aws

Guide du développeur

Amazon Textract

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon Textract: Guide du développeur

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Les marques commerciales et la présentation commerciale d'Amazon ne peuvent pas être utilisées en relation avec un produit ou un service extérieur à Amazon, d'une manière susceptible d'entraîner une confusion chez les clients, ou d'une manière qui dénigre ou discrédite Amazon. Toutes les autres marques commerciales qui ne sont pas la propriété d'Amazon appartiennent à leurs propriétaires respectifs, qui peuvent ou non être affiliés ou connectés à Amazon, ou sponsorisés par Amazon.

Table of Contents

Qu'est-ce qu'Amazon Textract ?

Amazon Textract facilite l'ajout d'une détection et de l'analyse du texte d'un document à vos applications. En utilisant Amazon Textract, les clients peuvent :

- Détectez le texte dactylographié et manuscrit dans divers documents, y compris les rapports financiers, les dossiers médicaux et les formulaires fiscaux.
- Extrayez du texte, des formulaires et des tableaux de documents contenant des données structurées, à l'aide de l'API Amazon Textract Document Analysis.
- Traitez les factures et les reçus avec l'API AnalyzeExpense.
- Traiter les documents d'identification tels que les permis de conduire et les passeports délivrés par le gouvernement américain, à l'aide de l'API AnalyzeID.

Amazon Textract est basé sur la même technologie éprouvée et hautement évolutive d'apprentissage profond, développée par les imaiers d'Amazon pour analyser des milliards d'images chaque jour. Vous n'avez pas besoin d'être un expert en matière d'apprentissage automatique pour l'utiliser. Amazon Textract inclut des API simples et faciles à utiliser qui peuvent analyser des fichiers image et des fichiers PDF. Amazon Textract s'enrichit en permanence de nouvelles données, et Amazon ajoute constamment de nouvelles fonctionnalités au service.

Voici quelques cas d'utilisation courants pour utiliser Amazon Textract :

- Création d'un index de recherche intelligent— À l'aide d'Amazon Textract, vous pouvez créer des bibliothèques de texte détectées dans des fichiers image et PDF.
- Utilisation de l'extraction intelligente de texte pour le traitement du langage naturel (NLP)— Amazon Textract vous permet de contrôler comment le texte est regroupé en entrée pour les applications NLP. Elle peut extraire du texte sous forme de mots et de lignes. Il regroupe également le texte par cellules du tableau si l'analyse de la table de documents Amazon Textract est activée.
- Accélération de la capture et de la normalisation des données provenant de différentes sources— Amazon Textract permet l'extraction de données textuelles et tabulaires à partir d'une grande variété de documents, tels que des documents financiers, des rapports de recherche et des notes médicales. Avec les API Amazon Textract Analyze Document, vous pouvez facilement et rapidement extraire des données non structurées et structurées de vos documents.

- Automatisation de la capture de données depuis les formulaires— Amazon Textract permet d'extraire des données structurées à partir de formulaires. Avec les API Amazon Textract Analysis, vous pouvez intégrer des capacités d'extraction dans des flux de travail professionnels existants afin que les données utilisateur envoyées via des formulaires puissent être extraites dans un format utilisable.

Certains avantages liés à l'utilisation d'Amazon Textract sont les suivants :

- Intégration de la détection de texte de documents dans vos applications— Amazon Textract élimine la complexité liée à la création de fonctions de détection de texte dans vos applications en rendant possible l'analyse précise et performante grâce à une simple API. Vous n'avez pas besoin de vision par ordinateur, ni d'être un expert en deep learning pour utiliser Amazon Textract pour détecter le texte d'un document. Avec les API Amazon Textract Textract, vous pouvez facilement intégrer la détection de texte dans n'importe quel site web ou application d'appareil mobile ou connecté.
- Analyse évolutive d'un document— Amazon Textract vous permet d'analyser et d'extraire rapidement des données de millions de documents, ce qui peut accélérer la prise de décision.
- Faible coût— Avec Amazon Textract, vous ne payez que pour les documents que vous analysez. Il n'y a pas de tarif minimum ni aucun engagement initial. Vous pouvez démarrer avec notre offre gratuite et économisez davantage grâce à notre modèle de tarification échelonnée.

Avec le traitement synchrone, Amazon Textract peut analyser des documents d'une seule page pour les applications où la latence est essentielle. Amazon Textract fournit également des opérations asynchrones pour étendre la prise en charge de documents multipages.

Première utilisation d'Amazon Textract

Si c'est la première fois que vous utilisez Amazon Textract, nous vous recommandons de lire les sections suivantes dans l'ordre :

1. Fonctionnement d'Amazon Textract— Cette section présente les composants Amazon Textract et comment ils fonctionnent ensemble pour une expérience de bout en bout.
2. Mise en route avec Amazon Textract— Dans cette section, vous configurez votre compte et testez l'API Amazon Textract.

Utilisation d'Amazon Textract avec unAWSKIT DE DÉVELOPPEMENT LOGICIEL

Les kits de développement (SDK) AWS sont disponibles pour de nombreux langages de programmation populaires. Chaque kit SDK fournit une API, des exemples de code et de la documentation qui facilitent la création d'applications par les développeurs dans leur langage préféré.

Documentation des kits SDK	Exemples de code
AWS SDK pour C++	Exemples de code AWS SDK pour C++
AWS SDK pour Go	Exemples de code AWS SDK pour Go
AWS SDK pour Java	Exemples de code AWS SDK pour Java
AWS SDK pour JavaScript	Exemples de code AWS SDK pour JavaScript
AWS SDK pour .NET	Exemples de code AWS SDK pour .NET
AWS SDK pour PHP	Exemples de code AWS SDK pour PHP
AWS SDK pour Python (Boto3)	Exemples de code AWS SDK pour Python (Boto3)
AWS SDK pour Ruby	Exemples de code AWS SDK pour Ruby

Exemple de disponibilité

Vous n'avez pas trouvé ce dont vous avez besoin ? Demandez un exemple de code en utilisant le lien Provide feedback (Fournir un commentaire) en bas de cette page.

Fonctionnement d'Amazon Textract

Amazon Textract vous permet de détecter et d'analyser du texte dans des documents d'entrée sur une ou plusieurs pages (voirDocuments de saisie).

Amazon Textract fournit des opérations pour les actions suivantes.

- Détection du texte uniquement. Pour plus d'informations, consultez la section .Détection de texte.
- Détecter et analyser les relations entre le texte. Pour plus d'informations, consultez la section .Analyse des documents.
- Détection et analyse du texte dans les factures et les reçus. Pour plus d'informations, consultez la section .Analyse des factures et des reçus.
- Détection et analyse de texte dans les documents d'identité gouvernementaux. Pour plus d'informations, consultez la section .Analyse des documents d'identité.

Amazon Textract fournit des opérations synchrone pour le traitement de petits documents d'une seule page et avec des réponses quasi en temps réel. Pour plus d'informations, consultez Traitement de documents avec des opérations synchrone. Amazon Textract fournit également des opérations asynchrones que vous pouvez utiliser pour traiter des documents plus volumineux sur plusieurs pages. Les réponses asynchrones ne sont pas en temps réel. Pour plus d'informations, consultez Traitement de documents avec des opérations asynchrones.

Lorsqu'une opération Amazon Textract traite un document, les résultats sont renvoyés dans un tableau dethe section called “Block”ou tableau d'objetsthe section called “ExpenseDocument”objets. Les deux objets contiennent des informations détectées sur les éléments, y compris leur emplacement sur le document et leur relation avec d'autres éléments du document. Pour plus d'informations, consultez Objets de réponse Amazon Textract. Pour obtenir des exemples illustrant l'utilisation`Block`objets, voirTutoriels.

Rubriques

- Détection de texte
- Analyse des documents
- Analyse des factures et des reçus
- Analyse des documents d'identité
- Documents de saisie

- Objets de réponse Amazon Textract
- Emplacement de l'article sur une page de document

Détection de texte

Amazon Textract fournit des opérations synchrone et asynchrone qui renvoient uniquement le texte détecté dans un document. Pour les deux ensembles d'opérations, les informations suivantes sont renvoyées en plusieursthe section called “Block”objets.

- Les lignes et les mots du texte détecté
- Les relations entre les lignes et les mots du texte détecté
- La page sur laquelle le texte détecté apparaît
- L'emplacement des lignes et des mots de texte sur la page de document

Pour plus d'informations, consultez the section called “Lignes et mots de texte”.

Pour détecter le texte de manière synchrone, utilisez leDetectDocumentTextFonctionnement de l'API et transmet un fichier de document en entrée. L'ensemble des résultats est renvoyé par l'opération. Pour plus d'informations et pour voir un exemple, consultez Traitement de documents avec des opérations synchrone.

> **Note**
>
> Opération d'API Amazon Rekognition`DetectText`est différent de`DetectDocumentText`. Vous utilisez`DetectText`pour détecter du texte dans des scènes en direct, telles que des affiches ou des panneaux de signalisation routière.

Pour détecter du texte de manière asynchrone, utilisezStartDocumentTextDetectionpour commencer le traitement d'un fichier de document d'entrée. Pour obtenir les résultats, appelezGetDocumentTextDetection. Les résultats sont renvoyés dans une ou plusieurs réponses de`GetDocumentTextDetection`. Pour plus d'informations et pour voir un exemple, consultez Traitement de documents avec des opérations asynchrones.

Analyse des documents

Amazon Textract analyse les documents et les formulaires à la recherche de relations entre le texte détecté. Les opérations d'analyse Amazon Textract renvoient trois catégories d'extraction de documents : texte, formulaires et tableaux. L'analyse des factures et des reçus est gérée par un processus différent. Pour plus d'informations, voirAnalyse des factures et des reçus.

Extraction de texte

Texte brut extrait d'un document. Pour de plus amples informations, veuillez consulterLignes et mots de texte.

Extraction de formulaire

Les données de formulaire sont liées à des éléments de texte extraits d'un document. Amazon Textract représente les données de formulaire sous forme de paires clé-valeur. Dans l'exemple suivant, l'une des lignes de texte détectées par Amazon Textract estName : Jane Die. Amazon Textract identifie également une clé (Name :) et une valeur (Jane Die). Pour de plus amples informations, veuillez consulterDonnées de formulaire (paires clé-valeur).

Name : Jane Die

Adresse : 123 Any Street, Anytown, États-Unis

Date de naissance : 26-1980

Les paires clé-valeur sont également utilisées pour représenter des cases à cocher ou des boutons d'option (boutons radio) extraits des formulaires.

Homme :☑

Pour de plus amples informations, veuillez consulterÉléments de sélection.

Extraction de table

Amazon Textract peut extraire des tables, des cellules de tableau et les éléments contenus dans des cellules de tableau et peut être programmé pour renvoyer les résultats dans un fichier JSON, .csv ou .txt.

Nom	Address
Ana Caroline	123 Any Town

Pour de plus amples informations, veuillez consulter Tables. Les éléments de sélection peuvent également être extraits des tableaux. Pour de plus amples informations, veuillez consulterÉléments de sélection.

Pour les articles analysés, Amazon Textract renvoie ce qui suit en plusieursthe section called "Block"objets :

- Les lignes et les mots du texte détecté
- Le contenu des éléments détectés
- La relation entre les éléments détectés
- La page sur laquelle l'élément a été détecté
- L'emplacement de l'élément sur la page de document

Vous pouvez utiliser des opérations synchrone ou asynchrone pour analyser du texte dans un document. Pour analyser du texte de manière synchrone, utilisez leAnalyzeDocumentet transmettez un document en entrée.`AnalyzeDocument`renvoie l'ensemble des résultats. Pour plus d'informations, consultez Analyse du texte du document avec Amazon Textract.

Pour détecter du texte de manière asynchrone, utilisezStartDocumentAnalysispour commencer le traitement. Pour obtenir les résultats, appelezGetDocumentAnalysis. Les résultats sont renvoyés dans une ou plusieurs réponses de`GetDocumentAnalysis`. Pour plus d'informations et pour voir un exemple, consultez Détection ou analyse de texte dans un document multipage.

Pour spécifier le type d'analyse à effectuer, vous pouvez utiliser le`FeatureTypes`paramètre d'entrée de liste. Ajoutez TABLES à la liste pour renvoyer des informations sur les tables détectées dans le document en entrée, par exemple les cellules du tableau, le texte de cellule et les éléments de sélection dans les cellules. Ajoutez FORMS pour renvoyer des relations entre mots, telles que des paires clé-valeur et des éléments de sélection. Pour effectuer les deux types d'analyse, ajoutez TABLES et FORMS à`FeatureTypes`.

Toutes les lignes et tous les mots détectés dans le document sont inclus dans la réponse (y compris le texte non lié à la valeur de`FeatureTypes`).

Analyse des factures et des reçus

Amazon Textract extrait des données pertinentes telles que les informations de contact, les articles achetés et le nom du fournisseur, à partir de presque n'importe quelle facture ou reçu sans aucun

modèle ni configuration. Les factures et les reçus utilisent souvent diverses mises en page, ce qui rend difficile et chronophage l'extraction manuelle des données à grande échelle. Amazon Textract utilise ML pour comprendre le contexte des factures et des reçus et extrait automatiquement des données telles que la date de facture ou de réception, le numéro de facture ou de réception, le prix des articles, le montant total et les conditions de paiement pour répondre aux besoins de votre entreprise.

Amazon Textract identifie également les noms de fournisseurs critiques pour vos flux de travail, mais peuvent ne pas être explicitement étiquetés. Par exemple, Amazon Textract peut trouver le nom du fournisseur sur un reçu même s'il n'est indiqué que dans un logo en haut de la page sans combinaison clé-valeur explicite. Amazon Textract vous permet également de consolider facilement les entrées provenant de divers reçus et factures utilisant différents mots pour le même concept. Par exemple, Amazon Textract met en correspondance les relations entre les noms de champs dans différents documents tels que le numéro de client, le numéro de client et l'ID de compte, en affichant une taxonomie standard comme`INVOICE_RECEIPT_ID`. Dans ce cas, Amazon Textract représente les données de manière cohérente entre différents types de documents. Les champs qui ne sont pas alignés sur la taxonomie standard sont classés comme suit :`OTHER`.

Voici une liste des champs standard pris en charge actuellement par AnalyzeExpense :

- Nom du fournisseur :`VENDOR_NAME`
- Total :`TOTAL`
- Adresse du destinataire :`RECEIVER_ADDRESS`
- Date de facturation/réception :`INVOICE_RECEIPT_DATE`
- ID de facturation/reçu :`INVOICE_RECEIPT_ID`
- Conditions de paiement :`PAYMENT_TERMS`
- Total partiel :`SUBTOTAL`
- Date d'échéance :`DUE_DATE`
- Taxe :`TAX`
- ID du contribuable sur facture (SSN/ITIN ou EIN) :`TAX_PAYER_ID`
- Nom de l'article :`ITEM_NAME`
- Prix de l'article :`PRICE`
- Quantité de l'article :`QUANTITY`

L'API AnalyzeExpense renvoie les éléments suivants pour une page de document donnée :

- Le nombre de reçus ou de factures dans une page représentée par`ExpenseIndex`
- Le nom normalisé des champs individuels représentés par`Type`
- Nom réel du champ tel qu'il apparaît sur le document, représenté sous forme de`LabelDetection`
- La valeur du champ correspondant représentée par`ValueDetection`
- Le nombre de pages dans le document soumis représenté par`Pages`
- Numéro de page sur lequel le champ, la valeur ou les éléments de ligne ont été détectés, représenté par`PageNumber`
- La géométrie, qui inclut le cadre de sélection et les coordonnées de l'emplacement du champ, de la valeur ou des éléments de ligne individuels sur la page, représentée par`Geometry`
- Le score de confiance associé à chaque élément de données détecté sur le document, représenté par`Confidence`
- La ligne complète des articles de ligne individuels achetés, représentée par`EXPENSE_ROW`

Ce qui suit est une partie de la sortie de l'API pour un reçu traité par AnalyzeExpense qui affiche le total : 55,64$ dans le champ Document extrait en tant que standard`TOTAL`, texte réel sur le document sous la forme « Total », Score de confiance de « 97,1 », Numéro de page « 1 », Valeur totale « 55,64$ » et le cadre de sélection et les coordonnées surfaciques :


```
{
    "Type": {
        "Text": "TOTAL",
        "Confidence": 99.94717407226562
    },
    "LabelDetection": {
        "Text": "Total:",
        "Geometry": {
            "BoundingBox": {
                "Width": 0.09809663146734238,
                "Height": 0.0234375,
                "Left": 0.36822840571403503,
                "Top": 0.8017578125
            },
            "Polygon": [
                {
                    "X": 0.36822840571403503,
                    "Y": 0.8017578125
                },
                {
```

```
                    "X": 0.466325044631958,
                    "Y": 0.8017578125
                },
                {
                    "X": 0.466325044631958,
                    "Y": 0.8251953125
                },
                {
                    "X": 0.36822840571403503,
                    "Y": 0.8251953125
                }
            ]
        },
        "Confidence": 97.10792541503906
},
        "ValueDetection": {
            "Text": "$55.64",
            "Geometry": {
                "BoundingBox": {
                    "Width": 0.10395314544439316,
                    "Height": 0.0244140625,
                    "Left": 0.66837477684021,
                    "Top": 0.802734375
                },
                "Polygon": [
                    {
                        "X": 0.66837477684021,
                        "Y": 0.802734375
                    },
                    {
                        "X": 0.7723279595375061,
                        "Y": 0.802734375
                    },
                    {
                        "X": 0.7723279595375061,
                        "Y": 0.8271484375
                    },
                    {
                        "X": 0.66837477684021,
                        "Y": 0.8271484375
                    }
                ]
            },
        "Confidence": 99.85165405273438
```

```
},
"PageNumber": 1
}
```

Vous pouvez utiliser des opérations synchrone pour analyser une facture ou un reçu. Pour analyser ces documents, vous utilisez l'opération AnalyzeExpense et vous lui transmettez un reçu ou une facture.`AnalyzeExpense`renvoie l'ensemble des résultats. Pour plus d'informations, consultez Analyse des factures et des reçus avec Amazon Textract.

