

Leitfaden

AWS CodeCommit



API-Version 2015-04-13

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS CodeCommit: Leitfaden

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Die Marken und Handelsmarken von Amazon dürfen nicht in einer Weise in Verbindung mit nicht von Amazon stammenden Produkten oder Services verwendet werden, die geeignet ist, Kunden irrezuführen oder Amazon in irgendeiner Weise herabzusetzen oder zu diskreditieren. Alle anderen Handelsmarken, die nicht Eigentum von Amazon sind, gehören den jeweiligen Besitzern, die möglicherweise zu Amazon gehören oder nicht, mit Amazon verbunden sind oder von Amazon gesponsert werden.

Table of Contents

.....	xv
Was ist CodeCommit?	1
Wir stellen vor CodeCommit	2
CodeCommit, Git und die Wahl des richtigen AWS Dienstes für Ihre Bedürfnisse	3
Wie funktioniert das? CodeCommit	7
Wie CodeCommit unterscheidet sich die Dateiversionierung in Amazon S3?	9
Wie fange ich an mit CodeCommit?	9
Wo kann ich mehr über Git erfahren?	9
Einrichtung	11
Ihre Anmeldedaten einsehen und verwalten	11
Einrichtung mit Git-Anmeldeinformationen	12
Einrichtung mit anderen Methoden	12
Kompatibilität für CodeCommit Git und andere Komponenten	14
Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden	15
Schritt 1: Erstkonfiguration für CodeCommit	16
Schritt 2: Installieren von Git	17
Schritt 3: Erstellen Sie Git-Anmeldeinformationen für HTTPS-Verbindungen zu CodeCommit	18
Schritt 4: Connect zur CodeCommit Konsole her und klonen Sie das Repository	20
Nächste Schritte	21
Für HTTPS-Verbindungen mit git-remote-codecommit	22
Schritt 0: Installieren Sie die Voraussetzungen für git-remote-codecommit	23
Schritt 1: Erstkonfiguration für CodeCommit	24
Schritt 2: Installieren git-remote-codecommit	28
Schritt 3: Connect zur CodeCommit Konsole her und klonen Sie das Repository	29
Nächste Schritte	30
Für Verbindungen von Entwicklungstools	30
Integrieren AWS Cloud9 mit AWS CodeCommit	33
Integrieren Sie Visual Studio mit AWS CodeCommit	38
Für SSH-Benutzer, die das nicht verwenden AWS CLI	39
Schritt 1: Verknüpfen Sie Ihren öffentlichen Schlüssel mit Ihrem IAM-Benutzer	40
Schritt 2: Zu Ihrer CodeCommit SSH-Konfiguration hinzufügen	41
Nächste Schritte	42
Für SSH-Verbindungen unter Linux, macOS oder Unix	42

Schritt 1: Erstkonfiguration für CodeCommit	43
Schritt 2: Installieren von Git	44
Schritt 3: Anmeldeinformationen unter Linux, macOS oder Unix konfigurieren	45
Schritt 4: Connect zur CodeCommit Konsole her und klonen Sie das Repository	49
Nächste Schritte	51
Für SSH-Verbindungen unter Windows	51
Schritt 1: Erstkonfiguration für CodeCommit	51
Schritt 2: Installieren von Git	52
Schritt 3: Richten Sie die öffentlichen und privaten Schlüssel für Git ein und CodeCommit	53
Schritt 4: Connect zur CodeCommit Konsole her und klonen Sie das Repository	57
Nächste Schritte	59
Für HTTPS-Verbindungen unter Linux, macOS oder Unix mit dem AWS CLI Credential Helper	59
Schritt 1: Erstkonfiguration für CodeCommit	60
Schritt 2: Installieren von Git	64
Schritt 3: Richten Sie den Credential Helper ein	64
Schritt 4: Connect zur CodeCommit Konsole her und klonen Sie das Repository	67
Nächste Schritte	68
Für HTTPS-Verbindungen unter Windows mit dem Credential Helper AWS CLI	68
Schritt 1: Erstkonfiguration für CodeCommit	69
Schritt 2: Installieren von Git	73
Schritt 3: Richten Sie den Credential Helper ein	74
Schritt 4: Connect zur CodeCommit Konsole her und klonen Sie das Repository	76
Nächste Schritte	77
Erste Schritte	78
Erste Schritte mit CodeCommit	78
Voraussetzungen	80
Schritt 1: Erstellen eines CodeCommit-Repositorys	80
Schritt 2: Fügen Sie Dateien zu Ihrem Repository hinzu	83
Schritt 3: Durchsuchen Sie den Inhalt Ihres Repositorys	85
Schritt 4: Einen Pull-Request erstellen und gemeinsam daran arbeiten	90
Schritt 5: Bereinigen	97
Schritt 6: Nächste Schritte	98
Erste Schritte mit Git und CodeCommit	98
Schritt 1: Erstellen Sie ein CodeCommit Repository	100
Schritt 2: Erstellen Sie ein lokales Repo	101

Schritt 3: Erstelle deinen ersten Commit	104
Schritt 4: Push deinen ersten Commit	105
Schritt 5: Teile das CodeCommit Repository und führe einen weiteren Commit per Push und Pull durch	105
Schritt 6: Einen Branch erstellen und teilen	108
Schritt 7: Erstelle und teile ein Tag	110
Schritt 8: Zugriffsberechtigungen einrichten	111
Schritt 9: Bereinigen	115
Produkt- und Service-Integrationen	117
Integration mit anderen AWS Diensten	117
Integrationsbeispiele der Community	126
Blog-Posts	126
Codebeispiele	130
Arbeiten mit Repositorien	131
Erstellen eines -Repositorys	132
Erstellen Sie ein Repository (Konsole)	133
Erstellen Sie ein Repository ()AWS CLI	135
Herstellen einer Verbindung mit einem Repository	138
Voraussetzungen für die Verbindung mit einem CodeCommit Repository	138
Connect zum CodeCommit Repository her, indem Sie das Repository klonen	139
Ein lokales Repo mit dem CodeCommit Repository Connect	141
Teilen Sie ein Repository	142
Wählen Sie das Verbindungsprotokoll, das Sie mit Ihren Benutzern teilen möchten	144
Erstellen Sie IAM-Richtlinien für Ihr Repository	145
Erstellen Sie eine IAM-Gruppe für Repository-Benutzer	146
Teilen Sie die Verbindungsinformationen mit Ihren Benutzern	148
Benachrichtigungen für Repository-Ereignisse konfigurieren	149
Verwenden von Repository-Benachrichtigungsregeln	151
Erstellen einer Benachrichtigungsregel	152
Benachrichtigungen ändern oder deaktivieren	155
Benachrichtigungen löschen	156
Markieren eines Repositorys	158
Hinzufügen eines Tags zu einem Repository	159
Anzeigen von Tags für ein Repository	161
Bearbeiten von Tags für ein Repository	163
Entfernen Sie ein Tag aus einem Repository	165

Verwalten von Auslösern für ein Repository	166
Erstellen Sie die Ressource und fügen Sie Berechtigungen hinzu für CodeCommit	167
Einen Auslöser für ein Amazon SNS SNS-Thema erstellen	168
Erstellen Sie einen Trigger für eine Lambda-Funktion	176
Erstellen Sie einen Trigger für eine bestehende Lambda-Funktion	181
Trigger für ein Repository bearbeiten	190
Testen Sie Trigger für ein Repository	192
Trigger aus einem Repository löschen	194
Ein Repository mit Amazon CodeGuru Reviewer verknüpfen oder die Zuordnung aufheben	197
Ordnen Sie dem CodeGuru Rezensenten ein Repository zu	199
Trennen Sie die Zuordnung eines Repositorys zu Reviewer CodeGuru	200
Repository-Details anzeigen	200
Repository-Details anzeigen (Konsole)	201
CodeCommit Repository-Details anzeigen (Git)	201
CodeCommit Repository-Details anzeigen (AWS CLI)	203
Ändern von Repository-Einstellungen	207
Repository-Einstellungen ändern (Konsole)	207
AWS CodeCommit Repository-Einstellungen ändern (AWS CLI)	209
Synchronisieren Sie Änderungen zwischen Repositories	211
Push-Commits an zwei Repositories	213
Konfigurieren Sie den kontoübergreifenden Zugriff auf ein Repository mithilfe von Rollen	218
Kontoübergreifender Zugriff auf das Repository: Aktionen für den Administrator in AccountA	219
Kontoübergreifender Zugriff auf das Repository: Aktionen für den Administrator in AccountB	223
Kontoübergreifender Repository-Zugriff: Aktionen für den Repository-Benutzer in AccountB	225
Löscht ein Repository	231
Löschen Sie ein CodeCommit Repository (Konsole)	232
Löscht ein lokales Repo	233
Löscht ein CodeCommit Repository (AWS CLI)	233
Mit Dateien arbeiten	235
Dateien in einem Repository durchsuchen	236
Ein CodeCommit Repository durchsuchen	237
Eine Datei erstellen oder hinzufügen	238
Eine Datei erstellen oder hochladen (Konsole)	239

Füge eine Datei hinzu (AWS CLI)	241
Eine Datei hinzufügen (Git)	242
Bearbeiten Sie den Inhalt einer Datei	242
Bearbeiten Sie eine Datei (Konsole)	243
Bearbeiten oder löschen Sie eine Datei (AWS CLI)	244
Eine Datei bearbeiten (Git)	247
Verwenden von Pull-Anforderungen	248
Erstellen einer Pull-Anforderung	252
Erstellen Sie eine Pull-Anfrage (Konsole)	252
Erstellen Sie eine Pull-Anfrage ()AWS CLI	254
Erstellen einer Genehmigungsregel	256
Erstellen Sie eine Genehmigungsregel für eine Pull-Anfrage (Konsole)	257
Erstellen Sie eine Genehmigungsregel für eine Pull-Anfrage ()AWS CLI	260
Anzeigen von Pull-Anforderungen	262
Pull-Requests anzeigen (Konsole)	262
Pull-Requests anzeigen (AWS CLI)	263
Überprüfen einer Pull-Anforderung	268
Überprüfen Sie eine Pull-Anfrage (Konsole)	268
Überprüfen Sie Pull-Requests (AWS CLI)	274
Aktualisieren einer Pull-Anforderung	279
Aktualisiere eine Pull-Anfrage (Konsole)	280
Pull-Anfragen aktualisieren ()AWS CLI	280
Bearbeiten oder Löschen einer Genehmigungsregel	283
Bearbeiten oder löschen Sie eine Genehmigungsregel für eine Pull-Anfrage (Konsole)	284
Bearbeiten oder löschen Sie eine Genehmigungsregel für eine Pull-Anfrage (AWS CLI)	285
Genehmigungsregeln für eine Pull-Anfrage überschreiben	287
Genehmigungsregeln überschreiben (Konsole)	288
Genehmigungsregeln außer Kraft setzen ()AWS CLI	289
Zusammenführen einer Pull-Anforderung	290
Eine Pull-Anfrage zusammenführen (Konsole)	291
Eine Pull-Anfrage zusammenführen (AWS CLI)	294
Konflikte in einer Pull-Anfrage lösen	301
Konflikte in einer Pull-Anfrage lösen (Konsole)	302
Konflikte in einer Pull-Anfrage lösen (AWS CLI)	304
Schließen einer Pull-Anforderung	313
Schließt eine Pull-Anfrage (Konsole)	313

Schließt eine Pull-Anfrage (AWS CLI)	314
Arbeiten mit Genehmigungsregelvorlagen	316
Erstellen einer Genehmigungsregelvorlage	318
Erstellen Sie eine Vorlage für Genehmigungsregeln (Konsole)	318
Erstellen Sie eine Vorlage für Genehmigungsregeln (AWS CLI)	323
Ordnen Sie eine Vorlage für Genehmigungsregeln einem Repository zu	325
Ordnen Sie eine Vorlage für Genehmigungsregeln zu (Konsole)	325
Ordnen Sie eine Vorlage für Genehmigungsregeln zu (AWS CLI)	326
Verwalten von Genehmigungsregelvorlagen	327
Vorlagen für Genehmigungsregeln verwalten (Konsole)	327
Vorlagen für Genehmigungsregeln verwalten (AWS CLI)	328
Trennen Sie die Zuordnung einer Vorlage für eine Genehmigungsregel	333
Trennen Sie die Zuordnung einer Vorlage für Genehmigungsregeln (Konsole)	333
Trennen Sie die Zuordnung einer Vorlage für Genehmigungsregeln (AWS CLI)	334
Löschen Sie eine Vorlage für Genehmigungsregeln	335
Löschen Sie eine Vorlage für Genehmigungsregeln (Konsole)	335
Löschen Sie eine Vorlage für Genehmigungsregeln (AWS CLI)	336
Mit Commits arbeiten	337
Erstellen Sie einen Commit	338
Erstellen Sie den ersten Commit für ein Repository mit dem AWS CLI	339
Einen Commit mit einem Git-Client erstellen	340
Erstelle einen Commit AWS CLI	344
Anzeigen von Commit-Details	347
Durchsuche Commits in einem Repository	347
Commit-Details anzeigen (AWS CLI)	352
Commit-Details anzeigen (Git)	358
Vergleichen von Commits	361
Einen Commit mit seinem übergeordneten Element vergleichen	361
Vergleichen von zwei beliebigen Commit-Spezifizierern	364
Kommentar zu einem Commit	367
Kommentare zu einem Commit in einem Repository anzeigen	367
Kommentare zu einem Commit in einem Repository hinzufügen und darauf antworten	368
Kommentare anzeigen, hinzufügen, aktualisieren und beantworten (AWS CLI)	373
Ein Git-Tag erstellen	382
Verwende Git, um ein Tag zu erstellen	382
Tag-Details anzeigen	383

Tag-Details anzeigen (Konsole)	384
Git-Tag-Details anzeigen (Git)	385
Löschen eines Tags	387
Verwende Git, um ein Git-Tag zu löschen	387
Arbeiten mit Zweigen	389
Erstellen eines Zweigs	391
Erstelle einen Branch (Konsole)	391
Einen Zweig erstellen (Git)	392
Erstelle einen Zweig (AWS CLI)	394
Beschränken Sie Pushs und Merges auf Branches	395
Konfigurieren Sie eine IAM-Richtlinie, um Pushs und Merges auf einen Branch zu beschränken	396
Wenden Sie die IAM-Richtlinie auf eine IAM-Gruppe oder -Rolle an	398
Testen Sie die Richtlinie	399
Filialdetails anzeigen	399
Branch-Details anzeigen (Konsole)	400
Filialdetails anzeigen (Git)	401
Details zur Filiale anzeigen (AWS CLI)	402
Zweige vergleichen und zusammenführen	403
Vergleichen Sie einen Branch mit dem Standard-Branch	404
Vergleichen Sie zwei spezifische Branches	404
Zwei Zweige zusammenführen (AWS CLI)	405
Ändern Sie die Filialeinstellungen	408
Ändern Sie den Standardzweig (Konsole)	408
Ändern Sie den Standardzweig (AWS CLI)	409
Löschen Sie einen Zweig	410
Löscht einen Branch (Konsole)	411
Lösche einen Zweig (AWS CLI)	411
Einen Zweig löschen (Git)	412
Mit Benutzereinstellungen arbeiten	414
Migrieren Sie zu CodeCommit	415
Migrieren Sie ein Git-Repository zu AWS CodeCommit	415
Schritt 0: Für den Zugriff auf ist eine Einrichtung erforderlich CodeCommit	416
Schritt 1: Erstellen Sie ein Repository CodeCommit	422
Schritt 2: Das Repository klonen und zum CodeCommit Repository pushen	425
Schritt 3: Dateien anzeigen in CodeCommit	426

Schritt 4: Teilen Sie das Repository CodeCommit	427
Migrieren Sie Inhalte zu CodeCommit	430
Schritt 0: Für den Zugriff auf ist eine Einrichtung erforderlich CodeCommit	431
Schritt 1: Erstellen Sie ein Repository CodeCommit	437
Schritt 2: Migrieren Sie lokale Inhalte in das CodeCommit Repository	438
Schritt 3: Dateien anzeigen in CodeCommit	440
Schritt 4: Teilen Sie das CodeCommit Repository	440
Migrieren Sie ein Repository schrittweise	443
Schritt 0: Stellen Sie fest, ob eine inkrementelle Migration erfolgen soll	444
Schritt 1: Installieren Sie die erforderlichen Komponenten und fügen Sie das CodeCommit Repository als Remote-Repository hinzu	445
Schritt 2: Erstellen Sie das Skript, das für die inkrementelle Migration verwendet werden soll	447
Schritt 3: Führen Sie das Skript aus und migrieren Sie schrittweise zu CodeCommit	447
Anhang: Beispielskript <code>incremental-repo-migration.py</code>	449
Sicherheit	457
Datenschutz	457
AWS KMS und Verschlüsselung	458
Verwendung rotierender Anmeldeinformationen	461
Identitäts- und Zugriffsverwaltung	466
Zielgruppe	467
Authentifizierung mit Identitäten	468
Verwalten des Zugriffs mit Richtlinien	471
Authentifizierung und Zugriffskontrolle	474
Wie AWS CodeCommit funktioniert mit IAM	548
Ressourcenbasierte CodeCommit-Richtlinien	549
Autorisierung auf der Grundlage von Tags CodeCommit	549
CodeCommit IAM-Rollen	553
Beispiele für identitätsbasierte Richtlinien	554
Fehlerbehebung	557
Ausfallsicherheit	560
Sicherheit der Infrastruktur	561
Überwachung CodeCommit	562
Überwachung von CodeCommit Ereignissen	563
referenceCreated-Ereignis	564
referenceUpdated-Ereignis	565

referenceDeleted-Ereignis	565
unreferencedMergeCommitEreignis wurde erstellt	566
commentOnCommitEreignis wurde erstellt	567
commentOnCommitEreignis wurde aktualisiert	568
commentOnPullRequestCreated Ereignis	569
commentOnPullRequestUpdated Ereignis	570
pullRequestCreated Ereignis	571
pullRequestSourceBranchUpdated Ereignis	572
pullRequestStatusGeändertes Ereignis	573
pullRequestMergeStatusUpdated Ereignis	574
approvalRuleTemplateEreignis wurde erstellt	575
approvalRuleTemplateDas Ereignis wurde aktualisiert	575
approvalRuleTemplateGelöschtes Ereignis	576
approvalRuleTemplateAssociatedWithRepository Ereignis	577
approvalRuleTemplateDisassociatedWithRepository Ereignis	578
approvalRuleTemplateBatchAssociatedWithRepositories Ereignis	579
approvalRuleTemplateBatchDisassociatedFromRepositories Ereignis	580
pullRequestApprovalRuleCreated Ereignis	580
pullRequestApprovalRuleDeleted Ereignis	582
pullRequestApprovalRuleOverridden Ereignis	583
pullRequestApprovalStateChanged Ereignis	585
pullRequestApprovalRuleUpdated Ereignis	587
ReactionCreated-Ereignis	588
ReactionUpdated-Ereignis	589
AWS CodeCommit API-Aufrufe protokollieren mit AWS CloudTrail	590
CodeCommit Informationen in CloudTrail	590
Grundlegendes zu CodeCommit Einträgen in Protokolldateien	592
AWS CloudFormation Ressourcen	600
CodeCommit und AWS CloudFormation Vorlagen	600
Beispiele für Vorlagen	601
AWS CloudFormation, CodeCommit, und die AWS Cloud Development Kit (AWS CDK)	602
Erfahren Sie mehr über AWS CloudFormation	603
Fehlerbehebung	604
Fehlerbehebung bei Git-Anmeldeinformationen (HTTPS)	604

Git-Anmeldeinformationen für AWS CodeCommit: Ich werde immer wieder zur Eingabe von Anmeldeinformationen aufgefordert, wenn ich über das Terminal oder die Befehlszeile eine Verbindung zu meinem CodeCommit Repository herstelle	605
Git-Anmeldeinformationen für AWS CodeCommit: Ich habe Git-Anmeldeinformationen eingerichtet, aber mein System verwendet sie nicht	605
Fehlerbehebung für git-remote-codecommit	606
Ich sehe einen Fehler: git: 'remote-codecommit' ist kein Git-Befehl	606
Mir wird ein Fehler angezeigt: Schwerwiegender Fehler: Der Remote-Helper für 'codecommit' konnte nicht gefunden werden	607
Fehler beim Klonen: Ich kann ein CodeCommit Repository nicht von einer IDE klonen	607
Push- oder Pull-Fehler: Ich kann keine Commits per Push oder Pull von einer IDE in ein CodeCommit-Repository verschieben	608
Installationsfehler: Beim Versuch, die Installation durchzuführen, wird ein externally-managed-environment Fehler angezeigt git-remote-codecommit	608
Problembehandlung bei SSH-Verbindungen	608
Zugriffsfehler: Der öffentliche Schlüssel wurde erfolgreich in IAM hochgeladen, aber die Verbindung schlägt auf Linux-, macOS- oder Unix-Systemen fehl	609
Zugriffsfehler: Der öffentliche Schlüssel wurde erfolgreich in IAM hochgeladen und SSH wurde erfolgreich getestet, aber die Verbindung schlägt auf Windows-Systemen fehl	610
Authentifizierungsherausforderung: Die Authentizität des Hosts kann nicht festgestellt werden, wenn eine Verbindung zu einem Repository hergestellt wird CodeCommit	611
IAM-Fehler: „Ungültiges Format“ beim Versuch, IAM einen öffentlichen Schlüssel hinzuzufügen	618
Ich muss mit SSH-Anmeldeinformationen auf CodeCommit Repositories in mehreren Amazon Web Services Services-Konten zugreifen	619
Git unter Windows: Bash-Emulator oder Befehlszeilenfenster stürzt bei Verbindungsversuchen mit SSH ab	620
Das Public-Key-Format erfordert in einigen Linux-Distributionen eine Spezifikation	621
Zugriffsfehler: Der öffentliche SSH-Schlüssel wurde beim Herstellen einer Verbindung zu einem CodeCommit Repository verweigert	621
Problembehandlung beim Credential Helper (HTTPS)	621
Ich erhalte eine Fehlermeldung, wenn ich den git config Befehl zur Konfiguration des Credential Helper ausführe	622
Beim Versuch, ein Repository zu klonen, wird beim Lesen des Benutzernamens ein Fehler angezeigt	623

Zurückgegebene Fehlermeldung "Befehl nicht gefunden" unter Windows bei Verwendung des Hilfsprogramms für Anmeldeinformationen	623
Ich werde zur Eingabe eines Benutzernamens aufgefordert, wenn ich eine Verbindung zu einem Repository herstelle CodeCommit	624
Git für macOS: Ich habe den Credential Helper erfolgreich konfiguriert, aber jetzt wird mir der Zugriff auf mein Repository verweigert (403)	624
Git für Windows: Ich habe Git für Windows installiert, aber mir wird der Zugriff auf mein Repository verweigert (403)	628
Problembehandlung bei Git-Clients	631
Git-Fehler: Fehler: RPC fehlgeschlagen; Ergebnis=56, HTTP-Code = 200 fatal: Das Remote-Ende ist unerwartet hängen geblieben	631
Git-Fehler: Zu viele Referenz-Update-Befehle	632
Git-Fehler: Push-Übertragung über HTTPS funktioniert in einigen Git-Versionen nicht	632
Git-Fehler: "gnutls_handshake() failed"	632
Git-Fehler: Git kann das CodeCommit Repository nicht finden oder hat keine Berechtigung, auf das Repository zuzugreifen	633
Git unter Windows: Keine unterstützten Authentifizierungsmethoden verfügbar (publickey) ..	633
Behebung von Zugriffsfehlern	634
Zugriffsfehler: Ich werde zur Eingabe eines Benutzernamens und eines Kennworts aufgefordert, wenn ich von Windows aus eine Verbindung zu einem CodeCommit Repository herstelle	634
Zugriffsfehler: Verweigerung des öffentlichen Schlüssels bei der Verbindung mit einem CodeCommit-Repository	635
Zugriffsfehler: Meldung „Rate Exceeded“ oder „429“ beim Herstellen einer Verbindung zu einem Repository CodeCommit	635
Behebung von Konfigurationsfehlern	636
Konfigurationsfehler: AWS CLI Anmeldeinformationen können auf macOS nicht konfiguriert werden	637
Behebung von Konsolenfehlern	637
Zugriffsfehler: Zugriff auf den Verschlüsselungsschlüssel für ein CodeCommit Repository über die Konsole verweigert oder AWS CLI	635
Verschlüsselungsfehler: Das Repository kann nicht entschlüsselt werden	638
Konsolenfehler: Der Code in einem CodeCommit Repository kann nicht von der Konsole aus durchsucht werden	638
Anzeigefehler: Eine Datei oder ein Vergleich zwischen Dateien kann nicht angezeigt werden	639

Behebung von Auslösern	639
Auslöserfehler: Ein Repository-Auslöser wird nicht zum erwarteten Zeitpunkt ausgeführt	639
Schalten Sie das Debuggen ein	640
CodeCommit Referenz	642
Regionen und Git-Verbindungsendpunkte	642
Unterstützt AWS-Regionen für CodeCommit	643
Endpunkte für Git-Verbindungen	644
Server-Fingerabdrücke für CodeCommit	651
Verwendung AWS CodeCommit mit VPC-Endpunkten mit Schnittstelle	658
Verfügbarkeit	659
VPC-Endpoints erstellen für CodeCommit	660
Erstellen einer VPC-Endpunktrichtlinie für CodeCommit	661
Kontingente	662
Befehlszeilenreferenz	670
Grundlegende Git-Befehle	676
Konfigurationsvariablen	677
Remote-Repositorys	677
Commits	679
Branches	681
Tags	682
Dokumentverlauf	684
Frühere Aktualisierungen	696
AWS Glossar	704

AWS CodeCommit ist für Neukunden nicht mehr verfügbar. Bestandskunden von AWS CodeCommit können den Service weiterhin wie gewohnt nutzen. [Erfahren Sie mehr](#)

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.

Was ist AWS CodeCommit?

AWS CodeCommit ist ein von Amazon Web Services gehosteter Service zur Versionskontrolle, mit dem Sie Assets (wie Dokumente, Quellcode und Binärdateien) privat in der Cloud speichern und verwalten können. Informationen zur Preisgestaltung für finden Sie CodeCommit unter [Preisgestaltung](#).

Note

CodeCommit fällt in den Geltungsbereich vieler Compliance-Programme. Einzelheiten zu den Compliance-Maßnahmen AWS und den Bemühungen zur Einhaltung der Vorschriften finden Sie unter [AWS Services im Umfang der einzelnen Compliance-Programme](#).

Dies ist ein HIPAA-berechtigter Service. [Weitere Informationen AWS zum US-amerikanischen Health Insurance Portability and Accountability Act von 1996 \(HIPAA\) und zur Nutzung von AWS Diensten zur Verarbeitung, Speicherung und Übertragung geschützter Gesundheitsinformationen \(PHI\) finden Sie unter HIPAA Overview](#).

Informationen zu diesem Service und ISO 27001, einer Norm für das Sicherheitsmanagement, die bewährte Methoden für das Sicherheitsmanagement festlegt, finden Sie unter [ISO 27001-Übersicht](#).

Weitere Informationen zu diesem Service und dem Payment Card Industry Data Security Standard (PCI DSS) finden Sie unter [Übersicht über PCI DSS](#).

Informationen zu diesem Service und zum US-Regierungsstandard Federal Information Processing Standard (FIPS) Publication 140-2, der die Sicherheitsanforderungen für kryptografische Module zum Schutz vertraulicher Informationen angibt, finden Sie unter [Federal Information Processing Standard \(FIPS\) 140-2 – Übersicht](#) und [Endpunkte für Git-Verbindungen](#).

Themen

- [Wir stellen vor CodeCommit](#)
- [CodeCommit, Git und die Wahl des richtigen AWS Dienstes für Ihre Bedürfnisse](#)
- [Wie funktioniert das? CodeCommit](#)
- [Wie CodeCommit unterscheidet sich die Dateiversionierung in Amazon S3?](#)
- [Wie fange ich an mit CodeCommit?](#)
- [Wo kann ich mehr über Git erfahren?](#)

Wir stellen vor CodeCommit

CodeCommit ist ein sicherer, hoch skalierbarer, verwalteter Quellcodeverwaltungsdienst, der private Git-Repositorys hostet. CodeCommit macht es überflüssig, dass Sie Ihr eigenes Quellcodeverwaltungssystem verwalten oder sich Gedanken über die Skalierung der Infrastruktur machen müssen. Sie können CodeCommit damit alles speichern, von Code bis hin zu Binärdateien. Es unterstützt die Standardfunktionen von Git, sodass Sie Ihre vorhandenen Git-basierten Tools weiterhin problemlos nutzen können.

Mit CodeCommit können Sie:

- Profitieren Sie von einem vollständig verwalteten Service, der von bereitgestellt wird AWS. CodeCommit bietet eine hohe Serviceverfügbarkeit und Beständigkeit und macht den administrativen Aufwand für die Verwaltung Ihrer eigenen Hardware und Software überflüssig. Es ist nicht mehr nötig, Hardware bereitzustellen und zu skalieren oder Serversoftware zu installieren, zu konfigurieren und zu aktualisieren.
- Bewahren Sie Ihren Code sicher auf. CodeCommit Repositorys werden sowohl im Ruhezustand als auch bei der Übertragung verschlüsselt.
- Arbeiten Sie gemeinsam an Code. CodeCommit Repositorys unterstützen Pull-Requests, bei denen Benutzer die Codeänderungen anderer Benutzer überprüfen und kommentieren können, bevor sie sie zu Branches zusammenführen, Benachrichtigungen, mit denen Benutzer automatisch E-Mails über Pull-Requests und Kommentare erhalten, und vieles mehr.
- Skalieren Sie auf einfache Weise Ihre Versionskontrollprojekte. CodeCommit-Repositorys sind skalierbar und können an Ihre Entwicklungsanforderungen angepasst werden. Der Service kann Repositorys mit einer großen Anzahl an Dateien oder Branches, mit großen Dateien und mit umfangreichen Revisionsverläufen verarbeiten.
- Speichere alles, jederzeit. CodeCommit hat keine Begrenzung der Größe Ihrer Repositorys oder der Dateitypen, die Sie speichern können.
- Integrieren Sie in andere Dienste AWS und Dienste von Drittanbietern. CodeCommit hält Ihre Repositorys in der Nähe Ihrer anderen Produktionsressourcen in der AWS Cloud, was dazu beiträgt, die Geschwindigkeit und Häufigkeit Ihres Entwicklungszyklus zu erhöhen. Es ist in IAM integriert und kann mit anderen AWS Diensten und parallel zu anderen Repositorys verwendet werden. Weitere Informationen finden Sie unter [Produkt- und Serviceintegrationen mit AWS CodeCommit](#).
- Migrieren Sie problemlos Dateien aus anderen Remote-Repositorys. Sie können CodeCommit von jedem Git-basierten Repository zu migrieren.

- Verwende die Git-Tools, die du bereits kennst. CodeCommit unterstützt Git-Befehle sowie eigene AWS CLI Befehle und APIs.

CodeCommit, Git und die Wahl des richtigen AWS Dienstes für Ihre Bedürfnisse

Als Git-basierter Dienst CodeCommit ist er für die meisten Versionskontrollanforderungen gut geeignet. Es gibt keine willkürlichen Einschränkungen hinsichtlich Dateigröße, Dateityp und Repository-Größe. Es gibt jedoch inhärente Einschränkungen für Git, die insbesondere im Laufe der Zeit negative Auswirkungen auf die Leistung bestimmter Vorgänge haben können. Sie können eine potenzielle Verschlechterung der CodeCommit Repository-Leistung vermeiden, indem Sie es vermeiden, ihn für Anwendungsfälle zu verwenden, in denen andere AWS Dienste für die Aufgabe besser geeignet sind. Sie können die Git-Leistung auch für komplexe Repositories optimieren. Hier sind einige Anwendungsfälle, in denen Git und daher CodeCommit möglicherweise nicht die beste Lösung für Sie sind oder in denen Sie möglicherweise zusätzliche Schritte unternehmen müssen, um für Git zu optimieren.

Anwendungsfall	Beschreibung	Weitere zu berücksichtigende Services
Große Dateien mit häufigen Änderungen	Git verwendet zum Speichern von Unterschieden zwischen Dateiversionen Delta-Kodierung. Beispiel: Wenn Sie ein paar Wörter in einem Dokument ändern, speichert Git nur diese geänderten Wörter. Bei Dateien oder Objekten über 5 MB mit vielen Änderungen muss Git möglicherweise eine große Kette von Delta-Unterschiede rekonstruieren. Dies kann eine zunehmende Menge an Rechenressourcen sowohl auf Ihrem lokalen Computer	Wenn Sie große Dateien versionieren möchten, sollten Sie Amazon Simple Storage Service (Amazon S3) in Betracht ziehen. Weitere Informationen finden Sie unter Using Versioning im Amazon Simple Storage Service-Benutzerhandbuch.

Anwendungsfall	Beschreibung	Weitere zu berücksichtigende Services
	CodeCommit als auch in dem Maße verbrauchen, in dem diese Dateien mit der Zeit wachsen.	
Datenbank	Git-Repositorys werden im Laufe immer größer. Da beim Versioning alle Änderungen verfolgt werden, nimmt die Größe des Repositorys mit jeder Änderung zu. Bei der Übermittlung von Daten mit Commit werden also selbst dann, wenn während der Übermittlung Daten gelöscht werden, Daten zu einem Repository hinzugefügt. Da im Laufe der Zeit mehr Daten verarbeitet und übertragen werden, wird Git immer langsamer. Dies wirkt sich insbesondere bei einem Datenbank-Anwendungsfall nachteilig aus. Git wurde nicht als Datenbank konzipiert.	Wenn Sie unabhängig von ihrer Größe eine Datenbank mit gleichbleibender Leistung erstellen und verwenden möchten, sollten Sie Amazon DynamoDB in Betracht ziehen. Weitere Informationen finden Sie im Amazon DynamoDB Getting Started Guide .

Anwendungsfall	Beschreibung	Weitere zu berücksichtigende Services
Audit-Trails	In der Regel werden Audit-Trails über einen langen Zeitraum aufbewahrt und kontinuierlich von Systemprozessen in sehr häufigen Intervallen erstellt. Git ist auf das sichere Speichern von Quellcode ausgelegt, der von Gruppen von Entwicklern in einem Entwicklungszyklus erstellt wird. Bei sich schnell ändernden Repositorys, in denen programmgesteuerte Systemänderungen kontinuierlich gespeichert werden, vermindert sich die Leistung im Laufe der Zeit.	Um Prüfprotokolle zu speichern, sollten Sie Amazon Simple Storage Service (Amazon S3) in Betracht ziehen. Um AWS Aktivitäten zu überprüfen, sollten Sie je nach Anwendungsfall die Verwendung von AWS CloudTrail, AWS Config, oder Amazon in Betracht ziehen CloudWatch.

Anwendungsfall	Beschreibung	Weitere zu berücksichtigende Services
Sicherungen	Git wurde zum Versioning des von Entwicklern geschriebenen Quellcodes konzipiert. Als Backup-Strategie können Sie Commits per Push auf zwei Remote-Repositorys übertragen, darunter ein CodeCommit Repository. Git ist jedoch nicht auf die Handhabung von Sicherungen Ihres Computer-Dateisystems, Datenbank-Dumps oder ähnlicher Sicherungsinhalte ausgelegt. Dies könnte Ihr System verlangsamen und den Zeitaufwand erhöhen, der zum Klonen und für die Push-Übertragung eines Repositorys erforderlich ist.	Informationen zur Sicherung in der AWS Cloud finden Sie unter Backup und Wiederherstellen.

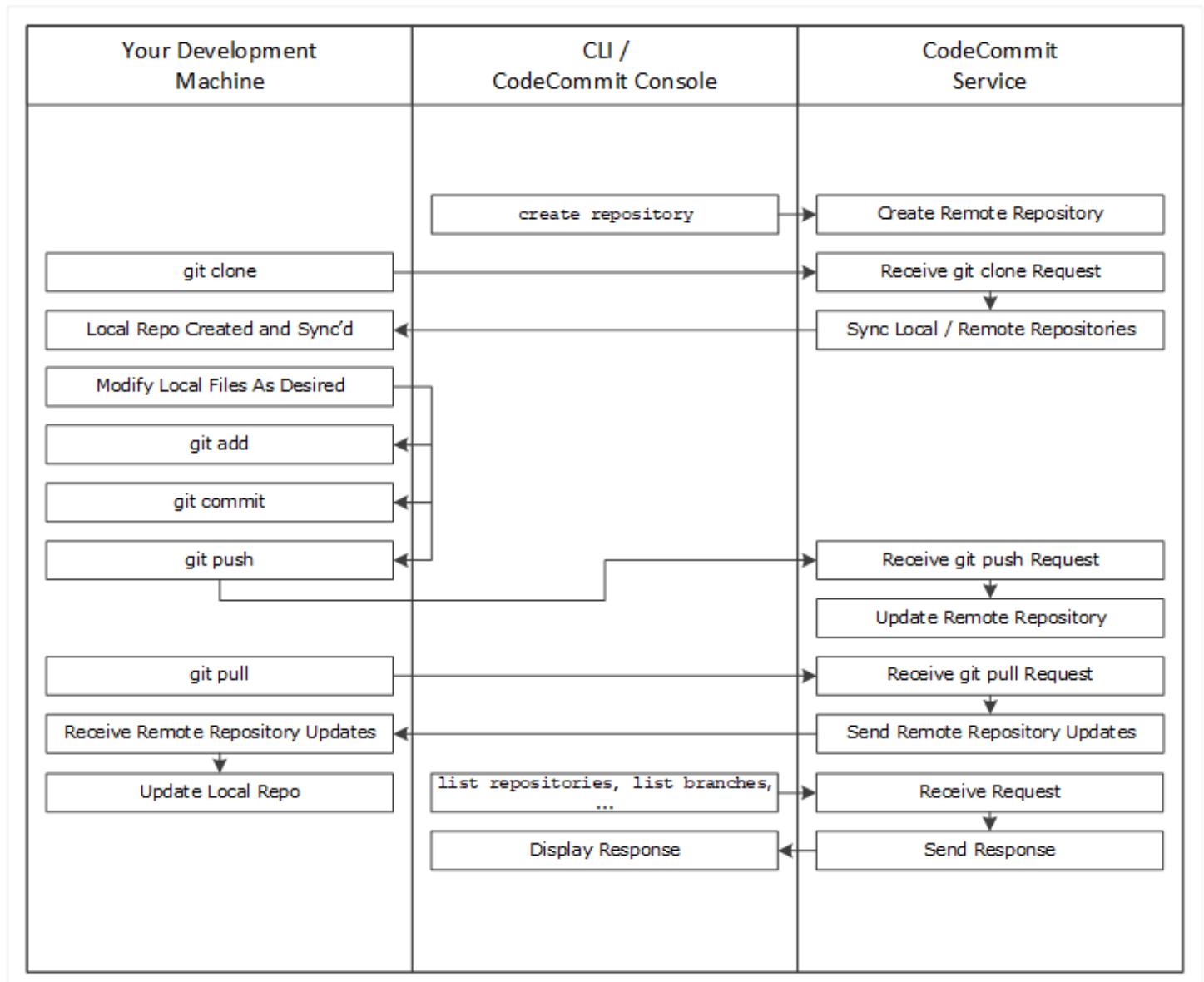
Anwendungsfall	Beschreibung	Weitere zu berücksichtigende Services
Große Anzahl von Verzweigungen oder Verweisen	Wenn ein Git-Client Repository-Daten per Push and Pull hochlädt oder abrufen muss, muss der Remote-Server alle Verzweigungen und Verweise wie z. B. Tags senden, auch wenn Sie nur an einer Verzweigung interessiert sind. Wenn Tausende von Verzweigungen und Verweise vorhanden sind, kann die Verarbeitung und das Senden (Paketaushandlung) einige Zeit in Anspruch nehmen und eine scheinbar langsame Repository-Reaktion zur Folge haben. Je mehr Verzweigungen und Tags vorhanden sind, desto länger kann dieser Vorgang dauern. Wir empfehlen CodeCommit, Branches und Tags, die nicht mehr benötigt werden, zu verwenden, aber zu löschen.	Um die Anzahl der Referenzen in einem CodeCommit Repository zu analysieren und festzustellen, welche möglicherweise nicht benötigt werden, können Sie einen der folgenden Befehle verwenden: - Linux-, macOS- oder Unix- oder Bash-Emulator unter Windows:<pre>git ls-remote wc -l</pre> - Powershell:<pre>git ls-remote Measure-Object -line</pre>

Wie funktioniert das? CodeCommit

CodeCommit ist Benutzern von Git-basierten Repositories vertraut, aber selbst Unbekannte sollten den Übergang zu CodeCommit Git-basierten Repositories als relativ einfach empfinden. CodeCommit bietet eine Konsole für die einfache Erstellung von Repositories und die Auflistung vorhandener Repositories und Branches. In wenigen einfachen Schritten können Benutzer Informationen über ein Repository finden und sie auf ihren Computer klonen. Dabei können sie ein lokales Repository erstellen, in dem sie Änderungen vornehmen und diese dann per Push in das Repository übertragen

können. CodeCommit Benutzer können die Befehlszeile des lokalen Computers oder einen GUI-basierten Editor nutzen.

Die folgende Abbildung zeigt, wie Sie Ihren Entwicklungscomputer, die AWS CLI CodeCommit OR-Konsole und den CodeCommit Dienst verwenden, um Repositories zu erstellen und zu verwalten:



1. Verwenden Sie die AWS CLI oder die CodeCommit Konsole, um ein CodeCommit Repository zu erstellen.
2. Verwenden Sie Git zum Ausführen `git clone` von Ihrem Entwicklungscomputer aus und geben Sie dabei den Namen des CodeCommit Repositories an. Dadurch wird ein lokales Repo erstellt, das eine Verbindung zum CodeCommit Repository herstellt.

3. Verwenden Sie das lokale Repository auf Ihrem Entwicklungscomputer, um Dateien zu ändern (hinzufügen, zu bearbeiten und zu löschen), und führen Sie es dann aus, `git add` um die geänderten Dateien lokal bereitzustellen. Führen Sie aus, `git commit` um die Dateien lokal zu übertragen, und starten Sie dann, `git push` um die Dateien an das CodeCommit Repository zu senden.
4. Laden Sie Änderungen von anderen Benutzern herunter. Führen Sie `git pull` aus, um die Dateien im CodeCommit Repository mit Ihrem lokalen Repository zu synchronisieren. So wird sichergestellt, dass Sie mit den neuesten Versionen dieser Dateien arbeiten.

Sie können die Konsole AWS CLI oder die CodeCommit Konsole verwenden, um Ihre Repositories zu verfolgen und zu verwalten.

Wie CodeCommit unterscheidet sich die Dateiversionierung in Amazon S3?

CodeCommit ist für die Softwareentwicklung im Team optimiert. Es verwaltet Stapel von Änderungen in mehreren Dateien, die parallel zu Änderungen von anderen Entwicklern auftreten können. Die Amazon S3 S3-Versionierung unterstützt die Wiederherstellung früherer Versionen von Dateien, konzentriert sich jedoch nicht auf Funktionen zur gemeinsamen Dateiverfolgung, die Softwareentwicklungsteams benötigen.

Wie fange ich an mit CodeCommit?

Um loszulegen mit CodeCommit:

1. Führen Sie die Schritte unter [Einrichtung](#) aus, um die Entwicklungscomputer vorzubereiten.
2. Führen Sie die Schritte in einem oder mehreren Tutorials unter [Erste Schritte](#) aus.
3. [Erstellen Sie](#) Versionskontrollprojekte in CodeCommit oder [migrieren Sie](#) Versionskontrollprojekte zu CodeCommit.

Wo kann ich mehr über Git erfahren?

Sie sollten sich mit der [Verwendung von Git vertraut machen](#), sofern noch nicht geschehen. Nachfolgend finden Sie einige hilfreiche Ressourcen:

- [Pro Git](#) ist eine Online-Version des Buchs Pro Git. Verfasst von Scott Chacon. Veröffentlicht von Apress.
- [Git Immersion](#), eine try-it-yourself geführte Tour, die Sie durch die Grundlagen der Verwendung von Git führt. Veröffentlicht von Neo Innovation, Inc.
- [Git Reference](#) ist ein Online-Schnelleinstieg, der auch als ausführlicheres Git-Tutorial genutzt werden kann. Veröffentlicht vom Team. GitHub
- [Git Cheat Sheet](#) bietet die grundlegende Git-Befehlssyntax. Vom GitHub Team veröffentlicht.
- [Git Pocket Guide](#). Verfasst von Richard E. Silverman. Veröffentlicht von O'Reilly Media, Inc.

Einrichtung für AWS CodeCommit

Sie können sich direkt von der AWS CodeCommit Konsole aus bei der anmelden AWS Management Console und [eine Datei hochladen, hinzufügen oder in ein Repository bearbeiten](#). Dies ist eine schnelle Möglichkeit, um eine Änderung vorzunehmen. Wenn Sie jedoch mit mehreren Dateien, Branch-übergreifenden Dateien usw. arbeiten möchten, sollten Sie Ihren lokalen Computer so einrichten, dass er Repositories unterstützt. Die einfachste Methode zur Einrichtung CodeCommit besteht darin, HTTPS-Git-Anmeldeinformationen für zu konfigurieren AWS CodeCommit. Diese HTTPS-Authentifizierungsmethode:

- Nutzt einen statischen Benutzernamen und ein statisches Passwort.
- Funktioniert mit allen Betriebssystemen, die von unterstützt werden CodeCommit.
- Ist auch kompatibel mit integrierten Entwicklungsumgebungen (IDEs) und anderen Entwicklungstools, die Git-Anmeldeinformationen unterstützen.

Sie können andere Methoden verwenden, wenn Sie dies wünschen, oder aus betrieblichen Gründen keine Git-Anmeldeinformationen verwenden können. Wenn Sie beispielsweise über Verbundzugriff, temporäre Anmeldeinformationen oder einen Web-Identitätsanbieter auf CodeCommit Repositories zugreifen, können Sie keine Git-Anmeldeinformationen verwenden. Es wird empfohlen, den lokalen Computer mit dem `git-remote-codecommit`-Befehl einzurichten. Lesen Sie diese anderen Optionen aufmerksam durch, um zu entscheiden, welche alternative Methode am besten für Sie geeignet ist.

- [Einrichtung mit Git-Anmeldeinformationen](#)
- [Einrichtung mit anderen Methoden](#)
- [Kompatibilität für CodeCommit Git und andere Komponenten](#)

Informationen zur Verwendung von CodeCommit Amazon Virtual Private Cloud finden Sie unter [Verwendung AWS CodeCommit mit VPC-Endpunkten mit Schnittstelle](#).

Ihre Anmeldedaten einsehen und verwalten

Sie können Ihre CodeCommit Anmeldeinformationen von der AWS Konsole aus über Meine Sicherheitsanmeldedaten einsehen und verwalten.

 Note

Diese Option ist nicht für Benutzer verfügbar, die Verbundzugriff, temporäre Anmeldeinformationen oder einen Web-Identitätsanbieter verwenden.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter <https://console.aws.amazon.com/iam/>
2. Wählen Sie auf der Navigationsleiste rechts oben Ihren Benutzernamen und dann Security Credentials (Sicherheits-Anmeldeinformationen) aus.
3. Wählen Sie den Tab „AWS CodeCommit Anmeldeinformationen“.

Einrichtung mit Git-Anmeldeinformationen

Mit HTTPS-Verbindungen und Git-Anmeldeinformationen generieren Sie einen statischen Benutzernamen und ein Kennwort in IAM. Sie können dieselben Anmeldeinformationen dann auch mit Git und jedem Drittanbieter-Tool verwenden, das die Authentifizierung mit Git-Benutzername und -Passwort unterstützt. Diese Methode wird von den meisten IDEs Entwicklungstools unterstützt. Es ist die einfachste und am leichtesten zu verwendende Verbindungsmethode CodeCommit.

- [Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden](#): Folgen Sie diesen Anweisungen, um mithilfe von Git-Anmeldeinformationen Verbindungen zwischen Ihrem lokalen Computer und CodeCommit Repositorys einzurichten.
- [Für Verbindungen von Entwicklungstools](#): Folgen Sie diesen Richtlinien, um mithilfe von Git-Anmeldeinformationen Verbindungen zwischen Ihrer IDE oder anderen Entwicklungstools und CodeCommit Repositorys einzurichten. IDEs zu den Git-Anmeldeinformationen gehören (sind aber nicht darauf beschränkt) Visual Studio, Xcode und IntelliJ.

Einrichtung mit anderen Methoden

Sie können das SSH-Protokoll anstelle von HTTPS verwenden, um eine Verbindung zu Ihrem CodeCommit Repository herzustellen. Mit SSH-Verbindungen erstellen Sie öffentliche und private Schlüsseldateien auf Ihrem lokalen Computer, die Git CodeCommit verwenden und für die SSH-Authentifizierung verwenden. Sie verknüpfen den öffentlichen Schlüssel mit Ihrem IAM-Benutzer. Den privaten Schlüssel speichern Sie auf dem lokalen Computer. Da SSH die manuelle Erstellung

und Verwaltung von öffentlichen und privaten Schlüsseldateien erfordert, kann es einfacher sein, Git-Anmeldeinformationen mit CodeCommit zu verwenden.

Anders als bei Git-Anmeldeinformationen ist die Einrichtung von SSH-Verbindungen je nach Betriebssystem auf dem lokalen Computer unterschiedlich.

- [Für SSH-Benutzer, die das nicht verwenden AWS CLI](#): Befolgen Sie diese verkürzten Anweisungen, wenn Sie bereits über ein Schlüsselpaar aus öffentlichem und privatem Schlüssel verfügen und mit SSH-Verbindungen auf dem lokalen Computer vertraut sind.
- [Für SSH-Verbindungen unter Linux, macOS oder Unix](#): Folgen Sie diesen Anweisungen, um eine step-by-step exemplarische Vorgehensweise zum Erstellen eines öffentlich-privaten key pair und zum Einrichten von Verbindungen auf Linux-, macOS- oder Unix-Betriebssystemen zu erhalten.
- [Für SSH-Verbindungen unter Windows](#): Folgen Sie diesen Anweisungen, um eine step-by-step exemplarische Vorgehensweise zum Erstellen eines öffentlich-privaten key pair und zum Einrichten von Verbindungen unter Windows-Betriebssystemen zu erhalten.

Wenn Sie eine Verbindung AWS mit Verbundzugriff, einem Identitätsanbieter oder temporären Anmeldeinformationen herstellen CodeCommit und diese verwenden oder wenn Sie keine IAM-Benutzer oder Git-Anmeldeinformationen für IAM-Benutzer konfigurieren möchten, können Sie Verbindungen zu CodeCommit Repositorys auf zwei Arten einrichten:

- Installieren und verwenden Sie `git-remote-codecommit` (empfohlen).
- Installieren und verwenden Sie das Hilfsprogramm für Anmeldeinformationen, der in der AWS CLI enthalten ist.

Beide Methoden unterstützen den Zugriff auf CodeCommit Repositorys, ohne dass ein IAM-Benutzer erforderlich ist. Das bedeutet, dass Sie mithilfe von Verbundzugriff und temporären Anmeldeinformationen eine Verbindung zu Repositorys herstellen können. Das `git-remote-codecommit` Hilfsprogramm ist der empfohlene Ansatz. Es erweitert Git und ist mit einer Vielzahl von Git-Versionen und Hilfsprogrammen für Anmeldeinformationen kompatibel. Allerdings IDEs unterstützen nicht alle das von verwendete Klon-URL-Format `git-remote-codecommit`. Möglicherweise müssen Sie Repositorys manuell auf Ihren lokalen Computer klonen, bevor Sie mit ihnen in der IDE arbeiten können.

- Folgen Sie den Anweisungen [unter Einrichtungsschritte für HTTPS-Verbindungen zu AWS CodeCommit Repositorys mit git-remote-codecommit](#), um die Installation und Einrichtung `git-remote-codecommit` unter Windows, Linux, macOS oder Unix durchzuführen.

Der in der enthaltene Credential Helper AWS CLI ermöglicht es Git, HTTPS und eine kryptografisch signierte Version Ihrer IAM-Benutzeranmeldedaten oder EC2 Amazon-Instance-Rolle zu verwenden, wann immer Git sich authentifizieren muss, um mit AWS Repositorys zu interagieren. CodeCommit Einige Betriebssysteme und Git-Versionen verfügen über eigene Hilfsprogramme für Anmeldeinformationen, die mit dem in der AWS CLI enthaltenen Hilfsprogramm für Anmeldeinformationen in Konflikt stehen. Sie können Verbindungsprobleme für verursachen. CodeCommit

- [Für HTTPS-Verbindungen unter Linux, macOS oder Unix mit dem AWS CLI Credential Helper:](#) Folgen Sie diesen Anweisungen, um eine step-by-step exemplarische Vorgehensweise zur Installation und Einrichtung des Credential Helper auf Linux-, macOS- oder Unix-Systemen zu erhalten.
- [Für HTTPS-Verbindungen unter Windows mit dem Credential Helper AWS CLI :](#) Folgen Sie diesen Anweisungen, um eine step-by-step exemplarische Vorgehensweise zur Installation und Einrichtung des Credential Helper auf Windows-Systemen zu erhalten.

Wenn Sie eine Verbindung zu einem CodeCommit Repository herstellen, das in einem anderen Amazon Web Services Services-Konto gehostet wird, können Sie den Zugriff konfigurieren und Verbindungen mithilfe von Rollen, Richtlinien und dem Credential Helper einrichten, der in der AWS CLI enthalten ist.

- [Konfigurieren Sie den kontoübergreifenden Zugriff auf ein AWS CodeCommit Repository mithilfe von Rollen:](#) Folgen Sie diesen Anweisungen für eine step-by-step exemplarische Vorgehensweise zur Konfiguration des kontoübergreifenden Zugriffs in einem Amazon Web Services Services-Konto für Benutzer in einer IAM-Gruppe in einem anderen Amazon Web Services Services-Konto.

Kompatibilität für CodeCommit Git und andere Komponenten

Wenn Sie mit arbeiten CodeCommit, verwenden Sie Git. Sie können aber auch andere Programme verwenden. Die folgende Tabelle enthält die neuesten Hinweise zur Versionskompatibilität. Als bewährte Methode empfehlen wir, die neuesten Versionen von Git und anderer Software zu verwenden.

Informationen zur Versionskompatibilität für AWS CodeCommit

Komponente	Version
Git	CodeCommit unterstützt Git-Versionen 1.7.9 und höher. Git Version 2.28 unterstützt die Konfiguration des Branchnamens für anfängliche Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git.
Curl	CodeCommit benötigt Curl 7.33 und höher. Es gibt jedoch ein bekanntes Problem mit HTTPS und dem Curl-Update 7.41.0. Weitere Informationen finden Sie unter Fehlerbehebung .
Python (nur git-remote-codecommit)	git-remote-codecommit benötigt Version 3 und höher.
Pip (nur git-remote-codecommit)	git-remote-codecommit benötigt Version 9.0.3 und höher.
AWS CLI (nur git-remote-codecommit)	Wir empfehlen allen CodeCommit Benutzern eine aktuelle AWS CLI Version von Version 2. git-remote-codecommit erfordert AWS CLI Version 2 zur Unterstützung von AWS SSO und Verbindungen, für die temporäre Anmeldeinformationen erforderlich sind, wie z. B. Verbundbenutzer.

Einrichtung für HTTPS-Benutzer mit Git-Anmeldeinformationen

Die einfachste Methode, Verbindungen zu AWS CodeCommit Repositorys einzurichten, besteht darin, Git-Anmeldeinformationen für CodeCommit in der IAM-Konsole zu konfigurieren und diese Anmeldeinformationen dann für HTTPS-Verbindungen zu verwenden. Sie können dieselben Anmeldeinformationen auch mit jedem Drittanbieter-Tool oder jeder integrierten Entwicklungsumgebung (IDE) verwenden, die die HTTPS-Authentifizierung mit einem statischen Benutzernamen und Passwort unterstützt. Beispiele finden Sie unter [Für Verbindungen von Entwicklungstools](#).

 Note

Wenn Sie Ihren lokalen Computer zuvor so konfiguriert haben, dass er den Credential Helper für verwendet CodeCommit, müssen Sie Ihre .gitconfig-Datei bearbeiten, um die Credential Helper-Informationen aus der Datei zu entfernen, bevor Sie Git-Anmeldeinformationen verwenden können. Wenn auf Ihrem lokalen Computer macOS ausgeführt wird, müssen Sie möglicherweise die zwischengespeicherten Anmeldeinformationen aus Keychain Access löschen.

Schritt 1: Erstkonfiguration für CodeCommit

Gehen Sie wie folgt vor, um ein Amazon Web Services Services-Konto einzurichten, einen IAM-Benutzer zu erstellen und den Zugriff auf zu CodeCommit konfigurieren.

Um einen IAM-Benutzer für den Zugriff zu erstellen und zu konfigurieren CodeCommit

1. Erstellen Sie ein Amazon Web Services Services-Konto, indem Sie zu <http://aws.amazon.com> gehen und Sign Up wählen.
2. Erstellen Sie einen IAM-Benutzer oder verwenden Sie einen vorhandenen in Ihrem Amazon Web Services Services-Konto. Stellen Sie sicher, dass Sie über eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel verfügen, die diesem IAM-Benutzer zugeordnet sind. Weitere Informationen finden Sie unter [Einen IAM-Benutzer in Ihrem Amazon Web Services Services-Konto erstellen](#).

 Note

CodeCommit erfordert AWS Key Management Service. Wenn Sie einen vorhandenen IAM-Benutzer verwenden, stellen Sie sicher, dass dem Benutzer keine Richtlinien zugeordnet sind, die ausdrücklich die von CodeCommit erforderlichen AWS KMS Aktionen verweigern. Weitere Informationen finden Sie unter [AWS KMS und Verschlüsselung](#).

3. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter. <https://console.aws.amazon.com/iam/>
4. Wählen Sie in der IAM-Konsole im Navigationsbereich Benutzer und dann den IAM-Benutzer aus, den Sie für den Zugriff konfigurieren möchten. CodeCommit

5. Wählen Sie auf der Registerkarte Permissions die Option Add Permissions.
6. Wählen Sie unter Grant permissions die Option Attach existing policies directly aus.
7. Wählen Sie AWSCodeCommitPowerUser aus der Liste der Richtlinien eine andere verwaltete Richtlinie für CodeCommit den Zugriff aus. Weitere Informationen finden Sie unter [AWS verwaltete Richtlinien für CodeCommit](#).

Nachdem Sie die Richtlinie ausgewählt haben, die Sie anhängen möchten, wählen Sie Weiter: Überprüfen aus, um die Liste der Richtlinien zu überprüfen, die an den IAM-Benutzer angehängt werden sollen. Ist die Liste korrekt, wählen Sie Add permissions aus.

Weitere Informationen zu CodeCommit verwalteten Richtlinien und zur gemeinsamen Nutzung des Zugriffs auf Repositorys mit anderen Gruppen und Benutzern finden Sie unter [Teilen Sie ein Repository](#) und [Authentifizierung und Zugriffskontrolle für AWS CodeCommit](#)

Wenn Sie AWS CLI Befehle mit verwenden möchten CodeCommit, installieren Sie den AWS CLI. Wir empfehlen Ihnen, ein Profil für die Verwendung von AWS CLI with zu erstellen CodeCommit. Weitere Informationen finden Sie unter [Befehlszeilenreferenz](#) [Benannte Profile verwenden](#).

Schritt 2: Installieren von Git

Um mit Dateien, Commits und anderen Informationen in CodeCommit Repositorys zu arbeiten, müssen Sie Git auf Ihrem lokalen Computer installieren. CodeCommit unterstützt Git-Versionen 1.7.9 und höher. Git Version 2.28 unterstützt die Konfiguration des Branchnamens für anfängliche Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git.

Um Git zu installieren, empfehlen wir Websites wie [Git Downloads](#).

Note

Git ist eine sich entwickelnde, regelmäßig aktualisierte Plattform. Gelegentlich kann sich eine Änderung der Funktionen auf die Art und Weise auswirken, mit der es funktioniert CodeCommit. Wenn du Probleme mit einer bestimmten Version von Git und hast CodeCommit, sieh dir die Informationen unter an [Fehlerbehebung](#).

Schritt 3: Erstellen Sie Git-Anmeldeinformationen für HTTPS-Verbindungen zu CodeCommit

Nachdem Sie Git installiert haben, erstellen Sie Git-Anmeldeinformationen für Ihren IAM-Benutzer in IAM.

So richten Sie HTTPS-Git-Anmeldeinformationen ein für CodeCommit

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter <https://console.aws.amazon.com/iam/>.

Stellen Sie sicher, dass Sie sich als der IAM-Benutzer anmelden, der die Git-Anmeldeinformationen für Verbindungen zu CodeCommit erstellt und verwendet.

2. Wählen Sie in der IAM-Konsole im Navigationsbereich Benutzer aus und wählen Sie aus der Benutzerliste Ihren IAM-Benutzer aus.

Note

Sie können Ihre CodeCommit Anmeldeinformationen direkt unter Meine Sicherheitsanmeldedaten einsehen und verwalten. Weitere Informationen finden Sie unter [Ihre Anmeldedaten einsehen und verwalten](#).

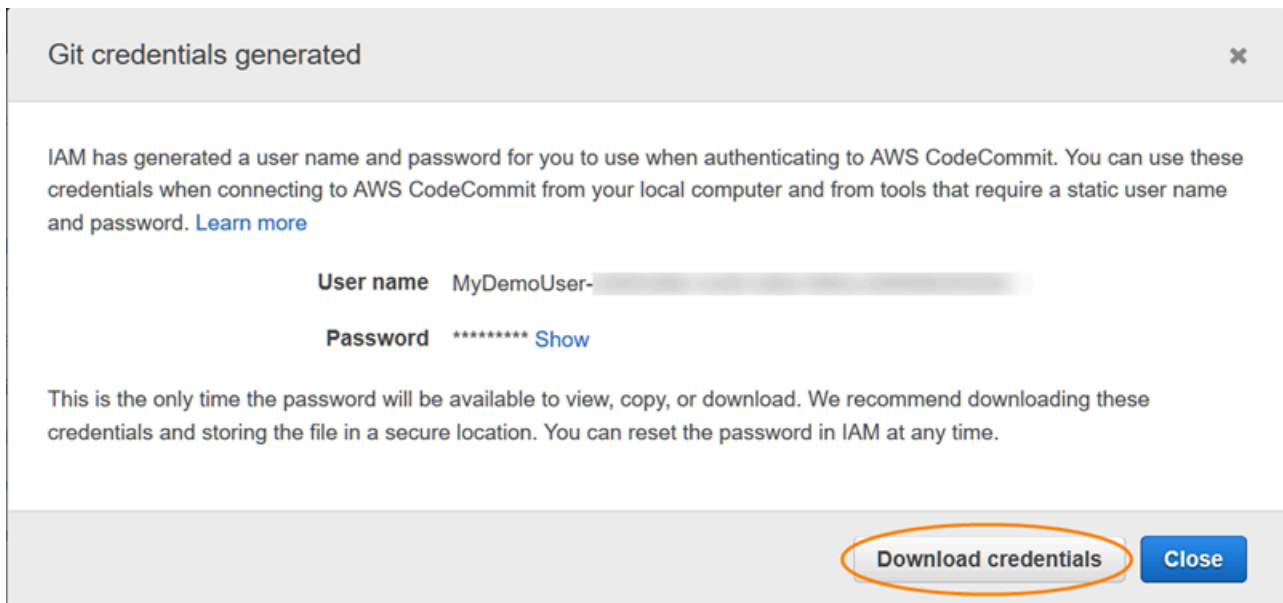
3. Wählen Sie auf der Seite mit den Benutzerdetails die Registerkarte Sicherheitsanmeldedaten und wählen Sie unter HTTPS-Git-Anmeldeinformationen für die AWS CodeCommit Option Generieren aus.



Note

Sie können nicht Ihren eigenen Benutzernamen oder das Passwort für Git-Anmeldeinformationen wählen. Weitere Informationen finden Sie unter [Verwenden von Git-Anmeldeinformationen und HTTPS mit CodeCommit](#).

4. Kopieren Sie den Benutzernamen und das Passwort, die IAM für Sie generiert hat, indem Sie diese Informationen entweder anzeigen, kopieren und dann in eine sichere Datei auf Ihrem lokalen Computer einfügen, oder indem Sie Anmeldeinformationen herunterladen wählen, um diese Informationen als CSV-Datei herunterzuladen. Sie benötigen diese Informationen für die Verbindung mit CodeCommit.



Wählen Sie, nachdem Sie Ihre Anmeldeinformationen gespeichert haben, die Option Close aus.

Important

Dies ist Ihre einzige Möglichkeit, den Benutzernamen und das Passwort zu speichern. Wenn Sie sie nicht speichern, können Sie den Benutzernamen von der IAM-Konsole kopieren, aber Sie können das Passwort nicht nachschlagen. Sie müssen das Passwort zurücksetzen und es dann speichern.

Schritt 4: Connect zur CodeCommit Konsole her und klonen Sie das Repository

Wenn Sie den Namen und die Verbindungsdetails für das CodeCommit-Repository bereits von einem Administrator erhalten haben, können Sie diesen Schritt überspringen und das Repository direkt klonen.

Um eine Verbindung zu einem CodeCommit Repository herzustellen

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie in der Regionsauswahl den Ort aus, AWS-Region an dem das Repository erstellt wurde. Repositorys sind spezifisch für ein. AWS-Region Weitere Informationen finden Sie unter [Regionen und Git-Verbindungsendpunkte](#).
3. Suchen Sie das Repository, zu dem Sie eine Verbindung herstellen möchten, in der Liste und wählen Sie es aus. Wählen Sie Clone URL (URL klonen) und dann das Protokoll aus, das Sie beim Klonen oder bei der Verbindung zu dem Repository verwenden möchten. Dadurch wird die Klon-URL kopiert.
 - Kopieren Sie die HTTPS-URL, wenn Sie entweder Git-Anmeldeinformationen mit Ihrem IAM-Benutzer oder den Credential Helper verwenden, der im Lieferumfang von enthalten ist. AWS CLI
 - Kopieren Sie die HTTPS-URL (GRC), wenn Sie den Befehl git-remote-codecommit auf Ihrem lokalen Computer verwenden.
 - Kopieren Sie die SSH-URL, wenn Sie ein öffentliches/privates SSH-Schlüsselpaar mit Ihrem IAM-Benutzer verwenden.

Note

Wenn Sie statt einer Liste von Repositorys eine Willkommensseite sehen, sind in dem Land, in dem Sie angemeldet sind, keine Repositorys mit Ihrem AWS Konto verknüpft. AWS-Region Informationen zur Erstellung eines Repositorys finden Sie unter [the section called "Erstellen eines -Repositorys"](#), oder befolgen Sie die Schritte im Tutorial [Erste Schritte mit Git und CodeCommit](#).

4. Öffnen Sie ein Terminal- oder Befehlszeilenfenster oder eine Git-Shell. Führen Sie den `git clone`-Befehl mit der HTTPS-Klon-URL aus, die Sie kopiert haben, um das Repository zu klonen. Um beispielsweise ein Repository mit dem Namen *MyDemoRepo* eines lokalen Repositorys mit dem Namen *my-demo-repo* der Region USA Ost (Ohio) zu klonen:

```
git clone https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
```

Wenn Sie zum ersten Mal die Verbindung herstellen, werden Sie aufgefordert, den Benutzernamen und das Passwort für das Repository einzugeben. Abhängig von der Konfiguration Ihres lokalen Computers stammt diese Aufforderung entweder von einem Anmeldeinformationsverwaltungssystem für das Betriebssystem, einem Credential Manager-Hilfsprogramm für Ihre Version von Git (z. B. dem in Git für Windows enthaltenen Git Credential Manager), Ihrer IDE oder von Git selbst. Geben Sie den Benutzernamen und das Passwort ein, die für Git-Anmeldeinformationen in IAM generiert wurden (die, die Sie in [Schritt 3: Erstellen Sie Git-Anmeldeinformationen für HTTPS-Verbindungen zu CodeCommit](#) erstellt haben). Je nach Betriebssystem und anderer Software werden diese Daten möglicherweise für Sie in einem Speicher für Anmeldeinformationen oder einem Dienstprogramm zur Verwaltung von Anmeldeinformationen gespeichert. In diesem Fall sollten Sie nicht erneut dazu aufgefordert werden, es sei denn, Sie ändern das Passwort, deaktivieren die Git-Anmeldeinformationen oder löschen die Git-Anmeldeinformationen in IAM.

Wenn auf dem lokalen Computer kein Speicher für Anmeldeinformationen und kein Dienstprogramm zur Verwaltung von Anmeldeinformationen konfiguriert ist, können Sie einen Speicher bzw. ein Dienstprogramm installieren. Weitere Informationen über Git und darüber, wie es Anmeldeinformationen verwaltet, finden Sie in der Git-Dokumentation unter [Credential Storage](#).

Weitere Informationen erhalten Sie unter [Connect zum CodeCommit Repository her, indem Sie das Repository klonen](#) und [Erstellen Sie einen Commit](#).

Nächste Schritte

Sie haben die Voraussetzungen erfüllt. Folgen Sie den Schritten unter, um mit der Nutzung [Erste Schritte mit CodeCommit](#) zu beginnen. CodeCommit

Informationen dazu, wie Sie Ihren ersten Commit erstellen und pushen, finden Sie unter [Erstellen Sie einen Commit in AWS CodeCommit](#). Wenn Sie mit Git noch nicht vertraut sind, finden Sie unter

[Wo kann ich mehr über Git erfahren?](#) und [Erste Schritte mit Git und AWS CodeCommit](#) genauere Informationen.

Einrichtungsschritte für HTTPS-Verbindungen AWS CodeCommit mit git-remote-codecommit

Wenn Sie CodeCommit mit einem Root-Konto, Verbundzugriff oder temporären Anmeldeinformationen eine Verbindung herstellen möchten, sollten Sie den Zugriff über git-remote-codecommit einrichten. Dieses Hilfsprogramm bietet eine einfache Methode zum Pushen und Abrufen von Code aus CodeCommit Repositorys, indem es Git erweitert. Es ist die empfohlene Methode zur Unterstützung von Verbindungen, die mit Verbundzugriff, Identitätsanbietern und temporären Anmeldeinformationen hergestellt werden. Um einer föderierten Identität Berechtigungen zuzuweisen, erstellen Sie eine Rolle und definieren Berechtigungen für die Rolle. Wird eine Verbundidentität authentifiziert, so wird die Identität der Rolle zugeordnet und erhält die von der Rolle definierten Berechtigungen. Informationen zu Rollen für den Verbund finden Sie unter [Erstellen von Rollen für externe Identitätsanbieter \(Verbund\)](#) im IAM-Benutzerhandbuch. Wenn Sie IAM Identity Center verwenden, konfigurieren Sie einen Berechtigungssatz. Wenn Sie steuern möchten, worauf Ihre Identitäten nach der Authentifizierung zugreifen können, korreliert IAM Identity Center den Berechtigungssatz mit einer Rolle in IAM. Informationen zu Berechtigungssätzen finden Sie unter [Berechtigungssätze](#) im AWS IAM Identity Center -Benutzerhandbuch.

Sie können es auch git-remote-codecommit mit einem IAM-Benutzer verwenden. Im Gegensatz zu anderen HTTPS-Verbindungsmethoden erfordert git-remote-codecommit das Einrichten von Git-Anmeldeinformationen für den Benutzer nicht.

Note

Einige unterstützen das von verwendete Klon-URL-Format IDEs nicht. git-remote-codecommit Möglicherweise müssen Sie Repositorys manuell auf Ihren lokalen Computer klonen, bevor Sie mit ihnen in der bevorzugten IDE arbeiten können. Weitere Informationen finden Sie unter [Problembehebung git-remote-codecommit und AWS CodeCommit](#).

Diese Verfahren wurden unter der Annahme geschrieben, dass Sie über ein Amazon Web Services Services-Konto verfügen, mindestens ein Repository erstellt haben und einen IAM-Benutzer mit einer verwalteten Richtlinie verwenden CodeCommit, wenn Sie eine Verbindung zu CodeCommit Repositorys herstellen. Informationen zum Konfigurieren des Zugriffs für verbundener Benutzer und

andere rotierende Anmeldeinformationstypen finden Sie unter [Verbindung zu AWS CodeCommit Repositories mit rotierenden Anmeldeinformationen herstellen](#).

Themen

- [Schritt 0: Installieren Sie die Voraussetzungen für git-remote-codecommit](#)
- [Schritt 1: Erstkonfiguration für CodeCommit](#)
- [Schritt 2: Installieren git-remote-codecommit](#)
- [Schritt 3: Connect zur CodeCommit Konsole her und klonen Sie das Repository](#)
- [Nächste Schritte](#)

Schritt 0: Installieren Sie die Voraussetzungen für git-remote-codecommit

Bevor Sie git-remote-codecommit verwenden können, müssen Sie einige Voraussetzungen auf Ihrem lokalen Computer installieren. Dazu zählen:

- Eine unterstützte Version von Python. Weitere Hinweise zu unterstützten Versionen von Python finden Sie unter [git-remote-codecommit](#).

Weitere Informationen zur Einrichtung und Verwendung git-remote-codecommit finden Sie unter [Einrichtungsschritte für HTTPS-Verbindungen AWS CodeCommit mit git-remote-codecommit](#).

- Git

Note

Wenn Sie Python unter Windows installieren, stellen Sie sicher, dass Sie die Option auswählen, mit der sich Python zum Pfad hinzufügen lässt.

git-remote-codecommit benötigt pip Version 9.0.3 oder höher. Um Ihre pip-Version zu überprüfen, öffnen Sie ein Terminal oder eine Befehlszeile und führen Sie den folgenden Befehl aus:

```
pip --version
```

Sie können die folgenden beiden Befehle ausführen, um Ihre pip-Version auf die neueste Version zu aktualisieren:

```
curl -O https://bootstrap.pypa.io/get-pip.py
python3 get-pip.py --user
```

Um mit Dateien, Commits und anderen Informationen in CodeCommit Repositorys zu arbeiten, müssen Sie Git auf Ihrem lokalen Computer installieren. CodeCommit unterstützt Git-Versionen 1.7.9 und höher. Git Version 2.28 unterstützt die Konfiguration des Branchnamens für anfängliche Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git.

Um Git zu installieren, empfehlen wir Websites wie [Git Downloads](#).

Note

Git ist eine sich entwickelnde, regelmäßig aktualisierte Plattform. Gelegentlich kann sich eine Änderung der Funktionen auf die Art und Weise auswirken, mit der es funktioniert CodeCommit. Wenn du Probleme mit einer bestimmten Version von Git und hast CodeCommit, sieh dir die Informationen unter an [Fehlerbehebung](#).

Schritt 1: Erstkonfiguration für CodeCommit

Gehen Sie wie folgt vor, um einen IAM-Benutzer zu erstellen, ihn mit den entsprechenden Richtlinien zu konfigurieren, einen Zugriffsschlüssel und einen geheimen Schlüssel zu erhalten und den AWS CLI zu installieren und zu konfigurieren.

Um einen IAM-Benutzer für den Zugriff zu erstellen und zu konfigurieren CodeCommit

1. Erstellen Sie ein Amazon Web Services Services-Konto, indem Sie zu <http://aws.amazon.com> gehen und Sign Up wählen.
2. Erstellen Sie einen IAM-Benutzer oder verwenden Sie einen vorhandenen in Ihrem Amazon Web Services Services-Konto. Stellen Sie sicher, dass Sie über eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel verfügen, die diesem IAM-Benutzer zugeordnet sind. Weitere Informationen finden Sie unter [Einen IAM-Benutzer in Ihrem Amazon Web Services Services-Konto erstellen](#).

Note

CodeCommit erfordert AWS Key Management Service. Wenn Sie einen vorhandenen IAM-Benutzer verwenden, stellen Sie sicher, dass dem Benutzer keine Richtlinien

zugeordnet sind, die ausdrücklich die von CodeCommit erforderlichen AWS KMS Aktionen verweigern. Weitere Informationen finden Sie unter [AWS KMS und Verschlüsselung](#).

3. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter <https://console.aws.amazon.com/iam/>
4. Wählen Sie in der IAM-Konsole im Navigationsbereich Benutzer und dann den IAM-Benutzer aus, den Sie für den Zugriff konfigurieren möchten. CodeCommit
5. Wählen Sie auf der Registerkarte Permissions die Option Add Permissions.
6. Wählen Sie unter Grant permissions die Option Attach existing policies directly aus.
7. Wählen Sie AWSCodeCommitPowerUser aus der Liste der Richtlinien eine andere verwaltete Richtlinie für CodeCommit den Zugriff aus. Weitere Informationen finden Sie unter [AWS verwaltete Richtlinien für CodeCommit](#).

Nachdem Sie die Richtlinie ausgewählt haben, die Sie anhängen möchten, wählen Sie Weiter: Überprüfen aus, um die Liste der Richtlinien zu überprüfen, die an den IAM-Benutzer angehängt werden sollen. Ist die Liste korrekt, wählen Sie Add permissions aus.

Weitere Informationen zu CodeCommit verwalteten Richtlinien und zur gemeinsamen Nutzung des Zugriffs auf Repositorys mit anderen Gruppen und Benutzern finden Sie unter [Teilen Sie ein Repository](#) und [Authentifizierung und Zugriffskontrolle für AWS CodeCommit](#)

Um das zu installieren und zu konfigurieren AWS CLI

1. Laden Sie auf Ihrem lokalen Computer den herunter und installieren Sie ihn AWS CLI. Dies ist eine Voraussetzung für die Interaktion mit CodeCommit über die Befehlszeile. Wir empfehlen, AWS CLI Version 2 zu installieren. Es ist die neueste Hauptversion von AWS CLI und unterstützt alle aktuellen Funktionen. Es ist die einzige Version von AWS CLI , die die Verwendung eines Root-Kontos, Verbundzugriff oder temporärer Anmeldeinformationen mit git-remote-codecommit unterstützt.

Weitere Informationen finden Sie [unter Einrichtung der AWS Befehlszeilenschnittstelle](#).

Note

CodeCommit funktioniert nur mit den AWS CLI Versionen 1.7.38 und höher. Es hat sich bewährt, die neueste verfügbare Version zu installieren oder AWS CLI auf sie zu

aktualisieren. Führen AWS CLI Sie den `aws --version` Befehl aus, um festzustellen, welche Version von Sie installiert haben.

Informationen zum Upgrade einer älteren Version von AWS CLI auf die neueste Version finden Sie unter [Installation von AWS Command Line Interface](#).

2. Führen Sie diesen Befehl aus, um zu überprüfen, ob die CodeCommit Befehle für installiert AWS CLI sind.

```
aws codecommit help
```

Dieser Befehl gibt eine Liste von CodeCommit Befehlen zurück.

3. Konfigurieren Sie das AWS CLI mit einem Profil, indem Sie den `configure` Befehl verwenden, wie folgt:.

```
aws configure
```

Wenn Sie dazu aufgefordert werden, geben Sie den AWS Zugriffsschlüssel und den AWS geheimen Zugriffsschlüssel des IAM-Benutzers an, mit CodeCommit dem Sie ihn verwenden möchten. Stellen Sie außerdem sicher, dass Sie angeben, AWS-Region wo sich das Repository befindet, z. B. `us-east-2` Wenn Sie nach dem standardmäßigen Ausgabeformat gefragt werden, geben Sie `json` an. Wenn Sie beispielsweise ein Profil für einen IAM-Benutzer konfigurieren:

```
AWS Access Key ID [None]: Type your IAM user AWS access key ID here, and then press Enter
AWS Secret Access Key [None]: Type your IAM user AWS secret access key here, and then press Enter
Default region name [None]: Type a supported region for CodeCommit here, and then press Enter
Default output format [None]: Type json here, and then press Enter
```

Weitere Informationen zum Erstellen und Konfigurieren von Profilen zur Verwendung mit dem AWS CLI finden Sie im Folgenden:

- [Benannte Profile](#)
- [Verwenden einer IAM-Rolle in der AWS CLI](#)
- [Befehl „Set“](#)

- [Verbindung zu AWS CodeCommit Repositorys mit rotierenden Anmeldeinformationen herstellen](#)

Um eine Verbindung zu einem Repository oder einer Ressource in einem anderen Repository herzustellen AWS-Region, müssen Sie das AWS CLI mit dem Standardregionsnamen neu konfigurieren. Zu den unterstützten Standardregionsnamen für CodeCommit gehören:

- us-east-2
- us-east-1
- eu-west-1
- us-west-2
- ap-northeast-1
- ap-southeast-1
- ap-southeast-2
- ap-southeast-3
- me-central-1
- eu-central-1
- ap-northeast-2
- sa-east-1
- us-west-1
- eu-west-2
- ap-south-1
- ap-south-1
- ca-central-1
- us-gov-west-1
- us-gov-east-1
- eu-north-1
- ap-east-1
- me-south-1
- cn-north-1
- cn-northwest-1

- eu-south-1
- ap-northeast-3
- af-south-1
- il-central-1

Weitere Informationen zu CodeCommit und finden AWS-Region Sie unter [Regionen und Git-Verbindungsendpunkte](#). Weitere Informationen zu IAM, Zugriffsschlüsseln und geheimen Schlüsseln finden Sie unter [Wie erhalte ich Anmeldeinformationen?](#) und [Zugriffsschlüssel für IAM-Benutzer verwalten](#). Weitere Informationen zu den Profilen AWS CLI und finden Sie unter [Benannte Profile](#).

Schritt 2: Installieren git-remote-codecommit

Führen Sie zur Installation von git-remote-codecommit die folgenden Schritte aus.

Um zu installieren git-remote-codecommit

1. Führen Sie am Terminal oder in der Befehlszeile den folgenden Befehl aus:

```
pip install git-remote-codecommit
```

Note

Je nach Betriebssystem und Konfiguration müssen Sie diesen Befehl möglicherweise mit erhöhten Rechten ausführen, z. B. `sudo`, oder den Parameter `--user` verwenden, um in einem Verzeichnis zu installieren, für das keine besonderen Rechte erforderlich sind, wie z. B. Ihr aktuelles Benutzerkonto. Zum Beispiel auf einem Computer, auf dem Linux, macOS oder Unix ausgeführt wird:

```
sudo pip install git-remote-codecommit
```

Auf einem Computer mit Windows:

```
pip install --user git-remote-codecommit
```

2. Überwachen Sie den Installationsvorgang, bis Sie eine Erfolgsmeldung sehen.

Schritt 3: Connect zur CodeCommit Konsole her und klonen Sie das Repository

Wenn Ihnen ein Administrator bereits die Klon-URL geschickt hat, die Sie git-remote-codecommit für das CodeCommit Repository verwenden möchten, können Sie die Verbindung zur Konsole überspringen und das Repository direkt klonen.

Um eine Verbindung zu einem CodeCommit Repository herzustellen

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie in der Regionsauswahl den Ort aus, AWS-Region an dem das Repository erstellt wurde. Repositories sind spezifisch für eine AWS-Region. Weitere Informationen finden Sie unter [Regionen und Git-Verbindungsendpunkte](#).
3. Suchen Sie das Repository, zu dem Sie eine Verbindung herstellen möchten, in der Liste und wählen Sie es aus. Wählen Sie Clone URL (URL klonen) und dann das Protokoll aus, das Sie beim Klonen oder bei der Verbindung zu dem Repository verwenden möchten. Dadurch wird die Klon-URL kopiert.
 - Kopieren Sie die HTTPS-URL, wenn Sie entweder Git-Anmeldeinformationen mit Ihrem IAM-Benutzer oder den Credential Helper verwenden, der im Lieferumfang von enthalten ist. AWS CLI
 - Kopieren Sie die HTTPS-URL (GRC), wenn Sie den Befehl git-remote-codecommit auf Ihrem lokalen Computer verwenden.
 - Kopieren Sie die SSH-URL, wenn Sie ein öffentliches/privates SSH-Schlüsselpaar mit Ihrem IAM-Benutzer verwenden.

Note

Wenn Sie statt einer Liste von Repositories eine Willkommenseite sehen, sind Ihrem AWS Konto in dem Land, in dem Sie angemeldet sind, keine Repositories zugeordnet. AWS-Region Informationen zur Erstellung eines Repositories finden Sie unter [the section called "Erstellen eines -Repositories"](#), oder befolgen Sie die Schritte im Tutorial [Erste Schritte mit Git und CodeCommit](#).

4. Klonen Sie am Terminal oder an der Eingabeaufforderung das Repository mit dem `git clone`-Befehl. Verwenden Sie die kopierte `git-remote-codecommit` HTTPS-URL und den Namen des AWS CLI Profils, wenn Sie ein benanntes Profil erstellt haben. Wenn Sie kein Profil angeben, nimmt der Befehl das Standardprofil an. Dadurch wird das lokale Repository in einem Unterverzeichnis des Verzeichnisses erstellt, in dem Sie den Befehl ausgeführt haben. Im folgenden Beispiel wird ein Repository mit dem Namen *MyDemoRepo* in ein lokales Repository mit dem Namen *my-demo-repo* geklont:

```
git clone codecommit://MyDemoRepo my-demo-repo
```

Um dasselbe Repository mit einem Profil mit dem Namen zu klonen *CodeCommitProfile*:

```
git clone codecommit://CodeCommitProfile@MyDemoRepo my-demo-repo
```

Um ein Repository in einem anderen AWS-Region als dem in Ihrem Profil konfigurierten zu klonen, geben Sie den AWS-Region Namen an. Zum Beispiel:

```
git clone codecommit::ap-northeast-1://MyDemoRepo my-demo-repo
```

Nächste Schritte

Sie haben die Voraussetzungen erfüllt. Folgen Sie den Anweisungen unter [Erste Schritte mit CodeCommit](#), um mit der Verwendung zu beginnen CodeCommit.

Informationen dazu, wie Sie Ihren ersten Commit erstellen und pushen, finden Sie unter [Erstellen Sie einen Commit in AWS CodeCommit](#). Wenn Sie mit Git noch nicht vertraut sind, finden Sie unter [Wo kann ich mehr über Git erfahren?](#) und [Erste Schritte mit Git und AWS CodeCommit](#) genauere Informationen.

Verbindungen von Entwicklungstools mithilfe von Git-Anmeldeinformationen einrichten

Nachdem Sie die Git-Anmeldeinformationen für AWS CodeCommit in der IAM-Konsole konfiguriert haben, können Sie diese Anmeldeinformationen mit jedem Entwicklungstool verwenden, das Git-Anmeldeinformationen unterstützt. Sie können beispielsweise den Zugriff auf Ihr CodeCommit Repository in Visual Studio AWS Cloud9, Xcode, IntelliJ oder einer beliebigen integrierten

Entwicklungsumgebung (IDE) konfigurieren, die Git-Anmeldeinformationen integriert. Wenn Sie den Zugriff konfiguriert haben, können Sie den Code bearbeiten, die Änderungen per Commit übergeben und direkt per Push von der IDE oder einem anderen Entwicklungstool übertragen.

Note

Wenn Sie über Verbundzugriff, temporäre Anmeldeinformationen oder einen Web-Identitätsanbieter auf CodeCommit Repositorys zugreifen, können Sie keine Git-Anmeldeinformationen verwenden. Es wird empfohlen, den lokalen Computer mit dem `git-remote-codecommit`-Befehl einzurichten. Allerdings IDEs sind nicht alle vollständig kompatibel mit Git-Remote-Helfern wie `git-remote-codecommit`. Bei Problemen finden Sie weitere Informationen unter [Problembehebung git-remote-codecommit und AWS CodeCommit](#).

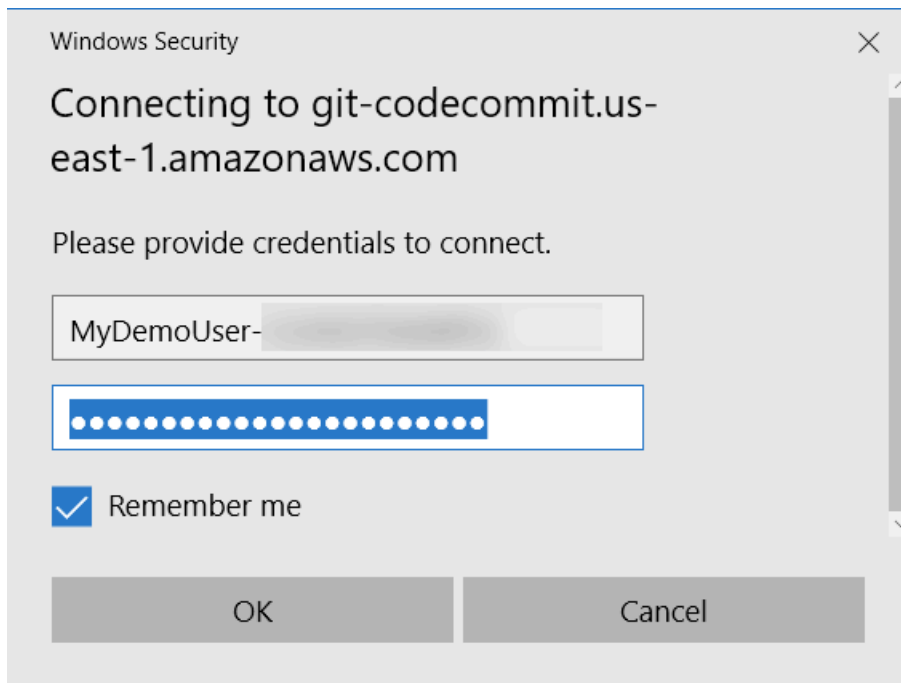
Themen

- [Integrieren AWS Cloud9 mit AWS CodeCommit](#)
- [Integrieren Sie Visual Studio mit AWS CodeCommit](#)

Wenn Sie von Ihrer IDE oder Ihrem Entwicklungstool nach dem Benutzernamen und dem Passwort gefragt werden, die für die Verbindung mit dem CodeCommit Repository verwendet wurden, geben Sie die Git-Anmeldeinformationen für den Benutzernamen und das Passwort ein, die Sie in IAM erstellt haben.

Weitere Informationen zu AWS-Regionen und Endpunkten für finden Sie CodeCommit unter [Regionen und Git-Verbindungsendpunkte](#)

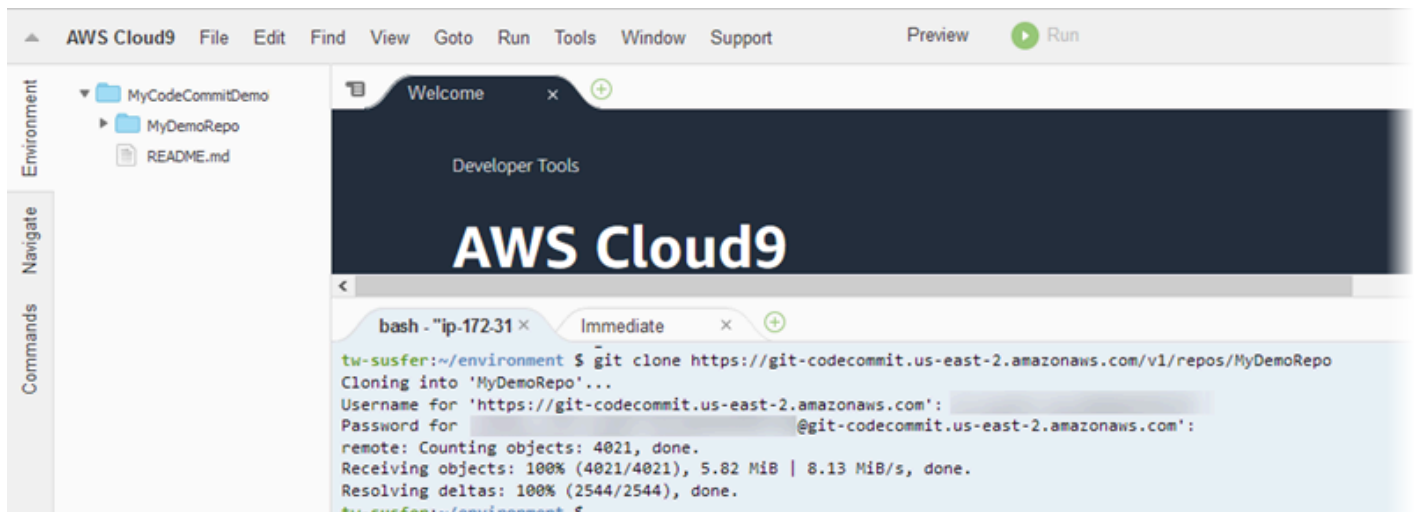
Möglicherweise werden Sie auch vom Betriebssystem aufgefordert, Ihren Benutzernamen und das Passwort zu speichern. Beispielsweise geben Sie in Windows Ihre Git-Anmeldeinformationen wie folgt ein:



Weitere Informationen zur Konfiguration der Git-Anmeldeinformationen für ein bestimmtes Softwareprogramm oder Entwicklungstool finden Sie in der Dokumentation zum jeweiligen Produkt.

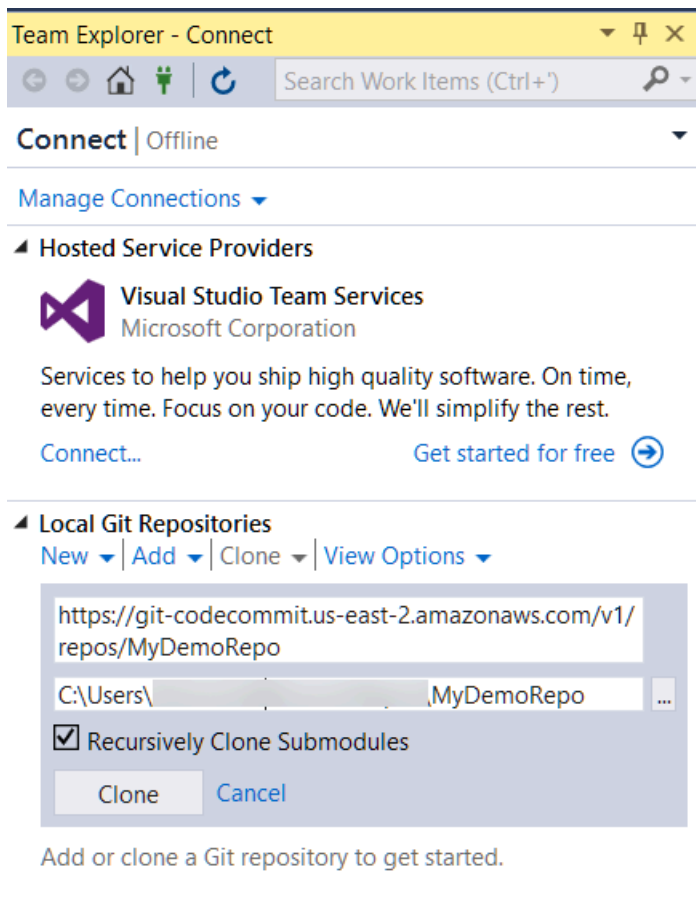
Die folgende Liste ist nicht vollständig. IDEs Die Links dienen ausschließlich dazu, Ihnen zu helfen, mehr über diese Tools zu erfahren. AWS ist nicht verantwortlich für den Inhalt dieser Themen.

- [AWS Cloud9](#)



- [Visual Studio](#)

Installieren Sie alternativ die AWS Toolkit for Visual Studio. Weitere Informationen finden Sie unter [Integrieren Sie Visual Studio mit AWS CodeCommit](#).



- [XCode](#)

Integrieren AWS Cloud9 mit AWS CodeCommit

Sie können AWS Cloud9 verwenden, um Codeänderungen in einem CodeCommit Repository vorzunehmen. AWS Cloud9 enthält eine Sammlung von Tools, mit denen Sie Code schreiben und Software erstellen, ausführen, testen, debuggen und veröffentlichen können. Sie können bestehende Repositories klonen, Repositories erstellen, Codeänderungen in ein Repository übertragen und dort speichern und vieles mehr — alles von Ihrer AWS Cloud9 EC2 Entwicklungsumgebung aus. Die AWS Cloud9 EC2 Entwicklungsumgebung ist in der Regel mit der AWS CLI, einer EC2 Amazon-Rolle und Git vorkonfiguriert, sodass Sie in den meisten Fällen ein paar einfache Befehle ausführen und mit der Interaktion mit Ihrem Repository beginnen können.

Für die Verwendung AWS Cloud9 mit CodeCommit benötigen Sie Folgendes:

- Eine AWS Cloud9 EC2 Entwicklungsumgebung, die auf Amazon Linux läuft.
- Die AWS Cloud9 IDE wird in einem Webbrowser geöffnet.

- Ein IAM-Benutzer, auf den eine der CodeCommit verwalteten Richtlinien und eine der AWS Cloud9 verwalteten Richtlinien angewendet wurden.

Weitere Informationen finden Sie unter [AWS verwaltete Richtlinien für CodeCommit](#) und [Verstehen und Abrufen Ihrer Sicherheitsanmeldeinformationen](#).

Note

In diesem Thema wird die Einrichtung der Integration mit CodeCommit und AWS Cloud9 mit allgemeinem Zugriff über das Internet beschrieben. Sie können den Zugriff auf CodeCommit und AWS Cloud9 in einer isolierten Umgebung einrichten, dafür sind jedoch zusätzliche Schritte erforderlich. Weitere Informationen finden Sie unter:

- [Verwendung AWS CodeCommit mit VPC-Endpunkten mit Schnittstelle](#)
- [Zugriff auf EC2 No-Ingress-Amazon-Instances mit AWS Systems Manager](#)
- [Arbeiten mit gemeinsamen Umgebungen](#)
- [Teilen Sie Ihre VPC mit anderen Konten](#)
- [Blogbeitrag: Isolieren Sie den Netzwerkzugriff auf Ihre Umgebungen AWS Cloud9](#)

Themen

- [Schritt 1: Erstellen Sie eine AWS Cloud9 Entwicklungsumgebung](#)
- [Schritt 2: Konfigurieren Sie den AWS CLI Credential Helper in Ihrer AWS Cloud9 EC2 Entwicklungsumgebung](#)
- [Schritt 3: Klonen Sie ein CodeCommit Repository in Ihre Entwicklungsumgebung AWS Cloud9 EC2](#)
- [Nächste Schritte](#)

Schritt 1: Erstellen Sie eine AWS Cloud9 Entwicklungsumgebung

AWS Cloud9 hostet Ihre Entwicklungsumgebung auf einer EC2 Amazon-Instance. Dies ist der einfachste Weg zur Integration, da Sie die AWS verwalteten temporären Anmeldeinformationen für die Instance verwenden können, um eine Verbindung zu Ihrem CodeCommit Repository herzustellen. Wenn Sie stattdessen Ihren eigenen Server verwenden möchten, finden Sie weitere Informationen im [AWS Cloud9 Benutzerhandbuch](#).

Um eine AWS Cloud9 Umgebung zu erstellen

1. Melden Sie sich AWS als der IAM-Benutzer an, den Sie konfiguriert haben, und öffnen Sie die AWS Cloud9 Konsole.
2. Wählen Sie in der AWS Cloud9 Konsole Umgebung erstellen aus.
3. Geben Sie unter Step 1: Name environment (Schritt 1: Benennen der Umgebung) einen Namen und eine optionale Beschreibung für die Umgebung ein und klicken Sie dann auf Next step (Nächster Schritt).
4. Konfigurieren Sie Ihre Umgebung unter Step 2: Configure Settings wie folgt:
 - Wählen Sie unter Umgebungstyp die Option Neue Instanz für Umgebung erstellen (EC2) aus.
 - Klicken Sie unter Instance type auf den entsprechenden Instance-Typ für Ihre Entwicklungsumgebung. Wenn Sie sich nur mit dem Service vertraut machen möchten, wählen Sie die Standardoption t2.micro aus. Wenn Sie diese Umgebung für die Entwicklung verwenden möchten, wählen Sie einen größeren Instance-Typ aus.
 - Akzeptieren Sie die anderen Standardeinstellungen, es sei denn, Sie haben Gründe, sich anders zu entscheiden (z. B. verwendet Ihre Organisation eine bestimmte VPC oder Ihr Amazon Web Services Services-Konto hat keine VPCs konfiguriert), und wählen Sie dann Weiter.
5. Überprüfen Sie unter Step 3: Review die Einstellungen. Wählen Sie Previous step aus, wenn Sie Änderungen vornehmen möchten. Wählen Sie andernfalls Create environment aus.

Das Erstellen einer Umgebung und das erstmalige Herstellen einer Verbindung dauert einige Minuten. Wenn dieser Vorgang ungewöhnlich lange dauert, finden Sie weitere Informationen unter [Fehlerbehebung](#) im AWS Cloud9 Benutzerhandbuch.

6. Nachdem Sie eine Verbindung zu Ihrer Umgebung hergestellt haben, prüfen Sie, ob Git bereits installiert und eine unterstützte Version ist, indem Sie den Git-Befehl `git --version` im Terminal-Fenster ausführen.

Wenn Git nicht installiert ist oder es sich nicht um eine unterstützte Version handelt, installieren Sie eine unterstützte Version. CodeCommit unterstützt Git-Versionen 1.7.9 und höher. Git Version 2.28 unterstützt die Konfiguration des Branchnamens für anfängliche Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git. Um Git zu installieren, empfehlen wir Websites wie [Git Downloads](#).

 Tip

Je nach Betriebssystem Ihrer Umgebung können Sie mit dem Befehl `yum` und der Option `sudo` Updates, einschließlich Git, installieren. Eine administrative Befehlssequenz ähnelt beispielsweise den folgenden drei Befehlen:

```
sudo yum -y update
sudo yum -y install git
git --version
```

7. Konfigurieren Sie einen Benutzernamen und eine E-Mail-Adresse, die mit Ihren Git-Commits verknüpft werden, indem Sie den Befehl `git config` ausführen. Zum Beispiel:

```
git config --global user.name "Mary Major"
git config --global user.email mary.major@example.com
```

Schritt 2: Konfigurieren Sie den AWS CLI Credential Helper in Ihrer AWS Cloud9 EC2 Entwicklungsumgebung

Nachdem Sie eine AWS Cloud9 Umgebung erstellt haben, können Sie den AWS CLI Credential Helper so konfigurieren, dass er die Anmeldeinformationen für Verbindungen zu Ihrem CodeCommit Repository verwaltet. Die AWS Cloud9 Entwicklungsumgebung enthält AWS verwaltete temporäre Anmeldeinformationen, die Ihrem IAM-Benutzer zugeordnet sind. Sie verwenden diese Anmeldeinformationen mit dem AWS CLI Credential Helper.

1. Öffnen Sie das Terminal-Fenster und führen Sie den folgenden Befehl aus, um zu überprüfen, dass die AWS CLI installiert ist:

```
aws --version
```

Im Erfolgsfall gibt dieser Befehl die derzeit installierte Version der AWS CLI zurück. Informationen zum Upgrade einer älteren Version von AWS CLI auf die neueste Version finden Sie unter [Installation von](#). AWS Command Line Interface

2. Führen Sie am Terminal die folgenden Befehle aus, um den AWS CLI Credential Helper für HTTPS-Verbindungen zu konfigurieren:

```
git config --global credential.helper '!aws codecommit credential-helper $@'
git config --global credential.UseHttpPath true
```

 Tip

Der Credential Helper verwendet die standardmäßige EC2 Amazon-Instance-Rolle für Ihre Entwicklungsumgebung. Wenn Sie die Entwicklungsumgebung verwenden möchten, um eine Verbindung zu Repositorys herzustellen, die nicht in diesen Repositorys gehostet werden CodeCommit, konfigurieren Sie entweder SSH-Verbindungen zu diesen Repositorys oder konfigurieren Sie eine lokale `.gitconfig` Datei, um ein alternatives Anmeldeinformationsverwaltungssystem zu verwenden, wenn Sie eine Verbindung zu diesen anderen Repositorys herstellen. Weitere Informationen finden Sie unter [Git Tools - Credential Storage](#) auf der Git-Website.

Schritt 3: Klonen Sie ein CodeCommit Repository in Ihre Entwicklungsumgebung AWS Cloud9 EC2

Nachdem Sie den AWS CLI Credential Helper konfiguriert haben, können Sie Ihr CodeCommit Repository darauf klonen. Anschließend können Sie den Code bearbeiten.

1. Führen Sie am Terminal den Befehl `git clone` unter Angabe der HTTPS-Klon-URL des zu klonenden Repositorys aus. Wenn Sie beispielsweise ein Repository klonen möchten, das MyDemoRepo in der Region USA Ost (Ohio) benannt ist, geben Sie Folgendes ein:

```
git clone https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo
```

 Tip

Sie finden die Klon-URL für Ihr Repository in der CodeCommit Konsole, indem Sie URL klonen wählen.

2. Wenn das Klonen abgeschlossen ist, erweitern Sie den Ordner für das Repository in der seitlichen Navigationsleiste und wählen Sie die Datei aus, die Sie bearbeiten möchten. Alternativ können Sie File (Datei) und New File (Neue Datei) auswählen, um eine Datei zu erstellen.
3. Wenn Sie die Bearbeitung oder Erstellung der Dateien abgeschlossen haben, wechseln Sie im Terminal-Fenster zu dem Verzeichnis mit Ihrem geklonten Repository und führen Sie einen Commit durch. Übertragen Sie Ihre Änderungen dann per Push. Wenn Sie beispielsweise eine neue Datei mit dem Namen hinzugefügt haben *MyFile.py*:

```
cd MyDemoRepo
git commit -a MyFile.py
git commit -m "Added a new file with some code improvements"
git push
```

Nächste Schritte

Weitere Informationen finden Sie im [AWS Cloud9 Benutzerhandbuch](#) und im [CodeCommit Beispiel für AWS Cloud9](#). Weitere Informationen zur Verwendung von Git mit CodeCommit finden Sie unter [Erste Schritte mit Git und AWS CodeCommit](#).

Integrieren Sie Visual Studio mit AWS CodeCommit

Sie können Visual Studio verwenden, um Codeänderungen in einem CodeCommit Repository vorzunehmen. Das enthält AWS Toolkit for Visual Studio jetzt Funktionen, die das Arbeiten mit Visual Studio CodeCommit einfacher und komfortabler machen. Das Toolkit for Visual Studio Studio-Integration ist so konzipiert, dass es mit Git-Anmeldeinformationen und einem IAM-Benutzer funktioniert. Sie können vorhandene Repositories klonen, Repositories erstellen, Code-Änderungen bestätigen und speichern, und vieles mehr.

Important

Das Toolkit for Visual Studio ist nur für die Installation auf Windows-Betriebssystemen verfügbar. Informationen zur Arbeit mit Visual Studio Code finden Sie unter [AWS Toolkit for Visual Studio Code](#).

Wenn Sie das Toolkit for Visual Studio schon einmal verwendet haben, sind Sie wahrscheinlich bereits mit der Einrichtung von AWS Anmeldeinformationsprofilen vertraut, die einen Zugriffsschlüssel

und einen geheimen Schlüssel enthalten. Anmeldeinformationsprofile werden im Toolkit for Visual Studio verwendet, um AWS Serviceanfragen zu ermöglichen APIs (z. B. an Amazon S3, um Buckets aufzulisten oder Repositorys CodeCommit aufzulisten). Um Code abzurufen und in ein CodeCommit Repository zu übertragen, benötigen Sie außerdem Git-Anmeldeinformationen. Wenn Sie keine Git-Anmeldeinformationen haben, kann das Toolkit for Visual Studio diese Anmeldeinformationen für Sie generieren und anwenden. Dadurch können Sie viel Zeit sparen.

Um Visual Studio mit verwenden zu können CodeCommit, benötigen Sie Folgendes:

- Ein IAM-Benutzer, für den ein gültiger Satz von Anmeldeinformationen (ein Zugriffsschlüssel und ein geheimer Schlüssel) konfiguriert ist. Dieser IAM-Benutzer sollte außerdem über Folgendes verfügen:

Eine der CodeCommit verwalteten Richtlinien und die IAMSelfManageServiceSpecificCredentials verwaltete Richtlinie, die darauf angewendet wurde.

ODER

Wenn der IAM-Benutzer bereits Git-Anmeldeinformationen konfiguriert hat, eine der CodeCommit verwalteten Richtlinien oder gleichwertige Berechtigungen.

Weitere Informationen finden Sie unter [AWS verwaltete Richtlinien für CodeCommit](#) und [Verstehen und Abrufen Ihrer Sicherheitsanmeldeinformationen](#).

- Das AWS Toolkit for Visual Studio ist auf dem Computer installiert, auf dem Sie Visual Studio installiert haben. Weitere Informationen finden Sie unter [Einrichten von AWS Toolkit for Visual Studio](#).

Weitere Informationen zur Verwendung von AWS Toolkit for Visual Studio finden Sie unter [Verwenden AWS CodeCommit mit Visual Studio Team Explorer](#) im Toolkit for Visual Studio Studio-Benutzerhandbuch. CodeCommit

Einrichtung für SSH-Benutzer, die das nicht verwenden AWS CLI

Wenn Sie SSH-Verbindungen für Ihr Repository verwenden möchten, können Sie eine Verbindung herstellen, ohne das zu installieren. AWS CodeCommit AWS CLI AWS CLI Dazu gehören Befehle, die nützlich sind, wenn Sie CodeCommit Repositorys verwenden und verwalten, aber für die Ersteinrichtung nicht erforderlich sind.

In diesem Thema wird Folgendes vorausgesetzt:

- Sie haben einen IAM-Benutzer mit den Richtlinien oder Berechtigungen eingerichtet, die für das Hochladen von Schlüsseln erforderlich sind, CodeCommit und der IAMUserSSHKeysverwalteten Richtlinie oder den entsprechenden Berechtigungen, die für das Hochladen von Schlüsseln erforderlich sind. Weitere Informationen finden Sie unter [Verwendung identitätsbasierter Richtlinien \(IAM-Richtlinien\) für CodeCommit](#).
- Sie haben bereits ein Schlüsselpaar aus einem privaten und einem öffentlichen Schlüssel bzw. wissen, wie ein solches erstellt wird. Wir empfehlen dringend, eine sichere Pass-Phrase für den SSH-Schlüssel zu verwenden.
- Sie sind mit SSH, Ihrem Git-Client und den entsprechenden Konfigurationsdateien vertraut.
- Wenn Sie Windows verwenden, haben Sie ein Befehlszeilen-Dienstprogramm wie Git Bash installiert, das die Bash-Shell emuliert.

Wenn Sie weitere Anleitungen benötigen, folgen Sie den Anweisungen unter [Für SSH-Verbindungen unter Linux, macOS oder Unix](#) oder [Für SSH-Verbindungen unter Windows](#).

Themen

- [Schritt 1: Verknüpfen Sie Ihren öffentlichen Schlüssel mit Ihrem IAM-Benutzer](#)
- [Schritt 2: Zu Ihrer CodeCommit SSH-Konfiguration hinzufügen](#)
- [Nächste Schritte](#)

Schritt 1: Verknüpfen Sie Ihren öffentlichen Schlüssel mit Ihrem IAM-Benutzer

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter <https://console.aws.amazon.com/iam/>
2. Wählen Sie in der IAM-Konsole im Navigationsbereich Benutzer aus und wählen Sie aus der Benutzerliste Ihren IAM-Benutzer aus.
3. Wählen Sie auf der Registerkarte Security Credentials die Option Upload SSH public key aus.
4. Kopieren Sie die Inhalte des öffentlichen SSH-Schlüssels in das Feld und wählen Sie dann Upload SSH Key aus.

Tip

Das Schlüsselpaar aus einem privaten und einem öffentlichen Schlüssel muss vom Schlüsseltyp SSH-2 RSA sein, im OpenSSH-Format vorliegen und 2048 Bits enthalten. Der Schlüssel sieht wie folgt aus:

```
ssh-rsa EXAMPLE-
AfICcQD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMCVVMxCzAJB
gNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBA
5zb2x1MRIwEAYDVQQDEw1UZXRhbnQ2L2sYWMxHZAAdBgkqhkiG9w0BCQEWEG5vb251QG
FtYXpvaW5jb20wHhcNMTEwNDI1MjA0NTIxWhcNMTEwNDI1MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJB
gNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDAS=EXAMPLE user-
name@ip-192-0-2-137
```

IAM akzeptiert nur öffentliche Schlüssel im OpenSSH-Format. Wenn Sie den öffentlichen Schlüssel in einem anderen Format bereitstellen, wird eine Fehlermeldung angezeigt, dass das Schlüsselformat ungültig ist.

- Kopieren Sie die SSH-Schlüssel-ID (z. B. **APKAEIBAERJR2EXAMPLE**) und schließen Sie die Konsole.

SSH keys for AWS CodeCommit			
Use SSH public keys to authenticate to AWS CodeCommit repositories. Learn more about SSH keys.			
Upload SSH public key			
SSH Key ID	Uploaded	Status	Actions
APKAEIBAERJR2EXAMPLE	2015-07-21 16:32 PDT	Active	Make Inactive Show SSH Key Delete

Schritt 2: Zu Ihrer CodeCommit SSH-Konfiguration hinzufügen

- Bearbeiten Sie am Terminal (Linux, macOS oder Unix) oder am Bash-Emulator (Windows) Ihre SSH-Konfigurationsdatei, indem Sie Folgendes eingeben: `cat >> ~/.ssh/config`

```
Host git-codecommit.*.amazonaws.com
  User Your-SSH-Key-ID, such as APKAEIBAERJR2EXAMPLE
  IdentityFile Your-Private-Key-File, such as ~/.ssh/codecommit_rsa or ~/.ssh/id_rsa
```


 Tip

Wenn Sie mehr als eine SSH-Konfiguration haben, stellen Sie sicher, dass Sie die Leerzeilen vor und nach dem Inhalt einfügen. Speichern Sie die Datei, indem Sie gleichzeitig die Tasten `Ctrl` und `d` drücken.

2. Führen Sie den folgenden Befehl aus, um Ihre SSH-Konfiguration zu testen:

```
ssh git-codecommit.us-east-2.amazonaws.com
```

Geben Sie die Pass-Phrase für die SSH-Schlüsseldatei ein, wenn Sie dazu aufgefordert werden. Wenn alles ordnungsgemäß konfiguriert ist, sollten Sie folgende Erfolgsmeldung sehen:

```
You have successfully authenticated over SSH. You can use Git to interact with  
CodeCommit.
```

Nächste Schritte

Sie haben die Voraussetzungen erfüllt. Folgen Sie den Schritten unter, um mit der Verwendung [Erste Schritte mit CodeCommit](#) zu beginnen. CodeCommit

Um die Verbindung zu einem bestehenden Repository herzustellen, folgen Sie den Schritten unter [Herstellen einer Verbindung mit einem Repository](#). Um ein Repository zu erstellen, folgen Sie den Schritten unter [Erstellen eines -Repositorys](#).

Einrichtungsschritte für SSH-Verbindungen zu AWS CodeCommit Repositorys unter Linux, macOS oder Unix

Bevor Sie CodeCommit zum ersten Mal eine Verbindung herstellen können, müssen Sie einige Schritte zur Erstkonfiguration ausführen. Nachdem Sie Ihren Computer und Ihr AWS Profil eingerichtet haben, können Sie eine Verbindung zu einem CodeCommit Repository herstellen und dieses Repository auf Ihren Computer klonen (auch bekannt als Erstellen eines lokalen Repos). Wenn Sie mit Git noch nicht vertraut sind, finden Sie unter [Wo kann ich mehr über Git erfahren?](#) genauere Informationen.

Themen

- [Schritt 1: Erstkonfiguration für CodeCommit](#)
- [Schritt 2: Installieren von Git](#)
- [Schritt 3: Anmeldeinformationen unter Linux, macOS oder Unix konfigurieren](#)
- [Schritt 4: Connect zur CodeCommit Konsole her und klonen Sie das Repository](#)
- [Nächste Schritte](#)

Schritt 1: Erstkonfiguration für CodeCommit

Gehen Sie wie folgt vor, um ein Amazon Web Services Services-Konto einzurichten, einen IAM-Benutzer zu erstellen und den Zugriff auf zu CodeCommit konfigurieren.

Um einen IAM-Benutzer für den Zugriff zu erstellen und zu konfigurieren CodeCommit

1. Erstellen Sie ein Amazon Web Services Services-Konto, indem Sie zu <http://aws.amazon.com> gehen und Sign Up wählen.
2. Erstellen Sie einen IAM-Benutzer oder verwenden Sie einen vorhandenen in Ihrem Amazon Web Services Services-Konto. Stellen Sie sicher, dass Sie über eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel verfügen, die diesem IAM-Benutzer zugeordnet sind. Weitere Informationen finden Sie unter [Einen IAM-Benutzer in Ihrem Amazon Web Services Services-Konto erstellen](#).

Note

CodeCommit erfordert AWS Key Management Service. Wenn Sie einen vorhandenen IAM-Benutzer verwenden, stellen Sie sicher, dass dem Benutzer keine Richtlinien zugeordnet sind, die ausdrücklich die von CodeCommit erforderlichen AWS KMS Aktionen verweigern. Weitere Informationen finden Sie unter [AWS KMS und Verschlüsselung](#).

3. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter <https://console.aws.amazon.com/iam/>
4. Wählen Sie in der IAM-Konsole im Navigationsbereich Benutzer und dann den IAM-Benutzer aus, den Sie für den Zugriff konfigurieren möchten. CodeCommit
5. Wählen Sie auf der Registerkarte Permissions die Option Add Permissions.
6. Wählen Sie unter Grant permissions die Option Attach existing policies directly aus.

7. Wählen Sie `AWSCodeCommitPowerUser` aus der Liste der Richtlinien eine andere verwaltete Richtlinie für CodeCommit den Zugriff aus. Weitere Informationen finden Sie unter [AWS verwaltete Richtlinien für CodeCommit](#).

Nachdem Sie die Richtlinie ausgewählt haben, die Sie anhängen möchten, wählen Sie Weiter: Überprüfen aus, um die Liste der Richtlinien zu überprüfen, die an den IAM-Benutzer angehängt werden sollen. Ist die Liste korrekt, wählen Sie Add permissions aus.

Weitere Informationen zu CodeCommit verwalteten Richtlinien und zur gemeinsamen Nutzung des Zugriffs auf Repositorys mit anderen Gruppen und Benutzern finden Sie unter [Teilen Sie ein Repository](#) und [Authentifizierung und Zugriffskontrolle für AWS CodeCommit](#)

Note

Wenn Sie AWS CLI Befehle mit verwenden möchten CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter [Befehlszeilenreferenz](#).

Schritt 2: Installieren von Git

Um mit Dateien, Commits und anderen Informationen in CodeCommit Repositorys zu arbeiten, müssen Sie Git auf Ihrem lokalen Computer installieren. CodeCommit unterstützt Git-Versionen 1.7.9 und höher. Git Version 2.28 unterstützt die Konfiguration des Branchnamens für anfängliche Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git.

Um Git zu installieren, empfehlen wir Websites wie [Git Downloads](#).

Note

Git ist eine sich entwickelnde, regelmäßig aktualisierte Plattform. Gelegentlich kann sich eine Änderung der Funktionen auf die Art und Weise auswirken, mit der es funktioniert CodeCommit. Wenn du Probleme mit einer bestimmten Version von Git und hast CodeCommit, sieh dir die Informationen unter an [Fehlerbehebung](#).

Schritt 3: Anmeldeinformationen unter Linux, macOS oder Unix konfigurieren

SSH und Linux, macOS oder Unix: Richten Sie die öffentlichen und privaten Schlüssel für Git ein und CodeCommit

Um die öffentlichen und privaten Schlüssel für Git einzurichten und CodeCommit

1. Führen Sie im Terminalfenster Ihres lokalen Computers den Befehl `ssh-keygen` aus und folgen Sie den Anweisungen, um die Datei im Verzeichnis ".ssh" für Ihr Profil zu speichern.

Note

Fragen Sie den Systemadministrator nach dem gewünschten Speicherort der Schlüsseldateien und nach dem Muster, nach dem die Dateien benannt werden sollten.

Zum Beispiel:

```
$ ssh-keygen
```

```
Generating public/private rsa key pair.
```

```
Enter file in which to save the key (/home/user-name/.ssh/id_rsa): Type /home/  
your-user-name/.ssh/ and a file name here, for example /home/your-user-name/.ssh/  
codecommit_rsa
```

```
Enter passphrase (empty for no passphrase): <Type a passphrase, and then press  
Enter>
```

```
Enter same passphrase again: <Type the passphrase again, and then press Enter>
```

```
Your identification has been saved in /home/user-name/.ssh/codecommit_rsa.
```

```
Your public key has been saved in /home/user-name/.ssh/codecommit_rsa.pub.
```

```
The key fingerprint is:
```

```
45:63:d5:99:0e:99:73:50:5e:d4:b3:2d:86:4a:2c:14 user-name@client-name
```

```
The key's randomart image is:
```

```
+--[ RSA 2048 ]-----+
|      E.+o*.++|
|      .o .=.o.|
|      . .. *.+|
|      ..o . +..|
|      So . . . |
```

```
|      .      |  
|              |  
|              |  
+-----+
```

Folgendes wird generiert:

- Die `codecommit_rsa` Datei, bei der es sich um die private Schlüsseldatei handelt.
- Die `codecommit_rsa.pub` Datei, bei der es sich um die Datei mit dem öffentlichen Schlüssel handelt.

 Tip

ssh-keygen generiert standardmäßig einen 2048-Bit-Schlüssel. Sie können die Parameter `-t` und `-b` verwenden, um den Typ und die Länge des Schlüssels anzugeben. Wenn Sie einen 4096-Bit-Schlüssel im RSA-Format benötigen, würden Sie dies angeben, indem Sie den Befehl mit den folgenden Parametern ausführen:

```
ssh-keygen -t rsa -b 4096
```

Weitere Informationen zu den für SSH-Schlüssel erforderlichen Formaten und Längen finden Sie unter [Using IAM with. CodeCommit](#)

2. Führen Sie den folgenden Befehl aus, um den Wert der Datei mit dem öffentlichen Schlüssel (`codecommit_rsa.pub`) anzuzeigen:

```
cat ~/.ssh/codecommit_rsa.pub
```

Kopieren Sie diesen Wert. Er sieht in etwa wie folgt aus:

```
ssh-rsa EXAMPLE-AfICCQD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC01BTSBDb25zb2x1MRIwEAYDVQQDEw1UZXRhbnQ21sYWMxH2AdBgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvaW5jb20wHhcNMTEwNDI1MjA0NTIxWhcNMTEwNDI1MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDAS=EXAMPLE user-name@ip-192-0-2-137
```

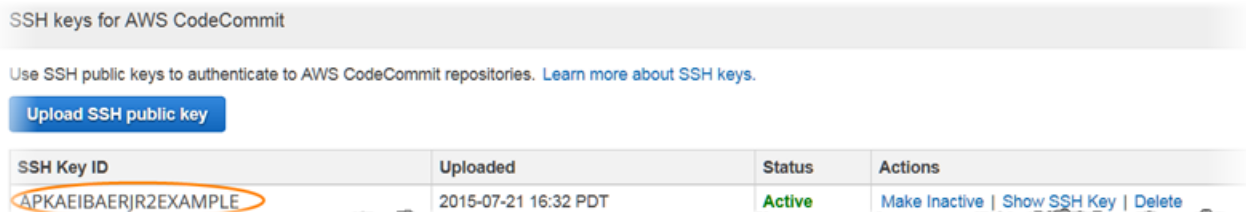
3. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter <https://console.aws.amazon.com/iam/>.

Note

Sie können Ihre CodeCommit Anmeldeinformationen direkt unter Meine Sicherheitsanmeldedaten einsehen und verwalten. Weitere Informationen finden Sie unter [Ihre Anmeldedaten einsehen und verwalten](#).

4. Wählen Sie in der IAM-Konsole im Navigationsbereich Benutzer aus und wählen Sie aus der Benutzerliste Ihren IAM-Benutzer aus.
5. Wählen Sie auf der Detailseite des Benutzers auf der Registerkarte Security Credentials die Option Upload SSH public key aus.
6. Kopieren Sie den Inhalt des öffentlichen SSH-Schlüssels in das Feld und wählen Sie dann Upload SSH public key aus.
7. Kopieren oder speichern Sie die Informationen in der SSH-Schlüssel-ID (z. B.).

APKAEIBAERJR2EXAMPLE



SSH keys for AWS CodeCommit			
Use SSH public keys to authenticate to AWS CodeCommit repositories. Learn more about SSH keys.			
Upload SSH public key			
SSH Key ID	Uploaded	Status	Actions
APKAEIBAERJR2EXAMPLE	2015-07-21 16:32 PDT	Active	Make inactive Show SSH Key Delete

Note

Wenn Sie mehr als einen SSH-Schlüssel IDs hochgeladen haben, werden die Schlüssel alphabetisch nach der Schlüssel-ID und nicht nach dem Upload-Datum aufgelistet. Stellen Sie sicher, dass Sie die Schlüssel-ID kopiert haben, die dem richtigen Upload-Datum zugeordnet ist.

8. Verwenden Sie auf Ihrem lokalen Computer einen Texteditor, um eine Konfigurationsdatei im Verzeichnis ~/.ssh zu erstellen, und fügen Sie dann der Datei die folgenden Zeilen hinzu, wobei der Wert für die SSH-Schlüssel-ID **User** ist, die Sie zuvor kopiert haben:

```
Host git-codecommit.*.amazonaws.com
  User APKAEIBAERJR2EXAMPLE
  IdentityFile ~/.ssh/codecommit_rsa
```

 Note

Wenn Sie Ihrer privaten Schlüsseldatei einen anderen Namen als gegeben haben, achten Sie darauf `codecommit_rsa`, ihn hier zu verwenden.

Sie können SSH-Zugriff auf Repositorys in mehreren Amazon Web Services Services-Konten einrichten. Weitere Informationen finden Sie unter. [Fehlerbehebung bei SSH-Verbindungen zu AWS CodeCommit](#)

Speichern Sie die Datei unter dem Namen `config`.

9. Führen Sie im Terminalfenster den folgenden Befehl aus, um die Berechtigungen für die Konfigurationsdatei zu ändern:

```
chmod 600 config
```

10. Führen Sie den folgenden Befehl aus, um Ihre SSH-Konfiguration zu testen:

```
ssh git-codecommit.us-east-2.amazonaws.com
```

Sie werden aufgefordert, die Verbindung zu bestätigen, da sie noch nicht in Ihrer Datei mit bekannten Hosts enthalten `git-codecommit.us-east-2.amazonaws.com` ist. Der CodeCommit Server-Fingerabdruck wird als Teil der Überprüfung (a9:6d:03:ed:08:42:21:be:06:e1:e0:2a:d1:75:31:5e für MD5 oder 31B1W2g5xn/NA2Ck6dyeJIrQ0Wvn7n8UEs56fG6ZIzQ für SHA256) angezeigt.

 Note

CodeCommit Server-Fingerabdrücke sind für jeden AWS-Region einzigartig. Informationen zum Anzeigen der Server-Fingerabdrücke für einen finden Sie AWS-Region unter. [Server-Fingerabdrücke für CodeCommit](#)

Nachdem Sie die Verbindung bestätigt haben, sollte jeweils eine Bestätigung angezeigt werden, dass der Server der Datei mit den bekannten Hosts hinzugefügt und die Verbindung hergestellt wurde. Wenn Sie keine Erfolgsmeldung sehen, überprüfen Sie, ob Sie die `config` Datei im

Verzeichnis ~/.ssh des IAM-Benutzers gespeichert haben, für den Sie den Zugriff konfiguriert haben CodeCommit, und ob Sie die richtige private Schlüsseldatei angegeben haben.

Wenn Sie Informationen zur Problembehandlung benötigen, führen Sie den Befehl mit dem ssh Parameter aus. -v Zum Beispiel:

```
ssh -v git-codecommit.us-east-2.amazonaws.com
```

Weitere Informationen zum Beheben von Verbindungsproblemen finden Sie unter [Fehlerbehebung bei SSH-Verbindungen zu AWS CodeCommit](#).

Schritt 4: Connect zur CodeCommit Konsole her und klonen Sie das Repository

Wenn Sie den Namen und die Verbindungsdetails für das CodeCommit-Repository bereits von einem Administrator erhalten haben, können Sie diesen Schritt überspringen und das Repository direkt klonen.

Um eine Verbindung zu einem CodeCommit Repository herzustellen

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie in der Regionsauswahl den Ort aus, AWS-Region an dem das Repository erstellt wurde. Repositories sind spezifisch für eine AWS-Region. Weitere Informationen finden Sie unter [Regionen und Git-Verbindungsendpunkte](#).
3. Suchen Sie das Repository, zu dem Sie eine Verbindung herstellen möchten, in der Liste und wählen Sie es aus. Wählen Sie Clone URL (URL klonen) und dann das Protokoll aus, das Sie beim Klonen oder bei der Verbindung zu dem Repository verwenden möchten. Dadurch wird die Klon-URL kopiert.
 - Kopieren Sie die HTTPS-URL, wenn Sie entweder Git-Anmeldeinformationen mit Ihrem IAM-Benutzer oder den Credential Helper verwenden, der im Lieferumfang von enthalten ist. AWS CLI
 - Kopieren Sie die HTTPS-URL (GRC), wenn Sie den Befehl git-remote-codecommit auf Ihrem lokalen Computer verwenden.

- Kopieren Sie die SSH-URL, wenn Sie ein öffentliches/privates SSH-Schlüsselpaar mit Ihrem IAM-Benutzer verwenden.

 Note

Wenn Sie statt einer Liste von Repositorys eine Willkommensseite sehen, sind Ihrem AWS Konto in dem Land, in dem Sie angemeldet sind, keine Repositorys zugeordnet. AWS-Region Informationen zur Erstellung eines Repositorys finden Sie unter [the section called “Erstellen eines -Repositorys”](#), oder befolgen Sie die Schritte im Tutorial [Erste Schritte mit Git und CodeCommit](#).

4. Öffnen Sie ein -Terminalfenster. Führen Sie im Verzeichnis "/tmp" den git clone-Befehl mit der SSH-URL aus, die Sie kopiert haben, um das Repository zu klonen. Um beispielsweise ein Repository mit dem Namen *MyDemoRepo* eines lokalen Repositorys mit dem Namen *my-demo-repo* der Region USA Ost (Ohio) zu klonen:

```
git clone ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
```

 Note

Wenn der Verbindungstest erfolgreich war, der Klonbefehl jedoch nicht ausgeführt wurde, besitzen Sie vielleicht nicht die erforderlichen Zugriffsrechte für die Konfigurationsdatei oder eine andere Einstellung verursacht einen Konflikt mit der Konfigurationsdatei. Versuchen Sie erneut, die Verbindung herzustellen. Geben Sie dabei die SSH-Schlüssel-ID als Teil des Befehls an. Zum Beispiel:

```
git clone ssh://Your-SSH-Key-ID@git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
```

Weitere Informationen finden Sie unter [Zugriffsfehler: Der öffentliche Schlüssel wurde erfolgreich in IAM hochgeladen, aber die Verbindung schlägt auf Linux-, macOS- oder Unix-Systemen fehl](#).

Weitere Informationen über die Verbindungserstellung zu Repositorys finden Sie unter [Connect zum CodeCommit Repository her, indem Sie das Repository klonen](#).

Nächste Schritte

Sie haben die Voraussetzungen erfüllt. Folgen Sie den Anweisungen unter, um mit der Verwendung [Erste Schritte mit CodeCommit](#) CodeCommit zu beginnen.

Einrichtungsschritte für SSH-Verbindungen zu AWS CodeCommit Repositorys unter Windows

Bevor Sie AWS CodeCommit zum ersten Mal eine Verbindung herstellen können, müssen Sie einige Schritte zur Erstkonfiguration ausführen. Nachdem Sie Ihren Computer und Ihr AWS Profil eingerichtet haben, können Sie eine Verbindung zu einem CodeCommit Repository herstellen und dieses Repository auf Ihren Computer klonen (auch bekannt als Erstellen eines lokalen Repos). Wenn Sie mit Git noch nicht vertraut sind, finden Sie unter [Wo kann ich mehr über Git erfahren?](#) genauere Informationen.

Themen

- [Schritt 1: Erstkonfiguration für CodeCommit](#)
- [Schritt 2: Installieren von Git](#)
- [Schritt 3: Richten Sie die öffentlichen und privaten Schlüssel für Git ein und CodeCommit](#)
- [Schritt 4: Connect zur CodeCommit Konsole her und klonen Sie das Repository](#)
- [Nächste Schritte](#)

Schritt 1: Erstkonfiguration für CodeCommit

Gehen Sie wie folgt vor, um ein Amazon Web Services Services-Konto einzurichten, einen IAM-Benutzer zu erstellen und den Zugriff auf zu CodeCommit konfigurieren.

Um einen IAM-Benutzer für den Zugriff zu erstellen und zu konfigurieren CodeCommit

1. Erstellen Sie ein Amazon Web Services Services-Konto, indem Sie zu <http://aws.amazon.com> gehen und Sign Up wählen.
2. Erstellen Sie einen IAM-Benutzer oder verwenden Sie einen vorhandenen in Ihrem Amazon Web Services Services-Konto. Stellen Sie sicher, dass Sie über eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel verfügen, die diesem IAM-Benutzer zugeordnet sind. Weitere Informationen finden Sie unter [Einen IAM-Benutzer in Ihrem Amazon Web Services Services-Konto erstellen](#).

 Note

CodeCommit erfordert AWS Key Management Service. Wenn Sie einen vorhandenen IAM-Benutzer verwenden, stellen Sie sicher, dass dem Benutzer keine Richtlinien zugeordnet sind, die ausdrücklich die von CodeCommit erforderlichen AWS KMS Aktionen verweigern. Weitere Informationen finden Sie unter [AWS KMS und Verschlüsselung](#).

3. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter <https://console.aws.amazon.com/iam/>
4. Wählen Sie in der IAM-Konsole im Navigationsbereich Benutzer und dann den IAM-Benutzer aus, den Sie für den Zugriff konfigurieren möchten. CodeCommit
5. Wählen Sie auf der Registerkarte Permissions die Option Add Permissions.
6. Wählen Sie unter Grant permissions die Option Attach existing policies directly aus.
7. Wählen Sie AWSCodeCommitPowerUser aus der Liste der Richtlinien eine andere verwaltete Richtlinie für CodeCommit den Zugriff aus. Weitere Informationen finden Sie unter [AWS verwaltete Richtlinien für CodeCommit](#).

Nachdem Sie die Richtlinie ausgewählt haben, die Sie anhängen möchten, wählen Sie Weiter: Überprüfen aus, um die Liste der Richtlinien zu überprüfen, die an den IAM-Benutzer angehängt werden sollen. Ist die Liste korrekt, wählen Sie Add permissions aus.

Weitere Informationen zu CodeCommit verwalteten Richtlinien und zur gemeinsamen Nutzung des Zugriffs auf Repositorys mit anderen Gruppen und Benutzern finden Sie unter [Teilen Sie ein Repository](#) und [Authentifizierung und Zugriffskontrolle für AWS CodeCommit](#)

 Note

Wenn Sie AWS CLI Befehle mit verwenden möchten CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter [Befehlszeilenreferenz](#).

Schritt 2: Installieren von Git

Um mit Dateien, Commits und anderen Informationen in CodeCommit Repositorys zu arbeiten, müssen Sie Git auf Ihrem lokalen Computer installieren. CodeCommit unterstützt Git-Versionen

1.7.9 und höher. Git Version 2.28 unterstützt die Konfiguration des Branchnamens für anfängliche Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git.

Um Git zu installieren, empfehlen wir Websites wie [Git Downloads](#).

Note

Git ist eine sich entwickelnde, regelmäßig aktualisierte Plattform. Gelegentlich kann sich eine Änderung der Funktionen auf die Art und Weise auswirken, mit der es funktioniert CodeCommit. Wenn du Probleme mit einer bestimmten Version von Git und hast CodeCommit, sieh dir die Informationen unter an [Fehlerbehebung](#).

Wenn die installierte Git-Version keinen Bash-Emulator, z. B. Git Bash, umfasst, installieren Sie einen. Sie verwenden diesen Emulator beim Erstellen von SSH-Verbindungen anstelle der Windows-Befehlszeile.

Schritt 3: Richten Sie die öffentlichen und privaten Schlüssel für Git ein und CodeCommit

Um die öffentlichen und privaten Schlüssel für Git und CodeCommit Windows einzurichten

1. Öffnen Sie den Bash-Emulator.

Note

Eventuell müssen Sie den Emulator mit Administratorberechtigungen ausführen.

Führen Sie im Emulator den Befehl `ssh-keygen` aus und befolgen Sie die Anweisungen, um die Datei im Verzeichnis ".ssh" für Ihr Profil zu speichern.

Zum Beispiel:

```
$ ssh-keygen
```

```
Generating public/private rsa key pair.
```

```
Enter file in which to save the key (/drive/Users/user-name/.ssh/id_rsa): Type a  
file name here, for example /c/Users/user-name/.ssh/codecommit_rsa
```

Enter passphrase (empty for no passphrase): *<Type a passphrase, and then press Enter>*

Enter same passphrase again: *<Type the passphrase again, and then press Enter>*

Your identification has been saved in *drive/Users/user-name/.ssh/codecommit_rsa*.

Your public key has been saved in *drive/Users/user-name/.ssh/codecommit_rsa.pub*.

The key fingerprint is:

45:63:d5:99:0e:99:73:50:5e:d4:b3:2d:86:4a:2c:14 *user-name@client-name*

The key's randomart image is:

```
+--[ RSA 2048]-----+
|           E.+..o*.++|
|           .o .=.o.|
|           . .. *. +|
|           ..o . +..|
|           So . . . |
|           .         |
|                   |
|                   |
|                   |
|                   |
+-----+

```

Folgendes wird generiert:

- Die *codecommit_rsa* Datei, bei der es sich um die private Schlüsseldatei handelt.
- *codecommit_rsa*Die.pub-Datei, bei der es sich um die Datei mit dem öffentlichen Schlüssel handelt.

Tip

ssh-keygenGeneriert standardmäßig einen 2048-Bit-Schlüssel. Sie können die Parameter -t und -b verwenden, um den Typ und die Länge des Schlüssels anzugeben. Wenn Sie einen 4096-Bit-Schlüssel im RSA-Format benötigen, würden Sie dies angeben, indem Sie den Befehl mit den folgenden Parametern ausführen:

```
ssh-keygen -t rsa -b 4096
```

Weitere Informationen zu den für SSH-Schlüssel erforderlichen Formaten und Längen finden Sie unter [Using IAM with CodeCommit](#)

- Führen Sie die folgenden Befehle aus, um den Wert der Datei mit dem öffentlichen Schlüssel (`codecommit_rsa.pub`) anzuzeigen:

```
cd .ssh
notepad codecommit_rsa.pub
```

Kopieren Sie den Inhalt der Datei und schließen Sie Notepad, ohne zu speichern. Der Inhalt der Datei ähnelt dem folgenden Beispiel:

```
ssh-rsa EXAMPLE-AfICCQD6m7oRw0uX0jANBgqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAwTC01BTSBDb25zb2x1MRIwEAYDVQQDEw1UZXR0Q21sYWMxHzAdBgqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTcwNDI1MjA0NTIxWhcNMTcwNDI1MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDAS=EXAMPLE user-name@computer-name
```

- Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter <https://console.aws.amazon.com/iam/>

Note

Sie können Ihre CodeCommit Anmeldeinformationen direkt unter Meine Sicherheitsanmeldedaten einsehen und verwalten. Weitere Informationen finden Sie unter [Ihre Anmeldedaten einsehen und verwalten](#).

- Wählen Sie in der IAM-Konsole im Navigationsbereich Benutzer aus und wählen Sie aus der Benutzerliste Ihren IAM-Benutzer aus.
- Wählen Sie auf der Detailseite des Benutzers auf der Registerkarte Security Credentials die Option Upload SSH public key aus.
- Kopieren Sie den Inhalt des öffentlichen SSH-Schlüssels in das Feld und wählen Sie dann Upload SSH public key aus.
- Kopieren oder speichern Sie die Informationen in der SSH-Schlüssel-ID (z. B.).
APKAEIBAERJR2EXAMPLE

SSH keys for AWS CodeCommit			
Use SSH public keys to authenticate to AWS CodeCommit repositories. Learn more about SSH keys.			
Upload SSH public key			
SSH Key ID	Uploaded	Status	Actions
APKAEIBAERJR2EXAMPLE	2015-07-21 16:32 PDT	Active	Make Inactive Show SSH Key Delete

 Note

Wenn Sie mehr als einen SSH-Schlüssel IDs hochgeladen haben, werden die Schlüssel alphabetisch nach der Schlüssel-ID und nicht nach dem Upload-Datum aufgelistet. Stellen Sie sicher, dass Sie die Schlüssel-ID kopiert haben, die dem richtigen Upload-Datum zugeordnet ist.

8. Führen Sie im Bash-Emulator die folgenden Befehle aus, um im Verzeichnis "~/.ssh" eine Konfigurationsdatei zu erstellen bzw. zu bearbeiten, falls bereits eine solche Datei vorhanden ist:

```
notepad ~/.ssh/config
```

9. Fügen Sie der Datei die folgenden Zeilen hinzu, wobei der Wert für die SSH-Schlüssel-ID *User* ist, die Sie zuvor kopiert haben, und der Wert für der Pfad und Name der privaten Schlüsseldatei *IdentityFile* ist:

```
Host git-codecommit.*.amazonaws.com
  User APKAEIBAERJR2EXAMPLE
  IdentityFile ~/.ssh/codecommit_rsa
```

 Note

Wenn Sie Ihrer privaten Schlüsseldatei einen anderen Namen gegeben haben als *codecommit_rsa*, achten Sie darauf, ihn hier zu verwenden. Sie können SSH-Zugriff auf Repositories in mehreren Amazon Web Services Services-Konten einrichten. Weitere Informationen finden Sie unter [Fehlerbehebung bei SSH-Verbindungen zu AWS CodeCommit](#)

Speichern Sie die Datei als "config" (nicht "config.txt") und schließen Sie Notepad.

 Important

Der Name der Datei muss config ohne Dateierweiterung lauten. Andernfalls schlagen die SSH-Verbindungen fehl.

10. Führen Sie den folgenden Befehl aus, um Ihre SSH-Konfiguration zu testen:

```
ssh git-codecommit.us-east-2.amazonaws.com
```

Sie werden aufgefordert, die Verbindung zu bestätigen, da sie noch nicht in Ihrer Datei mit bekannten Hosts enthalten `git-codecommit.us-east-2.amazonaws.com` ist. Der CodeCommit Server-Fingerabdruck wird als Teil der Überprüfung (a9:6d:03:ed:08:42:21:be:06:e1:e0:2a:d1:75:31:5e für MD5 oder 31B1W2g5xn/NA2Ck6dyeJIrQ0Wvn7n8UES56fG6ZIzQ für SHA256) angezeigt.

 Note

CodeCommit Server-Fingerabdrücke sind für jeden AWS-Region einzigartig. Informationen zum Anzeigen der Server-Fingerabdrücke für einen finden Sie AWS-Region unter. [Server-Fingerabdrücke für CodeCommit](#)

Nachdem Sie die Verbindung bestätigt haben, sollte jeweils eine Bestätigung angezeigt werden, dass der Server der Datei mit den bekannten Hosts hinzugefügt und die Verbindung hergestellt wurde. Wenn Sie keine Erfolgsmeldung sehen, überprüfen Sie, ob Sie die `config` Datei im Verzeichnis `~/.ssh` des IAM-Benutzers gespeichert haben, für den Sie den Zugriff konfiguriert haben CodeCommit, ob die `config` Datei keine Dateierweiterung hat (sie darf beispielsweise nicht `config.txt` heißen) und ob Sie die richtige private Schlüsseldatei (nicht `.pub`) angegeben haben. `codecommit_rsa codecommit_rsa`

Führen Sie den Befehl mit dem Parameter aus, um Probleme zu beheben. `ssh -v` Zum Beispiel:

```
ssh -v git-codecommit.us-east-2.amazonaws.com
```

Weitere Informationen zum Beheben von Verbindungsproblemen finden Sie unter [Fehlerbehebung bei SSH-Verbindungen zu AWS CodeCommit](#).

Schritt 4: Connect zur CodeCommit Konsole her und klonen Sie das Repository

Wenn Sie den Namen und die Verbindungsdetails für das CodeCommit-Repository bereits von einem Administrator erhalten haben, können Sie diesen Schritt überspringen und das Repository direkt klonen.

Um eine Verbindung zu einem CodeCommit Repository herzustellen

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie in der Regionsauswahl den Ort aus, AWS-Region an dem das Repository erstellt wurde. Repositories sind spezifisch für eine AWS-Region. Weitere Informationen finden Sie unter [Regionen und Git-Verbindungsendpunkte](#).
3. Suchen Sie das Repository, zu dem Sie eine Verbindung herstellen möchten, in der Liste und wählen Sie es aus. Wählen Sie Clone URL (URL klonen) und dann das Protokoll aus, das Sie beim Klonen oder bei der Verbindung zu dem Repository verwenden möchten. Dadurch wird die Klon-URL kopiert.
 - Kopieren Sie die HTTPS-URL, wenn Sie entweder Git-Anmeldeinformationen mit Ihrem IAM-Benutzer oder den Credential Helper verwenden, der im Lieferumfang von enthalten ist. AWS CLI
 - Kopieren Sie die HTTPS-URL (GRC), wenn Sie den Befehl `git-remote-codecommit` auf Ihrem lokalen Computer verwenden.
 - Kopieren Sie die SSH-URL, wenn Sie ein öffentliches/privates SSH-Schlüsselpaar mit Ihrem IAM-Benutzer verwenden.

Note

Wenn Sie statt einer Liste von Repositories eine Willkommensseite sehen, sind Ihrem AWS Konto in dem Land, in dem Sie angemeldet sind, keine Repositories zugeordnet. AWS-Region Informationen zur Erstellung eines Repositories finden Sie unter [the section called "Erstellen eines -Repositories"](#), oder befolgen Sie die Schritte im Tutorial [Erste Schritte mit Git und CodeCommit](#).

4. Führen Sie im Bash-Emulator den `git clone`-Befehl mit der SSH-URL aus, die Sie kopiert haben, um das Repository zu klonen. Durch diesen Befehl wird in einem Unterverzeichnis des Verzeichnisses, in dem Sie den Befehl ausgeführt haben, das lokale Repository erstellt. Um beispielsweise ein Repository mit dem Namen *MyDemoRepo* eines lokalen Repositories mit dem Namen *my-demo-repo* der Region USA Ost (Ohio) zu klonen:

```
git clone ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
```

Öffnen Sie alternativ eine Befehlszeile und führen Sie den Befehl mit der URL und der SSH-Schlüssel-ID für den öffentlichen Schlüssel aus, den Sie in IAM hochgeladen haben. `git clone` Dadurch wird das lokale Repository in einem Unterverzeichnis des Verzeichnisses erstellt, in dem Sie den Befehl ausgeführt haben. Im folgenden Beispiel wird ein Repository mit dem Namen *MyDemoRepo* in ein lokales Repository mit dem Namen *my-demo-repo* geklont:

```
git clone ssh://Your-SSH-Key-ID@git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
```

Weitere Informationen erhalten Sie unter [Connect zum CodeCommit Repository her, indem Sie das Repository klonen](#) und [Erstellen Sie einen Commit](#).

Nächste Schritte

Sie haben die Voraussetzungen erfüllt. Folgen Sie den Anweisungen unter, um mit der [Erste Schritte mit CodeCommit](#) Verwendung zu beginnen. CodeCommit

Einrichtungsschritte für HTTPS-Verbindungen zu AWS CodeCommit Repositorys unter Linux, macOS oder Unix mit dem AWS CLI Credential Helper

Bevor Sie AWS CodeCommit zum ersten Mal eine Verbindung herstellen können, müssen Sie die ersten Konfigurationsschritte abschließen. Für die meisten Benutzer lässt sich das am einfachsten mit den Schritten unter [Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden](#) umsetzen. Wenn Sie jedoch CodeCommit über ein Root-Konto, Verbundzugriff oder temporäre Anmeldeinformationen eine Verbindung herstellen möchten, können Sie den Credential Helper verwenden, der in der enthalten ist. AWS CLI

Note

Der Credential Helper ist zwar eine unterstützte Methode, um CodeCommit mithilfe von Verbundzugriff, einem Identitätsanbieter oder temporären Anmeldeinformationen eine Verbindung herzustellen, es wird jedoch empfohlen, das Hilfsprogramm zu installieren und zu verwenden. `git-remote-codecommit` Weitere Informationen finden Sie unter [Einrichtungsschritte für HTTPS-Verbindungen AWS CodeCommit mit git-remote-codecommit](#).

Themen

- [Schritt 1: Erstkonfiguration für CodeCommit](#)
- [Schritt 2: Installieren von Git](#)
- [Schritt 3: Richten Sie den Credential Helper ein](#)
- [Schritt 4: Connect zur CodeCommit Konsole her und klonen Sie das Repository](#)
- [Nächste Schritte](#)

Schritt 1: Erstkonfiguration für CodeCommit

Gehen Sie wie folgt vor, um ein Amazon Web Services Services-Konto einzurichten, einen IAM-Benutzer zu erstellen und zu konfigurieren und den AWS CLI zu installieren.

Um einen IAM-Benutzer für den Zugriff zu erstellen und zu konfigurieren CodeCommit

1. Erstellen Sie ein Amazon Web Services Services-Konto, indem Sie zu <http://aws.amazon.com> gehen und Sign Up wählen.
2. Erstellen Sie einen IAM-Benutzer oder verwenden Sie einen vorhandenen in Ihrem Amazon Web Services Services-Konto. Stellen Sie sicher, dass Sie über eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel verfügen, die diesem IAM-Benutzer zugeordnet sind. Weitere Informationen finden Sie unter [Einen IAM-Benutzer in Ihrem Amazon Web Services Services-Konto erstellen](#).

Note

CodeCommit erfordert AWS Key Management Service. Wenn Sie einen vorhandenen IAM-Benutzer verwenden, stellen Sie sicher, dass dem Benutzer keine Richtlinien zugeordnet sind, die ausdrücklich die von CodeCommit erforderlichen AWS KMS Aktionen verweigern. Weitere Informationen finden Sie unter [AWS KMS und Verschlüsselung](#).

3. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter <https://console.aws.amazon.com/iam/>
4. Wählen Sie in der IAM-Konsole im Navigationsbereich Benutzer und dann den IAM-Benutzer aus, den Sie für den Zugriff konfigurieren möchten. CodeCommit
5. Wählen Sie auf der Registerkarte Permissions die Option Add Permissions.
6. Wählen Sie unter Grant permissions die Option Attach existing policies directly aus.

7. Wählen Sie `AWSCodeCommitPowerUser` aus der Liste der Richtlinien eine andere verwaltete Richtlinie für CodeCommit den Zugriff aus. Weitere Informationen finden Sie unter [AWS verwaltete Richtlinien für CodeCommit](#).

Nachdem Sie die Richtlinie ausgewählt haben, die Sie anhängen möchten, wählen Sie Weiter: Überprüfen aus, um die Liste der Richtlinien zu überprüfen, die an den IAM-Benutzer angehängt werden sollen. Ist die Liste korrekt, wählen Sie Add permissions aus.

Weitere Informationen zu CodeCommit verwalteten Richtlinien und zur gemeinsamen Nutzung des Zugriffs auf Repositorys mit anderen Gruppen und Benutzern finden Sie unter [Teilen Sie ein Repository](#) und [Authentifizierung und Zugriffskontrolle für AWS CodeCommit](#)

Um das zu installieren und zu konfigurieren AWS CLI

1. Laden Sie auf Ihrem lokalen Computer den herunter und installieren Sie ihn AWS CLI. Dies ist eine Voraussetzung für die Interaktion mit CodeCommit über die Befehlszeile. Wir empfehlen, AWS CLI Version 2 zu installieren. Es ist die neueste Hauptversion von AWS CLI und unterstützt alle aktuellen Funktionen. Es ist die einzige Version von AWS CLI , die die Verwendung eines Root-Kontos, Verbundzugriff oder temporärer Anmeldeinformationen mit `git-remote-codecommit` unterstützt.

Weitere Informationen finden Sie [unter Einrichtung der AWS Befehlszeilenschnittstelle](#).

 Note

CodeCommit funktioniert nur mit den AWS CLI Versionen 1.7.38 und höher. Es hat sich bewährt, die neueste verfügbare Version zu installieren oder AWS CLI auf sie zu aktualisieren. Führen AWS CLI Sie den `aws --version` Befehl aus, um festzustellen, welche Version von Sie installiert haben.

Informationen zum Upgrade einer älteren Version von AWS CLI auf die neueste Version finden Sie unter [Installation von AWS Command Line Interface](#).

2. Führen Sie diesen Befehl aus, um zu überprüfen, ob die CodeCommit Befehle für installiert AWS CLI sind.

```
aws codecommit help
```

Dieser Befehl gibt eine Liste von CodeCommit Befehlen zurück.

3. Konfigurieren Sie das AWS CLI mit einem Profil, indem Sie den `configure` Befehl wie folgt verwenden:.

```
aws configure
```

Wenn Sie dazu aufgefordert werden, geben Sie den AWS Zugriffsschlüssel und den AWS geheimen Zugriffsschlüssel des IAM-Benutzers an, mit CodeCommit dem Sie ihn verwenden möchten. Stellen Sie außerdem sicher, dass Sie angeben, AWS-Region wo sich das Repository befindet, z. B. `us-east-2`. Wenn Sie nach dem standardmäßigen Ausgabeformat gefragt werden, geben Sie `json` an. Wenn Sie beispielsweise ein Profil für einen IAM-Benutzer konfigurieren:

```
AWS Access Key ID [None]: Type your IAM user AWS access key ID here, and then press Enter
AWS Secret Access Key [None]: Type your IAM user AWS secret access key here, and then press Enter
Default region name [None]: Type a supported region for CodeCommit here, and then press Enter
Default output format [None]: Type json here, and then press Enter
```

Weitere Informationen zum Erstellen und Konfigurieren von Profilen zur Verwendung mit dem AWS CLI finden Sie im Folgenden:

- [Benannte Profile](#)
- [Verwenden einer IAM-Rolle in der AWS CLI](#)
- [Befehl „Set“](#)
- [Verbindung zu AWS CodeCommit Repositories mit rotierenden Anmeldeinformationen herstellen](#)

Um eine Verbindung zu einem Repository oder einer Ressource in einem anderen Repository herzustellen AWS-Region, müssen Sie das AWS CLI mit dem Standardregionsnamen neu konfigurieren. Zu den unterstützten Standardregionsnamen für CodeCommit gehören:

- `us-east-2`
- `us-east-1`
- `eu-west-1`
- `us-west-2`

- ap-northeast-1
- ap-southeast-1
- ap-southeast-2
- ap-southeast-3
- me-central-1
- eu-central-1
- ap-northeast-2
- sa-east-1
- us-west-1
- eu-west-2
- ap-south-1
- ap-south-1
- ca-central-1
- us-gov-west-1
- us-gov-east-1
- eu-north-1
- ap-east-1
- me-south-1
- cn-north-1
- cn-northwest-1
- eu-south-1
- ap-northeast-3
- af-south-1
- il-central-1

Weitere Informationen zu CodeCommit und finden AWS-Region Sie unter [Regionen und Git-Verbindungsendpunkte](#). Weitere Informationen zu IAM, Zugriffsschlüsseln und geheimen Schlüsseln finden Sie unter [Wie erhalte ich Anmeldeinformationen?](#) und [Zugriffsschlüssel für IAM-Benutzer verwalten](#). Weitere Informationen zu den Profilen AWS CLI und finden Sie unter [Benannte Profile](#).

Schritt 2: Installieren von Git

Um mit Dateien, Commits und anderen Informationen in CodeCommit Repositorys zu arbeiten, müssen Sie Git auf Ihrem lokalen Computer installieren. CodeCommit unterstützt Git-Versionen 1.7.9 und höher. Git Version 2.28 unterstützt die Konfiguration des Branchnamens für anfängliche Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git.

Um Git zu installieren, empfehlen wir Websites wie [Git Downloads](#).

Note

Git ist eine sich entwickelnde, regelmäßig aktualisierte Plattform. Gelegentlich kann sich eine Änderung der Funktionen auf die Art und Weise auswirken, mit der es funktioniert CodeCommit. Wenn Sie Probleme mit einer bestimmten Version von Git und haben CodeCommit, lesen Sie die Informationen unter [Fehlerbehebung](#).

Schritt 3: Richten Sie den Credential Helper ein

1. Verwenden Sie vom Terminal aus Git zum Ausführen `git config`, geben Sie die Verwendung des Git Credential Helper mit dem Credential Profile an und aktivieren Sie den Git AWS Credential Helper, um den Pfad an Repositorys zu senden:

```
git config --global credential.helper '!aws codecommit credential-helper $@'
git config --global credential.UseHttpPath true
```

Tip

Der Credential Helper verwendet das AWS Standard-Credential-Profil oder die EC2 Amazon-Instance-Rolle. Sie können das zu verwendende Profil vorgeben, z. B. `CodeCommitProfile`, wenn Sie ein AWS -Anmeldeinformationsprofil erstellt haben, das mit CodeCommit verwendet werden soll:

```
git config --global credential.helper '!aws --profile CodeCommitProfile
codecommit credential-helper $@'
```

Falls im Profilnamen Leerzeichen enthalten sind, muss der Name in Anführungszeichen (") gesetzt werden.

Sie können Profile anstatt global auch pro Repository konfigurieren, indem Sie `--local` anstelle von `--global` verwenden.

Das Git-Hilfsprogramm für Anmeldeinformationen schreibt folgenden Wert in `~/.gitconfig`:

```
[credential]
  helper = !aws --profile CodeCommitProfile codecommit credential-helper $@
  UseHttpPath = true
```

 Important

Wenn Sie einen anderen IAM-Benutzer auf demselben lokalen Computer für verwenden möchten CodeCommit, müssen Sie den `git config` Befehl erneut ausführen und ein anderes AWS Anmeldeinformationsprofil angeben.

2. Führen Sie den Befehl `git config --global --edit` aus und überprüfen Sie so, ob der oben genannte Wert in `~/.gitconfig` geschrieben wurde. War der Schreibvorgang erfolgreich, wird der Wert angezeigt (zusätzlich zu den Werten, die ggf. bereits in der globalen Git-Konfigurationsdatei vorhanden sind). Zum Beenden geben Sie `:q` ein und drücken die Eingabetaste.

Falls nach der Konfiguration des Hilfsprogramms für Anmeldeinformationen Probleme auftreten sollten, finden Sie weitere Informationen unter [Fehlerbehebung](#).

 Important

Wenn Sie macOS verwenden, gehen Sie wie folgt vor, um sicherzustellen, dass der Credential Helper korrekt konfiguriert ist.

3. Wenn Sie macOS verwenden, verwenden Sie HTTPS, um eine [Verbindung zu einem CodeCommit Repository](#) herzustellen. Nachdem Sie zum ersten Mal eine Verbindung zu einem CodeCommit Repository mit HTTPS hergestellt haben, schlägt der nachfolgende Zugriff nach etwa 15 Minuten fehl. Die Standard-Git-Version auf macOS verwendet das Keychain Access-Hilfsprogramm zum Speichern von Anmeldeinformationen. Aus Sicherheitsgründen ist das für den Zugriff auf dein CodeCommit Repository generierte Passwort temporär, sodass die im Schlüsselbund gespeicherten Anmeldeinformationen nach etwa 15 Minuten nicht mehr funktionieren. Damit die abgelaufenen Anmeldeinformationen nicht mehr verwendet werden, führen Sie einen der folgenden Schritte aus:

- Sie installieren eine Git-Version, in der nicht standardmäßig der Schlüsselbund verwendet wird.
 - Konfigurieren Sie das Keychain Access-Hilfsprogramm so, dass es keine Anmeldeinformationen für CodeCommit Repositorys bereitstellt.
1. Öffnen Sie das Schlüsselbund-Dienstprogramm. (Sie finden das Schlüsselbund-Dienstprogramm mit dem Finder.)
 2. Suchen Sie nach `git-codecommit.us-east-2.amazonaws.com`. Markieren Sie die Zeile, öffnen Sie das Kontextmenü oder klicken Sie mit der rechten Maustaste und wählen Sie dann Informationen abrufen aus.
 3. Wählen Sie die Registerkarte Access Control aus.
 4. Wählen Sie unter Confirm before allowing access den Eintrag `git-credential-osxkeychain` aus und klicken Sie auf das Minuszeichen, um den Eintrag aus der Liste zu entfernen.

 Note

Nachdem Sie `git-credential-osxkeychain` aus der Liste entfernt haben, wird bei jeder Ausführung eines Git-Befehls eine Popup-Meldung eingeblendet. Klicken Sie zum Fortfahren auf Deny. Falls Sie diese Dialogfelder umgehen möchten, finden Sie hier andere Optionen:

- Stellen Sie eine Verbindung CodeCommit mit SSH statt HTTPS her. Weitere Informationen finden Sie unter [Für SSH-Verbindungen unter Linux, macOS oder Unix](#).
- Wählen Sie im Schlüsselbund-Dienstprogramm auf der Registerkarte Access Control für `git-codecommit.us-east-2.amazonaws.com` die Option Allow all applications to access this item (access to this item is not restricted) aus. Auf diese Weise werden keine Popups angezeigt, aber die Anmeldeinformationen laufen ab (nach durchschnittlich etwa 15 Minuten) und die Fehlermeldung 403 wird angezeigt. In diesem Fall müssen Sie das Schlüsselbundobjekt löschen, um die Funktionalität wiederherzustellen.
- Weitere Informationen finden Sie unter [Git für macOS: Ich habe den Credential Helper erfolgreich konfiguriert, aber jetzt wird mir der Zugriff auf mein Repository verweigert \(403\)](#).

Schritt 4: Connect zur CodeCommit Konsole her und klonen Sie das Repository

Wenn Ihnen bereits ein Administrator den Namen und die Verbindungsdetails für das CodeCommit Repository gesendet hat, können Sie diesen Schritt überspringen und das Repository direkt klonen.

Um eine Verbindung zu einem CodeCommit Repository herzustellen

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie in der Regionsauswahl den Ort aus, AWS-Region an dem das Repository erstellt wurde. Repositories sind spezifisch für ein. AWS-Region Weitere Informationen finden Sie unter [Regionen und Git-Verbindungsendpunkte](#).
3. Suchen Sie das Repository, zu dem Sie eine Verbindung herstellen möchten, in der Liste und wählen Sie es aus. Wählen Sie Clone URL (URL klonen) und dann das Protokoll aus, das Sie beim Klonen oder bei der Verbindung zu dem Repository verwenden möchten. Dadurch wird die Klon-URL kopiert.
 - Kopieren Sie die HTTPS-URL, wenn Sie entweder Git-Anmeldeinformationen mit Ihrem IAM-Benutzer oder den Credential Helper verwenden, der im Lieferumfang von enthalten ist. AWS CLI
 - Kopieren Sie die HTTPS-URL (GRC), wenn Sie den Befehl git-remote-codecommit auf Ihrem lokalen Computer verwenden.
 - Kopieren Sie die SSH-URL, wenn Sie ein öffentliches/privates SSH-Schlüsselpaar mit Ihrem IAM-Benutzer verwenden.

Note

Wenn Sie statt einer Liste von Repositories eine Willkommensseite sehen, sind in dem Land, in dem Sie angemeldet sind, keine Repositories mit Ihrem AWS Konto verknüpft. AWS-Region Informationen zur Erstellung eines Repositories finden Sie unter [the section called "Erstellen eines -Repositories"](#), oder befolgen Sie die Schritte im Tutorial [Erste Schritte mit Git und CodeCommit](#).

4. Öffnen Sie ein Terminal und führen Sie den `git clone`-Befehl mit der kopierten HTTPS-URL aus. Um beispielsweise ein Repository mit dem Namen *MyDemoRepo* eines lokalen Repositorys mit dem Namen *my-demo-repo* der Region USA Ost (Ohio) zu klonen:

```
git clone https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
```

Nächste Schritte

Sie haben die Voraussetzungen erfüllt. Folgen Sie den Anweisungen unter, um mit der Verwendung [Erste Schritte mit CodeCommit](#) CodeCommit zu beginnen.

Einrichtungsschritte für HTTPS-Verbindungen zu AWS CodeCommit Repositorys unter Windows mit dem AWS CLI Credential Helper

Bevor Sie AWS CodeCommit zum ersten Mal eine Verbindung herstellen können, müssen Sie die Schritte zur Erstkonfiguration abschließen. Für die meisten Benutzer lässt sich das am einfachsten mit den Schritten unter [Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden](#) umsetzen. Wenn Sie jedoch CodeCommit über ein Root-Konto, Verbundzugriff oder temporäre Anmeldeinformationen eine Verbindung herstellen möchten, können Sie den Credential Helper verwenden, der in der enthalten ist. AWS CLI

Note

Der Credential Helper ist zwar eine unterstützte Methode, um CodeCommit mithilfe von Verbundzugriff, einem Identitätsanbieter oder temporären Anmeldeinformationen eine Verbindung herzustellen, es wird jedoch empfohlen, das Hilfsprogramm zu installieren und zu verwenden. `git-remote-codecommit` Weitere Informationen finden Sie unter [Einrichtungsschritte für HTTPS-Verbindungen AWS CodeCommit mit git-remote-codecommit](#).

In diesem Thema erfahren Sie, wie Sie das installieren AWS CLI, Ihren Computer und Ihr AWS Profil einrichten, eine Verbindung zu einem CodeCommit Repository herstellen und dieses Repository auf Ihrem Computer klonen. Dies wird auch als Erstellen eines lokalen Repositorys bezeichnet. Wenn

Sie mit Git noch nicht vertraut sind, finden Sie unter [Wo kann ich mehr über Git erfahren?](#) genauere Informationen.

Themen

- [Schritt 1: Erstkonfiguration für CodeCommit](#)
- [Schritt 2: Installieren von Git](#)
- [Schritt 3: Richten Sie den Credential Helper ein](#)
- [Schritt 4: Connect zur CodeCommit Konsole her und klonen Sie das Repository](#)
- [Nächste Schritte](#)

Schritt 1: Erstkonfiguration für CodeCommit

Gehen Sie wie folgt vor, um ein Amazon Web Services Services-Konto einzurichten, einen IAM-Benutzer zu erstellen und zu konfigurieren und den AWS CLI zu installieren. Das AWS CLI beinhaltet einen Credential Helper, den Sie für HTTPS-Verbindungen zu Ihren CodeCommit Repositories konfigurieren.

Um einen IAM-Benutzer für den Zugriff zu erstellen und zu konfigurieren CodeCommit

1. Erstellen Sie ein Amazon Web Services Services-Konto, indem Sie zu <http://aws.amazon.com> gehen und Sign Up wählen.
2. Erstellen Sie einen IAM-Benutzer oder verwenden Sie einen vorhandenen in Ihrem Amazon Web Services Services-Konto. Stellen Sie sicher, dass Sie über eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel verfügen, die diesem IAM-Benutzer zugeordnet sind. Weitere Informationen finden Sie unter [Einen IAM-Benutzer in Ihrem Amazon Web Services Services-Konto erstellen](#).

Note

CodeCommit erfordert AWS Key Management Service. Wenn Sie einen vorhandenen IAM-Benutzer verwenden, stellen Sie sicher, dass dem Benutzer keine Richtlinien zugeordnet sind, die ausdrücklich die von CodeCommit erforderlichen AWS KMS Aktionen verweigern. Weitere Informationen finden Sie unter [AWS KMS und Verschlüsselung](#).

3. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter. <https://console.aws.amazon.com/iam/>

4. Wählen Sie in der IAM-Konsole im Navigationsbereich Benutzer und dann den IAM-Benutzer aus, den Sie für den Zugriff konfigurieren möchten. CodeCommit
5. Wählen Sie auf der Registerkarte Permissions die Option Add Permissions.
6. Wählen Sie unter Grant permissions die Option Attach existing policies directly aus.
7. Wählen Sie AWSCodeCommitPowerUser aus der Liste der Richtlinien eine andere verwaltete Richtlinie für CodeCommit den Zugriff aus. Weitere Informationen finden Sie unter [AWS verwaltete Richtlinien für CodeCommit](#).

Nachdem Sie die Richtlinie ausgewählt haben, die Sie anhängen möchten, wählen Sie Weiter: Überprüfen aus, um die Liste der Richtlinien zu überprüfen, die an den IAM-Benutzer angehängt werden sollen. Ist die Liste korrekt, wählen Sie Add permissions aus.

Weitere Informationen zu CodeCommit verwalteten Richtlinien und zur gemeinsamen Nutzung des Zugriffs auf Repositorys mit anderen Gruppen und Benutzern finden Sie unter [Teilen Sie ein Repository](#) und [Authentifizierung und Zugriffskontrolle für AWS CodeCommit](#)

Um das zu installieren und zu konfigurieren AWS CLI

1. Laden Sie auf Ihrem lokalen Computer das herunter und installieren Sie es AWS CLI. Dies ist eine Voraussetzung für die Interaktion mit CodeCommit über die Befehlszeile. Wir empfehlen, AWS CLI Version 2 zu installieren. Es ist die neueste Hauptversion von AWS CLI und unterstützt alle aktuellen Funktionen. Es ist die einzige Version von AWS CLI , die die Verwendung eines Root-Kontos, Verbundzugriff oder temporärer Anmeldeinformationen mit git-remote-codecommit unterstützt.

Weitere Informationen finden Sie [unter Einrichtung der AWS Befehlszeilenschnittstelle](#).

 Note

CodeCommit funktioniert nur mit den AWS CLI Versionen 1.7.38 und höher. Es hat sich bewährt, die neueste verfügbare Version zu installieren oder AWS CLI auf sie zu aktualisieren. Führen AWS CLI Sie den `aws --version` Befehl aus, um festzustellen, welche Version von Sie installiert haben.

Informationen zum Upgrade einer älteren Version von AWS CLI auf die neueste Version finden Sie unter [Installation von AWS Command Line Interface](#).

2. Führen Sie diesen Befehl aus, um zu überprüfen, ob die CodeCommit Befehle für installiert AWS CLI sind.

```
aws codecommit help
```

Dieser Befehl gibt eine Liste von CodeCommit Befehlen zurück.

3. Konfigurieren Sie das AWS CLI mit einem Profil, indem Sie den configure Befehl wie folgt verwenden:.

```
aws configure
```

Wenn Sie dazu aufgefordert werden, geben Sie den AWS Zugriffsschlüssel und den AWS geheimen Zugriffsschlüssel des IAM-Benutzers an, mit CodeCommit dem Sie ihn verwenden möchten. Stellen Sie außerdem sicher, dass Sie angeben, AWS-Region wo sich das Repository befindet, z. B. us-east-2 Wenn Sie nach dem standardmäßigen Ausgabeformat gefragt werden, geben Sie json an. Wenn Sie beispielsweise ein Profil für einen IAM-Benutzer konfigurieren:

```
AWS Access Key ID [None]: Type your IAM user AWS access key ID here, and then press Enter
AWS Secret Access Key [None]: Type your IAM user AWS secret access key here, and then press Enter
Default region name [None]: Type a supported region for CodeCommit here, and then press Enter
Default output format [None]: Type json here, and then press Enter
```

Weitere Informationen zum Erstellen und Konfigurieren von Profilen zur Verwendung mit dem AWS CLI finden Sie im Folgenden:

- [Benannte Profile](#)
- [Verwenden einer IAM-Rolle in der AWS CLI](#)
- [Befehl „Set“](#)
- [Verbindung zu AWS CodeCommit Repositories mit rotierenden Anmeldeinformationen herstellen](#)

Um eine Verbindung zu einem Repository oder einer Ressource in einem anderen Repository herzustellen AWS-Region, müssen Sie das AWS CLI mit dem Standardregionsnamen neu konfigurieren. Zu den unterstützten Standardregionsnamen für CodeCommit gehören:

- us-east-2
- us-east-1
- eu-west-1
- us-west-2
- ap-northeast-1
- ap-southeast-1
- ap-southeast-2
- ap-southeast-3
- me-central-1
- eu-central-1
- ap-northeast-2
- sa-east-1
- us-west-1
- eu-west-2
- ap-south-1
- ap-south-1
- ca-central-1
- us-gov-west-1
- us-gov-east-1
- eu-north-1
- ap-east-1
- me-south-1
- cn-north-1
- cn-northwest-1
- eu-south-1
- ap-northeast-3

- af-south-1
- il-central-1

Weitere Informationen zu CodeCommit und finden AWS-Region Sie unter [Regionen und Git-Verbindungsendpunkte](#). Weitere Informationen zu IAM, Zugriffsschlüsseln und geheimen Schlüsseln finden Sie unter [Wie erhalte ich Anmeldeinformationen?](#) und [Zugriffsschlüssel für IAM-Benutzer verwalten](#). Weitere Informationen zu den Profilen AWS CLI und finden Sie unter [Benannte Profile](#).

Schritt 2: Installieren von Git

Um mit Dateien, Commits und anderen Informationen in CodeCommit Repositorys zu arbeiten, müssen Sie Git auf Ihrem lokalen Computer installieren. CodeCommit unterstützt Git-Versionen 1.7.9 und höher. Git Version 2.28 unterstützt die Konfiguration des Branchnamens für anfängliche Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git.