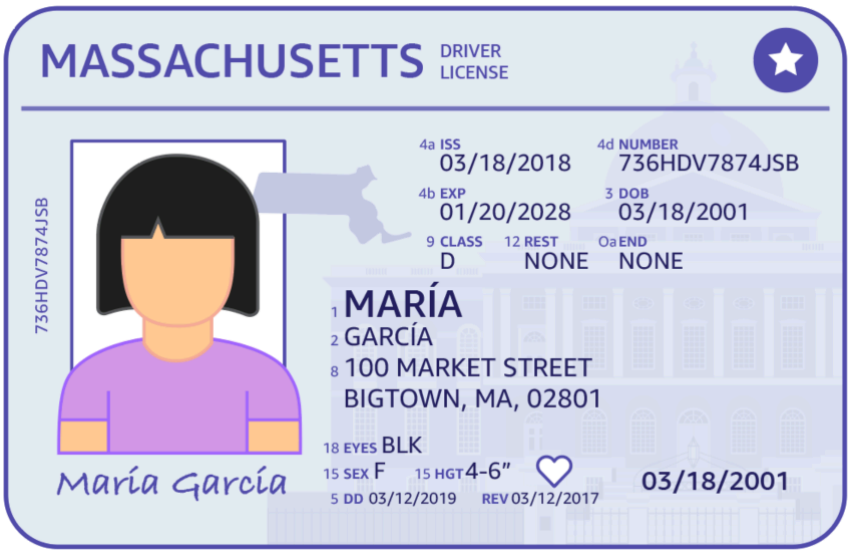
Pour analyser les factures et les reçus de manière asynchrone, utilisezStartExpenseAnalysispour commencer le traitement d'un fichier de document d'entrée. Pour obtenir les résultats, appelezGetExpenseAnalysis. Les résultats d'un appel donné àStartExpenseAnalysissont renvoyés par`GetExpenseAnalysis`. Pour plus d'informations et pour voir un exemple, consultez Traitement de documents avec des opérations asynchrones.

Analyse des documents d'identité

Amazon Textract peut extraire des informations pertinentes à partir de passeports, de permis de conduire et d'autres documents d'identité émis par le gouvernement américain à l'aide de l'API AnalyzeID. Avec Analyze ID, les entreprises peuvent extraire rapidement et avec précision des informations à partir d'identifiants tels que des permis de conduire américains, des identifiants d'État et des passeports ayant un modèle ou un format différent. L'API AnalyzeID renvoie deux catégories de types de données :

- Paires clé-valeur disponibles sur l'ID, telles que la date de naissance, la date d'émission, le numéro d'identification, la classe et les restrictions.
- Champs implicites du document qui peuvent ne pas avoir de clés explicites associées à eux tels que Nom, Adresse et Émis par.

Les noms clés sont normalisés dans la réponse. Par exemple, si votre permis de conduire indique numéro LIC (numéro de permis) et que le passeport indique le numéro de passeport, la réponse Analyze ID renvoie la clé normalisée sous la forme d' « ID de document » avec la clé brute (par exemple LIC #). Cette normalisation permet aux clients de combiner facilement des informations sur de nombreux identifiants utilisant des termes différents pour le même concept.



Analyser ID renvoie des informations dans les structures appeléesIdentityDocumentFields. Il s'agit deJSONstructures contenant deux informations : le type normalisé et la valeur associée au type. Ils ont également un score de confiance. Pour plus d'informations, consultez Objets de réponse de documentation d'identité.

Vous pouvez utiliser des opérations synchrone pour analyser un permis de conduire ou un passeport. Pour analyser ces documents, vous utilisez l'opération AnalyzeID et vous lui transmettez un document d'identité.AnalyzeIDrenvoie l'ensemble des résultats. Pour plus d'informations, consultez Analyse de la documentation d'identité avec Amazon Textract.

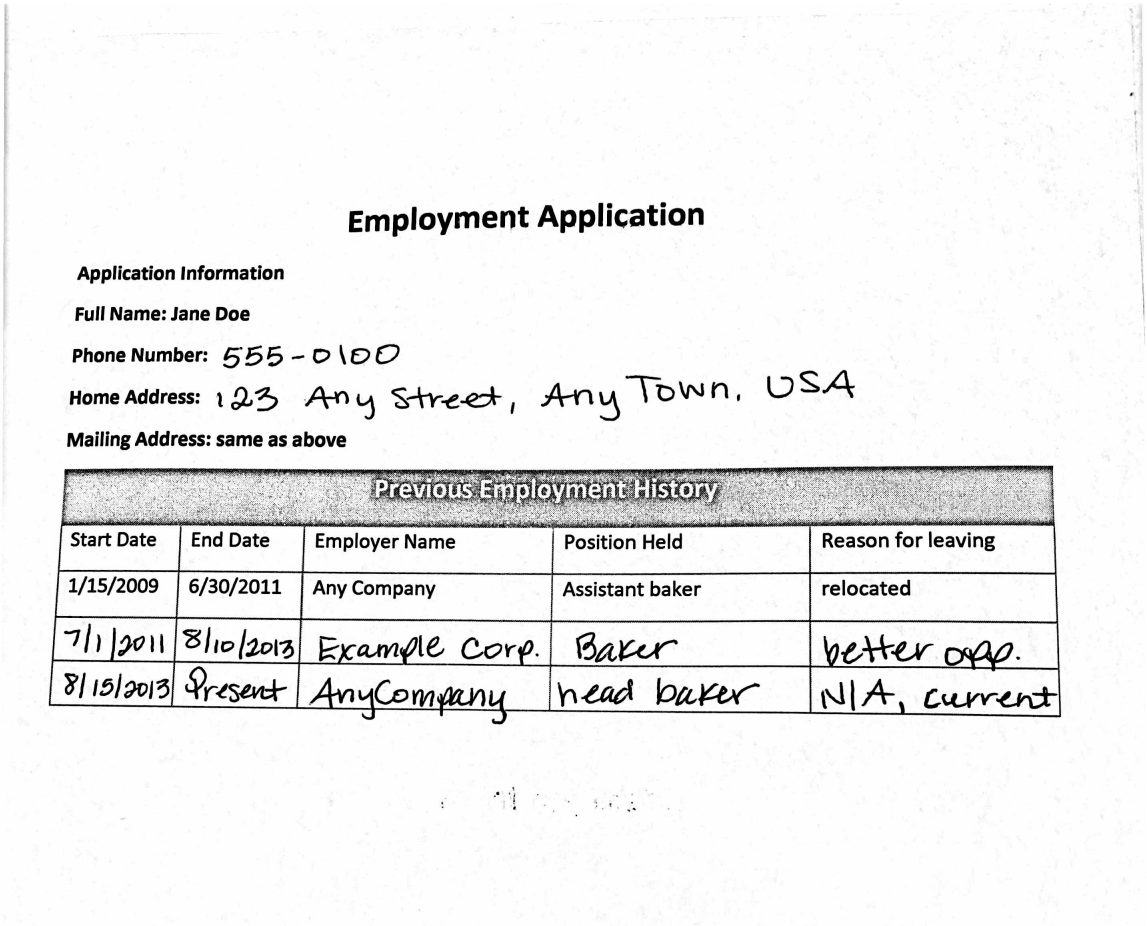
> **Note**
>
> Certains documents d'identité, tels que les permis de conduire, ont deux côtés. Vous pouvez transmettre les images avant et arrière des permis de conduire sous forme d'images distinctes dans la même demande d'API Analyze ID.

Documents de saisie

Une entrée appropriée pour une opération Amazon Textract est un document sur une ou plusieurs pages. Par exemple, il s'agit d'un document juridique, d'un formulaire, d'une pièce d'identité ou d'une lettre. Un formulaire est un document contenant des questions ou des invites permettant à un utilisateur de fournir des réponses. On peut citer par exemple un formulaire d'inscription des patients, un formulaire fiscal ou un formulaire de demande d'assurance.

Un document peut être au format JPEG, PNG, PDF ou TIFF. Avec les fichiers au format PDF et TIFF, vous pouvez traiter des documents multipages. Pour plus d'informations sur la façon dont Amazon Textract représente les documents en tant que`Block`objets, voirObjets de réponse Détection de texte et analyse de documents.

Voici un exemple de document d'entrée acceptable.



Employment Application

Application Information

Full Name: Jane Doe

Phone Number: 555-0100

Home Address: 123 Any Street, Any Town, USA

Mailing Address: same as above

Previous Employment History

Start Date	End Date	Employer Name	Position Held	Reason for leaving
1/15/2009	6/30/2011	Any Company	Assistant baker	relocated
7/1/2011	8/10/2013	Example Corp.	Baker	better opp.
8/15/2013	Present	AnyCompany	head baker	N/A, current

Pour obtenir des informations sur les limites de documents, consultezLimites strictes dans Amazon Textract.

Pour les opérations synchrone Amazon Textract, vous pouvez utiliser des documents d'entrée stockés dans un compartiment Amazon S3 ou vous pouvez transmettre des octets d'image codés en base64. Pour plus d'informations, consultez Appel d'opérations synchrone Amazon Textract. Pour les opérations asynchrones, vous devez fournir des documents d'entrée dans un compartiment Amazon S3. Pour plus d'informations, consultez Appel d'opérations asynchrones Amazon Textract.

Objets de réponse Amazon Textract

Les opérations Amazon Textract renvoient différents types d'objets en fonction des opérations exécutées. Pour détecter du texte et analyser un document générique, l'opération renvoie un objet Block. Pour analyser une facture ou un reçu, l'opération renvoie un objet ExpenseDocuments. Pour analyser la documentation d'identité, l'opération renvoie un objet IdentityDocumentFields. Pour plus d'informations sur ces objets de réponse, consultez les sections suivantes :

Rubriques

Objets de réponse Détection de texte et analyse de documents

Lorsque Amazon Textract traite un document, il crée une liste deBlockobjets pour le texte détecté ou analysé. Chaque bloc contient des informations sur un article détecté, où il se trouve, et la confiance d'Amazon Textract dans la précision du traitement.

Un document est composé des types suivants de`Block`objets.

- Pages
- Lignes et mots de texte
- Données de formulaire (paires clé-valeur)
- Tables et cellules
- Éléments de sélection

Le contenu d'un bloc dépend de l'opération que vous appelez. Si vous appelez l'une des opérations de détection de texte, les pages, les lignes et les mots du texte détecté sont renvoyés. Pour plus d'informations, consultez Détection de texte. Si vous appelez l'une des opérations d'analyse de documents, des informations sur les pages détectées, les paires clé-valeur, les tableaux, les éléments de sélection et le texte sont renvoyées. Pour plus d'informations, consultez Analyse des documents.

Momentanée`Block`Les champs d'objets sont communs aux deux types de traitement. Par exemple, chaque bloc possède un identifiant unique.

Pour obtenir des exemples illustrant l'utilisation`Block`objets, voirTutoriels.

Disposition du document

Amazon Textract renvoie une représentation d'un document sous forme de liste de différents types de`Block`objets liés dans une relation parent-enfant ou une paire clé-valeur. Les métadonnées indiquant le nombre de pages d'un document sont également renvoyées. Voici le JSON d'un type type`Block`objet de type`PAGE`.


```
{
    "Blocks": [
        {
            "Geometry": {
                "BoundingBox": {
                    "Width": 1.0,
                    "Top": 0.0,
                    "Left": 0.0,
                    "Height": 1.0
                },
                "Polygon": [
                    {
                        "Y": 0.0,
                        "X": 0.0
                    },
                    {
                        "Y": 0.0,
                        "X": 1.0
                    },
                    {
                        "Y": 1.0,
                        "X": 1.0
                    },
                    {
                        "Y": 1.0,
                        "X": 0.0
                    }
                ]
            },
            "Relationships": [
                {
                    "Type": "CHILD",
                    "Ids": [
                        "2602b0a6-20e3-4e6e-9e46-3be57fd0844b",
```

```
                        "82aedd57-187f-43dd-9eb1-4f312ca30042",
                        "52be1777-53f7-42f6-a7cf-6d09bdc15a30",
                        "7ca7caa6-00ef-4cda-b1aa-5571dfed1a7c"
                    ]
                }
            ],
            "BlockType": "PAGE",
            "Id": "8136b2dc-37c1-4300-a9da-6ed8b276ea97"
        }.....

    ],
    "DocumentMetadata": {
        "Pages": 1
    }
}
```

Un document est fabriqué à partir d'un ou plusieursPAGE. Chaque page contient une liste de blocs enfants pour les éléments principaux détectés sur la page, tels que des lignes de texte et des tableaux. Pour plus d'informations, consultez Pages.

Vous pouvez déterminer le type d'objetBlocken inspectant l'objetBlockType.

UNBlockcontient une liste deBlockobjets dans leRelationships, qui est un tableau deRelationshipobjets. UNRelationshipstableau est soit de type CHILD, soit de type VALUE. Un tableau de type CHILD permet de répertorier les éléments enfants du bloc actuel. Par exemple, si le bloc actuel est de type LINE,Relationshipscontient une liste d'ID des blocs WORD qui composent la ligne de texte. Un tableau de type VALUE est utilisé pour contenir des paires clé-valeur. Vous pouvez déterminer le type de relation en inspectant leTypedu champRelationshipobjet.

Les blocs enfants ne contiennent pas d'informations sur leurs objets Block parents.

Pour des exemples qui montrentBlock, consultezTraitement de documents avec des opérations synchrone.

Fiabilité

Les opérations Amazon Textract indiquent le pourcentage de fiabilité déterminé par Amazon Textract pour l'exactitude de l'article détecté. Pour obtenir la confiance, utilisez leConfidencedu champBlockobjet. Une valeur supérieure indique une confiance supérieure. Selon le scénario, les détections avec une faible confiance peuvent nécessiter une confirmation visuelle de la part d'un humain.

Geometry

Les opérations Amazon Textract, à l'exception de l'analyse d'identité, renvoient des informations de localisation concernant l'emplacement des éléments détectés sur une page de document. Pour obtenir cet emplacement, utilisez le`Geometry`du champ`Block`objet. Pour de plus amples informations, veuillez consulterEmplacement de l'article sur une page de document

Pages

Un document est composé d'une ou plusieurs pages. UNthe section called “Block”objet de type`PAGE`existe pour chaque page du document. UN`PAGE`bloc objet contient une liste des ID enfants des lignes de texte, des paires clé-valeur et des tables détectées sur la page de document.