Um Git zu installieren, empfehlen wir Websites wie [Git für Windows](#). Wenn du diesen Link verwendest, um Git zu installieren, kannst du alle Standardeinstellungen der Installation akzeptieren, mit Ausnahme der folgenden:

- Wenn Sie während des Schritts Anpassen Ihrer PATH-Umgebung dazu aufgefordert werden, wählen Sie in der Befehlszeile die Option, Git zu verwenden.
- (Optional) Wenn Sie beabsichtigen, HTTPS mit dem Credential Helper zu verwenden, der in AWS CLI statt der Konfiguration von Git-Anmeldeinformationen für enthalten ist CodeCommit, auf der Seite Zusätzliche Optionen konfigurieren, stellen Sie sicher, dass die Option Git Credential Manager aktivieren deaktiviert ist. Der Git Credential Manager ist nur kompatibel, CodeCommit wenn IAM-Benutzer Git-Anmeldeinformationen konfigurieren. Weitere Informationen erhalten Sie unter [Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden](#) und [Git für Windows: Ich habe Git für Windows installiert, aber mir wird der Zugriff auf mein Repository verweigert \(403\)](#).

Note

Git ist eine sich entwickelnde, regelmäßig aktualisierte Plattform. Gelegentlich kann sich eine Änderung der Funktionen auf die Art und Weise auswirken, mit der es funktioniert

CodeCommit. Wenn Sie Probleme mit einer bestimmten Version von Git und haben CodeCommit, lesen Sie die Informationen unter [Fehlerbehebung](#).

Schritt 3: Richten Sie den Credential Helper ein

Das AWS CLI beinhaltet einen Git-Helfer für Anmeldeinformationen, mit dem CodeCommit Sie ihn verwenden können. Der Git Credential Helper benötigt ein AWS Anmeldeinformationsprofil, das eine Kopie der AWS Zugriffsschlüssel-ID und des AWS geheimen Zugriffsschlüssels eines IAM-Benutzers (zusammen mit einem AWS-Region Standardnamen und einem Standardausgabeformat) speichert. Der Git Credential Helper verwendet diese Informationen, um sich automatisch zu authentifizieren, CodeCommit sodass Sie diese Informationen nicht jedes Mal eingeben müssen, wenn Sie Git für die Interaktion verwenden. CodeCommit

1. Öffnen Sie eine Befehlszeile und verwenden Sie Git zum Ausführengit config. Geben Sie dabei die Verwendung des Git Credential Helper mit dem AWS Credential Profile an, wodurch der Git Credential Helper den Pfad an Repositorys senden kann:

```
git config --global credential.helper "!aws codecommit credential-helper $@"  
git config --global credential.UseHttpPath true
```

Das Git-Hilfsprogramm für Anmeldeinformationen schreibt folgende Werte in die .gitconfig-Datei:

```
[credential]  
  helper = !aws codecommit credential-helper $@  
  UseHttpPath = true
```

Important

- Falls Sie anstelle der Windows-Befehlszeile einen Bash-Emulator einsetzen, müssen Sie einfache (statt doppelte) Anführungszeichen eingeben.
- Der Credential Helper verwendet das AWS Standardprofil oder die EC2 Amazon-Instance-Rolle. Wenn Sie ein zu verwendendes AWS Anmeldeinformationsprofil erstellt haben, können Sie den Befehl wie folgt ändern *CodeCommitProfile*, um ihn stattdessen zu verwenden:

```
git config --global credential.helper "!aws codecommit credential-helper  
--profile CodeCommitProfile $@"
```

Dann werden folgende Werte in die .gitconfig-Datei geschrieben:

```
[credential]  
  helper = !aws codecommit credential-helper --profile=CodeCommitProfile  
  $@  
  UseHttpPath = true
```

- Falls Ihr Profilname Leerzeichen enthält, müssen Sie nach der Ausführung dieses Befehls den Namen in der .gitconfig-Datei in einfache Anführungszeichen (') setzen. Andernfalls können Sie das Hilfsprogramm für Anmeldeinformationen nicht nutzen.
- Wenn Sie im Rahmen der Installation von Git für Windows auch das Git-Dienstprogramm zur Verwaltung von Anmeldeinformationen (Git Credential Manager) implementiert haben, werden Ihnen nach den ersten Verbindungsversuchen entweder 403-Fehlermeldungen angezeigt oder Sie werden vom Dienstprogramm aufgefordert, Ihre Anmeldeinformationen einzugeben. Dieses Problem lässt sich am besten lösen, indem Sie Git für Windows deinstallieren und dann ohne den Git Credential Manager erneut installieren, da das Dienstprogramm mit CodeCommit nicht kompatibel ist. Wenn Sie das Git Credential Manager-Hilfsprogramm behalten möchten, müssen Sie zusätzliche Konfigurationsschritte ausführen, um es ebenfalls zu verwenden. Dazu gehört auch das manuelle Ändern der Datei .gitconfig CodeCommit, um anzugeben, ob der Credential Helper für AWS CodeCommit die Verbindung zu verwendet werden soll. CodeCommit Entfernen Sie sämtliche gespeicherten Anmeldeinformationen aus dem Credential Manager-Dienstprogramm (dies finden Sie in der Systemsteuerung). Nachdem Sie alle gespeicherten Anmeldeinformationen gelöscht haben, fügen Sie folgenden Abschnitt zur .gitconfig-Datei hinzu, speichern diese und versuchen, über ein neues Befehlszeilenfenster eine Verbindung herzustellen:

```
[credential "https://git-codecommit.us-east-2.amazonaws.com"]  
  helper = !aws codecommit credential-helper $@  
  UseHttpPath = true  
  
[credential "https://git-codecommit.us-east-1.amazonaws.com"]  
  helper = !aws codecommit credential-helper $@
```

```
UseHttpPath = true
```

Möglicherweise ist zusätzlich eine Neukonfiguration der Einstellungen für git config erforderlich. Dabei geben Sie `--system` anstelle von `--global` oder `--local` an, damit alle Verbindungen ordnungsgemäß hergestellt werden können.

- Wenn Sie verschiedene IAM-Benutzer auf demselben lokalen Computer für verwenden möchten CodeCommit, sollten Sie `git config --local` anstelle von `git config --global` die Konfiguration für jedes Anmeldeinformationsprofil angeben und die Konfiguration ausführen. AWS

2. Führen Sie den Befehl `git config --global --edit` aus, um zu prüfen, ob die vorigen Werte in die `.gitconfig`-Datei Ihres Benutzerprofils geschrieben wurden (standardmäßig `%HOME%\.gitconfig` oder `drive:\Users\UserName\.gitconfig`). War der Schreibvorgang erfolgreich, werden die Werte angezeigt (zusätzlich zu den Werten, die ggf. bereits in der globalen Git-Konfigurationsdatei vorhanden sind). Zum Beenden geben Sie `:q` ein und drücken die Eingabetaste.

Schritt 4: Connect zur CodeCommit Konsole her und klonen Sie das Repository

Wenn Ihnen bereits ein Administrator den Namen und die Verbindungsdetails für das CodeCommit Repository gesendet hat, können Sie diesen Schritt überspringen und das Repository direkt klonen.

Um eine Verbindung zu einem CodeCommit Repository herzustellen

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie in der Regionsauswahl den Ort aus, AWS-Region an dem das Repository erstellt wurde. Repositories sind spezifisch für eine AWS-Region. Weitere Informationen finden Sie unter [Regionen und Git-Verbindungsendpunkte](#).
3. Suchen Sie das Repository, zu dem Sie eine Verbindung herstellen möchten, in der Liste und wählen Sie es aus. Wählen Sie Clone URL (URL klonen) und dann das Protokoll aus, das Sie beim Klonen oder bei der Verbindung zu dem Repository verwenden möchten. Dadurch wird die Klon-URL kopiert.

- Kopieren Sie die HTTPS-URL, wenn Sie entweder Git-Anmeldeinformationen mit Ihrem IAM-Benutzer oder den Credential Helper verwenden, der im Lieferumfang von enthalten ist. AWS CLI
- Kopieren Sie die HTTPS-URL (GRC), wenn Sie den Befehl `git-remote-codecommit` auf Ihrem lokalen Computer verwenden.
- Kopieren Sie die SSH-URL, wenn Sie ein öffentliches/privates SSH-Schlüsselpaar mit Ihrem IAM-Benutzer verwenden.

Note

Wenn Sie statt einer Liste von Repositorys eine Willkommensseite sehen, sind in dem Land, in dem Sie angemeldet sind, keine Repositorys mit Ihrem AWS Konto verknüpft. AWS-Region Informationen zur Erstellung eines Repositorys finden Sie unter [the section called “Erstellen eines -Repositorys”](#), oder befolgen Sie die Schritte im Tutorial [Erste Schritte mit Git und CodeCommit](#).

4. Öffnen Sie eine Befehlszeile und führen Sie den `git clone` Befehl mit der HTTPS-URL aus, die Sie kopiert haben. Dadurch wird das lokale Repository in einem Unterverzeichnis des Verzeichnisses erstellt, in dem Sie den Befehl ausgeführt haben. Um beispielsweise ein Repository mit dem Namen *MyDemoRepo* eines lokalen Repositorys mit dem Namen *my-demo-repo* in der Region USA Ost (Ohio) zu klonen:

```
git clone https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
```

Bei einigen Windows-Versionen werden Sie möglicherweise über eine Popup-Meldung aufgefordert, Ihren Benutzernamen und Ihr Passwort einzugeben. Dabei handelt es sich um das integrierte System für die Anmeldeinformationsverwaltung von Windows, dieses ist jedoch nicht mit dem AWS CodeCommit-Hilfsprogramm für Anmeldeinformationen kompatibel. Klicken Sie auf Abbrechen.

Nächste Schritte

Sie haben die Voraussetzungen erfüllt. Folgen Sie den Anweisungen unter, um mit der Verwendung [Erste Schritte mit CodeCommit](#) CodeCommit zu beginnen.

Erste Schritte

Der einfachste Weg, damit zu beginnen, CodeCommit besteht darin, die Schritte unter zu befolgen [Erste Schritte mit CodeCommit](#) . Wenn du neu bei Git und bist CodeCommit, solltest du auch in Betracht ziehen, die Schritte unter zu befolgen [Erste Schritte mit Git und CodeCommit](#). Dies hilft dir, dich mit den Grundlagen CodeCommit und der Verwendung von Git vertraut zu machen, wenn du mit deinen CodeCommit Repositorys interagierst.

Du kannst auch dem Tutorial in [Simple Pipeline Walkthrough](#) folgen, CodeCommit um zu erfahren, wie du dein CodeCommit Repository als Teil einer Continuous Delivery-Pipeline verwendest. CodePipeline

Die Tutorials in diesem Abschnitt wurden unter der Annahme verfasst, dass Sie die [Voraussetzungen und die Einrichtung](#) abgeschlossen haben, einschließlich:

- Zuweisen von Berechtigungen für den IAM-Benutzer.
- Einrichten der Verwaltung von Anmeldeinformationen für HTTPS- oder SSH-Verbindungen auf dem lokalen Computer, der für dieses Tutorial verwendet wird.
- Konfiguration, AWS CLI ob Sie die Befehlszeile oder das Terminal für alle Operationen verwenden möchten, einschließlich der Erstellung des Repositorys.

Themen

- [Erste Schritte mit AWS CodeCommit](#)
- [Erste Schritte mit Git und AWS CodeCommit](#)

Erste Schritte mit AWS CodeCommit

Dieses Tutorial zeigt Ihnen, wie Sie einige wichtige CodeCommit Funktionen verwenden. Zunächst erstellen Sie ein Repository und tragen einige Änderungen ein (Commit). Dann durchsuchen Sie die Dateien und sehen sich die Änderungen an. Sie können auch eine Pull-Anforderung erstellen, damit andere Benutzer die Änderungen an Ihrem Code prüfen und ggf. kommentieren.

Informationen dazu, wie Sie vorhandenen Code migrieren möchten CodeCommit, finden Sie unter [Migrieren zu AWS CodeCommit](#).

Wenn Sie mit Git nicht vertraut sind, sollten Sie auch [Erste Schritte mit Git und CodeCommit](#) durcharbeiten. Nachdem Sie diese Tutorials abgeschlossen haben, sollten Sie über ausreichend

Erfahrung verfügen, um damit zu beginnen, sie CodeCommit für Ihre eigenen Projekte und in Teamumgebungen zu verwenden.

Die CodeCommit Konsole enthält hilfreiche Informationen in einem zusammenklappbaren Bereich, den Sie über das Informationssymbol



oder einen beliebigen Informationslink auf der Seite öffnen können. Sie können diesen Bereich jederzeit schließen.

The screenshot shows the AWS CodeCommit console interface. At the top, there's a breadcrumb trail: Developer Tools > CodeCommit > Repositories > MyDemoRepo. Below this, the repository name 'MyDemoRepo' is displayed. There are buttons for 'Notify', a dropdown for 'main', 'Create pull request', and 'Clone URL'. The main content area shows the file 'MyDemoRepo / cl_sample.js' with an 'Edit' button. The file content is a JavaScript snippet using the AWS SDK for CodeCommit. On the right, a sidebar titled 'View a file in a repository' provides instructions on viewing file contents, switching between rendered and source code views, and editing files. It also includes a 'Learn more' section with links to 'Browse files', 'Edit a file', and 'Add a file'.

Die CodeCommit Konsole bietet auch eine Möglichkeit, schnell nach Ihren Ressourcen wie Repositories, Build-Projekten, Bereitstellungsanwendungen und Pipelines zu suchen. Wählen Sie Go to Ressource (Zur Ressource) oder drücken Sie die Taste / und geben Sie dann den Namen der Ressource ein. Alle Übereinstimmungen werden in der Liste angezeigt. Bei der Suche wird nicht zwischen Groß- und Kleinschreibung unterschieden. Sie sehen nur die Ressourcen, für die Sie die Berechtigung besitzen. Weitere Informationen finden Sie unter [Anzeigen von Ressourcen in der Konsole](#).

Voraussetzungen

Bevor Sie beginnen, müssen Sie die [Voraussetzungen und das Einrichtungsverfahren](#) abschließen, einschließlich:

- Dem IAM-Benutzer Berechtigungen zuweisen.
- Einrichten der Verwaltung von Anmeldeinformationen für HTTPS- oder SSH-Verbindungen auf dem lokalen Computer, der für dieses Tutorial verwendet wird.
- Konfiguration, AWS CLI ob Sie die Befehlszeile oder das Terminal für alle Operationen verwenden möchten, einschließlich der Erstellung des Repositorys.

Themen

- [Schritt 1: Erstellen eines CodeCommit-Repositorys](#)
- [Schritt 2: Fügen Sie Dateien zu Ihrem Repository hinzu](#)
- [Schritt 3: Durchsuchen Sie den Inhalt Ihres Repositorys](#)
- [Schritt 4: Einen Pull-Request erstellen und gemeinsam daran arbeiten](#)
- [Schritt 5: Bereinigen](#)
- [Schritt 6: Nächste Schritte](#)

Schritt 1: Erstellen eines CodeCommit-Repositorys

Sie können die CodeCommit Konsole verwenden, um ein CodeCommit Repository zu erstellen. Wenn Sie bereits ein Repository haben und für dieses Tutorial verwenden möchten, können Sie diesen Schritt überspringen.

Note

Je nach Nutzung können Ihnen Gebühren für die Erstellung oder den Zugriff auf ein Repository berechnet werden. Weitere Informationen finden Sie auf der CodeCommit Produktinformationsseite unter [Preise](#).

Um das CodeCommit Repository zu erstellen

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.

2. Verwenden Sie die Regionsauswahl, um den Ort auszuwählen, AWS-Region an dem Sie das Repository erstellen möchten. Weitere Informationen finden Sie unter [Regionen und Git-Verbindungsendpunkte](#).
3. Wählen Sie auf der Seite Repositorys die Option Repository auswählen.
4. Geben Sie auf der Seite Create repository (Repository erstellen) unter Repository name (Repository-Name) einen Namen für Ihr Repository ein (z. B. **MyDemoRepo**).

 Note

Bei Repository-Namen muss die Groß- und Kleinschreibung beachtet werden. Er darf nicht mehr als 100 Zeichen enthalten. Weitere Informationen finden Sie unter [Limits](#).

5. (Optional) Geben Sie unter Description (Beschreibung) eine Beschreibung ein (z. B. **My demonstration repository**). Dadurch können Sie und andere Benutzer den Zweck des Repositorys leichter erkennen.
6. (Optional) Wählen Sie Tag hinzufügen, um Ihrem Repository ein oder mehrere Repository-Tags (eine benutzerdefinierte Attributbezeichnung, mit der Sie Ihre AWS Ressourcen organisieren und verwalten können) hinzuzufügen. Weitere Informationen finden Sie unter [Kennzeichnen von Repositorys in AWS CodeCommit](#).
7. (Optional) Erweitern Sie Zusätzliche Konfiguration, um anzugeben, ob Sie den Standardschlüssel Von AWS verwalteter Schlüssel oder Ihren eigenen, vom Kunden verwalteten Schlüssel zum Verschlüsseln und Entschlüsseln der Daten in diesem Repository verwenden möchten. Wenn Sie Ihren eigenen, vom Kunden verwalteten Schlüssel verwenden möchten, müssen Sie sicherstellen, dass er dort verfügbar ist, AWS-Region wo Sie das Repository erstellen, und dass der Schlüssel aktiv ist. Weitere Informationen finden Sie unter [AWS Key Management Service und Verschlüsselung für AWS CodeCommit Repositorien](#).
8. (Optional) Wählen Sie Amazon CodeGuru Reviewer für Java und Python aktivieren, wenn dieses Repository Java- oder Python-Code enthält und Sie möchten, dass CodeGuru Reviewer diesen Code analysiert. CodeGuru Reviewer verwendet mehrere Modelle für maschinelles Lernen, um Codefehler zu finden und automatisch Verbesserungen und Korrekturen in Pull-Requests vorzuschlagen. Weitere Informationen finden Sie im Amazon CodeGuru Reviewer-Benutzerhandbuch.
9. Wählen Sie Create (Erstellen) aus.

Create repository

Create a secure repository to store and share your code. Begin by typing a repository name and a description for your repository. Repository names are included in the URLs for that repository.

Repository settings

Repository name

100 characters maximum. Other limits apply.

Description - *optional*

1,000 characters maximum

Tags

☐ **Enable Amazon CodeGuru Reviewer for Java and Python - *optional***

Get recommendations to improve the quality of the Java and Python code for all pull requests in this repository.

A service-linked role will be created in IAM on your behalf if it does not exist.

[Cancel](#)[Create](#)

Note

Wenn Sie einen anderen Namen als MyDemoRepo für das Repository verwenden, muss dieser auch in den weiteren Schritten des Tutorials verwendet werden.

Wenn das Repository geöffnet wird, werden Informationen darüber angezeigt, wie Sie Dateien direkt von der CodeCommit Konsole aus hinzufügen können.

Schritt 2: Fügen Sie Dateien zu Ihrem Repository hinzu

Sie können Dateien wie folgt zu Ihrem Repository hinzufügen:

- Eine Datei in der CodeCommit Konsole erstellen. Wenn Sie die erste Datei für ein Repository in der Konsole erstellen, wird für Sie ein Branch mit dem Namen main erstellt. Dieser Branch ist der Standard-Branch für dein Repository.
- Eine Datei mithilfe der CodeCommit Konsole von Ihrem lokalen Computer hochladen. Wenn Sie die erste Datei für ein Repository von der Konsole hochladen, wird für Sie ein Zweig mit dem Namen main erstellt. Dieser Branch ist der Standard-Branch für dein Repository.
- Verwenden Sie einen Git-Client, um das Repository auf Ihren lokalen Computer zu klonen, und fügen Sie dann Dateien hinzu, übertragen Sie sie und übertragen Sie sie in das CodeCommit Repository. Ein Branch wird für dich als Teil des ersten Commits von Git erstellt und als Standard-Branch für dein Repository festgelegt. Der Name des Branches ist die Standardauswahl Ihres Git-Clients. Erwägen Sie, Ihren Git-Client so zu konfigurieren, dass er main als Namen für den ersten Branch verwendet.

Note

Du kannst jederzeit Branches erstellen und den Standard-Branch für ein Repository ändern. Weitere Informationen finden Sie unter [Mit Branches in AWS CodeCommit Repositorys arbeiten](#).

Der einfachste Weg, um loszulegen, besteht darin, die CodeCommit Konsole zu öffnen und eine Datei hinzuzufügen. Auf diese Weise erstellen Sie auch einen Standardzweig für Ihr Repository mit dem Namen main. Anweisungen zum Hinzufügen einer Datei und zum Erstellen eines ersten Commits zu einem Repository mit dem AWS CLI finden [Sie unter Erstellen Sie den ersten Commit für ein Repository mit dem AWS CLI](#).

So fügen Sie dem Repository eine Datei hinzu

1. Wählen Sie in der Navigationsleiste des Repositorys die Option Code aus.
2. Wählen Sie Add file (Datei hinzufügen) aus. Erstellen Sie dann eine Datei oder laden Sie eine Datei vom Computer hoch. Dieses Tutorial erläutert beide Verfahren.
3. Gehen Sie folgendermaßen vor, um eine Datei hinzuzufügen:

- a. Wählen Sie in der Dropdown-Liste der Branches den Branch aus, dem die Datei hinzugefügt werden soll. Der Standard-Branch wird automatisch für Sie ausgewählt. In dem hier gezeigten Beispiel wird der Standard-Branch benannt *main*. Wenn Sie die Datei einem anderen Branch hinzufügen möchten, wählen Sie einen anderen Branch aus.
- b. Geben Sie in das Feld File name (Dateiname) einen Namen für die Datei ein. Geben Sie den Code für die Datei in den Code-Editor ein.
- c. Geben Sie im Feld Author name (Name des Autors) den Namen ein, der anderen Repository-Benutzern angezeigt werden soll.
- d. Geben Sie eine E-Mail-Adresse in das Feld Email address (E-Mail-Adresse) ein.
- e. (Optional) Geben Sie in das Feld Commit message (Commit-Nachricht) eine kurze Nachricht ein. Dies ist zwar optional, wir empfehlen jedoch, eine Commit-Nachricht hinzuzufügen, da dies dazu beiträgt, dass Ihre Teammitglieder verstehen, warum Sie diese Datei hinzugefügt haben. Wenn Sie keine Commit-Nachricht eingeben, wird eine Standardnachricht verwendet.
- f. Wählen Sie Commit changes (Änderungen übernehmen) aus.

Gehen Sie folgendermaßen vor, um eine Datei hochzuladen:

- Wenn Sie eine Datei hochladen möchten, wählen Sie die gewünschte Datei aus.

Upload a file

MyDemoRepo [Info](#)

Name	Size	Actions
Upload file Choose a file to upload. Choose file		

Commit changes to master

Author name

Email address

Commit message - optional
A default commit message will be used if you do not provide one.

- Geben Sie im Feld Author name (Name des Autors) den Namen ein, der anderen Repository-Benutzern angezeigt werden soll.
- Geben Sie eine E-Mail-Adresse in das Feld Email address (E-Mail-Adresse) ein.
- (Optional) Geben Sie in das Feld Commit message (Commit-Nachricht) eine kurze Nachricht ein. Dies ist zwar optional, wir empfehlen jedoch, eine Commit-Nachricht hinzuzufügen, da dies dazu beiträgt, dass Ihre Teammitglieder verstehen, warum Sie diese Datei hinzugefügt haben. Wenn Sie keine Commit-Nachricht eingeben, wird eine Standardnachricht verwendet.
- Wählen Sie Commit changes (Änderungen übernehmen) aus.

Weitere Informationen finden Sie unter [Arbeiten mit Dateien in AWS CodeCommit Repositorys](#).

Um das Repository mit einem Git-Client zu klonen, installieren Sie Git auf Ihrem lokalen Computer und klonen Sie dann das CodeCommit Repository. Fügen Sie dem lokalen Repository einige Dateien hinzu und übertragen Sie diese per Push an das CodeCommit -Repository. Eine umfassende Einführung finden Sie unter [Erste Schritte mit Git und CodeCommit](#). Wenn du mit Git vertraut bist, dir aber nicht sicher bist, wie du das mit einem CodeCommit Repository bewerkstelligen kannst, findest du Beispiele und Anleitungen in [Erstellen Sie einen Commit](#)[Schritt 2: Erstellen Sie ein lokales Repo](#), oder [Herstellen einer Verbindung mit einem Repository](#).

Nachdem Sie dem CodeCommit Repository einige Dateien hinzugefügt haben, können Sie sie in der Konsole ansehen.

Schritt 3: Durchsuchen Sie den Inhalt Ihres Repositorys

Sie können die CodeCommit Konsole verwenden, um die Dateien in einem Repository zu überprüfen oder den Inhalt einer Datei schnell zu lesen. So können Sie feststellen, welchen Branch Sie sich genauer ansehen oder ob Sie eine lokale Kopie eines Repositorys erstellen sollten.

So durchsuchen Sie das Repository

1. Wählen Sie unter Repositorys die Option MyDemoRepo.
2. Die Seite zeigt den Inhalt im Standard-Branch des Repositorys an. Um einen anderen Branch oder den Code an einem bestimmten Tag anzuzeigen, wählen Sie den Branch oder das Tag in der Liste aus. Im folgenden Screenshot ist die Ansicht auf den Hauptzweig eingestellt.

Developer Tools > CodeCommit > Repositories > MyDemoRepo

MyDemoRepo

Notify main Create pull request Clone URL

MyDemoRepo Info Add file

Name
batch files for https
batch files for ssh
tmp
ahs_count.py
apis_meliponini.txt
bees.txt
bird.txt
bumblebee.txt
cat.txt
cl_sample.js

3. Um den Inhalt einer Datei in Ihrem Repository anzuzeigen, wählen Sie die Datei in der Liste aus. Wählen Sie das Einstellungssymbol aus, um die Farbe des angezeigten Codes zu ändern.

Developer Tools > CodeCommit > Repositories > MyDemoRepo

MyDemoRepo

Notify main Create pull request Clone URL

MyDemoRepo / cl_sample.js Info Edit

```

1 var aws = require('aws-sdk');
2 var codecommit = new aws.CodeCommit({ apiVersion: '2015-04-13' });
3
4 exports.handler = function(event, context) {
5   //Log the updated references from the event
6   var references = event.Records[0].codecommit.references.map(function(reference) {return reference.ref;});
7   console.log('References:', references);
8
9   //Get the repository from the event and show its git clone URL
10  var repository = event.Records[0].eventSourceARN.split(":")[5];
11  var params = {
12    repositoryName: repository
13  };
14  codecommit.getRepository(params, function(err, data) {
15    if (err) {
16      console.log(err);
17      var message = "Error getting repository metadata for repository " + repository;
18      console.log(message);
19      context.fail(message);
20    } else {
21      console.log('Clone URL', data.repositoryMetadata.cloneUrlHttp);
22      context.succeed(data.repositoryMetadata.cloneUrlHttp);
23    }
24  });
25 };
```

Weitere Informationen finden Sie unter [Dateien in einem Repository durchsuchen](#).

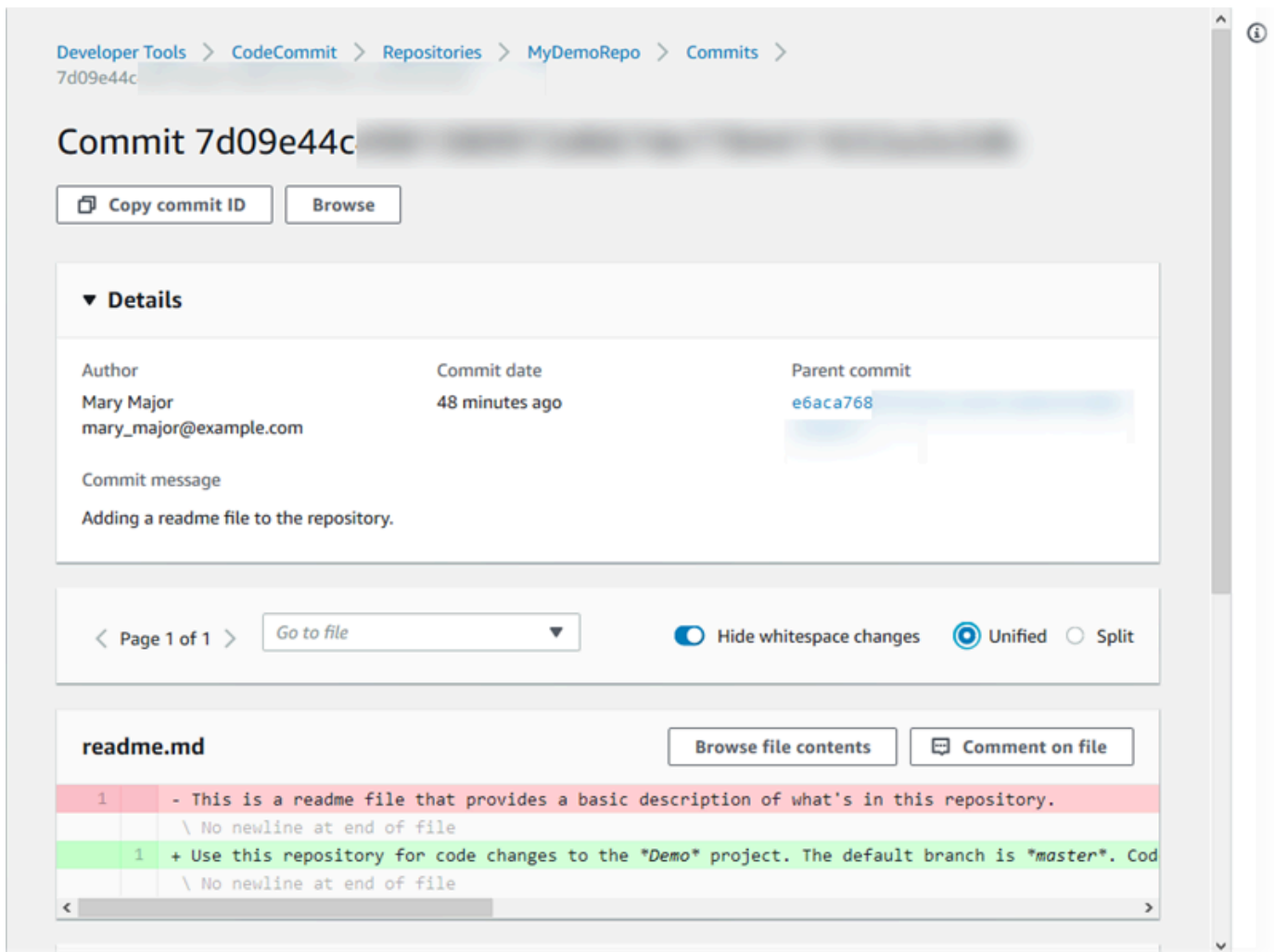
- Um den Commit-Verlauf des Repositorys zu durchsuchen, wählen Sie Commits. Die Konsole zeigt den Commit-Verlauf für den Standard-Branch in umgekehrter chronologischer Reihenfolge an. Prüfen Sie die Commit-Details anhand von Autor, Datum usw.

Commit ID	Commit message	Commit date	Author	Actions
56bde83f	Cleaned up old variable	2 minutes ago	Saanvi Sarkar	Copy ID Browse
e836ef2f	Minor changes to one file for upcoming class	4 minutes ago	Li Juan	Copy ID Browse
6d91526d	Eliminated blank line at end of file	5 minutes ago	Maria Garcia	Copy ID Browse
e4817953	Adding another analyzer	8 minutes ago	Jane Doe	Copy ID Browse
8a62eb3c	Updated a file with changes	10 minutes ago	Saanvi Sarkar	Copy ID Browse
3b459eca	Minor changes to incremental-repo-migration.py	12 minutes ago	Mary Major	Copy ID Browse
ac6730ba	Added a new sample and fixed comment spacing in another sample	16 minutes ago	Mary Major	Copy ID Browse
bfe67f5d	fixed instruction	19 minutes ago	Saanvi Sarkar	Copy ID Browse
9e32138a	Updating instructions	20 minutes ago	Mary Major	Copy ID Browse
98aa807b	add image files for new feature	30 minutes ago	Li Juan	Copy ID Browse
c357d32f	testing again from eclipse	30 minutes ago	Maria Garcia	Copy ID Browse
18a6c2c9	more testing	40 minutes ago	Mary Major	Copy ID Browse

- Um den Commit-Verlauf nach [Branch](#) oder [Git-Tag](#) anzuzeigen, wählen Sie den Branch oder das Tag, den bzw. das Sie anzeigen möchten, in der Liste aus.
- Wählen Sie die gekürzte Commit-ID aus, um die Unterschiede zwischen einem Commit und seinem übergeordneten Commit anzuzeigen. Sie können wählen, wie die Änderungen angezeigt werden sollen, z. B. ob Leerzeichenänderungen ein- oder ausgeblendet werden sollen und ob Änderungen inline (Ansicht Unified (Vereinheitlicht) oder nebeneinander (Ansicht Split (Geteilt)) angezeigt werden sollen.

Note

Ihre Präferenzen für die Anzeige des Codes sowie andere Konsoleinstellungen werden als Browser-Cookies gespeichert, wenn Sie sie ändern. Weitere Informationen finden Sie unter [Mit Benutzereinstellungen arbeiten](#).



- Um alle Kommentare zu einem Commit anzuzeigen, wählen Sie den Commit aus und blättern Sie durch die Änderungen, um sie inline anzuzeigen. Sie können auch Ihre eigenen Kommentare hinzufügen und auf Kommentare anderer Benutzer antworten.

Weitere Informationen finden Sie unter [Kommentar zu einem Commit](#).

- Um die Unterschiede zwischen zwei beliebigen Commit-Spezifikatoren, einschließlich Tags, Branches und Commit, anzuzeigen IDs, wählen Sie im Navigationsbereich Commits und dann Commits vergleichen aus.

The screenshot shows the 'Compare' page in the AWS CodeCommit console for a repository named 'MyDemoRepo'. The breadcrumb navigation at the top reads: 'Developer Tools > CodeCommit > Repositories > MyDemoRepo > Compare'. Below the repository name, there are three tabs: 'Commits', 'Commit visualizer', and 'Compare commits', with the last one being active. The 'Destination' dropdown is set to 'AnotherBranch' and the 'Source' dropdown is set to '6b65eb76'. There are 'Compare' and 'Cancel' buttons. Below this, there's a 'Page 1 of 1' indicator, a 'Go to file' dropdown, and toggle options for 'Hide whitespace changes' (checked), 'Unified' (selected), and 'Split'. The main content area displays a diff for the file 'ahs_count.py'. It shows a comparison between two versions of the file. Line 8 has a red background with a minus sign and the text '- print(ahs.format(total))'. Line 8 also has a green background with a plus sign and the text '+ print(alv.format(total))'. Line 10 shows a comment: '#When using this script, make sure that you ask the subject to use one of the provided texts, such as bumblebee.txt.'. At the bottom, there's a section for 'anothernew/dir2/anothertest.txt' with the status 'Added'. There are 'Browse file contents' and 'Comment on file' buttons for each file.

Developer Tools > CodeCommit > Repositories > MyDemoRepo > Compare

MyDemoRepo

Commits | Commit visualizer | **Compare commits**

Destination: AnotherBranch Source: 6b65eb76 **Compare** Cancel

< Page 1 of 1 > Go to file Hide whitespace changes Unified Split

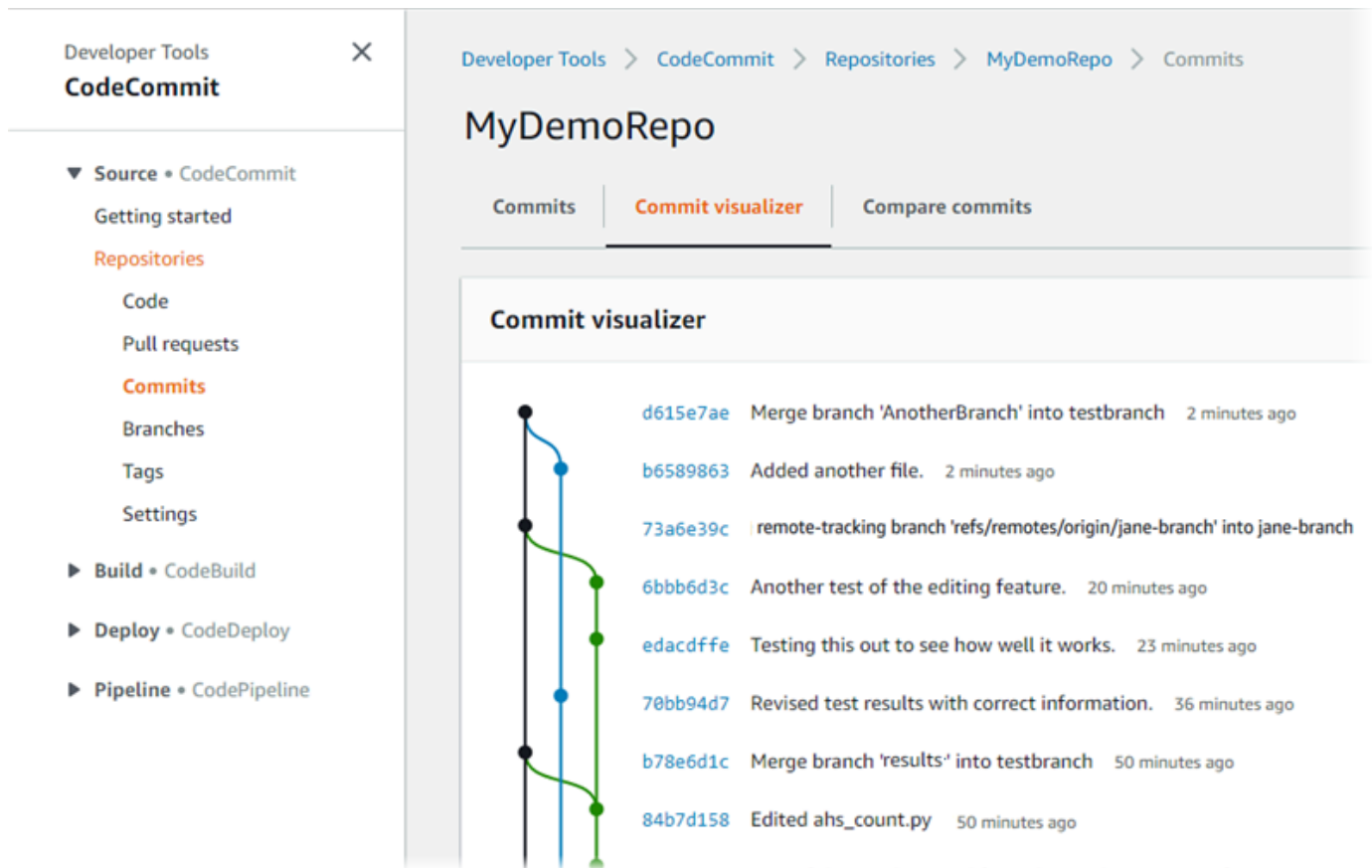
ahs_count.py Browse file contents Comment on file

```
***      ***      @@ -5,6 +5,6 @@
5      5
6      6      total = (ess + z)
7      7      ahs = "Number of alveolar hissing sibilants: {}"
8      - print(ahs.format(total))
8      + print(alv.format(total))
9      9
10     10     #When using this script, make sure that you ask the subject to use one of the provided texts, such as bumblebee.txt.
```

anothernew/dir2/anothertest.txt Added Browse file contents Comment on file

Weitere Informationen erhalten Sie unter [Den Commit-Verlauf eines Repositorys durchsuchen](#) und [Vergleichen von Commits](#).

- Wählen Sie unter Commits die Commit visualizer-Registerkarte aus.



Die Commit-Grafik wird angezeigt. Neben dem jeweiligen Punkt wird die Betreffzeile der einzelnen Commits in der Grafik aufgeführt. Die Anzeige der Betreffzeile ist auf 80 Zeichen beschränkt.

- Um weitere Details zu einem Commit anzuzeigen, wählen Sie die abgekürzte Commit-ID aus. Um das Diagramm eines bestimmten Commits zu rendern, wählen Sie diesen Punkt im Diagramm aus. Weitere Informationen finden Sie unter [Sehen Sie sich ein Diagramm des Commit-Verlaufs eines Repositories an](#).

Schritt 4: Einen Pull-Request erstellen und gemeinsam daran arbeiten

Wenn Sie mit anderen Benutzern in einem Repository arbeiten, können Sie Code gemeinsam bearbeiten und Änderungen prüfen. Sie können eine Pull-Anforderung erstellen, damit andere Benutzer die Codeänderungen in einem Branch prüfen und kommentieren können. Sie können auch einzelne oder mehrere Genehmigungsregeln für die Pull-Anforderung erstellen. Sie können beispielsweise eine Genehmigungsregel erstellen, die voraussetzt, dass mindestens zwei andere Benutzer die Pull-Anforderung genehmigen, bevor die Zusammenführung erfolgen kann. Nachdem die Pull-Anforderung genehmigt wurde, können Sie diese Änderungen im Ziel-Branch

zusammenführen. Wenn Sie Benachrichtigungen für Ihr Repository einrichten, können Repository-Benutzer E-Mails zu Repository-Ereignissen erhalten, z. B. für Pull-Anforderungen oder wenn jemand Kommentare zu Code hinterlässt. Weitere Informationen finden Sie unter [Benachrichtigungen für Ereignisse in einem Repository konfigurieren AWS CodeCommit](#).

 Important

Bevor Sie eine Pull-Anforderung erstellen können, müssen Sie einen Branch erstellen, der die Codeänderungen enthält, die Sie überprüfen möchten. Weitere Informationen finden Sie unter [Erstellen eines Zweigs](#).

So erstellen Sie eine Pull-Anforderung und arbeiten gemeinsam daran

1. Wählen Sie im Navigationsbereich Pull requests (Pull-Anforderungen) aus.
2. Wählen Sie unter Pull request (Pull-Anforderung) die Option Create pull request (Pull-Anforderung erstellen) aus.

 Tip

Darüber hinaus können Sie Pull-Anforderungen über Branches und Code erstellen.

Wählen Sie in Create pull request unter Source den Branch aus, der die zu prüfenden Änderungen enthält. Wählen Sie unter Destination (Ziel) den Branch aus, in dem der geprüfte Code nach dem Schließen der Pull-Anforderung zusammengeführt werden soll. Wählen Sie Compare aus.

3. Prüfen Sie die Details der Zusammenführung und die Änderungen, um sicherzustellen, dass die Pull-Anforderung die Änderungen und Commits enthält, die geprüft werden sollen. Wenn dies der Fall ist, geben Sie unter Title (Titel) einen Titel für diese Prüfung an. Dieser Titel wird in der Liste der Pull-Anforderungen für das Repository angezeigt. Unter Description (Beschreibung) können Sie eingeben, wofür diese Prüfung vorgesehen ist, sowie weitere nützliche Informationen für Prüfer hinterlassen. Wählen Sie Create (Erstellen) aus.

4. Ihre Pull-Anforderung wird in der Liste der Pull-Anforderungen für das Repository angezeigt. Sie können die Ansicht filtern, sodass nur offene Anforderungen, geschlossene Anforderungen, von Ihnen erstellte Anforderungen usw. angezeigt werden.

Pull requests [Info](#)

Open pull requests [Create pull request](#)

1

Pull request	Author	Destination	Last activity	Status	Approval status
31: testing this	Saamvi_Sarkar	preprod	4 minutes ago	Open	No approval rules
25: Updated some of our Java samples	Li_Juan	main	5 minutes ago	Open	0 of 1 rules satisfied
29: Changing duplicate value	Li_Juan	main	19 minutes ago	Open	0 of 1 rules satisfied
22: Test pull request	Saamvi_Sarkar	preprod	20 minutes ago	Open	No approval rules
28: Changes to some of our code samples	Li_Juan	main	1 month ago	Open	0 of 1 rules satisfied
20: A bugfix to add missing examples for S3	Saamvi_Sarkar	main	1 month ago	Open	0 of 1 rules satisfied

5. Du kannst deinem Pull Request eine Genehmigungsregel hinzufügen, um sicherzustellen, dass bestimmte Bedingungen erfüllt sind, bevor er zusammengeführt werden kann. Um der Pull-Anforderung eine Genehmigungsregel hinzuzufügen, wählen Sie die Pull-Anforderung in der Liste aus. Wählen Sie auf der Registerkarte Approvals (Genehmigungen) die Option Create approval rule (Genehmigungsregel erstellen) aus.

6. Geben Sie in das Feld Rule name (Regelname) einen beschreibenden Namen für die Regel ein. Wenn Sie z. B. möchten, dass eine Pull-Anforderung von zwei Personen genehmigt werden muss, bevor sie zusammengeführt werden kann, können Sie die Regel beispielsweise **Require two approvals before merge** nennen. Geben Sie in das Feld Number of approvals needed (Anzahl erforderlicher Genehmigungen) den gewünschten Wert (**2**) ein. Der Standardwert ist 1. Wählen Sie Absenden aus. Weitere Informationen zu Genehmigungsregeln und Genehmigungs-Pool-Mitgliedern finden Sie unter [Erstellen Sie eine Genehmigungsregel für eine Pull-Anfrage](#).

Create approval rule

Rule details

Rule name
Require two approvals before merge

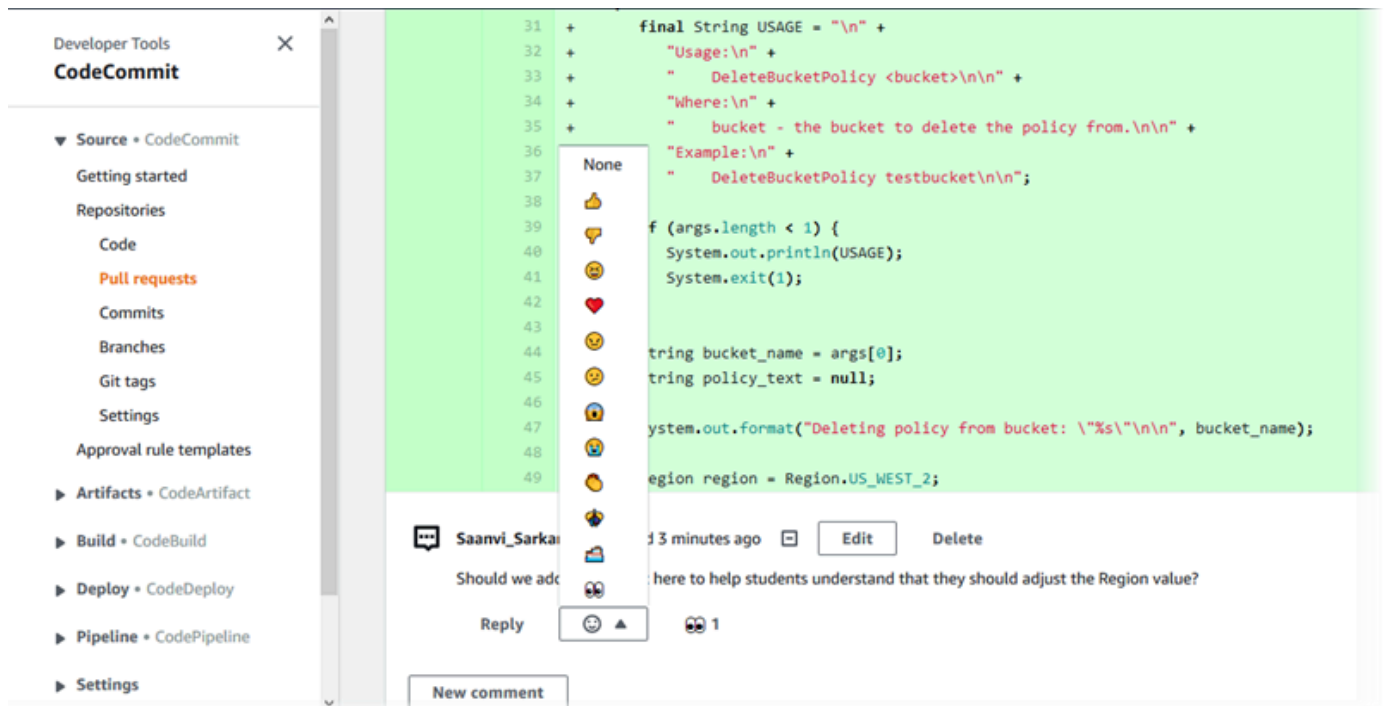
Number of approvals needed
2

Approval pool members - optional
If approval pool members are specified, only approvals from these members will count toward satisfying this rule. Use a wildcard to match multiple approvers with one value.

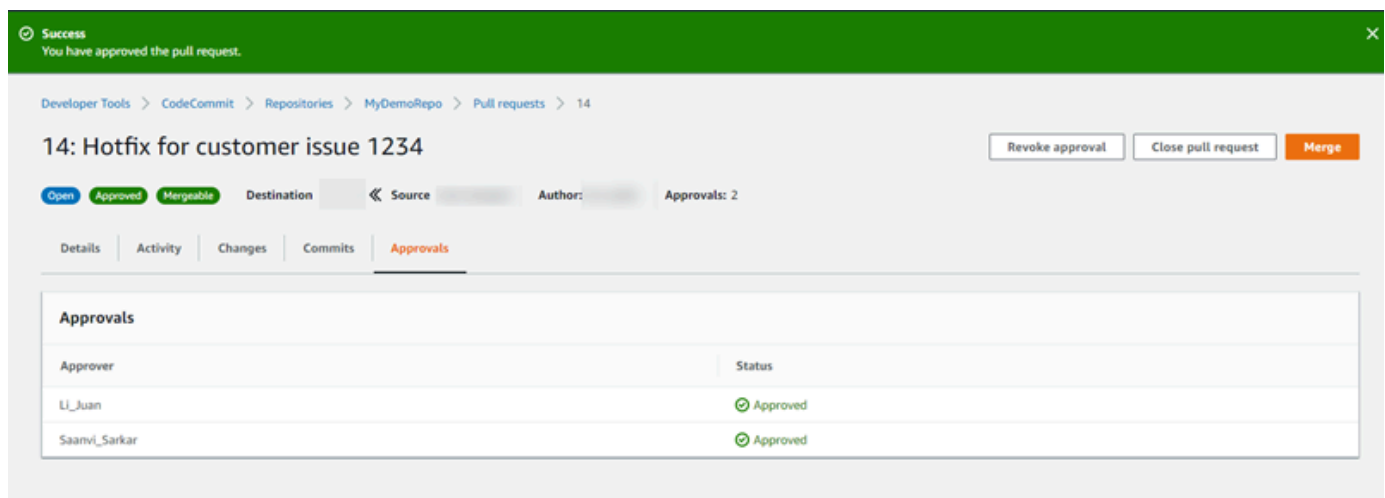
Add

Cancel Submit

7. Wenn Sie Benachrichtigungen für Ihr Repository konfiguriert haben und Benutzer über Pull-Anforderungsereignisse benachrichtigen wollen, erhalten die Benutzer eine E-Mail über Ihre neue Pull-Anforderung. Benutzer können die Änderungen für bestimmte Codezeilen, Dateien und die Pull-Anforderung selbst sehen. Sie können auch auf Kommentare mit Text und Emojis antworten. Falls erforderlich, können Sie Änderungen an den Pull-Anforderung-Branch senden, der die Pull-Anforderung aktualisiert.



8. Wenn Sie mit den Änderungen in der Anforderung zufrieden sind, wählen Sie Approve (Genehmigen). Sie können eine Pull-Anforderung auch dann genehmigen, wenn für diese Pull-Anforderung keine Genehmigungsregeln konfiguriert sind. Dies belegt, dass Sie die Pull-Anforderung geprüft haben und die Änderungen genehmigen. Sie können Ihre Genehmigung auch widerrufen, wenn Sie Ihre Meinung ändern.



Note

Sie können eine Pull-Anforderung nicht genehmigen, wenn Sie sie erstellt haben.

9. Wenn alle Codeänderungen geprüft wurden und Sie mit diesen Änderungen einverstanden sind, führen Sie von der Pull-Anforderung einen der folgenden Schritte aus:
- Wenn Sie die Pull-Anforderung schließen möchten, ohne Branches zusammenzuführen, wählen Sie Close pull request (Pull-Anforderung schließen).
 - Wenn Sie die Branches zusammenführen und die Pull-Anforderung schließen möchten, wählen Sie Merge (Zusammenführen) aus. Sie können unter den für Ihren Code verfügbaren Zusammenführungsstrategien wählen. Dabei spielen die Unterschiede zwischen den Quell- und Ziel-Banches eine Rolle und ob der Quell-Branch automatisch gelöscht werden soll, sobald die Zusammenführung abgeschlossen ist. Wenn Sie Ihre Auswahl getroffen haben, wählen Sie Merge pull request (Pull-Anforderung zusammenführen) aus, um die Zusammenführung abzuschließen.

Merge pull request 9: Bug fix for unhandled exception

Merge request details

Pull request: #9 Bug fix for unhandled exception

Destination main **Source** bugfix-bug1234

Merge strategy [Info](#)
Determines the way in which the current pull request will be merged into the destination branch

☐ **Fast forward merge**
`git merge --ff-only`
Merges the branches and moves the destination branch pointer to the tip of the source branch. This is the default merge strategy in Git.


☒ **Squash and merge**
`git merge --squash`
Combine all commits from the source branch into a single merge commit in the destination branch.


☐ **3-way merge**
`git merge --no-ff`
Create a merge commit and adds individual source commits to the destination branch.


Commit message - optional Preview markdown

Squashed commit of the following

commit d49940ad
Author: Li Juan <li_juan@example.com>
Date: Tue May 07 2019 15:12:48 GMT-0700 (Pacific Daylight Time)

Fixing the bug reported in 1234.

Author name

Email address

☒ Delete source branch bugfix-bug1234 after merging?

Cancel Merge pull request

- Wenn es in den Branches Merge-Konflikte gibt, die nicht automatisch gelöst werden können, kannst du sie in der CodeCommit Konsole lösen, oder du kannst deinen lokalen Git-Client verwenden, um die Branches zusammenzuführen und dann die Zusammenführung zu pushen. Weitere Informationen finden Sie unter [Konflikte in einer Pull-Anfrage in einem AWS CodeCommit Repository lösen](#).

 Note

Sie können Branches jederzeit manuell zusammenführen, einschließlich von Pull-Anforderungs-Banches, indem Sie den Befehl `git mergein` Ihrem lokalen Repository ausführen und Ihre Änderungen per Push senden.

Weitere Informationen erhalten Sie unter [Verwenden von Pull-Anforderungen](#) und [Arbeiten mit Genehmigungsregelvorlagen](#).

Schritt 5: Bereinigen

Wenn Sie das CodeCommit Repository nicht mehr benötigen, sollten Sie das CodeCommit Repository und andere Ressourcen, die Sie in dieser Übung verwendet haben, löschen, damit Ihnen der Speicherplatz nicht weiter in Rechnung gestellt wird.

 Important

Diese Aktion kann nicht rückgängig gemacht werden. Nachdem Sie dieses Repository gelöscht haben, können Sie es nicht mehr in ein lokales oder gemeinsam genutztes Repository klonen. Sie können auch keine Daten mehr aus einem lokalen oder gemeinsam genutzten Repository abrufen oder Daten dorthin übertragen oder Git-Operationen ausführen. Wenn Sie Benachrichtigungen für Ihr Repository konfiguriert haben, wird beim Löschen des Repositories auch die für das Repository erstellte Amazon CloudWatch Events-Regel gelöscht. Das Amazon SNS SNS-Thema, das als Ziel für diese Regel verwendet wurde, wird nicht gelöscht.

Wenn Sie Trigger für Ihr Repository konfiguriert haben, werden beim Löschen des Repositories nicht die Amazon SNS SNS-Themen oder Lambda-Funktionen gelöscht, die Sie als Ziele dieser Trigger konfiguriert haben. Achten Sie darauf, diese Ressourcen zu löschen, wenn sie nicht benötigt werden. Weitere Informationen finden Sie unter [Trigger aus einem Repository löschen](#).

Um das Repository zu löschen CodeCommit

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.

2. Wählen Sie unter Repositories (Repositoryys) das Repositorys aus, das Sie löschen möchten. Wenn Sie die Namenskonvention in diesem Thema befolgt haben, hat es den Namen MyDemoRepo erhalten.
3. Wählen Sie im Navigationsbereich Settings (Einstellungen).
4. Klicken Sie auf der Seite Settings unter Delete repository auf Delete repository.
5. Geben Sie **delete** ein und wählen Sie dann Delete (Löschen) aus. Das Repository wird dauerhaft gelöscht.

Schritt 6: Nächste Schritte

Nachdem Sie sich mit CodeCommit einigen Funktionen vertraut gemacht haben, sollten Sie Folgendes in Betracht ziehen:

- Wenn du neu bei Git bist und/oder CodeCommit Beispiele für die Verwendung von Git mit lesen möchtest CodeCommit, fahre mit dem [Erste Schritte mit Git und CodeCommit](#) Tutorial fort.
- Wenn Sie mit anderen in einem CodeCommit Repository arbeiten möchten, finden Sie weitere Informationen unter [Teilen Sie ein Repository](#). (Wenn Sie Ihr Repository mit Benutzern in einem anderen Amazon Web Services Services-Konto teilen möchten, finden Sie weitere Informationen unter [Konfigurieren den kontoübergreifenden Zugriff auf ein AWS CodeCommit Repository mithilfe von Rollen](#).)
- Wenn Sie ein Repository zu migrieren möchten CodeCommit, folgen Sie den Schritten unter [Migrieren Sie zu CodeCommit](#).
- Wenn Sie das Repository einer kontinuierlichen Bereitstellungs-Pipeline hinzufügen möchten, führen Sie die schrittweise [Anleitung zum Erstellen einer einfachen Pipeline](#) aus.
- Wenn Sie sich für Produkte und Services interessieren, die in CodeCommit integriert werden können, sowie für Beispiele aus der Community, lesen Sie [Produkt- und Service-Integrationen](#).

Erste Schritte mit Git und AWS CodeCommit

Wenn du neu bei Git und bist CodeCommit, hilft dir dieses Tutorial dabei, einige einfache Befehle zu erlernen, um dir den Einstieg zu erleichtern. Wenn Sie bereits mit Git vertraut sind, können Sie dieses Tutorial überspringen und mit [Erste Schritte mit CodeCommit](#) fortfahren.

In diesem Tutorial erstellst du ein Repository, das eine lokale Kopie des CodeCommit Repositorys darstellt, das wir als lokales Repo bezeichnen.

Nachdem Sie das lokale Repo erstellt haben, nehmen Sie einige Änderungen daran vor. Dann senden (pushen) Sie Ihre Änderungen an das CodeCommit Repository.

Sie simulieren auch eine Teamumgebung, in der zwei Benutzer unabhängig voneinander Änderungen in ihr lokales Repository übernehmen und diese Änderungen in das CodeCommit Repository übertragen. Die Benutzer ziehen dann die Änderungen aus dem CodeCommit Repository in ihr eigenes lokales Repository, um die Änderungen zu sehen, die der andere Benutzer vorgenommen hat.

Außerdem erstellen Sie Branches und Tags und verwalten einige Zugriffsberechtigungen im CodeCommit Repository.

Wenn Sie dieses Tutorial abgeschlossen haben, sollten Sie ausreichend Übung im Umgang mit den grundlegenden Git- und CodeCommit -Konzepten haben, um sie für Ihre eigenen Projekte anzuwenden.

Sorgen Sie dafür, dass die [Voraussetzungen erfüllt sind und die Einrichtung abgeschlossen ist](#), einschließlich:

- Weisen Sie dem IAM-Benutzer Berechtigungen zu.
- Richten Sie CodeCommit die Verbindung zu einem Repository über [HTTPS](#), SSH oder ein. [git-remote-codecommit](#) Weitere Informationen zu diesen Optionen finden Sie unter [Einrichtung für AWS CodeCommit](#).
- Konfigurieren Sie, AWS CLI ob Sie die Befehlszeile oder das Terminal für alle Operationen verwenden möchten, einschließlich der Erstellung des Repositorys.

Themen

- [Schritt 1: Erstellen Sie ein CodeCommit Repository](#)
- [Schritt 2: Erstellen Sie ein lokales Repo](#)
- [Schritt 3: Erstelle deinen ersten Commit](#)
- [Schritt 4: Push deinen ersten Commit](#)
- [Schritt 5: Teile das CodeCommit Repository und führe einen weiteren Commit per Push und Pull durch](#)
- [Schritt 6: Einen Branch erstellen und teilen](#)
- [Schritt 7: Erstelle und teile ein Tag](#)
- [Schritt 8: Zugriffsberechtigungen einrichten](#)

- [Schritt 9: Bereinigen](#)

Schritt 1: Erstellen Sie ein CodeCommit Repository

In diesem Schritt verwenden Sie die CodeCommit Konsole, um das Repository zu erstellen.

Sie können diesen Schritt überspringen, wenn Sie bereits über ein CodeCommit Repository verfügen, das Sie verwenden möchten.

Note

Abhängig von Ihrer Nutzung können Ihnen Gebühren für die Erstellung eines Repositorys oder den Zugriff auf ein Repository in Rechnung gestellt werden. Weitere Informationen finden Sie auf der CodeCommit Produktinformationsseite unter [Preise](#).

Um das CodeCommit Repository zu erstellen

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Verwenden Sie die Regionsauswahl, um den Ort auszuwählen, AWS-Region an dem Sie das Repository erstellen möchten. Weitere Informationen finden Sie unter [Regionen und Git-Verbindungsendpunkte](#).
3. Wählen Sie auf der Seite Repositorys die Option Repository auswählen.
4. Geben Sie auf der Seite Create repository (Repository erstellen) unter Repository name (Repository-Name) einen Namen für Ihr Repository ein (z. B. **MyDemoRepo**).

Note

Bei Repository-Namen muss die Groß- und Kleinschreibung beachtet werden. Er darf nicht mehr als 100 Zeichen enthalten. Weitere Informationen finden Sie unter [Limits](#).

5. (Optional) Geben Sie unter Description (Beschreibung) eine Beschreibung ein (z. B. **My demonstration repository**). Dadurch können Sie und andere Benutzer den Zweck des Repositorys leichter erkennen.
6. (Optional) Wählen Sie Tag hinzufügen, um Ihrem Repository ein oder mehrere Repository-Tags (eine benutzerdefinierte Attributbezeichnung, mit der Sie Ihre AWS Ressourcen organisieren

und verwalten können) hinzuzufügen. Weitere Informationen finden Sie unter [Kennzeichen von Repositorys in AWS CodeCommit](#).

7. (Optional) Erweitern Sie Zusätzliche Konfiguration, um anzugeben, ob Sie den Standardschlüssel Von AWS verwalteter Schlüssel oder Ihren eigenen, vom Kunden verwalteten Schlüssel zum Verschlüsseln und Entschlüsseln der Daten in diesem Repository verwenden möchten. Wenn Sie Ihren eigenen, vom Kunden verwalteten Schlüssel verwenden möchten, müssen Sie sicherstellen, dass er dort verfügbar ist, AWS-Region wo Sie das Repository erstellen, und dass der Schlüssel aktiv ist. Weitere Informationen finden Sie unter [AWS Key Management Service und Verschlüsselung für AWS CodeCommit Repositorien](#).
8. (Optional) Wählen Sie Amazon CodeGuru Reviewer für Java und Python aktivieren, wenn dieses Repository Java- oder Python-Code enthält und Sie möchten, dass CodeGuru Reviewer diesen Code analysiert. CodeGuru Reviewer verwendet mehrere Modelle für maschinelles Lernen, um Codefehler zu finden und automatisch Verbesserungen und Korrekturen in Pull-Requests vorzuschlagen. Weitere Informationen finden Sie im Amazon CodeGuru Reviewer-Benutzerhandbuch.
9. Wählen Sie Create (Erstellen) aus.

Note

Die verbleibenden Schritte in diesem Tutorial werden MyDemoRepo für den Namen des CodeCommit Repositorys verwendet. Wenn Sie einen anderen Namen auswählen, müssen Sie diesen im gesamten Tutorial verwenden.

Weitere Informationen zum Erstellen von Repositorys, darunter auch die Erstellung eines Repositorys vom Terminal oder über die Befehlszeile, finden Sie unter [Erstellen eines -Repositorys](#).

Schritt 2: Erstellen Sie ein lokales Repo

In diesem Schritt richten Sie ein lokales Repository auf Ihrem lokalen Computer ein, um eine Verbindung zu Ihrem Repository herzustellen. Dazu wählen Sie ein Verzeichnis auf Ihrem lokalen Computer aus, das das lokale Repository darstellt. Sie verwenden Git, um eine Kopie Ihres leeren CodeCommit Repositorys in diesem Verzeichnis zu klonen und zu initialisieren. Anschließend geben Sie den Git-Benutzernamen und die E-Mail-Adresse an, die Sie zum Kommentieren Ihrer Commits verwendet haben.

1. [Öffne die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home](https://console.aws.amazon.com/codesuite/codecommit/home).
2. Wählen Sie in der Regionsauswahl den Ort aus, AWS-Region an dem das Repository erstellt wurde. Repositorys sind spezifisch für ein. AWS-Region Weitere Informationen finden Sie unter [Regionen und Git-Verbindungsendpunkte](#).
3. Suchen Sie das Repository, zu dem Sie eine Verbindung herstellen möchten, in der Liste und wählen Sie es aus. Wählen Sie Clone URL (URL klonen) und dann das Protokoll aus, das Sie beim Klonen oder bei der Verbindung zu dem Repository verwenden möchten. Dadurch wird die Klon-URL kopiert.
 - Kopieren Sie die HTTPS-URL, wenn Sie entweder Git-Anmeldeinformationen mit Ihrem IAM-Benutzer oder den Credential Helper verwenden, der im Lieferumfang von enthalten ist. AWS CLI
 - Kopieren Sie die HTTPS-URL (GRC), wenn Sie den Befehl git-remote-codecommit auf Ihrem lokalen Computer verwenden.
 - Kopieren Sie die SSH-URL, wenn Sie ein öffentliches/privates SSH-Schlüsselpaar mit Ihrem IAM-Benutzer verwenden.

 Note

Wenn Sie statt einer Liste von Repositorys eine Willkommensseite sehen, sind Ihrem AWS Konto in dem Land, in dem Sie angemeldet sind, keine Repositorys zugeordnet. AWS-Region Informationen zur Erstellung eines Repositorys finden Sie unter [the section called “Erstellen eines -Repositorys”](#), oder befolgen Sie die Schritte im Tutorial [Erste Schritte mit Git und CodeCommit](#).

4. (Optional) Wir empfehlen, dass du deinen lokalen Git-Client so konfigurierst, dass er **main** als Namen für den Standard-Branch für dein Repository verwendet wird. Dies ist der Name, der in allen Beispielen in diesem Handbuch für den Standard-Branch verwendet wird. Es ist auch derselbe Standard-Branch-Name, den Sie CodeCommit verwenden, wenn Sie Ihren ersten Commit in der Konsole durchführen. Führen Sie den folgenden Befehl aus, um den Standard-Branch-Namen global für Ihr System zu konfigurieren:

```
git config --global init.defaultBranch main
```

Wenn Sie es vorziehen, einen anderen Standard-Branch-Namen für all Ihre Repositorys zu verwenden, **main** ersetzen Sie ihn durch Ihren bevorzugten Namen. In diesem Tutorial wird davon ausgegangen, dass Ihr Standard-Branch den Namen **main** trägt.

Wenn Sie unterschiedliche Standard-Branch-Namen für verschiedene Repositorys verwenden möchten, können Sie dieses Attribut lokal (`--local`) statt global (`--global`) setzen.

5. Klonen Sie das Repository am Terminal oder in der Befehlszeile mit dem `git clone` Befehl und geben Sie die Klon-URL ein, die Sie in Schritt 3 kopiert haben. Ihre Klon-URL hängt davon ab, welches Protokoll und welche Konfiguration Sie verwenden. Wenn Sie beispielsweise HTTPS mit Git-Anmeldeinformationen verwenden, um ein Repository zu klonen, das **MyDemoRepo** in der Region USA Ost (Ohio) benannt ist:

```
git clone https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
```

Wenn Sie HTTPS mit `git-remote-codecommit` verwenden:

```
git clone codecommit://MyDemoRepo my-demo-repo
```

Bei Verwendung von SSH:

```
git clone ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
```

Note

Wenn bei dem Versuch, ein Repository zu klonen, ein Fehler angezeigt wird, haben Sie möglicherweise die Einrichtung nicht abgeschlossen, die für Ihren lokalen Computer erforderlich ist. Weitere Informationen finden Sie unter [Einrichtung für AWS CodeCommit](#).

Schritt 3: Erstelle deinen ersten Commit

In diesem Schritt erstellen Sie Ihren ersten Commit in Ihrem lokalen Repo. Dazu erstellen Sie zwei Beispieldateien in Ihrem lokalen Repo. Sie verwenden Git, um die Änderung in Ihrem lokalen Repository bereitzustellen und die Änderung dann in Ihr lokales Repository zu übertragen.

1. Erstellen Sie mit einem Texteditor die beiden folgenden Beispieltextdateien in Ihrem Verzeichnis. Geben Sie den Dateien die Namen `cat.txt` und `dog.txt`:

```
cat.txt
-----
The domestic cat (Felis catus or Felis silvestris catus) is a small, usually furry,
domesticated, and carnivorous mammal.
```

```
dog.txt
-----
The domestic dog (Canis lupus familiaris) is a canid that is known as man's best
friend.
```

2. Führen Sie `git config` aus, um Ihren Benutzernamen und Ihre E-Mail-Adresse, dargestellt durch Platzhalter `your-user-name`, `your-email-address` zu Ihrem lokalen Repository hinzuzufügen. Dies vereinfacht die Identifizierung der erstellten Commits:

```
git config --local user.name "your-user-name"
git config --local user.email your-email-address
```

3. Wenn Sie Ihren Standard-Branch-Namen nicht global festgelegt haben, als Sie das lokale Repository erstellt haben, führen Sie den folgenden Befehl aus, um den Standard-Branch-Namen auf festzulegen: **main**

```
git config --local init.defaultBranch main
```

4. Führen Sie `git add` aus, um die Änderung per Stage zu übertragen:

```
git add cat.txt dog.txt
```

5. Führen Sie `git commit` aus, um die Änderung festzuschreiben:

```
git commit -m "Added cat.txt and dog.txt"
```

 Tip

Um Details zu dem soeben durchgeführten Commit anzuzeigen, führen Sie `git log` aus.

Schritt 4: Push deinen ersten Commit

In diesem Schritt pushen Sie den Commit von Ihrem lokalen Repository in Ihr CodeCommit Repository.

Führen Sie den Befehl `git push` aus, um Ihren Commit über den Standard-Remote-Namen, den Git für Ihr CodeCommit Repository (`origin`) verwendet, aus dem Standard-Branch in Ihrem lokalen Repo (`main`) zu pushen:

```
git push -u origin main
```

 Tip

Nachdem du Dateien in dein CodeCommit Repository gepusht hast, kannst du die CodeCommit Konsole verwenden, um den Inhalt anzusehen. Weitere Informationen finden Sie unter [Dateien in einem Repository durchsuchen](#).

Schritt 5: Teile das CodeCommit Repository und führe einen weiteren Commit per Push und Pull durch

In diesem Schritt teilst du Informationen über das CodeCommit Repository mit einem anderen Teammitglied. Das Teammitglied verwendet diese Informationen, um eine lokale Kopie zu erstellen, einige Änderungen daran vorzunehmen und dann die geänderte lokale Kopie in Ihr CodeCommit Repository zu übertragen. Anschließend ziehen Sie die Änderungen aus dem CodeCommit Repository in Ihr lokales Repository.

In diesem Tutorial wird der zweite Benutzer simuliert. Dazu lassen Sie Git zusätzlich zu dem in [Schritt 2](#) erstellten Verzeichnis ein weiteres Verzeichnis erstellen. (In der Regel befindet sich dieses Verzeichnis auf einem anderen Computer.) Dieses neue Verzeichnis ist eine Kopie Ihres CodeCommit Repositorys. Alle Änderungen, die Sie an dem vorhandenen oder diesem neuen Verzeichnis vornehmen, werden unabhängig voneinander vorgenommen. Die einzige Möglichkeit,

Änderungen an diesen Verzeichnissen zu erkennen, besteht darin, sie aus dem CodeCommit Repository abzurufen.

Obwohl sie sich auf demselben lokalen Computer befinden, nennen wir das vorhandene Verzeichnis lokales Repository (local repo) und das neue Verzeichnis freigegebenes Repository (shared repo).

Aus dem neuen Verzeichnis erhalten Sie eine separate Kopie des CodeCommit Repositorys. Anschließend fügen Sie eine neue Beispieldatei hinzu, übertragen die Änderungen in das gemeinsam genutzte Repository und übertragen dann den Commit aus dem gemeinsam genutzten Repository in Ihr CodeCommit Repository.

Schließlich ziehen Sie die Änderungen aus Ihrem Repository in Ihr lokales Repo und durchsuchen es dann, um die Änderungen zu sehen, die vom anderen Benutzer übernommen wurden.

1. Wechseln Sie in das Verzeichnis /tmp oder in das Verzeichnis c:\temp.
2. Führen Sie git clone aus, um eine Pull-Übertragung einer Kopie des Repositorys in das freigegebene Repository (shared repo) durchzuführen:

Für HTTPS:

```
git clone https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo  
shared-demo-repo
```

Für HTTPS mit git-remote-codecommit:

```
git clone codecommit://MyDemoRepo shared-demo-repo
```

Für SSH:

```
git clone ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo shared-  
demo-repo
```

Note

Beim Klonen eines Repositorys mit SSH auf Windows-Betriebssystemen müssen Sie möglicherweise die SSH-Schlüssel-ID wie folgt der Verbindungszeichenfolge hinzufügen:

```
git clone ssh://Your-SSH-Key-ID@git-codecommit.us-east-2.amazonaws.com/v1/  
repos/MyDemoRepo my-demo-repo
```

Weitere Informationen finden Sie unter [Für SSH-Verbindungen unter Windows](#).

In diesem Befehl MyDemoRepo steht der Name Ihres CodeCommit Repositorys. shared-demo-repo ist der Name des Verzeichnisses, das Git in dem /tmp Verzeichnis oder dem c:\temp Verzeichnis erstellt. Wenn Git das Verzeichnis erstellt hat, überträgt Git eine Kopie Ihres Repositorys per Pull in das Verzeichnis shared-demo-repo.

3. Wechseln Sie in das Verzeichnis shared-demo-repo:

```
(For Linux, macOS, or Unix) cd /tmp/shared-demo-repo  
(For Windows) cd c:\temp\shared-demo-repo
```

4. Führen Sie git config den Befehl aus, um einen weiteren Benutzernamen und eine E-Mail-Adresse hinzuzufügen, die durch Platzhalter *other-user-name* und *other-email-address* dargestellt werden. Dadurch wird es einfacher, festzustellen, welche Commits von dem anderen Benutzer durchgeführt werden:

```
git config --local user.name "other-user-name"  
git config --local user.email other-email-address
```

5. Erstellen Sie mit einem Texteditor die folgende Beispieldatei im Verzeichnis shared-demo-repo. Benennen Sie die Datei horse.txt:

```
horse.txt  
-----  
The horse (Equus ferus caballus) is one of two extant subspecies of Equus ferus.
```

6. Führen Sie git add aus, um die Änderung per Stage an das freigegebene Repository (shared repo) zu übertragen:

```
git add horse.txt
```

7. Führen Sie git commit aus, um die Änderung im dem freigegebenen Repository (shared repo) festzuschreiben:

```
git commit -m "Added horse.txt"
```

8. Führen Sie `git push` aus, um Ihren ersten Commit über den Standard-Remote-Namen, den Git für Ihr CodeCommit Repository (`origin`) verwendet, aus dem Standard-Branch in Ihrem lokalen Repo (`main`) zu pushen:

```
git push -u origin main
```

9. Wechsle zu deinem lokalen Repo und führe den Befehl `git pull` aus, um den Commit, den das gemeinsame Repo für das Repository vorgenommen hat, in dein lokales Repo zu ziehen. CodeCommit Führen Sie anschließend `git log` aus, um den Commit anzeigen zu lassen, der von dem freigegebenen Repository (`shared repo`) initiiert wurde.

Schritt 6: Einen Branch erstellen und teilen

In diesem Schritt erstellen Sie einen Branch in Ihrem lokalen Repository, nehmen einige Änderungen vor und übertragen den Branch dann per Push in Ihr CodeCommit Repository. Anschließend ziehen Sie den Branch aus Ihrem CodeCommit Repository in das gemeinsam genutzte Repo.

Ein Branch ermöglicht Ihnen die unabhängige Entwicklung einer abweichenden Version der Repository-Inhalte (zum Beispiel, um an einer neuen Softwarefunktion zu arbeiten, ohne die Arbeit Ihrer Teammitglieder zu beeinträchtigen). Wenn diese Funktion stabil ist, können Sie den Branch mit einem stabileren Software-Branch zusammenführen.

Sie erstellen den Branch mit Git und verweisen damit auf den ersten Commit, den Sie durchgeführt haben. Sie verwenden Git, um den Branch in das CodeCommit Repository zu übertragen. Dann wechselst du zu deinem gemeinsamen Repo und verwendest Git, um den neuen Branch in dein gemeinsames lokales Repo zu ziehen und den Branch zu erkunden.

1. Führen Sie `git checkout` von Ihrem lokalen Repo aus den Befehl aus und geben Sie dabei den Namen des Branches (zum Beispiel `MyNewBranch`) und die ID des ersten Commits an, den Sie im lokalen Repository vorgenommen haben.

Wenn Sie die Commit-ID nicht kennen, führen Sie `git log` aus, um sie abzurufen. Stellen Sie sicher, dass in dem Commit Ihr Benutzername und Ihre E-Mail-Adresse und nicht die Daten des anderen Benutzers verwendet werden. Damit wird simuliert, dass `main` es sich um eine stabile Version des CodeCommit Projektarchivs handelt und der `MyNewBranch` Branch für ein neues, relativ instabiles Feature vorgesehen ist:

```
git checkout -b MyNewBranch commit-ID
```

2. Führen Sie den Befehl `git push`, um den neuen Branch vom lokalen Repository an das CodeCommit Repository zu senden:

```
git push origin MyNewBranch
```

3. Übertragen Sie nun den Branch per Pull an das freigegebene Repository (shared repo) und überprüfen Sie die Ergebnisse:
 1. Wechseln Sie in das Verzeichnis des freigegebenen Repositorys (shared-demo-repo).
 2. Rufen Sie den neuen Branch per Pull ab (`git fetch origin`).
 3. Überprüfen Sie, ob der Branch per Pull übertragen wurde (`git branch --all` zeigt eine Liste aller Branches für das Repository an).
 4. Umstellen auf den neuen Branch (`git checkout MyNewBranch`).
 5. Überprüfen Sie, ob Sie zum Branch `MyNewBranch` gewechselt haben, indem Sie `git status` oder `git branch` ausführen. Die Ausgabe zeigt Ihnen, in welchem Branch Sie sich befinden. In diesem Fall sollte es sich um `MyNewBranch` handeln.
 6. Lassen Sie die Liste der Commits in dem Branch anzeigen (`git log`).

Im Folgenden sind die aufzurufenden Git-Befehle aufgeführt:

```
git fetch origin
git branch --all
git checkout MyNewBranch
git branch or git status
git log
```

4. Wechseln Sie zurück zum Branch `main` und lassen Sie die Liste der zugehörigen Commits anzeigen. Die Git-Befehle sollten wie folgt aussehen:

```
git checkout main
git log
```

5. Wechseln Sie zu dem `main` Zweig in Ihrem lokalen Repo. Sie können `git status` oder `git branch` ausführen. Die Ausgabe zeigt Ihnen, in welchem Branch Sie sich befinden. In diesem Fall sollte es sich um `main` handeln. Die Git-Befehle sollten wie folgt aussehen:

```
git checkout main  
git branch or git status
```

Schritt 7: Erstelle und teile ein Tag

In diesem Schritt erstellen Sie zwei Tags in Ihrem lokalen Repository, verknüpfen die Tags mit Commits und übertragen die Tags dann in Ihr CodeCommit Repository. Anschließend ziehen Sie die Änderungen aus dem CodeCommit Repository in das gemeinsam genutzte Repo.

Ein Tag wird verwendet, um einem Commit (oder einem Branch oder sogar einem anderen Tag) einen für Menschen lesbaren Namen zuzuweisen. Dies könnte zum Beispiel vorkommen, wenn Sie einen Commit mit `v2.1` markieren möchten. Ein Commit, ein Branch oder ein Tag kann mit beliebig vielen Tags verknüpft sein, aber ein bestimmter Tag ist mit nur einem bestimmten Commit, Branch oder Tag verknüpft. In diesem Tutorial markieren Sie einen Commit mit `release` und einen weiteren mit `beta`.

Sie erstellen die Tags mit Git und verweisen mit dem Tag `release` auf den ersten Commit und mit dem Tag `beta` auf den vom anderen Benutzer durchgeführten Commit. Anschließend verwenden Sie Git, um die Tags in das CodeCommit Repository zu übertragen. Dann wechselst du zu deinem gemeinsamen Repo und verwendest Git, um die Tags in dein geteiltes lokales Repo zu ziehen und die Tags zu erkunden.

1. Führen `git tag` Sie von Ihrem lokalen Repo aus den Befehl aus und geben Sie dabei den Namen des neuen Tags (`release`) und die ID des ersten Commits an, den Sie im lokalen Repo vorgenommen haben.

Wenn Sie die Commit-ID nicht kennen, führen Sie `git log` aus, um sie abzurufen. Stellen Sie sicher, dass in dem Commit Ihr Benutzername und Ihre E-Mail-Adresse und nicht die Daten des anderen Benutzers verwendet werden. Dies dient dazu, zu simulieren, dass es sich bei Ihrem Commit um eine stabile Version des CodeCommit Repositorys handelt:

```
git tag release commit-ID
```

Führen Sie `git tag` erneut aus, um den Commit des anderen Benutzers mit dem Tag `beta` zu versehen. Dadurch wird simuliert, dass der Commit für eine neue, relativ instabile Funktion durchgeführt wird:

```
git tag beta commit-ID
```

2. Führen Sie `git push --tags`, um die Tags an das CodeCommit Repository zu senden.
3. Übertragen Sie nun die Tags per Pull an das freigegebene Repository (shared repo) und überprüfen Sie die Ergebnisse:
 1. Wechseln Sie in das Verzeichnis des freigegebenen Repositorys (shared-demo-repo).
 2. Rufen Sie die neuen Tags per Pull ab (`git fetch origin`).
 3. Überprüfen Sie, ob die Tags per Pull übertragen wurden (`git tag` zeigt eine Liste aller Tags für das Repository an).
 4. Zeigen Sie Informationen zu jedem Tag an (`git log release` und `git log beta`).

Im Folgenden sind die aufzurufenden Git-Befehle aufgeführt:

```
git fetch origin
git tag
git log release
git log beta
```

4. Probieren Sie das auch im lokalen Repo aus:

```
git log release
git log beta
```

Schritt 8: Zugriffsberechtigungen einrichten

In diesem Schritt erteilen Sie einem Benutzer die Erlaubnis, das gemeinsam genutzte Repo mit dem CodeCommit Repository zu synchronisieren. Dieser Schritt ist optional. Es wird Benutzern empfohlen, die mehr darüber erfahren möchten, wie der Zugriff auf CodeCommit Repositorys gesteuert werden kann, wenn Benutzer Git-Anmeldeinformationen verwenden oder SSH-Schlüsselpaare mit IAM-Benutzern für den Zugriff auf Repositorys verwendet werden. CodeCommit

Note

Wenn Sie Verbundzugriff, temporäre Anmeldeinformationen oder einen Web-Identitätsanbieter wie IAM Identity Center verwenden, richten Sie Benutzer, Zugriff und

Berechtigungen für Ihren Identitätsanbieter ein und verwenden Sie dann. `git-remote-codecommit` Weitere Informationen erhalten Sie unter [Einrichtungsschritte für HTTPS-Verbindungen AWS CodeCommit mit git-remote-codecommit](#) und [Verbindung zu AWS CodeCommit Repositorys mit rotierenden Anmeldeinformationen herstellen](#).

Dazu verwenden Sie die IAM-Konsole, um einen Benutzer zu erstellen, der standardmäßig nicht berechtigt ist, das gemeinsam genutzte Repository mit dem Repository zu synchronisieren. CodeCommit Sie können `git pull` ausführen, um dies zu überprüfen. Wenn der neue Benutzer keine Berechtigung zur Synchronisierung hat, funktioniert der Befehl nicht. Dann kehren Sie zur IAM-Konsole zurück und wenden eine Richtlinie an, die dem Benutzer die Verwendung ermöglicht. `git pull` Sie können wieder `git pull` ausführen, um dies zu überprüfen.

Dieser Schritt wurde unter der Annahme geschrieben, dass Sie berechtigt sind, IAM-Benutzer in Ihrem Amazon Web Services Services-Konto zu erstellen. Wenn Sie diese Berechtigungen nicht haben, können Sie diesen Schritt nicht ausführen. Fahren Sie mit [Schritt 9: Bereinigen](#) fort, um die Ressourcen zu bereinigen, die Sie für das Tutorial verwendet haben.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter. <https://console.aws.amazon.com/iam/>

Stellen Sie sicher, dass Sie sich mit demselben Benutzernamen und demselben Passwort anmelden, das Sie auch in [Einrichtung](#) verwendet haben.

2. Wählen Sie im Navigationsbereich Users und dann Create New Users aus.
3. Geben Sie im ersten Feld unter Enter User Names (Benutzernamen eingeben) einen Beispiel-Benutzernamen ein (zum Beispiel **JaneDoe-CodeCommit**). Aktivieren Sie das Kontrollkästchen Generate an access key for each user (Zugriffsschlüssel für jeden Benutzer generieren) und wählen Sie dann Create (Erstellen).
4. Wählen Sie die Option Show User Security Credentials. Notieren Sie sich die Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel oder wählen Sie Download Credentials.
5. Folgen Sie den Anweisungen unter [Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden](#), um die Anmeldeinformationen des IAM-Benutzers zu generieren und bereitzustellen.

Wenn Sie SSH verwenden möchten, befolgen Sie die Anweisungen in [SSH und Linux, macOS oder Unix: Richten Sie die öffentlichen und privaten Schlüssel für Git ein und CodeCommit](#) oder [Schritt 3: Richten Sie die öffentlichen und privaten Schlüssel für Git ein und CodeCommit](#), um für den Benutzer öffentliche und private Schlüssel einzurichten.

6. Führen Sie `git pull`. Die folgende Fehlermeldung sollte angezeigt werden:

Für HTTPS:

```
fatal: unable to access 'https://git-codecommit.us-east-2.amazonaws.com/v1/repos/repository-name': The requested URL returned error: 403.
```

Für SSH:

```
fatal: unable to access 'ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/repository-name': The requested URL returned error: 403.
```

Der Fehler tritt auf, weil der neue Benutzer nicht berechtigt ist, das gemeinsam genutzte Repository mit dem CodeCommit Repository zu synchronisieren.

7. Kehren Sie zur IAM-Konsole zurück. Wählen Sie im Navigationsbereich Richtlinien und dann Richtlinie erstellen. (Wenn die Schaltfläche Get Started (Erste Schritte) angezeigt wird, klicken Sie darauf und wählen Sie anschließend Create Policy (Richtlinie erstellen) aus.)
8. Klicken Sie neben Create Your Own Policy auf Select.
9. Geben Sie in das Feld Policy Name (Richtliniennamen) einen Namen ein (z. B. **CodeCommitAccess-GettingStarted**).
10. Geben Sie im Feld Richtliniendokument Folgendes ein, sodass ein IAM-Benutzer Daten aus jedem Repository abrufen kann, das dem IAM-Benutzer zugeordnet ist:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codecommit:GitPull"
      ],
      "Resource": "*"
    }
  ]
}
```


 Tip

Wenn Sie möchten, dass der IAM-Benutzer Commits in jedes Repository übertragen kann, das dem IAM-Benutzer zugeordnet ist, geben Sie stattdessen Folgendes ein:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codecommit:GitPull",
        "codecommit:GitPush"
      ],
      "Resource": "*"
    }
  ]
}
```

Informationen zu anderen CodeCommit Aktions- und Ressourcenberechtigungen, die Sie Benutzern gewähren können, finden Sie unter [Authentifizierung und Zugriffskontrolle für AWS CodeCommit](#)

11. Klicken Sie im Navigationsbereich auf Users (Benutzer).
12. Wählen Sie den Beispiel-Benutzernamen (zum Beispiel **JaneDoe-CodeCommit**), dem Sie die Richtlinie anfügen möchten.
13. Wählen Sie die Registerkarte Berechtigungen.
14. Klicken Sie unter Managed Policies auf Attach Policy.
15. Wählen Sie die **CodeCommitAccess-GettingStarted**-Richtlinie aus, die Sie eben erstellt haben, und dann Attach Policy (Richtlinie anfügen).
16. Führen Sie git pull. Nun sollte der Befehl funktionieren und eine Meldung des Typs Already up-to-date sollte angezeigt werden.
17. Wenn Sie HTTPS verwenden, wechseln Sie zu Ihren ursprünglichen Git-Anmeldeinformationen oder, falls Sie git-remote-codecommit verwenden, zu Ihrem üblichen Profil. Weitere Informationen finden Sie in den Anweisungen unter [Einrichtung für HTTPS-Benutzer mit Git-Anmeldeinformationen](#) oder [Einrichtungsschritte für HTTPS-Verbindungen AWS CodeCommit mit git-remote-codecommit](#).

Wenn Sie SSH verwenden, wechseln Sie zu Ihren ursprünglichen Schlüsseln. Weitere Informationen finden Sie unter [SSH und Linux, macOS oder Unix: Richten Sie die öffentlichen und privaten Schlüssel für Git ein und CodeCommit](#) oder [Schritt 3: Richten Sie die öffentlichen und privaten Schlüssel für Git ein und CodeCommit](#).

Sie haben das Ende dieses Tutorials erreicht.

Schritt 9: Bereinigen

In diesem Schritt löschen Sie das CodeCommit Repository, das Sie in diesem Tutorial verwendet haben, sodass Ihnen der Speicherplatz nicht weiter in Rechnung gestellt wird.

Sie entfernen auch das lokale Repository und das gemeinsam genutzte Repo auf Ihrem lokalen Computer, da sie nach dem Löschen des CodeCommit Repositorys nicht mehr benötigt werden.

Important

Nachdem Sie dieses Repository gelöscht haben, können Sie es nicht mehr in ein lokales Repo oder ein gemeinsam genutztes Repository klonen. Sie werden auch nicht in der Lage sein, Daten aus einem lokalen oder gemeinsam genutzten Repo daraus abzurufen oder Daten dorthin zu übertragen. Diese Aktion kann nicht rückgängig gemacht werden.

Um das CodeCommit Repository zu löschen (Konsole)

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie auf der Seite Dashboard in der Liste der Repositorys den Eintrag MyDemoRepo.
3. Wählen Sie im Navigationsbereich Settings (Einstellungen).
4. Klicken Sie auf der Seite Settings unter Delete repository auf Delete repository.
5. Geben Sie in das Feld neben Type the name of the repository to confirm deletion (Namen des Repositorys eingeben, um das Löschen zu bestätigen) die Zeichenfolge **MyDemoRepo** ein und klicken Sie dann auf Delete (Löschen).

Um das Repository zu löschen () CodeCommit AWS CLI

Führen Sie den Befehl [delete-repository](#) aus:

```
aws codecommit delete-repository --repository-name MyDemoRepo
```

Um das lokale Repo und das gemeinsame Repo zu löschen

Für Linux, macOS oder Unix:

```
cd /tmp  
rm -rf /tmp/my-demo-repo  
rm -rf /tmp/shared-demo-repo
```

Für Windows:

```
cd c:\temp  
rd /s /q c:\temp\my-demo-repo  
rd /s /q c:\temp\shared-demo-repo
```

Produkt- und Serviceintegrationen mit AWS CodeCommit

Standardmäßig CodeCommit ist es in eine Reihe von AWS Diensten integriert. Sie können es auch CodeCommit mit Produkten und Dienstleistungen außerhalb von verwenden AWS. Die folgenden Informationen können Ihnen die Konfiguration von CodeCommit zum Integrieren in die von Ihnen verwendeten Produkte und Services erleichtern.

Themen

- [Integration mit anderen AWS Diensten](#)
- [Integrationsbeispiele der Community](#)

Integration mit anderen AWS Diensten

CodeCommit ist in die folgenden AWS Dienste integriert:

AWS Amplify

[AWS Amplify](#) macht es einfach, skalierbare mobile Anwendungen zu erstellen, zu konfigurieren und zu implementieren, die von AWS. Amplify sorgt für eine nahtlose Bereitstellung und Verwaltung Ihres mobilen Backends und bietet ein einfaches Framework für die problemlose Integration Ihres Backends in Ihre iOS-, Android-, Web- und React Native-Frontends. Außerdem automatisiert Amplify den Anwendungsfreigabeprozess Ihres Frontends und Backends, sodass Sie Funktionen schneller bereitstellen können.

Sie können Ihr CodeCommit Repository in der Amplify-Konsole verbinden. Nachdem Sie die Amplify-Konsole autorisiert haben, ruft Amplify ein Zugriffstoken vom Repository-Anbieter ab, speichert das Token jedoch nicht auf den Servern. AWS Amplify greift auf Ihr Repository nur mit Bereitstellungsschlüsseln zu, die in einem bestimmten Repository installiert sind.

Weitere Informationen:

- [AWS Amplify Benutzerhandbuch](#)
- [Erste Schritte](#)

AWS Cloud9

[AWS Cloud9](#) enthält eine Sammlung von Tools, mit denen Sie Software kodieren, erstellen, ausführen, testen, debuggen und in der Cloud veröffentlichen können. Diese Sammlung von Tools wird als AWS Cloud9 integrierte Entwicklungsumgebung oder IDE bezeichnet.

Sie greifen über einen Webbrowser auf die AWS Cloud9 IDE zu. Die IDE bietet eine umfassende Codebearbeitung mit Unterstützung mehrerer Programmiersprachen und Laufzeit-Debugger sowie ein integriertes Terminal.

Weitere Informationen:

- [AWS Cloud9 Benutzerhandbuch](#)
- [AWS CodeCommit Beispiel für AWS Cloud9](#)
- [Integrieren AWS Cloud9 mit AWS CodeCommit](#)

AWS CloudFormation

[AWS CloudFormation](#) ist ein Service, der Sie bei der Modellierung und Einrichtung Ihrer AWS Ressourcen unterstützt, sodass Sie weniger Zeit mit der Verwaltung dieser Ressourcen verbringen und sich mehr auf Ihre Anwendungen konzentrieren können. Sie erstellen eine Vorlage, die Ressourcen, einschließlich eines CodeCommit Repositorys, beschreibt und die Bereitstellung und Konfiguration dieser Ressourcen für Sie AWS CloudFormation übernimmt.

Weitere Informationen:

- [AWS CodeCommit Seite mit den Ressourcen für das Repository](#)

AWS CloudTrail

[CloudTrail](#) erfasst AWS API-Aufrufe und zugehörige Ereignisse, die von oder im Namen eines Amazon Web Services Services-Kontos getätigt wurden, und übermittelt Protokolldateien an einen von Ihnen angegebenen Amazon S3 S3-Bucket. Sie können so konfigurieren CloudTrail, dass API-Aufrufe von der AWS CodeCommit Konsole, CodeCommit Befehle vom AWS CLI lokalen Git-Client und von der CodeCommit API erfasst werden.

Weitere Informationen:

- [AWS CodeCommit API-Aufrufe protokollieren mit AWS CloudTrail](#)

CloudWatch Amazon-Veranstaltungen

[CloudWatch Events](#) bietet einen Stream von Systemereignissen, die Änderungen an AWS Ressourcen beschreiben, nahezu in Echtzeit. Mithilfe einfacher Regeln, die Sie schnell einrichten können, können Sie Ereignisse zuordnen und sie an eine oder mehrere Zielfunktionen oder Streams weiterleiten. CloudWatch Events erkennt betriebliche Änderungen, sobald sie eintreten. CloudWatch Events reagiert auf diese betrieblichen Änderungen und ergreift bei Bedarf Maßnahmen, indem es Nachrichten sendet, um auf die Umgebung zu reagieren, Funktionen aktiviert, Änderungen vornimmt und Statusinformationen erfasst.

Sie können CloudWatch Ereignisse so konfigurieren, dass sie CodeCommit Repositorys überwachen und auf Repository-Ereignisse reagieren, indem Sie auf Streams, Funktionen, Aufgaben oder andere Prozesse in anderen AWS Services wie Amazon Simple Queue Service, Amazon Kinesis und vielen AWS Lambda mehr abzielen.

Weitere Informationen:

- [CloudWatch Benutzerleitfaden für Veranstaltungen](#)
- [AWS CodeCommit -Ereignisse](#)
- Blogbeitrag: [Serverlose AWS CodeCommit Workflows mit Amazon CloudWatch Events erstellen](#) und JGit

AWS CodeBuild

[CodeBuild](#) ist ein vollständig verwalteter Service für die Codeerstellung in der Cloud. Sie können damit Quellcode kompilieren, Einheiten tests ausführen und implementierbare Artefakte generieren. Sie können den zu erstellenden Quellcode und die Erstellungsspezifikation in einem CodeCommit-Repository speichern. Sie können es CodeBuild direkt mit CodeCommit verwenden oder Sie können beide integrieren. CodeBuild und CodeCommit in eine Continuous-Delivery-Pipeline mit CodePipeline integrieren.

Weitere Informationen:

- [Planen eines Builds](#)
- [Erstellen eines Build-Projekts](#)
- [Verwenden Sie CodePipeline with AWS CodeBuild , um Builds auszuführen](#)

CodeGuru Amazon-Rezensent

Amazon CodeGuru Reviewer ist ein automatisierter Code-Review-Service, der mithilfe von Programmanalyse und maschinellem Lernen häufig auftretende Probleme erkennt und Korrekturen in Ihrem Java- oder Python-Code empfiehlt. Sie können Repositories in Ihrem Amazon Web Services Services-Konto mit CodeGuru Reviewer verknüpfen. Wenn Sie dies tun, erstellt der CodeGuru Prüfer eine servicebezogene Rolle, die es dem CodeGuru Prüfer ermöglicht, den Code in allen Pull-Requests zu analysieren, die nach der Zuordnung erstellt wurden.

Weitere Informationen:

- [Ein AWS CodeCommit Repository mit Amazon CodeGuru Reviewer verknüpfen oder die Zuordnung aufheben](#)
- [Amazon CodeGuru Reviewer-Benutzerhandbuch](#)

AWS CodePipeline

[CodePipeline](#) ist ein kontinuierlicher Bereitstellungsservice, mit dem Sie die für die Freigabe Ihrer Software erforderlichen Schritte entwickeln, visualisieren und automatisieren können. Sie können so konfigurieren CodePipeline, dass ein CodeCommit Repository als Quellaktion in einer Pipeline verwendet wird, und das Erstellen, Testen und Bereitstellen Ihrer Änderungen automatisieren.

Weitere Informationen:

- [Einfache Pipeline-Komplettlösung mit und CodePipeline AWS CodeCommit](#)
- [Migrieren Sie zu Amazon CloudWatch Events Change Detection für Pipelines mit einem Repository CodeCommit](#)
- [Änderungserkennungsmethoden zum automatischen Starten von Pipelines](#)

AWS Elastic Beanstalk

[Elastic Beanstalk](#) ist ein Managed Service, der es einfach macht, Anwendungen in der AWS Cloud bereitzustellen und zu verwalten, ohne sich Gedanken über die Infrastruktur machen zu müssen, auf der diese Anwendung ausgeführt werden. Sie können die Elastic Beanstalk Beanstalk-Befehlszeilenschnittstelle (EB CLI) verwenden, um Ihre Anwendung direkt aus einem neuen oder bestehenden CodeCommit Repository bereitzustellen.

Weitere Informationen:

- [Verwenden der EB CLI mit AWS CodeCommit](#)
- [Verwenden eines vorhandenen Repositorys AWS CodeCommit](#)
- [eb codesource \(EB CLI-Befehl\)](#)

AWS Key Management Service

[AWS KMS](#) ist ein verwalteter Service, der das Erstellen und Kontrollieren der Schlüssel zum Verschlüsseln Ihrer Daten vereinfacht. Wird standardmäßig AWS KMS zum Verschlüsseln von Repositorys CodeCommit verwendet.

Weitere Informationen:

- [AWS KMS und Verschlüsselung](#)

AWS Lambda

Mit [Lambda](#) können Sie Code ausführen, ohne Server bereitzustellen oder zu verwalten. Sie können Trigger für CodeCommit Repositories konfigurieren, die Lambda-Funktionen als Reaktion auf Repository-Ereignisse aufrufen.

Weitere Informationen:

- [Erstellen Sie einen Trigger für eine Lambda-Funktion](#)
- [AWS Lambda Entwicklerhandbuch](#)

Amazon Simple Notification Service

[Amazon SNS](#) ist ein Webservice, der es Anwendungen, Endbenutzern und Geräten ermöglicht, sofort Benachrichtigungen aus der Cloud zu senden und zu empfangen. Sie können Auslöser für CodeCommit Repositories konfigurieren, die Amazon SNS SNS-Benachrichtigungen als Reaktion auf Repository-Ereignisse senden. Sie können Amazon SNS SNS-Benachrichtigungen auch verwenden, um sie in andere AWS Dienste zu integrieren. Sie können beispielsweise eine Amazon SNS SNS-Benachrichtigung verwenden, um Nachrichten an eine Amazon Simple Queue Service-Warteschlange zu senden.

Weitere Informationen:

- [Einen Auslöser für ein Amazon SNS SNS-Thema erstellen](#)
- [Entwicklerhandbuch zu Amazon Simple Notification Service](#)

Integrationsbeispiele der Community

In den folgenden Abschnitten werden Links zu Blog-Posts, Artikel und von der Community bereitgestellte Beispiele vorgestellt.

Note

Diese Links werden nur zu Informationszwecken bereitgestellt und sollten weder als umfassende Liste noch als Bestätigung des Inhalts der Beispiele betrachtet werden. AWS ist nicht verantwortlich für den Inhalt oder die Richtigkeit externer Inhalte.

Themen

- [Blog-Posts](#)
- [Codebeispiele](#)

Blog-Posts

- [Integration SonarQube als Pull Request Approver auf AWS CodeCommit](#)

Erfahren Sie, wie Sie ein CodeCommit Repository erstellen, für das eine erfolgreiche SonarQube Qualitätsanalyse erforderlich ist, bevor Pull-Requests zusammengeführt werden können.

Veröffentlichung: 12. Dezember 2019

- [Migration zu AWS CodeCommitAWS CodePipeline, und AWS CodeBuild Von GitLab](#)

Erfahren Sie, wie Sie mehrere Repositories zu „AWS CodeCommit von“ migrieren GitLab und mithilfe von und eine CI/CD-Pipeline einrichten. AWS CodePipeline AWS CodeBuild

Veröffentlichung: 22. November 2019

- [Implementierung GitFlow mithilfe von AWS CodePipeline,, und AWS CodeCommitAWS CodeBuildAWS CodeDeploy](#)

Erfahren Sie, wie Sie GitFlow mithilfe von AWS CodePipeline AWS CodeCommit, AWS CodeBuild, und implementieren AWS CodeDeploy.

Veröffentlichung: 22. Februar 2019

- [Git mit AWS CodeCommit mehreren AWS Konten verwenden](#)

Erfahren Sie, wie Sie Ihre Git-Konfiguration für mehrere Amazon Web Services Services-Konten verwalten.

Veröffentlichung: 12. Februar 2019

- [Validierung von AWS CodeCommit Pull-Requests mit und AWS CodeBuildAWS Lambda](#)

Erfahre, wie du Pull-Requests mit AWS CodeCommit AWS CodeBuild, und AWS Lambda validierst. Indem Sie Tests mit den vorgeschlagenen Änderungen durchführen, bevor Sie sie in den Standard-Branch zusammenführen, können Sie dazu beitragen, ein hohes Qualitätsniveau bei Pull-Requests sicherzustellen, potenzielle Probleme zu catch und das Vertrauen der Entwickler in ihre Änderungen zu stärken.

Veröffentlichung: 11. Februar 2019

- [Verwenden von föderierten Identitäten mit AWS CodeCommit](#)

Erfahren Sie, wie Sie AWS CodeCommit mithilfe der in Ihrem Unternehmen verwendeten Identitäten auf Repositorys zugreifen können.

Veröffentlicht am 5. Oktober 2018

- [Verfeinerung des Zugriffs auf Filialen in AWS CodeCommit](#)

Erfahren Sie, wie Sie Commits auf Repository-Banches beschränken können, indem Sie eine IAM-Richtlinie erstellen und anwenden, die einen Kontextschlüssel verwendet.

Veröffentlicht am 16. Mai 2018

- [AWS CodeCommit Repositorys mit Fargate zwischen Regionen replizieren AWS](#)

Erfahren Sie, wie Sie mithilfe einer serverlosen Architektur die kontinuierliche Replikation eines CodeCommit Repositorys von einer AWS Region in eine andere einrichten.

Veröffentlicht am 11. April 2018

- [Verteilung Ihrer Infrastruktur AWS OpsWorks for Chef Automate](#)

Erfahren Sie CodePipeline,, CodeCommit, und AWS Lambda wie Sie sicherstellen können CodeBuild, dass Kochbücher und andere Konfigurationen konsistent auf zwei oder mehr Chef-Servern bereitgestellt werden, die sich auf einem oder mehreren befinden. AWS-Regionen

Veröffentlicht am 9. März 2018

- [Peanut Butter und Chocolate: Azure Functions CI/CD-Pipeline mit AWS CodeCommit](#)

Erfahren Sie, wie Sie eine PowerShell auf Azure Functions basierende CI/CD-Pipeline erstellen, bei der der Code in einem Repository gespeichert wird. CodeCommit

Veröffentlicht am 19. Februar 2018

- [Kontinuierliche Bereitstellung auf Kubernetes mithilfe von AWS CodePipeline,, AWS CodeCommitAWS CodeBuild, Amazon ECR und AWS Lambda](#)

Erfahren Sie, wie Sie Kubernetes verwenden und AWS gemeinsam eine vollständig verwaltete, kontinuierliche Bereitstellungs-pipeline für containerbasierte Anwendungen erstellen können.

Veröffentlicht am 11. Januar 2018

- [Verwenden Sie AWS CodeCommit Pull-Requests, um Code-Reviews anzufordern und Code zu besprechen](#)

Erfahre, wie du mithilfe von Pull Requests Codeänderungen in einem CodeCommit Repository überprüfen, kommentieren und interaktiv wiederholen kannst.

Veröffentlicht am 20. November 2017

- [Erstellen Sie serverlose AWS CodeCommit Workflows mit Amazon CloudWatch Events und JGit](#)

Erfahren Sie, wie Sie mithilfe von CloudWatch Repository-Ereignissen und Zielaktionen in anderen AWS Services Regeln für Ereignisse erstellen, die Änderungen in einem CodeCommit Repository verarbeiten. Beispiele hierfür sind AWS Lambda Funktionen, die Git-Commit-Nachrichtenrichtlinien für Commits durchsetzen, ein CodeCommit Repository replizieren und ein CodeCommit Repository auf Amazon S3 sichern.

Veröffentlicht am 3. August 2017

- [Migration zu AWS CodeCommit](#)

Erfahre, wie du im Rahmen der Migration von einem anderen Git-Repository auf CodeCommit wann Code in zwei Repositories pushen kannst. SourceTree

Veröffentlicht am 6. September 2016

- [Richten Sie kontinuierliche Tests mit Appium AWS CodeCommit, Jenkins und ein AWS Device Farm](#)

Erfahren Sie, wie Sie mit Appium, CodeCommit Jenkins und Device Farm einen kontinuierlichen Testprozess für mobile Geräte einrichten.

Veröffentlicht am 2. Februar 2016

- [Verwendung AWS CodeCommit mit Git-Repositorys in mehreren Amazon Web Services Services-Konten](#)

Erfahren Sie, wie Sie Ihr CodeCommit Repository klonen und mit einem Befehl den Credential Helper so konfigurieren, dass er eine bestimmte IAM-Rolle für Verbindungen zu diesem Repository verwendet.

Veröffentlichung November 2015

- [Integrieren und AWS OpsWorksAWS CodeCommit](#)

Erfahren Sie, wie AWS OpsWorks Sie Apps und Kochbücher von Chef automatisch abrufen können. CodeCommit

Veröffentlicht am 25. August 2015

- [Helper AWS CodeCommit und Credential Helpers GitHub](#)

Erfahren Sie, wie Sie Ihre Gitconfig-Datei so konfigurieren, dass sie sowohl mit Credential Helpers als auch mit GitHub Credential CodeCommit Helpers funktioniert.

Veröffentlichung September 2015

- [Von Eclipse aus verwenden AWS CodeCommit](#)

Erfahren Sie, wie Sie die EGit Tools in Eclipse verwenden, um damit zu arbeiten CodeCommit.

Veröffentlichung August 2015

- [AWS CodeCommit mit Amazon EC2 Role Credentials](#)

Erfahren Sie, wie Sie EC2 bei der Konfiguration des automatisierten Agentenzugriffs auf ein CodeCommit Repository ein Instance-Profil für Amazon verwenden.

Veröffentlichung Juli 2015

- [Integration AWS CodeCommit mit Jenkins](#)

Erfahren Sie, wie Sie Jenkins verwenden können CodeCommit , um zwei einfache CI-Szenarien (Continuous Integration) zu unterstützen.

Veröffentlichung Juli 2015

- [Integration AWS CodeCommit mit Review Board](#)

Erfahren Sie, wie Sie sich mithilfe des [Review Board-Code-Review-Systems CodeCommit](#) in einen Entwicklungsworkflow integrieren können.

Veröffentlichung Juli 2015

Codebeispiele

Im Folgenden finden Sie Codebeispiele, die für CodeCommit Benutzer von Interesse sein könnten.

- [Mac OS X Script to Periodically Delete Cached Credentials in the OS X Certificate Store](#)

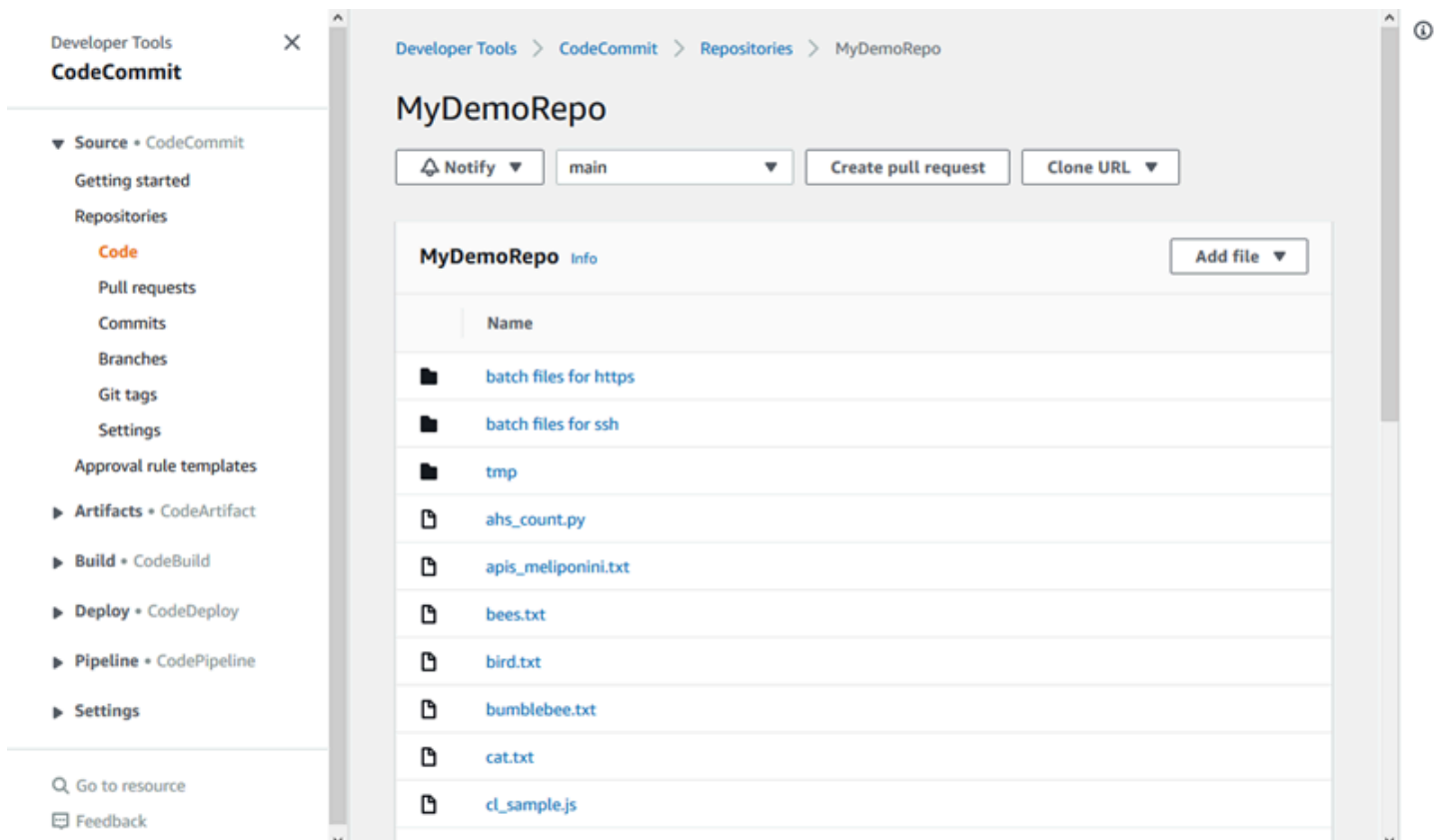
Wenn Sie den Credential Helper für CodeCommit unter Mac OS X verwenden, kennen Sie wahrscheinlich das Problem mit zwischengespeicherten Anmeldeinformationen. Dieses Skript veranschaulicht eine Lösung.

Autor: Nico Coetzee

Veröffentlichung Februar 2016

Arbeiten mit Repositorien in AWS CodeCommit

Ein Repository ist das grundlegende Objekt für die Versionskontrolle in CodeCommit. Hier können Sie Code und Dateien für Ihr Projekt sicher speichern. Zudem speichert es den gesamten Projektverlauf – vom ersten Commit bis zur letzten Änderung. Sie können das Repository mit anderen Benutzern teilen, um gemeinsam an einem Projekt zu arbeiten. Wenn Sie AWS Tags zu Repositorys hinzufügen, können Sie Benachrichtigungen einrichten, sodass Repository-Benutzer E-Mails über Ereignisse erhalten (z. B. wenn ein anderer Benutzer Code kommentiert). Außerdem können Sie die Standardeinstellungen für das Repository ändern, seine Inhalte durchsuchen und vieles mehr. Sie können Auslöser für das Repository erstellen, sodass Code per Push ausgeführt wird oder andere Ereignisse Aktionen wie E-Mails oder Codefunktionen auslösen. Wenn Sie ein Repository auf dem lokalen Computer konfigurieren (ein lokales Repository), können Sie Änderungen per Push sogar an mehrere Repositorys übertragen.



Bevor Sie Änderungen an ein CodeCommit Repository übertragen können, müssen Sie einen IAM-Benutzer in Ihrem Amazon Web Services Services-Konto konfigurieren oder den Zugriff für Verbundzugriff oder temporäre Anmeldeinformationen einrichten. Weitere Informationen erhalten Sie

unter [Schritt 1: Erstkonfiguration für CodeCommit](#) und [Einrichtungsschritte für HTTPS-Verbindungen](#) [AWS CodeCommit mit git-remote-codecommit](#).

Informationen zur Arbeit mit anderen Aspekten Ihres Repositorys finden Sie unter CodeCommit, [Mit Dateien arbeiten](#), [Verwenden von Pull-Anforderungen](#) [Mit Commits arbeiten](#) [Arbeiten mit Zweigen](#), und. [Mit Benutzereinstellungen arbeiten](#) Hinweise zur Migration zu finden Sie CodeCommit unter [Migrieren Sie zu CodeCommit](#).

Themen

- [Erstellen Sie ein Repository AWS CodeCommit](#)
- [Stellen Sie eine Connect zu einem AWS CodeCommit Repository her](#)
- [Teilen Sie ein Repository AWS CodeCommit](#)
- [Benachrichtigungen für Ereignisse in einem Repository konfigurieren AWS CodeCommit](#)
- [Kennzeichnen von Repositorys in AWS CodeCommit](#)
- [Trigger für ein AWS CodeCommit Repository verwalten](#)
- [Ein AWS CodeCommit Repository mit Amazon CodeGuru Reviewer verknüpfen oder die Zuordnung aufheben](#)
- [CodeCommit Repository-Details anzeigen](#)
- [AWS CodeCommit Repository-Einstellungen ändern](#)
- [Synchronisieren Sie Änderungen zwischen einem lokalen Repo und einem Repository AWS CodeCommit](#)
- [Push Commits in ein zusätzliches Git-Repository](#)
- [Konfigurieren Sie den kontoübergreifenden Zugriff auf ein AWS CodeCommit Repository mithilfe von Rollen](#)
- [Lösche ein Repository AWS CodeCommit](#)

Erstellen Sie ein Repository AWS CodeCommit

Verwenden Sie die AWS CodeCommit Konsole oder die AWS Command Line Interface (AWS CLI), um ein leeres CodeCommit Repository zu erstellen. Informationen zum Hinzufügen von Tags zu einem von Ihnen erstellten Repository finden Sie unter [Hinzufügen eines Tags zu einem Repository](#).

Diese Anleitungen setzen die Durchführung der Schritte unter [Einrichtung](#) voraus.

 Note

Je nach Nutzung können Ihnen Gebühren für die Erstellung oder den Zugriff auf ein Repository berechnet werden. Weitere Informationen finden Sie auf der CodeCommit Produktinformationsseite unter [Preise](#).

Themen

- [Erstellen Sie ein Repository \(Konsole\)](#)
- [Erstellen Sie ein Repository \(\)AWS CLI](#)

Erstellen Sie ein Repository (Konsole)

Um ein CodeCommit Repository zu erstellen

 Important

Das Erstellen eines Repositories in AWS CodeCommit ist für Neukunden nicht mehr verfügbar. Bestandskunden von AWS CodeCommit können die API weiterhin wie gewohnt verwenden. [Erfahren Sie mehr](#)

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie in der Regionsauswahl den Ort aus, AWS-Region an dem Sie das Repository erstellen möchten. Weitere Informationen finden Sie unter [Regionen und Git-Verbindungsendpunkte](#).
3. Wählen Sie auf der Seite Repositories die Option Repository auswählen.
4. Geben Sie auf der Seite Create repository (Repository erstellen) im Feld Repository name (Repository-Name) einen Namen für das Repository ein.

 Note

Bei den Repository-Namen wird nach Groß- und Kleinschreibung unterschieden. Der Name muss AWS-Region für Ihr Amazon Web Services Services-Konto eindeutig sein.

5. (Optional) Geben Sie unter Description (Beschreibung) eine Beschreibung für das Repository ein. Dadurch können Sie und andere Benutzer den Zweck des Repositorys leichter erkennen.

 Note

Das Beschreibungsfeld zeigt Markdown in der Konsole an und akzeptiert alle HTML-Zeichen und gültigen Unicode-Zeichen. Wenn Sie als Anwendungsentwickler das `GetRepository` oder verwenden `BatchGetRepositories` APIs und das Repository-Beschreibungsfeld in einem Webbrowser anzeigen möchten, finden Sie weitere Informationen in der [CodeCommit API-Referenz](#).

6. (Optional) Wählen Sie Tag hinzufügen, um Ihrem Repository ein oder mehrere Repository-Tags (eine benutzerdefinierte Attributbezeichnung, mit der Sie Ihre AWS Ressourcen organisieren und verwalten können) hinzuzufügen. Weitere Informationen finden Sie unter [Kennzeichnen von Repositories in AWS CodeCommit](#).
7. (Optional) Erweitern Sie Zusätzliche Konfiguration, um anzugeben, ob Sie den Standardschlüssel Von AWS verwalteter Schlüssel oder Ihren eigenen, vom Kunden verwalteten Schlüssel zum Verschlüsseln und Entschlüsseln von Daten in diesem Repository verwenden möchten. Wenn Sie Ihren eigenen, vom Kunden verwalteten Schlüssel verwenden möchten, müssen Sie sicherstellen, dass er dort verfügbar ist, AWS-Region wo Sie das Repository erstellen, und dass der Schlüssel aktiv ist. Weitere Informationen finden Sie unter [AWS Key Management Service und Verschlüsselung für AWS CodeCommit Repositorien](#).
8. (Optional) Wählen Sie Amazon CodeGuru Reviewer für Java und Python aktivieren, wenn dieses Repository Java- oder Python-Code enthält und Sie möchten, dass CodeGuru Reviewer ihn analysiert. CodeGuru Reviewer verwendet mehrere Modelle für maschinelles Lernen, um Codefehler zu finden und Verbesserungen und Korrekturen in Pull-Requests vorzuschlagen. Weitere Informationen finden Sie im [Amazon CodeGuru Reviewer-Benutzerhandbuch](#).
9. Wählen Sie Erstellen aus.

Nachdem Sie ein Repository erstellt haben, können Sie eine Verbindung zu diesem herstellen und mit dem Hinzufügen von Code beginnen, entweder über die CodeCommit Konsole oder einen lokalen Git-Client oder indem Sie Ihr CodeCommit Repository in Ihre bevorzugte IDE integrieren. Weitere Informationen finden Sie unter [Einrichtung für AWS CodeCommit](#). Sie können das Repository auch zu einer kontinuierlichen Bereitstellungs-Pipeline hinzufügen. Weitere Informationen finden Sie unter [Anleitung zum Erstellen einer einfachen Pipeline](#).

Um Informationen über das neue CodeCommit Repository zu erhalten, z. B. die beim Klonen des Repositorys URLs zu verwenden, wählen Sie den Namen des Repositorys aus der Liste oder wählen Sie einfach das Verbindungsprotokoll, das Sie neben dem Namen des Repositorys verwenden möchten.

Um dieses Repository für andere Benutzer freizugeben, müssen Sie ihnen den zum Klonen des Repositorys zu verwendenden HTTPS- oder SSH-Link senden. Vergewissern Sie sich, dass sie über die Berechtigungen verfügen, die zum Zugriff auf das Repository erforderlich sind. Weitere Informationen erhalten Sie unter [Teilen Sie ein Repository](#) und [Authentifizierung und Zugriffskontrolle für AWS CodeCommit](#).

Erstellen Sie ein Repository (AWS CLI)

Sie können den verwenden AWS CLI , um ein CodeCommit Repository zu erstellen. Anders als bei der Konsole können Sie dem Repository Tags hinzufügen, wenn Sie es mithilfe der AWS CLI erstellen.

Important

Das Erstellen eines Repositorys in AWS CodeCommit ist für Neukunden nicht mehr verfügbar. Bestandskunden von AWS CodeCommit können die API weiterhin wie gewohnt verwenden. [Erfahren Sie mehr](#)

1. Stellen Sie sicher, dass Sie das AWS CLI mit dem konfiguriert haben AWS-Region , wo sich das Repository befindet. Um die Region zu verifizieren, geben Sie den folgenden Befehl in das Befehlszeilen- oder Terminalfenster ein. Prüfen Sie dann die Angaben des Standardregionsnamens.

```
aws configure
```

Der Name der Standardregion muss mit dem Namen des AWS-Region Repositorys in übereinstimmen CodeCommit. Weitere Informationen finden Sie unter [Regionen und Git-Verbindungsendpunkte](#).

2. Führen Sie den Befehl create-repository aus und geben Sie Folgendes an:
 - Ein Name, der das CodeCommit Repository eindeutig identifiziert (mit der `--repository-name` Option).

 Note

Dieser Name muss in einem Amazon Web Services Services-Konto eindeutig sein.

- Ein optionaler Kommentar zum CodeCommit Repository (mit der `--repository-description` Option).
- Ein oder mehrere optionale Schlüssel-Wert-Paare, die als Tags für das CodeCommit Repository verwendet werden sollen (mit der `--tags` Option).
- Ein optionaler, vom Kunden verwalteter Schlüssel, der beim Verschlüsseln und Entschlüsseln dieses Repositories verwendet werden kann. Alle Repositories werden bei der Übertragung und im Ruhezustand mit einem Schlüssel verschlüsselt. AWS KMS Wenn kein Schlüssel angegeben ist, wird der AWS verwaltete Standardschlüssel `aws/codecommit` verwendet.

Um beispielsweise ein CodeCommit Repository `MyDemoRepo` mit der Beschreibung `"My demonstration repository"` und einem Tag mit einem Schlüssel `Team` mit dem Wert von `Saanvi` zu erstellen, verwenden Sie diesen Befehl.

```
aws codecommit create-repository --repository-name MyDemoRepo --repository-description "My demonstration repository" --tags Team=Saanvi
```

 Note

Das Beschreibungsfeld zeigt Markdown in der Konsole an und akzeptiert alle HTML-Zeichen und gültigen Unicode-Zeichen. Wenn Sie als Anwendungsentwickler das `GetRepository` oder verwenden `BatchGetRepositories` APIs und das Repository-Beschreibungsfeld in einem Webbrowser anzeigen möchten, finden Sie weitere Informationen in der [CodeCommit API-Referenz](#).

3. Ist der Befehl erfolgreich, wird ein `repositoryMetadata`-Objekt mit den folgenden Informationen ausgegeben:
 - Beschreibung (`repositoryDescription`)
 - Eindeutige, systemgenerierte ID (`repositoryId`)
 - Name (`repositoryName`)

- Die ID des Amazon Web Services Services-Kontos, das dem CodeCommit Repository zugeordnet ist (accountId).

Es folgt eine Beispielausgabe basierend auf dem vorangegangenen Beispielbefehl:

```
{
  "repositoryMetadata": {
    "repositoryName": "MyDemoRepo",
    "cloneUrlSsh": "ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo",
    "lastModifiedDate": 1446071622.494,
    "repositoryDescription": "My demonstration repository",
    "cloneUrlHttp": "https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo",
    "defaultBranch": main,
    "kmsKeyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "creationDate": 1446071622.494,
    "repositoryId": "f7579e13-b83e-4027-aaef-650c0EXAMPLE",
    "Arn": "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo",
    "accountId": "111111111111"
  }
}
```

Note

Tags, die beim Erstellen des Repository hinzugefügt wurden, werden nicht in der Ausgabe zurückgegeben. Um die einem Repository zugewiesenen Tags aufzulisten, führen Sie den Befehl [list-tags-for-resource](#) aus.

4. Notieren Sie sich den Namen und die ID des CodeCommit Repositorys. Sie benötigen sie, um Informationen über das CodeCommit Repository zu überwachen und zu ändern, insbesondere wenn Sie es verwenden AWS CLI.

Wenn Sie den Namen oder die ID vergessen haben, führen Sie die Anweisungen unter [CodeCommit Repository-Details anzeigen \(AWS CLI\)](#) aus.

Nach dem Erstellen eines Repositorys können Sie eine Verbindung mit diesem herstellen und Code hinzufügen. Weitere Informationen finden Sie unter [Herstellen einer Verbindung mit einem](#)

[Repository](#). Sie können das Repository auch zu einer kontinuierlichen Bereitstellungs-Pipeline hinzufügen. Weitere Informationen finden Sie unter [Anleitung zum Erstellen einer einfachen Pipeline](#).

Stellen Sie eine Connect zu einem AWS CodeCommit Repository her

Wenn Sie zum ersten Mal eine Verbindung zu einem CodeCommit Repository herstellen, klonen Sie in der Regel dessen Inhalt auf Ihren lokalen Computer. Sie können einem Repository auch direkt von der CodeCommit Konsole aus [Dateien hinzufügen und Dateien darin bearbeiten](#). Wenn Sie bereits ein lokales Repository besitzen, können Sie ein CodeCommit-Repository auch als Remote-Repository hinzufügen. Dieses Thema enthält Anweisungen zum Herstellen einer Verbindung zu einem CodeCommit Repository. Informationen dazu, wie Sie ein vorhandenes Repository migrieren möchten CodeCommit, finden Sie unter [Migrieren Sie zu CodeCommit](#).

Note

Je nach Nutzung können Ihnen Gebühren für die Erstellung eines Repositories oder den Zugriff auf ein Repository in Rechnung gestellt werden. Weitere Informationen finden Sie auf der CodeCommit Produktinformationsseite unter [Preise](#).

Themen

- [Voraussetzungen für die Verbindung mit einem CodeCommit Repository](#)
- [Connect zum CodeCommit Repository her, indem Sie das Repository klonen](#)
- [Ein lokales Repo mit dem CodeCommit Repository Connect](#)

Voraussetzungen für die Verbindung mit einem CodeCommit Repository

Bevor Sie ein CodeCommit Repository klonen oder ein lokales Repository mit einem CodeCommit Repository verbinden können:

- Der lokale Computer muss mit der entsprechenden Software und den Einstellungen konfiguriert sein, die eine Verbindung mit CodeCommit zulassen. Dazu gehört die Installation und Konfiguration von Git. Weitere Informationen erhalten Sie unter [Einrichtung](#) und [Erste Schritte mit Git und AWS CodeCommit](#).

- Sie benötigen die Klon-URL des CodeCommit Repositorys, zu dem Sie eine Verbindung herstellen möchten. Weitere Informationen finden Sie unter [Repository-Details anzeigen](#).

Wenn Sie noch kein CodeCommit Repository erstellt haben, folgen Sie den Anweisungen unter [Erstellen eines -Repositorys](#), kopieren Sie die Klon-URL des CodeCommit Repositorys und kehren Sie zu dieser Seite zurück.

Wenn Sie ein CodeCommit Repository haben, dessen Namen Sie jedoch nicht kennen, folgen Sie den Anweisungen unter [Repository-Details anzeigen](#).

- Sie benötigen einen Speicherort auf Ihrem lokalen Computer, an dem Sie eine lokale Kopie des CodeCommit Repositorys speichern können, zu dem Sie eine Verbindung herstellen. (Diese lokale Kopie des CodeCommit Repositorys wird als lokales Repo bezeichnet.) Sie wechseln dann zu Git-Befehlen, die Sie an diesem Ort ausführen. Sie könnten beispielsweise /tmp (für Linux, macOS oder Unix) oder c:\temp (für Windows) verwenden, wenn Sie zu Testzwecken einen temporären Klon erstellen. Dies ist der Verzeichnispfad, der in diesen Beispielen verwendet wird.

Note

Sie können jedes beliebige Verzeichnis verwenden. Wenn Sie ein Repository für eine langfristige Nutzung klonen, sollten Sie den Klon aus einem Arbeitsverzeichnis erstellen, das nicht für temporäre Dateien genutzt wird. Bei Verwendung eines anderen Verzeichnisses als /tmp oder c:\temp müssen Sie das von uns angegebene Verzeichnis in diesen Anweisungen mit diesem Verzeichnis ersetzen.

Connect zum CodeCommit Repository her, indem Sie das Repository klonen

Wenn Sie noch kein lokales Repository haben, folgen Sie den Schritten in diesem Verfahren, um das CodeCommit Repository auf Ihren lokalen Computer zu klonen.

1. Sorgen Sie dafür, dass die Voraussetzungen erfüllt sind, einschließlich [Einrichtung](#).

Important

Wenn Sie die Einrichtung nicht abgeschlossen haben, können Sie weder eine Verbindung herstellen noch das Repository klonen.

2. Verwenden Sie im Verzeichnis /tmp oder c:\temp Git, um den Befehl clone auszuführen. Die folgenden Beispiele zeigen, wie Sie ein Repository klonen, das *MyDemoRepo* in der Region USA Ost (Ohio) benannt ist.

Für HTTPS mit [Git-Anmeldeinformationen](#) oder dem Hilfsprogramm für Anmeldeinformationen, der in der AWS CLI enthalten ist:

```
git clone https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
```

Für die Verwendung von [git-remote-codecommit](#) HTTPS wird das Standardprofil vorausgesetzt und AWS-Region konfiguriert in AWS CLI:

```
git clone codecommit://MyDemoRepo my-demo-repo
```

Für SSH:

```
git clone ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
```

In diesem Beispiel `git-codecommit.us-east-2.amazonaws.com` steht der Git-Verbindungspunkt für die Region USA Ost (Ohio), in der das Repository existiert, `MyDemoRepo` für den Namen Ihres CodeCommit Repositories und `my-demo-repo` für den Namen des Verzeichnisses, das Git in dem /tmp Verzeichnis oder dem c:\temp Verzeichnis erstellt. Weitere Informationen zur That-Unterstützung CodeCommit und zu den Git-Verbindungen für diese AWS-Regionen finden Sie unter [Regionen und Git-Verbindungsendpunkte](#). AWS-Regionen

Note

Beim Klonen eines Repositories mit SSH auf Windows-Betriebssystemen müssen Sie gegebenenfalls die SSH-Schlüssel-ID wie folgt der Verbindungszeichenfolge hinzufügen:

```
git clone ssh://Your-SSH-Key-ID@git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
```

Weitere Informationen erhalten Sie unter [Für SSH-Verbindungen unter Windows](#) und [Fehlerbehebung](#).

Nachdem Git das Verzeichnis erstellt hat, wird eine Kopie Ihres CodeCommit Repositorys in das neu erstellte Verzeichnis heruntergeladen.

Wenn das CodeCommit Repository neu oder anderweitig leer ist, siehst du eine Meldung, dass du ein leeres Repository klonst. Das ist normal.

Note

Wenn Sie eine Fehlermeldung erhalten, dass Git das CodeCommit Repository nicht finden kann oder dass Sie nicht berechtigt sind, eine Verbindung zum CodeCommit Repository herzustellen, stellen Sie sicher, dass Sie die [Voraussetzungen erfüllt](#) haben, einschließlich der Zuweisung von Berechtigungen für den IAM-Benutzer und der Einrichtung Ihrer IAM-Benutzeranmeldeinformationen für Git und CodeCommit auf dem lokalen Computer. Vergewissern Sie sich auch, dass der Repository-Name stimmt.

Nachdem du dein lokales Repo erfolgreich mit deinem CodeCommit Repository verbunden hast, kannst du jetzt damit beginnen, Git-Befehle vom lokalen Repo aus auszuführen, um Commits, Branches und Tags zu erstellen und zum Repository zu pushen und daraus zu ziehen. CodeCommit

Ein lokales Repo mit dem CodeCommit Repository Connect

Führen Sie die folgenden Schritte aus, wenn Sie bereits ein lokales Repository besitzen und ein CodeCommit-Repository als Remote-Repository hinzufügen möchten. Wenn du bereits über ein Remote-Repository verfügst und deine Commits in dieses andere Remote-Repository übertragen möchtest, befolge die Schritte unter [CodeCommit Push-Commits an zwei Repositorys](#)

1. Sorgen Sie dafür, dass die [Voraussetzungen](#) erfüllt sind.
2. Wechseln Sie von der Befehlszeile oder dem Terminal aus zu Ihrem lokalen Repo-Verzeichnis und führen Sie den `git remote add` Befehl aus, um das CodeCommit Repository als Remote-Repository für Ihr lokales Repo hinzuzufügen.

Mit dem folgenden Befehl wird beispielsweise die Fernbedienung mit dem Spitznamen **origin** mit `us-east-2.amazonaws.com/v1/repos/MyDemoRepo` hinzugefügt.

Für HTTPS:

```
git remote add origin https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo
```

Für SSH:

```
git remote add origin ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo
```

Mit diesem Befehl wird kein Inhalt zurückgegeben.

3. Um zu überprüfen, ob Sie das CodeCommit Repository als Remote-Repository für Ihr lokales Repository hinzugefügt haben, führen Sie den `git remote -v` Befehl aus, der eine Ausgabe ähnlich der folgenden erzeugen sollte:

Für HTTPS:

```
origin https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (fetch)
origin https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (push)
```

Für SSH:

```
origin ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (fetch)
origin ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (push)
```

Nachdem Sie Ihr lokales Repo erfolgreich mit Ihrem CodeCommit Repository verbunden haben, können Sie damit beginnen, Git-Befehle vom lokalen Repo aus auszuführen, um Commits, Branches und Tags zu erstellen und Push zum Repository zu senden und daraus zu ziehen. CodeCommit

Teilen Sie ein Repository AWS CodeCommit

Nachdem Sie ein CodeCommit Repository erstellt haben, können Sie es mit anderen Benutzern teilen. Stellen Sie zunächst fest, ob Sie beim Zugriff CodeCommit einen Verbundzugriff, temporäre Anmeldeinformationen oder einen Web-Identitätsanbieter wie IAM Identity Center verwenden

oder ob Sie Git-Anmeldeinformationen oder SSH-Schlüsselpaare mit IAM-Benutzern verwenden möchten. Wenn Sie ersteres verwenden, müssen Sie Benutzer, Zugriff und Berechtigungen für Ihren Identitätsanbieter einrichten und dann Anweisungen zur Verwendung durch Ihre Benutzer bereitstellen. git-remote-codecommit Weitere Informationen erhalten Sie unter [Einrichtungsschritte für HTTPS-Verbindungen AWS CodeCommit mit git-remote-codecommit](#) und [Verbindung zu AWS CodeCommit Repositories mit rotierenden Anmeldeinformationen herstellen](#).

Sie können Git-Anmeldeinformationen oder SSH-Schlüsselpaare nicht mit Verbundzugriffs- oder Identitätsanbietern verwenden, aber viele IDEs funktionieren am besten mit diesen Anmeldeinformationen. Entscheiden Sie in diesem Fall, welches Protokoll (HTTPS oder SSH) Sie Benutzern empfehlen möchten, wenn Sie klonen und einen Git-Client oder eine IDE verwenden, um eine Verbindung zu Ihrem Repository herzustellen. Senden Sie dann die URL- und Verbindungsinformationen an die Benutzer, für die Sie das Repository freigeben möchten. Abhängig von Ihren Sicherheitsanforderungen kann es für die gemeinsame Nutzung eines Repositories auch erforderlich sein, eine IAM-Gruppe zu erstellen, verwaltete Richtlinien auf diese Gruppe anzuwenden und IAM-Richtlinien zu bearbeiten, um den Zugriff zu verfeinern, oder IAM-Rollen zu erstellen und zu verwenden.

Note

Nachdem Sie Benutzern Konsolenzugriff auf das Repository gewährt haben, können sie Dateien direkt in der Konsole hinzufügen oder bearbeiten, ohne einen Git-Client einrichten oder eine andere Verbindung herstellen zu müssen. Weitere Informationen erhalten Sie unter [Eine Datei erstellen oder zu einem AWS CodeCommit Repository hinzufügen](#) und [Den Inhalt einer Datei in einem AWS CodeCommit Repository bearbeiten](#).

Für diese Anweisungen wird davon ausgegangen, dass Sie bereits die Schritte unter [Einrichtung](#) und [Erstellen eines -Repositorys](#) ausgeführt haben.

Note

Abhängig von Ihrer Nutzung können Ihnen Gebühren für die Erstellung eines Repositories oder den Zugriff auf dieses Repository in Rechnung gestellt werden. Weitere Informationen finden Sie auf der CodeCommit Produktinformationsseite unter [Preise](#).

Themen

- [Wählen Sie das Verbindungsprotokoll, das Sie mit Ihren Benutzern teilen möchten](#)
- [Erstellen Sie IAM-Richtlinien für Ihr Repository](#)
- [Erstellen Sie eine IAM-Gruppe für Repository-Benutzer](#)
- [Teilen Sie die Verbindungsinformationen mit Ihren Benutzern](#)

Wählen Sie das Verbindungsprotokoll, das Sie mit Ihren Benutzern teilen möchten

Wenn Sie ein Repository in erstellen CodeCommit, werden zwei Endpunkte generiert: einer für HTTPS-Verbindungen und einer für SSH-Verbindungen. Beide bieten sichere Verbindungen über ein Netzwerk. Ihre Benutzer können beide Protokolle verwenden. Beide Endpunkte bleiben aktiv, unabhängig davon, welches Protokoll Sie Ihren Benutzern empfehlen.

HTTPS-Verbindungen erfordern:

- Git-Anmeldeinformationen, die IAM-Benutzer in IAM für sich selbst generieren können. Git-Anmeldeinformationen stellen die für Benutzer Ihres Repositories am einfachsten einzurichtende und zu nutzende Methode dar.
- Ein AWS Zugriffsschlüssel oder eine Rolle, die Sie annehmen müssen und die Ihre Repository-Benutzer in ihrem Anmeldeinformationsprofil konfigurieren müssen. Sie können git-remote-codecommit konfigurieren (empfohlen) oder das Hilfsprogramm für Anmeldeinformationen, der in der AWS CLI enthalten ist, verwenden. Diese stellen die einzige mögliche Methode für Root-Konto- oder verbundene Benutzer dar.

Für SSH-Verbindungen müssen Ihre Benutzer Folgendes beachten:

- Generieren Sie ein Schlüsselpaar aus öffentlichem und privatem Schlüssel.
- Speichern Sie den öffentlichen Schlüssel.
- Ordnen Sie den öffentlichen Schlüssel ihrem IAM-Benutzer zu.
- Konfigurieren Sie bekannte Host-Dateien auf dem lokalen Computer.
- Erstellen und verwalten Sie eine Konfigurationsdatei auf dem lokalen Computer.

Da dies ein komplexerer Konfigurationsprozess ist, empfehlen wir, dass Sie HTTPS- und Git-Anmeldeinformationen für Verbindungen zu wählen CodeCommit.

Weitere Informationen über HTTPS, SSH, Git, git-remote-codecommit und Remote-Repositorys finden Sie unter [Einrichtung](#), [Verbindung zu AWS CodeCommit Repositorys mit rotierenden Anmeldeinformationen herstellen](#) oder in der Git-Dokumentation.

Note

Git unterstützt zwar unterschiedliche Verbindungsprotokolle, jedoch werden Verbindungen mit ungesicherten Protokollen, wie etwa das lokale Protokoll oder allgemeines HTTP von CodeCommit nicht unterstützt.

Erstellen Sie IAM-Richtlinien für Ihr Repository

AWS bietet drei verwaltete Richtlinien in IAM für CodeCommit. Diese Richtlinien können nicht bearbeitet werden und gelten für alle Repositorys, die mit Ihrem Amazon Web Services Services-Konto verknüpft sind. Sie können diese Richtlinien jedoch als Vorlagen verwenden, um eigene kundenverwaltete Richtlinien zu erstellen, die nur für das Repository gelten, das Sie freigeben möchten. Ihre kundenverwaltete Richtlinie kann speziell für das Repository gelten, das Sie freigeben möchten. Weitere Informationen finden Sie unter [Verwaltete Richtlinien](#) und [IAM-Benutzer und -Gruppen](#).

Tip

Für eine genauere Steuerung des Zugriffs auf Ihr Repository können Sie mehr als eine vom Kunden verwaltete Richtlinie erstellen und die Richtlinien auf verschiedene IAM-Benutzer und -Gruppen anwenden.

Weitere Informationen zum Überprüfen der Inhalte von verwalteten Richtlinien und Verwenden von Richtlinien zum Erstellen und Anwenden von Berechtigungen finden Sie unter [Authentifizierung und Zugriffskontrolle für AWS CodeCommit](#).

Erstellen einer kundenverwalteten Richtlinie für das Repository

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter <https://console.aws.amazon.com/iam/>
2. Wählen Sie im Navigationsbereich Dashboard die Optionen Policies und dann Create Policy aus.
3. Wählen Sie auf der Seite „Richtlinie erstellen“ die Option Verwaltete Richtlinie importieren aus.

4. Geben Sie auf der Seite Verwaltete Richtlinien importieren im Feld Richtlinien filtern den Text **einAWSCodeCommitPowerUser**. Klicken Sie auf die Schaltfläche neben dem Richtliniennamen und wählen Sie dann Importieren aus.
5. Wählen Sie auf der Seite Create policy (Richtlinie erstellen) die Option JSON aus. Ersetzen Sie den Teil „*“ in der Resource Zeile für CodeCommit Aktionen durch den Amazon-Ressourcennamen (ARN) des CodeCommit Repositorys, wie hier gezeigt:

```
"Resource": [  
  "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo"  
]
```

 Tip

Um den ARN für das CodeCommit Repository zu finden, gehen Sie zur CodeCommit Konsole, wählen Sie den Repository-Namen aus der Liste aus und wählen Sie dann Einstellungen. Weitere Informationen finden Sie unter [Repository-Details anzeigen](#).

Wenn Sie diese Richtlinie auf mehr als ein Repository anwenden möchten, fügen Sie jedes Repository als Ressource hinzu, indem Sie seinen ARN angeben. Fügen Sie zwischen den Ressourcenanweisungen ein Komma ein, wie hier dargestellt:

```
"Resource": [  
  "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo",  
  "arn:aws:codecommit:us-east-2:111111111111:MyOtherDemoRepo"  
]
```

Wenn Sie mit der Bearbeitung fertig sind, wählen Sie „Richtlinie überprüfen“.

6. Geben Sie auf der Seite „Richtlinie überprüfen“ im Feld Name einen neuen Namen für die Richtlinie ein (z. B. **AWSCodeCommitPowerUser-MyDemoRepo**). Geben Sie optional eine Beschreibung für diese Richtlinie ein.
7. Wählen Sie Richtlinie erstellen aus.

Erstellen Sie eine IAM-Gruppe für Repository-Benutzer

Um den Zugriff auf Ihr Repository zu verwalten, erstellen Sie eine IAM-Gruppe für deren Benutzer, fügen Sie dieser Gruppe IAM-Benutzer hinzu und fügen Sie dann die vom Kunden verwaltete

Richtlinie hinzu, die Sie im vorherigen Schritt erstellt haben. Alternativ können Sie eine Rolle mit einer angehängten, vom Kunden verwalteten Richtlinie erstellen und Benutzer diese Rolle übernehmen lassen.

Wenn Sie SSH verwenden, müssen Sie der IAMUser SSHKeys Gruppe eine weitere verwaltete Richtlinie hinzufügen, nämlich die von IAM verwaltete Richtlinie, die es Benutzern ermöglicht, ihren öffentlichen SSH-Schlüssel hochzuladen und ihn dem IAM-Benutzer zuzuordnen, mit dem sie sich verbinden. CodeCommit

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter <https://console.aws.amazon.com/iam/>
2. Wählen Sie im Navigationsbereich Dashboard die Option Groups und dann Create New Group aus.
3. Geben Sie auf der Seite Set Group Name (Name für Gruppe festlegen) im Feld Group Name (Gruppenname) einen Namen für die Gruppe ein (zum Beispiel *MyDemoRepoGroup*) und wählen Sie dann Next Step (Nächster Schritt) aus. Sie sollten den Repository-Namen als Teil des Gruppennamens einfügen.

 Note

Dieser Name muss in einem Amazon Web Services Services-Konto eindeutig sein.

4. Markieren Sie das Kästchen neben der vom Kunden verwalteten Richtlinie, die Sie im vorherigen Abschnitt erstellt haben (z. B. AWSCodeCommitPowerUser-MyDemoRepo).
5. Klicken Sie auf der Seite Review auf Create Group. IAM erstellt diese Gruppe mit den angegebenen Richtlinien, die bereits angehängt sind. Die Gruppe erscheint in der Liste der Gruppen, die mit Ihrem Amazon Web Services Services-Konto verknüpft sind.
6. Wählen Sie Ihre Gruppe in der Liste aus.
7. Wählen Sie auf der Gruppenübersichtsseite die Registerkarte Users und anschließend die Option Add Users to Groups aus. Markieren Sie in der Liste, in der alle mit Ihrem Amazon Web Services Services-Konto verknüpften Benutzer angezeigt werden, die Kästchen neben den Benutzern, denen Sie Zugriff auf das CodeCommit Repository gewähren möchten, und wählen Sie dann Benutzer hinzufügen.

 Tip

Sie können über das Suchfeld schnell nach Benutzern anhand ihres Namens suchen.

8. Wenn Sie Ihre Benutzer hinzugefügt haben, schließen Sie die IAM-Konsole.

Teilen Sie die Verbindungsinformationen mit Ihren Benutzern

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie in der Regionsauswahl den Ort aus, AWS-Region an dem das Repository erstellt wurde. Repositories sind spezifisch für eine AWS-Region. Weitere Informationen finden Sie unter [Regionen und Git-Verbindungsendpunkte](#).
3. Wählen Sie auf der Seite Repositories (Repositories) das Repository aus, das Sie freigeben möchten.
4. Wählen Sie unter Clone URL (Klon-URL) das Protokoll aus, das die Benutzer verwenden sollen. Dadurch wird die Klon-URL für das Verbindungsprotokoll kopiert.
5. Senden Sie Ihren Benutzern die Klon-URL zusammen mit allen anderen Anweisungen, z. B. zur Installation von AWS CLI, zur Konfiguration eines Profils oder zur Installation von Git. Stellen Sie sicher, dass Sie die Konfigurationsinformationen für das Verbindungsprotokoll (z. B. HTTPS) angeben.

Die folgende Beispiel-E-Mail enthält Informationen für Benutzer, die sich mit dem HTTPS-Verbindungsprotokoll und Git-Anmeldeinformationen in der Region USA Ost (Ohio) (us-east-2) mit dem MyDemoRepo Repository verbinden. Diese E-Mail setzt voraus, dass der Benutzer Git bereits installiert hat und mit der Verwendung vertraut ist.

I've created a CodeCommit repository for us to use while working on our project. The name of the repository is *MyDemoRepo*, and it is in the US East (Ohio) (us-east-2) region. Here's what you need to do in order to get started using it:

1. Make sure that your version of Git on your local computer is 1.7.9 or later.
2. Generate Git credentials for your IAM user by signing into the IAM console here: <https://console.aws.amazon.com/iam/>.

Switch to the **Security credentials** tab for your IAM user and choose the **Generate** button in **HTTPS Git credentials for CodeCommit**.

Make sure to save your credentials in a secure location!

3. Switch to a directory of your choice and clone the CodeCommit repository to your local machine by running the following command:

```
git clone https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
```

4. When prompted for user name and password, use the Git credentials you just saved.

That's it! If you'd like to learn more about using CodeCommit, you can start with the tutorial [here](#).

Umfassende Einrichtungsanweisungen finden Sie unter [Einrichtung](#).

Benachrichtigungen für Ereignisse in einem Repository konfigurieren AWS CodeCommit

Sie können Benachrichtigungsregeln für ein Repository einrichten, sodass Repository-Benutzer E-Mails zu den von Ihnen angegebenen Repository-Ereignistypen erhalten. Es werden Benachrichtigungen gesendet, wenn Ereignisse mit den Einstellungen der Benachrichtigungsregel übereinstimmen. Sie können ein Amazon SNS SNS-Thema für Benachrichtigungen erstellen oder ein vorhandenes Thema in Ihrem Amazon Web Services Services-Konto verwenden. Sie können die CodeCommit Konsole und die verwenden AWS CLI , um Benachrichtigungsregeln zu konfigurieren.

Developer Tools > CodeCommit > Repositories > MyDemoRepo > Settings

Create notification rule

Notification rules set up a subscription to events that happen with your resources. When these events occur, you will receive notifications sent to the targets you designate. You can manage your notification preferences in Settings. [Info](#)

Notification rule settings

Notification name

Detail type

Choose the level of detail you want in notifications. [Learn more about notifications and security](#)

☒ **Full**
Includes any supplemental information about events provided by the resource or the notifications feature.

☐ **Basic**
Includes only information provided in resource events.

Events that trigger notifications

Select noneSelect all

Comments	Pull request	Branches and tags
☐ On Commits	☒ Source Updated	☐ Created
☒ On Pull requests	☒ Created	☒ Deleted
	☒ Status Changed	☐ Updated
	☒ Merged	

Targets

Choose an SNS topic to use as the target for the notification rule. Users can subscribe to the notification topic to receive emails about events. You can also configure integration between the SNS topic and AWS Chatbot, so that users receive notifications in Slack channels or Amazon Chime chatrooms.

You can also configure integration between the SNS topic and AWS Chatbot, so that users receive notifications in Slack channels or Amazon Chime chatrooms. [Learn more](#)

Amazon SNS topic ARN

× Remove target

Add target

Create SNS topic

Cancel Submit

Themen

- [Verwenden von Repository-Benachrichtigungsregeln](#)
- [Erstellen einer Benachrichtigungsregel](#)

- [Benachrichtigungen ändern oder deaktivieren](#)
- [Benachrichtigungen löschen](#)

Verwenden von Repository-Benachrichtigungsregeln

Durch das Konfigurieren von Benachrichtigungsregeln werden Ihre Repository-Benutzer unterstützt, da Benutzer E-Mails erhalten, wenn eine Aktion ausgeführt wird, die sich auf einen anderen Benutzer auswirkt. Sie können Benachrichtigungsregeln beispielsweise so konfigurieren, dass Benachrichtigungen gesendet werden, wenn Kommentare zu Commits erstellt werden. Wenn in dieser Konfiguration ein Repository-Benutzer eine Codezeile in einem Commit kommentiert, werden andere Repository-Benutzer per E-Mail benachrichtigt. Sie können sich anmelden und den Kommentar ansehen. Mit den Antworten auf Kommentare werden ebenfalls E-Mails generiert, sodass Repository-Benutzer informiert bleiben.

Benachrichtigungsregeln unterscheiden sich von Repository-Trigger und sie unterscheiden sich auch von den Benachrichtigungen, die Sie vor dem 5. November 2019 in der CodeCommit Konsole konfigurieren konnten.

- Sie können zwar einen Trigger so konfigurieren, dass er Amazon SNS verwendet, um E-Mails über bestimmte Repository-Ereignisse zu senden, aber diese Ereignisse sind auf betriebliche Ereignisse beschränkt, z. B. das Erstellen von Branches und das Übertragen von Code an einen Branch. Trigger verwenden keine CloudWatch Ereignisregeln zur Auswertung von Repository-Ereignissen. Sie sind begrenzt. Weitere Informationen zur Verwendung von Auslösern finden Sie unter [Verwalten von Auslösern für ein Repository](#).
- Für Benachrichtigungen, die vor dem 5. November 2019 konfiguriert wurden, waren weniger Ereignistypen verfügbar und sie konnten nicht für die Integration mit Amazon Chime Chime-Chatrooms oder Slack-Kanälen konfiguriert werden. Sie können weiterhin Benachrichtigungen verwenden, die vor dem 5. November 2019 konfiguriert wurden. Sie können jedoch keine Benachrichtigungen dieses Typs erstellen. Erstellen und verwenden Sie stattdessen Benachrichtigungsregeln. Wir empfehlen, Benachrichtigungsregeln zu verwenden und Benachrichtigungen, die vor dem 5. November 2019 erstellt wurden, zu deaktivieren oder zu löschen. Weitere Informationen erhalten Sie unter [Erstellen einer Benachrichtigungsregel](#) und [Benachrichtigungen löschen](#).

Erstellen einer Benachrichtigungsregel

Sie können Benachrichtigungsregeln verwenden, um Benutzer über wichtige Änderungen zu informieren, z. B. wenn eine Pull-Anfrage in einem Repository erstellt wird. In den Benachrichtigungsregeln werden sowohl die Ereignisse als auch das Amazon SNS SNS-Thema festgelegt, das zum Senden von Benachrichtigungen verwendet wird. Weitere Informationen finden Sie unter [Was sind Benachrichtigungen?](#).

Note

Diese Funktion ist in der Region Europa (Mailand) nicht verfügbar. Informationen zur Konfiguration von Benachrichtigungen in dem in dieser Region verfügbaren Erlebnis finden Sie unter [Repository-Benachrichtigungen konfigurieren](#).

Sie können die Konsole oder die verwenden AWS CLI , um Benachrichtigungsregeln für zu erstellen AWS CodeCommit.

So erstellen Sie eine Benachrichtigungsregel (Konsole):

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codecommit/>.
2. Wählen Sie Repositories (Repositorys) und dann ein Repository aus, zu dem Sie Benachrichtigungsregeln hinzufügen möchten.
3. Wählen Sie auf der Repository-Seite Notify (Benachrichtigen) und dann Create notification rule (Benachrichtigungsregel erstellen). Sie können auch zur Seite Settings (Einstellungen) für das Repository navigieren und Create notification rule (Benachrichtigungsregel erstellen) wählen.
4. Geben Sie unter Notification name (Benachrichtigungsname) einen Namen für die Regel ein.
5. Wählen Sie unter Detailtyp die Option Basic aus, wenn Sie möchten, dass nur die Informationen, die Amazon zur Verfügung gestellt wurden, in der Benachrichtigung EventBridge enthalten sind. Wählen Sie Vollständig, wenn Sie Informationen, die Amazon zur Verfügung gestellt wurden, EventBridge und Informationen, die möglicherweise vom CodeCommit oder vom Notification Manager bereitgestellt wurden, einbeziehen möchten.

Weitere Informationen finden Sie unter [Informationen zu Inhalten und Sicherheit von Benachrichtigungen](#).

6. Wählen Sie unter Events that trigger notifications (Ereignisse, die Benachrichtigungen auslösen) die Ereignisse aus, für die Sie Benachrichtigungen senden möchten. Weitere Informationen finden Sie unter [Ereignisse für Benachrichtigungsregeln in Repositories](#).
7. Führen Sie unter Targets (Ziele) einen der folgenden Schritte aus:
 - Wenn Sie bereits eine Ressource für die Verwendung mit Benachrichtigungen konfiguriert haben, wählen Sie unter Zieltyp auswählen entweder Amazon Q Developer in Chat-Anwendungen (Slack) oder SNS-Thema aus. Wählen Sie unter Ziel auswählen den Namen des Clients (für einen in Amazon Q Developer konfigurierten Slack-Client in Chat-Anwendungen) oder den Amazon-Ressourcennamen (ARN) des Amazon SNS-Themas (für Amazon SNS SNS-Themen, die bereits mit der für Benachrichtigungen erforderlichen Richtlinie konfiguriert wurden).
 - Wenn Sie keine Ressource für die Verwendung mit Benachrichtigungen konfiguriert haben, wählen Sie Create target (Ziel erstellen) und dann SNS topic (SNS-Thema) aus. Geben Sie nach codestar-notifications- einen Namen für das Thema an und wählen Sie dann Create (Erstellen).

Note

- Wenn Sie das Amazon-SNS-Thema im Rahmen des Erstellens der Benachrichtigungsregel erstellen, wird die Richtlinie, die es ermöglicht, Ereignisse in dem Thema zu veröffentlichen, für Sie angewendet. Durch die Verwendung eines Themas, das für Benachrichtigungsregeln erstellt wurde, kann sichergestellt werden, dass Sie das Thema nur für die Benutzer abonnieren, die Benachrichtigungen zu dieser Ressource erhalten sollen.
- Sie können im Rahmen der Erstellung einer Benachrichtigungsregel keinen Amazon Q Developer in Chat-Anwendungen Client erstellen. Wenn Sie Amazon Q Developer in Chat-Anwendungen (Slack) wählen, wird Ihnen eine Schaltfläche angezeigt, mit der Sie in Chat-Anwendungen einen Client in Amazon Q Developer konfigurieren können. Wenn Sie diese Option wählen, wird die Amazon Q Developer in Chat-Anwendungskonsole geöffnet. Weitere Informationen finden [Sie unter Konfigurieren von Integrationen zwischen Benachrichtigungen und Amazon Q Developer in Chat-Anwendungen](#).
- Wenn Sie ein vorhandenes Amazon SNS SNS-Thema als Ziel verwenden möchten, müssen Sie die erforderliche Richtlinie für AWS CodeStar Benachrichtigungen

zusätzlich zu allen anderen Richtlinien hinzufügen, die möglicherweise für dieses Thema existieren. Weitere Informationen finden Sie unter [Konfigurieren vorhandener Amazon SNS-Themen für Benachrichtigungen](#) und [Informationen zu Inhalten und Sicherheit von Benachrichtigungen](#).

8. Um die Erstellung der Regel abzuschließen, wählen Sie Submit (Absenden) aus.
9. Sie müssen das Amazon SNS SNS-Thema für die Regel abonnieren, bevor sie Benachrichtigungen erhalten können. Weitere Informationen finden [Sie unter Amazon SNS SNS-Themen, die Ziele sind für Benutzer abonnieren](#). Sie können auch die Integration zwischen Benachrichtigungen und Amazon Q Developer in Chat-Anwendungen einrichten, um Benachrichtigungen an Amazon Chime-Chatrooms zu senden. Weitere Informationen finden [Sie unter Konfiguration der Integration zwischen Benachrichtigungen und Amazon Q Developer in Chat-Anwendungen](#).

So erstellen Sie eine Benachrichtigungsregel (AWS CLI):

1. Führen Sie in einem Terminal oder einer Eingabeaufforderung den Befehl `create-notification rule` aus, um das JSON-Skelett zu generieren:

```
aws codestar-notifications create-notification-rule --generate-cli-skeleton  
> rule.json
```

Sie können die Datei beliebig benennen. In diesem Beispiel heißt die Datei *rule.json*.

2. Öffnen Sie die JSON-Datei in einem Texteditor, und bearbeiten Sie sie so, dass sie die Ressource, die Ereignistypen und das gewünschte Ziel für die Regel enthält. Das folgende Beispiel zeigt eine Benachrichtigungsregel, die **MyNotificationRule** nach einem Repository benannt ist, das *MyDemoRepo* in einem AWS Konto mit der ID *123456789012* benannt ist. Benachrichtigungen mit dem vollständigen Detailtyp werden an ein Amazon SNS SNS-Thema mit dem Namen *MyNotificationTopic* gesendet, wenn Branches und Tags erstellt werden:

```
{  
  "Name": "MyNotificationRule",  
  "EventTypeIds": [  
    "codecommit-repository-branches-and-tags-created"  
  ],  
  "Resource": "arn:aws:codecommit:us-east-1:123456789012:MyDemoRepo",  
  "Targets": [  
    {
```

```
        "TargetType": "SNS",
        "TargetAddress": "arn:aws:sns:us-
east-1:123456789012:MyNotificationTopic"
    },
    "Status": "ENABLED",
    "DetailType": "FULL"
}
```

Speichern Sie die Datei.

3. Führen Sie unter Verwendung der soeben bearbeiteten Datei am Terminal oder in der Befehlszeile erneut den Befehl `create-notification-rule` aus, um die Benachrichtigungsregel zu erstellen:

```
aws codestar-notifications create-notification-rule --cli-input-json
file://rule.json
```

4. Bei Erfolg gibt der Befehl den ARN der Benachrichtigungsregel zurück, der ähnlich wie im Folgenden dargestellt aussieht:

```
{
  "Arn": "arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/
dc82df7a-EXAMPLE"
}
```

Benachrichtigungen ändern oder deaktivieren

Sie können die AWS CodeCommit Konsole verwenden, um die Konfiguration von Benachrichtigungen zu ändern, die vor dem 5. November 2019 erstellt wurden, einschließlich der Ereignistypen, mit denen E-Mails an Benutzer gesendet werden, und des Amazon SNS SNS-Themas, das zum Senden von E-Mails über das Repository verwendet wird. Sie können die CodeCommit-Konsole auch verwenden, um die Liste der E-Mail-Adressen und Endpunkte zu verwalten, die das Thema abonniert haben, oder Benachrichtigungen zu deaktivieren.

So ändern Sie Benachrichtigungseinstellungen

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.

2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys, in dem Sie Benachrichtigungen konfigurieren möchten, die vor dem 5. November 2019 erstellt wurden.
3. Wählen Sie im Navigationsbereich Settings und dann Notifications aus. Wenn ein Banner angezeigt wird, das Sie darüber informiert, dass Sie anstelle von Benachrichtigungsregeln über Benachrichtigungen verfügen, wählen Sie Manage existing notifications (Vorhandene Benachrichtigungen verwalten).
4. Wählen Sie Bearbeiten aus.
5. Nehmen Sie die gewünschten Änderungen vor und wählen Sie dann Save.

Das Deaktivieren von Benachrichtigungen ist eine einfache Methode, den Empfang von E-Mails zu Repository-Ereignissen durch Benutzer vorübergehend zu verhindern.

Um eine Benachrichtigung, die vor dem 5. November 2019 erstellt wurde, dauerhaft zu löschen, führen Sie die unter [Benachrichtigungen löschen](#) beschriebenen Schritte aus.

So deaktivieren Sie Benachrichtigungen

1. [Öffnen Sie die CodeCommit Konsole unter codecommit/home. https://console.aws.amazon.com/codesuite/](https://console.aws.amazon.com/codesuite/)
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, in dem Sie Benachrichtigungen deaktivieren möchten.
3. Wählen Sie im Navigationsbereich Settings und dann Notifications aus. Wählen Sie Manage existing notifications (Vorhandene Benachrichtigungen verwalten) aus.
4. Wählen Sie Edit (Bearbeiten) und deaktivieren Sie unter Event status (Ereignisstatus) die Option Enable notifications (Benachrichtigungen aktivieren) mithilfe des Schiebereglers. Wählen Sie Speichern.
5. Der Ereignisstatus ändert sich in Disabled (Deaktiviert). Es werden keine E-Mails zu Ereignissen gesendet. Wenn Sie Benachrichtigungen deaktivieren, wird die CloudWatch Ereignisregel für das Repository automatisch deaktiviert. Ändern Sie ihren Status in der CloudWatch Ereigniskonsole nicht manuell.

Benachrichtigungen löschen

Wenn Sie Benachrichtigungen, die vor dem 5. November 2019 für ein Repository erstellt wurden, nicht mehr verwenden möchten, können Sie die mit der Benachrichtigung verknüpfte

Amazon CloudWatch Events-Regel löschen. Dadurch wird die Benachrichtigung automatisch gelöscht. Es werden keine Abonnements oder das Amazon SNS SNS-Thema gelöscht, das für Benachrichtigungen verwendet wird.

 Note

Wenn Sie den Namen eines Repositorys über die Konsole ändern, funktionieren Benachrichtigungen, die vor dem 5. November 2019 erstellt wurden, weiterhin ohne Änderung. Wenn Sie den Namen Ihres Repositorys jedoch über die Befehlszeile oder über die API ändern, funktionieren Benachrichtigungen nicht mehr. Die einfachste Methode zum Wiederherstellen von Benachrichtigungen ist, die Benachrichtigungseinstellungen zu löschen und sie dann erneut zu konfigurieren.

So löschen Sie Benachrichtigungseinstellungen

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, aus dem Benachrichtigungen entfernt werden sollen, die vor dem 5. November 2019 erstellt wurden.
3. Wählen Sie im Navigationsbereich Settings und dann Notifications aus. Wenn ein Banner angezeigt wird, das Sie darüber informiert, dass Sie anstelle von Benachrichtigungsregeln über Benachrichtigungen verfügen, wählen Sie Manage existing notifications (Vorhandene Benachrichtigungen verwalten).
4. Kopieren Sie im Feld CloudWatch Event-Regel den Namen der Regel, die für die Benachrichtigung erstellt wurde.
5. Melden Sie sich bei der an AWS Management Console und öffnen Sie die CloudWatch Konsole unter <https://console.aws.amazon.com/cloudwatch/>.
6. Wählen Sie unter Events (Ereignisse) die Option Rules (Regeln). Fügen Sie unter Name den Namen der für die Benachrichtigung erstellten Regel ein. Wählen Sie die Regel und anschließend unter Actions (Aktionen) die Option Delete (Löschen).
7. (Optional) Um das für Benachrichtigungen verwendete Amazon SNS SNS-Thema zu ändern oder zu löschen, nachdem Sie die Benachrichtigungseinstellungen gelöscht haben, gehen Sie zur Amazon SNS SNS-Konsole unter <https://console.aws.amazon.com/sns/v3/home>. Weitere Informationen finden Sie unter [Clean Up](#) im [Amazon Simple Notification Service Developer Guide](#).

Kennzeichnen von Repositorys in AWS CodeCommit

Ein Tag ist eine benutzerdefinierte Attributbezeichnung, die Sie einer AWS Ressource AWS zuweisen oder zuweisen. AWS Tags unterscheiden sich von Git-Tags, die auf Commits angewendet werden können. Jedes AWS Tag besteht aus zwei Teilen:

- einem Tag-Schlüssel (z. B. `CostCenter`, `Environment`, `Project` oder `Secret`). Bei Tag-Schlüsseln wird zwischen Groß- und Kleinschreibung unterschieden.
- einem optionalen Feld, das als Tag-Wert bezeichnet wird (z. B. `111122223333`, `Production` oder ein Team-Name). Ein nicht angegebener Tag-Wert entspricht einer leeren Zeichenfolge. Wie bei Tag-Schlüsseln wird auch bei Tag-Werten zwischen Groß- und Kleinschreibung unterschieden.

Zusammen werden sie als Schlüssel-Wert-Paare bezeichnet. Informationen zu den Limits hinsichtlich der Anzahl von Tags für ein Repository und zu Einschränkungen bei Tag-Schlüsseln und -Werten finden Sie unter [Limits](#).

Mithilfe von Tags können Sie Ihre AWS Ressourcen identifizieren und organisieren. Viele AWS Dienste unterstützen Tagging, sodass Sie Ressourcen aus verschiedenen Diensten dasselbe Tag zuweisen können, um anzuzeigen, dass die Ressourcen miteinander verknüpft sind. Sie können beispielsweise einem CodeCommit Repository dasselbe Tag zuweisen, das Sie einem Amazon S3 S3-Bucket zuweisen. Weitere Informationen zu Tagging-Strategien finden Sie unter [Tagging Resources AWS](#).

In ist CodeCommit die primäre Ressource ein Repository. Sie können die CodeCommit Konsole, oder verwenden AWS CLI CodeCommit APIs, AWS SDKs um Tags für ein Repository hinzuzufügen, zu verwalten und zu entfernen. Neben der Identifizierung, Organisation und Nachverfolgung Ihres Repositorys mithilfe von Tags können Sie mithilfe von Tags in IAM-Richtlinien steuern, wer Ihr Repository aufrufen und mit ihm interagieren kann. Beispiele für Tag-basierte Zugriffsrichtlinien finden Sie unter [Beispiel 5: Aktionen für Repositorys mit Tags verweigern oder zulassen](#).

Themen

- [Hinzufügen eines Tags zu einem Repository](#)
- [Anzeigen von Tags für ein Repository](#)
- [Bearbeiten von Tags für ein Repository](#)
- [Entfernen Sie ein Tag aus einem Repository](#)

Hinzufügen eines Tags zu einem Repository

Das Hinzufügen von Tags zu einem Repository kann Ihnen dabei helfen, Ihre AWS Ressourcen zu identifizieren und zu organisieren und den Zugriff darauf zu verwalten. Zuerst fügen Sie ein oder mehrere Tags (Schlüssel-Wert-Paare) zu einem Repository hinzu. Denken Sie daran, dass es hinsichtlich der Anzahl von Tags für ein Repository Grenzen gibt. Es gibt Einschränkungen im Hinblick auf die Zeichen, die Sie in die Felder für Schlüssel und Wert eingeben können. Weitere Informationen finden Sie unter [Limits](#). Sobald Sie über Tags verfügen, können Sie IAM-Richtlinien erstellen, um den Zugriff auf das Repository auf der Grundlage dieser Tags zu verwalten. Sie können die CodeCommit Konsole oder die verwenden AWS CLI , um Tags zu einem Repository hinzuzufügen.

Important

Das Hinzufügen von Tags zu einem Repository kann Auswirkungen auf den Zugriff auf dieses Repository haben. Bevor Sie einem Repository ein Tag hinzufügen, sollten Sie alle IAM-Richtlinien überprüfen, die möglicherweise Tags verwenden, um den Zugriff auf Ressourcen wie Repositories zu steuern. Beispiele für Tag-basierte Zugriffsrichtlinien finden Sie unter [Beispiel 5: Aktionen für Repositories mit Tags verweigern oder zulassen](#).

Weitere Informationen zum Hinzufügen von Tags zu einem Repository während der Erstellung finden Sie unter [Erstellen Sie ein Repository \(Konsole\)](#).

Themen

- [Fügen Sie einem Repository \(Konsole\) ein Tag hinzu](#)
- [Fügt einem Repository ein Tag hinzu \(\)AWS CLI](#)

Fügen Sie einem Repository (Konsole) ein Tag hinzu

Sie können die CodeCommit Konsole verwenden, um einem CodeCommit Repository ein oder mehrere Tags hinzuzufügen.

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories (Repositories) den Namen des Repositories aus, zu dem Sie Tags hinzufügen möchten.

3. Wählen Sie im Navigationsbereich Settings (Einstellungen). Wählen Sie Repository tags (Repository-Tags) aus.
4. Wenn noch keine Tags zum Repository hinzugefügt wurden, wählen Sie Add tag (Tag hinzufügen) aus. Wählen Sie andernfalls Edit (Bearbeiten) und Add tag (Tag hinzufügen) aus.
5. Geben Sie für Schlüssel einen Namen für das Tag ein. Sie können einen optionalen Wert für das Tag unter Wert hinzufügen.