Le JSON pour un`PAGE`ressemble à ce qui suit.

```
{

    "Geometry": ....
    "Relationships": [
        {
            "Type": "CHILD",
            "Ids": [
                "2602b0a6-20e3-4e6e-9e46-3be57fd0844b", // Line - Hello, world.
                "82aedd57-187f-43dd-9eb1-4f312ca30042", // Line - How are you?
                "52be1777-53f7-42f6-a7cf-6d09bdc15a30",
                "7ca7caa6-00ef-4cda-b1aa-5571dfed1a7c"
            ]
        }
    ],
    "BlockType": "PAGE",
    "Id": "8136b2dc-37c1-4300-a9da-6ed8b276ea97"  // Page identifier
},
```

Si vous utilisez des opérations asynchrones avec un document multipage au format PDF, vous pouvez déterminer la page sur laquelle se trouve un bloc en inspectant le`Page`du champ`Block`objet. Une image numérisée (image au format JPEG, PNG, PDF ou TIFF) est considérée comme un document d'une seule page, même s'il y a plusieurs pages de document sur l'image. Les opérations asynchrones renvoient toujours un`Page`valeur 1 pour les images numérisées.

Le nombre total de pages est renvoyé dans la`Pages`field ofDocumentMetadata.`DocumentMetadata`est renvoyé avec chaque liste de`Block`objets renvoyés par une opération Amazon Textract.

Lignes et mots de texte

Le texte détecté renvoyé par les opérations Amazon Textract est renvoyé dans une liste dethe section called “Block”objets. Ces objets représentent des lignes de texte ou des mots textuels détectés sur une page de document. Le texte suivant présente deux lignes de texte composées de plusieurs mots.

Il s'agit d'un texte.

En deux lignes distinctes.

Le texte détecté est renvoyé dans le`Text`d'un champ`Block`objet. Le`BlockType`détermine si le texte est une ligne de texte (LINE) ou un mot (WORD). UNMOTest un ou plusieurs caractères latins de base ISO non séparés par des espaces. UNLIGNEest une chaîne de mots contigus et délimités par des tabulations.

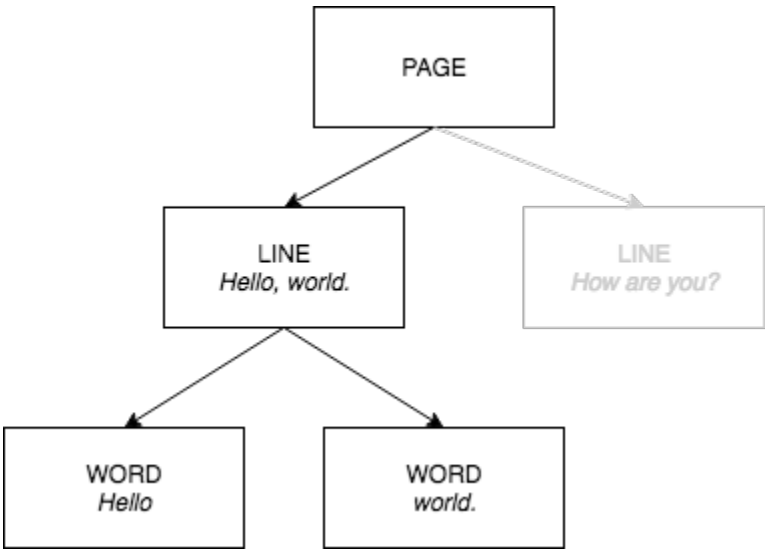
En outre, Amazon Textract déterminera si un morceau de texte a été manuscrit ou imprimé à l'aide de la`TextTypes`. Ils reviennent respectivement sous forme d'écriture manuscrite et d'impression.

L'autre`Block`sont communes à tous les types de blocs, tels que l'ID, la confiance et les informations de géométrie. Pour plus d'informations, consultez the section called “Objets de réponse Détection de texte et analyse de documents”.

Pour détecter uniquement les lignes et les mots, vous pouvez utiliserDetectDocumentTextouStartDocumentTextDetection. Pour plus d'informations, consultez Détection de texte. Pour obtenir le texte détecté (lignes et mots) et des informations sur sa relation avec d'autres parties du document, telles que les tableaux, vous pouvez utiliserAnalyzeDocumentouStartDocumentAnalysis. Pour plus d'informations, consultez Analyse des documents.

`PAGE`,`LINE`, et`WORD`les blocs sont liés les uns aux autres dans une relation parent-enfant. UNPAGEblock est le parent de tous`LINE`blocs d'objets sur une page de document. Parce qu'une ligne peut comporter un ou plusieurs mots, la`Relationships`pour un bloc LINE stocke les ID des blocs WORD enfants qui constituent la ligne de texte.

Le schéma suivant montre comment la ligneHello World.dans le texteHello World. Comment allez-vous ?est représenté par`Block`objets.



Voici la sortie JSON de`DetectDocumentText`quand la phraseHello World. Comment allez-vous ?est détecté. Le premier exemple est le JSON de la page de document. Notez comment les ID CHILD vous permettent de naviguer dans le document.

```
{
    "Geometry": {...},
    "Relationships": [
        {
            "Type": "CHILD",
            "Ids": [
                "d7fbd604-d609-4d69-857d-247a3f591238", // Line - Hello, world.
                "b6c19a93-6493-4d8e-958f-853c8f7ca055" //  Line - How are you?
            ]
        }
    ],
    "BlockType": "PAGE",
    "Id": "56ec1d77-171f-4881-9852-2b5b7e761608"
},
```

Voici le JSON pour les blocs LINE qui composent la ligne « Hello, World » :

```
{
    "Relationships": [
        {
            "Type": "CHILD",
            "Ids": [
                "7f97e2ca-063e-47a8-981c-8beee31afc01", // Word - Hello,
```

```
                "4b990aa0-af96-4369-b90f-dbe02538ed21"  // Word - world.
            ]
        }
    ],
    "Confidence": 99.63229370117188,
    "Geometry": {...},
    "Text": "Hello, world.",
    "BlockType": "LINE",
    "Id": "d7fbd604-d609-4d69-857d-247a3f591238"
},
```

Voici le code JSON du bloc WORD pour le motHello, :

```
{
    "Geometry": {...},
    "Text": "Hello,",
    "TextType": "PRINTED",
    "BlockType": "WORD",
    "Confidence": 99.74746704101562,
    "Id": "7f97e2ca-063e-47a8-981c-8beee31afc01"
},
```

Le JSON final est le bloc WORD du mot. :

```
{
    "Geometry": {...},
    "Text": "world.",
    "TextType": "PRINTED",
    "BlockType": "WORD",
    "Confidence": 99.5171127319336,
    "Id": "4b990aa0-af96-4369-b90f-dbe02538ed21"
},
```

Données de formulaire (paires clé-valeur)

Amazon Textract peut extraire les données de formulaire de documents sous forme de paires clé-valeur. Par exemple, dans le texte suivant, Amazon Textract peut identifier une clé (Name :) et une valeur (Ana Caroline).

Name : Ana Caroline

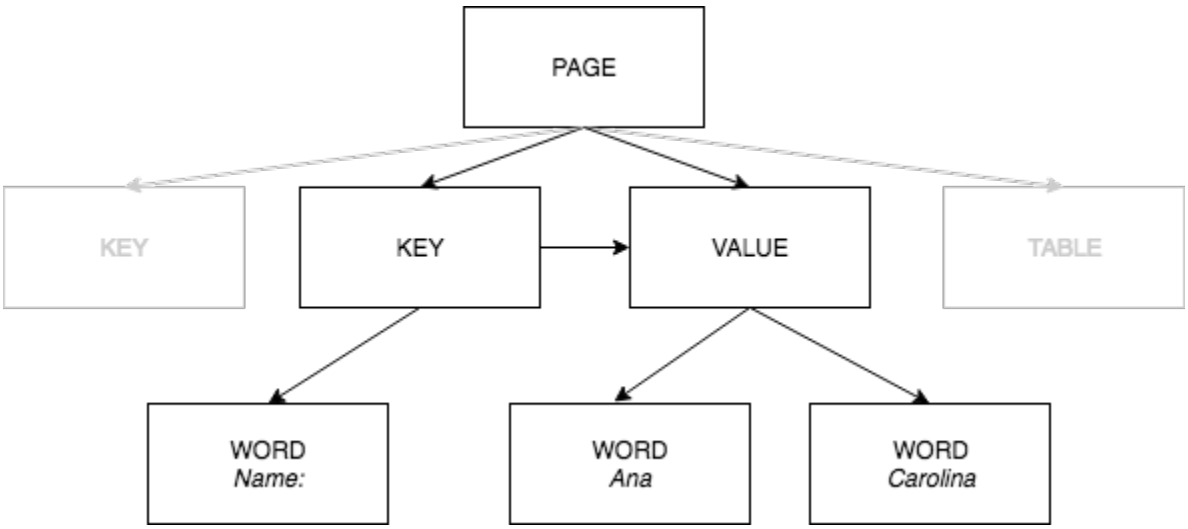
Les paires clé-valeur détectées sont renvoyées sous forme deBlockobjets dans les réponses deAnalyzeDocumentetGetDocumentAnalysis. Vous pouvez utiliser le pluginFeatureTypesparamètre d'entrée pour récupérer des informations sur les paires clé-valeur, les tables ou les deux. Pour les paires clé-valeur uniquement, utilisez la valeurFORMS. Pour voir un exemple, consultez Extraction de paires clé-valeur à partir d'un document de formulaire. Pour obtenir des informations générales sur la représentation d'un documentBlockobjets, voirObjets de réponse Détection de texte et analyse de documents.

Les objets Block de type KEY_VALUE_SET sont les conteneurs des objets KEY ou VALUE Block qui stockent des informations sur les éléments de texte liés détectés dans un document. Vous pouvez utiliser le pluginEntityTypepour déterminer si un bloc est une CLÉ ou une VALEUR.

- UNCLÉObjets contient des informations sur la clé du texte lié. Par exemple,Name :. Un bloc KEY comporte deux listes de relations. Une relation de type VALUE est une liste contenant l'ID du bloc VALUE associé à la clé. Une relation de type CHILD est une liste d'ID des blocs WORD qui composent le texte de la clé.
- UNVALEURObjets contient des informations sur le texte associé à une clé. Dans l'exemple précédent,Ana Carolinereprésente la valeur de la cléName :. Un bloc VALUE a une relation avec une liste de blocs CHILD qui identifient les blocs WORD. Chaque bloc WORD contient l'un des mots qui composent le texte de la valeur. UNVALUEObjets peuvent également contenir des informations sur les éléments sélectionnés. Pour plus d'informations, consultez Éléments de sélection.

Chaque instance d'un KEY_VALUE_SETBlockest un enfant de la pageBlockobjet correspondant à la page actuelle.

Le schéma suivant montre comment la paire clé-valeurName : Ana Carolineest représenté parBlockobjets.



Les exemples suivants montrent comment la paire clé-valeurName : Ana Carolineest représenté par JSON.

Le bloc PAGE comporte des blocs CHILD de typeKEY_VALUE_SETpour chaque bloc KEY et VALUE détecté dans le document.

```
{
    "Geometry": ....
    "Relationships": [
        {
            "Type": "CHILD",
            "Ids": [
                "2602b0a6-20e3-4e6e-9e46-3be57fd0844b",
                "82aedd57-187f-43dd-9eb1-4f312ca30042",
                "52be1777-53f7-42f6-a7cf-6d09bdc15a30", // Key - Name:
                "7ca7caa6-00ef-4cda-b1aa-5571dfed1a7c"  // Value - Ana Caroline
            ]
        }
    ],
    "BlockType": "PAGE",
    "Id": "8136b2dc-37c1-4300-a9da-6ed8b276ea97"  // Page identifier
},
```

Le JSON suivant indique que le bloc KEY (52be1777-53f7-42f6-a7cf-6d09bdc15a30) a une relation avec le bloc VALUE (7ca7caa6-00ef-4cda-b1aa-5571dfed1a7c). Il possède également un bloc CHILD pour le bloc WORD (c734fca6-c4c4-415c-b6c1-30f7510b72ee) qui contient le texte de la clé (Name :).

```
{
    "Relationships": [
        {
            "Type": "VALUE",
            "Ids": [
                "7ca7caa6-00ef-4cda-b1aa-5571dfed1a7c"  // Value identifier
            ]
        },
        {
            "Type": "CHILD",
            "Ids": [
                "c734fca6-c4c4-415c-b6c1-30f7510b72ee"  // Name:
            ]
        }
    ],
    "Confidence": 51.55965805053711,
    "Geometry": ....,
    "BlockType": "KEY_VALUE_SET",
    "EntityTypes": [
        "KEY"
    ],
    "Id": "52be1777-53f7-42f6-a7cf-6d09bdc15a30"  //Key identifier
},
```

Le JSON suivant indique que le bloc VALUE 7ca7caa6-00ef-4cda-b1aa-5571dfed1a7c possède une liste ENFANT d'ID pour les blocs WORD qui composent le texte de la valeur (AnaetCaroline).

```
{
    "Relationships": [
        {
            "Type": "CHILD",
            "Ids": [
                "db553509-64ef-4ecf-ad3c-bea62cc1cd8a", // Ana
                "e5d7646c-eaa2-413a-95ad-f4ae19f53ef3"  // Carolina
            ]
        }
    ],
    "Confidence": 51.55965805053711,
    "Geometry": ....,
    "BlockType": "KEY_VALUE_SET",
    "EntityTypes": [
        "VALUE"
    ],
```

```
    "Id": "7ca7caa6-00ef-4cda-b1aa-5571dfed1a7c" // Value identifier
}
```

Le fichier JSON suivant affiche le`Block`objets pour les motsName :,Ana, etCaroline.

```
{
    "Geometry": {...},
    "Text": "Name:",
    "TextType": "PRINTED".
    "BlockType": "WORD",
    "Confidence": 99.56285858154297,
    "Id": "c734fca6-c4c4-415c-b6c1-30f7510b72ee"
},
 {
    "Geometry": {...},
    "Text": "Ana",
    "TextType": "PRINTED",
    "BlockType": "WORD",
    "Confidence": 99.52057647705078,
    "Id": "db553509-64ef-4ecf-ad3c-bea62cc1cd8a"
},
{
    "Geometry": {...},
    "Text": "Carolina",
    "TextType": "PRINTED",
    "BlockType": "WORD",
    "Confidence": 99.84207916259766,
    "Id": "e5d7646c-eaa2-413a-95ad-f4ae19f53ef3"
},
```

Tables

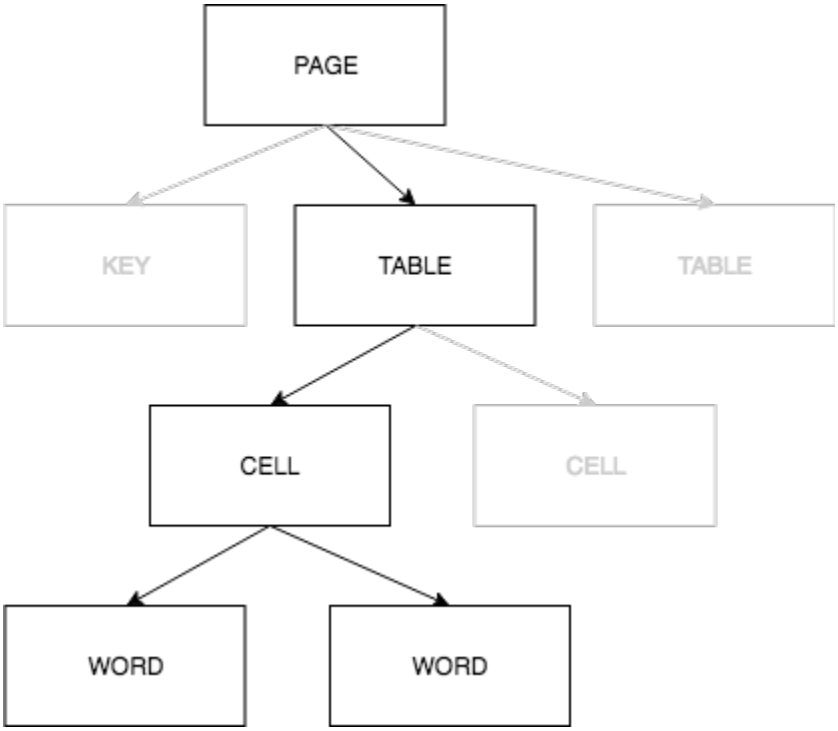
Amazon Textract peut extraire des tables et des cellules d'une table. Par exemple, lorsque le tableau suivant est détecté sur un formulaire, Amazon Textract détecte une table comportant quatre cellules.

Nom	Address
Ana Caroline	123 Any Town

Les tables détectées sont renvoyées sous formeBlockobjets dans les réponses deAnalyzeDocumentetGetDocumentAnalysis. Vous pouvez utiliser le

pluginFeatureTypesparamètre d'entrée pour récupérer des informations sur les paires clé-valeur, les tables ou les deux. Pour les tables uniquement, utilisez la valeurTABLES. Pour voir un exemple, consultez Exportation de tables dans un fichier CSV. Pour obtenir des informations générales sur la représentation d'un documentBlockobjets, voirObjets de réponse Détection de texte et analyse de documents.