Developer Tools **CodeCommit**

▼ Source • CodeCommit

- Getting started
- Repositories
- Code
- Pull requests
- Commits
- Branches
- Tags
- Settings**
- Build • CodeBuild
- Deploy • CodeDeploy
- Pipeline • CodePipeline
- Settings

Q Go to resource

Feedback

Developer Tools > CodeCommit > Repositories > MyDemoRepo > Settings > Repository tags

Edit Repository Tags

Repository Tags Info

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key	Value - optional	
Team	Saanvi	Remove tag

Add tag

Cancel Submit

6. (Optional) Zum Hinzufügen eines weiteren Tags wählen Sie Tag hinzufügen erneut aus.
7. Wenn Sie mit dem Hinzufügen von Tags fertig sind, klicken Sie auf Submit (Übermitteln).

Fügt einem Repository ein Tag hinzu (AWS CLI)

Gehen Sie wie folgt vor AWS CLI, um ein Tag zu einem CodeCommit Repository hinzuzufügen. Informationen darüber, wie Sie beim Erstellen eines Repository ein Tag hinzufügen können, finden Sie unter [Erstellen Sie ein Repository \(AWS CLI\)](#).

Bei diesen Schritten wird davon ausgegangen, dass Sie bereits eine aktuelle Version der AWS CLI installiert oder eine Aktualisierung auf die aktuelle Version vorgenommen haben. Weitere Informationen finden Sie unter [Installieren der AWS Command Line Interface](#).

Führen Sie am Terminal oder über die Befehlszeile den Befehl `tag-resource` aus und geben Sie dabei den Amazon-Ressourcennamen (ARN) des Repositorys an, für den Sie Tags hinzufügen möchten, sowie den Schlüssel und Wert des hinzuzufügenden Tags. Sie können mehrere Tags zu einem

Repository hinzufügen. Um beispielsweise ein Repository *MyDemoRepo* mit zwei Tags zu taggen, einem Tag-Schlüssel *Status* mit dem Tag-Wert von *Secret* und einem Tag-Schlüssel *Team* mit dem Tag-Wert von *Saanvi*:

```
aws codecommit tag-resource --resource-arn arn:aws:codecommit:us-west-2:111111111111:MyDemoRepo --tags Status=Secret,Team=Saanvi
```

Bei erfolgreicher Ausführung gibt dieser Befehl nichts zurück.

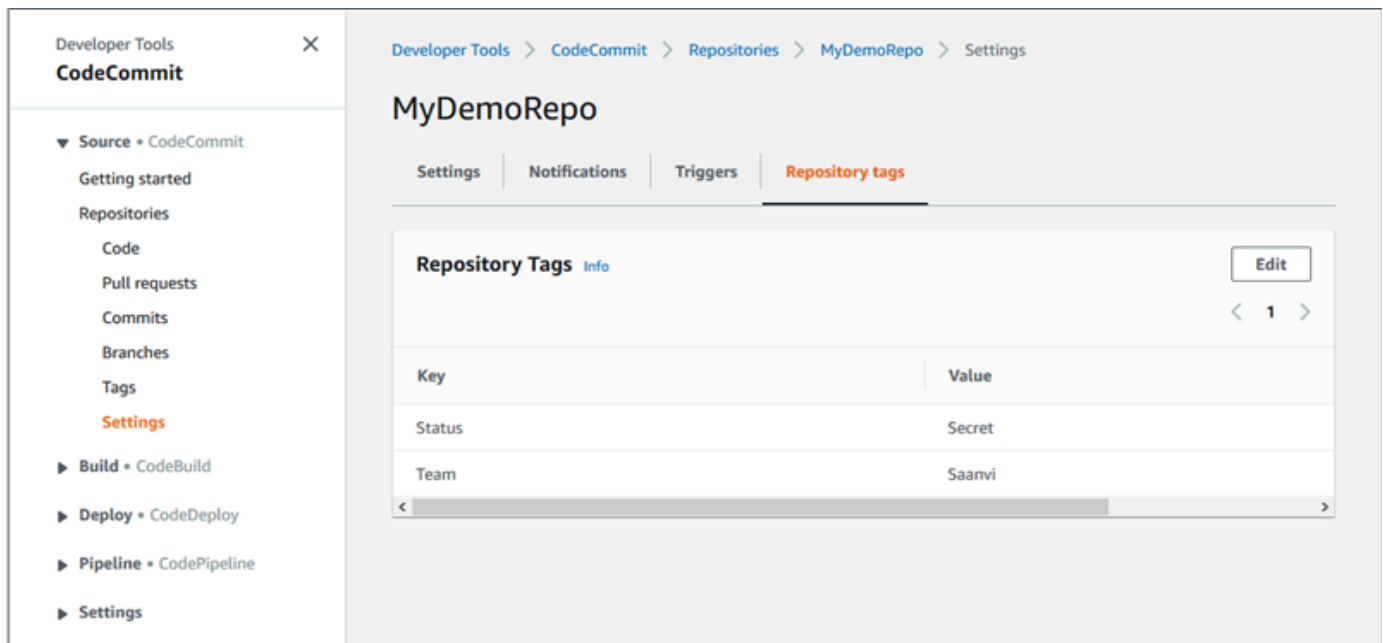
Anzeigen von Tags für ein Repository

Mithilfe von Tags können Sie Ihre AWS Ressourcen identifizieren und organisieren und den Zugriff darauf verwalten. Weitere Informationen zu Tagging-Strategien finden Sie unter Ressourcen [taggen AWS](#). Beispiele für Tag-basierte Zugriffsrichtlinien finden Sie unter [Beispiel 5: Aktionen für Repositorys mit Tags verweigern oder zulassen](#).

Tags für ein Repository anzeigen (Konsole)

Sie können die CodeCommit Konsole verwenden, um die mit einem CodeCommit Repository verknüpften Tags anzuzeigen.

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, in dem Sie Tags anzeigen möchten.
3. Wählen Sie im Navigationsbereich Settings (Einstellungen). Wählen Sie Repository tags (Repository-Tags) aus.



Tags für ein Repository anzeigen (AWS CLI)

Gehen Sie wie folgt vor AWS CLI, um die AWS Tags für ein CodeCommit Repository anzuzeigen. Wenn keine Tags hinzugefügt wurden, ist die zurückgegebene Liste leer.

Führen Sie am Terminal oder über die Befehlszeile den Befehl `list-tags-for-resource` aus. Um beispielsweise eine Liste der Tag-Schlüssel und Tag-Werte für ein Repository mit dem Namen *MyDemoRepo* mit dem ARN *arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo* anzuzeigen:

```
aws codecommit list-tags-for-resource --resource-arn arn:aws:codecommit:us-west-2:111111111111:MyDemoRepo
```

Bei erfolgreicher Ausführung gibt dieser Befehl etwa wie folgt aussehende Informationen zurück:

```
{
  "tags": {
    "Status": "Secret",
    "Team": "Saanvi"
  }
}
```

Bearbeiten von Tags für ein Repository

Sie können den Wert für ein Tag ändern, das mit dem Repository verknüpft ist. Sie können auch den Namen des Schlüssels ändern. Dies entspricht dem Entfernen des aktuellen Tags und dem Hinzufügen eines anderen Tags mit dem neuen Namen und demselben Wert wie dem des anderen. Denken Sie daran, dass es hinsichtlich der Zeichen, die Sie in die Felder für Schlüssel und Wert eingeben können, Einschränkungen gibt. Weitere Informationen finden Sie unter [Limits](#).

Important

Das Bearbeiten von Tags für ein Repository kann Auswirkungen auf den Repository-Zugriff haben. Bevor Sie den Namen (Schlüssel) oder Wert eines Tags für ein Repository bearbeiten, sollten Sie alle IAM-Richtlinien überprüfen, die den Schlüssel oder Wert für ein Tag möglicherweise verwenden, um den Zugriff auf Ressourcen wie Repositories zu steuern. Beispiele für Tag-basierte Zugriffsrichtlinien finden Sie unter [Beispiel 5: Aktionen für Repositories mit Tags verweigern oder zulassen](#).

Bearbeiten Sie ein Tag für ein Repository (Konsole)

Sie können die CodeCommit Konsole verwenden, um die mit einem CodeCommit Repository verknüpften Tags zu bearbeiten.

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories (Repositories) den Namen des Repositories aus, für das Sie Tags bearbeiten möchten.
3. Wählen Sie im Navigationsbereich Settings (Einstellungen). Wählen Sie Repository tags (Repository-Tags) aus.
4. Wählen Sie Bearbeiten aus.

5.

Developer Tools > CodeCommit > Repositories > MyDemoRepo > Settings > Repository tags

Edit Repository Tags

Repository Tags [Info](#)

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs. Each resource can have up to 50 tags. Keys cannot begin with "AWS:".

Key	Value - optional	
Status	Secret	Remove tag
Team	Saanvi	Remove tag

Führen Sie eine der folgenden Aktionen aus:

- Zum Ändern des Tags geben Sie einen neuen Namen unter Key (Schlüssel) ein. Das Ändern des Namens eines Tags entspricht dem Entfernen eines Tags und Hinzufügen eines neuen Tags mit dem neuen Schlüsselnamen.
- Geben Sie zum Ändern des Werts eines Tags einen neuen Wert ein. Wenn Sie den Wert in „kein“ ändern möchten, löschen Sie den aktuellen Wert und lassen das Feld leer.

6. Wenn Sie mit dem Bearbeiten der Tags fertig sind, wählen Sie Submit (Übermitteln) aus.

Bearbeiten Sie die Tags für ein Repository (AWS CLI)

Gehen Sie wie folgt vor, AWS CLI um mit dem ein Tag für ein CodeCommit Repository zu aktualisieren. Sie können den Wert für einen vorhandenen Schlüssel ändern oder einen anderen Schlüssel hinzufügen.

Führen Sie im Terminal oder in der Befehlszeile den Befehl `tag-resource` aus und geben Sie dabei den Amazon-Ressourcennamen (ARN) des Repositories, für das Sie ein Tag aktualisieren möchten, sowie den Schlüssel und den Wert des Tags an:

```
aws codecommit tag-resource --resource-arn arn:aws:codecommit:us-west-2:111111111111:MyDemoRepo --tags Team=Li
```

Entfernen Sie ein Tag aus einem Repository

Sie können ein oder mehrere Tags entfernen, die mit einem Repository verknüpft sind. Durch das Entfernen eines Tags wird das Tag nicht aus anderen AWS Ressourcen gelöscht, die mit diesem Tag verknüpft sind.

Important

Das Entfernen von Tags für ein Repository kann Auswirkungen auf den Repository-Zugriff haben. Bevor Sie ein Tag aus einem Repository entfernen, sollten Sie alle IAM-Richtlinien überprüfen, die den Schlüssel oder Wert für ein Tag verwenden könnten, um den Zugriff auf Ressourcen wie Repositorys zu steuern. Beispiele für Tag-basierte Zugriffsrichtlinien finden Sie unter [Beispiel 5: Aktionen für Repositorys mit Tags verweigern oder zulassen](#).

Entfernen Sie ein Tag aus einem Repository (Konsole)

Sie können die CodeCommit Konsole verwenden, um die Zuordnung zwischen einem Tag und einem CodeCommit Repository zu entfernen.

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, aus dem Sie Tags entfernen möchten.
3. Wählen Sie im Navigationsbereich Settings (Einstellungen). Wählen Sie Repository tags (Repository-Tags) aus.
4. Wählen Sie Bearbeiten aus.
5. Suchen Sie die Tags, die Sie entfernen möchten, und wählen Sie dann Remove tag (Tag entfernen) aus.
6. Wenn Sie die Tags entfernt haben, klicken Sie auf Submit (Übermitteln).

Entferne ein Tag aus einem Repository ()AWS CLI

Gehen Sie wie folgt vor, AWS CLI um mit dem ein Tag aus einem CodeCommit Repository zu entfernen. Wenn ein Tag entfernt wird, wird es nicht gelöscht, sondern lediglich die Zuordnung zwischen dem Tag und dem Repository entfernt.

Note

Wenn Sie ein CodeCommit Repository löschen, werden alle Tag-Verknüpfungen aus dem gelöschten Repository entfernt. Sie müssen vor dem Löschen eines Repositories keine Tags entfernen.

Führen Sie am Terminal oder über die Befehlszeile den Befehl `untag-resource` aus und geben Sie dabei den Amazon-Ressourcennamen (ARN) des Repositories an, für den Sie Tags hinzufügen möchten, sowie den Tag-Schlüssel des zu entfernenden Tags. Um beispielsweise ein Tag in einem Repository zu entfernen, das *MyDemoRepo* mit dem Tag-Schlüssel benannt ist *Status*:

```
aws codecommit untag-resource --resource-arn arn:aws:codecommit:us-west-2:111111111111:MyDemoRepo --tag-keys Status
```

Bei erfolgreicher Ausführung gibt dieser Befehl nichts zurück. Um die dem Repository zugeordneten Tags zu überprüfen, führen Sie den Befehl `list-tags-for-resource` aus.

Trigger für ein AWS CodeCommit Repository verwalten

Sie können ein CodeCommit Repository so konfigurieren, dass Code-Pushs oder andere Ereignisse Aktionen auslösen, z. B. das Senden einer Benachrichtigung von Amazon Simple Notification Service (Amazon SNS) oder das Aufrufen einer Funktion in AWS Lambda. Sie können bis zu 10 Trigger für jedes Repository erstellen. CodeCommit

Auslöser werden häufig für folgende Aktionen konfiguriert:

- Versenden von E-Mails an Benutzer mit Abonnement bei jedem Push-Vorgang auf das Repository
- Benachrichtigen eines externen Build-Systems, einen Build zu starten, nachdem der Haupt-Branch des Repositories verschoben wurde

Szenarien wie die Benachrichtigung eines externen Build-Systems erfordern das Schreiben einer Lambda-Funktion, um mit anderen Anwendungen zu interagieren. Das E-Mail-Szenario erfordert lediglich die Erstellung eines Amazon SNS SNS-Themas.

In diesem Thema erfahren Sie, wie Sie Berechtigungen festlegen, mit denen CodeCommit Aktionen in Amazon SNS und Lambda ausgelöst werden können. Außerdem enthält es Links zu Beispielen zum Erstellen, Bearbeiten, Testen und Löschen von Auslösern.

Themen

- [Erstellen Sie die Ressource und fügen Sie Berechtigungen hinzu für CodeCommit](#)
- [Beispiel: Einen AWS CodeCommit Trigger für ein Amazon SNS SNS-Thema erstellen](#)
- [Beispiel: Erstellen Sie einen AWS CodeCommit Trigger für eine Funktion AWS Lambda](#)
- [Beispiel: Erstellen Sie einen Trigger AWS CodeCommit für eine bestehende AWS Lambda Funktion](#)
- [Trigger für ein Repository bearbeiten AWS CodeCommit](#)
- [Testen Sie Trigger für ein AWS CodeCommit Repository](#)
- [Trigger aus einem AWS CodeCommit Repository löschen](#)

Erstellen Sie die Ressource und fügen Sie Berechtigungen hinzu für CodeCommit

Sie können Amazon SNS SNS-Themen und Lambda-Funktionen mit Triggern integrieren CodeCommit, aber Sie müssen zuerst Ressourcen mit einer Richtlinie erstellen und dann konfigurieren, CodeCommit die die Berechtigungen für die Interaktion mit diesen Ressourcen gewährt. Sie müssen die Ressource in derselben Umgebung AWS-Region wie das CodeCommit Repository erstellen. Wenn sich das Repository beispielsweise in USA Ost (Ohio) (us-east-2) befindet, muss sich das Amazon SNS SNS-Thema oder die Lambda-Funktion in USA Ost (Ohio) befinden.

- Für Amazon SNS SNS-Themen müssen Sie keine zusätzlichen IAM-Richtlinien oder -Berechtigungen konfigurieren, wenn das Amazon SNS SNS-Thema mit demselben Konto wie das Repository erstellt wurde. CodeCommit Sie können den CodeCommit Auslöser erstellen, sobald Sie das Amazon SNS SNS-Thema erstellt und abonniert haben.
- Weitere Informationen zum Erstellen von Themen in Amazon SNS finden Sie in der [Amazon SNS SNS-Dokumentation](#).

- Informationen zur Verwendung von Amazon SNS zum Senden von Nachrichten an Amazon SQS SQS-Warteschlangen finden Sie unter [Senden von Nachrichten an Amazon SQS SQS-Warteschlangen im Amazon SNS SNS-Entwicklerhandbuch](#).
- Informationen zur Verwendung von Amazon SNS zum Aufrufen einer Lambda-Funktion finden Sie unter [Aufrufen von Lambda-Funktionen](#) im Amazon SNS SNS-Entwicklerhandbuch.
- Wenn Sie Ihren Auslöser so konfigurieren möchten, dass er ein Amazon SNS SNS-Thema in einem anderen AWS Konto verwendet, müssen Sie dieses Thema zunächst mit einer Richtlinie konfigurieren, die die Veröffentlichung CodeCommit zu diesem Thema ermöglicht. Weitere Informationen finden Sie unter [Beispiel 1: Erstellen Sie eine Richtlinie, die den kontoübergreifenden Zugriff auf ein Amazon SNS SNS-Thema ermöglicht](#).
- Sie können Lambda-Funktionen konfigurieren, indem Sie den Trigger in der Lambda-Konsole als Teil der Funktion erstellen. Dies ist die einfachste Methode, da Trigger, die in der Lambda-Konsole erstellt wurden, automatisch die Berechtigungen enthalten, die für CodeCommit den Aufruf der Lambda-Funktion erforderlich sind. Wenn Sie den Trigger in erstellen CodeCommit, müssen Sie eine Richtlinie angeben, die das Aufrufen der CodeCommit Funktion ermöglicht. Weitere Informationen erhalten Sie unter [Erstellen Sie einen Trigger für eine bestehende Lambda-Funktion](#) und [Beispiel 3: Erstellen Sie eine Richtlinie für die AWS Lambda Integration mit einem Trigger CodeCommit](#).

Beispiel: Einen AWS CodeCommit Trigger für ein Amazon SNS SNS-Thema erstellen

Sie können einen Trigger für ein CodeCommit Repository erstellen, sodass Ereignisse in diesem Repository Benachrichtigungen von einem Amazon Simple Notification Service (Amazon SNS) -Thema auslösen. Möglicherweise möchten Sie einen Auslöser für ein Amazon SNS SNS-Thema erstellen, damit Benutzer Benachrichtigungen über Repository-Ereignisse wie das Löschen von Branches abonnieren können. Sie können auch die Vorteile der Integration von Amazon SNS SNS-Themen mit anderen Diensten wie Amazon Simple Queue Service (Amazon SQS) und nutzen. AWS Lambda

Note

- Sie müssen den Auslöser auf ein vorhandenes Amazon SNS SNS-Thema verweisen, bei dem es sich um die Aktion handelt, die als Reaktion auf Repository-Ereignisse ergriffen

wurde. Weitere Informationen zum Erstellen und Abonnieren von Amazon SNS SNS-Themen finden Sie unter [Erste Schritte mit Amazon Simple Notification Service](#).

- Amazon SNS FIFO-Themen (first in, first out) werden für CodeCommit Auslöser nicht unterstützt.

Themen

- [Einen Auslöser für ein Amazon SNS SNS-Thema für ein CodeCommit Repository \(Konsole\) erstellen](#)
- [Einen Auslöser für ein Amazon SNS SNS-Thema für ein CodeCommit Repository erstellen \(\)AWS CLI](#)

Einen Auslöser für ein Amazon SNS SNS-Thema für ein CodeCommit Repository (Konsole) erstellen

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories (Repositorys) das Repository aus, in dem Sie Auslöser für Repository-Ereignisse erstellen möchten.
3. Wählen Sie im Navigationsbereich für das Repository Settings (Einstellungen) und anschließend Triggers (Auslöser).
4. Wählen Sie Create trigger (Auslöser erstellen) aus und gehen Sie folgendermaßen vor:
 - Geben Sie im Feld Triggernamen einen Namen für den Trigger ein (z. B.). *MyFirstTrigger*
 - Wählen Sie unter Ereignisse die Repository-Ereignisse aus, die das Amazon SNS SNS-Thema zum Senden von Benachrichtigungen veranlassen.

Wenn Sie All repository events wählen, können Sie keine anderen Ereignisse auswählen. Wenn Sie nur einen Teil der Ereignisse auswählen möchten, löschen Sie All repository events und wählen Sie dann die gewünschten Ereignisse in der Liste aus. Wenn der Auslöser beispielsweise nur ausgeführt werden soll, wenn ein Benutzer in dem CodeCommit-Repository ein Tag oder einen Branch erstellt, entfernen Sie All repository events (Alle Repository-Ereignisse) und wählen Sie anschließend Create branch or tag (Verzweigung oder Tag erstellen) aus.

- Wenn Sie möchten, dass der Auslöser für alle Branches des Repositorys unter Branches gilt, lassen Sie die Auswahl leer, da diese Standardoption den Auslöser automatisch auf alle Branches anwendet. Wenn Sie möchten, dass dieser Auslöser nur für bestimmte Branches gilt, wählen Sie bis zu 10 Branch-Namen aus der Liste der Repository-Branches aus.
- Wählen Sie unter Wählen Sie den zu verwendenden Service die Option Amazon SNS aus.
- Wählen Sie in Amazon SNS einen Themennamen aus der Liste aus oder geben Sie den ARN für das Thema ein.

 Note

Amazon SNS FIFO-Themen (first in, first out) werden für CodeCommit Auslöser nicht unterstützt. Sie müssen ein Amazon SNS SNS-Thema auswählen, dessen Typ auf Standard gesetzt ist.

- Geben Sie unter Benutzerdefinierte Daten alle optionalen Informationen ein, die Sie in die vom Amazon SNS SNS-Thema gesendete Benachrichtigung aufnehmen möchten (z. B. einen IRC-Kanalnamen, den Entwickler verwenden, wenn sie über Entwicklungen in diesem Repository sprechen). Dies ist ein Zeichenfolgenfeld. Es kann nicht zur Übergabe dynamischer Parameter verwendet werden.
5. (Optional) Wählen Sie Test trigger (Auslöser testen). Mit diesem Schritt können Sie überprüfen, ob der Zugriff zwischen CodeCommit und dem Amazon SNS SNS-Thema korrekt konfiguriert wurde. Es verwendet das Amazon SNS SNS-Thema, um eine Testbenachrichtigung mit Daten aus Ihrem Repository zu senden, sofern verfügbar. Falls keine echten Daten verfügbar sind, enthält die Testbenachrichtigung Beispieldaten.
 6. Wählen Sie Create trigger (Auslöser erstellen), um die Erstellung des Auslösers abzuschließen.

Einen Auslöser für ein Amazon SNS SNS-Thema für ein CodeCommit Repository erstellen ()AWS CLI

Sie können auch die Befehlszeile verwenden, um einen Trigger für ein Amazon SNS SNS-Thema als Reaktion auf CodeCommit Repository-Ereignisse zu erstellen, z. B. wenn jemand einen Commit in Ihr Repository überträgt.

Um einen Auslöser für ein Amazon SNS SNS-Thema zu erstellen

1. Öffnen Sie einen Texteditor und erstellen Sie eine JSON-Datei mit folgenden Angaben:

- Der Name des Amazon SNS SNS-Themas.

Note

Amazon SNS FIFO-Themen (first in, first out) werden für CodeCommit Auslöser nicht unterstützt. Sie müssen ein Amazon SNS SNS-Thema auswählen, dessen Typ auf Standard gesetzt ist.

- Repository und Branches, die mit diesem Auslöser überwacht werden sollen. (Wenn keine Branches angegeben werden, gilt der Auslöser für alle Branches im Repository.)
- Ereignisse, die diesen Auslöser aktivieren.

Speichern Sie die Datei.

Um beispielsweise einen Trigger für ein Repository mit dem Namen zu erstellen *MyDemoRepo*, der alle Repository-Ereignisse in einem Amazon SNS SNS-Thema veröffentlicht, das *MySNSTopic* nach zwei Branches benannt ist, *main* und *preprod*:

```
{
  "repositoryName": "MyDemoRepo",
  "triggers": [
    {
      "name": "MyFirstTrigger",
      "destinationArn": "arn:aws:sns:us-east-2:111122223333:MySNSTopic",
      "customData": "",
      "branches": [
        "main", "preprod"
      ],
      "events": [
        "all"
      ]
    }
  ]
}
```

In der JSON-Datei muss für jeden Auslöser eines Repositorys ein Auslöserblock vorhanden sein. Wenn Sie für ein Repository mehr als einen Auslöser erstellen möchten, fügen Sie mehrere Auslöserblöcke zur JSON-Datei hinzu. Beachten Sie, dass alle in dieser Datei erstellten Auslöser für das angegebene Repository gelten. Es ist nicht möglich, Auslöser für verschiedene Repositorys in einer JSON-Datei zu erstellen. Wenn Sie beispielsweise zwei Auslöser für ein Repository erstellen möchten, können Sie eine JSON-Datei mit zwei Auslöserblöcken erstellen. Im folgenden Beispiel werden für den zweiten Auslöser keine Branches angegeben, daher gilt dieser Auslöser für alle Branches:

```
{
  "repositoryName": "MyDemoRepo",
  "triggers": [
    {
      "name": "MyFirstTrigger",
      "destinationArn": "arn:aws:sns:us-east-2:111122223333:MySNSTopic",
      "customData": "",
      "branches": [
        "main", "preprod"
      ],
      "events": [
        "all"
      ]
    },
    {
      "name": "MySecondTrigger",
      "destinationArn": "arn:aws:sns:us-east-2:111122223333:MySNSTopic2",
      "customData": "",
      "branches": [],
      "events": [
        "updateReference", "deleteReference"
      ]
    }
  ]
}
```

Sie können Auslöser für bestimmte Ereignisse erstellen, z. B. wenn ein Commit per Push an ein Repository übertragen wird. Zu den Ereignistypen zählen:

- `all` für alle Ereignisse im angegebenen Repository und den angegebenen Branches

- `updateReference` für Fälle, in denen Commits per Push an das angegebene Repository und die angegebenen Branches übertragen werden
- `createReference` für Fälle, in denen in dem angegebenen Repository ein neuer Branch oder ein neues Tag erstellt wird
- `deleteReference` für Fälle, in denen in dem angegebenen Repository ein Branch oder ein Tag gelöscht wird

 Note

Sie können in einem Auslöser mehrere Ereignistypen verwenden. Wenn Sie `all` angeben, können Sie allerdings keine anderen Ereignisse angeben.

Geben Sie im Terminal oder in der Eingabeaufforderung `aws codecommit put-repository-triggers help` ein, um eine vollständige Liste der gültigen Ereignistypen anzuzeigen.

Darüber hinaus können Sie in `customData` eine Zeichenfolge einschließen (z. B. den Namen eines IRC-Kanal, den Entwickler verwenden, wenn Sie die Entwicklung in diesem Repository besprechen). Dies ist ein Zeichenfolgenfeld. Es kann nicht zur Übergabe dynamischer Parameter verwendet werden. Diese Zeichenfolge wird als Attribut an das CodeCommit JSON angehängt, das als Antwort auf den Trigger zurückgegeben wird.

2. (Optional) Führen Sie an einem Terminal oder an einer Eingabeaufforderung den Befehl `test-repository-triggers` aus. Dieser Test verwendet Beispieldaten aus dem Repository (oder generiert Beispieldaten, falls keine Daten verfügbar sind), um eine Benachrichtigung an die Abonnenten des Amazon SNS SNS-Themas zu senden. Mit dem Folgenden wird beispielsweise getestet, ob der JSON-Code in der Trigger-Datei mit dem Namen gültig `trigger.json` ist und im Amazon SNS SNS-Thema veröffentlicht werden CodeCommit kann:

```
aws codecommit test-repository-triggers --cli-input-json file://trigger.json
```

Bei erfolgreicher Ausführung gibt dieser Befehl etwa wie folgt aussehende Informationen zurück:

```
{
  "successfulExecutions": [
    "MyFirstTrigger"
  ],
```

```
"failedExecutions": []  
}
```

3. Führen Sie in einem Terminal oder einer Eingabeaufforderung den Befehl `put-repository-triggers` aus, um den Auslöser in CodeCommit zu erstellen. Um beispielsweise eine JSON-Datei mit dem Namen *trigger.json* zur Erstellung des Triggers zu verwenden:

```
aws codecommit put-repository-triggers --cli-input-json  
file://trigger.json
```

Die vom Befehl zurückgegebene [Konfigurations-ID](#) sollte wie folgt aussehen:

```
{  
  "configurationId": "0123456-I-AM-AN-EXAMPLE"  
}
```

4. Führen Sie zum Anzeigen der Konfiguration des Auslösers den Befehl `get-repository-triggers` aus und geben Sie dabei den Namen des Repositorys an:

```
aws codecommit get-repository-triggers --repository-name MyDemoRepo
```

Dieser Befehl gibt die Struktur aller Auslöser zurück, die für das Repository konfiguriert sind, ähnlich folgendem Beispiel:

```
{  
  "configurationId": "0123456-I-AM-AN-EXAMPLE",  
  "triggers": [  
    {  
      "events": [  
        "all"  
      ],  
      "destinationArn": "arn:aws:sns:us-east-2:111122223333:MySNSTopic",  
      "branches": [  
        "main",  
        "preprod"  
      ],  
      "name": "MyFirstTrigger",  
      "customData": "Project ID 12345"  
    }  
  ]  
}
```

- Erstellen Sie einen Commit und übertragen Sie ihn per Push an das Repository, für das Sie den Auslöser konfiguriert haben, um die Funktion des Auslösers zu prüfen. Sie sollten eine Antwort vom Amazon SNS SNS-Thema sehen. Wenn Sie beispielsweise das Amazon SNS-Thema so konfiguriert haben, dass eine E-Mail gesendet wird, sollte in dem E-Mail-Konto, das das Thema abonniert hat, eine E-Mail von Amazon SNS angezeigt werden.

Im Folgenden finden Sie ein Beispiel für die Ausgabe einer E-Mail, die von Amazon SNS als Antwort auf einen Push an ein CodeCommit Repository gesendet wurde:

```
{
  "Records": [
    {
      "awsRegion": "us-east-2",
      "codecommit": {
        "references": [
          {
            "commit": "317f8570EXAMPLE",
            "created": true,
            "ref": "refs/heads/NewBranch"
          },
          {
            "commit": "4c925148EXAMPLE",
            "ref": "refs/heads/preprod",
          }
        ]
      },
      "eventId": "11111-EXAMPLE-ID",
      "eventName": "ReferenceChange",
      "eventPartNumber": 1,
      "eventSource": "aws:codecommit",
      "eventSourceARN": "arn:aws:codecommit:us-east-2:111122223333:MyDemoRepo",
      "eventTime": "2016-02-09T00:08:11.743+0000",
      "eventTotalParts": 1,
      "eventTriggerConfigId": "0123456-I-AM-AN-EXAMPLE",
      "eventTriggerName": "MyFirstTrigger",
      "eventVersion": "1.0",
      "customData": "Project ID 12345",
      "userIdentityARN": "arn:aws:iam::111122223333:user/JaneDoe-CodeCommit",
    }
  ]
}
```

Beispiel: Erstellen Sie einen AWS CodeCommit Trigger für eine Funktion AWS Lambda

Sie können einen Trigger für ein CodeCommit Repository erstellen, sodass Ereignisse im Repository eine Lambda-Funktion aufrufen. In diesem Beispiel erstellen Sie eine Lambda-Funktion, die die URL zurückgibt, die zum Klonen des Repositories in ein CloudWatch Amazon-Protokoll verwendet wurde.

Themen

- [So erstellen Sie die Lambda-Funktion:](#)
- [Den Trigger für die Lambda-Funktion im AWS CodeCommit Repository anzeigen](#)

So erstellen Sie die Lambda-Funktion:

Wenn Sie die Lambda-Konsole verwenden, um die Funktion zu erstellen, können Sie auch einen CodeCommit Trigger für die Lambda-Funktion erstellen. Die folgenden Schritte beinhalten ein Beispiel für eine Lambda-Funktion. Das Beispiel ist in zwei Sprachen verfügbar: JavaScript und Python. Die Funktion gibt den zum Klonen eines Repositories URLs verwendeten Wert in ein CloudWatch Protokoll zurück.

So erstellen Sie eine Lambda-Funktion mithilfe eines Lambda-Blueprints

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Lambda Konsole unter: <https://console.aws.amazon.com/lambda/>
2. Wählen Sie auf der Seite Lambda-Funktionen die Option Funktion erstellen aus. (Wenn Sie Lambda noch nicht verwendet haben, wählen Sie Jetzt starten.)
3. Wählen Sie auf der Seite Create function die Option Author from scratch. Geben Sie im Feld Funktionsname beispielsweise *MyLambdaFunctionforCodeCommit* einen Namen für die Funktion ein. Wählen Sie unter Runtime (Laufzeit) die Sprache aus, die Sie verwenden möchten, um Ihre Funktion zu schreiben, und wählen Sie dann Create function (Funktion erstellen).
4. Wählen Sie auf der Registerkarte Configuration (Konfiguration) die Option Add trigger (Auslöser hinzufügen) aus.
5. Wählen Sie in der Trigger-Konfiguration eine Option CodeCommit aus der Dropdownliste Dienste aus.

Lambda > Add trigger

Add trigger

Trigger configuration



CodeCommit

aws

developer-tools

git



Repository name

Select the repository to add a trigger to.

MyDemoRepo



Trigger name

Provide a name for the trigger that will invoke this function.

MyLambdaFunctionTrigger

Events

Choose one or more events to listen for. If you choose "All repository events", you cannot choose other event types.

Push to existing branch X

Branch names

This trigger will be configured for all repository branches and tags by default. For a more specific configuration, choose up to 10 branches. If you choose "All branches", you cannot choose specific branches.



All branches X

Custom data - optional

Custom data is additional contextual information used to distinguish this trigger from other triggers that run for the same event, refer to external resources, or group triggers from different repositories. For example, you could include the channel ID # for a chat room used by your team to collaborate on development.

#

Lambda will add the necessary permissions for AWS CodeCommit to invoke your Lambda function from this trigger.

[Learn more](#) about the Lambda permissions model.

Cancel

Add

- Wählen Sie unter Repository-Name den Namen des Repositorys aus, für das Sie einen Trigger konfigurieren möchten, der die Lambda-Funktion als Reaktion auf Repository-Ereignisse verwendet.

- Geben Sie im Feld Triggernamen einen Namen für den Trigger ein (z. *B. `MyLambdaFunctionTrigger`*).
- Wählen Sie unter Ereignissen die Repository-Ereignisse aus, die die Lambda-Funktion auslösen. Wenn Sie All repository events wählen, können Sie keine anderen Ereignisse auswählen. Wenn Sie nur einen Teil der Ereignisse auswählen möchten, löschen Sie All repository events und wählen Sie dann die gewünschten Ereignisse in der Liste aus. Wenn Sie beispielsweise möchten, dass der Trigger nur ausgeführt wird, wenn ein Benutzer ein Tag oder einen Branch im AWS CodeCommit Repository erstellt, entfernen Sie Alle Repository-Ereignisse und wählen Sie dann Branch oder Tag erstellen aus.
- Wenn der Auslöser für alle Branches des Repositories gelten soll, wählen Sie unter Branches die Option All branches aus. Wählen Sie andernfalls Specific branches aus. Der Standard-Branch für das Repository wird standardmäßig hinzugefügt. Sie können diesen Branch behalten oder aus der Liste entfernen. Aus der Liste der Repository-Branches können Sie bis zu 10 Branch-Namen auswählen.
- (Optional) Geben Sie unter Benutzerdefinierte Daten Informationen ein, die Sie in die Lambda-Funktion aufnehmen möchten (z. B. den Namen des IRC-Kanals, der von Entwicklern verwendet wird, um die Entwicklung im Repository zu besprechen). Dies ist ein Zeichenfolgenfeld. Es kann nicht zur Übergabe dynamischer Parameter verwendet werden.

Wählen Sie Hinzufügen aus.

6. Wählen Sie auf der Seite Configuration (Konfiguration) unter Function Code (Funktionscode) und Code entry type (Codeeingabetyp) die Option Edit Code Inline (Code inline bearbeiten) aus. Wählen Sie unter Runtime (Laufzeit) die Option Node.js aus. Wenn Sie ein Python-Funktionsbeispiel erstellen möchten, wählen Sie Python aus.
7. Wählen Sie unter Code entry type die Option Edit code inline aus und ersetzen Sie den hello-world-Code durch eines der folgenden Beispiele.

Für Node.js:

```
import {
    CodeCommitClient,
    GetRepositoryCommand,
} from "@aws-sdk/client-codecommit";

const codecommit = new CodeCommitClient({ region: "your-region" });

/**
```

```

* @param {{ Records: { codecommit: { references: { ref: string }[] },
eventSourceARN: string }[] } event
*/
export const handler = async (event) => {
  // Log the updated references from the event
  const references = event.Records[0].codecommit.references.map(
    (reference) => reference.ref,
  );
  console.log("References:", references);

  // Get the repository from the event and show its git clone URL
  const repository = event.Records[0].eventSourceARN.split(":")[5];
  const params = {
    repositoryName: repository,
  };

  try {
    const data = await codecommit.send(new GetRepositoryCommand(params));
    console.log("Clone URL:", data.repositoryMetadata.cloneUrlHttp);
    return data.repositoryMetadata.cloneUrlHttp;
  } catch (error) {
    console.error("Error:", error);
    throw new Error(
      `Error getting repository metadata for repository ${repository}`,
    );
  }
};

```

Für Python:

```

import json
import boto3

codecommit = boto3.client("codecommit")

def lambda_handler(event, context):
    # Log the updated references from the event
    references = {
        reference["ref"]
        for reference in event["Records"][0]["codecommit"]["references"]
    }

```

```
print("References: " + str(references))

# Get the repository from the event and show its git clone URL
repository = event["Records"][0]["eventSourceARN"].split(":")[5]
try:
    response = codecommit.get_repository(repositoryName=repository)
    print("Clone URL: " + response["repositoryMetadata"]["cloneUrlHttp"])
    return response["repositoryMetadata"]["cloneUrlHttp"]
except Exception as e:
    print(e)
    print(
        "Error getting repository {}. Make sure it exists and that your
repository is in the same region as this function.".format(
            repository
        )
    )
    raise e
```

8. Wählen Sie auf der Registerkarte Berechtigungen unter Ausführungsrolle die Rolle aus, um sie in der IAM-Konsole zu öffnen. Bearbeiten Sie die angefügte Richtlinie, um dem Repository, das für den Auslöser verwendet werden soll, eine GetRepository-Berechtigung zu erteilen.

Den Trigger für die Lambda-Funktion im AWS CodeCommit Repository anzeigen

Nachdem Sie die Lambda-Funktion erstellt haben, können Sie den Trigger in AWS CodeCommit anzeigen und testen. Beim Testen des Auslösers wird die Funktion als Reaktion auf die von Ihnen angegebenen Repository-Ereignisse ausgeführt.

So zeigen Sie den Trigger für die Lambda-Funktion an und testen ihn

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories (Repositorys) das Repository aus, in dem Sie die Auslöser anzeigen möchten.
3. Wählen Sie im Navigationsbereich für das Repository Settings (Einstellungen) und anschließend Triggers (Auslöser).
4. Überprüfen Sie die Liste der Auslöser für das Repository. Sie sollten den Trigger sehen, den Sie in der Lambda-Konsole erstellt haben. Wählen Sie ihn aus der Liste aus und wählen Sie

Test trigger (Trigger testen). Damit wird die Funktion mit Beispieldaten zu Ihrem Repository, einschließlich der neuesten Commit-ID für das Repository, aufgerufen. (Falls kein Commit-Verlauf vorhanden ist, werden stattdessen aus Nullen bestehende Beispielwerte generiert.) Auf diese Weise können Sie überprüfen, ob Sie den Zugriff zwischen AWS CodeCommit und der Lambda-Funktion korrekt konfiguriert haben.

5. Erstellen Sie einen Commit und übertragen Sie ihn per Push an das Repository, für das Sie den Auslöser konfiguriert haben, um die Funktion des Auslösers weiter zu überprüfen. Sie sollten eine Antwort der Lambda-Funktion auf der Registerkarte Überwachung für diese Funktion in der Lambda-Konsole sehen. Wählen Sie auf der Registerkarte „Überwachung“ die Option „Logs in anzeigen“ aus. CloudWatch Die CloudWatch Konsole wird in einer neuen Registerkarte geöffnet und zeigt Ereignisse für Ihre Funktion an. Wählen Sie aus der Liste den Protokollstream mit dem Zeitpunkt aus, zu dem Sie den Commit per Push übertragen haben. Die angezeigten Ereignisdaten sollten wie folgt aussehen:

```
START RequestId: 70afdc9a-EXAMPLE Version: $LATEST
2015-11-10T18:18:28.689Z 70afdc9a-EXAMPLE References: [ 'refs/heads/main' ]
2015-11-10T18:18:29.814Z 70afdc9a-EXAMPLE Clone URL: https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo
END RequestId: 70afdc9a-EXAMPLE
REPORT RequestId: 70afdc9a-EXAMPLE Duration: 1126.87 ms Billed Duration: 1200 ms
Memory Size: 128 MB Max Memory Used: 14 MB
```

Beispiel: Erstellen Sie einen Trigger AWS CodeCommit für eine bestehende AWS Lambda Funktion

Der einfachste Weg, einen Trigger zu erstellen, der eine Lambda-Funktion aufruft, besteht darin, diesen Trigger in der Lambda-Konsole zu erstellen. Diese integrierte Integration stellt sicher, dass Sie CodeCommit über die zum Ausführen der Funktion erforderlichen Berechtigungen verfügen. Um einen Trigger für eine bestehende Lambda-Funktion hinzuzufügen, rufen Sie die Lambda-Konsole auf und wählen Sie die Funktion aus. Führen Sie auf der Registerkarte Triggers (Auslöser) zu der Funktion die Schritte unter Add trigger (Auslöser hinzufügen) aus. Diese Schritte gleichen denen unter [So erstellen Sie die Lambda-Funktion:](#).

Sie können auch einen Trigger für eine Lambda-Funktion in einem CodeCommit Repository erstellen. Dazu müssen Sie eine vorhandene Lambda-Funktion zum Aufrufen auswählen. Außerdem müssen Sie die für CodeCommit die Ausführung der Funktion erforderlichen Berechtigungen manuell konfigurieren.

Themen

- [Manuelles Konfigurieren von Berechtigungen, um die Ausführung einer Lambda-Funktion CodeCommit zu ermöglichen](#)
- [Erstellen Sie einen Trigger für die Lambda-Funktion in einem CodeCommit Repository \(Konsole\)](#)
- [Erstellen Sie einen Trigger für eine Lambda-Funktion für ein CodeCommit Repository \(\)AWS CLI](#)

Manuelles Konfigurieren von Berechtigungen, um die Ausführung einer Lambda-Funktion CodeCommit zu ermöglichen

Wenn Sie einen Trigger erstellen CodeCommit , der eine Lambda-Funktion aufruft, müssen Sie die Berechtigungen, mit denen CodeCommit die Lambda-Funktion ausgeführt werden kann, manuell konfigurieren. Um diese manuelle Konfiguration zu vermeiden, sollten Sie den Trigger für die Funktion stattdessen in der Lambda-Konsole erstellen.

Um die Ausführung einer Lambda-Funktion CodeCommit zu ermöglichen

1. Öffnen Sie einen Klartext-Editor und erstellen Sie eine JSON-Datei, die den Namen der Lambda-Funktion, die Details des CodeCommit Repositories und die Aktionen angibt, die Sie in Lambda zulassen möchten, ähnlich der folgenden:

```
{
  "FunctionName": "MyCodeCommitFunction",
  "StatementId": "1",
  "Action": "lambda:InvokeFunction",
  "Principal": "codecommit.amazonaws.com",
  "SourceArn": "arn:aws:codecommit:us-east-1:111122223333:MyDemoRepo",
  "SourceAccount": "111122223333"
}
```

2. Speichern Sie die Datei als JSON-Datei mit einem Namen, den Sie sich leicht merken können (z. B.. json). *AllowAccessfromMyDemoRepo*
3. Führen Sie mithilfe der JSON-Datei, die Sie gerade erstellt haben, im Terminal (Linux, macOS oder Unix) oder in der Befehlszeile (Windows) den `aws lambda add-permissions` Befehl aus, um der mit Ihrer Lambda-Funktion verknüpften Ressourcenrichtlinie eine Berechtigung hinzuzufügen:

```
aws lambda add-permission --cli-input-json file://AllowAccessfromMyDemoRepo.json
```

Dieser Befehl gibt den JSON-Wert der soeben hinzugefügten Richtlinienanweisung zurück, ähnlich folgendem Beispiel:

```
{
  "Statement": "{ \"Condition\": { \"StringEquals\": { \"AWS:SourceAccount\": \"111122223333\" }, \"ArnLike\": { \"AWS:SourceArn\": \"arn:aws:codecommit:us-east-1:111122223333:MyDemoRepo\" } }, \"Action\": [ \"lambda:InvokeFunction\" ], \"Resource\": \"arn:aws:lambda:us-east-1:111122223333:function:MyCodeCommitFunction\" }, \"Effect\": \"Allow\", \"Principal\": { \"Service\": \"codecommit.amazonaws.com\" }, \"Sid\": \"1\" }"
```

Weitere Informationen zu Ressourcenrichtlinien für Lambda-Funktionen finden Sie unter [AddPermission](#) und [The Pull/Push Event Models](#) im AWS Lambda Benutzerhandbuch.

4. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter <https://console.aws.amazon.com/iam/>
5. Wählen Sie im Navigationsbereich des Dashboards Rollen und dann in der Rollenliste die Option aus *lambda_basic_execution*.
6. Wählen Sie auf der Zusammenfassungsseite für die Rolle die Registerkarte Permissions (Berechtigungen) aus und wählen Sie unter Inline Policies (Eingebundene Richtlinien) die Option Create Role Policy (Rollenrichtlinie erstellen) aus.
7. Wählen Sie auf der Seite Set Permissions (Berechtigungen festlegen) die Option Policy Generator (Richtliniengenerator) und dann Select (Auswählen) aus.
8. Führen Sie auf der Seite Edit Permissions die folgenden Schritte aus:
 - Wählen Sie unter Effect die Option Allow aus.
 - Wählen Sie AWS unter Service die Option AWS CodeCommit.
 - Wählen Sie unter Aktionen die Option aus GetRepository.
 - Geben Sie unter Amazon Resource Name (ARN) (Amazon-Ressourcenname (ARN)) den ARN für das Repository ein (z. B. `arn:aws:codecommit:us-east-1:111122223333:MyDemoRepo`).

Wählen Sie Add Statement und anschließend Next Step aus.

9. Wählen Sie auf der Seite Review Policy (Richtlinie überprüfen) die Option Apply Policy (Richtlinie anwenden) aus.

Die Richtlinienanweisung sollte dann dem folgenden Beispiel ähneln:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt11111111",
      "Effect": "Allow",
      "Action": [
        "codecommit:GetRepository"
      ],
      "Resource": [
        "arn:aws:codecommit:us-east-1:111122223333:MyDemoRepo"
      ]
    }
  ]
}
```

Erstellen Sie einen Trigger für die Lambda-Funktion in einem CodeCommit Repository (Konsole)

Nachdem Sie die Lambda-Funktion erstellt haben, können Sie einen Trigger erstellen, der CodeCommit die Funktion als Reaktion auf die von Ihnen angegebenen Repository-Ereignisse ausführt.

 Note

Bevor Sie den Trigger für das Beispiel erfolgreich testen oder ausführen können, müssen Sie die Richtlinien konfigurieren, die es ermöglichen CodeCommit, die Funktion und die Lambda-Funktion aufzurufen, um Informationen über das Repository abzurufen. Weitere Informationen finden Sie unter [Um die Ausführung einer Lambda-Funktion CodeCommit zu ermöglichen](#).

Um einen Trigger für eine Lambda-Funktion zu erstellen

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.

2. Wählen Sie unter Repositories (Repositoryys) das Repository aus, in dem Sie Auslöser für Repository-Ereignisse erstellen möchten.
3. Wählen Sie im Navigationsbereich für das Repository Settings (Einstellungen) und anschließend Triggers (Auslöser).
4. Wählen Sie Create trigger.
5. Gehen Sie in Create trigger (Auslöser erstellen) wie folgt vor:

- Geben Sie im Feld Triggernamen einen Namen für den Trigger ein (z. B.).

MyLambdaFunctionTrigger

- Wählen Sie unter Ereignisse die Repository-Ereignisse aus, die die Lambda-Funktion auslösen.

Wenn Sie All repository events wählen, können Sie keine anderen Ereignisse auswählen. Wenn Sie nur einen Teil der Ereignisse auswählen möchten, löschen Sie All repository events und wählen Sie dann die gewünschten Ereignisse in der Liste aus. Wenn der Auslöser beispielsweise nur ausgeführt werden soll, wenn ein Benutzer in dem CodeCommit - Repository ein Tag oder einen Branch erstellt, entfernen Sie All repository events (Alle Repository-Ereignisse) und wählen Sie dann Create branch or tag (Branch oder Tag erstellen) aus.

- Wenn Sie möchten, dass der Auslöser für alle Branches des Repositorys unter Branches gilt, lassen Sie die Auswahl leer, da diese Standardoption den Auslöser automatisch auf alle Branches anwendet. Wenn Sie möchten, dass dieser Auslöser nur für bestimmte Branches gilt, wählen Sie bis zu 10 Branch-Namen aus der Liste der Repository-Banches aus.
 - Wählen Sie unter Choose the service to use (Zu verwendende Services auswählen) die AWS Lambda aus.
 - Wählen Sie Lambda Lambda-Funktion den Funktionsnamen aus der Liste aus, oder geben Sie den ARN für die Funktion ein.
 - (Optional) Geben Sie unter Benutzerdefinierte Daten Informationen ein, die Sie in die Lambda-Funktion aufnehmen möchten (z. B. den Namen des IRC-Kanals, der von Entwicklern verwendet wird, um die Entwicklung im Repository zu besprechen). Dies ist ein Zeichenfolgenfeld. Es kann nicht zur Übergabe dynamischer Parameter verwendet werden.
6. (Optional) Wählen Sie Test trigger (Auslöser testen). Damit wird die Funktion mit Beispieldaten zu Ihrem Repository, einschließlich der neuesten Commit-ID für das Repository, aufgerufen. (Falls kein Commit-Verlauf vorhanden ist, werden stattdessen aus Nullen bestehende

Beispielwerte generiert.) Auf diese Weise können Sie überprüfen, ob Sie den Zugriff zwischen CodeCommit und der Lambda-Funktion korrekt konfiguriert haben.

7. Wählen Sie Create trigger (Auslöser erstellen), um die Erstellung des Auslösers abzuschließen.
8. Erstellen Sie einen Commit und übertragen Sie ihn per Push an das Repository, für das Sie den Auslöser konfiguriert haben, um die Funktion des Auslösers zu überprüfen. Sie sollten eine Antwort der Lambda-Funktion auf der Registerkarte Überwachung für diese Funktion in der Lambda-Konsole sehen.

Erstellen Sie einen Trigger für eine Lambda-Funktion für ein CodeCommit Repository (AWS CLI)

Sie können auch die Befehlszeile verwenden, um einen Trigger für eine Lambda-Funktion als Reaktion auf CodeCommit Repository-Ereignisse zu erstellen, z. B. wenn jemand einen Commit in Ihr Repository überträgt.

Um einen Trigger für eine Lambda-Funktion zu erstellen

1. Öffnen Sie einen Texteditor und erstellen Sie eine JSON-Datei mit folgenden Angaben:
 - Der Name der Lambda-Funktion.
 - Repository und Branches, die mit diesem Auslöser überwacht werden sollen. (Wenn keine Branches angegeben werden, gilt der Auslöser für alle Branches im Repository.)
 - Ereignisse, die diesen Auslöser aktivieren.

Speichern Sie die Datei.

Wenn Sie beispielsweise einen Trigger für ein Repository mit dem Namen erstellen möchten *MyDemoRepo*, der alle Repository-Ereignisse in einer Lambda-Funktion veröffentlicht, die *MyCodeCommitFunction* nach zwei Branches benannt ist, *main* und *preprod*:

```
{
  "repositoryName": "MyDemoRepo",
  "triggers": [
    {
      "name": "MyLambdaFunctionTrigger",
      "destinationArn": "arn:aws:lambda:us-east-1:111122223333:function:MyCodeCommitFunction",
      "customData": "",
```

```

        "branches": [
            "main", "preprod"
        ],
        "events": [
            "all"
        ]
    }
}

```

In der JSON-Datei muss für jeden Auslöser eines Repositorys ein Auslöserblock vorhanden sein. Wenn Sie für ein Repository mehr als einen Auslöser erstellen möchten, fügen Sie in der JSON-Datei weitere Blöcke hinzu. Beachten Sie, dass alle in dieser Datei erstellten Auslöser für das angegebene Repository gelten. Es ist nicht möglich, Auslöser für verschiedene Repositorys in einer JSON-Datei zu erstellen. Wenn Sie beispielsweise zwei Auslöser für ein Repository erstellen möchten, können Sie eine JSON-Datei mit zwei Auslöserblöcken erstellen. Im folgenden Beispiel sind im zweiten Auslöserblock keine Branches angegeben, daher gilt dieser Auslöser für alle Branches:

```

{
    "repositoryName": "MyDemoRepo",
    "triggers": [
        {
            "name": "MyLambdaFunctionTrigger",
            "destinationArn": "arn:aws:lambda:us-east-1:111122223333:function:MyCodeCommitFunction",
            "customData": "",
            "branches": [
                "main", "preprod"
            ],
            "events": [
                "all"
            ]
        },
        {
            "name": "MyOtherLambdaFunctionTrigger",
            "destinationArn": "arn:aws:lambda:us-east-1:111122223333:function:MyOtherCodeCommitFunction",
            "customData": "",
            "branches": [],
            "events": [

```

```
        "updateReference", "deleteReference"
    ]
}
]
```

Sie können Auslöser für bestimmte Ereignisse erstellen, z. B. wenn ein Commit per Push an ein Repository übertragen wird. Zu den Ereignistypen zählen:

- `all` für alle Ereignisse im angegebenen Repository und den angegebenen Branches
- `updateReference` für Fälle, in denen Commits per Push an das angegebene Repository und die angegebenen Branches übertragen werden
- `createReference` für Fälle, in denen in dem angegebenen Repository ein neuer Branch oder ein neues Tag erstellt wird
- `deleteReference` für Fälle, in denen in dem angegebenen Repository ein Branch oder ein Tag gelöscht wird

 Note

Sie können in einem Auslöser mehrere Ereignistypen verwenden. Wenn Sie `all` angeben, können Sie allerdings keine anderen Ereignisse angeben.

Geben Sie im Terminal oder in der Eingabeaufforderung `aws codecommit put-repository-triggers help` ein, um eine vollständige Liste der gültigen Ereignistypen anzuzeigen.

Darüber hinaus können Sie in `customData` eine Zeichenfolge einschließen (z. B. den Namen eines IRC-Kanal, den Entwickler verwenden, wenn Sie die Entwicklung in diesem Repository besprechen). Dies ist ein Zeichenfolgenfeld. Es kann nicht zur Übergabe dynamischer Parameter verwendet werden. Diese Zeichenfolge wird als Attribut an die CodeCommit JSON-Datei angehängt, die als Antwort auf den Trigger zurückgegeben wird.

2. (Optional) Führen Sie an einem Terminal oder an einer Eingabeaufforderung den Befehl `test-repository-triggers` aus. Mit dem Folgenden wird beispielsweise getestet, ob die angegebene JSON-Datei gültig *trigger.json* ist und die Lambda-Funktion auslösen CodeCommit kann. Dieser Test verwendet Beispieldaten, um die Funktion auszulösen, wenn keine echten Daten vorhanden sind.

```
aws codecommit test-repository-triggers --cli-input-json file://trigger.json
```

Bei erfolgreicher Ausführung gibt dieser Befehl etwa wie folgt aussehende Informationen zurück:

```
{
  "successfulExecutions": [
    "MyLambdaFunctionTrigger"
  ],
  "failedExecutions": []
}
```

3. Führen Sie in einem Terminal oder einer Eingabeaufforderung den Befehl `put-repository-triggers` aus, um den Auslöser in CodeCommit zu erstellen. Um beispielsweise eine JSON-Datei mit dem Namen zu verwenden *trigger.json*, um den Trigger zu erstellen:

```
aws codecommit put-repository-triggers --cli-input-json
file://trigger.json
```

Dieser Befehl gibt eine Konfigurations-ID wie die folgende zurück:

```
{
  "configurationId": "0123456-I-AM-AN-EXAMPLE"
}
```

4. Führen Sie zum Anzeigen der Konfiguration des Auslösers den Befehl `get-repository-triggers` aus und geben Sie dabei den Namen des Repositorys an:

```
aws codecommit get-repository-triggers --repository-name MyDemoRepo
```

Dieser Befehl gibt die Struktur aller Auslöser zurück, die für das Repository konfiguriert sind, ähnlich folgendem Beispiel:

```
{
  "configurationId": "0123456-I-AM-AN-EXAMPLE",
  "triggers": [
    {
      "events": [
        "all"
      ],

```

```
        "destinationArn": "arn:aws:lambda:us-  
east-1:111122223333:MyCodeCommitFunction",  
        "branches": [  
            "main",  
            "preprod"  
        ],  
        "name": "MyLambdaFunctionTrigger",  
        "customData": "Project ID 12345"  
    }  
]  
}
```

5. Erstellen Sie einen Commit und übertragen Sie ihn per Push an das Repository, für das Sie den Auslöser konfiguriert haben, um die Funktion des Auslösers zu prüfen. Sie sollten eine Antwort der Lambda-Funktion auf der Registerkarte Überwachung für diese Funktion in der Lambda-Konsole sehen.

Trigger für ein Repository bearbeiten AWS CodeCommit

Sie können die Trigger bearbeiten, die für ein CodeCommit Repository erstellt wurden. Sie können die Ereignisse und Branches für den Auslöser, die als Reaktion auf das Ereignis auszuführende Aktion und andere Einstellungen bearbeiten.

Themen

- [Bearbeiten Sie einen Trigger für ein Repository \(Konsole\)](#)
- [Bearbeiten Sie einen Trigger für ein Repository \(\)AWS CLI](#)

Bearbeiten Sie einen Trigger für ein Repository (Konsole)

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories (Repositorys) das Repository aus, bei dem Sie einen Auslöser für Repository-Ereignisse bearbeiten möchten.
3. Wählen Sie im Navigationsbereich für das Repository Settings (Einstellungen) und anschließend Triggers (Auslöser).
4. Wählen Sie aus der Liste der Auslöser für das Repository den Auslöser aus, den Sie bearbeiten möchten, und klicken Sie auf Edit (Bearbeiten).

5. Nehmen Sie die gewünschten Änderungen am Auslöser vor und wählen Sie anschließend Save (Speichern) aus.

Bearbeiten Sie einen Trigger für ein Repository ()AWS CLI

1. Führen Sie an einem Terminal (Linux, macOS oder Unix) oder einer Befehlszeile (Windows) den `get-repository-triggers` Befehl aus, um eine JSON-Datei mit der Struktur aller Trigger zu erstellen, die für Ihr Repository konfiguriert sind. Um beispielsweise eine JSON-Datei *MyTriggers.json* mit dem Namen der Struktur aller Trigger zu erstellen, die für ein Repository konfiguriert sind *MyDemoRepo*:

```
aws codecommit get-repository-triggers --repository-name MyDemoRepo
>MyTriggers.json
```

Dieser Befehl gibt nichts zurück, aber eine Datei mit dem Namen *MyTriggers.json* wird in dem Verzeichnis erstellt, in dem Sie den Befehl ausgeführt haben.

2. Bearbeiten Sie die JSON-Datei in einem Texteditor und ändern Sie den Auslöserblock für den zu bearbeitenden Auslöser. Ersetzen Sie das `configurationId`-Paar durch ein `repositoryName`-Paar. Speichern Sie die Datei.

Wenn Sie beispielsweise einen Trigger mit dem Namen *MyFirstTrigger* des Projektarchivs *MyDemoRepo* so bearbeiten möchten, dass er für alle Zweige gilt, ersetzen Sie `configurationId` durch den angegebenen Wert `main` und `preprod` entfernen Sie ihn. Wenn keine Branches angegeben werden, gilt der Auslöser standardmäßig für alle Branches im Repository:

```
{
  "repositoryName": "MyDemoRepo",
  "triggers": [
    {
      "destinationArn": "arn:aws:sns:us-
east-2:111122223333:MyCodeCommitTopic",
      "branches": [
        "main",
        "preprod"
      ],
      "name": "MyFirstTrigger",
      "customData": "",
      "events": [
```

```
        "all"
      ]
    }
  ]
}
```

3. Führen Sie am Terminal oder über die Befehlszeile den Befehl `put-repository-triggers` aus. Dadurch werden alle Trigger für das Repository aktualisiert, einschließlich der Änderungen, die Sie am *MyFirstTrigger* Trigger vorgenommen haben:

```
aws codecommit put-repository-triggers --repository-name MyDemoRepo
file://MyTriggers.json
```

Dieser Befehl gibt eine Konfigurations-ID wie die folgende zurück:

```
{
  "configurationId": "0123456-I-AM-AN-EXAMPLE"
}
```

Testen Sie Trigger für ein AWS CodeCommit Repository

Sie können die Trigger testen, die für ein CodeCommit Repository erstellt wurden. Beim Testen wird der Auslöser mit Beispieldaten aus Ihrem Repository ausgeführt, einschließlich der aktuellen Commit-ID. Wenn für das Repository kein Commit-Verlauf existiert, werden stattdessen aus Nullen bestehende Beispielwerte generiert. Durch das Testen von Triggern können Sie überprüfen, ob Sie den Zugriff zwischen CodeCommit und dem Ziel des Triggers korrekt konfiguriert haben, unabhängig davon, ob es sich um eine AWS Lambda Funktion oder eine Amazon Simple Notification Service-Benachrichtigung handelt.

Themen

- [Testen Sie einen Trigger für ein Repository \(Konsole\)](#)
- [Testen Sie einen Trigger für ein Repository \(AWS CLI\)](#)

Testen Sie einen Trigger für ein Repository (Konsole)

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.

2. Wählen Sie unter Repositories (Repositoryys) das Repository aus, bei dem Sie einen Auslöser für Repository-Ereignisse testen möchten.
3. Wählen Sie im Navigationsbereich für das Repository Settings (Einstellungen) und anschließend Triggers (Auslöser).
4. Wählen Sie den Auslöser aus, den Sie testen möchten, und klicken Sie auf Test trigger (Auslöser testen). Es sollte eine Erfolgs- oder Fehlermeldung angezeigt werden. Bei Erfolg sollte Ihnen auch eine entsprechende Aktionsantwort von der Lambda-Funktion oder dem Amazon SNS SNS-Thema angezeigt werden.

Testen Sie einen Trigger für ein Repository ()AWS CLI

1. Führen Sie an einem Terminal (Linux, macOS oder Unix) oder einer Befehlszeile (Windows) den `get-repository-triggers` Befehl aus, um eine JSON-Datei mit der Struktur aller Trigger zu erstellen, die für Ihr Repository konfiguriert sind. Um beispielsweise eine JSON-Datei *TestTrigger.json* mit dem Namen der Struktur aller Trigger zu erstellen, die für ein Repository konfiguriert sind MyDemoRepo:

```
aws codecommit get-repository-triggers --repository-name MyDemoRepo  
>TestTrigger.json
```

Dieser Befehl erstellt eine Datei mit dem Namen *TestTriggers.json* in dem Verzeichnis, in dem Sie den Befehl ausgeführt haben.

2. Bearbeiten Sie die JSON-Datei in einem Texteditor und nehmen Sie die gewünschten Änderungen an der Auslöseranweisung vor. Ersetzen Sie das `configurationId`-Paar durch ein `repositoryName`-Paar. Speichern Sie die Datei.

Wenn Sie beispielsweise einen Trigger testen möchten, der *MyFirstTrigger* im Repository *MyDemoRepo* so benannt ist, dass er für alle Branches gilt, ersetzen Sie den `configurationId` Trigger durch `repositoryName` und speichern Sie dann eine Datei, die der folgenden ähnelt, unter *TestTrigger.json*:

```
{  
  "repositoryName": "MyDemoRepo",  
  "triggers": [  
    {  
      "destinationArn": "arn:aws:sns:us-  
east-2:111122223333:MyCodeCommitTopic",
```



```
    "branches": [
      "main",
      "preprod"
    ],
    "name": "MyFirstTrigger",
    "customData": "",
    "events": [
      "all"
    ]
  }
]
```

3. Führen Sie am Terminal oder über die Befehlszeile den Befehl `test-repository-triggers` aus. Dadurch werden alle Trigger für das Repository aktualisiert, einschließlich der Änderungen, die Sie am *MyFirstTrigger* Trigger vorgenommen haben:

```
aws codecommit test-repository-triggers --cli-input-json file://TestTrigger.json
```

Dieser Befehl gibt eine Reaktion wie die folgende zurück:

```
{
  "successfulExecutions": [
    "MyFirstTrigger"
  ],
  "failedExecutions": []
}
```

Trigger aus einem AWS CodeCommit Repository löschen

Sie können Auslöser, die nicht mehr benötigt werden, löschen. Dieser Löschvorgang kann nicht rückgängig gemacht werden, aber Sie können die Auslöser erneut erstellen.

Note

Wenn Sie einen oder mehrere Trigger für Ihr Repository konfiguriert haben, werden beim Löschen des Repositories nicht die Amazon SNS SNS-Themen oder Lambda-Funktionen gelöscht, die Sie als Ziele dieser Trigger konfiguriert haben. Achten Sie darauf, diese Ressourcen ebenfalls zu löschen, sofern sie nicht mehr benötigt werden.

Themen

- [Löscht einen Trigger aus einem Repository \(Konsole\)](#)
- [Löscht einen Trigger aus einem Repository \(\)AWS CLI](#)

Löscht einen Trigger aus einem Repository (Konsole)

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories (Repositorys) das Repository aus, in dem Sie Auslöser für Repository-Ereignisse löschen möchten.
3. Wählen Sie im Navigationsbereich des Repositorys die Option Settings aus. Klicken Sie unter Settings auf die Option Triggers.
4. Wählen Sie den zu löschenden Auslöser in der Liste aus und wählen Sie dann Delete (Löschen).
5. Geben Sie zur Bestätigung im Dialogfeld delete ein.

Löscht einen Trigger aus einem Repository ()AWS CLI

1. Führen Sie an einem Terminal (Linux, macOS oder Unix) oder einer Befehlszeile (Windows) den `get-repository-triggers` Befehl aus, um eine JSON-Datei mit der Struktur aller Trigger zu erstellen, die für Ihr Repository konfiguriert sind. Um beispielsweise eine JSON-Datei *MyTriggers.json* mit dem Namen der Struktur aller Trigger zu erstellen, die für ein Repository konfiguriert sind `MyDemoRepo`:

```
aws codecommit get-repository-triggers --repository-name MyDemoRepo  
>MyTriggers.json
```

Dieser Befehl erstellt eine Datei mit dem Namen *MyTriggers.json* in dem Verzeichnis, in dem Sie den Befehl ausgeführt haben.

2. Bearbeiten Sie die JSON-Datei in einem Texteditor und entfernen Sie den Auslöserblock für den zu löschenden Auslöser. Ersetzen Sie das `configurationId`-Paar durch ein `repositoryName`-Paar. Speichern Sie die Datei.

Wenn Sie beispielsweise einen Trigger mit dem Namen *MyFirstTrigger* aus dem Repository `configurationId` mit `repositoryName` dem Namen entfernen möchten *MyDemoRepo*, ersetzen Sie ihn durch und entfernen Sie die Anweisung in *red italic text*:

```
{
  "repositoryName": "MyDemoRepo",
  "triggers": [
    {
      "destinationArn": "arn:aws:sns:us-east-2:111122223333:MyCodeCommitTopic",
      "branches": [
        "main",
        "preprod"
      ],
      "name": "MyFirstTrigger",
      "customData": "",
      "events": [
        "all"
      ]
    },
    {
      "destinationArn": "arn:aws:lambda:us-east-2:111122223333:function:MyCodeCommitJSFunction",
      "branches": [],
      "name": "MyLambdaTrigger",
      "events": [
        "all"
      ]
    }
  ]
}
```

3. Führen Sie am Terminal oder über die Befehlszeile den Befehl `put-repository-triggers` aus. Dadurch werden die Trigger für das Repository aktualisiert und der *MyFirstTrigger* Trigger gelöscht:

```
aws codecommit put-repository-triggers --repository-name MyDemoRepo
file://MyTriggers.json
```

Dieser Befehl gibt eine Konfigurations-ID wie die folgende zurück:

```
{
  "configurationId": "0123456-I-AM-AN-EXAMPLE"
}
```

 Note

Wenn Sie alle Auslöser für das Repository *MyDemoRepo* löschen möchten, muss die JSON-Datei wie folgt aussehen:

```
{
  "repositoryName": "MyDemoRepo",
  "triggers": []
}
```

Ein AWS CodeCommit Repository mit Amazon CodeGuru Reviewer verknüpfen oder die Zuordnung aufheben

Amazon CodeGuru Reviewer ist ein automatisierter Code-Review-Service, der mithilfe von Programmanalyse und maschinellem Lernen häufig auftretende Probleme erkennt und Korrekturen in Ihrem Java- oder Python-Code empfiehlt. Sie können Repositories in Ihrem Amazon Web Services Services-Konto mit CodeGuru Reviewer verknüpfen. Wenn Sie dies tun, erstellt der CodeGuru Prüfer eine servicebezogene Rolle, die es dem CodeGuru Prüfer ermöglicht, den Code in allen Pull-Requests zu analysieren, die nach der Zuordnung erstellt wurden.

Nachdem Sie ein Repository zugeordnet haben, analysiert und kommentiert CodeGuru Reviewer alle Probleme, die bei der Erstellung von Pull Requests festgestellt werden. Jeder Kommentar ist deutlich als von einem CodeGuru Rezensenten mit der Bezeichnung Amazon CodeGuru Reviewer gekennzeichnet. Sie können auf diese Kommentare wie auf alle anderen Kommentare in einer Pull-Anforderung reagieren und auch Feedback zur Qualität des Vorschlags übermitteln. Dieses Feedback wird an den CodeGuru Rezensenten weitergegeben und kann dazu beitragen, den Service und seine Vorschläge zu verbessern.

 Note

In Pull-Requests, die erstellt wurden, bevor das Repository mit dem Repository verknüpft wurde, werden dir keine Kommentare von CodeGuru Reviewer angezeigt. Aus folgenden Gründen werden Kommentare in Pull-Anforderungen, die nach der Zuordnung erstellt wurden, möglicherweise nicht angezeigt:

- Die Pull-Anfrage enthält keinen Java- oder Python-Code.

- CodeGuru Der Prüfer hatte nicht genug Zeit, um den Code in der Pull-Anfrage auszuführen und zu überprüfen. Dieser Vorgang kann bis zu 30 Minuten dauern. Kommentare können im Verlauf der Überprüfung erscheinen, aber das Kommentieren ist erst abgeschlossen, wenn der Jobstatus als Abgeschlossen angezeigt wird.
- CodeGuru Der Prüfer hat in der Pull-Anfrage keine Probleme im Java- oder Python-Code festgestellt.
- Der Code-Prüfungsauftrag konnte nicht ausgeführt werden. Den Status einer Überprüfung für einen Pull Request findest du auf der Registerkarte Aktivität des Pull-Requests.
- Sie sehen die Änderungen am Pull Request auf der Registerkarte Änderungen, der Pull Request wurde aktualisiert und Amazon CodeGuru Reviewer hat bei den Änderungen keine Probleme festgestellt. Kommentare von Amazon CodeGuru Reviewer werden nur dann auf der Registerkarte „Änderungen“ angezeigt, wenn sie zur letzten Version des Pull-Requests abgegeben wurden. Sie werden immer auf der Registerkarte Aktivität angezeigt.

The screenshot displays the AWS CodeCommit Pull Request interface for a repository named 'MyDemoRepo'. The pull request is titled '25: Updated some of our Java samples' and is authored by 'Li_Juan'. The status is 'In progress'. The interface shows the 'Activity' tab selected, displaying a comment from the 'Amazon CodeGuru Reviewer' bot. The comment states: 'This code might not produce accurate results if the operation returns paginated results instead of all results. Consider adding another call to check for additional results. Leave feedback on this recommendation by selecting "Reply".' The comment also includes a note that feedback and comments will be shared with Amazon CodeGuru Reviewer and might be used to improve the service. The comment is dated '2 minutes ago' and has 1 reply. The interface also shows the 'Activity history' section with a message: 'Pull request updated 1 minute ago. One or more commits added. Li_Juan updated the pull request.'

Weitere Informationen finden Sie unter [Mit Pull-Requests in AWS CodeCommit Repositorys arbeiten](#) und im [Amazon CodeGuru Reviewer-Benutzerhandbuch](#).

 Note

Sie müssen mit einem IAM-Benutzer oder einer IAM-Rolle angemeldet sein, der über ausreichende Berechtigungen verfügt, um ein Repository mit CodeGuru Reviewer zu verknüpfen oder die Zuordnung aufzuheben. Informationen zu den verwalteten Richtlinien, die diese Berechtigungen beinhalten CodeCommit , finden Sie unter [AWS verwaltete Richtlinien für CodeCommit](#) und. [AWS CodeCommit verwaltete Richtlinien und Amazon CodeGuru Reviewer](#) Informationen zu den Berechtigungen und zur Sicherheit von CodeGuru Rezensenten finden Sie im Amazon CodeGuru Reviewer-Benutzerhandbuch.

Themen

- [Ordnen Sie dem CodeGuru Rezensenten ein Repository zu](#)
- [Trennen Sie die Zuordnung eines Repositorys zu Reviewer CodeGuru](#)

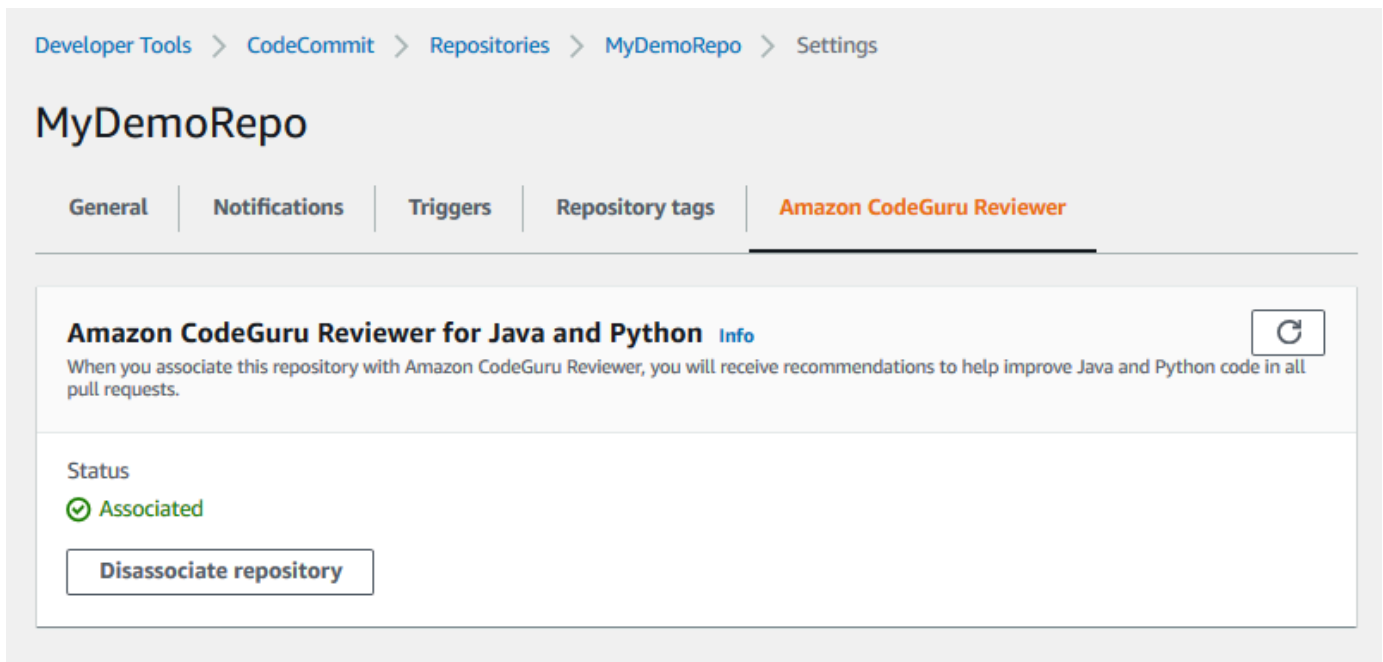
Ordnen Sie dem CodeGuru Rezensenten ein Repository zu

Verwenden Sie die AWS CodeCommit Konsole, um CodeGuru Reviewer schnell ein Repository zuzuordnen. Weitere Methoden finden Sie im Amazon CodeGuru Reviewer-Benutzerhandbuch.

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories den Namen des Repositorys aus, das Reviewer zugeordnet werden soll. CodeGuru
3. Wählen Sie Einstellungen und dann Amazon CodeGuru Reviewer aus.
4. Wählen Sie Associate repository (Repository zuordnen) aus.

 Note

Es kann bis zu 10 Minuten dauern, bis ein Repository vollständig mit CodeGuru Reviewer verknüpft ist. Der Status wird nicht automatisch aktualisiert. Um den aktuellen Status anzuzeigen, wählen Sie die Schaltfläche "Refresh (Aktualisieren)".



Trennen Sie die Zuordnung eines Repositorys zu Reviewer CodeGuru

Verwenden Sie die AWS CodeCommit Konsole, um ein Repository schnell von CodeGuru Reviewer zu trennen. Weitere Methoden finden Sie im Amazon CodeGuru Reviewer-Benutzerhandbuch.

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories den Namen des Repositorys aus, das Sie von Reviewer trennen möchten. CodeGuru
3. Wählen Sie Einstellungen und dann Amazon CodeGuru Reviewer aus.
4. Wählen Sie Disassociate repository (Repository-Zuordnung aufheben).

CodeCommit Repository-Details anzeigen

Sie können die AWS CodeCommit Konsole oder Git von einem lokalen Repo aus verwenden AWS CLI, das mit dem CodeCommit Repository verbunden ist, um Informationen über verfügbare Repositorys anzuzeigen.

Bevor Sie diese Anweisungen befolgen, führen Sie die Schritte unter [Einrichtung](#) aus.

Themen

- [Repository-Details anzeigen \(Konsole\)](#)
- [CodeCommit Repository-Details anzeigen \(Git\)](#)
- [CodeCommit Repository-Details anzeigen \(AWS CLI\)](#)

Repository-Details anzeigen (Konsole)

Verwenden Sie die AWS CodeCommit Konsole, um schnell alle Repositorys anzuzeigen, die mit Ihrem Amazon Web Services Services-Konto erstellt wurden.

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Sehen Sie sich unter Repositorys die Details zu den Repositorys in dem Bereich an, in AWS-Region dem Sie angemeldet sind. Verwenden Sie die Regionsauswahl, um ein anderes AWS-Region T auszuwählen, um Repositorys in dieser Region anzuzeigen.
3. Wählen Sie den Namen des Repositorys, für das Sie weitere Details anzeigen möchten, und führen Sie dann einen der folgenden Schritte aus:
 - Wählen Sie zum Anzeigen der URL für das Klonen des Repositorys Clone URL (URL klonen) und anschließend das Protokoll aus, das Sie beim Klonen des Repositorys verwenden möchten. Dadurch wird die Klon-URL kopiert. Um sie zu überprüfen, fügen Sie sie in einen Texteditor ein.
 - Um konfigurierbare Optionen für das Repository sowie Details wie den Repository-ARN und die Repository-ID anzuzeigen, wählen Sie im Navigationsbereich Einstellungen aus.

Note

Wenn Sie als IAM-Benutzer angemeldet sind, können Sie Ihre Einstellungen für die Anzeige von Code und anderen Konsoleneinstellungen konfigurieren und speichern. Weitere Informationen finden Sie unter [Mit Benutzereinstellungen arbeiten](#).

CodeCommit Repository-Details anzeigen (Git)

Um Git von einem lokalen Repo aus zu verwenden, um Details zu CodeCommit Repositorys anzuzeigen, führen Sie den `git remote show` Befehl aus.

Bevor du diese Schritte ausführst, verbinde das lokale Repository mit dem Repository. CodeCommit
Detaillierte Anweisungen finden Sie unter [Herstellen einer Verbindung mit einem Repository](#).

1. Führen Sie den `git remote show remote-name` Befehl aus, wobei der Alias des CodeCommit Repositorys **remote-name** steht (standardmäßig `origin`).

 Tip

Um eine Liste der CodeCommit Repository-Namen und ihrer Namen zu erhalten URLs, führen Sie den `git remote -v` Befehl aus.

Um beispielsweise Details über das CodeCommit Repository mit dem Alias `origin` anzuzeigen:

```
git remote show origin
```

2. Für HTTPS:

```
* remote origin
Fetch URL: https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo
Push URL: https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo
HEAD branch: (unknown)
Remote branches:
  MyNewBranch tracked
  main tracked
Local ref configured for 'git pull':
  MyNewBranch merges with remote MyNewBranch (up to date)
Local refs configured for 'git push':
  MyNewBranch pushes to MyNewBranch (up to date)
  main pushes to main (up to date)
```

Für SSH:

```
* remote origin
Fetch URL: ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo
Push URL: ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo
HEAD branch: (unknown)
Remote branches:
  MyNewBranch tracked
  main tracked
Local ref configured for 'git pull':
```

```
MyNewBranch merges with remote MyNewBranch (up to date)
Local refs configured for 'git push':
  MyNewBranch pushes to MyNewBranch (up to date)
  main pushes to main (up to date)
```

 Tip

Um die SSH-Schlüssel-ID für Ihren IAM-Benutzer nachzuschlagen, öffnen Sie die IAM-Konsole und erweitern Sie auf der Seite mit den IAM-Benutzerdetails die Option Sicherheitsanmeldedaten. Die SSH-Schlüssel-ID finden Sie unter SSH-Schlüssel für AWS CodeCommit

Weitere Optionen findest du in deiner Git-Dokumentation.

CodeCommit Repository-Details anzeigen (AWS CLI)

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter [Befehlszeilenreferenz](#).

Führen Sie die folgenden Befehle aus, AWS CLI um die Repository-Details anzuzeigen:

- Um eine Liste der CodeCommit Repository-Namen und der entsprechenden Namen anzuzeigen IDs, führen Sie [list-repositories](#) aus.
- [Um Informationen zu einem einzelnen CodeCommit Repository anzuzeigen, führen Sie get-repository aus.](#)
- Um Informationen über mehrere Repositories in anzuzeigen CodeCommit, führen Sie den Befehl aus. [batch-get-repositories](#)

So zeigen Sie eine Liste der CodeCommit-Repositories an

1. Führen Sie den Befehl list-repositories aus:

```
aws codecommit list-repositories
```

Sie können den optionalen Parameter `--sort-by` oder `--order` verwenden, um die Sortierreihenfolge der zurückgegebenen Informationen zu ändern.

2. Bei Erfolg gibt dieser Befehl ein `repositories` Objekt aus, das die Namen und IDs alle Repositories enthält, die dem Amazon Web Services Services-Konto CodeCommit zugeordnet sind.

Es folgt eine Beispielausgabe basierend auf dem vorangehenden Befehl:

```
{
  "repositories": [
    {
      "repositoryName": "MyDemoRepo",
      "repositoryId": "f7579e13-b83e-4027-aaef-650c0EXAMPLE"
    },
    {
      "repositoryName": "MyOtherDemoRepo",
      "repositoryId": "cfc29ac4-b0cb-44dc-9990-f6f51EXAMPLE"
    }
  ]
}
```

Um Details zu einem einzelnen CodeCommit Repository anzuzeigen

1. Führen Sie den `get-repository` Befehl aus und geben Sie den Namen des CodeCommit Repositories mit der `--repository-name` Option an.

Tip

Um den Namen des CodeCommit Repositories abzurufen, führen Sie den Befehl [list-repositories](#) aus.

Um beispielsweise Details zu einem CodeCommit Repository mit dem Namen anzuzeigen: `MyDemoRepo`

```
aws codecommit get-repository --repository-name MyDemoRepo
```

2. Ist der Befehl erfolgreich, wird ein `repositoryMetadata`-Objekt mit den folgenden Informationen ausgegeben:
- Name des Repositories (`repositoryName`)

- Beschreibung des Repositorys (`repositoryDescription`)
- Eindeutige, systemgenerierte ID des Repositorys (`repositoryId`)
- Die ID des Amazon Web Services Services-Kontos, das dem Repository zugeordnet ist (`accountId`).

Es folgt eine Beispielausgabe basierend auf dem vorangehenden Beispielbefehl:

```
{
  "repositoryMetadata": {
    "creationDate": 1429203623.625,
    "defaultBranch": "main",
    "repositoryName": "MyDemoRepo",
    "cloneUrlSsh": "ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo",
    "lastModifiedDate": 1430783812.0869999,
    "repositoryDescription": "My demonstration repository",
    "cloneUrlHttp": "https://codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo",
    "repositoryId": "f7579e13-b83e-4027-aaef-650c0EXAMPLE",
    "Arn": "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo",
    "accountId": "111111111111"
  }
}
```

So zeigen Sie Details zu mehreren CodeCommit -Repositorys; an

1. Führen Sie den Befehl `batch-get-repositories` mit der Option `--repository-names` aus. Fügen Sie zwischen den einzelnen CodeCommit Repository-Namen ein Leerzeichen ein.

 Tip

Um die Namen der Repositorys abzurufen CodeCommit, führen Sie den Befehl [list-repositories](#) aus.

Um beispielsweise Details zu zwei CodeCommit Repositorys mit dem Namen und anzuzeigen:
MyDemoRepo MyOtherDemoRepo

```
aws codecommit batch-get-repositories --repository-names MyDemoRepo MyOtherDemoRepo
```

2. Ist der Befehl erfolgreich, wird ein Objekt mit den folgenden Informationen ausgegeben:

- Eine Liste aller CodeCommit Repositorys, die nicht gefunden werden konnten (`repositoriesNotFound`).
- Eine Liste von CodeCommit Repositorien (`repositories`). Auf jeden CodeCommit Repository-Namen folgt:
 - Beschreibung des Repositorys (`repositoryDescription`)
 - Eindeutige, systemgenerierte ID des Repositorys (`repositoryId`)
 - Die ID des Amazon Web Services Services-Kontos, das dem Repository zugeordnet ist (`accountId`).

Es folgt eine Beispielausgabe basierend auf dem vorangehenden Beispielbefehl:

```
{
  "repositoriesNotFound": [],
  "repositories": [
    {
      "creationDate": 1429203623.625,
      "defaultBranch": "main",
      "repositoryName": "MyDemoRepo",
      "cloneUrlSsh": "ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo",
      "lastModifiedDate": 1430783812.0869999,
      "repositoryDescription": "My demonstration repository",
      "cloneUrlHttp": "https://codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo",
      "repositoryId": "f7579e13-b83e-4027-aaef-650c0EXAMPLE",
      "Arn": "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo",
      "accountId": "111111111111"
    },
    {
      "creationDate": 1429203623.627,
      "defaultBranch": "main",
      "repositoryName": "MyOtherDemoRepo",
      "cloneUrlSsh": "ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyOtherDemoRepo",
      "lastModifiedDate": 1430783812.0889999,
```

```
        "repositoryDescription": "My other demonstration repository",
        "cloneUrlHttp": "https://codecommit.us-east-2.amazonaws.com/v1/
repos/MyOtherDemoRepo",
        "repositoryId": "cfc29ac4-b0cb-44dc-9990-f6f51EXAMPLE",
        "Arn": "arn:aws:codecommit:us-east-2:111111111111:MyOtherDemoRepo",
        "accountId": "111111111111"
    }
],
"repositoriesNotFound": []
}
```

AWS CodeCommit Repository-Einstellungen ändern

Sie können die Konsole AWS CLI und die AWS CodeCommit Konsole verwenden, um die Einstellungen eines CodeCommit Repositorys zu ändern, z. B. dessen Beschreibung oder seinen Namen.

Important

Wenn der Name eines Repositorys geändert wird, können möglicherweise lokale Repos nicht mehr aufgerufen werden, die den alten Namen in ihrer Remote-URL verwenden. Führen Sie den Befehl `git remote set-url` aus, um die Remote-URL für die Verwendung des neuen Repository-Namens zu aktualisieren.

Themen

- [Repository-Einstellungen ändern \(Konsole\)](#)
- [AWS CodeCommit Repository-Einstellungen ändern \(AWS CLI\)](#)

Repository-Einstellungen ändern (Konsole)

Gehen Sie wie folgt vor, um mit der AWS CodeCommit Konsole die Einstellungen eines CodeCommit Repositorys zu ändern. AWS CodeCommit

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.

2. Wählen Sie unter Repositories (Repositoryys) den Namen des Repositorys aus, bei dem Sie Einstellungen ändern möchten.
3. Wählen Sie im Navigationsbereich Settings (Einstellungen).
4. Um den Namen des Repositorys zu ändern, geben Sie unter Repository name (Repository-Name) in das Textfeld Name einen neuen Namen ein und klicken Sie auf Save (Speichern). Wenn Sie dazu aufgefordert werden, überprüfen Sie Ihre Wahl.

Important

Wenn Sie den Namen des AWS CodeCommit Repositorys ändern, ändern sich auch die SSH- und HTTPS-Werte URLs, die Benutzer benötigen, um sich mit dem Repository zu verbinden. Benutzer können erst eine Verbindung mit diesem Repository herstellen, wenn sie ihre Verbindungseinstellungen aktualisiert haben. Da sich der ARN des Repositorys ändert, werden durch die Änderung des Repository-Namens auch alle IAM-Benutzerrichtlinien ungültig, die auf dem ARN dieses Repositorys basieren.

Um nach einer Namensänderung eine Verbindung mit dem Repository herzustellen, muss der Benutzer mit dem `git remote set-url` die neue URL angeben. Wenn Sie beispielsweise den Namen des Repositorys von `MyDemoRepo` in ändern würden, würden Benutzer `MyRenamedDemoRepo`, die HTTPS verwenden, um eine Verbindung zum Repository herzustellen, den folgenden Git-Befehl ausführen:

```
git remote set-url origin https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyRenamedDemoRepo
```

Benutzer, die mit SSH eine Verbindung mit dem Repository herstellen, führen den folgenden Git-Befehl aus:

```
git remote set-url origin ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyRenamedDemoRepo
```

Weitere Optionen findest du in deiner Git-Dokumentation.

5. Um die Beschreibung des Repositorys zu ändern, geben Sie den Text in das Textfeld Description (Beschreibung) ein und klicken Sie auf Save (Speichern).

 Note

Das Beschreibungsfeld zeigt Markdown in der Konsole an und akzeptiert alle HTML-Zeichen und gültigen Unicode-Zeichen. Wenn Sie als Anwendungsentwickler das `GetRepository` oder verwenden `BatchGetRepositories` APIs und das Repository-Beschreibungsfeld in einem Webbrowser anzeigen möchten, finden Sie weitere Informationen in der [CodeCommit API-Referenz](#).

6. Zum Ändern des Standard-Banches wählen Sie unter Default branch (Standard-Branch) die Branch-Dropdown-Liste und wählen Sie einen anderen Branch aus. Wählen Sie Speichern.
7. Um den AWS KMS Verschlüsselungsschlüssel zu ändern, der zum Verschlüsseln und Entschlüsseln von Daten im Repository verwendet wird, wählen Sie unter Repository-Verschlüsselungsschlüssel entweder Von AWS verwalteter Schlüssel oder Vom Kunden verwalteter Schlüssel, um den Typ des zu verwendenden Schlüssels anzugeben. Wenn Sie einen vom Kunden verwalteten Schlüssel wählen, geben Sie den ARN des Schlüssels ein. Wählen Sie Speichern.
8. Um das Repository zu löschen, klicken Sie auf Delete repository. Geben Sie in das Feld neben Type the name of the repository to confirm deletion (Namen des Repositorys eingeben, um das Löschen zu bestätigen) die Zeichenfolge **delete** ein und klicken Sie dann auf Delete (Löschen).

 Important

Nachdem Sie dieses Repository gelöscht haben AWS CodeCommit, können Sie es nicht mehr in ein lokales oder gemeinsam genutztes Repository klonen. Sie werden auch nicht mehr in der Lage sein, Daten aus einem lokalen oder gemeinsam genutzten Repo daraus abzurufen oder Daten dorthin zu übertragen. Diese Aktion kann nicht rückgängig gemacht werden.

AWS CodeCommit Repository-Einstellungen ändern (AWS CLI)

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter [Befehlszeilenreferenz](#).

Um die Einstellungen eines CodeCommit Repositorys in AWS CLI zu ändern AWS CodeCommit, führen Sie einen oder mehrere der folgenden Befehle aus:

- [update-repository-description](#) um die Beschreibung eines CodeCommit Repositorys zu ändern.
- [update-repository-name](#) um den Namen eines CodeCommit Repositorys zu ändern.

Um die Beschreibung eines CodeCommit Repositorys zu ändern

1. Führen Sie den Befehl `update-repository-description` aus und geben Sie Folgendes an:

- Der Name des CodeCommit Repositorys (mit der `--repository-name` Option).

Tip

Führen Sie den [list-repositories](#) Befehl aus, um den Namen des CodeCommit Repositorys abzurufen.

- Neue Beschreibung des Repositorys (mit der Option `--repository-description`).

Note

Das Beschreibungsfeld zeigt Markdown in der Konsole an und akzeptiert alle HTML-Zeichen und gültigen Unicode-Zeichen. Wenn Sie als Anwendungsentwickler das `GetRepository` oder verwenden `BatchGetRepositories` APIs und das Repository-Beschreibungsfeld in einem Webbrowser anzeigen möchten, finden Sie weitere Informationen in der [CodeCommit API-Referenz](#).

Um beispielsweise die Beschreibung für das CodeCommit Repository mit dem Namen wie folgt `MyDemoRepo` zu ändern `This description was changed`:

```
aws codecommit update-repository-description --repository-name MyDemoRepo --  
repository-description "This description was changed"
```

Dieser Befehl liefert nur eine Ausgabe, wenn Fehler aufgetreten sind.

2. Um die geänderte Beschreibung zu überprüfen, führen Sie den `get-repository` Befehl aus und geben Sie den Namen des CodeCommit Repositorys an, dessen Beschreibung Sie mit der `--repository-name` Option geändert haben.

Die Ausgabe des Befehls zeigt den geänderten Text in `repositoryDescription`.

Um den Namen eines CodeCommit Repositorys zu ändern

1. Führen Sie den Befehl `update-repository-name` aus und geben Sie Folgendes an:
 - Der aktuelle Name des CodeCommit Repositorys (mit der `--old-name` Option).

Tip

Um den Namen des CodeCommit Repositorys abzurufen, führen Sie den Befehl [list-repositories](#) aus.

- Der neue Name des CodeCommit Repositorys (mit der `--new-name` Option).

So ändern Sie z. B. das Repository mit dem Namen `MyDemoRepo` in `MyRenamedDemoRepo`:

```
aws codecommit update-repository-name --old-name MyDemoRepo --new-name
MyRenamedDemoRepo
```

Dieser Befehl liefert nur eine Ausgabe, wenn Fehler aufgetreten sind.

Important

Wenn Sie den Namen des AWS CodeCommit Repositorys ändern, werden SSH und HTTPS geänderte URLs, die Benutzer benötigen, um eine Verbindung zum Repository herzustellen. Benutzer können erst eine Verbindung mit diesem Repository herstellen, nachdem sie ihre Verbindungseinstellungen aktualisiert haben. Da sich der ARN des Repositorys ändert, macht eine Änderung des Repository-Namens außerdem alle IAM-Benutzerrichtlinien ungültig, die auf dem ARN dieses Repositorys basieren.

2. Um den geänderten Namen zu überprüfen, führen Sie den Befehl `list-repositories` aus und prüfen Sie die Liste mit den Repository-Namen.

Synchronisieren Sie Änderungen zwischen einem lokalen Repo und einem Repository AWS CodeCommit

Sie verwenden Git, um Änderungen zwischen einem lokalen Repo und dem mit dem lokalen Repo verbundenen CodeCommit Repository zu synchronisieren.

Um Änderungen vom lokalen Repo in das CodeCommit Repository zu übertragen, führe den Befehl aus. `git push remote-name branch-name`

Um Änderungen am lokalen Repo aus dem CodeCommit Repository abzurufen, führen Sie den Befehl aus. `git pull remote-name branch-name`

Sowohl für Push als auch für Pull **remote-name** ist dies der Spitzname, den das lokale Repo für das CodeCommit Projektarchiv verwendet. **branch-name** ist der Name des Branches im CodeCommit Projektarchiv, zu dem gepusht oder von dem abgerufen werden soll.

Tip

Um den Spitznamen zu ermitteln, den das lokale Repo für das CodeCommit Repository verwendet, führen Sie `git remote` den Befehl aus. Um eine Liste der Branch-Namen zu erhalten, führen Sie den Befehl `git branch` aus. Ein Sternchen (*) wird neben dem aktuell geöffneten Branch angezeigt. (Sie können auch `git status` ausführen, um den aktuellen Branch-Namen anzuzeigen.)

Note

Wenn Sie das Repository geklont haben, **remote-name** ist das aus Sicht des lokalen Repos nicht der Name des Repositorys. CodeCommit Wenn Sie ein Repository klonen, **remote-name** wird automatisch auf `origin` gesetzt.

Um beispielsweise Änderungen aus dem lokalen Repository in den `main` Zweig im CodeCommit Repository mit dem Spitznamen `origin` zu übertragen:

```
git push origin main
```

Um auf ähnliche Weise Änderungen am lokalen Repo aus dem `main` Branch im CodeCommit Repository mit dem Spitznamen abzurufen: `origin`

```
git pull origin main
```

 Tip

Durch Hinzufügen der Option `-u` zum Befehl `git push` legen Sie Upstream-Nachverfolgungsdaten fest. Zum Beispiel, wenn du `rennstgit push -u origin main`), kannst du in future laufen `git push` und `git pull` ohne `remote-namebranch-name`. Führen Sie `git remote show remote-name` aus (z. B. `git remote show origin`), um Upstream-Nachverfolgungsdaten abzurufen.

Weitere Optionen findest du in deiner Git-Dokumentation.

Push Commits in ein zusätzliches Git-Repository

Sie können Ihr lokales Repository so konfigurieren, dass Änderungen mithilfe von Push an zwei Remote-Repositorys übertragen werden. Sie könnten beispielsweise Ihre vorhandene Git-Repository-Lösung weiterhin verwenden, während Sie AWS CodeCommit testen. Folgen Sie diesen grundlegenden Schritten, um Änderungen in Ihrem lokalen Repository in ein separates Git-Repository zu CodeCommit übertragen.

 Tip

Wenn Sie kein Git-Repository haben, können Sie ein leeres in einem anderen Dienst als diesem erstellen CodeCommit und dann Ihr CodeCommit Repository dorthin migrieren. Hierzu führen Sie Schritte aus, die den Anweisungen unter [Migrieren Sie zu CodeCommit](#) weitestgehend entsprechen.

1. Wechseln Sie über das Befehlszeilen- oder Terminalfenster zu Ihrem lokalen Repository-Verzeichnis und führen Sie den Befehl `git remote -v` aus. Die Ausgabe sollte folgendermaßen oder ähnlich aussehen:

Für HTTPS:

```
origin https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (fetch)
origin https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (push)
```

Für SSH:

```
origin  ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (fetch)
origin  ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (push)
```

2. Führen Sie den `git remote set-url --add --push origin git-repository-name` Befehl mit der URL und dem Namen des Git-Repositorys aus, in dem Sie Ihren Code hosten möchten. **git-repository-name** Hierdurch wird das Push-Ziel von origin in dieses Git-Repository geändert.

Note

Mit dem Befehl `git remote set-url --add --push` wird die Standard-URL für Pushes überschrieben, sodass Sie diesen Befehl zweimal ausführen müssen, wie weiter unten erläutert.

Mit dem folgenden Befehl wird beispielsweise der Push von origin auf **some-URL/** geändert `MyDestinationRepo`:

```
git remote set-url --add --push origin some-URL/MyDestinationRepo
```

Mit diesem Befehl wird kein Inhalt zurückgegeben.

Tip

Wenn Sie zu einem Git-Repository pushen, für das Anmeldeinformationen erforderlich sind, stellen Sie sicher, dass Sie diese Anmeldeinformationen in einem Credential Helper oder in der Konfiguration der **some-URL** Zeichenfolge konfigurieren. Andernfalls schlagen Ihre Pushes an dieses Repository fehl.

3. Führen Sie den Befehl `git remote -v` erneut aus. Die Befehlsausgabe sollte der folgenden ähneln:

Für HTTPS:

```
origin  https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (fetch)
origin  some-URL/MyDestinationRepo (push)
```

Für SSH:

```
origin  ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (fetch)
origin  some-URL/MyDestinationRepo (push)
```

4. Fügen Sie nun das CodeCommit Repository hinzu. Führen Sie den Befehl `git remote set-url --add --push origin` erneut aus, dieses Mal mit der URL und dem Namen Ihres CodeCommit-Repositorys.

Der folgende Befehl fügt beispielsweise den Push of origin zu `https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo`:

Für HTTPS:

```
git remote set-url --add --push origin https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo
```

Für SSH:

```
git remote set-url --add --push origin ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo
```

Mit diesem Befehl wird kein Inhalt zurückgegeben.

5. Führen Sie den Befehl `git remote -v` erneut aus. Die Befehlsausgabe sollte der folgenden ähneln:

Für HTTPS:

```
origin  https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (fetch)
origin  some-URL/MyDestinationRepo (push)
origin  https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (push)
```

Für SSH:

```
origin  ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (fetch)
origin  some-URL/MyDestinationRepo (push)
origin  ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (push)
```

Sie haben jetzt zwei Git-Repositorys als Ziel für Ihre Pushs, aber Ihre Pushs gehen zuerst zu **some-URL**/. MyDestinationRepo Wenn der Push an dieses Repository fehlschlägt, werden Ihre Commits nicht mithilfe von Push an die Repositorys übertragen.

 Tip

Wenn für das andere Repository Anmeldeinformationen erforderlich sind, die Sie manuell eingeben möchten, sollten Sie die Reihenfolge der Pushs so ändern, dass Sie zuerst zu pushen beginnen. CodeCommit Führen Sie den Befehl `git remote set-url --delete` aus, um das Repository zu löschen, an das Pushes zuerst ausgeführt werden. Führen Sie anschließend den Befehl `git remote set-url --add` aus, um das Repository erneut hinzuzufügen, sodass es zum zweiten Push-Ziel in der Liste wird. Weitere Optionen findest du in deiner Git-Dokumentation.

6. Um zu überprüfen, ob Sie nun einen Push an beide Remote-Repositorys ausführen, erstellen Sie mit einem Texteditor die folgende Textdatei in Ihrem lokalen Repository:

```
bees.txt
-----
Bees are flying insects closely related to wasps and ants, and are known for their
role in pollination and for producing honey and beeswax.
```

7. Führen Sie `git add` aus, um die Änderung per Stage an Ihr lokales Repository zu übertragen:

```
git add bees.txt
```

8. Führen Sie `git commit` aus, um die Änderung per Commit an Ihr lokales Repository zu übertragen:

```
git commit -m "Added bees.txt"
```

9. Um den Commit vom lokalen Repo in deine Remote-Repositorys zu übertragen, führe den Befehl aus, `git push -u remote-name branch-name` wo der Spitzname **remote-name** ist, den das lokale Repo für die Remote-Repositorys verwendet, und der Name des **branch-name** Branches ist, der in das Repository übertragen werden soll.

 Tip

Sie müssen die Option `-u` nur für die erste Push-Ausführung verwenden. Dann werden die Upstream-Nachverfolgungsdaten festgelegt.

Beispielsweise wird durch die Ausführung von `git push -u origin main` deutlich, dass der Push an beide Remote-Repositorys in den erwarteten Branches übertragen wurde. Die Ausgabe entspricht weitgehend der folgenden:

Für HTTPS:

```
Counting objects: 5, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 5.61 KiB | 0 bytes/s, done.
Total 3 (delta 1), reused 0 (delta 0)
To some-URL/MyDestinationRepo
   a5ba4ed..250f6c3  main -> main
Counting objects: 5, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 5.61 KiB | 0 bytes/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote:
To https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo
   a5ba4ed..250f6c3  main -> main
```

Für SSH:

```
Counting objects: 5, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 5.61 KiB | 0 bytes/s, done.
Total 3 (delta 1), reused 0 (delta 0)
To some-URL/MyDestinationRepo
   a5ba4ed..250f6c3  main -> main
Counting objects: 5, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (3/3), done.
```



```
Writing objects: 100% (3/3), 5.61 KiB | 0 bytes/s, done.  
Total 3 (delta 1), reused 0 (delta 0)  
remote:  
To ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo  
a5ba4ed..250f6c3  main -> main
```

Weitere Optionen findest du in deiner Git-Dokumentation.

Konfiguriere den kontoübergreifenden Zugriff auf ein AWS CodeCommit Repository mithilfe von Rollen

Sie können den Zugriff auf CodeCommit Repositorys für IAM-Benutzer und -Gruppen in einem anderen Konto konfigurieren. AWS Dieser Vorgang wird oft als kontoübergreifender Zugriff bezeichnet. Dieser Abschnitt enthält Beispiele und step-by-step Anweisungen für die Konfiguration des kontoübergreifenden Zugriffs auf ein Repository mit dem Namen *MySharedDemoRepo* in der Region USA Ost (Ohio) in einem AWS Konto (als AccountA bezeichnet) für IAM-Benutzer, die zu einer IAM-Gruppe gehören, die *DeveloPERSWithCrossAccountRepositoryAccess* in einem anderen AWS Konto benannt ist (als AccountB bezeichnet).

Dieser Abschnitt ist in drei Teile untergliedert:

- Aktionen für den Administrator in Konto A
- Aktionen für den Administrator in Konto B
- Aktionen für den Repository-Benutzer in Konto B

So konfigurieren den kontoübergreifenden Zugriff:

- Der Administrator in AccountA meldet sich als IAM-Benutzer mit den erforderlichen Berechtigungen an, um Repositorys in IAM zu erstellen und zu verwalten und Rollen in CodeCommit IAM zu erstellen. Wenn Sie verwaltete Richtlinien verwenden, wenden Sie IAMFull Access und auf diesen IAM-Benutzer AWSCode CommitFullAccess an.

Die Beispielkonto-ID für Konto A lautet *111122223333*.

- Der Administrator in AccountB meldet sich als IAM-Benutzer mit den erforderlichen Berechtigungen an, um IAM-Benutzer und -Gruppen zu erstellen und zu verwalten und Richtlinien für Benutzer und Gruppen zu konfigurieren. Wenn Sie verwaltete Richtlinien verwenden, wenden Sie IAMFull Access auf diesen IAM-Benutzer an.

Die Beispielkonto-ID für Konto B lautet **888888888888**.

- Um die Aktivitäten eines Entwicklers nachzuahmen, meldet sich der Repository-Benutzer in AccountB als IAM-Benutzer an, der Mitglied der IAM-Gruppe ist, die erstellt wurde, um den Zugriff auf das CodeCommit Repository in AccountA zu ermöglichen. Für dieses Konto müssen folgende Elemente konfiguriert sein:
 - AWS Zugriff auf die Managementkonsole.
 - Ein Zugriffsschlüssel und ein geheimer Schlüssel, die verwendet werden, wenn eine Verbindung zu AWS Ressourcen hergestellt wird, sowie der ARN der Rolle, die beim Zugriff auf Repositorys in AccountA verwendet werden soll.
 - Das git-remote-codecommit-Dienstprogramm auf dem lokalen Computer, auf dem das Repository geklont wird. Dieses Dienstprogramm erfordert Python und dessen Installationsprogramm „pip“. Sie können das Dienstprogramm von [git-remote-codecommit](#) auf der Python Package Index-Website herunterladen.

Weitere Informationen finden Sie unter [Einrichtungsschritte für HTTPS-Verbindungen AWS CodeCommit mit git-remote-codecommit](#) und [IAM-Benutzer](#).

Themen

- [Kontoübergreifender Zugriff auf das Repository: Aktionen für den Administrator in AccountA](#)
- [Kontoübergreifender Zugriff auf das Repository: Aktionen für den Administrator in AccountB](#)
- [Kontoübergreifender Repository-Zugriff: Aktionen für den Repository-Benutzer in AccountB](#)

Kontoübergreifender Zugriff auf das Repository: Aktionen für den Administrator in AccountA

Um Benutzern oder Gruppen in Konto B den Zugriff auf ein Repository in Konto A zu erlauben, muss der Administrator von Konto A folgende Schritte ausführen:

- Erstellen einer Richtlinie in Konto A, die Zugriff auf das Repository gewährt
- Erstellen Sie eine Rolle in AccountA, die von IAM-Benutzern und Gruppen in AccountB übernommen werden kann.
- Fügen Sie der Rolle die -Richtlinie an.

In den folgenden Abschnitten werden die Schritte dazu mit Beispielen beschrieben.

Themen

- [Schritt 1: Erstellen Sie eine Richtlinie für den Repository-Zugriff in AccountA](#)
- [Schritt 2: Erstellen Sie eine Rolle für den Repository-Zugriff in AccountA](#)

Schritt 1: Erstellen Sie eine Richtlinie für den Repository-Zugriff in AccountA

Sie können in AccountA eine Richtlinie erstellen, die Benutzern in KontoB Zugriff auf das Repository in AccountA gewährt. Abhängig von der Zugriffsebene, die Sie zulassen möchten, führen Sie einen der folgenden Schritte aus:

- Konfigurieren Sie die Richtlinie so, dass Benutzern in Konto B Zugriff auf ein bestimmtes Repository gewährt wird. Erlauben Sie ihnen aber nicht, eine Liste aller Repositories in Konto A anzuzeigen.
- Konfigurieren Sie zusätzlichen Zugriff, um Benutzern in Konto B die Auswahl des Repositories aus einer Liste aller Repositories in Konto A zu erlauben.

So erstellen Sie eine Richtlinie für den Repository-Zugriff

1. Melden Sie sich bei der AWS Management Console als IAM-Benutzer mit Berechtigungen zum Erstellen von Richtlinien in AccountA an.
2. Öffnen Sie unter <https://console.aws.amazon.com/iam/> die IAM-Konsole.
3. Wählen Sie im Navigationsbereich Richtlinien.
4. Wählen Sie Richtlinie erstellen aus.
5. Wählen Sie die Registerkarte JSON aus und kopieren Sie das folgende JSON-Richtliniendokument in das JSON-Textfeld. *us-east-2* Ersetzen Sie durch den AWS-Region für das Repository, *111122223333* durch die Konto-ID für AccountA und *MySharedDemoRepo* durch den Namen für Ihr CodeCommit Repository in AccountA:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codecommit:BatchGet*",

```

```

        "codecommit:Create*",
        "codecommit>DeleteBranch",
        "codecommit:Get*",
        "codecommit:List*",
        "codecommit:Describe*",
        "codecommit:Put*",
        "codecommit:Post*",
        "codecommit:Merge*",
        "codecommit:Test*",
        "codecommit:Update*",
        "codecommit:GitPull",
        "codecommit:GitPush"
    ],
    "Resource": [
        "arn:aws:codecommit:us-east-2:111122223333:MySharedDemoRepo"
    ]
}
]
}
```

Wenn Sie möchten, dass Benutzer, die diese Rolle übernehmen, eine Liste von Repositorys auf der CodeCommit Konsolen-Startseite einsehen können, fügen Sie der Richtlinie eine zusätzliche Erklärung hinzu, die wie folgt lautet:

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codecommit:BatchGet*",
                "codecommit:Create*",
                "codecommit>DeleteBranch",
                "codecommit:Get*",
                "codecommit:List*",
                "codecommit:Describe*",
                "codecommit:Put*",
                "codecommit:Post*",
                "codecommit:Merge*",
                "codecommit:Test*",
                "codecommit:Update*",
                "codecommit:GitPull",
                "codecommit:GitPush"
            ]
        }
    ]
}
```

```
    ],
    "Resource": [
        "arn:aws:codecommit:us-east-2:111122223333:MySharedDemoRepo"
    ]
  },
  {
    "Effect": "Allow",
    "Action": "codecommit:ListRepositories",
    "Resource": "*"
  }
]
```

Dieser Zugriff macht es für Benutzer, die diese Rolle übernehmen, mit dieser Richtlinie einfacher, das Repository zu finden, auf das sie Zugriff haben. Sie können den Namen des Repositorys aus der Liste auswählen und gelangen dann auf die Startseite des freigegebenen Repositorys (Code). Die Benutzer können nicht auf andere in der Liste angezeigten Repositorys zugreifen, sie können aber die Repositorys in Konto A auf der Seite Dashboard anzeigen.

Wenn Sie nicht möchten, dass Benutzer, die diese Rolle übernehmen, eine Liste aller Repositorys in AccountA einsehen können, verwenden Sie das erste Richtlinienbeispiel, stellen Sie jedoch sicher, dass Sie diesen Benutzern einen direkten Link zur Startseite des gemeinsam genutzten Repositorys in der CodeCommit Konsole senden.

6. Wählen Sie Richtlinie prüfen. Der Richtlinienvalidator meldet Syntaxfehler (wenn Sie beispielsweise vergessen, die Amazon Web Services Services-Konto-ID und den Repository-Namen durch Ihre Amazon Web Services Services-Konto-ID und Ihren Repository-Namen zu ersetzen).
7. Geben Sie auf der Seite „Richtlinie überprüfen“ einen Namen für die Richtlinie ein (z. B. *CrossAccountAccessForMySharedDemoRepo*). Sie können optional auch eine Beschreibung für die Richtlinie eingeben. Wählen Sie Richtlinie erstellen aus.

Schritt 2: Erstellen Sie eine Rolle für den Repository-Zugriff in AccountA

Nachdem Sie eine Richtlinie konfiguriert haben, erstellen Sie eine Rolle, die IAM-Benutzer und -Gruppen in AccountB annehmen können, und fügen Sie die Richtlinie dieser Rolle hinzu.

So erstellen Sie eine Rolle für den Repository-Zugriff

1. Wählen Sie in der IAM-Konsole Roles (Rollen) aus.

2. Wählen Sie Rolle erstellen aus.
3. Wählen Sie ein anderes Amazon Web Services Services-Konto.
4. Geben Sie unter Konto-ID die Amazon Web Services Services-Konto-ID für AccountB ein (z. B. **888888888888**). Wählen Sie Weiter: Berechtigungen aus.
5. Wählen Sie unter Zugriffsrichtlinien anhängen die Richtlinie aus, die Sie im vorherigen Verfahren erstellt haben (***CrossAccountAccessForMySharedDemoRepo***). Wählen Sie Weiter: Prüfen aus.
6. Geben Sie im Feld Rollenname einen Namen für die Rolle ein (z. B. ***MyCrossAccountRepositoryContributorRole***). Sie können optional auch eine Beschreibung eingeben, um anderen den Zweck dieser Rolle mitzuteilen.
7. Wählen Sie Rolle erstellen aus.
8. Öffnen Sie die Rolle, die Sie gerade erstellt haben, und kopieren Sie den ARN der Rolle (z. B. ***arn:aws:iam::111122223333:role/MyCrossAccountRepositoryContributorRole***). Sie müssen diesen ARN dem Administrator von Konto B mitteilen.

Kontoübergreifender Zugriff auf das Repository: Aktionen für den Administrator in AccountB

Um Benutzern oder Gruppen in Konto B den Zugriff auf ein Repository in Konto A zu erlauben, muss der Administrator von Konto B eine Gruppe in Konto B erstellen. Diese Gruppe muss mit einer Richtlinie konfiguriert werden, die es den Gruppenmitgliedern erlaubt, die vom Administrator von Konto A erstellte Rolle zu übernehmen.

In den folgenden Abschnitten werden die Schritte dazu mit Beispielen beschrieben.

Themen

- [Schritt 1: Erstellen Sie eine IAM-Gruppe für den Repository-Zugriff für AccountB-Benutzer](#)
- [Schritt 2: Erstellen Sie eine Richtlinie und fügen Sie Benutzer zur IAM-Gruppe hinzu](#)

Schritt 1: Erstellen Sie eine IAM-Gruppe für den Repository-Zugriff für AccountB-Benutzer

Die einfachste Methode, um zu verwalten, welche IAM-Benutzer in AccountB auf das AccountA-Repository zugreifen können, besteht darin, eine IAM-Gruppe in AccountB zu erstellen, die über die

Berechtigung verfügt, die Rolle in AccountA anzunehmen, und dann die IAM-Benutzer zu dieser Gruppe hinzuzufügen.

So erstellen Sie eine Gruppe für den kontoübergreifenden Repository-Zugriff

1. Melden Sie sich bei der AWS Management Console als IAM-Benutzer mit den erforderlichen Berechtigungen an, um IAM-Gruppen und -Richtlinien zu erstellen und IAM-Benutzer in AccountB zu verwalten.
2. Öffnen Sie unter <https://console.aws.amazon.com/iam/> die IAM-Konsole.
3. Wählen Sie in der IAM-Konsole Gruppen aus.
4. Wählen Sie Create New Group (Neue Gruppe erstellen).
5. Geben Sie im Feld Gruppenname einen Namen für die Gruppe ein (z. B. *DevelopersWithCrossAccountRepositoryAccess*). Wählen Sie Next Step (Weiter) aus.
6. In Richtlinien anfügen wählen Sie Next Step aus. Sie erstellen die kontoübergreifende Richtlinie im nächsten Verfahren. Schließen Sie die Erstellung der Gruppe ab.

Schritt 2: Erstellen Sie eine Richtlinie und fügen Sie Benutzer zur IAM-Gruppe hinzu

Nachdem Sie nun über eine Gruppe verfügen, erstellen Sie die Richtlinie, die es den Mitgliedern dieser Gruppe erlaubt, die Rolle zu übernehmen, die ihnen Zugriff auf das Repository in Konto A gewährt. Fügen Sie dann der Gruppe die IAM-Benutzer in AccountB hinzu, denen Sie den Zugriff in AccountA gewähren möchten.

So erstellen Sie eine Richtlinie für die Gruppe und fügen dieser Benutzer hinzu

1. Wählen Sie in der IAM-Konsole Gruppen und dann den Namen der Gruppe aus, die Sie gerade erstellt haben (z. B.). *DevelopersWithCrossAccountRepositoryAccess*
2. Wählen Sie die Registerkarte Berechtigungen. Erweitern Sie Inline Policies (Eingebundene Richtlinien) und wählen Sie den Link, um eine eingebundene Richtlinie zu erstellen. (Wenn Sie eine Gruppe konfigurieren, die bereits über eine eingebundene Richtlinie verfügt, wählen Sie Create Group Policy (Gruppenrichtlinie erstellen).)
3. Wählen Sie Custom Policy und dann Select aus.
4. Geben Sie unter Policy Name (Richtliniennamen) einen Namen für die Richtlinie ein (zum Beispiel *AccessPolicyForSharedRepository*).
5. Fügen Sie die folgende Richtlinie unter Policy Document (Richtliniendokument) ein. Ersetzen Sie in den ARN durch den ARN der Richtlinie **Resource**, die vom Administrator in AccountA

erstellt wurde (z. B. `arn:aws:iam:111122223333:role/`), und wählen Sie dann Apply Policy aus. ***MyCrossAccountRepositoryContributorRole*** Weitere Informationen zu der Richtlinie, die vom Administrator von Konto A erstellt wurde, finden Sie unter [Schritt 1: Erstellen Sie eine Richtlinie für den Repository-Zugriff in AccountA](#).

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource":
      "arn:aws:iam::111122223333:role/MyCrossAccountRepositoryContributorRole"
  }
}
```

- Wählen Sie die Registerkarte Users. Wählen Sie Benutzer zur Gruppe hinzufügen und fügen Sie dann die AccountB IAM-Benutzer hinzu. Sie können der Gruppe beispielsweise einen IAM-Benutzer mit dem Benutzernamen ***Saanvi_Sarkar*** hinzufügen.

Note

Benutzer in AccountB benötigen programmatischen Zugriff, einschließlich eines Zugriffsschlüssels und eines geheimen Schlüssels, um ihre lokalen Computer für den Zugriff auf das gemeinsam genutzte CodeCommit Repository zu konfigurieren. Wenn Sie IAM-Benutzer erstellen, achten Sie darauf, den Zugriffsschlüssel und den geheimen Schlüssel zu speichern. Damit die Sicherheit des AWS -Kontos gewährleistet ist, kann auf den geheimen Zugriffsschlüssel nur während seiner Erstellung zugegriffen werden.

Kontoübergreifender Repository-Zugriff: Aktionen für den Repository-Benutzer in AccountB

Um auf das Repository in Konto A zugreifen zu können, müssen die Benutzer in der Gruppe von Konto B ihre lokalen Computer für den Repository-Zugriff konfigurieren. In den folgenden Abschnitten werden die Schritte dazu mit Beispielen beschrieben.

Themen

- [Schritt 1: Konfiguration von AWS CLI und Git für einen AccountB-Benutzer, um auf das Repository in AccountA zuzugreifen](#)

- [Schritt 2: Klonen Sie das CodeCommit Repository in AccountA und greifen Sie darauf zu](#)

Schritt 1: Konfiguration von AWS CLI und Git für einen AccountB-Benutzer, um auf das Repository in AccountA zuzugreifen

Sie können SSH-Schlüssel oder Git-Anmeldeinformationen nicht verwenden, um auf Repositories in einem anderen Amazon Web Services Services-Konto zuzugreifen. AccountB-Benutzer müssen ihre Computer so konfigurieren, dass sie entweder git-remote-codecommit (empfohlen) oder den Credential Helper verwenden, um auf das gemeinsame CodeCommit Repository in AccountA zuzugreifen. Sie können jedoch weiterhin SSH-Schlüssel oder Git-Anmeldeinformationen verwenden, wenn Sie auf Repositories in Konto B zugreifen.

Gehen Sie folgendermaßen vor, um den Zugriff mithilfe von git-remote-codecommit zu konfigurieren. Falls Sie es noch nicht installiert haben git-remote-codecommit, laden Sie es von [git-remote-codecommit](#) der Python Package Index-Website herunter.

Um Git AWS CLI und Git für den kontoübergreifenden Zugriff zu konfigurieren

1. Installieren Sie das AWS CLI auf dem lokalen Computer. Anleitungen für Ihr Betriebssystem finden Sie unter [Installieren der AWS CLI](#).
2. Installieren Sie Git auf dem lokalen Computer. Für die Installation von Git empfehlen wir Websites wie [Git Downloads](#) oder [Git for Windows](#).

Note

CodeCommit unterstützt Git-Versionen 1.7.9 und höher. Git Version 2.28 unterstützt die Konfiguration des Branchnamens für anfängliche Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git. Git ist eine sich entwickelnde, regelmäßig aktualisierte Plattform. Gelegentlich kann sich eine Änderung der Funktionen auf die Art und Weise auswirken, mit der es funktioniert CodeCommit. Wenn du Probleme mit einer bestimmten Version von Git und hast CodeCommit, sieh dir die Informationen unter [anFehlerbehebung](#).

3. Führen Sie am Terminal oder über die Befehlszeile an dem Speicherort, an dem Sie das Repository klonen möchten, die Befehle git config --local user.name und git config --local user.email aus, um den Benutzernamen und die E-Mail-Adresse für die Commits für das Repository anzugeben. Zum Beispiel:

```
git config --local user.name "Saanvi Sarkar"  
git config --local user.email saanvi_sarkar@example.com
```

Diese Befehle geben nichts zurück. Die E-Mail-Adresse und der Benutzername, die Sie angeben, werden aber mit den Commits verknüpft, die Sie für das Repository in Konto A vornehmen.

4. Führen Sie den Befehl `aws configure --profile` aus, um ein Standardprofil für das Herstellen einer Verbindung zu Ressourcen in Konto B zu konfigurieren. Wenn Sie dazu aufgefordert werden, geben Sie den Zugriffsschlüssel und den geheimen Schlüssel für Ihren IAM-Benutzer ein.

 Note

Wenn Sie das Profil bereits installiert AWS CLI und konfiguriert haben, können Sie diesen Schritt überspringen.

Führen Sie beispielsweise den folgenden Befehl aus, um ein AWS CLI Standardprofil zu erstellen, das Sie für den Zugriff auf AWS Ressourcen in AccountB in US East (Ohio) (us-east-2) verwenden:

```
aws configure
```

Wenn Sie dazu aufgefordert werden, geben Sie die folgenden Informationen an:

```
AWS Access Key ID [None]: Your-IAM-User-Access-Key  
AWS Secret Access Key ID [None]: Your-IAM-User-Secret-Access-Key  
Default region name ID [None]: us-east-2  
Default output format [None]: json
```

5. Führen Sie den Befehl `aws configure --profile` erneut aus, um ein Profil zu konfigurieren, das für die Verbindung mit dem Repository in Konto A verwendet werden soll. Wenn Sie dazu aufgefordert werden, geben Sie den Zugriffsschlüssel und den geheimen Schlüssel für Ihren IAM-Benutzer ein. Führen Sie beispielsweise den folgenden Befehl aus, um ein AWS CLI Profil mit dem Namen zu erstellen *MyCrossAccountAccessProfile*, das Sie für den Zugriff auf ein Repository in AccountA in US East (Ohio) (us-east-2) verwenden:

```
aws configure --profile MyCrossAccountAccessProfile
```

Wenn Sie dazu aufgefordert werden, geben Sie die folgenden Informationen an:

```
AWS Access Key ID [None]: Your-IAM-User-Access-Key
AWS Secret Access Key ID [None]: Your-IAM-User-Secret-Access-Key
Default region name ID [None]: us-east-2
Default output format [None]: json
```

6. Öffnen Sie in einem Texteditor die Datei config (die AWS CLI -Konfigurationsdatei). Je nach Betriebssystem befindet sich diese Datei `~/.aws/config` unter Linux, macOS oder Unix oder unter `drive:\ Users\ USERNAME .aws\ config` unter Windows.
7. Suchen Sie in der Datei den Eintrag, der dem Standardprofil entspricht, das Sie für den Zugriff auf Repositorys in Konto B konfiguriert haben. Das sollte bei Ihnen ähnlich wie im folgenden Bild aussehen:

```
[default]
region = us-east-2
output = json
```

Fügen Sie account der Profilkonfiguration hinzu. Geben Sie die AWS Konto-ID von AccountB an. Zum Beispiel:

```
[default]
account = 888888888888
region = us-east-2
output = json
```

8. Suchen Sie in der Datei den Eintrag, der dem *MyCrossAccountAccessProfile* Profil entspricht, das Sie gerade erstellt haben. Das sollte bei Ihnen ähnlich wie im folgenden Bild aussehen:

```
[profile MyCrossAccountAccessProfile]
region = us-east-2
output = json
```

Fügen Sie account, role_arn und source_profile der Profilkonfiguration hinzu. Geben Sie die Amazon Web Services Services-Konto-ID von AccountA, den ARN der Rolle in AccountA, von der Sie annehmen, dass sie auf das Repository in dem anderen Konto zugreift, und den Namen Ihres AWS CLI Standardprofils in AccountB an. Zum Beispiel:

```
[profile MyCrossAccountAccessProfile]  
region = us-east-2  
account = 111122223333  
role_arn = arn:aws:iam::111122223333:role/MyCrossAccountRepositoryContributorRole  
source_profile = default  
output = json
```

Speichern Sie Ihre Änderungen und schließen Sie den Texteditor.

Schritt 2: Klonen Sie das CodeCommit Repository in AccountA und greifen Sie darauf zu

Führen Sie `git clone`, `git push` und `git pull` aus, um das kontoübergreifende CodeCommit-Repository zu klonen und darauf eine Push- und Pull-Operation auszuführen. Sie können sich auch bei der AWS Management Console anmelden, die Rollen wechseln und die CodeCommit Konsole verwenden, um mit dem Repository des anderen Kontos zu interagieren.

Note

Je nachdem, wie die IAM-Rolle konfiguriert wurde, können Sie Repositories möglicherweise auf der Standardseite für anzeigen. CodeCommit Wenn Sie die Repositories nicht einsehen können, bitten Sie den Repository-Administrator, Ihnen per E-Mail einen URL-Link zur Codepage für das gemeinsam genutzte Repository in der Konsole zu senden. CodeCommit Die URL sollte in etwa wie folgt aussehen:

```
https://console.aws.amazon.com/codecommit/home?region=us-east-2#/  
repository/MySharedDemoRepo/browse/HEAD/--/
```

So klonen Sie das kontoübergreifende Repository auf Ihrem lokalen Computer

1. Führen Sie über die Befehlszeile oder am Terminal in dem Verzeichnis, in dem Sie das Repository klonen möchten, den Befehl `git clone` mit der HTTPS (GRC)-Klon-URL aus. Zum Beispiel:

```
git clone codecommit://MyCrossAccountAccessProfile@MySharedDemoRepo
```

Sofern Sie nichts anderes angeben, wird das Repository in ein Unterverzeichnis mit demselben Namen wie das Repository geklont.

2. Ändern Sie die Verzeichnisse zum geklonten Repository und fügen Sie eine Datei hinzu oder nehmen Sie eine Änderung an einer Datei vor. Sie können beispielsweise eine Datei mit dem Namen *NewFile.txt* hinzufügen.
3. Fügen Sie die Datei zu den verfolgten Änderungen für das lokale Repository hinzu, übertragen Sie die Änderung und übertragen Sie die Datei per Push in das CodeCommit Repository. Zum Beispiel:

```
git add NewFile.txt
git commit -m "Added a file to test cross-account access to this repository"
git push
```

Weitere Informationen finden Sie unter [Erste Schritte mit Git und AWS CodeCommit](#).

Nachdem du eine Datei hinzugefügt hast, kannst du in der CodeCommit Konsole deinen Commit ansehen, die Änderungen anderer Benutzer am Repo überprüfen, an Pull-Requests teilnehmen und vieles mehr.

So greifen Sie in der Konsole auf das kontoübergreifende Repository zu CodeCommit

1. Melden Sie sich AWS Management Console bei AccountB (*888888888888*) als der IAM-Benutzer an, dem kontoübergreifender Zugriff auf das Repository in AccountA gewährt wurde.
2. Wählen Sie in der Navigationsleiste Ihren Benutzernamen aus. Wählen Sie dann in der Dropdown-Liste Rollen wechseln aus.

 Note

Wenn Sie diese Option zum ersten Mal auswählen, überprüfen Sie die Informationen auf der Seite und klicken Sie dann erneut auf Rollen wechseln.

3. Führen Sie auf der Seite Rollen wechseln die folgenden Schritte aus:
 - Geben Sie unter Account ID (Konto-ID) die Konto-ID für Konto A an (z. B. *111122223333*).
 - Geben Sie im Feld Rolle den Namen der Rolle ein, die Sie für den Zugriff auf das Repository in AccountA übernehmen möchten (z. B. *MyCrossAccountRepositoryContributorRole*).

- Geben Sie unter Display Name (Anzeigenamen) einen Anzeigenamen für die Rolle ein. Dieser Name wird in der Konsole angezeigt, wenn Sie diese Rolle übernehmen. Er wird auch in der Liste der angenommenen Rollen angezeigt, wenn Sie in der Konsole das nächste Mal die Rolle wechseln möchten.
- (Optional) Wählen Sie unter Farbe eine Farbkennzeichnung für den Anzeigenamen aus.
- Wählen Sie Switch Role.

Weitere Informationen finden Sie unter [Wechseln zu einer Rolle \(AWS Management Console\)](#).

4. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.

Wenn die angenommene Rolle berechtigt ist, die Namen von Repositorys in Konto A anzuzeigen, sehen Sie eine Liste der Repositorys sowie eine Fehlermeldung, die Sie darüber informiert, dass Sie keine Berechtigungen zum Anzeigen der zugehörigen Statuswerte haben. Dieses Verhalten wird erwartet. Wählen Sie den Namen des freigegebenen Repositorys aus der Liste aus.

Wenn die angenommene Rolle nicht berechtigt ist, die Namen von Repositorys in Konto A anzuzeigen, sehen Sie eine Fehlermeldung und eine leere Liste ohne Repositorys. Fügen Sie den URL-Link zum Repository ein oder ändern Sie den Konsolenlink und ändern Sie `/list` in den Namen des freigegebenen Repositorys (z. B. `/MySharedDemoRepo`).

5. Suchen Sie unter Code den Namen der Datei, die Sie vom lokalen Computer hinzugefügt haben. Wählen Sie die Datei aus und durchsuchen Sie den Code in der Datei. Durchsuchen Sie dann das restliche Repository und beginnen Sie mit der Verwendung der Funktionen.

Weitere Informationen finden Sie unter [Erste Schritte mit AWS CodeCommit](#).

Lösche ein Repository AWS CodeCommit

Sie können die CodeCommit Konsole oder die verwenden AWS CLI , um ein CodeCommit Repository zu löschen.

 Note

Beim Löschen eines Repositorys werden lokale Kopien dieses Repositorys (lokale Repos) nicht gelöscht. Um ein lokales Repo zu löschen, verwenden Sie die Verzeichnis- und Dateiverwaltungstools Ihres lokalen Computers.

Themen

- [Löschen Sie ein CodeCommit Repository \(Konsole\)](#)
- [Löscht ein lokales Repo](#)
- [Löscht ein CodeCommit Repository \(AWS CLI\)](#)

Löschen Sie ein CodeCommit Repository (Konsole)

Gehen Sie wie folgt vor, um mit der CodeCommit Konsole ein CodeCommit Repository zu löschen.

 Important

Nachdem Sie ein CodeCommit Repository gelöscht haben, können Sie es nicht mehr in ein lokales oder gemeinsam genutztes Repository klonen. Sie sind auch nicht mehr in der Lage, Daten aus einem lokalen oder gemeinsam genutzten Repo daraus abzurufen oder Daten dorthin zu übertragen. Diese Aktion kann nicht rückgängig gemacht werden.

1. [Öffnen Sie die CodeCommit Konsole unter `https://console.aws.amazon.com/codesuite/codecommit/home`.](https://console.aws.amazon.com/codesuite/codecommit/home)
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, das Sie löschen möchten.
3. Wählen Sie im Navigationsbereich Settings (Einstellungen).
4. Wählen Sie auf der Registerkarte General (Allgemein) unter Delete repository (Repository löschen) die Option Delete repository (Repository löschen) aus. Geben Sie **delete** ein und klicken Sie auf Delete (Löschen). Das Repository wird dauerhaft gelöscht.

 Note

Durch das Löschen des Repositorys in werden CodeCommit keine lokalen Repos gelöscht.

Löscht ein lokales Repo

Verwenden Sie die Verzeichnis- und Dateiverwaltungstools Ihres lokalen Computers, um das Verzeichnis zu löschen, das das lokale Repository enthält.

Durch das Löschen eines lokalen Repos wird kein CodeCommit Repository gelöscht, mit dem es möglicherweise verbunden ist.

Löscht ein CodeCommit Repository ()AWS CLI

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter [Befehlszeilenreferenz](#).

AWS CLI Um das zum Löschen eines CodeCommit Repositorys zu verwenden, führen Sie den delete-repository Befehl aus und geben Sie den Namen des zu löschenden CodeCommit Repositorys an (mit der --repository-name Option).

 Important

Nachdem Sie ein CodeCommit Repository gelöscht haben, können Sie es nicht mehr in ein lokales oder gemeinsam genutztes Repository klonen. Sie sind auch nicht mehr in der Lage, Daten aus einem lokalen oder gemeinsam genutzten Repo daraus abzurufen oder Daten dorthin zu übertragen. Diese Aktion kann nicht rückgängig gemacht werden.

 Tip

Um den Namen des CodeCommit Repositorys abzurufen, führen Sie den Befehl [list-repositories](#) aus.

Beispielsweise löschen Sie das Repository MyDemoRepo wie folgt:


```
aws codecommit delete-repository --repository-name MyDemoRepo
```

Bei Erfolg erscheint die ID des CodeCommit Repositorys, das dauerhaft gelöscht wurde, in der Ausgabe:

```
{  
  "repositoryId": "f7579e13-b83e-4027-aaef-650c0EXAMPLE"  
}
```

Durch das Löschen eines CodeCommit Repositorys werden keine lokalen Repos gelöscht, die möglicherweise damit verbunden sind.

Arbeiten mit Dateien in AWS CodeCommit Repositorys

In CodeCommit ist eine Datei eine versionsgesteuerte, eigenständige Information, die Ihnen und anderen Benutzern des Repositorys und der Filiale, in der die Datei gespeichert ist, zur Verfügung steht. Sie können Ihre Repository-Dateien mithilfe einer Verzeichnisstruktur wie auf einem Computer strukturieren. Verfolgt im Gegensatz zu Ihrem Computer CodeCommit automatisch jede Änderung an einer Datei. Sie können die Versionen einer Datei vergleichen und verschiedene Versionen einer Datei in verschiedenen Branches des Repositorys speichern.

Um eine Datei in einem Repository hinzuzufügen oder zu bearbeiten, können Sie einen Git-Client verwenden. Sie können auch die CodeCommit Konsole AWS CLI, die oder die CodeCommit API verwenden.

The screenshot shows the 'Create a file' interface in the AWS CodeCommit console. At the top, a breadcrumb trail reads: 'Developer Tools > CodeCommit > Repositories > MyDemoRepo > File'. A dropdown menu on the right is set to 'main'. The main area is titled 'MyDemoRepo Info' and contains a code editor with the following Python code:

```
1 import sys
2
3 print('Hello, World!')
4
5 print('The sum of 2 and 3 is 5.')
6
7 sum = int(sys.argv[1]) + int(sys.argv[2])
8
9 print('The sum of {0} and {1} is {2}.'.format(sys.argv[1], sys.argv[2], sum))
```

Below the code editor, there is a section titled 'Commit changes to master'. It contains two input fields: 'File name' and 'Author name'. The 'File name' field has a hint 'For example, file.txt' and contains the text 'samples/hello.py'. Below this field, the path 'MyDemoRepo/samples/hello.py' is displayed. The 'Author name' field contains the text 'Saanvi Sarkar'.

Hinweise zur Arbeit mit anderen Aspekten Ihres Repositorys finden Sie unter CodeCommit [Arbeiten mit Repositorien](#), [Verwenden von Pull-Anforderungen](#), [Arbeiten mit Zweigen](#), [Mit Commits arbeiten](#), und [Mit Benutzereinstellungen arbeiten](#).

Themen

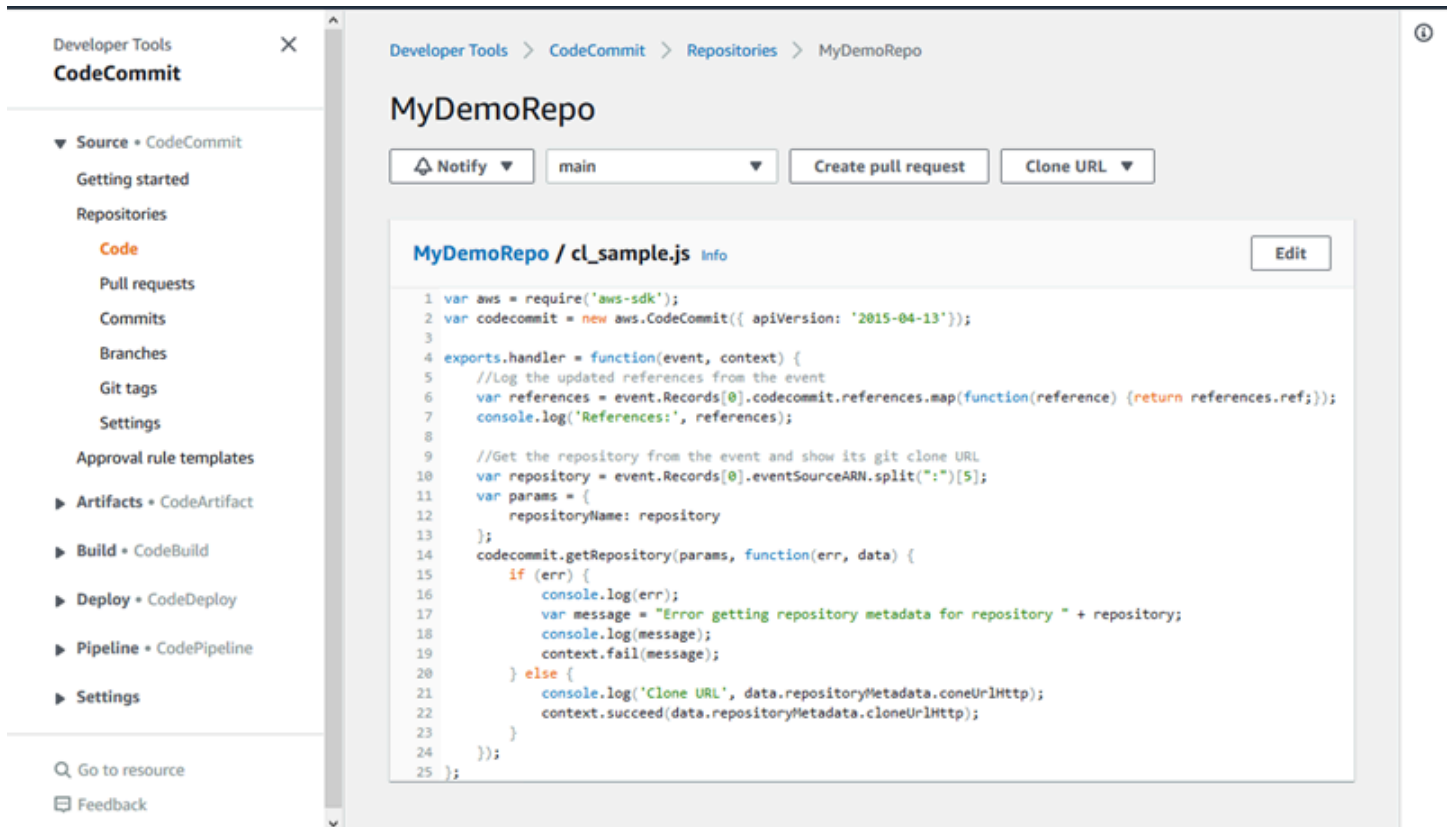
- [Dateien in einem AWS CodeCommit Repository durchsuchen](#)
- [Eine Datei erstellen oder zu einem AWS CodeCommit Repository hinzufügen](#)
- [Den Inhalt einer Datei in einem AWS CodeCommit Repository bearbeiten](#)

Dateien in einem AWS CodeCommit Repository durchsuchen

Nachdem Sie eine Verbindung zu einem CodeCommit Repository hergestellt haben, können Sie es in ein lokales Repository klonen oder die CodeCommit Konsole verwenden, um seinen Inhalt zu durchsuchen. In diesem Thema wird beschrieben, wie Sie die CodeCommit Konsole verwenden, um den Inhalt eines CodeCommit Repositorys zu durchsuchen.

Note

Für aktive CodeCommit Benutzer fallen keine Gebühren für das Durchsuchen von Code über die CodeCommit Konsole an. Weitere Informationen zu den Fällen, wann Gebühren anfallen können, finden Sie unter [Preise](#).



Ein CodeCommit Repository durchsuchen

Sie können die CodeCommit Konsole verwenden, um die in einem Repository enthaltenen Dateien zu überprüfen oder den Inhalt einer Datei schnell zu lesen.

So durchsuchen Sie den Inhalt eines Repositorys

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie auf der Seite Repositories (Repositorys) in der Liste der Repositorys das Repository aus, das Sie durchsuchen möchten.
3. Durchsuchen Sie in der Codeansicht die Inhalte des Standard-Branches für Ihr Repository.

Um einen anderen Branch oder Tag in der Ansicht anzuzeigen, klicken Sie auf die Schaltfläche der Ansichtsauswahl. Wählen Sie entweder in der Dropdown-Liste einen Branch- oder Tag-Namen aus oder geben Sie diesen im Filterfeld ein und wählen Sie ihn dann in der Liste aus.

4. Führen Sie eine der folgenden Aktionen aus:

- Um den Inhalt eines Verzeichnisses anzuzeigen, wählen Sie es in der Liste aus. Sie können in der Navigationsliste ein beliebiges Verzeichnis auswählen, um zu dieser Verzeichnisansicht zurückzukehren. Sie können auch den Aufwärtspfeil oben in der Verzeichnisliste verwenden.
- Um den Inhalt einer Datei anzuzeigen, wählen Sie sie in der Liste aus. Wenn die Größe der Datei den Grenzwert des Commit-Objekts überschreitet, kann sie nicht in der Konsole angezeigt werden. Stattdessen muss sie in einem lokalen Repository aufgerufen werden. Weitere Informationen finden Sie unter [Kontingente](#). Zum Beenden der Dateiansicht wählen Sie in der Code-Navigationsleiste das anzuzeigende Verzeichnis aus.

Note

Nicht alle Binärdateien können in der Konsole angezeigt werden. Wenn Sie eine Binärdatei auswählen und diese möglicherweise angezeigt werden kann, wird eine Warnmeldung angezeigt, in der Sie aufgefordert werden, zu bestätigen, dass Sie den Inhalt anzeigen möchten. Um die Datei anzuzeigen, wählen Sie die Option Show file contents aus. Wenn Sie die Datei nicht anzeigen möchten, wählen Sie in der Code-Navigationsleiste das anzuzeigende Verzeichnis aus.

Wenn Sie eine Markdown-Datei (.md) auswählen, klicken Sie auf die Schaltflächen Rendered Markdown und Markdown Source, um zwischen der gerenderten Ansicht und der Syntaxansicht zu wechseln. Weitere Informationen finden Sie unter [Verwenden von Markdown in der Konsole](#).

Eine Datei erstellen oder zu einem AWS CodeCommit Repository hinzufügen

Du kannst die CodeCommit Konsole oder einen Git-Client verwenden AWS CLI, um eine Datei zu einem Repository hinzuzufügen. Sie können eine Datei vom lokalen Computer in das Repository hochladen oder den Code-Editor in der Konsole zum Erstellen der Datei verwenden. Der Editor ist eine schnelle und einfache Möglichkeit zum Hinzufügen einer einfachen Datei (z. B. einer readme.md-Datei) zu einem Branch in einem Repository.

Upload a file

MyDemoRepo [Info](#)

Name	Size	Actions
Upload file Choose a file to upload. Choose file		

Commit changes to master

Author name

Email address

Commit message - optional
A default commit message will be used if you do not provide one.

[Cancel](#) [Commit changes](#)

Themen

- [Eine Datei erstellen oder hochladen \(Konsole\)](#)
- [Füge eine Datei hinzu \(AWS CLI\)](#)
- [Eine Datei hinzufügen \(Git\)](#)

Eine Datei erstellen oder hochladen (Konsole)

Sie können die CodeCommit Konsole verwenden, um eine Datei zu erstellen und sie einem Zweig in einem CodeCommit Repository hinzuzufügen. Beim Erstellen der Datei können Sie Ihren Benutzernamen und eine E-Mail-Adresse angeben. Sie können auch eine Commit-Nachricht hinzufügen, sodass andere Benutzer wissen, wer die Datei hinzugefügt hat und aus welchem Grund. Sie können eine Datei auch direkt von Ihrem lokalen Computer in einen Branch in einem Repository hochladen.

So fügen Sie eine Datei zu einem Repository hinzu

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.

2. Wählen Sie unter Repositories (Repositorys) das Repository aus, in dem Sie eine Datei hinzufügen möchten.
3. Wählen Sie in der Codeansicht den Branch aus, in dem Sie die Datei hinzufügen möchten. Standardmäßig werden beim Öffnen der Codeansicht die Inhalte der Standardverzweigung angezeigt.

Um einen anderen Branch anzuzeigen, wählen Sie die Schaltfläche der Ansichtsauswahl. Wählen Sie entweder in der Dropdown-Liste einen Branch-Namen aus oder geben Sie diesen im Filterfeld ein und wählen Sie ihn dann aus der Liste aus.

4. Wählen Sie Add file (Datei hinzufügen) und dann eine der folgenden Optionen:
 - Um den Code-Editor zum Erstellen des Inhalts einer Datei und zum Hinzufügen der Datei zum Repository zu verwenden, wählen Sie Create file (Datei erstellen).
 - Um eine Datei vom lokalen Computer in das Repository hochzuladen, wählen Sie Upload file (Datei hochladen).
5. Stellen Sie den anderen Benutzern Informationen darüber bereit, wer diese Datei zum Repository hinzugefügt hat und warum.
 - Geben Sie im Feld Author name (Name des Autors) Ihren Namen ein. Dieser Name wird sowohl als Autorennamen als auch als Name des Committers in den Commit-Informationen verwendet. CodeCommit verwendet standardmäßig Ihren IAM-Benutzernamen oder eine Ableitung Ihres Konsolen-Logins als Autorennamen.
 - Geben Sie im Feld E-Mail-Adresse eine E-Mail-Adresse ein, damit andere Repository-Benutzer Sie bezüglich dieser Änderung kontaktieren können.
 - Geben Sie in das Feld Commit message (Commit-Nachricht) eine kurze Beschreibung ein. Dies ist zwar optional, wird aber ausdrücklich empfohlen. Andernfalls wird eine Standard-Commit-Nachricht verwendet.
6. Führen Sie eine der folgenden Aktionen aus:
 - Wenn Sie eine Datei hochladen, wählen Sie sie auf dem lokalen Computer aus.
 - Wenn Sie eine Datei erstellen, geben Sie den gewünschten Inhalt im Code-Editor ein und geben Sie einen Namen für die Datei an.
7. Wählen Sie Commit changes (Änderungen übernehmen) aus.

Füge eine Datei hinzu (AWS CLI)

Sie können den Befehl AWS CLI und den put-file Befehl verwenden, um eine Datei zu einem CodeCommit Repository hinzuzufügen. Sie können den Befehl put-file auch nutzen, um ein Verzeichnis oder eine Pfadstruktur für die Datei hinzuzufügen.

Note

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter [Befehlszeilenreferenz](#).

So fügen Sie eine Datei zu einem Repository hinzu

1. Erstellen Sie auf dem lokalen Computer die Datei, die Sie dem CodeCommit-Repository hinzufügen möchten.
2. Führen Sie am Terminal oder in der Befehlszeile den Befehl put-file unter Angabe der folgenden Informationen aus:
 - Repository, dem Sie die Datei hinzufügen möchten
 - Branch, dem Sie die Datei hinzufügen möchten
 - Vollständige Commit-ID des letzten Commits an diesem Branch (auch bezeichnet als oberster oder HEAD-Commit)
 - Der lokale Speicherort der Datei Die Syntax für diesen Standort hängt von Ihrem lokalen Betriebssystem ab.
 - Der Name der Datei, die Sie hinzufügen möchten, einschließlich des Pfads, in sich dem die aktualisierte Datei im Repository befindet (falls vorhanden)
 - Der Benutzername und die E-Mail-Adresse, den/die Sie mit dieser Datei verknüpfen möchten
 - Eine Commit-Nachricht, die erklärt, warum Sie diese Datei hinzugefügt haben

Der Benutzername, die E-Mail-Adresse und die Commit-Nachricht sind optional. Teilen Sie aber anderen Benutzern mit, wer die Änderung warum vorgenommen hat. Wenn Sie keinen Benutzernamen angeben, wird CodeCommit standardmäßig Ihr IAM-Benutzername oder eine Ableitung Ihres Konsolen-Logins als Autorennamen verwendet.

Um beispielsweise eine Datei zu einem Repository hinzuzufügen, *ExampleSolution.py* das *MyDemoRepo* nach einem Branch benannt ist, *feature-randomizationfeature* dessen neuester Commit die ID lautet: *4c925148EXAMPLE*

```
aws codecommit put-file --repository-name MyDemoRepo --branch-name feature-  
randomizationfeature --file-content file://MyDirectory/ExampleSolution.py --file-  
path /solutions/ExampleSolution.py --parent-commit-id 4c925148EXAMPLE --name "María  
García" --email "maría_garcía@example.com" --commit-message "I added a third  
randomization routine."
```

Note

Wenn Sie Binärdateien hinzufügen, stellen Sie sicher, dass Sie für die Angabe des lokalen Speicherorts der Datei `fileb://` verwenden.

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

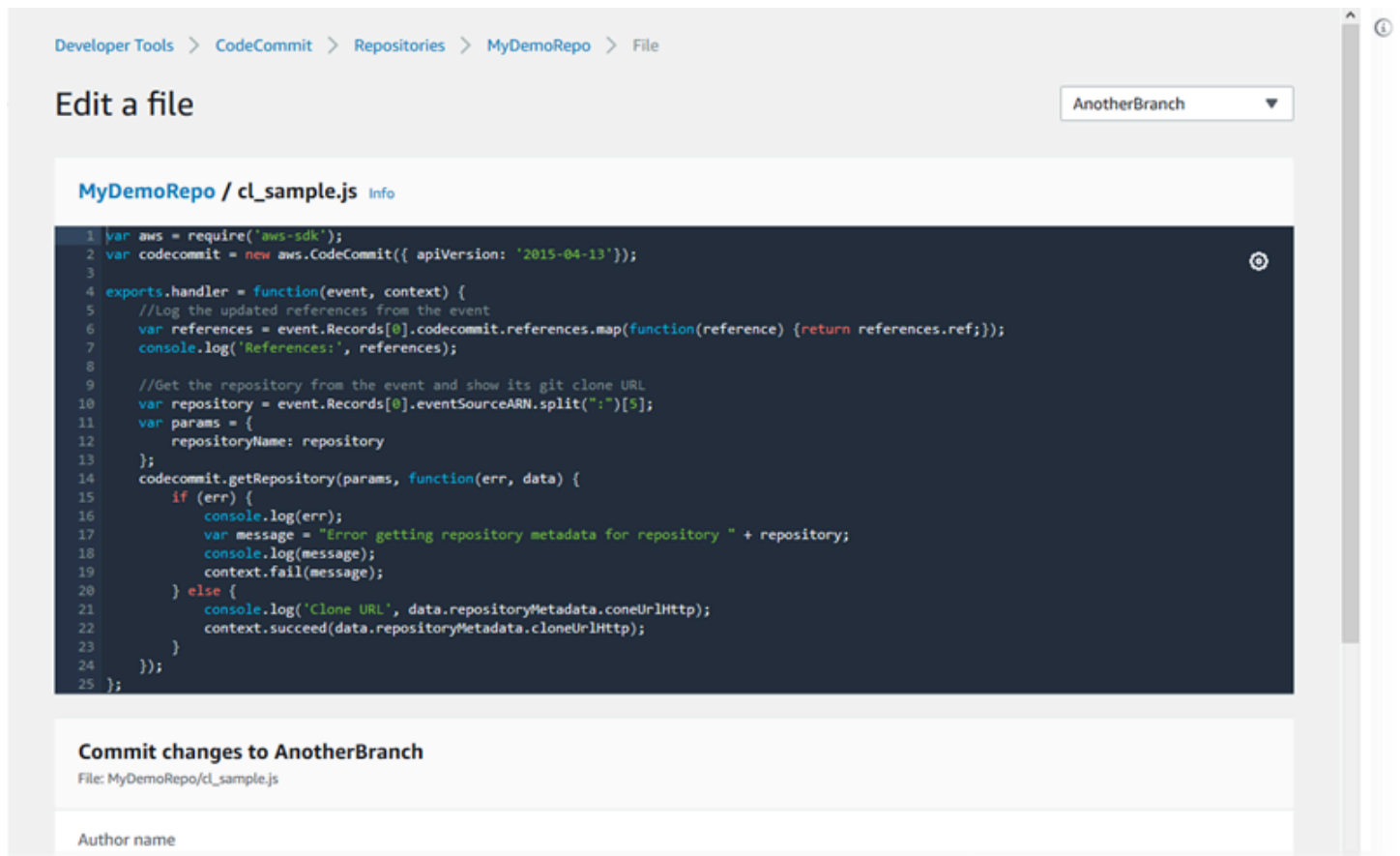
```
{  
  "blobId": "2eb4af3bEXAMPLE",  
  "commitId": "317f8570EXAMPLE",  
  "treeId": "347a3408EXAMPLE"  
}
```

Eine Datei hinzufügen (Git)

Sie können Dateien in einem lokalen Repo hinzufügen und Ihre Änderungen in ein CodeCommit Repository übertragen. Weitere Informationen finden Sie unter [Erste Schritte mit Git und AWS CodeCommit](#).

Den Inhalt einer Datei in einem AWS CodeCommit Repository bearbeiten

Sie können die CodeCommit Konsole oder einen Git-Client verwenden AWS CLI, um den Inhalt einer Datei in einem CodeCommit Repository zu bearbeiten.



Themen

- [Bearbeiten Sie eine Datei \(Konsole\)](#)
- [Bearbeiten oder löschen Sie eine Datei \(AWS CLI\)](#)
- [Eine Datei bearbeiten \(Git\)](#)

Bearbeiten Sie eine Datei (Konsole)

Sie können die CodeCommit Konsole verwenden, um eine Datei zu bearbeiten, die einem Zweig in einem CodeCommit Repository hinzugefügt wurde. Beim Bearbeiten der Datei können Sie den Benutzernamen und eine E-Mail-Adresse angeben. Sie können auch eine Commit-Nachricht hinzufügen, sodass andere Benutzer wissen, wer die Datei geändert hat und aus welchem Grund.

So bearbeiten Sie eine Datei in einem Repository

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.

2. Wählen Sie unter Repositories (Repositoryys) das Repository aus, in dem Sie eine Datei bearbeiten möchten.
3. Wählen Sie in der Codeansicht den Branch aus, in dem Sie die Datei bearbeiten möchten. Standardmäßig werden beim Öffnen der Codeansicht die Inhalte der Standardverzweigung angezeigt.

Um einen anderen Branch anzuzeigen, wählen Sie die Schaltfläche der Ansichtsauswahl. Wählen Sie entweder in der Dropdown-Liste einen Branch-Namen aus oder geben Sie diesen im Filterfeld ein und wählen Sie ihn dann aus der Liste aus.

4. Navigieren Sie durch den Inhalt des Branch und wählen Sie die Datei aus, die Sie bearbeiten möchten. Wählen Sie in der Dateiansicht die Option Edit (Bearbeiten).

Note

Wenn Sie eine Binärdatei auswählen, wird eine Warnmeldung eingeblendet, in der Sie zur Bestätigung aufgefordert werden, ob Sie den Inhalt anzeigen möchten. Sie sollten die CodeCommit Konsole nicht zum Bearbeiten von Binärdateien verwenden.

5. Bearbeiten Sie die Datei. Stellen Sie den anderen Benutzern Informationen darüber bereit, wer diese Änderung getätigt hat und warum.
 - Geben Sie im Feld Author name (Name des Autors) Ihren Namen ein. Dieser Name wird sowohl als Autorennamen als auch als Name des Committers in den Commit-Informationen verwendet. CodeCommit verwendet standardmäßig Ihren IAM-Benutzernamen oder eine Ableitung Ihres Konsolen-Logins als Autorennamen.
 - Geben Sie im Feld E-Mail-Adresse eine E-Mail-Adresse ein, damit andere Repository-Benutzer Sie bezüglich dieser Änderung kontaktieren können.
 - Geben Sie in das Feld Commit message (Commit-Nachricht) eine kurze Beschreibung Ihrer Änderungen ein.
6. Wählen Sie Commit changes (Änderungen übernehmen), um Ihre Änderungen an der Datei zu speichern und diese in das Repository zu übernehmen.

Bearbeiten oder löschen Sie eine Datei (AWS CLI)

Sie können den Befehl AWS CLI und den put-file Befehl verwenden, um Änderungen an einer Datei in einem CodeCommit Repository vorzunehmen. Sie können auch den Befehl put-file verwenden,

um ein Verzeichnis oder eine Pfadstruktur für die geänderte Datei hinzuzufügen, wenn Sie diese an einem anderen Speicherort als dem ursprünglichen Ort speichern möchten. Wenn Sie eine Datei vollständig löschen möchten, können Sie den Befehl `delete-file` verwenden.

Note

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter [Befehlszeilenreferenz](#).

So bearbeiten Sie eine Datei in einem Repository

1. Nehmen Sie unter Verwendung einer lokalen Kopie der Datei die Änderungen vor, die Sie dem CodeCommit-Repository hinzufügen möchten.
2. Führen Sie am Terminal oder in der Befehlszeile den Befehl `put-file` unter Angabe der folgenden Informationen aus:
 - Das Repository, dem Sie die bearbeitete Datei hinzufügen möchten
 - Der Branch, dem Sie die bearbeitete Datei hinzufügen möchten
 - Vollständige Commit-ID des letzten Commits an diesem Branch (auch bezeichnet als oberster oder HEAD-Commit)
 - Der lokale Speicherort der Datei
 - Der Name der aktualisierten Datei, die Sie hinzufügen möchten, einschließlich des Pfads, in dem sich die aktualisierte Datei im Repository befindet (falls vorhanden)
 - Der Benutzername und die E-Mail-Adresse, den/die Sie mit dieser Dateiänderung verknüpfen möchten
 - Eine Commit-Nachricht, die die Änderung erläutert

Der Benutzername, die E-Mail-Adresse und die Commit-Nachricht sind optional. Teilen Sie aber anderen Benutzern mit, wer die Änderung warum vorgenommen hat. Wenn Sie keinen Benutzernamen angeben, wird CodeCommit standardmäßig Ihr IAM-Benutzername oder eine Ableitung Ihres Konsolenanmeldenamens verwendet.

Um beispielsweise Änderungen an einer Datei mit dem Namen einem Repository hinzuzufügen, das *ExampleSolution.py MyDemoRepo* nach einem Branch benannt ist, *feature-randomizationfeature* dessen neuester Commit die ID lautet: *4c925148EXAMPLE*

```
aws codecommit put-file --repository-name MyDemoRepo --branch-name feature-randomizationfeature --file-content file://MyDirectory/ExampleSolution.py --file-path /solutions/ExampleSolution.py --parent-commit-id 4c925148EXAMPLE --name "María García" --email "maría_garcía@example.com" --commit-message "I fixed the bug Mary found."
```

 Note

Wenn Sie eine geänderte Binärdatei hinzufügen möchten, sollten Sie `--file-content` mit der Notation **fileb://MyDirectory/MyFile.raw** verwenden.

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
  "blobId": "2eb4af3bEXAMPLE",
  "commitId": "317f8570EXAMPLE",
  "treeId": "347a3408EXAMPLE"
}
```

Verwenden Sie zum Löschen einer Datei den Befehl `delete-file`. Um beispielsweise eine Datei zu löschen, die *README.md* in einem Branch benannt ist, der *main* mit der neuesten Commit-ID benannt ist, *c5709475EXAMPLE* in einem Repository mit dem *MyDemoRepo* Namen:

```
aws codecommit delete-file --repository-name MyDemoRepo --branch-name main --file-path README.md --parent-commit-id c5709475EXAMPLE
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
  "blobId": "559b44fEXAMPLE",
  "commitId": "353cf655EXAMPLE",
  "filePath": "README.md",
  "treeId": "6bc824cEXAMPLE"
}
```

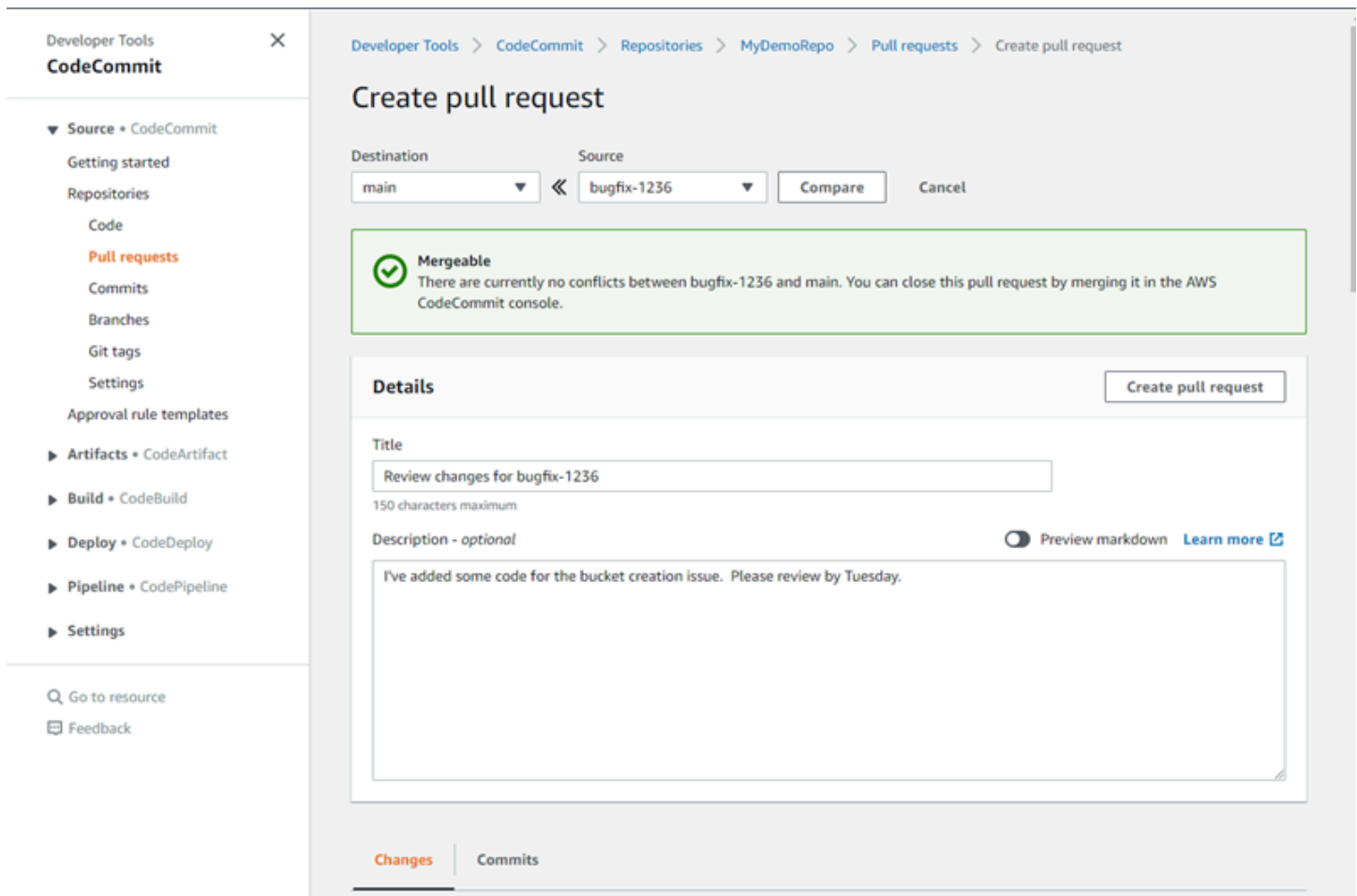
Eine Datei bearbeiten (Git)

Sie können Dateien in einem lokalen Repo bearbeiten und Ihre Änderungen in ein CodeCommit Repository übertragen. Weitere Informationen finden Sie unter [Erste Schritte mit Git und AWS CodeCommit](#).

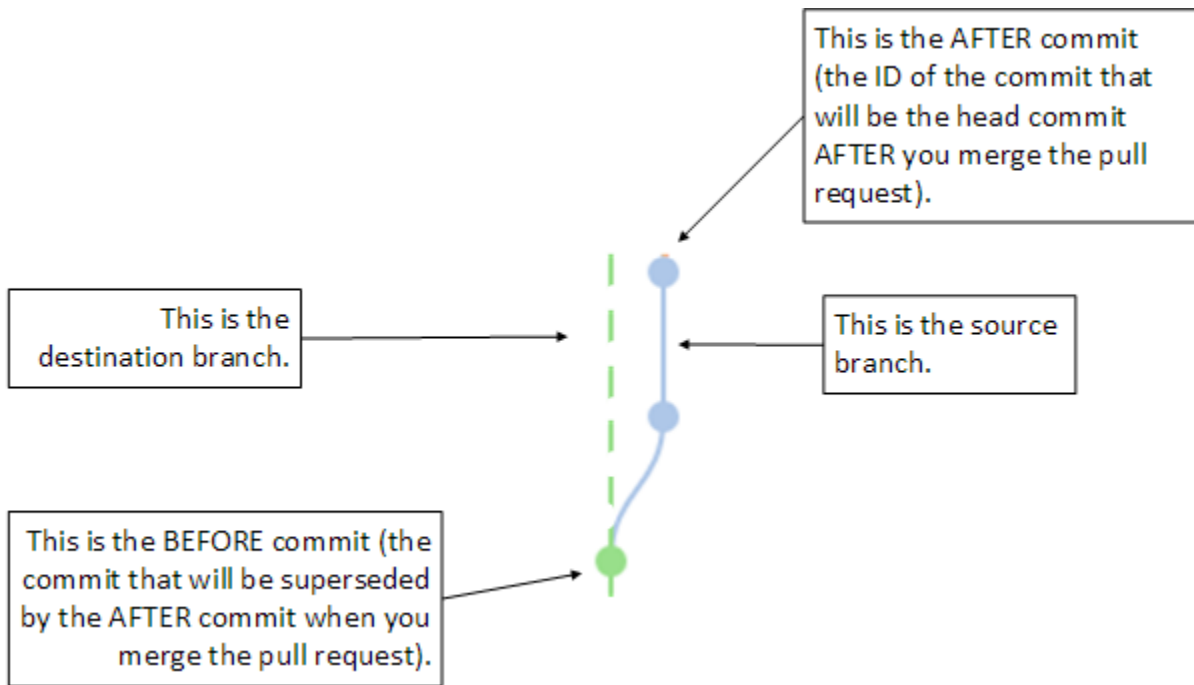
Mit Pull-Requests in AWS CodeCommit Repositorys arbeiten

Eine Pull-Anforderung ist die primäre Vorgehensweise, wie Sie und andere Repository-Benutzer Codeänderungen prüfen, kommentieren und über Branches hinweg zusammenführen können. Mit Pull-Anforderungen können Sie Codeänderungen auf kleine Änderungen oder Fehlerkorrekturen, größere Funktionserweiterungen oder neue Versionen Ihrer veröffentlichten Software gemeinsam prüfen. Hier finden Sie einen möglichen Workflow für eine Pull-Anforderung:

Li Juan, ein Entwickler, der in einem Repo mit dem Namen arbeitet MyDemoRepo, möchte an einer neuen Funktion für eine kommende Version eines Produkts arbeiten. Um ihre Arbeit vom produktionsreifen Code zu trennen, erstellt sie einen Zweig aus dem Standardzweig und benennt ihn. *feature-randomizationfeature* Sie schreibt Code, führt Commits durch und überträgt den neuen Funktionscode auf diesen Branch. Sie möchte die Codequalität durch andere Repository-Benutzer überprüfen lassen, bevor sie ihre Änderungen in den Standard-Branch einfügt. Zu diesem Zweck erstellt sie eine Pull-Anforderung. Die Pull-Anforderung enthält den Vergleich zwischen dem Branch mit ihrer Arbeit und dem Branch mit dem Code, mit dem sie ihre Änderungen zusammenführen möchte (in diesem Fall der Standard-Branch). Sie kann außerdem eine Genehmigungsregel erstellen, die voraussetzt, dass die Pull-Anforderung von einer bestimmten Anzahl von Benutzern genehmigt wird. Und sie kann einen Genehmigungs-Pool von Benutzern angeben. Andere Benutzer prüfen ihren Code sowie die Änderungen und fügen Kommentare und Vorschläge hinzu. Als Reaktion auf Kommentare aktualisiert sie den Branch mit ihrer Arbeit möglicherweise mehrmals mit Codeänderungen. Ihre Änderungen werden jedes Mal, wenn sie sie an diesen Branch-In überträgt, in den Pull-Request übernommen. CodeCommit Sie kann zudem Änderungen einbeziehen, die im vorgesehenen Ziel-Branch vorgenommen wurden, während die Pull-Anforderung offen ist, sodass Benutzer sicher sein können, dass sie alle Änderungsvorschläge im Kontext sehen. Wenn sie und die Prüfer zufrieden sind und die Bedingungen für Genehmigungsregeln (falls vorhanden) erfüllt sind, führt sie oder einer der Prüfer den Code zusammen und schließt die Pull-Anforderung.



Pull-Anforderungen erfordern zwei Branches: einen Quell-Branch mit dem Code, den Sie prüfen lassen möchten, und einen Ziel-Branch, mit dem der geprüfte Code zusammengeführt werden soll. Der Quell-Branch enthält den AFTER-Commit, bei dem es sich um den Commit mit den Änderungen handelt, die Sie mit dem Ziel-Branch zusammenführen möchten. Der Ziel-Branch enthält den BEFORE-Commit, der den Status des Codes repräsentiert, bevor der Pull-Anforderungs-Branch mit dem Ziel-Branch zusammengeführt wird. Die Wahl der Merge-Strategie wirkt sich auf die Details aus, wie Commits zwischen den Quell- und Ziel-Branches in der CodeCommit Konsole zusammengeführt werden. Weitere Informationen zu Merge-Strategien finden Sie CodeCommit unter [Eine Pull-Anfrage zusammenführen \(Konsole\)](#).



Die Pull-Anforderung zeigt die Unterschiede zwischen der Spitze des Source-Branch und dem neuesten Commit auf dem Ziel-Branch, auf dem die Pull-Anforderung erstellt wird, sodass die Benutzer Änderungen ansehen und kommentieren können. Sie können die Pull-Anforderung aufgrund der Kommentare mit weiteren Änderungen aktualisieren, indem Sie Commits für Änderungen durchführen und diese per Push auf den Quell-Branch übertragen.

Developer Tools
CodeCommit

Source • CodeCommit

- Getting started
- Repositories
- Code
- Pull requests**
- Commits
- Branches
- Tags
- Settings
- Build • CodeBuild
- Deploy • CodeDeploy
- Pipeline • CodePipeline

Mergeable [Learn more](#)

Details Activity **Changes** Commits

< Page 1 of 1 >

☒ Hide whitespace changes ☐ Unified ☒ Split

ahs_count.py

```

*** @ -5,6 +5,6 @@
5
6 total = (ess + z)
7 ahs = "Number of alveolar hissing sibilants: {}"
8 - print(ahs.format(total))
8 + print(alv.format(total))

```

New comment ☒ Preview markdown [Learn more](#)

You've switched back to the old variable, which won't work. This should be `ahs`.

Wenn Ihr Code geprüft wurde und die Genehmigungsregeln (falls vorhanden) erfüllt sind, können Sie die Pull-Anforderung auf verschiedene Arten schließen:

- Führen Sie die Branches lokal zusammen und übertragen Sie Ihre Änderungen per Push. Dadurch wird die Anforderung automatisch geschlossen, wenn die Fast-Forward-Merge-Strategie verwendet wird und es keine Zusammenführungskonflikte gibt.
- Verwenden Sie die AWS CodeCommit Konsole, um die Pull-Anfrage ohne Zusammenführung zu schließen, Konflikte bei einer Zusammenführung zu lösen oder, falls keine Konflikte vorliegen, die Branches mit einer der verfügbaren Merge-Strategien zu schließen und zusammenzuführen.
- Verwenden Sie die AWS CLI.

Bevor Sie eine Pull-Anforderung erstellen, müssen Sie folgende Schritte ausführen:

- Stellen Sie sicher, dass Sie die zu prüfenden Codeänderungen bestätigt und per Push an einen Branch (den Quell-Branch) übergeben haben.
- Richten Sie Benachrichtigungen für Ihr Repository ein, sodass die Benutzer über die Pull-Anforderung und entsprechende Änderungen informiert werden. (Dieser Schritt ist zwar optional, wird aber empfohlen.)
- Erstellen Sie Genehmigungsregelvorlagen mit dem Repository und ordnen Sie sie zu, damit automatisch Genehmigungsregeln für Pull-Anforderungen erstellt werden und die Code-Qualität sichergestellt bleibt. Weitere Informationen finden Sie unter [Arbeiten mit Genehmigungsregelvorlagen](#).

Pull-Requests sind effektiver, wenn Sie IAM-Benutzer für Ihre Repository-Benutzer in Ihrem Amazon Web Services Services-Konto eingerichtet haben. Das erleichtert das Identifizieren des Benutzers, der einen Kommentar abgegeben hat. Der andere Vorteil besteht darin, dass IAM-Benutzer Git-Anmeldeinformationen für den Repository-Zugriff verwenden können. Weitere Informationen finden Sie unter [Schritt 1: Erstkonfiguration für CodeCommit](#). Sie können Pull-Anforderungen mit anderen Arten von Benutzern verwenden, einschließlich Benutzern mit verbundenem Zugriff.

Informationen zur Arbeit mit anderen Aspekten Ihres Repositorys finden Sie unter [CodeCommitArbeiten mit Repositorien](#), [Arbeiten mit Genehmigungsregelvorlagen](#), [Mit Dateien arbeiten](#), [Mit Commits arbeitenArbeiten mit Zweigen](#), und [Mit Benutzereinstellungen arbeiten](#).

Themen

- [Erstellen einer Pull-Anforderung](#)
- [Erstellen Sie eine Genehmigungsregel für eine Pull-Anfrage](#)
- [Pull-Requests in einem AWS CodeCommit Repository anzeigen](#)

- [Überprüfen einer Pull-Anforderung](#)
- [Aktualisieren einer Pull-Anforderung](#)
- [Genehmigungsregel für einen Pull Request bearbeiten oder löschen](#)
- [Genehmigungsregeln für eine Pull-Anfrage überschreiben](#)
- [Einen Pull-Request in einem AWS CodeCommit Repository zusammenführen](#)
- [Konflikte in einer Pull-Anfrage in einem AWS CodeCommit Repository lösen](#)
- [Einen Pull-Request in einem AWS CodeCommit Repository schließen](#)

Erstellen einer Pull-Anforderung

Durch das Erstellen von Pull-Anforderungen können andere Benutzer Ihre Codeänderungen sehen und prüfen, bevor Sie diese mit einem anderen Branch zusammenführen. Zunächst erstellen Sie einen Branch für Ihre Codeänderungen. Dieser wird als Quell-Branch einer Pull-Anforderung bezeichnet. Nachdem Sie einen Commit für diese Änderungen durchgeführt und sie per Push auf das Repository übertragen haben, können Sie eine Pull-Anforderung erstellen, die den Inhalt dieses Branch (Quell-Branch) mit dem Branch vergleicht, mit dem Sie Ihre Änderungen nach dem Schließen der Pull-Anforderung zusammenführen möchten (Ziel-Branch).

Sie können die AWS CodeCommit Konsole oder die verwenden AWS CLI , um Pull-Requests für Ihr Repository zu erstellen.

Themen

- [Erstellen Sie eine Pull-Anfrage \(Konsole\)](#)
- [Erstellen Sie eine Pull-Anfrage \(\)AWS CLI](#)

Erstellen Sie eine Pull-Anfrage (Konsole)

Sie können die CodeCommit Konsole verwenden, um eine Pull-Anfrage in einem CodeCommit Repository zu erstellen. Wenn Ihr Repository [mit Benachrichtigungen konfiguriert](#) ist, erhalten Benutzer mit Abonnement eine E-Mail, wenn Sie eine Pull-Anforderung erstellen.

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, in dem Sie eine Pull-Anforderung erstellen möchten.

3. Wählen Sie im Navigationsbereich Pull Requests aus.



Tip

Darüber hinaus können Sie Pull-Anforderungen über Branches und Code erstellen.

4. Wählen Sie Create pull request aus.

</

- Wählen Sie in Create pull request unter Source den Branch aus, der die zu prüfenden Änderungen enthält.
- Wählen Sie unter Destination (Ziel) den Branch aus, mit dem Sie Ihre Codeänderungen nach dem Schließen der Pull-Anforderung zusammenführen möchten.
- Wählen Sie Compare aus. Die beiden Branches werden verglichen und die Unterschiede zwischen ihnen angezeigt. Zudem wird eine Analyse durchgeführt, um zu ermitteln, ob die beiden Branches nach dem Schließen der Pull-Anforderung automatisch zusammengeführt werden können.
- Prüfen Sie die Details des Vergleichs und die Änderungen, um sicherzustellen, dass die Pull-Anforderung die Änderungen und Commits enthält, die geprüft werden sollen. Andernfalls passen Sie Ihre Auswahl für Quell- und Ziel-Branch an und wählen erneut Compare aus.
- Wenn Sie mit den Vergleichsergebnissen für die Pull-Anforderung zufrieden sind, geben Sie unter Title (Titel) einen kurzen, aussagekräftigen Namen für diese Prüfung an. Dieser Titel wird in der Liste der Pull-Anforderungen für das Repository angezeigt.
- (Optional) Unter Description (Beschreibung) können Sie angeben, wofür diese Prüfung vorgesehen ist, sowie weitere nützliche Informationen für Prüfer hinterlassen.
- Wählen Sie Create (Erstellen) aus.

Developer Tools > CodeCommit > Repositories > MyDemoRepo > Pull requests > Create pull request

Create pull request

Destination: main Source: bugfix-1236 Compare Cancel

Mergeable
There are currently no conflicts between bugfix-1236 and main. You can close this pull request by merging it in the AWS CodeCommit console.

Details Create pull request

Title: Review changes for bugfix-1236
150 characters maximum

Description - optional: I've added some code for the bucket creation issue. Please review by Tuesday.
Preview markdown Learn more

Changes Commits

Ihre Pull-Anforderung wird in der Liste der Pull-Anforderungen für das Repository angezeigt. Wenn Sie [Benachrichtigungen konfiguriert](#) haben, erhalten Abonnenten des Amazon SNS SNS-Themas eine E-Mail, um sie über die neu erstellte Pull-Anfrage zu informieren.

Erstellen Sie eine Pull-Anfrage ()AWS CLI

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter [Befehlszeilenreferenz](#).

Um den zu verwenden AWS CLI , um einen Pull-Request in einem CodeCommit Repository zu erstellen

1. Führen Sie den Befehl create-pull-request aus und geben Sie Folgendes an:
 - Name der Pull-Anforderung (mit der Option --title).
 - Beschreibung der Pull-Anforderung (mit der Option --description).
 - Liste der Ziele für den Befehl create-pull-request, einschließlich:

- Der Name des CodeCommit Repositorys, in dem der Pull-Request erstellt wird (mit dem `repositoryName` Attribut).
- Name des Branches, der die zu prüfenden Codeänderungen enthält, auch Quell-Branch genannt (mit dem Attribut `sourceReference`).
- (Optional) Name des Branches, mit dem Sie Ihre Codeänderungen zusammenführen möchten (auch Ziel-Branch genannt), falls Sie sie nicht mit dem Standard-Branch zusammenführen möchten (mit dem Attribut `destinationReference`).
- Einzigartiger, vom Client generierter Idempotenz-Token (mit der Option `--client-request-token`).

In diesem Beispiel wird eine Pull-Anfrage *Pronunciation difficulty analyzer* mit dem Namen und einer Beschreibung erstellt *Please review these changes by Tuesday*, die auf den *jane-branch* Quell-Branch abzielt. Der Pull-Request soll mit dem Standard-Branch *main* in einem CodeCommit Repository mit dem Namen *MyDemoRepo* zusammengeführt werden.