Le schéma suivant montre comment une cellule unique d'un tableau est représentée parBlockobjets.



Une cellule contientWORDblocs pour les mots détectés, etSELECTION_ELEMENTblocs pour les éléments de sélection tels que les cases à cocher.

Ce qui suit est un JSON partiel pour le tableau précédent, qui comporte quatre cellules.

L'objet PAGE Block contient une liste d'ID de bloc ENFANT pour le bloc TABLE et chaque LIGNE de texte détectée.

```
{
    "Geometry": {...},
    "Relationships": [
        {
            "Type": "CHILD",
            "Ids": [
                "f2a4ad7b-f21d-4966-b548-c859b84f66a4",   // Line - Name
```

```
                "4dce3516-ffeb-45e0-92a2-60770e9cb744",   // Line  - Address
                "ee506578-768f-4696-8f4b-e4917e429f50",   // Line - Ana Carolina
                "33fc7223-411b-4399-8a90-ccd3c5a2c196",   // Line  - 123 Any Town
                "3f9665be-379d-4ae7-be44-d02f32b049c2"    // Table
            ]
        }
    ],
    "BlockType": "PAGE",
    "Id": "78c3ce84-ae70-418e-add7-27058418adf6"
},
```

Le bloc TABLE inclut une liste d'ID enfants pour les cellules du tableau. Un bloc TABLE inclut également des informations de géométrie pour l'emplacement de la table dans le document. Le fichier JSON suivant indique que la table comporte quatre cellules répertoriées dans le`Ids`tableau.

```
{
    "Geometry": {...},
    "Relationships": [
        {
            "Type": "CHILD",
            "Ids": [
                "505e9581-0d1c-42fb-a214-6ff736822e8c",
                "6fca44d4-d3d3-46ab-b22f-7fca1fbaaf02",
                "9778bd78-f3fe-4ae1-9b78-e6d29b89e5e9",
                "55404b05-ae12-4159-9003-92b7c129532e"
            ]
        }
    ],
    "BlockType": "TABLE",
    "Confidence": 92.5705337524414,
    "Id": "3f9665be-379d-4ae7-be44-d02f32b049c2"
},
```

Le type de bloc des cellules du tableau est CELL. Le`Block`pour chaque cellule contient des informations sur l'emplacement de la cellule par rapport aux autres cellules du tableau. Il inclut également des informations géométriques pour l'emplacement de la cellule sur le document. Dans l'exemple précédent,`505e9581-0d1c-42fb-a214-6ff736822e8c`est l'ID enfant de la cellule contenant le motNom. L'exemple suivant illustre les informations relatives à la cellule.

```
{
    "Geometry": {...},
    "Relationships": [
```

```
        {
            "Type": "CHILD",
            "Ids": [
                "e9108c8e-0167-4482-989e-8b6cd3c3653e"
            ]
        }
    ],
    "Confidence": 100.0,
    "RowSpan": 1,
    "RowIndex": 1,
    "ColumnIndex": 1,
    "ColumnSpan": 1,
    "BlockType": "CELL",
    "Id": "505e9581-0d1c-42fb-a214-6ff736822e8c"
},
```

Chaque cellule a un emplacement dans un tableau, la première cellule étant 1,1. Dans l'exemple précédent, la cellule avec la valeurNomse trouve à la ligne 1, colonne 1. La cellule avec la valeur123 Any Townse trouve à la ligne 2, colonne 2. Un objet de bloc de cellules contient ces informations dans le`RowIndex`et`ColumnIndex`. La liste enfant contient les ID des objets WORD Block qui contiennent le texte qui se trouve dans la cellule. Les mots de la liste sont dans l'ordre dans lequel ils sont détectés, du haut à gauche de la cellule jusqu'en bas à droite de la cellule. Dans l'exemple précédent, la cellule a un ID enfant avec la valeur e9108c8e-0167-4482-989e-8b6cd3c3653e. La sortie suivante concerne le bloc WORD dont la valeur d'ID est e9108c8e-0167-4482-989e-8b6cd3c3653e :

```
"Geometry": {...},
"Text": "Name",
"TextType": "Printed",
"BlockType": "WORD",
"Confidence": 99.81139373779297,
"Id": "e9108c8e-0167-4482-989e-8b6cd3c3653e"
},
```

Éléments de sélection

Amazon Textract peut détecter des éléments de sélection tels que les boutons d'option (boutons radio) et les cases à cocher sur une page de document. Les éléments de sélection peuvent être détectés dansdonnées de formulaireet danstables. Par exemple, lorsque le tableau suivant est détecté sur un formulaire, Amazon Textract détecte les cases à cocher dans les cellules du tableau.

	D'accord	Neutral	En désaccord
Bon service	☑	☐	☐
Facile à utiliser	☐	☑	☐
Prix équitable	☑	☐	☐

Les éléments de sélection détectés sont renvoyés commeBlockobjets dans les réponses deAnalyzeDocumentetGetDocumentAnalysis.

> **Note**
>
> Vous pouvez utiliser le pluginFeatureTypesparamètre d'entrée pour récupérer des informations sur les paires clé-valeur, les tables ou les deux. Par exemple, si vous filtrez sur des tables, la réponse inclut les éléments de sélection détectés dans les tables. Les éléments de sélection détectés dans des paires clé-valeur ne sont pas inclus dans la réponse.

Les informations relatives à un élément de sélection sont contenues dans unBlockobjet de typeSELECTION_ELEMENT. Pour déterminer l'état d'un élément sélectionnable, utilisez laSelectionStatusdans le domaineSELECTION_ELEMENT. Le statut peut être l'un ou l'autreSÉLECTIONNÉouNOT_SELECTED. Par exemple, la valeur deSelectionStatuspour l'image précédente estSÉLECTIONNÉ.

UNSELECTION_ELEMENT Blockest associé à une paire clé-valeur ou à une cellule de table. UNSELECTION_ELEMENT Blockcontient des informations de cadre de sélection pour un élément de sélection dans la zone de sélectionGeometry. UNSELECTION_ELEMENT Blockn'est pas un enfant d'un objetPAGE Blockobjet.

Données de formulaire (paires clé-valeur)

Une paire clé-valeur est utilisée pour représenter un élément de sélection détecté sur un formulaire. LeKEYcontient le texte de l'élément de sélection. LeVALUEcontient le bloc SELECTION_ELEMENT. Le schéma suivant montre comment les éléments de sélection sont représentés parthe section called “Block”objets.

Pour plus d'informations sur les paires clé-valeur, consultezDonnées de formulaire (paires clé-valeur).

L'extrait JSON suivant affiche la clé d'une paire clé-valeur contenant un élément de sélection (male ☑). L'ID enfant (ID bd14cfd5-9005-498b-a7f3-45ceb171f0ff) est l'ID du bloc WORD qui contient le texte de l'élément de sélection (masculin). L'ID de valeur (Id 24aaac7f-fcce-49c7-a4f0-3688b05586d4) est l'ID duVALUEqui contient le blocSELECTION_ELEMENTbloc d'objet.

```
{
    "Relationships": [
        {
            "Type": "VALUE",
            "Ids": [
                "24aaac7f-fcce-49c7-a4f0-3688b05586d4"  // Value containing Selection
 Element
            ]
        },
        {
            "Type": "CHILD",
            "Ids": [
                "bd14cfd5-9005-498b-a7f3-45ceb171f0ff"  // WORD - male
            ]
        }
    ],
    "Confidence": 94.15619659423828,
    "Geometry": {
        "BoundingBox": {
            "Width": 0.022914813831448555,
            "Top": 0.08072036504745483,
            "Left": 0.18966935575008392,
            "Height": 0.014860388822853565
        },
        "Polygon": [
            {
                "Y": 0.08072036504745483,
                "X": 0.18966935575008392
            },
            {
                "Y": 0.08072036504745483,
                "X": 0.21258416771888733
            },
            {
                "Y": 0.09558075666427612,
                "X": 0.21258416771888733
```

```
            },
            {
                "Y": 0.09558075666427612,
                "X": 0.18966935575008392
            }
        ]
    },
    "BlockType": "KEY_VALUE_SET",
    "EntityTypes": [
        "KEY"
    ],
    "Id": "a118dc43-d5f7-49a2-a20a-5f876d9ffd79"
}
```

L'extrait JSON suivant est le bloc WORD du motHomme. Le bloc WORD possède également un bloc LINE parent.

```
{
    "Geometry": {
        "BoundingBox": {
            "Width": 0.022464623674750328,
            "Top": 0.07842985540628433,
            "Left": 0.18863198161125183,
            "Height": 0.01617223583161831
        },
        "Polygon": [
            {
                "Y": 0.07842985540628433,
                "X": 0.18863198161125183
            },
            {
                "Y": 0.07842985540628433,
                "X": 0.2110965996980667
            },
            {
                "Y": 0.09460209310054779,
                "X": 0.2110965996980667
            },
            {
                "Y": 0.09460209310054779,
                "X": 0.18863198161125183
            }
        ]
```

```
    },
    "Text": "Male",
    "BlockType": "WORD",
    "Confidence": 54.06439208984375,
    "Id": "bd14cfd5-9005-498b-a7f3-45ceb171f0ff"
},
```

Le bloc VALUE a un enfant (Id f2f5e8cd-e73a-4e99-a095-053acd3b6bfb) qui est le bloc SELECTION_ELEMENT.

```
{
    "Relationships": [
        {
            "Type": "CHILD",
            "Ids": [
                "f2f5e8cd-e73a-4e99-a095-053acd3b6bfb"  // Selection element
            ]
        }
    ],
    "Confidence": 94.15619659423828,
    "Geometry": {
        "BoundingBox": {
            "Width": 0.017281491309404373,
            "Top": 0.07643391191959381,
            "Left": 0.2271782010793686,
            "Height": 0.026274094358086586
        },
        "Polygon": [
            {
                "Y": 0.07643391191959381,
                "X": 0.2271782010793686
            },
            {
                "Y": 0.07643391191959381,
                "X": 0.24445968866348267
            },
            {
                "Y": 0.10270800441503525,
                "X": 0.24445968866348267
            },
            {
                "Y": 0.10270800441503525,
                "X": 0.2271782010793686
```

```
            }
        ]
    },
    "BlockType": "KEY_VALUE_SET",
    "EntityTypes": [
        "VALUE"
    ],
    "Id": "24aaac7f-fcce-49c7-a4f0-3688b05586d4"
},
}
```

Le JSON suivant est le bloc SELECTION_ELEMENT. Pour`SelectionStatus`indique que la case est cochée.

```
{
    "Geometry": {
        "BoundingBox": {
            "Width": 0.020316146314144135,
            "Top": 0.07575977593660355,
            "Left": 0.22590067982673645,
            "Height": 0.027631107717752457
        },
        "Polygon": [
            {
                "Y": 0.07575977593660355,
                "X": 0.22590067982673645
            },
            {
                "Y": 0.07575977593660355,
                "X": 0.2462168186903
            },
            {
                "Y": 0.1033908873796463,
                "X": 0.2462168186903
            },
            {
                "Y": 0.1033908873796463,
                "X": 0.22590067982673645
            }
        ]
    },
    "BlockType": "SELECTION_ELEMENT",
    "SelectionStatus": "SELECTED",
```

```
    "Confidence": 74.14942932128906,
    "Id": "f2f5e8cd-e73a-4e99-a095-053acd3b6bfb"
}
```

Cellules de

Amazon Textract peut détecter des éléments de sélection à l'intérieur d'une cellule de tableau. Par exemple, les cellules du tableau suivant comportent des cases à cocher.

	D'accord	Neutral	En désaccord
Bon service	☑	☐	☐
Facile à utiliser	☐	☑	☐
Prix équitable	☑	☐	☐

UNCELLpeut contenir un enfantSELECTION_ELEMENTobjets pour éléments de sélection, ainsi que pour enfantsWORDblocs pour le texte détecté.

Pour plus d'informations sur les tableaux, consultezTables.

La TABLEBlockpour le tableau précédent ressemble à ceci.

```
{
    "Geometry": {.....},
    "Relationships": [
        {
            "Type": "CHILD",
            "Ids": [
                "652c09eb-8945-473d-b1be-fa03ac055928",
                "37efc5cc-946d-42cd-aa04-e68e5ed4741d",
                "4a44940a-435a-4c5c-8a6a-7fea341fa295",
                "2de20014-9a3b-4e26-b453-0de755144b1a",
                "8ed78aeb-5c9a-4980-b669-9e08b28671d2",
                "1f8e1c68-2c97-47b2-847c-a19619c02ca9",
                "9927e1d1-6018-4960-ac17-aadb0a94f4d9",
                "68f0ed8b-a887-42a5-b618-f68b494a6034",
                "fcba16e0-6bd7-4ea5-b86e-36e8330b68ea",
```

```
                "2250357c-ae34-4ed9-86da-45dac5a5e903",
                "c63ad40d-5a14-4646-a8df-2d4304213dbc",   // Cell
                "2b8417dc-e65f-4fcd-aa0f-61a23f1e8cb0",
                "26c62932-72f0-4dc2-9893-1ae27829c060",
                "27f291cc-abf4-4c23-aa24-676abe99cb1e",
                "7e5ce028-1bcd-4d9f-ad42-15ac181c5b47",
                "bf32e3d2-efa2-4fc1-b09b-ab9cc52ff734"
            ]
        }
    ],
    "BlockType": "TABLE",
    "Confidence": 99.99993896484375,
    "Id": "f66eac36-2e74-406e-8032-14d1c14e0b86"
}
```

CellBLOCKobjet (Id c63ad40d-5a14-4646-a8df-2d4304213dbc) pour la cellule contenant la case à cocherBon serviceOn dirait ce qui suit. Il comprend un enfant`Block`(Id = 26d122fd-c5f4-4b53-92c4-0ae92730ee1e) c'est le`SELECTION_ELEMENT` `Block`pour la case à cocher.

```
{
    "Geometry": {.....},
    "Relationships": [
        {
            "Type": "CHILD",
            "Ids": [
                "26d122fd-c5f4-4b53-92c4-0ae92730ee1e"  // Selection Element
            ]
        }
    ],
    "Confidence": 79.741689682006836,
    "RowSpan": 1,
    "RowIndex": 3,
    "ColumnIndex": 3,
    "ColumnSpan": 1,
    "BlockType": "CELL",
    "Id": "c63ad40d-5a14-4646-a8df-2d4304213dbc"
}
```

Le SELECTION_ELEMENT`Block`objet de la case à cocher est le suivant. Pour`SelectionStatus`indique que la case est cochée.

```
{
```

```
    "Geometry": {.......},
    "BlockType": "SELECTION_ELEMENT",
    "SelectionStatus": "SELECTED",
    "Confidence": 88.79517364501953,
    "Id": "26d122fd-c5f4-4b53-92c4-0ae92730ee1e"
}
```

Objets de réponse de facture et de réception

Lorsque vous soumettez une facture ou un reçu à l'API AnalyzeExpense, il renvoie une série d'objets ExpenseDocuments. Chaque document ExpenseDocument est séparé en`LineItemGroups`et`SummaryFields`. La plupart des factures et des reçus contiennent des informations telles que le nom du fournisseur, le numéro de réception, la date de réception ou le montant total. AnalyzeExpense renvoie ces informations sous`SummaryFields`. Les reçus et les factures contiennent également des informations sur les articles achetés. L'API AnalyzeExpense renvoie ces informations sous`LineItemGroups`. Le`ExpenseIndex`identifie de manière unique la dépense et associe le champ approprié`SummaryFields`et`LineItemGroups`détectés dans cette dépense.

Le niveau de données le plus granulaire de la réponse AnalyzeExpense consiste en :`Type`,`ValueDetection`, et`LabelDetection`(Facultatif). Les entités individuelles sont les suivantes :

- Type: désigne le type d'information détecté à un niveau élevé.
- Détection d'étiquettes: fait référence à l'étiquette d'une valeur associée dans le texte du document.`LabelDetection`est facultatif et n'est renvoyé que si l'étiquette est écrite.
- Détection de valeur: fait référence à la valeur de l'étiquette ou du type renvoyé.