```
aws codecommit create-pull-request --title "Pronunciation difficulty analyzer"
--description "Please review these changes by Tuesday" --client-request-token
123Example --targets repositoryName=MyDemoRepo,sourceReference=jane-branch
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
  "pullRequest": {
    "approvalRules": [
      {
        "approvalRuleContent": "{\"Version\": \"2018-11-08\",
        \"DestinationReferences\": [\"refs/heads/main\"], \"Statements\": [{\"Type
        \": \"Approvers\", \"NumberOfApprovalsNeeded\": 2, \"ApprovalPoolMembers\":
        [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
        "approvalRuleId": "dd8b17fe-EXAMPLE",
        "approvalRuleName": "2-approver-rule-for-main",
        "creationDate": 1571356106.936,
        "lastModifiedDate": 1571356106.936,
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
        "originApprovalRuleTemplate": {
          "approvalRuleTemplateId": "dd3d22fe-EXAMPLE",
          "approvalRuleTemplateName": "2-approver-rule-for-main"
        },
        "ruleContentSha256": "4711b576EXAMPLE"
      }
    ]
  }
}
```

```
    }
  ],
  "authorArn": "arn:aws:iam::111111111111:user/Jane_Doe",
  "description": "Please review these changes by Tuesday",
  "title": "Pronunciation difficulty analyzer",
  "pullRequestTargets": [
    {
      "destinationCommit": "5d036259EXAMPLE",
      "destinationReference": "refs/heads/main",
      "repositoryName": "MyDemoRepo",
      "sourceCommit": "317f8570EXAMPLE",
      "sourceReference": "refs/heads/jane-branch",
      "mergeMetadata": {
        "isMerged": false
      }
    }
  ],
  "lastActivityDate": 1508962823.285,
  "pullRequestId": "42",
  "clientRequestToken": "123Example",
  "pullRequestStatus": "OPEN",
  "creationDate": 1508962823.285
}
```

Erstellen Sie eine Genehmigungsregel für eine Pull-Anfrage

Durch das Erstellen von Genehmigungsregeln für Pull-Anforderungen wird die Qualität des Codes sichergestellt, weil Benutzer die Pull-Anforderung genehmigen müssen, bevor der Code im Ziel-Branche zusammengeführt werden kann. Sie können die Anzahl der Benutzer angeben, die eine Pull-Anforderung genehmigen müssen. Sie können auch einen Genehmigungs-Pool von Benutzern für die Regel angeben. In diesem Fall werden nur Genehmigungen dieser Benutzer auf die Anzahl der erforderlichen Genehmigungen für die Regel angerechnet.

Note

Sie können auch Vorlagen für Genehmigungsregeln erstellen, mit deren Hilfe Sie die repositoryübergreifende Erstellung von Genehmigungsregeln automatisieren können,

die für jeden Pull Request gelten. Weitere Informationen finden Sie unter [Arbeiten mit Genehmigungsregelvorlagen](#).

Sie können die AWS CodeCommit Konsole oder die verwenden AWS CLI , um Genehmigungsregeln für Ihr Repository zu erstellen.

Themen

- [Erstellen Sie eine Genehmigungsregel für eine Pull-Anfrage \(Konsole\)](#)
- [Erstellen Sie eine Genehmigungsregel für eine Pull-Anfrage \(\)AWS CLI](#)

Erstellen Sie eine Genehmigungsregel für eine Pull-Anfrage (Konsole)

Sie können die CodeCommit Konsole verwenden, um eine Genehmigungsregel für eine Pull-Anfrage in einem CodeCommit Repository zu erstellen.

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, in dem eine Genehmigungsregel für eine Pull-Anforderung erstellt werden soll.
3. Wählen Sie im Navigationsbereich Pull Requests aus.
4. Wählen Sie in der Liste die Pull-Anforderung aus, für die Sie eine Genehmigungsregel erstellen möchten. Sie können Genehmigungsregeln nur für offene Pull-Anforderungen erstellen.

Pull request	Author	Destination	Last activity	Status	Approval status
31: testing this	Saanvi_Sarkar	preprod	4 minutes ago	Open	No approval rules
25: Updated some of our Java samples	Li_Juan	main	5 minutes ago	Open	0 of 1 rules satisfied
29: Changing duplicate value	Li_Juan	main	19 minutes ago	Open	0 of 1 rules satisfied
22: Test pull request	Saanvi_Sarkar	preprod	20 minutes ago	Open	No approval rules
28: Changes to some of our code samples	Li_Juan	main	1 month ago	Open	0 of 1 rules satisfied
20: A bugfix to add missing examples for S3	Saanvi_Sarkar	main	1 month ago	Open	0 of 1 rules satisfied

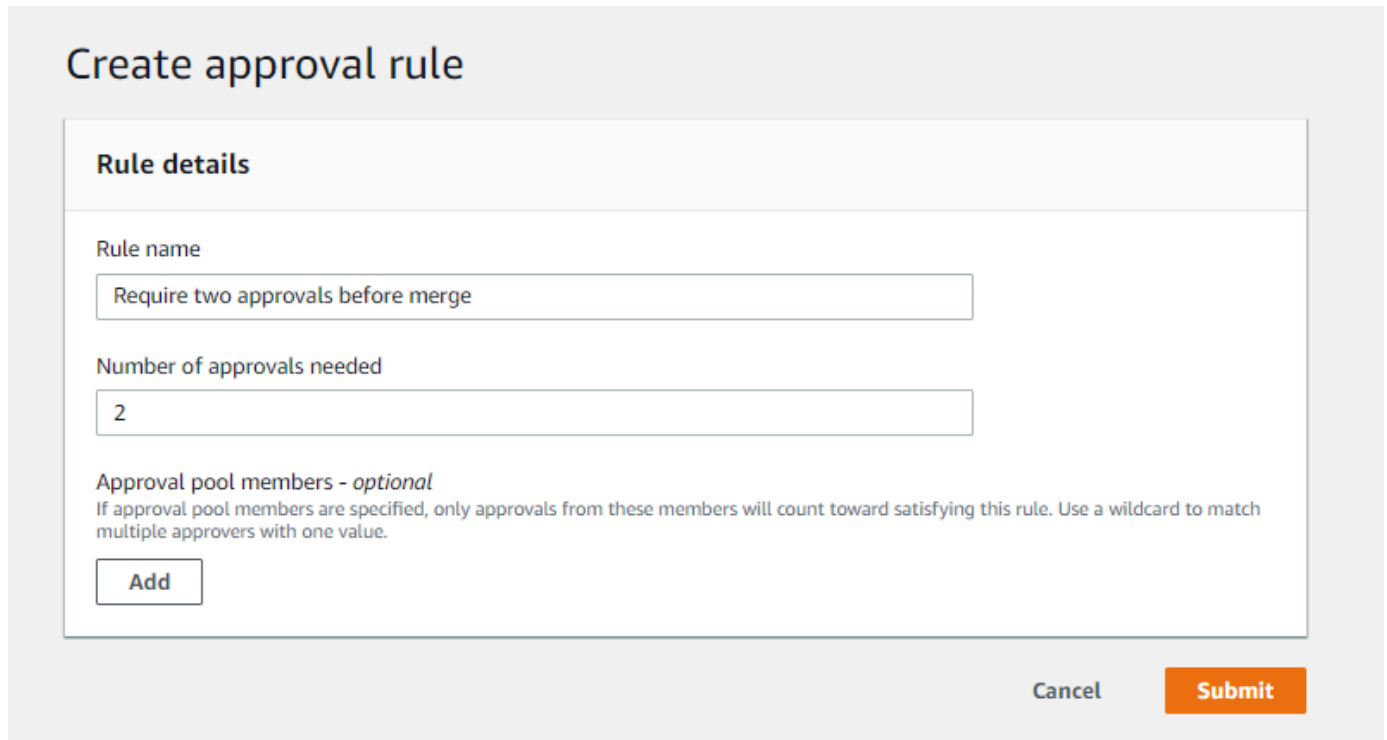
5. Wählen Sie in der Pull-Anforderung Approvals (Genehmigungen) und dann Create approval rule (Genehmigungsregel erstellen) aus.
6. Geben Sie in das Feld Rule name (Regelname) einen aussagekräftigen Namen ein. Wenn Sie z. B. möchten, dass eine Pull-Anforderung von zwei Personen genehmigt werden muss,

bevor sie zusammengeführt werden kann, können Sie die Regel beispielsweise **Require two approvals before merge** nennen.

 Note

Sie können den Namen einer Genehmigungsregel nicht mehr ändern, nachdem sie erstellt wurde.

Geben Sie in das Feld Number of approvals needed (Anzahl erforderlicher Genehmigungen) die gewünschte Anzahl ein. Der Standardwert ist 1.



Create approval rule

Rule details

Rule name

Require two approvals before merge

Number of approvals needed

2

Approval pool members - optional
If approval pool members are specified, only approvals from these members will count toward satisfying this rule. Use a wildcard to match multiple approvers with one value.

Add

Cancel Submit

7. (Optional) Wenn Sie voraussetzen möchten, dass die Genehmigungen für eine Pull-Anforderung von einer bestimmten Gruppe von Benutzern stammen, wählen Sie unter Approval rule members (Genehmigungsregelmitglieder) die Option Add (Hinzufügen) aus. Wählen Sie unter Approver type (Genehmigertyp) eine der folgenden Optionen aus:

- IAM-Benutzername oder angenommene Rolle: Bei dieser Option wird die AWS Konto-ID automatisch mit dem Konto aufgefüllt, mit dem Sie sich angemeldet haben, und es wird lediglich ein Name benötigt. Sie kann sowohl für IAM-Benutzer als auch für Benutzer mit Verbundzugriff verwendet werden, deren Name mit dem angegebenen Namen übereinstimmt. Dies ist eine sehr leistungsfähige Option, die hohe Flexibilität bietet. Wenn Sie beispielsweise

mit dem Amazon Web Services Services-Konto 123456789012 angemeldet sind und diese Option wählen und Sie angeben **Mary_Major**, dass alle folgenden Informationen als Genehmigungen von diesem Benutzer gezählt werden:

- Ein IAM-Benutzer im Konto () `arn:aws:iam::123456789012:user/Mary_Major`
- Ein Verbundbenutzer, der in IAM als Mary_Major () identifiziert wurde `arn:aws:sts::123456789012:federated-user/Mary_Major`

Diese Option erkennt keine aktive Sitzung eines Benutzers, der die Rolle **CodeCommitReview** angenommen hat und den Rollensitzungsnamen Mary_Major (`arn:aws:sts::123456789012:assumed-role/CodeCommitReview/Mary_Major`) verwendet, sofern Sie kein Platzhalterzeichen angegeben (*Mary_Major). Sie können den Rollennamen auch explizit angeben (`CodeCommitReview/Mary_Major`).

- Vollqualifizierter ARN: Mit dieser Option können Sie den vollqualifizierten Amazon-Ressourcennamen (ARN) des IAM-Benutzers oder der IAM-Rolle angeben. Diese Option unterstützt auch angenommene Rollen, die von anderen AWS Diensten verwendet werden, wie z. B. AWS Lambda und AWS CodeBuild. Bei angenommenen Rollen sollte das ARN-Format bei Rollen „`arn:aws:sts::AccountID:assumed-role/RoleName“` und bei Funktionen „`arn:aws:sts::AccountID:assumed-role/FunctionName“` sein.

Wenn Sie als Genehmigertyp den IAM-Benutzernamen oder die angenommene Rolle ausgewählt haben, geben Sie im Feld Wert den Namen des IAM-Benutzers oder der IAM-Rolle oder den vollqualifizierten ARN des Benutzers oder der Rolle ein. Wählen Sie erneut Add (Hinzufügen) aus, um weitere Benutzer oder Rollen hinzuzufügen, bis Sie alle Benutzer oder Rollen hinzugefügt haben, deren Genehmigungen auf die Anzahl der erforderlichen Genehmigungen angerechnet werden sollen.

Für beide Genehmigertypen können Platzhalterzeichen (*) in den Werten verwendet werden. Wenn Sie beispielsweise die Option „IAM-Benutzername“ oder „Angenommene Rolle“ wählen und angeben, dass alle Benutzer **CodeCommitReview/***, die die Rolle von annehmen, im **CodeCommitReview** Genehmigungspool gezählt werden. Die individuellen Rollensitzungsnamen zählen zum Erreichen der erforderlichen Anzahl von Genehmigern. Auf diese Weise zählen sowohl Mary_Major als auch Li_Juan als Genehmigungen, wenn sie angemeldet sind und die Rolle CodeCommitReview annehmen. [Weitere Informationen zu IAM ARNs, Platzhaltern und Formaten finden Sie unter IAM-Identifikatoren.](#)

 Note

Genehmigungsregeln unterstützen keine kontoübergreifenden Genehmigungen.

8. Wählen Sie nach dem Konfigurieren der Genehmigungsregel Submit (Senden).

Erstellen Sie eine Genehmigungsregel für eine Pull-Anfrage ()AWS CLI

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter [Befehlszeilenreferenz](#).

Um eine Genehmigungsregel für eine Pull-Anfrage in einem CodeCommit Repository zu erstellen

1. Führen Sie den Befehl `create-pull-request-approval-rule` aus und geben Sie Folgendes an:
 - ID der Pull-Anforderung (mit der Option `--id`).
 - Name der Genehmigungsregel (mit der Option `--approval-rule-name`).
 - Inhalt der Genehmigungsregel (mit der Option `--approval-rule-content`).

Wenn Sie die Genehmigungsregel erstellen, können Sie Genehmiger in einem Genehmigungs-Pool auf zwei Arten angeben:

- **CodeCommitApprovers:** Für diese Option sind nur ein Amazon Web Services Services-Konto und eine Ressource erforderlich. Sie kann sowohl für IAM-Benutzer als auch für Benutzer mit Verbundzugriff verwendet werden, deren Name mit dem angegebenen Ressourcennamen übereinstimmt. Dies ist eine sehr leistungsfähige Option, die hohe Flexibilität bietet. Wenn Sie beispielsweise das Amazon Web Services Services-Konto 123456789012 und angeben **Mary_Major**, werden alle folgenden Angaben als Genehmigungen gezählt, die von diesem Benutzer kommen:
 - Ein IAM-Benutzer im Konto () `arn:aws:iam::123456789012:user/Mary_Major`
 - Ein Verbundbenutzer, der in IAM als `Mary_Major` () identifiziert wurde `arn:aws:sts::123456789012:federated-user/Mary_Major`

Diese Option erkennt keine aktive Sitzung eines Benutzers, der die Rolle **CodeCommitReview** angenommen hat und den Rollensitzungsnamen `Mary_Major`

(arn:aws:sts::123456789012:assumed-role/CodeCommitReview/Mary_Major) verwendet, sofern Sie kein Platzhalterzeichen angegeben (*Mary_Major).

- Vollqualifizierter ARN: Mit dieser Option können Sie den vollqualifizierten Amazon-Ressourcennamen (ARN) des IAM-Benutzers oder der IAM-Rolle angeben.

[Weitere Informationen zu IAM ARNs, Platzhaltern und Formaten finden Sie unter IAM-Identifikatoren.](#)

Im folgenden Beispiel wird eine Genehmigungsregel namens `Require two approved approvers` für eine Pull-Anforderung mit der ID 27 erstellt. Die Regel gibt an, dass zwei Genehmigungen aus einem Genehmigungs-Pool erforderlich sind. Der Pool umfasst alle Benutzer, die auf das 123456789012 Amazon Web Services Services-Konto zugreifen CodeCommit und dessen Rolle übernehmen. **CodeCommitReview** Es umfasst auch entweder einen IAM-Benutzer oder einen Verbundbenutzer mit `Nikhil_Jayashankar` demselben Amazon Web Services Services-Konto:

```
aws codecommit create-pull-request-approval-rule --pull-request-id 27
--approval-rule-name "Require two approved approvers" --approval-
rule-content "{\"Version\": \"2018-11-08\", \"Statements\": [{\"Type\":
  \"Approvers\", \"NumberOfApprovalsNeeded\": 2, \"ApprovalPoolMembers
\": [\"CodeCommitApprovers:123456789012:Nikhil_Jayashankar\",
  \"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}"
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
  "approvalRule": {
    "approvalRuleName": "Require two approved approvers",
    "lastModifiedDate": 1570752871.932,
    "ruleContentSha256": "7c44e6ebEXAMPLE",
    "creationDate": 1570752871.932,
    "approvalRuleId": "aac33506-EXAMPLE",
    "approvalRuleContent": "{\"Version\": \"2018-11-08\", \"Statements\":
  [{\"Type\": \"Approvers\", \"NumberOfApprovalsNeeded\": 2, \"ApprovalPoolMembers
\": [\"CodeCommitApprovers:123456789012:Nikhil_Jayashankar\",
  \"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
    "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major"
  }
}
```

Pull-Requests in einem AWS CodeCommit Repository anzeigen

Sie können die AWS CodeCommit Konsole oder die verwenden AWS CLI , um Pull-Requests für Ihr Repository anzuzeigen. Standardmäßig sehen Sie nur offene Pull-Anforderungen. Sie können jedoch den Filter ändern und alle Pull-Anforderungen, nur geschlossene Anforderungen, nur von Ihnen erstellte Pull-Anforderungen usw. anzeigen.

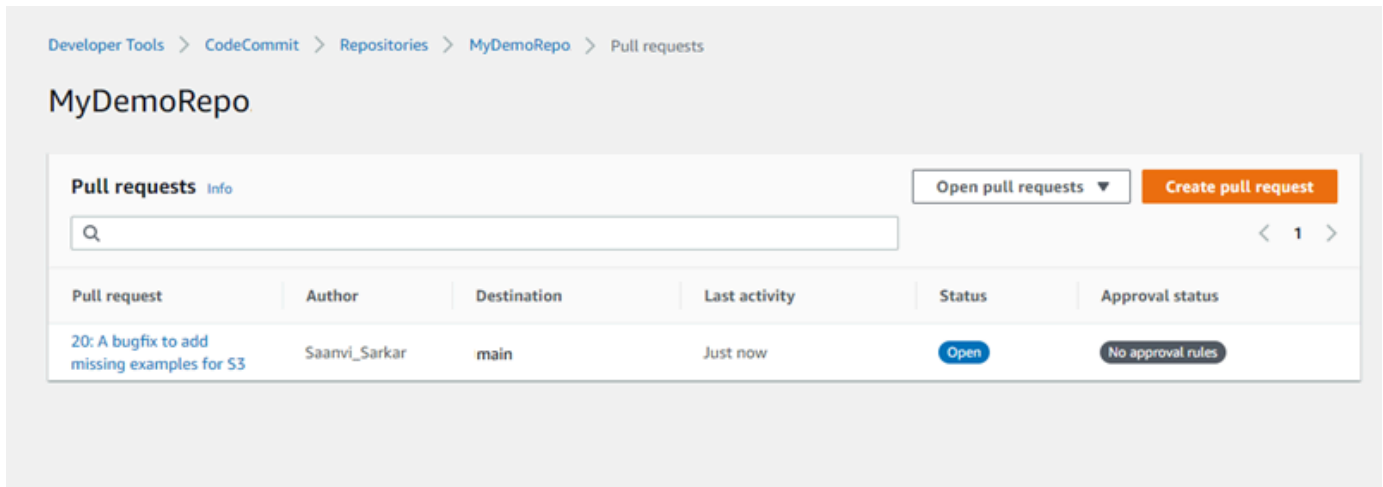
Themen

- [Pull-Requests anzeigen \(Konsole\)](#)
- [Pull-Requests anzeigen \(AWS CLI\)](#)

Pull-Requests anzeigen (Konsole)

Sie können die AWS CodeCommit Konsole verwenden, um eine Liste von Pull-Requests in einem CodeCommit Repository anzuzeigen. Durch Änderung des Filters können Sie die Listenanzeige so ändern, dass nur eine bestimmte Gruppe von Pull-Anforderungen angezeigt wird. Beispielsweise können Sie eine Liste mit Pull-Anforderungen anzeigen, die Sie erstellt haben und deren Status Open ist, oder Sie können einen anderen Filter wählen und von Ihnen erstellte Pull-Anforderungen mit dem Status Closed ansehen.

1. Öffne die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, in dem Sie Pull-Anforderungen anzeigen möchten.
3. Wählen Sie im Navigationsbereich Pull Requests aus.
4. Standardmäßig wird eine Liste aller offenen Pull-Anforderungen angezeigt.



5. Um den Anzeigefilter zu ändern, wählen Sie aus der Liste der verfügbaren Filter aus:
 - Open pull requests (Offene Pull-Anforderungen) (Standard): Zeigt alle Pull-Anforderungen mit dem Status Open (Offen) an.
 - All pull requests (Alle Pull-Anforderungen): Zeigt alle Pull-Anforderungen an.
 - All closed requests (Alle geschlossenen Pull-Anforderungen): Zeigt alle Pull-Anforderungen mit dem Status Closed (Geschlossen) an.
 - My pull requests (Meine Pull-Anforderungen): Zeigt alle von Ihnen erstellten Pull-Anforderungen unabhängig vom Status an. Es werden keine Überprüfungen angezeigt, die Sie kommentiert haben, oder an denen Sie anderweitig beteiligt waren.
 - My open pull requests (Meine offenen Pull-Anforderungen): Zeigt alle von Ihnen erstellten Pull-Anforderungen mit dem Status Open (Offen) an.
 - My closed pull requests (Meine geschlossenen Pull-Anforderungen): Zeigt alle von Ihnen erstellten Pull-Anforderungen mit dem Status Closed (Geschlossen) an.
6. Wenn Sie eine Pull-Anforderung in der angezeigten Liste finden, die Sie anzeigen möchten, wählen Sie diese aus.

Pull-Requests anzeigen (AWS CLI)

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter [Befehlszeilenreferenz](#).

Gehen Sie wie folgt vor, AWS CLI um Pull-Requests in einem CodeCommit Repository anzuzeigen.

1. Um eine Liste mit Pull-Anforderungen in einem Repository anzuzeigen, führen Sie den Befehl `list-pull-requests` aus. Geben Sie dabei Folgendes an:
 - Der Name des CodeCommit Repositorys, in dem Sie Pull-Requests anzeigen möchten (mit der `--repository-name` Option).
 - (Optional) Den Status der Pull-Anforderung (mit der Option `--pull-request-status`).
 - (Optional) Der Amazon-Ressourcenname (ARN) des IAM-Benutzers, der die Pull-Anfrage erstellt hat (mit der `--author-arn` Option).
 - (Optional) Ein Aufzählungs-Token, das verwendet werden kann, um Ergebnisstapel zurückzugeben (mit der Option `--next-token`).
 - (Optional) Eine Einschränkung in Bezug auf die Anzahl der zurückgegebenen Ergebnisse pro Anforderung (mit der Option `--max-results`).

Um beispielsweise Pull-Requests aufzulisten, die von einem IAM-Benutzer mit dem ARN `arn:aws:iam::111111111111:user/Li_Juan` und dem Status von erstellt wurden, `CLOSED` in einem CodeCommit Repository mit dem Namen `MyDemoRepo`:

```
aws codecommit list-pull-requests --author-arn arn:aws:iam::111111111111:user/Li_Juan --pull-request-status CLOSED --repository-name MyDemoRepo
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
  "nextToken": "",
  "pullRequestIds": ["2", "12", "16", "22", "23", "35", "30", "39", "47"]
}
```

Pull-Requests IDs werden in der Reihenfolge der letzten Aktivität angezeigt.

2. Um Details zu einer bestimmten Pull-Anforderung anzuzeigen, führen Sie den Befehl `get-pull-request` mit der Option `--pull-request-id` aus, wobei Sie die ID der Pull-Anforderung angeben. Zum Beispiel, um Informationen über eine Pull-Anfrage mit der folgenden ID anzuzeigen`27`:

```
aws codecommit get-pull-request --pull-request-id 27
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
  "pullRequest": {
    "approvalRules": [
      {
        "approvalRuleContent": "{\"Version\": \"2018-11-08\", \"Statements\": [{\"Type\": \"Approvers\", \"NumberOfApprovalsNeeded\": 2, \"ApprovalPoolMembers\": [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
        "approvalRuleId": "dd8b17fe-EXAMPLE",
        "approvalRuleName": "2-approver-rule-for-main",
        "creationDate": 1571356106.936,
        "lastModifiedDate": 571356106.936,
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
        "ruleContentSha256": "4711b576EXAMPLE"
      }
    ],
    "lastActivityDate": 1562619583.565,
    "pullRequestTargets": [
      {
        "sourceCommit": "ca45e279EXAMPLE",
        "sourceReference": "refs/heads/bugfix-1234",
        "mergeBase": "a99f5ddbEXAMPLE",
        "destinationReference": "refs/heads/main",
        "mergeMetadata": {
          "isMerged": false
        },
        "destinationCommit": "2abfc6beEXAMPLE",
        "repositoryName": "MyDemoRepo"
      }
    ],
    "revisionId": "e47def21EXAMPLE",
    "title": "Quick fix for bug 1234",
    "authorArn": "arn:aws:iam::123456789012:user/Nikhil_Jayashankar",
    "clientRequestToken": "d8d7612e-EXAMPLE",
    "creationDate": 1562619583.565,
    "pullRequestId": "27",
    "pullRequestStatus": "OPEN"
  }
}
```

3.

Um Genehmigungen für eine Pull-Anforderung anzuzeigen, führen Sie den Befehl `get-pull-request-approval-state` aus. Geben Sie dabei Folgendes an:

- ID der Pull-Anforderung (mit der Option `--pull-request-id`).
- Revisions-ID der Pull-Anforderung (mit `--revision-id` option)). Sie können die aktuelle Revisions-ID für eine Pull-Anforderung mit dem Befehl [get-pull-request](#) abrufen.

Um beispielsweise Genehmigungen für einen Pull Request mit der ID **8** und der Revision-ID wie folgt anzuzeigen: **9f29d167EXAMPLE**

```
aws codecommit get-pull-request-approval-state --pull-request-id 8 --revision-id 9f29d167EXAMPLE
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
  "approvals": [
    {
      "userArn": "arn:aws:iam::123456789012:user/Mary_Major",
      "approvalState": "APPROVE"
    }
  ]
}
```

4. Um Ereignisse in einer Pull-Anforderung anzuzeigen, führen Sie den Befehl `describe-pull-request-events` mit der Option `--pull-request-id` aus, wobei Sie die ID der Pull-Anforderung angeben. Um beispielsweise die Ereignisse für eine Pull-Anfrage mit der folgenden ID anzuzeigen: **8**

```
aws codecommit describe-pull-request-events --pull-request-id 8
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
  "pullRequestEvents": [
    {
      "pullRequestId": "8",
      "pullRequestEventType": "PULL_REQUEST_CREATED",
      "eventDate": 1510341779.53,
      "actor": "arn:aws:iam::111111111111:user/Zhang_Wei"
    },
    {
```

```

        "pullRequestStatusChangedEventMetadata": {
            "pullRequestStatus": "CLOSED"
        },
        "pullRequestId": "8",
        "pullRequestEventType": "PULL_REQUEST_STATUS_CHANGED",
        "eventDate": 1510341930.72,
        "actor": "arn:aws:iam::111111111111:user/Jane_Doe"
    }
}

```

5. Wenn Sie sehen möchten, ob für eine Pull-Anforderung Konflikte für die Zusammenführung vorliegen, führen Sie den Befehl `get-merge-conflicts` aus. Geben Sie dabei Folgendes an:
- Der Name des CodeCommit Repositorys (mit der `--repository-name` Option).
 - Branch, Tag, HEAD oder eine andere vollständig qualifizierte Referenz für die Quelle der Änderungen, die bei der Bewertung der Zusammenführung verwendet werden soll (mit der Option `--source-commit-specifier`).
 - Branch, Tag, HEAD oder eine andere vollständig qualifizierte Referenz für das Ziel der Änderungen, das bei der Bewertung der Zusammenführung verwendet werden soll (mit der Option `--destination-commit-specifier`).
 - Die zu verwendende Zusammenführungsoption (mit der Option `--merge-option`).

Um beispielsweise herauszufinden, ob es Zusammenführungskonflikte zwischen der Spitze eines Quell-Branches mit dem Namen *my-feature-branch* und einem Ziel-Branch mit dem Namen *main* eines Repositorys mit dem Namen *gibtMyDemoRepo*:

```

aws codecommit get-merge-conflicts --repository-name MyDemoRepo --source-commit-specifier my-feature-branch --destination-commit-specifier main --merge-option FAST_FORWARD_MERGE

```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```

{
    "destinationCommitId": "fac04518EXAMPLE",
    "mergeable": false,
    "sourceCommitId": "16d097f03EXAMPLE"
}

```

Überprüfen einer Pull-Anforderung

Du kannst die AWS CodeCommit Konsole verwenden, um die in einer Pull-Anfrage enthaltenen Änderungen zu überprüfen. Sie können der Anforderung, den Dateien und einzelnen Codezeilen Kommentare hinzufügen. Sie können auch auf Kommentare von anderen Benutzern antworten. Wenn Ihr Repository [mit Benachrichtigungen konfiguriert](#) ist, erhalten Sie E-Mails, wenn Benutzer auf Ihre Kommentare antworten oder eine Pull-Anforderung kommentieren.

Sie können die verwenden AWS CLI , um eine Pull-Anfrage zu kommentieren und auf Kommentare zu antworten. Um die Änderungen zu überprüfen, müssen Sie die CodeCommit Konsole, den git diff Befehl oder ein Diff-Tool verwenden.

Themen

- [Überprüfen Sie eine Pull-Anfrage \(Konsole\)](#)
- [Überprüfen Sie Pull-Requests \(AWS CLI\)](#)

Überprüfen Sie eine Pull-Anfrage (Konsole)

Sie können die CodeCommit Konsole verwenden, um eine Pull-Anfrage in einem CodeCommit Repository zu überprüfen.

1. Öffne die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie im Bereich Repositories (Repositorys) den Namen des Repositorys aus.
3. Wählen Sie im Navigationsbereich Pull requests (Pull-Anforderungen) aus.
4. Standardmäßig wird eine Liste aller offenen Pull-Anforderungen angezeigt. Wählen Sie die offene Pull-Anforderung aus, die Sie überprüfen möchten.

Developer Tools > CodeCommit > Repositories > MyDemoRepo > Pull requests

MyDemoRepo

Pull requests [Info](#)

Open pull requests ▼ Create pull request

Q

< 1 >

Pull request	Author	Destination	Last activity	Status	Approval status
20: A bugfix to add missing examples for S3	Saanvi_Sarkar	main	Just now	Open	No approval rules

Note

Sie können eine geschlossene oder zusammengeführte Pull-Anforderung kommentieren, aber Sie können sie nicht zusammenführen oder erneut öffnen.

- Wählen Sie in der Pull-Anforderung Changes.
- Führen Sie eine der folgenden Aktionen aus:
 - Um einen Kommentar zur Pull-Anforderung insgesamt hinzuzufügen, geben Sie unter Comments on changes (Kommentare zu Änderungen), New comment (Neuer Kommentar) einen Kommentar ein und wählen Sie dann Save (Speichern) aus. Sie können [Markdown](#) verwenden oder Ihren Kommentar als Klartext eingeben.

Comments on changes

New comment Preview markdown Learn more

Did we also change the variable name in blf.py and concat.py?

Save

- Um einer Datei im Commit einen Kommentar hinzuzufügen, suchen Sie in Changes den Namen der Datei. Wählen Sie das Kommentarsymbol



neben dem Dateinamen aus, geben Sie einen Kommentar ein und wählen Sie dann Save (Speichern) aus.

ahs_count.py [Browse file contents](#) [Comment on file](#)

New comment ☐ Preview markdown [Learn more](#)

[Save](#) [Cancel](#)

- Um einer geänderte Zeile in der Pull-Anfrage einen Kommentar hinzuzufügen, suchen Sie in Changes (Änderungen) die Zeile, die kommentiert werden soll. Wählen Sie das für die Zeile angezeigte Kommentarsymbol



aus, geben Sie einen Kommentar ein und wählen Sie dann Save (Speichern) aus.

Developer Tools [Close](#)

CodeCommit

- Source • CodeCommit
 - Getting started
 - Repositories
 - Code
 - Pull requests**
 - Commits
 - Branches
 - Tags
 - Settings
- Build • CodeBuild
- Deploy • CodeDeploy
- Pipeline • CodePipeline

Mergeable [Learn more](#)

Details Activity **Changes** Commits

< Page 1 of 1 > ▼

☒ Hide whitespace changes ☐ Unified ☒ Split

ahs_count.py [Browse file contents](#) [Comment on file](#)

```

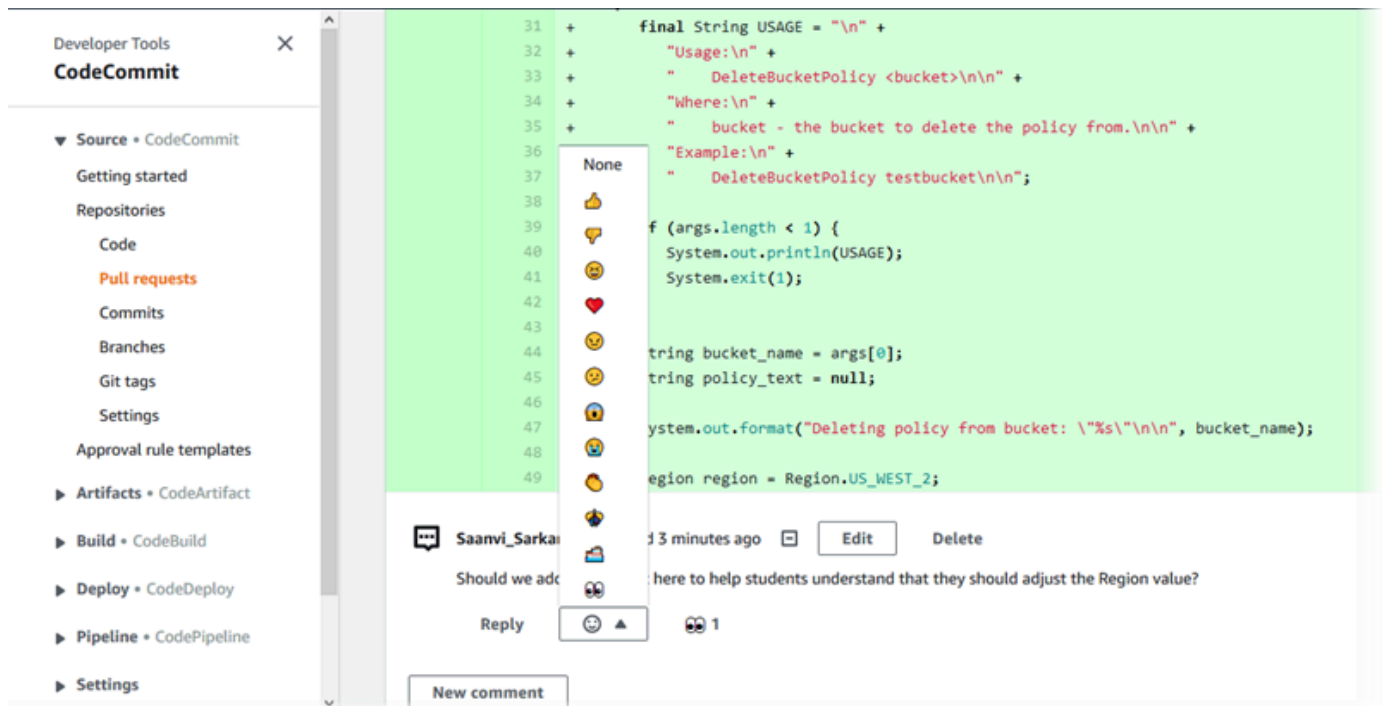
*** @ -5,6 +5,6 @@
5
6 total = (ess + z)
7 ahs = "Number of alveolar hissing sibilants: {}"
8 - print(ahs.format(total))
  + print(ahv.format(total))
  
```

New comment ☐ Preview markdown [Learn more](#)

You've switched back to the old variable, which won't work. This should be ahs.

[Save](#) [Cancel](#)

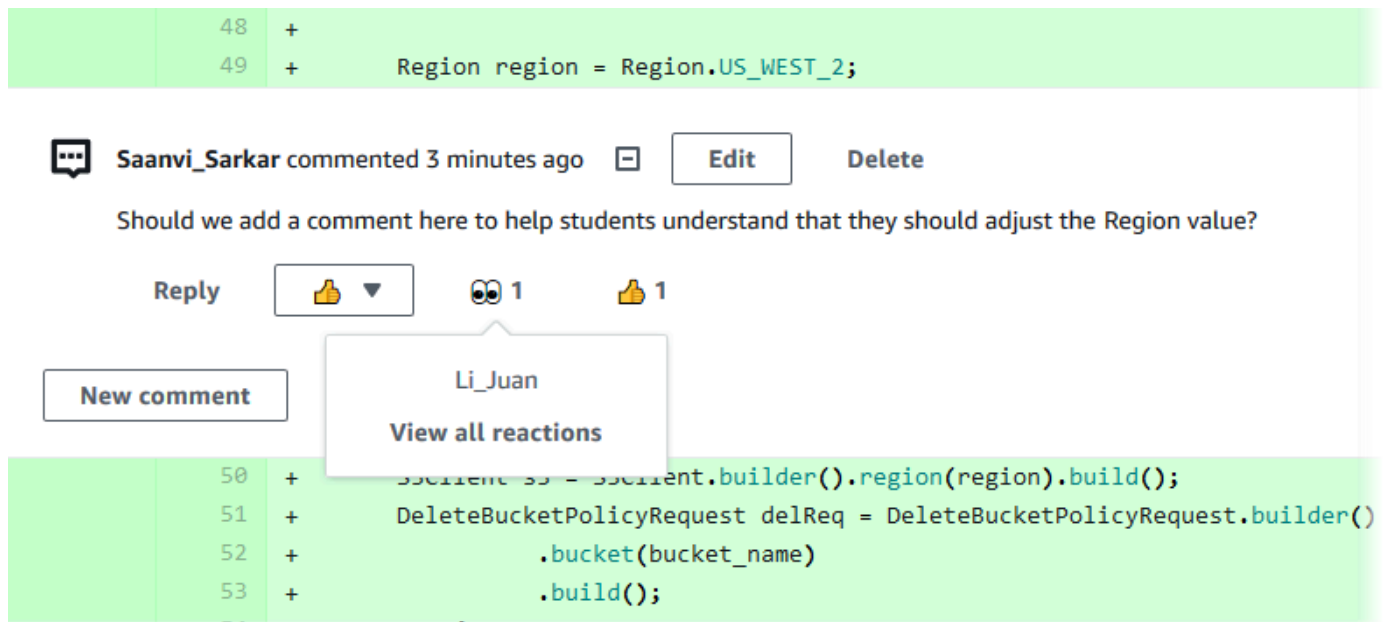
7. Zum Beantworten von Kommentaren im Commit wählen Sie unter Changes oder Activity die Option Reply aus. Sie können mit Text und Emojis antworten.



Sie können die Namen der Personen anzeigen, die mit einer bestimmten Emoji-Reaktion geantwortet haben, indem Sie diese auswählen. Um alle Emoji-Reaktionen und Informationen darüber zu sehen, wer mit welchen Emojis geantwortet hat, wähle Alle Reaktionen anzeigen. Wenn du mit einem Emoji auf einen Kommentar geantwortet hast, wird deine Antwort im Symbol für die Emoji-Reaktionsschaltfläche angezeigt.

Note

Die Anzahl der Reaktionen, die in der Konsole angezeigt werden, entspricht dem Zeitpunkt, zu dem die Seite geladen wurde. Aktuellste Informationen zur Anzahl der Emoji-Reaktionen erhalten Sie, wenn Sie entweder die Seite aktualisieren oder „Alle Reaktionen anzeigen“ wählen.



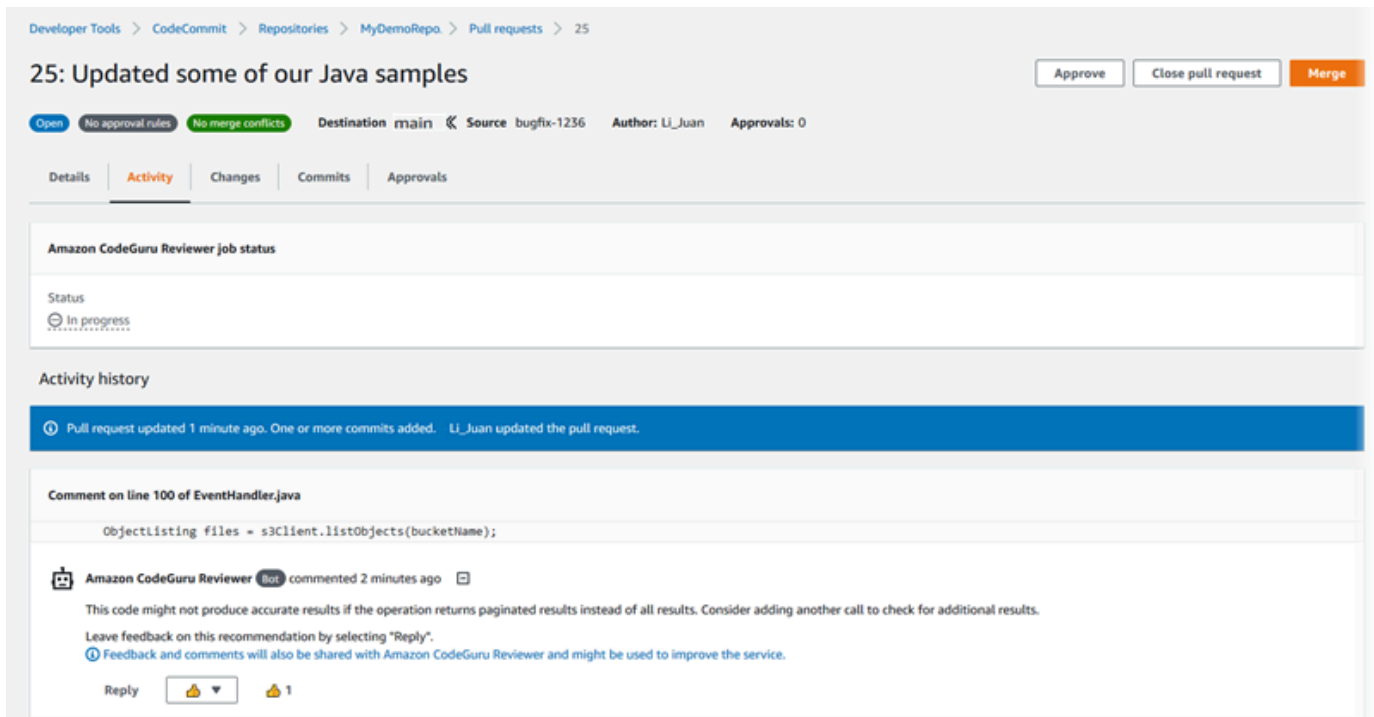
8. (Optional) Um auf eine von Amazon CodeGuru Reviewer erstellte Empfehlung zu antworten und auch Feedback zur Qualität der Empfehlung zu geben, wählen Sie Antworten. Verwenden Sie die Reaktionsschaltflächen, um allgemein anzugeben, ob Sie die Empfehlung genehmigen oder ablehnen. Verwenden Sie das Kommentarfeld, um weitere Details zur Reaktion anzugeben.

Note

Amazon CodeGuru Reviewer ist ein automatisierter Code-Review-Service, der mithilfe von Programmanalyse und maschinellem Lernen häufig auftretende Probleme erkennt und Korrekturen in Ihrem Java- oder Python-Code empfiehlt.

- Sie sehen Amazon CodeGuru Reviewer-Kommentare nur, wenn Sie das Repository mit Amazon CodeGuru Reviewer verknüpft haben, wenn die Analyse abgeschlossen ist und wenn der Code in der Pull-Anfrage Java- oder Python-Code ist. Weitere Informationen finden Sie unter [Ein AWS CodeCommit Repository mit Amazon CodeGuru Reviewer verknüpfen oder die Zuordnung aufheben](#).
- Kommentare von Amazon CodeGuru Reviewer werden nur dann auf der Registerkarte „Änderungen“ angezeigt, wenn sie zur letzten Version der Pull-Anfrage abgegeben wurden. Sie werden immer auf der Registerkarte Aktivität angezeigt.
- Sie können zwar mit allen verfügbaren Emoji-Reaktionen auf Empfehlungen von Amazon CodeGuru Rezensenten antworten, aber nur Emoji-Reaktionen mit Daumen

hoch und Daumen runter werden verwendet, um die Nützlichkeit der Empfehlung zu bewerten.



9. Um die in einer Pull-Anforderung vorgenommenen Änderungen zu genehmigen, wählen Sie Approve (Genehmigen).

Note

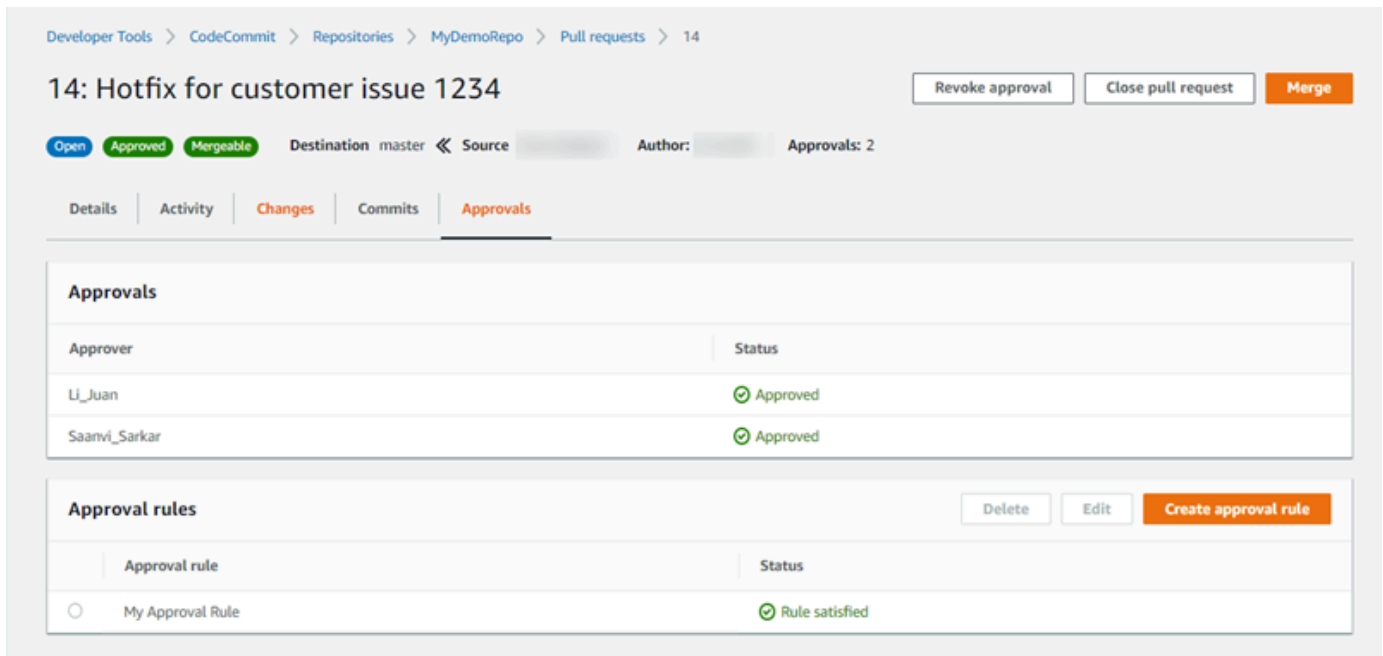
Sie können eine von Ihnen erstellte Pull-Anforderung nicht genehmigen.

Genehmigungen, Genehmigungsregeln für eine Pull-Anforderung und von Genehmigungsregelvorlagen erstellte Genehmigungsregeln sehen Sie unter Approvals (Genehmigungen). Wenn Sie die Pull-Anforderung anschließend nicht genehmigen möchten, können Sie die Option Revoke approval (Genehmigung widerrufen) auswählen.

Note

Sie können Genehmigungen nur für offene Pull-Anforderungen bestätigen oder widerrufen. Sie können keine Genehmigung für eine Pull-Anforderung bestätigen oder

widerrufen, die den Status „Merged (Zusammengeführt)“ oder „Closed (Geschlossen)“ hat.



Überprüfen Sie Pull-Requests (AWS CLI)

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter [Befehlszeilenreferenz](#).

Sie können Pull-Requests mit den folgenden AWS CLI Befehlen überprüfen:

- [post-comment-for-pull-request](#), um einen Kommentar zu einer Pull-Anfrage hinzuzufügen
- [get-comments-for-pull-request](#), um Kommentare zu sehen, die zu einer Pull-Anfrage hinterlassen wurden
- [update-pull-request-approval-state](#), um die Genehmigung für einen Pull-Request zu genehmigen oder zu widerrufen
- [post-comment-reply](#), um auf einen Kommentar in einer Pull-Anfrage zu antworten

Du kannst Emojis auch mit Kommentaren in einer Pull-Anfrage mit den folgenden Befehlen verwenden:

- Um auf einen Kommentar mit einem Emoji zu antworten, führe den Befehl aus. [put-comment-reaction](#)
- Um Emoji-Reaktionen auf einen Kommentar anzuzeigen, führe den Befehl aus. [get-comment-reactions](#)

Um den zu verwenden AWS CLI , um Pull-Requests in einem CodeCommit Repository zu überprüfen

1. Um einer Pull-Anforderung in einem Repository einen Kommentar hinzuzufügen, führen Sie den Befehl `post-comment-for-pull-request` aus. Geben Sie dabei Folgendes an:
 - ID der Pull-Anforderung (mit der Option `--pull-request-id`).
 - Den Namen des Repositorys, das die Pull-Anforderung enthält (mit der Option `--repository-name`).
 - Die vollständige Commit-ID des Commits im Ziel-Branch, in dem die Pull-Anforderung zusammengeführt wird (mit der Option `--before-commit-id`).
 - Die vollständige Commit-ID des Commits im Quell-Branch, die die aktuelle Spitze des Branch für die Pull-Anforderung darstellt, wenn Sie den Kommentar posten (mit der Option `--after-commit-id`).
 - Einzigartiger, vom Client generierter Idempotenz-Token (mit der Option `--client-request-token`).
 - Der Inhalt Ihres Kommentars (mit der Option `--content`).
 - Eine Liste der Speicherortangaben, wo der Kommentar abgelegt werden soll, unter anderem:
 - Der Name der zu vergleichenden Datei, einschließlich der Erweiterung und des Unterverzeichnisses (mit dem Attribut `filePath`).
 - Die Zeilennummer der Änderung innerhalb einer verglichenen Datei (mit dem Attribut `filePosition`).
 - Die Angabe, ob der Kommentar zu der Änderung „vor“ oder „nach“ dem Vergleich zwischen den Quell- und Ziel-Branches steht (mit dem Attribut `relativeFileVersion`).

Verwenden Sie diesen Befehl beispielsweise, um der `ahs_count.py` Datei einen Kommentar *"These don't appear to be used anywhere. Can we remove them?"* zur Änderung in einer Pull-Anfrage mit der ID von `47` in einem Repository mit dem Namen `MyDemoRepo` hinzuzufügen.

```
aws codecommit post-comment-for-pull-request --pull-request-id "47" --  
repository-name MyDemoRepo --before-commit-id 317f8570EXAMPLE --after-
```

```
commit-id 5d036259EXAMPLE --client-request-token 123Example --content
"These don't appear to be used anywhere. Can we remove them?" --location
filePath=ahs_count.py,filePosition=367,relativeFileVersion=AFTER
```

Ist der Befehl erfolgreich, wird eine Ausgabe wie die folgende erzeugt.

```
{
  "afterBlobId": "1f330709EXAMPLE",
  "afterCommitId": "5d036259EXAMPLE",
  "beforeBlobId": "80906a4cEXAMPLE",
  "beforeCommitId": "317f8570EXAMPLE",
  "comment": {
    "authorArn": "arn:aws:iam::111111111111:user/Saanvi_Sarkar",
    "clientRequestToken": "123Example",
    "commentId": "abcd1234EXAMPLEb5678efgh",
    "content": "These don't appear to be used anywhere. Can we remove
them?",
    "creationDate": 1508369622.123,
    "deleted": false,
    "lastModifiedDate": 1508369622.123,
    "callerReactions": [],
    "reactionCounts": []
  },
  "location": {
    "filePath": "ahs_count.py",
    "filePosition": 367,
    "relativeFileVersion": "AFTER"
  },
  "repositoryName": "MyDemoRepo",
  "pullRequestId": "47"
}
```

2. Zum Anzeigen von Kommentaren für eine Pull-Anforderung führen Sie den Befehl `get-comments-for-pull-request` aus. Geben Sie dabei Folgendes an:
 - Der Name des CodeCommit Repositorys (mit der `--repository-name` Option).
 - Die vom System generierte ID der Pull-Anforderung (mit der Option `--pull-request-id`).
 - (Optional) Ein Aufzählungs-Token zum Zurückgeben des nächsten Ergebnisstapels (mit der Option `--next-token`).
 - (Optional) Eine nicht negative ganze Zahl, um die Anzahl der zurückgegebenen Ergebnisse zu begrenzen (mit der Option `--max-results`).

Verwenden Sie diesen Befehl beispielsweise, um Kommentare für eine Pull-Anforderung mit der ID 42 anzuzeigen.

```
aws codecommit get-comments-for-pull-request --pull-request-id 42
```

Ist der Befehl erfolgreich, wird eine Ausgabe wie die folgende erzeugt.

```
{
  "commentsForPullRequestData": [
    {
      "afterBlobId": "1f330709EXAMPLE",
      "afterCommitId": "5d036259EXAMPLE",
      "beforeBlobId": "80906a4cEXAMPLE",
      "beforeCommitId": "317f8570EXAMPLE",
      "comments": [
        {
          "authorArn": "arn:aws:iam::111111111111:user/Saanvi_Sarkar",
          "clientRequestToken": "",
          "commentId": "abcd1234EXAMPLEb5678efgh",
          "content": "These don't appear to be used anywhere. Can we remove
them?",
          "creationDate": 1508369622.123,
          "deleted": false,
          "lastModifiedDate": 1508369622.123,
          "callerReactions": [],
          "reactionCounts": {
            "THUMBSUP" : 6,
            "CONFUSED" : 1
          }
        },
        {
          "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
          "clientRequestToken": "",
          "commentId": "442b498bEXAMPLE5756813",
          "content": "Good catch. I'll remove them.",
          "creationDate": 1508369829.104,
          "deleted": false,
          "lastModifiedDate": 150836912.273,
          "callerReactions": ["THUMBSUP"]
          "reactionCounts":
```

```
        {
            "THUMBSUP" : 14
        }
    ],
    "location": {
        "filePath": "ahs_count.py",
        "filePosition": 367,
        "relativeFileVersion": "AFTER"
    },
    "repositoryName": "MyDemoRepo",
    "pullRequestId": "42"
}
],
"nextToken": "exampleToken"
}
```

3.

Um die Genehmigung für eine Pull-Anforderung zu bestätigen oder zu widerrufen, führen Sie den Befehl `update-pull-request-approval-state` aus und geben Sie Folgendes an:

- ID der Pull-Anforderung (mit der Option `--pull-request-id`).
- Revisions-ID der Pull-Anforderung (mit `--revision-id` option)). Sie können die aktuelle Revisions-ID für eine Pull-Anforderung mit dem Befehl [get-pull-request](#) abrufen.
- Der Genehmigungsstatus, den Sie zuweisen möchten (mit der Option `--approval-state`). Gültige Genehmigungsstatus sind `APPROVE` und `REVOKE`.

Verwenden Sie diesen Befehl beispielsweise, um eine Pull-Anfrage mit der ID von **27** und der Revision-ID von zu genehmigen **9f29d167EXAMPLE**.

```
aws codecommit update-pull-request-approval-state --pull-request-id 27 --revision-id 9f29d167EXAMPLE --approval-state "APPROVE"
```

Bei erfolgreicher Ausführung gibt dieser Befehl nichts zurück.

4. Um eine Antwort auf einen Kommentar in einer Pull-Anforderung hinzuzufügen, führen Sie den Befehl `post-comment-reply` aus. Geben Sie dabei Folgendes an:

- Die vom System generierte ID des Kommentars, auf den Sie antworten wollen (mit der Option `--in-reply-to`).
- Einzigartiger, vom Client generierter Idempotenz-Token (mit der Option `--client-request-token`).

- Der Inhalt Ihrer Antwort (mit der Option `--content`).

Verwenden Sie diesen Befehl beispielsweise, um die Antwort *"Good catch. I'll remove them."* auf den Kommentar mit der vom System generierten ID von hinzuzufügen. *abcd1234EXAMPLEb5678efgh*

```
aws codecommit post-comment-reply --in-reply-to abcd1234EXAMPLEb5678efgh --
content "Good catch. I'll remove them." --client-request-token 123Example
```

Ist der Befehl erfolgreich, wird eine Ausgabe wie die folgende erzeugt.

```
{
  "comment": {
    "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
    "clientRequestToken": "123Example",
    "commentId": "442b498bEXAMPLE5756813",
    "content": "Good catch. I'll remove them.",
    "creationDate": 1508369829.136,
    "deleted": false,
    "lastModifiedDate": 150836912.221,
    "callerReactions": [],
    "reactionCounts": []
  }
}
```

Aktualisieren einer Pull-Anforderung

Sie können eine Pull-Anfrage mit weiteren Codeänderungen aktualisieren, indem Sie Commits in den Quell-Branch eines offenen Pull-Requests übertragen. Weitere Informationen finden Sie unter [Erstellen Sie einen Commit in AWS CodeCommit](#).

Du kannst die AWS CodeCommit Konsole oder die verwenden AWS CLI , um den Titel oder die Beschreibung eines Pull-Requests zu aktualisieren. Möglicherweise möchten Sie den Titel oder die Beschreibung der Pull-Anfrage aus folgenden Gründen aktualisieren:

- Andere Benutzer verstehen die Beschreibung nicht oder der ursprüngliche Titel ist irreführend.
- Sie möchten, dass der Titel oder die Beschreibung die Änderungen berücksichtigt, die am Quell-Branch einer offenen Pull-Anforderung vorgenommen wurden.

Aktualisiere eine Pull-Anfrage (Konsole)

Sie können die CodeCommit Konsole verwenden, um den Titel und die Beschreibung einer Pull-Anfrage in einem CodeCommit Repository zu aktualisieren. Um den Code in der Pull-Anfrage zu aktualisieren, senden Sie Push-Commits an den Quell-Branch eines offenen Pull-Requests.

1. Öffne die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, in dem Sie eine Pull-Anforderungen aktualisieren möchten.
3. Wählen Sie im Navigationsbereich Pull requests (Pull-Anforderungen) aus.
4. Standardmäßig wird eine Liste aller offenen Pull-Anforderungen angezeigt. Wählen Sie die offene Pull-Anforderung aus, die Sie aktualisieren möchten.
5. Wählen Sie in der Pull-Anforderung Details und dann Edit Details (Details bearbeiten) zum Bearbeiten des Titels oder der Beschreibung aus.

Note

Den Titel oder die Beschreibung einer geschlossenen oder zusammengeführten Pull-Anforderung können Sie nicht aktualisieren.

Pull-Anfragen aktualisieren (AWS CLI)

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter [Befehlszeilenreferenz](#).

Auch folgende Befehle können interessant sein:

- [update-pull-request-approval-state](#), um die Genehmigung für eine Pull-Anforderung zu gewähren oder zu widerrufen.
- [create-pull-request-approval-rule](#), um eine Genehmigungsregel für eine Pull-Anforderung zu erstellen.
- [delete-pull-request-approval-rule](#), um eine Genehmigungsregel für eine Pull-Anforderung zu löschen.
- [Erstelle einen Commit AWS CLI](#) oder [Einen Commit mit einem Git-Client erstellen](#), um zusätzliche Codeänderungen zu erstellen und in den Quell-Branch einer offenen Pull-Anfrage zu übertragen.

Um den zu verwenden AWS CLI , um Pull-Requests in einem CodeCommit Repository zu aktualisieren

1. Um den Titel einer Pull-Anforderung in einem Repository zu aktualisieren, führen Sie den Befehl `update-pull-request-title` aus, wobei Sie Folgendes angeben:

- ID der Pull-Anforderung (mit der Option `--pull-request-id`).
- Den Titel der Pull-Anforderung (mit der Option `--title`).

Um beispielsweise den Titel einer Pull-Anfrage mit der folgenden ID zu aktualisieren⁴⁷:

```
aws codecommit update-pull-request-title --pull-request-id 47 --title
"Consolidation of global variables - updated review"
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
  "pullRequest": {
    "approvalRules": [
      {
        "approvalRuleContent": "{\"Version\": \"2018-11-08\",
        \"DestinationReferences\": [\"refs/heads/main\"], \"Statements\": [{\"Type
        \": \"Approvers\", \"NumberOfApprovalsNeeded\": 2, \"ApprovalPoolMembers\":
        [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
        "approvalRuleId": "dd8b17fe-EXAMPLE",
        "approvalRuleName": "2-approver-rule-for-main",
        "creationDate": 1571356106.936,
        "lastModifiedDate": 571356106.936,
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
        "originApprovalRuleTemplate": {
          "approvalRuleTemplateId": "dd8b26gr-EXAMPLE",
          "approvalRuleTemplateName": "2-approver-rule-for-main"
        },
        "ruleContentSha256": "4711b576EXAMPLE"
      }
    ],
    "authorArn": "arn:aws:iam::123456789012:user/Li_Juan",
    "clientRequestToken": "",
    "creationDate": 1508530823.12,
```



```

    "description": "Review the latest changes and updates to the global
variables. I have updated this request with some changes, including removing some
unused variables.",
    "lastActivityDate": 1508372657.188,
    "pullRequestId": "47",
    "pullRequestStatus": "OPEN",
    "pullRequestTargets": [
        {
            "destinationCommit": "9f31c968EXAMPLE",
            "destinationReference": "refs/heads/main",
            "mergeMetadata": {
                "isMerged": false,
            },
            "repositoryName": "MyDemoRepo",
            "sourceCommit": "99132ab0EXAMPLE",
            "sourceReference": "refs/heads/variables-branch"
        }
    ],
    "title": "Consolidation of global variables - updated review"
}

```

2. Um die Beschreibung einer Pull-Anforderung zu aktualisieren, führen Sie den Befehl `update-pull-request-description` aus, wobei Sie Folgendes angeben:
 - ID der Pull-Anforderung (mit der Option `--pull-request-id`).
 - Die Beschreibung (mit der `--description`-Option).

Um beispielsweise die Beschreibung einer Pull-Anfrage mit der folgenden ID zu aktualisieren⁴⁷:

```
aws codecommit update-pull-request-description --pull-request-id 47 --description
"Updated the pull request to remove unused global variable."
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```

{
  "pullRequest": {
    "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
    "clientRequestToken": "",
    "creationDate": 1508530823.155,
    "description": "Updated the pull request to remove unused global variable.",

```

```
{
  "lastActivityDate": 1508372423.204,
  "pullRequestId": "47",
  "pullRequestStatus": "OPEN",
  "pullRequestTargets": [
    {
      "destinationCommit": "9f31c968EXAMPLE",
      "destinationReference": "refs/heads/main",
      "mergeMetadata": {
        "isMerged": false,
      },
      "repositoryName": "MyDemoRepo",
      "sourceCommit": "99132ab0EXAMPLE",
      "sourceReference": "refs/heads/variables-branch"
    }
  ],
  "title": "Consolidation of global variables"
}
```

Genehmigungsregel für einen Pull Request bearbeiten oder löschen

Falls eine Genehmigungsregel für eine Pull-Anforderung vorliegt, können Sie diese Pull-Anforderung erst zusammenführen, wenn die Bedingungen erfüllt sind. Sie können die Genehmigungsregeln für Pull-Anforderungen ändern, um die Erfüllung der Bedingungen zu erleichtern oder für striktere Prüfungen zu sorgen. Sie können die Anzahl der Benutzer ändern, die eine Pull-Anforderung genehmigen müssen. Sie können für die Regel Mitgliedschaften in einem Genehmigungs-Pool von Benutzern hinzufügen, entfernen oder ändern. Wenn Sie eine Genehmigungsregel für eine Pull-Anforderung nicht mehr verwenden möchten, können Sie diese löschen.

Note

Sie können Genehmigungsregeln für eine Pull-Anforderung auch außer Kraft setzen. Weitere Informationen finden Sie unter [Genehmigungsregeln für eine Pull-Anfrage überschreiben](#).

Du kannst die AWS CodeCommit Konsole oder die verwenden AWS CLI , um Genehmigungsregeln für dein Repository zu bearbeiten und zu löschen.

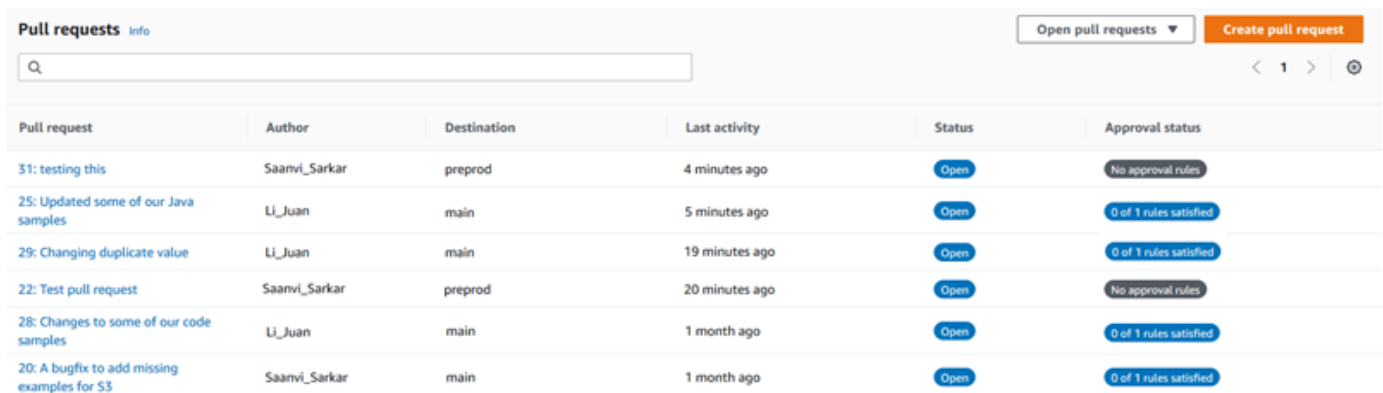
Themen

- [Bearbeiten oder löschen Sie eine Genehmigungsregel für eine Pull-Anfrage \(Konsole\)](#)
- [Bearbeiten oder löschen Sie eine Genehmigungsregel für eine Pull-Anfrage \(AWS CLI\)](#)

Bearbeiten oder löschen Sie eine Genehmigungsregel für eine Pull-Anfrage (Konsole)

Sie können die CodeCommit Konsole verwenden, um eine Genehmigungsregel für eine Pull-Anfrage in einem CodeCommit Repository zu bearbeiten oder zu löschen.

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories (Repositoryys) den Namen des Repositorys aus, in dem Sie eine Genehmigungsregel für eine Pull-Anforderung bearbeiten oder löschen möchten.
3. Wählen Sie im Navigationsbereich Pull Requests aus.
4. Wählen Sie die Pull-Anforderung aus, in der Sie eine Genehmigungsregel bearbeiten oder löschen möchten. Sie können Genehmigungsregeln nur für offene Pull-Anforderungen bearbeiten und löschen.



Pull request	Author	Destination	Last activity	Status	Approval status
31: testing this	Saanvi_Sarkar	preprod	4 minutes ago	Open	No approval rules
25: Updated some of our Java samples	Li_Juan	main	5 minutes ago	Open	0 of 1 rules satisfied
29: Changing duplicate value	Li_Juan	main	19 minutes ago	Open	0 of 1 rules satisfied
22: Test pull request	Saanvi_Sarkar	preprod	20 minutes ago	Open	No approval rules
28: Changes to some of our code samples	Li_Juan	main	1 month ago	Open	0 of 1 rules satisfied
20: A bugfix to add missing examples for S3	Saanvi_Sarkar	main	1 month ago	Open	0 of 1 rules satisfied

5. Wählen Sie in der Pull-Anforderung Approvals (Genehmigungen) und dann die Regel aus, die Sie bearbeiten oder aus der Liste löschen möchten. Führen Sie eine der folgenden Aktionen aus:
 - Wenn Sie die Regel bearbeiten möchten, wählen Sie Edit (Bearbeiten) aus.
 - Wenn Sie die Regel löschen möchten, wählen Sie Delete (Löschen) aus und befolgen Sie dann die Anweisungen, mit denen das Löschen der Regel verifiziert werden kann.
6. Nehmen Sie unter Edit approval rule (Genehmigungsregel bearbeiten) die gewünschten Änderungen an der Regel vor und wählen Sie dann Submit (Senden).

Edit approval rule

Rule details

Rule name

My Approval Rule

Number of approvals needed

1

Approval pool members - *optional*

If approval pool members are specified, only approvals from these members will count toward satisfying this rule. Use a wildcard to match multiple approvers with one value.

Approver type [Info](#)

Value

CodeCommit ▼

Mary_Major

Remove

CodeCommit ▼

Li_Juan

Remove

CodeCommit ▼

Saanvi_Sarkar

Remove

Fully qualified ARN ▼

arn:aws:iam:::user/

Remove

Add

Cancel

Submit

7. Wählen Sie nach dem Konfigurieren der Genehmigungsregel Submit (Senden).

Bearbeiten oder löschen Sie eine Genehmigungsregel für eine Pull-Anfrage (AWS CLI)

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter [Befehlszeilenreferenz](#).

Sie können die verwenden AWS CLI , um den Inhalt einer Genehmigungsregel zu bearbeiten und eine Genehmigungsregel zu löschen.

 Note

Auch folgende Befehle können interessant sein:

- [update-pull-request-approval-state](#), um die Genehmigung für eine Pull-Anforderung zu gewähren oder zu widerrufen.
- [get-pull-request-approval-states](#), um die Genehmigungen für die Pull-Anforderung anzuzeigen.
- [evaluate-pull-request-approval-rules](#), um festzustellen, ob die Bedingungen der Genehmigungsregeln für eine Pull-Anforderung erfüllt sind.

Um die zu verwenden AWS CLI , um eine Genehmigungsregel für einen Pull Request in einem CodeCommit Repository zu bearbeiten oder zu löschen

1. Um eine Genehmigungsregel zu bearbeiten, führen Sie den Befehl `update-pull-request-approval-rule-content` aus, indem Sie Folgendes angeben:

- ID der Pull-Anforderung (mit der Option `--id`).
- Name der Genehmigungsregel (mit der Option `--approval-rule-name`).
- Inhalt der Genehmigungsregel (mit der Option `--approval-rule-content`).

In diesem Beispiel wird eine Genehmigungsregel aktualisiert, *Require two approved approvers* die nach einer Pull-Anfrage benannt ist und die ID von `27` hat. Die Regel erfordert eine Benutzergenehmigung aus einem Genehmigungspool, der alle IAM-Benutzer im *123456789012* Amazon Web Services Services-Konto umfasst:

```
aws codecommit update-pull-request-approval-rule-content --pull-request-id 27
--approval-rule-name "Require two approved approvers" --approval-rule-content
"{Version: 2018-11-08, Statements: [{Type: \"Approvers\", NumberOfApprovalsNeeded:
1, ApprovalPoolMembers:[\"CodeCommitApprovers:123456789012:user/*\"]}]}"
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
  "approvalRule": {
```

```

    "approvalRuleContent": "{Version: 2018-11-08, Statements:
    [{Type: \"Approvers\", NumberOfApprovalsNeeded: 1, ApprovalPoolMembers:
    [\"CodeCommitApprovers:123456789012:user/*\"]}]}",
    "approvalRuleId": "aac33506-EXAMPLE",
    "originApprovalRuleTemplate": {},
    "creationDate": 1570752871.932,
    "lastModifiedDate": 1570754058.333,
    "approvalRuleName": Require two approved approvers,
    "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
    "ruleContentSha256": "cd93921cEXAMPLE",
  }
}

```

3.

Um eine Genehmigungsregel zu löschen, führen Sie den Befehl `delete-pull-request-approval-rule` aus, indem Sie Folgendes angeben:

- ID der Pull-Anforderung (mit der Option `--id`).
- Name der Genehmigungsregel (mit der Option `--approval-rule-name`).

Um beispielsweise eine Genehmigungsregel mit dem Namen *My Approval Rule* für eine Pull-Anfrage mit der folgenden ID zu löschen: *15*

```
aws codecommit delete-pull-request-approval-rule --pull-request-id 15 --approval-rule-name "My Approval Rule"
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```

{
  "approvalRuleId": "077d8e8a8-EXAMPLE"
}

```

Genehmigungsregeln für eine Pull-Anfrage überschreiben

Im normalen Verlauf der Entwicklung möchten Sie, dass Benutzer die Bedingungen der Genehmigungsregeln erfüllen, bevor Sie Pull-Anforderungen zusammenführen. Es kann jedoch vorkommen, dass Sie das Zusammenführen einer Pull-Anforderung beschleunigen müssen. Beispiel: Sie möchten eine Fehlerbehebung in die Produktion übernehmen, aber niemand im Genehmigungs-Pool ist verfügbar, um die Pull-Anforderung zu genehmigen. In solchen Fällen

können Sie die Genehmigungsregeln für eine Pull-Anforderung außer Kraft setzen. Sie können alle Genehmigungsregeln für eine Pull-Anforderung außer Kraft setzen, einschließlich derjenigen, die speziell für die Pull-Anforderung erstellt und aus einer Genehmigungsregelvorlage generiert wurden. Sie können nicht selektiv eine bestimmte Genehmigungsregel außer Kraft setzen, sondern nur alle Regeln. Nachdem Sie die Genehmigungsregelanforderungen durch Außerkraftsetzen der Regeln aufgehoben haben, können Sie die Pull-Anforderung mit dem Ziel-Branch zusammenführen.

Wenn Sie Genehmigungsregeln für eine Pull-Anforderung außer Kraft setzen, werden Informationen über den Benutzer, der die Regeln außer Kraft setzt, in der Aktivität für die Pull-Anforderung aufgezeichnet. Auf diese Weise können Sie den Verlauf einer Pull-Anforderung rückverfolgen und ermitteln, wer die Regeln außer Kraft gesetzt hat. Sie können die Außerkraftsetzung auch widerrufen, wenn die Pull-Anforderung noch geöffnet ist. Nachdem die Pull-Anforderung zusammengeführt wurde, können Sie die Außerkraftsetzung nicht mehr widerrufen.

Themen

- [Genehmigungsregeln überschreiben \(Konsole\)](#)
- [Genehmigungsregeln außer Kraft setzen \(\)AWS CLI](#)

Genehmigungsregeln überschreiben (Konsole)

Sie können die Anforderungen von Genehmigungsregeln für eine Pull-Anforderung in der Konsole im Rahmen der Prüfung einer Pull-Anforderung außer Kraft setzen. Wenn Sie Ihre Meinung ändern, können Sie die Außerkraftsetzung widerrufen. Die Genehmigungsregelanforderungen werden dann wieder angewendet. Sie können Genehmigungsregeln nur außer Kraft setzen oder eine Außerkraftsetzung widerrufen, wenn die Pull-Anforderung noch geöffnet ist. Nachdem sie zusammengeführt oder geschlossen wurde, können Sie den Außerkraftsetzungsstatus nicht mehr ändern.

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie im Bereich Repositories (Repositoryys) den Namen des Repositorys aus.
3. Wählen Sie im Navigationsbereich Pull requests (Pull-Anforderungen) aus. Wählen Sie die Pull-Anforderung, für die die Genehmigungsregel außer Kraft gesetzt werden soll, oder widerrufen Sie eine Außerkraftsetzung.
4. Wählen Sie auf der Registerkarte Approvals (Genehmigungen) die Option Override approval rules (Genehmigungsregel außer Kraft setzen) aus. Die Anforderungen werden verlagert und

der Schaltflächentext wird in Revoke override (Außerkraftsetzung widerrufen) geändert. Um die Genehmigungsregelanforderungen wieder anzuwenden, wählen Sie Revoke override (Außerkraftsetzung widerrufen).

Genehmigungsregeln außer Kraft setzen (AWS CLI)

Sie können die verwenden AWS CLI , um die Anforderungen der Genehmigungsregeln außer Kraft zu setzen. Sie können auch den Außerkraftsetzungsstatus für eine Pull-Anforderung anzeigen.

So setzen Sie Genehmigungsregelanforderungen für eine Pull-Anforderung außer Kraft

1. Führen Sie am Terminal oder in der Befehlszeile den Befehl `override-pull-request-approval-rules` unter Angabe der folgenden Informationen aus:
 - Die vom System generierte ID der Pull-Anforderung.
 - Die neueste Revisions-ID der Pull-Anforderung. Verwenden Sie `get-pull-request`, um diese Informationen anzuzeigen.
 - Der zu überschreibende Status: `OVERRIDE` oder `REVOKE`. Der Status `REVOKE` entfernt den Status `OVERRIDE`, wird aber nicht gespeichert.

So setzen Sie beispielsweise Genehmigungsregeln für eine Pull-Anforderung mit der ID **34** und der Revisions-ID **927df8d8EXAMPLE** außer Kraft:

```
aws codecommit override-pull-request-approval-rules --pull-request-id 34 --  
revision-id 927df8d8EXAMPLE --override-status OVERRIDE
```

2. Bei erfolgreicher Ausführung gibt dieser Befehl nichts zurück.
3. So widerrufen Sie die Außerkraftsetzung für eine Pull-Anforderung mit der ID **34** und der Revisions-ID **927df8d8EXAMPLE**:

```
aws codecommit override-pull-request-approval-rules --pull-request-id 34 --  
revision-id 927df8d8EXAMPLE --override-status REVOKE
```

So erhalten Sie Informationen zum Außerkraftsetzungsstatus einer Pull-Anforderung

1. Führen Sie am Terminal oder in der Befehlszeile den Befehl `get-pull-request-override-state` unter Angabe der folgenden Informationen aus:

- Die vom System generierte ID der Pull-Anforderung.
- Die neueste Revisions-ID der Pull-Anforderung. Verwenden Sie `get-pull-request`, um diese Informationen anzuzeigen.

So zeigen Sie beispielsweise den Außerkraftsetzungsstatus für eine Pull-Anforderung mit der ID **34** und der Revisions-ID **927df8d8EXAMPLE** an:

```
aws codecommit get-pull-request-override-state --pull-request-id 34 --revision-id 927df8d8EXAMPLE
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
  "overridden": true,
  "overrider": "arn:aws:iam::123456789012:user/Mary_Major"
}
```

Einen Pull-Request in einem AWS CodeCommit Repository zusammenführen

Nachdem der Code geprüft wurde und alle Genehmigungsregeln (falls vorhanden) für die Pull-Anforderung erfüllt wurden, können Sie eine Pull-Anforderung auf verschiedene Arten zusammenführen:

- In der Konsole können Sie den Quell-Branch mit einer der verfügbaren Merge-Strategien mit dem Ziel-Branch zusammenführen, wodurch auch die Pull-Anforderung geschlossen wird. Sie können Zusammenführungskonflikte auch in der Konsole lösen. Die Konsole zeigt eine Meldung an, die angibt, ob die Pull-Anforderung zusammengeführt werden kann oder Konflikte gelöst werden müssen. Sobald alle Konflikte gelöst wurden und Sie Merge (Zusammenführen) auswählen, wird die Zusammenführung unter Anwendung der von Ihnen ausgewählten Merge-Strategie durchgeführt. `FAST_FORWARD` ist die Merge-Standardstrategie und die Standardoption für Git. Je nach dem Status des Codes in den Quell- und Ziel-Branche ist diese Strategie möglicherweise nicht verfügbar, dafür aber andere Optionen, wie z. B. Squashmerge oder 3-Wege-Merge.
- Sie können den verwenden AWS CLI , um den Pull-Request zusammenzuführen und zu schließen, indem Sie die Fast-Forward-, Squash- oder 3-Way-Merge-Strategie verwenden.

- Verwenden Sie auf Ihrem lokalen Computer den Befehl `git merge`, um den Quell-Branch mit dem Ziel-Branch zusammenzuführen, und übertragen Sie dann den zusammengeführten Code an den Ziel-Branch. Dieser Herangehensweise hat Nachteile, die Sie berücksichtigen müssen. Die Pull-Anforderung wird unabhängig davon zusammengeführt, ob die Anforderungen der Genehmigungsregeln für die Pull-Anforderung erfüllt wurden – diese Kontrollen werden umgangen. Durch das Zusammenführen und Pushen des Ziel-Branches wird die Pull-Anforderung auch automatisch geschlossen, wenn die Pull-Anforderung mithilfe der Merge-Strategie mit Vorlauf zusammengeführt wird. Ein Vorteil dieses Ansatzes besteht darin, dass Sie mit dem `git merge` Befehl Optionen oder Strategien zum Zusammenführen auswählen können, die in der Konsole nicht verfügbar sind. CodeCommit Weitere Informationen zu `git merge` und den Mergeoptionen finden Sie unter [git-merge](#) oder in Ihrer Git-Dokumentation.

CodeCommit schließt eine Pull-Anfrage automatisch, wenn entweder der Quell- oder der Ziel-Branch der Pull-Anfrage gelöscht wird.

Themen

- [Eine Pull-Anfrage zusammenführen \(Konsole\)](#)
- [Eine Pull-Anfrage zusammenführen \(AWS CLI\)](#)

Eine Pull-Anfrage zusammenführen (Konsole)

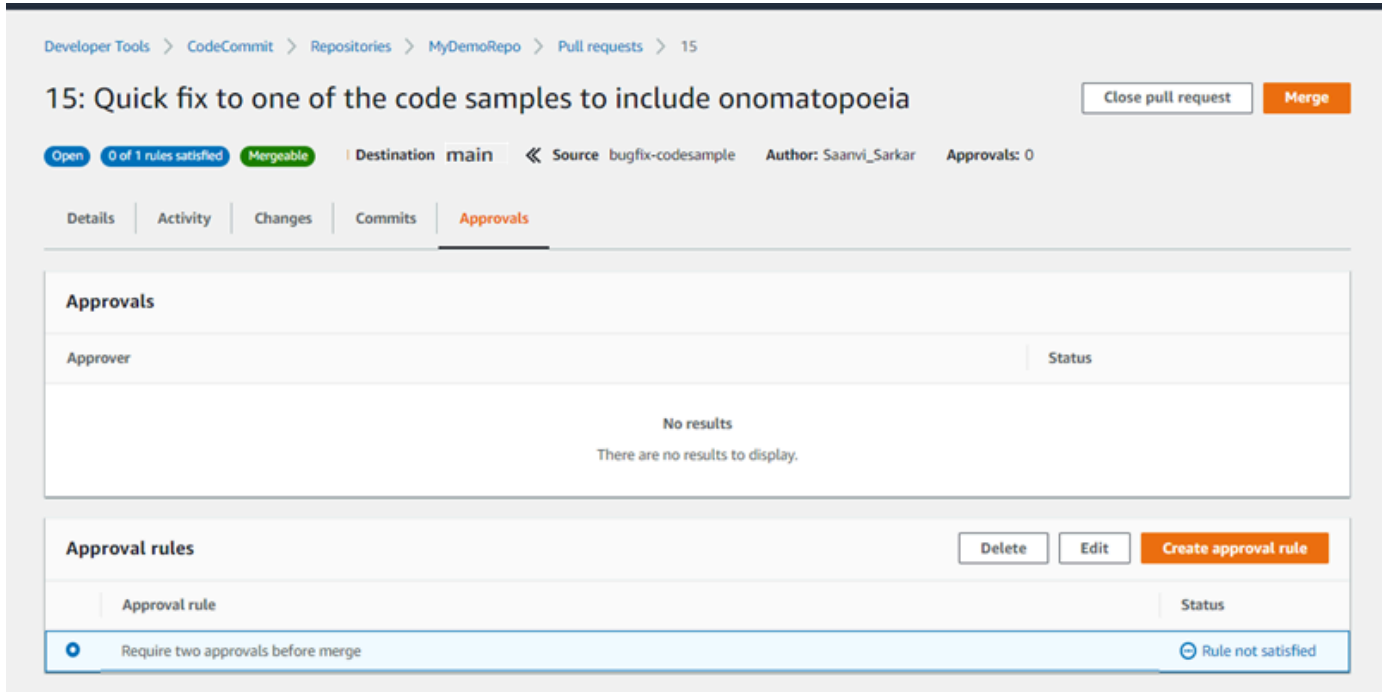
Du kannst die CodeCommit Konsole verwenden, um eine Pull-Anfrage in einem CodeCommit Repository zusammenzuführen. Nachdem der Status einer Pull-Anforderung in Merged (Zusammengeführt) geändert wurde, wird sie nicht mehr in der Liste offener Pull-Anforderungen angezeigt. Eine zusammengeführte Pull-Anforderung wird als geschlossen kategorisiert. Ihr Status kann nicht wieder in Open (Geöffnet) geändert werden, aber Benutzer können die Änderungen weiterhin kommentieren und auf Kommentare antworten. Nachdem eine Pull-Anforderung zusammengeführt oder geschlossen wurde, können Sie sie nicht mehr genehmigen, eine erteilte Genehmigung für sie widerrufen oder die Genehmigungsregeln außer Kraft setzen, die der Pull-Anforderung zugewiesen wurden.

1. Öffne die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie im Bereich Repositories (Repositorys) den Namen des Repositorys aus.
3. Wählen Sie im Navigationsbereich Pull requests (Pull-Anforderungen) aus.

4. Standardmäßig wird eine Liste aller offenen Pull-Anforderungen angezeigt. Wählen Sie die offene Pull-Anforderung aus, die Sie zusammenführen möchten.
5. Wählen Sie in der Pull-Anforderung die Option Approvals (Genehmigungen). Verifizieren Sie anhand der Liste der Genehmiger, dass die Bedingungen aller Genehmigungsregeln (falls vorhanden) erfüllt wurden. Sie können eine Pull-Anforderung nicht zusammenführen, wenn einzelne oder mehrere Genehmigungsregeln den Status Rule not satisfied (Regel nicht erfüllt) haben. Wenn kein Benutzer die Pull-Anforderung genehmigt hat, müssen Sie abwägen, ob sie zusammengeführt werden soll oder ob Sie auf Genehmigungen warten möchten.

Note

Wenn eine Genehmigungsregel für eine Pull-Anforderung erstellt wurde, können Sie sie bearbeiten oder löschen, um die Blockierung der Zusammenführung aufzuheben. Wenn die Genehmigungsregel mit einer Genehmigungsregelvorlage erstellt wurde, können Sie sie nicht bearbeiten oder löschen. Sie können sich nur entscheiden, die Anforderungen außer Kraft zu setzen. Weitere Informationen finden Sie unter [Genehmigungsregeln für eine Pull-Anfrage überschreiben](#).



The screenshot displays the AWS CodeCommit Pull Request interface for a pull request titled "15: Quick fix to one of the code samples to include onomatopoeia". The interface includes a breadcrumb trail: Developer Tools > CodeCommit > Repositories > MyDemoRepo > Pull requests > 15. At the top right, there are buttons for "Close pull request" and "Merge". Below the title, a status bar shows "Open", "0 of 1 rules satisfied", "Mergeable", "Destination: main", "Source: bugfix-codesample", "Author: Saanvi_Sarkar", and "Approvals: 0". The "Approvals" tab is selected, showing a table with columns "Approver" and "Status". The table is empty, with a message: "No results. There are no results to display." Below the table, the "Approval rules" section is visible, containing a table with one rule: "Require two approvals before merge". The status of this rule is "Rule not satisfied". Buttons for "Delete", "Edit", and "Create approval rule" are present.

6. Wählen Sie Merge (Zusammenführen).

7. Wählen Sie in der Pull-Anforderung zwischen den verfügbaren Mergestrategien. Merge-Strategien, die nicht angewendet werden können, werden abgedunkelt dargestellt. Wenn keine Merge-Strategien verfügbar sind, kannst du Konflikte manuell in der CodeCommit Konsole oder lokal mit deinem Git-Client lösen. Weitere Informationen finden Sie unter [Konflikte in einer Pull-Anfrage in einem AWS CodeCommit Repository lösen](#).

Merge pull request 9: Bug fix for unhandled exception

Merge request details

Pull request: #9 Bug fix for unhandled exception

Destination main **Source** bugfix-bug1234

Merge strategy Info
Determines the way in which the current pull request will be merged into the destination branch

☐ **Fast forward merge**
`git merge --ff-only`
Merges the branches and moves the destination branch pointer to the tip of the source branch. This is the default merge strategy in Git.


☒ **Squash and merge**
`git merge --squash`
Combine all commits from the source branch into a single merge commit in the destination branch.


☐ **3-way merge**
`git merge --no-ff`
Create a merge commit and adds individual source commits to the destination branch.


Commit message - optional Preview markdown

Squashed commit of the following

commit d49940ad
Author: Li Juan <li_juan@example.com>
Date: Tue May 07 2019 15:12:48 GMT-0700 (Pacific Daylight Time)

Fixing the bug reported in 1234.

Author name
Maria Garcia

Email address
maria_garcia@example.com

☒ Delete source branch bugfix-bug1234 after merging?

Cancel Merge pull request

- Bei einem Merge mit Vorlauf bewegt sich die Referenz für den Ziel-Branch vorwärts zum neuesten Commit des Quell-Branches. Dies ist das Standardverhalten von Git, soweit möglich. Es wird kein Merge-Commit erstellt, der gesamte Commit-Verlauf vom Quell-Branch bleibt

aber erhalten, als ob er im Ziel-Branch aufgetreten wäre. Zusammenführungen mit Vorlauf werden nicht als Branch-Zusammenführungen in der Commit-Ansicht des Verlaufs für den Ziel-Branch dargestellt, weil kein Merge-Commit erstellt wird. Die Spitze des Quell-Branches wird per Vorlauf an die Spitze des Ziel-Branches weitergeleitet.

- Bei einem Squash-Merge wird ein Commit mit den Änderungen im Quell-Branch erstellt und weist diesen einzelnen Squash-Commit dem Ziel-Branch zu. Standardmäßig enthält die Commit-Nachricht für diesen Squash-Commit alle Commit-Nachrichten der Änderungen im Quell-Branch. Individuelle Commit-Verläufe der Branch-Änderungen bleiben nicht erhalten. Dadurch lässt sich der Repository-Verlauf vereinfachen, während gleichzeitig eine grafische Darstellung der Zusammenführung in der Commit Visualizer-Ansicht der Ziel-Branch-Verlaufs beibehalten wird.
 - Ein Dreiwege-Merge erstellt einen Merge-Commit für die Zusammenführung im Ziel-Branch, behält als Teil des Verlaufs des Ziel-Branches aber auch die einzelnen Commits bei, die im Quell-Branch vorgenommen wurden. Dies kann bei der Wahrung eines vollständigen Verlaufs der Änderungen an Ihrem Repository hilfreich sein.
8. Wenn Sie als Strategie den Squash-Merge oder den Dreiwege-Merge wählen, überprüfen Sie die automatisch generierte Commit-Nachricht und ändern Sie sie, wenn Sie die Informationen ändern möchten. Fügen Sie den Namen und die E-Mail-Adresse für den Commit-Verlauf hinzu.
 9. (Optional) Deaktivieren Sie die Option, um den Quell-Branch im Rahmen der Zusammenführung zu löschen. Standardmäßig wird der Quell-Branch gelöscht, wenn eine Pull-Anforderung zusammengeführt wird.
 10. Wählen Sie Merge pull request (Pull-Anforderung zusammenführen) um die Zusammenführung abzuschließen.

Eine Pull-Anfrage zusammenführen (AWS CLI)

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter [Befehlszeilenreferenz](#).

Um das AWS CLI zum Zusammenführen von Pull-Requests in einem CodeCommit Repository zu verwenden

1. Um auszuwerten, ob eine Pull-Anforderung alle Genehmigungsregeln erfüllt und zusammengeführt werden kann, führen Sie den Befehl `evaluate-pull-request-approval-rules` aus, indem Sie Folgendes angeben:

- ID der Pull-Anforderung (mit der Option `--pull-request-id`).
- Revisions-ID der Pull-Anforderung (mit `--revision-id` option)). Sie können die aktuelle Revisions-ID für eine Pull-Anforderung mit dem Befehl [get-pull-request](#) abrufen.

Zum Beispiel, um den Status der Genehmigungsregeln für eine Pull-Anfrage mit einer ID von **27** und einer Revisions-ID von zu bewerten **9f29d167EXAMPLE**:

```
aws codecommit evaluate-pull-request-approval-rules --pull-request-id 27 --  
revision-id 9f29d167EXAMPLE
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{  
  "evaluation": {  
    "approved": false,  
    "approvalRulesNotSatisfied": [  
      "Require two approved approvers"  
    ],  
    "overridden": false,  
    "approvalRulesSatisfied": []  
  }  
}
```

Note

Diese Ausgabe gibt an, dass eine Pull-Anforderung nicht zusammengeführt werden kann, weil die Anforderungen einer Genehmigungsregel nicht erfüllt wurden. Damit diese Pull-Anforderung zusammengeführt werden kann, können Sie voraussetzen, dass Genehmiger sie genehmigen und so die Bedingungen der Regel erfüllen. Abhängig von Ihren Berechtigungen und der Art der Regelerstellung können Sie die Regel möglicherweise auch bearbeiten, überschreiben oder löschen. Weitere Informationen finden Sie unter [Überprüfen einer Pull-Anforderung](#), [Genehmigungsregeln für eine Pull-Anfrage überschreiben](#) und [Genehmigungsregel für einen Pull Request bearbeiten oder löschen](#).

2. Um eine Pull-Anforderung mittels der Mergestrategie mit Vorlauf zusammenzuführen und zu schließen, führen Sie den Befehl `merge-pull-request-by-fast-forward` aus, wobei Sie Folgendes angeben:

- ID der Pull-Anforderung (mit der Option `--pull-request-id`).
- Die vollständige Commit-ID des obersten Commits des Quell-Branches (mit der Option `--source-commit-id`).
- Der Name des Repositorys (mit der Option `--repository-name`).

Um beispielsweise eine Pull-Anfrage mit der ID von und der Quell-Commit-ID von `99132ab0EXAMPLE` in einem Repository mit dem Namen zusammenzuführen `47` und zu schließen `MyDemoRepo`:

```
aws codecommit merge-pull-request-by-fast-forward --pull-request-id 47 --source-commit-id 99132ab0EXAMPLE --repository-name MyDemoRepo
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
  "pullRequest": {
    "approvalRules": [
      {
        "approvalRuleContent": "{\n\"Version\": \"2018-11-08\", \"Statements\n\": [{\n\"Type\": \"Approvers\", \"NumberOfApprovalsNeeded\": 1, \"ApprovalPoolMembers\n\": [\n\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]\n}]}\",
        "approvalRuleId": "dd8b17fe-EXAMPLE",
        "approvalRuleName": "I want one approver for this pull request",
        "creationDate": 1571356106.936,
        "lastModifiedDate": 571356106.936,
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
        "ruleContentSha256": "4711b576EXAMPLE"
      }
    ],
    "authorArn": "arn:aws:iam::123456789012:user/Li_Juan",
    "clientRequestToken": "",
    "creationDate": 1508530823.142,
    "description": "Review the latest changes and updates to the global variables",
    "lastActivityDate": 1508887223.155,
    "pullRequestId": "47",
```

```
    "pullRequestStatus": "CLOSED",
    "pullRequestTargets": [
      {
        "destinationCommit": "9f31c968EXAMPLE",
        "destinationReference": "refs/heads/main",
        "mergeMetadata": {
          "isMerged": true,
          "mergedBy": "arn:aws:iam::123456789012:user/Mary_Major"
        },
        "repositoryName": "MyDemoRepo",
        "sourceCommit": "99132ab0EXAMPLE",
        "sourceReference": "refs/heads/variables-branch"
      }
    ],
    "title": "Consolidation of global variables"
  }
}
```

3. Um eine Pull-Anforderung mittels der Squashmerge-Strategie zusammenzuführen und zu schließen, führen Sie den Befehl `merge-pull-request-by-squash` aus, wobei Sie Folgendes angeben:
- ID der Pull-Anforderung (mit der Option `--pull-request-id`).
 - Die vollständige Commit-ID des obersten Commits des Quell-Branches (mit der Option `--source-commit-id`).
 - Der Name des Repositorys (mit der Option `--repository-name`).
 - Die Konflikt-Detailgenauigkeit, die Sie verwenden möchten (mit der Option `--conflict-detail-level`). Wenn dieser Parameter nicht angegeben wird, wird der Standardwert **FILE_LEVEL** verwendet.
 - Die Konfliktlösungsstrategie, die Sie verwenden möchten (mit der Option `--conflict-resolution-strategy`). Wenn dieser Parameter nicht angegeben wird, gilt der Standardwert **NONE**, und Konflikte müssen manuell behoben werden.
 - Die einzuschließende Commit-Nachricht (mit der Option `--commit-message`).
 - Der für den Commit zu verwendende Name (mit der Option `--author-name`).
 - Die für den Commit zu verwendende E-Mail-Adresse (mit der Option `--email`).
 - Angabe, ob leere Ordner beibehalten werden sollen (mit der Option `--keep-empty-folders`).

Im folgenden Beispiel wird eine Pull-Anfrage mit der ID von **47** und der Quell-Commit-ID von **99132ab0EXAMPLE** in einem Repository mit dem Namen **MyDemoRepo** zusammengeführt und geschlossen. Es verwendet das Konfliktdetail **LINE_LEVEL** und die Konfliktlösungsstrategie **ACCEPT_SOURCE**:

```
aws codecommit merge-pull-request-by-squash --pull-request-id 47 --source-commit-id 99132ab0EXAMPLE --repository-name MyDemoRepo --conflict-detail-level LINE_LEVEL --conflict-resolution-strategy ACCEPT_SOURCE --author-name "Jorge Souza" --email "jorge_souza@example.com" --commit-message "Merging pull request 47 by squash and accepting source in merge conflicts"
```

Ist der Befehl erfolgreich, erstellt er die gleiche Art von Ausgabe wie ein Merge mit Vorlauf, vergleichbar mit der folgenden Ausgabe:

```
{
  "pullRequest": {
    "approvalRules": [
      {
        "approvalRuleContent": "{\"Version\": \"2018-11-08\", \"DestinationReferences\": [\"refs/heads/main\"], \"Statements\": [{\"Type\": \"Approvers\", \"NumberOfApprovalsNeeded\": 2, \"ApprovalPoolMembers\": [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}\",
        "approvalRuleId": "dd8b17fe-EXAMPLE",
        "approvalRuleName": "2-approver-rule-for-main",
        "creationDate": 1571356106.936,
        "lastModifiedDate": 571356106.936,
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
        "originApprovalRuleTemplate": {
          "approvalRuleTemplateId": "dd8b17fe-EXAMPLE",
          "approvalRuleTemplateName": "2-approver-rule-for-main"
        },
        "ruleContentSha256": "4711b576EXAMPLE"
      }
    ],
    "authorArn": "arn:aws:iam::123456789012:user/Li_Juan",
    "clientRequestToken": "",
    "creationDate": 1508530823.142,
    "description": "Review the latest changes and updates to the global variables",
    "lastActivityDate": 1508887223.155,
```

```
{
  "pullRequestId": "47",
  "pullRequestStatus": "CLOSED",
  "pullRequestTargets": [
    {
      "destinationCommit": "9f31c968EXAMPLE",
      "destinationReference": "refs/heads/main",
      "mergeMetadata": {
        "isMerged": true,
        "mergedBy": "arn:aws:iam::123456789012:user/Mary_Major"
      },
      "repositoryName": "MyDemoRepo",
      "sourceCommit": "99132ab0EXAMPLE",
      "sourceReference": "refs/heads/variables-branch"
    }
  ],
  "title": "Consolidation of global variables"
}
```

4. Um eine Pull-Anforderung mittels der Dreiwege-Mergestrategie zusammenzuführen und zu schließen, führen Sie den Befehl `merge-pull-request-by-three-way` aus und geben Sie Folgendes an:

- ID der Pull-Anforderung (mit der Option `--pull-request-id`).
- Die vollständige Commit-ID des obersten Commits des Quell-Branches (mit der Option `--source-commit-id`).
- Der Name des Repositorys (mit der Option `--repository-name`).
- Die Konflikt-Detailgenauigkeit, die Sie verwenden möchten (mit der Option `--conflict-detail-level`). Wenn dieser Parameter nicht angegeben wird, wird der Standardwert **FILE_LEVEL** verwendet.
- Die Konfliktlösungsstrategie, die Sie verwenden möchten (mit der Option `--conflict-resolution-strategy`). Wenn dieser Parameter nicht angegeben wird, gilt der Standardwert **NONE**, und Konflikte müssen manuell behoben werden.
- Die einzuschließende Commit-Nachricht (mit der Option `--commit-message`).
- Der für den Commit zu verwendende Name (mit der Option `--author-name`).
- Die für den Commit zu verwendende E-Mail-Adresse (mit der Option `--email`).
- Angabe, ob leere Ordner beibehalten werden sollen (mit der Option `--keep-empty-folders`).

Im folgenden Beispiel wird eine Pull-Anfrage mit der ID von **47** und der Quell-Commit-ID von **99132ab0EXAMPLE** in einem Repository mit dem Namen zusammengeführt und geschlossen. **MyDemoRepo** Es verwendet die Standardoptionen für Konfliktdetail und Konfliktlösungsstrategie:

```
aws codecommit merge-pull-request-by-three-way --pull-request-id 47 --source-
commit-id 99132ab0EXAMPLE --repository-name MyDemoRepo --author-name "Maria Garcia"
--email "maria_garcia@example.com" --commit-message "Merging pull request 47 by
three-way with default options"
```

Ist der Befehl erfolgreich, erstellt er die gleiche Art von Ausgabe wie ein Merge mit Vorlauf, vergleichbar mit der folgenden Ausgabe:

```
{
  "pullRequest": {
    "approvalRules": [
      {
        "approvalRuleContent": "{\"Version\": \"2018-11-08\",
        \"DestinationReferences\": [\"refs/heads/main\"], \"Statements\": [{\"Type
        \": \"Approvers\", \"NumberOfApprovalsNeeded\": 2, \"ApprovalPoolMembers\":
        [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
        "approvalRuleId": "dd8b17fe-EXAMPLE",
        "approvalRuleName": "2-approver-rule-for-main",
        "creationDate": 1571356106.936,
        "lastModifiedDate": 571356106.936,
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
        "originApprovalRuleTemplate": {
          "approvalRuleTemplateId": "dd8b17fe-EXAMPLE",
          "approvalRuleTemplateName": "2-approver-rule-for-main"
        },
        "ruleContentSha256": "4711b576EXAMPLE"
      }
    ],
    "authorArn": "arn:aws:iam::123456789012:user/Li_Juan",
    "clientRequestToken": "",
    "creationDate": 1508530823.142,
    "description": "Review the latest changes and updates to the global
    variables",
    "lastActivityDate": 1508887223.155,
    "pullRequestId": "47",
    "pullRequestStatus": "CLOSED",
```

```
    "pullRequestTargets": [
      {
        "destinationCommit": "9f31c968EXAMPLE",
        "destinationReference": "refs/heads/main",
        "mergeMetadata": {
          "isMerged": true,
          "mergedBy": "arn:aws:iam::123456789012:user/Mary_Major"
        },
        "repositoryName": "MyDemoRepo",
        "sourceCommit": "99132ab0EXAMPLE",
        "sourceReference": "refs/heads/variables-branch"
      }
    ],
    "title": "Consolidation of global variables"
  }
}
```

Konflikte in einer Pull-Anfrage in einem AWS CodeCommit Repository lösen

Wenn Ihre Pull-Anforderung Konflikte hat und nicht zusammengeführt werden kann, können Sie versuchen, die Konflikte auf eine von mehreren Arten zu lösen:

- Auf Ihrem lokalen Computer können Sie mit dem Befehl `git diff` nach Konflikten zwischen den beiden Branches suchen und Änderungen vornehmen, um sie zu lösen. Sie können auch ein anderes Tool oder ein anderes Programm verwenden, um sie beim Suchen und Beheben von Unterschieden zu unterstützen. Sobald Sie sie zu Ihrer Zufriedenheit gelöst haben, können Sie Ihren Quell-Branch mit den Änderungen, die die gelösten Konflikte enthalten, weiterleiten, wodurch der Pull-Request aktualisiert wird. Weitere Informationen über `git diff` und `git difftool` finden Sie in der Git-Dokumentation.
- In der Konsole können Sie **Resolve conflicts** (Konflikte lösen) wählen. Daraufhin wird ein Texteditor geöffnet, der die Konflikte auf ähnliche Weise wie der Befehl `git diff` anzeigt. Sie können die Konflikte manuell in jeder Datei, in der sie enthalten sind, überprüfen, Änderungen vornehmen und die Pull-Anforderung dann mit Ihren Änderungen aktualisieren.
- In der AWS CLI können Sie Informationen über Merge-Konflikte abrufen und einen Merge-Commit ohne Referenz erstellen, um eine Zusammenführung zu testen.

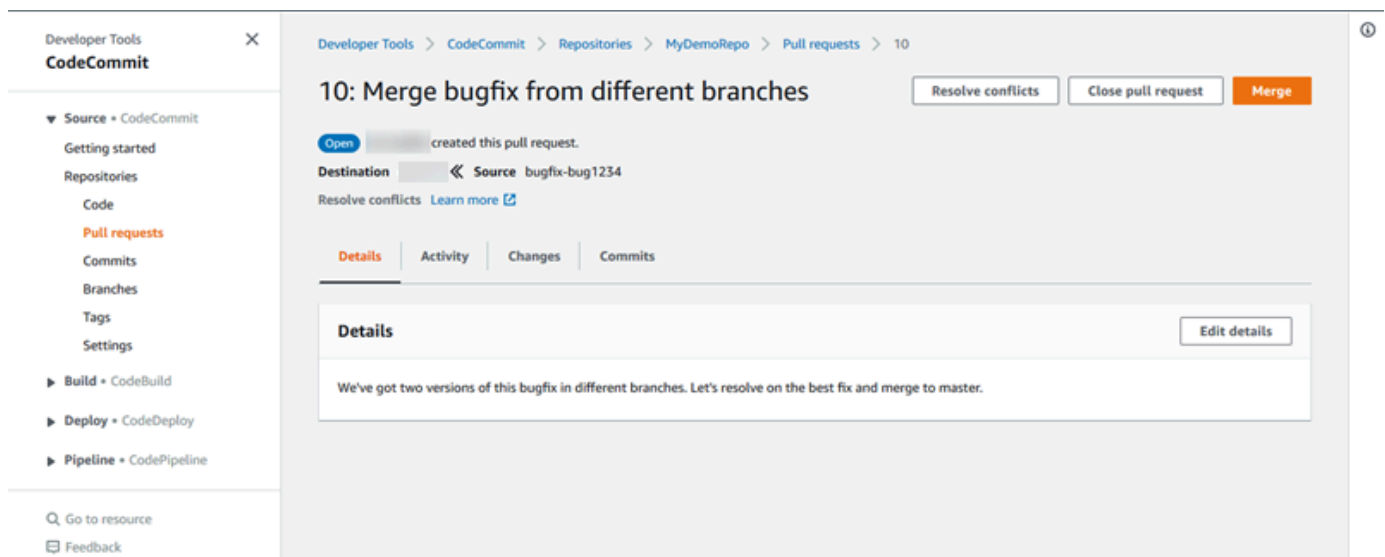
Themen

- [Konflikte in einer Pull-Anfrage lösen \(Konsole\)](#)
- [Konflikte in einer Pull-Anfrage lösen \(AWS CLI\)](#)

Konflikte in einer Pull-Anfrage lösen (Konsole)

Sie können die CodeCommit Konsole verwenden, um Konflikte in einer Pull-Anfrage in einem CodeCommit Repository zu lösen.

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie im Bereich Repositories (Repositorys) den Namen des Repositorys aus.
3. Wählen Sie im Navigationsbereich Pull requests (Pull-Anforderungen) aus.
4. Standardmäßig wird eine Liste aller offenen Pull-Anforderungen angezeigt. Wählen Sie die offene Pull-Anforderung aus, die Sie zusammenführen möchten, die aber Konflikte enthält.
5. Wählen Sie in der Pull-Anforderung Resolve conflicts (Konflikte lösen). Diese Option wird nur angezeigt, wenn Konflikte vorhanden sind, die behoben werden müssen, bevor die Pull-Anforderung zusammengeführt werden kann.



6. In dem nun geöffneten Konfliktlösungsfenster werden die einzelnen Dateien mit Konflikten aufgeführt, die behoben werden müssen. Wählen Sie die einzelnen Dateien in der Liste aus, um die Konflikte zu überprüfen. Nehmen Sie alle erforderlichen Änderungen vor, bis alle Konflikte behoben wurden.

Developer Tools > CodeCommit > Repositories > MyDemoRepo > Pull requests > 10 > Resolve conflicts

Resolve conflicts

Resolve conflicts in each of the files in the list. When you have resolved all conflicts, update the pull request and review the merge strategies available. [Info](#)

Destination << Source bugfix-bug1234

Editing: helloworld.py

Reset file

Delete file

Use source content

Use destination content

⊗ helloworld.py

1

```
1 import sys
2
3 print('Hello, World!')
4
5 <<<<<< HEAD:helloworld.py
6 print('The sum of 2 and 3 is 5.')
7 =====
8 print('The sum of 3 and 2 is 5.')
9 >>>>>> bugfix-bug1234:helloworld.py
10
11 sum = int(sys.argv[1]) + int(sys.argv[2])
12
13 print('The sum of {0} and {1} is {2}.'.format(sys.argv[1], sys.argv[2], sum))
```

☐ Allow the merge to proceed with Git conflict markers still present.

Cancel

Update pull request

- Sie können wählen, ob Sie den Inhalt der Quelldatei oder der Zieldatei verwenden oder ob Sie bei einer anderen Datei als einer binären Datei ihren Inhalt manuell so bearbeiten möchten, dass sie nur die von Ihnen gewünschten Änderungen enthält. Standard-Git-Diff-Marker werden zum Aufzeigen der Konflikte zwischen den Ziel(HEAD)- und Quell-Banches in der Datei verwendet.
- Wenn es sich bei der Datei um eine binäre Datei oder ein Git-Untermodule handelt oder wenn ein Konflikt bei den Datei-/Ordernamen vorliegt, müssen Sie entscheiden, ob die Quelldatei oder die Zieldatei zum Lösen der Konflikte verwendet werden soll. Sie können Binärdateien in der Konsole nicht anzeigen oder bearbeiten. CodeCommit

- Bei Dateimoduskonflikten wird die Option angezeigt, diesen Konflikt zu lösen, indem Sie zwischen dem Dateimodus der Quelldatei und dem Dateimodus der Zieldatei wählen.
 - Wenn Sie Ihre Änderungen an einer Datei verwerfen und deren konfigurierten Zustand wiederherstellen möchten, wählen Sie Reset file (Datei zurücksetzen). Dies ermöglicht Ihnen, die Konflikte auf andere Weise zu lösen.
7. Wenn Sie mit Ihren Änderungen zufrieden sind, wählen Sie Update pull request (Pull-Anforderung aktualisieren).

 Note

Sie müssen alle Konflikte in allen Dateien lösen, bevor Sie die Pull-Anforderung erfolgreich mit Ihren Änderungen aktualisieren können.

8. Die Pull-Anforderung wird mit Ihren Änderungen aktualisiert und kann zusammengeführt werden. Sie sehen die Seite zum Zusammenführen. Sie können bestimmen, dass die Pull-Anforderung zu diesem Zeitpunkt zusammengeführt werden soll, oder zur Liste der Pull-Anforderungen zurückkehren.

Konflikte in einer Pull-Anfrage lösen (AWS CLI)

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter [Befehlszeilenreferenz](#).

Kein einziger AWS CLI Befehl ermöglicht es Ihnen, Konflikte in einer Pull-Anfrage zu lösen und diese Anfrage zusammenzuführen. Sie können mit einzelnen Befehlen jedoch Konflikte aufdecken und versuchen, sie zu lösen, und testen, ob eine Pull-Anforderung zusammengeführt werden kann. Sie können Folgendes verwenden:

- `get-merge-options`, um herauszufinden, welche Mergeoptionen für eine Zusammenführung zwischen zwei Commit-Spezifizierern verfügbar sind.
- `get-merge-conflicts`, um eine Liste von Dateien mit Konflikten beim Zusammenführen zwischen zwei Commit-Spezifizierern abzurufen.
- `batch-describe-merge-conflicts`, um Informationen über alle Zusammenführungskonflikte in Dateien in einer Zusammenführung zwischen zwei Commits mithilfe einer bestimmten Merge-Strategie abzurufen.

- `describe-merge-conflicts`, um detaillierte Informationen über Zusammenführungskonflikte für eine bestimmte Datei zwischen zwei Commits mit einer bestimmten Strategie zu erhalten.
- `create-unreferenced-merge-commit`, um das Ergebnis des Zusammenführens von zwei Commit-Spezifizierern mit einer bestimmten Strategie zu testen.

1.

Um herauszufinden, welche Mergeoptionen für eine Zusammenführung zwischen zwei Commit-Spezifizierern verfügbar sind, führen Sie den Befehl `get-merge-options` aus und geben Sie Folgendes an:

- Einen Commit-Spezifizierer für die Quelle der Zusammenführung (mit der Option `--source-commit-specifier`).
- Einen Commit-Spezifizierer für das Ziel der Zusammenführung (mit der Option `--destination-commit-specifier`).
- Der Name des Repositorys (mit der Option `--repository-name`).
- (Optional) Eine Konfliktlösungsstrategie, die verwendet werden soll (mit der Option `--conflict-resolution-strategy`).
- (Optional) Die gewünschte Detailgenauigkeit aller Konflikte (mit der Option `--conflict-detail-level`).

Um beispielsweise zu ermitteln, welche Merge-Strategien für das Zusammenführen eines Quell-Banches *bugfix-1234* mit einem Ziel-Branch *main* in einem Repository mit dem Namen: *MyDemoRepo*

```
aws codecommit get-merge-options --source-commit-specifier bugfix-1234 --  
destination-commit-specifier main --repository-name MyDemoRepo
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{  
  "mergeOptions": [  
    "FAST_FORWARD_MERGE",  
    "SQUASH_MERGE",  
    "THREE_WAY_MERGE"  
  ],  
  "sourceCommitId": "d49940adEXAMPLE",  
  "destinationCommitId": "86958e0aEXAMPLE",  
}
```



```
"baseCommitId": "86958e0aEXAMPLE"
}
```

2.

Um eine Liste der Dateien mit Zusammenführungskonflikten in einer Zusammenführung zwischen zwei Commit-Spezifizierern zu erhalten, führen Sie den Befehl `get-merge-conflicts` aus und geben Sie Folgendes an:

- Einen Commit-Spezifizierer für die Quelle der Zusammenführung (mit der Option `--source-commit-specifier`).
- Einen Commit-Spezifizierer für das Ziel der Zusammenführung (mit der Option `--destination-commit-specifier`).
- Der Name des Repositorys (mit der Option `--repository-name`).
- Die Zusammenführungsoption, die Sie verwenden möchten (mit der Option `--merge-option`).
- (Optional) Die gewünschte Detailgenauigkeit aller Konflikte (mit der Option `--conflict-detail-level`).
- (Optional) Eine Konfliktlösungsstrategie, die verwendet werden soll (mit der Option `--conflict-resolution-strategy`).
- (Optional) Die maximale Anzahl von Dateien mit Konflikten, die zurückgegeben werden soll (mit der Option `--max-conflict-files`).

Um beispielsweise eine Liste von Dateien abzurufen, die Konflikte bei einer Zusammenführung zwischen einem Quell-Branch namens `feature-randomizationfeature` und einem Ziel-Branch namens `main` enthalten, verwenden Sie die Drei-Wege-Zusammenführungsstrategie in einem Repository mit dem Namen: `MyDemoRepo`

```
aws codecommit get-merge-conflicts --source-commit-specifier feature-
randomizationfeature --destination-commit-specifier main --merge-option
THREE_WAY_MERGE --repository-name MyDemoRepo
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
  "mergeable": false,
  "destinationCommitId": "86958e0aEXAMPLE",
  "sourceCommitId": "6ccd57fdEXAMPLE",
  "baseCommitId": "767b6958EXAMPLE",
  "conflictMetadataList": [
```

```
{
  "filePath": "readme.md",
  "fileSizes": {
    "source": 139,
    "destination": 230,
    "base": 85
  },
  "fileModes": {
    "source": "NORMAL",
    "destination": "NORMAL",
    "base": "NORMAL"
  },
  "objectTypes": {
    "source": "FILE",
    "destination": "FILE",
    "base": "FILE"
  },
  "numberOfConflicts": 1,
  "isBinaryFile": {
    "source": false,
    "destination": false,
    "base": false
  },
  "contentConflict": true,
  "fileModeConflict": false,
  "objectTypeConflict": false,
  "mergeOperations": {
    "source": "M",
    "destination": "M"
  }
}
```

3.

Um Informationen über Zusammenführungskonflikte in allen Dateien und einem Teilsatz von Dateien in einer Zusammenführung zwischen zwei Commit-Spezifizierern zu erhalten, führen Sie den Befehl `batch-describe-merge-conflicts` aus und geben Sie Folgendes an:

- Einen Commit-Spezifizierer für die Quelle der Zusammenführung (mit der Option `--source-commit-specifier`).
- Einen Commit-Spezifizierer für das Ziel der Zusammenführung (mit der Option `--destination-commit-specifier`).

- Die Zusammenführungsoption, die Sie verwenden möchten (mit der Option `--merge-option`).
- Der Name des Repositorys (mit der Option `--repository-name`).
- (Optional) Eine Konfliktlösungsstrategie, die verwendet werden soll (mit der Option `--conflict-resolution-strategy`).
- (Optional) Die gewünschte Detailgenauigkeit aller Konflikte (mit der Option `--conflict-detail-level`).
- (Optional) Die maximale Anzahl von Zusammenführungsstücken, die zurückgegeben werden soll (mit der Option `--max-merge-hunks`).
- (Optional) Die maximale Anzahl von Dateien mit Konflikten, die zurückgegeben werden soll (mit der Option `--max-conflict-files`).
- (Optional) Der Pfad der Ziel-Dateien, der zur Beschreibung der Konflikte verwendet werden soll (mit der Option `--file-paths`).

Um beispielsweise die Zusammenführungskonflikte beim Zusammenführen eines Quellzweigs mit einem Zielzweig *feature-randomizationfeature* mit dem Namen zu ermitteln, *main* verwenden Sie die Strategie in einem Repository mit dem Namen: *THREE_WAY_MERGE MyDemoRepo*

```
aws codecommit batch-describe-merge-conflicts --source-commit-specifier feature-randomizationfeature --destination-commit-specifier main --merge-option THREE_WAY_MERGE --repository-name MyDemoRepo
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
  "conflicts": [
    {
      "conflictMetadata": {
        "filePath": "readme.md",
        "fileSizes": {
          "source": 139,
          "destination": 230,
          "base": 85
        },
        "fileModes": {
          "source": "NORMAL",
          "destination": "NORMAL",
```

```

        "base": "NORMAL"
    },
    "objectTypes": {
        "source": "FILE",
        "destination": "FILE",
        "base": "FILE"
    },
    "numberOfConflicts": 1,
    "isBinaryFile": {
        "source": false,
        "destination": false,
        "base": false
    },
    "contentConflict": true,
    "fileModeConflict": false,
    "objectTypeConflict": false,
    "mergeOperations": {
        "source": "M",
        "destination": "M"
    }
},
"mergeHunks": [
    {
        "isConflict": true,
        "source": {
            "startLine": 0,
            "endLine": 3,
            "hunkContent": "VGhpcyBpEXAMPLE=="
        },
        "destination": {
            "startLine": 0,
            "endLine": 1,
            "hunkContent": "VXNlIHRoEXAMPLE="
        }
    }
]
},
"errors": [],
"destinationCommitId": "86958e0aEXAMPLE",
"sourceCommitId": "6ccd57fdEXAMPLE",
"baseCommitId": "767b6958EXAMPLE"
}

```

4.

Um detaillierte Informationen über Zusammenführungskonflikte für eine bestimmte Datei in einer Zusammenführung zwischen zwei Commit-Spezifizierern zu erhalten, führen Sie den Befehl `describe-merge-conflicts` aus und geben Sie Folgendes an:

- Einen Commit-Spezifizierer für die Quelle der Zusammenführung (mit der Option `--source-commit-specifier`).
- Einen Commit-Spezifizierer für das Ziel der Zusammenführung (mit der Option `--destination-commit-specifier`).
- Die Zusammenführungsoption, die Sie verwenden möchten (mit der Option `--merge-option`).
- Der Pfad der Zieldatei, die zur Beschreibung der Konflikte verwendet werden soll (mit der Option `--file-path`).
- Der Name des Repositorys (mit der Option `--repository-name`).
- (Optional) Eine Konfliktlösungsstrategie, die verwendet werden soll (mit der Option `--conflict-resolution-strategy`).
- (Optional) Die gewünschte Detailgenauigkeit aller Konflikte (mit der Option `--conflict-detail-level`).
- (Optional) Die maximale Anzahl von Zusammenführungsstücken, die zurückgegeben werden soll (mit der Option `--max-merge-hunks`).
- (Optional) Die maximale Anzahl von Dateien mit Konflikten, die zurückgegeben werden soll (mit der Option `--max-conflict-files`).

Um beispielsweise die Zusammenführungskonflikte für eine Datei zu ermitteln, die `readme.md` in einem Quell-Branch `feature-randomizationfeature` mit einem Ziel-Branch benannt ist, `main` mit der `THREE_WAY_MERGE` Strategie in einem Repository namens `MyDemoRepo`:

```
aws codecommit describe-merge-conflicts --source-commit-specifier feature-randomizationfeature --destination-commit-specifier main --merge-option THREE_WAY_MERGE --file-path readme.md --repository-name MyDemoRepo
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
  "conflictMetadata": {
    "filePath": "readme.md",
    "fileSizes": {
```

```
        "source": 139,
        "destination": 230,
        "base": 85
    },
    "fileModes": {
        "source": "NORMAL",
        "destination": "NORMAL",
        "base": "NORMAL"
    },
    "objectTypes": {
        "source": "FILE",
        "destination": "FILE",
        "base": "FILE"
    },
    "numberOfConflicts": 1,
    "isBinaryFile": {
        "source": false,
        "destination": false,
        "base": false
    },
    "contentConflict": true,
    "fileModeConflict": false,
    "objectTypeConflict": false,
    "mergeOperations": {
        "source": "M",
        "destination": "M"
    }
},
"mergeHunks": [
    {
        "isConflict": true,
        "source": {
            "startLine": 0,
            "endLine": 3,
            "hunkContent": "VGhpcyBpEXAMPLE=="
        },
        "destination": {
            "startLine": 0,
            "endLine": 1,
            "hunkContent": "VXNlIHRoEXAMPLE="
        }
    }
],
"destinationCommitId": "86958e0aEXAMPLE",
```

```
"sourceCommitId": "6ccd57fdEXAMPLE",  
"baseCommitId": "767b69580EXAMPLE"  
}
```

5.

Um einen uneferenzierten Commit zu erstellen, der das Ergebnis des Zusammenführens von zwei Commit-Spezifizierern repräsentiert, führen Sie den Befehl `create-unreferenced-merge-commit` aus und geben Sie Folgendes an:

- Einen Commit-Spezifizierer für die Quelle der Zusammenführung (mit der Option `--source-commit-specifier`).
- Einen Commit-Spezifizierer für das Ziel der Zusammenführung (mit der Option `--destination-commit-specifier`).
- Die Zusammenführungsoption, die Sie verwenden möchten (mit der Option `--merge-option`).
- Der Name des Repositorys (mit der Option `--repository-name`).
- (Optional) Eine Konfliktlösungsstrategie, die verwendet werden soll (mit der Option `--conflict-resolution-strategy`).
- (Optional) Die gewünschte Detailgenauigkeit aller Konflikte (mit der Option `--conflict-detail-level`).
- (Optional) Die einzuschließende Commit-Nachricht (mit der Option `--commit-message`).
- (Optional) Der für den Commit zu verwendende Name (mit der Option `--name`).
- (Optional) Die für den Commit zu verwendende E-Mail-Adresse (mit der Option `--email`).
- (Optional) Angabe, ob leere Ordner beibehalten werden sollen (mit der Option `--keep-empty-folders`).

Um beispielsweise die Zusammenführungskonflikte beim Zusammenführen eines Quellzweigs *bugfix-1234* mit einem Zielzweig mit dem Namen `ACCEPT_SOURCE` in einem Repository *main* mit dem Namen zu ermitteln: *MyDemoRepo*

```
aws codecommit create-unreferenced-merge-commit --source-commit-  
specifier bugfix-1234 --destination-commit-specifier main --merge-option  
THREE_WAY_MERGE --repository-name MyDemoRepo --name "Maria Garcia" --email  
"maria_garcia@example.com" --commit-message "Testing the results of this merge."
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
```

```
"commitId": "4f178133EXAMPLE",  
"treeId": "389765daEXAMPLE"  
}
```

Einen Pull-Request in einem AWS CodeCommit Repository schließen

Wenn Sie eine Pull-Anforderung schließen möchten, ohne den Code zusammenzuführen, stehen Ihnen dazu mehrere Optionen zur Verfügung:

- In der Konsole können Sie eine Pull-Anforderung schließen, ohne den Code zusammenzuführen. Dies ist sinnvoll, wenn Sie Branches manuell mithilfe des Befehls `git merge` zusammenführen möchten oder wenn der Code im Quell-Branch der Pull-Anforderung nicht der Code ist, der mit dem Ziel-Branch zusammengeführt werden soll.
- Sie können den in der Pull-Anfrage angegebenen Quell-Branch löschen. CodeCommit schließt einen Pull-Request automatisch, wenn entweder der Quell- oder der Ziel-Branch des Pull-Requests gelöscht wird.
- In der AWS CLI kannst du den Status einer Pull-Anfrage von `OPEN` bis aktualisieren `CLOSED`. Damit wird die Pull-Anforderung geschlossen, ohne den Code zusammenzuführen.

Themen

- [Schließt eine Pull-Anfrage \(Konsole\)](#)
- [Schließt eine Pull-Anfrage \(AWS CLI\)](#)

Schließt eine Pull-Anfrage (Konsole)

Du kannst die CodeCommit Konsole verwenden, um eine Pull-Anfrage in einem CodeCommit Repository zu schließen. Nachdem der Status einer Pull-Anforderung in `Closed` geändert wurde, kann er nicht wieder zu `Open` wechseln. Die Benutzer können jedoch weiterhin Änderungen kommentieren und auf Kommentare antworten.

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie im Bereich Repositories (Repositorys) den Namen des Repositorys aus.
3. Wählen Sie im Navigationsbereich Pull requests (Pull-Anforderungen) aus.

4. Standardmäßig wird eine Liste aller offenen Pull-Anforderungen angezeigt. Wählen Sie die offene Pull-Anforderung aus, die Sie schließen möchten.

Pull request	Author	Destination	Last activity	Status	Approval status
31: testing this	Saanvi_Sarkar	preprod	4 minutes ago	Open	No approval rules
25: Updated some of our Java samples	Li_Juan	main	5 minutes ago	Open	0 of 1 rules satisfied
29: Changing duplicate value	Li_Juan	main	19 minutes ago	Open	0 of 1 rules satisfied
22: Test pull request	Saanvi_Sarkar	preprod	20 minutes ago	Open	No approval rules
28: Changes to some of our code samples	Li_Juan	main	1 month ago	Open	0 of 1 rules satisfied
20: A bugfix to add missing examples for S3	Saanvi_Sarkar	main	1 month ago	Open	0 of 1 rules satisfied

5. Wählen Sie in der Pull-Anforderung Close pull request (Pull-Anforderung schließen). Diese Option schließt die Pull-Anforderung ohne zu versuchen, den Quell-Branch mit dem Ziel-Branch zusammenzuführen. Diese Option bietet keine Möglichkeit, den Quell-Branch beim Schließen der Pull-Anforderung zu löschen. Sie können diesen Vorgang aber selbst ausführen, sobald die Anforderung geschlossen ist.

Schließt eine Pull-Anfrage (AWS CLI)

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter [Befehlszeilenreferenz](#).

Um den zu verwenden AWS CLI , um Pull-Requests in einem CodeCommit Repository zu schließen

- Um den Status einer Pull-Anforderung in einem Repository von OPEN zu CLOSED zu ändern, führen Sie den Befehl `update-pull-request-status` aus, wobei Sie Folgendes angeben:
 - ID der Pull-Anforderung (mit der Option `--pull-request-id`).
 - Der Status der Pull-Anforderung (mit der Option `--pull-request-status`).

Um beispielsweise den Status einer Pull-Anfrage mit der ID von auf den Status **CLOSED** in einem CodeCommit Repository mit dem Namen **42** zu aktualisierenMyDemoRepo:

```
aws codecommit update-pull-request-status --pull-request-id 42 --pull-request-status CLOSED
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```

{
  "pullRequest": {
    "approvalRules": [
      {
        "approvalRuleContent": "{\\"Version\\": \\"2018-11-08\\",\\"Statements\\": [{\\"Type\\": \\"Approvers\\",\\"NumberOfApprovalsNeeded\\": 2,\\"ApprovalPoolMembers\\": [\\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\\"]]}}",
        "approvalRuleId": "dd8b17fe-EXAMPLE",
        "approvalRuleName": "2-approvers-needed-for-this-change",
        "creationDate": 1571356106.936,
        "lastModifiedDate": 571356106.936,
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
        "ruleContentSha256": "4711b576EXAMPLE"
      }
    ],
    "authorArn": "arn:aws:iam::123456789012:user/Li_Juan",
    "clientRequestToken": "",
    "creationDate": 1508530823.165,
    "description": "Updated the pull request to remove unused global variable.",
    "lastActivityDate": 1508372423.12,
    "pullRequestId": "47",
    "pullRequestStatus": "CLOSED",
    "pullRequestTargets": [
      {
        "destinationCommit": "9f31c968EXAMPLE",
        "destinationReference": "refs/heads/main",
        "mergeMetadata": {
          "isMerged": false,
        },
        "repositoryName": "MyDemoRepo",
        "sourceCommit": "99132ab0EXAMPLE",
        "sourceReference": "refs/heads/variables-branch"
      }
    ],
    "title": "Consolidation of global variables"
  }
}

```

Arbeiten mit Genehmigungsregelvorlagen

Sie können Genehmigungsregeln für Pull-Anforderungen erstellen. Verwenden Sie Vorlagen für Genehmigungsregeln, um Genehmigungsregeln automatisch auf einige oder alle in Repositorys erstellten Pull Requests anzuwenden. Genehmigungsregelvorlagen erleichtern das Anpassen von Entwicklungs-Workflows über Repositorys hinweg, sodass für verschiedene Branches ausreichende Genehmigungen und Kontrollen vorgesehen sind. Sie können verschiedene Regeln für Produktions- und Entwicklungs-Banches definieren. Diese Regeln werden angewendet, wenn eine Pull-Anforderung erstellt wird, die den Regelbedingungen entspricht. Weitere Informationen zu verwalteten Richtlinien und Berechtigungen für Genehmigungsregelvorlagen finden Sie unter [Berechtigungen für Aktionen auf Vorlagen für Genehmigungsregeln](#) und [AWS verwaltete Richtlinien für CodeCommit](#).

Sie können eine Vorlage für Genehmigungsregeln einem oder mehreren Repositorys dort zuordnen AWS-Region , wo sie erstellt wurden. Wenn eine Vorlage einem Repository zugeordnet ist, erstellt sie beim Erstellen von Pull-Anforderungen automatisch Genehmigungsregeln für Pull-Anforderungen in diesem Repository. Genau wie eine Genehmigungsregel definiert eine Genehmigungsregelvorlage eine Genehmigungsregelstruktur, einschließlich der Anzahl der erforderlichen Genehmigungen und eines optionalen Pools der Benutzer, von denen die Genehmigungen stammen müssen. Im Unterschied zu einer Genehmigungsregel können Sie außerdem Zielverweise (Branches) definieren, die auch als Branch-Filter bezeichnet werden. Wenn Sie Zielverweise definieren, werden nur für Pull-Anforderungen, deren Branch-Zielnamen den angegebenen Branch-Namen (Zielverweise) in der Vorlage entsprechen, Regeln erstellt. Wenn Sie z. B. **refs/heads/main** als Zielverweis angeben, wird die in der Vorlage definierte Genehmigungsregel nur auf Pull-Anforderungen angewendet, wenn der Ziel-Branch **main** ist.

Approval rule template

Approval rule template name

Require 1 approver from a senior developer for main branch

Description - *optional*

Before a pull request can be merged to the main branch, at least one senior developer must give an approval to the changes.

Number of approvals needed

1

Approval pool members - *optional*

If approval pool members are specified, only approvals from these members will count toward satisfying this rule. You can use wildcards to match multiple approvers with one value.

Approver type [Info](#)

Value

IAM user name or assumed role ▼

SeniorDevelopers/*

Remove

Fully qualified ARN ▼

:s:iam::123456789012:user/Mary_Major

Remove

Add

Branch filters - *optional*

Use branch filters to only apply this template to a pull request if the destination branch name matches a name in the filter list.

Branch name

main

Remove

Add

▼ Associated repositories

Repositories - *optional*

MyDemoRepo ✕

MyTestRepo ✕

Themen

- [Erstellen einer Genehmigungsregelvorlage](#)
- [Ordnen Sie eine Vorlage für Genehmigungsregeln einem Repository zu](#)
- [Verwalten von Genehmigungsregelvorlagen](#)
- [Trennen Sie die Zuordnung einer Vorlage für eine Genehmigungsregel](#)
- [Löschen Sie eine Vorlage für Genehmigungsregeln](#)

Erstellen einer Genehmigungsregelvorlage

Sie können einzelne oder mehrere Genehmigungsregelvorlagen erstellen, um Entwicklungs-Workflows über Repositorys hinweg anzupassen. Durch das Erstellen mehrerer Vorlagen können Sie die automatische Erstellung von Genehmigungsregeln so gestalten, dass für unterschiedliche Branches die jeweils erforderlichen Genehmigungen vorausgesetzt und Kontrollen bereitgestellt werden. Sie können beispielsweise unterschiedliche Vorlagen für Produktions- und Entwicklungs-Banches erstellen und diese Vorlagen einzelnen oder mehreren Repositorys zuweisen. Wenn Benutzer Pull-Anforderungen in diesen Repositorys erstellen, wird die Anforderung anhand dieser Vorlagen ausgewertet. Wenn die Anforderung den Bedingungen in den zugewiesenen Vorlagen entspricht, werden Genehmigungsregeln für die Pull-Anforderung erstellt.

Sie können die Konsole verwenden oder Vorlagen AWS CLI für Genehmigungsregeln erstellen. Weitere Informationen zu verwalteten Richtlinien und Berechtigungen für Genehmigungsregelvorlagen finden Sie unter [Berechtigungen für Aktionen auf Vorlagen für Genehmigungsregeln](#) und [AWS verwaltete Richtlinien für CodeCommit](#).

Themen

- [Erstellen Sie eine Vorlage für Genehmigungsregeln \(Konsole\)](#)
- [Erstellen Sie eine Vorlage für Genehmigungsregeln \(AWS CLI\)](#)

Erstellen Sie eine Vorlage für Genehmigungsregeln (Konsole)

Genehmigungsregelvorlagen sind standardmäßig keinem Repository zugeordnet. Sie können eine Zuordnung zwischen einer Vorlage und einzelnen oder mehreren Repositorys herstellen, wenn Sie die Vorlage erstellen. Sie können die Zuordnungen aber auch zu einem späteren Zeitpunkt vornehmen.

So erstellen Sie eine Genehmigungsregelvorlage (Konsole)

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie Approval rule templates (Genehmigungsregelvorlagen) und dann Create template (Vorlage erstellen) aus.
3. Geben Sie in das Feld Approval rule template name (Name der Genehmigungsregelvorlage) einen aussagekräftigen Namen an, der den Zweck beschreibt. Wenn Sie beispielsweise möchten, dass eine Person aus einer Gruppe leitender Entwickler eine Pull-Anforderung genehmigen muss, bevor sie zusammengeführt werden kann, können Sie die Regel beispielsweise **Require 1 approver from a senior developer** nennen.
4. (Optional) Geben Sie in das Feld Description (Beschreibung) eine Beschreibung des Zwecks dieser Vorlage ein. Dies erleichtert anderen Benutzern die Entscheidung, ob diese Vorlage für ihre Repositorys geeignet ist.
5. Geben Sie in das Feld Number of approvals needed (Anzahl erforderlicher Genehmigungen) die gewünschte Anzahl ein. Der Standardwert ist 1.
6. (Optional) Wenn Sie voraussetzen möchten, dass die Genehmigungen für eine Pull-Anforderung von einer bestimmten Gruppe von Benutzern stammen, wählen Sie unter Approval rule members (Genehmigungsregelmitglieder) die Option Add (Hinzufügen) aus. Wählen Sie unter Approver type (Genehmigertyp) eine der folgenden Optionen aus:
 - IAM-Benutzername oder angenommene Rolle: Diese Option füllt die Amazon Web Services Services-Konto-ID für das Konto, mit dem Sie sich angemeldet haben, vorab aus und erfordert nur einen Namen. Sie kann sowohl für IAM-Benutzer als auch für Benutzer mit Verbundzugriff verwendet werden, deren Name mit dem angegebenen Namen übereinstimmt. Dies ist eine sehr leistungsfähige Option, die hohe Flexibilität bietet. Wenn Sie beispielsweise diese Option wählen und mit dem Amazon Web Services Services-Konto 123456789012 angemeldet sind und angeben **Mary_Major**, werden alle folgenden Informationen als Genehmigungen von diesem Benutzer gezählt:
 - Ein IAM-Benutzer im Konto () `arn:aws:iam::123456789012:user/Mary_Major`
 - Ein Verbundbenutzer, der in IAM als Mary_Major () identifiziert wurde `arn:aws:sts::123456789012:federated-user/Mary_Major`

Diese Option erkennt keine aktive Sitzung eines Benutzers, der die Rolle

CodeCommitReview angenommen hat und den Rollensitzungsnamen Mary_Major

(`arn:aws:sts::123456789012:assumed-role/CodeCommitReview/Mary_Major`)

verwendet, sofern Sie kein Platzhalterzeichen angeben (*Mary_Major). Sie können den Rollennamen auch explizit angeben (CodeCommitReview/Mary_Major).

- Vollqualifizierter ARN: Mit dieser Option können Sie den vollqualifizierten Amazon-Ressourcennamen (ARN) des IAM-Benutzers oder der IAM-Rolle angeben. Diese Option unterstützt auch angenommene Rollen, die von anderen AWS Diensten verwendet werden, wie z. B. AWS Lambda und AWS CodeBuild. Bei angenommenen Rollen sollte das ARN-Format bei Rollen „arn:aws:sts::*AccountID*:assumed-role/*RoleName*“ und bei Funktionen „arn:aws:sts::*AccountID*:assumed-role/*FunctionName*“ sein.

Wenn Sie als Genehmigertyp den IAM-Benutzernamen oder die angenommene Rolle ausgewählt haben, geben Sie im Feld Wert den Namen des IAM-Benutzers oder der IAM-Rolle oder den vollqualifizierten ARN des Benutzers oder der Rolle ein. Wählen Sie erneut Add (Hinzufügen) aus, um weitere Benutzer oder Rollen hinzuzufügen, bis Sie alle Benutzer oder Rollen hinzugefügt haben, deren Genehmigungen auf die Anzahl der erforderlichen Genehmigungen angerechnet werden sollen.

Für beide Genehmigertypen können Platzhalterzeichen (*) in den Werten verwendet werden. Wenn Sie beispielsweise die Option „IAM-Benutzername“ oder „Angenommene Rolle“ wählen und angeben, dass alle Benutzer **CodeCommitReview/***, die die Rolle von annehmen, im **CodeCommitReview** Genehmigungspool gezählt werden. Die individuellen Rollensitzungsnamen zählen zum Erreichen der erforderlichen Anzahl von Genehmigern. Auf diese Weise zählen sowohl Mary_Major als auch Li_Juan als Genehmigungen, wenn sie angemeldet sind und die Rolle CodeCommitReview annehmen. [Weitere Informationen zu IAM ARNs, Platzhaltern und Formaten finden Sie unter IAM-Identifikatoren.](#)

 Note

Genehmigungsregeln unterstützen keine kontoübergreifenden Genehmigungen.

7. (Optional) Geben Sie unter Branch filters (Branch-Filter) die Branch-Zielnamen ein, die zum Filtern der Erstellung von Genehmigungsregeln verwendet werden sollen. Wenn Sie beispielsweise angeben *main*, dass eine Genehmigungsregel für Pull Requests in zugehörigen Repositorys nur erstellt wird, wenn der Ziel-Branch für den Pull Request ein Branch mit dem Namen ist. *main* Sie können Platzhalterzeichen (*) in Branch-Namen verwenden, um Genehmigungsregeln allen Branch-Namen zuzuweisen, die den Namen mit Platzhaltern entsprechen. Sie dürfen Platzhalterzeichen jedoch nicht am Anfang eines Branch-Namens

verwenden. Sie können bis zu 100 Branch-Namen angeben. Wenn Sie keine Filter angeben, gilt die Vorlage für alle Branches in einem zugeordneten Repository.

8. (Optional) Wählen Sie unter Zugeordnete Repositorys in der Liste Repositorys die Repositorys aus, AWS-Region die Sie dieser Genehmigungsregel zuordnen möchten.

 Note

Nach dem Erstellen der Vorlage können Sie Repositorys zuordnen. Weitere Informationen finden Sie unter [Ordnen Sie eine Vorlage für Genehmigungsregeln einem Repository zu](#).

9. Wählen Sie Create (Erstellen) aus.

Approval rule template

Approval rule template name

Require 1 approver from a senior developer for main branch

Description - *optional*

Before a pull request can be merged to the main branch, at least one senior developer must give an approval to the changes.

Number of approvals needed

1

Approval pool members - *optional*

If approval pool members are specified, only approvals from these members will count toward satisfying this rule. You can use wildcards to match multiple approvers with one value.

Approver type [Info](#)

Value

IAM user name or assumed role ▼

SeniorDevelopers/*

Remove

Fully qualified ARN ▼

:s:iam::123456789012:user/Mary_Major

Remove

Add

Branch filters - *optional*

Use branch filters to only apply this template to a pull request if the destination branch name matches a name in the filter list.

Branch name

main

Remove

Add

▼ Associated repositories

Repositories - *optional*

MyDemoRepo ✕

MyTestRepo ✕

Erstellen Sie eine Vorlage für Genehmigungsregeln ()AWS CLI

Sie können die verwenden AWS CLI , um Vorlagen für Genehmigungsregeln zu erstellen. Wenn Sie die verwenden AWS CLI, können Sie Zielverweise für die Vorlage angeben, sodass sie nur für Pull Requests gilt, deren Ziel-Branches mit denen in der Vorlage übereinstimmen.

So erstellen Sie eine Genehmigungsregelvorlage (AWS CLI)

1. Führen Sie am Terminal oder in der Befehlszeile den Befehl `create-approval-rule-template` unter Angabe der folgenden Informationen aus:
 - Name der Genehmigungsregelvorlage. Empfohlen wird ein Name, der ihren Zweck beschreibt.
 - Beschreibung der Genehmigungsregelvorlage. Wie beim Namen sollte die Eingabe aussagekräftig sein.
 - JSON-Struktur der Genehmigungsregelvorlage. Diese Struktur kann Anforderungen für Zielverweise umfassen, d. h. die Ziel-Branches für Pull-Anforderungen, für die die Genehmigungsregel gilt, und Genehmigungs-Pool-Mitglieder, also Benutzer, deren Genehmigungen für die Anzahl erforderlicher Genehmigungen gezählt werden.

Beim Erstellen des Inhalts der Genehmigungsregel können Sie Genehmiger in einem Genehmigungs-Pool auf zwei Arten angeben:

- **CodeCommitApprovers:** Für diese Option sind nur ein Amazon Web Services Services-Konto und eine Ressource erforderlich. Sie kann sowohl für IAM-Benutzer als auch für Benutzer mit Verbundzugriff verwendet werden, deren Name mit dem angegebenen Ressourcennamen übereinstimmt. Dies ist eine sehr leistungsfähige Option, die hohe Flexibilität bietet. Wenn Sie beispielsweise das AWS Konto 123456789012 und angeben, werden alle folgenden Angaben als Genehmigungen gezählt **Mary_Major**, die von diesem Benutzer stammen:
 - Ein IAM-Benutzer im Konto () `arn:aws:iam::123456789012:user/Mary_Major`
 - Ein Verbundbenutzer, der in IAM als `Mary_Major` () identifiziert wurde `arn:aws:sts::123456789012:federated-user/Mary_Major`

Diese Option erkennt keine aktive Sitzung, in der jemand die Rolle von *SeniorDevelopers* mit dem Rollensitzungsnamen () übernimmt, es sei denn, Sie fügen einen Platzhalter *Mary_Major* (`arn:aws:sts::123456789012:assumed-role/SeniorDevelopers/Mary_Major`) hinzu. `*Mary_Major`

- Vollqualifizierter ARN: Mit dieser Option können Sie den vollqualifizierten Amazon-Ressourcennamen (ARN) des IAM-Benutzers oder der IAM-Rolle angeben.

[Weitere Informationen zu IAM ARNs, Platzhaltern und Formaten finden Sie unter IAM-Identifikatoren.](#)

Im folgenden Beispiel wird eine Genehmigungsregelvorlage namens **2-approver-rule-for-main** mit der Beschreibung **Requires two developers from the team to approve the pull request if the destination branch is main** erstellt. Für die Vorlage müssen zwei Benutzer, die die Rolle **CodeCommitReview** angenommen haben, alle Pull-Anforderungen genehmigen, bevor sie im Branch **main** zusammengeführt werden können:

```
aws codecommit create-approval-rule-template --approval-rule-template-name 2-
approver-rule-for-main --approval-rule-template-description "Requires two
developers from the team to approve the pull request if the destination branch
is main" --approval-rule-template-content "{\"Version\": \"2018-11-08\",
\"DestinationReferences\": [\"refs/heads/main\"],\"Statements\": [{\"Type
\": \"Approvers\", \"NumberOfApprovalsNeeded\": 2, \"ApprovalPoolMembers\":
[\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}"
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
  "approvalRuleTemplate": {
    "approvalRuleTemplateName": "2-approver-rule-for-main",
    "creationDate": 1571356106.936,
    "approvalRuleTemplateId": "dd8b17fe-EXAMPLE",
    "approvalRuleTemplateContent": "{\"Version\": \"2018-11-08\",
\"DestinationReferences\": [\"refs/heads/main\"],\"Statements\": [{\"Type
\": \"Approvers\", \"NumberOfApprovalsNeeded\": 2, \"ApprovalPoolMembers\":
[\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
    "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
    "approvalRuleTemplateDescription": "Requires two developers from the team
to approve the pull request if the destination branch is main",
    "lastModifiedDate": 1571356106.936,
    "ruleContentSha256": "4711b576EXAMPLE"
  }
}
```

Ordnen Sie eine Vorlage für Genehmigungsregeln einem Repository zu

Vorlagen für Genehmigungsregeln werden in einem bestimmten Bereich erstellt AWS-Region, wirken sich jedoch AWS-Region erst dann auf die Repositorys in diesem Bereich aus, wenn sie verknüpft werden. Um eine Vorlage einzelnen oder mehreren Repositorys zuzuweisen, müssen Sie die Vorlage den betreffenden Repositorys zuordnen. Sie können eine einzelne Vorlage auf mehrere Repositorys in einem anwenden. AWS-Region Auf diese Weise können Sie den Entwicklungs-Workflow in Repositorys automatisieren und standardisieren, indem Sie konsistente Bedingungen für das Genehmigen und das Zusammenführen von Pull-Anforderungen schaffen.

Sie können eine Genehmigungsregelvorlage nur Repositorys zuordnen, in AWS-Region denen die Vorlage für Genehmigungsregeln erstellt wurde.

Weitere Informationen zu verwalteten Richtlinien und Berechtigungen für Genehmigungsregelvorlagen finden Sie unter [Berechtigungen für Aktionen auf Vorlagen für Genehmigungsregeln](#) und [AWS verwaltete Richtlinien für CodeCommit](#).

Themen

- [Ordnen Sie eine Vorlage für Genehmigungsregeln zu \(Konsole\)](#)
- [Ordnen Sie eine Vorlage für Genehmigungsregeln zu \(AWS CLI\)](#)

Ordnen Sie eine Vorlage für Genehmigungsregeln zu (Konsole)

Möglicherweise haben Sie einer Genehmigungsregelvorlage bereits bei der Erstellung Repositorys zugeordnet. (Dieser Schritt ist optional.) Sie können Zuordnungen hinzufügen oder entfernen, indem Sie die Vorlage bearbeiten.

So ordnen Sie eine Genehmigungsregelvorlage Repositorys zu

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie Approval rule templates (Genehmigungsregelvorlagen) aus. Wählen Sie die Vorlage und dann Edit (Bearbeiten) aus.
3. Wählen Sie unter Associated Repositories (Zugeordnete Repositorys) die Repositorys in der Liste Repositories (Repositorys) aus. Jedes zugeordnete Repository wird unter dem Listenfeld angezeigt.

4. Wählen Sie **Save (Speichern)** aus. Genehmigungsregeln werden nun allen in den zugeordneten Repositorys erstellten Pull-Anforderungen zugewiesen.

Ordnen Sie eine Vorlage für Genehmigungsregeln zu (AWS CLI)

Sie können die verwenden AWS CLI , um eine Vorlage für Genehmigungsregeln einem oder mehreren Repositorys zuzuordnen.

So ordnen Sie eine Vorlage einem einzelnen Repository zu

1. Führen Sie am Terminal oder in der Befehlszeile den Befehl `associate-approval-rule-template-with-repository` unter Angabe der folgenden Informationen aus:
 - Name der Genehmigungsregelvorlage, die Sie einem Repository zuordnen möchten.
 - Name des Repositorys, das der Genehmigungsregelvorlage zugeordnet werden soll.

Um beispielsweise eine Genehmigungsregelvorlage mit dem Namen einem Repository **2-approver-rule-for-main** mit dem Namen zu verknüpfen: **MyDemoRepo**

```
aws codecommit associate-approval-rule-template-with-repository --repository-name MyDemoRepo --approval-rule-template-name 2-approver-rule-for-main
```

2. Bei erfolgreicher Ausführung gibt dieser Befehl nichts zurück.

So ordnen Sie eine Vorlage mehreren Repositorys zu

1. Führen Sie am Terminal oder in der Befehlszeile den Befehl `batch-associate-approval-rule-template-with-repositories` unter Angabe der folgenden Informationen aus:
 - Name der Genehmigungsregelvorlage, die Sie einem Repository zuordnen möchten.
 - Namen der Repositorys, die der Genehmigungsregelvorlage zugeordnet werden sollen.

So ordnen Sie beispielsweise eine Genehmigungsregelvorlage namens **2-approver-rule-for-main** Repositorys mit den Namen **MyDemoRepo** und **MyOtherDemoRepo** zu:

```
aws codecommit batch-associate-approval-rule-template-with-repositories --  
repository-names "MyDemoRepo", "MyOtherDemoRepo" --approval-rule-template-name 2-  
approver-rule-for-main
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{  
  "associatedRepositoryNames": [  
    "MyDemoRepo",  
    "MyOtherDemoRepo"  
  ],  
  "errors": []  
}
```

Verwalten von Genehmigungsregelvorlagen

Sie können die Vorlagen für Genehmigungsregeln in einer verwalteten AWS-Region, um zu verstehen, wie sie verwendet werden und wofür sie bestimmt sind. Sie können beispielsweise die Namen und Beschreibungen von Genehmigungsregelvorlagen bearbeiten, um andere Benutzer über den Verwendungszweck zu informieren. Sie können alle Vorlagen für Genehmigungsregeln in einer AWS-Region auflisten und Informationen zu Inhalt und Struktur einer Vorlage abrufen. Sie können ermitteln, welche Vorlagen einem Repository und welche Repositorys einer Vorlage zugeordnet sind.

Weitere Informationen zu verwalteten Richtlinien und Berechtigungen für Genehmigungsregelvorlagen finden Sie unter [Berechtigungen für Aktionen auf Vorlagen für Genehmigungsregeln](#) und [AWS verwaltete Richtlinien für CodeCommit](#).

Vorlagen für Genehmigungsregeln verwalten (Konsole)

Sie können Ihre Vorlagen für Genehmigungsregeln in der CodeCommit Konsole anzeigen und verwalten.

So verwalten Sie Genehmigungsregelvorlagen

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie Vorlagen für Genehmigungsregeln, um die Liste der Vorlagen für Genehmigungsregeln in dem Bereich anzuzeigen, in AWS-Region dem Sie angemeldet sind.

 Note

Vorlagen für Genehmigungsregeln sind nur dort verfügbar AWS-Region , wo sie erstellt wurden.

3. Wenn Sie Änderungen an einer Vorlage vornehmen möchten, wählen Sie sie in der Liste und dann Edit (Bearbeiten) aus.
4. Nehmen Sie die gewünschten Änderungen vor und wählen Sie dann Save.

Vorlagen für Genehmigungsregeln verwalten (AWS CLI)

Sie können Genehmigungsregelvorlagen mit den folgenden AWS CLI -Befehlen verwalten:

- [list-approval-rule-templates](#), um eine Liste aller Vorlagen für Genehmigungsregeln in einem AWS-Region
- [get-approval-rule-template](#) zeigt den Inhalt einer Genehmigungsregelvorlage an.
- [update-approval-rule-template-content](#) ändert den Inhalt einer Genehmigungsregelvorlage.
- [update-approval-rule-template-name](#) ändert den Namen einer Genehmigungsregelvorlage.
- [update-approval-rule-template-description](#) ändert die Beschreibung einer Genehmigungsregelvorlage.
- [list-repositories-for-approval-rule-template](#) zeigt alle Repositorys an, die einer Genehmigungsregelvorlage zugeordnet sind.
- [list-associated-approval-rule-templates-for-repository](#) zeigt alle Genehmigungsregelvorlagen an, die einem Repository zugeordnet sind.

Um alle Vorlagen für Genehmigungsregeln in einem aufzulisten AWS-Region

1. Führen Sie am Terminal oder über die Befehlszeile den Befehl `list-approval-rule-templates` aus. Um beispielsweise alle Vorlagen für Genehmigungsregeln in der Region USA Ost (Ohio) aufzulisten:

```
aws codecommit list-approval-rule-templates --region us-east-2
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
  "approvalRuleTemplateName": [
    "2-approver-rule-for-main",
    "1-approver-rule-for-all-pull-requests"
  ]
}
```

So rufen Sie den Inhalt einer Genehmigungsregelvorlage ab

1. Führen Sie am Terminal oder in der Befehlszeile den Befehl `get-approval-rule-template` aus und geben den Namen der Genehmigungsregelvorlage an:

```
aws codecommit get-approval-rule-template --approval-rule-template-name 1-approver-
rule-for-all-pull-requests
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
  "approvalRuleTemplate": {
    "approvalRuleTemplateContent": "{\"Version\": \"2018-11-08\", \"Statements \": [{\"Type\": \"Approvers\", \"NumberOfApprovalsNeeded\": 1, \"ApprovalPoolMembers \": [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
    "ruleContentSha256": "621181bbEXAMPLE",
    "lastModifiedDate": 1571356106.936,
    "creationDate": 1571356106.936,
    "approvalRuleTemplateName": "1-approver-rule-for-all-pull-requests",
    "lastModifiedUser": "arn:aws:iam::123456789012:user/Li_Juan",
    "approvalRuleTemplateId": "a29abb15-EXAMPLE",
    "approvalRuleTemplateDescription": "All pull requests must be approved by one developer on the team."
  }
}
```

So aktualisieren Sie den Inhalt einer Genehmigungsregelvorlage

1. Führen Sie am Terminal oder an der Eingabeaufforderung den Befehl `update-approval-rule-template-content` aus und geben Sie den Namen der Vorlage und den geänderten Inhalt an. So ändern Sie beispielsweise den Inhalt einer Genehmigungsregelvorlage namens **1-approver-**

rule, um den Genehmigungs-Pool für Benutzer, die die Rolle **CodeCommitReview** annehmen, neu zu definieren:

```
aws codecommit update-approval-rule-template-content --approval-rule-template-name 1-approver-rule --new-rule-content "{\"Version\": \"2018-11-08\", \"DestinationReferences\": [\"refs/heads/main\"], \"Statements\": [{\"Type\": \"Approvers\", \"NumberOfApprovalsNeeded\": 2, \"ApprovalPoolMembers\": [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}"
```

- Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
  "approvalRuleTemplate": {
    "creationDate": 1571352720.773,
    "approvalRuleTemplateDescription": "Requires 1 approval for all pull requests from the CodeCommitReview pool",
    "lastModifiedDate": 1571358728.41,
    "approvalRuleTemplateId": "41de97b7-EXAMPLE",
    "approvalRuleTemplateContent": "{\"Version\": \"2018-11-08\", \"Statements\": [{\"Type\": \"Approvers\", \"NumberOfApprovalsNeeded\": 1, \"ApprovalPoolMembers\": [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
    "approvalRuleTemplateName": "1-approver-rule-for-all-pull-requests",
    "ruleContentSha256": "2f6c21a5EXAMPLE",
    "lastModifiedUser": "arn:aws:iam::123456789012:user/Li_Juan"
  }
}
```

So aktualisieren Sie den Namen einer Genehmigungsregelvorlage

- Führen Sie den Befehl `update-approval-rule-template-name` am Terminal oder an der Eingabeaufforderung aus und geben Sie den aktuellen Namen und den neuen Namen an. So ändern Sie beispielsweise den Namen einer Genehmigungsregelvorlage von **1-approver-rule** in **1-approver-rule-for-all-pull-requests**:

```
aws codecommit update-approval-rule-template-name --old-approval-rule-template-name "1-approver-rule" --new-approval-rule-template-name "1-approver-rule-for-all-pull-requests"
```

- Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
```

```

    "approvalRuleTemplate": {
      "approvalRuleTemplateName": "1-approver-rule-for-all-pull-requests",
      "lastModifiedDate": 1571358241.619,
      "approvalRuleTemplateId": "41de97b7-EXAMPLE",
      "approvalRuleTemplateContent": "{\"Version\": \"2018-11-08\", \"Statements\": [{\"Type\": \"Approvers\", \"NumberOfApprovalsNeeded\": 1, \"ApprovalPoolMembers\": [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
      "creationDate": 1571352720.773,
      "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
      "approvalRuleTemplateDescription": "All pull requests must be approved by one developer on the team.",
      "ruleContentSha256": "2f6c21a5cEXAMPLE"
    }
  }
}

```

So aktualisieren Sie die Beschreibung einer Genehmigungsregelvorlage

1. Führen Sie am Terminal oder in der Befehlszeile den Befehl `update-approval-rule-template-description` aus und geben Sie den Namen der Genehmigungsregelvorlage und die neue Beschreibung an:

```

aws codecommit update-approval-rule-template-description --approval-rule-template-name "1-approver-rule-for-all-pull-requests" --approval-rule-template-description "Requires 1 approval for all pull requests from the CodeCommitReview pool"

```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```

{
  "approvalRuleTemplate": {
    "creationDate": 1571352720.773,
    "approvalRuleTemplateDescription": "Requires 1 approval for all pull requests from the CodeCommitReview pool",
    "lastModifiedDate": 1571358728.41,
    "approvalRuleTemplateId": "41de97b7-EXAMPLE",
    "approvalRuleTemplateContent": "{\"Version\": \"2018-11-08\", \"Statements\": [{\"Type\": \"Approvers\", \"NumberOfApprovalsNeeded\": 1, \"ApprovalPoolMembers\": [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
    "approvalRuleTemplateName": "1-approver-rule-for-all-pull-requests",
    "ruleContentSha256": "2f6c21a5cEXAMPLE",
    "lastModifiedUser": "arn:aws:iam::123456789012:user/Li_Juan"
  }
}

```

```
}
```

So listen Sie alle Repositorys auf, die einer Vorlage zugeordnet sind

1. Führen Sie in der Befehlszeile oder am Terminal den Befehl `list-repositories-for-approval-rule-template` aus und geben Sie den Namen der Vorlage an:

```
aws codecommit list-repositories-for-approval-rule-template --approval-rule-template-name 2-approver-rule-for-main
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
  "repositoryNames": [
    "MyDemoRepo",
    "MyClonedRepo"
  ]
}
```

So listen Sie alle Vorlagen auf, die einem Repository zugeordnet sind

1. Führen Sie in der Befehlszeile oder am Terminal den Befehl `list-associated-approval-rule-templates-for-repository` aus und geben Sie den Namen des Repositorys an:

```
aws codecommit list-associated-approval-rule-templates-for-repository --repository-name MyDemoRepo
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
  "approvalRuleTemplateName": [
    "2-approver-rule-for-main",
    "1-approver-rule-for-all-pull-requests"
  ]
}
```

Trennen Sie die Zuordnung einer Vorlage für eine Genehmigungsregel

Wenn die von einer Genehmigungsregelvorlage generierten Genehmigungsregeln für den Workflow Ihres Teams in einem Repository nicht mehr geeignet sind, können Sie die Zuordnung der Vorlage zu diesem Repository aufheben. Durch das Aufheben der Zuordnung einer Vorlage werden keine Genehmigungsregeln entfernt, die erstellt wurden, während die Vorlage dem Repository zugeordnet war.

Weitere Informationen zu verwalteten Richtlinien und Berechtigungen für Genehmigungsregelvorlagen finden Sie unter [Berechtigungen für Aktionen auf Vorlagen für Genehmigungsregeln](#) und [AWS verwaltete Richtlinien für CodeCommit](#).

Trennen Sie die Zuordnung einer Vorlage für Genehmigungsregeln (Konsole)

Mit der Konsole können Sie die Zuordnung zwischen einem Repository und einer Genehmigungsregelvorlage aufheben.

So heben Sie die Zuordnung einer Genehmigungsregelvorlage zu Repositories auf

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie Approval rule templates (Genehmigungsregelvorlagen) aus. Wählen Sie die Vorlage aus, deren Zuordnung zu einzelnen oder mehreren Repositories aufgehoben werden soll. Wählen Sie dann Edit (Bearbeiten) aus.
3. Wählen Sie unter Associated repositories (Zugeordnete Repositories) das X neben den Repositories aus, deren Zuordnung aufgehoben werden soll. Die Repository-Namen werden nicht mehr angezeigt.
4. Wählen Sie Save (Speichern) aus. Genehmigungsregeln werden auf in diesen Repositories erstellte Pull-Anforderungen nicht angewendet. Die Regeln werden weiterhin auf Pull-Anforderungen angewendet, die bei bestehender Zuordnung getätigt wurden.

Trennen Sie die Zuordnung einer Vorlage für Genehmigungsregeln ()AWS CLI

Sie können den verwenden AWS CLI , um die Zuordnung eines oder mehrerer Repositorys zu einer Vorlage für Genehmigungsregeln aufzuheben.

So heben Sie die Zuordnung einer Genehmigungsregelvorlage zu einem Repository auf

1. Führen Sie am Terminal oder in der Befehlszeile den Befehl `disassociate-approval-rule-template-from-repository` unter Angabe der folgenden Informationen aus:

- Name der Genehmigungsregelvorlage.
- Name des Repositorys.

So heben Sie beispielsweise die Zuordnung einer Genehmigungsregelvorlage namens **1-approver-rule-for-all-pull-requests** zu einem Repository namens **MyDemoRepo** auf:

```
aws codecommit disassociate-approval-rule-template-from-repository --repository-name MyDemoRepo --approval-rule-template-name 1-approver-rule-for-all-pull-requests
```

2. Bei erfolgreicher Ausführung gibt dieser Befehl nichts zurück.

So heben Sie die Zuordnung einer Genehmigungsregelvorlage zu mehreren Repositorys auf

1. Führen Sie am Terminal oder in der Befehlszeile den Befehl `batch-disassociate-approval-rule-template-from-repositories` unter Angabe der folgenden Informationen aus:

- Name der Genehmigungsregelvorlage.
- Namen der Repositorys.

So heben Sie beispielsweise die Zuordnung einer Genehmigungsregelvorlage namens **1-approver-rule-for-all-pull-requests** zu den Repositorys **MyDemoRepo** und **MyOtherDemoRepo** auf:

```
aws codecommit batch-disassociate-approval-rule-template-from-repositories --repository-names "MyDemoRepo", "MyOtherDemoRepo" --approval-rule-template-name 1-approver-rule-for-all-pull-requests
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
  "disassociatedRepositoryNames": [
    "MyDemoRepo",
    "MyOtherDemoRepo"
  ],
  "errors": []
}
```

Löschen Sie eine Vorlage für Genehmigungsregeln

Sie können eine Genehmigungsregelvorlage löschen, wenn Sie sie nicht in Repositorys verwenden. Das Löschen nicht verwendeter Genehmigungsregelvorlagen erleichtert das Verwalten der und die Suche nach Vorlagen, die für Ihre Workflows geeignet sind.

Weitere Informationen zu verwalteten Richtlinien und Berechtigungen für Genehmigungsregelvorlagen finden Sie unter [Berechtigungen für Aktionen auf Vorlagen für Genehmigungsregeln](#) und [AWS verwaltete Richtlinien für CodeCommit](#).

Themen

- [Löschen Sie eine Vorlage für Genehmigungsregeln \(Konsole\)](#)
- [Löschen Sie eine Vorlage für Genehmigungsregeln \(\)AWS CLI](#)

Löschen Sie eine Vorlage für Genehmigungsregeln (Konsole)

Sie können eine Genehmigungsregelvorlage löschen, wenn sie für Ihre Entwicklungsarbeit nicht mehr benötigt wird. Wenn Sie eine Genehmigungsregelvorlage mit der Konsole löschen, wird beim Löschen die Zuordnung zu allen Repositorys aufgehoben.

So löschen Sie eine Genehmigungsregelvorlage

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie Approval rule templates (Genehmigungsregelvorlagen) aus. Wählen Sie die zu löschende Vorlage aus und klicken Sie auf Delete (Löschen).

Löschen Sie eine Vorlage für Genehmigungsregeln ()AWS CLI

Sie können die verwenden AWS CLI , um eine Genehmigungsregel zu löschen, wenn sie von allen Repositorys getrennt wurde. Weitere Informationen finden Sie unter [Trennen Sie die Zuordnung einer Vorlage für Genehmigungsregeln \(\)AWS CLI](#).

So löschen Sie eine Genehmigungsregelvorlage

1. Führen Sie an einem Terminal oder in der Befehlszeile den Befehl `delete-approval-rule-template` aus und geben Sie den Namen der zu löschenden Genehmigungsregelvorlage an:

```
aws codecommit delete-approval-rule-template --approval-rule-template-name 1-approver-for-all-pull-requests
```

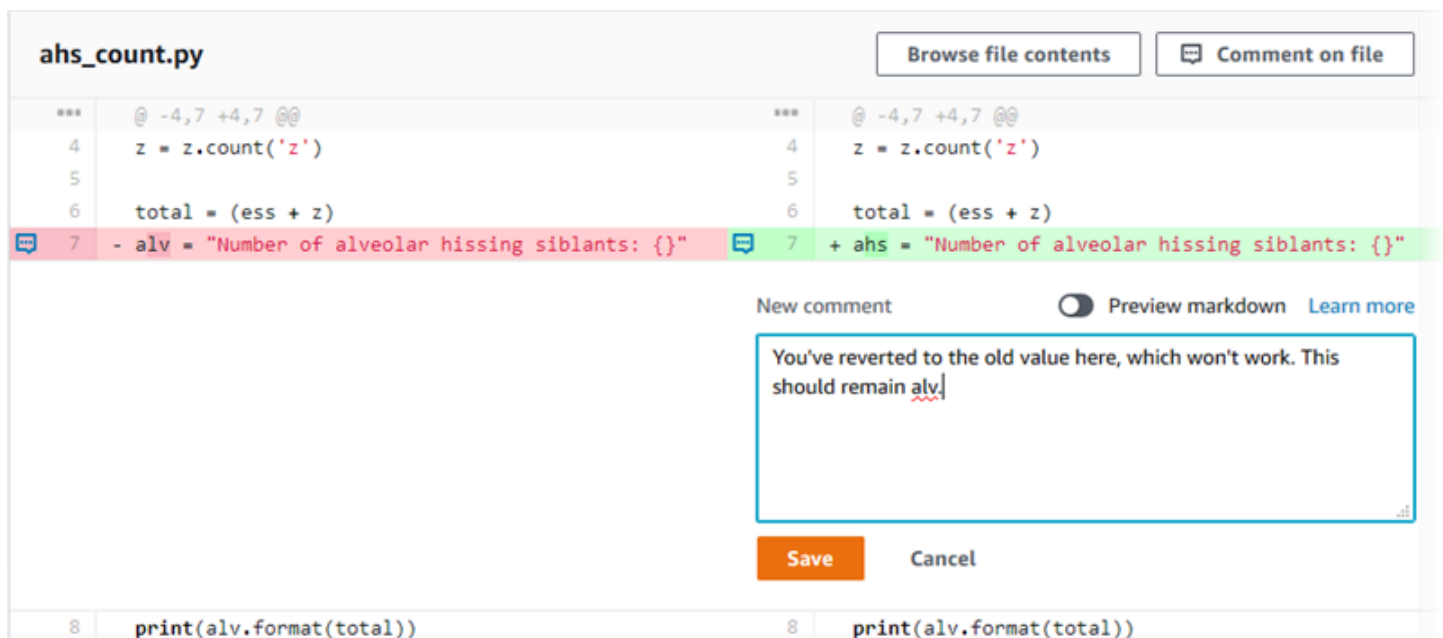
2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte. Wenn die Genehmigungsregelvorlage bereits gelöscht wurde, gibt dieser Befehl keinen Wert zurück.