L'API AnalyzeExpense détecte également`ITEM`,`QUANTITY`, et`PRICE`dans les éléments de ligne sous forme de champs normalisés. S'il y a un autre texte dans une ligne sur l'image de réception, tel que le SKU ou la description détaillée, il sera inclus dans le JSON comme suit :`EXPENSE_ROW`comme illustré dans l'exemple ci-dessous :

```
            {
                                "Type": {
                                    "Text": "EXPENSE_ROW",
                                    "Confidence": 99.95216369628906
```

```
                                    },
                                    "ValueDetection": {
                                        "Text": "Banana 5 $2.5",
                                        "Geometry": {
                                          …
                                        },
                                        "Confidence": 98.11214447021484
                                    }
```

L'exemple ci-dessus montre comment l'API AnalyzeExpense renvoie la ligne entière d'un reçu contenant des informations de ligne sur 5 bananes vendues pour 2,5$.

Type

Voici un exemple du type standard ou normalisé de la paire clé-valeur :

```
            {
                 "PageNumber": 1,
                 "Type": {
                     "Text": "VENDOR_NAME",
                     "Confidence": 70.0
                 },
                 "ValueDetection": {
                     "Geometry": { ... },
                     "Text": "AMAZON",
                     "Confidence": 87.89806365966797
                 }
             }
```

Le « Nom du vendeur » ne figurait pas explicitement sur le reçu. Toutefois, l'API Analyze Expense a reconnu le document comme un reçu et a classé la valeur « AMAZON » comme Type`VENDOR_NAME`.

Détection d'étiquettes

Voici un exemple de texte tel qu'il est affiché sur une page de document client :

```
            {
```

```
            "PageNumber": 1,
            "Type": {
                "Text": "OTHER",
                "Confidence": 70.0
            },
            "LabelDetection": {
                "Geometry": { ... },
                "Text": "CASHIER",
                "Confidence": 88.19171142578125
            },
            "ValueDetection": {
                "Geometry": { ... },
                "Text": "Mina",
                "Confidence": 87.89806365966797
            }
        }
```

L'exemple de document contenait « CAISSIER Mina ». L'API Analyze Expense a extrait la valeur telle quelle et la renvoie sous`LabelDetection`. Pour les valeurs implicites telles que « Nom du vendeur », où la « clé » n'est pas explicitement affichée dans le reçu,`LabelDetection`ne sera pas inclus dans l'élément AnalyzeExpense. Dans ce cas, l'API AnalyzeExpense ne renvoie pas`LabelDetection`.

Détection de valeur

L'exemple suivant illustre la « valeur » de la paire clé-valeur.

```
        {
            "PageNumber": 1,
            "Type": {
                "Text": "OTHER",
                "Confidence": 70.0
            },
            "LabelDetection": {
                "Geometry": { ... },
                "Text": "CASHIER",
                "Confidence": 88.19171142578125
            },
            "ValueDetection": {
                "Geometry": { ... },
                "Text": "Mina",
```

```
                    "Confidence": 87.89806365966797
                }
            }
```

Dans cet exemple, le document contenait « CAISSIER Mina ». L'API AnalyzeExpense a détecté la valeur Caissier comme Mina et l'a renvoyée sous`ValueDetection`.

Objets de réponse de documentation d'identité

Lorsque vous soumettez un document d'identité à l'API AnalyzeID, il renvoie une série de`IdentityDocumentField`objets. Chacun de ces objets contient`Type`, et`Value`.`Type`enregistre le champ normalisé détecté par Amazon Textract, et`Value`enregistre le texte associé au champ normalisé.

Vous trouverez ci-dessous un exemple d'`IdentityDocumentField`, raccourci pour plus de brièveté.

```
{
    "DocumentMetadata": {
        "Pages": 1
    },
    "IdentityDocumentFields": [
        {
            "Type": {
                "Text": "first name"
            },
            "ValueDetection": {
                "Text": "jennifer",
                "Confidence": 99.99908447265625
            }
        },
        {
            "Type": {
                "Text": "last name"
            },
            "ValueDetection": {
                "Text": "sample",
                "Confidence": 99.99758911132812
            }
        },
```

Voici deux exemples d'IdentityDocumentFields découpés d'une réponse plus longue. Il existe une séparation entre le type détecté et la valeur de ce type. Ici, c'est respectivement le prénom et le nom de famille. Cette structure se répète avec toutes les informations contenues. Si un type n'est pas reconnu comme champ normalisé, il sera répertorié comme « autre ».

Voici une liste de champs normalisés pour les permis de conduire :

- prénom
- nom
- deuxième prénom
- suffixe
- adresse de ville
- code postal dans l'adresse
- adresse d'état
- comté
- numéro de document
- Date d'expiration
- Date de naissance
- nom de l'état
- Date d'émission
- class
- restrictions
- avenants
- type d'ID
- vétéran
- adresse

Voici une liste de champs normalisés pour les passeports américains :

- prénom
- nom

- deuxième prénom
- numéro de document
- Date d'expiration
- Date de naissance
- lieu de naissance
- Date d'émission
- type d'ID

Emplacement de l'article sur une page de document

Les opérations Amazon Textract renvoient l'emplacement et la géométrie des éléments présents sur une page de document.DetectDocumentTextetGetDocumentTextDetectionrenvoie l'emplacement et la géométrie des lignes et des mots, tandis queAnalyzeDocumentetGetDocumentAnalysisrenvoie l'emplacement et la géométrie des paires clé-valeur, des tableaux, des cellules et des éléments de sélection.

Pour déterminer où se trouve un élément sur une page de document, utilisez le cadre de délimitation (Geometry) informations renvoyées par l'opération Amazon Textract dans unBlockobjet. Le`Geometry`Objets contient deux types d'informations d'emplacement et de géométrie pour les éléments détectés :

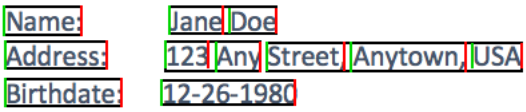
- Un axe alignéBoundingBoxObjets contenant la coordonnée supérieure gauche ainsi que la largeur et la hauteur de l'élément.
- Objet polygone qui décrit le contour de l'élément, spécifié sous la forme d'un tableau dePointobjets contenant`X`(axe horizontal) et`Y`(axe vertical) les coordonnées de page de document de chaque point.

Le JSON pour un`Block`Objets ressemble à ce qui suit. Notez le`BoundingBox`et`Polygon`.

```
{
    "Geometry": {
        "BoundingBox": {
            "Width": 0.053907789289951324,
            "Top": 0.08913730084896088,
            "Left": 0.11085548996925354,
            "Height": 0.013171200640499592
```

```
        },
        "Polygon": [
            {
                "Y": 0.08985357731580734,
                "X": 0.11085548996925354
            },
            {
                "Y": 0.08913730084896088,
                "X": 0.16447919607162476
            },
            {
                "Y": 0.10159222036600113,
                "X": 0.16476328670978546
            },
            {
                "Y": 0.10230850428342819,
                "X": 0.11113958805799484
            }
        ]
    },
    "Text": "Name:",
    "TextType": "PRINTED",
    "BlockType": "WORD",
    "Confidence": 99.56285858154297,
    "Id": "c734fca6-c4c4-415c-b6c1-30f7510b72ee"
},
```


Vous pouvez utiliser les informations de géométrie pour dessiner des zones de sélection autour des éléments détectés. Pour un exemple qui utilise`BoundingBox`et`Polygon`informations permettant de dessiner des cases autour des lignes et des lignes verticales au début et à la fin de chaque mot, voirDétection du texte de document avec Amazon Textract. L'exemple de sortie est similaire à ce qui suit.



Bounding Box

Un cadre de délimitation (`BoundingBox`) possède les propriétés suivantes :

- Height — Hauteur du cadre de délimitation sous forme de ratio de la hauteur globale de la page du document.

- Left — Coordonnée X du point supérieur gauche du cadre de délimitation sous forme de ratio de la largeur globale de la page du document.
- Top — Coordonnée Y du point supérieur gauche du cadre de délimitation sous forme de ratio de la hauteur globale de la page du document.
- Width — Largeur du cadre de délimitation sous forme de ratio de la largeur globale de la page du document.

Chaque propriété BoundingBox est définie sur une valeur comprise entre 0 et 1. La valeur est un ratio de la largeur d'image globale (s'applique à`Left`et`Width`) ou hauteur (s'applique à`Height`et`Top`). Par exemple, si l'image d'entrée a une résolution de 700 x 200 pixels, et que la coordonnée supérieure gauche du cadre de délimitation est de (350,50) pixels, l'API renvoie un`Left`valeur de 0,5 (350/700) et un`Top`valeur de 0,25 (50/200).

Le schéma suivant illustre la plage d'une page de document couverte par chaque propriété BoundingBox.

Pour afficher le cadre de délimitation avec l'emplacement et la taille appropriés, vous devez multiplier les valeurs de BoundingBox par la largeur ou la hauteur de la page du document (selon la valeur que vous souhaitez) pour obtenir les valeurs en pixels. Vous utilisez les valeurs en pixels pour afficher le cadre de délimitation. Un exemple consiste à utiliser une page de document de 608 pixels de largeur x 588 pixels de hauteur, et les valeurs de cadre de sélection suivantes pour le texte analysé :

```
BoundingBox.Left: 0.3922065
BoundingBox.Top: 0.15567766
BoundingBox.Width: 0.284666
BoundingBox.Height: 0.2930403
```

L'emplacement du cadre de délimitation du texte en pixels est calculé comme suit :

```
Left coordinate = BoundingBox.Left (0.3922065) * document page width (608)
= 238

Top coordinate = BoundingBox.Top (0.15567766) * document page height (588)
= 91

Bounding box width = BoundingBox.Width (0.284666) * document page width
(608) = 173
```

`Bounding box height = BoundingBox.Height (0.2930403) * document page height (588) = 172`

Vous utilisez ces valeurs pour afficher un cadre de délimitation autour du texte analysé. Les exemples Java et Python suivants montrent comment afficher un cadre de délimitation.

Java

```
   public void ShowBoundingBox(int imageHeight, int imageWidth, BoundingBox box,
Graphics2D g2d) {

        float left = imageWidth * box.getLeft();
        float top = imageHeight * box.getTop();

        // Display bounding box.
        g2d.setColor(new Color(0, 212, 0));
        g2d.drawRect(Math.round(left / scale), Math.round(top / scale),
                Math.round((imageWidth * box.getWidth()) / scale),
Math.round((imageHeight * box.getHeight())) / scale);

   }
```

Python

Cet exemple Python prend en compte le`response`renvoyé par leDetectDocumentTextOpération d'API.

```
def process_text_detection(response):

    # Get the text blocks
    blocks = response['Blocks']
    width, height = image.size
    draw = ImageDraw.Draw(image)
    print('Detected Document Text')

    # Create image showing bounding box/polygon the detected lines/text
    for block in blocks:

        draw = ImageDraw.Draw(image)

        if block['BlockType'] == "LINE":
            box=block['Geometry']['BoundingBox']
```

```
            left = width * box['Left']
            top = height * box['Top']
            draw.rectangle([left,top, left + (width * box['Width']), top +(height *
box['Height'])],outline='black')

    # Display the image
    image.show()

    return len(blocks)
```

Polygone

Le polygone renvoyé par`AnalyzeDocument`est un tableau de[Point](#)objets. EACH`Point`possède une coordonnée X et Y pour un emplacement spécifique sur la page de document. Comme les coordonnées BoundingBox, les coordonnées surfaciques sont normalisées à la largeur et à la hauteur du document, et se situent entre 0 et 1.

Vous pouvez utiliser des points dans le tableau de polygones pour afficher un cadre de sélection à grain fin autour d'un`Block`objet. Vous calculez la position de chaque point surfacique sur la page de document à l'aide de la même technique utilisée pour`BoundingBoxes`. Multipliez la coordonnée X par la largeur de la page de document et multipliez la coordonnée Y par la hauteur de la page de document.

L'exemple suivant montre comment afficher les lignes verticales d'un polygone.

```
   public void ShowPolygonVerticals(int imageHeight, int imageWidth, List <Point>
points, Graphics2D g2d) {

       g2d.setColor(new Color(0, 212, 0));
       Object[] parry = points.toArray();
       g2d.setStroke(new BasicStroke(2));

       g2d.drawLine(Math.round(((Point) parry[0]).getX() * imageWidth),
               Math.round(((Point) parry[0]).getY() * imageHeight),
Math.round(((Point) parry[3]).getX() * imageWidth),
               Math.round(((Point) parry[3]).getY() * imageHeight));

       g2d.setColor(new Color(255, 0, 0));
       g2d.drawLine(Math.round(((Point) parry[1]).getX() * imageWidth),
               Math.round(((Point) parry[1]).getY() * imageHeight),
Math.round(((Point) parry[2]).getX() * imageWidth),
```

```
                Math.round(((Point) parry[2]).getY() * imageHeight));

    }
```

Mise en route avec Amazon Textract

Cette section inclut des rubriques vous permettant de commencer à utiliser Amazon Textract. Si vous découvrez Amazon Textract, nous vous recommandons dans un premier temps de passer en revue les concepts et la terminologie deFonctionnement d'Amazon Textract.

Vous pouvez essayer l'API à l'aide de la démonstration dans la console Amazon Textract. Pour de plus amples informations, veuillez consulterhttps://console.aws.amazon.com/textract/.

Rubriques

- Étape 1 : Configuration d'un compte AWS et création d'un utilisateur IAM
- Étape 2 : Configuration de l'AWS CLIetAWSKits SDK
- Étape 3 : Premiers pas avec l'AWS CLIetAWSAPI SDK

Étape 1 : Configuration d'un compte AWS et création d'un utilisateur IAM

Avant d'utiliser Amazon Textract pour la première fois, exécutez les tâches suivantes :

1. Inscrivez-vous à AWS.
2. Créer un utilisateur IAM.

Inscrivez-vous à AWS

Lorsque vous vous inscrivez à Amazon Web Services (AWS), votre compte AWS est automatiquement inscrit à tous les services publiés dans AWS. Seuls les services que vous utilisez vous sont facturés.

Avec Amazon Textract, vous ne payez que les ressources que vous utilisez. Pour de plus amples informations sur les taux d'utilisation d'Amazon Textract, consultezTarification Amazon Textract. Si vous êtes un nouveau client AWS, vous pouvez démarrer gratuitement avec Amazon Textract. Pour plus d'informations, consultez la page Niveau d'offre gratuite d'AWS.

Si vous possédez déjà un compte AWS, passez à l'étape suivante. Si ce vous ne disposez pas d'un compte AWS, effectuez les étapes de la procédure suivante pour en créer un.

Pour créer un compte AWS

1. Ouvrez https://portal.aws.amazon.com/billing/signup.
2. Suivez les instructions en ligne.

   Dans le cadre de la procédure d'inscription, vous recevrez un appel téléphonique et vous saisirez un code de vérification en utilisant le clavier numérique du téléphone.

Notez votre ID de compte AWS, car vous en aurez besoin lors de la prochaine tâche.

Créer un utilisateur IAM

Les services d'AWS, comme Amazon Textract, nécessitent que vous fournissiez vos informations d'identification lors de l'accès. Cela permet au service de déterminer si vous avez les autorisations nécessaires pour accéder aux ressources détenues par ce service. La console exige votre mot de passe. Vous pouvez créer des clés d'accès pour votre compte AWS afin d'accéder à l'AWS CLI ou à l'API. Cependant, nous vous déconseillons d'accéder à AWS en utilisant les informations d'identification de votre compte AWS. A la place, nous vous recommandons de procéder comme suit :

- UtiliserGestion des identités et des accès AWS(IAM) pour créer un utilisateur IAM.
- Ajoutez l'utilisateur à un groupe IAM avec des autorisations administratives.

Vous pouvez alors accéder à AWS à l'aide d'une URL spéciale et des informations d'identification de cet utilisateur IAM.

Si vous êtes inscrit à AWS, mais que vous n'avez pas créé d'utilisateur IAM pour vous-même, vous pouvez le faire avec la console IAM. Suivez la procédure permettant de créer un utilisateur IAM dans votre compte.

Pour créer un utilisateur IAM et vous connecter à la console

1. Créez un utilisateur IAM avec des autorisations d'administrateur dans votre compte AWS. Pour obtenir des instructions, veuillez consulter Création de votre premier groupe d'utilisateurs et d'administrateurs IAM dans le Guide de l'utilisateur IAM.

2. En tant qu'utilisateur IAM, connectez-vous à AWS Management Console en utilisant une URL spéciale. Pour de plus amples informations, veuillez consulterComment les utilisateurs se connectent à votre comptedans leIAM User Guide.

> **Note**
>
> Un utilisateur IAM disposant d'autorisations d'administrateur dispose d'un accès illimité auAWSservices dans votre compte. Les exemples de code présentés dans ce guide supposent que l'un de vos utilisateurs dispose de`AmazonTextractFullAccess`Autorisations.`AmazonS3ReadOnlyAccess`est nécessaire pour les exemples qui accèdent à des documents stockés dans un compartiment Amazon S3. Selon vos exigences de sécurité, vous pouvez utiliser un groupe IAM qui se limite à ces autorisations. Pour de plus amples informations, veuillez consulterCréation de groupes IAM.

Pour plus d'informations sur IAM, consultez les ressources suivantes :

- Gestion des identités et des accès AWS (IAM)
- Prise en main
- Guide de l'utilisateur IAM

Étape suivante

Étape 2 : Configuration de l'AWS CLIetAWSKits SDK

Étape 2 : Configuration de l'AWS CLIetAWSKits SDK

Les étapes suivantes vous montrent comment installer l'AWS Command Line Interface (AWS CLI) et les kits SDK AWS utilisés dans les exemples de cette documentation.

Il existe plusieurs façons d'authentifier les appels des kits SDK AWS. Les exemples présentés dans ce guide supposent que vous utilisez un profil d'informations d'identification par défaut pour appeler les commandes de l'AWS CLI et les opérations d'API des kits SDK AWS. Vos informations d'identification par défaut fonctionneront sur tous les services. Par conséquent, si vous avez déjà configuré vos informations d'identification, vous n'avez pas besoin de le refaire. Toutefois, si vous

souhaitez créer un autre ensemble d'informations d'identification pour ce service, vous pouvez créer un profil de nom. Pour plus d'informations sur la création de profils, consultez :Consultez Profils nommés.

Pour obtenir la liste des disponiblesAWSRégions, voirRégions et points de terminaisondans leRéférence générale Amazon Web Services.

Pour configurer l'AWS CLI et les kits SDK AWS

1. Téléchargez et installez l'AWS CLI et les kits SDK AWS que vous souhaitez utiliser. Ce guide fournit des exemples pour laAWS CLI, Java et Python. Pour en savoir plus sur les autres kits SDK AWS, consultez Outils pour Amazon Web Services.

   - AWS CLI
   - AWS SDK pour Java
   - AWS SDK pour Python (Boto3)

2. Créez une clé d'accès pour l'utilisateur que vous avez créé dansCréer un utilisateur IAM.

   a. Connectez-vous à la AWS Management Console et ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.

   b. Dans le panneau de navigation, sélectionnez Users (Utilisateurs).

   c. Choisissez le nom de l'utilisateur que vous avez créé dansCréer un utilisateur IAM.

   d. Choisissez l'onglet Informations d'identification de sécurité.

   e. Choisissez Create access key (Créer une clé d'accès). Choisissez ensuite Download .csv file (Télécharger un fichier .csv) pour enregistrer l'ID de clé d'accès et la clé d'accès secrète dans un fichier CSV sur votre ordinateur. Stockez le fichier dans un emplacement sûr. Vous ne pourrez pas accéder à la clé d'accès secrète à nouveau une fois que cette boîte de dialogue se sera fermée. Après avoir téléchargé le fichier CSV, choisissezFermer.

3. Définissez des informations d'identification dans le fichier de profils d'informations d'identification AWS situé dans :

   - `~/.aws/credentials`sous Linux, macOS ou Unix.
   - `C:\Users\USERNAME\.aws\credentials`sous Windows.

   Le`.aws`n'existe pas avant la première configuration initiale de votre instance AWS. La première fois que vous configurez vos informations d'identification avec l'interface de ligne de commande,

ce dossier sera créé. Pour de plus amples informations sur les informations d'identification AWS, consultezParamètres des fichiers de configuration et d'informations d'identification.

Ce fichier doit contenir des lignes au format suivant :

```
[default]
aws_access_key_id = your_access_key_id
aws_secret_access_key = your_secret_access_key
```

Remplacez votre ID de clé d'accès et votre clé d'accès secrètevotre_access_key_idetvotre_secret_access_key.

4. Définissez la région AWS par défaut dans AWS`config`sur votre système local, situé dans :

   - `~/.aws/config`sous Linux, macOS ou Unix.
   - `C:\Users\USERNAME\.aws\config`sous Windows.

   Le`.aws`n'existe pas avant la première configuration initiale de votre instance AWS. La première fois que vous configurez vos informations d'identification avec l'interface de ligne de commande, ce dossier sera créé. Pour de plus amples informations sur les informations d'identification AWS, consultezParamètres des fichiers de configuration et d'informations d'identification.

   Ce fichier doit contenir des lignes suivantes :

   ```
   [default]
   region = your_aws_region
   ```

   Remplacez la région AWS que vous souhaitez (par exemple, « us-west-2 ») parvotre_aws_region.

   > **Note**
   >
   > Si vous ne choisissez pas une région, us-east-1 est utilisé par défaut.

Étape suivante

Étape 3 : Premiers pas avec l'AWS CLIetAWSAPI SDK

Étape 3 : Premiers pas avec l'AWS CLIetAWSAPI SDK

Une fois que vous avez configuré leAWS CLIetAWSLes kits SDK que vous souhaitez utiliser, vous pouvez créer des applications qui utilisent Amazon Textract. Les rubriques suivantes vous montrent comment démarrer à l'aide d'Amazon Textract.

- Analyse du texte du document avec Amazon Textract

Mise en forme des exemples de l'AWS CLI

Les exemples de l'AWS CLI présentés dans ce manuel sont formatés pour le système d'exploitation Linux. Pour utiliser les exemples avec Microsoft Windows, vous devez modifier le format JSON du paramètre `--document` et remplacer les barres obliques inverses (\) de saut de ligne par des accents circonflexes (^). Pour plus d'informations sur le format JSON, consultez Spécification des valeurs des paramètres pour l'interface de ligne de commande AWS.

Traitement de documents avec des opérations synchrone

Amazon Textract peut détecter et analyser du texte dans des documents d'une seule page fournis sous forme d'images au format JPEG, PNG, PDF et TIFF. Les opérations sont synchrones et renvoient les résultats en quasi temps réel. Pour plus d'informations sur les documents, consultez Objets de réponse Détection de texte et analyse de documents.

Cette section explique comment utiliser Amazon Textract pour détecter et analyser du texte dans un document d'une seule page de manière synchrone. Pour détecter et analyser du texte dans des documents multipages ou pour détecter des documents JPEG et PNG de manière asynchrone, reportez-vous à la sectionTraitement de documents avec des opérations asynchrones.

Vous pouvez utiliser les opérations synchrone Amazon Textract aux fins suivantes :

- Détection de texte : vous pouvez détecter des lignes et des mots sur une image de document d'une seule page à l'aide de la commandeDetectDocumentText. Pour plus d'informations, consultez Détection de texte.
- Analyse de texte : vous pouvez identifier les relations entre le texte détecté sur un document d'une seule page à l'aide de laAnalyzeDocument. Pour plus d'informations, consultez Analyse des documents.
- Analyse des factures et des reçus : vous pouvez identifier les relations financières entre le texte détecté sur une facture ou un reçu d'une seule page à l'aide de l'opération AnalyzeExpense. Pour de plus amples informations, veuillez consulterAnalyse des factures et des reçus
- Analyse des documents d'identité : vous pouvez analyser les documents d'identité émis par le gouvernement américain et extraire des informations ainsi que des types d'informations courants trouvés sur les documents d'identité. Pour plus d'informations, consultez la section .Analyse des documents d'identité.

Rubriques

- Appel d'opérations synchrone Amazon Textract
- Détection du texte de document avec Amazon Textract
- Analyse du texte du document avec Amazon Textract
- Analyse des factures et des reçus avec Amazon Textract
- Analyse de la documentation d'identité avec Amazon Textract

Appel d'opérations synchrone Amazon Textract

Les opérations Amazon Textract traitent les images de documents stockées sur un système de fichiers local ou les images de documents stockées dans un compartiment Amazon S3. Vous spécifiez où se trouve le document en entrée à l'aide de la commandeDocumentparamètre d'entrée. L'image du document peut être au format PNG, JPEG, PDF ou TIFF. Les résultats des opérations synchrone sont renvoyés immédiatement et ne sont pas stockés pour récupération.

Pour un exemple complet, consultezDétection du texte de document avec Amazon Textract.

Requête

La section suivante décrit comment les demandes fonctionnent dans Amazon Textract.

Documents transmis sous forme d'octets d'image

Vous pouvez transmettre une image de document à une opération Amazon Textract en transmettant l'image sous la forme d'un tableau d'octets codé en base64. Par exemple, une image de document chargée à partir d'un système de fichiers local. Votre code n'a peut-être pas besoin d'encoder des octets de fichier de document si vous utilisez unAWSSDK pour appeler les opérations de l'API Amazon Textract.

Les octets de l'image sont spécifiés dans le`Bytes`du domaine`Document`paramètre d'entrée. L'exemple suivant illustre le JSON en entrée pour une opération Amazon Textract qui transmet les octets d'image dans la zone`Bytes`paramètre d'entrée.

```
{
    "Document": {
        "Bytes": "/9j/4AAQSk....."
    }
}
```

> **Note**
>
> Si vous utilisez leAWS CLI, vous ne pouvez pas transmettre d'octets d'image aux opérations Amazon Textract. Vous devez plutôt référencer une image stockée dans un compartiment Amazon S3.

Le code Java suivant montre comment charger une image à partir d'un système de fichiers local et comment appeler une opération Amazon Textract.

```
String document="input.png";

ByteBuffer imageBytes;
try (InputStream inputStream = new FileInputStream(new File(document))) {
    imageBytes = ByteBuffer.wrap(IOUtils.toByteArray(inputStream));
}
AmazonTextract client = AmazonTextractClientBuilder.defaultClient();

DetectDocumentTextRequest request = new DetectDocumentTextRequest()
        .withDocument(new Document()
                .withBytes(imageBytes));


DetectDocumentTextResult result = client.detectDocumentText(request);
```

Documents stockés dans un compartiment Amazon S3

Amazon Textract peut analyser les images de documents stockées dans un compartiment Amazon S3. Vous spécifiez le compartiment et le nom de fichier à l'aide deS3Objectdu domaine`Document`paramètre d'entrée. L'exemple suivant illustre le JSON en entrée pour une opération Amazon Textract qui traite un document stocké dans un compartiment Amazon S3.

```
{
    "Document": {
        "S3Object": {
            "Bucket": "bucket",
            "Name": "input.png"
        }
    }
}
```

L'exemple suivant montre comment appeler une opération Amazon Textract à l'aide d'une image stockée dans un compartiment Amazon S3.

```
String document="input.png";
String bucket="bucket";

AmazonTextract client = AmazonTextractClientBuilder.defaultClient();
```

```
DetectDocumentTextRequest request = new DetectDocumentTextRequest()
        .withDocument(new Document()
                .withS3Object(new S3Object()
                        .withName(document)
                        .withBucket(bucket)));

DetectDocumentTextResult result = client.detectDocumentText(request);
```