```
{
  "approvalRuleTemplateId": "41de97b7-EXAMPLE"
}
```

Mit Commits in Repositorys arbeiten AWS CodeCommit

Commits sind Snapshots der Inhalte und der Änderungen, die an den Inhalten des Repositorys vorgenommen wurden. Jedes Mal, wenn ein Benutzer einen Commit ausführt und eine Änderung anstößt, werden diese Informationen gespeichert. Zudem wird gespeichert, wer die Änderung ausgeführt hat, sowie das Datum und die Uhrzeit des Commits und die Änderungen, die mit dem Commit ausgeführt wurden. Zur einfachen Identifizierung spezifischer Commits lassen sich Tags hinzufügen. In CodeCommit können Sie:

- Commits überprüfen
- Den Verlauf von Commits in einem Diagramm darstellen
- Einen Commit mit dem übergeordneten Commit oder einem anderen Spezifizierer vergleichen
- Ihren Commits Kommentare hinzufügen und auf Kommentare anderer Benutzer antworten



Bevor Sie Commits in ein CodeCommit Repository übertragen können, müssen Sie Ihren lokalen Computer so einrichten, dass er eine Verbindung zum Repository herstellt. Die einfachste Methode dazu finden Sie unter [Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden](#).

Hinweise zur Arbeit mit anderen Aspekten Ihres Repositorys finden Sie unter [CodeCommitArbeiten mit Repositorien](#), [Mit Dateien arbeiten](#), [Verwenden von Pull-AnforderungenArbeiten mit Zweigen](#), und [Mit Benutzereinstellungen arbeiten](#).

Themen

- [Erstellen Sie einen Commit in AWS CodeCommit](#)
- [Commit-Details anzeigen in AWS CodeCommit](#)
- [Vergleiche Commits in AWS CodeCommit](#)
- [Kommentieren Sie einen Commit in AWS CodeCommit](#)
- [Erstellen Sie ein Git-Tag in AWS CodeCommit](#)
- [Git-Tag-Details anzeigen in AWS CodeCommit](#)
- [Lösche ein Git-Tag in AWS CodeCommit](#)

Erstellen Sie einen Commit in AWS CodeCommit

Wenn Sie den ersten Commit für ein neues Repository erstellen, verwenden Sie den Befehl AWS CLI und den `put-file` Befehl. Dadurch wird der erste Commit erstellt und Sie können den Standard-Branch für Ihr neues Repository erstellen und angeben. Du kannst Git oder the verwenden AWS CLI , um einen Commit in einem CodeCommit Repository zu erstellen. Wenn das lokale Repo mit einem CodeCommit Repository verbunden ist, verwendest du Git, um den Commit vom lokalen Repo in das Repository zu übertragen. CodeCommit Informationen zum Erstellen eines Commits direkt in der CodeCommit Konsole finden Sie unter [Eine Datei erstellen oder zu einem AWS CodeCommit Repository hinzufügen](#) und. [Den Inhalt einer Datei in einem AWS CodeCommit Repository bearbeiten](#)

Note

Als bewährte Methode empfehlen wir, die neuesten unterstützten Versionen von AWS CLI, Git und anderer Software zu verwenden. Wenn Sie die verwenden AWS CLI, stellen Sie sicher, dass Sie eine aktuelle Version installiert haben, um sicherzustellen, dass Sie eine Version verwenden, die den `create-commit` Befehl enthält.

Themen

- [Erstellen Sie den ersten Commit für ein Repository mit dem AWS CLI](#)
- [Einen Commit mit einem Git-Client erstellen](#)
- [Erstelle einen Commit AWS CLI](#)

Erstellen Sie den ersten Commit für ein Repository mit dem AWS CLI

Sie können den Befehl AWS CLI und den `put-file` Befehl verwenden, um Ihren ersten Commit für ein Repository zu erstellen. Mit `put-file` wird ein erster Commit erstellt, der eine Datei zu Ihrem leeren Repository hinzufügt, und es wird ein Branch mit dem von Ihnen angegebenen Namen erstellt. Es bestimmt den neuen Branch als Standard-Branch für dein Repository.

Note

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den. AWS CLI Weitere Informationen finden Sie unter [Befehlszeilenreferenz](#).

Um den ersten Commit für ein Repository mit dem zu erstellen AWS CLI

1. Erstellen Sie auf Ihrem lokalen Computer die Datei, die Sie als erste Datei zum CodeCommit Repository hinzufügen möchten. Eine gängige Praxis besteht darin, eine README .md Markdown-Datei zu erstellen, die anderen Repository-Benutzern den Zweck dieses Repositories erklärt. Wenn Sie eine README .md Datei einschließen, wird der Inhalt der Datei automatisch unten auf der Codeseite für Ihr Repository in der CodeCommit Konsole angezeigt.
2. Führen Sie am Terminal oder in der Befehlszeile den Befehl `put-file` unter Angabe der folgenden Informationen aus:
 - Der Name des Repositories, dem Sie die erste Datei hinzufügen möchten.
 - Der Name des Branches, den Sie als Standard-Branch erstellen möchten.
 - Der lokale Speicherort der Datei Die Syntax für diesen Standort hängt von Ihrem lokalen Betriebssystem ab.
 - Der Name der Datei, die Sie hinzufügen möchten, einschließlich des Pfads, in dem die aktualisierte Datei im Repository gespeichert ist.
 - Der Benutzername und die E-Mail-Adresse, die Sie dieser Datei zuordnen möchten.
 - Eine Commit-Nachricht, die erklärt, warum Sie diese Datei hinzugefügt haben

Der Benutzername, die E-Mail-Adresse und die Commit-Nachricht sind optional, können aber anderen Benutzern helfen, zu erfahren, wer die Änderung vorgenommen hat und warum. Wenn Sie keinen Benutzernamen angeben, wird CodeCommit standardmäßig Ihr IAM-Benutzername oder eine Ableitung Ihres Konsolen-Logins als Autorennamen verwendet.

Um beispielsweise eine Datei *README.md* mit dem Namen Base-6-codierten Dateiinhalt zu einem Repository hinzuzufügen, das nach einem Branch namens benannt *MyDemoRepo* ist: *development*

```
aws codecommit put-file --repository-name MyDemoRepo --branch-name development --
file-path README.md --file-content "EXAMPLE0123456789example0123456789EXAMPLE1"
--name "Mary Major" --email "mary_major@example.com" --commit-message "I added a
quick readme for our new team repository."
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
  "commitId": "724caa36EXAMPLE",
  "blobId": "a8a94062EXAMPLE",
  "treeId": "08b2fc73EXAMPLE"
}
```

Einen Commit mit einem Git-Client erstellen

Du kannst Commits mit einem Git-Client erstellen, der auf deinem lokalen Computer installiert ist, und diese Commits dann in dein CodeCommit Repository übertragen.

1. Sorgen Sie dafür, dass die Voraussetzungen erfüllt sind, einschließlich [Einrichtung](#).

Important

Wenn Sie die Einrichtung nicht abgeschlossen haben, können Sie mit Git weder eine Verbindung herstellen noch einen Commit für das Repository durchführen.

2. Vergewissern Sie sich, dass Sie den Commit für den richtigen Branch erstellen. Um eine Liste der verfügbaren Branches zu sehen und festzustellen, welchen Branch Sie derzeit verwenden, führen Sie `git branch` aus. Alle Branches werden angezeigt. Ihr aktueller Branch ist mit einem Sternchen (*) gekennzeichnet. Wenn Sie zu einem anderen Branch wechseln möchten, führen Sie `git checkout branch-name` aus. Wenn dies dein erster Commit ist, führe den `git config` Befehl aus, um deinen Git-Client so zu konfigurieren, dass er einen ersten Branch mit dem Namen erstellt, den du für diesen Branch verwenden möchtest. Wenn du zum Beispiel möchtest, dass dein Standard-Branch den folgenden Namen hat *development*:

```
git config --local init.defaultBranch development
```

 Tip

Dieser Befehl ist nur in Git v.2.28 und höher verfügbar.

Du kannst diesen Befehl auch ausführen, um deinen Standard-Branch-Namen **development** für alle neu erstellten Repositorys auf festzulegen:

```
git config --global init.defaultBranch development
```

3. Ändern Sie den Branch (z. B. durch Hinzufügen, Ändern oder Löschen einer Datei).

Erstellen Sie beispielsweise im lokalen Repository eine Datei `bird.txt` mit dem folgenden Text:

```
bird.txt
-----
Birds (class Aves or clade Avialae) are feathered, winged, two-legged, warm-blooded, egg-laying vertebrates.
```

4. Führen Sie `git status` aus. Die Befehlsausgabe sollte besagen, dass `bird.txt` noch in keinem schwebenden Commit enthalten ist:

```
...
Untracked files:
  (use "git add <file>..." to include in what will be committed)

    bird.txt
```

5. Führen Sie `git add bird.txt` aus, um die neue Datei in den schwebenden Commit aufzunehmen.
6. Wenn Sie den Befehl `git status` erneut ausführen, sollte die Befehlsausgabe der folgenden ähneln. Damit wird angegeben, dass `bird.txt` nun zum schwebenden Commit gehört oder für den Commit bereitgestellt wird:

```
...
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)
```

```
new file:   bird.txt
```

7. Um den Commit abzuschließen, führen Sie `git commit` mit der Option `-m` aus (z. B. `git commit -m "Adding bird.txt to the repository."`). Mit der Option `-m` wird die Commit-Nachricht erstellt.
8. Wenn Sie den Befehl `git status` erneut ausführen, sollte die Befehlsausgabe der folgenden ähneln. Es zeigt an, dass der Commit bereit ist, vom lokalen Repo in das Repository übertragen zu werden: CodeCommit

```
...
nothing to commit, working directory clean
```

9. Bevor Sie den finalisierten Commit vom lokalen Repo in das CodeCommit Projektarchiv übertragen, können Sie sehen, was Sie durch Ausführen übertragen. Dabei *remote-name* steht der Nickname `git diff --stat remote-name/branch-name`, den das lokale Repo für das CodeCommit Projektarchiv verwendet, und der Name des Branches, der verglichen werden *branch-name* soll.

Tip

Um den Remote-Namen zu erhalten, führen Sie `git remote` aus. Um eine Liste der Branch-Namen zu erhalten, führen Sie den Befehl `git branch` aus. Der aktuelle Branch ist mit einem Sternchen (*) gekennzeichnet. Sie können auch `git status` ausführen, um den aktuellen Branch-Namen abzurufen.

Note

Wenn Sie das Repository geklont haben, *remote-name* ist das aus der Sicht des lokalen Repos nicht der Name des Repositories. CodeCommit Wenn Sie ein Repository klonen, *remote-name* wird automatisch auf `origin` gesetzt.

Beispielsweise würde die Ausgabe von `git diff --stat origin/main` der folgenden gleichen:

```
bird.txt | 1 +
1 file changed, 1 insertion(+)
```

In der Ausgabe wird davon ausgegangen, dass Sie das lokale Repository bereits mit dem CodeCommit Repository verbunden haben. (Detaillierte Anweisungen finden Sie unter [Herstellen einer Verbindung mit einem Repository](#).)

10. Wenn Sie bereit sind, den Commit vom lokalen Repo in das CodeCommit Repository zu übertragen, führen Sie den Befehl aus. Dabei **remote-name** steht der Spitznamegit push **remote-name branch-name**, den das lokale Repo für das CodeCommit Repository verwendet, und **branch-name** ist der Name des Branches, der in das Repository übertragen werden soll. CodeCommit

Beispielsweise würde die Ausgabe von git push origin main der folgenden gleichen:

Für HTTPS:

```
Counting objects: 7, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 516 bytes | 0 bytes/s, done.
Total 5 (delta 2), reused 0 (delta 0)
remote:
To https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo
    b9e7aa6..3dbf4dd main -> main
```

Für SSH:

```
Counting objects: 7, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 516 bytes | 0 bytes/s, done.
Total 5 (delta 2), reused 0 (delta 0)
remote:
To ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo
    b9e7aa6..3dbf4dd main -> main
```

Tip

Wenn Sie dem Befehl git push die Option -u hinzufügen (z. B. git push -u origin main), müssen Sie künftig nur noch git push ausführen, da die Upstream-Nachverfolgungsdaten

bereits festgelegt wurden. Führen Sie `git remote show remote-name` aus (z. B. `git remote show origin`), um Upstream-Nachverfolgungsdaten abzurufen.

Weitere Optionen findest du in deiner Git-Dokumentation.

Erstelle einen Commit AWS CLI

Sie können den Befehl `AWS CLI` und den `create-commit` Befehl verwenden, um einen Commit für ein Repository an der Spitze eines bestimmten Branches zu erstellen. Sie können auch einen nicht referenzierten Mergecommit für die Zusammenführung von zwei Commit-Spezifizierern erstellen. Weitere Informationen finden Sie unter [Erstellen eines nicht referenzierten Commits](#).

Note

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter [Befehlszeilenreferenz](#).

So erstellen Sie einen Commit

1. Nehmen Sie auf dem lokalen Computer die Änderungen vor, die an das CodeCommit-Repository übertragen werden sollen.
2. Führen Sie am Terminal oder in der Befehlszeile den Befehl `create-commit` unter Angabe der folgenden Informationen aus:
 - Das Repository, an das Sie die Änderungen übertragen möchten.
 - Der Branch, an den Sie die Änderungen übertragen möchten.
 - Die vollständige Commit-ID des letzten an diesem Branch vorgenommenen Commits (auch als Spitzen- oder Head-Commit- oder übergeordnete Commit-ID bezeichnet).
 - Gibt an, ob leere Ordner beibehalten werden, wenn die von Ihnen vorgenommenen Änderungen den Inhalt dieser Ordner löschen. Der Standardwert ist "false".
 - Die Informationen zu den Dateien, die Sie hinzufügen, ändern oder löschen möchten.
 - Der Benutzername und die E-Mail-Adresse, die Sie mit diesen Änderungen verknüpfen möchten.
 - Eine Commit-Nachricht, die erklärt, warum Sie diese Änderungen vorgenommen haben.

Der Benutzername, die E-Mail-Adresse und die Commit-Nachricht sind optional. Sie machen anderen Benutzern aber leichter verständlich, von wem die Änderungen aus welchem Grund vorgenommen wurden. Wenn Sie keinen Benutzernamen angeben, wird CodeCommit standardmäßig Ihr IAM-Benutzername oder eine Ableitung Ihres Konsolen-Logins als Autorennamen verwendet.

Zum Beispiel, um einen Commit für ein Repository zu erstellen, der eine README.md Datei zu einem *MyDemoRepo* im Branch benannten Repository hinzufügt. *main* Der Inhalt der Datei ist Base64 und lautet „Willkommen in unserem Team-Repository!“ :

```
aws codecommit create-commit --repository-name MyDemoRepo --  
branch-name main --parent-commit-id 4c925148EXAMPLE --put-files  
"filePath=README.md,fileContent=V2VsY29tZSB0byBvdXIgdGVhbSBByZXBvc2l0b3J5IQo="
```

 Tip

Führen Sie den Befehl [get-branch](#) aus, um die übergeordnete Commit-ID abzurufen.

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{  
  "commitId": "4df8b524-EXAMPLE",  
  "treeId": "55b57003-EXAMPLE",  
  "filesAdded": [  
    {  
      "blobId": "5e1c309dEXAMPLE",  
      "absolutePath": "meeting.md",  
      "fileMode": "NORMAL"  
    }  
  ],  
  "filesDeleted": [],  
  "filesUpdated": []  
}
```

Um einen Commit zu erstellen, der Änderungen an Dateien mit dem Namen *file1.py* und vornimmt *file2.txt*, eine Datei von *picture.png* bis *pictures* umbenennt *image1.png* und sie aus einem Verzeichnis mit dem Namen *images*, verschiebt und eine Datei löscht, die

ExampleSolution.py in einem Repository benannt *MyDemoRepo* ist, *MyFeatureBranch* dessen neuester Commit die ID lautet: *4c925148EXAMPLE*

```
aws codecommit create-commit --repository-name MyDemoRepo --branch-
name MyFeatureBranch --parent-commit-id 4c925148EXAMPLE --author-name "Saanvi
Sarkar"
--email "saanvi_sarkar@example.com" --commit-message "I'm creating this commit to
update a variable name in a number of files."
--keep-empty-folders false --put-files '{"filePath": "file1.py", "fileMode":
"EXECUTABLE", "fileContent": "bucket_name = sys.argv[1] region = sys.argv[2]"}'
'{"filePath": "file2.txt", "fileMode": "NORMAL", "fileContent": "//Adding a comment
to explain the variable changes in file1.py"}' '{"filePath": "images/image1.png",
"fileMode": "NORMAL", "sourceFile": {"filePath": "pictures/picture.png", "isMove":
true}}' --delete-files filePath="ExampleSolution.py"
```

Note

Die Syntax für das `--put-files` Segment hängt von Ihrem Betriebssystem ab. Das obige Beispiel ist für Linux-, MacOS- oder Unix-Benutzer und Windows-Benutzer mit einem Bash-Emulator optimiert. Windows-Benutzer in der Befehlszeile oder in Powershell sollten eine für diese Systeme geeignete Syntax verwenden.

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
  "commitId": "317f8570EXAMPLE",
  "treeId": "347a3408EXAMPLE",
  "filesAdded": [
    {
      "absolutePath": "images/image1.png",
      "blobId": "d68ba6ccEXAMPLE",
      "fileMode": "NORMAL"
    }
  ],
  "filesUpdated": [
    {
      "absolutePath": "file1.py",
      "blobId": "0a4d55a8EXAMPLE",
      "fileMode": "EXECUTABLE"
    },
  ],
}
```

```
{
  "absolutePath": "file2.txt",
  "blobId": "915766bbEXAMPLE",
  "fileMode": "NORMAL"
},
"filesDeleted": [
  {
    "absolutePath": "ExampleSolution.py",
    "blobId": "4f9cebe6aEXAMPLE",
    "fileMode": "EXECUTABLE"
  },
  {
    "absolutePath": "pictures/picture.png",
    "blobId": "fb12a539EXAMPLE",
    "fileMode": "NORMAL"
  }
]
}
```

Commit-Details anzeigen in AWS CodeCommit

Sie können die AWS CodeCommit Konsole verwenden, um den Verlauf der Commits in einem Repository zu durchsuchen. Auf diese Weise können Sie Änderungen an einem Repository erkennen und auch:

- Wann und von wem die Änderungen vorgenommen wurden.
- Wann bestimmte Commits in einen Branch zusammengeführt wurden.

Wenn Sie den Verlauf von Commits für einen Branch anzeigen, erhalten Sie zudem einen besseren Einblick in die Unterschiede zwischen den Branches. Wenn Sie das Markieren verwenden, können Sie außerdem schnell einen Commit anzeigen, der mit einem bestimmten Tag markiert wurde, und Sie sehen dessen übergeordnete Commits. In der Befehlszeile kannst du Git verwenden, um Details zu den Commits in einem lokalen Repo oder einem CodeCommit Repository anzuzeigen.

Durchsuche Commits in einem Repository

Sie können die AWS CodeCommit Konsole verwenden, um den Verlauf der Commits in einem Repository zu durchsuchen. Zudem können Sie eine Grafik der Commits im Repository und in dessen

Branches im Zeitverlauf anzeigen. Auf diese Weise können Sie die Vorgänge des Repository im Verlauf erkennen. Sie sehen auch, wann Änderungen vorgenommen wurden.

Note

Wenn Sie mit dem Befehl `git rebase` einen Rebase für ein Repository ausführen, ändert sich der Verlauf des Repositorys. Commits werden dann möglicherweise nicht mehr in der richtigen Reihenfolge angezeigt. Weitere Informationen finden Sie unter [Git Branching-Rebasing](#) oder in Ihrer Git-Dokumentation.

Themen

- [Den Commit-Verlauf eines Repositorys durchsuchen](#)
- [Sehen Sie sich ein Diagramm des Commit-Verlaufs eines Repositorys an](#)

Den Commit-Verlauf eines Repositorys durchsuchen

Sie können den Commit-Verlauf für einen bestimmten Branch oder Tag des Repositorys durchsuchen, u. a. können Sie Informationen zum Committer und die Commit-Nachricht einsehen. Sie können auch den Code für einen Commit anzeigen.

So durchsuchen Sie den Commit-Verlauf

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories (Repositorys) das Repository aus, für das Sie den Commit-Verlauf prüfen möchten.
3. Wählen Sie im Navigationsbereich Commits aus. In der Ansicht des Commit-Verlaufs wird der Verlauf der Commits für das Repository im Standard-Branch in umgekehrter chronologischer Reihenfolge des Commit-Datums angezeigt. Datum und Uhrzeit entsprechen der Zeitzone Coordinated Universal Time (UTC). Sie können den Commit-Verlauf eines anderen Branches anzeigen, indem Sie auf die Schaltfläche der Ansichtsauswahl klicken und einen anderen Branch in der Liste auswählen. Wenn Sie Tags in Ihrem Repository verwenden, können Sie einen Commit mit einem bestimmten Tag und dessen übergeordnete Commits anzeigen. Klicken Sie dazu auf die Schaltfläche der Ansichtsauswahl.

Developer Tools > CodeCommit > Repositories > MyDemoRepo > Commits

MyDemoRepo

master

Commits | Commit visualizer | Compare commits

Commits

< 1 ... >

Commit ID	Commit message	Commit date	Author	Actions
d615e7ae	Merge branch 'AnotherBranch' into testbranch	5 days ago	Maria Garcia	Copy ID Browse
b6589863	Added another file.	5 days ago	Li Juan	Copy ID Browse
73a6e39c	Merge branch 'master' into testbranch	5 days ago	Maria Garcia	Copy ID Browse
6bbb6d3c	Another test of the editing feature.	5 days ago	Li Juan	Copy ID Browse
edacdfef	Testing this out to see how well it works.	5 days ago	Li Juan	Copy ID Browse
70bb94d7	Revised test results with correct information.	5 days ago	Li Juan	Copy ID Browse
b78e6d1c	Merge branch 'master' into testbranch	5 days ago	Maria Garcia	Copy ID Browse
84b7d158	Edited ahs_count.py	22 days ago	Maria Garcia	Copy ID Browse

4. Um den Unterschied zwischen einem Commit und seinem übergeordneten Commit sowie alle Kommentare zu den Änderungen anzuzeigen, wählen Sie die abgekürzte Commit-ID. Weitere Informationen erhalten Sie unter [Einen Commit mit seinem übergeordneten Element vergleichen](#) und [Kommentar zu einem Commit](#). Wenn Sie den Unterschied zwischen einem Commit und einem beliebigen anderen Commit-Spezifizierer, einschließlich Branch, Tag und Commit-ID, anzeigen möchten, finden Sie unter [Vergleichen von zwei beliebigen Commit-Spezifizierern](#) weitere Informationen.
5. Führen Sie eine oder mehrere der folgenden Aktionen aus:
 - Um das Datum und die Uhrzeit einer Änderung anzuzeigen, bewegen Sie den Mauszeiger über das Commit-Datum.
 - Um die vollständige Commit-ID anzuzeigen, kopieren Sie sie und fügen Sie sie in einem Texteditor oder an anderer Stelle ein. Um sie zu kopieren, wählen Sie Copy ID (ID kopieren) aus.
 - Um den Code so anzuzeigen, wie er zum Zeitpunkt des Commits war, wählen Sie Browse (Durchsuchen) aus. Der Inhalt des Repositorys zum Zeitpunkt des Commits wird in der Ansicht Code angezeigt. Die Schaltfläche der Ansichtsauswahl zeigt die gekürzte Commit-ID an anstelle eines Branches oder Tags.

Sehen Sie sich ein Diagramm des Commit-Verlaufs eines Repositorys an

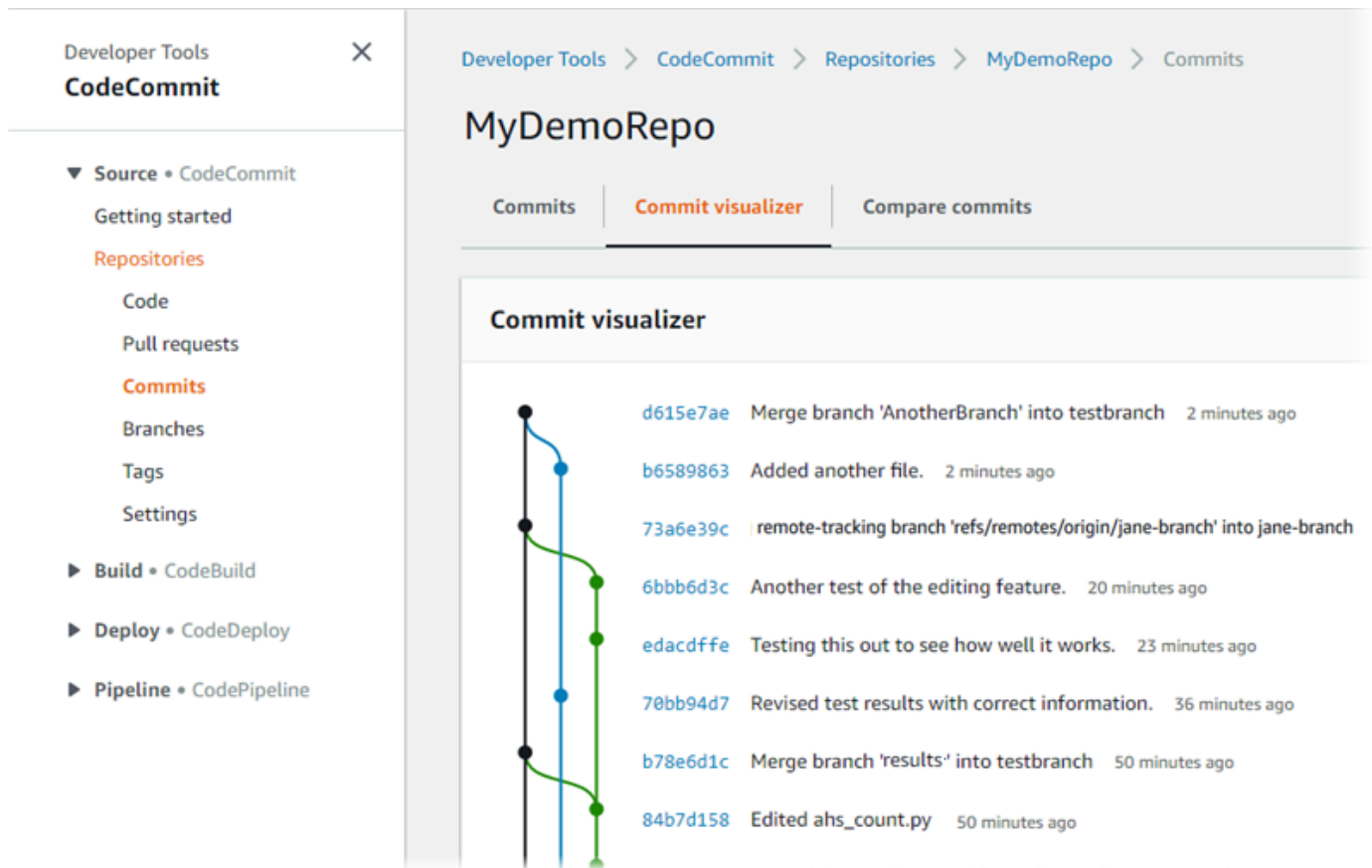
Sie können eine Grafik der Commits anzeigen, die für ein Repository durchgeführt wurden. Die Ansicht Commit Visualizer ist eine DAG-Darstellung (Directed Acyclic Graph, gerichteter azyklischer Graph) aller Commits, die für einen Branch eines Repository durchgeführt wurden. Anhand dieser grafische Darstellung können Sie besser erkennen, wann Commits und verbundene Funktionen hinzugefügt oder zusammengeführt wurden. Sie erhalten damit auch einen genaueren Einblick, wann eine Änderung in Relation zu anderen Änderungen vorgenommen wurde.

Note

Commits, die mithilfe der Fast-Forward-Methode zusammengeführt wurden, werden in der Commit-Grafik nicht in gesonderten Zeilen aufgeführt.

So zeigen Sie eine Commit-Grafik an

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories (Repositorys) das Repository aus, für das Sie eine Commit-Grafik anzeigen möchten.
3. Wählen Sie im Navigationsbereich Commits und dann die Registerkarte Commit Visualizer aus.



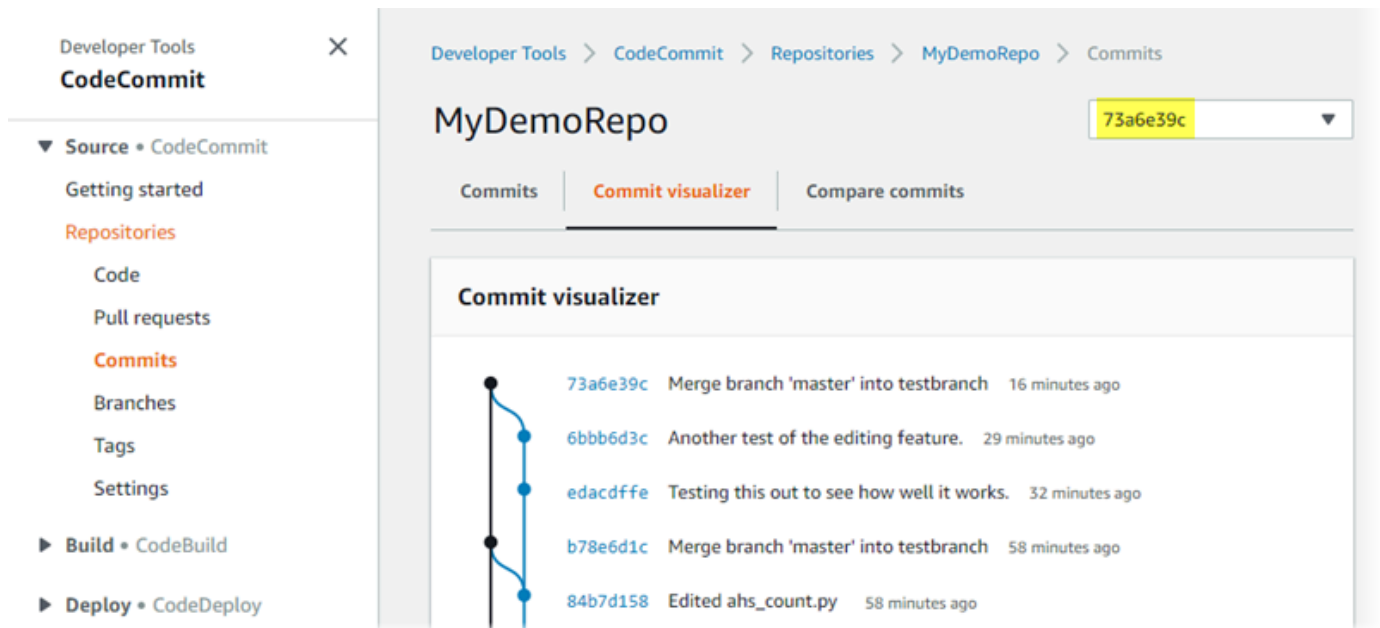
In der Commit-Grafik wird die gekürzte Commit-ID und der Betreff jeder Commit-Nachricht neben diesem Punkt in der Grafik angezeigt.

Note

Die Grafik kann bis zu 35 Branches auf einer Seite anzeigen. Falls es mehr als 35 Branches gibt, ist die Grafik zu komplex für eine Anzeige. Es gibt zwei Methoden zum Vereinfachen der Ansicht:

- Indem Sie die Grafik mithilfe der Schaltfläche der Ansichtsauswahl für einen bestimmten Branch anzeigen.
- Indem Sie eine vollständige Commit-ID in das Suchfeld kopieren, um die Grafik für diesen Commit zu rendern.

4. Um eine neue Grafik von einem Commit zu rendern, wählen Sie den Punkt in der Grafik aus, der diesem Commit entspricht. Die Schaltfläche der Ansichtsauswahl ändert sich in die gekürzte Commit-ID.



Commit-Details anzeigen (AWS CLI)

Mit Git können Sie die Details zu Commits anzeigen. Sie können den auch verwenden AWS CLI , um Details zu den Commits in einem lokalen Repo oder in einem CodeCommit Repository anzuzeigen, indem Sie die folgenden Befehle ausführen:

- Um Informationen zu einem Commit anzuzeigen, führen Sie [aws codecommit get-commit](#) aus.
- Um Informationen zu mehreren Commits anzuzeigen, führen Sie [aws codecommit batch-get-commits](#) aus.
- Führen Sie zum Anzeigen von Informationen zu einem Merge-Commit [aws codecommit get-merge-commit](#) aus.
- Um Informationen zu Änderungen an einem Commit-Spezifizierer (Branch, Tag, HEAD oder andere vollqualifizierte Referenzen wie Commit IDs) anzuzeigen, führen Sie folgenden Befehl aus. [aws codecommit get-differences](#)
- Um den base64-codierten Inhalt eines einzelnen Git-Blob-Objekts in einem Repository anzuzeigen, führen Sie [aws codecommit get-blob](#) aus.

So zeigen Sie Informationen zu einem Commit an

1. Führen Sie den Befehl `aws codecommit get-commit` aus und geben Sie Folgendes an:

- Der Name des CodeCommit Repositorys (mit der `--repository-name` Option).
- Die vollständige Commit-ID

Um beispielsweise Informationen über einen Commit mit der ID `317f8570EXAMPLE` in einem CodeCommit Repository mit dem Namen `MyDemoRepo` anzuzeigen:

```
aws codecommit get-commit --repository-name MyDemoRepo --commit-id
317f8570EXAMPLE
```

2. Ist der Befehl erfolgreich, enthält die Ausgabe Folgendes:

- Informationen zum Autor des Commits (wie in Git konfiguriert), einschließlich des Datums im Zeitstempelformat und der Abweichung zur Coordinated Universal Time (UTC).
- Informationen zum Committer (wie in Git konfiguriert), einschließlich des Datums im Zeitstempelformat und der Abweichung zur UTC.
- Die ID der Git-Struktur, in der sich der Commit befindet.
- Die Commit-ID des übergeordneten Commits.
- Die Commit-Nachricht.

Es folgt eine Beispielausgabe basierend auf dem vorangehenden Beispielbefehl:

```
{
  "commit": {
    "additionalData": "",
    "committer": {
      "date": "1484167798 -0800",
      "name": "Mary Major",
      "email": "mary_major@example.com"
    },
    "author": {
      "date": "1484167798 -0800",
      "name": "Mary Major",
      "email": "mary_major@example.com"
    },
    "treeId": "347a3408EXAMPLE",
    "parents": [
      "4c925148EXAMPLE"
    ],

```



```
    "message": "Fix incorrect variable name"
  }
}
```

So zeigen Sie Informationen zu einem Commit für die Zusammenführung an

1. Führen Sie den Befehl `get-merge-commit` aus und geben Sie Folgendes an:

- Einen Commit-Spezifizierer für die Quelle der Zusammenführung (mit der Option `--source-commit-specifier`).
- Einen Commit-Spezifizierer für das Ziel der Zusammenführung (mit der Option `--destination-commit-specifier`).
- Die Zusammenführungsoption, die Sie verwenden möchten (mit der Option `--merge-option`).
- Der Name des Repositorys (mit der Option `--repository-name`).

Um beispielsweise Informationen über einen Merge-Commit für den Quell-Branch *bugfix-bug1234* mit einem Ziel-Branch anzuzeigen, der die *THREE_WAY_MERGE* Strategie in einem Repository mit dem Namen *main* verwendet *MyDemoRepo*:

```
aws codecommit get-merge-commit --source-commit-specifier bugfix-bug1234 --
destination-commit-specifier main --merge-option THREE_WAY_MERGE --repository-
name MyDemoRepo
```

2. Ist der Befehl erfolgreich, gibt seine Ausgabe Informationen zurück, die wie folgt aussehen sollten:

```
{
  "sourceCommitId": "c5709475EXAMPLE",
  "destinationCommitId": "317f8570EXAMPLE",
  "baseCommitId": "fb12a539EXAMPLE",
  "mergeCommitId": "ffc4d608eEXAMPLE"
}
```

So zeigen Sie Informationen zu mehreren Commits an

1. Führen Sie den Befehl `batch-get-commits` aus und geben Sie Folgendes an:

- Der Name des CodeCommit Repositorys (mit der `--repository-name` Option).
- Eine Liste der vollständigen Commits IDs für jeden Commit, zu dem Sie Informationen anzeigen möchten.

Um beispielsweise Informationen über Commits mit dem IDs 317f8570EXAMPLE und 4c925148EXAMPLE in einem CodeCommit Repository mit dem Namen MyDemoRepo anzuzeigen:

```
aws codecommit batch-get-commits --repository-name MyDemoRepo --commit-ids
317f8570EXAMPLE 4c925148EXAMPLE
```

2. Ist der Befehl erfolgreich, enthält die Ausgabe Folgendes:

- Informationen zu den Autoren des Commits (wie in Git konfiguriert), einschließlich des Datums im Zeitstempelformat und der Abweichung zur Coordinated Universal Time (UTC).
- Informationen zu Committern (wie in Git konfiguriert), einschließlich des Datums im Zeitstempelformat und der Abweichung zur UTC.
- Der IDs des Git-Baums, in dem der Commit existiert.
- Der Commit IDs des übergeordneten Commits.
- Die Commit-Nachrichten.

Es folgt eine Beispielausgabe basierend auf dem vorangehenden Beispielbefehl:

```
{
  "commits": [
    {
      "additionalData": "",
      "committer": {
        "date": "1508280564 -0800",
        "name": "Mary Major",
        "email": "mary_major@example.com"
      },
      "author": {
        "date": "1508280564 -0800",
        "name": "Mary Major",
        "email": "mary_major@example.com"
      },
      "commitId": "317f8570EXAMPLE",
```

```

    "treeId": "1f330709EXAMPLE",
    "parents": [
        "6e147360EXAMPLE"
    ],
    "message": "Change variable name and add new response element"
},
{
    "additionalData": "",
    "committer": {
        "date": "1508280542 -0800",
        "name": "Li Juan",
        "email": "li_juan@example.com"
    },
    "author": {
        "date": "1508280542 -0800",
        "name": "Li Juan",
        "email": "li_juan@example.com"
    },
    "commitId": "4c925148EXAMPLE",
    "treeId": "1f330709EXAMPLE",
    "parents": [
        "317f8570EXAMPLE"
    ],
    "message": "Added new class"
}
}

```

So zeigen Sie Informationen zu den Änderungen eines Commit-Spezifizierers an

1. Führen Sie den Befehl `aws codecommit get-differences` aus und geben Sie Folgendes an:

- Der Name des CodeCommit Repositorys (mit der `--repository-name` Option).
- Der Commit-Spezifizierer, zu dem Sie Informationen erhalten möchten. Nur `--after-commit-specifier` ist erforderlich. Wenn Sie `--before-commit-specifier` nicht angeben, werden alle aktuellen Dateien ab `--after-commit-specifier` angezeigt.

Um beispielsweise Informationen zu den Unterschieden zwischen Commits mit dem IDs `317f8570EXAMPLE` und `4c925148EXAMPLE` in einem CodeCommit Repository mit dem Namen `MyDemoRepo` anzuzeigen:

```
aws codecommit get-differences --repository-name MyDemoRepo --before-commit-specifier 317f8570EXAMPLE --after-commit-specifier 4c925148EXAMPLE
```

2. Ist der Befehl erfolgreich, enthält die Ausgabe Folgendes:

- Eine Liste der Unterschiede, einschließlich Änderungstyp: "A" für Added (hinzugefügt), "D" für Deleted (gelöscht) oder "M" für Modified (geändert).
- Der Modus des Dateiänderungstyps.
- Die ID des Git-Blobs, das die Änderung enthält.

Es folgt eine Beispielausgabe basierend auf dem vorangehenden Beispielbefehl:

```
{
  "differences": [
    {
      "afterBlob": {
        "path": "blob.txt",
        "blobId": "2eb4af3bEXAMPLE",
        "mode": "100644"
      },
      "changeType": "M",
      "beforeBlob": {
        "path": "blob.txt",
        "blobId": "bf7fcf28fEXAMPLE",
        "mode": "100644"
      }
    }
  ]
}
```

So zeigen Sie Informationen zu einem Git-Blob an

1. Führen Sie den Befehl `aws codecommit get-blob` aus und geben Sie Folgendes an:

- Der Name des CodeCommit Repositorys (mit der `--repository-name` Option).
- Die ID des Git-Blobs (mit der Option `--blob-id`).

Um beispielsweise Informationen über einen Git-Blob mit der ID von 2eb4af3bEXAMPLE in einem CodeCommit Repository mit dem Namen MyDemoRepo anzuzeigen:

```
aws codecommit get-blob --repository-name MyDemoRepo --blob-id 2eb4af3bEXAMPLE
```

2. Ist der Befehl erfolgreich, enthält die Ausgabe Folgendes:

- Der base64-kodierte Inhalt des Blobs ist in der Regel eine Datei.

Beispielsweise könnte die Ausgabe des vorherigen Befehls etwa so aussehen:

```
{
  "content": "QSBCaW5hcnkgTGFGyToEXAMPLE="
}
```

Commit-Details anzeigen (Git)

Bevor Sie diese Schritte ausführen, sollten Sie das lokale Repository bereits mit dem CodeCommit Repository verbunden und die Änderungen übernommen haben. Detaillierte Anweisungen finden Sie unter [Herstellen einer Verbindung mit einem Repository](#).

Um die Änderungen für den letzten Commit in ein Repository anzuzeigen, führen Sie den `git show` Befehl aus.

```
git show
```

Die Ausgabe des Befehls ähnelt der folgenden:

```
commit 4f8c6f9d
Author: Mary Major <mary.major@example.com>
Date:   Mon May 23 15:56:48 2016 -0700

    Added bumblebee.txt

diff --git a/bumblebee.txt b/bumblebee.txt
new file mode 100644
index 0000000..443b974
--- /dev/null
```

```
+++ b/bumblebee.txt
@@ -0,0 +1 @@
+A bumblebee, also written bumble bee, is a member of the bee genus Bombus, in the
  family Apidae.
\ No newline at end of file
```

Note

In diesem und den folgenden Beispielen wurde Commit IDs abgekürzt. Der vollständige Commit IDs wird nicht angezeigt.

Sie können den Befehl `git show` mit der Commit-ID verwenden, um die vorgenommenen Änderungen anzuzeigen:

```
git show 94ba1e60

commit 94ba1e60
Author: John Doe <johndoe@example.com>
Date:   Mon May 23 15:39:14 2016 -0700

    Added horse.txt

diff --git a/horse.txt b/horse.txt
new file mode 100644
index 0000000..080f68f
--- /dev/null
+++ b/horse.txt
@@ -0,0 +1 @@
+The horse (Equus ferus caballus) is one of two extant subspecies of Equus ferus.
```

Um die Unterschiede zwischen zwei Commits zu sehen, führe den `git diff` Befehl aus und füge die beiden Commits IDs hinzu.

```
git diff ce22850d 4f8c6f9d
```

In diesem Beispiel besteht der Unterschied zwischen den Commits darin, dass zwei Dateien hinzugefügt wurden. Die Ausgabe des Befehls ähnelt der folgenden:

```
diff --git a/bees.txt b/bees.txt
new file mode 100644
```

```
index 00000000..cf57550
--- /dev/null
+++ b/bees.txt
@@ -0,0 +1 @@
+Bees are flying insects closely related to wasps and ants, and are known for their
  role in pollination and for producing honey and beeswax.
diff --git a/bumblebee.txt b/bumblebee.txt
new file mode 100644
index 00000000..443b974
--- /dev/null
+++ b/bumblebee.txt
@@ -0,0 +1 @@
+A bumblebee, also written bumble bee, is a member of the bee genus Bombus, in the
  family Apidae.
\ No newline at end of file
```

Um Git zu verwenden, um Details zu den Commits in einem lokalen Repo anzuzeigen, führe den git log folgenden Befehl aus:

```
git log
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
commit 94ba1e60
Author: John Doe <johndoe@example.com>
Date:   Mon May 23 15:39:14 2016 -0700

    Added horse.txt

commit 4c925148
Author: Jane Doe <janedoe@example.com>
Date:   Mon May 22 14:54:55 2014 -0700

    Added cat.txt and dog.txt
```

Um nur Commit IDs und Nachrichten anzuzeigen, führe den git log --pretty=oneline folgenden Befehl aus:

```
git log --pretty=oneline
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
94ba1e60 Added horse.txt
4c925148 Added cat.txt and dog.txt
```

Weitere Optionen findest du in deiner Git-Dokumentation.

Vergleiche Commits in AWS CodeCommit

Sie können die CodeCommit Konsole verwenden, um die Unterschiede zwischen Commit-Spezifizierern in einem CodeCommit Repository anzuzeigen. Sie können den Unterschied zwischen einem Commit und dem übergeordneten Commit auf einen Blick erkennen. Sie können auch zwei beliebige Referenzen vergleichen, einschließlich Commit IDs.

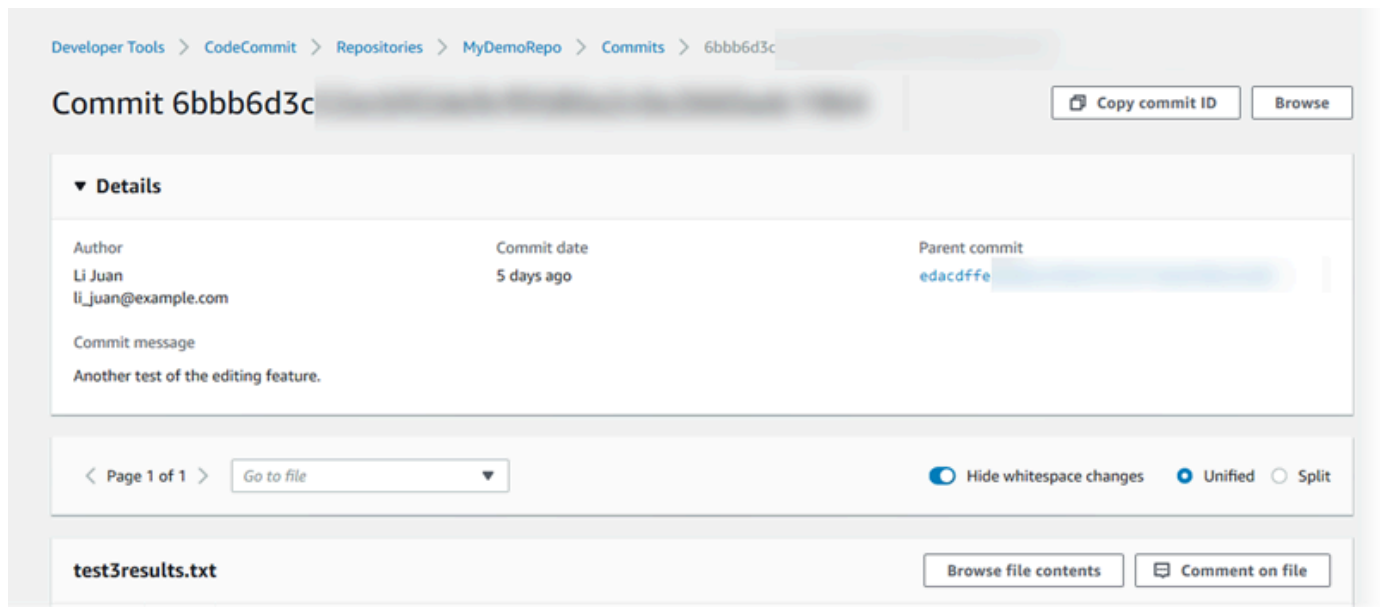
Themen

- [Einen Commit mit seinem übergeordneten Element vergleichen](#)
- [Vergleichen von zwei beliebigen Commit-Spezifizierern](#)

Einen Commit mit seinem übergeordneten Element vergleichen

Sie können den Unterschied zwischen einem Commit und dem übergeordneten Commit auf einen Blick erkennen und die Commit-Nachricht, den Committer und die Änderung prüfen.

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie auf der Seite Repositories (Repositorys) das Repository aus, bei dem Sie den Unterschied zwischen einem Commit und dem übergeordneten Commit anzeigen möchten.
3. Wählen Sie im Navigationsbereich Commits aus.
4. Wählen Sie die gekürzte Commit-ID eines beliebigen Commits in der Liste aus. Die Ansicht wird geändert und zeigt nun Details zu diesem Commit an, darunter auch die Unterschiede zum übergeordneten Commit.



Sie können die Änderungen nebeneinander (Ansicht Split) oder inline (Ansicht Unified) anzeigen. Außerdem können Sie geänderte Leerzeichen ein- oder ausblenden. Sie können auch Kommentare hinzufügen. Weitere Informationen finden Sie unter [Kommentar zu einem Commit](#).

Note

Ihre Präferenzen für die Anzeige des Codes sowie andere Konsoleneinstellungen werden als Browser-Cookies gespeichert, wenn Sie sie ändern. Weitere Informationen finden Sie unter [Mit Benutzereinstellungen arbeiten](#).

[Developer Tools](#) > [CodeCommit](#) > [Repositories](#) > [MyDemoRepo](#) > [Commits](#) > 7d09e44c

Commit 7d09e44c

 Copy commit ID

Browse

▼ Details

Author	Commit date	Parent commit
Mary Major mary_major@example.com	48 minutes ago	e6aca768
Commit message		
Adding a readme file to the repository.		

< Page 1 of 1 >

Go to file ▼

☒ Hide whitespace changes

☒ Unified

☐ Split

readme.md

Browse file contents

 Comment on file

1

- This is a readme file that provides a basic description of what's in this repository.

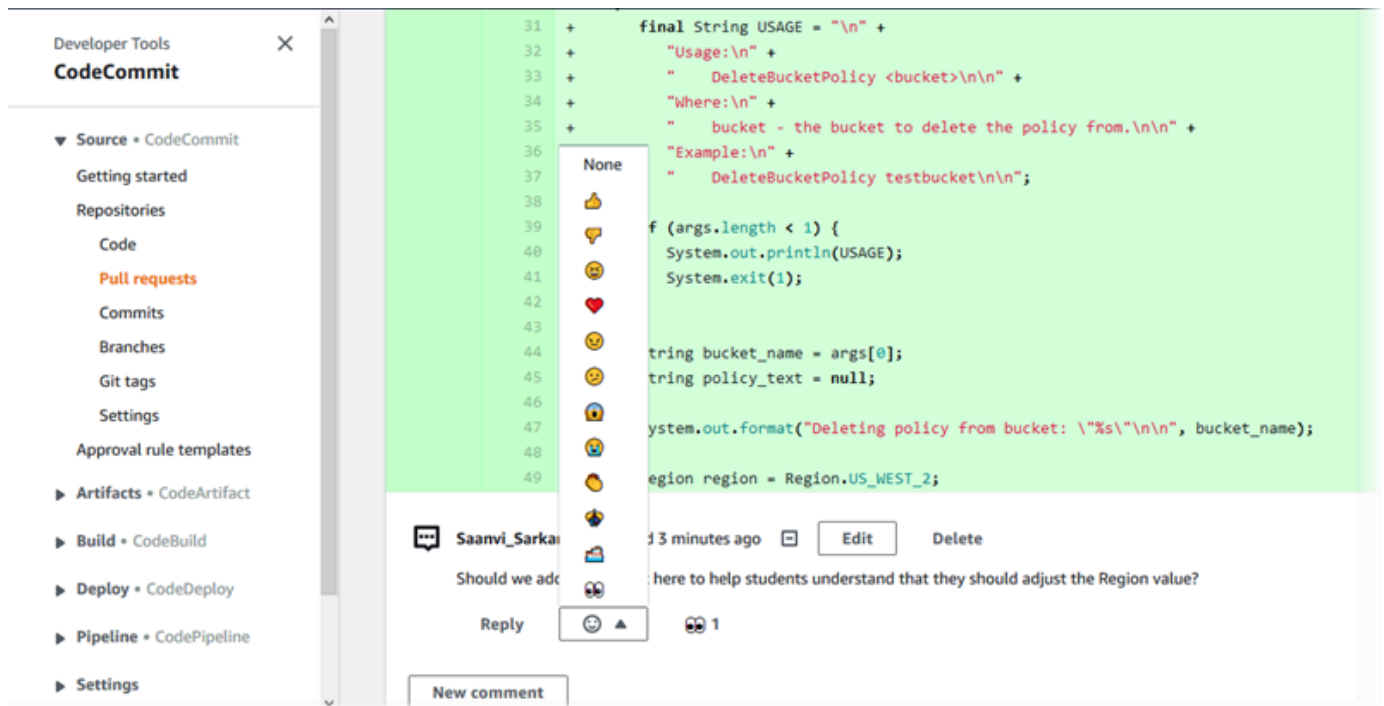
\ No newline at end of file

1

+ Use this repository for code changes to the *Demo* project. The default branch is *master*. Cod

\ No newline at end of file

< >



Note

Abhängig von der Art der Zeilenenden, Ihrem Codeeditor und weiteren Faktoren sehen Sie möglicherweise nicht die spezifischen Änderungen in einer Zeile, sondern ganze hinzugefügte oder gelöschte Zeilen. Die Details entsprechen dem, was der Befehl `git show` oder `git diff` zurückgibt.

5. Für den Vergleich eines Commits mit dem übergeordneten Commit von der Commit Visualizer-Registerkarte wählen Sie die abgekürzte Commit-ID. Die Commit-Details werden angezeigt, einschließlich der Änderungen zwischen dem Commit und dem übergeordneten Commit.

Vergleichen von zwei beliebigen Commit-Spezifizierern

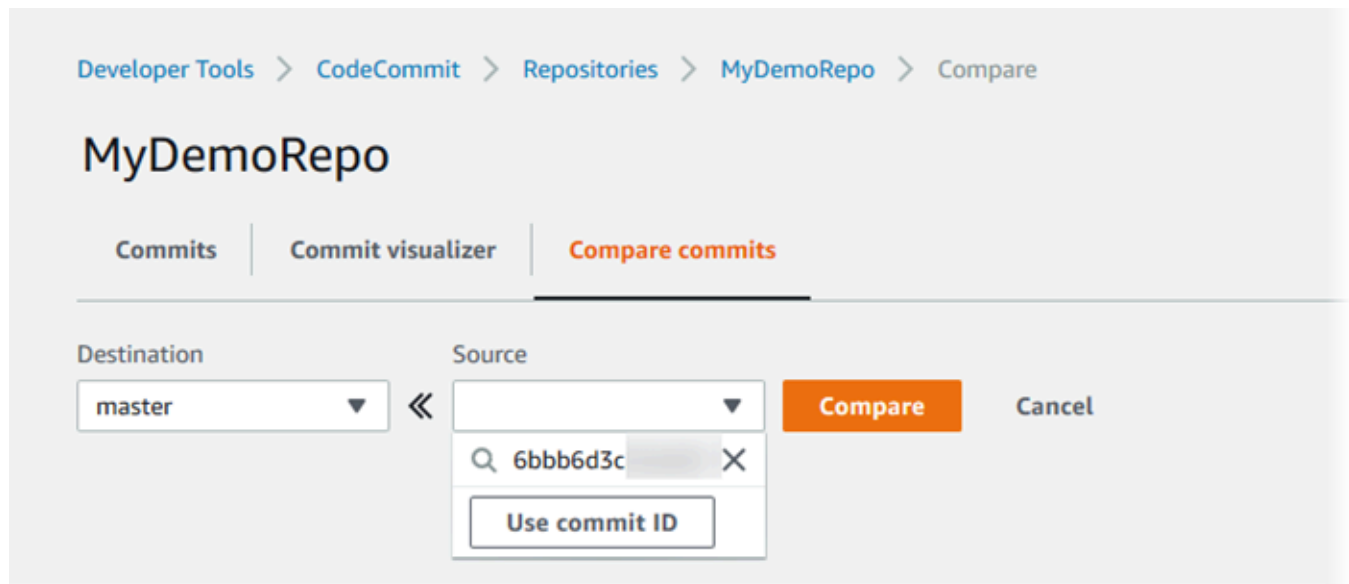
Sie können die Unterschiede zwischen zwei beliebigen Commit-Spezifizierern in der CodeCommit Konsole einsehen. Commit-Spezifizierer sind Verweise wie Branches, Tags und Commit. IDs

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie auf der Seite Repositories (Repositorys) das Repository aus, bei dem Sie Commits, Branches oder Commits mit Tags vergleichen möchten.

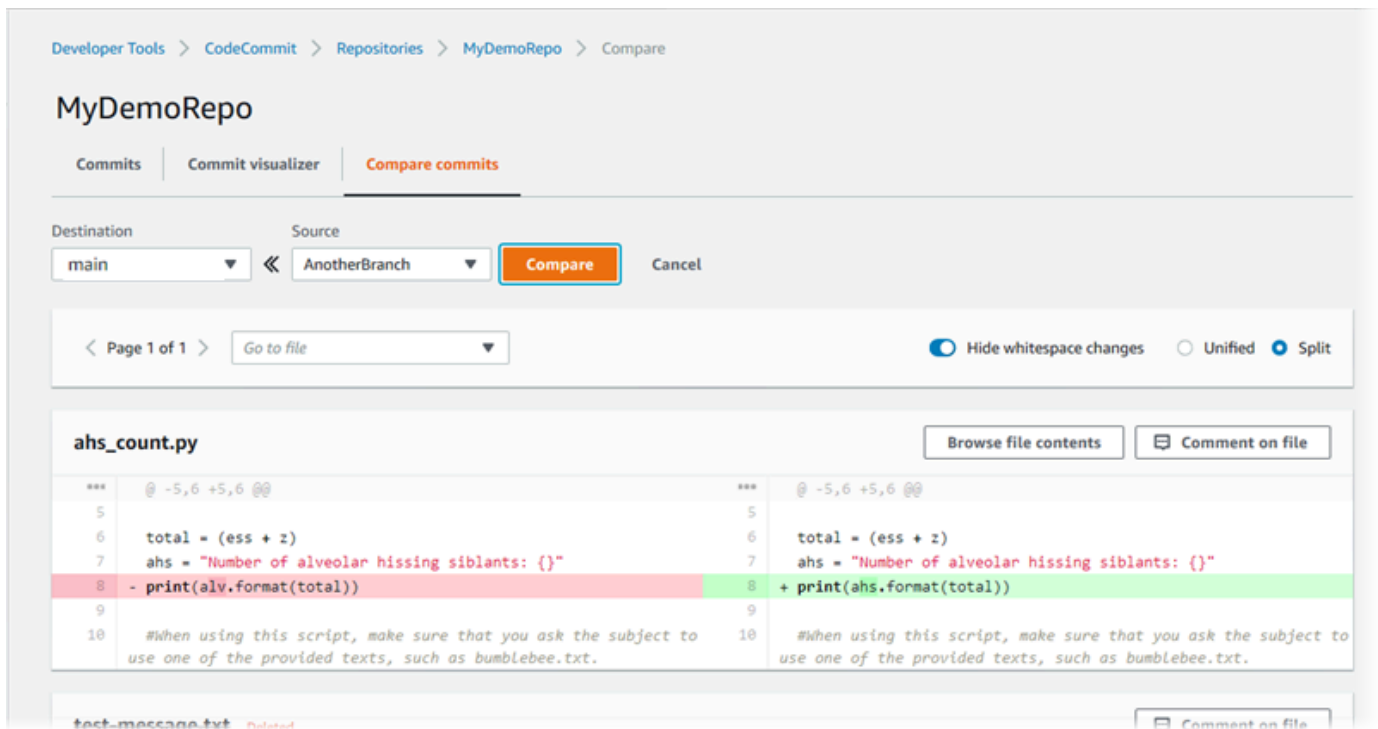
3. Wählen Sie im Navigationsbereich Commits und dann Compare commits (Commits vergleichen) aus.

The screenshot shows the AWS CodeCommit web interface for a repository named 'MyDemoRepo'. At the top, a breadcrumb trail reads: [Developer Tools](#) > [CodeCommit](#) > [Repositories](#) > [MyDemoRepo](#) > [Compare](#). Below this, the repository name 'MyDemoRepo' is displayed in a large font. Underneath the name are three tabs: 'Commits', 'Commit visualizer', and 'Compare commits', with the third tab being the active one. The main area contains two dropdown menus labeled 'Destination' and 'Source', separated by a double-left arrow icon. To the right of these dropdowns are two buttons: an orange 'Compare' button and a grey 'Cancel' button.

4. Verwenden Sie die Felder, um zwei Commit-Spezifizierer zu vergleichen.
- Wählen Sie den Branch-Namen aus der Liste aus, um den obersten Commit vom Branch zu vergleichen. Damit wird der neueste Commit von diesem Branch für den Vergleich ausgewählt.
 - Wählen Sie den Tag-Namen aus der Liste aus, sofern vorhanden, um einen Commit mit einem spezifischen zugeordneten Tag zu vergleichen. Damit wird der Commit mit Tag für den Vergleich ausgewählt.
 - Geben oder fügen Sie die Commit-ID in das Feld ein, um einen bestimmten Commit zu vergleichen. Für die vollständige Commit-ID wählen Sie Commits in der Navigationsleiste aus, dann kopieren Sie die Commit-ID aus der Liste. Fügen Sie auf der Seite Compare commits (Commits vergleichen) die vollständige Commit-ID in das Textfeld ein und wählen Sie Use commit ID (Commit-ID verwenden) aus.



5. Wenn Sie die Spezifizierer ausgewählt haben, wählen Sie Compare aus.



Sie können die Unterschiede nebeneinander (Ansicht Split) oder inline (Ansicht Unified) anzeigen. Außerdem können Sie geänderte Leerzeichen ein- oder ausblenden.

6. Um die Auswahl für den Vergleich zu löschen, wählen Sie Cancel (Abbrechen) aus.

Kommentieren Sie einen Commit in AWS CodeCommit

Sie können die CodeCommit Konsole verwenden, um Commits in einem Repository zu kommentieren und Kommentare anderer Benutzer zu Commits anzusehen und darauf zu antworten. Auf diese Weise können Sie Änderungen an einem Repository diskutiert werden, unter anderem:

- Warum Änderungen vorgenommen wurden.
- Ob weitere Änderungen erforderlich sind.
- Ob Änderungen mit einem anderen Branch zusammengeführt werden sollen.

Sie können einen allgemeinen Commit, eine Datei in einem Commit oder eine bestimmte Zeile oder Änderung in einer Datei kommentieren. Sie können auch auf eine Codezeile verlinken, indem Sie die Zeile auswählen und dann die resultierende URL in Ihren Browser kopieren.

Note

Die besten Ergebnisse erzielen Sie, wenn Sie das Kommentieren verwenden, wenn Sie als IAM-Benutzer angemeldet sind. Die Kommentarfunktion ist nicht optimiert für Benutzer, die sich mit Anmeldeinformationen des Root-Kontos, über einen verbundenen Zugriff oder mit temporären Anmeldeinformationen anmelden.

Themen

- [Kommentare zu einem Commit in einem Repository anzeigen](#)
- [Kommentare zu einem Commit in einem Repository hinzufügen und darauf antworten](#)
- [Kommentare anzeigen, hinzufügen, aktualisieren und beantworten \(\)AWS CLI](#)

Kommentare zu einem Commit in einem Repository anzeigen

Sie können die CodeCommit Konsole verwenden, um Kommentare zu einem Commit anzuzeigen.

So zeigen Sie Kommentare zu einem Commit an

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.

2. Wählen Sie unter Repositories (Repositorys) das Repository aus, für das Sie Kommentare zu Commits prüfen möchten.
3. Wählen Sie im Navigationsbereich Commits aus. Wählen Sie die Commit-ID des Commits aus, in dem Sie Kommentare anzeigen möchten.

Die Seite für den betreffenden Commit wird mit den vorhandenen Kommentaren angezeigt.

Kommentare zu einem Commit in einem Repository hinzufügen und darauf antworten

Sie können die CodeCommit Konsole verwenden, um Kommentare zum Vergleich zwischen einem Commit und einem übergeordneten Commit oder zum Vergleich zwischen zwei angegebenen Commits hinzuzufügen. Du kannst auf Kommentare auch mit Emojis, mit deinen eigenen Kommentaren oder beidem antworten.

Kommentare zu einem Commit hinzufügen und darauf antworten (Konsole)

Sie können Kommentare zu einem Commit mit Text und Emojis hinzufügen und darauf antworten. Ihre Kommentare und Emojis sind so gekennzeichnet, dass sie dem IAM-Benutzer oder der IAM-Rolle gehören, mit der Sie sich bei der Konsole angemeldet haben.

So fügen Sie Kommentare zu einem Commit hinzu und antworten darauf

1. [Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home](https://console.aws.amazon.com/codesuite/codecommit/home).
2. Wählen Sie unter Repositories (Repositorys) das Repository aus, in dem Sie Commits kommentieren möchten.
3. Wählen Sie im Navigationsbereich Commits aus. Wählen Sie die Commit-ID des Commits aus, in dem Sie Kommentare hinzufügen oder beantworten möchten.

Die Seite für den betreffenden Commit wird mit den vorhandenen Kommentaren angezeigt.

4. Um einen Kommentar hinzuzufügen, gehen Sie wie folgt vor:
 - Zum Hinzufügen eines allgemeinen Kommentars geben Sie in Comments on changes (Kommentare zu Änderungen) einen Kommentar ein und wählen dann Save (Speichern) aus. Sie können [Markdown](#) verwenden oder Ihren Kommentar als Klartext eingeben.

Comments on changes

New comment

☐ Preview markdown [Learn more](#)

Did we also change the variable name in blf.py and concat.py?

Save

- Um einer Datei im Commit einen Kommentar hinzuzufügen, suchen Sie den Namen der Datei. Wählen Sie Comment on file (Kommentar zu Datei) aus, geben Sie Ihren Kommentar ein und wählen Sie dann Save (Speichern) aus.

ahs_count.py

[Browse file contents](#) [Comment on file](#)

New comment

☐ Preview markdown [Learn more](#)

Save

Cancel

- Um einer geänderten Zeile in dem Commit einen Kommentar hinzuzufügen, suchen Sie die Zeile mit der Änderung. Wählen Sie die Kommentarblase



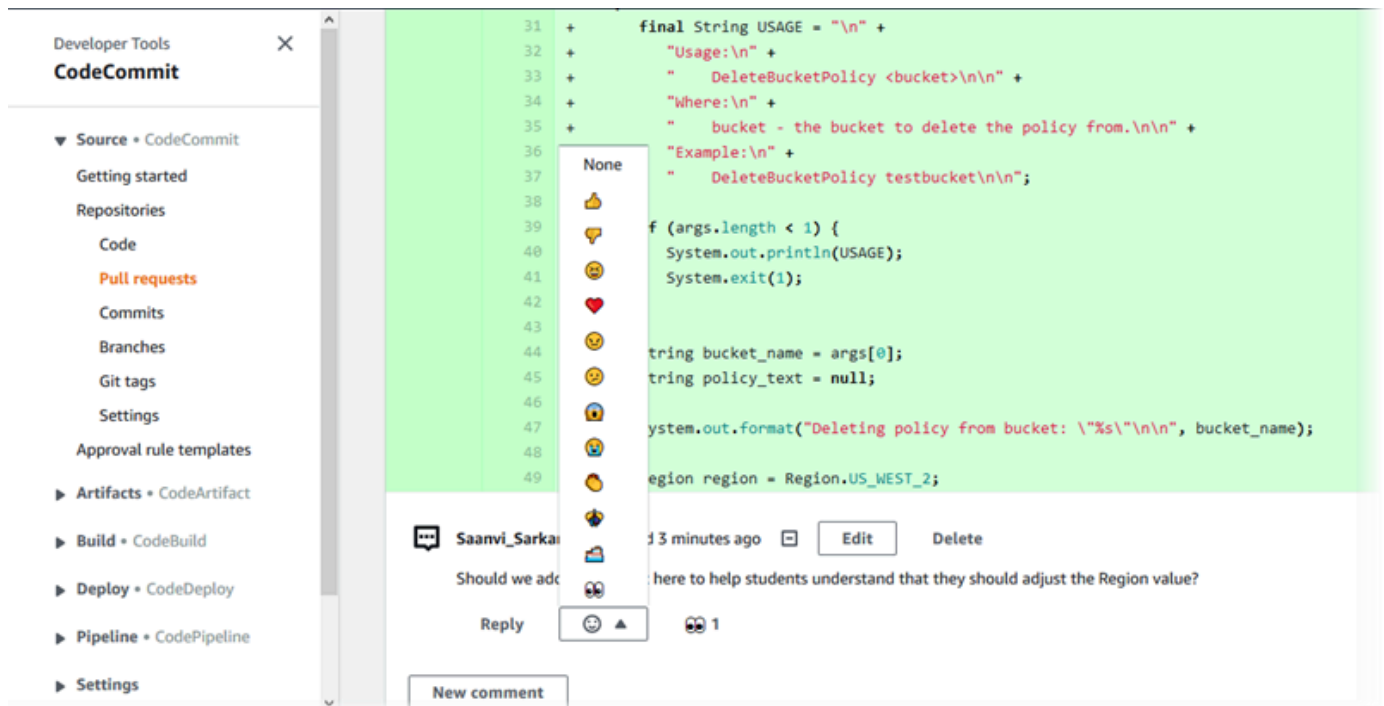
aus, geben Sie Ihren Kommentar ein und wählen Sie dann Save (Speichern) aus.



Note

Sie können Ihren Kommentar nach dem Speichern bearbeiten. Sie können den Inhalt auch löschen. Der Kommentar bleibt mit einer Meldung erhalten, dass der Inhalt gelöscht wurde. Verwenden Sie gegebenenfalls den Modus Preview markdown (Markdown-Vorschau) für Ihren Kommentar, bevor Sie ihn speichern.

5. Wählen Sie Reply (Antworten) aus, um Kommentare zu einem Commit zu beantworten. Um auf einen Kommentar mit einem Emoji zu antworten, wählen Sie das gewünschte Emoji aus der Liste aus. Du kannst nur ein Emoji pro Kommentar auswählen. Wenn du deine Emoji-Reaktion ändern möchtest, wähle ein anderes aus der Liste oder wähle Keine, um deine Reaktion zu entfernen.

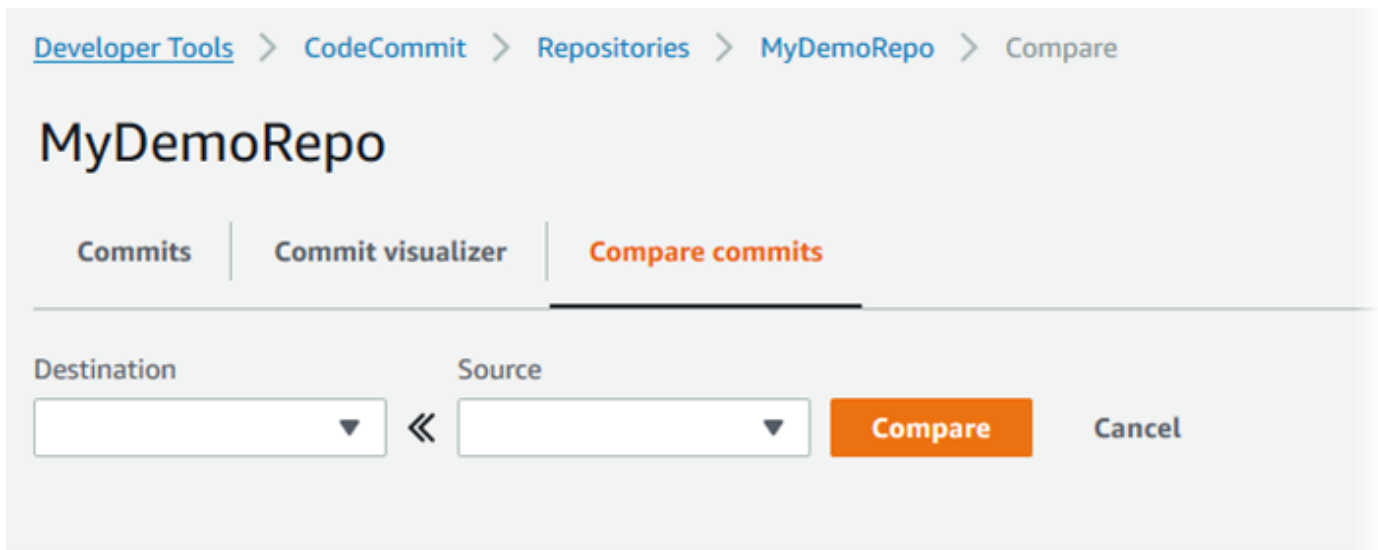


Füge Kommentare hinzu und beantworte sie, wenn du zwei Commit-Spezifizierer vergleichst

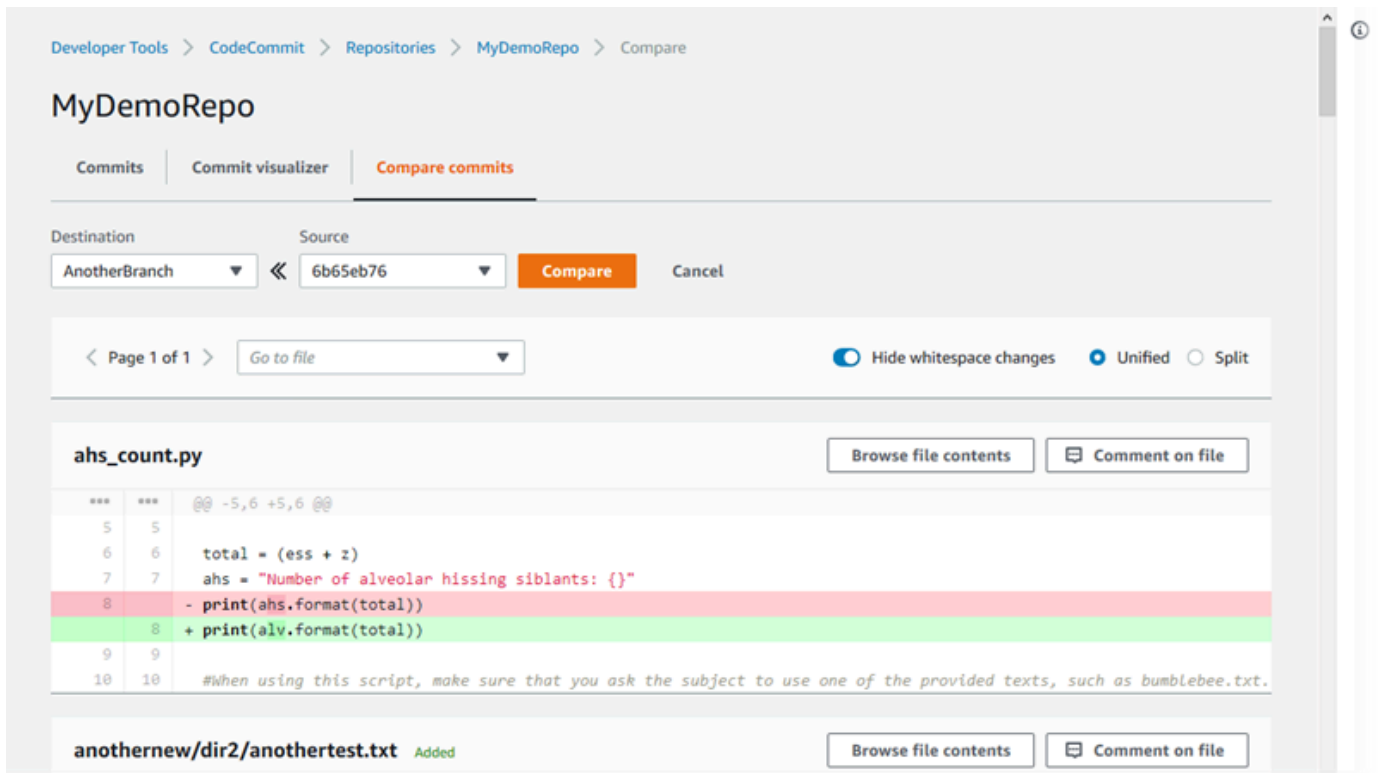
Sie können einem Vergleich zwischen Branches, Tags oder Commits Kommentare hinzufügen.

Beim Vergleich von Commit-Kennungen Kommentare hinzufügen und darauf antworten

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories (Repositoryys) das Repository aus, in dem Sie Commits, Branches oder Commits mit Tags vergleichen möchten.
3. Wählen Sie im Navigationsbereich Commits und dann die Compare commits (Commits vergleichen)-Registerkarte aus.



4. Verwenden Sie die Felder Ziel und Quelle, um zwei Commit-Kennungen zu vergleichen. Verwenden Sie die Drop-down-Listen oder fügen Sie Commit ein. IDs Wählen Sie Compare aus.



5. Führen Sie eine oder mehrere der folgenden Aktionen aus:
- Um Kommentare für Dateien oder Zeilen einzufügen, wählen Sie die Kommentarblase  aus.

- Um allgemeine Kommentare zu den verglichenen Änderungen hinzuzufügen, wählen Sie `Comments on changes`.

Kommentare anzeigen, hinzufügen, aktualisieren und beantworten ()AWS CLI

Sie können den Inhalt eines Kommentars mit den folgenden Befehlen anzeigen, hinzufügen, beantworten, aktualisieren und löschen:

- Um die Kommentare zum Vergleich zwischen zwei Commits anzuzeigen, führen Sie [get-comments-for-compared-commit](#) aus.
- Um Details zu einem bestimmten Kommentar anzuzeigen, führen Sie [get-comment](#) aus.
- Um den Inhalt eines Kommentars zu löschen, den Sie erstellt haben, führen Sie [delete-comment-content](#) aus.
- Um einen Kommentar zum Vergleich zwischen zwei Commits zu erstellen, führen Sie [post-comment-for-compared-commit](#) aus.
- Um einen Kommentar zu aktualisieren, führen Sie [update-comment](#) aus.
- Um auf einen Kommentar zu antworten, führen Sie [post-comment-reply](#) den Befehl aus.
- Um auf einen Kommentar mit einem Emoji zu antworten, führe [put-comment-reaction](#) den Befehl aus.
- Um Emoji-Reaktionen auf einen Kommentar anzuzeigen, führe den Befehl aus. [get-comment-reactions](#)

So zeigen Sie Kommentare zu einem Commit an

1. Führen Sie den Befehl `get-comments-for-compared-commit` aus und geben Sie Folgendes an:
 - Der Name des CodeCommit Repositorys (mit der `--repository-name` Option).
 - Die vollständige Commit-ID des AFTER-Commits, um die Richtung des Vergleichs festzulegen (mit der `--after-commit-id` option).
 - Die vollständige Commit-ID des BEFORE-Commits, um die Richtung des Vergleichs festzulegen (mit der `--before-commit-id`-Option).
 - (Optional) Ein Aufzählungs-Token zum Zurückgeben des nächsten Ergebnisstapels (mit der Option `--next-token`).

- (Optional) Eine nicht negative ganze Zahl, um die Anzahl der zurückgegebenen Ergebnisse zu begrenzen (mit der Option `--max-results`).

So zeigen Sie beispielsweise Kommentare zu dem Vergleich zwischen zwei Commits in einem Repository mit dem Namen *MyDemoRepo* an:

```
aws codecommit get-comments-for-compared-commit --repository-name MyDemoRepo --
before-commit-id 6e147360EXAMPLE --after-commit-id 317f8570EXAMPLE
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
  "commentsForComparedCommitData": [
    {
      "afterBlobId": "1f330709EXAMPLE",
      "afterCommitId": "317f8570EXAMPLE",
      "beforeBlobId": "80906a4cEXAMPLE",
      "beforeCommitId": "6e147360EXAMPLE",
      "comments": [
        {
          "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
          "clientRequestToken": "123Example",
          "commentId": "ff30b348EXAMPLEb9aa670f",
          "content": "Whoops - I meant to add this comment to the line, not
the file, but I don't see how to delete it.",
          "creationDate": 1508369768.142,
          "deleted": false,
          "CommentId": "123abc-EXAMPLE",
          "lastModifiedDate": 1508369842.278,
          "callerReactions": [],
          "reactionCounts":
            {
              "SMILE" : 6,
              "THUMBSUP" : 1
            }
        },
        {
          "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
          "clientRequestToken": "123Example",
          "commentId": "553b509bEXAMPLE56198325",
          "content": "Can you add a test case for this?",
          "creationDate": 1508369612.240,
```

```

        "deleted": false,
        "commentId": "456def-EXAMPLE",
        "lastModifiedDate": 1508369612.240,
        "callerReactions": [],
        "reactionCounts":
        {
            "THUMBSUP" : 2
        }
    },
    "location": {
        "filePath": "cl_sample.js",
        "filePosition": 1232,
        "relativeFileVersion": "after"
    },
    "repositoryName": "MyDemoRepo"
},
"nextToken": "exampleToken"
}

```

So zeigen Sie die Details eines Kommentars zu einem Commit an

1. Führen Sie den Befehl `get-comment` aus und geben Sie die vom System generierte Kommentar-ID an. Zum Beispiel:

```
aws codecommit get-comment --comment-id ff30b348EXAMPLEb9aa670f
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```

{
  "comment": {
    "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
    "clientRequestToken": "123Example",
    "commentId": "ff30b348EXAMPLEb9aa670f",
    "content": "Whoops - I meant to add this comment to the line, but I don't see
how to delete it.",
    "creationDate": 1508369768.142,
    "deleted": false,
    "commentId": "",
    "lastModifiedDate": 1508369842.278,
    "callerReactions": [],

```

```
    "reactionCounts":  
      {  
        "SMILE" : 6,  
        "THUMBSUP" : 1  
      }  
  }  
}
```

So löschen Sie den Inhalt eines Kommentars zu einem Commit

1. Führen Sie den Befehl `delete-comment-content` aus und geben Sie die vom System generierte Kommentar-ID an. Zum Beispiel:

```
aws codecommit delete-comment-content --comment-id ff30b348EXAMPLEb9aa670f
```

Note

Sie können den Inhalt eines Kommentars nur löschen, wenn Sie die `AWSCodeCommitFullAccess` Richtlinie angewendet haben oder wenn Sie die `DeleteCommentContent` Berechtigung auf Zulassen gesetzt haben.

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{  
  "comment": {  
    "creationDate": 1508369768.142,  
    "deleted": true,  
    "lastModifiedDate": 1508369842.278,  
    "clientRequestToken": "123Example",  
    "commentId": "ff30b348EXAMPLEb9aa670f",  
    "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",  
    "callerReactions": [],  
    "reactionCounts":  
      {  
        "CLAP" : 1  
      }  
  }  
}
```

So erstellen Sie einen Kommentar zu einem Commit

1. Führen Sie den Befehl `post-comment-for-compared-commit` aus und geben Sie Folgendes an:
 - Der Name des CodeCommit Repositorys (mit der `--repository-name` Option).
 - Die vollständige Commit-ID des AFTER-Commits, um die Richtung des Vergleichs festzulegen (mit der Option `--after-commit-id`).
 - Die vollständige Commit-ID des BEFORE-Commits, um die Richtung des Vergleichs festzulegen (mit der `--before-commit-id`-Option).
 - Einzigartiger, vom Client generierter Idempotenz-Token (mit der Option `--client-request-token`).
 - Der Inhalt Ihres Kommentars (mit der Option `--content`).
 - Eine Liste der Speicherortangaben, wo der Kommentar abgelegt werden soll, unter anderem:
 - Der Name der zu vergleichenden Datei, einschließlich der Erweiterung und des Unterverzeichnisses (mit dem Attribut `filePath`).
 - Die Zeilennummer der Änderung innerhalb einer verglichenen Datei (mit dem Attribut `filePosition`).
 - Die Angabe, ob der Kommentar zu der Änderung vor oder nach dem Vergleich zwischen den Quell- und Ziel-Branches steht (mit dem Attribut `relativeFileVersion`).

Zum Beispiel, um den Kommentar zur Änderung *"Can you add a test case for this?"* an der *cl_sample.js* Datei beim Vergleich zwischen zwei Commits in einem Repository mit dem Namen *MyDemoRepo* hinzuzufügen:

```
aws codecommit post-comment-for-compared-commit --repository-name MyDemoRepo
--before-commit-id 317f8570EXAMPLE --after-commit-id 5d036259EXAMPLE --client-
request-token 123Example --content "Can you add a test case for this?" --location
filePath=cl_sample.js,filePosition=1232,relativeFileVersion=AFTER
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
  "afterBlobId": "1f330709EXAMPLE",
  "afterCommitId": "317f8570EXAMPLE",
  "beforeBlobId": "80906a4cEXAMPLE",
  "beforeCommitId": "6e147360EXAMPLE",
  "comment": {
    "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
```



```

        "clientRequestToken": "",
        "commentId": "553b509bEXAMPLE56198325",
        "content": "Can you add a test case for this?",
        "creationDate": 1508369612.203,
        "deleted": false,
        "commentId": "abc123-EXAMPLE",
        "lastModifiedDate": 1508369612.203,
        "callerReactions": [],
        "reactionCounts": []
    },
    "location": {
        "filePath": "cl_sample.js",
        "filePosition": 1232,
        "relativeFileVersion": "AFTER"
    },
    "repositoryName": "MyDemoRepo"
}

```

So aktualisieren Sie einen Kommentar zu einem Commit

1. Führen Sie den Befehl `update-comment` aus und geben Sie die vom System generierte Kommentar-ID und den Inhalt an, mit dem Sie vorhandenen Inhalt ersetzen möchten.

Zum Beispiel, um den Inhalt *"Fixed as requested. I'll update the pull request."* zu einem Kommentar mit der *442b498bEXAMPLE5756813* ID hinzuzufügen:

```
aws codecommit update-comment --comment-id 442b498bEXAMPLE5756813 --content "Fixed as requested. I'll update the pull request."
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```

{
  "comment": {
    "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
    "clientRequestToken": "",
    "commentId": "442b498bEXAMPLE5756813",
    "content": "Fixed as requested. I'll update the pull request.",
    "creationDate": 1508369929.783,
    "deleted": false,
    "lastModifiedDate": 1508369929.287,
    "callerReactions": [],

```

```
    "reactionCounts":  
      {  
        "THUMBSUP" : 2  
      }  
  }  
}
```

So antworten Sie auf einen Kommentar zu einem Commit

1. Um eine Antwort auf einen Kommentar in einer Pull-Anforderung hinzuzufügen, führen Sie den Befehl `post-comment-reply` aus und geben Folgendes an:
 - Die vom System generierte ID des Kommentars, auf den Sie antworten wollen (mit der Option `--in-reply-to`).
 - Einzigartiger, vom Client generierter Idempotenz-Token (mit der Option `--client-request-token`).
 - Der Inhalt Ihrer Antwort (mit der Option `--content`).

Um beispielsweise die Antwort auf *"Good catch. I'll remove them."* den Kommentar mit der vom System generierten ID hinzuzufügen: *abcd1234EXAMPLEb5678efgh*

```
aws codecommit post-comment-reply --in-reply-to abcd1234EXAMPLEb5678efgh --  
content "Good catch. I'll remove them." --client-request-token 123Example
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{  
  "comment": {  
    "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",  
    "clientRequestToken": "123Example",  
    "commentId": "442b498bEXAMPLE5756813",  
    "content": "Good catch. I'll remove them.",  
    "creationDate": 1508369829.136,  
    "deleted": false,  
    "CommentId": "abcd1234EXAMPLEb5678efgh",  
    "lastModifiedDate": 150836912.221,  
    "callerReactions": [],  
    "reactionCounts": []  
  }  
}
```

Um auf einen Kommentar zu einem Commit mit einem Emoji zu antworten

1. Um auf einen Kommentar in einer Pull-Anfrage mit einem Emoji zu antworten oder den Wert deiner Emoji-Reaktion zu ändern, führe den `put-comment-reaction` Befehl aus und gib Folgendes an:
 - Die vom System generierte ID des Kommentars, auf den Sie mit einem Emoji antworten möchten.
 - Der Wert der Reaktion, die Sie hinzufügen oder aktualisieren möchten. Zu den akzeptablen Werten gehören unterstützte Emojis, Shortcodes und Unicode-Werte.

Die folgenden Werte werden für Emojis in unterstützt: CodeCommit

Emoji	Shortcode	Unicode
#	:thumbsup:	U+1F44D
#	:thumbsdown:	U+1F44E
#	:smile:	U+1F604
♥	:heart:	U+2764
#	:angry:	U+1F620
#	:confused:	U+1F615
#	:scream:	U+1F631
#	:sob:	U+1F62D
#	:clap:	U+1F44F
#	:confetti_ball:	U+1F38A
#	:ship:	U+1F6A2
#	:eyes:	U+1F440

Emoji	Shortcode	Unicode
	Keine	U+0000

Zum Beispiel, um das Emoji `:thumbsup:` dem Kommentar mit der vom System generierten ID hinzuzufügen: `abcd1234EXAMPLEb5678efgh`

```
aws codecommit put-comment-reaction --comment-id abcd1234EXAMPLEb5678efgh --
reaction-value :thumbsup:
```

2. Bei Erfolg erzeugt dieser Befehl keine Ausgabe.

Um Emoji-Reaktionen auf einen Kommentar zu sehen

1. Um Emoji-Reaktionen auf einen Kommentar anzuzeigen, einschließlich der Benutzer, die mit diesen Emojis reagiert haben, führen Sie den `get-comment-reactions` Befehl aus und geben Sie die vom System generierte ID des Kommentars an.

Um beispielsweise Emoji-Reaktionen auf den Kommentar mit der systemgenerierten ID anzuzeigen: `abcd1234EXAMPLEb5678efgh`

```
aws codecommit get-comment-reactions --comment-id abcd1234EXAMPLEb5678efgh
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
  "reactionsForComment": {
    [
      {
        "reaction": {
          "emoji": "#",
          "shortCode": "thumbsup",
          "unicode": "U+1F44D"
        },
        "users": [
          "arn:aws:iam::123456789012:user/Li_Juan",
          "arn:aws:iam::123456789012:user/Mary_Major",
          "arn:aws:iam::123456789012:user/Jorge_Souza"
        ]
      }
    ],
  },
}
```

```
{
  "reaction": {
    "emoji": "#",
    "shortCode": "thumbsdown",
    "unicode": "U+1F44E"
  },
  "users": [
    "arn:aws:iam::123456789012:user/Nikhil_Jayashankar"
  ]
},
{
  "reaction": {
    "emoji": "#",
    "shortCode": "confused",
    "unicode": "U+1F615"
  },
  "users": [
    "arn:aws:iam::123456789012:user/Saanvi_Sarkar"
  ]
}
]
```

Erstellen Sie ein Git-Tag in AWS CodeCommit

Sie können ein Git-Tag verwenden, um ein Commit mit einer Kennung zu versehen, damit andere Repository-Benutzer dessen Bedeutung verstehen. Um ein Git-Tag in einem CodeCommit Repository zu erstellen, können Sie Git von einem lokalen Repo aus verwenden, das mit dem CodeCommit Repository verbunden ist. Nachdem Sie ein Git-Tag im lokalen Repo erstellt haben, können Sie es verwenden, `git push --tags` um es in das CodeCommit Repository zu übertragen.

Weitere Informationen finden Sie unter [Tag-Details anzeigen](#).

Verwende Git, um ein Tag zu erstellen

Gehen Sie wie folgt vor, um mit Git aus einem lokalen Repository ein Git-Tag in einem CodeCommit Repository zu erstellen.

In diesen Schritten gehen wir davon aus, dass Sie das lokale Repo bereits mit dem Repository verbunden haben. CodeCommit Detaillierte Anweisungen finden Sie unter [Herstellen einer Verbindung mit einem Repository](#).

1. Führen Sie den git tag **new-tag-name commit-id** Befehl aus, wobei **new-tag-name** der Name des neuen Git-Tags und **commit-id** die ID des Commits stehen, der dem Git-Tag zugeordnet werden soll.

Beispielsweise wird mit dem folgenden Befehl das neue Git-Tag beta erstellt und mit der Commit-ID dc082f9a...af873b88 verknüpft:

```
git tag beta dc082f9a...af873b88
```

2. Um das neue Git-Tag vom lokalen Repository in das CodeCommit Repository zu übertragen, führen Sie den git push **remote-name new-tag-name** Befehl aus, wobei **remote-name** der Name des CodeCommit Repositorys und der Name des neuen Git-Tags **new-tag-name** steht.

Um beispielsweise ein neues Git-Tag mit dem Namen in ein CodeCommit Repository mit dem Namen beta zu übertragen:

```
git push origin beta
```

Note

Um alle neuen Git-Tags von deinem lokalen Repo in das CodeCommit Repository zu übertragen, führe git push --tags den Befehl aus.

Um sicherzustellen, dass Ihr lokales Repo mit allen Git-Tags im CodeCommit Repository aktualisiert wird, führen Sie den Befehl git fetch gefolgt von git fetch --tags aus.

Weitere Optionen findest du in deiner Git-Dokumentation.

Git-Tag-Details anzeigen in AWS CodeCommit

In Git ist ein Tag eine Bezeichnung für eine Referenz wie ein Commit, um es mit Informationen zu kennzeichnen, die möglicherweise wichtig für andere Repository-Benutzer sind. Sie können beispielsweise das Commit, das der Beta-Release-Punkt für ein Projekt war, mit dem Tag **beta**

kennzeichnen. Weitere Informationen finden Sie unter [Verwende Git, um ein Tag zu erstellen](#). Git-Tags unterscheiden sich von Repository-Tags. Weitere Informationen zur Verwendung von Repository-Tags finden Sie unter [Hinzufügen eines Tags zu einem Repository](#).

Sie können die AWS CodeCommit Konsole verwenden, um Informationen zu Git-Tags in Ihrem Repository anzuzeigen, einschließlich des Datums und der Commit-Nachricht des Commits, auf den jedes Git-Tag verweist. In der Konsole können Sie das Commit, auf das das Tag verweist, mit dem Kopf des Standard-Branch Ihres Repositories vergleichen. Wie bei jedem anderen Commit können Sie auch den Code am Punkt dieses Git-Tags anzeigen.

Sie können Git auch von Ihrem Terminal oder der Befehlszeile aus verwenden, um Details zu Git-Tags in einem lokalen Repo anzuzeigen.

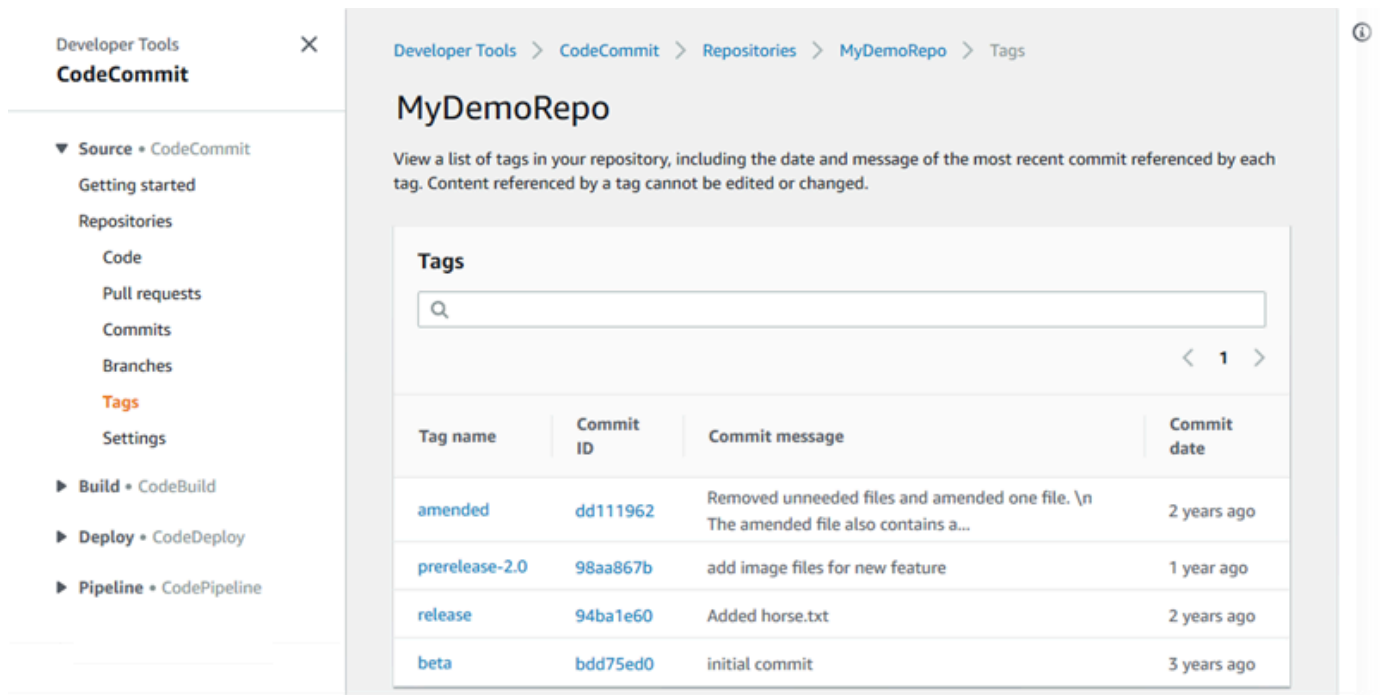
Themen

- [Tag-Details anzeigen \(Konsole\)](#)
- [Git-Tag-Details anzeigen \(Git\)](#)

Tag-Details anzeigen (Konsole)

Verwende die AWS CodeCommit Konsole, um schnell eine Liste der Git-Tags für dein Repository und Details zu den Commits anzuzeigen, auf die in den Git-Tags verwiesen wird.

1. Öffne die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositories aus, in dem Sie Tags anzeigen möchten.
3. Wählen Sie im Navigationsbereich die Option Git tags (Git-Tags) aus.



4. Führen Sie eine der folgenden Aktionen aus:

- Zum Anzeigen des Codes, wie er an diesem Commit war, wählen Sie den Git-Tag-Namen aus.
- Um Details des Commits anzuzeigen, einschließlich der vollständigen Commit-Nachricht, des Committers und des Autors, wählen Sie die gekürzte Commit-ID aus.

Git-Tag-Details anzeigen (Git)

Um Git zu verwenden, um Details zu Git-Tags in einem lokalen Repository anzuzeigen, führen Sie einen der folgenden Befehle aus:

- [git tag](#), um eine Liste der Git-Tag-Namen zu erhalten.
- [git show](#), um Informationen über ein bestimmtes Git-Tag anzuzeigen.
- [Git ls-remote](#), um Informationen über Git-Tags in einem CodeCommit Repository anzuzeigen.

Note

Um sicherzustellen, dass Ihr lokales Repo mit allen Git-Tags im CodeCommit Repository aktualisiert wird, führen Sie den Befehl `git fetch` gefolgt von `git fetch --tags` aus.

In den folgenden Schritten gehen wir davon aus, dass Sie das lokale Repo bereits mit einem CodeCommit Repository verbunden haben. Detaillierte Anweisungen finden Sie unter [Herstellen einer Verbindung mit einem Repository](#).

Um eine Liste von Git-Tags in einem lokalen Repo anzuzeigen

1. Führen Sie den Befehl `git tag` aus:

```
git tag
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
beta  
release
```

Note

Wenn keine Tags definiert wurden, gibt `git tag` nichts zurück.

Weitere Optionen findest du in deiner Git-Dokumentation.

So zeigen Sie Informationen zu einem Git-Tag in einem lokalen Repo an

1. Führen Sie den Befehl `git show tag-name` aus. Führen Sie beispielsweise den folgenden Befehl aus, um Informationen zu einem Git-Tag mit dem Namen `beta` anzuzeigen:

```
git show beta
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
commit 317f8570...ad9e3c09  
Author: John Doe <johndoe@example.com>  
Date:   Tue Sep 23 13:49:51 2014 -0700  
  
    Added horse.txt  
  
diff --git a/horse.txt b/horse.txt  
new file mode 100644  
index 0000000..df42ff1  
--- /dev/null
```

```
+++ b/horse.txt
@@ -0,0 +1 @@
+The horse (Equus ferus caballus) is one of two extant subspecies of Equus ferus
\ No newline at end of file
```

 Note

Geben Sie `:q` ein, um die Ausgabe der Tag-Informationen zu verlassen.

Weitere Optionen findest du in deiner Git-Dokumentation.

So zeigen Sie Informationen zu Git-Tags in einem CodeCommit Repository an

1. Führen Sie den Befehl `git ls-remote --tags` aus.

```
git ls-remote --tags
```

2. Bei Erfolg erzeugt dieser Befehl als Ausgabe eine Liste der Git-Tags im CodeCommit Repository:

```
129ce87a...70fbffba    refs/tags/beta
785de9bd...59b402d8    refs/tags/release
```

Wenn keine Git-Tags definiert wurden, gibt `git ls-remote --tags` eine leere Zeile zurück.

Weitere Optionen findest du in deiner Git-Dokumentation.

Lösche ein Git-Tag in AWS CodeCommit

Um ein Git-Tag in einem CodeCommit Repository zu löschen, verwende Git aus einem lokalen Repo, das mit dem CodeCommit Repository verbunden ist.

Verwende Git, um ein Git-Tag zu löschen

Gehen Sie wie folgt vor, um mit Git aus einem lokalen Repository ein Git-Tag in einem CodeCommit Repository zu löschen.

Diese Schritte wurden unter der Annahme geschrieben, dass Sie das lokale Repo bereits mit dem Repository verbunden haben. CodeCommit Detaillierte Anweisungen finden Sie unter [Herstellen einer Verbindung mit einem Repository](#).

1. Um das Git-Tag aus dem lokalen Repository zu löschen, führen Sie den `git tag -d tag-name` Befehl aus, bei dem der Name des Git-Tags **tag-name** steht, das Sie löschen möchten.

 Tip

Führen Sie den Befehl `git tag` aus, um eine Liste der Git-Tag-Namen zu erhalten.

Um beispielsweise ein Git-Tag im lokalen Repo mit dem Namen `beta` zu löschen:

```
git tag -d beta
```

2. Um das Git-Tag aus dem CodeCommit Repository zu löschen, führen Sie den `git push remote-name --delete tag-name` Befehl aus, bei dem **remote-name** es sich um den Spitznamen handelt, den das lokale Repo für das CodeCommit Repository verwendet, und **tag-name** um den Namen des Git-Tags, das Sie aus dem CodeCommit Repository löschen möchten.

 Tip

Um eine Liste der CodeCommit Repository-Namen und ihrer Namen zu erhalten URLs, führen Sie den `git remote -v` Befehl aus.

Um beispielsweise ein Git-Tag mit dem Namen `beta` im CodeCommit Repository mit dem Namen `origin` zu löschen:

```
git push origin --delete beta
```

Mit Branches in AWS CodeCommit Repositorys arbeiten

Was ist ein Branch? In Git sind Branches Zeiger oder Verweise auf einen Commit. In der Entwicklung sind sie eine praktische Möglichkeit, die Arbeit zu organisieren. Du kannst Branches verwenden, um die Arbeit an einer neuen oder anderen Version von Dateien voneinander zu trennen, ohne die Arbeit in anderen Branches zu beeinträchtigen. Sie können Branches verwenden, um neue Funktionen zu entwickeln, spezifische Projektversionen von einem bestimmten Commit zu speichern und mehr. Wenn Sie Ihren ersten Commit erstellen, wird ein Standard-Branch für Sie erstellt. Dieser Standard-Branch wird als Basis- oder Standard-Branch in lokalen Repositorys (Repos) verwendet, wenn Benutzer das Repository klonen. Der Name dieses Standard-Branche hängt davon ab, wie Sie Ihren ersten Commit erstellen. Wenn Sie die erste Datei mithilfe der CodeCommit Konsole zu Ihrem Repository hinzufügen AWS CLI, lautet der SDKs Name des Standard-Branche main. Dies ist der Standard-Branch-Name, der in den Beispielen in diesem Handbuch verwendet wird. Wenn du deinen ersten Commit mit einem Git-Client pushst, gibt der Git-Client den Namen des Standard-Branche als Standard an. Erwägen Sie, Ihren Git-Client so zu konfigurieren, dass er main als Namen für den ersten Branch verwendet.

CodeCommitIn kannst du den Standard-Branch für dein Repository ändern. Sie können auch Branches erstellen und löschen und Details über einen Branch anzeigen. Sie können die Unterschiede zwischen einem Branch und dem Standard-Branch (oder zwei anderen Branches) schnell ermitteln. Um den Verlauf der Branches und Merges in deinem Repository anzusehen, kannst du den [Commit-Visualizer](#) verwenden, der in der folgenden Grafik dargestellt ist.

Developer Tools **CodeCommit**

▼ Source • CodeCommit

- Getting started
- Repositories
- Code
- Pull requests
- Commits**
- Branches
- Tags
- Settings

► Build • CodeBuild

► Deploy • CodeDeploy

► Pipeline • CodePipeline

Developer Tools > CodeCommit > Repositories > MyDemoRepo > Commits

MyDemoRepo

Commits | **Commit visualizer** | Compare commits

Commit visualizer

Commit history visualization showing a vertical timeline with colored branches (blue, green, black) and commit points. The commits are listed on the right:

Commit Hash	Description	Time Ago
d615e7ae	Merge branch 'AnotherBranch' into testbranch	2 minutes ago
b6589863	Added another file.	2 minutes ago
73a6e39c	remote-tracking branch 'refs/remotes/origin/jane-branch' into jane-branch	
6bbb6d3c	Another test of the editing feature.	20 minutes ago
edacdfce	Testing this out to see how well it works.	23 minutes ago
70bb94d7	Revised test results with correct information.	36 minutes ago
b78e6d1c	Merge branch 'results' into testbranch	50 minutes ago
84b7d158	Edited ahs_count.py	50 minutes ago

Informationen zur Arbeit mit anderen Aspekten Ihres Repositorys finden Sie unter [CodeCommit](#), [Arbeiten mit Repositorien](#), [Mit Dateien arbeiten](#), [Verwenden von Pull-Anforderungen](#), [Mit Commits arbeiten](#), und [Mit Benutzereinstellungen arbeiten](#)

Themen

- [Erstellen Sie einen Zweig in AWS CodeCommit](#)
- [Beschränken Sie die Anzahl von Pushs und Merges auf Branches in AWS CodeCommit](#)
- [Branch-Details finden Sie in AWS CodeCommit](#)
- [Vergleichen und zusammenführen in AWS CodeCommit](#)
- [Ändern Sie die Zweigeinstellungen in AWS CodeCommit](#)
- [Lösche einen Zweig in AWS CodeCommit](#)

Erstellen Sie einen Zweig in AWS CodeCommit

Sie können die CodeCommit Konsole oder die verwenden AWS CLI , um Branches für Ihr Repository zu erstellen. Dies ist eine schnelle Möglichkeit, um separat an einer neuen oder anderen Version von Dateien zu arbeiten, ohne den Standard-Branch zu beeinträchtigen. Nachdem Sie einen Branch in der CodeCommit Konsole erstellt haben, müssen Sie diese Änderung in Ihr lokales Repository übernehmen. Alternativ können Sie lokal einen Branch erstellen und dann Git von einem lokalen Repo aus verwenden, das mit dem CodeCommit Repository verbunden ist, um diese Änderung zu pushen.

Themen

- [Erstelle einen Branch \(Konsole\)](#)
- [Einen Zweig erstellen \(Git\)](#)
- [Erstelle einen Zweig \(AWS CLI\)](#)

Erstelle einen Branch (Konsole)

Sie können die CodeCommit Konsole verwenden, um einen Branch in einem CodeCommit Repository zu erstellen. Wenn Benutzer das nächste Mal Änderungen vom Repository abrufen, sehen sie den neuen Branch.

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, in dem Sie einen Branch erstellen möchten.
3. Wählen Sie im Navigationsbereich Branches aus.
4. Klicken Sie auf Create branch.

Create branch

Branch name
feature_advanced-class

Branch from

Search:

Branches

- main
Default branch
- bugfix-1236
- feature-lambdafunctions
- feature-new-wizard
- feature-randomizationfeature

Git tags

- release
- prerelease-2.0
- beta
- amended

Create branch

Geben Sie im Feld Branch name (Branch-Name) einen Namen für den Branch ein. Wählen Sie unter Branch from (Verzweigen von) entweder einen Branch oder ein Tag aus der Liste aus oder fügen Sie eine Commit-ID ein. Klicken Sie auf Create branch.

Einen Zweig erstellen (Git)

Gehen Sie wie folgt vor, um mit Git von einem lokalen Repository aus einen Branch in einem lokalen Repository zu erstellen und diesen Branch dann in das Repository zu übertragen. CodeCommit

Diese Schritte wurden unter der Annahme geschrieben, dass Sie das lokale Repository bereits mit dem Repository verbunden haben. CodeCommit Detaillierte Anweisungen finden Sie unter [Herstellen einer Verbindung mit einem Repository](#).

1. Erstellen Sie einen Branch in Ihrem lokalen Repository, indem Sie den `git checkout -b new-branch-name` Befehl ausführen. Dabei **new-branch-name** steht der Name des neuen Branches.

Mit dem folgenden Befehl wird beispielsweise ein Branch mit dem Namen MyNewBranch im lokalen Repository erstellt:

```
git checkout -b MyNewBranch
```

2. Um den neuen Branch vom lokalen Repository in das CodeCommit Repository zu übertragen, führen Sie den `git push` Befehl aus **remote-name** und geben Sie dabei sowohl den als auch den an. **new-branch-name**

Um beispielsweise einen neuen Branch im lokalen Repository mit dem Namen in das CodeCommit Repository mit dem Spitznamen MyNewBranch zu übertragen: `origin`

```
git push origin MyNewBranch
```

Note

Wenn Sie die `-u` Option hinzufügen `git push` (z. B. `git push -u origin main`), können Sie in future `git push` ohne arbeiten **remote-namebranch-name**. Es werden Upstream-Nachverfolgungsdaten festgelegt. Führen Sie `git remote show remote-name` aus (z. B. `git remote show origin`), um Upstream-Nachverfolgungsdaten abzurufen.

Führen Sie `git branch --all` aus, um eine Liste aller lokalen und Remote-Nachverfolgungs-Branches anzuzeigen.

Um einen Zweig im lokalen Repository einzurichten, der mit einem Zweig im CodeCommit Repository verbunden ist, führen Sie `git checkout remote-branch-name` den Befehl aus.

Weitere Optionen findest du in deiner Git-Dokumentation.

Erstelle einen Zweig (AWS CLI)

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter [Befehlszeilenreferenz](#).

Gehen Sie wie folgt vor, um mit dem einen Branch in einem CodeCommit Repository AWS CLI zu erstellen und diesen Branch dann per Push in das CodeCommit Repository zu übertragen. Die Schritte zum Erstellen eines ersten Commits und zum Angeben des Namens des Standard-Banches für ein leeres Repository finden [Sie unter Erstellen Sie den ersten Commit für ein Repository mit dem AWS CLI](#).

1. Führen Sie den Befehl `create-branch` aus und geben Sie Folgendes an:
 - Der Name des CodeCommit Repositorys, in dem der Branch erstellt wird (mit der `--repository-name` Option).

Note

Um den Namen des CodeCommit Repositorys abzurufen, führen Sie den Befehl [list-repositories](#) aus.

- Der Name des neuen Branches (mit der Option `--branch-name`).
- Die ID des Commits, auf den der neue Branch verweist (mit der Option `--commit-id`).

Um beispielsweise einen Branch mit dem Namen zu erstellen, der auf MyNewBranch die Commit-ID 317f8570EXAMPLE in einem CodeCommit Repository mit dem Namen verweist: MyDemoRepo

```
aws codecommit create-branch --repository-name MyDemoRepo --branch-name MyNewBranch
--commit-id 317f8570EXAMPLE
```

Dieser Befehl liefert nur eine Ausgabe, wenn Fehler aufgetreten sind.

2. Um die Liste der verfügbaren CodeCommit Repository-Banches in Ihrem lokalen Repository mit dem neuen Namen der Remote-Banches zu aktualisieren, führen Sie `git remote update remote-name` folgenden Befehl aus.

Um beispielsweise die Liste der verfügbaren Branches für das CodeCommit Repository mit dem Spitznamen `origin` zu aktualisieren:

```
git remote update origin
```

Note

Alternativ können Sie den Befehl `git fetch` ausführen. Sie können sich auch alle Remote-Banches ansehen mit `git branch --all`, indem Sie den Befehl ausführen, aber bis Sie die Liste Ihres lokalen Repos aktualisiert haben, erscheint der von Ihnen erstellte Remote-Branch nicht in der Liste.

Weitere Optionen findest du in deiner Git-Dokumentation.

3. Um einen Branch im lokalen Repository einzurichten, der mit dem neuen Branch im CodeCommit Repository verbunden ist, führe `git checkout remote-branch-name` folgenden Befehl aus.

Note

Um eine Liste der CodeCommit Repository-Namen und ihrer Namen zu erhalten URLs, führen Sie den `git remote -v` Befehl aus.

Beschränken Sie die Anzahl von Pushs und Merges auf Branches in AWS CodeCommit

Standardmäßig kann jeder CodeCommit Repository-Benutzer, der über ausreichende Berechtigungen verfügt, um Code in das Repository zu übertragen, zu jedem Branch in diesem Repository beitragen. Dies gilt unabhängig davon, wie Sie einen Branch zum Repository hinzuzufügen: über die Konsole, die Befehlszeile oder Git. Sie können jedoch einen Branch konfigurieren, sodass nur einige Repository-Benutzer Code zu diesem Branch senden oder mit ihm zusammenführen können. Beispielsweise können Sie eine Verzweigung für den Produktionscode so konfigurieren, dass nur bestimmte erfahrene Entwickler Änderungen an dieser Verzweigung vornehmen oder zusammenführen können. Andere Entwickler können etwas aus der Verzweigung abrufen, ihre eigenen Verzweigungen erstellen und Pull-Anforderungen erstellen, aber sie können keine Änderungen an diese Verzweigung senden oder mit dieser zusammenführen. Sie können diesen Zugriff konfigurieren, indem Sie eine bedingte Richtlinie erstellen, die einen Kontextschlüssel für einen oder mehrere Zweige in IAM verwendet.

 Note

Um einige der Verfahren in diesem Thema abzuschließen, müssen Sie sich mit einem Administratorbenutzer anmelden, der über ausreichende Berechtigungen zum Konfigurieren und Anwenden von IAM-Richtlinien verfügt. Weitere Informationen finden Sie unter [Einen IAM-Admin-Benutzer und eine IAM-Admin-Gruppe erstellen](#).

Themen

- [Konfigurieren Sie eine IAM-Richtlinie, um Pushs und Merges auf einen Branch zu beschränken](#)
- [Wenden Sie die IAM-Richtlinie auf eine IAM-Gruppe oder -Rolle an](#)
- [Testen Sie die Richtlinie](#)

Konfigurieren Sie eine IAM-Richtlinie, um Pushs und Merges auf einen Branch zu beschränken

Sie können in IAM eine Richtlinie erstellen, die verhindert, dass Benutzer einen Branch aktualisieren, einschließlich der Übertragung von Commits an einen Branch und der Zusammenführung von Pull-Requests an einen Branch. Dazu verwendet Ihre Richtlinie eine bedingte Anweisung, sodass die Wirkung der Anweisung Deny nur stattfindet, wenn die Bedingung erfüllt ist. Die Angaben APIs, die Sie in die Deny Anweisung aufnehmen, bestimmen, welche Aktionen nicht zulässig sind. Sie können diese Richtlinie so konfigurieren, dass sie nur für einen Branch in einem Repository, eine Reihe von Branches in einem Repository oder für alle Branches gilt, die den Kriterien in allen Repositories in einem Amazon Web Services Services-Konto entsprechen.

So erstellen Sie eine bedingte Richtlinie für Verzweigungen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter <https://console.aws.amazon.com/iam/>
2. Wählen Sie im Navigationsbereich Richtlinien.
3. Wählen Sie Richtlinie erstellen aus.
4. Wählen Sie JSON, und fügen Sie dann die folgende Beispielrichtlinie ein. Ersetzen Sie den Wert von Resource durch den ARN des Repositories, das die Verzweigung enthält, für die Sie den Zugriff einschränken möchten. Ersetzen Sie den Wert von `codecommit:References` durch einen Verweis auf den Branch oder die Branches, für die Sie den Zugriff einschränken möchten. Diese Richtlinie verbietet beispielsweise das Senden von Commits, das Zusammenführen von

Branches, das Löschen von Branches, das Löschen von Dateien, das Zusammenführen von Pull-Requests und das Hinzufügen von Dateien zu einem Branch mit dem Namen *main* und einem Branch mit dem Namen *prod: MyDemoRepo*

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "codecommit:GitPush",
        "codecommit>DeleteBranch",
        "codecommit>DeleteFile",
        "codecommit:PutFile",
        "codecommit:MergeBranchesByFastForward",
        "codecommit:MergeBranchesBySquash",
        "codecommit:MergeBranchesByThreeWay",
        "codecommit:MergePullRequestByFastForward",
        "codecommit:MergePullRequestBySquash",
        "codecommit:MergePullRequestByThreeWay"
      ],
      "Resource": "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo",
      "Condition": {
        "StringEqualsIfExists": {
          "codecommit:References": [
            "refs/heads/main",
            "refs/heads/prod"
          ]
        },
        "Null": {
          "codecommit:References": "false"
        }
      }
    }
  ]
}
```

Branches in Git sind einfach Zeiger (Referenzen) auf den SHA-1-Wert des HEAD-Commit, weshalb die Bedingung References verwendet. Die Null -Anweisung in jeder Richtlinie erforderlich, die ein Deny bewirkt, und wobei eine der Aktionen GitPush ist. Dies ist aufgrund der Art und Weise erforderlich, wie Git und git-receive-pack arbeiten, wenn sie Änderungen von einem lokalen Repository an CodeCommit senden.

 Tip

Um eine Richtlinie zu erstellen, die für alle Branches mit dem Namen main in allen Repositorys eines Amazon Web Services Services-Kontos gilt, ändern Sie den Wert `Resource` von einem Repository-ARN in ein Sternchen (*)^{*}.

5. Wählen Sie Richtlinie prüfen. Korrigieren Sie alle Fehler in Ihrer Richtlinienanweisung und fahren Sie dann mit Create policy (Richtlinie erstellen) fort.
6. Wenn JSON validiert wird, wird die Seite Seite Create policy (Richtlinie erstellen) angezeigt. Eine Warnung wird im Abschnitt Summary (Zusammenfassung) angezeigt, die Sie darauf hinweist, dass diese Richtlinie keine Berechtigungen gewährt. Das ist normal.
 - Geben Sie unter Name einen Namen für diese Richtlinie wie **DenyChangesToMain** ein.
 - Geben Sie in Description (Beschreibung) eine Beschreibung zum Zweck der Richtlinie ein. Dies ist zwar optional, wird aber empfohlen.
 - Wählen Sie Richtlinie erstellen aus.

Wenden Sie die IAM-Richtlinie auf eine IAM-Gruppe oder -Rolle an

Sie haben eine Richtlinie erstellt, die Push- und Merge-Vorgänge auf einen Branch beschränkt, aber die Richtlinie ist erst wirksam, wenn Sie sie auf einen IAM-Benutzer, eine IAM-Gruppe oder eine IAM-Rolle anwenden. Es hat sich bewährt, die Richtlinie auf eine IAM-Gruppe oder -Rolle anzuwenden. Die Anwendung von Richtlinien auf einzelne IAM-Benutzer lässt sich nicht gut skalieren.

Anwendung der bedingten Richtlinie auf eine Gruppe oder Rolle

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter <https://console.aws.amazon.com/iam/>
2. Wenn Sie die Richtlinie auf eine IAM-Gruppe anwenden möchten, wählen Sie im Navigationsbereich Gruppen aus. Wenn Sie die Richtlinie auf eine Rolle anwenden möchten, die Benutzer übernehmen, wählen Sie Rolle aus. Wählen Sie den Namen der Gruppe oder Rolle aus.
3. Wählen Sie auf der Registerkarte Permissions (Berechtigungen) die Option Attach Policy (Richtlinie zuweisen) aus.
4. Wählen Sie die bedingte Richtlinie, die Sie zuvor erstellt haben, aus der Liste der Richtlinien aus, und klicken Sie auf Attach policy (Richtlinie hinzufügen).

Weitere Informationen finden Sie unter [IAM-Richtlinien anhängen und trennen](#).

Testen Sie die Richtlinie

Sie sollten die Auswirkungen der von Ihnen auf die Gruppe oder Rolle angewendeten Richtlinie testen, um sicherzustellen, dass sie wie erwartet funktioniert. Es gibt viele Möglichkeiten, dies zu realisieren. Wenn Sie beispielsweise eine ähnliche Richtlinie wie die oben gezeigte testen möchten, können Sie wie folgt vorgehen:

- Melden Sie sich bei der CodeCommit Konsole mit einem IAM-Benutzer an, der entweder Mitglied einer IAM-Gruppe ist, auf die die Richtlinie angewendet wurde, oder der eine Rolle annimmt, auf die die Richtlinie angewendet wurde. In der Konsole fügen Sie der Verzweigung, für die die Einschränkungen gelten, eine Datei hinzu. Sie sollten eine Fehlermeldung sehen, wenn Sie versuchen, eine Datei in dieser Verzweigung zu speichern oder eine Datei in die Verzweigung hochzuladen. Fügen Sie einer anderen Verzweigung eine Datei hinzu. Die Operation sollte erfolgreich sein.
- Melden Sie sich bei der CodeCommit Konsole mit einem IAM-Benutzer an, der entweder Mitglied einer IAM-Gruppe ist, auf die die Richtlinie angewendet wurde, oder der eine Rolle annimmt, auf die die Richtlinie angewendet wurde. Erstellen Sie eine Pull-Anforderung, die mit dem Branch zusammengeführt wird, für den die Einschränkungen gelten. Sie sollten in der Lage sein, die Pull-Anforderung zu erstellen, erhalten aber einen Fehler, wenn Sie versuchen, sie zusammenzuführen.
- Erstellen Sie über das Terminal oder die Befehlszeile einen Commit für den Branch, für den die Einschränkungen gelten, und übertragen Sie diesen Commit dann per Push in das CodeCommit Repository. Sie sollten eine Fehlermeldung sehen. Commits und Sendeoperationen von anderen Verzweigungen aus sollten wie gewohnt funktionieren.

Branch-Details finden Sie in AWS CodeCommit

Sie können die CodeCommit Konsole verwenden, um Details zu den Branches in einem CodeCommit Repository anzuzeigen. Sie können das Datum des letzten Commit für einen Branche, die Commit-Nachrichtung und vieles andere mehr anzeigen. Sie können auch das AWS CLI oder Git von einem lokalen Repo verwenden, das mit dem CodeCommit Repository verbunden ist.

Themen

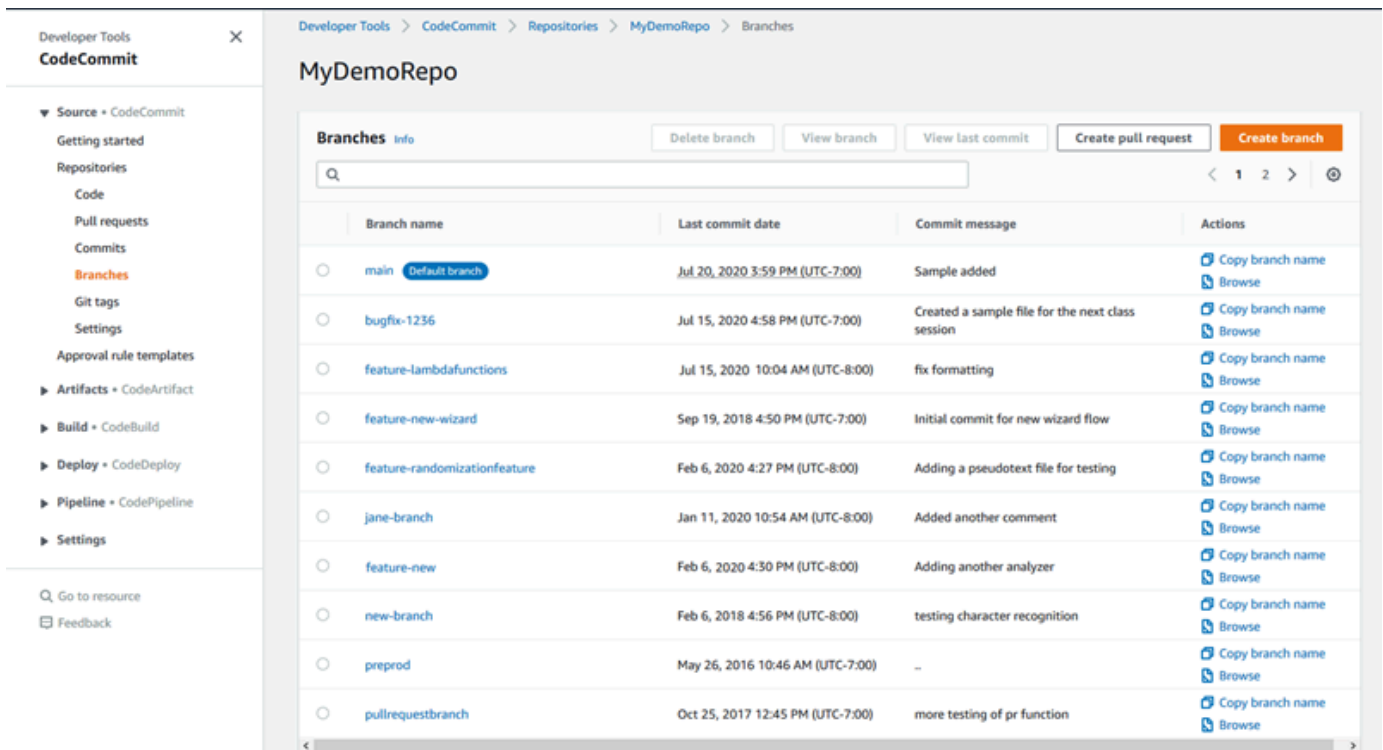
- [Branch-Details anzeigen \(Konsole\)](#)
- [Filiatldetails anzeigen \(Git\)](#)

- [Details zur Filiale anzeigen \(AWS CLI\)](#)

Branch-Details anzeigen (Konsole)

Verwenden Sie die CodeCommit Konsole, um schnell eine Liste der Branches für Ihr Repository und Details zu den Branches anzuzeigen.

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, in dem Sie Branch-Details anzeigen möchten.
3. Wählen Sie im Navigationsbereich Branches aus.



The screenshot shows the AWS CodeCommit console interface. On the left is a navigation sidebar with options like 'Source', 'Artifacts', 'Build', 'Deploy', 'Pipeline', and 'Settings'. The main area displays the 'MyDemoRepo' repository. At the top, there's a breadcrumb trail: 'Developer Tools > CodeCommit > Repositories > MyDemoRepo > Branches'. Below this, the 'Branches' section is active, showing a table of branches. The 'main' branch is marked as the 'Default branch'. Each row in the table includes a radio button, the branch name, the last commit date, the commit message, and a set of actions (Copy branch name, Browse).

Branch name	Last commit date	Commit message	Actions
main (Default branch)	Jul 20, 2020 3:59 PM (UTC-7:00)	Sample added	Copy branch name, Browse
bugfix-1236	Jul 15, 2020 4:58 PM (UTC-7:00)	Created a sample file for the next class session	Copy branch name, Browse
feature-lambdafunctions	Jul 15, 2020 10:04 AM (UTC-8:00)	fix formatting	Copy branch name, Browse
feature-new-wizard	Sep 19, 2018 4:50 PM (UTC-7:00)	Initial commit for new wizard flow	Copy branch name, Browse
feature-randomizationfeature	Feb 6, 2020 4:27 PM (UTC-8:00)	Adding a pseudotext file for testing	Copy branch name, Browse
jane-branch	Jan 11, 2020 10:54 AM (UTC-8:00)	Added another comment	Copy branch name, Browse
feature-new	Feb 6, 2020 4:30 PM (UTC-8:00)	Adding another analyzer	Copy branch name, Browse
new-branch	Feb 6, 2018 4:56 PM (UTC-8:00)	testing character recognition	Copy branch name, Browse
preprod	May 26, 2016 10:46 AM (UTC-7:00)	-	Copy branch name, Browse
pullrequestbranch	Oct 25, 2017 12:45 PM (UTC-7:00)	more testing of pr function	Copy branch name, Browse

4. Der Name des Branches, der als Standard für das Repository verwendet wird, wird neben Standard-Branch angezeigt. Um Details über das letzte Commit für einen Branch anzuzeigen, wählen Sie den Branch und dann View last commit (Letztes Commit anzeigen) aus. Um die Dateien und den Code in einem Branch anzuzeigen, wählen Sie den Branch-Namen.

Filialdetails anzeigen (Git)

Um Git von einem lokalen Repo aus zu verwenden, um Details sowohl zu den lokalen als auch zu den Remote-Tracking-Branches für ein CodeCommit Repository anzuzeigen, führen Sie den `git branch` Befehl aus.

Die folgenden Schritte wurden unter der Annahme geschrieben, dass Sie das lokale Repository bereits mit dem Repository verbunden haben. CodeCommit Detaillierte Anweisungen finden Sie unter [Herstellen einer Verbindung mit einem Repository](#).

1. Führen Sie den Befehl `git branch` aus und geben Sie dabei die Option `--all` an:

```
git branch --all
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
MyNewBranch
* main
remotes/origin/MyNewBranch
remotes/origin/main
```

Das Sternchen (*) wird neben dem aktuell geöffneten Branch angezeigt. Die Einträge danach sind Remotenachverfolgungs-Referenzen.

Tip

`git branch` zeigt lokale Branches an.

`git branch -r` zeigt Remote-Branches an.

`git checkout existing-branch-name` wechselt zu dem angegebenen Branch-Namen. Sofern direkt danach `git branch` ausgeführt wird, wird daneben ein Sternchen angezeigt (*).

`git remote update remote-name` aktualisiert Ihr lokales Repo mit der Liste der verfügbaren CodeCommit Repository-Zweige. (Um eine Liste der CodeCommit Repository-Namen und ihrer Namen zu erhalten URLs, führen Sie den `git remote -v` Befehl aus.)

Weitere Optionen findest du in deiner Git-Dokumentation.

Details zur Filiale anzeigen (AWS CLI)

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter [Befehlszeilenreferenz](#).

Um Details AWS CLI zu den Branches in einem CodeCommit Repository anzuzeigen, führen Sie einen oder mehrere der folgenden Befehle aus:

- Um eine Liste von Branch-Namen anzuzeigen, führen Sie [list-branches](#) aus.
- [Um Informationen zu einem bestimmten Zweig anzuzeigen, führen Sie get-branch aus.](#)

So zeigen Sie eine Liste mit Branch-Namen an

1. Führen Sie den list-branches Befehl aus und geben Sie den Namen des CodeCommit Repositorys an (mit der --repository-name Option).

Tip

Um den Namen des CodeCommit Repositorys abzurufen, führen Sie den Befehl [list-repositories](#) aus.

Um beispielsweise Details zu den Branches in einem CodeCommit Repository mit dem Namen anzuzeigen: MyDemoRepo

```
aws codecommit list-branches --repository-name MyDemoRepo
```

2. Ist der Befehl erfolgreich, wird ein branchNameList-Objekt mit einem Eintrag für jeden Branch ausgegeben.

Es folgt eine Beispielausgabe basierend auf dem vorangehenden Beispielbefehl:

```
{
  "branches": [
    "MyNewBranch",
    "main"
  ]
}
```

So zeigen Sie Informationen über einen Branch an

1. Führen Sie den Befehl `get-branch` aus und geben Sie Folgendes an:

- Den Repository-Namen (mit der Option `--repository-name`).
- Den Branch-Namen (mit der Option `--branch-name`).

Um beispielsweise Informationen über einen Branch anzuzeigen, der `MyNewBranch` in einem CodeCommit Repository mit dem Namen `MyDemoRepo`:

```
aws codecommit get-branch --repository-name MyDemoRepo --branch-name MyNewBranch
```

2. Ist der Befehl erfolgreich, werden der Name des Branches und die ID des letzten, für den Branch ausgeführten Commits ausgegeben.

Es folgt eine Beispielausgabe basierend auf dem vorangehenden Beispielbefehl:

```
{
  "branch": {
    "branchName": "MyNewBranch",
    "commitID": "317f8570EXAMPLE"
  }
}
```

Vergleichen und zusammenführen in AWS CodeCommit

Sie können die CodeCommit Konsole verwenden, um Branches in einem CodeCommit Repository zu vergleichen. Durch den Vergleich von Branches können Sie schnell die Unterschiede zwischen einem Branch und dem Standard-Branch erkennen, oder die Unterschiede zwischen zwei Branches.

Themen

- [Vergleichen Sie einen Branch mit dem Standard-Branch](#)
- [Vergleichen Sie zwei spezifische Branches](#)
- [Zwei Zweige zusammenführen \(\)AWS CLI](#)

Vergleichen Sie einen Branch mit dem Standard-Branch

Verwende die CodeCommit Konsole, um dir schnell die Unterschiede zwischen einem Branch und dem Standard-Branch für dein Repository anzusehen.

1. Öffne die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories (Repositoryys) den Namen des Repositorys aus, in dem Sie Branches vergleichen möchten.
3. Wählen Sie im Navigationsbereich Commits und dann die Compare commits (Commits vergleichen)-Registerkarte aus.
4. Wählen Sie unter Destination (Ziel) den Namen des Standard-Branches aus. Wählen Sie unter Source (Quelle) den Branch aus, den Sie mit dem Standard-Branch vergleichen möchten. Wählen Sie Compare aus.

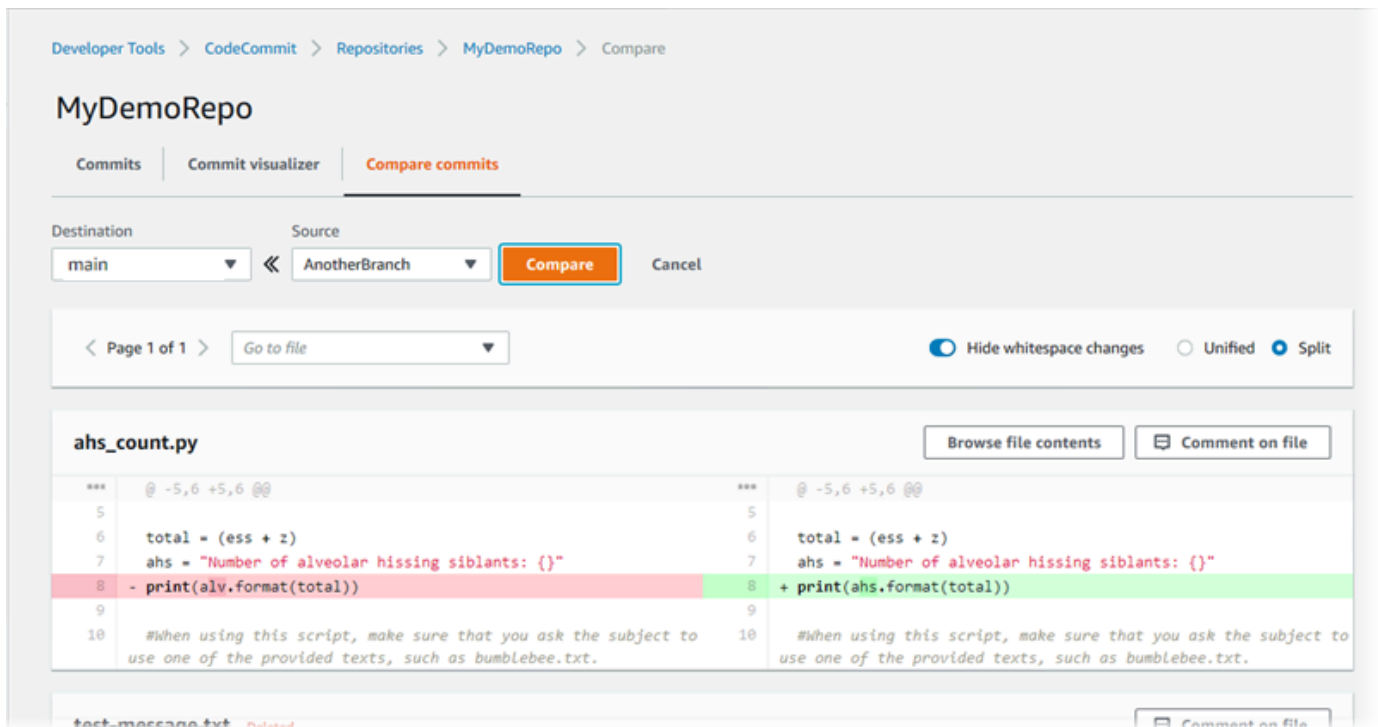
Vergleichen Sie zwei spezifische Branches

Verwenden Sie die CodeCommit Konsole, um die Unterschiede zwischen zwei Branches anzuzeigen, die Sie vergleichen möchten.

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories (Repositoryys) den Namen des Repositorys aus, in dem Sie Branches vergleichen möchten.
3. Wählen Sie im Navigationsbereich Commits und dann die Compare commits (Commits vergleichen)-Registerkarte aus.
4. Wählen Sie unter Destination (Ziel) und Source (Quelle) die zwei zu vergleichenden Branches aus und wählen Sie dann Compare (Vergleichen) aus. Um die Liste der geänderten Dateien anzuzeigen, erweitern Sie die Liste der geänderten Dateien. Sie können die Änderungen an den Dateien nebeneinander (Ansicht Split) oder inline (Ansicht Unified) anzeigen.

Note

Wenn Sie als IAM-Benutzer angemeldet sind, können Sie Ihre Einstellungen für die Anzeige von Code und andere Konsoleinstellungen konfigurieren und speichern. Weitere Informationen finden Sie unter [Mit Benutzereinstellungen arbeiten](#).



Zwei Zweige zusammenführen ()AWS CLI

Sie können zwei Branches in einem CodeCommit Repository zusammenführen, indem Sie eine der verfügbaren Merge-Strategien AWS CLI verwenden, indem Sie einen der folgenden Befehle ausführen:

- Führen Sie den Befehl [merge-branches-by-fast-forward](#) aus, um zwei Branches mithilfe der Mergestrategie mit Vorlauf zusammenzuführen.
- Führen Sie den Befehl [merge-branches-by-squash](#) aus, um zwei Branches mithilfe der Squashmerge-Strategie zusammenzuführen.
- Führen Sie den Befehl [merge-branches-by-three-way](#) aus, um zwei Branches mithilfe der Dreiwegemerge-Strategie zusammenzuführen.

Sie können Zusammenführungen auch testen, indem Sie den Befehl `create-unreferenced-merge-commit` ausführen. Weitere Informationen finden Sie unter [Auflösen von Konflikten in einer Pull-Anforderung](#).

 Note

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter [Befehlszeilenreferenz](#).

Um das zu verwenden AWS CLI , um zwei Zweige in einem CodeCommit Repository zusammenzuführen

1.

Um zwei Branches mithilfe der Mergestrategie mit Vorlauf zusammenzuführen, führen Sie den Befehl `merge-branches-by-fast-forward` aus und geben Sie Folgendes an:

- Den Namen des Quell-Branches, der die Änderungen enthält, die Sie zusammenführen möchten (mit der Option `--source-commit-specifier`).
- Den Namen des Ziel-Branches, in dem Sie Ihre Änderungen zusammenführen möchten (mit der Option `--destination-commit-specifier`).
- Der Name des Repositorys (mit der Option `--repository-name`).

Um zum Beispiel einen Quell-Branch mit einem Ziel-Branch *bugfix-1234* zusammenzuführen, der *preprod* in einem Repository mit dem Namen *MyDemoRepo*:

```
aws codecommit merge-branches-by-fast-forward --source-commit-specifier bugfix-  
bug1234 --destination-commit-specifier preprod --repository-name MyDemoRepo
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{  
  "commitId": "4f178133EXAMPLE",  
  "treeId": "389765daEXAMPLE"  
}
```

2.

Um zwei Branches mithilfe der Squashmerge-Strategie zusammenzuführen, führen Sie den Befehl `merge-branches-by-squash` aus und geben Sie Folgendes an:

- Den Namen des Quell-Branches, der die Änderungen enthält, die Sie zusammenführen möchten (mit der Option `--source-commit-specifier`).

- Den Namen des Ziel-Banches, in dem Sie Ihre Änderungen zusammenführen möchten (mit der Option `--destination-commit-specifier`).
- Der Name des Repositorys (mit der Option `--repository-name`).
- Die einzuschließende Commit-Nachricht (mit der Option `--commit-message`).
- Der für den Commit zu verwendende Name (mit der Option `--name`).
- Die für den Commit zu verwendende E-Mail-Adresse (mit der Option `--email`).

Um beispielsweise einen Quellzweig *bugfix-bug1234* mit dem Namen eines Zielzweigs *bugfix-quarterly* in einem Repository mit dem Namen zusammenzuführen *MyDemoRepo*:

```
aws codecommit merge-branches-by-squash --source-commit-specifier bugfix-bug1234 --  
destination-commit-specifier bugfix-quarterly --author-name "Maria Garcia" --email  
"maria_garcia@example.com" --commit-message "Merging in fix branches to prepare  
for a general patch." --repository-name MyDemoRepo
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{  
  "commitId": "4f178133EXAMPLE",  
  "treeId": "389765daEXAMPLE"  
}
```

3.

Um zwei Branches mithilfe der Dreiwegemerge-Strategie zusammenzuführen, führen Sie den Befehl `merge-branches-by-three-way` aus und geben Sie Folgendes an:

- Den Namen des Quell-Banches, der die Änderungen enthält, die Sie zusammenführen möchten (mit der Option `--source-commit-specifier`).
- Den Namen des Ziel-Banches, in dem Sie Ihre Änderungen zusammenführen möchten (mit der Option `--destination-commit-specifier`).
- Der Name des Repositorys (mit der Option `--repository-name`).
- Die einzuschließende Commit-Nachricht (mit der Option `--commit-message`).
- Der für den Commit zu verwendende Name (mit der Option `--name`).
- Die für den Commit zu verwendende E-Mail-Adresse (mit der Option `--email`).

Um beispielsweise einen Quellzweig *main* mit dem Namen eines Zielzweigs *bugfix-1234* in einem Repository mit dem Namen *MyDemoRepo* zusammenzuführen:

```
aws codecommit merge-branches-by-three-way --source-commit-specifier main --  
destination-commit-specifier bugfix-bug1234 --author-name "Jorge Souza" --email  
"jorge_souza@example.com" --commit-message "Merging changes from main to bugfix  
branch before additional testing." --repository-name MyDemoRepo
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{  
  "commitId": "4f178133EXAMPLE",  
  "treeId": "389765daEXAMPLE"  
}
```

Ändern Sie die Zweigeinstellungen in AWS CodeCommit

Sie können in der AWS CodeCommit Konsole oder mit dem `aws cli` ändern, welcher Zweig als Standardzweig verwendet werden soll. Wenn du beispielsweise deinen ersten Commit mit einem Git-Client erstellt hast, der den Standard-Branch auf `master` setzt, könntest du einen Branch namens `main` erstellen und dann die Branch-Einstellungen so ändern, dass der neue Branch als Standard-Branch für das Repository festgelegt wird. Um andere Branch-Einstellungen zu ändern, kannst du Git von einem lokalen Repo aus verwenden, das mit dem CodeCommit Repository verbunden ist.

Themen

- [Ändern Sie den Standardzweig \(Konsole\)](#)
- [Ändern Sie den Standardzweig \(AWS CLI\)](#)

Ändern Sie den Standardzweig (Konsole)

Sie können in der AWS CodeCommit Konsole angeben, welcher Branch der Standard-Branch in einem CodeCommit Repository ist.

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.

2. Wählen Sie unter Repositories (Repositoryys) den Namen des Repositorys aus, bei dem Sie Einstellungen ändern möchten.
3. Wählen Sie im Navigationsbereich Settings (Einstellungen).
4. Wählen Sie unter Default branch (Standard-Branch) die Dropdown-Liste für Branches aus und wählen Sie dann einen anderen Branch. Wählen Sie Speichern.

 Tip

- Wenn Sie in der Drop-down-Liste keinen anderen Zweig sehen, haben Sie keine zusätzlichen Zweige erstellt. Sie können den Standardzweig eines Repositorys nicht ändern, wenn das Repository nur einen Zweig hat. Weitere Informationen finden Sie unter [Erstellen Sie einen Zweig in AWS CodeCommit](#).
- Wenn Sie den Abschnitt Standardverzweigung nicht sehen, sondern stattdessen Elemente für Benachrichtigungsregeln und Verbindungen sehen, befinden Sie sich im Menü mit den allgemeinen Einstellungen für die Konsole. Das Einstellungsmenü für Repositorys ist unter Repositorys auf derselben Ebene wie Code - und Pull-Anfragen aufgeführt.

Ändern Sie den Standardzweig ()AWS CLI

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter [Befehlszeilenreferenz](#).

AWS CLI Um die Branch-Einstellungen eines Repositorys in einem CodeCommit Repository zu ändern, führen Sie den folgenden Befehl aus:

- [update-default-branch](#) (Damit ändern Sie den Standard-Branch.)

So ändern Sie den Standard-Branch

1. Führen Sie den Befehl update-default-branch aus und geben Sie Folgendes an:
 - Der Name des CodeCommit Repositorys, in dem der Standard-Branch aktualisiert wird (mit der --repository-name Option).

 Tip

Um den Namen des CodeCommit Repositorys abzurufen, führen Sie den Befehl [list-repositories](#) aus.

- Der Name des neuen Standard-Branches (mit der Option `--default-branch-name`).

 Tip

[Um den Namen des Branches abzurufen, führen Sie den Befehl `list-branches` aus.](#)

2. Um beispielsweise den Standard-Branch `MyNewBranch` in ein CodeCommit Repository mit dem Namen zu ändern: `MyDemoRepo`

```
aws codecommit update-default-branch --repository-name MyDemoRepo --default-branch-name MyNewBranch
```

Dieser Befehl liefert nur eine Ausgabe, wenn Fehler aufgetreten sind.

Weitere Optionen findest du in deiner Git-Dokumentation.

Lösche einen Zweig in AWS CodeCommit

Sie können die CodeCommit Konsole verwenden, um einen Branch in einem Repository zu löschen. Wenn Sie einen Branch in löschen, wird dieser Branch in einem lokalen Repository CodeCommit nicht gelöscht, sodass Benutzer möglicherweise weiterhin Kopien dieses Branches haben, bis sie das nächste Mal Änderungen abrufen. Um einen Branch lokal zu löschen und diese Änderung in das CodeCommit Repository zu übertragen, verwende Git von einem lokalen Repo, das mit dem CodeCommit Repository verbunden ist.

Durch das Löschen eines Branch werden keine Commits gelöscht, sondern alle Verweise auf die Commits in diesem Branch. Wenn du einen Branch löschst, der Commits enthält, die nicht mit einem anderen Branch im Repository zusammengeführt wurden, kannst du diese Commits nur abrufen, wenn du den vollständigen Commit hast. IDs

 Note

Sie können die Anweisungen in diesem Thema nicht verwenden, um den Standard-Branch eines Repository zu löschen. Wenn Sie den Standard-Branch löschen möchten, müssen Sie einen Branch erstellen, diesen zum Standard-Branch machen und dann den alten Branch löschen. Weitere Informationen erhalten Sie unter [Erstellen eines Zweigs](#) und [Ändern Sie die Filialeinstellungen](#).

Themen

- [Löscht einen Branch \(Konsole\)](#)
- [Lösche einen Zweig \(\)AWS CLI](#)
- [Einen Zweig löschen \(Git\)](#)

Löscht einen Branch (Konsole)

Sie können die CodeCommit Konsole verwenden, um einen Branch in einem CodeCommit Repository zu löschen.

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, in dem Sie einen Branch löschen möchten.
3. Wählen Sie im Navigationsbereich Branches aus.
4. Suchen Sie den Namen des Branches, den Sie löschen möchten, klicken Sie auf Delete branch (Branch löschen) und bestätigen Sie dies.

Lösche einen Zweig ()AWS CLI

Sie können den verwenden AWS CLI , um einen Branch in einem CodeCommit Repository zu löschen, falls dieser Branch nicht der Standard-Branch für das Repository ist. Weitere Hinweise zur Installation und Verwendung von finden Sie unter [Befehlszeilenreferenz](#). AWS CLI

1. Führen Sie am Terminal oder in der Befehlszeile den Befehl delete-branch unter Angabe der folgenden Informationen aus:

- Der Name des CodeCommit Repositorys, in dem der Branch gelöscht werden soll (mit der `--repository-name` Option).

 Tip

Um den Namen des CodeCommit Repositorys abzurufen, führen Sie den Befehl [list-repositories](#) aus.

- Name des zu löschenden Branches (mit der Option `branch-name`)

 Tip

[Um den Namen des Branches abzurufen, führen Sie den Befehl list-branches aus.](#)

2. Um beispielsweise einen Branch zu löschen, der MyNewBranch in einem CodeCommit Repository mit dem folgenden Namen benannt ist: MyDemoRepo

```
aws codecommit delete-branch --repository-name MyDemoRepo --branch-name MyNewBranch
```

Dieser Befehl gibt Informationen über den gelöschten Branch zurück, z. B. den Namen des gelöschten Branch sowie die vollständige Commit-ID des Commits, das Kopf des Branch war. Zum Beispiel:

```
"deletedBranch": {  
  "branchName": "MyNewBranch",  
  "commitId": "317f8570EXAMPLE"  
}
```

Einen Zweig löschen (Git)

Gehen Sie wie folgt vor, um mit Git aus einem lokalen Repository einen Branch in einem CodeCommit Repository zu löschen.

Diese Schritte wurden unter der Annahme geschrieben, dass Sie das lokale Repository bereits mit dem Repository verbunden haben. CodeCommit Detaillierte Anweisungen finden Sie unter [Herstellen einer Verbindung mit einem Repository](#).

1. Um den Branch aus dem lokalen Repository zu löschen, führen Sie den `git branch -D branch-name` Befehl aus, bei dem der Name des Branches **branch-name** steht, den Sie löschen möchten.

 Tip

Um eine Liste der Branch-Namen zu erhalten, führen Sie den Befehl `git branch --all` aus.

Um beispielsweise einen Zweig im lokalen Repository mit dem Namen zu löschen:

MyNewBranch

```
git branch -D MyNewBranch
```

2. Um den Branch aus dem CodeCommit Repository zu löschen, führen Sie den `git push remote-name --delete branch-name` Befehl aus, bei dem **remote-name** es sich um den Spitznamen handelt, den das lokale Repository für das CodeCommit Repository verwendet, und **branch-name** um den Namen des Branches, den Sie aus dem CodeCommit Repository löschen möchten.

 Tip

Um eine Liste der CodeCommit Repository-Namen und ihrer Namen zu erhalten URLs, führen Sie den `git remote -v` Befehl aus.

Um beispielsweise einen Branch zu löschen, der MyNewBranch im CodeCommit Repository wie folgt benannt ist `origin`:

```
git push origin --delete MyNewBranch
```

 Tip

Mit diesem Befehl wird der Branch nicht gelöscht, wenn es sich dabei um den Standard-Branch handelt.

Weitere Optionen findest du in deiner Git-Dokumentation.

Mit Benutzereinstellungen arbeiten

Sie können die AWS CodeCommit Konsole verwenden, um einige Standardeinstellungen zu konfigurieren. Beispielsweise können Sie Ihre Einstellungen für die Anzeige der Codeänderungen als inline oder in einer geteilten Ansicht festlegen. Wenn Sie eine dieser Einstellungen ändern, setzt die AWS CodeCommit Konsole ein Cookie in Ihrem Browser, das Ihre Auswahl bei jeder Nutzung der Konsole speichert und anwendet. Diese Einstellungen werden jedes Mal, wenn Sie mit diesem Browser auf die AWS CodeCommit Konsole zugreifen, auf alle Repositorys in allen Regionen angewendet. Diese Einstellungen gelten weder nur für ein bestimmtes Repository noch für eine bestimmte Region. Sie haben keinen Einfluss auf Ihre Interaktionen mit der AWS CLI AWS CodeCommit API oder anderen Diensten, mit AWS CodeCommit denen Sie interagieren.

Note

Die Cookies für die Benutzereinstellungen sind browserspezifisch. Wenn Sie die Cookies in Ihrem Browser löschen, werden auch die Einstellungen gelöscht. Greifen Sie mit einem anderen Browser auf ein Repository zu, haben Sie mit diesem Browser keinen Zugriff auf die Cookies des ursprünglichen Browsers. Ihre Einstellungen werden nicht gespeichert.

Benutzereinstellungen umfassen:

- Ob beim Anzeigen von Codeänderungen die Ansicht Unified oder Split verwendet und ob geänderte Leerzeichen ein- oder ausgeblendet werden sollen.
- Ob im Code-Editorfenster beim Anzeigen, Bearbeiten oder Erstellen von Code, ein heller oder dunkler Hintergrund verwendet werden soll.

Es gibt keine spezielle Seite, um all Ihre Einstellungen festzulegen. Stattdessen werden Ihre Änderungen an den Einstellungen in der Konsole, beispielsweise wie Sie Codeänderungen anzeigen möchten, gespeichert und nach Bedarf angewendet.

Migrieren zu AWS CodeCommit

Du kannst ein Git-Repository auf verschiedene Arten in ein CodeCommit Repository migrieren: indem du es klonst, spiegelst, alle oder nur einige der Branches migrierst und so weiter. Auch lokale und nicht versionierte Inhalte von Ihrem Computer können Sie in CodeCommit migrieren.

In den folgenden Themen werden einige Methoden für die Repository-Migration vorgestellt. Ihre Schritte sind möglicherweise anders, dies ist abhängig vom Typ, von der Art oder der Komplexität Ihres Repositories und von den Entscheidungen, die Sie in Bezug auf die zu migrierenden Inhalte und die Migrationsmethode getroffen haben. Bei sehr umfangreichen Repositories kann eine [inkrementelle Migration](#) sinnvoll sein.

Note

Sie können zu CodeCommit anderen Versionskontrollsystemen wie Perforce, Subversion oder TFS migrieren, müssen aber zuerst zu Git migrieren.

Weitere Optionen findest du in deiner Git-Dokumentation.

Alternativ können Sie die Informationen für eine [Migration in Git](#) im Buch Pro Git von Scott Chacon und Ben Straub heranziehen.

Themen

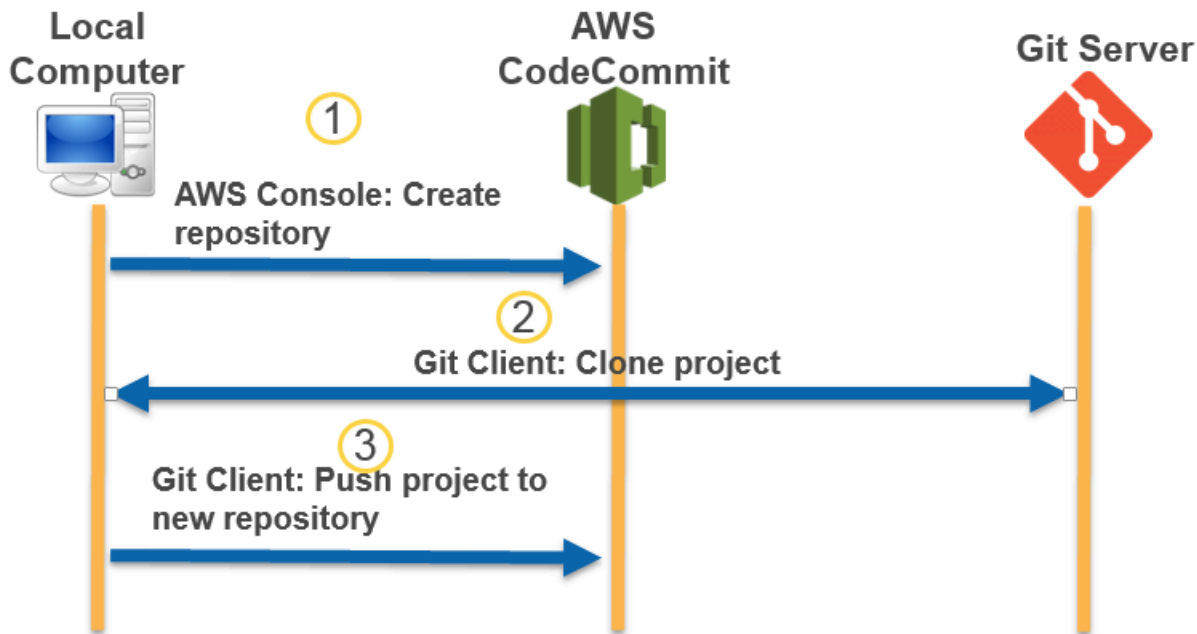
- [Migrieren Sie ein Git-Repository zu AWS CodeCommit](#)
- [Migrieren Sie lokalen oder unversionierten Inhalt zu AWS CodeCommit](#)
- [Ein Repository inkrementell migrieren](#)

Migrieren Sie ein Git-Repository zu AWS CodeCommit

Sie können ein vorhandenes Git-Repository in ein CodeCommit Repository migrieren. Das Verfahren in diesem Thema führt Sie durch die Migration eines auf einem anderen Git-Repository gehosteten Projekts zu CodeCommit. Als Teil dieses Prozesses werden Sie Folgendes durchführen:

- Schließen Sie die Ersteinrichtung ab, die für erforderlich ist CodeCommit.
- Erstellen Sie ein CodeCommit Repository.
- Klonen Sie das Repository und übertragen Sie es dorthin CodeCommit.

- Dateien im CodeCommit Repository anzeigen.
- Teilen Sie das CodeCommit Repository mit Ihrem Team.



Themen

- [Schritt 0: Für den Zugriff auf ist eine Einrichtung erforderlich CodeCommit](#)
- [Schritt 1: Erstellen Sie ein Repository CodeCommit](#)
- [Schritt 2: Das Repository klonen und zum CodeCommit Repository pushen](#)
- [Schritt 3: Dateien anzeigen in CodeCommit](#)
- [Schritt 4: Teilen Sie das Repository CodeCommit](#)

Schritt 0: Für den Zugriff auf ist eine Einrichtung erforderlich CodeCommit

Bevor Sie ein Repository zu migrieren können CodeCommit, müssen Sie einen IAM-Benutzer für den Zugriff erstellen und konfigurieren CodeCommit und Ihren lokalen Computer für den Zugriff konfigurieren. Außerdem sollten Sie die AWS CLI für die Verwaltung von CodeCommit installieren. Obwohl Sie die meisten CodeCommit Aufgaben auch ohne Git ausführen können, AWS CLI bietet es Flexibilität, wenn Sie mit Git über die Befehlszeile oder das Terminal arbeiten.

Wenn Sie bereits dafür eingerichtet sind CodeCommit, können Sie direkt zu [Schritt 1: Erstellen Sie ein Repository CodeCommit](#).

Um einen IAM-Benutzer für den Zugriff zu erstellen und zu konfigurieren CodeCommit

1. Erstellen Sie ein Amazon Web Services Services-Konto, indem Sie zu <http://aws.amazon.com> gehen und Sign Up wählen.
2. Erstellen Sie einen IAM-Benutzer oder verwenden Sie einen vorhandenen in Ihrem Amazon Web Services Services-Konto. Stellen Sie sicher, dass Sie über eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel verfügen, die diesem IAM-Benutzer zugeordnet sind. Weitere Informationen finden Sie unter [Einen IAM-Benutzer in Ihrem Amazon Web Services Services-Konto erstellen](#).

 Note

CodeCommit erfordert AWS Key Management Service. Wenn Sie einen vorhandenen IAM-Benutzer verwenden, stellen Sie sicher, dass dem Benutzer keine Richtlinien zugeordnet sind, die ausdrücklich die von CodeCommit erforderlichen AWS KMS Aktionen verweigern. Weitere Informationen finden Sie unter [AWS KMS und Verschlüsselung](#).

3. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter <https://console.aws.amazon.com/iam/>
4. Wählen Sie in der IAM-Konsole im Navigationsbereich Benutzer und dann den IAM-Benutzer aus, den Sie für den Zugriff konfigurieren möchten. CodeCommit
5. Wählen Sie auf der Registerkarte Permissions die Option Add Permissions.
6. Wählen Sie unter Grant permissions die Option Attach existing policies directly aus.
7. Wählen Sie AWSCodeCommitPowerUser aus der Liste der Richtlinien eine andere verwaltete Richtlinie für CodeCommit den Zugriff aus. Weitere Informationen finden Sie unter [AWS verwaltete Richtlinien für CodeCommit](#).

Nachdem Sie die Richtlinie ausgewählt haben, die Sie anhängen möchten, wählen Sie Weiter: Überprüfen aus, um die Liste der Richtlinien zu überprüfen, die an den IAM-Benutzer angehängt werden sollen. Ist die Liste korrekt, wählen Sie Add permissions aus.

Weitere Informationen zu CodeCommit verwalteten Richtlinien und zur gemeinsamen Nutzung des Zugriffs auf Repositories mit anderen Gruppen und Benutzern finden Sie unter [Teilen Sie ein Repository](#) und [Authentifizierung und Zugriffskontrolle für AWS CodeCommit](#)

Um das zu installieren und zu konfigurieren AWS CLI

1. Laden Sie auf Ihrem lokalen Computer das herunter und installieren Sie es AWS CLI. Dies ist eine Voraussetzung für die Interaktion mit CodeCommit über die Befehlszeile. Wir empfehlen, AWS CLI Version 2 zu installieren. Es ist die neueste Hauptversion von AWS CLI und unterstützt alle aktuellen Funktionen. Es ist die einzige Version von AWS CLI , die die Verwendung eines Root-Kontos, Verbundzugriff oder temporärer Anmeldeinformationen mit git-remote-codecommit unterstützt.

Weitere Informationen finden Sie [unter Einrichtung der AWS Befehlszeilenschnittstelle](#).

 Note

CodeCommit funktioniert nur mit den AWS CLI Versionen 1.7.38 und höher. Es hat sich bewährt, die neueste verfügbare Version zu installieren oder AWS CLI auf sie zu aktualisieren. Führen AWS CLI Sie den `aws --version` Befehl aus, um festzustellen, welche Version von Sie installiert haben.

Informationen zum Upgrade einer älteren Version von AWS CLI auf die neueste Version finden Sie unter [Installation von AWS Command Line Interface](#).

2. Führen Sie diesen Befehl aus, um zu überprüfen, ob die CodeCommit Befehle für installiert AWS CLI sind.

```
aws codecommit help
```

Dieser Befehl gibt eine Liste von CodeCommit Befehlen zurück.

3. Konfigurieren Sie das AWS CLI mit einem Profil, indem Sie den `configure` Befehl verwenden, wie folgt:.

```
aws configure
```

Wenn Sie dazu aufgefordert werden, geben Sie den AWS Zugriffsschlüssel und den AWS geheimen Zugriffsschlüssel des IAM-Benutzers an, mit CodeCommit dem Sie ihn verwenden möchten. Stellen Sie außerdem sicher, dass Sie angeben, AWS-Region wo sich das Repository befindet, z. B. `us-east-2`. Wenn Sie nach dem standardmäßigen Ausgabeformat gefragt werden, geben Sie `json` an. Wenn Sie beispielsweise ein Profil für einen IAM-Benutzer konfigurieren:

```
AWS Access Key ID [None]: Type your IAM user AWS access key ID here, and then press Enter
AWS Secret Access Key [None]: Type your IAM user AWS secret access key here, and then press Enter
Default region name [None]: Type a supported region for CodeCommit here, and then press Enter
Default output format [None]: Type json here, and then press Enter
```

Weitere Informationen zum Erstellen und Konfigurieren von Profilen zur Verwendung mit dem AWS CLI finden Sie im Folgenden:

- [Benannte Profile](#)
- [Verwenden einer IAM-Rolle in der AWS CLI](#)
- [Befehl „Set“](#)
- [Verbindung zu AWS CodeCommit Repositories mit rotierenden Anmeldeinformationen herstellen](#)

Um eine Verbindung zu einem Repository oder einer Ressource in einem anderen Repository herzustellen, müssen Sie das AWS CLI mit dem Standardregionsnamen neu konfigurieren. Zu den unterstützten Standardregionsnamen für CodeCommit gehören:

- `us-east-2`
- `us-east-1`
- `eu-west-1`
- `us-west-2`
- `ap-northeast-1`
- `ap-southeast-1`
- `ap-southeast-2`

- ap-southeast-3
- me-central-1
- eu-central-1
- ap-northeast-2
- sa-east-1
- us-west-1
- eu-west-2
- ap-south-1
- ap-south-1
- ca-central-1
- us-gov-west-1
- us-gov-east-1
- eu-north-1
- ap-east-1
- me-south-1
- cn-north-1
- cn-northwest-1
- eu-south-1
- ap-northeast-3
- af-south-1
- il-central-1

Weitere Informationen zu CodeCommit und finden AWS-Region Sie unter [Regionen und Git-Verbindungsendpunkte](#). Weitere Informationen zu IAM, Zugriffsschlüsseln und geheimen Schlüsseln finden Sie unter [Wie erhalte ich Anmeldeinformationen?](#) und [Zugriffsschlüssel für IAM-Benutzer verwalten](#). Weitere Informationen zu den Profilen AWS CLI und finden Sie unter [Benannte Profile](#).

Als Nächstes müssen Sie Git installieren.

- Für Linux, macOS oder Unix:

Um mit Dateien, Commits und anderen Informationen in CodeCommit Repositorys zu arbeiten, müssen Sie Git auf Ihrem lokalen Computer installieren. CodeCommit unterstützt Git-Versionen 1.7.9 und höher. Git Version 2.28 unterstützt die Konfiguration des Branchnamens für anfängliche Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git.

Um Git zu installieren, empfehlen wir Websites wie [Git Downloads](#).

 Note

Git ist eine sich entwickelnde, regelmäßig aktualisierte Plattform. Gelegentlich kann sich eine Änderung der Funktionen auf die Art und Weise auswirken, mit der es funktioniert CodeCommit. Wenn du Probleme mit einer bestimmten Version von Git und hast CodeCommit, sieh dir die Informationen unter an [Fehlerbehebung](#).

- Für Windows:

Um mit Dateien, Commits und anderen Informationen in CodeCommit Repositorys zu arbeiten, müssen Sie Git auf Ihrem lokalen Computer installieren. CodeCommit unterstützt Git-Versionen 1.7.9 und höher. Git Version 2.28 unterstützt die Konfiguration des Branchnamens für anfängliche Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git.

Um Git zu installieren, empfehlen wir Websites wie [Git für Windows](#). Wenn du diesen Link verwendest, um Git zu installieren, kannst du alle Standardeinstellungen der Installation akzeptieren, mit Ausnahme der folgenden:

- Wenn Sie während des Schritts Anpassen Ihrer PATH-Umgebung dazu aufgefordert werden, wählen Sie in der Befehlszeile die Option, Git zu verwenden.
- (Optional) Wenn Sie beabsichtigen, HTTPS mit dem Credential Helper zu verwenden, der in AWS CLI statt der Konfiguration von Git-Anmeldeinformationen für enthalten ist CodeCommit, auf der Seite Zusätzliche Optionen konfigurieren, stellen Sie sicher, dass die Option Git Credential Manager aktivieren deaktiviert ist. Der Git Credential Manager ist nur kompatibel, CodeCommit wenn IAM-Benutzer Git-Anmeldeinformationen konfigurieren. Weitere Informationen erhalten Sie unter [Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden](#) und [Git für Windows: Ich habe Git für Windows installiert, aber mir wird der Zugriff auf mein Repository verweigert \(403\)](#).

 Note

Git ist eine sich entwickelnde, regelmäßig aktualisierte Plattform. Gelegentlich kann sich eine Änderung der Funktionen auf die Art und Weise auswirken, mit der es funktioniert CodeCommit. Wenn du Probleme mit einer bestimmten Version von Git und hast CodeCommit, sieh dir die Informationen unter [anFehlerbehebung](#).

CodeCommit unterstützt sowohl HTTPS- als auch SSH-Authentifizierung. Um die Einrichtung abzuschließen, müssen Sie Git-Anmeldeinformationen für CodeCommit (HTTPS, für die meisten Benutzer empfohlen), ein SSH-Schlüsselpaar für den Zugriff CodeCommit (SSH) `git-remote-codecommit` (empfohlen für Benutzer, die Verbundzugriff verwenden) oder den AWS CLI in (HTTPS) enthaltenen Credential Helper konfigurieren.

- Informationen zu Git-Anmeldeinformationen auf allen unterstützten Betriebssystemen finden Sie unter [Schritt 3: Erstellen Sie Git-Anmeldeinformationen für HTTPS-Verbindungen zu CodeCommit](#).
- Informationen zu SSH unter Linux, macOS oder Unix finden Sie unter [SSH und Linux, macOS oder Unix: Richten Sie die öffentlichen und privaten Schlüssel für Git ein und CodeCommit](#).
- Informationen zu SSH unter Windows finden Sie unter [Schritt 3: Richten Sie die öffentlichen und privaten Schlüssel für Git ein und CodeCommit](#).
- Informationen für `git-remote-codecommit` finden Sie unter [Einrichtungsschritte für HTTPS-Verbindungen AWS CodeCommit mit git-remote-codecommit](#).
- Informationen zum Credential Helper unter Linux, macOS oder Unix finden Sie unter [Credential Helper einrichten \(Linux, macOS oder Unix\)](#).
- Informationen zum Hilfsprogramm für Anmeldeinformationen unter Windows finden Sie unter [Einrichten des Hilfsprogramms für Anmeldeinformationen \(Windows\)](#).

Schritt 1: Erstellen Sie ein Repository CodeCommit

In diesem Abschnitt verwenden Sie die CodeCommit Konsole, um das CodeCommit Repository zu erstellen, das Sie für den Rest dieses Tutorials verwenden. Informationen zur Verwendung von AWS CLI zum Erstellen des Repositorys finden Sie unter [Erstellen Sie ein Repository \(\)AWS CLI](#).

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.

2. Wählen Sie in der Regionsauswahl den Ort aus, AWS-Region an dem Sie das Repository erstellen möchten. Weitere Informationen finden Sie unter [Regionen und Git-Verbindungsendpunkte](#).
3. Wählen Sie auf der Seite Repositorys die Option Repository auswählen.
4. Geben Sie auf der Seite Create repository (Repository erstellen) im Feld Repository name (Repository-Name) einen Namen für das Repository ein.

 Note

Bei den Repository-Namen wird nach Groß- und Kleinschreibung unterschieden. Der Name muss AWS-Region für Ihr Amazon Web Services Services-Konto eindeutig sein.

5. (Optional) Geben Sie unter Description (Beschreibung) eine Beschreibung für das Repository ein. Dadurch können Sie und andere Benutzer den Zweck des Repositorys leichter erkennen.

 Note

Das Beschreibungsfeld zeigt Markdown in der Konsole an und akzeptiert alle HTML-Zeichen und gültigen Unicode-Zeichen. Wenn Sie als Anwendungsentwickler das GetRepository oder verwenden BatchGetRepositories APIs und das Repository-Beschreibungsfeld in einem Webbrowser anzeigen möchten, finden Sie weitere Informationen in der [CodeCommit API-Referenz](#).

6. (Optional) Wählen Sie Tag hinzufügen, um Ihrem Repository ein oder mehrere Repository-Tags (eine benutzerdefinierte Attributbezeichnung, mit der Sie Ihre AWS Ressourcen organisieren und verwalten können) hinzuzufügen. Weitere Informationen finden Sie unter [Kennzeichen von Repositorys in AWS CodeCommit](#).
7. (Optional) Erweitern Sie Zusätzliche Konfiguration, um anzugeben, ob Sie den Standardschlüssel Von AWS verwalteter Schlüssel oder Ihren eigenen, vom Kunden verwalteten Schlüssel zum Verschlüsseln und Entschlüsseln von Daten in diesem Repository verwenden möchten. Wenn Sie Ihren eigenen, vom Kunden verwalteten Schlüssel verwenden möchten, müssen Sie sicherstellen, dass er dort verfügbar ist, AWS-Region wo Sie das Repository erstellen, und dass der Schlüssel aktiv ist. Weitere Informationen finden Sie unter [AWS Key Management Service und Verschlüsselung für AWS CodeCommit Repositorien](#).
8. (Optional) Wählen Sie Amazon CodeGuru Reviewer für Java und Python aktivieren, wenn dieses Repository Java- oder Python-Code enthält und Sie möchten, dass CodeGuru Reviewer ihn analysiert. CodeGuru Reviewer verwendet mehrere Modelle für maschinelles Lernen, um

Codefehler zu finden und Verbesserungen und Korrekturen in Pull-Requests vorzuschlagen. Weitere Informationen finden Sie im [Amazon CodeGuru Reviewer-Benutzerhandbuch](#).

9. Wählen Sie Create (Erstellen) aus.

Developer Tools > CodeCommit > Repositories > Create repository

Create repository

Create a secure repository to store and share your code. Begin by typing a repository name and a description for your repository. Repository names are included in the URLs for that repository.

Repository settings

Repository name

MyClonedRepository

100 characters maximum. Other limits apply.

createRepository.form.field.repositoryDescription.label - optional

This repository was initially cloned from https://

1,000 characters maximum

Cancel Create

Nachdem das Repository erstellt wurde, wird es in der Liste Repositories angezeigt. Wählen Sie in der URL-Spalte das Kopiersymbol aus und wählen Sie dann das Protokoll (SSH oder HTTPS) aus, das Sie zum Herstellen der Verbindung mit CodeCommit verwenden möchten. Kopieren Sie die URL.

Wenn Sie beispielsweise Ihrem Repository einen Namen gegeben haben *MyClonedRepository* und Git-Anmeldeinformationen mit HTTPS in der Region USA Ost (Ohio) verwenden, sieht die URL wie folgt aus:

```
https://git-codecommit.us-east-2.amazonaws.com/MyClonedRepository
```

Sie benötigen diese URL später in [Schritt 2: Das Repository klonen und zum CodeCommit Repository pushen](#).

Schritt 2: Das Repository klonen und zum CodeCommit Repository pushen

In diesem Abschnitt klonen Sie ein Git-Repository auf Ihren lokalen Computer und erstellen somit ein lokales Repository. Anschließend übertragen Sie den Inhalt des lokalen Repos in das CodeCommit Repository, das Sie zuvor erstellt haben.

1. Führen Sie über das Terminal oder die Befehlszeile auf Ihrem lokalen Computer den `git clone` Befehl mit der `--mirror` Option aus, eine bloße Kopie des Remote-Repositorys in einen neuen Ordner mit dem Namen `aws-codecommit-demo` zu klonen. Hierbei handelt es sich um ein einfaches Repository nur für die Migration. Es ist nicht das lokale Repo für die Interaktion mit dem migrierten Repository in CodeCommit. Sie können es später erstellen, nachdem die Migration zu abgeschlossen CodeCommit ist.

Im folgenden Beispiel wird eine auf GitHub (<https://github.com/aws-labs/aws-demo-php-simple-app.git>) gehostete Demo-Anwendung in ein lokales Repository in einem Verzeichnis mit dem Namen geklont. `aws-codecommit-demo`

```
git clone --mirror https://github.com/aws-labs/aws-demo-php-simple-app.git aws-codecommit-demo
```

2. Ändern Sie die Verzeichnisse in das Verzeichnis, in dem Sie den Klon erstellt haben.

```
cd aws-codecommit-demo
```

3. Führen Sie den `git push` Befehl aus und geben Sie die URL und den Namen des CodeCommit Ziel-Repositorys sowie die Option `--all`. (Dies ist die URL, die Sie in [Schritt 1: Erstellen Sie ein Repository CodeCommit](#) kopiert haben.)

Wenn Sie beispielsweise Ihr Repository benannt haben `MyClonedRepository` und für die Verwendung von HTTPS eingerichtet sind, würden Sie den folgenden Befehl ausführen:

```
git push https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyClonedRepository --all
```

Note

Die Option `--all` überträgt per Push ausschließlich alle Branches für das Repository. Andere Referenzen, z. B. Tags, werden nicht übertragen. Wenn Sie Tags per Push

übertragen möchten, warten Sie, bis der ursprüngliche Push-Vorgang abgeschlossen ist, und führen Sie einen weiteren aus, diesmal unter Verwendung der Option `--tags`:

```
git push ssh://git-codecommit.us-east-2.amazonaws.com/v1/
repos/MyClonedRepository --tags
```

Weitere Informationen finden Sie unter [Git Push](#) auf der Git-Website. Informationen zur Push-Übertragung großer Repositories, insbesondere wenn alle Referenzen gleichzeitig per Push übertragen werden sollen, (z. B. mit der Option `--mirror`), finden Sie unter [Migrieren Sie ein Repository schrittweise](#).

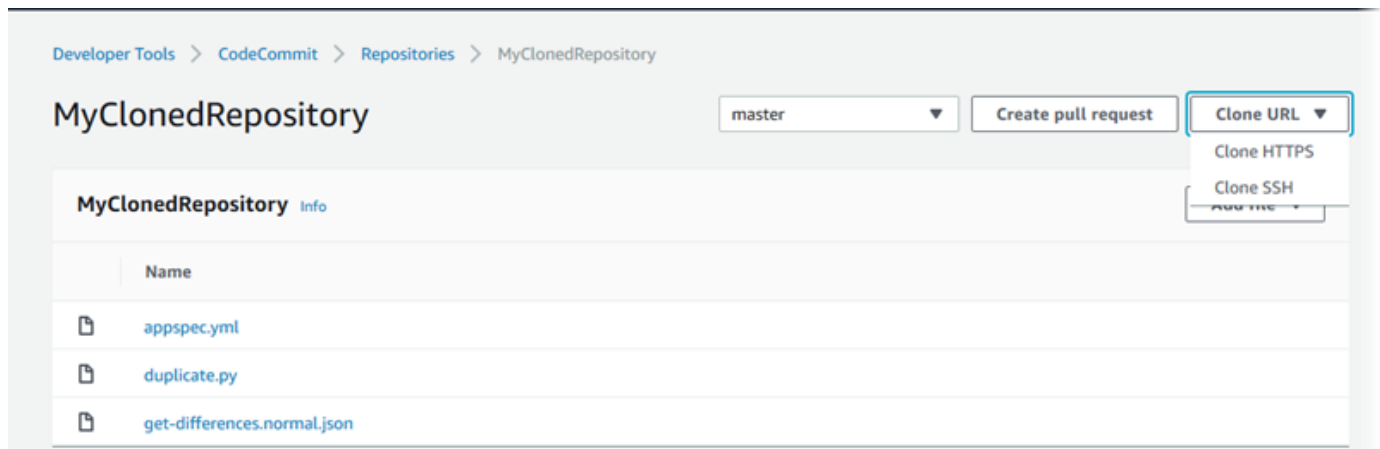
Sie können den `aws-codecommit-demo` Ordner und seinen Inhalt löschen, nachdem Sie das Repository zu CodeCommit migriert haben. Um ein lokales Repo mit allen korrekten Referenzen für die Arbeit mit dem Repository zu erstellen CodeCommit, führen Sie den `git clone` Befehl ohne die `--mirror` Option aus:

```
git clone https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyClonedRepository
```

Schritt 3: Dateien anzeigen in CodeCommit

Nachdem Sie den Inhalt Ihres Verzeichnisses per Push übertragen haben, können Sie die CodeCommit Konsole verwenden, um schnell alle Dateien in diesem Repository anzuzeigen.

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories den Namen des Repositories aus (z. B.). `MyClonedRepository`
3. Sehen Sie sich die Dateien im Repository für die Branches, den Clone URLs, die Einstellungen und mehr an.



Schritt 4: Teilen Sie das Repository CodeCommit

Wenn Sie ein Repository in erstellen CodeCommit, werden zwei Endpunkte generiert: einer für HTTPS-Verbindungen und einer für SSH-Verbindungen. Beide bieten sichere Verbindungen über ein Netzwerk. Ihre Benutzer können beide Protokolle verwenden. Beide Endpunkte bleiben aktiv, unabhängig davon, welches Protokoll Sie Ihren Benutzern empfehlen. Bevor Sie Ihr Repository mit anderen teilen können, müssen Sie IAM-Richtlinien erstellen, die anderen Benutzern Zugriff auf Ihr Repository gewähren. Stellen Sie Ihren Benutzern diese Zugriffsanweisungen zur Verfügung.

Erstellen einer kundenverwalteten Richtlinie für das Repository

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter <https://console.aws.amazon.com/iam/>
2. Wählen Sie im Navigationsbereich Dashboard die Optionen Policies und dann Create Policy aus.
3. Wählen Sie auf der Seite „Richtlinie erstellen“ die Option Verwaltete Richtlinie importieren aus.
4. Geben Sie auf der Seite Verwaltete Richtlinien importieren im Feld Richtlinien filtern den Text ein **AWSCodeCommitPowerUser**. Klicken Sie auf die Schaltfläche neben dem Richtliniennamen und wählen Sie dann Importieren aus.
5. Wählen Sie auf der Seite Create policy (Richtlinie erstellen) die Option JSON aus. Ersetzen Sie den Teil „*“ in der Resource Zeile für CodeCommit Aktionen durch den Amazon-Ressourcennamen (ARN) des CodeCommit Repositorys, wie hier gezeigt:

```
"Resource": [  
  "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo"  
]
```

 Tip

Um den ARN für das CodeCommit Repository zu finden, gehen Sie zur CodeCommit Konsole, wählen Sie den Repository-Namen aus der Liste aus und wählen Sie dann Einstellungen. Weitere Informationen finden Sie unter [Repository-Details anzeigen](#).

Wenn Sie diese Richtlinie auf mehr als ein Repository anwenden möchten, fügen Sie jedes Repository als Ressource hinzu, indem Sie seinen ARN angeben. Fügen Sie zwischen den Ressourcenanweisungen ein Komma ein, wie hier dargestellt:

```
"Resource": [  
  "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo",  
  "arn:aws:codecommit:us-east-2:111111111111:MyOtherDemoRepo"  
]
```

Wenn Sie mit der Bearbeitung fertig sind, wählen Sie „Richtlinie überprüfen“.

6. Geben Sie auf der Seite „Richtlinie überprüfen“ im Feld Name einen neuen Namen für die Richtlinie ein (z. B. *AWSCodeCommitPowerUser-MyDemoRepo*). Geben Sie optional eine Beschreibung für diese Richtlinie ein.
7. Wählen Sie Create Policy (Richtlinie erstellen) aus.

Um den Zugriff auf Ihr Repository zu verwalten, erstellen Sie eine IAM-Gruppe für deren Benutzer, fügen Sie dieser Gruppe IAM-Benutzer hinzu und fügen Sie dann die vom Kunden verwaltete Richtlinie hinzu, die Sie im vorherigen Schritt erstellt haben. Fügen Sie alle anderen Richtlinien hinzu, die für den Zugriff erforderlich sind, z. B. oder IAMUser SSHKeys IAMSelfManageServiceSpecificCredentials.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter <https://console.aws.amazon.com/iam/>.
2. Wählen Sie im Navigationsbereich Dashboard die Option Groups und dann Create New Group aus.
3. Geben Sie auf der Seite Set Group Name (Name für Gruppe festlegen) im Feld Group Name (Gruppenname) einen Namen für die Gruppe ein (zum Beispiel *MyDemoRepoGroup*) und wählen Sie dann Next Step (Nächster Schritt) aus. Sie sollten den Repository-Namen als Teil des Gruppennamens einfügen.

 Note

Dieser Name muss in einem Amazon Web Services Services-Konto eindeutig sein.

4. Markieren Sie das Kästchen neben der vom Kunden verwalteten Richtlinie, die Sie im vorherigen Abschnitt erstellt haben (z. B. AWSCodeCommitPowerUser-MyDemoRepo).
5. Klicken Sie auf der Seite Review auf Create Group. IAM erstellt diese Gruppe mit den angegebenen Richtlinien, die bereits angehängt sind. Die Gruppe erscheint in der Liste der Gruppen, die mit Ihrem Amazon Web Services Services-Konto verknüpft sind.
6. Wählen Sie Ihre Gruppe in der Liste aus.
7. Wählen Sie auf der Gruppenübersichtsseite die Registerkarte Users und anschließend die Option Add Users to Groups aus. Markieren Sie in der Liste, in der alle mit Ihrem Amazon Web Services Services-Konto verknüpften Benutzer angezeigt werden, die Kästchen neben den Benutzern, denen Sie Zugriff auf das CodeCommit Repository gewähren möchten, und wählen Sie dann Benutzer hinzufügen.

 Tip

Sie können über das Suchfeld schnell nach Benutzern anhand ihres Namens suchen.

8. Wenn Sie Ihre Benutzer hinzugefügt haben, schließen Sie die IAM-Konsole.

Nachdem Sie mit der von Ihnen konfigurierten Richtliniengruppe und den Richtlinien einen IAM-Benutzer für den CodeCommit Zugriff erstellt haben, senden Sie diesem Benutzer die Informationen, die für die Verbindung mit dem Repository erforderlich sind.

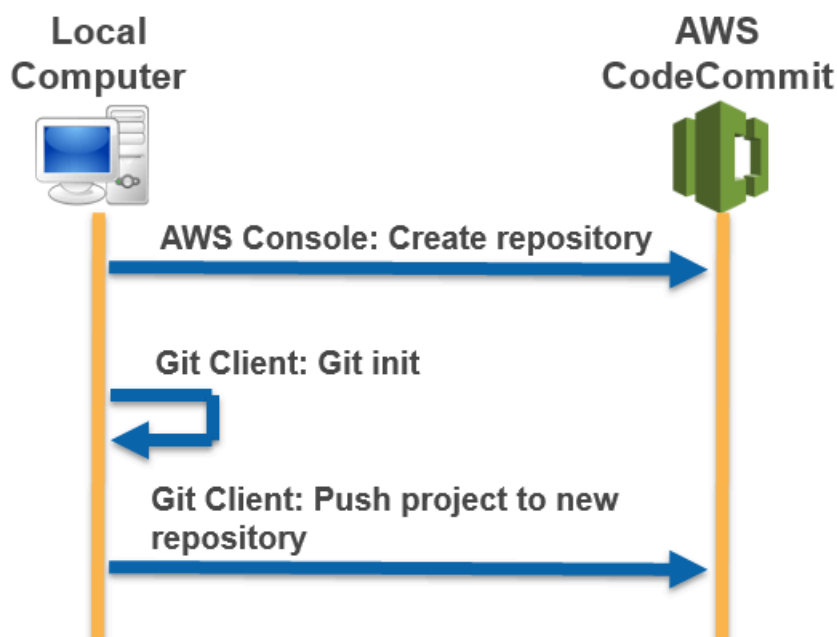
1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie in der Regionsauswahl den Ort aus, AWS-Region an dem das Repository erstellt wurde. Repositorys sind spezifisch für ein. AWS-Region Weitere Informationen finden Sie unter [Regionen und Git-Verbindungsendpunkte](#).
3. Wählen Sie auf der Seite Repositories (Repositorys) das Repository aus, das Sie freigeben möchten.
4. Wählen Sie unter Clone URL (Klon-URL) das Protokoll aus, das die Benutzer verwenden sollen. Dadurch wird die Klon-URL für das Verbindungsprotokoll kopiert.

5. Senden Sie Ihren Benutzern die Klon-URL zusammen mit allen anderen Anweisungen, z. B. zur Installation von AWS CLI, zur Konfiguration eines Profils oder zur Installation von Git. Stellen Sie sicher, dass Sie die Konfigurationsinformationen für das Verbindungsprotokoll (z. B. HTTPS) angeben.

Migrieren Sie lokalen oder unversionierten Inhalt zu AWS CodeCommit

Die Verfahren in diesem Thema zeigen Ihnen, wie Sie ein vorhandenes Projekt oder lokale Inhalte auf Ihrem Computer in ein CodeCommit Repository migrieren. Als Teil dieses Prozesses werden Sie Folgendes durchführen:

- Schließen Sie die Ersteinrichtung ab, die für erforderlich ist CodeCommit.
- Erstellen Sie ein CodeCommit Repository.
- Platzieren Sie einen lokalen Ordner unter der Git-Versionskontrolle und übertragen Sie den Inhalt dieses Ordners in das CodeCommit Repository.
- Dateien im CodeCommit Repository anzeigen.
- Teilen Sie das CodeCommit Repository mit Ihrem Team.



Themen

- [Schritt 0: Für den Zugriff auf ist eine Einrichtung erforderlich CodeCommit](#)
- [Schritt 1: Erstellen Sie ein Repository CodeCommit](#)
- [Schritt 2: Migrieren Sie lokale Inhalte in das CodeCommit Repository](#)
- [Schritt 3: Dateien anzeigen in CodeCommit](#)
- [Schritt 4: Teilen Sie das CodeCommit Repository](#)

Schritt 0: Für den Zugriff auf ist eine Einrichtung erforderlich CodeCommit

Bevor Sie lokale Inhalte migrieren können CodeCommit, müssen Sie einen IAM-Benutzer für den Zugriff erstellen und konfigurieren CodeCommit und Ihren lokalen Computer für den Zugriff konfigurieren. Außerdem sollten Sie die AWS CLI für die Verwaltung von CodeCommit installieren. Obwohl Sie die meisten CodeCommit Aufgaben auch ohne Git ausführen können, AWS CLI bietet es Flexibilität bei der Arbeit mit Git.

Wenn Sie bereits dafür eingerichtet sind CodeCommit, können Sie direkt zu [Schritt 1: Erstellen Sie ein Repository CodeCommit](#).

Um einen IAM-Benutzer für den Zugriff zu erstellen und zu konfigurieren CodeCommit

1. Erstellen Sie ein Amazon Web Services Services-Konto, indem Sie zu <http://aws.amazon.com> gehen und Sign Up wählen.
2. Erstellen Sie einen IAM-Benutzer oder verwenden Sie einen vorhandenen in Ihrem Amazon Web Services Services-Konto. Stellen Sie sicher, dass Sie über eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel verfügen, die diesem IAM-Benutzer zugeordnet sind. Weitere Informationen finden Sie unter [Einen IAM-Benutzer in Ihrem Amazon Web Services Services-Konto erstellen](#).

Note

CodeCommit erfordert AWS Key Management Service. Wenn Sie einen vorhandenen IAM-Benutzer verwenden, stellen Sie sicher, dass dem Benutzer keine Richtlinien zugeordnet sind, die ausdrücklich die von CodeCommit erforderlichen AWS KMS Aktionen verweigern. Weitere Informationen finden Sie unter [AWS KMS und Verschlüsselung](#).

3. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter <https://console.aws.amazon.com/iam/>
4. Wählen Sie in der IAM-Konsole im Navigationsbereich Benutzer und dann den IAM-Benutzer aus, den Sie für den Zugriff konfigurieren möchten. CodeCommit
5. Wählen Sie auf der Registerkarte Permissions die Option Add Permissions.
6. Wählen Sie unter Grant permissions die Option Attach existing policies directly aus.
7. Wählen Sie AWSCodeCommitPowerUser aus der Liste der Richtlinien eine andere verwaltete Richtlinie für CodeCommit den Zugriff aus. Weitere Informationen finden Sie unter [AWS verwaltete Richtlinien für CodeCommit](#).

Nachdem Sie die Richtlinie ausgewählt haben, die Sie anhängen möchten, wählen Sie Weiter: Überprüfen aus, um die Liste der Richtlinien zu überprüfen, die an den IAM-Benutzer angehängt werden sollen. Ist die Liste korrekt, wählen Sie Add permissions aus.

Weitere Informationen zu CodeCommit verwalteten Richtlinien und zur gemeinsamen Nutzung des Zugriffs auf Repositorys mit anderen Gruppen und Benutzern finden Sie unter [Teilen Sie ein Repository](#) und [Authentifizierung und Zugriffskontrolle für AWS CodeCommit](#)

Um das zu installieren und zu konfigurieren AWS CLI

1. Laden Sie auf Ihrem lokalen Computer das herunter und installieren Sie es AWS CLI. Dies ist eine Voraussetzung für die Interaktion mit CodeCommit über die Befehlszeile. Wir empfehlen, AWS CLI Version 2 zu installieren. Es ist die neueste Hauptversion von AWS CLI und unterstützt alle aktuellen Funktionen. Es ist die einzige Version von AWS CLI , die die Verwendung eines Root-Kontos, Verbundzugriff oder temporärer Anmeldeinformationen mit git-remote-codecommit unterstützt.

Weitere Informationen finden Sie [unter Einrichtung der AWS Befehlszeilenschnittstelle](#).

 Note

CodeCommit funktioniert nur mit den AWS CLI Versionen 1.7.38 und höher. Es hat sich bewährt, die neueste verfügbare Version zu installieren oder AWS CLI auf sie zu aktualisieren. Führen AWS CLI Sie den `aws --version` Befehl aus, um festzustellen, welche Version von Sie installiert haben.

Informationen zum Upgrade einer älteren Version von AWS CLI auf die neueste Version finden Sie unter [Installation von AWS Command Line Interface](#).

2. Führen Sie diesen Befehl aus, um zu überprüfen, ob die CodeCommit Befehle für installiert AWS CLI sind.

```
aws codecommit help
```

Dieser Befehl gibt eine Liste von CodeCommit Befehlen zurück.

3. Konfigurieren Sie das AWS CLI mit einem Profil, indem Sie den configure Befehl wie folgt verwenden:.

```
aws configure
```

Wenn Sie dazu aufgefordert werden, geben Sie den AWS Zugriffsschlüssel und den AWS geheimen Zugriffsschlüssel des IAM-Benutzers an, mit CodeCommit dem Sie ihn verwenden möchten. Stellen Sie außerdem sicher, dass Sie angeben, AWS-Region wo sich das Repository befindet, z. B. us-east-2 Wenn Sie nach dem standardmäßigen Ausgabeformat gefragt werden, geben Sie json an. Wenn Sie beispielsweise ein Profil für einen IAM-Benutzer konfigurieren:

```
AWS Access Key ID [None]: Type your IAM user AWS access key ID here, and then press Enter
AWS Secret Access Key [None]: Type your IAM user AWS secret access key here, and then press Enter
Default region name [None]: Type a supported region for CodeCommit here, and then press Enter
Default output format [None]: Type json here, and then press Enter
```

Weitere Informationen zum Erstellen und Konfigurieren von Profilen zur Verwendung mit dem AWS CLI finden Sie im Folgenden:

- [Benannte Profile](#)
- [Verwenden einer IAM-Rolle in der AWS CLI](#)
- [Befehl „Set“](#)
- [Verbindung zu AWS CodeCommit Repositorys mit rotierenden Anmeldeinformationen herstellen](#)

Um eine Verbindung zu einem Repository oder einer Ressource in einem anderen Repository herzustellen AWS-Region, müssen Sie das AWS CLI mit dem Standardregionsnamen neu konfigurieren. Zu den unterstützten Standardregionsnamen für CodeCommit gehören:

- us-east-2
- us-east-1
- eu-west-1
- us-west-2
- ap-northeast-1
- ap-southeast-1
- ap-southeast-2
- ap-southeast-3
- me-central-1
- eu-central-1
- ap-northeast-2
- sa-east-1
- us-west-1
- eu-west-2
- ap-south-1
- ap-south-1
- ca-central-1
- us-gov-west-1
- us-gov-east-1
- eu-north-1
- ap-east-1
- me-south-1
- cn-north-1
- cn-northwest-1
- eu-south-1
- ap-northeast-3

- af-south-1
- il-central-1

Weitere Informationen zu CodeCommit und finden AWS-Region Sie unter [Regionen und Git-Verbindungsendpunkte](#). Weitere Informationen zu IAM, Zugriffsschlüsseln und geheimen Schlüsseln finden Sie unter [Wie erhalte ich Anmeldeinformationen?](#) und [Zugriffsschlüssel für IAM-Benutzer verwalten](#). Weitere Informationen zu den Profilen AWS CLI und finden Sie unter [Benannte Profile](#).

Als Nächstes müssen Sie Git installieren.

- Für Linux, macOS oder Unix:

Um mit Dateien, Commits und anderen Informationen in CodeCommit Repositorys zu arbeiten, müssen Sie Git auf Ihrem lokalen Computer installieren. CodeCommit unterstützt Git-Versionen 1.7.9 und höher. Git Version 2.28 unterstützt die Konfiguration des Branchnamens für anfängliche Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git.

Um Git zu installieren, empfehlen wir Websites wie [Git Downloads](#).

Note

Git ist eine sich entwickelnde, regelmäßig aktualisierte Plattform. Gelegentlich kann sich eine Änderung der Funktionen auf die Art und Weise auswirken, mit der es funktioniert CodeCommit. Wenn Sie Probleme mit einer bestimmten Version von Git und haben CodeCommit, lesen Sie die Informationen unter [Fehlerbehebung](#).

- Für Windows:

Um mit Dateien, Commits und anderen Informationen in CodeCommit Repositorys zu arbeiten, müssen Sie Git auf Ihrem lokalen Computer installieren. CodeCommit unterstützt Git-Versionen 1.7.9 und höher. Git Version 2.28 unterstützt die Konfiguration des Branchnamens für anfängliche Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git.

Um Git zu installieren, empfehlen wir Websites wie [Git für Windows](#). Wenn du diesen Link verwendest, um Git zu installieren, kannst du alle Standardeinstellungen der Installation akzeptieren, mit Ausnahme der folgenden:

- Wenn Sie während des Schritts Anpassen Ihrer PATH-Umgebung dazu aufgefordert werden, wählen Sie in der Befehlszeile die Option, Git zu verwenden.
- (Optional) Wenn Sie beabsichtigen, HTTPS mit dem Credential Helper zu verwenden, der in AWS CLI statt der Konfiguration von Git-Anmeldeinformationen für enthalten ist CodeCommit, auf der Seite Zusätzliche Optionen konfigurieren, stellen Sie sicher, dass die Option Git Credential Manager aktivieren deaktiviert ist. Der Git Credential Manager ist nur kompatibel, CodeCommit wenn IAM-Benutzer Git-Anmeldeinformationen konfigurieren. Weitere Informationen erhalten Sie unter [Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden](#) und [Git für Windows: Ich habe Git für Windows installiert, aber mir wird der Zugriff auf mein Repository verweigert \(403\)](#).

 Note

Git ist eine sich entwickelnde, regelmäßig aktualisierte Plattform. Gelegentlich kann sich eine Änderung der Funktionen auf die Art und Weise auswirken, mit der es funktioniert CodeCommit. Wenn Sie Probleme mit einer bestimmten Version von Git und haben CodeCommit, lesen Sie die Informationen unter [Fehlerbehebung](#).

CodeCommit unterstützt sowohl HTTPS- als auch SSH-Authentifizierung. Um die Installation abzuschließen, müssen Sie die Git-Anmeldeinformationen für CodeCommit (HTTPS, für die meisten Benutzer empfohlen), ein SSH-Schlüsselpaar (SSH) für den Zugriff git-remote-codecommit (empfohlen für Benutzer CodeCommit, die Verbundzugriff verwenden) oder den in der enthaltenen Credential Helper konfigurieren. AWS CLI

- Informationen zu Git-Anmeldeinformationen auf allen unterstützten Betriebssystemen finden Sie unter [Schritt 3: Erstellen Sie Git-Anmeldeinformationen für HTTPS-Verbindungen zu CodeCommit](#).
- Informationen zu SSH unter Linux, macOS oder Unix finden Sie unter [SSH und Linux, macOS oder Unix: Richten Sie die öffentlichen und privaten Schlüssel für Git ein und CodeCommit](#).
- Informationen zu SSH unter Windows finden Sie unter [Schritt 3: Richten Sie die öffentlichen und privaten Schlüssel für Git ein und CodeCommit](#).
- Informationen für git-remote-codecommit finden Sie unter [Einrichtungsschritte für HTTPS-Verbindungen AWS CodeCommit mit git-remote-codecommit](#).
- Informationen zum Credential Helper unter Linux, macOS oder Unix finden Sie unter [Credential Helper einrichten \(Linux, macOS oder Unix\)](#).

- Informationen zum Hilfsprogramm für Anmeldeinformationen unter Windows finden Sie unter [Einrichten des Hilfsprogramms für Anmeldeinformationen \(Windows\)](#).

Schritt 1: Erstellen Sie ein Repository CodeCommit

In diesem Abschnitt verwenden Sie die CodeCommit Konsole, um das CodeCommit Repository zu erstellen, das Sie für den Rest dieses Tutorials verwenden. Informationen zur Verwendung von AWS CLI zum Erstellen des Repositorys finden Sie unter [Erstellen Sie ein Repository \(\)AWS CLI](#).

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie in der Regionsauswahl den Ort aus, AWS-Region an dem Sie das Repository erstellen möchten. Weitere Informationen finden Sie unter [Regionen und Git-Verbindungsendpunkte](#).
3. Wählen Sie auf der Seite Repositorys die Option Repository auswählen.
4. Geben Sie auf der Seite Create repository (Repository erstellen) im Feld Repository name (Repository-Name) einen Namen für das Repository ein.

Note

Bei den Repository-Namen wird nach Groß- und Kleinschreibung unterschieden. Der Name muss AWS-Region für Ihr Amazon Web Services Services-Konto eindeutig sein.

5. (Optional) Geben Sie unter Description (Beschreibung) eine Beschreibung für das Repository ein. Dadurch können Sie und andere Benutzer den Zweck des Repositorys leichter erkennen.

Note

Das Beschreibungsfeld zeigt Markdown in der Konsole an und akzeptiert alle HTML-Zeichen und gültigen Unicode-Zeichen. Wenn Sie als Anwendungsentwickler das GetRepository oder verwenden BatchGetRepositories APIs und das Repository-Beschreibungsfeld in einem Webbrowser anzeigen möchten, finden Sie weitere Informationen in der [CodeCommit API-Referenz](#).

6. (Optional) Wählen Sie Tag hinzufügen, um Ihrem Repository ein oder mehrere Repository-Tags (eine benutzerdefinierte Attributbezeichnung, mit der Sie Ihre AWS Ressourcen organisieren

und verwalten können) hinzuzufügen. Weitere Informationen finden Sie unter [Kennzeichen von Repositorys in AWS CodeCommit](#).

7. (Optional) Erweitern Sie Zusätzliche Konfiguration, um anzugeben, ob Sie den Standardschlüssel Von AWS verwalteter Schlüssel oder Ihren eigenen, vom Kunden verwalteten Schlüssel zum Verschlüsseln und Entschlüsseln von Daten in diesem Repository verwenden möchten. Wenn Sie Ihren eigenen, vom Kunden verwalteten Schlüssel verwenden möchten, müssen Sie sicherstellen, dass er dort verfügbar ist, AWS-Region wo Sie das Repository erstellen, und dass der Schlüssel aktiv ist. Weitere Informationen finden Sie unter [AWS Key Management Service und Verschlüsselung für AWS CodeCommit Repositorien](#).
8. (Optional) Wählen Sie Amazon CodeGuru Reviewer für Java und Python aktivieren, wenn dieses Repository Java- oder Python-Code enthält und Sie möchten, dass CodeGuru Reviewer ihn analysiert. CodeGuru Reviewer verwendet mehrere Modelle für maschinelles Lernen, um Codefehler zu finden und Verbesserungen und Korrekturen in Pull-Requests vorzuschlagen. Weitere Informationen finden Sie im [Amazon CodeGuru Reviewer-Benutzerhandbuch](#).
9. Wählen Sie Create (Erstellen) aus.

Nachdem das Repository erstellt wurde, wird es in der Liste Repositories angezeigt. Wählen Sie in der URL-Spalte das Kopiersymbol aus und wählen Sie dann das Protokoll (HTTPS oder SSH) aus, das Sie zum Herstellen der Verbindung mit CodeCommit verwenden möchten. Kopieren Sie die URL.

Wenn Sie beispielsweise Ihrem Repository einen Namen *MyFirstRepo* gegeben haben und HTTPS verwenden, würde die URL wie folgt aussehen:

```
https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyFirstRepo
```

Sie benötigen diese URL später in [Schritt 2: Migrieren Sie lokale Inhalte in das CodeCommit Repository](#).

Schritt 2: Migrieren Sie lokale Inhalte in das CodeCommit Repository

Da Sie nun ein CodeCommit Repository haben, können Sie ein Verzeichnis auf Ihrem lokalen Computer auswählen, das in ein lokales Git-Repository konvertiert werden soll. Der `git init`-Befehl kann verwendet werden, um entweder bestehende, nicht versionierte Inhalte in ein Git-Repository zu konvertieren oder, wenn Sie noch keine Dateien oder Inhalte haben, um ein neues, leeres Repository zu initialisieren.

1. Ändern Sie im Terminal- oder Befehlszeilenfenster des lokalen Computers Verzeichnisse in das Verzeichnis, das Sie als Quelle für Ihr Repository verwenden möchten.
2. Führen Sie den folgenden Befehl aus, um Git so zu konfigurieren, dass es einen Standardzweig mit dem Namen verwendet **main**:

```
git config --local init.defaultBranch main
```

Du kannst diesen Befehl auch ausführen, um deinen Standard-Branch-Namen **main** für alle neu erstellten Repositories auf festzulegen:

```
git config --global init.defaultBranch main
```

3. Führen Sie den `git init`-Befehl aus, um die Git-Versionskontrolle im Verzeichnis zu initialisieren. Dadurch wird ein Unterverzeichnis `".git"` im Stamm des Verzeichnisses erstellt, das eine Nachverfolgung zur Versionskontrolle ermöglicht. Der Ordner `".git"` enthält außerdem alle für das Repository erforderlichen Metadaten.

```
git init
```

4. Überprüfen Sie den Status des initialisierten Verzeichnisses, indem Sie den folgenden Befehl ausführen:

```
git status
```

Fügen Sie die Dateien hinzu, die Sie der Versionskontrolle hinzufügen möchten. In diesem Tutorial führen Sie den `git add`-Befehl mit dem `.`-Spezifizierer aus, um alle Dateien in diesem Verzeichnis hinzuzufügen. Weitere Optionen finden Sie in der Git-Dokumentation.

```
git add .
```

5. Erstellen Sie einen Commit für die hinzugefügten Dateien mit einer Commit-Nachricht.

```
git commit -m "Initial commit"
```

6. Führen Sie den `git push` Befehl aus und geben Sie die URL und den Namen des CodeCommit Ziel-Repositorys sowie die `--all` Option an. (Dies ist die URL, die Sie in [Schritt 1: Erstellen Sie ein Repository CodeCommit](#) kopiert haben.)

Wenn Sie beispielsweise Ihr Repository benannt haben *MyFirstRepo* und für die Verwendung von HTTPS eingerichtet sind, würden Sie den folgenden Befehl ausführen:

```
git push https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyFirstRepo --all
```

Schritt 3: Dateien anzeigen in CodeCommit

Nachdem Sie den Inhalt Ihres Verzeichnisses per Push übertragen haben, können Sie die CodeCommit Konsole verwenden, um schnell alle Dateien im Repository anzuzeigen.

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie unter Repositories den Namen des Repositorys (z. B. *MyFirstRepository*) aus der Liste aus.
3. Sehen Sie sich die Dateien im Repository für die Branches, den Clone URLs, die Einstellungen und mehr an.

Schritt 4: Teilen Sie das CodeCommit Repository

Wenn Sie ein Repository in erstellen CodeCommit, werden zwei Endpunkte generiert: einer für HTTPS-Verbindungen und einer für SSH-Verbindungen. Beide bieten sichere Verbindungen über ein Netzwerk. Ihre Benutzer können beide Protokolle verwenden. Beide Endpunkte bleiben aktiv, unabhängig davon, welches Protokoll Sie Ihren Benutzern empfehlen. Bevor Sie Ihr Repository mit anderen teilen können, müssen Sie IAM-Richtlinien erstellen, die anderen Benutzern Zugriff auf Ihr Repository gewähren. Stellen Sie Ihren Benutzern diese Zugriffsanweisungen zur Verfügung.

Erstellen einer kundenverwalteten Richtlinie für das Repository

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter <https://console.aws.amazon.com/iam/>
2. Wählen Sie im Navigationsbereich Dashboard die Optionen Policies und dann Create Policy aus.
3. Wählen Sie auf der Seite „Richtlinie erstellen“ die Option Verwaltete Richtlinie importieren aus.
4. Geben Sie auf der Seite Verwaltete Richtlinien importieren im Feld Richtlinien filtern den Text **einAWSCodeCommitPowerUser**. Klicken Sie auf die Schaltfläche neben dem Richtliniennamen und wählen Sie dann Importieren aus.

5. Wählen Sie auf der Seite Create policy (Richtlinie erstellen) die Option JSON aus. Ersetzen Sie den Teil „*“ in der Resource Zeile für CodeCommit Aktionen durch den Amazon-Ressourcennamen (ARN) des CodeCommit Repositorys, wie hier gezeigt:

```
"Resource": [  
  "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo"  
]
```

 Tip

Um den ARN für das CodeCommit Repository zu finden, gehen Sie zur CodeCommit Konsole, wählen Sie den Repository-Namen aus der Liste aus und wählen Sie dann Einstellungen. Weitere Informationen finden Sie unter [Repository-Details anzeigen](#).

Wenn Sie diese Richtlinie auf mehr als ein Repository anwenden möchten, fügen Sie jedes Repository als Ressource hinzu, indem Sie seinen ARN angeben. Fügen Sie zwischen den Ressourcenanweisungen ein Komma ein, wie hier dargestellt:

```
"Resource": [  
  "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo",  
  "arn:aws:codecommit:us-east-2:111111111111:MyOtherDemoRepo"  
]
```

Wenn Sie mit der Bearbeitung fertig sind, wählen Sie „Richtlinie überprüfen“.

6. Geben Sie auf der Seite „Richtlinie überprüfen“ im Feld Name einen neuen Namen für die Richtlinie ein (z. B. *AWSCodeCommitPowerUser-MyDemoRepo*). Geben Sie optional eine Beschreibung für diese Richtlinie ein.
7. Wählen Sie Create Policy (Richtlinie erstellen) aus.

Um den Zugriff auf Ihr Repository zu verwalten, erstellen Sie eine IAM-Gruppe für deren Benutzer, fügen Sie dieser Gruppe IAM-Benutzer hinzu und fügen Sie dann die vom Kunden verwaltete Richtlinie hinzu, die Sie im vorherigen Schritt erstellt haben. Fügen Sie alle anderen für den Zugriff erforderlichen Richtlinien an, z. B. `IAMSelfManageServiceSpecificCredentials` oder `IAMUserSSHKeys`.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter. <https://console.aws.amazon.com/iam/>
2. Wählen Sie im Navigationsbereich Dashboard die Option Groups und dann Create New Group aus.
3. Geben Sie auf der Seite Set Group Name (Name für Gruppe festlegen) im Feld Group Name (Gruppenname) einen Namen für die Gruppe ein (zum Beispiel *MyDemoRepoGroup*) und wählen Sie dann Next Step (Nächster Schritt) aus. Sie sollten den Repository-Namen als Teil des Gruppennamens einfügen.

 Note

Dieser Name muss in einem Amazon Web Services Services-Konto eindeutig sein.

4. Markieren Sie das Kästchen neben der vom Kunden verwalteten Richtlinie, die Sie im vorherigen Abschnitt erstellt haben (z. B. AWSCodeCommitPowerUser-MyDemoRepo).
5. Klicken Sie auf der Seite Review auf Create Group. IAM erstellt diese Gruppe mit den angegebenen Richtlinien, die bereits angehängt sind. Die Gruppe erscheint in der Liste der Gruppen, die mit Ihrem Amazon Web Services Services-Konto verknüpft sind.
6. Wählen Sie Ihre Gruppe in der Liste aus.
7. Wählen Sie auf der Gruppenübersichtsseite die Registerkarte Users und anschließend die Option Add Users to Groups aus. Markieren Sie in der Liste, in der alle mit Ihrem Amazon Web Services Services-Konto verknüpften Benutzer angezeigt werden, die Kästchen neben den Benutzern, denen Sie Zugriff auf das CodeCommit Repository gewähren möchten, und wählen Sie dann Benutzer hinzufügen.

 Tip

Sie können über das Suchfeld schnell nach Benutzern anhand ihres Namens suchen.

8. Wenn Sie Ihre Benutzer hinzugefügt haben, schließen Sie die IAM-Konsole.

Nachdem Sie einen IAM-Benutzer für den Zugriff CodeCommit mit der von Ihnen konfigurierten Policy-Gruppe und den Richtlinien erstellt haben, senden Sie diesem Benutzer die Informationen, die für die Verbindung mit dem Repository erforderlich sind.

1. Öffnen Sie die CodeCommit Konsole unter <https://console.aws.amazon.com/codesuite/codecommit/home>.
2. Wählen Sie in der Regionsauswahl den Ort aus, AWS-Region an dem das Repository erstellt wurde. Repositories sind spezifisch für eine AWS-Region. Weitere Informationen finden Sie unter [Regionen und Git-Verbindungsendpunkte](#).
3. Wählen Sie auf der Seite Repositories (Repositories) das Repository aus, das Sie freigeben möchten.
4. Wählen Sie unter Clone URL (Klon-URL) das Protokoll aus, das die Benutzer verwenden sollen. Dadurch wird die Klon-URL für das Verbindungsprotokoll kopiert.
5. Senden Sie Ihren Benutzern die Klon-URL zusammen mit allen anderen Anweisungen, z. B. zur Installation von AWS CLI, zur Konfiguration eines Profils oder zur Installation von Git. Stellen Sie sicher, dass Sie die Konfigurationsinformationen für das Verbindungsprotokoll (z. B. HTTPS) angeben.

Ein Repository inkrementell migrieren

Erwägen Sie bei der Migration zu AWS CodeCommit, Ihr Repository inkrementell oder abschnittsweise zu pushen, um die Wahrscheinlichkeit zu verringern, dass ein zeitweiliges Netzwerkproblem oder eine verminderte Netzwerkleistung dazu führen, dass der gesamte Push fehlschlägt. Indem Sie inkrementelle Push-Übertragungen mit einem Skript wie dem hier enthaltenen ausführen, können Sie die Migration erneut starten und nur die Commits, die im vorherigen Versuch nicht erfolgreich übermittelt werden konnten, erneut per Push übertragen.

Die Verfahren in diesem Thema veranschaulichen, wie Sie ein Skript erstellen und ausführen, mit dem eine inkrementelle Migration des Repositories durchgeführt wird und nur die inkrementellen Schritte, die zuvor fehlgeschlagen sind, erneut ausgeführt werden, bis die Migration vollständig abgeschlossen ist.

Für diese Anweisungen wird davon ausgegangen, dass Sie bereits die Schritte unter [Einrichtung](#) und [Erstellen eines -Repositorys](#) ausgeführt haben.

Themen

- [Schritt 0: Stellen Sie fest, ob eine inkrementelle Migration erfolgen soll](#)
- [Schritt 1: Installieren Sie die erforderlichen Komponenten und fügen Sie das CodeCommit Repository als Remote-Repository hinzu](#)
- [Schritt 2: Erstellen Sie das Skript, das für die inkrementelle Migration verwendet werden soll](#)

- [Schritt 3: Führen Sie das Skript aus und migrieren Sie schrittweise zu CodeCommit](#)
- [Anhang: Beispielskript incremental-repo-migration.py](#)

Schritt 0: Stellen Sie fest, ob eine inkrementelle Migration erfolgen soll

Bei der Ermittlung der Repository-Gesamtgröße und der Bestimmung, ob eine inkrementelle Migration sinnvoll ist, sind zahlreiche Faktoren zu berücksichtigen. Am offensichtlichsten ist die Gesamtgröße der Artefakte im Repository. Aber auch andere Faktoren wie z. B. der gesamte Repository-Verlauf tragen zur Größe bei. Ein Repository mit jahrelangem Verlauf und Branches kann sehr umfangreich sein, auch wenn die einzelnen Komponenten nicht groß sind. Es gibt viele Strategien, die eine Migration dieser Repositories einfacher und effizienter gestalten können. Sie können beispielsweise einen flachen Klon beim Klonen eines Repositories mit einem umfangreichen Entwicklungsverlauf einsetzen oder die Deltakomprimierung bei großen Binärdateien deaktivieren. Sie können diese Optionen in der Git-Dokumentation prüfen oder Sie können inkrementelle Push-Übertragungen einrichten und konfigurieren, um Ihr Repository mit dem hier enthaltenen Skriptbeispiel `incremental-repo-migration.py` zu migrieren.

Sie sollten die Konfiguration inkrementeller Push-Übertragungen in Betracht ziehen, wenn eine der folgenden Bedingungen zutrifft:

- Das zu migrierende Repository weist einen Verlauf von mehr als fünf Jahren auf.
- Ihre Internetverbindung weist Probleme wie gelegentliche Ausfälle, nicht übertragene Pakete, langsame Geschwindigkeit oder andere Leistungsstörungen auf.
- Die Gesamtgröße des zu migrierenden Repositories übersteigt 2 GB.
- Das Repository enthält große Artefakte oder Binärdateien, die sich nicht gut komprimieren lassen (z. B. große Bilddateien mit mehr als fünf verfolgten Versionen).
- Sie haben bereits versucht, zu migrieren, CodeCommit und die Meldung „Internal Service Error“ erhalten.

Auch wenn keine der oben genannten Bedingungen zutrifft, können Sie sich für eine inkrementelle Push-Übertragung entscheiden.

Schritt 1: Installieren Sie die erforderlichen Komponenten und fügen Sie das CodeCommit Repository als Remote-Repository hinzu

Sie können ein eigenes Skript mit eigenen Voraussetzungen erstellen. Wenn Sie das Beispiel in diesem Thema verwenden können, müssen Sie Folgendes beachten:

- Installieren Sie die Voraussetzungen.
- Klonen Sie das Repository auf Ihren lokalen Computer.
- Fügen Sie das CodeCommit Repository als Remote-Repository für das Repository hinzu, das Sie migrieren möchten.

Für die Ausführung von incremental-repo-migration .py einrichten

1. Installieren Sie Python 2.6 oder eine neuere Version auf Ihrem lokalen Computer. Weitere Informationen und die neuesten Versionen finden Sie auf der [Python-Website](#).
2. Installieren Sie auf demselben Computer, eine Python-Bibliothek GitPython, die für die Interaktion mit Git-Repositorys verwendet wird. Weitere Informationen finden Sie in [der GitPython Dokumentation](#).
3. Verwenden Sie den Befehl `git clone --mirror`, um das Repository, das auf Ihren lokalen Computer migriert werden soll, zu klonen. Verwenden Sie im Terminal (Linux, macOS oder Unix) oder in der Befehlszeile (Windows) den `git clone --mirror` Befehl, um ein lokales Repository für das Repository zu erstellen, einschließlich des Verzeichnisses, in dem Sie das lokale Repository erstellen möchten. Um beispielsweise ein Git-Repository mit der URL in ein Verzeichnis *MyMigrationRepo* mit dem Namen `https://example.com/my-repo/` zu klonen *my-repo*:

```
git clone --mirror https://example.com/my-repo/MyMigrationRepo.git my-repo
```

Die Ausgabe sieht folgendermaßen aus und gibt an, dass das Repository in ein leeres lokales Repository mit der Bezeichnung "my-repo" geklont wurde:

```
Cloning into bare repository 'my-repo'...
remote: Counting objects: 20, done.
remote: Compressing objects: 100% (17/17), done.
remote: Total 20 (delta 5), reused 15 (delta 3)
Unpacking objects: 100% (20/20), done.
Checking connectivity... done.
```

4. Ändern Sie die Verzeichnisse für das Repository, das Sie gerade geklont haben, in das lokale Repository (z. B. *my-repo*). Fügen Sie für dieses Verzeichnis mit dem Befehl `git remote add DefaultRemoteName RemoteRepositoryURL` das CodeCommit -Repository als Remote-Repository für das lokale Repository hinzu.

 Note

Bei der Push-Übertragung von umfangreichen Repositories sollten Sie anstelle von HTTPS die Verwendung von SSH in Betracht ziehen. Wird eine umfassende Änderung, eine große Anzahl an Änderungen oder ein großes Repository per Push übertragen, werden HTTPS-Langzeitverbindungen häufig aufgrund von Problemen mit der Netzwerkverbindung oder den Firewall-Einstellungen vorzeitig beendet. Weitere Hinweise zur Einrichtung von SSH CodeCommit finden Sie unter oder. [Für SSH-Verbindungen unter Linux, macOS oder Unix](#) [Für SSH-Verbindungen unter Windows](#)

Verwenden Sie beispielsweise den folgenden Befehl, um den SSH-Endpunkt für ein CodeCommit Repository hinzuzufügen, das MyDestinationRepo als Remote-Repository für das Remote-Repository benannt ist: `codecommit`

```
git remote add codecommit ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDestinationRepo
```

 Tip

Da es sich um einen Klon handelt, ist der Standardname für das Remote-Repository (`origin`) bereits in Verwendung. Sie müssen daher einen anderen Namen festlegen. Obwohl im Beispiel `codecommit` verwendet wird, können Sie den Namen beliebig auswählen. Überprüfen Sie mit dem Befehl `git remote show` die Liste der Remote-Repositories, die für Ihr lokales Repository eingerichtet sind.

5. Zeigen Sie mit dem Befehl `git remote -v` die Fetch- und Push-Einstellungen für das lokale Repository an und überprüfen Sie, ob die Einstellungen korrekt sind. Zum Beispiel:

```
codecommit  ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDestinationRepo
(fetch)
```

```
codecommit ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDestinationRepo  
(push)
```

 Tip

Falls noch Fetch- und Push-Einträge für ein anderes Remote-Repository angezeigt werden (z. B. Einträge für "origin"), entfernen Sie diese mit dem Befehl `git remote set-url --delete`.

Schritt 2: Erstellen Sie das Skript, das für die inkrementelle Migration verwendet werden soll

Bei diesen Schritten wird davon ausgegangen, dass Sie das Beispiel-Skript `incremental-repo-migration.py` verwenden.

1. Öffnen Sie einen Texteditor und fügen Sie den Inhalt aus dem [Skriptbeispiel](#) in ein leeres Dokument ein.
2. Speichern Sie das Dokument in einem Dokumentverzeichnis (nicht im Arbeitsverzeichnis des lokalen Repositories) unter dem Namen `incremental-repo-migration.py`. Achten Sie darauf, dass das gewählte Verzeichnis für Ihre lokale Umgebung oder Pfadvariablen konfiguriert ist, damit Sie das Python-Skript über ein Befehlszeilen- oder Terminalfenster ausführen können.

Schritt 3: Führen Sie das Skript aus und migrieren Sie schrittweise zu CodeCommit

Nachdem Sie Ihr `incremental-repo-migration.py` Skript erstellt haben, können Sie es verwenden, um ein lokales Repo inkrementell in ein Repository zu migrieren. CodeCommit Standardmäßig überträgt das Skript Stapel von je 1 000 Commits. Zudem nutzt das Skript die Git-Einstellungen des Verzeichnisses, in dem es ausgeführt wird, als Einstellungen für das lokale und das Remote-Repository. Mit den in `incremental-repo-migration.py` enthaltenen Optionen können Sie bei Bedarf andere Einstellungen konfigurieren.

1. Ändern Sie über das Terminal- oder Befehlszeilenfenster die Verzeichnisverweise auf das zu migrierende lokale Repository.
2. Führen Sie aus dem Verzeichnis den folgenden Befehl aus:

```
python incremental-repo-migration.py
```

3. Das Skript wird ausgeführt, der Fortschritt wird im Terminal- oder Befehlszeilenfenster angezeigt. Bei sehr umfangreichen Repositorys ist nur ein langsamer Fortschritt sichtbar. Falls ein einzelner Push-Versuch dreimal fehlschlägt, wird das Skript angehalten. Sie können das Skript dann ab dem fehlgeschlagenen Stapel erneut ausführen. Das Skript kann so lange ausgeführt werden, bis alle Push-Übertragungen erfolgreich verlaufen sind und die Migration abgeschlossen ist.

Tip

Sie können `incremental-repo-migration.py` aus jedem Verzeichnis ausführen, sofern Sie die zu verwendenden Einstellungen für das lokale und das Remote-Repository mithilfe der Optionen `-l` und `-r` festgelegt haben. Um beispielsweise das Skript aus einem beliebigen Verzeichnis zu verwenden, um ein lokales Repo, das sich unter `my-repo /tmp/` befindet, zu einem Remote-Repository mit dem Spitznamen zu migrieren: `codecommit`

```
python incremental-repo-migration.py -l "/tmp/my-repo" -r "codecommit"
```

Sie können auch die Option `-b` verwenden, um die Standard-Batch-Größe für den inkrementellen Push zu ändern. Wenn Sie beispielsweise regelmäßig ein Repository mit sehr umfangreichen Binärdateien, an denen häufig Änderungen vorgenommen werden, per Push über eine langsame Netzwerkverbindung übertragen müssen, können Sie die Stapelgröße mithilfe der Option `-b` von 1 000 auf 500 verringern. Zum Beispiel:

```
python incremental-repo-migration.py -b 500
```

Auf diese Weise wird das lokale Repository inkrementell in Stapeln zu je 500 Commits per Push übertragen. Falls Sie die Stapelgröße bei der Repository-Migration erneut ändern möchten (z. B. um die Stapelgröße nach einem fehlgeschlagenen Versuch zu verringern), müssen Sie die Stapel-Tags erst mit der Option `-c` entfernen, bevor Sie die Stapelgröße mit `-b` ändern:

```
python incremental-repo-migration.py -c  
python incremental-repo-migration.py -b 250
```

⚠ Important

Verwenden Sie nicht die Option `-c`, wenn Sie das Skript nach einem Fehlschlag erneut ausführen möchten. Die Option `-c` löscht die Tags, mit denen die Commits in Stapeln zusammengefasst werden. Nutzen Sie daher die Option `-c` nur, wenn Sie die Stapelgröße ändern und neu starten möchten oder wenn Sie entschieden haben, das Skript nicht weiter zu verwenden.

Anhang: Beispielskript **incremental-repo-migration.py**

Wir haben für Sie das Python-Skriptbeispiel `incremental-repo-migration.py` für eine einfache inkrementelle Push-Übertragung von Repositorys entwickelt. Dieses Skript ist ein Open Source-Codebeispiel und wird wie gesehen bereitgestellt.

```
# Copyright 2015 Amazon.com, Inc. or its affiliates. All Rights Reserved. Licensed
# under the Amazon Software License (the "License").
# You may not use this file except in compliance with the License. A copy of the
# License is located at
#   http://aws.amazon.com/asl/
# This file is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY
# KIND, express or implied. See the License for
# the specific language governing permissions and limitations under the License.

#!/usr/bin/env python

import os
import sys
from optparse import OptionParser
from git import Repo, TagReference, RemoteProgress, GitCommandError

class PushProgressPrinter(RemoteProgress):
    def update(self, op_code, cur_count, max_count=None, message=""):
        op_id = op_code & self.OP_MASK
        stage_id = op_code & self.STAGE_MASK
        if op_id == self.WRITING and stage_id == self.BEGIN:
            print("\tObjects: %d" % max_count)

class RepositoryMigration:
```



```
MAX_COMMITS_TOLERANCE_PERCENT = 0.05
PUSH_RETRY_LIMIT = 3
MIGRATION_TAG_PREFIX = "codecommit_migration_"

def migrate_repository_in_parts(
    self, repo_dir, remote_name, commit_batch_size, clean
):
    self.next_tag_number = 0
    self.migration_tags = []
    self.walked_commits = set()
    self.local_repo = Repo(repo_dir)
    self.remote_name = remote_name
    self.max_commits_per_push = commit_batch_size
    self.max_commits_tolerance = (
        self.max_commits_per_push * self.MAX_COMMITS_TOLERANCE_PERCENT
    )

    try:
        self.remote_repo = self.local_repo.remote(remote_name)
        self.get_remote_migration_tags()
    except (ValueError, GitCommandError):
        print(
            "Could not contact the remote repository. The most common reasons for
this error are that the name of the remote repository is incorrect, or that you do not
have permissions to interact with that remote repository."
        )
        sys.exit(1)

    if clean:
        self.clean_up(clean_up_remote=True)
        return

    self.clean_up()

    print("Analyzing repository")
    head_commit = self.local_repo.head.commit
    sys.setrecursionlimit(max(sys.getrecursionlimit(), head_commit.count()))

    # tag commits on default branch
    leftover_commits = self.migrate_commit(head_commit)
    self.tag_commits([commit for (commit, commit_count) in leftover_commits])

    # tag commits on each branch
    for branch in self.local_repo.heads:
```

```
        leftover_commits = self.migrate_commit(branch.commit)
        self.tag_commits([commit for (commit, commit_count) in leftover_commits])

    # push the tags
    self.push_migration_tags()

    # push all branch references
    for branch in self.local_repo.heads:
        print("Pushing branch %s" % branch.name)
        self.do_push_with_retries(ref=branch.name)

    # push all tags
    print("Pushing tags")
    self.do_push_with_retries(push_tags=True)

    self.get_remote_migration_tags()
    self.clean_up(clean_up_remote=True)

    print("Migration to CodeCommit was successful")

def migrate_commit(self, commit):
    if commit in self.walked_commits:
        return []

    pending_ancestor_pushes = []
    commit_count = 1

    if len(commit.parents) > 1:
        # This is a merge commit
        # Ensure that all parents are pushed first
        for parent_commit in commit.parents:
            pending_ancestor_pushes.extend(self.migrate_commit(parent_commit))
    elif len(commit.parents) == 1:
        # Split linear history into individual pushes
        next_ancestor, commits_to_next_ancestor = self.find_next_ancestor_for_push(
            commit.parents[0]
        )
        commit_count += commits_to_next_ancestor
        pending_ancestor_pushes.extend(self.migrate_commit(next_ancestor))

    self.walked_commits.add(commit)

    return self.stage_push(commit, commit_count, pending_ancestor_pushes)
```

```

def find_next_ancestor_for_push(self, commit):
    commit_count = 0

    # Traverse linear history until we reach our commit limit, a merge commit, or
    an initial commit
    while (
        len(commit.parents) == 1
        and commit_count < self.max_commits_per_push
        and commit not in self.walked_commits
    ):
        commit_count += 1
        self.walked_commits.add(commit)
        commit = commit.parents[0]

    return commit, commit_count

def stage_push(self, commit, commit_count, pending_ancestor_pushes):
    # Determine whether we can roll up pending ancestor pushes into this push
    combined_commit_count = commit_count + sum(
        ancestor_commit_count
        for (ancestor, ancestor_commit_count) in pending_ancestor_pushes
    )

    if combined_commit_count < self.max_commits_per_push:
        # don't push anything, roll up all pending ancestor pushes into this
        pending push
        return [(commit, combined_commit_count)]

    if combined_commit_count <= (
        self.max_commits_per_push + self.max_commits_tolerance
    ):
        # roll up everything into this commit and push
        self.tag_commits([commit])
        return []

    if commit_count >= self.max_commits_per_push:
        # need to push each pending ancestor and this commit
        self.tag_commits(
            [
                ancestor
                for (ancestor, ancestor_commit_count) in pending_ancestor_pushes
            ]
        )
        self.tag_commits([commit])

```

```

        return []

    # push each pending ancestor, but roll up this commit
    self.tag_commits(
        [ancestor for (ancestor, ancestor_commit_count) in pending_ancestor_pushes]
    )
    return [(commit, commit_count)]

def tag_commits(self, commits):
    for commit in commits:
        self.next_tag_number += 1
        tag_name = self.MIGRATION_TAG_PREFIX + str(self.next_tag_number)

        if tag_name not in self.remote_migration_tags:
            tag = self.local_repo.create_tag(tag_name, ref=commit)
            self.migration_tags.append(tag)
        elif self.remote_migration_tags[tag_name] != str(commit):
            print(
                "Migration tags on the remote do not match the local tags. Most
likely your batch size has changed since the last time you ran this script. Please run
this script with the --clean option, and try again."
            )
            sys.exit(1)

def push_migration_tags(self):
    print("Will attempt to push %d tags" % len(self.migration_tags))
    self.migration_tags.sort(
        key=lambda tag: int(tag.name.replace(self.MIGRATION_TAG_PREFIX, ""))
    )
    for tag in self.migration_tags:
        print(
            "Pushing tag %s (out of %d tags), commit %s"
            % (tag.name, self.next_tag_number, str(tag.commit))
        )
        self.do_push_with_retries(ref=tag.name)

def do_push_with_retries(self, ref=None, push_tags=False):
    for i in range(0, self.PUSH_RETRY_LIMIT):
        if i == 0:
            progress_printer = PushProgressPrinter()
        else:
            progress_printer = None

        try:

```

```
        if push_tags:
            infos = self.remote_repo.push(tags=True, progress=progress_printer)
        elif ref is not None:
            infos = self.remote_repo.push(
                refspec=ref, progress=progress_printer
            )
        else:
            infos = self.remote_repo.push(progress=progress_printer)

        success = True
        if len(infos) == 0:
            success = False
        else:
            for info in infos:
                if (
                    info.flags & info.UP_TO_DATE
                    or info.flags & info.NEW_TAG
                    or info.flags & info.NEW_HEAD
                ):
                    continue
                success = False
                print(info.summary)

        if success:
            return
    except GitCommandError as err:
        print(err)

    if push_tags:
        print("Pushing all tags failed after %d attempts" %
              (self.PUSH_RETRY_LIMIT))
    elif ref is not None:
        print("Pushing %s failed after %d attempts" % (ref, self.PUSH_RETRY_LIMIT))
        print(
            "For more information about the cause of this error, run the following"
            "command from the local repo: 'git push %s %s'"
            % (self.remote_name, ref)
        )
    else:
        print(
            "Pushing all branches failed after %d attempts"
            % (self.PUSH_RETRY_LIMIT)
        )
    sys.exit(1)
```

```
def get_remote_migration_tags(self):
    remote_tags_output = self.local_repo.git.ls_remote(
        self.remote_name, tags=True
    ).split("\n")
    self.remote_migration_tags = dict(
        (tag.split()[1].replace("refs/tags/", ""), tag.split()[0])
        for tag in remote_tags_output
        if self.MIGRATION_TAG_PREFIX in tag
    )

def clean_up(self, clean_up_remote=False):
    tags = [
        tag
        for tag in self.local_repo.tags
        if tag.name.startswith(self.MIGRATION_TAG_PREFIX)
    ]

    # delete the local tags
    TagReference.delete(self.local_repo, *tags)

    # delete the remote tags
    if clean_up_remote:
        tags_to_delete = [":" + tag_name for tag_name in
self.remote_migration_tags]
        self.remote_repo.push(refspec=tags_to_delete)

parser = OptionParser()
parser.add_option(
    "-l",
    "--local",
    action="store",
    dest="localrepo",
    default=os.getcwd(),
    help="The path to the local repo. If this option is not specified, the script will
attempt to use current directory by default. If it is not a local git repo, the script
will fail.",
)
parser.add_option(
    "-r",
    "--remote",
    action="store",
    dest="remoterepo",
```

```
    default="codecommit",
    help="The name of the remote repository to be used as the push or migration
destination. The remote must already be set in the local repo ('git remote add ...').
If this option is not specified, the script will use 'codecommit' by default.",
)
parser.add_option(
    "-b",
    "--batch",
    action="store",
    dest="batchsize",
    default="1000",
    help="Specifies the commit batch size for pushes. If not explicitly set, the
default is 1,000 commits.",
)
parser.add_option(
    "-c",
    "--clean",
    action="store_true",
    dest="clean",
    default=False,
    help="Remove the temporary tags created by migration from both the local repo
and the remote repository. This option will not do any migration work, just cleanup.
Cleanup is done automatically at the end of a successful migration, but not after a
failure so that when you re-run the script, the tags from the prior run can be used to
identify commit batches that were not pushed successfully.",
)

(options, args) = parser.parse_args()

migration = RepositoryMigration()
migration.migrate_repository_in_parts(
    options.localrepo, options.remoterepo, int(options.batchsize), options.clean
)
```

Sicherheit in AWS CodeCommit

Cloud-Sicherheit AWS hat höchste Priorität. Als AWS Kunde profitieren Sie von einem Rechenzentrum und einer Netzwerkarchitektur, die darauf ausgelegt sind, die Anforderungen der sicherheitssensibelsten Unternehmen zu erfüllen.

Sicherheit ist eine gemeinsame Verantwortung von Ihnen AWS und Ihnen. Das [Modell der geteilten Verantwortung](#) beschreibt dies als Sicherheit der Cloud selbst und Sicherheit in der Cloud:

- Sicherheit der Cloud — AWS ist verantwortlich für den Schutz der Infrastruktur, die AWS Dienste in der AWS Cloud ausführt. AWS bietet Ihnen auch Dienste, die Sie sicher nutzen können. Externe Prüfer testen und verifizieren regelmäßig die Wirksamkeit unserer Sicherheitsmaßnahmen im Rahmen der [AWS](#). Weitere Informationen zu den Compliance-Programmen, die für gelten AWS CodeCommit, finden Sie unter [AWS Services im Umfang nach Compliance-Programmen AWS](#).
- Sicherheit in der Cloud — Ihre Verantwortung richtet sich nach dem AWS Dienst, den Sie nutzen. Sie sind auch für andere Faktoren verantwortlich, etwa für die Vertraulichkeit Ihrer Daten, für die Anforderungen Ihres Unternehmens und für die geltenden Gesetze und Vorschriften.

Diese Dokumentation hilft Ihnen zu verstehen, wie Sie das Modell der gemeinsamen Verantwortung bei der Nutzung anwenden können CodeCommit. In den folgenden Themen erfahren Sie, wie Sie die Konfiguration vornehmen CodeCommit, um Ihre Sicherheits- und Compliance-Ziele zu erreichen. Sie erfahren auch, wie Sie andere AWS Dienste nutzen können, die Sie bei der Überwachung und Sicherung Ihrer CodeCommit Ressourcen unterstützen.

Themen

- [Datenschutz in AWS CodeCommit](#)
- [Identity and Access Management für AWS CodeCommit](#)
- [Resilienz in AWS CodeCommit](#)
- [Infrastruktursicherheit in AWS CodeCommit](#)

Datenschutz in AWS CodeCommit

Als verwalteter Service ist es durch AWS globale Netzwerksicherheit geschützt. Informationen zu AWS Sicherheitsdiensten und zum AWS Schutz der Infrastruktur finden Sie unter [AWS Cloud-Sicherheit](#). Informationen zum Entwerfen Ihrer AWS Umgebung unter Verwendung der bewährten

Methoden für die Infrastruktursicherheit finden Sie unter [Infrastructure Protection](#) in Security Pillar AWS Well-Architected Framework.

Sie verwenden AWS veröffentlichte API-Aufrufe für den Zugriff über das Netzwerk. Kunden müssen Folgendes unterstützen:

- Transport Layer Security (TLS). Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Verschlüsselungs-Suiten mit Perfect Forward Secrecy (PFS) wie DHE (Ephemeral Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Die meisten modernen Systeme wie Java 7 und höher unterstützen diese Modi.

Außerdem müssen Anforderungen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signiert sein, der einem IAM-Prinzipal zugeordnet ist. Alternativ können Sie mit [AWS Security Token Service](#) (AWS STS) temporäre Sicherheitsanmeldeinformationen erstellen, um die Anforderungen zu signieren.

CodeCommit Repositorys werden im Ruhezustand automatisch verschlüsselt. Es ist keine Kundenaktion erforderlich. CodeCommit verschlüsselt Repository-Daten auch während der Übertragung. Sie können entweder das HTTPS-Protokoll, das SSH-Protokoll oder beide mit CodeCommit Repositorys verwenden. Weitere Informationen finden Sie unter [Einrichtung für AWS CodeCommit](#). Sie können auch den [kontoübergreifenden Zugriff auf](#) Repositorys konfigurieren.

Themen

- [AWS Key Management Service und Verschlüsselung für AWS CodeCommit Repositoryn](#)
- [Verbindung zu AWS CodeCommit Repositorys mit rotierenden Anmeldeinformationen herstellen](#)

AWS Key Management Service und Verschlüsselung für AWS CodeCommit Repositoryn

Daten in CodeCommit Repositorys werden bei der Übertragung und im Speicher verschlüsselt. Wenn Daten in ein CodeCommit Repository übertragen werden (z. B. durch einen Aufrufgit push), werden die empfangenen Daten so CodeCommit verschlüsselt, wie sie im Repository gespeichert sind. Wenn Daten aus einem CodeCommit Repository abgerufen werden (z. B. durch einen Aufrufgit pull), werden die Daten CodeCommit entschlüsselt und anschließend an den Aufrufer gesendet. Dabei wird davon ausgegangen, dass der mit der Push- oder Pull-Anfrage verknüpfte IAM-Benutzer von authentifiziert

wurde. AWS Gesendete oder empfangene Daten werden über die HTTPS- oder SSH-verschlüsselten Netzwerkprotokolle übertragen.

Sie können entweder einen Von AWS verwalteter Schlüssel oder einen vom Kunden verwalteten Schlüssel zum Verschlüsseln und Entschlüsseln der Daten in Ihrem Repository verwenden. Weitere Informationen zu den Unterschieden zwischen vom Kunden verwalteten Schlüsseln und finden Sie unter Von AWS verwaltete Schlüssel Vom [Kunden verwaltete Schlüssel](#) und. Von AWS verwaltete Schlüssel Wenn Sie keinen vom Kunden verwalteten Schlüssel angeben, verwenden CodeCommit Sie einen Von AWS verwalteter Schlüssel zum Verschlüsseln und Entschlüsseln der Daten in Ihrem Repository. Dieser Von AWS verwalteter Schlüssel wird automatisch für Sie in Ihrem erstellt. AWS-Konto Wenn Sie zum ersten Mal ein CodeCommit Repository in einem neuen Konto AWS-Region in Ihrem Amazon Web Services Services-Konto erstellen und keinen vom Kunden verwalteten Schlüssel angeben, CodeCommit wird in demselben AWS-Region in Von AWS verwalteter Schlüssel () ein AWS Key Management Service (der aws/codecommit Schlüssel AWS KMS) erstellt. Dieser aws/codecommit Schlüssel wird nur von verwendet CodeCommit. Es ist in Ihrem Amazon Web Services Services-Konto gespeichert. Je nachdem, was Sie angeben, wird CodeCommit entweder der vom Kunden verwaltete Schlüssel oder der Von AWS verwalteter Schlüssel zum Verschlüsseln und Entschlüsseln der Daten im Repository verwendet.

Important

CodeCommit führt die folgenden AWS KMS Aktionen mit dem AWS KMS Schlüssel aus, der zum Verschlüsseln und Entschlüsseln von Daten in einem Repository verwendet wird. Wenn Sie einen verwenden Von AWS verwalteter Schlüssel, benötigt ein Benutzer keine ausdrücklichen Berechtigungen für diese Aktionen, aber dem Benutzer dürfen keine Richtlinien zugeordnet sein, die diese Aktionen für den aws/codecommit Schlüssel verweigern. Wenn Sie einen vom Kunden verwalteten Schlüssel verwenden, für den eine AWS-Konto ID als Richtlinienprinzipal festgelegt ist, müssen diese Berechtigungen explizit auf festgelegt werdenallow. Insbesondere wenn Sie Ihr erstes Repository erstellen und Schlüssel für Ihr Repository aktualisieren, darf keine der folgenden Berechtigungen aktiviert sein, wenn Sie einen verwenden Von AWS verwalteter Schlüssel, und müssen auf gesetzt sein, deny wenn Sie einen vom Kunden verwalteten Schlüssel mit einem Policy-Prinzipal verwenden: allow

- "kms:Encrypt"
- "kms:Decrypt"

- "kms:ReEncrypt"(Je nach Kontext kann dies Folgendes erfordernkms:ReEncryptFrom:

kms:ReEncryptTo, oder kms:ReEncrypt* nicht auf Verweigern gesetzt)
- "kms:GenerateDataKey"
- "kms:GenerateDataKeyWithoutPlaintext"
- "kms:DescribeKey"

Wenn Sie Ihren eigenen, vom Kunden verwalteten Schlüssel verwenden möchten, muss der Schlüssel dort verfügbar sein, AWS-Region wo sich das Repository befindet. CodeCommit unterstützt die Verwendung von kundenverwalteten Schlüsseln sowohl für einzelne als auch für mehrere Regionen. Es werden zwar alle wichtigen Materialherkunftstypen unterstützt, wir empfehlen jedoch, die Standardoption KMS zu verwenden. Bei Kunden, die die Option Externer Schlüsselspeicher verwenden, kann es zu Verzögerungen bei ihrem Shop-Anbieter kommen. Darüber hinaus. CodeCommit hat die folgenden Anforderungen für vom Kunden verwaltete Schlüssel:

- CodeCommit unterstützt nur die Verwendung symmetrischer Schlüssel.
- Der Schlüsselverwendungstyp muss auf Verschlüsseln und Entschlüsseln gesetzt sein.

Weitere Informationen zur Erstellung von kundenverwalteten Schlüsseln finden Sie unter [Konzepte](#) und Schlüssel [erstellen](#).

Gehen Sie wie folgt vor, um Informationen über die Datei zu erhalten CodeCommit, die von Von AWS verwalteter Schlüssel generiert wurde:

1. Melden Sie sich bei der AWS Key Management Service (AWS KMS) -Konsole an AWS Management Console und öffnen Sie sie unter <https://console.aws.amazon.com/kms>.
2. Um das zu ändern AWS-Region, verwenden Sie die Regionsauswahl in der oberen rechten Ecke der Seite.
3. Wählen Sie im Navigationsbereich des Dienstes. Von AWS verwaltete Schlüssel Stellen Sie sicher, dass Sie bei dem Ort angemeldet sind, in AWS-Region dem Sie die Schlüssel überprüfen möchten.
4. Wählen Sie in der Liste der Verschlüsselungsschlüssel den Von AWS verwalteter Schlüssel mit dem Alias aws/codecommit aus. Grundlegende Informationen über die AWS-eigener Schlüssel werden angezeigt.

Sie können dies nicht ändern oder löschen Von AWS verwalteter Schlüssel.

Wie Verschlüsselungsalgorithmen zur Verschlüsselung von Repository-Daten verwendet werden

CodeCommit verwendet zwei verschiedene Ansätze zur Verschlüsselung von Daten. Einzelne Git-Objekte unter 6 MB werden mit AES-GCM-256 verschlüsselt, was eine Validierung der Datenintegrität ermöglicht. Objekte zwischen 6 MB und maximal 2 GB für ein einzelnes Blob werden mit AES-CBC-256 verschlüsselt. CodeCommit validiert immer den Verschlüsselungskontext.

Verschlüsselungskontext

Jeder Dienst, in AWS KMS den integriert ist, spezifiziert einen Verschlüsselungskontext für die Verschlüsselungs- und Entschlüsselungsvorgänge. Der Verschlüsselungskontext enthält zusätzliche authentifizierte Informationen, anhand derer AWS KMS die Datenintegrität überprüft. Wenn dieser für den Verschlüsselungsvorgang angegeben ist, muss er auch für den Entschlüsselungsvorgang angegeben werden. Andernfalls schlägt die Entschlüsselung fehl. CodeCommit verwendet die CodeCommit Repository-ID für den Verschlüsselungskontext. Sie können den `get-repository` Befehl oder die CodeCommit Konsole verwenden, um die Repository-ID zu ermitteln. Suchen Sie in den AWS CloudTrail Protokollen nach der CodeCommit Repository-ID, um zu erfahren, welche Verschlüsselungsvorgänge mit welchem Schlüssel durchgeführt wurden AWS KMS , um Daten im Repository zu verschlüsseln oder zu entschlüsseln. CodeCommit

Weitere Informationen AWS KMS dazu finden Sie im [AWS Key Management Service Entwicklerhandbuch](#).

Verbindung zu AWS CodeCommit Repositorys mit rotierenden Anmeldeinformationen herstellen

Sie können Benutzern Zugriff auf Ihre AWS CodeCommit Repositorys gewähren, ohne IAM-Benutzer für sie zu konfigurieren oder einen Zugriffsschlüssel und einen geheimen Schlüssel zu verwenden. Um einer föderierten Identität Berechtigungen zuzuweisen, erstellen Sie eine Rolle und definieren Berechtigungen für die Rolle. Wird eine Verbundidentität authentifiziert, so wird die Identität der Rolle zugeordnet und erhält die von der Rolle definierten Berechtigungen. Informationen zu Rollen für den Verbund finden Sie unter [Erstellen von Rollen für externe Identitätsanbieter \(Verbund\)](#) im IAM-Benutzerhandbuch. Wenn Sie IAM Identity Center verwenden, konfigurieren Sie einen Berechtigungssatz. Wenn Sie steuern möchten, worauf Ihre Identitäten nach der Authentifizierung zugreifen können, korreliert IAM Identity Center den Berechtigungssatz mit einer Rolle in IAM.

Informationen zu Berechtigungssätzen finden Sie unter [Berechtigungssätze](#) im AWS IAM Identity Center -Benutzerhandbuch. Sie können auch den rollenbasierten Zugriff für IAM-Benutzer konfigurieren, um auf CodeCommit Repositorys in separaten Amazon Web Services Services-Konten zuzugreifen (eine Technik, die als kontoübergreifender Zugriff bezeichnet wird). Eine schrittweise Anleitung zur Konfiguration des kontoübergreifenden Zugriffs auf ein Repository finden Sie unter [Konfigurieren des kontoübergreifenden Zugriffs auf ein AWS CodeCommit Repository mithilfe von Rollen](#).

Sie können den Zugriff für Benutzer konfigurieren, die sich über folgende Methoden authentifizieren möchten oder müssen:

- Security Assertion Markup Language (SAML)
- Multi-Faktor-Authentifizierung (MFA)
- Verbund
- Login with Amazon
- Amazon Cognito
- Facebook
- Google
- OpenID Connect-kompatibler (OIDC) Identitätsanbieter

Note

Die folgenden Informationen beziehen sich nur auf die Verwendung von `git-remote-codecommit` oder des AWS CLI Credential Helper zur Verbindung mit Repositorys. CodeCommit Da der empfohlene Ansatz für den temporären oder föderierten Zugriff auf die Einrichtung CodeCommit ist `git-remote-codecommit`, finden Sie in diesem Thema Beispiele für die Verwendung dieses Dienstprogramms. Weitere Informationen finden Sie unter [Einrichtungsschritte für HTTPS-Verbindungen AWS CodeCommit mit `git-remote-codecommit`](#). Sie können SSH- oder Git-Anmeldeinformationen und HTTPS nicht verwenden, um eine Verbindung zu CodeCommit Repositorys mit rotierenden oder temporären Zugangsdaten herzustellen.

Sie müssen die folgenden Schritte nicht ausführen, wenn alle der folgenden Bedingungen zutreffen:

- Sie sind bei einer EC2 Amazon-Instance angemeldet.

- Sie verwenden Git und HTTPS mit dem AWS CLI Credential Helper, um eine Verbindung von der EC2 Amazon-Instance zu CodeCommit Repositorys herzustellen.
- Die EC2 Amazon-Instance hat ein angehängtes IAM-Instance-Profil, das die unter [Für HTTPS-Verbindungen unter Linux, macOS oder Unix mit dem AWS CLI Credential Helper](#) oder [Für HTTPS-Verbindungen unter Windows mit dem Credential Helper AWS CLI](#) beschriebenen Zugriffsberechtigungen enthält.
- Sie haben den Git Credential Helper auf der EC2 Amazon-Instance installiert und konfiguriert, wie unter [Für HTTPS-Verbindungen unter Linux, macOS oder Unix mit dem AWS CLI Credential Helper](#) oder [Für HTTPS-Verbindungen unter Windows mit dem Credential Helper AWS CLI](#) beschrieben.

EC2 Amazon-Instances, die die oben genannten Anforderungen erfüllen, sind bereits so eingerichtet, dass sie in Ihrem Namen temporäre Zugangsdaten CodeCommit an sie weiterleiten.

Note

Sie können git-remote-codecommit auf EC2 Amazon-Instances konfigurieren und verwenden.

Gehen Sie wie folgt vor, um Benutzern temporären Zugriff auf Ihre CodeCommit Repositorys zu gewähren.

Schritt 1: Erfüllen der Voraussetzungen

Schließe die Einrichtungsschritte ab, um einem Benutzer mit wechselnden Zugangsdaten Zugriff auf deine CodeCommit Repositorys zu gewähren:

- Informationen zum kontoübergreifenden Zugriff finden Sie unter [Exemplarische Vorgehensweise: Delegieren des Zugriffs über Amazon Web Services Services-Konten mithilfe von IAM-Rollen](#) und [Konfigurieren des kontoübergreifenden Zugriffs auf ein AWS CodeCommit Repository mithilfe von Rollen](#)
- [Informationen zu SAML und Verbund finden Sie unter Verwenden des Authentifizierungssystems Ihrer Organisation, um Zugriff auf AWS Ressourcen zu gewähren und Über den SAML 2.0-basierten Verbund. AWS STS](#)
- Informationen zu MFA finden Sie unter [Verwenden von Multi-Factor Authentication \(MFA\) -Geräten mit AWS](#) und [Erstellen temporärer Sicherheitsanmeldedaten, um den Zugriff für IAM-Benutzer zu ermöglichen](#).

- Informationen zur Login with Amazon, Amazon Cognito, Facebook, Google oder einem anderen OIDC-kompatiblen Identitätsanbieter finden Sie unter [Über AWS STS](#) Web Identity Federation.

Verwenden Sie die Informationen in [Authentifizierung und Zugriffskontrolle für AWS CodeCommit](#), um die CodeCommit Berechtigungen anzugeben, die Sie dem Benutzer gewähren möchten.

Schritt 2: Holen Sie sich den Rollennamen oder die Zugangsdaten

Wenn Sie möchten, dass Ihre Benutzer auf Repositorys zugreifen, indem sie eine Rolle übernehmen, geben Sie Ihren Benutzern den Amazon-Ressourcennamen (ARN) dieser Rolle an. Andernfalls kann der Benutzer je nach Einrichtung des Zugriffs rotierende Anmeldeinformationen auf eine der folgenden Arten abrufen:

- Rufen Sie für kontoübergreifenden Zugriff den Befehl AWS CLI [assume-role](#) auf oder rufen Sie die API auf. AWS STS [AssumeRole](#)
- [Rufen Sie für SAML den AWS CLI assume-role-with-saml Befehl oder die SAML-API auf. AWS STS AssumeRoleWith](#)
- Rufen Sie für den Verbund die [get-federation-token](#) Befehle AWS CLI [assume-role](#) oder oder or auf. AWS STS [AssumeRoleGetFederationToken](#) APIs
- Rufen Sie für MFA den AWS CLI [get-session-token](#) Befehl oder die AWS STS [GetSessionToken](#) API auf.
- Um sich mit Amazon, Amazon Cognito, Facebook, Google oder einem anderen OIDC-kompatiblen Identitätsanbieter anzumelden, rufen Sie den Befehl AWS CLI [assume-role-with-web-identity](#) oder [die API](#) auf. AWS STS [AssumeRoleWithWebIdentity](#)

Schritt 3: Installieren und konfigurieren Sie den git-remote-codecommit AWS CLI

Sie müssen Ihren lokalen Computer so konfigurieren, dass die Anmeldeinformationen verwendet werden, indem Sie [git-remote-codecommit](#) installieren und ein Profil in der AWS CLI konfigurieren.

1. Folgen Sie den Anweisungen in [Einrichtung](#), um die AWS CLI einzurichten. Verwenden Sie den aws configure-Befehl, um ein oder mehrere Profile zu konfigurieren. Erwägen Sie, ein benanntes Profil zu erstellen, das Sie verwenden können, wenn Sie mithilfe rotierender Anmeldeinformationen eine Verbindung zu CodeCommit Repositorys herstellen.
2. Sie können die Anmeldeinformationen auf eine der folgenden Arten AWS CLI dem benannten Profil des Benutzers zuordnen.

- Wenn Sie eine Rolle für den Zugriff annehmen CodeCommit, konfigurieren Sie ein benanntes Profil mit den Informationen, die für die Übernahme dieser Rolle erforderlich sind. Wenn Sie beispielsweise eine Rolle annehmen möchten, die *CodeCommitAccess* im Amazon Web Services Services-Konto 1111111111 benannt ist, können Sie ein Standardprofil für die Arbeit mit anderen AWS Ressourcen und ein benanntes Profil für die Übernahme dieser Rolle konfigurieren. Die folgenden Befehle erstellen ein benanntes Profil mit dem Namen, das eine Rolle mit dem Namen *CodeAccess* annimmt. *CodeCommitAccess* Der Benutzername *Maria_Garcia* ist der Sitzung zugeordnet, und das Standardprofil wird als Quelle für die AWS Anmeldeinformationen festgelegt:

```
aws configure set role_arn arn:aws:iam::111111111111:role/CodeCommitAccess --  
profile CodeAccess  
aws configure set source_profile default --profile CodeAccess  
aws configure set role_session_name "Maria_Garcia" --profile CodeAccess
```

Wenn Sie die Änderungen überprüfen möchten, können Sie die ~/.aws/config-Datei (für Linux) oder die %UserProfile%.aws\config-Datei (für Windows) manuell anzeigen oder bearbeiten und die Informationen unter dem benannten Profil überprüfen. Zum Beispiel ähnelt Ihre Datei möglicherweise dem Folgenden:

```
[default]  
region = us-east-1  
output = json  
  
[profile CodeAccess]  
source_profile = default  
role_session_name = Maria_Garcia  
role_arn = arn:aws:iam::111111111111:role/CodeCommitAccess
```

Nachdem Sie Ihr benanntes Profil konfiguriert haben, können Sie CodeCommit -Repositorys mit dem git-remote-codecommit-Dienstprogramm mithilfe des benannten Profils klonen. Beispielsweise löschen Sie das Repository namens *MyDemoRepo* wie folgt:

```
git clone codecommit://CodeAccess@MyDemoRepo
```

- Wenn Sie Web Identity Federation und OpenID Connect (OIDC) verwenden, konfigurieren Sie ein benanntes Profil, das in Ihrem Namen den AssumeRoleWithWebIdentity API-Aufruf AWS Security Token Service (AWS STS) durchführt, um temporäre Anmeldeinformationen

zu aktualisieren. Verwenden Sie den `aws configure set` Befehl oder bearbeiten Sie die `~/.aws/credentials` Datei (für Linux) oder die `%UserProfile%.aws\credentials` Datei (für Windows) manuell, um ein AWS CLI benanntes Profil mit den erforderlichen Einstellungswerten hinzuzufügen. Um beispielsweise ein Profil zu erstellen, das die *CodeCommitAccess* Rolle übernimmt und eine Web-Identity-Token-Datei `~/my-credentials/my-token-file` verwendet:

```
[CodeCommitWebIdentity]
role_arn = arn:aws:iam::111111111111:role/CodeCommitAccess
web_identity_token_file=~/my-credentials/my-token-file
role_session_name = Maria_Garcia
```

Weitere Informationen finden Sie unter [Konfiguration AWS Command Line Interface](#) und [Verwendung einer IAM-Rolle AWS CLI im AWS Command Line Interface](#) Benutzerhandbuch.

Schritt 4: Greifen Sie auf die Repositorys zu CodeCommit

Angenommen, Ihr Benutzer hat die Anweisungen unter befolgt, [Herstellen einer Verbindung mit einem Repository](#) um eine Verbindung zu den CodeCommit Repositorys herzustellen, dann verwendet der Benutzer die erweiterten Funktionen von `git-remote-codecommit` und `Git`, um die CodeCommit Repositorys `git clone`, `git push`, `git pull` auf die er Zugriff hat, aufzurufen und zu klonen, zu ihnen zu pushen und von ihnen abzurufen. Beispiel zum Klonen eines Repositorys:

```
git clone codecommit://CodeAccess@MyDemoRepo
```

`Git-Commit`-, `Push`- und `Pull`-Befehle verwenden reguläre `Git`-Syntax.

Wenn der Benutzer das AWS CLI und das AWS CLI benannte Profil verwendet, das mit den rotierenden Zugangsdaten verknüpft ist, werden Ergebnisse zurückgegeben, die auf dieses Profil beschränkt sind.

Identity and Access Management für AWS CodeCommit

AWS Identity and Access Management (IAM) hilft einem Administrator AWS-Service, den Zugriff auf AWS Ressourcen sicher zu kontrollieren. IAM-Administratoren kontrollieren, wer authentifiziert (angemeldet) und autorisiert werden kann (über Berechtigungen verfügt), um Ressourcen zu

verwenden. CodeCommit IAM ist ein Programm AWS-Service , das Sie ohne zusätzliche Kosten nutzen können.

Themen

- [Zielgruppe](#)
- [Authentifizierung mit Identitäten](#)
- [Verwalten des Zugriffs mit Richtlinien](#)
- [Authentifizierung und Zugriffskontrolle für AWS CodeCommit](#)
- [Wie AWS CodeCommit funktioniert mit IAM](#)
- [Ressourcenbasierte CodeCommit-Richtlinien](#)
- [Autorisierung auf der Grundlage von Tags CodeCommit](#)
- [CodeCommit IAM-Rollen](#)
- [AWS CodeCommit Beispiele für identitätsbasierte Richtlinien](#)
- [Problembehebung bei AWS CodeCommit Identität und Zugriff](#)

Zielgruppe

Die Art und Weise, wie Sie AWS Identity and Access Management (IAM) verwenden, hängt von der Arbeit ab, in der Sie tätig sind. CodeCommit

Dienstbenutzer — Wenn Sie den CodeCommit Dienst für Ihre Arbeit verwenden, stellt Ihnen Ihr Administrator die erforderlichen Anmeldeinformationen und Berechtigungen zur Verfügung. Wenn Sie für Ihre Arbeit mehr CodeCommit Funktionen verwenden, benötigen Sie möglicherweise zusätzliche Berechtigungen. Wenn Sie die Funktionsweise der Zugriffskontrolle nachvollziehen, wissen Sie bereits, welche Berechtigungen Sie von Ihrem Administrator anfordern müssen. Unter [Problembehebung bei AWS CodeCommit Identität und Zugriff](#) finden Sie nützliche Informationen für den Fall, dass Sie keinen Zugriff auf eine Feature in CodeCommit haben.

Serviceadministrator — Wenn Sie in Ihrem Unternehmen für die CodeCommit Ressourcen verantwortlich sind, haben Sie wahrscheinlich vollen Zugriff auf CodeCommit. Es ist Ihre Aufgabe, zu bestimmen, auf welche CodeCommit Funktionen und Ressourcen Ihre Servicebenutzer zugreifen sollen. Anschließend müssen Sie Anforderungen an Ihren IAM-Administrator senden, um die Berechtigungen der Servicebenutzer zu ändern. Lesen Sie die Informationen auf dieser Seite, um die Grundkonzepte von IAM nachzuvollziehen. Weitere Informationen darüber, wie Ihr Unternehmen IAM nutzen kann CodeCommit, finden Sie unter [Wie AWS CodeCommit funktioniert mit IAM](#).

IAM-Administrator: Wenn Sie als IAM-Administrator fungieren, sollten Sie Einzelheiten dazu kennen, wie Sie Richtlinien zur Verwaltung des Zugriffs auf CodeCommit verfassen können. Beispiele für CodeCommit identitätsbasierte Richtlinien, die Sie in IAM verwenden können, finden Sie unter. [AWS CodeCommit Beispiele für identitätsbasierte Richtlinien](#)

Authentifizierung mit Identitäten

Authentifizierung ist die Art und Weise, wie Sie sich AWS mit Ihren Identitätsdaten anmelden. Sie müssen als IAM-Benutzer authentifiziert (angemeldet AWS) sein oder eine IAM-Rolle annehmen. Root-Benutzer des AWS-Kontos

Sie können sich AWS als föderierte Identität anmelden, indem Sie Anmeldeinformationen verwenden, die über eine Identitätsquelle bereitgestellt wurden. AWS IAM Identity Center (IAM Identity Center) -Benutzer, die Single Sign-On-Authentifizierung Ihres Unternehmens und Ihre Google- oder Facebook-Anmeldeinformationen sind Beispiele für föderierte Identitäten. Wenn Sie sich als Verbundidentität anmelden, hat der Administrator vorher mithilfe von IAM-Rollen einen Identitätsverbund eingerichtet. Wenn Sie über den Verbund darauf zugreifen AWS, übernehmen Sie indirekt eine Rolle.

Je nachdem, welcher Benutzertyp Sie sind, können Sie sich beim AWS Management Console oder beim AWS Zugangsportal anmelden. Weitere Informationen zur Anmeldung finden Sie AWS unter [So melden Sie sich bei Ihrem an AWS-Konto](#) im AWS-Anmeldung Benutzerhandbuch.

Wenn Sie AWS programmgesteuert zugreifen, AWS stellt es ein Software Development Kit (SDK) und eine Befehlszeilenschnittstelle (CLI) bereit, um Ihre Anfragen mithilfe Ihrer Anmeldeinformationen kryptografisch zu signieren. Wenn Sie keine AWS Tools verwenden, müssen Sie Anfragen selbst signieren. Weitere Informationen zur Verwendung der empfohlenen Methode für die Selbstsignierung von Anforderungen finden Sie unter [AWS Signature Version 4 für API-Anforderungen](#) im IAM-Benutzerhandbuch.

Unabhängig von der verwendeten Authentifizierungsmethode müssen Sie möglicherweise zusätzliche Sicherheitsinformationen bereitstellen. AWS empfiehlt beispielsweise, die Multi-Faktor-Authentifizierung (MFA) zu verwenden, um die Sicherheit Ihres Kontos zu erhöhen. Weitere Informationen finden Sie unter [Multi-Faktor-Authentifizierung](#) im AWS IAM Identity Center - Benutzerhandbuch und [AWS Multi-Faktor-Authentifizierung \(MFA\) in IAM](#) im IAM-Benutzerhandbuch.

AWS-Konto Root-Benutzer

Wenn Sie ein neues AWS-Konto erstellen, beginnen Sie mit einer Anmeldeidentität, die vollständigen Zugriff auf alle AWS-Services Ressourcen im Konto hat. Diese Identität wird als AWS-Konto Root-

Benutzer bezeichnet. Sie können darauf zugreifen, indem Sie sich mit der E-Mail-Adresse und dem Passwort anmelden, mit denen Sie das Konto erstellt haben. Wir raten ausdrücklich davon ab, den Root-Benutzer für Alltagsaufgaben zu verwenden. Schützen Sie Ihre Root-Benutzer-Anmeldeinformationen. Verwenden Sie diese nur, um die Aufgaben auszuführen, die nur der Root-Benutzer ausführen kann. Eine vollständige Liste der Aufgaben, für die Sie sich als Root-Benutzer anmelden müssen, finden Sie unter [Aufgaben, die Root-Benutzer-Anmeldeinformationen erfordern](#) im IAM-Benutzerhandbuch.

IAM-Benutzer und -Gruppen

Ein [IAM-Benutzer](#) ist eine Identität innerhalb von Ihrem AWS-Konto, die über spezifische Berechtigungen für eine einzelne Person oder Anwendung verfügt. Wenn möglich, empfehlen wir, temporäre Anmeldeinformationen zu verwenden, anstatt IAM-Benutzer zu erstellen, die langfristige Anmeldeinformationen wie Passwörter und Zugriffsschlüssel haben. Bei speziellen Anwendungsfällen, die langfristige Anmeldeinformationen mit IAM-Benutzern erfordern, empfehlen wir jedoch, die Zugriffsschlüssel zu rotieren. Weitere Informationen finden Sie unter [Regelmäßiges Rotieren von Zugriffsschlüsseln für Anwendungsfälle, die langfristige Anmeldeinformationen erfordern](#) im IAM-Benutzerhandbuch.

Eine [IAM-Gruppe](#) ist eine Identität, die eine Sammlung von IAM-Benutzern angibt. Sie können sich nicht als Gruppe anmelden. Mithilfe von Gruppen können Sie Berechtigungen für mehrere Benutzer gleichzeitig angeben. Gruppen vereinfachen die Verwaltung von Berechtigungen, wenn es zahlreiche Benutzer gibt. Sie könnten beispielsweise eine Gruppe benennen IAMAdmins und dieser Gruppe Berechtigungen zur Verwaltung von IAM-Ressourcen erteilen.

Benutzer unterscheiden sich von Rollen. Ein Benutzer ist einer einzigen Person oder Anwendung eindeutig zugeordnet. Eine Rolle kann von allen Personen angenommen werden, die sie benötigen. Benutzer besitzen dauerhafte Anmeldeinformationen. Rollen stellen temporäre Anmeldeinformationen bereit. Weitere Informationen finden Sie unter [Anwendungsfälle für IAM-Benutzer](#) im IAM-Benutzerhandbuch.

IAM-Rollen

Eine [IAM-Rolle](#) ist eine Identität innerhalb von Ihrem AWS-Konto, die über bestimmte Berechtigungen verfügt. Sie ist einem IAM-Benutzer vergleichbar, jedoch nicht mit einer bestimmten Person verknüpft. Um vorübergehend eine IAM-Rolle in der zu übernehmen AWS Management Console, können Sie [von einer Benutzer- zu einer IAM-Rolle \(Konsole\) wechseln](#). Sie können eine Rolle übernehmen, indem Sie eine AWS CLI oder AWS API-Operation aufrufen oder eine

benutzerdefinierte URL verwenden. Weitere Informationen zu Methoden für die Verwendung von Rollen finden Sie unter [Methoden für die Übernahme einer Rolle](#) im IAM-Benutzerhandbuch.

IAM-Rollen mit temporären Anmeldeinformationen sind in folgenden Situationen hilfreich:

- **Verbundbenutzerzugriff** – Um einer Verbundidentität Berechtigungen zuzuweisen, erstellen Sie eine Rolle und definieren Berechtigungen für die Rolle. Wird eine Verbundidentität authentifiziert, so wird die Identität der Rolle zugeordnet und erhält die von der Rolle definierten Berechtigungen. Informationen zu Rollen für den Verbund finden Sie unter [Erstellen von Rollen für externe Identitätsanbieter \(Verbund\)](#) im IAM-Benutzerhandbuch. Wenn Sie IAM Identity Center verwenden, konfigurieren Sie einen Berechtigungssatz. Wenn Sie steuern möchten, worauf Ihre Identitäten nach der Authentifizierung zugreifen können, korreliert IAM Identity Center den Berechtigungssatz mit einer Rolle in IAM. Informationen zu Berechtigungssätzen finden Sie unter [Berechtigungssätze](#) im AWS IAM Identity Center -Benutzerhandbuch.
- **Temporäre IAM-Benutzerberechtigungen** – Ein IAM-Benutzer oder eine -Rolle kann eine IAM-Rolle übernehmen, um vorübergehend andere Berechtigungen für eine bestimmte Aufgabe zu erhalten.
- **Kontoübergreifender Zugriff** – Sie können eine IAM-Rolle verwenden, um einem vertrauenswürdigen Prinzipal in einem anderen Konto den Zugriff auf Ressourcen in Ihrem Konto zu ermöglichen. Rollen stellen die primäre Möglichkeit dar, um kontoübergreifendem Zugriff zu gewähren. Bei einigen können Sie AWS-Services jedoch eine Richtlinie direkt an eine Ressource anhängen (anstatt eine Rolle als Proxy zu verwenden). Informationen zu den Unterschieden zwischen Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter [Kontoübergreifender Ressourcenzugriff in IAM](#) im IAM-Benutzerhandbuch.
- **Serviceübergreifender Zugriff** — Einige AWS-Services verwenden Funktionen in anderen AWS-Services. Wenn Sie beispielsweise einen Service aufrufen, ist es üblich, dass dieser Service Anwendungen in Amazon ausführt EC2 oder Objekte in Amazon S3 speichert. Ein Dienst kann dies mit den Berechtigungen des aufrufenden Prinzipals mit einer Servic Rolle oder mit einer serviceverknüpften Rolle tun.
- **Forward Access Sessions (FAS)** — Wenn Sie einen IAM-Benutzer oder eine IAM-Rolle verwenden, um Aktionen auszuführen AWS, gelten Sie als Principal. Bei einigen Services könnte es Aktionen geben, die dann eine andere Aktion in einem anderen Service initiieren. FAS verwendet die Berechtigungen des Prinzipals, der einen aufruft AWS-Service, in Kombination mit der Anfrage, Anfragen an AWS-Service nachgelagerte Dienste zu stellen. FAS-Anfragen werden nur gestellt, wenn ein Dienst eine Anfrage erhält, für deren Abschluss Interaktionen mit anderen AWS-Services oder Ressourcen erforderlich sind. In diesem Fall müssen Sie über

Berechtigungen zum Ausführen beider Aktionen verfügen. Einzelheiten zu den Richtlinien für FAS-Anfragen finden Sie unter [Zugriffssitzungen weiterleiten](#).

- **Servicerolle** – Eine Servicerolle ist eine [IAM-Rolle](#), die ein Service übernimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle innerhalb von IAM erstellen, ändern und löschen. Weitere Informationen finden Sie unter [Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service](#) im IAM-Benutzerhandbuch.
- **Dienstbezogene Rolle** — Eine dienstbezogene Rolle ist eine Art von Servicerolle, die mit einer verknüpft ist. AWS-Service Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Servicebezogene Rollen erscheinen in Ihrem Dienst AWS-Konto und gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.
- **Auf Amazon ausgeführte Anwendungen EC2** — Sie können eine IAM-Rolle verwenden, um temporäre Anmeldeinformationen für Anwendungen zu verwalten, die auf einer EC2 Instance ausgeführt werden und AWS API-Anfragen stellen AWS CLI . Dies ist dem Speichern von Zugriffsschlüsseln innerhalb der EC2 Instance vorzuziehen. Um einer EC2 Instanz eine AWS Rolle zuzuweisen und sie allen ihren Anwendungen zur Verfügung zu stellen, erstellen Sie ein Instanzprofil, das an die Instanz angehängt ist. Ein Instanzprofil enthält die Rolle und ermöglicht Programmen, die auf der EC2 Instanz ausgeführt werden, temporäre Anmeldeinformationen abzurufen. Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter [Verwenden einer IAM-Rolle, um Berechtigungen für Anwendungen zu gewähren, die auf EC2 Amazon-Instances ausgeführt werden](#).

Verwalten des Zugriffs mit Richtlinien

Sie kontrollieren den Zugriff, AWS indem Sie Richtlinien erstellen und diese an AWS Identitäten oder Ressourcen anhängen. Eine Richtlinie ist ein Objekt, AWS das, wenn es einer Identität oder Ressource zugeordnet ist, deren Berechtigungen definiert. AWS wertet diese Richtlinien aus, wenn ein Prinzipal (Benutzer, Root-Benutzer oder Rollensitzung) eine Anfrage stellt. Die Berechtigungen in den Richtlinien legen fest, ob eine Anforderung zugelassen oder abgelehnt wird. Die meisten Richtlinien werden AWS als JSON-Dokumente gespeichert. Weitere Informationen zu Struktur und Inhalten von JSON-Richtliniendokumenten finden Sie unter [Übersicht über JSON-Richtlinien](#) im IAM-Benutzerhandbuch.

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Standardmäßig haben Benutzer, Gruppen und Rollen keine Berechtigungen. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern die Berechtigung erteilen, Aktionen für die Ressourcen auszuführen, die sie benötigen. Der Administrator kann dann die IAM-Richtlinien zu Rollen hinzufügen, und Benutzer können die Rollen annehmen.

IAM-Richtlinien definieren Berechtigungen für eine Aktion unabhängig von der Methode, die Sie zur Ausführung der Aktion verwenden. Angenommen, es gibt eine Richtlinie, die Berechtigungen für die `iam:GetRole`-Aktion erteilt. Ein Benutzer mit dieser Richtlinie kann Rolleninformationen von der AWS Management Console AWS CLI, der oder der AWS API abrufen.

Identitätsbasierte Richtlinien

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität anfügen können, wie z. B. IAM-Benutzern, -Benutzergruppen oder -Rollen. Diese Richtlinien steuern, welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter [Definieren benutzerdefinierter IAM-Berechtigungen mit vom Kunden verwalteten Richtlinien](#) im IAM-Benutzerhandbuch.

Identitätsbasierte Richtlinien können weiter als Inline-Richtlinien oder verwaltete Richtlinien kategorisiert werden. Inline-Richtlinien sind direkt in einen einzelnen Benutzer, eine einzelne Gruppe oder eine einzelne Rolle eingebettet. Verwaltete Richtlinien sind eigenständige Richtlinien, die Sie mehreren Benutzern, Gruppen und Rollen in Ihrem System zuordnen können AWS-Konto. Zu den verwalteten Richtlinien gehören AWS verwaltete Richtlinien und vom Kunden verwaltete Richtlinien. Informationen dazu, wie Sie zwischen einer verwalteten Richtlinie und einer Inline-Richtlinie wählen, finden Sie unter [Auswählen zwischen verwalteten und eingebundenen Richtlinien](#) im IAM-Benutzerhandbuch.

Ressourcenbasierte Richtlinien

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. Beispiele für ressourcenbasierte Richtlinien sind IAM-Rollen-Vertrauensrichtlinien und Amazon-S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Für die Ressource, an welche die Richtlinie angehängt ist, legt die Richtlinie fest, welche Aktionen ein bestimmter Prinzipal unter welchen Bedingungen für diese Ressource ausführen kann. Sie müssen in einer ressourcenbasierten Richtlinie [einen Prinzipal angeben](#). Zu den Prinzipalen können Konten, Benutzer, Rollen, Verbundbenutzer oder gehören. AWS-Services

Ressourcenbasierte Richtlinien sind Richtlinien innerhalb dieses Diensts. Sie können AWS verwaltete Richtlinien von IAM nicht in einer ressourcenbasierten Richtlinie verwenden.

Zugriffskontrolllisten () ACLs

Zugriffskontrolllisten (ACLs) steuern, welche Principals (Kontomitglieder, Benutzer oder Rollen) über Zugriffsberechtigungen für eine Ressource verfügen. ACLs ähneln ressourcenbasierten Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

Amazon S3 und Amazon VPC sind Beispiele für Dienste, die Unterstützung ACLs bieten. AWS WAF Weitere Informationen finden Sie unter [Übersicht über ACLs die Zugriffskontrollliste \(ACL\)](#) im Amazon Simple Storage Service Developer Guide.

Weitere Richtlinientypen

AWS unterstützt zusätzliche, weniger verbreitete Richtlinientypen. Diese Richtlinientypen können die maximalen Berechtigungen festlegen, die Ihnen von den häufiger verwendeten Richtlinientypen erteilt werden können.

- **Berechtigungsgrenzen** – Eine Berechtigungsgrenze ist ein erweitertes Feature, mit der Sie die maximalen Berechtigungen festlegen können, die eine identitätsbasierte Richtlinie einer IAM-Entität (IAM-Benutzer oder -Rolle) erteilen kann. Sie können eine Berechtigungsgrenze für eine Entität festlegen. Die daraus resultierenden Berechtigungen sind der Schnittpunkt der identitätsbasierten Richtlinien einer Entität und ihrer Berechtigungsgrenzen. Ressourcenbasierte Richtlinien, die den Benutzer oder die Rolle im Feld `Principal` angeben, werden nicht durch Berechtigungsgrenzen eingeschränkt. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen über Berechtigungsgrenzen finden Sie unter [Berechtigungsgrenzen für IAM-Entitäten](#) im IAM-Benutzerhandbuch.
- **Dienststeuerungsrichtlinien (SCPs)** — SCPs sind JSON-Richtlinien, die die maximalen Berechtigungen für eine Organisation oder Organisationseinheit (OU) in festlegen. AWS Organizations AWS Organizations ist ein Dienst zur Gruppierung und zentralen Verwaltung mehrerer Objekte AWS-Konten , die Ihrem Unternehmen gehören. Wenn Sie alle Funktionen in einer Organisation aktivieren, können Sie Richtlinien zur Servicesteuerung (SCPs) auf einige oder alle Ihre Konten anwenden. Das SCP schränkt die Berechtigungen für Entitäten in Mitgliedskonten ein, einschließlich der einzelnen Root-Benutzer des AWS-Kontos Entitäten. Weitere Informationen zu Organizations und SCPs finden Sie unter [Richtlinien zur Servicesteuerung](#) im AWS Organizations Benutzerhandbuch.

- **Ressourcenkontrollrichtlinien (RCPs)** — RCPs sind JSON-Richtlinien, mit denen Sie die maximal verfügbaren Berechtigungen für Ressourcen in Ihren Konten festlegen können, ohne die IAM-Richtlinien aktualisieren zu müssen, die jeder Ressource zugeordnet sind, deren Eigentümer Sie sind. Das RCP schränkt die Berechtigungen für Ressourcen in Mitgliedskonten ein und kann sich auf die effektiven Berechtigungen für Identitäten auswirken, einschließlich der Root-Benutzer des AWS-Kontos, unabhängig davon, ob sie zu Ihrer Organisation gehören. Weitere Informationen zu Organizations RCPs, einschließlich einer Liste AWS-Services dieser Support-Leistungen RCPs, finden Sie unter [Resource Control Policies \(RCPs\)](#) im AWS Organizations Benutzerhandbuch.
- **Sitzungsrichtlinien** – Sitzungsrichtlinien sind erweiterte Richtlinien, die Sie als Parameter übergeben, wenn Sie eine temporäre Sitzung für eine Rolle oder einen verbundenen Benutzer programmgesteuert erstellen. Die resultierenden Sitzungsberechtigungen sind eine Schnittmenge der auf der Identität des Benutzers oder der Rolle basierenden Richtlinien und der Sitzungsrichtlinien. Berechtigungen können auch aus einer ressourcenbasierten Richtlinie stammen. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen finden Sie unter [Sitzungsrichtlinien](#) im IAM-Benutzerhandbuch.

Mehrere Richtlinientypen

Wenn mehrere auf eine Anforderung mehrere Richtlinientypen angewendet werden können, sind die entsprechenden Berechtigungen komplizierter. Informationen darüber, wie AWS bestimmt wird, ob eine Anfrage zulässig ist, wenn mehrere Richtlinientypen betroffen sind, finden Sie im IAM-Benutzerhandbuch unter [Bewertungslogik für Richtlinien](#).

Authentifizierung und Zugriffskontrolle für AWS CodeCommit

Für den Zugriff auf AWS CodeCommit sind Anmeldeinformationen erforderlich. Diese Anmeldeinformationen müssen über Zugriffsberechtigungen für AWS Ressourcen wie CodeCommit Repositories und Ihren IAM-Benutzer verfügen, mit dem Sie Ihre Git-Anmeldeinformationen oder den öffentlichen SSH-Schlüssel verwalten, den Sie für die Herstellung von Git-Verbindungen verwenden. In den folgenden Abschnitten erfahren Sie, wie Sie [AWS Identity and Access Management \(IAM\)](#) verwenden und CodeCommit wie Sie den Zugriff auf Ihre Ressourcen sichern können:

- [Authentifizierung](#)
- [Zugriffskontrolle](#)

Authentifizierung

Da CodeCommit Repositorys Git-basiert sind und die grundlegenden Funktionen von Git unterstützen, einschließlich Git-Anmeldeinformationen, empfehlen wir, dass Sie bei der Arbeit mit einem IAM-Benutzer arbeiten. CodeCommit Du kannst CodeCommit mit anderen Identitätstypen darauf zugreifen, aber die anderen Identitätstypen unterliegen Einschränkungen, wie unten beschrieben.

Identitätstypen:

- IAM-Benutzer — Ein [IAM-Benutzer](#) ist eine Identität innerhalb Ihres Amazon Web Services Services-Kontos, die über spezifische benutzerdefinierte Berechtigungen verfügt. Ein IAM-Benutzer kann beispielsweise über Berechtigungen zum Erstellen und Verwalten von Git-Anmeldeinformationen für den Zugriff auf CodeCommit Repositorys verfügen. Dies ist der empfohlene Benutzertyp für die Arbeit mit. CodeCommit [Sie können einen IAM-Benutzernamen und ein Passwort verwenden, um sich auf sicheren AWS Webseiten wie den AWS Management Console AWS Diskussionsforen oder dem AWS -Support Center anzumelden.](#)

Sie können Git-Anmeldeinformationen generieren oder öffentliche SSH-Schlüssel mit Ihrem IAM-Benutzer verknüpfen, oder Sie können sie installieren und konfigurieren. `git-remote-codecommit` Dies sind die einfachsten Möglichkeiten, Git so einzurichten, dass es mit deinen CodeCommit Repositorys funktioniert. Mit [Git-Anmeldeinformationen](#) generieren Sie einen statischen Benutzernamen und ein statisches Passwort in IAM. Sie können dieselben Anmeldeinformationen für HTTPS-Verbindungen dann auch mit Git und jedem Drittanbieter-Tool verwenden, das die Authentifizierung mit Git-Benutzername und -Passwort unterstützt. Mit SSH-Verbindungen erstellen Sie öffentliche und private Schlüsseldateien auf Ihrem lokalen Computer, die Git CodeCommit verwenden und für die SSH-Authentifizierung verwenden. Sie verknüpfen den öffentlichen Schlüssel mit Ihrem IAM-Benutzer und speichern den privaten Schlüssel auf Ihrem lokalen Computer. `git-remote-codecommit` erweitert Git selbst und erfordert keine Einrichtung von Git-Anmeldeinformationen für den Benutzer.

Darüber hinaus können Sie [Zugriffsschlüssel](#) für jeden Benutzer erstellen. Verwenden Sie Zugriffstasten, wenn Sie programmgesteuert auf AWS Dienste zugreifen, entweder über [einen der AWS SDKs](#) oder mithilfe von [AWS Command Line Interface \(AWS CLI\)](#). Das SDK und die CLI-Tools verwenden die Zugriffsschlüssel, um Ihre Anfragen verschlüsselt zu signieren. Wenn Sie die AWS Tools nicht verwenden, müssen Sie die Anfragen selbst signieren. CodeCommit unterstützt Signature Version 4, ein Protokoll zur Authentifizierung eingehender API-Anfragen.

Weitere Informationen zur Authentifizierung von Anfragen finden Sie unter [Signature Version 4-Signaturprozess](#) im Allgemeine AWS-Referenz.

- Root-Benutzer für das Amazon Web Services Services-Konto — Wenn Sie sich registrieren AWS, geben Sie eine E-Mail-Adresse und ein Passwort an, die mit Ihrem Amazon Web Services Services-Konto verknüpft sind. Dies sind Ihre Root-Anmeldeinformationen. Sie bieten vollständigen Zugriff auf alle Ihre AWS -Ressourcen. Einige CodeCommit Funktionen sind für Benutzer mit Root-Konten nicht verfügbar. Darüber hinaus besteht die einzige Möglichkeit, Git mit Ihrem Root-Konto zu verwenden, darin, entweder den Credential Helper zu installieren und zu konfigurieren git-remote-codecommit (empfohlen) oder den AWS Credential Helper zu konfigurieren, der AWS CLI im Lieferumfang von enthalten ist. Sie können mit Ihrem Stammkonto-Benutzer keine GIT-Anmeldeinformationen oder öffentliche/private SSH-Schlüsselpaare verwenden. Aus diesen Gründen empfehlen wir nicht, den Benutzer Ihres Root-Kontos zu verwenden, wenn Sie mit CodeCommit interagieren.

 Important

Aus Sicherheitsgründen empfehlen wir, die Root-Anmeldeinformationen nur zum Erstellen eines Administrator-Benutzers zu verwenden. Hierbei handelt es sich um einen IAM-Benutzer mit vollständigen Berechtigungen für Ihr AWS -Konto. Anschließend können Sie mit diesem Administrator-Benutzer andere IAM-Benutzer und -Rollen mit eingeschränkten Berechtigungen erstellen. Weitere Informationen finden Sie unter [Bewährte Methoden für IAM](#) und [Erstellen eines Administratorbenutzers und einer Gruppe](#) im IAM-Benutzerhandbuch.

- IAM Identity Center und Benutzer in IAM Identity Center — AWS IAM Identity Center erweitert die Funktionen von AWS Identity and Access Management und bietet einen zentralen Ort, an dem die Benutzerverwaltung und deren Zugriff auf Cloud-Anwendungen sowie deren Zugriff auf Cloud-Anwendungen AWS-Konten zusammengeführt werden. Obwohl IAM Identity Center für die meisten Benutzer, die damit arbeiten AWS, als bewährte Methode empfohlen wird, bietet es derzeit keine Mechanismen für Git-Anmeldeinformationen oder SSH-Schlüsselpaare. Diese Benutzer können CodeCommit Repositorys so installieren und konfigurieren, git-remote-codecommit dass sie lokal geklont werden, aber nicht alle integrierten Entwicklungsumgebungen (IDEs) unterstützen das Klonen, Pushen oder Ziehen mit. git-remote-codecommit

Als bewährte Methode sollten menschliche Benutzer, einschließlich Benutzer, die Administratorzugriff benötigen, für den Zugriff AWS-Services mithilfe temporärer Anmeldeinformationen den Verbund mit einem Identitätsanbieter verwenden.

Eine föderierte Identität ist ein Benutzer aus Ihrem Unternehmensbenutzerverzeichnis, einem Web-Identitätsanbieter AWS Directory Service, dem Identity Center-Verzeichnis oder einem beliebigen Benutzer, der mithilfe AWS-Services von Anmeldeinformationen zugreift, die über eine Identitätsquelle bereitgestellt wurden. Wenn föderierte Identitäten darauf zugreifen AWS-Konten, übernehmen sie Rollen, und die Rollen stellen temporäre Anmeldeinformationen bereit.

Für die zentrale Zugriffsverwaltung empfehlen wir Ihnen, AWS IAM Identity Center zu verwenden. Sie können Benutzer und Gruppen in IAM Identity Center erstellen, oder Sie können eine Verbindung zu einer Gruppe von Benutzern und Gruppen in Ihrer eigenen Identitätsquelle herstellen und diese synchronisieren, um sie in all Ihren AWS-Konten Anwendungen zu verwenden. Informationen zu IAM Identity Center finden Sie unter [Was ist IAM Identity Center?](#) im AWS IAM Identity Center -Benutzerhandbuch.

- IAM-Rolle — Wie ein IAM-Benutzer ist auch eine [IAM-Rolle eine IAM-Identität](#), die Sie in Ihrem Konto erstellen können, um bestimmte Berechtigungen zu gewähren.

Eine [IAM-Rolle](#) ist eine Identität innerhalb Ihres AWS-Konto Unternehmens, für die bestimmte Berechtigungen gelten. Sie ist einem IAM-Benutzer vergleichbar, jedoch nicht mit einer bestimmten Person verknüpft. Um vorübergehend eine IAM-Rolle in der zu übernehmen AWS Management Console, können Sie [von einer Benutzer- zu einer IAM-Rolle \(Konsole\) wechseln](#). Sie können eine Rolle übernehmen, indem Sie eine AWS CLI oder AWS API-Operation aufrufen oder eine benutzerdefinierte URL verwenden. Weitere Informationen zu Methoden für die Verwendung von Rollen finden Sie unter [Methoden für die Übernahme einer Rolle](#) im IAM-Benutzerhandbuch.

IAM-Rollen mit temporären Anmeldeinformationen sind in folgenden Situationen hilfreich:

- Verbundbenutzerzugriff – Um einer Verbundidentität Berechtigungen zuzuweisen, erstellen Sie eine Rolle und definieren Berechtigungen für die Rolle. Wird eine Verbundidentität authentifiziert, so wird die Identität der Rolle zugeordnet und erhält die von der Rolle definierten Berechtigungen. Informationen zu Rollen für den Verbund finden Sie unter [Erstellen von Rollen für externe Identitätsanbieter \(Verbund\)](#) im IAM-Benutzerhandbuch. Wenn Sie IAM Identity Center verwenden, konfigurieren Sie einen Berechtigungssatz. Wenn Sie steuern möchten, worauf Ihre Identitäten nach der Authentifizierung zugreifen können, korreliert IAM Identity Center den Berechtigungssatz mit einer Rolle in IAM. Informationen zu Berechtigungssätzen finden Sie unter [Berechtigungssätze](#) im AWS IAM Identity Center -Benutzerhandbuch.
- Temporäre IAM-Benutzerberechtigungen – Ein IAM-Benutzer oder eine -Rolle kann eine IAM-Rolle übernehmen, um vorübergehend andere Berechtigungen für eine bestimmte Aufgabe zu erhalten.

- **Kontoübergreifender Zugriff** – Sie können eine IAM-Rolle verwenden, um einem vertrauenswürdigen Prinzipal in einem anderen Konto den Zugriff auf Ressourcen in Ihrem Konto zu ermöglichen. Rollen stellen die primäre Möglichkeit dar, um kontoübergreifendem Zugriff zu gewähren. Bei einigen können Sie AWS-Services jedoch eine Richtlinie direkt an eine Ressource anhängen (anstatt eine Rolle als Proxy zu verwenden). Informationen zu den Unterschieden zwischen Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter [Kontoübergreifender Ressourcenzugriff in IAM](#) im IAM-Benutzerhandbuch.
- **Serviceübergreifender Zugriff** — Einige AWS-Services verwenden Funktionen in anderen AWS-Services. Wenn Sie beispielsweise in einem Service einen Anruf tätigen, ist es üblich, dass dieser Service Anwendungen in Amazon ausführt EC2 oder Objekte in Amazon S3 speichert. Ein Dienst kann dies mit den Berechtigungen des aufrufenden Prinzipals mit einer Servicerolle oder mit einer serviceverknüpften Rolle tun.
- **Forward Access Sessions (FAS)** — Wenn Sie einen IAM-Benutzer oder eine IAM-Rolle verwenden, um Aktionen auszuführen AWS, gelten Sie als Principal. Bei einigen Services könnte es Aktionen geben, die dann eine andere Aktion in einem anderen Service initiieren. FAS verwendet die Berechtigungen des Prinzipals, der einen aufruft AWS-Service, in Kombination mit der Anfrage, Anfragen an AWS-Service nachgelagerte Dienste zu stellen. FAS-Anfragen werden nur gestellt, wenn ein Dienst eine Anfrage erhält, für deren Abschluss Interaktionen mit anderen AWS-Services oder Ressourcen erforderlich sind. In diesem Fall müssen Sie über Berechtigungen zum Ausführen beider Aktionen verfügen. Einzelheiten zu den Richtlinien für FAS-Anfragen finden Sie unter [Zugriffssitzungen weiterleiten](#).
- **Servicerolle** – Eine Servicerolle ist eine [IAM-Rolle](#), die ein Service übernimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle innerhalb von IAM erstellen, ändern und löschen. Weitere Informationen finden Sie unter [Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service](#) im IAM-Benutzerhandbuch.
- **Dienstbezogene Rolle** — Eine dienstbezogene Rolle ist eine Art von Servicerolle, die mit einer verknüpft ist. AWS-Service Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Servicebezogene Rollen erscheinen in Ihrem Dienst AWS-Konto und gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.
- **Auf Amazon ausgeführte Anwendungen EC2** — Sie können eine IAM-Rolle verwenden, um temporäre Anmeldeinformationen für Anwendungen zu verwalten, die auf einer EC2 Instance ausgeführt werden und AWS API-Anfragen stellen AWS CLI . Dies ist dem Speichern von Zugriffsschlüsseln innerhalb der EC2 Instance vorzuziehen. Um einer EC2 Instanz eine AWS Rolle zuzuweisen und sie allen ihren Anwendungen zur Verfügung zu stellen, erstellen Sie ein

Instanzprofil, das an die Instanz angehängt ist. Ein Instanzprofil enthält die Rolle und ermöglicht Programmen, die auf der EC2 Instanz ausgeführt werden, temporäre Anmeldeinformationen abzurufen. Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter [Verwenden einer IAM-Rolle, um Berechtigungen für Anwendungen zu gewähren, die auf EC2 Amazon-Instances ausgeführt werden](#).

Note

Sie können mit verbundenen Benutzern keine GIT-Anmeldeinformationen oder öffentliche/private SSH-Schlüsselpaare verwenden. Darüber hinaus stehen für verbundene Benutzer keine Benutzereinstellungen zur Verfügung. Informationen zum Einrichten von Verbindungen mithilfe des Verbundzugriffs finden Sie unter [Einrichtungsschritte für HTTPS-Verbindungen AWS CodeCommit mit git-remote-codecommit](#).

Zugriffskontrolle

Sie können über gültige Anmeldeinformationen verfügen, um Ihre Anfragen zu authentifizieren, aber ohne die entsprechenden Berechtigungen können Sie keine CodeCommit Ressourcen erstellen oder darauf zugreifen. Beispielsweise müssen Sie über Berechtigungen zum Anzeigen von Repositories, zum Senden von Code, zum Erstellen und Verwalten von Git-Anmeldeinformationen usw. verfügen.

In den folgenden Abschnitten wird beschrieben, wie Sie Berechtigungen für CodeCommit verwalten. Wir empfehlen Ihnen, zunächst die Übersicht zu lesen.

- [Überblick über die Verwaltung von Zugriffsberechtigungen für Ihre CodeCommit Ressourcen](#)
- [Verwendung identitätsbasierter Richtlinien \(IAM-Richtlinien\) für CodeCommit](#)
- [Referenz für CodeCommit-Berechtigungen](#)

Überblick über die Verwaltung von Zugriffsberechtigungen für Ihre CodeCommit Ressourcen

Jede AWS Ressource gehört einem Amazon Web Services Services-Konto. Die Berechtigungen zum Erstellen einer Ressource oder für den Zugriff darauf werden durch Berechtigungsrichtlinien geregelt. Ein Kontoadministrator kann Berechtigungsrichtlinien an IAM-Identitäten (Benutzer, Gruppen und Rollen) anfügen. Einige Dienste, wie z. B. AWS Lambda, unterstützen auch das Anhängen von Berechtigungsrichtlinien an Ressourcen.

 Note

Ein Kontoadministrator (oder Administratorbenutzer) ist ein Benutzer mit Administratorrechten. Weitere Informationen finden Sie unter [Bewährte Methoden für IAM](#) im IAM-Benutzerhandbuch.

Beim Erteilen von Berechtigungen entscheiden Sie, wer die Berechtigungen erhält, für welche Ressourcen die Berechtigungen gelten und welche Aktionen an diesen Ressourcen gestattet werden sollen.

Themen

- [CodeCommit Ressourcen und Abläufe](#)
- [Grundlegendes zum Eigentum an Ressourcen](#)
- [Verwaltung des Zugriffs auf -Ressourcen](#)
- [Eingrenzung des Ressourcenbereichs CodeCommit](#)
- [Angaben der Richtlinienelemente: Ressourcen, Aktionen, Effekte und Prinzipale](#)
- [Angaben von Bedingungen in einer Richtlinie](#)

CodeCommit Ressourcen und Abläufe

CodeCommit ist die primäre Ressource ein Repository. Jeder dieser Ressourcen ist ein eindeutiger Amazon-Ressourcenname (ARN) zugeordnet. In einer Richtlinie identifizieren Sie die Ressource, für welche die Richtlinie gilt, mithilfe eines Amazon-Ressourcenname (ARN). Weitere Informationen zu ARNs finden Sie unter [Amazon Resource Names \(ARN\) und AWS Service Namespaces](#) in der Allgemeinen Amazon Web Services-Referenz. CodeCommit unterstützt derzeit keine anderen Ressourcentypen, die als Unterressourcen bezeichnet werden.

In der folgenden Tabelle wird beschrieben, wie Ressourcen angegeben CodeCommit werden.

Ressourcentyp	ARN-Format
Repository	arn:aws:codecommit::: <i>region account-id repository-name</i>
Alle CodeCommit Repositorien	arn:aws:codecommit:*

Ressourcentyp	ARN-Format
Alle CodeCommit Repositorys, die dem angegebenen Konto im angegebenen Konto gehören AWS-Region	arn:aws:codecommit::: * <i>region account-id</i>

Note

Die meisten AWS Dienste behandeln einen Doppelpunkt (:) oder einen Schrägstrich (/) als dasselbe Zeichen. ARNs CodeCommit Erfordert jedoch eine exakte Übereinstimmung der Ressourcenmuster und Regeln. Verwenden Sie also die richtigen ARN-Zeichen zum Erstellen von Ereignismustern, sodass sie mit der ARN-Syntax in der Ressource übereinstimmen.

Sie können beispielsweise folgendermaßen ein bestimmtes Repository (*MyDemoRepo*) mit dem ARN in der Anweisung angeben:

```
"Resource": "arn:aws:codecommit:us-west-2:111111111111:MyDemoRepo"
```

Um alle Repositorys anzugeben, die zu einem bestimmten Konto gehören, verwenden Sie das Platzhalterzeichen (*) wie folgt:

```
"Resource": "arn:aws:codecommit:us-west-2:111111111111:*"
```

Um alle Ressourcen anzugeben oder falls eine bestimmte API-Aktion dies nicht unterstützt ARNs, verwenden Sie das Platzhalterzeichen (*) im Resource Element wie folgt:

```
"Resource": "*"
```

Sie können auch das Platzhalterzeichen (*) verwenden, um alle Ressourcen anzugeben, die mit einem Teil eines Repository-Namen übereinstimmen. Der folgende ARN gibt beispielsweise jedes CodeCommit Repository an, das mit dem Namen beginnt MyDemo und das für das Amazon Web Services Services-Konto registriert ist 111111111111 in us-east-2 AWS-Region:


```
arn:aws:codecommit:us-east-2:111111111111:MyDemo*
```

Eine Liste der verfügbaren Operationen, die mit den CodeCommit Ressourcen arbeiten, finden Sie unter [Referenz für CodeCommit-Berechtigungen](#).

Grundlegendes zum Eigentum an Ressourcen

Das Amazon Web Services Services-Konto besitzt die Ressourcen, die im Konto erstellt wurden, unabhängig davon, wer sie erstellt hat. Insbesondere ist der Ressourcenbesitzer das Amazon Web Services Services-Konto der [Prinzipalidentität](#) (d. h. das Root-Konto, ein IAM-Benutzer oder eine IAM-Rolle), das die Anfrage zur Ressourcenerstellung authentifiziert. Die Funktionsweise wird anhand der folgenden Beispiele deutlich:

- Wenn Sie in Ihrem Amazon Web Services Services-Konto einen IAM-Benutzer erstellen und diesem Benutzer Berechtigungen zum Erstellen von CodeCommit Ressourcen gewähren, kann der Benutzer CodeCommit Ressourcen erstellen. Ihr Amazon Web Services Services-Konto, zu dem der Benutzer gehört, besitzt jedoch die CodeCommit Ressourcen.
- Wenn Sie die Root-Kontoanmeldeinformationen Ihres Amazon Web Services Services-Kontos verwenden, um eine Regel zu erstellen, ist Ihr Amazon Web Services Services-Konto der Eigentümer der CodeCommit Ressource.
- Wenn Sie in Ihrem Amazon Web Services Services-Konto eine IAM-Rolle mit Berechtigungen zum Erstellen von CodeCommit Ressourcen erstellen, kann jeder, der die Rolle übernehmen kann, CodeCommit Ressourcen erstellen. Ihr Amazon Web Services Services-Konto, zu dem die Rolle gehört, besitzt die CodeCommit Ressourcen.

Verwaltung des Zugriffs auf -Ressourcen

Um den Zugriff auf AWS Ressourcen zu verwalten, verwenden Sie Berechtigungsrichtlinien. Eine Berechtigungsrichtlinie beschreibt, wer Zugriff auf welche Objekte hat. Im folgenden Abschnitt werden die Optionen zum Erstellen von Berechtigungsrichtlinien erläutert.

Note

In diesem Abschnitt wird die Verwendung von IAM im Kontext von CodeCommit beschrieben. Er enthält keine detaillierten Informationen über den IAM-Service. Weitere Informationen zu IAM finden Sie unter [Was ist IAM?](#) im IAM-Benutzerhandbuch. Für Informationen über die

Syntax und Beschreibungen von [IAM-Richtlinien](#) lesen Sie die IAM-Richtlinienreferenz im IAM-Benutzerhandbuch.

Berechtigungsrichtlinien, die mit einer IAM-Identität verknüpft sind, werden als identitätsbasierte Richtlinien (IAM-Richtlinien) bezeichnet. An Ressourcen angehängte Berechtigungsrichtlinien werden als ressourcenbasierte Richtlinien bezeichnet. CodeCommit unterstützt derzeit nur identitätsbasierte Richtlinien (IAM-Richtlinien).

Themen

- [Identitätsbasierte Richtlinien \(IAM-Richtlinien\)](#)
- [Ressourcenbasierte Richtlinien](#)

Identitätsbasierte Richtlinien (IAM-Richtlinien)

Um den Zugriff auf AWS Ressourcen zu verwalten, fügen Sie Berechtigungsrichtlinien an IAM-Identitäten an. In verwenden Sie identitätsbasierte Richtlinien CodeCommit, um den Zugriff auf Repositorys zu steuern. Sie können z. B. Folgendes tun:

- Ordnen Sie einem Benutzer oder einer Gruppe in Ihrem Konto eine Berechtigungsrichtlinie zu — Um einem Benutzer Berechtigungen zum Anzeigen von CodeCommit Ressourcen in der CodeCommit Konsole zu gewähren, fügen Sie einem Benutzer oder einer Gruppe, zu der der Benutzer gehört, eine identitätsbasierte Berechtigungsrichtlinie hinzu.
- Einer Rolle eine Berechtigungsrichtlinie zuordnen (um kontoübergreifende Berechtigungen zu gewähren) — Delegation, z. B. wenn Sie kontenübergreifenden Zugriff gewähren möchten, beinhaltet die Einrichtung einer Vertrauensstellung zwischen dem Konto, dem die Ressource gehört (das vertrauenswürdige Konto), und dem Konto, das die Benutzer enthält, die auf die Ressource zugreifen müssen (dem vertrauenswürdigen Konto). Eine Berechtigungsrichtlinie erteilt dem Benutzer einer Rolle die erforderlichen Berechtigungen zum Ausführen der vorgesehenen Aufgaben auf der Ressource. Eine Vertrauensrichtlinie gibt an, welche vertrauenswürdigen Konten ihren Benutzern Berechtigungen zum Übernehmen der Rolle erteilen dürfen. Weitere Informationen finden Sie unter [IAM-Begriffe](#) und Konzepte.

Um kontoübergreifende Berechtigungen zu gewähren, fügen Sie einer IAM-Rolle eine identitätsbasierte Berechtigungsrichtlinie hinzu. Beispielsweise kann der Administrator in Konto A wie folgt eine Rolle erstellen, um einem anderen Amazon Web Services Services-Konto (z. B. Konto B) oder einem AWS Service kontoübergreifende Berechtigungen zu gewähren:

1. Der Administrator von Konto A erstellt eine IAM-Rolle und fügt ihr eine Berechtigungsrichtlinie an, die Berechtigungen für Ressourcen in Konto A erteilt.
2. Der Administrator von Konto A weist der Rolle eine Vertrauensrichtlinie zu, die Konto B als den Prinzipal identifiziert, der die Rolle übernehmen kann.
3. Der Administrator von Konto B kann dann die Berechtigungen zur Übernahme der Rolle an alle Benutzer in Konto B delegieren. Auf diese Weise können Benutzer in Konto B Ressourcen in Konto A erstellen oder darauf zugreifen. Wenn Sie eine AWS Dienstberechtigung zur Übernahme der Rolle erteilen möchten, kann der Principal in der Vertrauensrichtlinie auch ein AWS Dienstprinzipal sein. Weitere Informationen finden Sie unter Delegation in [IAM-Begriffen und Konzepten](#).

Weitere Informationen zum Delegieren von Berechtigungen mithilfe von IAM finden Sie unter [Zugriffsverwaltung](#) im IAM-Benutzerhandbuch.

Die folgende Beispielrichtlinie ermöglicht einem Benutzer das Erstellen eines Branch in einem Repository mit dem Namen *MyDemoRepo*:

```
{
  "Version": "2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "codecommit:CreateBranch"
      ],
      "Resource" : "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo"
    }
  ]
}
```

Um die Aufrufe und Ressourcen einzuschränken, auf die Benutzer in Ihrem Konto Zugriff haben, erstellen Sie spezifische IAM-Richtlinien und fügen Sie diese Richtlinien dann IAM-Benutzern hinzu. Weitere Informationen zum Erstellen von IAM-Rollen und Beispiele für IAM-Richtlinienerklärungen finden Sie unter. CodeCommit [Beispiele für vom Kunden verwaltete Identitätsrichtlinien](#)

Ressourcenbasierte Richtlinien

Einige Dienste, wie Amazon S3, unterstützen auch ressourcenbasierte Berechtigungsrichtlinien. Sie können beispielsweise eine ressourcenbasierte Richtlinie an einen S3-Bucket anhängen, um die Zugriffsberechtigungen für diesen Bucket zu verwalten. CodeCommit unterstützt keine ressourcenbasierten Richtlinien, aber Sie können Tags verwenden, um Ressourcen zu identifizieren, die Sie dann in IAM-Richtlinien verwenden können. Ein Beispiel für eine Tag-basierte Richtlinien finden Sie unter [Identitätsbasierte Richtlinien \(IAM-Richtlinien\)](#).

Eingrenzung des Ressourcenbereichs CodeCommit

In CodeCommit können Sie identitätsbasierte Richtlinien und Berechtigungen auf Ressourcen beschränken, wie unter beschrieben. [CodeCommit Ressourcen und Abläufe](#) Sie können jedoch nicht die `ListRepositories`- Berechtigung auf eine Ressource anwenden. Stattdessen müssen Sie sie auf alle Ressourcen anwenden (mit dem Platzhalter `*`). Andernfalls schlägt die Aktion fehl.

Alle anderen CodeCommit Berechtigungen können auf Ressourcen beschränkt werden.

Angaben der Richtlinienelemente: Ressourcen, Aktionen, Effekte und Prinzipale

Sie können Richtlinien erstellen, um Benutzern den Zugriff auf Ressourcen zu gewähren oder zu verweigern oder Benutzern zu erlauben oder zu verweigern, bestimmte Aktionen mit diesen Ressourcen durchzuführen. CodeCommit definiert eine Reihe von öffentlichen API-Vorgängen, die definieren, wie Benutzer mit dem Dienst arbeiten, unabhängig davon, ob dies über die CodeCommit Konsole SDKs, die AWS CLI, den oder durch deren direkten Aufruf erfolgt APIs. CodeCommit Definiert eine Reihe von Aktionen, die Sie in einer Richtlinie angeben können, um Berechtigungen für diese API-Operationen zu gewähren.

Für einige API-Operationen können Berechtigungen für mehrere Aktionen erforderlich sein. Weitere Informationen zu Ressourcen und API-Operationen finden Sie unter [CodeCommit Ressourcen und Abläufe](#) und [Referenz für CodeCommit-Berechtigungen](#).

Die Grundelemente einer Richtlinie sind:

- **Ressource** — Um die Ressource zu identifizieren, für die die Richtlinie gilt, verwenden Sie einen Amazon-Ressourcennamen (ARN). Weitere Informationen finden Sie unter [CodeCommit Ressourcen und Abläufe](#).
- **Aktion** — Um Ressourcenoperationen zu identifizieren, die Sie zulassen oder verweigern möchten, verwenden Sie Aktionsschlüsselwörter. Je nach Angabe `Effect` erlaubt oder verweigert die `codecommit:GetBranch` Berechtigung dem Benutzer beispielsweise die Ausführung des

GetBranch Vorgangs, mit dem Details zu einem Branch in einem CodeCommit Repository abgerufen werden.

- **Wirkung** — Sie geben den Effekt an, der eintritt, wenn der Benutzer die bestimmte Aktion anfordert, entweder zulassen oder verweigern. Wenn Sie den Zugriff auf eine Ressource nicht ausdrücklich gestatten („Allow“), wird er automatisch verweigert. Sie können den Zugriff auf eine Ressource auch explizit verweigern. So können Sie sicherstellen, dass Benutzer nicht darauf zugreifen können, auch wenn der Zugriff durch eine andere Richtlinie gestattet wird.
- **Prinzipal** — In identitätsbasierten Richtlinien (IAM-Richtlinien) CodeCommit unterstützt der einzige Richtlinienentyp den Benutzer, dem die Richtlinie zugeordnet ist, der implizite Prinzipal.

Weitere Informationen zur IAM-Richtliniensyntax finden Sie unter [IAM-Richtlinienreferenz im IAM-Benutzerhandbuch](#).

Eine Tabelle mit allen CodeCommit API-Aktionen und den Ressourcen, für die sie gelten, finden Sie unter [Referenz für CodeCommit-Berechtigungen](#)

Angeben von Bedingungen in einer Richtlinie

Wenn Sie Berechtigungen gewähren, verwenden Sie die Sprache der Zugriffsrichtlinie für IAM, um die Bedingungen anzugeben, unter denen eine Richtlinie wirksam werden soll. Beispielsweise kann festgelegt werden, dass eine Richtlinie erst ab einem bestimmten Datum gilt. Weitere Informationen zur Angabe von Bedingungen in einer Richtlinienensprache finden Sie unter [Bedingungen](#) und [Richtliniengrammatik](#) im IAM-Benutzerhandbuch.

Bedingungen werden mithilfe vordefinierter Bedingungsschlüssel formuliert. Für CodeCommit gibt es keine speziellen Bedingungsschlüssel. Es gibt jedoch Bedingungsschlüssel für AWS alle Bereiche, die Sie je nach Bedarf verwenden können. Eine vollständige Liste der AWS-weiten Schlüssel finden Sie unter [Verfügbare Schlüssel für Bedingungen](#) im IAM-Benutzerhandbuch.

Verwendung identitätsbasierter Richtlinien (IAM-Richtlinien) für CodeCommit

Die folgenden Beispiele für identitätsbasierte Richtlinien zeigen, wie ein Kontoadministrator Berechtigungsrichtlinien an IAM-Identitäten (Benutzer, Gruppen und Rollen) anhängen kann, um Berechtigungen zur Ausführung von Vorgängen mit Ressourcen zu erteilen. CodeCommit

Important

Wir empfehlen Ihnen, zunächst die einführenden Themen zu lesen, in denen die grundlegenden Konzepte und Optionen für die Verwaltung des Zugriffs auf Ihre Ressourcen

erläutert werden. CodeCommit Weitere Informationen finden Sie unter [Überblick über die Verwaltung von Zugriffsberechtigungen für Ihre CodeCommit Ressourcen](#).

Themen

- [Erforderliche Berechtigungen für die Verwendung der CodeCommit-Konsole](#)
- [Anzeigen von Ressourcen in der Konsole](#)
- [AWS verwaltete Richtlinien für CodeCommit](#)
- [Beispiele für vom Kunden verwaltete Richtlinien](#)

Nachstehend finden Sie ein Beispiel für eine identitätsbasierte Berechtigungsrichtlinie:

```
{
  "Version": "2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "codecommit:BatchGetRepositories"
      ],
      "Resource" : [
        "arn:aws:codecommit:us-east-2:111111111111:MyDestinationRepo",
        "arn:aws:codecommit:us-east-2:111111111111:MyDemo*"
      ]
    }
  ]
}
```

Diese Richtlinie enthält eine Erklärung, die es einem Benutzer ermöglicht, Informationen über das angegebene CodeCommit Repository `MyDestinationRepo` und alle CodeCommit Repositories, die mit dem Namen beginnen, `MyDemo` in der **us-east-2** Region abzurufen.

Erforderliche Berechtigungen für die Verwendung der CodeCommit-Konsole

Informationen zu den erforderlichen Berechtigungen für jeden CodeCommit API-Vorgang sowie weitere Informationen zu CodeCommit Vorgängen finden Sie unter [Referenz für CodeCommit-Berechtigungen](#).

Damit Benutzer die CodeCommit Konsole verwenden können, muss ihnen der Administrator Berechtigungen für CodeCommit Aktionen gewähren. Sie könnten beispielsweise die [AWSCodeCommitPowerUser](#) verwaltete Richtlinie oder eine entsprechende Richtlinie einem Benutzer oder einer Gruppe zuordnen.

Zusätzlich zu den Berechtigungen, die Benutzern durch identitätsbasierte Richtlinien gewährt werden, sind auch Berechtigungen für AWS Key Management Service (AWS KMS) -Aktionen CodeCommit erforderlich. Ein IAM-Benutzer benötigt keine ausdrücklichen Allow Berechtigungen für diese Aktionen, dem Benutzer dürfen jedoch keine Richtlinien angehängt sein, die die folgenden Berechtigungen festlegen auf: Deny

```
"kms:Encrypt",  
"kms:Decrypt",  
"kms:ReEncrypt",  
"kms:GenerateDataKey",  
"kms:GenerateDataKeyWithoutPlaintext",  
"kms:DescribeKey"
```

Weitere Informationen zur Verschlüsselung und finden Sie CodeCommit unter [AWS KMS und Verschlüsselung](#).

Anzeigen von Ressourcen in der Konsole

Die CodeCommit Konsole benötigt die ListRepositories Erlaubnis, eine Liste der Repositorys für Ihr Amazon Web Services Services-Konto in dem Land anzuzeigen, in AWS-Region dem Sie angemeldet sind. Die Konsole umfasst auch eine Funktion Go to resource (Zu Ressource wechseln), um schnell ohne Berücksichtigung der Groß- und Kleinschreibung nach Ressourcen zu suchen. Diese Suche wird in Ihrem Amazon Web Services Services-Konto durchgeführt, in AWS-Region dem Sie angemeldet sind. Die folgenden Ressourcen werden für die folgenden Services angezeigt:

- AWS CodeBuild: Build-Projekte
- AWS CodeCommit: Repositorys
- AWS CodeDeploy: Anwendungen
- AWS CodePipeline: Pipelines

Für diese Suche unter den Ressourcen in allen Services müssen Sie über die folgenden Berechtigungen verfügen:

- CodeBuild: ListProjects

- CodeCommit: `ListRepositories`
- CodeDeploy: `ListApplications`
- CodePipeline: `ListPipelines`

Es werden keine Ergebnisse für die Ressourcen eines Service zurückgegeben, wenn Sie nicht über Berechtigungen für diesen Service verfügen. Auch wenn Sie über Berechtigungen zum Anzeigen von Ressourcen verfügen, werden bestimmte Ressourcen nicht zurückgegeben, wenn die Anzeige dieser Ressourcen ausdrücklich verweigert wird (Deny).

AWS verwaltete Richtlinien für CodeCommit

Um Benutzern, Gruppen und Rollen Berechtigungen hinzuzufügen, ist es einfacher, AWS verwaltete Richtlinien zu verwenden, als Richtlinien selbst zu schreiben. Es erfordert Zeit und Fachwissen, um [von Kunden verwaltete IAM-Richtlinien zu erstellen](#), die Ihrem Team nur die benötigten Berechtigungen bieten. Um schnell loszulegen, können Sie unsere AWS verwalteten Richtlinien verwenden. Diese Richtlinien decken allgemeine Anwendungsfälle ab und sind in Ihrem AWS-Konto verfügbar. Weitere Informationen zu AWS verwalteten Richtlinien finden Sie im IAM-Benutzerhandbuch unter [AWS Verwaltete Richtlinien](#).

AWS Dienste verwalten und aktualisieren AWS verwaltete Richtlinien. Sie können die Berechtigungen in AWS verwalteten Richtlinien nicht ändern. Services fügen einer von AWS verwalteten Richtlinien gelegentlich zusätzliche Berechtigungen hinzu, um neue Features zu unterstützen. Diese Art von Update betrifft alle Identitäten (Benutzer, Gruppen und Rollen), an welche die Richtlinie angehängt ist. Services aktualisieren eine von AWS verwaltete Richtlinie am ehesten, ein neues Feature gestartet wird oder neue Vorgänge verfügbar werden. Dienste entfernen keine Berechtigungen aus einer AWS verwalteten Richtlinie, sodass durch Richtlinienaktualisierungen Ihre bestehenden Berechtigungen nicht beeinträchtigt werden.

AWS Unterstützt außerdem verwaltete Richtlinien für Jobfunktionen, die sich über mehrere Dienste erstrecken. Die `ReadOnlyAccess` AWS verwaltete Richtlinie bietet beispielsweise schreibgeschützten Zugriff auf alle AWS Dienste und Ressourcen. Wenn ein Dienst eine neue Funktion startet, werden nur Leseberechtigungen für neue Operationen und Ressourcen AWS hinzugefügt. Eine Liste und Beschreibungen der Richtlinien für Auftragsfunktionen finden Sie in [Verwaltete AWS -Richtlinien für Auftragsfunktionen](#) im IAM-Leitfaden.

AWS adressiert viele gängige Anwendungsfälle durch die Bereitstellung eigenständiger IAM-Richtlinien, die von erstellt und verwaltet werden. AWS Diese AWS verwalteten Richtlinien gewähren die erforderlichen Berechtigungen für allgemeine Anwendungsfälle. Die verwalteten Richtlinien für

gewähren CodeCommit auch Berechtigungen zur Ausführung von Vorgängen in anderen Diensten wie IAM, Amazon SNS und Amazon CloudWatch Events, je nach den Zuständigkeiten der Benutzer, denen die betreffende Richtlinie erteilt wurde. Bei der `AWSCodeCommitFullAccess` Richtlinie handelt es sich beispielsweise um eine Benutzerrichtlinie auf Administrationsebene, die es Benutzern mit dieser Richtlinie ermöglicht, CloudWatch Ereignisregeln für Repositorys (Regeln, deren Namen ein Präfix `habencodecommit`) und Amazon SNS SNS-Themen für Benachrichtigungen über repository-bezogene Ereignisse (Themen, deren Namen ein Präfix haben) zu erstellen und zu verwalten sowie Repositorys in zu verwalten. `codecommit` CodeCommit

Die folgenden AWS verwalteten Richtlinien, die Sie Benutzern in Ihrem Konto zuordnen können, sind spezifisch für. CodeCommit

Themen

- [AWS verwaltete Richtlinie: `AWSCodeCommitFullAccess`](#)
- [AWS verwaltete Richtlinie: `AWSCodeCommitPowerUser`](#)
- [AWS verwaltete Richtlinie: `AWSCodeCommitReadOnly`](#)
- [CodeCommit verwaltete Richtlinien und Benachrichtigungen](#)
- [AWS CodeCommit verwaltete Richtlinien und Amazon CodeGuru Reviewer](#)
- [CodeCommit Aktualisierungen der AWS verwalteten Richtlinien](#)

AWS verwaltete Richtlinie: `AWSCodeCommitFullAccess`

Sie können die `AWSCodeCommitFullAccess`-Richtlinie an Ihre IAM-Identitäten anfügen. Diese Richtlinie gewährt vollen Zugriff auf CodeCommit. Wenden Sie diese Richtlinie nur auf Benutzer auf Administratorebene an, denen Sie die volle Kontrolle über CodeCommit Repositorys und zugehörige Ressourcen in Ihrem Amazon Web Services Services-Konto gewähren möchten, einschließlich der Möglichkeit, Repositorys zu löschen.

Die `AWSCodeCommitFullAccess` Richtlinie enthält die folgende Grundsatzerklärung:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codecommit:*"
      ],
    }
  ],
}
```

```
    "Resource": "*"
  },
  {
    "Sid": "CloudWatchEventsCodeCommitRulesAccess",
    "Effect": "Allow",
    "Action": [
      "events:DeleteRule",
      "events:DescribeRule",
      "events:DisableRule",
      "events:EnableRule",
      "events:PutRule",
      "events:PutTargets",
      "events:RemoveTargets",
      "events:ListTargetsByRule"
    ],
    "Resource": "arn:aws:events:*:*:rule/codecommit*"
  },
  {
    "Sid": "SNSTopicAndSubscriptionAccess",
    "Effect": "Allow",
    "Action": [
      "sns:CreateTopic",
      "sns:DeleteTopic",
      "sns:Subscribe",
      "sns:Unsubscribe",
      "sns:SetTopicAttributes"
    ],
    "Resource": "arn:aws:sns:*:*:codecommit*"
  },
  {
    "Sid": "SNSTopicAndSubscriptionReadAccess",
    "Effect": "Allow",
    "Action": [
      "sns:ListTopics",
      "sns:ListSubscriptionsByTopic",
      "sns:GetTopicAttributes"
    ],
    "Resource": "*"
  },
  {
    "Sid": "LambdaReadOnlyListAccess",
    "Effect": "Allow",
    "Action": [
      "lambda:ListFunctions"
```

```
    ],
    "Resource": "*"
  },
  {
    "Sid": "IAMReadOnlyListAccess",
    "Effect": "Allow",
    "Action": [
      "iam:ListUsers"
    ],
    "Resource": "*"
  },
  {
    "Sid": "IAMReadOnlyConsoleAccess",
    "Effect": "Allow",
    "Action": [
      "iam:ListAccessKeys",
      "iam:ListSSHPublicKeys",
      "iam:ListServiceSpecificCredentials"
    ],
    "Resource": "arn:aws:iam::*:user/${aws:username}"
  },
  {
    "Sid": "IAMUserSSHKeys",
    "Effect": "Allow",
    "Action": [
      "iam:DeleteSSHPublicKey",
      "iam:GetSSHPublicKey",
      "iam:ListSSHPublicKeys",
      "iam:UpdateSSHPublicKey",
      "iam:UploadSSHPublicKey"
    ],
    "Resource": "arn:aws:iam::*:user/${aws:username}"
  },
  {
    "Sid": "IAMSelfManageServiceSpecificCredentials",
    "Effect": "Allow",
    "Action": [
      "iam:CreateServiceSpecificCredential",
      "iam:UpdateServiceSpecificCredential",
      "iam:DeleteServiceSpecificCredential",
      "iam:ResetServiceSpecificCredential"
    ],
    "Resource": "arn:aws:iam::*:user/${aws:username}"
  },
  ],
```

```
{
  "Sid": "CodeStarNotificationsReadWriteAccess",
  "Effect": "Allow",
  "Action": [
    "codestar-notifications:CreateNotificationRule",
    "codestar-notifications:DescribeNotificationRule",
    "codestar-notifications:UpdateNotificationRule",
    "codestar-notifications>DeleteNotificationRule",
    "codestar-notifications:Subscribe",
    "codestar-notifications:Unsubscribe"
  ],
  "Resource": "*",
  "Condition": {
    "StringLike": {
      "codestar-notifications:NotificationsForResource": "arn:aws:codecommit:*"
    }
  }
},
{
  "Sid": "CodeStarNotificationsListAccess",
  "Effect": "Allow",
  "Action": [
    "codestar-notifications:ListNotificationRules",
    "codestar-notifications:ListTargets",
    "codestar-notifications:ListTagsForResource",
    "codestar-notifications:ListEventTypes"
  ],
  "Resource": "*"
},
{
  "Sid": "CodeStarNotificationsSNSTopicCreateAccess",
  "Effect": "Allow",
  "Action": [
    "sns:CreateTopic",
    "sns:SetTopicAttributes"
  ],
  "Resource": "arn:aws:sns:*:*:codestar-notifications*"
},
{
  "Sid": "AmazonCodeGuruReviewerFullAccess",
  "Effect": "Allow",
  "Action": [
    "codeguru-reviewer:AssociateRepository",
    "codeguru-reviewer:DescribeRepositoryAssociation",
```

```

        "codeguru-reviewer:ListRepositoryAssociations",
        "codeguru-reviewer:DisassociateRepository",
        "codeguru-reviewer:DescribeCodeReview",
        "codeguru-reviewer:ListCodeReviews"
    ],
    "Resource": "*"
},
{
    "Sid": "AmazonCodeGuruReviewerSLRCreation",
    "Action": "iam:CreateServiceLinkedRole",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::*:role/aws-service-role/codeguru-
reviewer.amazonaws.com/AWSServiceRoleForAmazonCodeGuruReviewer",
    "Condition": {
        "StringLike": {
            "iam:AWSServiceName": "codeguru-reviewer.amazonaws.com"
        }
    }
},
{
    "Sid": "CloudWatchEventsManagedRules",
    "Effect": "Allow",
    "Action": [
        "events:PutRule",
        "events:PutTargets",
        "events>DeleteRule",
        "events:RemoveTargets"
    ],
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "events:ManagedBy": "codeguru-reviewer.amazonaws.com"
        }
    }
},
{
    "Sid": "CodeStarNotificationsChatbotAccess",
    "Effect": "Allow",
    "Action": [
        "chatbot:DescribeSlackChannelConfigurations",
        "chatbot:ListMicrosoftTeamsChannelConfigurations"
    ],
    "Resource": "*"
},

```

```

{
  "Sid": "CodeStarConnectionsReadOnlyAccess",
  "Effect": "Allow",
  "Action": [
    "codestar-connections:ListConnections",
    "codestar-connections:GetConnection"
  ],
  "Resource": "arn:aws:codestar-connections:*:*:connection/*"
}
]
}

```

AWS verwaltete Richtlinie: AWSCodeCommitPowerUser

Sie können die AWSCodeCommitPowerUser-Richtlinie an Ihre IAM-Identitäten anfügen. Diese Richtlinie ermöglicht Benutzern den Zugriff auf alle Funktionen CodeCommit und Ressourcen im Zusammenhang mit Repositorys, mit der Ausnahme, dass sie ihnen nicht erlaubt, Repositorys zu löschen oder CodeCommit repository-bezogene Ressourcen in anderen Diensten wie Amazon Events zu erstellen oder zu löschen. AWS CloudWatch Wir empfehlen, dass diese Richtlinie auf die meisten Benutzer anzuwenden.

Die AWSCodeCommitPowerUser Richtlinie enthält die folgende Grundsatzerklärung:

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codecommit:AssociateApprovalRuleTemplateWithRepository",
        "codecommit:BatchAssociateApprovalRuleTemplateWithRepositories",
        "codecommit:BatchDisassociateApprovalRuleTemplateFromRepositories",
        "codecommit:BatchGet*",
        "codecommit:BatchDescribe*",
        "codecommit:Create*",
        "codecommit>DeleteBranch",
        "codecommit>DeleteFile",
        "codecommit:Describe*",
        "codecommit:DisassociateApprovalRuleTemplateFromRepository",
        "codecommit:EvaluatePullRequestApprovalRules",
        "codecommit:Get*",
        "codecommit:List*",
        "codecommit:Merge*",

```

```
        "codecommit:OverridePullRequestApprovalRules",
        "codecommit:Put*",
        "codecommit:Post*",
        "codecommit:TagResource",
        "codecommit:Test*",
        "codecommit:UntagResource",
        "codecommit:Update*",
        "codecommit:GitPull",
        "codecommit:GitPush"
    ],
    "Resource": "*"
},
{
    "Sid": "CloudWatchEventsCodeCommitRulesAccess",
    "Effect": "Allow",
    "Action": [
        "events:DeleteRule",
        "events:DescribeRule",
        "events:DisableRule",
        "events:EnableRule",
        "events:PutRule",
        "events:PutTargets",
        "events:RemoveTargets",
        "events:ListTargetsByRule"
    ],
    "Resource": "arn:aws:events:*:*:rule/codecommit*"
},
{
    "Sid": "SNSTopicAndSubscriptionAccess",
    "Effect": "Allow",
    "Action": [
        "sns:Subscribe",
        "sns:Unsubscribe"
    ],
    "Resource": "arn:aws:sns:*:*:codecommit*"
},
{
    "Sid": "SNSTopicAndSubscriptionReadAccess",
    "Effect": "Allow",
    "Action": [
        "sns:ListTopics",
        "sns:ListSubscriptionsByTopic",
        "sns:GetTopicAttributes"
    ],
    ],
```

```
    "Resource": "*"
  },
  {
    "Sid": "LambdaReadOnlyListAccess",
    "Effect": "Allow",
    "Action": [
      "lambda:ListFunctions"
    ],
    "Resource": "*"
  },
  {
    "Sid": "IAMReadOnlyListAccess",
    "Effect": "Allow",
    "Action": [
      "iam:ListUsers"
    ],
    "Resource": "*"
  },
  {
    "Sid": "IAMReadOnlyConsoleAccess",
    "Effect": "Allow",
    "Action": [
      "iam:ListAccessKeys",
      "iam:ListSSHPublicKeys",
      "iam:ListServiceSpecificCredentials"
    ],
    "Resource": "arn:aws:iam::*:user/${aws:username}"
  },
  {
    "Sid": "IAMUserSSHKeys",
    "Effect": "Allow",
    "Action": [
      "iam:DeleteSSHPublicKey",
      "iam:GetSSHPublicKey",
      "iam:ListSSHPublicKeys",
      "iam:UpdateSSHPublicKey",
      "iam:UploadSSHPublicKey"
    ],
    "Resource": "arn:aws:iam::*:user/${aws:username}"
  },
  {
    "Sid": "IAMSelfManageServiceSpecificCredentials",
    "Effect": "Allow",
    "Action": [
```



```

        "iam:CreateServiceSpecificCredential",
        "iam:UpdateServiceSpecificCredential",
        "iam>DeleteServiceSpecificCredential",
        "iam:ResetServiceSpecificCredential"
    ],
    "Resource": "arn:aws:iam::*:user/${aws:username}"
},
{
    "Sid": "CodeStarNotificationsReadWriteAccess",
    "Effect": "Allow",
    "Action": [
        "codestar-notifications:CreateNotificationRule",
        "codestar-notifications:DescribeNotificationRule",
        "codestar-notifications:UpdateNotificationRule",
        "codestar-notifications:Subscribe",
        "codestar-notifications:Unsubscribe"
    ],
    "Resource": "*",
    "Condition": {
        "StringLike": {
            "codestar-notifications:NotificationsForResource": "arn:aws:codecommit:*"
        }
    }
},
{
    "Sid": "CodeStarNotificationsListAccess",
    "Effect": "Allow",
    "Action": [
        "codestar-notifications:ListNotificationRules",
        "codestar-notifications:ListTargets",
        "codestar-notifications:ListTagsForResource",
        "codestar-notifications:ListEventTypes"
    ],
    "Resource": "*"
},
{
    "Sid": "AmazonCodeGuruReviewerFullAccess",
    "Effect": "Allow",
    "Action": [
        "codeguru-reviewer:AssociateRepository",
        "codeguru-reviewer:DescribeRepositoryAssociation",
        "codeguru-reviewer:ListRepositoryAssociations",
        "codeguru-reviewer:DisassociateRepository",
        "codeguru-reviewer:DescribeCodeReview",

```

```

        "codeguru-reviewer:ListCodeReviews"
    ],
    "Resource": "*"
},
{
    "Sid": "AmazonCodeGuruReviewerSLRCreation",
    "Action": "iam:CreateServiceLinkedRole",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::*:role/aws-service-role/codeguru-
reviewer.amazonaws.com/AWSServiceRoleForAmazonCodeGuruReviewer",
    "Condition": {
        "StringLike": {
            "iam:AWSServiceName": "codeguru-reviewer.amazonaws.com"
        }
    }
},
{
    "Sid": "CloudWatchEventsManagedRules",
    "Effect": "Allow",
    "Action": [
        "events:PutRule",
        "events:PutTargets",
        "events>DeleteRule",
        "events:RemoveTargets"
    ],
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "events:ManagedBy": "codeguru-reviewer.amazonaws.com"
        }
    }
},
{
    "Sid": "CodeStarNotificationsChatbotAccess",
    "Effect": "Allow",
    "Action": [
        "chatbot:DescribeSlackChannelConfigurations",
        "chatbot:ListMicrosoftTeamsChannelConfigurations"
    ],
    "Resource": "*"
},
{
    "Sid": "CodeStarConnectionsReadOnlyAccess",
    "Effect": "Allow",

```

```

    "Action": [
        "codestar-connections:ListConnections",
        "codestar-connections:GetConnection"
    ],
    "Resource": "arn:aws:codestar-connections:*:*:connection/*"
}
]
}

```

AWS verwaltete Richtlinie: AWSCode CommitReadOnly

Sie können die `AWSCodeCommitReadOnly`-Richtlinie an Ihre IAM-Identitäten anfügen. Diese Richtlinie gewährt schreibgeschützten Zugriff auf CodeCommit und Repository-bezogene Ressourcen in anderen AWS Diensten sowie die Möglichkeit, eigene CodeCommit zugehörige Ressourcen zu erstellen und zu verwalten (z. B. Git-Anmeldeinformationen und SSH-Schlüssel, die ihre IAM-Benutzer beim Zugriff auf Repositories verwenden können). Wenden Sie diese Richtlinie auf Benutzer an, denen Sie die Möglichkeit geben möchten, den Inhalt eines Repositories zu lesen, ohne jedoch dessen Inhalt zu ändern.

Die `AWSCodeCommitReadOnly` Richtlinie enthält die folgende Grundsatzerklärung:

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codecommit:BatchGet*",
        "codecommit:BatchDescribe*",
        "codecommit:Describe*",
        "codecommit:EvaluatePullRequestApprovalRules",
        "codecommit:Get*",
        "codecommit:List*",
        "codecommit:GitPull"
      ],
      "Resource": "*"
    },
    {
      "Sid": "CloudWatchEventsCodeCommitRulesReadOnlyAccess",
      "Effect": "Allow",
      "Action": [
        "events:DescribeRule",

```

```
        "events:ListTargetsByRule"
    ],
    "Resource": "arn:aws:events:*:*:rule/codecommit*"
},
{
    "Sid": "SNSSubscriptionAccess",
    "Effect": "Allow",
    "Action": [
        "sns:ListTopics",
        "sns:ListSubscriptionsByTopic",
        "sns:GetTopicAttributes"
    ],
    "Resource": "*"
},
{
    "Sid": "LambdaReadOnlyListAccess",
    "Effect": "Allow",
    "Action": [
        "lambda:ListFunctions"
    ],
    "Resource": "*"
},
{
    "Sid": "IAMReadOnlyListAccess",
    "Effect": "Allow",
    "Action": [
        "iam:ListUsers"
    ],
    "Resource": "*"
},
{
    "Sid": "IAMReadOnlyConsoleAccess",
    "Effect": "Allow",
    "Action": [
        "iam:ListAccessKeys",
        "iam:ListSSHPublicKeys",
        "iam:ListServiceSpecificCredentials",
        "iam:GetSSHPublicKey"
    ],
    "Resource": "arn:aws:iam:*:*:user/${aws:username}"
},
{
    "Sid": "CodeStarNotificationsReadOnlyAccess",
    "Effect": "Allow",
```

```

        "Action": [
            "codestar-notifications:DescribeNotificationRule"
        ],
        "Resource": "*",
        "Condition": {
            "StringLike": {
                "codestar-
notifications:NotificationsForResource": "arn:aws:codecommit:*"
            }
        }
    },
    {
        "Sid": "CodeStarNotificationsListAccess",
        "Effect": "Allow",
        "Action": [
            "codestar-notifications:ListNotificationRules",
            "codestar-notifications:ListEventTypes",
            "codestar-notifications:ListTargets"
        ],
        "Resource": "*"
    },
    {
        "Sid": "AmazonCodeGuruReviewerReadOnlyAccess",
        "Effect": "Allow",
        "Action": [
            "codeguru-reviewer:DescribeRepositoryAssociation",
            "codeguru-reviewer:ListRepositoryAssociations",
            "codeguru-reviewer:DescribeCodeReview",
            "codeguru-reviewer:ListCodeReviews"
        ],
        "Resource": "*"
    },
    {
        "Sid": "CodeStarConnectionsReadOnlyAccess",
        "Effect": "Allow",
        "Action": [
            "codestar-connections:ListConnections",
            "codestar-connections:GetConnection"
        ],
        "Resource": "arn:aws:codestar-connections:*:*:connection/*"
    }
]
}

```

CodeCommit verwaltete Richtlinien und Benachrichtigungen

AWS CodeCommit unterstützt Benachrichtigungen, mit denen Benutzer über wichtige Änderungen an Repositorys informiert werden können. Zu den verwalteten Richtlinien CodeCommit gehören auch Richtlinienenerklärungen für die Benachrichtigungsfunktion. Weitere Informationen finden Sie unter [Was sind Benachrichtigungen?](#).