Réponse

L'exemple suivant est la réponse JSON d'un appel à`DetectDocumentText`. Pour plus d'informations, consultez Détection de texte.

```
{
{
  "DocumentMetadata": {
    "Pages": 1
  },
  "Blocks": [
    {
      "BlockType": "PAGE",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.9995205998420715,
          "Height": 1.0,
          "Left": 0.0,
          "Top": 0.0
        },
        "Polygon": [
          {
            "X": 0.0,
            "Y": 0.0
          },
          {
            "X": 0.9995205998420715,
            "Y": 2.297314024515845E-16
          },
          {
            "X": 0.9995205998420715,
            "Y": 1.0
          },
          {
            "X": 0.0,
```

```
            "Y": 1.0
          }
        ]
      },
      "Id": "ca4b9171-7109-4adb-a811-e09bbe4834dd",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "26085884-d005-4144-b4c2-4d83dc50739b",
            "ee9d01bc-d91c-401d-8c0a-eec76f5f7862",
            "404bb3d3-d7ab-4008-a195-5dec87a08664",
            "8ae1b4ba-67c1-4486-bd20-54f461886ce9",
            "47aab5ab-be2c-4c73-97c7-d0a45454e843",
            "dd06bb49-6a56-4ea7-beec-a2aa09835c3c",
            "8837153d-81b8-4031-a49f-83a3d81803c2",
            "5dae3b74-9e95-4b62-99b7-93b88fe70648",
            "4508da80-64d8-42a8-8846-cfafe6eab10c",
            "e87be7a9-5519-42e1-b18e-ae10e2d3ed13",
            "f04bb223-d075-41c3-b328-7354611c826b",
            "a234f0e8-67de-46f4-a7c7-0bbe8d5159ce",
            "61b20e27-ff8a-450a-a8b1-bc0259f82fd6",
            "445f4fdd-c77b-4a7b-a2fc-6ca07cfe9ed7",
            "359f3870-7183-43f5-b638-970f5cefe4d5",
            "b9deea0a-244c-4d54-b774-cf03fbaaa8b1",
            "e2a43881-f620-44f2-b067-500ce7dc8d4d",
            "41756974-64ef-432d-b4b2-34702505975a",
            "93d96d32-8b4a-4a98-9578-8b4df4f227a6",
            "bc907357-63d6-43c0-ab87-80d7e76d377e",
            "2d727ca7-3acb-4bb9-a564-5885c90e9325",
            "f32a5989-cbfb-41e6-b0fc-ce1c77c014bd",
            "e0ba06d0-dbb6-4962-8047-8cac3adfe45a",
            "b6ed204d-ae01-4b75-bb91-c85d4147a37e",
            "ac4b9ee0-c9b2-4239-a741-5753e5282033",
            "ebc18885-48d7-45b8-90e3-d172b4357802",
            "babf6360-789e-49c1-9c78-0784acc14a0c"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.93761444091797,
      "Text": "Employment Application",
```

```
      "Geometry": {
        "BoundingBox": {
          "Width": 0.3391372561454773,
          "Height": 0.06906412541866302,
          "Left": 0.29548385739326477,
          "Top": 0.027493247762322426
        },
        "Polygon": [
          {
            "X": 0.29548385739326477,
            "Y": 0.027493247762322426
          },
          {
            "X": 0.6346210837364197,
            "Y": 0.027493247762322426
          },
          {
            "X": 0.6346210837364197,
            "Y": 0.0965573713183403
          },
          {
            "X": 0.29548385739326477,
            "Y": 0.0965573713183403
          }
        ]
      },
      "Id": "26085884-d005-4144-b4c2-4d83dc50739b",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "ed48dacc-d089-498f-8e93-1cee1e5f39f3",
            "ac7370f3-cbb7-4cd9-a8f9-bdcb2252caaf"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.91246795654297,
      "Text": "Application Information",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.19878505170345306,
```

```
          "Height": 0.03754019737243652,
          "Left": 0.03988289833068848,
          "Top": 0.14050349593162537
        },
        "Polygon": [
          {
            "X": 0.03988289833068848,
            "Y": 0.14050349593162537
          },
          {
            "X": 0.23866795003414154,
            "Y": 0.14050349593162537
          },
          {
            "X": 0.23866795003414154,
            "Y": 0.1780436933040619
          },
          {
            "X": 0.03988289833068848,
            "Y": 0.1780436933040619
          }
        ]
      },
      "Id": "ee9d01bc-d91c-401d-8c0a-eec76f5f7862",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "efe3fc6d-becb-4520-80ee-49a329386aee",
            "c2260852-6cfd-4a71-9fc6-62b2f9b02355"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.88693237304688,
      "Text": "Full Name: Jane Doe",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.16733919084072113,
          "Height": 0.031106337904930115,
          "Left": 0.03899926319718361,
          "Top": 0.21361036598682404
```

```
        },
        "Polygon": [
          {
            "X": 0.03899926319718361,
            "Y": 0.21361036598682404
          },
          {
            "X": 0.20633845031261444,
            "Y": 0.21361036598682404
          },
          {
            "X": 0.20633845031261444,
            "Y": 0.24471670389175415
          },
          {
            "X": 0.03899926319718361,
            "Y": 0.24471670389175415
          }
        ]
      },
      "Id": "404bb3d3-d7ab-4008-a195-5dec87a08664",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "e94eb587-9545-4215-b0fc-8e8cb1172958",
            "090aeba5-8428-4b7a-a54b-7a95a774120e",
            "64ff0abb-736b-4a6b-aa8d-ad2c0086ae1d",
            "565ffc30-89d6-4295-b8c6-d22b4ed76584"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.9206314086914,
      "Text": "Phone Number: 555-0100",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.3115004599094391,
          "Height": 0.047169625759124756,
          "Left": 0.03604753687977791,
          "Top": 0.2812676727771759
        },
```

```
        "Polygon": [
          {
            "X": 0.03604753687977791,
            "Y": 0.2812676727771759
          },
          {
            "X": 0.3475480079650879,
            "Y": 0.2812676727771759
          },
          {
            "X": 0.3475480079650879,
            "Y": 0.32843729853630066
          },
          {
            "X": 0.03604753687977791,
            "Y": 0.32843729853630066
          }
        ]
      },
      "Id": "8ae1b4ba-67c1-4486-bd20-54f461886ce9",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "d782f847-225b-4a1b-b52d-f252f8221b1f",
            "fa69c5cd-c80d-4fac-81df-569edae8d259",
            "d4bbc0f1-ae02-41cf-a26f-8a1e899968cc"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.48902893066406,
      "Text": "Home Address: 123 Any Street, Any Town. USA",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.7431139945983887,
          "Height": 0.09577702730894089,
          "Left": 0.03359385207295418,
          "Top": 0.3258342146873474
        },
        "Polygon": [
          {
```

```
            "X": 0.03359385207295418,
            "Y": 0.3258342146873474
          },
          {
            "X": 0.7767078280448914,
            "Y": 0.3258342146873474
          },
          {
            "X": 0.7767078280448914,
            "Y": 0.4216112196445465
          },
          {
            "X": 0.03359385207295418,
            "Y": 0.4216112196445465
          }
        ]
      },
      "Id": "47aab5ab-be2c-4c73-97c7-d0a45454e843",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "acfbed90-4a00-42c6-8a90-d0a0756eea36",
            "046c8a40-bb0e-4718-9c71-954d3630e1dd",
            "82b838bc-4591-4287-8dea-60c94a4925e4",
            "5cdcde7a-f5a6-4231-a941-b6396e42e7ba",
            "beafd497-185f-487e-b070-db4df5803e94",
            "ef1b77fb-8ba6-41fe-ba53-dce039af22ed",
            "7b555310-e7f8-4cd2-bb3d-dcec37f3d90e",
            "b479c24d-448d-40ef-9ed5-36a6ef08e5c7"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.89382934570312,
      "Text": "Mailing Address: same as above",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.26575741171836853,
          "Height": 0.039571404457092285,
          "Left": 0.03068041242659092,
          "Top": 0.43351811170578003
```

```
        },
        "Polygon": [
          {
            "X": 0.03068041242659092,
            "Y": 0.43351811170578003
          },
          {
            "X": 0.2964377999305725,
            "Y": 0.43351811170578003
          },
          {
            "X": 0.2964377999305725,
            "Y": 0.4730895161628723
          },
          {
            "X": 0.03068041242659092,
            "Y": 0.4730895161628723
          }
        ]
      },
      "Id": "dd06bb49-6a56-4ea7-beec-a2aa09835c3c",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "d7261cdc-6ac5-4711-903c-4598fe94952d",
            "287f80c3-6db2-4dd7-90ec-5f017c80aa31",
            "ce31c3ad-b51e-4068-be64-5fc9794bc1bc",
            "e96eb92c-6774-4d6f-8f4a-68a7618d4c66",
            "88b85c05-427a-4d4f-8cc4-3667234e8364"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 94.67343139648438,
      "Text": "Previous Employment History",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.3309842050075531,
          "Height": 0.051920413970947266,
          "Left": 0.3194798231124878,
          "Top": 0.5172380208969116
```

```
        },
        "Polygon": [
          {
            "X": 0.3194798231124878,
            "Y": 0.5172380208969116
          },
          {
            "X": 0.6504639983177185,
            "Y": 0.5172380208969116
          },
          {
            "X": 0.6504639983177185,
            "Y": 0.5691584348678589
          },
          {
            "X": 0.3194798231124878,
            "Y": 0.5691584348678589
          }
        ]
      },
      "Id": "8837153d-81b8-4031-a49f-83a3d81803c2",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "8b324501-bf38-4ce9-9777-6514b7ade760",
            "b0cea99a-5045-464d-ac8a-a63ab0470995",
            "b92a6ee5-ca59-44dc-9c47-534c133b11e7"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.66949462890625,
      "Text": "Start Date",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.08310240507125854,
          "Height": 0.030944595113396645,
          "Left": 0.034429505467414856,
          "Top": 0.6123942136764526
        },
        "Polygon": [
```

```
          {
            "X": 0.034429505467414856,
            "Y": 0.6123942136764526
          },
          {
            "X": 0.1175319030880928,
            "Y": 0.6123942136764526
          },
          {
            "X": 0.1175319030880928,
            "Y": 0.6433387994766235
          },
          {
            "X": 0.034429505467414856,
            "Y": 0.6433387994766235
          }
        ]
      },
      "Id": "5dae3b74-9e95-4b62-99b7-93b88fe70648",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "ffe8b8e0-df59-4ac5-9aba-6b54b7c51b45",
            "91e582cd-9871-4e9c-93cc-848baa426338"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.86717224121094,
      "Text": "End Date",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.07581500709056854,
          "Height": 0.03223184868693352,
          "Left": 0.14846202731132507,
          "Top": 0.6120467782020569
        },
        "Polygon": [
          {
            "X": 0.14846202731132507,
            "Y": 0.6120467782020569
```

```
          },
          {
            "X": 0.22427703440189362,
            "Y": 0.6120467782020569
          },
          {
            "X": 0.22427703440189362,
            "Y": 0.6442786455154419
          },
          {
            "X": 0.14846202731132507,
            "Y": 0.6442786455154419
          }
        ]
      },
      "Id": "4508da80-64d8-42a8-8846-cfafe6eab10c",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "7c97b56b-699f-49b0-93f4-98e6d90b107c",
            "7af04e27-0c15-447e-a569-b30edb99a133"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.9539794921875,
      "Text": "Employer Name",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.1347292959690094,
          "Height": 0.0392492413520813,
          "Left": 0.2647075653076172,
          "Top": 0.6140711903572083
        },
        "Polygon": [
          {
            "X": 0.2647075653076172,
            "Y": 0.6140711903572083
          },
          {
            "X": 0.3994368314743042,
```

```
            "Y": 0.6140711903572083
          },
          {
            "X": 0.3994368314743042,
            "Y": 0.6533204317092896
          },
          {
            "X": 0.2647075653076172,
            "Y": 0.6533204317092896
          }
        ]
      },
      "Id": "e87be7a9-5519-42e1-b18e-ae10e2d3ed13",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "a9bfeb55-75cd-47cd-b953-728e602a3564",
            "9f0f9c06-d02c-4b07-bb39-7ade70be2c1b"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.35584259033203,
      "Text": "Position Held",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.11393272876739502,
          "Height": 0.03415105864405632,
          "Left": 0.49973347783088684,
          "Top": 0.614840030670166
        },
        "Polygon": [
          {
            "X": 0.49973347783088684,
            "Y": 0.614840030670166
          },
          {
            "X": 0.6136661767959595,
            "Y": 0.614840030670166
          },
          {
```

```
            "X": 0.6136661767959595,
            "Y": 0.6489911079406738
          },
          {
            "X": 0.49973347783088684,
            "Y": 0.6489911079406738
          }
        ]
      },
      "Id": "f04bb223-d075-41c3-b328-7354611c826b",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "6d5edf02-845c-40e0-9514-e56d0d652ae0",
            "3297ab59-b237-45fb-ae60-a108f0c95ac2"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.9817886352539,
      "Text": "Reason for leaving",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.16511960327625275,
          "Height": 0.04062700271606445,
          "Left": 0.7430596351623535,
          "Top": 0.6116235852241516
        },
        "Polygon": [
          {
            "X": 0.7430596351623535,
            "Y": 0.6116235852241516
          },
          {
            "X": 0.9081792235374451,
            "Y": 0.6116235852241516
          },
          {
            "X": 0.9081792235374451,
            "Y": 0.6522505879402161
          },
```

```
          {
            "X": 0.7430596351623535,
            "Y": 0.6522505879402161
          }
        ]
      },
      "Id": "a234f0e8-67de-46f4-a7c7-0bbe8d5159ce",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "f4b8cf26-d2da-4a76-8345-69562de3cc11",
            "386d4a63-1194-4c0e-a18d-4d074a0b1f93",
            "a8622541-1896-4d54-8d10-7da2c800ec5c"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.77413177490234,
      "Text": "1/15/2009",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.08799663186073303,
          "Height": 0.03832906484603882,
          "Left": 0.03175082430243492,
          "Top": 0.691371738910675
        },
        "Polygon": [
          {
            "X": 0.03175082430243492,
            "Y": 0.691371738910675
          },
          {
            "X": 0.11974745243787766,
            "Y": 0.691371738910675
          },
          {
            "X": 0.11974745243787766,
            "Y": 0.7297008037567139
          },
          {
            "X": 0.03175082430243492,
```

```
            "Y": 0.7297008037567139
          }
        ]
      },
      "Id": "61b20e27-ff8a-450a-a8b1-bc0259f82fd6",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "da7a6482-0964-49a4-bc7d-56942ff3b4e1"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.72286224365234,
      "Text": "6/30/2011",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.08843101561069489,
          "Height": 0.03991425037384033,
          "Left": 0.14642837643623352,
          "Top": 0.6919752955436707
        },
        "Polygon": [
          {
            "X": 0.14642837643623352,
            "Y": 0.6919752955436707
          },
          {
            "X": 0.2348593920469284,
            "Y": 0.6919752955436707
          },
          {
            "X": 0.2348593920469284,
            "Y": 0.731889545917511
          },
          {
            "X": 0.14642837643623352,
            "Y": 0.731889545917511
          }
        ]
      },
```

```
      "Id": "445f4fdd-c77b-4a7b-a2fc-6ca07cfe9ed7",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "5a8da66a-ecce-4ee9-a765-a46d6cdc6cde"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.86936950683594,
      "Text": "Any Company",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.11800950765609741,
          "Height": 0.03943679481744766,
          "Left": 0.2626699209213257,
          "Top": 0.6972727179527283
        },
        "Polygon": [
          {
            "X": 0.2626699209213257,
            "Y": 0.6972727179527283
          },
          {
            "X": 0.3806794285774231,
            "Y": 0.6972727179527283
          },
          {
            "X": 0.3806794285774231,
            "Y": 0.736709475517273
          },
          {
            "X": 0.2626699209213257,
            "Y": 0.736709475517273
          }
        ]
      },
      "Id": "359f3870-7183-43f5-b638-970f5cefe4d5",
      "Relationships": [
        {
          "Type": "CHILD",
```

```
          "Ids": [
            "77749c2b-aa7f-450e-8dd2-62bcaf253ba2",
            "713bad19-158d-4e3e-b01f-f5707ddb04e5"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.582275390625,
      "Text": "Assistant baker",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.13280922174453735,
          "Height": 0.032666124403476715,
          "Left": 0.49814170598983765,
          "Top": 0.699238657951355
        },
        "Polygon": [
          {
            "X": 0.49814170598983765,
            "Y": 0.699238657951355
          },
          {
            "X": 0.630950927734375,
            "Y": 0.699238657951355
          },
          {
            "X": 0.630950927734375,
            "Y": 0.7319048047065735
          },
          {
            "X": 0.49814170598983765,
            "Y": 0.7319048047065735
          }
        ]
      },
      "Id": "b9deea0a-244c-4d54-b774-cf03fbaaa8b1",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "989944f9-f684-4714-87d8-9ad9a321d65c",
            "ae82e2aa-1601-4e0c-8340-1db7ad0c9a31"
```

```
        ]
      }
    ]
  },
  {
    "BlockType": "LINE",
    "Confidence": 99.96180725097656,
    "Text": "relocated",
    "Geometry": {
      "BoundingBox": {
        "Width": 0.08668994903564453,
        "Height": 0.033302485942840576,
        "Left": 0.7426905632019043,
        "Top": 0.6974037289619446
      },
      "Polygon": [
        {
          "X": 0.7426905632019043,
          "Y": 0.6974037289619446
        },
        {
          "X": 0.8293805122375488,
          "Y": 0.6974037289619446
        },
        {
          "X": 0.8293805122375488,
          "Y": 0.7307062149047852
        },
        {
          "X": 0.7426905632019043,
          "Y": 0.7307062149047852
        }
      ]
    },
    "Id": "e2a43881-f620-44f2-b067-500ce7dc8d4d",
    "Relationships": [
      {
        "Type": "CHILD",
        "Ids": [
          "a9cf9a8c-fdaa-413e-9346-5a28a98aebdb"
        ]
      }
    ]
  },
```

```
    {
      "BlockType": "LINE",
      "Confidence": 99.98190307617188,
      "Text": "7/1/2011",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.09747002273797989,
          "Height": 0.07067441940307617,
          "Left": 0.028500309213995934,
          "Top": 0.7745237946510315
        },
        "Polygon": [
          {
            "X": 0.028500309213995934,
            "Y": 0.7745237946510315
          },
          {
            "X": 0.12597033381462097,
            "Y": 0.7745237946510315
          },
          {
            "X": 0.12597033381462097,
            "Y": 0.8451982140541077
          },
          {
            "X": 0.028500309213995934,
            "Y": 0.8451982140541077
          }
        ]
      },
      "Id": "41756974-64ef-432d-b4b2-34702505975a",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "0f711065-1872-442a-ba6d-8fababaa452a"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.98418426513672,
      "Text": "8/10/2013",
```

```
      "Geometry": {
        "BoundingBox": {
          "Width": 0.10664612054824829,
          "Height": 0.06439518928527832,
          "Left": 0.14159755408763885,
          "Top": 0.7791688442230225
        },
        "Polygon": [
          {
            "X": 0.14159755408763885,
            "Y": 0.7791688442230225
          },
          {
            "X": 0.24824367463588715,
            "Y": 0.7791688442230225
          },
          {
            "X": 0.24824367463588715,
            "Y": 0.8435640335083008
          },
          {
            "X": 0.14159755408763885,
            "Y": 0.8435640335083008
          }
        ]
      },
      "Id": "93d96d32-8b4a-4a98-9578-8b4df4f227a6",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "a92d8eef-db28-45ba-801a-5da0f589d277"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.98075866699219,
      "Text": "Example Corp.",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.2114926278591156,
          "Height": 0.058415766805410385,
```

```
          "Left": 0.26764172315597534,
          "Top": 0.794414758682251
        },
        "Polygon": [