Berechtigungen in Zusammenhang mit Benachrichtigungen in verwalteten Vollzugriffsrichtlinien

Die von `AWSCodeCommitFullAccess` verwaltete Richtlinie enthält die folgenden Anweisungen, um den vollständigen Zugriff auf Benachrichtigungen zu ermöglichen. Benutzer, für die diese verwaltete Richtlinie gilt, können auch Amazon SNS SNS-Themen für Benachrichtigungen erstellen und verwalten, Benutzer für Themen abonnieren und abbestellen, Themen auflisten, die als Ziele für Benachrichtigungsregeln ausgewählt werden sollen, und Amazon Q Developer in Chat-Anwendungsclients auflisten, die für Slack konfiguriert sind.

```
{
  "Sid": "CodeStarNotificationsReadWriteAccess",
  "Effect": "Allow",
  "Action": [
    "codestar-notifications:CreateNotificationRule",
    "codestar-notifications:DescribeNotificationRule",
    "codestar-notifications:UpdateNotificationRule",
    "codestar-notifications>DeleteNotificationRule",
    "codestar-notifications:Subscribe",
    "codestar-notifications:Unsubscribe"
  ],
  "Resource": "*",
  "Condition": {
    "StringLike": {"codestar-notifications:NotificationsForResource" :
"arn:aws:codecommit:*"}
  }
},
{
  "Sid": "CodeStarNotificationsListAccess",
  "Effect": "Allow",
  "Action": [
    "codestar-notifications:ListNotificationRules",
    "codestar-notifications:ListTargets",
    "codestar-notifications:ListTagsForResource",
    "codestar-notifications:ListEventTypes"
  ],
```

```

    "Resource": "*"
  },
  {
    "Sid": "CodeStarNotificationsSNSTopicCreateAccess",
    "Effect": "Allow",
    "Action": [
      "sns:CreateTopic",
      "sns:SetTopicAttributes"
    ],
    "Resource": "arn:aws:sns:*:*:codestar-notifications*"
  },
  {
    "Sid": "CodeStarNotificationsChatbotAccess",
    "Effect": "Allow",
    "Action": [
      "chatbot:DescribeSlackChannelConfigurations",
      "chatbot:ListMicrosoftTeamsChannelConfigurations"
    ],
    "Resource": "*"
  }
}

```

Berechtigungen in Zusammenhang mit Benachrichtigungen in schreibgeschützten verwalteten Richtlinien

Die verwaltete Richtlinie `AWSCodeCommitReadOnlyAccess` enthält die folgenden Anweisungen, um schreibgeschützten Zugriff auf Benachrichtigungen zu ermöglichen. Benutzer mit dieser verwalteten Richtlinie können Benachrichtigungen für Ressourcen anzeigen, sie können sie jedoch nicht erstellen, verwalten oder abonnieren.

```

{
  "Sid": "CodeStarNotificationsPowerUserAccess",
  "Effect": "Allow",
  "Action": [
    "codestar-notifications:DescribeNotificationRule"
  ],
  "Resource": "*",
  "Condition" : {
    "StringLike" : {"codestar-notifications:NotificationsForResource" :
"arn:aws:codecommit:*"}
  }
},
{
  "Sid": "CodeStarNotificationsListAccess",

```

```

    "Effect": "Allow",
    "Action": [
        "codestar-notifications:ListNotificationRules",
        "codestar-notifications:ListEventTypes",
        "codestar-notifications:ListTargets"
    ],
    "Resource": "*"
}

```

Berechtigungen in Zusammenhang mit Benachrichtigungen in anderen verwalteten Richtlinien

Die verwaltete Richtlinie `AWSCodeCommitPowerUser` enthält die folgenden Anweisungen, mit denen Sie Benutzern erlauben können, Benachrichtigungen zu erstellen, zu bearbeiten und zu abonnieren. Benutzer können Benachrichtigungsregeln nicht löschen und auch keine Tags für Ressourcen verwalten.

```

{
    "Sid": "CodeStarNotificationsReadWriteAccess",
    "Effect": "Allow",
    "Action": [
        "codestar-notifications:CreateNotificationRule",
        "codestar-notifications:DescribeNotificationRule",
        "codestar-notifications:UpdateNotificationRule",
        "codestar-notifications>DeleteNotificationRule",
        "codestar-notifications:Subscribe",
        "codestar-notifications:Unsubscribe"
    ],
    "Resource": "*",
    "Condition" : {
        "StringLike" : {"codestar-notifications:NotificationsForResource" :
"arn:aws:codecommit*"}
    }
},
{
    "Sid": "CodeStarNotificationsListAccess",
    "Effect": "Allow",
    "Action": [
        "codestar-notifications:ListNotificationRules",
        "codestar-notifications:ListTargets",
        "codestar-notifications:ListTagsForResource",
        "codestar-notifications:ListEventTypes"
    ],
    "Resource": "*"
}

```



```
    },
    {
      "Sid": "SNSTopicListAccess",
      "Effect": "Allow",
      "Action": [
        "sns:ListTopics"
      ],
      "Resource": "*"
    },
    {
      "Sid": "CodeStarNotificationsChatbotAccess",
      "Effect": "Allow",
      "Action": [
        "chatbot:DescribeSlackChannelConfigurations",
        "chatbot:ListMicrosoftTeamsChannelConfigurations"
      ],
      "Resource": "*"
    }
  ]
}
```

Weitere Informationen zu IAM und Benachrichtigungen finden Sie unter [Identity and Access Management für AWS CodeStar Benachrichtigungen](#).

AWS CodeCommit verwaltete Richtlinien und Amazon CodeGuru Reviewer

CodeCommit unterstützt Amazon CodeGuru Reviewer, einen automatisierten Code-Review-Service, der Programmanalyse und maschinelles Lernen nutzt, um häufig auftretende Probleme zu erkennen und Korrekturen in Ihrem Java- oder Python-Code zu empfehlen. Zu den verwalteten Richtlinien CodeCommit gehören auch Richtlinienenerklärungen für die CodeGuru Reviewer-Funktionalität. Weitere Informationen finden Sie unter [Was ist Amazon CodeGuru Reviewer](#).

Berechtigungen im Zusammenhang mit dem CodeGuru Prüfer in AWSCodeCommitFullAccess

Die AWSCodeCommitFullAccess verwaltete Richtlinie enthält die folgenden Anweisungen, mit denen CodeGuru Prüfer CodeCommit Repositorys zugeordnet oder getrennt werden können. Benutzer, auf die diese verwaltete Richtlinie angewendet wurde, können auch den Zuordnungsstatus zwischen CodeCommit Repositorys und CodeGuru Reviewer sowie den Status von Review-Jobs für Pull Requests einsehen.

```
{
  "Sid": "AmazonCodeGuruReviewerFullAccess",
  "Effect": "Allow",
```

```

    "Action": [
        "codeguru-reviewer:AssociateRepository",
        "codeguru-reviewer:DescribeRepositoryAssociation",
        "codeguru-reviewer:ListRepositoryAssociations",
        "codeguru-reviewer:DisassociateRepository",
        "codeguru-reviewer:DescribeCodeReview",
        "codeguru-reviewer:ListCodeReviews"
    ],
    "Resource": "*"
},
{
    "Sid": "AmazonCodeGuruReviewerSLRCreation",
    "Action": "iam:CreateServiceLinkedRole",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::*:role/aws-service-role/codeguru-
reviewer.amazonaws.com/AWSServiceRoleForAmazonCodeGuruReviewer",
    "Condition": {
        "StringLike": {
            "iam:AWSServiceName": "codeguru-reviewer.amazonaws.com"
        }
    }
},
{
    "Sid": "CloudWatchEventsManagedRules",
    "Effect": "Allow",
    "Action": [
        "events:PutRule",
        "events:PutTargets",
        "events>DeleteRule",
        "events:RemoveTargets"
    ],
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "events:ManagedBy": "codeguru-reviewer.amazonaws.com"
        }
    }
}
}

```

Berechtigungen im Zusammenhang mit dem CodeGuru Prüfer in AWSCode CommitPowerUser

Die AWSCodeCommitPowerUser verwaltete Richtlinie umfasst die folgenden Anweisungen, die es Benutzern ermöglichen, Repositories dem CodeGuru Prüfer zuzuordnen und die Verknüpfung

aufzuheben, den Zuordnungsstatus einzusehen und den Status von Review-Jobs für Pull Requests einzusehen.

```
{
  "Sid": "AmazonCodeGuruReviewerFullAccess",
  "Effect": "Allow",
  "Action": [
    "codeguru-reviewer:AssociateRepository",
    "codeguru-reviewer:DescribeRepositoryAssociation",
    "codeguru-reviewer:ListRepositoryAssociations",
    "codeguru-reviewer:DisassociateRepository",
    "codeguru-reviewer:DescribeCodeReview",
    "codeguru-reviewer:ListCodeReviews"
  ],
  "Resource": "*"
},
{
  "Sid": "AmazonCodeGuruReviewerSLRCreation",
  "Action": "iam:CreateServiceLinkedRole",
  "Effect": "Allow",
  "Resource": "arn:aws:iam::*:role/aws-service-role/codeguru-reviewer.amazonaws.com/AWSServiceRoleForAmazonCodeGuruReviewer",
  "Condition": {
    "StringLike": {
      "iam:AWSServiceName": "codeguru-reviewer.amazonaws.com"
    }
  }
},
{
  "Sid": "CloudWatchEventsManagedRules",
  "Effect": "Allow",
  "Action": [
    "events:PutRule",
    "events:PutTargets",
    "events>DeleteRule",
    "events:RemoveTargets"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "events:ManagedBy": "codeguru-reviewer.amazonaws.com"
    }
  }
}
```

```
}
```

Berechtigungen im Zusammenhang mit Reviewer in CodeGuru AWSCode CommitReadOnly

Die AWSCodeCommitReadOnlyAccess verwaltete Richtlinie umfasst die folgenden Anweisungen, um nur Lesezugriff auf den Zuordnungsstatus von CodeGuru Prüfern zu gewähren und den Status von Überprüfungsaufträgen für Pull-Requests anzuzeigen. Benutzer, denen diese verwaltete Richtlinie zugewiesen wurde, können Repositorys nicht zuordnen und deren Zuordnung nicht aufheben.

```
{
  "Sid": "AmazonCodeGuruReviewerReadOnlyAccess",
  "Effect": "Allow",
  "Action": [
    "codeguru-reviewer:DescribeRepositoryAssociation",
    "codeguru-reviewer:ListRepositoryAssociations",
    "codeguru-reviewer:DescribeCodeReview",
    "codeguru-reviewer:ListCodeReviews"
  ],
  "Resource": "*"
}
```

Rolle im Zusammenhang mit dem Service von Amazon CodeGuru Reviewer

Wenn Sie CodeGuru Reviewer ein Repository zuordnen, wird eine serviceverknüpfte Rolle erstellt, sodass CodeGuru Reviewer Probleme erkennen und Korrekturen für Java- oder Python-Code in Pull-Requests empfehlen kann. Die Service-verknüpfte Rolle wird AWSServiceRoleForAmazonCodeGuruReviewer benannt. Weitere Informationen finden Sie unter [Verwenden von serviceverknüpften Rollen für Amazon CodeGuru Reviewer](#).

Weitere Informationen finden Sie unter [AWS -verwaltete Richtlinien](#) im IAM-Benutzerhandbuch.

CodeCommit Aktualisierungen der AWS verwalteten Richtlinien

Hier finden Sie Informationen zu Aktualisierungen AWS verwalteter Richtlinien, die CodeCommit seit Beginn der Nachverfolgung dieser Änderungen durch diesen Dienst vorgenommen wurden. Abonnieren Sie den RSS-Feed auf, um automatische Benachrichtigungen über Änderungen an dieser Seite zu erhalten [AWS CodeCommit Dokumenthistorie des Benutzerhandbuchs](#).

Änderung	Beschreibung	Datum
AWS verwaltete Richtlinie: AWSCodeCommitFullAccess und AWS verwaltete Richtlinie: AWSCodeCommitPowerUser — Aktualisierung vorhandener Richtlinien	CodeCommit hat diesen Richtlinien eine Berechtigung hinzugefügt, um einen zusätzlichen Benachrichtigungstyp mithilfe von Amazon Q Developer in Chat-Anwendungen zu unterstützen. Die AWSCodeCommitFullAccess Richtlinien AWSCodeCommitPowerUser und wurden geändert, um eine Berechtigung hinzuzufügen, <code>chatbot:ListMicrosoftTeamsChannelConfigurations</code>.	16. Mai 2023
AWS verwaltete Richtlinie: AWSCodeCommitReadOnly – Aktualisierung auf eine bestehende Richtlinie	CodeCommit eine doppelte Berechtigung wurde aus der Richtlinie entfernt. Die AWSCodeCommitReadOnly wurde geändert, um eine doppelte Berechtigung zu entfernen, <code>"iam:ListAccessKeys"</code>.	18. August 2021
CodeCommit hat begonnen, Änderungen zu verfolgen	CodeCommit hat begonnen, Änderungen für die AWS verwalteten Richtlinien zu verfolgen.	18. August 2021

Beispiele für vom Kunden verwaltete Richtlinien

Sie können Ihre eigenen benutzerdefinierten IAM-Richtlinien erstellen, um Berechtigungen für CodeCommit Aktionen und Ressourcen zu gewähren. Die benutzerdefinierten Richtlinien können Sie dann den IAM-Benutzern oder -Gruppen zuweisen, die diese Berechtigungen benötigen. Sie können auch Ihre eigenen benutzerdefinierten IAM-Richtlinien für die Integration zwischen CodeCommit und anderen AWS Diensten erstellen.

Themen

- [Beispiele für vom Kunden verwaltete Identitätsrichtlinien](#)
- [Beispiele für vom Kunden verwaltete Integrationsrichtlinien](#)

Beispiele für vom Kunden verwaltete Identitätsrichtlinien

Das folgende Beispiel für IAM-Richtlinien gewährt Berechtigungen für verschiedene CodeCommit Aktionen. Verwenden Sie sie, um den CodeCommit Zugriff für Ihre IAM-Benutzer und -Rollen einzuschränken. Diese Richtlinien steuern die Fähigkeit, Aktionen mit der CodeCommit Konsole, der API oder dem AWS CLI auszuführen. AWS SDKs

Note

Alle Beispiele verwenden die Region USA West (Oregon) (us-west-2) und enthalten ein fiktives Konto. IDs

Beispiele

- [Beispiel 1: Erlauben Sie einem Benutzer, Operationen CodeCommit in einem einzigen AWS-Region](#)
- [Beispiel 2: Erlaube einem Benutzer, Git für ein einzelnes Repository zu verwenden](#)
- [Beispiel 3: Erlaubt einem Benutzer, der von einem bestimmten IP-Adressbereich aus eine Verbindung herstellt, Zugriff auf ein Repository](#)
- [Beispiel 4: Aktionen für Branches verweigern oder zulassen](#)
- [Beispiel 5: Aktionen für Repositorys mit Tags verweigern oder zulassen](#)

Beispiel 1: Erlauben Sie einem Benutzer, Operationen CodeCommit in einem einzigen AWS-Region

Die folgende Berechtigungsrichtlinie verwendet ein Platzhalterzeichen ("codecommit:*"), damit Benutzer alle CodeCommit Aktionen in der Region us-east-2 und nicht von anderen aus ausführen können. AWS-Regionen

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codecommit:*",
      "Resource": "arn:aws:codecommit:us-east-2:111111111111:*",
      "Condition": {
        "StringEquals": {
          "aws:RequestedRegion": "us-east-2"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "codecommit:ListRepositories",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:RequestedRegion": "us-east-2"
        }
      }
    }
  ]
}
```

Beispiel 2: Erlaube einem Benutzer, Git für ein einzelnes Repository zu verwenden

CodeCommitIn gelten die `GitPull` IAM-Richtlinienberechtigungen für jeden Git-Client-Befehl CodeCommit, von dem Daten abgerufen werden `git fetch` `git clone`, einschließlich, usw. In ähnlicher Weise gelten die `GitPush` IAM-Richtlinienberechtigungen für jeden Git-Client-Befehl, an den Daten gesendet CodeCommit werden. Wenn die `GitPush` IAM-Richtlinienberechtigung beispielsweise auf `Allow` gesetzt ist, kann ein Benutzer das Löschen eines Branches mithilfe des Git-Protokolls vorantreiben. Dieser Push wird nicht von den Berechtigungen beeinflusst, die für diesen IAM-Benutzer auf den `DeleteBranch` Vorgang angewendet wurden. Die `DeleteBranch` Berechtigung

gilt für Aktionen, die mit der Konsole, der AWS CLI SDKs, der und der API ausgeführt werden, aber nicht mit dem Git-Protokoll.

Das folgende Beispiel ermöglicht es dem angegebenen Benutzer, Daten aus dem angegebenen CodeCommit Repository abzurufen und dorthin zu pushenMyDemoRepo:

```
{
  "Version": "2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "codecommit:GitPull",
        "codecommit:GitPush"
      ],
      "Resource" : "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo"
    }
  ]
}
```

Beispiel 3: Erlaubt einem Benutzer, der von einem bestimmten IP-Adressbereich aus eine Verbindung herstellt, Zugriff auf ein Repository

Sie können eine Richtlinie erstellen, die es Benutzern nur dann erlaubt, sich mit einem CodeCommit-Repository zu verbinden, wenn ihre IP-Adresse innerhalb eines bestimmten IP-Adressbereichs liegt. Dazu gibt es zwei gleichwertige Ansätze. Sie können eine Deny Richtlinie erstellen, die CodeCommit Operationen verbietet, wenn sich die IP-Adresse des Benutzers nicht in einem bestimmten Block befindet, oder Sie können eine Allow Richtlinie erstellen, die CodeCommit Operationen zulässt, wenn sich die IP-Adresse des Benutzers innerhalb eines bestimmten Blocks befindet.

Sie können eine Deny-Richtlinie erstellen, die den Zugriff für alle Benutzer verweigert, die sich nicht aus einem bestimmten IP-Adressbereich stammen. Beispielsweise können Sie die verwaltete Richtlinie AWSCodeCommitPowerUser und eine vom Kunden verwaltete Richtlinie allen Benutzern anfügen, die Zugriff auf Ihr Repository benötigen. Die folgende Beispielrichtlinie verweigert Benutzern, deren IP-Adressen nicht innerhalb des angegebenen IP-Adressblocks 203.0.113.0/16 liegen, alle CodeCommit Berechtigungen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {

```



```

    "Effect": "Deny",
    "Action": [
        "codecommit:*"
    ],
    "Resource": "*",
    "Condition": {
        "NotIpAddress": {
            "aws:SourceIp": [
                "203.0.113.0/16"
            ]
        }
    }
}
]
}

```

Mit der folgenden Beispielrichtlinie kann der angegebene Benutzer nur dann auf ein CodeCommit Repository zugreifen, das MyDemoRepo mit den entsprechenden Berechtigungen der AWSCodeCommitPowerUser verwalteten Richtlinie benannt ist, wenn sich seine IP-Adresse innerhalb des angegebenen Adressblocks 203.0.113.0/16 befindet:

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codecommit:BatchGetRepositories",
        "codecommit:CreateBranch",
        "codecommit:CreateRepository",
        "codecommit:Get*",
        "codecommit:GitPull",
        "codecommit:GitPush",
        "codecommit:List*",
        "codecommit:Put*",
        "codecommit:Post*",
        "codecommit:Merge*",
        "codecommit:TagResource",
        "codecommit:Test*",
        "codecommit:UntagResource",
        "codecommit:Update*"
      ],
      "Resource": "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo",
    }
  ]
}

```

```
    "Condition": {
      "IpAddress": {
        "aws:SourceIp": [
          "203.0.113.0/16"
        ]
      }
    }
  ]
}
```

Beispiel 4: Aktionen für Branches verweigern oder zulassen

Sie können eine Richtlinie erstellen, die Benutzern die Berechtigungen für Aktionen verweigert, die Sie für eine oder mehrere Verzweigungen angeben. Alternativ können Sie eine Richtlinie erstellen, die Aktionen für eine oder mehrere Verzweigungen erlaubt, die sie sonst nicht in anderen Verzweigungen eines Repositories haben. Sie können diese Richtlinien mit den entsprechenden verwalteten (vordefinierten) Richtlinien verwenden. Weitere Informationen finden Sie unter [Beschränken Sie die Anzahl von Pushs und Merges auf Branches in AWS CodeCommit](#).

Sie können beispielsweise eine Deny Richtlinie erstellen, die Benutzern die Möglichkeit verweigert, Änderungen an einem Branch namens main vorzunehmen, einschließlich des Löschens dieses Branches, in einem Repository mit dem Namen *MyDemoRepo*. Sie können diese Richtlinien mit der verwalteten Richtlinie *AWSCodeCommitPowerUser* verwenden. Benutzer, auf die diese beiden Richtlinien angewendet wurden, könnten Branches erstellen und löschen, Pull Requests erstellen und alle anderen Aktionen ausführen, sofern dies von erlaubt ist. Sie wären jedoch nicht in der Lage *AWSCodeCommitPowerUser*, Änderungen an den Branch mit dem Namen main zu pushen, eine Datei im Main Branch in der CodeCommit Konsole hinzuzufügen oder zu bearbeiten oder Branches oder eine Pull-Anfrage mit dem Main Branch zusammenzuführen. Da Deny auf *GitPush* angewendet wird, müssen Sie eine Null-Anweisung in die Richtlinie aufnehmen, um die Analyse anfänglicher *GitPush*-Aufrufe auf Gültigkeit zu gestatten, wenn Benutzer Sendeoperationen von ihren lokalen Repositories aus ausführen.

Tip

Wenn Sie eine Richtlinie erstellen möchten, die für alle Branches mit dem Namen main in allen Repositories in Ihrem Amazon Web Services Services-Konto gilt, geben Sie ein Sternchen (*) anstelle eines Repository-ARN ein.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "codecommit:GitPush",
        "codecommit>DeleteBranch",
        "codecommit:PutFile",
        "codecommit:Merge*"
      ],
      "Resource": "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo",
      "Condition": {
        "StringEqualsIfExists": {
          "codecommit:References": [
            "refs/heads/main"
          ]
        },
        "Null": {
          "codecommit:References": "false"
        }
      }
    }
  ]
}
```

Die folgende Beispielrichtlinie ermöglicht es einem Benutzer, Änderungen an einem Branch namens main in allen Repositorys eines Amazon Web Services Services-Kontos vorzunehmen. Sie erlaubt keine Änderungen an anderen Zweigen. Sie können diese Richtlinie zusammen mit der AWSCodeCommitReadOnly verwalteten Richtlinie verwenden, um automatisierte Pushs an das Repository im Hauptzweig zu ermöglichen. Der Effekt ist Allow, deshalb würde diese Beispielrichtlinie nicht mit verwalteten Richtlinien funktionieren, wie z. B. AWSCodeCommitPowerUser.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codecommit:GitPush",
        "codecommit:Merge*"
      ],
    }
  ]
}
```

```

        "Resource": "*",
        "Condition": {
            "StringEqualsIfExists": {
                "codecommit:References": [
                    "refs/heads/main"
                ]
            }
        }
    }
}

```

Beispiel 5: Aktionen für Repositorys mit Tags verweigern oder zulassen

Sie können eine Richtlinie erstellen, die Aktionen für Repositorys auf der Grundlage der mit diesen Repositorys verknüpften AWS Tags erlaubt oder verweigert, und diese Richtlinien dann auf die IAM-Gruppen anwenden, die Sie für die Verwaltung von IAM-Benutzern konfigurieren. Sie können beispielsweise eine Richtlinie erstellen, die alle CodeCommit Aktionen in beliebigen Repositorys mit dem AWS Tag-Schlüssel Status und dem Schlüsselwert Secret verweigert, und diese Richtlinie dann auf die IAM-Gruppe anwenden, die Sie für allgemeine Entwickler erstellt haben (). *Developers* Anschließend müssen Sie sicherstellen, dass die Entwickler, die an diesen markierten Repositorys arbeiten, nicht Mitglieder dieser allgemeinen *Developers* Gruppe sind, sondern einer anderen IAM-Gruppe angehören, auf die die restriktive Richtlinie nicht angewendet wurde (). SecretDevelopers

Im folgenden Beispiel werden alle CodeCommit Aktionen für Repositorys verweigert, die mit dem Schlüssel Status und dem Schlüsselwert Secret gekennzeichnet sind:

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "codecommit:Associate*",
        "codecommit:Batch*",
        "codecommit:CancelUploadArchive",
        "codecommit:CreateBranch",
        "codecommit:CreateCommit",
        "codecommit:CreatePullRequest*",
        "codecommit:CreateRepository",
        "codecommit:CreateUnreferencedMergeCommit",

```

```
"codecommit:DeleteBranch",
"codecommit:DeleteCommentContent",
"codecommit:DeleteFile",
"codecommit:DeletePullRequest*",
"codecommit:DeleteRepository",
"codecommit:Describe*",
"codecommit:DisassociateApprovalRuleTemplateFromRepository",
"codecommit:EvaluatePullRequestApprovalRules",
"codecommit:GetBlob",
"codecommit:GetBranch",
"codecommit:GetComment*",
"codecommit:GetCommit",
"codecommit:GetDifferences*",
"codecommit:GetFile",
"codecommit:GetFolder",
"codecommit:GetMerge*",
"codecommit:GetObjectIdentifier",
"codecommit:GetPullRequest*",
"codecommit:GetReferences",
"codecommit:GetRepository*",
"codecommit:GetTree",
"codecommit:GetUploadArchiveStatus",
"codecommit:Git*",
"codecommit:ListAssociatedApprovalRuleTemplatesForRepository",
"codecommit:ListBranches",
"codecommit:ListPullRequests",
"codecommit:ListTagsForResource",
"codecommit:Merge*",
"codecommit:OverridePullRequestApprovalRules",
"codecommit:Post*",
"codecommit:Put*",
"codecommit:TagResource",
"codecommit:TestRepositoryTriggers",
"codecommit:UntagResource",
"codecommit:UpdateComment",
"codecommit:UpdateDefaultBranch",
"codecommit:UpdatePullRequest*",
"codecommit:UpdateRepository*",
"codecommit:UploadArchive"
],
"Resource": "*",
"Condition": {
  "StringEquals": {
    "aws:ResourceTag/Status": "Secret"
  }
}
```

```

    }
  }
}
]
}

```

Sie können diese Strategie weiter verfeinern, indem Sie bestimmte Repositorys und nicht alle Repositorys als Ressourcen angeben. Sie können auch Richtlinien erstellen, die CodeCommit Aktionen für alle Repositorys zulassen, die nicht mit bestimmten Tags gekennzeichnet sind. Die folgende Richtlinie erlaubt beispielsweise das Äquivalent von AWSCodeCommitPowerUserBerechtigungen für CodeCommit Aktionen, mit der Ausnahme, dass sie nur Aktionen für Repositorys zulässt CodeCommit , die nicht mit den angegebenen Tags gekennzeichnet sind:

Note

Dieses Richtlinienbeispiel umfasst nur Aktionen für CodeCommit. Es umfasst keine Aktionen für andere AWS Dienste, die in der AWSCodeCommitPowerUserverwalteten Richtlinie enthalten sind. Weitere Informationen finden Sie unter [AWS verwaltete Richtlinie: AWSCodeCommitPowerUser..](#)

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codecommit:Associate*",
        "codecommit:Batch*",
        "codecommit:CancelUploadArchive",
        "codecommit:CreateBranch",
        "codecommit:CreateCommit",
        "codecommit:CreatePullRequest*",
        "codecommit:CreateRepository",
        "codecommit:CreateUnreferencedMergeCommit",
        "codecommit>DeleteBranch",
        "codecommit>DeleteCommentContent",
        "codecommit>DeleteFile",
        "codecommit>DeletePullRequest*",
        "codecommit:Describe*",

```

```

    "codecommit:DisassociateApprovalRuleTemplateFromRepository",
    "codecommit:EvaluatePullRequestApprovalRules",
    "codecommit:GetBlob",
    "codecommit:GetBranch",
    "codecommit:GetComment*",
    "codecommit:GetCommit",
    "codecommit:GetDifferences*",
    "codecommit:GetFile",
    "codecommit:GetFolder",
    "codecommit:GetMerge*",
    "codecommit:GetObjectIdentifier",
    "codecommit:GetPullRequest*",
    "codecommit:GetReferences",
    "codecommit:GetRepository*",
    "codecommit:GetTree",
    "codecommit:GetUploadArchiveStatus",
    "codecommit:Git*",
    "codecommit:ListAssociatedApprovalRuleTemplatesForRepository",
    "codecommit:ListBranches",
    "codecommit:ListPullRequests",
    "codecommit:ListTagsForResource",
    "codecommit:Merge*",
    "codecommit:OverridePullRequestApprovalRules",
    "codecommit:Post*",
    "codecommit:Put*",
    "codecommit:TagResource",
    "codecommit:TestRepositoryTriggers",
    "codecommit:UntagResource",
    "codecommit:UpdateComment",
    "codecommit:UpdateDefaultBranch",
    "codecommit:UpdatePullRequest*",
    "codecommit:UpdateRepository*",
    "codecommit:UploadArchive"
  ],
  "Resource": "*",
  "Condition": {
    "StringNotEquals": {
      "aws:ResourceTag/Status": "Secret",
      "aws:ResourceTag/Team": "Saanvi"
    }
  }
},
{
  "Effect": "Allow",

```

```
"Action": [  
  "codecommit:CreateApprovalRuleTemplate",  
  "codecommit:GetApprovalRuleTemplate",  
  "codecommit:ListApprovalRuleTemplates",  
  "codecommit:ListRepositories",  
  "codecommit:ListRepositoriesForApprovalRuleTemplate",  
  "codecommit:UpdateApprovalRuleTemplateContent",  
  "codecommit:UpdateApprovalRuleTemplateDescription",  
  "codecommit:UpdateApprovalRuleTemplateName"  
],  
"Resource": "*" ]  
}
```

Beispiele für vom Kunden verwaltete Integrationsrichtlinien

Dieser Abschnitt enthält Beispiele für vom Kunden verwaltete Benutzerrichtlinien, die Berechtigungen für Integrationen zwischen CodeCommit und anderen AWS Diensten gewähren. Spezifische Beispiele für Richtlinien, die den kontoübergreifenden Zugriff auf ein CodeCommit-Repository erlauben, finden Sie unter [Konfiguriere den kontoübergreifenden Zugriff auf ein AWS CodeCommit Repository mithilfe von Rollen](#).

Note

Alle Beispiele verwenden die Region USA West (Oregon) (us-west-2), wenn ein erforderlich AWS-Region ist, und enthalten ein fiktives Konto. IDs

Beispiele

- [Beispiel 1: Erstellen Sie eine Richtlinie, die den kontoübergreifenden Zugriff auf ein Amazon SNS SNS-Thema ermöglicht](#)
- [Beispiel 2: Erstellen Sie eine Amazon Simple Notification Service \(Amazon SNS\) -Themenrichtlinie, damit Amazon CloudWatch Events Ereignisse zu diesem Thema veröffentlichen CodeCommit kann](#)
- [Beispiel 3: Erstellen Sie eine Richtlinie für die AWS Lambda Integration mit einem Trigger CodeCommit](#)

Beispiel 1: Erstellen Sie eine Richtlinie, die den kontoübergreifenden Zugriff auf ein Amazon SNS SNS-Thema ermöglicht

Sie können ein CodeCommit Repository so konfigurieren, dass Code-Pushs oder andere Ereignisse Aktionen auslösen, z. B. das Senden einer Benachrichtigung von Amazon Simple Notification Service (Amazon SNS). Wenn Sie das Amazon SNS SNS-Thema mit demselben Konto erstellen, das für die Erstellung des CodeCommit Repositorys verwendet wurde, müssen Sie keine zusätzlichen IAM-Richtlinien oder -Berechtigungen konfigurieren. Sie können das Thema und dann den Auslöser für das Repository erstellen. Weitere Informationen finden Sie unter [Einen Auslöser für ein Amazon SNS SNS-Thema erstellen](#).

Wenn Sie Ihren Auslöser jedoch so konfigurieren möchten, dass er ein Amazon SNS SNS-Thema in einem anderen Amazon Web Services Services-Konto verwendet, müssen Sie dieses Thema zunächst mit einer Richtlinie konfigurieren, die es ermöglicht, CodeCommit zu diesem Thema zu veröffentlichen. Öffnen Sie von diesem anderen Konto aus die Amazon SNS SNS-Konsole, wählen Sie das Thema aus der Liste aus und wählen Sie für Andere Themenaktionen die Option Themenrichtlinie bearbeiten aus. Ändern Sie auf der Registerkarte „Erweitert“ die Richtlinie für das Thema, sodass es CodeCommit zu diesem Thema veröffentlicht werden kann. Wenn es sich bei der Richtlinie beispielsweise um die Standardrichtlinie handelt, würden Sie die Richtlinie wie folgt ändern und die Elemente so ändern, *red italic text* dass sie den Werten für Ihr Repository, Ihr Amazon SNS SNS-Thema und Ihr Konto entsprechen:

```
{
  "Version": "2008-10-17",
  "Id": "__default_policy_ID",
  "Statement": [
    {
      "Sid": "__default_statement_ID",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "sns:Subscribe",
        "sns:ListSubscriptionsByTopic",
        "sns>DeleteTopic",
        "sns:GetTopicAttributes",
        "sns:Publish",
        "sns:RemovePermission",
        "sns:AddPermission",
        "sns:SetTopicAttributes"
      ]
    }
  ]
}
```

```

    "Resource": "arn:aws:sns:us-east-2:111111111111:NotMySNSTopic",
    "Condition": {
      "StringEquals": {
        "AWS:SourceOwner": "111111111111"
      }
    },
  },
  {
    "Sid": "CodeCommit-Policy_ID",
    "Effect": "Allow",
    "Principal": {
      "Service": "codecommit.amazonaws.com"
    },
    "Action": "sns:Publish",
    "Resource": "arn:aws:sns:us-east-2:111111111111:NotMySNSTopic",
    "Condition": {
      "StringEquals": {
        "AWS:SourceArn": "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo",
        "AWS:SourceAccount": "111111111111"
      }
    }
  }
]
}

```

Beispiel 2: Erstellen Sie eine Amazon Simple Notification Service (Amazon SNS) -Themenrichtlinie, damit Amazon CloudWatch Events Ereignisse zu diesem Thema veröffentlichen CodeCommit kann

Sie können CloudWatch Ereignisse so konfigurieren, dass sie bei Eintreten von Ereignissen, einschließlich CodeCommit Ereignissen, in einem Amazon SNS SNS-Thema veröffentlicht werden. Dazu müssen Sie sicherstellen, dass CloudWatch Events berechtigt ist, Ereignisse zu Ihrem Amazon SNS SNS-Thema zu veröffentlichen, indem Sie eine Richtlinie für das Thema erstellen oder eine bestehende Richtlinie für das Thema ändern, die der folgenden ähnelt:

```

{
  "Version": "2008-10-17",
  "Id": "__default_policy_ID",
  "Statement": [
    {
      "Sid": "__default_statement_ID",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      }
    }
  ]
}

```

```
    },
    "Action": "sns:Publish",
    "Resource": "arn:aws:sns:us-east-2:123456789012:MyTopic",
    "Condition": {
      "StringEquals": {
        "AWS:SourceOwner": "123456789012"
      }
    }
  },
  {
    "Sid": "Allow_Publish_Events",
    "Effect": "Allow",
    "Principal": {
      "Service": "events.amazonaws.com"
    },
    "Action": "sns:Publish",
    "Resource": "arn:aws:sns:us-east-2:123456789012:MyTopic"
  }
]
```

Weitere Informationen zu CodeCommit und CloudWatch Veranstaltungen finden Sie unter [Beispiele für CloudWatch Ereignisse und Ereignisse von Supported Services](#). Weitere Informationen zu IAM und der Richtlinienprache finden Sie unter [Grammatik der IAM-JSON-Richtliniensprache](#).

Beispiel 3: Erstellen Sie eine Richtlinie für die AWS Lambda Integration mit einem Trigger CodeCommit

Sie können ein CodeCommit Repository so konfigurieren, dass Code-Pushs oder andere Ereignisse Aktionen auslösen, z. B. das Aufrufen einer Funktion in AWS Lambda. Weitere Informationen finden Sie unter [Erstellen Sie einen Trigger für eine Lambda-Funktion](#). Diese Informationen beziehen sich nur auf Trigger und nicht CloudWatch auf Ereignisse.

Wenn Sie möchten, dass Ihr Trigger eine Lambda-Funktion direkt ausführt (anstatt ein Amazon SNS Thema zum Aufrufen der Lambda-Funktion zu verwenden), und Sie den Trigger nicht in der Lambda-Konsole konfigurieren, müssen Sie eine Anweisung ähnlich der folgenden in die ressourcenbasierte Richtlinie der Funktion aufnehmen:

```
{
  "Statement":{
    "StatementId":"Id-1",
    "Action":"lambda:InvokeFunction",
```

```
"Principal": "codecommit.amazonaws.com",
"SourceArn": "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo",
"SourceAccount": "111111111111"
}
}
```

Wenn Sie einen CodeCommit Trigger manuell konfigurieren, der eine Lambda-Funktion aufruft, müssen Sie auch den [AddPermission](#) Lambda-Befehl verwenden, um die Erlaubnis zum Aufrufen der Funktion CodeCommit zu erteilen. Ein Beispiel finden Sie im Abschnitt [Um die Ausführung einer Lambda-Funktion CodeCommit zu ermöglichen](#) von [Erstellen Sie einen Trigger für eine bestehende Lambda-Funktion](#).

Weitere Informationen zu Ressourcenrichtlinien für Lambda-Funktionen finden Sie unter [AddPermission](#) und [The Pull/Push Event Models im AWS Lambda Developer](#) Guide.

Referenz für CodeCommit-Berechtigungen

In den folgenden Tabellen sind die einzelnen CodeCommit API-Operationen, die entsprechenden Aktionen, für die Sie Berechtigungen erteilen können, und das Format des Ressourcen-ARN aufgeführt, der für die Erteilung von Berechtigungen verwendet werden soll. Sie CodeCommit APIs sind auf der Grundlage des Umfangs der Aktionen, die von dieser API zugelassen sind, in Tabellen gruppiert. Beziehen Sie sich darauf, wenn Sie Berechtigungsrichtlinien einrichten [Zugriffskontrolle](#) und schreiben, die Sie einer IAM-Identität zuordnen können (identitätsbasierte Richtlinien).

Beim Erstellen einer Berechtigungsrichtlinie geben Sie die Aktionen im Feld `Action` der Richtlinie an. Sie geben den Ressourcenwert im Feld `Resource` der Richtlinie als ARN mit oder ohne Platzhalterzeichen (*) an.

Verwenden AWS Sie Bedingungsschlüssel, um Bedingungen in Ihren CodeCommit Richtlinien auszudrücken. Eine vollständige Liste der AWS-weiten Schlüssel finden Sie unter [Verfügbare Schlüssel](#) im IAM-Benutzerhandbuch. Vollständige Informationen zu Aktionen, Ressourcen und Bedingungsschlüsseln für CodeCommit in IAM-Richtlinien finden Sie unter [Aktionen, Ressourcen und Bedingungsschlüssel](#) für AWS CodeCommit

Note

Um eine Aktion anzugeben, verwenden Sie das Präfix `codecommit:` gefolgt vom Namen der API-Operation (z. B. `codecommit:GetRepository` oder `codecommit>CreateRepository`).

Verwenden von Platzhaltern

Sie können ein Platzhalterzeichen (*) in Ihrem ARN verwenden, um mehrere Aktionen oder Ressourcen anzugeben. `codecommit:*` Gibt beispielsweise alle Aktionen an und `codecommit:Get*` gibt alle CodeCommit CodeCommit Aktionen an, die mit dem Wort `Get` beginnen. Im folgenden Beispiel wird der Zugriff auf alle Repositorys erteilt, deren Name mit `MyDemo` beginnt:

```
arn:aws:codecommit:us-west-2:111111111111:MyDemo*
```

Sie können Platzhalter nur für die in der folgenden Tabelle aufgeführten *repository-name* Ressourcen verwenden. Sie können Platzhalter nicht zusammen mit *region* Ressourcen verwenden. *account-id* Weitere Informationen zu Platzhaltern finden Sie unter [IAM-Identifikatoren im IAM-Benutzerhandbuch](#).

Themen

- [Erforderliche Berechtigungen für Git-Client-Befehle](#)
- [Berechtigungen für Aktionen in Branches](#)
- [Berechtigungen für Aktionen bei Zusammenführungen](#)
- [Berechtigungen für Aktionen bei Pull-Requests](#)
- [Berechtigungen für Aktionen auf Vorlagen für Genehmigungsregeln](#)
- [Berechtigungen für Aktionen an einzelnen Dateien](#)
- [Berechtigungen für Aktionen im Zusammenhang mit Kommentaren](#)
- [Berechtigungen für Aktionen an festgeschriebenem Code](#)
- [Berechtigungen für Aktionen in Repositorys](#)
- [Berechtigungen für Aktionen mit Tags](#)
- [Berechtigungen für Aktionen auf Triggern](#)
- [Berechtigungen für Aktionen bei der CodePipeline Integration](#)

Erforderliche Berechtigungen für Git-Client-Befehle

CodeCommitIn gelten die `GitPull` IAM-Richtlinienberechtigungen für jeden Git-Client-Befehl CodeCommit, von dem Daten abgerufen werden `git fetch` `git clone`, einschließlich, usw. In ähnlicher

Weise gelten die `GitPush` IAM-Richtlinienberechtigungen für jeden `Git-Client`-Befehl, an den Daten gesendet `CodeCommit` werden. Wenn die `GitPush` IAM-Richtlinienberechtigung beispielsweise auf `gesetzt istAllow`, kann ein Benutzer das Löschen eines Branches mithilfe des `Git-Protokolls` vorantreiben. Dieser Push wird nicht von den Berechtigungen beeinflusst, die für diesen IAM-Benutzer auf den `DeleteBranch` Vorgang angewendet wurden. Die `DeleteBranch` Berechtigung gilt für Aktionen, die mit der Konsole, der AWS CLI SDKs, der und der API ausgeführt werden, aber nicht mit dem `Git-Protokoll`.

`GitPull` und `GitPush` sind IAM-Richtlinienberechtigungen. Es handelt sich dabei nicht um API-Aktionen.

CodeCommit Erforderliche Berechtigungen für Aktionen für `Git-Client`-Befehle

`GitPull`

Aktion(en): `codecommit:GitPull`

Erforderlich, um Informationen aus einem `CodeCommit Repository` in ein lokales `Repository` abzurufen. Dies ist nur eine IAM-Richtlinienberechtigung, keine API-Aktion.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

`GitPush`

Aktion(en): `codecommit:Git Push`

Erforderlich, um Informationen von einem lokalen `Repository` in ein `CodeCommit Repository` zu übertragen. Dies ist nur eine IAM-Richtlinienberechtigung, keine API-Aktion.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

Berechtigungen für Aktionen in Branches

Die folgenden Berechtigungen erlauben oder verweigern Aktionen für Branches in `CodeCommit Repositorys`. Diese Berechtigungen beziehen sich nur auf Aktionen, die in der `CodeCommit Konsole` und mit der `CodeCommit API` ausgeführt werden, sowie auf Befehle, die mit der ausgeführt werden. `AWS CLI` Sie gelten nicht für ähnliche Aktionen, die mit dem `Git-Protokoll` ausgeführt werden können. Beispielsweise zeigt der Befehl `git show-branch -r` eine Liste externer Branches für ein `Repository` und seine Commits unter Verwendung des `Git-Protokolls` an. Sie wird durch keine Berechtigungen für den `CodeCommit ListBranches` Vorgang beeinflusst.

Weitere Informationen zu Richtlinien für Filialen finden Sie unter [Beschränken Sie die Anzahl von Pushs und Merges auf Branches in AWS CodeCommit](#) und [Beispiele für vom Kunden verwaltete Richtlinien](#).

CodeCommit API-Operationen und erforderliche Berechtigungen für Aktionen in Zweigen

[CreateBranch](#)

Aktion(en): `codecommit:CreateBranch`

Erforderlich, um einen Branch in einem CodeCommit Repository zu erstellen.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

[DeleteBranch](#)

Aktion(en): `codecommit>DeleteBranch`

Erforderlich, um einen Branch aus einem CodeCommit Repository zu löschen.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

[GetBranch](#)

Aktion(en): `codecommit:GetBranch`

Erforderlich, um Details zu einem Branch in einem CodeCommit Repository abzurufen.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

[ListBranches](#)

Aktion(en): `codecommit:ListBranches`

Erforderlich, um eine Liste von Branches in einem CodeCommit Repository abzurufen.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

[MergeBranchesByFastForward](#)

Aktion(en): `codecommit:MergeBranchesByFastForward`

Erforderlich, um zwei Branches mithilfe der Fast-Forward-Merge-Strategie in einem CodeCommit Repository zusammenzuführen.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

MergeBranchesBySquash

Aktion(en): `codecommit:ListBranches`

Erforderlich zum Zusammenführen von zwei Branches mit der Squashmerge-Strategie in einem CodeCommit -Repository.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

MergeBranchesByThreeWay

Aktion(en): `codecommit:ListBranches`

Erforderlich zum Zusammenführen von zwei Branches mithilfe der Dreiwegemerge-Strategie in einem CodeCommit -Repository.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

UpdateDefaultBranch

Aktion(en): `codecommit:UpdateDefaultBranch`

Erforderlich, um den Standard-Branch in einem CodeCommit Repository zu ändern.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

Berechtigungen für Aktionen bei Zusammenführungen

Die folgenden Berechtigungen erlauben oder verweigern Aktionen bei Zusammenführungen in CodeCommit Repositorys. Diese Berechtigungen beziehen sich auf Aktionen, die mit der CodeCommit Konsole und der CodeCommit API ausgeführt werden, sowie auf Befehle, die mit der ausgeführt werden. AWS CLI Sie gelten nicht für ähnliche Aktionen, die mit dem Git-Protokoll ausgeführt werden können. Entsprechende Berechtigungen für Branches finden Sie unter [Berechtigungen für Aktionen in Branches](#). Entsprechende Berechtigungen für Pull-Anforderungen finden Sie unter [Berechtigungen für Aktionen bei Pull-Requests](#).

CodeCommit API-Operationen und erforderliche Berechtigungen für Aktionen für Merge-Befehle

BatchDescribeMergeConflicts

Aktion(en): `codecommit:BatchDescribeMergeConflicts`

Erforderlich, um Informationen über Konflikte bei einer Zusammenführung zwischen Commits in einem CodeCommit Repository zurückzugeben.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

CreateUnreferencedMergeCommit

Aktion(en): `codecommit:CreateUnreferencedMergeCommit`

Erforderlich, um einen nicht referenzierten Commit zwischen zwei Branches oder Commits in einem CodeCommit Repository zu erstellen, um diese zu vergleichen und mögliche Konflikte zu identifizieren.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

DescribeMergeConflicts

Aktion(en): `codecommit:DescribeMergeConflicts`

Erforderlich zum Abruf von Informationen über Zusammenführungskonflikte zwischen den Basis-, Quell- und Ziel-Versionen einer Datei in einer potenziellen Zusammenführung in einem CodeCommit -Repository.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetMergeCommit

Aktion(en): `codecommit:GetMergeCommit`

Erforderlich, um Informationen über die Zusammenführung zwischen einem Quell- und einem Ziel-Commit in einem CodeCommit Repository zurückzugeben.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetMergeOptions

Aktion(en): `codecommit:GetMergeOptions`

Erforderlich, um Informationen über die verfügbaren Zusammenführungsoptionen zwischen zwei Branches oder Commit-Spezifizierern in einem CodeCommit Repository zurückzugeben.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

Berechtigungen für Aktionen bei Pull-Requests

Die folgenden Berechtigungen erlauben oder verweigern Aktionen für Pull-Anfragen in CodeCommit-Repositorys. Diese Berechtigungen beziehen sich auf Aktionen, die mit der CodeCommit Konsole

und der CodeCommit API ausgeführt werden, sowie auf Befehle, die mit der AWS CLI ausgeführt werden. Sie gelten nicht für ähnliche Aktionen, die mit dem Git-Protokoll ausgeführt werden können. Weitere Berechtigungen für Kommentare finden Sie unter [Berechtigungen für Aktionen im Zusammenhang mit Kommentaren](#).

CodeCommit API-Operationen und erforderliche Berechtigungen für Aktionen auf Pull-Requests

BatchGetPullRequests

Aktion(en): `codecommit:BatchGetPullRequests`

Erforderlich, um Informationen zu einer oder mehreren Pull-Anforderungen in einem CodeCommit-Repository zurückzugeben. Dies ist nur eine IAM-Richtlinienberechtigung, keine API-Aktion, die Sie aufrufen können.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

[CreatePullRequest](#)

Aktion(en): `codecommit>CreatePullRequest`

Erforderlich, um eine Pull-Anfrage in einem CodeCommit Repository zu erstellen.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

[CreatePullRequestApprovalRule](#)

Aktion(en): `codecommit>CreatePullRequestApprovalRule`

Erforderlich, um eine Genehmigungsregel für eine Pull-Anforderung in einem CodeCommit-Repository zu erstellen.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

[DeletePullRequestApprovalRule](#)

Aktion(en): `codecommit>DeletePullRequestApprovalRule`

Erforderlich, um eine Genehmigungsregel für eine Pull-Anforderung in einem CodeCommit-Repository zu löschen.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

[DescribePullRequestEvents](#)

Aktion(en): `codecommit:DescribePullRequestEvents`

Erforderlich, um Informationen über ein oder mehrere Pull-Request-Ereignisse in einem CodeCommit Repository zurückzugeben.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

[EvaluatePullRequestApprovalRules](#)

Aktion(en): `codecommit:EvaluatePullRequestApprovalRules`

Erforderlich, um zu bewerten, ob eine Pull-Anfrage alle Bedingungen erfüllt hat, die in den zugehörigen Genehmigungsregeln in einem CodeCommit Repository angegeben sind.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

[GetCommentsForPullRequest](#)

Aktion(en): `codecommit:GetCommentsForPullRequest`

Erforderlich, um Kommentare in einer Pull-Anfrage zurückzugeben.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

[GetCommitsFromMergeBase](#)

Aktion(en): `codecommit:GetCommitsFromMergeBase`

Erforderlich, um Informationen über den Unterschied zwischen Commits im Zusammenhang mit einer potenziellen Zusammenführung zurückzugeben. Dies ist nur eine IAM-Richtlinienberechtigung, keine API-Aktion, die Sie aufrufen können.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

[GetMergeConflicts](#)

Aktion(en): `codecommit:GetMergeConflicts`

Erforderlich, um Informationen über Merge-Konflikte zwischen dem Quell- und dem Ziel-Branch in einer Pull-Anfrage zurückzugeben.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

[GetPullRequest](#)

Aktion(en): `codecommit:GetPullRequest`

Erforderlich, um Informationen zu einer Pull-Anforderung in einem CodeCommit-Repository zurückzugeben.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetPullRequestApprovalStates

Aktion(en): `codecommit:GetPullRequestApprovalStates`

Erforderlich, um Informationen zu den Genehmigungsstatus für eine angegebene Pull-Anforderung zurückzugeben.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetPullRequestOverrideState

Aktion(en): `codecommit:GetPullRequestOverrideState`

Erforderlich, um Informationen darüber zurückzugeben, ob Genehmigungsregeln für eine Pull-Anfrage reserviert (überschrieben) wurden, und falls ja, den Amazon-Ressourcennamen (ARN) des Benutzers oder der Identität, der die Regeln und deren Anforderungen für die Pull-Anfrage außer Kraft gesetzt hat.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

ListPullRequests

Aktion(en): `codecommit:ListPullRequests`

Erforderlich für das Auflisten von Pull-Anfragen in einem Repository.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

MergePullRequestByFastForward

Aktion(en): `codecommit:MergePullRequestByFastForward`

Erforderlich für das Schließen einer Pull-Anforderung und die versuchte Zusammenführung des Quell-Branch mit dem Ziel-Branch einer Pull-Anforderung unter Verwendung der Mergestrategie mit Vorlauf.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

MergePullRequestBySquash

Aktion(en): `codecommit:MergePullRequestBySquash`

Erforderlich für das Schließen einer Pull-Anforderung und für die versuchte Zusammenführung des Quell-Branch mit dem Ziel-Branch einer Pull-Anforderung unter Verwendung der Squashmerge-Strategie.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

MergePullRequestByThreeWay

Aktion(en): `codecommit:MergePullRequestByThreeWay`

Erforderlich für das Schließen einer Pull-Anforderung und die versuchte Zusammenführung des Quell-Branch mit dem Ziel-Branch einer Pull-Anforderung unter Verwendung der „Three-Way“-Strategie.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

OverridePullRequestApprovalRules

Aktion(en): `codecommit:OverridePullRequestApprovalRules`

Erforderlich, um alle Anforderungen an Genehmigungsregeln für eine Pull-Anfrage in einem Repository aufzuheben. CodeCommit

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

PostCommentForPullRequest

Aktion(en): `codecommit:PostCommentForPullRequest`

Erforderlich, um einen Kommentar zu einer Pull-Anforderung in einem CodeCommit-Repository zu veröffentlichen.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

UpdatePullRequestApprovalRuleContent

Aktion(en): `codecommit:UpdatePullRequestApprovalRuleContent`

Erforderlich, um die Struktur einer Genehmigungsregel für eine Pull-Anfrage in einem CodeCommit Repository zu ändern.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

UpdatePullRequestApprovalState

Aktion(en): `codecommit:UpdatePullRequestApprovalState`

Erforderlich, um den Status einer Genehmigung für eine Pull-Anforderung in einem CodeCommit -Repository zu aktualisieren.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

[UpdatePullRequestDescription](#)

Aktion(en): `codecommit:UpdatePullRequestDescription`

Erforderlich zum Ändern der Beschreibung einer Pull-Anfrage in einem CodeCommit-Repository.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

[UpdatePullRequestStatus](#)

Aktion(en): `codecommit:UpdatePullRequestStatus`

Erforderlich zum Ändern des Status einer Pull-Anfrage in einem CodeCommit-Repository.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

[UpdatePullRequestTitle](#)

Aktion(en): `codecommit:UpdatePullRequestTitle`

Erforderlich zum Ändern des Titels einer Pull-Anfrage in einem CodeCommit-Repository.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

Berechtigungen für Aktionen auf Vorlagen für Genehmigungsregeln

Die folgenden Berechtigungen erlauben oder verweigern Aktionen mit Genehmigungsregelvorlagen in CodeCommit -Repositories. Diese Berechtigungen beziehen sich nur auf Aktionen, die in der CodeCommit Konsole und der CodeCommit API ausgeführt werden, und auf Befehle, die mit der AWS CLI ausgeführt werden. Sie gelten nicht für ähnliche Aktionen, die mit dem Git-Protokoll ausgeführt werden können. Entsprechende Berechtigungen für Pull-Anforderungen finden Sie unter [Berechtigungen für Aktionen bei Pull-Requests](#).

CodeCommit API-Operationen und erforderliche Berechtigungen für Aktionen auf Vorlagen für Genehmigungsregeln

[AssociateApprovalRuleTemplateWithRepository](#)

Aktion(en): `codecommit:AssociateApprovalRuleTemplateWithRepository`

Erforderlich, um eine Vorlage mit einem bestimmten Repository in einem Amazon Web Services Services-Konto zu verknüpfen. Nach der Zuordnung werden automatisch Genehmigungsregeln erstellt, die den Vorlagenbedingungen für jede im angegebenen Repository erstellte Pull-Anforderung entsprechen.

Ressource: *

[BatchAssociateApprovalRuleTemplateWithRepositories](#)

Aktion(en): `codecommit:BatchAssociateApprovalRuleTemplateWithRepositories`

Erforderlich, um eine Vorlage mit einem oder mehreren angegebenen Repositories in einem Amazon Web Services Services-Konto zu verknüpfen.

Ressource: *

[BatchDisassociateApprovalRuleTemplateFromRepositories](#)

Aktion(en):

`codecommit:BatchDisassociateApprovalRuleTemplateFromRepositories`

Erforderlich, um eine Vorlage von einem oder mehreren angegebenen Repositories in einem Amazon Web Services Services-Konto zu trennen.

Ressource: *

[CreateApprovalRuleTemplate](#)

Aktion(en): `codecommit>CreateApprovalRuleTemplate`

Erforderlich, um eine Vorlage für Genehmigungsregeln zu erstellen, die dann mit einem oder mehreren Repositories in Ihrem Amazon Web Services Services-Konto verknüpft werden kann.

Ressource: *

[DeleteApprovalRuleTemplate](#)

Aktion(en): `codecommit>DeleteApprovalRuleTemplate`

Erforderlich, um eine Genehmigungsregelvorlage aus einem AWS -Konto zu löschen.

Ressource: *

[DisassociateApprovalRuleTemplateFromRepository](#)

Aktion(en): `codecommit:DisassociateApprovalRuleTemplateFromRepository`

Erforderlich, um die angegebene Vorlage von einem Repository in einem Amazon Web Services Services-Konto zu trennen. Genehmigungsregeln für bereits mit der Vorlage erstellte Pull-Anforderungen werden nicht entfernt.

Ressource: *

[GetApprovalRuleTemplate](#)

Aktion(en): `codecommit:GetApprovalRuleTemplate`

Erforderlich, um Informationen zu einer Vorlage für Genehmigungsregeln in einem Amazon Web Services Services-Konto zurückzugeben.

Ressource: *

[ListApprovalRuleTemplates](#)

Aktion(en): `codecommit:ListApprovalRuleTemplates`

Erforderlich, um Vorlagen für Genehmigungsregeln in einem Amazon Web Services Services-Konto aufzulisten.

Ressource: *

[ListAssociatedApprovalRuleTemplatesForRepository](#)

Aktion(en): `codecommit:ListAssociatedApprovalRuleTemplatesForRepository`

Erforderlich, um alle Vorlagen für Genehmigungsregeln aufzulisten, die einem bestimmten Repository in einem Amazon Web Services Services-Konto zugeordnet sind.

Ressource: *

[ListRepositoriesForApprovalRuleTemplate](#)

Aktion(en): `codecommit:ListRepositoriesForApprovalRuleTemplate`

Erforderlich, um alle Repositories aufzulisten, die mit einer bestimmten Genehmigungsregelvorlage in einem Amazon Web Services Services-Konto verknüpft sind.

Ressource: *

[UpdateApprovalRuleTemplateContent](#)

Aktion(en): `codecommit:UpdateApprovalRuleTemplateContent`

Erforderlich, um den Inhalt einer Vorlage für Genehmigungsregeln in einem Amazon Web Services Services-Konto zu aktualisieren.

Ressource: *

[UpdateApprovalRuleTemplateDescription](#)

Aktion(en): `codecommit:UpdateApprovalRuleTemplateDescription`

Erforderlich, um die Beschreibung einer Vorlage für Genehmigungsregeln in einem Amazon Web Services Services-Konto zu aktualisieren.

Ressource: *

[UpdateApprovalRuleTemplateName](#)

Aktion(en): `codecommit:UpdateApprovalRuleTemplateName`

Erforderlich, um den Namen einer Genehmigungsregelvorlage in einem AWS -Konto zu aktualisieren.

Ressource: *

Berechtigungen für Aktionen an einzelnen Dateien

Die folgenden Berechtigungen erlauben bzw. verweigern Aktionen auf einzelne Dateien in CodeCommit-Repositorys. Diese Berechtigungen beziehen sich nur auf Aktionen, die in der CodeCommit Konsole, der CodeCommit API ausgeführt werden, und auf Befehle, die mit der AWS CLI ausgeführt werden. Sie gelten nicht für ähnliche Aktionen, die mit dem Git-Protokoll ausgeführt werden können. Mit dem Befehl `git push` beispielsweise werden neue und geänderte Dateien im Push-Verfahren an ein CodeCommit-Repository mithilfe des Git-Protokolls übertragen. Sie wird durch keine Berechtigungen für den CodeCommit PutFile Vorgang beeinflusst.

CodeCommit API-Operationen und erforderliche Berechtigungen für Aktionen an einzelnen Dateien

[DeleteFile](#)

Aktion(en): `codecommit:DeleteFile`

Erforderlich, um eine bestimmte Datei aus einem bestimmten Zweig in einem CodeCommit Repository von der CodeCommit Konsole aus zu löschen.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetBlob

Aktion(en): `codecommit:GetBlob`

Erforderlich, um den codierten Inhalt einer einzelnen Datei in einem CodeCommit Repository von der CodeCommit Konsole aus anzuzeigen.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetFile

Aktion(en): `codecommit:GetFile`

Erforderlich, um den codierten Inhalt einer bestimmten Datei und ihrer Metadaten in einem CodeCommit Repository von der CodeCommit Konsole aus anzuzeigen.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetFolder

Aktion(en): `codecommit:GetFolder`

Erforderlich, um den Inhalt eines bestimmten Ordners in einem CodeCommit Repository von der CodeCommit Konsole aus anzuzeigen.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

PutFile

Aktion(en): `codecommit:PutFile`

Erforderlich, um einem CodeCommit Repository über die CodeCommit Konsole, CodeCommit API oder die eine neue oder geänderte Datei hinzuzufügen AWS CLI.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

Berechtigungen für Aktionen im Zusammenhang mit Kommentaren

Die folgenden Berechtigungen erlauben oder verbieten Aktionen für Kommentare in CodeCommit Repositories. Diese Berechtigungen beziehen sich auf Aktionen, die mit der CodeCommit Konsole und der CodeCommit API ausgeführt werden, sowie auf Befehle, die mit der ausgeführt werden. AWS CLI Weitere Berechtigungen für Kommentare in Pull-Anfragen finden Sie unter [Berechtigungen für Aktionen bei Pull-Requests](#).

CodeCommit API-Operationen und erforderliche Berechtigungen für Aktionen in Repositorys

DeleteCommentContent

Aktion(en): `codecommit:DeleteCommentContent`

Erforderlich zum Löschen des Inhalts eines Kommentars an einer Änderung, eine Datei oder einem Commit in einem Repository. Kommentare können nicht gelöscht werden, aber der Inhalt eines Kommentars kann entfernt werden, wenn der Benutzer über die entsprechende Berechtigung verfügt.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetComment

Aktion(en): `codecommit:GetComment`

Erforderlich, um Informationen über einen Kommentar zu einer Änderung, Datei oder einem Commit in einem CodeCommit Repository zurückzugeben.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetCommentReactions

Aktion(en): `codecommit:GetCommentReactions`

Erforderlich, um Informationen über Emoji-Reaktionen auf einen Kommentar zu einer Änderung, Datei oder einem Commit in einem CodeCommit Repository zurückzugeben.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetCommentsForComparedCommit

Aktion(en): `codecommit:GetCommentsForComparedCommit`

Erforderlich, um Informationen über Kommentare zurückzugeben, die beim Vergleich zwischen zwei Commits in einem CodeCommit Repository abgegeben wurden.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

PostCommentForComparedCommit

Aktion(en): `codecommit:PostCommentForComparedCommit`

Erforderlich zum Erstellen eines Kommentars zum Vergleich zwischen zwei Commits in einem CodeCommit-Repository.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

PostCommentReply

Aktion(en): `codecommit:PostCommentReply`

Erforderlich, um eine Antwort auf einen Kommentar zu einem Vergleich zwischen Commits oder zu einer Pull-Anfrage in einem CodeCommit Repository zu erstellen.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

PutCommentReaction

Aktion(en): `codecommit:PutCommentReaction`

Erforderlich, um auf einen Kommentar mit einem Emoji zu einem Commit oder einer Pull-Anfrage in einem CodeCommit Repository zu antworten.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

UpdateComment

Aktion(en): `codecommit:UpdateComment`

Erforderlich zum Bearbeiten eines Kommentars zu einem Vergleich zwischen Commits oder zu einer Pull-Anfrage. Kommentare können nur von ihrem Autor bearbeitet werden.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

Berechtigungen für Aktionen an festgeschriebenem Code

Die folgenden Berechtigungen erlauben oder verweigern Aktionen für festgeschriebenen Code CodeCommit-Repositories. Diese Berechtigungen beziehen sich auf Aktionen, die mit der CodeCommit Konsole und der CodeCommit API ausgeführt werden, sowie auf Befehle, die mit der AWS CLI ausgeführt werden. Sie gelten nicht für ähnliche Aktionen, die mit dem Git-Protokoll ausgeführt werden können. Beispiel: Der Befehl `git commit` erstellt einen Commit für einen Branch in einem Repository mithilfe des Git-Protokolls. Es wird durch keine Berechtigungen für den CodeCommit `CreateCommit` Vorgang beeinflusst.

Die ausdrückliche Verweigerung einiger dieser Berechtigungen kann zu unerwarteten Konsequenzen in der CodeCommit Konsole führen. Wird beispielsweise `GetTree` auf Deny gesetzt, wird verhindert, dass Benutzer durch die Inhalte eines Repositories in der Konsole navigieren. Benutzer werden jedoch nicht daran gehindert, die Inhalte einer Datei im Repository anzuzeigen (wenn sie

beispielsweise einen Link zu der Datei per E-Mail gesendet bekommen). Durch das Festlegen von `GetBlob` auf `Deny` wird verhindert, dass Benutzer die Inhalte von Dateien anzeigen. Benutzer werden jedoch nicht daran gehindert, die Struktur eines Repositorys zu durchsuchen. Durch das Festlegen von `GetCommit` auf `Deny` wird verhindert, dass Benutzer Details über Commits abrufen. Durch das Festlegen von `GetObjectIdentifier` auf `Deny` wird der Großteil der Funktionalität von Code-Browsing blockiert. Wenn Sie alle drei Aktionen `Deny` in einer Richtlinie auf festlegen, kann ein Benutzer mit dieser Richtlinie keinen Code in der CodeCommit Konsole durchsuchen.

CodeCommit API-Operationen und erforderliche Berechtigungen für Aktionen an festgeschriebenem Code

BatchGetCommits

Aktion(en): `codecommit:BatchGetCommits`

Erforderlich, um Informationen zu einem oder mehreren Commits in einem CodeCommit-Repository zurückzugeben. Dies ist nur eine IAM-Richtlinienberechtigung, keine API-Aktion, die Sie aufrufen können.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

[CreateCommit](#)

Aktion(en): `codecommit>CreateCommit`

Erforderlich zum Erstellen eines Commits.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

[GetCommit](#)

Aktion(en): `codecommit:GetCommit`

Erforderlich zum Abrufen von Informationen zu einem Commit.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetCommitHistory

Aktion(en): `codecommit:GetCommitHistory`

Erforderlich zum Abfragen von Informationen über den Verlauf von Commits in einem Repository. Dies ist nur eine IAM-Richtlinienberechtigung, keine API-Aktion, die Sie aufrufen können.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetDifferences

Aktion(en): `codecommit:GetDifferences`

Erforderlich zum Abfragen von Informationen über die Unterschiede in einem Commit-Spezifizierer (wie etwa ein Branch, ein Tag, HEAD, eine Commit-ID oder andere vollständig qualifizierte Referenzen).

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetObjectIdentifier

Aktion(en): `codecommit:GetObjectIdentifier`

Erforderlich zum Auflösen von Blobs, Strukturen und Commits zu ihrem Bezeichner. Dies ist nur eine IAM-Richtlinienberechtigung, keine API-Aktion, die Sie aufrufen können.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetReferences

Aktion(en): `codecommit:GetReferences`

Erforderlich für die Rückgabe aller Referenzen, wie etwa Branches und Tags. Dies ist nur eine IAM-Richtlinienberechtigung, keine API-Aktion, die Sie aufrufen können.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetTree

Aktion(en): `codecommit:GetTree`

Erforderlich, um den Inhalt einer bestimmten Struktur in einem CodeCommit Repository von der CodeCommit Konsole aus anzuzeigen. Dies ist nur eine IAM-Richtlinienberechtigung, keine API-Aktion, die Sie aufrufen können.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

Berechtigungen für Aktionen in Repositorys

Die folgenden Berechtigungen erlauben oder verweigern Aktionen in CodeCommit Repositorys. Diese Berechtigungen beziehen sich auf Aktionen, die mit der CodeCommit Konsole und der

CodeCommit API ausgeführt werden, sowie auf Befehle, die mit der ausgeführt werden. AWS CLI Sie gelten nicht für ähnliche Aktionen, die mit dem Git-Protokoll ausgeführt werden können.

CodeCommit API-Operationen und erforderliche Berechtigungen für Aktionen in Repositorys

BatchGetRepositories

Aktion(en): `codecommit:BatchGetRepositories`

Erforderlich, um Informationen über mehrere CodeCommit Repositorys abzurufen, die sich in einem Amazon Web Services Services-Konto befinden. In Resource müssen Sie die Namen aller CodeCommit Repositorys angeben, für die einem Benutzer Informationen erlaubt (oder verweigert) werden.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

CreateRepository

Aktion(en): `codecommit>CreateRepository`

Erforderlich, um ein CodeCommit Repository zu erstellen.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

DeleteRepository

Aktion(en): `codecommit>DeleteRepository`

Erforderlich, um ein CodeCommit Repository zu löschen.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetRepository

Aktion(en): `codecommit:GetRepository`

Erforderlich, um Informationen über ein einzelnes CodeCommit Repository abzurufen.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

ListRepositories

Aktion(en): `codecommit>ListRepositories`

Erforderlich, um eine Liste der Namen und IDs des Systems mehrerer CodeCommit Repositories für ein Amazon Web Services Services-Konto abzurufen. Der einzige erlaubte Wert für Resource für diese Aktion bezieht sich auf alle Repositories (*).

Ressource: *

UpdateRepositoryDescription

Aktion(en): codecommit:UpdateRepositoryDescription

Erforderlich, um die Beschreibung eines CodeCommit Repositories zu ändern.

Ressource: arn:aws:codecommit:*region*:*account-id*:*repository-name*

UpdateRepositoryName

Aktion(en): codecommit:UpdateRepositoryName

Erforderlich, um den Namen eines CodeCommit Repositories zu ändern. In Resource müssen Sie sowohl die CodeCommit Repositories angeben, die geändert werden dürfen, als auch die neuen Repository-Namen.

Ressource: arn:aws:codecommit:*region*:*account-id*:*repository-name*

Berechtigungen für Aktionen mit Tags

Die folgenden Berechtigungen erlauben oder verweigern Aktionen mit AWS Tags für CodeCommit Ressourcen.

CodeCommit API-Operationen und erforderliche Berechtigungen für Aktionen mit Tags

ListTagsForResource

Aktion(en): codecommit:ListTagsForResource

Erforderlich, um Informationen zu AWS Tags zurückzugeben, die für eine Ressource in konfiguriert sind CodeCommit.

Ressource: arn:aws:codecommit:*region*:*account-id*:*repository-name*

TagResource

Aktion(en): codecommit:TagResource

Erforderlich, um AWS Tags für ein Repository hinzuzufügen oder zu bearbeiten.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

UntagResource

Aktion(en): `codecommit:UntagResource`

Erforderlich, um AWS Tags aus einer Ressource in zu entfernen CodeCommit.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

Berechtigungen für Aktionen auf Triggern

Die folgenden Berechtigungen genehmigen oder verweigern Aktionen für Auslöser von CodeCommit-Repositorys.

CodeCommit API-Operationen und erforderliche Berechtigungen für Aktionen auf Triggern

GetRepositoryTriggers

Aktion(en): `codecommit:GetRepositoryTriggers`

Erforderlich für die Rückgabe von Informationen über für ein Repository konfigurierte Auslöser.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

PutRepositoryTriggers

Aktion(en): `codecommit:PutRepositoryTriggers`

Erforderlich zum Erstellen, Bearbeiten oder Löschen von Auslösern für ein Repository.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

TestRepositoryTriggers

Aktion(en): `codecommit:TestRepositoryTriggers`

Erforderlich zum Testen der Funktionalität eines Repository-Auslösers, indem Daten an das Thema oder die Funktion gesendet werden, das bzw. die für den Auslöser konfiguriert wurde.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

Berechtigungen für Aktionen bei der CodePipeline Integration

Um ein CodeCommit Repository in einer Quellaktion für eine Pipeline verwenden CodePipeline zu können, müssen Sie der Servicerolle für alle in der folgenden Tabelle aufgeführten Berechtigungen gewähren CodePipeline. Wenn diese Berechtigungen nicht in der Servicerolle angegeben oder auf **Deny** festgelegt sind, wird die Pipeline nicht automatisch ausgeführt, wenn eine Änderung an dem Repository vorgenommen wird, und Änderungen können nicht manuell veröffentlicht werden.

CodeCommit API-Operationen und erforderliche Berechtigungen für Integrationsaktionen CodePipeline

GetBranch

Aktion(en): `codecommit:GetBranch`

Erforderlich, um Details zu einem Branch in einem CodeCommit Repository abzurufen.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetCommit

Aktion(en): `codecommit:GetCommit`

Erforderlich zum Abrufen von Informationen zu einem Commit.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

UploadArchive

Aktion(en): `codecommit:UploadArchive`

Erforderlich, damit die Servicerolle für CodePipeline Repository-Änderungen in eine Pipeline hochladen kann. Dies ist nur eine IAM-Richtlinienberechtigung, keine API-Aktion, die Sie aufrufen können.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetUploadArchiveStatus

Aktion(en): `codecommit:GetUploadArchiveStatus`

Erforderlich zum Bestimmen des Status des Hochladens des Archivs, etwa ob es sich in Bearbeitung befindet, abgeschlossen ist, abgebrochen wurde oder ob ein Fehler aufgetreten ist. Dies ist nur eine IAM-Richtlinienberechtigung, keine API-Aktion, die Sie aufrufen können.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

CancelUploadArchive

Aktion(en): `codecommit:CancelUploadArchive`

Erforderlich zum Abbrechen des Uploads eines Archivs in eine Pipeline. Dies ist nur eine IAM-Richtlinienberechtigung, keine API-Aktion, die Sie aufrufen können.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

Wie AWS CodeCommit funktioniert mit IAM

Bevor Sie IAM verwenden, um den Zugriff auf zu verwalten CodeCommit, sollten Sie wissen, mit welchen IAM-Funktionen Sie verwenden können. CodeCommit Einen allgemeinen Überblick darüber, wie CodeCommit und andere AWS Dienste mit IAM funktionieren, finden Sie im IAM-Benutzerhandbuch unter [AWS Services That Work with IAM](#).

Themen

- [Bedingungsschlüssel](#)
- [Beispiele](#)

Bedingungsschlüssel

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer auf was Zugriff hat. Das heißt, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Das Element `Condition` (oder `Condition block`) ermöglicht Ihnen die Angabe der Bedingungen, unter denen eine Anweisung wirksam ist. Das Element `Condition` ist optional. Sie können bedingte Ausdrücke erstellen, die [Bedingungsoperatoren](#) verwenden, z. B. `ist gleich` oder `kleiner als`, damit die Bedingung in der Richtlinie mit Werten in der Anforderung übereinstimmt.

Wenn Sie mehrere `Condition`-Elemente in einer Anweisung oder mehrere Schlüssel in einem einzelnen `Condition`-Element angeben, wertet AWS diese mittels einer logischen AND-Operation aus. Wenn Sie mehrere Werte für einen einzelnen Bedingungsschlüssel angeben, AWS wertet die Bedingung mithilfe einer logischen OR Operation aus. Alle Bedingungen müssen erfüllt werden, bevor die Berechtigungen der Anweisung gewährt werden.

Sie können auch Platzhaltervariablen verwenden, wenn Sie Bedingungen angeben. Beispielsweise können Sie einem IAM-Benutzer die Berechtigung für den Zugriff auf eine Ressource nur dann gewähren, wenn sie mit dessen IAM-Benutzernamen gekennzeichnet ist. Weitere Informationen finden Sie unter [IAM-Richtlinienelemente: Variablen und Tags](#) im IAM-Benutzerhandbuch.

AWS unterstützt globale Bedingungsschlüssel und dienstspezifische Bedingungsschlüssel. Eine Übersicht aller AWS globalen Bedingungsschlüssel finden Sie unter [Kontextschlüssel für AWS globale Bedingungen](#) im IAM-Benutzerhandbuch.

CodeCommit definiert seinen eigenen Satz von Bedingungsschlüsseln und unterstützt auch die Verwendung einiger globaler Bedingungsschlüssel. Eine Übersicht aller AWS globalen Bedingungsschlüssel finden Sie unter [AWS Globale Bedingungskontextschlüssel](#) im IAM-Benutzerhandbuch.

Einige CodeCommit Aktionen unterstützen den `codecommit:References` Bedingungsschlüssel. Eine Beispielrichtlinie, die diesen Schlüssel verwendet, finden Sie unter [Beispiel 4: Aktionen für Branches verweigern oder zulassen](#).

Eine Liste der CodeCommit Bedingungsschlüssel finden Sie unter [Bedingungsschlüssel für AWS CodeCommit](#) im IAM-Benutzerhandbuch. Informationen zu den Aktionen und Ressourcen, mit denen Sie einen Bedingungsschlüssel verwenden können, finden Sie unter [Definierte Aktionen von AWS CodeCommit](#).

Beispiele

Beispiele für CodeCommit identitätsbasierte Richtlinien finden Sie unter [AWS CodeCommit Beispiele für identitätsbasierte Richtlinien](#)

Ressourcenbasierte CodeCommit-Richtlinien

CodeCommit unterstützt keine ressourcenbasierten Richtlinien.

Autorisierung auf der Grundlage von Tags CodeCommit

Sie können Tags an CodeCommit Ressourcen anhängen oder Tags in einer Anfrage an übergeben CodeCommit. Um den Zugriff auf der Grundlage von Tags zu steuern, geben Sie im Bedingungelement einer [Richtlinie Tag-Informationen](#) an, indem Sie die Schlüssel `codecommit:ResourceTag/key-name`, `aws:RequestTag/key-name`, oder Bedingung `aws:TagKeys` verwenden. Weitere Informationen zum Markieren von CodeCommit Ressourcen

finden Sie unter [Beispiel 5: Aktionen für Repositorys mit Tags verweigern oder zulassen](#). Weitere Informationen zu Tagging-Strategien finden Sie unter [AWS Tagging Resources](#).

CodeCommit unterstützt auch Richtlinien, die auf Sitzungs-Tags basieren. Weitere Informationen finden Sie unter [Sitzungs-Tags](#)

Verwendung von Tags zur Bereitstellung von Identitätsinformationen in CodeCommit

CodeCommit unterstützt die Verwendung von Sitzungs-Tags, bei denen es sich um Schlüssel-Wert-Paarattribute handelt, die Sie übergeben, wenn Sie eine IAM-Rolle übernehmen, temporäre Anmeldeinformationen verwenden oder einen Benutzer in () verbinden. AWS Security Token Service AWS STS Sie können Tags auch einem IAM-Benutzer zuordnen. Sie können die in diesen Tags enthaltenen Informationen verwenden, um leichter zu ermitteln, wer eine Änderung vorgenommen oder ein Ereignis verursacht hat. CodeCommit schließt die Werte für Tags mit den folgenden Schlüsselnamen in CodeCommit Ereignisse ein:

Tastename	Wert
displayName	Der für den Benutzer lesbare Name, der angezeigt und mit dem Benutzer verknüpft werden soll (z. B. Mary Major oder Saanvi Sarkar).
emailAddress	Die E-Mail-Adresse, die für den Benutzer angezeigt und mit ihm verknüpft werden soll (z. B. mary_major@example.com oder saanvi_sarkar@example.com).

Wenn diese Informationen bereitgestellt werden, werden sie CodeCommit in Veranstaltungen aufgenommen, die an Amazon EventBridge und Amazon CloudWatch Events gesendet werden. Weitere Informationen finden Sie unter [Überwachung von CodeCommit Ereignissen in Amazon EventBridge und Amazon CloudWatch Events](#).

Um die Sitzungsmarkierung zu verwenden, müssen Rollen Richtlinien enthalten, bei denen die `sts:TagSession`-Berechtigung auf `Allow` gesetzt ist. Wenn Sie den Verbundzugriff verwenden, können Sie den Anzeigenamen und die E-Mail-Tag-Informationen im Rahmen Ihrer Einrichtung konfigurieren. Wenn Sie beispielsweise Azure Active Directory verwenden, können Sie die folgenden Antragsinformationen angeben:

Name des Antrags	Wert
<code>https://aws.amazon.com/SAML/Attributes/PrincipalTag:displayName</code>	<code>user.displayName</code>
<code>https://aws.amazon.com/SAML/Attributes/PrincipalTag:emailAddress</code>	<code>user.mail</code>

Sie können die verwenden AWS CLI , um Sitzungs-Tags zu übergeben `displayName` und zu `emailAddress` verwenden `AssumeRole`. Beispielsweise *Mary Major* könnte ein Benutzer, der eine Rolle mit dem Namen *Developer* Wer möchte ihm seinen Namen zuordnen möchten, den `assume-role` Befehl ähnlich dem folgenden verwenden:

```
aws sts assume-role \  
--role-arn arn:aws:iam::123456789012:role/Developer \  
--role-session-name Mary-Major \  
--tags Key=displayName,Value="Mary Major" \  
Key=emailAddress,Value="mary_major@example.com" \  
--external-id Example987
```

Weitere Informationen finden Sie unter [AssumeRole](#).

Sie können den Vorgang `AssumeRoleWithSAML` verwenden, um einen Satz temporärer Anmeldeinformationen zurückzugeben, die die Tags `emailAddress` und `displayName` enthalten. Sie können diese Tags verwenden, wenn Sie auf CodeCommit-Repositorys zugreifen. Dies setzt voraus, dass Ihr Unternehmen oder Ihre Gruppe Ihre SAML-Lösung eines Drittanbieters bereits integriert hat. AWS Wenn dies der Fall ist, können Sie SAML-Attribute als Sitzungs-Tags übergeben. Wenn Sie beispielsweise Identitätsattribute für einen Anzeigenamen und eine E-Mail-Adresse für einen Benutzer übergeben möchten, der *Saanvi Sarkar* als Sitzungs-Tags benannt ist:

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:displayName">  
  <AttributeValue>Saanvi Sarkar</AttributeValue>  
</Attribute>  
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:emailAddress">  
  <AttributeValue>saanvi_sarkar@example.com</AttributeValue>  
</Attribute>
```

Weitere Informationen finden Sie unter [Übergeben von Sitzungs-Tags mithilfe von AssumeRoleWithSAML](#).

Sie können den Vorgang AssumeRoleWithIdentity verwenden, um einen Satz temporärer Anmeldeinformationen zurückzugeben, die die Tags emailAddress und displayName enthalten. Sie können diese Tags verwenden, wenn Sie auf CodeCommit-Repositorys zugreifen. Um Sitzungs-Tags von OpenID Connect (OIDC) zu übergeben, müssen Sie die Sitzungs-Tags in das JSON Web Token (JWT) einbeziehen. Zum Beispiel enthält das dekodierte JWP-Token, das für den Aufruf verwendet wird AssumeRoleWithWebIdentity, die Tags displayName und die emailAddress Sitzungs-Tags für einen Benutzer mit dem Namen: *Li Juan*

```
{
  "sub": "lijuan",
  "aud": "ac_oic_client",
  "jti": "ZYUCeREXAMPLE",
  "iss": "https://xyz.com",
  "iat": 1566583294,
  "exp": 1566583354,
  "auth_time": 1566583292,
  "https://aws.amazon.com/tags": {
    "principal_tags": {
      "displayName": ["Li Juan"],
      "emailAddress": ["li_juan@example.com"],
    },
    "transitive_tag_keys": [
      "displayName",
      "emailAddress"
    ]
  }
}
```

Weitere Informationen finden Sie unter [Übergeben von Sitzungs-Tags mithilfe von AssumeRoleWithWebIdentity](#).

Sie können den Vorgang GetFederationToken verwenden, um einen Satz temporärer Anmeldeinformationen zurückzugeben, die die Tags emailAddress und displayName enthalten. Sie können diese Tags verwenden, wenn Sie auf CodeCommit-Repositorys zugreifen. Um beispielsweise den zu verwenden, AWS CLI um ein Verbund-Token abzurufen, das die emailAddress Tags displayName und enthält:

```
aws sts get-federation-token \
```

```
--name my-federated-user \  
--tags key=displayName,value="Nikhil Jayashankar"  
      key=emailAddress,value=nikhil_jayashankar@example.com
```

Weitere Informationen finden Sie unter [Übergeben von Sitzungs-Tags mithilfe von GetFederationToken](#).

CodeCommit IAM-Rollen

Eine [IAM-Rolle](#) ist eine Entität in Ihrem Amazon-Web-Services-Konto mit spezifischen Berechtigungen.

Verwenden temporärer Anmeldeinformationen mit CodeCommit

Sie können temporäre Anmeldeinformationen verwenden, um sich über einen Verbund anzumelden, eine IAM-Rolle anzunehmen oder eine kontenübergreifende Rolle anzunehmen. Sie erhalten temporäre Sicherheitsanmeldedaten, indem Sie AWS STS API-Operationen wie [AssumeRole](#) oder aufrufen [GetFederationToken](#).

CodeCommit unterstützt die Verwendung temporärer Anmeldeinformationen. Weitere Informationen finden Sie unter [Verbindung zu AWS CodeCommit Repositorys mit rotierenden Anmeldeinformationen herstellen](#).

Service-verknüpfte Rollen

Mit [dienstbezogenen Rollen](#) können AWS Dienste auf Ressourcen in anderen Diensten zugreifen, um eine Aktion in Ihrem Namen auszuführen. Serviceverknüpfte Rollen werden in Ihrem IAM-Konto angezeigt und gehören zum Service. Ein IAM-Administrator kann die Berechtigungen für serviceverknüpfte Rollen anzeigen, aber nicht bearbeiten.

CodeCommit verwendet keine dienstbezogenen Rollen.

Service Rollen

Dieses Feature ermöglicht einem Service das Annehmen einer [Servicerolle](#) in Ihrem Namen. Diese Rolle gewährt dem Service Zugriff auf Ressourcen in anderen Diensten, um eine Aktion in Ihrem Namen auszuführen. Servicerollen werden in Ihrem IAM-Konto angezeigt und gehören zum Konto. Dies bedeutet, dass ein IAM-Administrator die Berechtigungen für diese Rolle ändern kann. Dies kann jedoch die Funktionalität des Dienstes beeinträchtigen.

CodeCommit verwendet keine Servicerollen.

AWS CodeCommit Beispiele für identitätsbasierte Richtlinien

IAM-Benutzer besitzen keine Berechtigungen zum Erstellen oder Ändern von CodeCommit - Ressourcen. Sie können auch keine Aufgaben mit der AWS Management Console AWS CLI, oder AWS API ausführen. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern und Rollen die Berechtigung zum Ausführen bestimmter API-Operationen für die angegebenen Ressourcen gewähren, die diese benötigen. Der Administrator muss diese Richtlinien anschließend den IAM-Benutzern oder -Gruppen anfügen, die diese Berechtigungen benötigen.

Beispiele für diese Richtlinien finden Sie unter:

- [Beispiel 1: Erlauben Sie einem Benutzer, Operationen CodeCommit in einem einzigen AWS-Region](#)
- [Beispiel 2: Erlaube einem Benutzer, Git für ein einzelnes Repository zu verwenden](#)
- [Beispiel 3: Erlaubt einem Benutzer, der von einem bestimmten IP-Adressbereich aus eine Verbindung herstellt, Zugriff auf ein Repository](#)
- [Beispiel 4: Aktionen für Branches verweigern oder zulassen](#)
- [Beispiel 5: Aktionen für Repositories mit Tags verweigern oder zulassen](#)
- [Konfiguriere den kontoübergreifenden Zugriff auf ein AWS CodeCommit Repository mithilfe von Rollen](#)

Informationen dazu, wie Sie unter Verwendung dieser beispielhaften JSON-Richtliniendokumente eine identitätsbasierte IAM-Richtlinie erstellen, finden Sie unter [Erstellen von Richtlinien auf der JSON-Registerkarte](#) im IAM-Benutzerhandbuch.

Themen

- [Bewährte Methoden für Richtlinien](#)
- [Verwenden der CodeCommit -Konsole](#)
- [Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer](#)
- [Anzeige CodeCommit repositories basierend auf Tags](#)

Bewährte Methoden für Richtlinien

Identitätsbasierte Richtlinien legen fest, ob jemand CodeCommit Ressourcen in Ihrem Konto erstellen, darauf zugreifen oder sie löschen kann. Dies kann zusätzliche Kosten für Ihr verursachen

AWS-Konto. Befolgen Sie beim Erstellen oder Bearbeiten identitätsbasierter Richtlinien die folgenden Anleitungen und Empfehlungen:

- Beginnen Sie mit AWS verwalteten Richtlinien und wechseln Sie zu Berechtigungen mit den geringsten Rechten — Verwenden Sie die AWS verwalteten Richtlinien, die Berechtigungen für viele gängige Anwendungsfälle gewähren, um Ihren Benutzern und Workloads zunächst Berechtigungen zu gewähren. Sie sind in Ihrem verfügbar. AWS-Konto Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom AWS Kunden verwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind. Weitere Informationen finden Sie unter [AWS -verwaltete Richtlinien](#) oder [AWS -verwaltete Richtlinien für Auftrags-Funktionen](#) im IAM-Benutzerhandbuch.
- Anwendung von Berechtigungen mit den geringsten Rechten – Wenn Sie mit IAM-Richtlinien Berechtigungen festlegen, gewähren Sie nur die Berechtigungen, die für die Durchführung einer Aufgabe erforderlich sind. Sie tun dies, indem Sie die Aktionen definieren, die für bestimmte Ressourcen unter bestimmten Bedingungen durchgeführt werden können, auch bekannt als die geringsten Berechtigungen. Weitere Informationen zur Verwendung von IAM zum Anwenden von Berechtigungen finden Sie unter [Richtlinien und Berechtigungen in IAM](#) im IAM-Benutzerhandbuch.
- Verwenden von Bedingungen in IAM-Richtlinien zur weiteren Einschränkung des Zugriffs – Sie können Ihren Richtlinien eine Bedingung hinzufügen, um den Zugriff auf Aktionen und Ressourcen zu beschränken. Sie können beispielsweise eine Richtlinienbedingung schreiben, um festzulegen, dass alle Anforderungen mithilfe von SSL gesendet werden müssen. Sie können auch Bedingungen verwenden, um Zugriff auf Serviceaktionen zu gewähren, wenn diese für einen bestimmten Zweck verwendet werden AWS-Service, z. AWS CloudFormation B. Weitere Informationen finden Sie unter [IAM-JSON-Richtlinienelemente: Bedingung](#) im IAM-Benutzerhandbuch.
- Verwenden von IAM Access Analyzer zur Validierung Ihrer IAM-Richtlinien, um sichere und funktionale Berechtigungen zu gewährleisten – IAM Access Analyzer validiert neue und vorhandene Richtlinien, damit die Richtlinien der IAM-Richtliniensprache (JSON) und den bewährten IAM-Methoden entsprechen. IAM Access Analyzer stellt mehr als 100 Richtlinienprüfungen und umsetzbare Empfehlungen zur Verfügung, damit Sie sichere und funktionale Richtlinien erstellen können. Weitere Informationen finden Sie unter [Richtlinienvvalidierung mit IAM Access Analyzer](#) im IAM-Benutzerhandbuch.
- Multi-Faktor-Authentifizierung (MFA) erforderlich — Wenn Sie ein Szenario haben, das IAM-Benutzer oder einen Root-Benutzer in Ihrem System erfordert AWS-Konto, aktivieren Sie MFA für zusätzliche Sicherheit. Um MFA beim Aufrufen von API-Vorgängen anzufordern, fügen Sie Ihren

Richtlinien MFA-Bedingungen hinzu. Weitere Informationen finden Sie unter [Sicherer API-Zugriff mit MFA](#) im IAM-Benutzerhandbuch.

Weitere Informationen zu bewährten Methoden in IAM finden Sie unter [Bewährte Methoden für die Sicherheit in IAM](#) im IAM-Benutzerhandbuch.

Verwenden der CodeCommit -Konsole

Um auf die AWS CodeCommit Konsole zugreifen zu können, benötigen Sie ein Mindestmaß an Berechtigungen. Diese Berechtigungen müssen es Ihnen ermöglichen, Informationen zu den CodeCommit Ressourcen in Ihrem Amazon Web Services Services-Konto aufzulisten und einzusehen. Wenn Sie eine identitätsbasierte Richtlinie erstellen, die strenger ist als die mindestens erforderlichen Berechtigungen, funktioniert die Konsole nicht wie vorgesehen für Entitäten (IAM-Benutzer oder -Rollen) mit dieser Richtlinie.

Um sicherzustellen, dass diese Entitäten die CodeCommit Konsole weiterhin verwenden können, fügen Sie den Entitäten außerdem die folgende AWS verwaltete Richtlinie hinzu. Weitere Informationen finden Sie unter [Hinzufügen von Berechtigungen](#) zu einem Benutzer im IAM-Benutzerhandbuch:

Weitere Informationen finden Sie unter [Verwendung identitätsbasierter Richtlinien \(IAM-Richtlinien\) für CodeCommit](#).

Sie müssen Benutzern, die nur die API AWS CLI oder die AWS API aufrufen, keine Mindestberechtigungen für die Konsole gewähren. Stattdessen sollten Sie nur Zugriff auf die Aktionen zulassen, die den API-Operation entsprechen, die Sie ausführen möchten.

Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer

In diesem Beispiel wird gezeigt, wie Sie eine Richtlinie erstellen, die IAM-Benutzern die Berechtigung zum Anzeigen der eingebundenen Richtlinien und verwalteten Richtlinien gewährt, die ihrer Benutzeridentität angefügt sind. Diese Richtlinie umfasst Berechtigungen zum Ausführen dieser Aktion auf der Konsole oder programmgesteuert mithilfe der API AWS CLI oder AWS .

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewOwnUserInfo",
```

```

        "Effect": "Allow",
        "Action": [
            "iam:GetUserPolicy",
            "iam:ListGroupsForUser",
            "iam:ListAttachedUserPolicies",
            "iam:ListUserPolicies",
            "iam:GetUser"
        ],
        "Resource": ["arn:aws:iam::*:user/${aws:username}"]
    },
    {
        "Sid": "NavigateInConsole",
        "Effect": "Allow",
        "Action": [
            "iam:GetGroupPolicy",
            "iam:GetPolicyVersion",
            "iam:GetPolicy",
            "iam:ListAttachedGroupPolicies",
            "iam:ListGroupPolicies",
            "iam:ListPolicyVersions",
            "iam:ListPolicies",
            "iam:ListUsers"
        ],
        "Resource": "*"
    }
]
}

```

Anzeige CodeCommit *repositories* basierend auf Tags

Sie können in Ihrer identitätsbasierten Richtlinie Bedingungen für die Steuerung des Zugriffs auf CodeCommit -Ressourcen auf der Basis von Tags verwenden. Eine Beispielrichtlinie, die die Vorgehensweise veranschaulicht, finden Sie unter [Beispiel 5: Aktionen für Repositorys mit Tags verweigern oder zulassen](#).

Weitere Informationen finden Sie unter [IAM-JSON-Richtlinienelemente: Bedingung](#) im IAM-Benutzerhandbuch.

Problembehebung bei AWS CodeCommit Identität und Zugriff

Verwenden Sie die folgenden Informationen, um häufig auftretende Probleme zu diagnostizieren und zu beheben, die bei der Arbeit mit CodeCommit und IAM auftreten können.

Themen

- [Ich bin nicht berechtigt, eine Aktion durchzuführen in CodeCommit](#)
- [Ich bin nicht berechtigt, iam auszuführen: PassRole](#)
- [Ich möchte meine Zugriffsschlüssel anzeigen](#)
- [Ich bin Administrator und möchte anderen den Zugriff ermöglichen CodeCommit](#)
- [Ich möchte Personen außerhalb meines Amazon Web Services Services-Kontos den Zugriff auf meine CodeCommit Ressourcen ermöglichen](#)

Ich bin nicht berechtigt, eine Aktion durchzuführen in CodeCommit

Wenn Ihnen AWS Management Console mitgeteilt wird, dass Sie nicht berechtigt sind, eine Aktion durchzuführen, müssen Sie sich an Ihren Administrator wenden, um Unterstützung zu erhalten. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Weitere Informationen finden Sie unter [Erforderliche Berechtigungen für die Verwendung der CodeCommit-Konsole](#)

Ich bin nicht berechtigt, iam auszuführen: PassRole

Wenn Sie die Fehlermeldung erhalten, dass Sie nicht zum Durchführen der `iam:PassRole`-Aktion autorisiert sind, müssen Ihre Richtlinien aktualisiert werden, um eine Rolle an CodeCommit übergeben zu können.

Einige AWS-Services ermöglichen es Ihnen, eine bestehende Rolle an diesen Dienst zu übergeben, anstatt eine neue Servicerolle oder eine dienstverknüpfte Rolle zu erstellen. Hierzu benötigen Sie Berechtigungen für die Übergabe der Rolle an den Dienst.

Der folgende Beispielfehler tritt auf, wenn ein IAM-Benutzer mit dem Namen `marymajor` versucht, die Konsole zu verwenden, um eine Aktion in CodeCommit auszuführen. Die Aktion erfordert jedoch, dass der Service über Berechtigungen verfügt, die durch eine Servicerolle gewährt werden. Mary besitzt keine Berechtigungen für die Übergabe der Rolle an den Dienst.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
iam:PassRole
```

In diesem Fall müssen die Richtlinien von Mary aktualisiert werden, um die Aktion `iam:PassRole` ausführen zu können.

Wenn Sie Hilfe benötigen, wenden Sie sich an Ihren AWS Administrator. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen odzur Verfügung gestellt.

Ich möchte meine Zugriffsschlüssel anzeigen

Nachdem Sie Ihre IAM-Benutzerzugriffsschlüssel erstellt haben, können Sie Ihre Zugriffsschlüssel-ID jederzeit anzeigen. Sie können Ihren geheimen Zugriffsschlüssel jedoch nicht erneut anzeigen. Wenn Sie den geheimen Zugriffsschlüssel verlieren, müssen Sie ein neues Zugriffsschlüsselpaar erstellen.

Zugriffsschlüssel bestehen aus zwei Teilen: einer Zugriffsschlüssel-ID (z. B. AKIAIOSFODNN7EXAMPLE) und einem geheimen Zugriffsschlüssel (z. B. wJa1rXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY). Ähnlich wie bei Benutzernamen und Passwörtern müssen Sie die Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel zusammen verwenden, um Ihre Anforderungen zu authentifizieren. Verwalten Sie Ihre Zugriffsschlüssel so sicher wie Ihren Benutzernamen und Ihr Passwort.

Important

Geben Sie Ihre Zugriffsschlüssel nicht an Dritte weiter, auch nicht für die [Suche nach Ihrer kanonischen Benutzer-ID](#). Auf diese Weise können Sie jemandem dauerhaften Zugriff auf Ihre gewähren AWS-Konto.

Während der Erstellung eines Zugriffsschlüsselpaars werden Sie aufgefordert, die Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel an einem sicheren Speicherort zu speichern. Der geheime Zugriffsschlüssel ist nur zu dem Zeitpunkt verfügbar, an dem Sie ihn erstellen. Wenn Sie Ihren geheimen Zugriffsschlüssel verlieren, müssen Sie Ihrem IAM-Benutzer neue Zugriffsschlüssel hinzufügen. Sie können maximal zwei Zugriffsschlüssel besitzen. Wenn Sie bereits zwei Zugriffsschlüssel besitzen, müssen Sie ein Schlüsselpaar löschen, bevor Sie ein neues erstellen. Anweisungen hierfür finden Sie unter [Verwalten von Zugriffsschlüsseln](#) im IAM-Benutzerhandbuch.

Ich bin Administrator und möchte anderen den Zugriff ermöglichen CodeCommit

Um anderen den Zugriff zu ermöglichen CodeCommit, müssen Sie den Personen oder Anwendungen, die Zugriff benötigen, die entsprechenden Berechtigungen erteilen. Wenn Sie Personen und Anwendungen verwalten, weisen Sie Benutzern oder Gruppen Berechtigungssätze zu, um deren Zugriffsebene zu definieren. AWS IAM Identity Center Mit Berechtigungssätzen werden automatisch IAM-Richtlinien erstellt und den IAM-Rollen zugewiesen, die der Person

oder Anwendung zugeordnet sind. Weitere Informationen finden Sie im AWS IAM Identity Center Benutzerhandbuch unter [Berechtigungssätze](#).

Wenn Sie IAM Identity Center nicht verwenden, müssen Sie IAM-Entitäten (Benutzer oder Rollen) für die Personen oder Anwendungen erstellen, die Zugriff benötigen. Anschließend müssen Sie der Entität eine Richtlinie anfügen, die dieser die korrekten Berechtigungen in CodeCommit gewährt. Nachdem die Berechtigungen erteilt wurden, stellen Sie dem Benutzer oder Anwendungsentwickler die Anmeldeinformationen zur Verfügung. Sie werden diese Anmeldeinformationen für den Zugriff verwenden AWS. Weitere Informationen zum Erstellen von IAM-Benutzern, -Gruppen, -Richtlinien und -Berechtigungen finden Sie im [IAM-Benutzerhandbuch unter IAM-Identitäten sowie Richtlinien und Berechtigungen in IAM](#).

Ich möchte Personen außerhalb meines Amazon Web Services Services-Kontos den Zugriff auf meine CodeCommit Ressourcen ermöglichen

Weitere Informationen finden Sie unter [Konfiguriere den kontoübergreifenden Zugriff auf ein AWS CodeCommit Repository mithilfe von Rollen](#).

Resilienz in AWS CodeCommit

Die AWS globale Infrastruktur basiert AWS-Regionen auf Availability Zones. AWS-Regionen bieten mehrere physisch getrennte und isolierte Availability Zones, die über Netzwerke mit niedriger Latenz, hohem Durchsatz und hoher Redundanz miteinander verbunden sind. Mithilfe von Availability Zones können Sie Anwendungen und Datenbanken erstellen und ausführen, die automatisch Failover zwischen Zonen ausführen, ohne dass es zu Unterbrechungen kommt. Availability Zones sind besser verfügbar, fehlertoleranter und skalierbarer als herkömmliche Infrastrukturen mit einem oder mehreren Rechenzentren.

Eine Vorlage für ein CodeCommit Repository oder eine CodeCommit Genehmigungsregel ist dort vorhanden AWS-Region , wo sie erstellt wurde. Weitere Informationen finden Sie unter [Regionen und Git-Verbindungsendpunkte für AWS CodeCommit](#). Um die Stabilität von Repositorys zu erhöhen, kannst du deinen Git-Client so konfigurieren, dass er auf zwei Repositorys gleichzeitig pusht. Weitere Informationen finden Sie unter [Push Commits in ein zusätzliches Git-Repository](#).

[Weitere Informationen zu Availability Zones AWS-Regionen und Availability Zones finden Sie unter Globale Infrastruktur AWS .](#)

Infrastruktursicherheit in AWS CodeCommit

Als verwalteter Service AWS CodeCommit ist er durch die AWS globalen Netzwerksicherheitsverfahren geschützt, die im Whitepaper [Amazon Web Services: Sicherheitsprozesse im Überblick](#) beschrieben sind.

Sie verwenden AWS veröffentlichte API-Aufrufe, um CodeCommit über das Netzwerk darauf zuzugreifen. Kunden müssen Transport Layer Security (TLS) 1.0 oder neuer unterstützen. Wir empfehlen TLS 1.2 oder höher. Clients müssen außerdem Verschlüsselungssammlungen mit PFS (Perfect Forward Secrecy) wie DHE (Ephemeral Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman) unterstützen. Die meisten modernen Systemen wie Java 7 und höher unterstützen diese Modi.

Anforderungen müssen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signiert sein, der mit einem IAM-Prinzipal verknüpft ist. Alternativ können Sie mit [AWS Security Token Service](#) (AWS STS) temporäre Sicherheitsanmeldeinformationen erstellen, um die Anforderungen zu signieren.

Sie können diese API-Operationen von jedem Netzwerkstandort aus aufrufen, CodeCommit unterstützt jedoch Einschränkungen, die auf der Quell-IP-Adresse basieren. Sie können auch CodeCommit Richtlinien verwenden, um den Zugriff von bestimmten Amazon Virtual Private Cloud (Amazon VPC) -Endpunkten oder bestimmten zu kontrollieren. VPCs Dadurch wird der Netzwerkzugriff auf eine bestimmte CodeCommit Ressource effektiv nur von der spezifischen VPC im AWS Netzwerk isoliert.

Weitere Informationen finden Sie hier:

- [Beispiel 1: Erlauben Sie einem Benutzer, Operationen CodeCommit in einem einzigen AWS-Region](#)
- [Beispiel 3: Erlaubt einem Benutzer, der von einem bestimmten IP-Adressbereich aus eine Verbindung herstellt, Zugriff auf ein Repository](#)
- [Verwendung AWS CodeCommit mit VPC-Endpunkten mit Schnittstelle](#)

Überwachung AWS CodeCommit

Die Überwachung ist ein wichtiger Bestandteil der Aufrechterhaltung der Zuverlässigkeit, Verfügbarkeit und Leistung Ihrer CodeCommit anderen AWS Lösungen. AWS bietet die folgenden Überwachungstools, mit denen Sie beobachten CodeCommit, melden können, wenn etwas nicht stimmt, und gegebenenfalls automatische Maßnahmen ergreifen können:

- Amazon EventBridge kann verwendet werden, um Ihre AWS Services zu automatisieren und automatisch auf Systemereignisse wie Probleme mit der Anwendungsverfügbarkeit oder Ressourcenänderungen zu reagieren. Ereignisse im AWS Rahmen von Services werden nahezu EventBridge in Echtzeit zugestellt. Sie können einfache Regeln schreiben, um anzugeben, welche Ereignisse für Sie interessant sind und welche automatisierten Aktionen ausgeführt werden sollen, wenn ein Ereignis mit einer Regel übereinstimmt. Weitere Informationen finden Sie im [EventBridge Amazon-Benutzerhandbuch](#) und [Überwachung von CodeCommit Ereignissen in Amazon EventBridge und Amazon CloudWatch Events](#).
- Amazon CloudWatch Events bietet einen Stream von Systemereignissen, die Änderungen an AWS Ressourcen beschreiben, nahezu in Echtzeit. CloudWatch Events ermöglicht automatisiertes ereignisgesteuertes Rechnen, da Sie Regeln schreiben können, die auf bestimmte Ereignisse achten und automatisierte Aktionen in anderen AWS Diensten auslösen können, wenn diese Ereignisse eintreten. Weitere Informationen finden Sie im [Amazon CloudWatch Events-Benutzerhandbuch](#) und [Überwachung von CodeCommit Ereignissen in Amazon EventBridge und Amazon CloudWatch Events](#).
- Amazon CloudWatch Logs kann verwendet werden, um Ihre Protokolldateien und andere Quellen zu überwachen, zu speichern CloudTrail und darauf zuzugreifen. CloudWatch Logs kann Informationen in den Protokolldateien überwachen und Sie benachrichtigen, wenn bestimmte Schwellenwerte erreicht werden. Sie können Ihre Protokolldaten auch in einem sehr robusten Speicher archivieren. Weitere Informationen finden Sie im [Amazon CloudWatch Logs-Benutzerhandbuch](#).
- AWS CloudTrail erfasst API-Aufrufe und zugehörige Ereignisse, die von oder im Namen Ihres Amazon Web Services Services-Kontos getätigt wurden, und übermittelt die Protokolldateien an einen von Ihnen angegebenen Amazon S3 S3-Bucket. Sie können die Benutzer und Konten, die AWS aufgerufen haben, identifizieren, sowie die Quell-IP-Adresse, von der diese Aufrufe stammen, und den Zeitpunkt der Aufrufe ermitteln. Weitere Informationen finden Sie im [AWS CloudTrail Benutzerhandbuch](#) und [AWS CodeCommit API-Aufrufe protokollieren mit AWS CloudTrail](#).

Überwachung von CodeCommit Ereignissen in Amazon EventBridge und Amazon CloudWatch Events

Sie können AWS CodeCommit Ereignisse überwachen EventBridge, wodurch ein Stream von Echtzeitdaten aus Ihren eigenen Anwendungen, software-as-a-service (SaaS) -Anwendungen und AWS Diensten bereitgestellt wird. EventBridge leitet diese Daten an Ziele wie AWS Lambda Amazon Simple Notification Service weiter. Diese Ereignisse sind identisch mit denen, die in Amazon CloudWatch Events erscheinen. Amazon Events liefert einen Stream von Systemereignissen, die Änderungen an AWS Ressourcen beschreiben, nahezu in Echtzeit.

Die folgenden Beispiele zeigen Ereignisse für CodeCommit.

Note

CodeCommit unterstützt die Bereitstellung von emailAddress Informationen displayName und die in Sitzungs-Tags enthaltenen Informationen in Ereignissen, sofern diese Informationen verfügbar sind. Weitere Informationen finden Sie unter [Sitzungs-Tags](#) und [Verwendung von Tags zur Bereitstellung von Identitätsinformationen in CodeCommit](#).

Themen

- [referenceCreated-Ereignis](#)
- [referenceUpdated-Ereignis](#)
- [referenceDeleted-Ereignis](#)
- [unreferencedMergeCommitEreignis wurde erstellt](#)
- [commentOnCommitEreignis wurde erstellt](#)
- [commentOnCommitEreignis wurde aktualisiert](#)
- [commentOnPullRequestCreated Ereignis](#)
- [commentOnPullRequestUpdated Ereignis](#)
- [pullRequestCreated Ereignis](#)
- [pullRequestSourceBranchUpdated Ereignis](#)
- [pullRequestStatusGeändertes Ereignis](#)
- [pullRequestMergeStatusUpdated Ereignis](#)
- [approvalRuleTemplateEreignis wurde erstellt](#)

- [approvalRuleTemplateDas Ereignis wurde aktualisiert](#)
- [approvalRuleTemplateGelöschtes Ereignis](#)
- [approvalRuleTemplateAssociatedWithRepository Ereignis](#)
- [approvalRuleTemplateDisassociatedWithRepository Ereignis](#)
- [approvalRuleTemplateBatchAssociatedWithRepositories Ereignis](#)
- [approvalRuleTemplateBatchDisassociatedFromRepositories Ereignis](#)
- [pullRequestApprovalRuleCreated Ereignis](#)
- [pullRequestApprovalRuleDeleted Ereignis](#)
- [pullRequestApprovalRuleOverridden Ereignis](#)
- [pullRequestApprovalStateChanged Ereignis](#)
- [pullRequestApprovalRuleUpdated Ereignis](#)
- [ReactionCreated-Ereignis](#)
- [ReactionUpdated-Ereignis](#)

referenceCreated-Ereignis

In diesem Beispielereignis wurde eine Verzweigung namens myBranch in einem Repository mit dem Namen MyDemoRepo erstellt.

```
{
  "version": "0",
  "id": "01234567-EXAMPLE",
  "detail-type": "CodeCommit Repository State Change",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-06-12T10:23:43Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "event": "referenceCreated",
    "repositoryName": "MyDemoRepo",
    "repositoryId": "12345678-1234-5678-abcd-12345678abcd",
    "referenceType": "branch",
    "referenceName": "myBranch",
    "referenceFullName": "refs/heads/myBranch",
```

```
    "commitId": "3e5983DESTINATION"
  }
}
```

referenceUpdated-Ereignis

In diesem Beispielereignis wurde eine Verzweigung mit dem Namen myBranch durch eine Zusammenführung in einem Repository mit dem Namen MyDemoRepo aktualisiert.

```
{
  "version": "0",
  "id": "01234567-EXAMPLE",
  "detail-type": "CodeCommit Repository State Change",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-06-12T10:23:43Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "event": "referenceUpdated",
    "repositoryName": "MyDemoRepo",
    "repositoryId": "12345678-1234-5678-abcd-12345678abcd",
    "referenceType": "branch",
    "referenceName": "myBranch",
    "referenceFullName": "refs/heads/myBranch",
    "commitId": "7f0103fMERGE",
    "oldCommitId": "3e5983DESTINATION",
    "baseCommitId": "3e5a9bf1BASE",
    "sourceCommitId": "26a8f2SOURCE",
    "destinationCommitId": "3e5983DESTINATION",
    "mergeOption": "THREE_WAY_MERGE",
    "conflictDetailsLevel": "LINE_LEVEL",
    "conflictResolutionStrategy": "AUTOMERGE"
  }
}
```

referenceDeleted-Ereignis

In diesem Beispielereignis wurde eine Verzweigung namens myBranch in einem Repository mit dem Namen MyDemoRepo gelöscht.

```
{
  "version": "0",
  "id": "01234567-EXAMPLE",
  "detail-type": "CodeCommit Repository State Change",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-06-12T10:23:43Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "event": "referenceDeleted",
    "repositoryName": "MyDemoRepo",
    "repositoryId": "12345678-1234-5678-abcd-12345678abcd",
    "referenceType": "branch",
    "referenceName": "myBranch",
    "referenceFullName": "refs/heads/myBranch",
    "oldCommitId": "26a8f2EXAMPLE"
  }
}
```

unreferencedMergeCommitEreignis wurde erstellt

In diesem Beispielergebnis wurde ein nicht referenziertes Merge-Commit in einem Repository namens MyDemoRepo erstellt.

```
{
  "version": "0",
  "id": "01234567-EXAMPLE",
  "detail-type": "CodeCommit Repository State Change",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-06-12T10:23:43Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "event": "unreferencedMergeCommitCreated",
    "repositoryName": "MyDemoRepo",
    "repositoryId": "12345678-1234-5678-abcd-12345678abcd",

```

```

    "commitId": "7f0103fMERGE",
    "baseCommitId": "3e5a9bf1BASE",
    "sourceCommitId": "26a8f2SOURCE",
    "destinationCommitId": "3e5983DESTINATION",
    "mergeOption": "SQUASH_MERGE",
    "conflictDetailsLevel": "LINE_LEVEL",
    "conflictResolutionStrategy": "AUTOMERGE"
  }
}

```

commentOnCommitEreignis wurde erstellt

In diesem Beispielereignis hat ein verbundener Benutzer mit dem Namen `Mary_Major` ein Commit kommentiert. In diesem Beispiel hat der Verbundidentitätsanbieter Sitzungs-Tags für `displayName` und `emailAddress` konfiguriert. Diese Informationen sind in der Veranstaltung enthalten.

```

{
  "version": "0",
  "id": "e9dce2e9-EXAMPLE",
  "detail-type": "CodeCommit Comment on Commit",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-09-29T20:20:39Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "beforeCommitId": "3c5dEXAMPLE",
    "repositoryId": "7dd1EXAMPLE...",
    "inReplyTo": "695bEXAMPLE...",
    "notificationBody": "A comment event occurred in the following repository:
MyDemoRepo. The display name for the user is Mary Major. The email address for
the user is mary_major@example.com. The user arn:aws:sts::123456789012:federated-
user/Mary_Major made a comment. The comment was made on the following comment ID:
463bEXAMPLE.... For more information, go to the AWS CodeCommit console at https://us-
east-2.console.aws.amazon.com/codecommit/home?region=us-east-2#/repository/MyDemoRepo/
compare/3c5dEXAMPLE...f4d5EXAMPLE#463bEXAMPLE....",
    "commentId": "463bEXAMPLE...",
    "afterCommitId": "f4d5EXAMPLE",
    "event": "commentOnCommitCreated",
    "repositoryName": "MyDemoRepo",
    "callerUserArn": "arn:aws:sts::123456789012:federated-user/Mary_Major",

```

```
"displayName": "Mary Major",  
"emailAddress": "mary_major@example.com"  
}  
}
```

commentOnCommitEreignis wurde aktualisiert

In diesem Beispielereignis hat ein Benutzer, der eine Rolle namens Admin mit einem Sitzungsnamen namens Mary_Major übernommen hat, einen Kommentar zu einem Commit bearbeitet. In diesem Beispiel enthielt die Rolle konfigurierte Sitzungs-Tags für displayName und emailAddress. Diese Informationen sind in der Veranstaltung enthalten.

```
{  
  "version": "0",  
  "id": "98377d67-EXAMPLE",  
  "detail-type": "CodeCommit Comment on Commit",  
  "source": "aws.codecommit",  
  "account": "123456789012",  
  "time": "2019-02-09T07:15:16Z",  
  "region": "us-east-2",  
  "resources": [  
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"  
  ],  
  "detail": {  
    "afterCommitId": "53812581",  
    "beforeCommitId": "03314446",  
    "callerUserArn": "arn:aws:sts::123456789012:assumed-role/Admin/Mary_Major",  
    "commentId": "a7e5471e-EXAMPLE",  
    "event": "commentOnCommitUpdated",  
    "inReplyTo": "bdb07d47-EXAMPLE",  
    "notificationBody": "A comment event occurred in the following AWS  
CodeCommit repository: MyDemoRepo. The display name for the user is Mary  
Major. The email address for the user is mary_major@example.com. The user  
arn:aws:sts::123456789012:federated-user/Mary_Major updated a comment or  
replied to a comment. The comment was made on the following comment ID:  
bdb07d47-6fe9-47b0-a839-b93cc743b2ac:468cd1cb-2dfb-4f68-9636-8de52431d1d6.  
For more information, go to the AWS CodeCommit console https://us-east-2.console.aws.amazon.com/codesuite/codecommit/repositories/MyDemoRepo/compare/0331444646178429589969823096709582251768/.../5381258150293783361471680277136017291382?  
region\u003dus-east-2",  
    "repositoryId": "12345678-1234-1234-1234-123456789012",  
    "repositoryName": "MyDemoRepo",  
    "displayName": "Mary Major",
```

```
    "emailAddress": "mary_major@example.com"
  }
}
```

commentOnPullRequestCreated Ereignis

In diesem Beispielergebnis hat ein verbundener Benutzer mit dem Namen Saanvi_Sarkar eine Pull-Anforderung kommentiert. In diesem Beispiel hat der Verbundidentitätsanbieter Sitzungs-Tags für `displayName` und `emailAddress` konfiguriert. Diese Informationen sind in der Veranstaltung enthalten.

```
{
  "version": "0",
  "id": "98377d67-EXAMPLE",
  "detail-type": "CodeCommit Comment on Pull Request",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-02-09T07:15:16Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "beforeCommitId": "3c5dEXAMPLE",
    "repositoryId": "7dd1EXAMPLE...",
    "inReplyTo": "695bEXAMPLE...",
    "notificationBody": "A comment event occurred in the following AWS
CodeCommit repository: MyDemoRepo. The display name for the user is Saanvi
Sarkar. The email address for the user is saanvi_sarkar@example.com. The user
arn:aws:sts::123456789012:federated-user/Saanvi_Sarkar made a comment. The comment
was made on the following Pull Request: 201. For more information, go to the AWS
CodeCommit console https://us-east-2.console.aws.amazon.com/codecommit/home?region=us-
east-2#/repository/MyDemoRepo/pull-request/201/activity#3276EXAMPLE...",
    "commentId": "463bEXAMPLE...",
    "afterCommitId": "f4d5EXAMPLE",
    "event": "commentOnPullRequestCreated",
    "repositoryName": "MyDemoRepo",
    "callerUserArn": "arn:aws:sts::123456789012:federated-user/Saanvi_Sarkar",
    "pullRequestId": "201",
    "displayName": "Saanvi Sarkar",
    "emailAddress": "saanvi_sarkar@example.com"
  }
}
```



```
}
```

commentOnPullRequestUpdated Ereignis

In diesem Beispielereignis hat ein verbundener Benutzer namens Saanvi_Sarkar einen Kommentar zu einer Pull-Anforderung bearbeitet. In diesem Beispiel hat der Verbundidentitätsanbieter Sitzungs-Tags für `displayName` und `emailAddress` konfiguriert. Diese Informationen sind in der Veranstaltung enthalten.

```
{
  "version": "0",
  "id": "98377d67-EXAMPLE",
  "detail-type": "CodeCommit Comment on Pull Request",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-02-09T07:15:16Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "afterCommitId": "96814774EXAMPLE",
    "beforeCommitId": "6031971EXAMPLE",
    "callerUserArn": "arn:aws:sts::123456789012:federated-user/Saanvi_Sarkar",
    "commentId": "40cb52f0-EXAMPLE",
    "event": "commentOnPullRequestUpdated",
    "inReplyTo": "1285e713-EXAMPLE",
    "notificationBody": "A comment event occurred in the following AWS
CodeCommit repository: MyDemoRepo. The display name for the user is Saanvi
Sarkar. The email address for the user is saanvi_sarkar@example.com. The user
arn:aws:sts::123456789012:federated-user/Saanvi_Sarkar updated a comment or
replied to a comment. The comment was made on the following Pull Request:
1. For more information, go to the AWS CodeCommit console https://us-
east-2.console.aws.amazon.com/codesuite/codecommit/repositories/MyDemoRepo/pull-
requests/1/activity#40cb52f0-aac7-4c43-b771-601eff02EXAMPLE",
    "pullRequestId": "1",
    "repositoryId": "12345678-1234-1234-1234-123456789012",
    "repositoryName": "MyDemoRepo"
  }
}
```

pullRequestCreated Ereignis

In diesem Beispiereignis wurde eine Pull-Anforderung in einem Repository namens MyDemoRepo von einem Benutzer erstellt, der eine Rolle namens Admin mit dem Sitzungsnamen Mary_Major angenommen hat. Es wurden keine Sitzungs-Tag-Informationen bereitgestellt, sodass diese Informationen nicht im Ereignis enthalten sind.

```
{
  "version": "0",
  "id": "98377d67-EXAMPLE",
  "detail-type": "CodeCommit Pull Request State Change",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-02-09T07:15:16Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "author": "arn:aws:sts::123456789012:assumed-role/Admin/Mary_Major",
    "callerUserArn": "arn:aws:sts::123456789012:assumed-role/Admin/Mary_Major",
    "creationDate": "Tue Feb 9 2019 10:18:42 PDT ",
    "description": "An example description.",
    "destinationCommit": "12241970EXAMPLE",
    "destinationReference": "refs/heads/main",
    "event": "pullRequestCreated",
    "isMerged": "False",
    "lastModifiedDate": "Tue Feb 9 2019 10:18:42 PDT",
    "notificationBody": "A pull request event occurred in the following AWS CodeCommit repository: MyDemoRepo. User: arn:aws:sts::123456789012:assumed-role/Admin/Mary_Major. Event: Created. The pull request was created with the following information: Pull Request ID as 1 and title as My Example Pull Request. For more information, go to the AWS CodeCommit console https://us-east-2.console.aws.amazon.com/codesuite/codecommit/repositories/MyDemoRepo/pull-requests/1",
    "pullRequestId": "1",
    "pullRequestStatus": "Open",
    "repositoryNames": ["MyDemoRepo"],
    "revisionId": "bdc0cb9bEXAMPLE",
    "sourceCommit": "2774290EXAMPLE",
    "sourceReference": "refs/heads/test-branch",
    "title": "My Example Pull Request"
  }
}
```

```
}
```

pullRequestSourceBranchUpdated Ereignis

In diesem Beispielereignis hat ein Benutzer, der eine Rolle namens Admin mit dem Sitzungsnamen Mary_Major übernommen hat, die Quellverzweigung namens test-branch für eine Pull-Anforderung mit der ID „1“ aktualisiert.

```
{
  "version": "0",
  "id": "98377d67-EXAMPLE",
  "detail-type": "CodeCommit Pull Request State Change",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-02-09T07:15:16Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "author": "arn:aws:sts::123456789012:assumed-role/Admin/Mary_Major",
    "callerUserArn": "arn:aws:sts::123456789012:assumed-role/Admin/Mary_Major",
    "creationDate": "Tue Feb 9 2019 10:18:42 PDT",
    "description": "An example description.",
    "destinationCommit": "7644990EXAMPLE",
    "destinationReference": "refs/heads/main",
    "event": "pullRequestSourceBranchUpdated",
    "isMerged": "False",
    "lastModifiedDate": "Tue Feb 9 2019 10:18:42 PDT",
    "notificationBody": "A pull request event occurred in the following AWS
CodeCommit repository: MyDemoRepo. User: arn:aws:sts::123456789012:assumed-role/
Admin/Mary_Major. Event: Updated. The user updated the following pull request:
1. The pull request was updated with one or more commits to the source branch:
test-branch. For more information, go to the AWS CodeCommit console https://us-
east-2.console.aws.amazon.com/codesuite/codecommit/repositories/MyDemoRepo/pull-
requests/1?region\u003dus-east-2",
    "pullRequestId": "1",
    "pullRequestStatus": "Open",
    "repositoryNames": ["MyDemoRepo"],
    "revisionId": "bdc0cb9b4EXAMPLE",
    "sourceCommit": "64875001EXAMPLE",
    "sourceReference": "refs/heads/test-branch",
    "title": "My Example Pull Request"
  }
}
```

```
}  
}
```

pullRequestStatusGeändertes Ereignis

In diesem Beispiereignis hat ein Benutzer, der eine Rolle namens Admin mit dem Sitzungsnamen Mary_Major angenommen hat, eine Pull-Anforderung mit der ID „1“ geschlossen. Die Pull-Anforderung wurde nicht zusammengeführt.

```
{  
  "version": "0",  
  "id": "98377d67-EXAMPLE",  
  "detail-type": "CodeCommit Pull Request State Change",  
  "source": "aws.codecommit",  
  "account": "123456789012",  
  "time": "2019-02-09T07:15:16Z",  
  "region": "us-east-2",  
  "resources": [  
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"  
  ],  
  "detail": {  
    "author": "arn:aws:sts::123456789012:assumed-role/Admin/Mary_Major",  
    "callerUserArn": "arn:aws:sts::123456789012:assumed-role/Admin/Mary_Major",  
    "creationDate": "Tue Jun 18 10:34:20 PDT 2019",  
    "description": "An example description.",  
    "destinationCommit": "95149731EXAMPLE",  
    "destinationReference": "refs/heads/main",  
    "event": "pullRequestStatusChanged",  
    "isMerged": "False",  
    "lastModifiedDate": "Tue Jun 18 10:34:20 PDT 2019",  
    "notificationBody": "A pull request event occurred in the following AWS CodeCommit repository: MyDemoRepo. arn:aws:sts::123456789012:assumed-role/Admin/Mary_Major updated the following PullRequest 1. The pull request status has been updated. The status is closed. For more information, go to the AWS CodeCommit console https://us-east-2.console.aws.amazon.com/codesuite/codecommit/repositories/MyDemoRepo/pull-requests/1?region\u003dus-east-2",  
    "pullRequestId": "1",  
    "pullRequestStatus": "Closed",  
    "repositoryNames": ["MyDemoRepo"],  
    "revisionId": "bdc0cb9bEXAMPLE",  
    "sourceCommit": "4409936EXAMPLE",  
    "sourceReference": "refs/heads/test-branch",  
    "title": "My Example Pull Request"  
  }  
}
```

```
}
}
```

pullRequestMergeStatusUpdated Ereignis

In diesem Beispiereignis hat ein Benutzer, der eine Rolle namens Admin mit dem Sitzungsnamen Mary_Major übernommen hat, eine Pull-Anforderung mit der ID „1“ zusammengeführt.

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "CodeCommit Pull Request State Change",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-06-12T10:23:43Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "author": "arn:aws:sts::123456789012:assumed-role/Admin/Mary_Major",
    "callerUserArn": "arn:aws:sts::123456789012:assumed-role/Admin/Mary_Major",
    "creationDate": "Mon Mar 11 14:42:31 PDT 2019",
    "description": "An example description.",
    "destinationCommit": "4376719EXAMPLE",
    "destinationReference": "refs/heads/main",
    "event": "pullRequestMergeStatusUpdated",
    "isMerged": "True",
    "lastModifiedDate": "Mon Mar 11 14:42:31 PDT 2019",
    "mergeOption": "FAST_FORWARD_MERGE",
    "notificationBody": "A pull request event occurred in the following AWS CodeCommit repository: MyDemoRepo. arn:aws:sts::123456789012:assumed-role/Admin/Mary_Major updated the following PullRequest 1. The pull request merge status has been updated. The status is merged. For more information, go to the AWS CodeCommit console https://us-east-2.console.aws.amazon.com/codesuite/codecommit/repositories/MyDemoRepo/pull-requests/1?region\u003dus-east-2",
    "pullRequestId": "1",
    "pullRequestStatus": "Closed",
    "repositoryNames": ["MyDemoRepo"],
    "revisionId": "bdc0cb9beEXAMPLE",
    "sourceCommit": "0701696EXAMPLE",
    "sourceReference": "refs/heads/test-branch",
    "title": "My Example Pull Request"
  }
}
```

```
}  
}
```

approvalRuleTemplateEreignis wurde erstellt

In diesem Beispielereignis hat ein Benutzer mit dem IAM-Benutzernamen eine Genehmigungsregelvorlage mit dem 2-approvers-required-for-main Namen Mary_Major erstellt.

```
{  
  "version": "0",  
  "id": "f7702227-EXAMPLE",  
  "detail-type": "CodeCommit Approval Rule Template Change",  
  "source": "aws.codecommit",  
  "account": "123456789012",  
  "time": "2019-11-06T19:02:27Z",  
  "region": "us-east-2",  
  "resources": [],  
  "detail": {  
    "approvalRuleTemplateContentSha256": "f742eebbEXAMPLE",  
    "approvalRuleTemplateId": "d7385967-EXAMPLE",  
    "approvalRuleTemplateName": "2-approvers-required-for-main",  
    "callerUserArn": "arn:aws:iam::123456789012:user/Mary_Major",  
    "creationDate": "Wed Nov 06 19:02:14 UTC 2019",  
    "event": "approvalRuleTemplateCreated",  
    "lastModifiedDate": "Wed Nov 06 19:02:14 UTC 2019",  
    "notificationBody": "A approval rule template event occurred in the following  
AWS CodeCommit account: 123456789012. User: arn:aws:iam::123456789012:user/Mary_Major.  
Additional information: An approval rule template with the following name has been  
created: 2-approvers-required-for-main. The ID of the created template is: d7385967-  
EXAMPLE. For more information, go to the AWS CodeCommit console.",  
    "repositories": {}  
  }  
}
```

approvalRuleTemplateDas Ereignis wurde aktualisiert

In diesem Beispielereignis hat ein Benutzer mit dem IAM-Benutzernamen eine Genehmigungsregelvorlage mit dem 2-approvers-required-for-main Namen Mary_Major bearbeitet. Die Genehmigungsregelvorlage ist keinem Repository zugeordnet.

```
{
```

```

"version": "0",
"id": "66403118-EXAMPLE",
"detail-type": "CodeCommit Approval Rule Template Change",
"source": "aws.codecommit",
"account": "123456789012",
"time": "2019-11-12T23:03:30Z",
"region": "us-east-2",
"resources": [

],
"detail": {
  "approvalRuleTemplateContentSha256": "f742eebbEXAMPLE",
  "approvalRuleTemplateId": "c9d2b844-EXAMPLE",
  "approvalRuleTemplateName": "2-approvers-required-for-main",
  "callerUserArn": "arn:aws:iam::123456789012:user\Mary_Major",
  "creationDate": "Tue Nov 12 23:03:06 UTC 2019",
  "event": "approvalRuleTemplateDeleted",
  "lastModifiedDate": "Tue Nov 12 23:03:20 UTC 2019",
  "notificationBody": "A approval rule template event occurred in the following AWS
CodeCommit account: 123456789012. User: arn:aws:iam::123456789012:user\Mary_Major.
Additional information: An approval rule template with the following name has been
deleted: 2-approvers-required-for-main. The ID of the updated template is: c9d2b844-
EXAMPLE. For more information, go to the AWS CodeCommit console.",
  "repositories": {}
}
}

```

approvalRuleTemplateGelöschtes Ereignis

In diesem Beispiereignis hat ein Benutzer mit dem IAM-Benutzernamen eine Genehmigungsregelvorlage mit dem 2-approvers-required-for-main Namen Mary_Major gelöscht. Die Genehmigungsregelvorlage ist keinem Repository zugeordnet.

```

{
  "version": "0",
  "id": "66403118-EXAMPLE",
  "detail-type": "CodeCommit Approval Rule Template Change",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-11-12T23:03:30Z",
  "region": "us-east-2",
  "resources": [],
  "detail": {

```

```

    "approvalRuleTemplateContentSha256": "4f3de6632EXAMPLE",
    "approvalRuleTemplateId": "c9d2b844-EXAMPLE",
    "approvalRuleTemplateName": "2-approvers-required-for-main",
    "callerUserArn": "arn:aws:iam::123456789012:user\Mary_Major",
    "creationDate": "Tue Nov 12 23:03:06 UTC 2019",
    "event": "approvalRuleTemplateUpdated",
    "lastModifiedDate": "Tue Nov 12 23:03:20 UTC 2019",
    "notificationBody": "A approval rule template event occurred in the following AWS
CodeCommit account: 123456789012. User: arn:aws:iam::123456789012:user\Mary_Major.
Additional information: An approval rule template with the following name has
been updated: 2-approvers-required-for-main. The ID of the updated template is:
c9d2b844-EXAMPLE. The after rule template content SHA256 is 4f3de663EXAMPLE. For more
information, go to the AWS CodeCommit console.",
    "repositories": {}
  }
}

```

approvalRuleTemplateAssociatedWithRepository Ereignis

In diesem Beispielergebnis hat ein Benutzer mit einem IAM-Benutzernamen eine Genehmigungsregelvorlage mit dem Namen eines Repositorys 2-approvers-required-for-main mit dem Namen Mary_Major MyDemoRepo verknüpft.

```

{
  "version": "0",
  "id": "ea1c6d73-EXAMPLE",
  "detail-type": "CodeCommit Approval Rule Template Change",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-11-06T19:02:27Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "approvalRuleTemplateContentSha256": "f742eebbEXAMPLE",
    "approvalRuleTemplateId": "d7385967-EXAMPLE",
    "approvalRuleTemplateName": "2-approvers-required-for-main",
    "callerUserArn": "arn:aws:iam::123456789012:user/Mary_Major",
    "creationDate": "Wed Nov 06 19:02:14 UTC 2019",
    "event": "approvalRuleTemplateAssociatedWithRepository",
    "lastModifiedDate": "Wed Nov 06 19:02:14 UTC 2019",
  }
}

```



```

    "notificationBody": "A approval rule template event occurred in the following
AWS CodeCommit account: 123456789012. User: arn:aws:iam::123456789012:user/Mary_Major.
Additional information: An approval rule template has been associated with the
following repository: [MyDemoRepo]. For more information, go to the AWS CodeCommit
console.",
    "repositories": {
        "MyDemoRepo": "92ca7bf2-d878-49ed-a994-336a6cc7c574"
    }
}

```

approvalRuleTemplateDisassociatedWithRepository Ereignis

In diesem Beispielereignis hat ein Benutzer mit dem IAM-Benutzernamen die Mary_Major Zuordnung einer Genehmigungsregelvorlage mit dem Namen zu einem Repository mit 2-approvers-required-for-main dem Namen aufgehoben. MyDemoRepo

```

{
    "version": "0",
    "id": "ea1c6d73-EXAMPLE",
    "detail-type": "CodeCommit Approval Rule Template Change",
    "source": "aws.codecommit",
    "account": "123456789012",
    "time": "2019-11-06T19:02:27Z",
    "region": "us-east-2",
    "resources": [
        "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
    ],
    "detail": {
        "approvalRuleTemplateContentSha256": "f742eebbEXAMPLE",
        "approvalRuleTemplateId": "d7385967-EXAMPLE",
        "approvalRuleTemplateName": "2-approvers-required-for-main",
        "callerUserArn": "arn:aws:iam::123456789012:user/Mary_Major",
        "creationDate": "Wed Nov 06 19:02:14 UTC 2019",
        "event": "approvalRuleTemplateDisassociatedFromRepository",
        "lastModifiedDate": "Wed Nov 06 19:02:14 UTC 2019",
        "notificationBody": "A approval rule template event occurred in the following
AWS CodeCommit account: 123456789012. User: arn:aws:iam::123456789012:user/Mary_Major.
Additional information: An approval rule template has been disassociated from the
following repository: [MyDemoRepo]. For more information, go to the AWS CodeCommit
console.",
        "repositories": {
            "MyDemoRepo": "92ca7bf2-d878-49ed-a994-336a6cc7c574"
        }
    }
}

```

```

    }
  }
}

```

approvalRuleTemplateBatchAssociatedWithRepositories Ereignis

In diesem Beispielergebnis hat ein Benutzer mit dem IAM-Benutzernamen `Mary_Major` batch eine Genehmigungsregelvorlage mit dem Namen eines Repositorys `MyDemoRepo` und einem Repository `2-approvers-required-for-main` mit dem Namen `MyTestRepo` verknüpft.

```

{
  "version": "0",
  "id": "0f861e5b-EXAMPLE",
  "detail-type": "CodeCommit Approval Rule Template Change",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-11-12T23:39:09Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "approvalRuleTemplateContentSha256": "f742eebbEXAMPLE",
    "approvalRuleTemplateId": "c71c1fe0-EXAMPLE",
    "approvalRuleTemplateName": "2-approvers-required-for-main",
    "callerUserArn": "arn:aws:iam::123456789012:user/Mary_Major",
    "creationDate": "Tue Nov 12 23:38:57 UTC 2019",
    "event": "batchAssociateApprovalRuleTemplateWithRepositories",
    "lastModifiedDate": "Tue Nov 12 23:38:57 UTC 2019",
    "notificationBody": "A approval rule template event occurred in the following
AWS CodeCommit account: 123456789012. User: arn:aws:iam::123456789012:user\Mary_Major.
Additional information: An approval rule template has been batch associated with the
following repository names: [MyDemoRepo, MyTestRepo]. For more information, go to the
AWS CodeCommit console.",
    "repositories": {
      "MyDemoRepo": "MyTestRepo"
    }
  }
}

```

approvalRuleTemplateBatchDisassociatedFromRepositories Ereignis

In diesem Beispiereignis hat ein Benutzer mit dem IAM-Benutzernamen `Mary_Major` batch die Zuordnung einer Genehmigungsregelvorlage zu einem Repository mit `2-approvers-required-for-main` dem Namen `MyDemoRepo` und einem Repository mit dem Namen `MyTestRepo` aufgehoben.

`MyTestRepo`

```
{
  "version": "0",
  "id": "e08fc996-EXAMPLE",
  "detail-type": "CodeCommit Approval Rule Template Change",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-11-12T23:39:09Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "approvalRuleTemplateContentSha256": "f742eebbEXAMPLE",
    "approvalRuleTemplateId": "c71c1fe0-ff91-4db4-9a45-a86a7b6c474f",
    "approvalRuleTemplateName": "2-approvers-required-for-main",
    "callerUserArn": "arn:aws:iam::123456789012:user/Mary_Major",
    "creationDate": "Tue Nov 12 23:38:57 UTC 2019",
    "event": "batchDisassociateApprovalRuleTemplateFromRepositories",
    "lastModifiedDate": "Tue Nov 12 23:38:57 UTC 2019",
    "notificationBody": "A approval rule template event occurred in the following
AWS CodeCommit account: 123456789012. User: arn:aws:iam::123456789012:user/Mary_Major.
Additional information: An approval rule template has been batch disassociated from
the following repository names: [MyDemoRepo, MyTestRepo]. For more information, go to
the AWS CodeCommit console.",
    "repositories": {
      "MyDemoRepo": "MyTestRepo"
    }
  }
}
```

pullRequestApprovalRuleCreated Ereignis

In diesem Beispiereignis hat ein Benutzer mit dem IAM-Benutzernamen `Mary_Major` eine Genehmigungsregel erstellt, die `1-approver-needed` nach einer Pull-Anfrage mit der ID von `227` benannt ist.

```

{
  "version": "0",
  "id": "ad860f12-EXAMPLE",
  "detail-type": "CodeCommit Pull Request State Change",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-11-06T19:12:19Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "approvalRuleContentSha256": "f742eebbEXAMPLE",
    "approvalRuleId": "0a9b5dfc-EXAMPLE",
    "approvalRuleName": "1-approver-needed",
    "author": "arn:aws:iam::123456789012:user/Mary_Major",
    "callerUserArn": "arn:aws:iam::123456789012:user/Mary_Major",
    "creationDate": "Wed Nov 06 19:10:58 UTC 2019",
    "description": "An An example description.",
    "destinationCommit": "194fdf00EXAMPLE",
    "destinationReference": "refs/heads/main",
    "event": "pullRequestApprovalRuleCreated",
    "isMerged": "False",
    "lastModifiedDate": "Wed Nov 06 19:10:58 UTC 2019",
    "notificationBody": "A pull request event occurred in the following AWS
CodeCommit repository: MyDemoRepo. User: arn:aws:iam::123456789012:user/Mary_Major.
Event: Updated. Pull request: 227. Additional information: An approval rule has been
created with the following name: 1-approver-needed. For more information, go to the
AWS CodeCommit console https://us-east-2.console.aws.amazon.com/codesuite/codecommit/
repositories/MyDemoRepo/pull-requests/227?region=us-east-2",
    "pullRequestId": "227",
    "pullRequestStatus": "Open",
    "repositoryNames": [
      "MyDemoRepo"
    ],
    "revisionId": "3b8cecab3EXAMPLE",
    "sourceCommit": "29964a17EXAMPLE",
    "sourceReference": "refs/heads/test-branch",
    "title": "My example pull request"
  }
}

```

pullRequestApprovalRuleDeleted Ereignis

In diesem Beispiereignis hat ein Benutzer mit dem IAM-Benutzernamen von eine Genehmigungsregel `Mary_Major` gelöscht, die `1-approver-needed` nach einer Pull-Anfrage mit der ID von 227 benannt ist. Ein IAM-Benutzer mit dem Namen hat die `Saanvi_Sarkar` Genehmigungsregel ursprünglich verfasst.

```
{
  "version": "0",
  "id": "c1c3509d-EXAMPLE",
  "detail-type": "CodeCommit Pull Request State Change",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-11-06T19:12:19Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "approvalRuleContentSha256": "f742eebbEXAMPLE",
    "approvalRuleId": "0a9b5dfc-EXAMPLE",
    "approvalRuleName": "1-approver-needed",
    "author": "arn:aws:iam::123456789012:user/Saanvi_Sarkar",
    "callerUserArn": "arn:aws:iam::123456789012:user/Mary_Major",
    "creationDate": "Wed Nov 06 19:10:58 UTC 2019",
    "description": "An An example description.",
    "destinationCommit": "194fdf00EXAMPLE",
    "destinationReference": "refs/heads/main",
    "event": "pullRequestApprovalRuleDeleted",
    "isMerged": "False",
    "lastModifiedDate": "Wed Nov 06 19:10:58 UTC 2019",
    "notificationBody": "A pull request event occurred in the following AWS
CodeCommit repository: MyDemoRepo. User: arn:aws:iam::123456789012:user/Mary_Major.
Event: Created. Pull request: 227. Additional information: An approval rule has been
deleted: 1-approver-needed was deleted. For more information, go to the AWS CodeCommit
console https://us-east-2.console.aws.amazon.com/codesuite/codecommit/repositories/
MyDemoRepo/pull-requests/227?region=us-east-2",
    "pullRequestId": "227",
    "pullRequestStatus": "Open",
    "repositoryNames": [
      "MyDemoRepo"
    ],
    "revisionId": "3b8cecabEXAMPLE",
```

```

    "sourceCommit": "29964a17EXAMPLE",
    "sourceReference": "refs/heads/test-branch",
    "title": "My example pull request"
  }
}

```

pullRequestApprovalRuleOverridden Ereignis

In diesem Beispielergebnis wurden die Genehmigungsregelanforderungen für eine Pull-Anfrage von einem Benutzer mit dem IAM-Benutzernamen von Mary_Major aufgehoben (OVERRIDE). Die Pull-Anfrage wurde von einem Benutzer mit dem IAM-Benutzernamen von verfasst. Li_Juan

```

{
  "version": "0",
  "id": "52d2cb73-EXAMPLE",
  "detail-type": "CodeCommit Pull Request State Change",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-11-06T19:12:19Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "author": "arn:aws:iam::123456789012:user/Li_Juan",
    "callerUserArn": "arn:aws:iam::123456789012:user/Mary_Major",
    "creationDate": "Wed Nov 06 19:10:58 UTC 2019",
    "description": "An An example description.",
    "destinationCommit": "194fdf00EXAMPLE",
    "destinationReference": "refs/heads/main",
    "event": "pullRequestApprovalRuleOverridden",
    "isMerged": "False",
    "lastModifiedDate": "Wed Nov 06 19:10:58 UTC 2019",
    "notificationBody": "A pull request event occurred in the following AWS
CodeCommit repository: MyDemoRepo. User: arn:aws:iam::123456789012:user/Mary_Major.
Event: Updated. Pull request name: 227. Additional information: An override
event has occurred for the approval rules for this pull request. Override status:
OVERRIDE. For more information, go to the AWS CodeCommit console https://us-
east-2.console.aws.amazon.com/codesuite/codecommit/repositories/MyDemoRepo/pull-
requests/227?region=us-east-2",
    "overrideStatus": "OVERRIDE",
    "pullRequestId": "227",
    "pullRequestStatus": "Open",

```

```

    "repositoryNames": [
        "MyDemoRepo"
    ],
    "revisionId": "3b8cecabEXAMPLE",
    "sourceCommit": "29964a17EXAMPLE",
    "sourceReference": "refs/heads/test-branch",
    "title": "My example pull request"
}
}

```

In diesem Beispiel wurden die Genehmigungsregelanforderungen für eine Pull-Anforderung wieder eingeführt („REVOKE“).

```

{
    "version": "0",
    "id": "2895482d-13eb-b783-270d-76588e6029fa",
    "detail-type": "CodeCommit Pull Request State Change",
    "source": "aws.codecommit",
    "account": "123456789012",
    "time": "2019-11-06T19:12:19Z",
    "region": "us-east-2",
    "resources": [
        "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
    ],
    "detail": {
        "author": "arn:aws:iam::123456789012:user/Li_Juan",
        "callerUserArn": "arn:aws:iam::123456789012:user/Mary_Major",
        "creationDate": "Wed Nov 06 19:10:58 UTC 2019",
        "description": "An An example description.",
        "destinationCommit": "194fdf00EXAMPLE",
        "destinationReference": "refs/heads/main",
        "event": "pullRequestApprovalRuleOverridden",
        "isMerged": "False",
        "lastModifiedDate": "Wed Nov 06 19:10:58 UTC 2019",
        "notificationBody": "A pull request event occurred in the following
AWS CodeCommit repository: MyDemoRepo. User: arn:aws:iam::123456789012:user/
Mary_Major. Event: Updated. Pull request name: 227. Additional information: An
override event has occurred for the approval rules for this pull request. Override
status: REVOKE. For more information, go to the AWS CodeCommit console https://
us-east-2.console.aws.amazon.com/codesuite/codecommit/repositories/MyDemoRepo/pull-
requests/227?region=us-east-2",
        "overrideStatus": "REVOKE",
        "pullRequestId": "227",
    }
}

```

```

    "pullRequestStatus": "Open",
    "repositoryNames": [
        "MyDemoRepo"
    ],
    "revisionId": "3b8cecabEXAMPLE",
    "sourceCommit": "29964a17EXAMPLE",
    "sourceReference": "refs/heads/test-branch",
    "title": "My example pull request"
}
}

```

pullRequestApprovalStateChanged Ereignis

In diesem Beispielereignis wurde eine Pull-Anfrage von einem Benutzer mit dem IAM-Benutzernamen genehmigt. Mary_Major

```

{
    "version": "0",
    "id": "53e5d7e9-986c-1ebf-9d8b-ebef5596da0e",
    "detail-type": "CodeCommit Pull Request State Change",
    "source": "aws.codecommit",
    "account": "123456789012",
    "time": "2019-11-06T19:12:19Z",
    "region": "us-east-2",
    "resources": [
        "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
    ],
    "detail": {
        "approvalStatus": "APPROVE",
        "author": "arn:aws:iam::123456789012:user/Li_Juan",
        "callerUserArn": "arn:aws:iam::123456789012:user/Mary_Major",
        "creationDate": "Wed Nov 06 19:10:58 UTC 2019",
        "description": "An An example description.",
        "destinationCommit": "194fdf00EXAMPLE",
        "destinationReference": "refs/heads/main",
        "event": "pullRequestApprovalStateChanged",
        "isMerged": "False",
        "lastModifiedDate": "Wed Nov 06 19:10:58 UTC 2019",
        "notificationBody": "A pull request event occurred in the following
AWS CodeCommit repository: MyDemoRepo. User: arn:aws:iam::123456789012:user/
Mary_Major. Event: Updated. Pull request name: 227. Additional information:
A user has changed their approval state for the pull request. State change:
APPROVE. For more information, go to the AWS CodeCommit console https://us-

```



```

east-2.console.aws.amazon.com/codesuite/codecommit/repositories/MyDemoRepo/pull-
requests/227?region=us-east-2",
    "pullRequestId": "227",
    "pullRequestStatus": "Open",
    "repositoryNames": [
        "MyDemoRepo"
    ],
    "revisionId": "3b8cecabEXAMPLE",
    "sourceCommit": "29964a17EXAMPLE",
    "sourceReference": "refs/heads/test-branch",
    "title": "My example pull request"
}
}

```

In diesem Beispielereignis wurde eine Genehmigung für eine Pull-Anfrage von einem Benutzer mit dem IAM-Benutzernamen von widerrufen. Mary_Major

```

{
    "version": "0",
    "id": "25e183d7-d01a-4e07-2bd9-b2d56ebecc81",
    "detail-type": "CodeCommit Pull Request State Change",
    "source": "aws.codecommit",
    "account": "123456789012",
    "time": "2019-11-06T19:12:19Z",
    "region": "us-east-2",
    "resources": [
        "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
    ],
    "detail": {
        "approvalStatus": "REVOKE",
        "author": "arn:aws:iam::123456789012:user/Li_Juan",
        "callerUserArn": "arn:aws:iam::123456789012:user/Mary_Major",
        "creationDate": "Wed Nov 06 19:10:58 UTC 2019",
        "description": "An An example description.",
        "destinationCommit": "194fdf00EXAMPLE",
        "destinationReference": "refs/heads/main",
        "event": "pullRequestApprovalStateChanged",
        "isMerged": "False",
        "lastModifiedDate": "Wed Nov 06 19:10:58 UTC 2019",
        "notificationBody": "A pull request event occurred in the following AWS
CodeCommit repository: MyDemoRepo. User: arn:aws:iam::123456789012:user/Mary_Major.
Event: Updated. Pull request name: 227. Additional information: A user has changed
their approval state for the pull request. State change: REVOKE. For more information,

```

```

go to the AWS CodeCommit console https://us-east-2.console.aws.amazon.com/codesuite/codecommit/repositories/MyDemoRepo/pull-requests/227?region=us-east-2,
    "pullRequestId": "227",
    "pullRequestStatus": "Open",
    "repositoryNames": [
        "MyDemoRepo"
    ],
    "revisionId": "3b8cecabEXAMPLE",
    "sourceCommit": "29964a17EXAMPLE",
    "sourceReference": "refs/heads/test-branch",
    "title": "My example pull request"
}
}

```

pullRequestApprovalRuleUpdated Ereignis

In diesem Beispiereignis wurde eine Genehmigungsregel für eine Pull-Anfrage von einem Benutzer mit dem IAM-Benutzernamen bearbeitet. `Mary_Major` Sie ist auch die Benutzerin, die die Pull-Anfrage erstellt hat.

```

{
    "version": "0",
    "id": "21b1c819-2889-3528-1cb8-3861aacf9d42",
    "detail-type": "CodeCommit Pull Request State Change",
    "source": "aws.codecommit",
    "account": "123456789012",
    "time": "2019-11-06T19:12:19Z",
    "region": "us-east-2",
    "resources": [
        "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
    ],
    "detail": {
        "approvalRuleContentSha256": "f742eebbEXAMPLE",
        "approvalRuleId": "0a9b5dfc-EXAMPLE",
        "approvalRuleName": "1-approver-needed",
        "author": "arn:aws:iam::123456789012:user/Mary_Major",
        "callerUserArn": "arn:aws:iam::123456789012:user/Mary_Major",
        "creationDate": "Wed Nov 06 19:10:58 UTC 2019",
        "description": "An example description.",
        "destinationCommit": "194fdf00EXAMPLE",
        "destinationReference": "refs/heads/main",
        "event": "pullRequestApprovalRuleUpdated",
        "isMerged": "False",
    }
}

```

```

    "lastModifiedDate": "Wed Nov 06 19:10:58 UTC 2019",
    "notificationBody": "A pull request event occurred in the following
AWS CodeCommit repository: MyDemoRepo. User: arn:aws:iam::123456789012:user/
Mary_Major. Event: Updated. Pull request name: 227. The content of an approval
rule has been updated for the pull request. The name of the updated rule is: 1-
approver-needed. For more information, go to the AWS CodeCommit console https://
us-east-2.console.aws.amazon.com/codesuite/codecommit/repositories/MyDemoRepo/pull-
requests/227?region=us-east-2",
    "pullRequestId": "227",
    "pullRequestStatus": "Open",
    "repositoryNames": [
        "MyDemoRepo"
    ],
    "revisionId": "3b8cecab3EXAMPLE",
    "sourceCommit": "29964a17EXAMPLE",
    "sourceReference": "refs/heads/test-branch",
    "title": "My example pull request"
}
}

```

ReactionCreated-Ereignis

In diesem Beispielereignis wurde eine Reaktion auf einen Kommentar von einem Benutzer mit dem IAM-Benutzernamen von hinzugefügt. Mary_Major

```

{
  "version":"0",
  "id":"59fccccd8-217a-32ce-2b05-561ed68a1c42",
  "detail-type":"CodeCommit Comment Reaction Change",
  "source":"aws.codecommit",
  "account":"123456789012",
  "time":"2020-04-14T00:49:03Z",
  "region":"us-east-2",
  "resources":[
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail":{
    "callerUserArn":"arn:aws:iam::123456789012:user/Mary_Major",
    "commentId":"28930161-EXAMPLE",
    "event":"commentReactionCreated",
    "notificationBody":"A comment reaction event occurred in the following AWS
CodeCommit Repository: MyDemoRepo. The user: arn:aws:iam::123456789012:user/Mary_Major
made a comment reaction # to the comment with comment ID: 28930161-EXAMPLE",

```

```
    "reactionEmojis":["#"],
    "reactionShortcodes":[":thumbsdown:"],
    "reactionUnicode":["U+1F44E"],
    "repositoryId":"12345678-1234-5678-abcd-12345678abcd",
    "repositoryName":"MyDemoRepo"
  }
}
```

ReactionUpdated-Ereignis

In diesem Beispielereignis wurde eine Reaktion auf einen Kommentar von einem Benutzer mit dem IAM-Benutzernamen aktualisiert. Mary_Major Benutzer können nur ihre eigenen Reaktionen aktualisieren.

```
{
  "version":"0",
  "id":"0844ed99-a53f-3bdb-6048-4de315516889",
  "detail-type":"CodeCommit Comment Reaction Change",
  "source":"aws.codecommit",
  "account":"123456789012",
  "time":"2020-04-22T23:19:42Z",
  "region":"us-east-2",
  "resources":[
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail":{
    "callerUserArn":"arn:aws:iam::123456789012:user/Mary_Major",
    "commentId":"28930161-EXAMPLE",
    "event":"commentReactionUpdated",
    "notificationBody":"A comment reaction event occurred in the following AWS
CodeCommit Repository: MyDemoRepo. The user: arn:aws:iam::123456789012:user/Mary_Major
updated a reaction :smile: to the comment with comment ID: 28930161-EXAMPLE",
    "reactionEmojis":[
      "#"
    ],
    "reactionShortcodes":[
      ":smile:"
    ],
    "reactionUnicode":["
U+1F604"
    ],
    "repositoryId":"12345678-1234-5678-abcd-12345678abcd",
    "repositoryName":"MyDemoRepo"
  }
}
```

```
}  
}
```

AWS CodeCommit API-Aufrufe protokollieren mit AWS CloudTrail

CodeCommit ist in einen Dienst integriert AWS CloudTrail, der eine Aufzeichnung der Aktionen bereitstellt, die ein Benutzer, eine Rolle oder ein AWS Dienst in ausgeführt hat CodeCommit. CloudTrail erfasst alle API-Aufrufe CodeCommit als Ereignisse, einschließlich Aufrufe von der CodeCommit Konsole, Ihrem Git-Client und von Codeaufrufen an den CodeCommit APIs. Wenn Sie einen Trail erstellen, können Sie die kontinuierliche Bereitstellung von CloudTrail Ereignissen an einen Amazon S3 S3-Bucket aktivieren, einschließlich Ereignissen für CodeCommit. Wenn Sie keinen Trail konfigurieren, können Sie die neuesten Ereignisse trotzdem in der CloudTrail Konsole im Ereignisverlauf anzeigen. Anhand der von gesammelten Informationen können Sie die Anfrage ermitteln CloudTrail, an die die Anfrage gestellt wurde CodeCommit, die IP-Adresse, von der aus die Anfrage gestellt wurde, wer die Anfrage gestellt hat, wann sie gestellt wurde, und weitere Details.

Weitere Informationen CloudTrail dazu finden Sie im [AWS CloudTrail Benutzerhandbuch](#).

CodeCommit Informationen in CloudTrail

CloudTrail ist in Ihrem Amazon Web Services Services-Konto aktiviert, wenn Sie das Konto erstellen. Wenn eine Aktivität in stattfindet CodeCommit, wird diese Aktivität zusammen mit anderen AWS Serviceereignissen in der CloudTrail Ereignishistorie in einem Ereignis aufgezeichnet. Sie können die neusten Ereignisse in Ihrem Amazon Web Services-Konto anzeigen, suchen und herunterladen. Weitere Informationen finden Sie unter [Ereignisse mit CloudTrail Ereignisverlauf anzeigen](#).

Für eine fortlaufende Aufzeichnung der Ereignisse in Ihrem Amazon Web Services Services-Konto, einschließlich der Ereignisse für CodeCommit, erstellen Sie einen Trail. Ein Trail ermöglicht CloudTrail die Übermittlung von Protokolldateien an einen Amazon S3 S3-Bucket. Wenn Sie einen Trail in der Konsole anlegen, gilt dieser standardmäßig für alle Regionen. Der Trail protokolliert Ereignisse aus allen Regionen der AWS Partition und übermittelt die Protokolldateien an den von Ihnen angegebenen Amazon S3 S3-Bucket. Darüber hinaus können Sie andere AWS Dienste konfigurieren, um die in den CloudTrail Protokollen gesammelten Ereignisdaten weiter zu analysieren und darauf zu reagieren. Weitere Informationen finden Sie unter:

- [Übersicht zum Erstellen eines Trails](#)
- [CloudTrail Unterstützte Dienste und Integrationen](#)
- [Konfiguration von Amazon SNS SNS-Benachrichtigungen für CloudTrail](#)

- [Empfangen von CloudTrail Protokolldateien aus mehreren Regionen](#) und [Empfangen von CloudTrail Protokolldateien von mehreren Konten](#)

Wenn die CloudTrail Protokollierung in Ihrem Amazon Web Services Services-Konto aktiviert ist, werden API-Aufrufe von CodeCommit Aktionen in CloudTrail Protokolldateien nachverfolgt, wo sie zusammen mit anderen AWS Servicedatensätzen geschrieben werden. CloudTrail bestimmt anhand eines Zeitraums und der Dateigröße, wann eine neue Datei erstellt und in sie geschrieben werden soll.

Alle CodeCommit Aktionen werden protokolliert CloudTrail, einschließlich einiger Aktionen (z. B. `GetObjectIdentifier`), die derzeit nicht in der [AWS CodeCommit API-Referenz](#) dokumentiert sind, sondern stattdessen als Zugriffsberechtigungen referenziert und unter dokumentiert sind [Referenz für CodeCommit-Berechtigungen](#). Beispielsweise generieren Aufrufe der Aktionen `ListRepositories` (in den AWS CLI, `aws codecommit list-repositories`), `CreateRepository` (`aws codecommit create-repository`) und `PutRepositoryTriggers` (`aws codecommit put-repository-triggers`) Einträge in den CloudTrail Protokolldateien, ebenso wie Git-Client-Aufrufe von `GitPull` und `GitPush`. Wenn du ein CodeCommit Repository als Quelle für eine Pipeline in konfiguriert hast `CodePipeline`, siehst du außerdem Aufrufe für CodeCommit Zugriffsberechtigungsaktionen wie `UploadArchive from CodePipeline`. Da Repositories AWS Key Management Service zum Ver- und Entschlüsseln CodeCommit verwendet werden, werden Sie auch Aufrufe von CodeCommit an `Encrypt` und `Decrypt` Aktionen von AWS KMS in Protokollen sehen. CloudTrail

Jeder Protokolleintrag enthält Informationen über den Ersteller der Anforderung. Der Benutzeridentität im Protokolleintrag können Sie folgende Informationen entnehmen:

- Ob die Anfrage mit Root- oder IAM-Benutzer-Anmeldeinformationen ausgeführt wurde
- Ob die Anfrage mit temporären Sicherheitsanmeldeinformationen für eine Funktion oder einen verbundenen Benutzer oder von einer angenommenen Rolle getätigt wurde
- Ob die Anfrage von einem anderen Dienst gestellt wurde AWS

Weitere Informationen finden Sie unter [CloudTrail userIdentity-Element](#).

Sie können Ihre Protokolldateien so lange in Ihrem Amazon S3 S3-Bucket speichern, wie Sie möchten, aber Sie können auch Amazon S3 S3-Lebenszyklusregeln definieren, um Protokolldateien automatisch zu archivieren oder zu löschen. Standardmäßig werden die Protokolldateien mit serverseitiger Amazon-S3-Verschlüsselung (SSE) verschlüsselt.

Grundlegendes zu CodeCommit Einträgen in Protokolldateien

CloudTrail Protokolldateien können einen oder mehrere Protokolleinträge enthalten. In jedem Eintrag werden mehrere Ereignisse im JSON-Format aufgelistet. Ein Protokollereignis stellt eine einzelne Anforderung aus einer beliebigen Quelle dar und enthält unter anderem Informationen über die angeforderte Aktion, das Datum und die Uhrzeit der Aktion sowie über die Anforderungsparameter. Protokolleinträge sind kein geordnetes Stacktrace der öffentlichen API-Aufrufe und erscheinen daher nicht in einer bestimmten Reihenfolge.

Note

Dieses Beispiel wurde zur Verbesserung der Lesbarkeit formatiert. In einer CloudTrail Protokolldatei sind alle Einträge und Ereignisse in einer einzigen Zeile zusammengefasst. Dieses Beispiel wurde ebenfalls auf einen einzigen Eintrag beschränkt. CodeCommit In einer echten CloudTrail Protokolldatei sehen Sie Einträge und Ereignisse von mehreren AWS Diensten.

Inhalt

- [Beispiel: Ein Protokolleintrag zum Auflisten von CodeCommit Repositorys](#)
- [Beispiel: Ein Logeintrag zum Erstellen eines Repositorys CodeCommit](#)
- [Beispiele: Protokolleinträge zum Abrufen von Git-Aufrufen an ein CodeCommit-Repository](#)
- [Beispiel: Ein Logeintrag für einen erfolgreichen Push in ein Repository CodeCommit](#)

Beispiel: Ein Protokolleintrag zum Auflisten von CodeCommit Repositorys

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag, der die `ListRepositories` Aktion demonstriert.

Note

Es wird zwar eine Liste von Repositorys `ListRepositories` zurückgegeben, nicht veränderbare Antworten werden jedoch nicht in CloudTrail Protokollen aufgezeichnet, sodass sie `responseElements` wie `null` in der Protokolldatei angezeigt werden.

```
{
```

```
"eventVersion": "1.05",
"userIdentity": {
  "type": "IAMUser",
  "principalId": "AIDACKCEVSQ6C2EXAMPLE",
  "arn": "arn:aws:iam::444455556666:user/Mary_Major",
  "accountId": "444455556666",
  "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
  "userName": "Mary_Major"
},
"eventTime": "2016-12-14T17:57:36Z",
"eventSource": "codecommit.amazonaws.com",
"eventName": "ListRepositories",
"awsRegion": "us-east-1",
"sourceIPAddress": "203.0.113.12",
"userAgent": "aws-cli/1.10.53 Python/2.7.9 Windows/8 botocore/1.4.43",
"requestParameters": null,
"responseElements": null,
"requestID": "cb8c167e-EXAMPLE",
"eventID": "e3c6f4ce-EXAMPLE",
"readOnly": true,
"eventType": "AwsApiCall",
"apiVersion": "2015-04-13",
"recipientAccountId": "444455556666"
}
```

Beispiel: Ein Logeintrag zum Erstellen eines Repositorys CodeCommit

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag, der die CreateRepository Aktion in der Region USA Ost (Ohio) veranschaulicht.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::444455556666:user/Mary_Major",
    "accountId": "444455556666",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "Mary_Major"
  },
  "eventTime": "2016-12-14T18:19:15Z",
  "eventSource": "codecommit.amazonaws.com",
  "eventName": "CreateRepository",
```



```

"awsRegion": "us-east-2",
"sourceIPAddress": "203.0.113.12",
"userAgent": "aws-cli/1.10.53 Python/2.7.9 Windows/8 botocore/1.4.43",
"requestParameters": {
  "repositoryDescription": "Creating a demonstration repository.",
  "repositoryName": "MyDemoRepo"
},
"responseElements": {
  "repositoryMetadata": {
    "arn": "arn:aws:codecommit:us-east-2:111122223333:MyDemoRepo",
    "creationDate": "Dec 14, 2016 6:19:14 PM",
    "repositoryId": "8afe792d-EXAMPLE",
    "cloneUrlSsh": "ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo",
    "repositoryName": "MyDemoRepo",
    "accountId": "111122223333",
    "cloneUrlHttp": "https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo",
    "repositoryDescription": "Creating a demonstration repository.",
    "lastModifiedDate": "Dec 14, 2016 6:19:14 PM"
  }
},
"requestID": "d148de46-EXAMPLE",
"eventID": "740f179d-EXAMPLE",
"readOnly": false,
"resources": [
  {
    "ARN": "arn:aws:codecommit:us-east-2:111122223333:MyDemoRepo",
    "accountId": "111122223333",
    "type": "AWS::CodeCommit::Repository"
  }
],
"eventType": "AwsApiCall",
"apiVersion": "2015-04-13",
"recipientAccountId": "111122223333"
}

```

Beispiele: Protokolleinträge zum Abrufen von Git-Aufrufen an ein CodeCommit-Repository

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag, der die `GitPull` Aktion demonstriert, bei der sich das lokale Repository bereits up-to-date befindet.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::444455556666:user/Mary_Major",
    "accountId": "444455556666",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "Mary_Major"
  },
  "eventTime": "2016-12-14T18:19:15Z",
  "eventSource": "codecommit.amazonaws.com",
  "eventName": "GitPull",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "203.0.113.12",
  "userAgent": "git/2.11.0.windows.1",
  "requestParameters": null,
  "responseElements": null,
  "additionalEventData": {
    "protocol": "HTTP",
    "dataTransferred": false,
    "repositoryName": "MyDemoRepo",
    "repositoryId": "8afe792d-EXAMPLE",
  },
  "requestID": "d148de46-EXAMPLE",
  "eventID": "740f179d-EXAMPLE",
  "readOnly": true,
  "resources": [
    {
      "ARN": "arn:aws:codecommit:us-east-2:111122223333:MyDemoRepo",
      "accountId": "111122223333",
      "type": "AWS::CodeCommit::Repository"
    }
  ],
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag, der die `GitPull` Aktion demonstriert, bei der sich das lokale Repository nicht befindet up-to-date und somit Daten vom CodeCommit Repository zum lokalen Repository übertragen werden.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::444455556666:user/Mary_Major",
    "accountId": "444455556666",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "Mary_Major"
  },
  "eventTime": "2016-12-14T18:19:15Z",
  "eventSource": "codecommit.amazonaws.com",
  "eventName": "GitPull",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "203.0.113.12",
  "userAgent": "git/2.10.1",
  "requestParameters": null,
  "responseElements": null,
  "additionalEventData": {
    "protocol": "HTTP",
    "capabilities": [
      "multi_ack_detailed",
      "side-band-64k",
      "thin-pack"
    ],
    "dataTransferred": true,
    "repositoryName": "MyDemoRepo",
    "repositoryId": "8afe792d-EXAMPLE",
    "shallow": false
  },
  "requestID": "d148de46-EXAMPLE",
  "eventID": "740f179d-EXAMPLE",
  "readOnly": true,
  "resources": [
    {
      "ARN": "arn:aws:codecommit:us-east-2:111122223333:MyDemoRepo",
      "accountId": "111122223333",
      "type": "AWS::CodeCommit::Repository"
    }
  ],
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```

Beispiel: Ein Logeintrag für einen erfolgreichen Push in ein Repository CodeCommit

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag, der eine erfolgreiche `GitPush` Aktion demonstriert. Die Aktion `GitPush` wird im Protokoll bei einem erfolgreichen Push-Vorgang zweimal angezeigt.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::444455556666:user/Mary_Major",
    "accountId": "444455556666",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "Mary_Major"
  },
  "eventTime": "2016-12-14T18:19:15Z",
  "eventSource": "codecommit.amazonaws.com",
  "eventName": "GitPush",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "203.0.113.12",
  "userAgent": "git/2.10.1",
  "requestParameters": null,
  "responseElements": null,
  "additionalEventData": {
    "protocol": "HTTP",
    "dataTransferred": false,
    "repositoryName": "MyDemoRepo",
    "repositoryId": "8afe792d-EXAMPLE",
  },
  "requestID": "d148de46-EXAMPLE",
  "eventID": "740f179d-EXAMPLE",
  "readOnly": false,
  "resources": [
    {
      "ARN": "arn:aws:codecommit:us-east-2:111122223333:MyDemoRepo",
      "accountId": "111122223333",
      "type": "AWS::CodeCommit::Repository"
    }
  ],
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
},
```

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::444455556666:user/Mary_Major",
    "accountId": "444455556666",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "Mary_Major"
  },
  "eventTime": "2016-12-14T18:19:15Z",
  "eventSource": "codecommit.amazonaws.com",
  "eventName": "GitPush",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "203.0.113.12",
  "userAgent": "git/2.10.1",
  "requestParameters": {
    "references": [
      {
        "commit": "100644EXAMPLE",
        "ref": "refs/heads/main"
      }
    ]
  },
  "responseElements": null,
  "additionalEventData": {
    "protocol": "HTTP",
    "capabilities": [
      "report-status",
      "side-band-64k"
    ],
    "dataTransferred": true,
    "repositoryName": "MyDemoRepo",
    "repositoryId": "8afe792d-EXAMPLE",
  },
  "requestID": "d148de46-EXAMPLE",
  "eventID": "740f179d-EXAMPLE",
  "readOnly": false,
  "resources": [
    {
      "ARN": "arn:aws:codecommit:us-east-2:111122223333:MyDemoRepo",
      "accountId": "111122223333",
      "type": "AWS::CodeCommit::Repository"
    }
  ]
}
```

```
    ],  
    "eventType": "AwsApiCall",  
    "recipientAccountId": "111122223333"  
}
```

CodeCommit Ressourcen erstellen mit AWS CloudFormation

AWS CodeCommit ist integriert in AWS CloudFormation, ein Service, der Ihnen hilft, Ihre AWS Ressourcen zu modellieren und einzurichten, sodass Sie weniger Zeit mit der Erstellung und Verwaltung Ihrer Ressourcen und Infrastruktur verbringen müssen. Sie erstellen eine Vorlage, die alle benötigten AWS Ressourcen (z. B. Repositories) beschreibt und diese Ressourcen für Sie AWS CloudFormation bereitstellt und konfiguriert.

Wenn Sie sie verwenden AWS CloudFormation, können Sie Ihre Vorlage wiederverwenden, um Ihre CodeCommit Ressourcen konsistent und wiederholt einzurichten. Beschreiben Sie Ihre Ressourcen einmal und stellen Sie dann dieselben Ressourcen immer wieder in mehreren AWS-Konten Regionen bereit.

CodeCommit und AWS CloudFormation Vorlagen

Um Ressourcen für und zugehörige Dienste bereitzustellen CodeCommit und zu konfigurieren, müssen Sie sich mit [AWS CloudFormation Vorlagen](#) auskennen. Vorlagen sind formatierte Textdateien in JSON oder YAML. Diese Vorlagen beschreiben die Ressourcen, die Sie in Ihren AWS CloudFormation Stacks bereitstellen möchten. Wenn Sie mit JSON oder YAML nicht vertraut sind, können Sie AWS CloudFormation Designer verwenden, um Ihnen die ersten Schritte mit Vorlagen zu erleichtern. AWS CloudFormation Weitere Informationen finden Sie unter [Was ist AWS CloudFormation -Designer?](#) im AWS CloudFormation -Benutzerhandbuch.

CodeCommit unterstützt das Erstellen von Repositories in AWS CloudFormation Im Gegensatz zum Erstellen von Repositories über die Konsole oder die Befehlszeile können Sie AWS CloudFormation damit Repositories erstellen und automatisch Code aus einer angegebenen .zip-Datei in einem Amazon S3 S3-Bucket in das neu erstellte Repository übertragen. Weitere Informationen, einschließlich Beispielen für JSON- und YAML-Vorlagen für Repositories, finden Sie unter [AWS::CodeCommit::Repository](#)

Wenn Sie ein CodeCommit Repository mit erstellen AWS CloudFormation, haben Sie die Möglichkeit, Code als Teil des Erstellungsprozesses in dieses Repository zu übertragen, sofern das Archiv weniger als 20 MB groß ist, indem Sie Eigenschaften in [AWS::CodeCommit::Repository Code](#) konfigurieren. Sie können den Amazon S3 S3-Bucket angeben, in dem der Code gespeichert ist, und optional die [BranchName Eigenschaft](#) verwenden, um den Namen des Standard-Branche anzugeben, der beim ersten Commit dieses Codes erstellt wird. Diese Eigenschaften werden nur

bei der ersten Repository-Erstellung verwendet und bei Stack-Updates ignoriert. Sie können diese Eigenschaften nicht verwenden, um zusätzliche Commits an ein Repository vorzunehmen oder den Namen des Standard-Branches nach dem ersten Commit zu ändern.

Note

Am 19. Januar 2021 AWS wurde der Name des Standard-Branches CodeCommit von master in main geändert. Diese Namensänderung wirkt sich auf das Standardverhalten aus CodeCommit, wenn der erste Commit für Repositories mithilfe der CodeCommit Konsole, der CodeCommit APIs AWS SDKs, und der AWS CLI erstellt wird. Repositories, die mit AWS CloudFormation oder AWS CDK mit einem ersten Code-Commit im Rahmen der Erstellung erstellt wurden, entsprechen ab dem 4. März 2021 dieser Änderung. Diese Änderung hat keine Auswirkungen auf bestehende Repositories oder Branches. Kunden, die lokale Git-Clients verwenden, um ihre ersten Commits zu erstellen, haben einen Standard-Branch-Namen, der der Konfiguration dieser Git-Clients folgt. Weitere Informationen findest du unter [Mit Branches arbeiten](#), [Einen Commit erstellen](#) und [Branch-Einstellungen ändern](#).

Sie können auch Vorlagen erstellen, mit denen verwandte Ressourcen erstellt werden, z. B. [Benachrichtigungsregeln](#) für Repositories, [AWS CodeBuild Build-Projekte](#), [AWS CodeDeploy Anwendungen](#) und [AWS CodePipeline Pipelines](#).

Beispiele für Vorlagen

In den folgenden Beispielen wird ein CodeCommit Repository mit dem Namen erstellt *MyDemoRepo*. Das neu erstellte Repository wird mit Code gefüllt, der in einem Amazon S3 S3-Bucket gespeichert ist *MySourceCodeBucket* und in einem Branch mit dem Namen platziert wird *development*, der der Standard-Branch für das Repository ist.

Note

Der Name des Amazon S3 S3-Buckets, der die ZIP-Datei mit dem Inhalt enthält, der in das neue Repository übertragen wird, kann mit einem ARN oder dem Namen des Buckets im Amazon Web Services Services-Konto angegeben werden. Der Amazon S3-Objektschlüssel entspricht der Definition im [Amazon S3 Developer Guide](#).

JSON:


```
{
  "MyRepo": {
    "Type": "AWS::CodeCommit::Repository",
    "Properties": {
      "RepositoryName": "MyDemoRepo",
      "RepositoryDescription": "This is a repository for my project with code
from MySourceCodeBucket.",
      "Code": {
        "BranchName": "development",
        "S3": {
          "Bucket": "MySourceCodeBucket",
          "Key": "MyKey",
          "ObjectVersion": "1"
        }
      }
    }
  }
}
```

YAML:

```
MyRepo:
  Type: AWS::CodeCommit::Repository
  Properties:
    RepositoryName: MyDemoRepo
    RepositoryDescription: This is a repository for my project with code from
MySourceCodeBucket.
  Code:
    BranchName: development
    S3:
      Bucket: MySourceCodeBucket,
      Key: MyKey,
      ObjectVersion: 1
```

Weitere Beispiele finden Sie unter [AWS::CodeCommit::Repository](#).

AWS CloudFormation, CodeCommit, und die AWS Cloud Development Kit (AWS CDK)

Repositorys, die bei ihrer Erstellung AWS CDK mithilfe der AWS CloudFormation Use-Funktionalität erstellt wurden. Wenn Sie verstehen, wie AWS CloudFormation Vorlagen mit

CodeCommit Ressourcen funktionieren, können Sie Ihren AWS CDK Code erstellen und verwalten. Weitere Informationen zu finden Sie im AWS CDK [AWS Cloud Development Kit \(AWS CDK\) Entwicklerhandbuch](#) und in der [AWS CDK API-Referenz](#).

Das folgende AWS CDK Typescript-Beispiel erstellt ein CodeCommit Repository mit dem Namen *MyDemoRepo*. Das neu erstellte Repository wird mit Code gefüllt, der in einem Amazon S3 S3-Bucket gespeichert ist *MySourceCodeBucket* und in einem Branch mit dem Namen platziert wird *development*, der der Standard-Branch für das Repository ist.

```
import * as cdk from '@aws-cdk/core';
import codecommit = require('@aws-cdk/aws-codecommit');
export class CdkCodecommitStack extends cdk.Stack {
  constructor(scope: cdk.Construct, id: string, props?: cdk.StackProps) {
    super(scope, id, props);
    // The code creates a CodeCommit repository with a default branch name development
    new codecommit.CfnRepository(this, 'MyRepoResource', {
      repositoryName: "MyDemoRepo",
      code: {
        "branchName": "development",
        "s3": {
          "bucket": "MySourceCodeBucket",
          "key": "MyKey"
        }
      }
    },
  );
}
```

Erfahren Sie mehr über AWS CloudFormation

Weitere Informationen AWS CloudFormation dazu finden Sie in den folgenden Ressourcen:

- [AWS CloudFormation](#)
- [AWS CloudFormation Benutzerhandbuch](#)
- [AWS CloudFormation Benutzerhandbuch für die Befehlszeilenschnittstelle](#)

Problembehebung AWS CodeCommit

Die folgenden Informationen helfen Ihnen möglicherweise bei der Lösung häufiger Probleme in AWS CodeCommit.

Themen

- [Fehlerbehebung bei Git-Anmeldeinformationen und HTTPS-Verbindungen zu AWS CodeCommit](#)
- [Problembehebung git-remote-codecommit und AWS CodeCommit](#)
- [Fehlerbehebung bei SSH-Verbindungen zu AWS CodeCommit](#)
- [Problembehandlung beim Credential Helper und bei HTTPS-Verbindungen zu AWS CodeCommit](#)
- [Problembehandlung bei Git-Clients und AWS CodeCommit](#)
- [Behebung von Zugriffsfehlern und AWS CodeCommit](#)
- [Behebung von Konfigurationsfehlern und AWS CodeCommit](#)
- [Behebung von Konsolenfehlern und AWS CodeCommit](#)
- [Fehlerbehebung bei Auslösern und AWS CodeCommit](#)
- [Schalten Sie das Debuggen ein](#)

Fehlerbehebung bei Git-Anmeldeinformationen und HTTPS-Verbindungen zu AWS CodeCommit

Die folgenden Informationen können Ihnen helfen, häufig auftretende Probleme zu beheben, wenn Sie Git-Anmeldeinformationen und HTTPS verwenden, um eine Verbindung zu AWS CodeCommit Repositorys herzustellen.

Themen

- [Git-Anmeldeinformationen für AWS CodeCommit: Ich werde immer wieder zur Eingabe von Anmeldeinformationen aufgefordert, wenn ich über das Terminal oder die Befehlszeile eine Verbindung zu meinem CodeCommit Repository herstelle](#)
- [Git-Anmeldeinformationen für AWS CodeCommit: Ich habe Git-Anmeldeinformationen eingerichtet, aber mein System verwendet sie nicht](#)

Git-Anmeldeinformationen für AWS CodeCommit: Ich werde immer wieder zur Eingabe von Anmeldeinformationen aufgefordert, wenn ich über das Terminal oder die Befehlszeile eine Verbindung zu meinem CodeCommit Repository herstelle

Problem: Wenn Sie versuchen, ein CodeCommit Repository über das Terminal oder die Befehlszeile zu pushen, abzurufen oder auf andere Weise mit ihm zu interagieren, werden Sie aufgefordert, einen Benutzernamen und ein Passwort einzugeben, und Sie müssen die Git-Anmeldeinformationen für Ihren IAM-Benutzer angeben.

Mögliche Lösungen: Die häufigsten Ursachen für diesen Fehler sind, dass auf Ihrem lokalen Computer ein Betriebssystem ausgeführt wird, das die Verwaltung von Anmeldeinformationen nicht unterstützt, oder dass kein Programm zur Verwaltung von Anmeldeinformationen installiert ist oder dass die Git-Anmeldeinformationen für Ihren IAM-Benutzer nicht in einem dieser Anmeldeinformationsverwaltungssysteme gespeichert wurden. Je nach Ihrem Betriebssystem und der lokalen Umgebung müssen Sie möglicherweise einen Anmeldeinformationsmanager installieren, den in Ihr Betriebssystem integrierten Anmeldeinformationsmanager konfigurieren oder Ihre lokale Umgebung so anpassen, dass Anmeldeinformationen gespeichert werden. Wenn auf Ihrem Computer beispielsweise macOS ausgeführt wird, können Sie das Keychain Access-Hilfsprogramm verwenden, um Ihre Anmeldeinformationen zu speichern. Wenn Sie einen Windows-Computer nutzen, können Sie das Git-Dienstprogramm zur Verwaltung von Anmeldeinformationen (Git Credential Manager) verwenden, das zusammen mit Git für Windows installiert wird. Weitere Informationen finden Sie unter [Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden](#) und [Credential Storage](#) in der Git-Dokumentation.

Git-Anmeldeinformationen für AWS CodeCommit: Ich habe Git-Anmeldeinformationen eingerichtet, aber mein System verwendet sie nicht

Problem: Wenn Sie versuchen, es CodeCommit mit einem Git-Client zu verwenden, scheint der Client die Git-Anmeldeinformationen für Ihren IAM-Benutzer nicht zu verwenden.

Mögliche Korrekturen: Die häufigste Ursache für diesen Fehler besteht darin, dass Sie zuvor die Verwendung des Hilfsprogramms für Anmeldeinformationen, das in der AWS CLI enthalten ist, auf Ihrem Computer eingerichtet haben. Überprüfen Sie Ihre GITCONFIG-Datei auf Konfigurationsabschnitte, die dem folgenden ähneln, und entfernen Sie sie:

```
[credential "https://git-codecommit.*.amazonaws.com"]
```

```
helper = !aws codecommit credential-helper $@  
UseHttpPath = true
```

Speichern Sie die Datei und öffnen Sie anschließend eine neue Befehlszeilen- oder Terminalsitzung, bevor Sie einen erneuten Verbindungsversuch unternehmen.

Möglicherweise wurden auch mehrere Hilfs- oder Verwaltungsprogramme für Anmeldeinformationen auf Ihrem Computer eingerichtet und Ihr System verwendet standardmäßig eine andere Konfiguration. Um die Einstellung zurückzusetzen, welches Hilfsprogramm für Anmeldeinformationen standardmäßig verwendet wird, können Sie die Option `--system` anstelle von `--global` oder `--local` verwenden, wenn Sie den Befehl `git config` ausführen.

Weitere Informationen finden Sie unter [Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden](#) und [Credential Storage](#) in der Git-Dokumentation.

Problembehebung git-remote-codecommit und AWS CodeCommit

Die folgenden Informationen können Ihnen helfen, Probleme mit git-remote-codecommit zu beheben, die bei der Verbindung mit AWS CodeCommit -Repositorys auftreten können.

Themen

- [Ich sehe einen Fehler: git: 'remote-codecommit' ist kein Git-Befehl](#)
- [Mir wird ein Fehler angezeigt: Schwerwiegender Fehler: Der Remote-Helper für 'codecommit' konnte nicht gefunden werden](#)
- [Fehler beim Klonen: Ich kann ein CodeCommit Repository nicht von einer IDE klonen](#)
- [Push- oder Pull-Fehler: Ich kann keine Commits per Push oder Pull von einer IDE in ein CodeCommit-Repository verschieben](#)
- [Installationsfehler: Beim Versuch, die Installation durchzuführen, wird ein externally-managed-environment Fehler angezeigt git-remote-codecommit](#)

Ich sehe einen Fehler: git: 'remote-codecommit' ist kein Git-Befehl

Problem: Wenn Sie versuchen, es zu verwenden git-remote-codecommit, wird ein Fehler angezeigt, bei dem es git-remote-codecommit sich nicht um einen Git-Befehl handelt. Siehe 'git --help'.

Mögliche Lösungen: Der häufigste Grund für diesen Fehler ist, dass Sie die git-remote-codecommit ausführbare Datei entweder nicht zu Ihrem PATH hinzugefügt haben oder dass die Zeichenfolge

einen Syntaxfehler enthält. Das kann passieren, wenn ein Bindestrich zwischen Git und remote-codecommit fehlt oder wenn ein zusätzlicher Git davor steht. `git-remote-codecommit`

Dieser Fehler kann auch auftreten, wenn Sie Ihre lokale Installation auf eine nicht unterstützte Version aktualisiert haben. Weitere Hinweise zu unterstützten Versionen von Python finden Sie unter [git-remote-codecommit](#).

Weitere Informationen zur Einrichtung und Verwendung `git-remote-codecommit` finden Sie unter [Einrichtungsschritte für HTTPS-Verbindungen AWS CodeCommit mit git-remote-codecommit](#).

Mir wird ein Fehler angezeigt: Schwerwiegender Fehler: Der Remote-Helper für 'codecommit' konnte nicht gefunden werden

Problem: Wenn Sie versuchen, es zu verwenden `git-remote-codecommit`, wird die Fehlermeldung „fatal: Der Remote-Helper für 'Codecommit' konnte nicht gefunden werden“ angezeigt.

Mögliche Lösungen: Die häufigsten Gründe für diesen Fehler sind:

- Die Einrichtung ist nicht abgeschlossen für `git-remote-codecommit`
- Sie haben die Installation `git-remote-codecommit` an einem Ort durchgeführt, der nicht in Ihrem Pfad liegt oder der nicht als Teil der Path Umgebungsvariablen konfiguriert ist
- Python befindet sich nicht in Ihrem Pfad oder ist nicht als Teil der Path Umgebungsvariablen konfiguriert
- Sie verwenden ein Terminal- oder Befehlszeilenfenster, das seit Abschluss der `git-remote-codecommit` Installation nicht neu gestartet wurde

Weitere Informationen zur Einrichtung und Verwendung finden [Einrichtungsschritte für HTTPS-Verbindungen AWS CodeCommit mit git-remote-codecommit](#) Sie `git-remote-codecommit` unter.

Fehler beim Klonen: Ich kann ein CodeCommit Repository nicht von einer IDE klonen

Problem: Wenn Sie versuchen, ein CodeCommit Repository in einer IDE zu klonen, wird eine Fehlermeldung angezeigt, die besagt, dass der Endpunkt oder die URL nicht gültig ist.

Mögliche Lösungen: Nicht alle IDEs unterstützen die URL, die `git-remote-codecommit` beim Klonen verwendet wurde. Klonen Sie das Repository lokal über das Terminal oder die Befehlszeile, und

fügen Sie Ihrer IDE das lokale Repo hinzu. Weitere Informationen finden Sie unter [Schritt 3: Connect zur CodeCommit Konsole her und klonen Sie das Repository](#).

Push- oder Pull-Fehler: Ich kann keine Commits per Push oder Pull von einer IDE in ein CodeCommit-Repository verschieben

Problem: Wenn Sie versuchen, Code per Push oder Pull von einer IDE zu verschieben, wird ein Verbindungsfehler angezeigt.

Mögliche Lösungen: Der häufigste Grund für diesen Fehler ist, dass die IDE nicht mit Git-Remote-Helfern wie `git-remote-codecommit` kompatibel ist. Anstatt die IDE-Funktionalität zu verwenden, um Commit, Push und Pull für Code durchzuführen, aktualisieren Sie das lokale Repo manuell über die Befehlszeile oder das Terminal mithilfe von Git-Befehlen.

Weitere Informationen zu Remote-Helfern und Git finden Sie in der [Git-Dokumentation](#).

Installationsfehler: Beim Versuch, die Installation durchzuführen, wird ein `externally-managed-environment` Fehler angezeigt `git-remote-codecommit`

Problem: Wenn Sie versuchen, den `pip install git-remote-codecommit` Befehl auszuführen, wird eine Fehlermeldung angezeigt, die darauf hinweist, dass die Umgebung extern verwaltet wird.

Mögliche Lösungen: Der häufigste Grund für diesen Fehler ist, dass Sie eine Python-Distribution (Distribution) ausführen, die eine `EXTERN VERWALTETE` Markerdatei definiert. Die beste Lösung hierfür besteht darin, eine virtuelle Umgebung zu erstellen und zu verwenden.

Weitere Informationen zu extern verwalteten Umgebungen und Python finden Sie unter [Extern verwaltete Umgebungen](#) und [Installieren von Paketen in einer virtuellen Umgebung mit pip und venv](#) in der Python-Dokumentation.

Fehlerbehebung bei SSH-Verbindungen zu AWS CodeCommit

Die folgenden Informationen können Ihnen bei der Behebung häufiger Probleme bei der Verwendung von SSH zur CodeCommit Verbindung mit Repositories helfen.

Themen

- [Zugriffsfehler: Der öffentliche Schlüssel wurde erfolgreich in IAM hochgeladen, aber die Verbindung schlägt auf Linux-, macOS- oder Unix-Systemen fehl](#)

- [Zugriffsfehler: Der öffentliche Schlüssel wurde erfolgreich in IAM hochgeladen und SSH wurde erfolgreich getestet, aber die Verbindung schlägt auf Windows-Systemen fehl](#)
- [Authentifizierungsherausforderung: Die Authentizität des Hosts kann nicht festgestellt werden, wenn eine Verbindung zu einem Repository hergestellt wird CodeCommit](#)
- [IAM-Fehler: „Ungültiges Format“ beim Versuch, IAM einen öffentlichen Schlüssel hinzuzufügen](#)
- [Ich muss mit SSH-Anmeldeinformationen auf CodeCommit Repositories in mehreren Amazon Web Services Services-Konten zugreifen](#)
- [Git unter Windows: Bash-Emulator oder Befehlszeilenfenster stürzt bei Verbindungsversuchen mit SSH ab](#)
- [Das Public-Key-Format erfordert in einigen Linux-Distributionen eine Spezifikation](#)
- [Zugriffsfehler: Der öffentliche SSH-Schlüssel wurde beim Herstellen einer Verbindung zu einem CodeCommit Repository verweigert](#)

Zugriffsfehler: Der öffentliche Schlüssel wurde erfolgreich in IAM hochgeladen, aber die Verbindung schlägt auf Linux-, macOS- oder Unix-Systemen fehl

Problem: Wenn Sie versuchen, eine Verbindung zu einem SSH-Endpunkt herzustellen, um mit einem CodeCommit Repository zu kommunizieren, entweder beim Testen der Verbindung oder beim Klonen eines Repositories, schlägt die Verbindung fehl oder wird abgelehnt.

Mögliche Lösungen: Die SSH-Schlüssel-ID, die Ihrem öffentlichen Schlüssel in IAM zugewiesen wurde, ist möglicherweise nicht mit Ihrem Verbindungsversuch verknüpft. [Möglicherweise haben Sie keine Konfigurationsdatei konfiguriert, Sie haben möglicherweise keinen Zugriff auf die Konfigurationsdatei, eine](#) andere Einstellung verhindert möglicherweise ein erfolgreiches Lesen der Konfigurationsdatei, Sie haben möglicherweise die falsche Schlüssel-ID angegeben oder Sie haben anstelle der Schlüssel-ID die ID des IAM-Benutzers angegeben.

Die SSH-Schlüssel-ID finden Sie in der IAM-Konsole im Profil Ihres IAM-Benutzers:



SSH keys for AWS CodeCommit			
Use SSH public keys to authenticate to AWS CodeCommit repositories. Learn more about SSH keys.			
Upload SSH public key			
SSH Key ID	Uploaded	Status	Actions
APKAEIBAERJR2EXAMPLE	2015-07-21 16:32 PDT	Active	Make inactive Show SSH Key Delete

 Note

Wenn Sie mehr als einen SSH-Schlüssel IDs hochgeladen haben, werden die Schlüssel alphabetisch nach der Schlüssel-ID und nicht nach dem Upload-Datum aufgelistet. Stellen Sie sicher, dass Sie die Schlüssel-ID kopiert haben, die dem richtigen Upload-Datum zugeordnet ist.

Testen Sie versuchsweise die Verbindung mit dem folgenden Befehl:

```
ssh Your-SSH-Key-ID@git-codecommit.us-east-2.amazonaws.com
```

Wenn nach der Bestätigung der Verbindung eine Erfolgsmeldung angezeigt wird, ist Ihre SSH-Schlüssel-ID gültig. Bearbeiten Sie Ihre Konfigurationsdatei, um Ihre Verbindungsversuche mit Ihrem öffentlichen Schlüssel in IAM zu verknüpfen. Wenn Sie die Config-Datei nicht bearbeiten möchten, können Sie allen Verbindungsversuchen mit Ihrem Repository Ihre SSH-Schlüssel-ID voranstellen. Wenn Sie beispielsweise ein Repository mit dem Namen klonen möchten, *MyDemoRepo* ohne Ihre Konfigurationsdatei zu ändern, um Ihre Verbindungsversuche zu verknüpfen, würden Sie den folgenden Befehl ausführen:

```
git clone ssh://Your-SSH-Key-ID@git-codecommit.us-east-2.amazonaws.com/v1/  
repos/MyDemoRepo my-demo-repo
```

Weitere Informationen finden Sie unter [Für SSH-Verbindungen unter Linux, macOS oder Unix](#).

Zugriffsfehler: Der öffentliche Schlüssel wurde erfolgreich in IAM hochgeladen und SSH wurde erfolgreich getestet, aber die Verbindung schlägt auf Windows-Systemen fehl

Problem: Wenn Sie versuchen, einen SSH-Endpunkt zum Klonen oder Kommunizieren mit einem CodeCommit Repository zu verwenden, wird eine Fehlermeldung mit dem folgenden Satz angezeigt.
No supported authentication methods available

Mögliche Lösungen: Die häufigste Ursache für diesen Fehler ist das Vorhandensein einer Windows-Systemumgebungsvariablen, die Windows anweist, ein anderes Programm zu verwenden, wenn Sie SSH zu verwenden versuchen. Möglicherweise verweist auf Ihrem System z. B. die Variable GIT_SSH auf eines der PuTTY-Tools (plink.exe). Dabei kann es sich um eine veraltete Konfiguration

handeln oder aber auch um eine notwendige Einstellung für ein oder mehrere Programme auf Ihrem Computer. Wenn Sie sicher sind, dass diese Umgebungsvariable nicht benötigt wird, können Sie sie entfernen. Öffnen Sie dazu die Systemeigenschaften.

Um dieses Problem zu umgehen, öffnen Sie einen Bash-Emulator und versuchen Sie erneut, die SSH-Verbindung herzustellen. Geben Sie dabei jedoch `GIT_SSH_COMMAND="SSH"` als Präfix an. Beispiel zum Klonen eines Repositorys mit SSH:

```
GIT_SSH_COMMAND="ssh" git clone ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
```

Ein ähnliches Problem kann auftreten, wenn bei Ihrer Windows-Version bei Verbindungen mithilfe von SSH über die Windows-Befehlszeile die SSH-Schlüssel-ID als Teil der Verbindungszeichenfolge angegeben werden muss. Versuchen Sie erneut, die Verbindung herzustellen, diesmal mit der SSH-Schlüssel-ID, die als Teil des Befehls aus IAM kopiert wurde. Zum Beispiel:

```
git clone ssh://Your-SSH-Key-ID@git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
```

Authentifizierungsherausforderung: Die Authentizität des Hosts kann nicht festgestellt werden, wenn eine Verbindung zu einem Repository hergestellt wird CodeCommit

Problem: Wenn Sie versuchen, einen SSH-Endpunkt für die Kommunikation mit einem CodeCommit Repository zu verwenden, wird eine Warnmeldung mit dem folgenden Satz angezeigt: The authenticity of host '*host-name*' can't be established.

Mögliche Lösungen: Ihre Anmeldeinformationen sind möglicherweise nicht richtig eingerichtet. Befolgen Sie die Anweisungen unter [Für SSH-Verbindungen unter Linux, macOS oder Unix](#) oder [Für SSH-Verbindungen unter Windows](#).

Wenn Sie diese Schritte befolgt haben und das Problem weiterhin besteht, versucht möglicherweise jemand einen man-in-the-middle Angriff. Wenn die folgende Meldung angezeigt wird, geben Sie `no` ein und drücken Sie die Eingabetaste:

```
Are you sure you want to continue connecting (yes/no)?
```

Stellen Sie sicher, dass der Fingerabdruck und der öffentliche Schlüssel für CodeCommit Verbindungen mit denen übereinstimmen, die in den Themen zur SSH-Setup dokumentiert sind, bevor Sie mit der Verbindung fortfahren.

Öffentliche Fingerabdrücke für CodeCommit

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.us-east-2.amazonaws.com	MD5	a9:6d:03:ed:08:42: 21:be:06:e1:e0:2a: d1:75:31:5e
git-codecommit.us-east-2.amazonaws.com	SHA256	3lB1W2g5xn/NA2Ck6d yeJIrQ0Wvn7n8UEs56 fG6ZIZQ
git-codecommit.us-east-1.amazonaws.com	MD5	a6:9c:7d:bc:35:f5: d4:5f:8b:ba:6f:c8: bc:d4:83:84
git-codecommit.us-east-1.amazonaws.com	SHA256	eLMY1j0DKA4uvDZc1/ KgtIayZANwX6t8+8is PtotBoY
git-codecommit.us-west-2.amazonaws.com	MD5	a8:68:53:e3:99:ac: 6e:d7:04:7e:f7:92: 95:77:a9:77
git-codecommit.us-west-2.amazonaws.com	SHA256	0pJx9SQpkbPUAHwy58 UVIq0IHcyo1fwCp00u VgcAWPo
git-codecommit.eu-west-1.amazonaws.com	MD5	93:42:36:ea:22:1f: f1:0f:20:02:4a:79: ff:ea:12:1d
git-codecommit.eu-west-1.amazonaws.com	SHA256	tKjRk0L8dmJyTmSbeS dN1S8F/f0iq13R1vqg TOP1UyQ

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.ap-northeast-1.amazonaws.com	MD5	8e:a3:f0:80:98:48: 1c:5c:6f:59:db:a7: 8f:6e:c6:cb
git-codecommit.ap-northeast-1.amazonaws.com	SHA256	Xk/WeYD/K/bnBybzhi uu4dWpBJtXPf7E30jH U7se40w
git-codecommit.ap-southeast-1.amazonaws.com	MD5	65:e5:27:c3:09:68: 0d:8e:b7:6d:94:25: 80:3e:93:cf
git-codecommit.ap-southeast-1.amazonaws.com	SHA256	ZISVa70VzxrTIf+Rk4 UbhPv6Es22mSB3uTBo jfPXIno
git-codecommit.ap-southeast-2.amazonaws.com	MD5	7b:d2:c1:24:e6:91: a5:7b:fa:c1:0c:35: 95:87:da:a0
git-codecommit.ap-southeast-2.amazonaws.com	SHA256	nYp+gHas80HY3DqbP4 yanCDFhqDVjseefVbH EXqH2Ec
git-codecommit.ap-southeast-3.amazonaws.com	MD5	64:d9:e0:53:19:4f: a8:91:9a:c3:53:22: a6:a8:ed:a6
git-codecommit.ap-southeast-3.amazonaws.com	SHA256	ATdkGSFhpqIu7RqUVT /1RZo6MLxxxUW9NoDV MbAc/6g
git-codecommit.me-central-1.amazonaws.com	MD5	bd:fa:e2:f9:05:84: d6:39:6f:bc:d6:8d: fe:de:61:76

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.me-central-1.amazonaws.com	SHA256	grceUDWubo4MzG1Noa KZKUfrgPvfN3ijli0n Qr1lTZA
git-codecommit.eu-central-1.amazonaws.com	MD5	74:5a:e8:02:fc:b2: 9c:06:10:b4:78:84: 65:94:22:2d
git-codecommit.eu-central-1.amazonaws.com	SHA256	MwGrkiEki8QkkBtlAg XbYt0hoZYBnZF62VY5 RzGJEUy
git-codecommit.ap-northeast-2.amazonaws.com	MD5	9f:68:48:9b:5f:fc: 96:69:39:45:58:87: 95:b3:69:ed
git-codecommit.ap-northeast-2.amazonaws.com	SHA256	eegAPQrWY9YsYo9ZHI K0mxeTfXBHzAZd8Eya 53Qcwko
git-codecommit.sa-east-1.amazonaws.com	MD5	74:99:9d:ff:2b:ef: 63:c6:4b:b4:6a:7f: 62:c5:4b:51
git-codecommit.sa-east-1.amazonaws.com	SHA256	kW+VKB0jpRaG/ZbXkg btMQbKgEDK7JnISV3S VoyCmzU
git-codecommit.us-west-1.amazonaws.com	MD5	3b:76:18:83:13:2c: f8:eb:e9:a3:d0:51: 10:32:e7:d1
git-codecommit.us-west-1.amazonaws.com	SHA256	gzauWTWXDK2u5KuMMi 5vbKTmfyerdIwgSbzY BODLpzg

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.eu-west-2.amazonaws.com	MD5	a5:65:a6:b1:84:02: b1:95:43:f9:0e:de: dd:ed:61:d3
git-codecommit.eu-west-2.amazonaws.com	SHA256	r0Rwz5k/IHp/QyrRnf iM9j02D5UEqMbtFNTu DG2hNbs
git-codecommit.ap-south-1.amazonaws.com	MD5	da:41:1e:07:3b:9e: 76:a0:c5:1e:64:88: 03:69:86:21
git-codecommit.ap-south-1.amazonaws.com	SHA256	hUKwnTj7+Xpx4Kddb6 p45j4RazIJ4IhAMD8k 29it0fE
git-codecommit.ap-south-2.amazonaws.com	MD5	bc:cc:9f:15:f8:f3: 58:a2:68:65:21:e2: 23:71:8d:ce
git-codecommit.ap-south-2.amazonaws.com	SHA256	Xe0CyZE0vgR5Xa2YUG qf+jn8/Ut7l7nX/Cms lSFNEig
git-codecommit.ca-central-1.amazonaws.com	MD5	9f:7c:a2:2f:8c:b5: 74:fd:ab:b7:e1:fd: af:46:ed:23
git-codecommit.ca-central-1.amazonaws.com	SHA256	Qz5puafQdANVprLlj6 r0Qyh4lCNsF6ob61dG cPtFS7w
git-codecommit.eu-west-3.amazonaws.com	MD5	1b:7f:97:dd:d7:76: 8a:32:2c:bd:2c:7b: 33:74:6a:76

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.eu-west-3.amazonaws.com	SHA256	uw7c2FL564jVoFgtc+ ikzILnKBsZz7t9+CFd SJjKbLI
Git-Code-Commit. us-gov-west-1.amazonaws.com	MD5	9f:6c:19:3b:88:cd: e8:88:1b:9c:98:6a: 95:31:8a:69
Git-Code-Commit. us-gov-west-1.amazonaws.com	SHA256	djXQoSIFcg8vHe0KVH 1xW/g0F9X37tWTqu4H kng75x4
Git-Code-Commit. us-gov-east-1.amazonaws.com	MD5	00:8d:b5:55:6f:05: 78:05:ed:ea:cb:3f: e6:f0:62:f2
Git-Code-Commit. us-gov-east-1.amazonaws.com	SHA256	fVb+R0z7qW7minenW+ rUpAABRCRBTCzmETAJ EQrg98
git-codecommit.eu-north-1.amazonaws.com	MD5	8e:53:d8:59:35:88: 82:fd:73:4b:60:8a: 50:70:38:f4
git-codecommit.eu-north-1.amazonaws.com	SHA256	b6KSK7xKq+V8jl7iuA cjQXsG7zkqoUZZmmhY YFBq1wQ
git-codecommit.me-south-1.amazonaws.com	MD5	0e:39:28:56:d5:41: e6:8d:fa:81:45:37: fb:f3:cd:f7
git-codecommit.me-south-1.amazonaws.com	SHA256	0+NToCGgjRHeKiBu0l 0ad7R0GEsz+DBLX0d/ c9wc0JU

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.ap-east-1.amazonaws.com	MD5	a8:00:3d:24:52:9d: 61:0e:f6:e3:88:c8: 96:01:1c:fe
git-codecommit.ap-east-1.amazonaws.com	SHA256	LafadYwUYW8h0NoTRp objbjNs9IRnbEwHtezD 3aAIBX0
git-codecommit.cn-north-1.amazonaws.com.cn	MD5	11:7e:2d:74:9e:3b: 94:a2:69:14:75:6f: 5e:22:3b:b3
git-codecommit.cn-north-1.amazonaws.com.cn	SHA256	IYUXxH20pTDsyYMLIp +JY8CTLS4UX+ZC5JVZ XPRaxc8
git-codecommit.cn-northwest-1.amazonaws.com.cn	MD5	2e:a7:fb:4c:33:ac: 6c:f9:aa:f2:bc:fb: 0a:7b:1e:b6
git-codecommit.cn-northwest-1.amazonaws.com.cn	SHA256	wqjd6eHd0+m0Bx+dCN uL0omUoCNjaDtZiEpW j5TmCfQ
git-codecommit.eu-south-1.amazonaws.com	MD5	b9:f6:5d:e2:48:92: 3f:a9:37:1e:c4:d0: 32:0e:fb:11
git-codecommit.eu-south-1.amazonaws.com	SHA256	1yXrWbCg3uQmJr11Xx B/ASR7ugW1Ysf5yzY0 JbudHsI
git-codecommit.ap-northeast-3.amazonaws.com	MD5	25:17:40:da:b9:d4: 18:c3:b6:b3:fb:ed: 1c:20:fe:29

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.ap-northeast-3.amazonaws.com	SHA256	2B815B9F0AvwLnRxSV xUz4kDYmtEQUGGdQYP 80QLXhA
git-codecommit.af-south-1.amazonaws.com	MD5	21:a0:ba:d7:c1:d1: b5:39:98:8d:4d:7c: 96:f5:ca:29
git-codecommit.af-south-1.amazonaws.com	SHA256	C34ji3x/cnsDZjUpyN GXdE5pjHYimqJrQZ3l eTgqJHM
git-codecommit.il-central-1.amazonaws.com	MD5	04:74:89:16:98:7a: 61:b1:69:46:42:3c: d1:b4:ac:a9
git-codecommit.il-central-1.amazonaws.com	SHA256	uFxhp51kUWh1eTLeYb xQVYm4RnNLNZ5Dbdm1 cgdS1/8

IAM-Fehler: „Ungültiges Format“ beim Versuch, IAM einen öffentlichen Schlüssel hinzuzufügen

Problem: Wenn Sie in IAM versuchen, die Verwendung von SSH mit einzurichten, wird eine Fehlermeldung angezeigt CodeCommit, die den Satz enthält, `Invalid format` wenn Sie versuchen, Ihren öffentlichen Schlüssel hinzuzufügen.

Mögliche Korrekturen: IAM erfordert, dass der öffentliche Schlüssel im `ssh-rsa`-Format oder `PEM`-Format codiert sein muss. Es akzeptiert nur öffentliche Schlüssel im `OpenSSH`-Format und hat zusätzliche Anforderungen, wie unter [SSH-Schlüssel verwenden mit CodeCommit](#) im IAM-Benutzerhandbuch beschrieben. Wenn Sie den öffentlichen Schlüssel in einem anderen Format bereitstellen oder wenn der Schlüssel nicht die erforderliche Bitzahl enthält, wird diese Fehlermeldung angezeigt.

- Beim Kopieren der öffentlichen SSH-Schlüssel wurden durch Ihr Betriebssystem möglicherweise Zeilenumbrüche eingeführt. Stellen Sie sicher, dass der öffentliche Schlüssel, den Sie zu IAM hinzufügen, keine Zeilenumbrüche enthält.
- Einige Windows-Betriebssysteme verwenden das OpenSSH-Format nicht. Informationen zum Generieren eines key pair und zum Kopieren des von IAM benötigten OpenSSH-Formats finden Sie unter: [the section called “Schritt 3: Richten Sie die öffentlichen und privaten Schlüssel für Git ein und CodeCommit”](#)

Weitere Informationen zu den Anforderungen für SSH-Schlüssel in IAM finden Sie unter [Use SSH Keys with CodeCommit](#) im IAM-Benutzerhandbuch.

Ich muss mit SSH-Anmeldeinformationen auf CodeCommit Repositorys in mehreren Amazon Web Services Services-Konten zugreifen

Problem: Ich möchte SSH-Zugriff auf CodeCommit Repositorys in mehr als einem Amazon Web Services Services-Konto einrichten.

Mögliche Lösungen: Sie können eine eindeutige public/private key pairs for each Amazon Web Services account and configure IAM with each key. You can then configure your ~/.ssh/config SSH-Datei mit Informationen zu jeder IAM-Benutzer-ID erstellen, die dem öffentlichen Schlüssel zugeordnet ist. Zum Beispiel:

```
Host codecommit-1
  Hostname git-codecommit.us-east-1.amazonaws.com
  User SSH-KEY-ID-1 # This is the SSH Key ID you copied from IAM in Amazon Web
  Services account 1 (for example, APKAEIBAERJR2EXAMPLE1).
  IdentityFile ~/.ssh/codecommit_rsa # This is the path to the associated public key
  file, such as id_rsa. We advise creating CodeCommit specific _rsa files.

Host codecommit-2
  Hostname git-codecommit.us-east-1.amazonaws.com
  User SSH-KEY-ID-2 # This is the SSH Key ID you copied from IAM in Amazon Web
  Services account 2 (for example, APKAEIBAERJR2EXAMPLE2).
  IdentityFile ~/.ssh/codecommit_2_rsa # This is the path to the other associated
  public key file. We advise creating CodeCommit specific _rsa files.
```

In dieser Konfiguration können Sie 'git-codecommit.us-east-1.amazonaws.com' durch 'codecommit-2' ersetzen. Um beispielsweise ein Repository in Ihrem zweiten Amazon Web Services Services-Konto zu klonen:

```
git clone ssh://codecommit-2/v1/repos/YourRepositoryName
```

Um eine Remote-Verbindung für Ihr Repository einzurichten, führen Sie den Befehl `git remote add`. Zum Beispiel:

```
git remote add origin ssh://codecommit-2/v1/repos/YourRepositoryName
```

Weitere Beispiele finden Sie in [diesem Forenbeitrag](#) und in [diesem Beitrag unter GitHub](#).

Git unter Windows: Bash-Emulator oder Befehlszeilenfenster stürzt bei Verbindungsversuchen mit SSH ab

Problem: Nachdem Sie SSH-Zugriff für Windows konfiguriert und Konnektivität im Befehlszeilen- oder Terminalfenster überprüft haben, wird eine Meldung angezeigt, dass der Host-Schlüssel des Servers nicht in der Registry zwischengespeichert wurde. Außerdem reagiert die Eingabeaufforderung zum Speichern des Schlüssels im Cache nicht (y/n/Eingabetaste wird nicht akzeptiert), wenn Sie versuchen, Befehle wie zum Beispiel `git pull`, `git push` oder `git clone` an der Eingabeaufforderung oder im Bash-Emulator zu verwenden.

Mögliche Lösungen: Die häufigste Ursache für diesen Fehler ist, dass Ihre Git-Umgebung zur Verwendung einer anderen Authentifizierungsmethode als OpenSSH konfiguriert ist (möglicherweise PuTTY). Es ist bekannt, dass dies in einigen Konfigurationen Probleme mit der Zwischenspeicherung von Schlüsseln verursachen kann. Um dieses Problem zu beheben, versuchen Sie es mit einer der folgenden Methoden:

- Öffnen Sie einen Bash-Emulator und fügen Sie den Parameter `GIT_SSH_COMMAND="ssh"` vor dem Git-Befehl hinzu. Wenn Sie zum Beispiel versuchen, ein Repository per Push zu übertragen, geben Sie anstelle von `git push` den folgenden Befehl ein:

```
GIT_SSH_COMMAND="ssh" git push
```

- Wenn Sie PuTTY installiert haben, öffnen Sie PuTTY und geben Sie unter Hostname (oder IP-Adresse) den CodeCommit Endpunkt ein, den Sie erreichen möchten (z. B. `git-codecommit.us-east-2.amazonaws.com`). Klicken Sie auf Open. Wenn Sie eine Eingabeaufforderung von PuTTY Security Alert erhalten, wählen Sie Yes (Ja), um den Schlüssel dauerhaft zwischenzuspeichern.
- Löschen Sie die Umgebungsvariable `GIT_SSH` oder benennen Sie sie um, wenn Sie sie nicht mehr verwenden. Öffnen Sie anschließend ein neues Befehlszeilenfenster oder eine neue Bash-Emulator-Sitzung und versuchen Sie erneut, den Befehl einzugeben.

Weitere Lösungen finden Sie unter [Git clone/pull continually freezing at Store key in cache](#) auf Stack Overflow.

Das Public-Key-Format erfordert in einigen Linux-Distributionen eine Spezifikation

Problem: Wenn Sie versuchen, ein öffentlich-privates key pair zu konfigurieren, erhalten Sie eine Fehlermeldung.

Mögliche Lösungen: Einige Linux-Distributionen erfordern eine zusätzliche Konfigurationszeile in der `~/.ssh/config` Datei, die die akzeptierten Typen von öffentlichen Schlüsseln spezifiziert. Weitere Informationen finden Sie in der Dokumentation zu Ihrer Distribution über `PubkeyAcceptedKeyTypes`.

Zugriffsfehler: Der öffentliche SSH-Schlüssel wurde beim Herstellen einer Verbindung zu einem CodeCommit Repository verweigert

Problem: Wenn Sie versuchen, einen SSH-Endpunkt für die Kommunikation mit einem CodeCommit Repository zu verwenden, wird eine Fehlermeldung mit dem folgenden Satz angezeigt. `Error: public key denied`

Mögliche Fehlerbehebungen: Der häufigste Grund für diesen Fehler ist, dass Sie die Einrichtung für die SSH-Verbindungen nicht abgeschlossen haben. Konfigurieren Sie ein öffentliches und ein privates SSH-Schlüsselpaar und ordnen Sie den öffentlichen Schlüssel dann Ihrem IAM-Benutzer zu. Weitere Informationen zur Konfiguration von SSH finden Sie unter [Für SSH-Verbindungen unter Linux, macOS oder Unix](#) und [Für SSH-Verbindungen unter Windows](#).

Problembehandlung beim Credential Helper und bei HTTPS-Verbindungen zu AWS CodeCommit

Die folgenden Informationen können Ihnen helfen, häufig auftretende Probleme zu beheben, wenn Sie den in den AWS CLI und HTTPS enthaltenen Credential Helper verwenden, um eine Verbindung zu CodeCommit Repositories herzustellen.

Note

Der Credential Helper ist zwar eine unterstützte Methode, um CodeCommit mithilfe von Federated Access, einem Identity Provider oder temporären Anmeldeinformationen

eine Verbindung herzustellen, es wird jedoch empfohlen, das Programm zu installieren und zu verwenden. `git-remote-codecommit` Weitere Informationen finden Sie unter [Einrichtungsschritte für HTTPS-Verbindungen AWS CodeCommit mit `git-remote-codecommit`](#).

Themen

- [Ich erhalte eine Fehlermeldung, wenn ich den `git config` Befehl zur Konfiguration des Credential Helper ausführe](#)
- [Beim Versuch, ein Repository zu klonen, wird beim Lesen des Benutzernamens ein Fehler angezeigt](#)
- [Zurückgegebene Fehlermeldung "Befehl nicht gefunden" unter Windows bei Verwendung des Hilfsprogramms für Anmeldeinformationen](#)
- [Ich werde zur Eingabe eines Benutzernamens aufgefordert, wenn ich eine Verbindung zu einem Repository herstelle CodeCommit](#)
- [Git für macOS: Ich habe den Credential Helper erfolgreich konfiguriert, aber jetzt wird mir der Zugriff auf mein Repository verweigert \(403\)](#)
- [Git für Windows: Ich habe Git für Windows installiert, aber mir wird der Zugriff auf mein Repository verweigert \(403\)](#)

Ich erhalte eine Fehlermeldung, wenn ich den **git config** Befehl zur Konfiguration des Credential Helper ausführe

Problem: Wenn Sie versuchen, den Befehl `Git config` auszuführen, um den Credential Helper für die Kommunikation mit einem CodeCommit Repository zu konfigurieren, erhalten Sie eine Fehlermeldung, dass zu wenige Argumente vorhanden sind, oder eine Eingabeaufforderung, die Git-Konfigurationsbefehle und Syntax vorschlägt.

Mögliche Lösungen: Der häufigste Grund für diesen Fehler ist, dass entweder einfache Anführungszeichen für den Befehl auf einem Windows-Betriebssystem oder doppelte Anführungszeichen für den Befehl in einem Linux-, MacOS- oder Unix-Betriebssystem verwendet werden. Die korrekte Syntax lautet wie folgt:

- Windows: `git config --global credential.helper "!aws codecommit credential-helper %@"`

- Linux, macOS oder Unix: `git config --global credential.helper '!aws codecommit credential-helper $@'`

Beim Versuch, ein Repository zu klonen, wird beim Lesen des Benutzernamens ein Fehler angezeigt

Problem: Wenn Sie versuchen, ein Repository mit dem Credential Helper zu klonen, wird eine Fehlermeldung angezeigt, die besagt, dass das System den Benutzernamen für die Repository-URL nicht lesen konnte.

Mögliche Lösungen: Der häufigste Grund für diesen Fehler ist, dass die `.gitconfig`-Datei auf Ihrem Computer nicht korrekt erstellt oder konfiguriert wurde. Öffnen Sie Ihre `.gitconfig`-Datei und stellen Sie sicher, dass der Credential Helper richtig eingestellt ist. Wenn Sie einen Computer verwenden, auf dem Linux, macOS oder Unix ausgeführt wird, stellen Sie außerdem sicher, dass der Wert von `$HOME` für Ihr System richtig eingestellt ist.

Zurückgegebene Fehlermeldung "Befehl nicht gefunden" unter Windows bei Verwendung des Hilfsprogramms für Anmeldeinformationen

Problem: Nach der Aktualisierung der AWS CLI schlagen Credential Helper-Verbindungen zu CodeCommit Repositorys fehl. `aws codecommit credential-helper $@ get: aws: command not found`

Ursache: Der häufigste Grund für diesen Fehler ist, dass Ihre AWS CLI-Version auf eine Version aktualisiert wurde, die Python 3 verwendet. Dies ist ein bekanntes Problem bei dem MSI-Paket. Um festzustellen, ob Sie über eine der betroffenen Versionen verfügen, öffnen Sie eine Befehlszeile und führen Sie den folgenden Befehl aus: `aws --version`

Wenn die Python-Version in der Ausgabe mit 3 beginnt, ist Ihre Version betroffen. Zum Beispiel:

```
aws-cli/1.16.62 Python/3.6.2 Darwin/16.7.0 botocore/1.12.52
```

Mögliche Lösungen: Sie können dieses Problem mit einer der folgenden Maßnahmen umgehen:

- Installieren und konfigurieren Sie das AWS CLI unter Windows mit Python und Pip anstelle von MSI. Weitere Informationen finden [Sie unter Python, pip und the AWS CLI unter Windows installieren](#).

- Bearbeiten Sie Ihre `.gitconfig`-Datei manuell, um den Abschnitt `[credential]` so zu ändern, dass er explizit auf `aws.cmd` auf Ihrem lokalen Computer verweist. Zum Beispiel:

```
[credential]
  helper = !"C:\\Program Files\\Amazon\\AWSCLI\\bin\\aws.cmd\" codecommit
credential-helper $@
UseHttpPath = true
```

- Führen Sie den Befehl `git config` aus, um Ihre `.gitconfig`-Datei so zu konfigurieren, dass sie explizit auf `aws.cmd` verweist. Aktualisieren Sie dann manuell die `PATH`-Umgebungsvariable nach Bedarf mit dem Pfad zu dem Befehl. Zum Beispiel:

```
git config --global credential.helper "!aws.cmd codecommit credential-helper $@"
git config --global credential.UseHttpPath true
```

Ich werde zur Eingabe eines Benutzernamens aufgefordert, wenn ich eine Verbindung zu einem Repository herstelle CodeCommit

Problem: Wenn Sie versuchen, den Credential Helper für die Kommunikation mit einem CodeCommit Repository zu verwenden, wird eine Meldung angezeigt, in der Sie zur Eingabe Ihres Benutzernamens aufgefordert werden.

Mögliche Lösungen: Konfigurieren Sie Ihr AWS Profil oder stellen Sie sicher, dass das von Ihnen verwendete Profil das ist, mit dem Sie arbeiten möchten. CodeCommit Weitere Informationen zur Einrichtung erhalten Sie unter [Einrichtungsschritte für HTTPS-Verbindungen zu AWS CodeCommit Repositorys unter Linux, macOS oder Unix mit dem AWS CLI Credential Helper](#) oder [Einrichtungsschritte für HTTPS-Verbindungen zu AWS CodeCommit Repositorys unter Windows mit dem AWS CLI Credential Helper](#). Weitere Informationen zu IAM, Zugriffsschlüsseln und geheimen Schlüsseln finden Sie unter [Zugriffsschlüssel für IAM-Benutzer verwalten](#) und [Wie erhalte ich Anmeldeinformationen?](#)

Git für macOS: Ich habe den Credential Helper erfolgreich konfiguriert, aber jetzt wird mir der Zugriff auf mein Repository verweigert (403)

Problem: Unter macOS scheint der Credential Helper nicht wie erwartet auf Ihre Anmeldeinformationen zuzugreifen oder diese zu verwenden. Dies kann durch zwei verschiedene Probleme verursacht werden:

- Der AWS CLI ist für einen AWS-Region anderen konfiguriert als den, in dem das Repository existiert.
- Das Schlüsselbund-Dienstprogramm (Keychain Access) hat Anmeldeinformationen gespeichert, die jetzt nicht mehr gültig sind.

Mögliche Lösungen: Um zu überprüfen, ob das für die richtige Region konfiguriert AWS CLI ist, führen Sie den `aws configure` Befehl aus und überprüfen Sie die angezeigten Informationen. Wenn sich das CodeCommit Repository in einem AWS-Region anderen Verzeichnis befindet als dem, das für die angezeigt wird AWS CLI, müssen Sie den `aws configure` Befehl ausführen und die Werte auf die Werte ändern, die für diese Region geeignet sind. Weitere Informationen finden Sie unter [Schritt 1: Erstkonfiguration für CodeCommit](#).

Die auf OS X und macOS veröffentlichte Standardversion von Git verwendet das Keychain Access-Hilfsprogramm, um generierte Anmeldeinformationen zu speichern. Aus Sicherheitsgründen ist das für den Zugriff auf dein CodeCommit Repository generierte Passwort temporär, sodass die im Schlüsselbund gespeicherten Anmeldeinformationen nach etwa 15 Minuten nicht mehr funktionieren. Wenn du nur mit auf Git zugreifst CodeCommit, versuche Folgendes:

1. Geben Sie am Terminal den Befehl `git config` ein, um die Git-Konfigurationsdatei (`gitconfig`) zu finden, in der das Schlüsselbund-Dienstprogramm definiert ist. Abhängig von Ihrem lokalen System und Ihren Einstellungen haben Sie möglicherweise mehr als eine `gitconfig`-Datei.

```
git config -l --show-origin | grep credential
```

Suchen Sie in der Ausgabe dieses Befehls nach Ergebnissen, die den folgenden ähneln:

```
file://path/to/gitconfig credential.helper=osxkeychain
```

Die am Anfang dieser Zeile aufgelistete Datei ist die Git-Konfigurationsdatei, die Sie bearbeiten müssen.

2. Verwenden Sie zum Bearbeiten der Git-Konfigurationsdatei einen Texteditor, oder führen Sie den folgenden Befehl aus:

```
nano /usr/local/git/etc/gitconfig
```

3. Ändern Sie die Konfiguration mit einer der folgenden Strategien:

- Kommentieren oder löschen Sie den Abschnitt mit den Anmeldeinformationen, der Folgendes enthält `helper = osxkeychain`. Zum Beispiel:

```
# helper = osxkeychain
```

- Aktualisieren Sie `aws credential helper` sowohl den Abschnitt als auch den Abschnitt `osxkeychain` Credential Helper, sodass sie einen Kontext haben. Es wird beispielsweise verwendet, um `osxkeychain` sich zu authentifizieren für: GitHub

```
[credential "https://git-codecommit.us-east-1.amazonaws.com"]
  helper = !aws --profile CodeCommitProfile codecommit credential-helper $@
  UseHttpPath = true
[credential "https://github.com"]
  helper = osxkeychain
```

In dieser Konfiguration verwendet Git den `osxkeychain` Helper, wenn der Remote-Host mit `"https://github.com"` übereinstimmt, und den Credential-Helper, wenn der Remote-Host mit `"https://git-codecommit.us-east-1.amazonaws.com"` übereinstimmt.

- Fügt vor dem Credential-Helper einen leeren String-Helper ein. Um beispielsweise den `osxkeychain` Helper nicht zu verwenden, wenn Sie das *CodeCommitProfile* mit der CLI benannte Profil verwenden:

```
[credential]
  helper =
  helper = !aws --profile CodeCommitProfile codecommit credential-helper $@
  UseHttpPath = true
```

Tip

Sie könnten auch die Zeile, die auf die leere Hilfszeile folgt, so konfigurieren, dass sie nicht übereinstimmt, CodeCommit wenn Sie möchten, dass sie alle Profile ausschließt:

```
helper = !aws codecommit credential-helper $@
```

Wenn Sie andernfalls das Schlüsselbund-Dienstprogramm weiterhin verwenden wollen, um Anmeldeinformationen für weitere Git-Repositorys zu cachern, modifizieren Sie die

Kopfzeile, anstatt die Zeile auszukommentieren. Um beispielsweise zwischengespeicherte Anmeldeinformationen für zuzulassen GitHub, könnten Sie den Header wie folgt ändern:

```
[credential "https://github.com"]  
  helper = osxkeychain
```

Wenn Sie mit Git auf andere Repositories zugreifen, können Sie das Keychain Access-Hilfsprogramm so konfigurieren, dass es keine Anmeldeinformationen für Ihre CodeCommit Repositories bereitstellt. So konfigurieren Sie das Schlüsselbund-Dienstprogramm (Keychain Access):

1. Öffnen Sie das Schlüsselbund-Dienstprogramm. (Sie finden das Schlüsselbund-Dienstprogramm mit dem Finder.)
2. Suchen Sie nach dem Ort, an *us-east-2* AWS-Region dem das Repository existiert, `git-codecommit.us-east-2.amazonaws.com` und ersetzen Sie es durch. Markieren Sie die Zeile, öffnen Sie das Kontextmenü (klicken Sie mit der rechten Maustaste) und wählen Sie dann Get Info.
3. Wählen Sie die Registerkarte Access Control aus.
4. Wählen Sie unter Confirm before allowing access den Eintrag `git-credential-osxkeychain` aus und klicken Sie auf das Minuszeichen, um den Eintrag aus der Liste zu entfernen.

Note

Nachdem Sie `git-credential-osxkeychain` aus der Liste entfernt haben, wird bei jeder Ausführung eines Git-Befehls ein Dialogfeld eingeblendet. Klicken Sie zum Fortfahren auf Deny. Falls Sie diese Dialogfelder umgehen möchten, finden Sie hier einige Alternativen:

- Stellen Sie eine Verbindung her, CodeCommit indem Sie SSH- oder Git-Anmeldeinformationen anstelle des Credential Helper mit HTTPS verwenden. Weitere Informationen erhalten Sie unter [Für SSH-Verbindungen unter Linux, macOS oder Unix](#) und [Einrichtung für HTTPS-Benutzer mit Git-Anmeldeinformationen](#).
- Wählen Sie im Schlüsselbund-Dienstprogramm auf der Registerkarte Access Control für `git-codecommit.us-east-2.amazonaws.com` die Option Allow all applications to access this item (access to this item is not restricted) aus. Auf diese Weise werden keine Popups angezeigt, aber die Anmeldeinformationen laufen ab

(nach durchschnittlich etwa 15 Minuten) und die Fehlermeldung 403 wird angezeigt. In diesem Fall müssen Sie das Schlüsselbundobjekt löschen, um die Funktionalität wiederherzustellen.

- Sie installieren eine Git-Version, in der nicht standardmäßig der Schlüsselbund verwendet wird.
- Ziehen Sie eine Skriptlösung zum Löschen des Schlüsselbundelements in Betracht. Ein von der Community generiertes Beispiel einer Skriptlösung finden Sie unter [Mac OS X Script to Periodically Delete Cached Credentials in the OS X Certificate Store](#) in [Produkt- und Service-Integrationen](#).

Wenn Sie nicht möchten, dass Git das Schlüsselbund-Dienstprogramm verwendet, können Sie Git so konfigurieren, dass "osxkeychain" nicht mehr als Hilfsprogramm für die Anmeldeinformationen verwendet wird. Wenn Sie beispielsweise ein Terminal öffnen und den Befehl `git config --system credential.helper` ausführen und `osxkeychain` zurückgegeben wird, ist Git zur Verwendung des Schlüsselbund-Dienstprogramms eingestellt. Sie können dies ändern, indem Sie den folgenden Befehl ausführen:

```
git config --system --unset credential.helper
```

Beachten Sie, dass die Ausführung dieses Befehls mit der `--system` Option das Git-Verhalten für alle Benutzer systemweit ändert. Dies kann unbeabsichtigte Folgen für andere Benutzer oder für andere Repositories haben, wenn Sie zusätzlich zu anderen Repository-Diensten verwenden. CodeCommit Außerdem ist für diesen Ansatz möglicherweise `sudo` erforderlich und Sie benötigen ausreichende Systemberechtigungen zum Anwenden dieser Änderung. Führen Sie den Befehl `git config --system credential.helper` erneut aus, um sicherzustellen, dass er korrekt ausgeführt wurde. Weitere Informationen finden Sie unter [Customizing Git - Git Configuration](#) und [in diesem Artikel zu Stack Overflow](#).

Git für Windows: Ich habe Git für Windows installiert, aber mir wird der Zugriff auf mein Repository verweigert (403)

Problem: Unter Windows greift das Hilfsprogramm für Anmeldeinformationen nicht wie erwartet auf Ihre Anmeldeinformationen zu bzw. verwendet diese nicht wie erwartet. Dies kann durch verschiedene Probleme verursacht werden:

- Der AWS CLI ist für einen AWS-Region anderen konfiguriert als für den, in dem das Repository existiert.
- Standardmäßig installiert Git für Windows ein Git Credential Manager-Hilfsprogramm, das nicht mit CodeCommit Verbindungen kompatibel ist, die den AWS Credential Helper verwenden. Wenn es installiert ist, führt es dazu, dass Verbindungen zum Repository fehlschlagen, obwohl der Credential Helper mit dem installiert AWS CLI und für Verbindungen zu konfiguriert wurde. CodeCommit
- Einige Versionen von Git für Windows erfüllen möglicherweise nicht vollständig die in [RFC 2617](#) und [RFC 4559](#) festgeschriebenen Bedingungen. Dies kann zu Problemen mit Git-Anmeldeinformationen und dem in die AWS CLI integrierten Hilfsprogramm für Anmeldeinformationen führen. Weitere Informationen finden Sie unter [Version 2.11.0\(3\) does not ask for username/password](#).

Mögliche Lösungen:

- Wenn Sie versuchen, den in der enthaltenen Credential Helper zu verwenden AWS CLI, sollten Sie erwägen, eine Verbindung mit Git-Anmeldeinformationen über HTTPS herzustellen, anstatt den Credential Helper zu verwenden. Die für Ihren IAM-Benutzer konfigurierten Git-Anmeldeinformationen sind mit dem Git Credential Manager für Windows kompatibel, im Gegensatz zum Credential Helper für. AWS CodeCommit Weitere Informationen finden Sie unter [Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden](#).

Wenn Sie den Credential Helper verwenden möchten, um zu überprüfen, ob der richtig konfiguriert AWS CLI ist AWS-Region, führen Sie den `aws configure` Befehl aus und überprüfen Sie die angezeigten Informationen. Wenn sich das CodeCommit Repository in einem AWS-Region anderen Verzeichnis befindet als dem, das für den angezeigt wird AWS CLI, müssen Sie den `aws configure` Befehl ausführen und die Werte auf die Werte ändern, die für diese Region geeignet sind. Weitere Informationen finden Sie unter [Schritt 1: Erstkonfiguration für CodeCommit](#).

- Wenn möglich, deinstallieren Sie Git für Windows und installieren Sie es dann erneut. Achten Sie bei der Installation von Git für Windows darauf, dass das Kontrollkästchen zur Installation des Git-Dienstprogramms zur Verwaltung von Anmeldeinformationen (Git Credential Manager) nicht aktiviert ist. Dieses Programm zur Verwaltung von Anmeldeinformationen ist nicht mit dem AWS CodeCommit-Hilfsprogramm für Anmeldeinformationen kompatibel. Wenn Sie den Git Credential Manager oder ein anderes Dienstprogramm zur Verwaltung von Anmeldeinformationen installiert haben und es nicht deinstallieren möchten, können Sie Ihre `.gitconfig`-Datei verändern und eine spezielle Anmeldeverwaltung für CodeCommit hinzufügen:

1. Öffnen Sie die Systemsteuerung, wählen Sie Credential Manager und entfernen Sie alle gespeicherten Anmeldeinformationen für CodeCommit.
2. Öffnen Sie Ihre `.gitconfig`-Datei in einem Texteditor wie zum Beispiel dem Windows-eigenen Editor.

 Note

Wenn Sie mit mehreren Git-Profilen arbeiten, haben Sie möglicherweise lokale und globale `.gitconfig`-Dateien. Stellen Sie sicher, dass Sie die richtige Datei bearbeiten.

3. Fügen Sie nun der `.gitconfig`-Datei den folgenden Abschnitt hinzu:

```
[credential "https://git-codecommit.*.amazonaws.com"]
  helper = !aws codecommit credential-helper $@
  UseHttpPath = true
```

4. Speichern Sie die Datei und öffnen Sie anschließend eine neue Befehlszeilensitzung, bevor Sie einen erneuten Verbindungsversuch unternehmen.

Sie können diesen Ansatz auch verwenden, wenn Sie den Credential Helper verwenden möchten, wenn Sie eine Verbindung zu Repositorys herstellen, und ein anderes Credential Management System, AWS CodeCommit wenn Sie eine Verbindung zu anderen gehosteten CodeCommit Repositorys, wie z. B. Repositorys, herstellen möchten. GitHub

Um die Einstellung zurückzusetzen, welches Hilfsprogramm für Anmeldeinformationen standardmäßig verwendet wird, können Sie die Option `--system` anstelle von `--global` oder `--local` verwenden, wenn Sie den Befehl `git config` ausführen.

- Wenn Sie Git-Anmeldeinformationen auf einem Windows-Computer verwenden, können Sie versuchen, Probleme aufgrund von mangelnder RFC-Compliance zu umgehen, indem Sie Ihren Git-Benutzernamen als Teil der Verbindungszeichenfolge eingeben. Um das Problem beispielsweise zu umgehen und ein Repository zu klonen, das *MyDemoRepo* in der Region USA Ost (Ohio) benannt ist, gehen Sie wie folgt vor:

```
git clone https://Your-Git-Credential-Username@git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
```

 Note

Dieser Ansatz funktioniert nicht, wenn in dem Benutzernamen Ihrer Git-Anmeldeinformationen ein @-Zeichen vorkommt. Sie müssen für das Zeichen eine URL-Codierung (auch bekannt als URL-Escaping oder [Prozentcodierung](#)) durchführen.

Problembehandlung bei Git-Clients und AWS CodeCommit

Die folgenden Informationen können Ihnen dabei helfen, häufige Probleme bei der Verwendung von Git mit AWS CodeCommit -Repositorys zu beheben. Weitere Informationen zur Fehlerbehebung bei Git-Clients bei der Verwendung von HTTPS oder SSH finden Sie unter [Fehlerbehebung bei Git-Anmeldeinformationen \(HTTPS\)](#), [Problembehandlung bei SSH-Verbindungen](#) und [Problembehandlung beim Credential Helper \(HTTPS\)](#).

Themen

- [Git-Fehler: Fehler: RPC fehlgeschlagen; Ergebnis=56, HTTP-Code = 200 fatal: Das Remote-Ende ist unerwartet hängen geblieben](#)
- [Git-Fehler: Zu viele Referenz-Update-Befehle](#)
- [Git-Fehler: Push-Übertragung über HTTPS funktioniert in einigen Git-Versionen nicht](#)
- [Git-Fehler: "gnutls_handshake\(\) failed"](#)
- [Git-Fehler: Git kann das CodeCommit Repository nicht finden oder hat keine Berechtigung, auf das Repository zuzugreifen](#)
- [Git unter Windows: Keine unterstützten Authentifizierungsmethoden verfügbar \(publickey\)](#)

Git-Fehler: Fehler: RPC fehlgeschlagen; Ergebnis=56, HTTP-Code = 200 fatal: Das Remote-Ende ist unerwartet hängen geblieben

Problem: Wird eine umfassende Änderung, eine hohe Anzahl an Änderungen oder ein großes Repository per Push übertragen, werden HTTPS-Langzeitverbindungen häufig aufgrund von Problemen mit der Netzwerkverbindung oder den Firewall-Einstellungen vorzeitig beendet.

Mögliche Lösungen: Führen Sie die Push-Übertragung alternativ mit SSH durch. Zur Migration eines großen Repositorys können Sie auch die Anweisungen unter [Migrieren Sie ein Repository](#)

[schrittweise](#) befolgen. Stellen Sie darüber hinaus sicher, dass Sie die Größenbeschränkung für einzelne Dateien nicht überschreiten. Weitere Informationen finden Sie unter [Kontingente](#).

Git-Fehler: Zu viele Referenz-Update-Befehle

Problem: Die maximale Anzahl für Referenz-Updates pro Push ist 4 000. Dieser Fehler tritt auf, wenn der Push mehr als 4 000 Referenz-Updates enthält.

Mögliche Lösungen: Versuchen Sie, die Push-Übertragung für Branches und Tags mit `git push --all` und `git push --tags` separat durchzuführen. Wenn Sie zu viele Tags haben, teilen Sie die Tags auf mehrere Push-Übertragungen auf. Weitere Informationen finden Sie unter [Kontingente](#).

Git-Fehler: Push-Übertragung über HTTPS funktioniert in einigen Git-Versionen nicht

Problem: Ein Problem mit dem Curl-Update auf Version 7.41.0 verursacht Fehlschläge bei der SSPI-basierten Digest-Authentifizierung. Betroffen ist unter anderem die Git-Version 1.9.5.msysgit.1. Einige Versionen von Git für Windows erfüllen möglicherweise nicht vollständig die in [RFC 2617](#) und [RFC 4559](#) festgeschriebenen Bedingungen. Dies kann zu Problemen mit HTTPS-Verbindungen führen, wenn Git-Anmeldeinformationen und das in der AWS CLI integrierte Hilfsprogramm für Anmeldeinformationen verwendet werden.

Mögliche Lösungen: Überprüfen Sie Ihre Git-Version auf bekannte Probleme oder verwenden Sie eine ältere oder neuere Version. Weitere Informationen zu msysgit finden Sie in den Foren unter [Push to HTTPS Is Broken](#). GitHub Weitere Informationen zu Problemen mit Git-Versionen für Windows finden Sie unter [Version 2.11.0\(3\) fragt nicht nach Benutzernamen/Passwort](#).

Git-Fehler: "gnutls_handshake() failed"

Problem: Wenn Sie unter Linux versuchen, Git für die Kommunikation mit einem CodeCommit Repository zu verwenden, erscheint eine Fehlermeldung, die den Ausdruck enthält `error: gnutls_handshake() failed`.

Mögliche Lösungen: Kompilieren Sie Git für OpenSSL. Einen Ansatz finden Sie unter ["Error: gnutls_handshake\(\) failed" When Connecting to HTTPS Servers](#) in den Ask Ubuntu-Foren.

Verwenden Sie alternativ SSH anstelle von HTTPS, um mit CodeCommit Repositorys zu kommunizieren.

Git-Fehler: Git kann das CodeCommit Repository nicht finden oder hat keine Berechtigung, auf das Repository zuzugreifen

Problem: Ein nachgestellter Schrägstrich in der Verbindungszeichenfolge kann dazu führen, dass Verbindungsversuche fehlschlagen.

Mögliche Lösungen: Stellen Sie sicher, dass Sie den richtigen Namen und die richtige Verbindungszeichenfolge für das Repository ohne nachgestellte Schrägstriche angegeben haben. Weitere Informationen finden Sie unter [Herstellen einer Verbindung mit einem Repository](#).

Git unter Windows: Keine unterstützten Authentifizierungsmethoden verfügbar (publickey)

Problem: Nachdem Sie SSH-Zugriff für Windows konfiguriert haben, wird eine Fehlermeldung des Typs "Zugriff verweigert" angezeigt, wenn Sie versuchen, Befehle wie `git pull` oder `git clone` zu verwenden.

Mögliche Lösungen: Die häufigste Ursache für diesen Fehler ist, dass eine `GIT_SSH`-Umgebungsvariable auf Ihrem Computer vorhanden ist, die zur Unterstützung eines anderen Verbindungsdienstprogramms wie zum Beispiel PuTTY konfiguriert ist. Um dieses Problem zu beheben, versuchen Sie es mit einer der folgenden Methoden:

- Öffnen Sie einen Bash-Emulator und fügen Sie den Parameter `GIT_SSH_COMMAND="ssh"` vor dem Git-Befehl hinzu. Wenn Sie zum Beispiel versuchen, ein Repository zu klonen, führen Sie anstelle von `git clone ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo` den folgenden Befehl aus:

```
GIT_SSH_COMMAND="ssh" git clone ssh://git-codecommit.us-east-2.amazonaws.com/v1/
repos/MyDemoRepo my-demo-repo
```

- Löschen Sie die Umgebungsvariable `GIT_SSH` oder benennen Sie sie um, wenn Sie sie nicht mehr verwenden. Öffnen Sie anschließend ein neues Befehlszeilenfenster oder eine neue Bash-Emulator-Sitzung und versuchen Sie erneut, den Befehl einzugeben.

Weitere Informationen zur Behebung von Git-Problemen unter Windows bei Verwendung von SSH finden Sie unter [Problembehandlung bei SSH-Verbindungen](#).

Behebung von Zugriffsfehlern und AWS CodeCommit

Die folgenden Informationen können Ihnen helfen, Zugriffsfehler zu beheben, die bei der Verbindung mit AWS CodeCommit -Repositorys auftreten können.

Themen

- [Zugriffsfehler: Ich werde zur Eingabe eines Benutzernamens und eines Kennworts aufgefordert, wenn ich von Windows aus eine Verbindung zu einem CodeCommit Repository herstelle](#)
- [Zugriffsfehler: Verweigerung des öffentlichen Schlüssels bei der Verbindung mit einem CodeCommit-Repository](#)
- [Zugriffsfehler: Meldung „Rate Exceeded“ oder „429“ beim Herstellen einer Verbindung zu einem Repository CodeCommit](#)

Zugriffsfehler: Ich werde zur Eingabe eines Benutzernamens und eines Kennworts aufgefordert, wenn ich von Windows aus eine Verbindung zu einem CodeCommit Repository herstelle

Problem: Wenn Sie versuchen, Git für die Kommunikation mit einem CodeCommit Repository zu verwenden, wird ein Dialogfeld angezeigt, in dem Sie nach Ihrem Benutzernamen und Passwort gefragt werden.

Mögliche Lösungen: Die Ursache ist möglicherweise das Windows-eigene Verwaltungssystem für Anmeldeinformationen. Abhängig von Ihrer Konfiguration führen Sie eine der folgenden Maßnahmen durch:

- Wenn Sie HTTPS mit Git-Anmeldeinformationen verwenden, sind Ihre Git-Anmeldeinformationen noch nicht im System gespeichert. Stellen Sie die Git-Anmeldeinformationen bereit und fahren Sie fort. Sie sollten nicht mehr zur Eingabe aufgefordert werden. Weitere Informationen finden Sie unter [Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden](#).
- Wenn Sie HTTPS mit dem Credential Helper für verwenden AWS CodeCommit, ist es nicht mit dem Windows-Anmeldeinformationsverwaltungssystem kompatibel. Klicken Sie auf Abbrechen.

Dieses Problem kann auch bedeuten, dass Sie das Git-Dienstprogramm zur Verwaltung von Anmeldeinformationen (Git Credential Manager) bei der Installation von Git für Windows installiert haben. Der Git Credential Manager ist nicht kompatibel mit dem Credential Helper für, der in der CodeCommit enthalten ist. AWS CLI Erwägen Sie, den Git-Anmeldeinformations-Manager zu

deinstallieren. Als Alternative zur Verwendung des Hilfsprogramms für Anmeldeinformationen für CodeCommit können Sie auch `git-remote-codecommit` installieren und konfigurieren.

Weitere Informationen finden Sie unter [Einrichtungsschritte für HTTPS-Verbindungen AWS CodeCommit mit git-remote-codecommit](#), [Für HTTPS-Verbindungen unter Windows mit dem Credential Helper AWS CLI](#) und [Git für Windows: Ich habe Git für Windows installiert, aber mir wird der Zugriff auf mein Repository verweigert \(403\)](#).

Zugriffsfehler: Verweigerung des öffentlichen Schlüssels bei der Verbindung mit einem CodeCommit-Repository

Problem: Wenn Sie versuchen, einen SSH-Endpunkt für die Kommunikation mit einem CodeCommit Repository zu verwenden, wird eine Fehlermeldung angezeigt, die den Ausdruck enthält. `Error: public key denied`

Mögliche Fehlerbehebungen: Der häufigste Grund für diesen Fehler ist, dass Sie die Einrichtung für die SSH-Verbindungen nicht abgeschlossen haben. Konfigurieren Sie ein öffentliches und ein privates SSH-Schlüsselpaar und ordnen Sie den öffentlichen Schlüssel dann Ihrem IAM-Benutzer zu. Weitere Informationen zur Konfiguration von SSH finden Sie unter [Für SSH-Verbindungen unter Linux, macOS oder Unix](#) und [Für SSH-Verbindungen unter Windows](#).

Zugriffsfehler: Meldung „Rate Exceeded“ oder „429“ beim Herstellen einer Verbindung zu einem Repository CodeCommit

Problem: Bei dem Versuch, mit einem CodeCommit-Repository zu kommunizieren, erscheint eine Meldung mit "Quote überschritten" oder dem Fehlercode "429". Die Kommunikation wird entweder erheblich verlangsamt oder schlägt fehl.

Ursache: Alle Aufrufe von CodeCommit, ob von einer Anwendung, dem AWS CLI, einem Git-Client oder der AWS Management Console, unterliegen einer maximalen Anzahl von Anfragen pro Sekunde und insgesamt aktiven Anfragen. Sie dürfen die maximal zulässige Anforderungsrate für ein Amazon Web Services Services-Konto in keinem Fall überschreiten AWS-Region. Wenn Anfragen die maximale Rate überschreiten, erhalten Sie eine Fehlermeldung und weitere Anrufe werden für Ihr Amazon Web Services Services-Konto vorübergehend gedrosselt. Während des Drosselungszeitraums werden Ihre Verbindungen mit CodeCommit verlangsamt und schlagen möglicherweise fehl.

Mögliche Lösungen: Ergreifen Sie Maßnahmen, um die Anzahl der Verbindungen oder Anrufe zu reduzieren CodeCommit oder Anfragen zu verteilen. Beachten Sie folgende Ansätze:

- Implementieren von Jitter in Anforderungen, insbesondere in regelmäßige Abfrageanforderungen

Wenn Sie eine Anwendung haben, die CodeCommit regelmäßig Abfragen durchführt und diese Anwendung auf mehreren EC2 Amazon-Instances läuft, führen Sie Jitter ein (eine zufällige Verzögerung), sodass verschiedene EC2 Amazon-Instances nicht gleichzeitig Abfragen durchführen. Wir empfehlen eine Zufallszahl von 0 bis 59 Sekunden, um Abfragemechanismen gleichmäßig über einen einminütigen Zeitraum zu verteilen.

- Verwenden einer ereignisbasierten Architektur anstelle von Abfragen

Verwenden Sie anstelle von Abfragen eine ereignisbasierte Architektur, sodass Aufrufe nur gemacht werden, wenn ein Ereignis eintritt. Erwägen Sie die Verwendung von CloudWatch Ereignisbenachrichtigungen für [AWS CodeCommit Ereignisse](#), um Ihren Workflow auszulösen.

- Implementieren Sie Fehlerwiederholungen und exponentielle Backoffs für APIs und automatisierte Git-Aktionen

Durch Wiederholversuche bei Fehlern und exponentielle Backoffs lässt sich die Rate der Aufrufe beschränken. Jedes AWS -SDK implementiert Logik für automatische Wiederholungsversuche und Algorithmen für exponentielles Backoff. Für automatisierte Git-Push- und Git-Pull-Aktionen müssen Sie möglicherweise Ihre eigene Logik für Wiederholversuche implementieren. Weitere Informationen finden Sie unter [Wiederholungen bei Fehlern](#) und exponentielles Backoff in. AWS

- Beantragen Sie im AWS Support Center eine Erhöhung des CodeCommit Servicekontingents

Um eine Erhöhung des Service Limits zu erhalten, müssen Sie bestätigen, dass Sie die hier gemachten Vorschläge bereits befolgt haben, einschließlich der Methoden für Wiederholversuche bei Fehlern und exponentielles Backoff. In Ihrer Anfrage müssen Sie auch das Amazon Web Services Services-Konto und den AWS-Region Zeitrahmen angeben, der von den Drosselungsproblemen betroffen ist.

Behebung von Konfigurationsfehlern und AWS CodeCommit

Die folgenden Informationen können Ihnen bei der Behebung von Konfigurationsfehlern helfen, die beim Herstellen einer Verbindung mit AWS CodeCommit Repositorys auftreten können.

Themen

- [Konfigurationsfehler: AWS CLI Anmeldeinformationen können auf macOS nicht konfiguriert werden](#)

Konfigurationsfehler: AWS CLI Anmeldeinformationen können auf macOS nicht konfiguriert werden

Problem: Wenn Sie ausführen, `aws configure` um das zu konfigurieren AWS CLI, wird eine `ConfigParseError` Meldung angezeigt.

Mögliche Lösungen: Die häufigste Ursache für diesen Fehler ist, dass bereits eine Datei mit Anmeldeinformationen existiert. Navigieren Sie zu `~/aws` und suchen Sie eine Datei mit dem Namen `credentials`. Löschen Sie diese Datei oder benennen Sie sie um und führen Sie anschließend `aws configure` erneut aus.

Behebung von Konsolenfehlern und AWS CodeCommit

Die folgenden Informationen können Ihnen helfen, Konsolenfehler zu beheben, die bei der Verbindung mit AWS CodeCommit -Repositorys auftreten können.

Themen

- [Zugriffsfehler: Zugriff auf den Verschlüsselungsschlüssel für ein CodeCommit Repository über die Konsole verweigert oder AWS CLI](#)
- [Verschlüsselungsfehler: Das Repository kann nicht entschlüsselt werden](#)
- [Konsolenfehler: Der Code in einem CodeCommit Repository kann nicht von der Konsole aus durchsucht werden](#)
- [Anzeigefehler: Eine Datei oder ein Vergleich zwischen Dateien kann nicht angezeigt werden](#)

Zugriffsfehler: Zugriff auf den Verschlüsselungsschlüssel für ein CodeCommit Repository über die Konsole verweigert oder AWS CLI

Problem: Wenn Sie versuchen, CodeCommit von der Konsole oder der aus darauf zuzugreifen AWS CLI, wird eine Fehlermeldung angezeigt, die den Ausdruck `EncryptionKeyAccessDeniedException` oder enthält `User is not authorized for the KMS default key for CodeCommit 'aws/codecommit' in your account.`

Mögliche Lösungen: Die häufigste Ursache für diesen Fehler ist, dass Ihr Amazon Web Services Services-Konto nicht abonniert ist AWS Key Management Service, was für CodeCommit erforderlich

ist. Öffnen Sie die AWS KMS Konsole, wählen Sie AWS Managed Keys und dann Get Started Now. Wenn Sie eine Meldung sehen, dass Sie den AWS Key Management Service Dienst derzeit nicht abonniert haben, folgen Sie den Anweisungen auf dieser Seite, um ihn zu abonnieren. Weitere Informationen zu CodeCommit und finden Sie AWS Key Management Service unter [AWS KMS und Verschlüsselung](#).

Verschlüsselungsfehler: Das Repository kann nicht entschlüsselt werden

Problem: Wenn Sie versuchen, über die Konsole oder die auf ein CodeCommit Repository zuzugreifen AWS CLI, erscheint eine Fehlermeldung, die den Ausdruck `Repository can't be decrypted` enthält.

Mögliche Lösungen: Die häufigste Ursache für diesen Fehler ist, dass der AWS KMS Schlüssel, der zum Verschlüsseln und Entschlüsseln von Daten für dieses Repository verwendet wird, nicht aktiv ist oder noch gelöscht werden muss. Für ist ein aktiver Von AWS verwalteter Schlüssel oder vom Kunden verwalteter Schlüssel AWS Key Management Service erforderlich. CodeCommit Öffnen Sie die AWS KMS Konsole, wählen Sie Von AWS verwaltete Schlüsseler oder Vom Kunden verwaltete Schlüssel und stellen Sie sicher, dass der für das Repository verwendete Schlüssel dort vorhanden ist, AWS-Region wo das Repository existiert, und dass sein Status Aktiv ist. Weitere Informationen zu CodeCommit und finden AWS Key Management Service Sie unter [AWS KMS und Verschlüsselung](#).

Important

Wenn der Schlüssel, der zum Verschlüsseln und Entschlüsseln der Daten für das Repository verwendet wurde, dauerhaft gelöscht wurde oder auf andere Weise nicht zugänglich ist, kann auf Daten in den Repositories, die mit diesem Schlüssel verschlüsselt wurden, nicht zugegriffen werden.

Konsolenfehler: Der Code in einem CodeCommit Repository kann nicht von der Konsole aus durchsucht werden

Problem: Wenn Sie versuchen, den Inhalt eines Repositories von der Konsole aus anzuzeigen, wird eine Fehlermeldung angezeigt und der Zugriff wird verweigert.

Mögliche Lösungen: Die häufigste Ursache für diesen Fehler ist, dass eine auf Ihr Amazon Web Services Services-Konto angewendete IAM-Richtlinie eine oder mehrere der Berechtigungen verweigert, die für das Durchsuchen von Code von der CodeCommit Konsole aus erforderlich sind.

Weitere Informationen zu CodeCommit Zugriffsberechtigungen und zum Surfen finden Sie unter [Authentifizierung und Zugriffskontrolle für AWS CodeCommit](#)

Anzeigefehler: Eine Datei oder ein Vergleich zwischen Dateien kann nicht angezeigt werden

Problem: Wenn Sie versuchen, eine Datei oder einen Vergleich zwischen zwei Versionen einer Datei in der CodeCommit Konsole anzuzeigen, wird ein Fehler angezeigt, der besagt, dass die Datei oder der Unterschied zu groß ist, um angezeigt zu werden.

Mögliche Lösungen: Die häufigste Ursache für diesen Fehler ist entweder, dass die Datei zu groß für die Anzeige ist, eine oder mehrere Zeilen enthält, die die Zeichenbeschränkung für eine einzelne Zeile in einer Datei überschreiten, oder dass der Unterschied zwischen den beiden Versionen der Datei die Zeilenbegrenzung überschreitet. Weitere Informationen finden Sie unter [Kontingente](#). Um die Datei oder die Unterschiede zwischen den Versionen der Datei anzuzeigen, können Sie die Datei lokal in Ihrer bevorzugten IDE öffnen, ein Git-Diff-Tool verwenden oder den `git diff` Befehl ausführen.

Fehlerbehebung bei Auslösern und AWS CodeCommit

Die folgenden Informationen können Ihnen helfen, Probleme mit Auslösern in AWS CodeCommit zu beheben.

Themen

- [Auslöserfehler: Ein Repository-Auslöser wird nicht zum erwarteten Zeitpunkt ausgeführt](#)

Auslöserfehler: Ein Repository-Auslöser wird nicht zum erwarteten Zeitpunkt ausgeführt

Problem: Ein oder mehrere der für ein Repository konfigurierten Auslöser werden offenbar gar nicht oder nicht wie erwartet ausgeführt.

Mögliche Lösungen: Wenn das Ziel des Auslösers eine AWS Lambda -Funktion ist, stellen Sie sicher, dass Sie die Ressourcenrichtlinie der Funktion für den Zugriff durch CodeCommit konfiguriert haben. Weitere Informationen finden Sie unter [Beispiel 3: Erstellen Sie eine Richtlinie für die AWS Lambda Integration mit einem Trigger CodeCommit](#).

Alternativ dazu können Sie den Auslöser bearbeiten. Stellen Sie sicher, dass die Ereignisse, für die Sie Aktionen auslösen möchten, ausgewählt wurden und dass die Branches für den Auslöser den

Branch umfassen, in dem Sie Reaktionen auf Aktionen sehen möchten. Ändern Sie versuchsweise die Einstellungen für den Auslöser in All repository events und All branches und testen Sie dann den Auslöser. Weitere Informationen finden Sie unter [Trigger für ein Repository bearbeiten](#).

Schalten Sie das Debuggen ein

Problem: Ich möchte Debugging aktivieren, um mehr Informationen zu meinem Repository zu erhalten und dazu, wie Git Befehle ausführt.

Mögliche Lösungen: Versuchen Sie die folgenden Maßnahmen:

1. Führen Sie in einem Terminal oder einer Eingabeaufforderung auf Ihrem lokalen Computer vor der Ausführung von Git-Befehlen die folgenden Befehle aus:

Unter Linux, macOS oder Unix:

```
export GIT_TRACE_PACKET=1
export GIT_TRACE=1
export GIT_CURL_VERBOSE=1
```

Unter Windows:

```
set GIT_TRACE_PACKET=1
set GIT_TRACE=1
set GIT_CURL_VERBOSE=1
```

Note

Die Einstellung von GIT_CURL_VERBOSE ist nur für HTTPS-Verbindungen hilfreich. SSH verwendet die `libcurl`-Bibliothek nicht.

2. Um mehr Informationen über dein Git-Repository zu erhalten, empfehlen wir, die neueste Version von [git-sizer](#) zu installieren. Folgen Sie den Anweisungen zur Installation des für Ihr Betriebssystem und Ihre Umgebung geeigneten Dienstprogramms. Wechseln Sie nach der Installation in der Befehlszeile oder im Terminal zu den Verzeichnissen Ihres lokalen Repositorys und führen Sie dann den folgenden Befehl aus:

```
git-sizer --verbose
```

 Tip

Erwägen Sie, die Ausgabe des Befehls in einer Datei zu speichern, sodass Sie sie bei der Behebung von Problemen, insbesondere im Laufe der Zeit, problemlos mit anderen teilen können.

AWS CodeCommit Referenz

Die folgenden Referenzthemen können Ihnen helfen CodeCommit, Git AWS-Regionen, Service Limits und mehr besser zu verstehen.

Themen

- [Regionen und Git-Verbindungsendpunkte für AWS CodeCommit](#)
- [Verwendung AWS CodeCommit mit VPC-Endpunkten mit Schnittstelle](#)
- [Kontingente in AWS CodeCommit](#)
- [AWS CodeCommit Befehlszeilenreferenz](#)
- [Grundlegende Git-Befehle](#)

Regionen und Git-Verbindungsendpunkte für AWS CodeCommit

Jedes CodeCommit Repository ist mit einem verknüpft. AWS-Region CodeCommit bietet regionale Endpunkte, über die Sie Ihre Anfragen an den Dienst stellen können. Darüber hinaus CodeCommit bietet es in jeder Region, in der es verfügbar CodeCommit ist, Git-Verbindungsendpunkte für SSH- und HTTPS-Protokolle.

Alle Beispiele in diesem Handbuch verwenden dieselbe Endpunkt-URL für Git in US East (Ohio):`git-codecommit.us-east-2.amazonaws.com`. Wenn Sie Git verwenden und Ihre Verbindungen konfigurieren, stellen Sie jedoch sicher, dass Sie den Git-Verbindungsendpunkt wählen, der dem entspricht AWS-Region , der Ihr CodeCommit Repository hostet. Wenn du beispielsweise eine Verbindung zu einem Repository im Osten der USA (Nord-Virginia) herstellen möchtest, verwende die Endpunkt-URL `git-codecommit.us-east-1.amazonaws.com`. Dies gilt auch für API-Aufrufe. Wenn Sie Verbindungen zu einem CodeCommit Repository mit AWS CLI oder dem herstellen SDKs, stellen Sie sicher, dass Sie den richtigen regionalen Endpunkt für das Repository verwenden.

Themen

- [Unterstützt AWS-Regionen für CodeCommit](#)
- [Endpunkte für Git-Verbindungen](#)
- [Server-Fingerabdrücke für CodeCommit](#)

Unterstützt AWS-Regionen für CodeCommit

Sie können CodeCommit Repositorys wie folgt AWS-Regionen erstellen und verwenden:

- US East (Ohio)
- USA Ost (Nord-Virginia)
- USA West (Nordkalifornien)
- US West (Oregon)
- Europa (Irland)
- Europa (London)
- Europa (Paris)
- Europa (Frankfurt)
- Europa (Stockholm)
- Europa (Milan)
- Afrika (Kapstadt)
- Israel (Tel Aviv)
- Asien-Pazifik (Tokio)
- Asien-Pazifik (Singapur)
- Asien-Pazifik (Sydney)
- Asien-Pazifik (Jakarta)
- Naher Osten (VAE)
- Asien-Pazifik (Seoul)
- Asien-Pazifik (Osaka)
- Asien-Pazifik (Mumbai)
- Asien-Pazifik (Hyderabad)
- Asien-Pazifik (Hongkong)
- Südamerika (São Paulo)
- Naher Osten (Bahrain)
- Kanada (Zentral)
- China (Peking)
- China (Ningxia)
- AWS GovCloud (US-West)

- AWS GovCloud (US-Ost)

CodeCommit hat in einigen Regionen Unterstützung für den Regierungsstandard Federal Information Processing Standard (FIPS) Publication 140-2 hinzugefügt. Weitere Informationen über FIPS und FIPS-Endpunkte finden Sie unter [Federal Information Processing Standard \(FIPS\) 140-2 – Übersicht](#). Informationen zu Git-Verbindungsendpunkten, die FIPS unterstützen, finden Sie unter [Endpunkte für Git-Verbindungen](#).

Weitere Informationen zu regionalen Endpunkten AWS CLI, Service- und API-Aufrufen finden Sie unter [AWS CodeCommit Endpunkte und CodeCommit Kontingente](#).

Endpunkte für Git-Verbindungen

Verwenden Sie Folgendes URLs , wenn Sie Git-Verbindungen zu CodeCommit Repositorys konfigurieren:

Git-Verbindungsendpunkte für AWS CodeCommit

Name der Region	Region	Endpunkt-URL	Protokoll
USA Ost (Ohio)	us-east-2	https://git-codecommit.us-east-2.amazonaws.com	HTTPS
USA Ost (Ohio)	us-east-2	ssh://git-codecommit.us-east-2.amazonaws.com	SSH
USA Ost (Ohio)	us-east-2	https://git-codecommit-fips.us-east-2.amazonaws.com	HTTPS
USA Ost (Nord-Virginia)	us-east-1	https://git-codecommit.us-east-1.amazonaws.com	HTTPS
USA Ost (Nord-Virginia)	us-east-1	ssh://git-codecommit.us-east-1.amazonaws.com	SSH

Name der Region	Region	Endpunkt-URL	Protokoll
USA Ost (Nord-Virginia)	us-east-1	https://git-codecommit-fips.us-east-1.amazonaws.com	HTTPS
USA West (Oregon)	us-west-2	https://git-codecommit.us-west-2.amazonaws.com	HTTPS
USA West (Oregon)	us-west-2	ssh://git-codecommit.us-west-2.amazonaws.com	SSH
USA West (Oregon)	us-west-2	https://git-codecommit-fips.us-west-2.amazonaws.com	HTTPS
USA West (Nordkalifornien)	us-west-1	https://git-codecommit.us-west-1.amazonaws.com	HTTPS
USA West (Nordkalifornien)	us-west-1	ssh://git-codecommit.us-west-1.amazonaws.com	SSH
USA West (Nordkalifornien)	us-west-1	https://git-codecommit-fips.us-west-1.amazonaws.com	HTTPS
Europa (Irland)	eu-west-1	https://git-codecommit.eu-west-1.amazonaws.com	HTTPS
Europa (Irland)	eu-west-1	ssh://git-codecommit.eu-west-1.amazonaws.com	SSH

Name der Region	Region	Endpunkt-URL	Protokoll
Asien-Pazifik (Tokio)	ap-northeast-1	https://git-codecommit.ap-northeast-1.amazonaws.com	HTTPS
Asien-Pazifik (Tokio)	ap-northeast-1	ssh://git-codecommit.ap-northeast-1.amazonaws.com	SSH
Asien-Pazifik (Singapur)	ap-southeast-1	https://git-codecommit.ap-southeast-1.amazonaws.com	HTTPS
Asien-Pazifik (Singapur)	ap-southeast-1	ssh://git-codecommit.ap-southeast-1.amazonaws.com	SSH
Asien-Pazifik (Sydney)	ap-southeast-2	https://git-codecommit.ap-southeast-2.amazonaws.com	HTTPS
Asien-Pazifik (Sydney)	ap-southeast-2	ssh://git-codecommit.ap-southeast-2.amazonaws.com	SSH
Asien-Pazifik (Jakarta)	ap-southeast-3	https://git-codecommit.ap-southeast-3.amazonaws.com	HTTPS
Asien-Pazifik (Jakarta)	ap-southeast-3	ssh://git-codecommit.ap-southeast-3.amazonaws.com	SSH
Naher Osten (VAE)	me-central-1	https://git-codecommit.me-central-1.amazonaws.com	HTTPS

Name der Region	Region	Endpunkt-URL	Protokoll
Naher Osten (VAE)	me-central-1	ssh://git-codecommit.me-central-1.amazonaws.com	SSH
Europa (Frankfurt)	eu-central-1	https://git-codecommit.eu-central-1.amazonaws.com	HTTPS
Europa (Frankfurt)	eu-central-1	ssh://git-codecommit.eu-central-1.amazonaws.com	SSH
Asien-Pazifik (Seoul)	ap-northeast-2	https://git-codecommit.ap-northeast-2.amazonaws.com	HTTPS
Asien-Pazifik (Seoul)	ap-northeast-2	ssh://git-codecommit.ap-northeast-2.amazonaws.com	SSH
Südamerika (São Paulo)	sa-east-1	https://git-codecommit.sa-east-1.amazonaws.com	HTTPS
Südamerika (São Paulo)	sa-east-1	ssh://git-codecommit.sa-east-1.amazonaws.com	SSH
Europa (London)	eu-west-2	https://git-codecommit.eu-west-2.amazonaws.com	HTTPS
Europa (London)	eu-west-2	ssh://git-codecommit.eu-west-2.amazonaws.com	SSH

Name der Region	Region	Endpunkt-URL	Protokoll
Asien-Pazifik (Mumbai)	ap-south-1	https://git-codecommit.ap-south-1.amazonaws.com	HTTPS
Asien-Pazifik (Mumbai)	ap-south-1	ssh://git-codecommit.ap-south-1.amazonaws.com	SSH
Asien-Pazifik (Hyderabad)	ap-south-2	https://git-codecommit.ap-south-2.amazonaws.com	HTTPS
Asien-Pazifik (Hyderabad)	ap-south-2	ssh://git-codecommit.ap-south-2.amazonaws.com	SSH
Kanada (Zentral)	ca-central-1	https://git-codecommit.ca-central-1.amazonaws.com	HTTPS
Kanada (Zentral)	ca-central-1	ssh://git-codecommit.ca-central-1.amazonaws.com	SSH
Kanada (Zentral)	ca-central-1	https://git-codecommit-fips.ca-central-1.amazonaws.com	HTTPS
Europa (Paris)	eu-west-3	https://git-codecommit.eu-west-3.amazonaws.com	HTTPS
Europa (Paris)	eu-west-3	ssh://git-codecommit.eu-west-3.amazonaws.com	SSH

Name der Region	Region	Endpunkt-URL	Protokoll
AWS GovCloud (US-West)	us-gov-west-1	https://git-codecommit.us-gov-west-1.amazonaws.com	HTTPS
AWS GovCloud (US-West)	us-gov-west-1	ssh://git-codecommit.us-gov-west-1.amazonaws.com	SSH
AWS GovCloud (US-West)	us-gov-west-1	https://git-codecommit-fips.us-gov-west-1.amazonaws.com	HTTPS
AWS GovCloud (US-Ost)	us-gov-east-1	https://git-codecommit.us-gov-east-1.amazonaws.com	HTTPS
AWS GovCloud (US-Ost)	us-gov-east-1	ssh://git-codecommit.us-gov-east-1.amazonaws.com	SSH
AWS GovCloud (US-Ost)	us-gov-east-1	https://git-codecommit-fips.us-gov-east-1.amazonaws.com	HTTPS
Europa (Stockholm)	eu-north-1	https://git-codecommit.eu-north-1.amazonaws.com	HTTPS
Europa (Stockholm)	eu-north-1	ssh://git-codecommit.eu-north-1.amazonaws.com	SSH
Naher Osten (Bahrain)	me-south-1	https://git-codecommit.me-south-1.amazonaws.com	HTTPS

Name der Region	Region	Endpunkt-URL	Protokoll
Naher Osten (Bahrain)	me-south-1	ssh://git-codecommit.me-south-1.amazonaws.com	SSH
Asien-Pazifik (Hongkong)	ap-east-1	https://git-codecommit.ap-east-1.amazonaws.com	HTTPS
Asien-Pazifik (Hongkong)	ap-east-1	ssh://git-codecommit.ap-east-1.amazonaws.com	SSH
China (Beijing)	cn-north-1	https://git-codecommit.cn-north-1.amazonaws.com.cn	HTTPS
China (Beijing)	cn-north-1	ssh://git-codecommit.cn-north-1.amazonaws.com.cn	SSH
China (Ningxia)	cn-northwest-1	https://git-codecommit.cn-northwest-1.amazonaws.com.cn	HTTPS
China (Ningxia)	cn-northwest-1	ssh://git-codecommit.cn-northwest-1.amazonaws.com.cn	SSH
Europa (Mailand)	eu-south-1	https://git-codecommit.eu-south-1.amazonaws.com	HTTPS
Europa (Mailand)	eu-south-1	ssh://git-codecommit.eu-south-1.amazonaws.com	SSH

Name der Region	Region	Endpunkt-URL	Protokoll
Asia Pacific (Osaka)	ap-northeast-3	https://git-codecommit.ap-northeast-3.amazonaws.com	HTTPS
Asia Pacific (Osaka)	ap-northeast-3	ssh://git-codecommit.ap-northeast-3.amazonaws.com	SSH
Afrika (Kapstadt)	af-south-1	https://git-codecommit.af-south-1.amazonaws.com	HTTPS
Afrika (Kapstadt)	af-south-1	ssh://git-codecommit.af-south-1.amazonaws.com	SSH
Israel (Tel Aviv)	il-central-1	https://git-codecommit.il-central-1.amazonaws.com	HTTPS
Israel (Tel Aviv)	il-central-1	ssh://git-codecommit.il-central-1.amazonaws.com	SSH

Server-Fingerabdrücke für CodeCommit

Die folgende Tabelle listet die öffentlichen Fingerprints für Git-Verbindungsendpunkte in CodeCommit auf. Diese Server-Fingerprints werden als Teil des Verifizierungsvorgangs beim Hinzufügen eines Endpunkts zu Ihrer bekannten Hosts-Datei angezeigt.

Öffentliche Fingerabdrücke für CodeCommit

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.us-east-2.amazonaws.com	MD5	a9:6d:03:ed:08:42: 21:be:06:e1:e0:2a: d1:75:31:5e
git-codecommit.us-east-2.amazonaws.com	SHA256	3lB1W2g5xn/NA2Ck6d yeJIrQ0Wvn7n8UEs56 fG6ZIZQ
git-codecommit.us-east-1.amazonaws.com	MD5	a6:9c:7d:bc:35:f5: d4:5f:8b:ba:6f:c8: bc:d4:83:84
git-codecommit.us-east-1.amazonaws.com	SHA256	eLMY1j0DKA4uvDZc1/ KgtIayZANwX6t8+8is PtotBoY
git-codecommit.us-west-2.amazonaws.com	MD5	a8:68:53:e3:99:ac: 6e:d7:04:7e:f7:92: 95:77:a9:77
git-codecommit.us-west-2.amazonaws.com	SHA256	0pJx9SQpkbPUAHwy58 UVIq0IHcyo1fwCp00u VgcAWPo
git-codecommit.eu-west-1.amazonaws.com	MD5	93:42:36:ea:22:1f: f1:0f:20:02:4a:79: ff:ea:12:1d
git-codecommit.eu-west-1.amazonaws.com	SHA256	tKjRk0L8dmJyTmSbeS dN1S8F/f0iq13Rlvqg TOP1UyQ
git-codecommit.ap-northeast-1.amazonaws.com	MD5	8e:a3:f0:80:98:48: 1c:5c:6f:59:db:a7: 8f:6e:c6:cb

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.ap-northeast-1.amazonaws.com	SHA256	Xk/WeYD/K/bnBybzhi uu4dWpBJtXPf7E30jH U7se40w
git-codecommit.ap-southeast-1.amazonaws.com	MD5	65:e5:27:c3:09:68: 0d:8e:b7:6d:94:25: 80:3e:93:cf
git-codecommit.ap-southeast-1.amazonaws.com	SHA256	ZIsVa70VzxrtIf+Rk4 UbhPv6Es22mSB3uTBo jfPXIno
git-codecommit.ap-southeast-2.amazonaws.com	MD5	7b:d2:c1:24:e6:91: a5:7b:fa:c1:0c:35: 95:87:da:a0
git-codecommit.ap-southeast-2.amazonaws.com	SHA256	nYp+gHas80HY3DqbP4 yanCDFhqDVjseefVbH EXqH2Ec
git-codecommit.ap-southeast-3.amazonaws.com	MD5	64:d9:e0:53:19:4f: a8:91:9a:c3:53:22: a6:a8:ed:a6
git-codecommit.ap-southeast-3.amazonaws.com	SHA256	ATdkGSFhpqIu7RqUVT /1RZo6MLxxxUW9NoDV MbAc/6g
git-codecommit.me-central-1.amazonaws.com	MD5	bd:fa:e2:f9:05:84: d6:39:6f:bc:d6:8d: fe:de:61:76
git-codecommit.me-central-1.amazonaws.com	SHA256	grceUDWubo4MzG1Noa KZKUfrgPvfN3ijli0n Qr11TZA

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.eu-central-1.amazonaws.com	MD5	74:5a:e8:02:fc:b2: 9c:06:10:b4:78:84: 65:94:22:2d
git-codecommit.eu-central-1.amazonaws.com	SHA256	MwGrkiEki8QkkBt1Ag XbYt0hoZYBnZF62VY5 RzGJEUy
git-codecommit.ap-northeast-2.amazonaws.com	MD5	9f:68:48:9b:5f:fc: 96:69:39:45:58:87: 95:b3:69:ed
git-codecommit.ap-northeast-2.amazonaws.com	SHA256	eegAPQrWY9YsYo9ZHI K0mxetfXBHzAZd8Eya 53Qcwko
git-codecommit.sa-east-1.amazonaws.com	MD5	74:99:9d:ff:2b:ef: 63:c6:4b:b4:6a:7f: 62:c5:4b:51
git-codecommit.sa-east-1.amazonaws.com	SHA256	kW+VKB0jpRaG/ZbXkg btMQbKgEDK7JnISV3S VoyCmzU
git-codecommit.us-west-1.amazonaws.com	MD5	3b:76:18:83:13:2c: f8:eb:e9:a3:d0:51: 10:32:e7:d1
git-codecommit.us-west-1.amazonaws.com	SHA256	gzauWTWDXDK2u5KuMMi 5vbKTmfyerdIwgSbzY B0DLpzig
git-codecommit.eu-west-2.amazonaws.com	MD5	a5:65:a6:b1:84:02: b1:95:43:f9:0e:de: dd:ed:61:d3

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.eu-west-2.amazonaws.com	SHA256	r0Rwz5k/IHp/QyrRnf iM9j02D5UEqMbtFNTu DG2hNbs
git-codecommit.ap-south-1.amazonaws.com	MD5	da:41:1e:07:3b:9e: 76:a0:c5:1e:64:88: 03:69:86:21
git-codecommit.ap-south-1.amazonaws.com	SHA256	hUKwnTj7+Xpx4Kddb6 p45j4RazIJ4IhAMD8k 29it0fE
git-codecommit.ap-south-2.amazonaws.com	MD5	bc:cc:9f:15:f8:f3: 58:a2:68:65:21:e2: 23:71:8d:ce
git-codecommit.ap-south-2.amazonaws.com	SHA256	Xe0CyZE0vgR5Xa2YUG qf+jn8/Ut7l7nX/Cms lSFNEig
git-codecommit.ca-central-1.amazonaws.com	MD5	9f:7c:a2:2f:8c:b5: 74:fd:ab:b7:e1:fd: af:46:ed:23
git-codecommit.ca-central-1.amazonaws.com	SHA256	Qz5puafQdANVprLlj6 r0Qyh4lCNsF6ob61dG cPtFS7w
git-codecommit.eu-west-3.amazonaws.com	MD5	1b:7f:97:dd:d7:76: 8a:32:2c:bd:2c:7b: 33:74:6a:76
git-codecommit.eu-west-3.amazonaws.com	SHA256	uw7c2FL564jVoFgtc+ ikzILnKBsZz7t9+CFd SJjKbLI

Server	Kryptografischer Hash-Typ	Fingerabdruck
Git-Code-Commit. us-gov-west-1.amazonaws.com	MD5	9f:6c:19:3b:88:cd: e8:88:1b:9c:98:6a: 95:31:8a:69
Git-Code-Commit. us-gov-west-1.amazonaws.com	SHA256	djXQoSIFcg8vHe0KVH 1xW/g0F9X37tWTqu4H kng75x4
Git-Code-Commit. us-gov-east-1.amazonaws.com	MD5	00:8d:b5:55:6f:05: 78:05:ed:ea:cb:3f: e6:f0:62:f2
Git-Code-Commit. us-gov-east-1.amazonaws.com	SHA256	fVb+R0z7qW7minenW+ rUpAABRCRBTczmETAJ EQrg98
git-codecommit.eu-north-1.amazonaws.com	MD5	8e:53:d8:59:35:88: 82:fd:73:4b:60:8a: 50:70:38:f4
git-codecommit.eu-north-1.amazonaws.com	SHA256	b6KSK7xKq+V8j17iuA cjQXsG7zkqoUZZmmhY YFBq1wQ
git-codecommit.me-south-1.amazonaws.com	MD5	0e:39:28:56:d5:41: e6:8d:fa:81:45:37: fb:f3:cd:f7
git-codecommit.me-south-1.amazonaws.com	SHA256	0+NTToCGgjRHeKiBu01 0ad7R0GESz+DBLX0d/ c9wc0JU
git-codecommit.ap-east-1.amazonaws.com	MD5	a8:00:3d:24:52:9d: 61:0e:f6:e3:88:c8: 96:01:1c:fe

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.ap-east-1.amazonaws.com	SHA256	LafadYwUYW8h0NoTRp objbjNs9IRnbEwHtezD 3aAIBX0
git-codecommit.cn-north-1.amazonaws.com.cn	MD5	11:7e:2d:74:9e:3b: 94:a2:69:14:75:6f: 5e:22:3b:b3
git-codecommit.cn-north-1.amazonaws.com.cn	SHA256	IYUXxH20pTDsyYMLIp +JY8CTLS4UX+ZC5JVZ XPRaxc8
git-codecommit.cn-northwest-1.amazonaws.com.cn	MD5	2e:a7:fb:4c:33:ac: 6c:f9:aa:f2:bc:fb: 0a:7b:1e:b6
git-codecommit.cn-northwest-1.amazonaws.com.cn	SHA256	wqjd6eHd0+m0Bx+dCN uL0omUoCNjaDtZiEpW j5TmCfQ
git-codecommit.eu-south-1.amazonaws.com	MD5	b9:f6:5d:e2:48:92: 3f:a9:37:1e:c4:d0: 32:0e:fb:11
git-codecommit.eu-south-1.amazonaws.com	SHA256	1yXrWbCg3uQmJr11Xx B/ASR7ugW1Ysf5yzY0 JbudHsI
git-codecommit.ap-northeast-3.amazonaws.com	MD5	25:17:40:da:b9:d4: 18:c3:b6:b3:fb:ed: 1c:20:fe:29
git-codecommit.ap-northeast-3.amazonaws.com	SHA256	2B815B9F0AvwLnRxSV xUz4kDYmtEQUGGdQYP 8OQLXhA

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.af-south-1.amazonaws.com	MD5	21:a0:ba:d7:c1:d1: b5:39:98:8d:4d:7c: 96:f5:ca:29
git-codecommit.af-south-1.amazonaws.com	SHA256	C34ji3x/cnsDZjUpyN GXdE5pjHYimqJrQZ3l eTgqJHM
git-codecommit.il-central-1.amazonaws.com	MD5	04:74:89:16:98:7a: 61:b1:69:46:42:3c: d1:b4:ac:a9
git-codecommit.il-central-1.amazonaws.com	SHA256	uFxhp51kUWhleTLeYb xQVYm4RnNLNZ5Dbdm1 cgdS1/8

Verwendung AWS CodeCommit mit VPC-Endpunkten mit Schnittstelle

Wenn Sie Amazon Virtual Private Cloud (Amazon VPC) zum Hosten Ihrer AWS Ressourcen verwenden, können Sie eine private Verbindung zwischen Ihrer VPC und herstellen. CodeCommit Sie können diese Verbindung verwenden, um CodeCommit die Kommunikation mit den Ressourcen in der VPC zu ermöglichen, ohne das öffentliche Internet verwenden zu müssen.

Amazon VPC ist ein AWS Service, mit dem Sie AWS Ressourcen in einem von Ihnen definierten virtuellen Netzwerk starten können. Mit einer VPC haben Sie die Kontrolle über Ihre Netzwerkeinstellungen, wie IP-Adressbereich, Subnetze, Routing-Tabellen und Netzwerk-Gateways. Bei VPC-Endpunkten wird das Routing zwischen der VPC und den AWS Diensten vom AWS Netzwerk abgewickelt, und Sie können IAM-Richtlinien verwenden, um den Zugriff auf Dienstressourcen zu steuern.

Um Ihre VPC zu verbinden CodeCommit, definieren Sie einen VPC-Schnittstellen-Endpunkt für. CodeCommit Ein Schnittstellenendpunkt ist eine elastic network interface mit einer privaten IP-Adresse, die als Einstiegspunkt für Datenverkehr dient, der für einen unterstützten AWS Dienst bestimmt ist. Der Endpunkt bietet zuverlässige, skalierbare Konnektivität, CodeCommit ohne dass

ein Internet-Gateway, eine NAT-Instanz (Network Address Translation) oder eine VPN-Verbindung erforderlich sind. Weitere Informationen finden Sie unter [Was ist Amazon VPC](#) im Benutzerhandbuch zu Amazon VPC.

Note

Andere AWS Services, die VPC-Unterstützung bieten und sich integrieren lassen CodeCommit, wie z. B. AWS CodePipeline, unterstützen möglicherweise nicht die Verwendung von Amazon VPC-Endpunkten für diese Integration. Beispielsweise kann der Verkehr zwischen CodePipeline und CodeCommit nicht auf den VPC-Subnetzbereich beschränkt werden. Dienste, die Integration unterstützen, wie z. B. [AWS Cloud9](#), benötigen möglicherweise zusätzliche Dienste wie AWS Systems Manager

Schnittstelle, auf der VPC-Endpunkte basieren AWS PrivateLink, eine AWS Technologie, die private Kommunikation zwischen AWS Diensten über eine elastic network interface mit privaten IP-Adressen ermöglicht. Weitere Informationen finden Sie unter [AWS PrivateLink](#).

Die folgenden Schritte sind für Benutzer von Amazon VPC vorgesehen. Weitere Informationen finden Sie unter [Erste Schritte](#) im Amazon VPC Benutzerhandbuch.

Verfügbarkeit

CodeCommit unterstützt derzeit VPC-Endpunkte in den folgenden Bereichen: AWS-Regionen

- US East (Ohio)
- USA Ost (Nord-Virginia)
- USA West (Nordkalifornien)
- US West (Oregon)
- Europa (Irland)
- Europa (London)
- Europa (Paris)
- Europa (Frankfurt)
- Europa (Stockholm)
- Europa (Milan)
- Afrika (Kapstadt)

- Israel (Tel Aviv)
- Asien-Pazifik (Tokio)
- Asien-Pazifik (Singapur)
- Asien-Pazifik (Sydney)
- Asien-Pazifik (Jakarta)
- Naher Osten (VAE)
- Asien-Pazifik (Seoul)
- Asien-Pazifik (Osaka)
- Asien-Pazifik (Mumbai)
- Asien-Pazifik (Hyderabad)
- Asien-Pazifik (Hongkong)
- Südamerika (São Paulo)
- Naher Osten (Bahrain)
- Kanada (Zentral)
- China (Peking)
- China (Ningxia)
- AWS GovCloud (US-West)
- AWS GovCloud (US-Ost)

VPC-Endpoints erstellen für CodeCommit

Um mit der Verwendung CodeCommit mit Ihrer VPC zu beginnen, erstellen Sie einen VPC-Schnittstellen-Endpunkt für CodeCommit. CodeCommit erfordert separate Endpunkte für Git-Operationen und CodeCommit API-Operationen. Je nach Ihren geschäftlichen Anforderungen müssen Sie möglicherweise mehr als einen VPC-Endpunkt erstellen. Wenn Sie einen VPC-Endpunkt für CodeCommit erstellen, wählen Sie AWS Services und unter Service Name eine der folgenden Optionen aus:

- `com.amazonaws.region.git-codecommit`: Wählen Sie diese Option, wenn Sie einen VPC-Endpunkt für Git-Operationen mit Repositorys erstellen möchten. Wählen Sie diese Option beispielsweise, wenn Ihre Benutzer einen Git-Client und Befehle wie `git pull`, `git commit`, verwenden und `git push` wenn sie mit CodeCommit Repositorys interagieren.

- `com.amazonaws. region. git-codecommit-fips`: Wählen Sie diese Option, wenn Sie einen VPC-Endpunkt für Git-Operationen mit CodeCommit Repositorys erstellen möchten, der dem US-Regierungsstandard Federal Information Processing Standard (FIPS) Publication 140-2 entspricht.

 Note

FIPS-Endpunkte für Git sind nicht in allen AWS Regionen verfügbar. Weitere Informationen finden Sie unter [Endpunkte für Git-Verbindungen](#).

- `com.amazonaws. region.codecommit`: Wählen Sie diese Option, wenn Sie einen VPC-Endpunkt für CodeCommit API-Operationen erstellen möchten. Wählen Sie diese Option beispielsweise, wenn Ihre Benutzer die AWS CLI, die CodeCommit API oder die verwenden, AWS SDKs um mit CodeCommit Vorgängen wie `CreateRepository`, `ListRepositories` und zu interagieren. `PutFile`
- `com.amazonaws. region.codecommit-fips`: Wählen Sie diese Option, wenn Sie einen VPC-Endpunkt für CodeCommit API-Operationen erstellen möchten, der dem US-Regierungsstandard Federal Information Processing Standard (FIPS) Publication 140-2 entspricht.

 Note

FIPS-Endpunkte sind nicht in allen Regionen verfügbar. AWS Weitere Informationen finden Sie unter dem Eintrag AWS CodeCommit in [Federal Information Processing Standard \(FIPS\) 140-2 Overview](#).

Erstellen einer VPC-Endpunktrichtlinie für CodeCommit

Sie können eine Richtlinie für Amazon VPC-Endpunkte erstellen, für die Sie CodeCommit Folgendes angeben können:

- Prinzipal, der die Aktionen ausführen kann.
- Aktionen, die ausgeführt werden können
- Ressourcen, für die Aktionen ausgeführt werden können

Beispiel: Ein Unternehmen möchte möglicherweise den Zugriff auf Repositorys auf den Netzwerkadressbereich einer VPC einschränken. Ein Beispiel für diese Art von Richtlinie finden Sie hier: [Beispiel 3: Erlaubt einem Benutzer, der von einem bestimmten IP-Adressbereich aus eine](#)

[Verbindung herstellt, Zugriff auf ein Repository](#) . Das Unternehmen konfigurierte zwei Git-VPC-Endpunkte für die Region USA Ost (Ohio): `com.amazonaws.us-east-2.codecommit` und `com.amazonaws.us-east-2.git-codecommit-fips`. Sie möchten Code-Pushs an ein CodeCommit Repository zulassen, das *MyDemoRepo* nur auf dem FIPS-konformen Endpunkt benannt ist. Um dies zu erzwingen, würden die Benutzer auf dem Endpunkt `com.amazonaws.us-east-2.codecommit` eine Richtlinie ähnlich der folgenden konfigurieren, die speziell Git-Push-Aktionen verweigert:

```
{
  "Statement": [
    {
      "Action": "*",
      "Effect": "Allow",
      "Resource": "*",
      "Principal": "*"
    },
    {
      "Action": "codecommit:GitPush",
      "Effect": "Deny",
      "Resource": "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo",
      "Principal": "*"
    }
  ]
}
```

Important

Der globale Bedingungsschlüssel `aws:VpcSourceIp` wird in CodeCommit Repositories in IAM-Richtlinien für Befehle nicht unterstützt. `git push`

Weitere Informationen finden Sie unter [Erstellen eines Schnittstellenendpunkts](#) im Amazon VPC Leitfaden.

Kontingente in AWS CodeCommit

In der folgenden Tabelle werden Kontingente in beschrieben CodeCommit. Informationen zu Kontingenten, die geändert werden können, finden Sie unter [AWS CodeCommit -Endpunkte und Kontingente](#). Informationen zur Beantragung einer Erhöhung der Service Quotas finden Sie unter

[AWS Servicekontingenten](#). Hinweise zu den erforderlichen Versionen von Git und anderer Software finden Sie unter [Kompatibilität für CodeCommit Git und andere Komponenten](#).

Namen für Genehmigungsregeln und Genehmigungsregelvorlagen	Beliebige Kombination aus Buchstaben, Ziffern, Punkten, Leerzeichen, Unterstrichen und Bindestrichen und mit einer Länge zwischen 1 und 100 Zeichen. Bei den Namen muss die Groß- und Kleinschreibung beachtet werden. Namen dürfen nicht auf .git enden und keines folgenden Zeichen enthalten: ! ? @ # \$ % ^ & * () + = { } [] \ / > < ~ ` ' " ; :
Länge des Inhalts von Genehmigungsregeln	3 000 Zeichen
Länge der Beschreibung einer Genehmigungsregelvorlage	1 000 Zeichen
Zielverweise für Genehmigungsregelvorlagen	100
Genehmigungsregelvorlagen	1000 in einem AWS-Region
Genehmigungsregeln für eine Pull-Anforderung	Maximal 30. 25 davon können aus Genehmigungsregelvorlagen stammen.
Genehmigungsregeln für eine Pull-Anforderung, die aus einer Genehmigungsregelvorlage erstellt werden	25
Genehmigungen für eine Pull-Anforderung	200
Genehmiger in einem Genehmigungs-Pool	50
Branch-Namen	Jede Kombination von zulässigen Zeichen mit einer Länge zwischen 1 und 256 Zeichen, mit der Ausnahme, dass Zweignamen mit genau 40 Hexadezimalzeichen nicht zulässig sind. Branch-Namen dürfen nicht:

	• mit einem Schrägstrich (/) oder Punkt (.) anfangen oder enden. • nur das einzelne Zeichen @ aufweisen. • zwei oder mehr aufeinanderfolgende Punkte (.), Schrägstriche (/) oder die folgende Zeichenkombination enthalten: @{ • Leerzeichen oder eines der folgenden Zeichen enthalten: ? ^ * [\ ~ : Branch-Namen sind Referenzen. Viele der Einschränkungen für Branch-Namen basieren auf dem Git-Referenzstandard. Weitere Informationen finden Sie unter Git Internals und git-check-ref-format.
Länge des Kommentars	Maximal 10.240 Zeichen.
Komprimierte Archivgröße für die Initialisierung eines CodeCommit Repositorys in AWS CloudFormation	Maximal 20 GB bei der Komprimierung.
Benutzerdefinierte Daten für Auslöser	Dieses Zeichenfolgenfeld darf maximal 1 000 Zeichen enthalten. Es kann nicht zur Übergabe dynamischer Parameter verwendet werden.
Anzeige in der Konsole	Eine Datei oder ein Vergleich zwischen Dateien kann in der Konsole möglicherweise nicht angezeigt werden, wenn: • Die Datei ist größer als 2 MB • Die Datei enthält mehr als 25.000 Zeichen in einer einzigen Zeile • Ein Vergleich enthält insgesamt mehr als 6.500 Zeilen mit Unterschieden

E-Mail-Adressen in Commits in der Konsole

Beliebige Kombination aus zulässigen Zeichen und mit einer Länge zwischen 1 und 256 Zeichen. E-Mail-Adressen werden nicht validiert.

Dateipfade

Beliebige Kombination aus zulässigen Zeichen und mit einer Länge zwischen 1 und 4,096 Zeichen. Dateipfade müssen einen eindeutigen Namen haben, in dem die Datei und der genaue Speicherort der Datei angegeben sind. Dateipfade dürfen nicht mehr als 20 Verzeichnisse umfassen. Außerdem gelten für Dateipfade folgende Regeln:

- Sie dürfen keine leeren Zeichenfolgen enthalten.
- Sie dürfen kein relativer Dateipfad sein.
- Sie dürfen keine der folgenden Zeichenkombinationen enthalten:

 `./`

 `../`

 `//`
- Sie dürfen nicht mit einem Schrägstrich oder umgekehrten Schrägstrich (Backslash) enden.

Dateinamen und -pfade müssen vollständig qualifiziert sein. Der Dateiname und der Pfad zu einer Datei auf Ihrem lokalen Computer müssen den Standards für das Betriebssystem entsprechen. Verwenden Sie bei der Angabe des Pfads zu einer Datei in einem CodeCommit Repository die Standards für Amazon Linux.

Dateigröße	Maximal 6 MB für jede einzelne Datei bei Verwendung der CodeCommit Konsole APIs,, oder AWS CLI.
Größe des Git-Blobs	Maximal 2 GB  Note Die Anzahl oder die Gesamtgröße aller Dateien in einem einzelnen Commit ist nicht begrenzt, sofern die Metadaten den Maximalwert von 6 MB nicht übersteigen und ein einzelner Blob nicht größer ist als 2 GB.
Grafische Branch-Darstellung im Commit Visualizer	35 pro Seite. Falls sich mehr als 35 Branches auf einer einzelnen Seite befinden, wird keine Grafik angezeigt.
Metadaten für Commit	Maximal 20 MB für die kombinierten Metadaten für einen Commit (z. B. die Kombination aus Autoreneninformationen, Datum, übergeordneter Commit-Liste und Commit-Nachrichten) bei Verwendung der CodeCommit Konsole APIs,, oder AWS CLI.  Note Es gibt keine Begrenzung für die Anzahl oder die Gesamtgröße aller Dateien in einem einzigen Commit, solange die Metadaten 6 MB nicht überschreiten, eine einzelne Datei 6 MB nicht überschreitet und ein einzelner Blob 2 GB nicht überschreitet.

Anzahl der Dateien in einem Commit	Maximal 100.
Anzahl der offenen Pull-Requests	Maximal 1.000.
Anzahl Referenzen für einzelnen Push	Maximal 4 000 (einschließlich Erstellen, Löschen und Aktualisieren). Für die Gesamtanzahl der Referenzen im Repository ist keine Einschränkung vorhanden.
Anzahl der Repositorys	Maximal 5.000 pro Amazon Web Services Services-Konto. Dieses Limit kann geändert werden. Weitere Informationen finden Sie unter AWS CodeCommit Endpunkte und Kontingente und AWS Service Quotas .
Anzahl Auslöser im Repository	Maximal 10

Regionen

CodeCommit ist in den folgenden AWS-Regionen Sprachen verfügbar:

- US East (Ohio)
- USA Ost (Nord-Virginia)
- USA West (Nordkalifornien)
- US West (Oregon)
- Europa (Irland)
- Europa (London)
- Europa (Paris)
- Europa (Frankfurt)
- Europa (Stockholm)
- Europa (Milan)
- Afrika (Kapstadt)
- Israel (Tel Aviv)
- Asien-Pazifik (Tokio)
- Asien-Pazifik (Singapur)
- Asien-Pazifik (Sydney)
- Asien-Pazifik (Jakarta)
- Naher Osten (VAE)
- Asien-Pazifik (Seoul)
- Asien-Pazifik (Osaka)
- Asien-Pazifik (Mumbai)
- Asien-Pazifik (Hyderabad)
- Asien-Pazifik (Hongkong)
- Südamerika (São Paulo)
- Naher Osten (Bahrain)
- Kanada (Zentral)
- China (Peking)
- China (Ningxia)
- AWS GovCloud (US-West)

	• AWS GovCloud (US-Ost) Weitere Informationen finden Sie unter Regionen und Git-Verbindungsendpunkte.
Repository-Beschreibungen	Beliebige Zeichenkombination und mit einer Länge zwischen 0 und 1 000 Zeichen. Repository-Beschreibungen sind optional.
Repository-Namen	Beliebige Kombination aus Buchstaben, Ziffern, Punkten, Unterstrichen und Bindestrichen und mit einer Länge zwischen 1 und 100 Zeichen. Bei den Namen muss die Groß- und Kleinschreibung beachtet werden. Repository-Namen dürfen nicht auf .git enden und keines folgenden Zeichen enthalten: ! ? @ # \$ % ^ & * () + = { } [] \ / > < ~ ` ' " ; :
Namen von Repository-Tag-Schlüsseln	Beliebige Kombination aus Unicode-Buchstaben, Zahlen, Leerstellen und zulässigen UTF-8-Zeichen mit einer Länge zwischen 1 und 128 Zeichen. Zulässige Zeichen sind + - = . _ : / @ Tag-Schlüsselnamen müssen eindeutig sein. Jeder Schlüssel darf nur einen Wert haben. Ein Tag darf nicht: • mit aws : beginnen • nur aus Leerstellen bestehen • mit einem Leerzeichen enden • Emojis oder eines der folgenden Zeichen enthalten: ? ^ * [\ ~ ! # \$ % & * () > < " ' ` [] { } ;

Repository-Tag-Werte	Beliebige Kombination aus Unicode-Buchstaben, Zahlen, Leerstellen und zulässigen UTF-8-Zeichen mit einer Länge zwischen 1 und 256 Zeichen. Zulässige Zeichen sind + - = . _ : / @ Ein Schlüssel darf nur einen Wert haben, viele Schlüssel können aber dasselbe Tag aufweisen . Ein Tag darf nicht: • nur aus Leerstellen bestehen • mit einem Leerzeichen enden • Emojis oder eines der folgenden Zeichen enthalten: ? ^ * [\ ~ ! # \$ % & * () > < " ' ` [] { } ;
Repository-Tags	Bei Tags muss die Groß- und Kleinschreibung beachtet werden. Maximal 50 pro Ressource. Tagnamen mit genau 40 Hexadezimalzeichen sind nicht zulässig.
Auslösernamen	Beliebige Kombination aus Buchstaben, Ziffern, Punkten, Unterstrichen und Bindestrichen und mit einer Länge zwischen 1 und 100 Zeichen. Auslösernamen dürfen weder Leerzeichen noch Kommata enthalten.
Benutzernamen in Commits in der Konsole	Beliebige Kombination aus zulässigen Zeichen und mit einer Länge zwischen 1 und 1,024 Zeichen.

AWS CodeCommit Befehlszeilenreferenz

Anhand dieser Referenz können Sie leichter die Verwendung der AWS CLI erlernen.

Um das zu installieren und zu konfigurieren AWS CLI

1. Laden Sie auf Ihrem lokalen Computer das herunter und installieren Sie es AWS CLI. Dies ist eine Voraussetzung für die Interaktion mit CodeCommit über die Befehlszeile. Wir empfehlen, AWS CLI Version 2 zu installieren. Es ist die neueste Hauptversion von AWS CLI und unterstützt alle aktuellen Funktionen. Es ist die einzige Version von AWS CLI , die die Verwendung eines Root-Kontos, Verbundzugriff oder temporärer Anmeldeinformationen mit git-remote-codecommit unterstützt.

Weitere Informationen finden Sie [unter Einrichtung der AWS Befehlszeilenschnittstelle](#).

Note

CodeCommit funktioniert nur mit den AWS CLI Versionen 1.7.38 und höher. Es hat sich bewährt, die neueste verfügbare Version zu installieren oder AWS CLI auf sie zu aktualisieren. Führen AWS CLI Sie den `aws --version` Befehl aus, um festzustellen, welche Version von Sie installiert haben.

Informationen zum Upgrade einer älteren Version von AWS CLI auf die neueste Version finden Sie unter [Installation von AWS Command Line Interface](#).

2. Führen Sie diesen Befehl aus, um zu überprüfen, ob die CodeCommit Befehle für installiert AWS CLI sind.

```
aws codecommit help
```

Dieser Befehl gibt eine Liste von CodeCommit Befehlen zurück.

3. Konfigurieren Sie das AWS CLI mit einem Profil, indem Sie den `configure` Befehl verwenden, wie folgt:.

```
aws configure
```

Wenn Sie dazu aufgefordert werden, geben Sie den AWS Zugriffsschlüssel und den AWS geheimen Zugriffsschlüssel des IAM-Benutzers an, mit CodeCommit dem Sie ihn verwenden möchten. Stellen Sie außerdem sicher, dass Sie angeben, AWS-Region wo sich das Repository befindet, z. B. `us-east-2` Wenn Sie nach dem standardmäßigen Ausgabeformat gefragt werden, geben Sie `json` an. Wenn Sie beispielsweise ein Profil für einen IAM-Benutzer konfigurieren:

AWS Access Key ID [None]: *Type your IAM user AWS access key ID here, and then press Enter*

AWS Secret Access Key [None]: *Type your IAM user AWS secret access key here, and then press Enter*

Default region name [None]: *Type a supported region for CodeCommit here, and then press Enter*

Default output format [None]: *Type json here, and then press Enter*

Weitere Informationen zum Erstellen und Konfigurieren von Profilen zur Verwendung mit dem AWS CLI finden Sie im Folgenden:

- [Benannte Profile](#)
- [Verwenden einer IAM-Rolle in der AWS CLI](#)
- [Befehl „Set“](#)
- [Verbindung zu AWS CodeCommit Repositories mit rotierenden Anmeldeinformationen herstellen](#)

Um eine Verbindung zu einem Repository oder einer Ressource in einem anderen Repository herzustellen AWS-Region, müssen Sie das AWS CLI mit dem Standardregionsnamen neu konfigurieren. Zu den unterstützten Standardregionsnamen für CodeCommit gehören:

- us-east-2
- us-east-1
- eu-west-1
- us-west-2
- ap-northeast-1
- ap-southeast-1
- ap-southeast-2
- ap-southeast-3
- me-central-1
- eu-central-1
- ap-northeast-2
- sa-east-1
- us-west-1

- eu-west-2
- ap-south-1
- ap-south-1
- ca-central-1
- us-gov-west-1
- us-gov-east-1
- eu-north-1
- ap-east-1
- me-south-1
- cn-north-1
- cn-northwest-1
- eu-south-1
- ap-northeast-3
- af-south-1
- il-central-1

Weitere Informationen zu CodeCommit und finden AWS-Region Sie unter [Regionen und Git-Verbindungsendpunkte](#). Weitere Informationen zu IAM, Zugriffsschlüsseln und geheimen Schlüsseln finden Sie unter [Wie erhalte ich Anmeldeinformationen?](#) und [Zugriffsschlüssel für IAM-Benutzer verwalten](#). Weitere Informationen zu den Profilen AWS CLI und finden Sie unter [Benannte Profile](#).

Führen Sie den folgenden Befehl aus, um eine Liste aller verfügbaren CodeCommit Befehle anzuzeigen:

```
aws codecommit help
```

Um Informationen zu einem CodeCommit Befehl anzuzeigen, führen Sie den folgenden Befehl aus, wobei der Name des Befehls *command-name* steht (z. B. create-repository):

```
aws codecommit command-name help
```


Im Folgenden finden Sie Beschreibungen und beispielhafte Anwendungen der Befehle in der AWS CLI:

- [associate-approval-rule-template-mit-Repository](#)
- [batch-associate-approval-rule-template-with-repositories](#)
- [batch-disassociate-approval-rule-template-from-repositories](#)
- [batch-describe-merge-conflicts](#)
- [batch-get-commits](#)
- [batch-get-repositories](#)
- [create-approval-rule-template](#)
- [Zweig erstellen](#)
- [create-commit](#)
- [create-pull-request](#)
- [create-pull-request-approval-Regel](#)
- [Repository erstellen](#)
- [create-unreferenced-merge-commit](#)
- [delete-approval-rule-template](#)
- [delete-branch](#)
- [delete-comment-content](#)
- [delete-file](#)
- [Projektarchiv löschen](#)
- [describe-merge-conflicts](#)
- [delete-pull-request-approval-Regel](#)
- [describe-pull-request-events](#)
- [disassociate-pull-request-approval-rule-template-from-repository](#)
- [evaluate-pull-request-approval-Regeln](#)
- [get-approval-rule-template](#)
- [get-blob](#)
- [Abzweigung abrufen](#)
- [get-comment](#)
- [get-comment-reactions](#)

- [get-comments-for-compared-verpflichten](#)
- [get-comments-for-pull-Anfrage](#)
- [get-commit](#)
- [get-differences](#)
- [get-merge-commit](#)
- [get-merge-conflicts](#)
- [get-merge-options](#)
- [get-pull-request](#)
- [get-pull-request-approval-Staaten](#)
- [get-pull-request-override-Staat](#)
- [Repository abrufen](#)
- [get-repository-triggers](#)
- [list-approval-rule-templates](#)
- [list-associated-approval-rule-templates-for-repository](#)
- [Zweige auflisten](#)
- [list-pull-requests](#)
- [Listen-Repositoryen](#)
- [list-repositories-for-approval-Regelvorlage](#)
- [list-tags-for-resource](#)
- [merge-branches-by-fast-vorwärts](#)
- [merge-branches-by-squash](#)
- [merge-branches-by-three-Weg](#)
- [merge-pull-request-by-schnell vorwärts](#)
- [merge-pull-request-by-Kürbis](#)
- [merge-pull-request-by-Dreiweg](#)
- [override-pull-request-approval-Regeln](#)
- [post-comment-for-compared-verpflichten](#)
- [post-comment-for-pull-Anfrage](#)
- [post-comment-reply](#)
- [put-comment-reaction](#)

- [put-file](#)
- [put-repository-triggers](#)
- [tag-resource](#)
- [test-repository-triggers](#)
- [untag-resource](#)
- [update-approval-rule-template-Inhalt](#)
- [update-approval-rule-template-Beschreibung](#)
- [update-approval-rule-template-name](#)
- [update-comment](#)
- [update-default-branch](#)
- [update-pull-request-approval-Regelinhalt](#)
- [update-pull-request-approval-Staat](#)
- [update-pull-request-description](#)
- [update-pull-request-status](#)
- [update-pull-request-title](#)
- [update-repository-description](#)
- [update-repository-name](#)

Grundlegende Git-Befehle

Du kannst Git verwenden, um mit einem lokalen Repo und dem CodeCommit Repository zu arbeiten, mit dem du das lokale Repo verbunden hast.

Es folgen einige grundlegende Beispiele für häufig verwendete Git-Befehle.

Weitere Optionen findest du in deiner Git-Dokumentation.

Themen

- [Konfigurationsvariablen](#)
- [Remote-Repositories](#)
- [Commits](#)
- [Branches](#)
- [Tags](#)

Konfigurationsvariablen

Listet alle Konfigurationsvariablen auf.	<code>git config --list</code>
Listet nur lokale Konfigurationsvariablen auf.	<code>git config --local -l</code>
Listet nur Systemkonfigurationsvariablen auf.	<code>git config --system -l</code>
Listet nur globale Konfigurationsvariablen auf.	<code>git config --global -l</code>
Legt eine Konfigurationsvariable in der angegebenen Konfigurationsdatei fest.	<code>git config [--local --global --system] <i>variable-name</i> <i>variable-value</i></code>
Setzt den Standard-Branch-Namen für alle lokalen Repositorys auf main, wenn ein erster Commit in ein Repository erfolgt, das noch keinen Standard-Branch hat	<code>git config --global init.defaultBranch main</code>
Bearbeitet eine Konfigurationsdatei direkt. Kann auch verwendet werden, um den Speicherort einer bestimmten Konfigurationsdatei zu ermitteln. Zum Verlassen des Bearbeitungsmodus geben Sie normalerweise :q (Beenden ohne Änderungen zu speichern) oder :wq (Änderungen speichern und dann beenden) ein und drücken die Eingabetaste.	<code>git config [--local --global --system] --edit</code>

Remote-Repositorys

Initialisiert ein lokales Repo als Vorbereitung für die Verbindung mit einem Repository. CodeCommit	<code>git init</code>
Kann verwendet werden, um eine Verbindung zwischen einem lokalen Repo und einem Remote-Repository (z. B. einem CodeCommit	<code>git remote add <i>remote-name</i> <i>remote-url</i></code>

Repository) unter Verwendung des angegebenen Nicknamens, den das lokale Repo für das Repository hat, und der angegebenen URL zum CodeCommit Repository einzurichten. CodeCommit	
Erstellt ein lokales Repo, indem eine Kopie eines CodeCommit Repositories unter der angegebenen URL im angegebenen Unterordner des aktuellen Ordners auf dem lokalen Computer erstellt wird. Dieser Befehl erstellt außerdem einen Remote-Tracking-Zweig für jeden Branch im geklonten CodeCommit Repository und erstellt einen ersten Branch, der vom aktuellen Standardbranch im geklonten Repository abgezweigt ist, und checkt ihn aus. CodeCommit	<code>git clone <i>remote-url</i> <i>local-subfolder-name</i></code>
Zeigt den Spitznamen an, den das lokale Repo für das Repository verwendet. CodeCommit	<code>git remote</code>
Zeigt den Spitznamen und die URL an, die das lokale Repo für Abrufe und Pushs in das Repository verwendet. CodeCommit	<code>git remote -v</code>
Leitet abgeschlossene Commits vom lokalen Repo in das Repository weiter und verwendet dabei den angegebenen Spitznamen, den das lokale Repo für das CodeCommit Repository und den angegebenen Branch hat. CodeCommit Außerdem werden Upstream-Tracking-Informationen für das lokale Repository während des Push-Vorgangs eingerichtet.	<code>git push -u <i>remote-name</i> <i>branch-name</i></code>
Überträgt abgeschlossene Commits aus dem lokalen Repository in das CodeCommit Repository, nachdem die Upstream-Tracking-Informationen festgelegt wurden.	<code>git push</code>

Ruft abgeschlossene Commits an das lokale Repository aus dem Repository ab und verwendet dabei den angegebenen Spitznamen, den das lokale CodeCommit Repository für das Repository und den angegebenen Branch hat CodeCommit	<code>git pull <i>remote-name</i> <i>branch-name</i></code>
Ruft abgeschlossene Commits an das lokale Repository aus dem Repository ab, nachdem die Upstream-Tracking-Informationen festgelegt wurden. CodeCommit	<code>git pull</code>
Trennt das lokale Repo vom CodeCommit Repository und verwendet dabei den angegebenen Spitznamen, den das lokale Repo für das Repository hat. CodeCommit	<code>git remote rm <i>remote-name</i></code>

Commits

Zeigt an, was dem schwebenden Commit im lokalen Repository hinzugefügt wurde oder was nicht.	<code>git status</code>
Zeigt in einem übersichtlichen Format an, was zum ausstehenden Commit im lokalen Repo hinzugefügt wurde oder nicht. (M = geändert, A = hinzugefügt, D = gelöscht usw.)	<code>git status -sb</code>
Zeigt Änderungen zwischen dem ausstehenden Commit und dem letzten Commit im lokalen Repository an.	<code>git diff HEAD</code>
Fügt dem ausstehenden Commit im lokalen Repository bestimmte Dateien hinzu.	<code>git add [<i>file-name-1</i> <i>file-name-2</i> <i>file-name-N</i> <i>file-pattern</i>]</code>

Fügt dem schwebenden Commit im lokalen Repository alle neuen, geänderten und gelöschten Dateien hinzu.	<code>git add</code>
Beginnt mit dem Finalisieren des ausstehenden Commits im lokalen Repository, das einen Editor mit einer Commit-Nachricht anzeigt. Nachdem die Mitteilung eingegeben wurde, wird der schwebende Commit finalisiert.	<code>git commit</code>
Finalisiert den ausstehenden Commit im lokalen Repository, einschließlich der gleichzeitigen Angabe einer Commit-Nachricht.	<code>git commit -m "<i>Some meaningful commit comment</i>"</code>
Listet die letzten Commits im lokalen Repo auf.	<code>git log</code>
Listet die letzten Commits im lokalen Repository in einem Grafikformat auf.	<code>git log --graph</code>
Listet die letzten Commits im lokalen Repository in einem vordefinierten komprimierten Format auf.	<code>git log --pretty=oneline</code>
Listet die letzten Commits im lokalen Repository in einem vordefinierten komprimierten Format mit einem Diagramm auf.	<code>git log --graph --pretty=oneline</code>
Listet die letzten Commits im lokalen Repository in einem benutzerdefinierten Format mit einem Diagramm auf. (Weitere Optionen finden Sie unter Understanding History: Commits in der Git-Dokumentation.)	<code>git log --graph --pretty=format:"%H (%h) : %cn : %ar : %s"</code>

Branches

Listet alle Branches im lokalen Repository auf, wobei neben Ihrem aktuellen Branch ein Sternchen (*) angezeigt wird.	<code>git branch</code>
Ruft Informationen über alle vorhandenen Branches im Repository in das CodeCommit lokale Repository ab.	<code>git fetch</code>
Listet alle Zweige im lokalen Repository und alle Remote-Tracking-Zweige im lokalen Repository auf.	<code>git branch -a</code>
Listet nur Remote-Tracking-Zweige im lokalen Repo auf.	<code>git branch -r</code>
Erstellt einen neuen Zweig im lokalen Repository unter Verwendung des angegebenen Zweignamens.	<code>git branch <i>new-branch-name</i></code>
Wechselt unter Verwendung des angegebenen Zweignamens zu einem anderen Zweig im lokalen Repository.	<code>git checkout <i>other-branch-name</i></code>
Erstellt einen neuen Zweig im lokalen Repository unter Verwendung des angegebenen Zweignamens und wechselt dann zu diesem.	<code>git checkout -b <i>new-branch-name</i></code>
Verschiebt eine neue Verzweigung vom lokalen Repository in das Repository, wobei der angegebene Nickname, den das lokale CodeCommit Repository für das Repository hat, und der CodeCommit angegebene Branch-Name verwendet wird. Außerdem werden während des Push-Vorgangs Upstream-Tracking-Informationen für den Branch im lokalen Repository eingerichtet.	<code>git push -u <i>remote-name new-branch-name</i></code>

Erstellt einen neuen Zweig im lokalen Repository unter Verwendung des angegebenen Zweignamens. Verbindet dann den neuen Branch im lokalen Repo mit einem vorhandenen Branch im CodeCommit Repository und verwendet dabei den angegebenen Spitznamen, den das lokale Repository für das CodeCommit Repository hat, und den angegebenen Branch-Namen.

```
git branch --track new-branch-name  
remote-name /remote-branch-name
```

Führt Änderungen von einem anderen Zweig im lokalen Repository mit dem aktuellen Zweig im lokalen Repository zusammen.

```
git merge from-other-branch-name
```

Löscht einen Zweig im lokalen Repository, es sei denn, er enthält Werke, die noch nicht zusammengeführt wurden.

```
git branch -d branch-name
```

Löscht einen Branch im CodeCommit Repository unter Verwendung des angegebenen Nicknamens, den das lokale Repository für das CodeCommit Repository hat, und des angegebenen Branch-Namens. (Beachten Sie die Verwendung des Doppelpunkts (:).)

```
git push remote-name :branch-name
```

Tags

Listet alle Tags im lokalen Repo auf.

```
git tag
```

Zieht alle Tags aus dem CodeCommit Repository in das lokale Repo.

```
git fetch --tags
```

Zeigt Informationen zu einem bestimmten Tag im lokalen Repo an.

```
git show tag-name
```

Erzeugt ein „Lightweight“-Tag im lokalen Repo.	<code>git tag <i>tag-name</i> <i>commit-id-to-point-tag-at</i></code>
Verschiebt ein bestimmtes Tag aus dem lokalen Repository in das CodeCommit Repository und verwendet dabei den angegebenen Spitznamen, den das lokale Repo für das CodeCommit Repository hat, und den angegebenen Tagnamen.	<code>git push <i>remote-name</i> <i>tag-name</i></code>
Verschiebt alle Tags aus dem lokalen Repo in das CodeCommit Repository und verwendet dabei den angegebenen Spitznamen, den das lokale Repo für das Repository hat. CodeCommit	<code>git push <i>remote-name</i> --tags</code>
Löscht ein Tag im lokalen Repo.	<code>git tag -d <i>tag-name</i></code>
Löscht ein Tag im CodeCommit Repository mit dem angegebenen Spitznamen, den das lokale Repo für das CodeCommit Repository hat, und dem angegebenen Tagnamen. (Beachten Sie die Verwendung des Doppelpunkts (:).)	<code>git push <i>remote-name</i> :<i>tag-name</i></code>

AWS CodeCommit Dokumenthistorie des Benutzerhandbuches

In der folgenden Tabelle werden wichtige Änderungen an der Dokumentation für CodeCommit beschrieben. Um Benachrichtigungen über Aktualisierungen dieser Dokumentation zu erhalten, können Sie einen RSS-Feed abonnieren.

- API-Version: 2015-04-13

Änderung	Beschreibung	Datum
AWS CodeCommit ist für Neukunden nicht mehr verfügbar.	AWS CodeCommit ist für Neukunden nicht mehr verfügbar. Bestandskunden von AWS CodeCommit können den Service weiterhin wie gewohnt nutzen. Erfahren Sie mehr	25. Juli 2024
Die Informationen zur Fehlerbehebung für den Credential Helper wurden aktualisiert	Es wurden Informationen zur Behebung von Fehlern hinzugefügt, die durch unvollständige oder falsch konfigurierte .gitconfig-Dateien verursacht wurden.	9. Juli 2024
Aktualisierte Informationen zur Fehlerbehebung für git-remote-codecommit	Zusätzliche Informationen zur Behebung von Fehlern in extern verwalteten Umgebungen wurden hinzugefügt.	9. Juli 2024
Verweise auf wurden entfernt AWS Toolkit for Eclipse	Der AWS Toolkit for Eclipse hat das Ende seiner Lebensdauer erreicht und wird nicht mehr unterstützt.	9. Juli 2024

Verweise auf dieses Toolkit wurden entfernt.

[CodeCommit unterstützt jetzt die Verwendung von vom Kunden verwalteten Schlüsseln](#)

Sie können jetzt einen vom Kunden verwalteten Schlüssel oder einen Von AWS verwalteten Schlüssel zum Verschlüsseln und Entschlüsseln von Daten in einem Repository verwenden. Weitere Informationen finden Sie unter [Verschlüsselung, Repository erstellen AWS KMS und Repository-Einstellungen ändern](#).

21. Dezember 2023

[CodeCommit ist jetzt in Israel \(Tel Aviv\) erhältlich;](#)

CodeCommit ist jetzt in Israel (Tel Aviv) erhältlich. Weitere Informationen finden Sie unter [Regionen und Git-Verbindungsendpunkte](#).

28. August 2023

[Änderungen an verwalteten Richtlinien für CodeCommit](#)

Die AWSCode CommitFullAccess Richtlinien AWSCode CommitPowerUser und wurden mit einer zusätzlichen Genehmigung aktualisiert. Weitere Informationen finden Sie unter [CodeCommit Aktualisierungen der AWS verwalteten Richtlinien](#).

16. Mai 2023

[CodeCommit ist jetzt in drei weiteren Versionen verfügbar AWS-Regionen](#)

CodeCommit ist jetzt in drei weiteren Ländern erhältlich
AWS-Regionen: Asien-Pazifik (Jakarta), Naher Osten (VAE) und Asien-Pazifik (Hyderabad). Weitere Informationen finden Sie unter [Regionen und Git-Verbindungsendpunkte](#).

28. Februar 2023

[CodeCommit ist jetzt in Afrika \(Kapstadt\) erhältlich](#)

CodeCommit ist jetzt in einer weiteren Version erhältlich
AWS-Region: Afrika (Kapstadt). Weitere Informationen finden Sie unter [Regionen und Git-Verbindungsendpunkte](#).

15. September 2021

[Änderungen an verwalteten Richtlinien für CodeCommit](#)

Einzelheiten zu Aktualisierungen der AWS verwalteten Richtlinien für CodeCommit sind jetzt verfügbar. Weitere Informationen finden Sie unter [CodeCommit Aktualisierungen AWS verwalteter Richtlinien](#).

18. August 2021

[CodeCommit ist jetzt im asiatisch-pazifischen Raum \(Osaka\) erhältlich](#)

CodeCommit ist jetzt in einer weiteren Region erhältlich
h AWS-Region: Asien-Pazifik (Osaka). Weitere Informationen finden Sie unter [Regionen und Git-Verbindungsendpunkte](#).

14. April 2021

[AWS CloudFormation und AWS Cloud Development Kit \(AWS CDK\) ändert das Benennungsverhalten für Standardzweige in CodeCommit](#)

Repositorys, die mit AWS CloudFormation oder AWS CDK mit einem ersten Commit von Code erstellt wurden, verwenden jetzt den Standard-Branch-Namen main. Diese Änderung hat keine Auswirkungen auf bestehende Repositorys oder Branches. Kunden, die lokale Git-Clients verwenden, um ihre ersten Commits zu erstellen, haben einen Standard-Branch-Namen, der der Konfiguration dieser Git-Clients folgt. Weitere Informationen finden Sie unter [CodeCommit Ressourcen erstellen mit AWS CloudFormation](#).

4. März 2021

[CodeCommit ändert das Benennungsverhalten für Standardzweige](#)

Seit dem 19. Januar 2021 lautet der Standard-Branch-Name, der durch einen ersten Commit in ein CodeCommit Repository erstellt wurde, main. Diese Änderung hat keine Auswirkungen auf bestehende Repositories oder Branches. Kunden, die lokale Git-Clients verwenden, um ihre ersten Commits zu erstellen, haben einen Standard-Branch-Namen, der der Konfiguration dieser Git-Clients folgt. Weitere Informationen findest du unter [Mit Branches arbeiten](#), [Einen Commit erstellen](#) und [Branch-Einstellungen ändern](#).

19. Januar 2021

[CodeCommit ist jetzt in Europa \(Mailand\) verfügbar](#)

CodeCommit ist jetzt in einer weiteren Version erhältlich AWS-Region: Europa (Mailand). Weitere Informationen finden Sie unter [Regionen und Git-Verbindungsendpunkte](#).

16. September 2020

[CodeCommit fügt Unterstützung für Emoji-Reaktionen auf Kommentare hinzu](#)

CodeCommit unterstützt jetzt das Reagieren auf Kommentare von anderen Benutzern mit Emojis. Weitere Informationen findest du unter [Einen Commit kommentieren und einen Pull Request überprüfen](#).

24. Juni 2020

[CodeCommit jetzt in China \(Peking\) und China \(Ningxia\) erhältlich](#)

CodeCommit ist jetzt in zwei weiteren Ländern erhältlich in AWS-Regionen: China (Peking) und China (Ningxia). Weitere Informationen finden Sie unter [Regionen und Git-Verbindungsendpunkte](#).

23. April 2020

[CodeCommit fügt Unterstützung hinzu für git-remote-codecommit](#)

CodeCommit unterstützt Verbindungen zu CodeCommit Repositorys über HTTPS mit git-remote-codecommit, einem Hilfsprogramm, das Git modifiziert. Dies ist der empfohlene Ansatz für Verbundverbindungen oder temporären Zugriff auf Repositorys. CodeCommit Sie können es auch git-remote-codecommit mit einem IAM-Benutzer verwenden. git-remote-codecommit erfordert nicht, dass Sie Git-Anmeldeinformationen für den Benutzer einrichten. Weitere Informationen finden Sie unter [Einrichtungsschritte für HTTPS-Verbindungen AWS CodeCommit mit git-remote-codecommit](#).

4. März 2020

[CodeCommit unterstützt Sitzungs-Tags](#)

CodeCommit unterstützt die Verwendung von Sitzungs-Tags, bei denen es sich um Schlüssel-Wert-Paarattribute handelt, die Sie übergeben, wenn Sie eine IAM-Rolle übernehmen, temporäre Anmeldeinformationen verwenden oder einen Benutzer in () verbinden. AWS Security Token Service AWS STS Sie können die Informationen in diesen Tags verwenden, um die Identifizierung zu erleichtern, wer eine Änderung vorgenommen hat oder ein Ereignis verursacht hat. Weitere Informationen finden Sie unter [CodeCommit-Überwachung](#) und [Verwenden von Tags zum Bereitstellen von Identitätsinformationen in CodeCommit](#).

19. Dezember 2019

[CodeCommit ist in Asien-Pazifik \(Hongkong\) verfügbar](#)

Sie können es jetzt CodeCommit im asiatisch-pazifischen Raum (Hongkong) verwenden. Weitere Informationen über Git-Verbindungspunkte finden Sie unter [Regionen](#).

11. Dezember 2019

[CodeCommit unterstützt Amazon CodeGuru Reviewer](#)

CodeCommit unterstützt Amazon CodeGuru Reviewer, einen automatisierten Code-Review-Service, der Programmanalyse und maschinelles Lernen nutzt, um häufig auftretende Probleme zu erkennen und Korrekturen in Ihrem Java- oder Python-Code zu empfehlen. Weitere Informationen finden Sie unter [Ein Repository mit Amazon CodeGuru Reviewer verknüpfen oder die Zuordnung aufheben und Mit Pull Requests arbeiten](#).

3. Dezember 2019

[CodeCommit unterstützt Genehmigungsregeln](#)

Sie können jetzt Genehmigungsregeln verwenden, um Entwicklungs-Workflows über Repositorys hinweg so anzupassen, dass verschiedene Branches über geeignete Genehmigungen und Kontrollen für Pull-Anforderungen verfügen. Weitere Informationen finden Sie unter [Working with Approval Rule Templates](#) und [Working with Pull Requests](#).

20. November 2019

[CodeCommit unterstützt Benachrichtigungsregeln](#)

Sie können jetzt Benachrichtigungsregeln verwenden , um Benutzer über wichtige Änderungen in Repositories zu informieren. Diese Funktion ersetzt Benachrichtigungen , die vor dem 5. November 2019 erstellt wurden. Weitere Informationen finden Sie unter [Erstellen einer Benachrichtigungsregel](#).

5. November 2019

[CodeCommit ist im Mittleren Osten \(Bahrain\) verfügbar](#)

Sie können es jetzt CodeCommit im Nahen Osten (Bahrain) verwenden. Weitere Informationen über Git-Verbindungsendpunkte finden Sie unter [Regionen](#).

30. Oktober 2019

[CodeCommit fügt Unterstützung für das Abrufen von Informationen über mehrere Commits hinzu](#)

Sie können Informationen zu mehreren Commits abrufen, indem Sie den batch-get-commits Befehl in der verwenden. AWS CLI Weitere Informationen finden Sie unter [Anzeigen von Commit-Details](#).

15. August 2019

[CodeCommit ist in Europa \(Stockholm\) verfügbar](#)

Sie können es jetzt CodeCommit in Europa (Stockholm) verwenden. Weitere Informationen über Git-Verbindungsendpunkte finden Sie unter [Regionen](#).

31. Juli 2019

[CodeCommit fügt Unterstützung für das Taggen von Repositorys in der Konsole hinzu](#) [CodeCommit](#)

Sie können jetzt Tags für ein Repository hinzufügen, verwalten und entfernen, um Ihre AWS Ressourcen von der CodeCommit Konsole aus zu verwalten. Weitere Informationen finden Sie unter [Markieren eines Repositorys](#).

2. Juli 2019

[CodeCommit fügt Unterstützung für zusätzliche Git-Merge-Strategien hinzu](#)

Sie können jetzt zwischen Git-Mergestrategien wählen, wenn Pull-Anforderungen in CodeCommit zusammenggeführt werden. Sie können Zusammenführungskonflikte auch in der CodeCommit Konsole lösen. Weitere Informationen finden Sie unter [Arbeiten mit Pull-Anforderungen](#).

10. Juni 2019

[CodeCommit ist in AWS GovCloud \(US-Ost\) verfügbar](#)

Sie können es jetzt CodeCommit in AWS GovCloud (US-Ost) verwenden. Weitere Informationen über Git-Verbindungsendpunkte finden Sie unter [Regionen](#).

31. Mai 2019

[CodeCommit fügt Unterstützung für das Taggen von Repositorys hinzu](#)

Sie können jetzt Tags für ein Repository hinzufügen, verwalten und entfernen, um Sie bei der Verwaltung Ihrer AWS Ressourcen zu unterstützen. Weitere Informationen finden Sie unter [Markieren eines Repositorys](#).

30. Mai 2019

[Suchen Sie in der Konsole nach Ressourcen](#)

Sie können jetzt Schnellsuchen nach Ressourcen, z. B. Repositorys, Build-Projekten, Bereitstellungsanwendungen und Pipelines, ausführen. Wählen Sie Go to Ressource (Zur Ressource) oder drücken Sie die Taste / und geben Sie dann den Namen der Ressource ein. Weitere Informationen finden Sie im [CodeCommit -Tutorial](#).

14. Mai 2019

[CodeCommit ist in AWS GovCloud \(US-West\) verfügbar](#)

Sie können es jetzt CodeCommit in AWS GovCloud (US-West) verwenden. Weitere Informationen über Git-Verbindungsendpunkte finden Sie unter [Regionen](#).

18. April 2019

[CodeCommit fügt Unterstützung für Amazon VPC-Endpunkte hinzu](#)

Sie können jetzt eine private Verbindung zwischen Ihrer VPC und CodeCommit herstellen. Weitere Informationen finden Sie unter [Verwenden von VPC-Endpunkten CodeCommit mit Interface](#).

7. März 2019

[CodeCommit fügt eine neue API hinzu](#)

CodeCommit hat eine API zum Erstellen von Commits hinzugefügt. Weitere Informationen finden Sie unter [Erstellen eines Commits](#).

20. Februar 2019

[Aktualisierung des Inhalts](#)

Der Inhalt in diesem Handbuch wurde aktualisiert und enthält jetzt kleinere Fehlerbehebungen und zusätzliche Anleitungen zur Fehlerbehebung.

2. Januar 2019

[Aktualisierung des Inhalts](#)

Der Inhalt dieses Handbuchs wurde aktualisiert, um das neue CodeCommit Konsolenerlebnis zu unterstützen.

30. Oktober 2018

[CodeCommit und der Federal Information Processing Standard \(FIPS\)](#)

CodeCommit hat in einigen Regionen Unterstützung für den Regierungsstandard Federal Information Processing Standard (FIPS) Publication 140-2 hinzugefügt. Weitere Informationen über FIPS und FIPS-Endpunkte finden Sie unter [Federal Information Processing Standard \(FIPS\) 140-2 – Übersicht](#). Weitere Informationen über Git-Verbindungsendpunkte finden Sie unter [Regionen](#).

25. Oktober 2018

[CodeCommit fügt drei hinzu APIs](#)

CodeCommit hat drei hinzugefügt APIs , um die Arbeit mit Dateien zu unterstützen. Weitere Informationen zu Git-Verbindungsendpunkten finden Sie unter [Berechtigungen für Aktionen für einzelne Dateien](#) und [AWS CodeCommit API-Referenz](#).

27. September 2018

[CodeCommit Benachrichtigung über den Dokumentationsverlauf als RSS-Feed verfügbar](#)

Sie können jetzt Benachrichtigungen über Aktualisierungen der CodeCommit Dokumentation erhalten, indem Sie einen RSS-Feed abonnieren.

29. Juni 2018

Frühere Aktualisierungen

In der folgenden Tabelle werden die wichtigen Änderungen an der Dokumentation vor dem 29.06.2018 beschrieben.

Änderung	Beschreibung	Datum geändert
Neues Thema	Das Thema Beschränken Sie Pushs und Merges auf Branches wurde hinzugefügt. Das Thema Referenz für CodeCommit-Berechtigungen wurde aktualisiert.	16. Mai 2018
Neuer -Abschnitt	Der Abschnitt Arbeiten mit Dateien in AWS CodeCommit Repositorys wurde hinzugefügt. Die Themen Referenz für CodeCommit-Berechtigungen und Erste Schritte mit AWS CodeCommit wurden aktualisiert.	21. Februar 2018
Neues Thema	Das Thema Konfiguriere den kontoübergreifenden Zugriff auf ein AWS CodeCommit Repository mithilfe von Rollen wurde hinzugefügt.	21. Februar 2018
Neues Thema	Das Thema Integrieren AWS Cloud9 mit AWS CodeCommit wurde hinzugefügt. Das Produkt- und Service-Integrationen Thema wurde mit Informationen über AWS Cloud9 aktualisiert.	1. Dezember 2017
Neuer -Abschnitt	Der Abschnitt Mit Pull-Requests in AWS CodeCommit Repositorys arbeiten wurde hinzugefügt. Der Abschnitt Authentifizierung und Zugriffskontrolle für AWS CodeCommit wurde mit Informationen zu Berechtig	20. November 2017

Änderung	Beschreibung	Datum geändert
	ungen für Pull-Anforderungen und die Kommentarfunktion aktualisiert. Außerdem enthält er aktualisierte verwaltete Richtlinienanweisungen.	
Aktualisierte Themen	Das Produkt- und Service-Integrationen Thema wurde aktualisiert und enthält nun Links für Kunden, die ihre bestehenden Pipelines aktualisieren möchten, um Amazon CloudWatch Events zum Starten von Pipelines als Reaktion auf Änderungen in einem CodeCommit Repository zu verwenden.	11. Oktober 2017
Neue Themen	Der Abschnitt Authentifizierung und Zugriffskontrolle für AWS CodeCommit wurde hinzugefügt. Er ersetzt das Referenz-Thema zu den Zugriffsberechtigungen.	11. September 2017
Aktualisierte Themen	Der Abschnitt Verwalten von Auslösern für ein Repository wurde aktualisiert, um die Änderung der Auslöser-Konfiguration zu berücksichtigen. Themen und Bilder wurden im gesamten Handbuch aktualisiert, um Änderungen in der Navigationsleiste zu berücksichtigen.	29. August 2017
Neues Thema	Das Thema Mit Benutzereinstellungen arbeiten wurde hinzugefügt. Das Thema Tag-Details anzeigen wurde aktualisiert. Die Produkt- und Service-Integrationen Themen wurden mit Informationen zur Integration mit Amazon CloudWatch Events aktualisiert.	3. August 2017
Neue Themen	Die Integrieren Sie Visual Studio mit AWS CodeCommit Themen „Eclipse integrieren mit“ AWS CodeCommit und „wurden hinzugefügt.	29. Juni 2017
Aktualisiertes Thema	CodeCommit ist jetzt in zwei weiteren Regionen verfügbar: Asien-Pazifik (Mumbai) und Kanada (Zentral). Das Thema Regionen und Git-Verbindungsendpunkte wurde aktualisiert.	29. Juni 2017

Änderung	Beschreibung	Datum geändert
Aktualisiertes Thema	CodeCommit ist jetzt in vier weiteren Regionen verfügbar : Asien-Pazifik (Seoul), Südamerika (São Paulo), USA West (Nordkalifornien) und Europa (London). Das Thema Regionen und Git-Verbindungsendpunkte wurde aktualisiert.	6. Juni 2017
Aktualisiertes Thema	CodeCommit ist jetzt in vier weiteren Regionen verfügbar: Asien-Pazifik (Tokio), Asien-Pazifik (Singapur), Asien-Pazifik (Sydney) und Europa (Frankfurt). Das Regionen und Git-Verbindungsendpunkte Thema wurde aktualisiert und enthält nun Informationen zu Git-Verbindungsendpunkten und unterstützten Regionen für CodeCommit.	25. Mai 2017
Neues Thema	Das Thema Zweige vergleichen und zusammenführen wurde hinzugefügt. Der Inhalt des Abschnitts Arbeiten mit Zweigen wurde aktualisiert mit Informationen zur Verwendung der CodeCommit -Konsole für die Arbeit mit Branches in einem Repository.	18. Mai 2017
Neues Thema	Das Thema Vergleichen von Commits mit Informationen über das Vergleichen von Commits wurde hinzugefügt. Die Struktur des Benutzerhandbuchs wurde mit Informationen zum Verwenden von Repositorys , Commits und Branches aktualisiert.	28. März 2017
Aktualisiertes Thema	Das Thema Anzeigen von Commit-Details wurde mit Informationen zum Anzeigen des Unterschieds zwischen einem Commit und dem übergeordneten Commit in der Konsole und dem Anzeigen der Unterschiede zwischen Commits mit der AWS CLI und dem Befehl get-differences aktualisiert.	24. Januar 2017

Änderung	Beschreibung	Datum geändert
Neues Thema	Das AWS CodeCommit API-Aufrufe protokollieren mit AWS CloudTrail Thema wurde mit Informationen zur Protokollierung von Verbindungen zur CodeCommit Verwendung AWS CloudFormation hinzugefügt.	11. Januar 2017
Neues Thema	Das Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden Thema wurde mit Informationen zum Einrichten von Verbindungen zur CodeCommit Verwendung von Git-Anmeldeinformationen über HTTPS hinzugefügt.	22. Dezember 2016
Aktualisiertes Thema	Das Produkt- und Service-Integrationen Thema wurde aktualisiert und enthält nun Informationen zur Integration mit AWS CodeBuild.	5. Dezember 2016
Aktualisiertes Thema	CodeCommit ist jetzt in einer anderen Region, Europa (Irland), verfügbar. Das Regionen und Git-Verbindungsendpunkte Thema wurde aktualisiert und enthält nun Informationen zu Git-Verbindungsendpunkten und unterstützten Regionen für CodeCommit.	16. November 2016
Aktualisiertes Thema	CodeCommit ist jetzt in einer anderen Region, USA West (Oregon), verfügbar. Das Regionen und Git-Verbindungsendpunkte Thema wurde aktualisiert und enthält nun Informationen zu Git-Verbindungsendpunkten und unterstützten Regionen für CodeCommit.	14. November 2016

Änderung	Beschreibung	Datum geändert
Neues Thema	Das Erstellen Sie einen Trigger für eine Lambda-Funktion Thema wurde aktualisiert, um der Möglichkeit Rechnung zu tragen, CodeCommit Trigger im Rahmen der Erstellung der Lambda-Funktion zu erstellen. Dieser vereinfachte Prozess optimiert die Triggererstellung und konfiguriert den Trigger automatisch mit den Berechtigungen, die CodeCommit zum Aufrufen der Lambda-Funktion erforderlich sind. Das Erstellen Sie einen Trigger für eine bestehende Lambda-Funktion Thema wurde hinzugefügt und enthält nun Informationen zum Erstellen von Triggern für bestehende Lambda-Funktionen in der CodeCommit Konsole.	19. Oktober 2016
Neues Thema	CodeCommit ist jetzt in einer anderen Region, USA Ost (Ohio), verfügbar. Das Regionen und Git-Verbindungsendpunkte Thema wurde hinzugefügt, um Informationen zu Git-Verbindungsendpunkten und unterstützten Regionen für CodeCommit bereitzustellen.	17. Oktober 2016
Aktualisiertes Thema	Das Produkt- und Service-Integrationen Thema wurde aktualisiert und enthält nun Informationen zur Integration mit AWS Elastic Beanstalk.	13. Oktober 2016
Aktualisiertes Thema	Das Produkt- und Service-Integrationen Thema wurde aktualisiert und enthält nun Informationen zur Integration mit AWS CloudFormation.	6. Oktober 2016
Aktualisiertes Thema	Das Thema Für SSH-Verbindungen unter Windows wurde überarbeitet und enthält nun eine Anleitung zur Verwendung des Bash-Emulators mit SSH-Verbindungen in Windows (anstatt der PuTTY-Tool-Suite).	29. September 2016

Änderung	Beschreibung	Datum geändert
Aktualisiertes Thema	Die Erste Schritte mit CodeCommit Themen Anzeigen von Commit-Details und wurden aktualisiert und enthalten nun Informationen über den Commit Visualizer in der CodeCommit Konsole. Das Thema Kontingente wurde mit Informationen über die Erhöhung der Anzahl zulässiger Referenzen für einen einzelnen Push aktualisiert.	14. September 2016
Aktualisiertes Thema	Die Erste Schritte mit CodeCommit Themen Anzeigen von Commit-Details und. Sie wurden aktualisiert und enthalten nun Informationen zur Anzeige des Commit-Verlaufs in der CodeCommit Konsole.	28. Juli 2016
Neue Themen	Die Themen Migrieren Sie ein Git-Repository zu AWS CodeCommit und Migrieren Sie lokalen oder unversionierten Inhalt zu AWS CodeCommit wurden hinzugefügt.	29. Juni 2016
Aktualisiertes Thema	An den Themen Fehlerbehebung und Für HTTPS-Verbindungen unter Windows mit dem Credential Helper AWS CLI wurden geringfügige Aktualisierungen vorgenommen.	22. Juni 2016
Aktualisiertes Thema	Die Themen Produkt- und Service-Integrationen und Referenz zu Zugriffsberechtigungen wurden aktualisiert und enthalten nun Informationen zur Integration mit CodePipeline.	18. April 2016
Neue Themen	Der Abschnitt Verwalten von Auslösern für ein Repository wurde hinzugefügt. Die neuen Themen enthalten Beispiele (darunter Richtlinien- und Codebeispiele) zum Erstellen, Bearbeiten und Löschen von Auslösern.	7. März 2016
Neues Thema	Das Thema Produkt- und Service-Integrationen wurde hinzugefügt. Am Thema Fehlerbehebung wurden geringfügige Aktualisierungen vorgenommen.	7. März 2016

Änderung	Beschreibung	Datum geändert
Aktualisiertes Thema	Zusätzlich zum MD5 Server-Fingerabdruck CodeCommit wurde der SHA256 Server-Fingerabdruck für zu Für SSH-Verbindungen unter Linux, macOS oder Unix und hinzugefügt Für SSH-Verbindungen unter Windows .	9. Dezember 2015
Neues Thema	Das Thema Dateien in einem Repository durchsuchen wurde hinzugefügt. Dem Thema Fehlerbehebung wurden neue Problemlösungen hinzugefügt. Am gesamten Benutzerhandbuch wurden kleinere Verbesserungen vorgenommen.	5. Oktober 2015
Neues Thema	Das Thema Für SSH-Benutzer, die das nicht verwenden AWS CLI wurde hinzugefügt. Die Themen im Abschnitt Einrichtung wurden überarbeitet. Für die Benutzer wurde eine Anleitung mit Schritten für ihre jeweiligen Betriebssysteme und bevorzugten Protokolle hinzugefügt.	5. August 2015
Aktualisiertes Thema	In den Themen SSH und Linux, macOS oder Unix: Richten Sie die öffentlichen und privaten Schlüssel für Git ein und CodeCommit und Schritt 3: Richten Sie die öffentlichen und privaten Schlüssel für Git ein und CodeCommit wurden Erläuterungen und Beispiele für die Schritte der SSH-Schlüssel-ID hinzugefügt.	24. Juli 2015
Aktualisiertes Thema	Die Schritte unter Schritt 3: Richten Sie die öffentlichen und privaten Schlüssel für Git ein und CodeCommit wurden aktualisiert, um ein Problem mit IAM und dem Speichern der öffentlichen Schlüsseldatei zu beheben.	22. Juli 2015
Aktualisiertes Thema	Das Thema Fehlerbehebung wurde mit Navigationshilfen aktualisiert. Zudem wurden weitere Informationen zur Behebung von Problemen mit dem Anmeldeinformationsschlüssel hinzugefügt.	20. Juli 2015

Änderung	Beschreibung	Datum geändert
Aktualisiertes Thema	Weitere Informationen zu AWS Key Management Service Berechtigungen wurden den Themen AWS KMS und Verschlüsselung und Referenzthemen zu Zugriffsberechtigungen hinzugefügt.	17. Juli 2015
Aktualisiertes Thema	Dem Thema Fehlerbehebung wurde ein weiterer Abschnitt mit Informationen zur Fehlerbehebung in AWS Key Management Service hinzugefügt.	10. Juli 2015
Erstversion	Dies ist die erste Version des CodeCommit -Benutzer handbuchs.	9. Juli 2015

AWS Glossar

Die neueste AWS Terminologie finden Sie im [AWS Glossar](#) in der AWS-Glossar Referenz.