          {
            "X": 0.26764172315597534,
            "Y": 0.794414758682251
          },
          {
            "X": 0.47913435101509094,
            "Y": 0.794414758682251
          },
          {
            "X": 0.47913435101509094,
            "Y": 0.8528305292129517
          },
          {
            "X": 0.26764172315597534,
            "Y": 0.8528305292129517
          }
        ]
      },
      "Id": "bc907357-63d6-43c0-ab87-80d7e76d377e",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "d6962efb-34ab-4ffb-9f2f-5f263e813558",
            "1876c8ea-d3e8-4c39-870e-47512b3b5080"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.91166687011719,
      "Text": "Baker",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.09931200742721558,
          "Height": 0.06008726358413696,
          "Left": 0.5098910331726074,
          "Top": 0.787897527217865
        },
```

```
        "Polygon": [
          {
            "X": 0.5098910331726074,
            "Y": 0.787897527217865
          },
          {
            "X": 0.609203040599823,
            "Y": 0.787897527217865
          },
          {
            "X": 0.609203040599823,
            "Y": 0.847984790802002
          },
          {
            "X": 0.5098910331726074,
            "Y": 0.847984790802002
          }
        ]
      },
      "Id": "2d727ca7-3acb-4bb9-a564-5885c90e9325",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "00adeaef-ed57-44eb-b8a9-503575236d62"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.93852233886719,
      "Text": "better opp.",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.18919607996940613,
          "Height": 0.06994765996932983,
          "Left": 0.7428008317947388,
          "Top": 0.7928366661071777
        },
        "Polygon": [
          {
            "X": 0.7428008317947388,
            "Y": 0.7928366661071777
```

```
          },
          {
            "X": 0.9319968819618225,
            "Y": 0.7928366661071777
          },
          {
            "X": 0.9319968819618225,
            "Y": 0.8627843260765076
          },
          {
            "X": 0.7428008317947388,
            "Y": 0.8627843260765076
          }
        ]
      },
      "Id": "f32a5989-cbfb-41e6-b0fc-ce1c77c014bd",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "c0fc9a58-7a4b-4f69-bafd-2cff32be2665",
            "bf6dc8ee-2fb3-4b6c-aee4-31e96912a2d8"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.92573547363281,
      "Text": "8/15/2013",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.10257463902235031,
          "Height": 0.05412459373474121,
          "Left": 0.027909137308597565,
          "Top": 0.8608770370483398
        },
        "Polygon": [
          {
            "X": 0.027909137308597565,
            "Y": 0.8608770370483398
          },
          {
            "X": 0.13048377633094788,
```

```
            "Y": 0.8608770370483398
          },
          {
            "X": 0.13048377633094788,
            "Y": 0.915001630783081
          },
          {
            "X": 0.027909137308597565,
            "Y": 0.915001630783081
          }
        ]
      },
      "Id": "e0ba06d0-dbb6-4962-8047-8cac3adfe45a",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "5384f860-f857-4a94-9438-9dfa20eed1c6"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.99625396728516,
      "Text": "Present",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.09982697665691376,
          "Height": 0.06888341903686523,
          "Left": 0.1420602649450302,
          "Top": 0.8511748909950256
        },
        "Polygon": [
          {
            "X": 0.1420602649450302,
            "Y": 0.8511748909950256
          },
          {
            "X": 0.24188724160194397,
            "Y": 0.8511748909950256
          },
          {
            "X": 0.24188724160194397,
```

```
            "Y": 0.9200583100318909
          },
          {
            "X": 0.1420602649450302,
            "Y": 0.9200583100318909
          }
        ]
      },
      "Id": "b6ed204d-ae01-4b75-bb91-c85d4147a37e",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "0bb96ed6-b2e6-4da4-90b3-b85561bbd89d"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.9826431274414,
      "Text": "AnyCompany",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.18611276149749756,
          "Height": 0.08581399917602539,
          "Left": 0.2615866959095001,
          "Top": 0.869536280632019
        },
        "Polygon": [
          {
            "X": 0.2615866959095001,
            "Y": 0.869536280632019
          },
          {
            "X": 0.4476994574069977,
            "Y": 0.869536280632019
          },
          {
            "X": 0.4476994574069977,
            "Y": 0.9553502798080444
          },
          {
            "X": 0.2615866959095001,
```

```
            "Y": 0.9553502798080444
          }
        ]
      },
      "Id": "ac4b9ee0-c9b2-4239-a741-5753e5282033",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "25343360-d906-440a-88b7-92eb89e95949"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.99549102783203,
      "Text": "head baker",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.1937451809644699,
          "Height": 0.056156039237976074,
          "Left": 0.49359121918678284,
          "Top": 0.8702592849731445
        },
        "Polygon": [
          {
            "X": 0.49359121918678284,
            "Y": 0.8702592849731445
          },
          {
            "X": 0.6873363852500916,
            "Y": 0.8702592849731445
          },
          {
            "X": 0.6873363852500916,
            "Y": 0.9264153242111206
          },
          {
            "X": 0.49359121918678284,
            "Y": 0.9264153242111206
          }
        ]
      },
```

```
      "Id": "ebc18885-48d7-45b8-90e3-d172b4357802",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "0ef3c194-8322-4575-94f1-82819ee57e3a",
            "d296acd9-3e9a-4985-95f8-f863614f2c46"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.98360443115234,
      "Text": "N/A, current",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.22544169425964355,
          "Height": 0.06588292121887207,
          "Left": 0.7411766648292542,
          "Top": 0.8722732067108154
        },
        "Polygon": [
          {
            "X": 0.7411766648292542,
            "Y": 0.8722732067108154
          },
          {
            "X": 0.9666183590888977,
            "Y": 0.8722732067108154
          },
          {
            "X": 0.9666183590888977,
            "Y": 0.9381561279296875
          },
          {
            "X": 0.7411766648292542,
            "Y": 0.9381561279296875
          }
        ]
      },
      "Id": "babf6360-789e-49c1-9c78-0784acc14a0c",
      "Relationships": [
        {
```

```
          "Type": "CHILD",
          "Ids": [
            "195cfb5b-ae06-4203-8520-4e4b0a73b5ce",
            "549ef3f9-3a13-4b77-bc25-fb2e0996839a"
          ]
        }
      ]
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.94815826416016,
      "Text": "Employment",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.17462396621704102,
          "Height": 0.06266549974679947,
          "Left": 0.29548385739326477,
          "Top": 0.03389188274741173
        },
        "Polygon": [
          {
            "X": 0.29548385739326477,
            "Y": 0.03389188274741173
          },
          {
            "X": 0.4701078236103058,
            "Y": 0.03389188274741173
          },
          {
            "X": 0.4701078236103058,
            "Y": 0.0965573862195015
          },
          {
            "X": 0.29548385739326477,
            "Y": 0.0965573862195015
          }
        ]
      },
      "Id": "ed48dacc-d089-498f-8e93-1cee1e5f39f3"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.92706298828125,
```

```
      "Text": "Application",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.15933875739574432,
          "Height": 0.062391020357608795,
          "Left": 0.47528234124183655,
          "Top": 0.027493247762322426
        },
        "Polygon": [
          {
            "X": 0.47528234124183655,
            "Y": 0.027493247762322426
          },
          {
            "X": 0.6346211433410645,
            "Y": 0.027493247762322426
          },
          {
            "X": 0.6346211433410645,
            "Y": 0.08988427370786667
          },
          {
            "X": 0.47528234124183655,
            "Y": 0.08988427370786667
          }
        ]
      },
      "Id": "ac7370f3-cbb7-4cd9-a8f9-bdcb2252caaf"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.9821548461914,
      "Text": "Application",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.09610454738140106,
          "Height": 0.03656719997525215,
          "Left": 0.03988289833068848,
          "Top": 0.14147649705410004
        },
        "Polygon": [
          {
```

```
            "X": 0.03988289833068848,
            "Y": 0.14147649705410004
          },
          {
            "X": 0.13598744571208954,
            "Y": 0.14147649705410004
          },
          {
            "X": 0.13598744571208954,
            "Y": 0.1780436933040619
          },
          {
            "X": 0.03988289833068848,
            "Y": 0.1780436933040619
          }
        ]
      },
      "Id": "efe3fc6d-becb-4520-80ee-49a329386aee"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.84278106689453,
      "Text": "Information",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.10029315203428268,
          "Height": 0.03209415823221207,
          "Left": 0.13837480545043945,
          "Top": 0.14050349593162537
        },
        "Polygon": [
          {
            "X": 0.13837480545043945,
            "Y": 0.14050349593162537
          },
          {
            "X": 0.23866795003414154,
            "Y": 0.14050349593162537
          },
          {
            "X": 0.23866795003414154,
            "Y": 0.17259766161441803
          },
```

```
          {
            "X": 0.13837480545043945,
            "Y": 0.17259766161441803
          }
        ]
      },
      "Id": "c2260852-6cfd-4a71-9fc6-62b2f9b02355"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.83993530273438,
      "Text": "Full",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.03039788082242012,
          "Height": 0.031106330454349518,
          "Left": 0.03899926319718361,
          "Top": 0.21361036598682404
        },
        "Polygon": [
          {
            "X": 0.03899926319718361,
            "Y": 0.21361036598682404
          },
          {
            "X": 0.06939714401960373,
            "Y": 0.21361036598682404
          },
          {
            "X": 0.06939714401960373,
            "Y": 0.24471670389175415
          },
          {
            "X": 0.03899926319718361,
            "Y": 0.24471670389175415
          }
        ]
      },
      "Id": "e94eb587-9545-4215-b0fc-8e8cb1172958"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.93611907958984,
```

```
      "Text": "Name:",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.05555811896920204,
          "Height": 0.030184319242835045,
          "Left": 0.07123806327581406,
          "Top": 0.2137702852487564
        },
        "Polygon": [
          {
            "X": 0.07123806327581406,
            "Y": 0.2137702852487564
          },
          {
            "X": 0.1267961859703064,
            "Y": 0.2137702852487564
          },
          {
            "X": 0.1267961859703064,
            "Y": 0.2439546138048172
          },
          {
            "X": 0.07123806327581406,
            "Y": 0.2439546138048172
          }
        ]
      },
      "Id": "090aeba5-8428-4b7a-a54b-7a95a774120e"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.91043853759766,
      "Text": "Jane",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.03905024006962776,
          "Height": 0.02941947989165783,
          "Left": 0.12933772802352905,
          "Top": 0.214289128780365
        },
        "Polygon": [
          {
```

```
            "X": 0.12933772802352905,
            "Y": 0.214289128780365
          },
          {
            "X": 0.16838796436786652,
            "Y": 0.214289128780365
          },
          {
            "X": 0.16838796436786652,
            "Y": 0.24370861053466797
          },
          {
            "X": 0.12933772802352905,
            "Y": 0.24370861053466797
          }
        ]
      },
      "Id": "64ff0abb-736b-4a6b-aa8d-ad2c0086ae1d"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.86123657226562,
      "Text": "Doe",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.035229459404945374,
          "Height": 0.030427640303969383,
          "Left": 0.17110899090766907,
          "Top": 0.21377210319042206
        },
        "Polygon": [
          {
            "X": 0.17110899090766907,
            "Y": 0.21377210319042206
          },
          {
            "X": 0.20633845031261444,
            "Y": 0.21377210319042206
          },
          {
            "X": 0.20633845031261444,
            "Y": 0.244199737906456
          },
```

```
          {
            "X": 0.17110899090766907,
            "Y": 0.244199737906456
          }
        ]
      },
      "Id": "565ffc30-89d6-4295-b8c6-d22b4ed76584"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.92633056640625,
      "Text": "Phone",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.052783288061618805,
          "Height": 0.03104414977133274,
          "Left": 0.03604753687977791,
          "Top": 0.28701552748680115
        },
        "Polygon": [
          {
            "X": 0.03604753687977791,
            "Y": 0.28701552748680115
          },
          {
            "X": 0.08883082121610641,
            "Y": 0.28701552748680115
          },
          {
            "X": 0.08883082121610641,
            "Y": 0.31805968284606934
          },
          {
            "X": 0.03604753687977791,
            "Y": 0.31805968284606934
          }
        ]
      },
      "Id": "d782f847-225b-4a1b-b52d-f252f8221b1f"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.86275482177734,
```

```
      "Text": "Number:",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.07424934208393097,
          "Height": 0.030300479382276535,
          "Left": 0.0915418416261673,
          "Top": 0.28639692068099976
        },
        "Polygon": [
          {
            "X": 0.0915418416261673,
            "Y": 0.28639692068099976
          },
          {
            "X": 0.16579118371009827,
            "Y": 0.28639692068099976
          },
          {
            "X": 0.16579118371009827,
            "Y": 0.3166973888874054
          },
          {
            "X": 0.0915418416261673,
            "Y": 0.3166973888874054
          }
        ]
      },
      "Id": "fa69c5cd-c80d-4fac-81df-569edae8d259"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.97282409667969,
      "Text": "555-0100",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.17021971940994263,
          "Height": 0.047169629484415054,
          "Left": 0.17732827365398407,
          "Top": 0.2812676727771759
        },
        "Polygon": [
          {
```

```
            "X": 0.17732827365398407,
            "Y": 0.2812676727771759
          },
          {
            "X": 0.3475480079650879,
            "Y": 0.2812676727771759
          },
          {
            "X": 0.3475480079650879,
            "Y": 0.32843729853630066
          },
          {
            "X": 0.17732827365398407,
            "Y": 0.32843729853630066
          }
        ]
      },
      "Id": "d4bbc0f1-ae02-41cf-a26f-8a1e899968cc"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.66238403320312,
      "Text": "Home",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.049357783049345016,
          "Height": 0.03134990110993385,
          "Left": 0.03359385207295418,
          "Top": 0.36172014474868774
        },
        "Polygon": [
          {
            "X": 0.03359385207295418,
            "Y": 0.36172014474868774
          },
          {
            "X": 0.0829516351222992,
            "Y": 0.36172014474868774
          },
          {
            "X": 0.0829516351222992,
            "Y": 0.3930700421333313
          },
```

```
        {
          "X": 0.03359385207295418,
          "Y": 0.3930700421333313
        }
      ]
    },
    "Id": "acfbed90-4a00-42c6-8a90-d0a0756eea36"
  },
  {
    "BlockType": "WORD",
    "Confidence": 99.6871109008789,
    "Text": "Address:",
    "TextType": "PRINTED",
    "Geometry": {
      "BoundingBox": {
        "Width": 0.07411003112792969,
        "Height": 0.0314042791724205,
        "Left": 0.08516156673431396,
        "Top": 0.3600046932697296
      },
      "Polygon": [
        {
          "X": 0.08516156673431396,
          "Y": 0.3600046932697296
        },
        {
          "X": 0.15927159786224365,
          "Y": 0.3600046932697296
        },
        {
          "X": 0.15927159786224365,
          "Y": 0.3914089798927307
        },
        {
          "X": 0.08516156673431396,
          "Y": 0.3914089798927307
        }
      ]
    },
    "Id": "046c8a40-bb0e-4718-9c71-954d3630e1dd"
  },
  {
    "BlockType": "WORD",
    "Confidence": 99.93781280517578,
```

```
      "Text": "123",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.05761868134140968,
          "Height": 0.05008566007018089,
          "Left": 0.1750781387090683,
          "Top": 0.35484206676483154
        },
        "Polygon": [
          {
            "X": 0.1750781387090683,
            "Y": 0.35484206676483154
          },
          {
            "X": 0.23269681632518768,
            "Y": 0.35484206676483154
          },
          {
            "X": 0.23269681632518768,
            "Y": 0.40492773056030273
          },
          {
            "X": 0.1750781387090683,
            "Y": 0.40492773056030273
          }
        ]
      },
      "Id": "82b838bc-4591-4287-8dea-60c94a4925e4"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.96530151367188,
      "Text": "Any",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.06814215332269669,
          "Height": 0.06354366987943649,
          "Left": 0.2550157308578491,
          "Top": 0.35471394658088684
        },
        "Polygon": [
          {
```

```
            "X": 0.2550157308578491,
            "Y": 0.35471394658088684
          },
          {
            "X": 0.3231579065322876,
            "Y": 0.35471394658088684
          },
          {
            "X": 0.3231579065322876,
            "Y": 0.41825762391090393
          },
          {
            "X": 0.2550157308578491,
            "Y": 0.41825762391090393
          }
        ]
      },
      "Id": "5cdcde7a-f5a6-4231-a941-b6396e42e7ba"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.87527465820312,
      "Text": "Street,",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.12156613171100616,
          "Height": 0.05449587106704712,
          "Left": 0.3357025980949402,
          "Top": 0.3550415635108948
        },
        "Polygon": [
          {
            "X": 0.3357025980949402,
            "Y": 0.3550415635108948
          },
          {
            "X": 0.45726871490478516,
            "Y": 0.3550415635108948
          },
          {
            "X": 0.45726871490478516,
            "Y": 0.4095374345779419
          },
```

```
        {
          "X": 0.3357025980949402,
          "Y": 0.4095374345779419
        }
      ]
    },
    "Id": "beafd497-185f-487e-b070-db4df5803e94"
  },
  {
    "BlockType": "WORD",
    "Confidence": 99.99514770507812,
    "Text": "Any",
    "TextType": "HANDWRITING",
    "Geometry": {
      "BoundingBox": {
        "Width": 0.07748188823461533,
        "Height": 0.07339789718389511,
        "Left": 0.47723668813705444,
        "Top": 0.3482133150100708
      },
      "Polygon": [
        {
          "X": 0.47723668813705444,
          "Y": 0.3482133150100708
        },
        {
          "X": 0.554718554019928,
          "Y": 0.3482133150100708
        },
        {
          "X": 0.554718554019928,
          "Y": 0.4216112196445465
        },
        {
          "X": 0.47723668813705444,
          "Y": 0.4216112196445465
        }
      ]
    },
    "Id": "ef1b77fb-8ba6-41fe-ba53-dce039af22ed"
  },
  {
    "BlockType": "WORD",
    "Confidence": 96.80656433105469,
```

```
      "Text": "Town.",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.11213835328817368,
          "Height": 0.057233039289712906,
          "Left": 0.5563329458236694,
          "Top": 0.3331930637359619
        },
        "Polygon": [
          {
            "X": 0.5563329458236694,
            "Y": 0.3331930637359619
          },
          {
            "X": 0.6684713363647461,
            "Y": 0.3331930637359619
          },
          {
            "X": 0.6684713363647461,
            "Y": 0.3904260993003845
          },
          {
            "X": 0.5563329458236694,
            "Y": 0.3904260993003845
          }
        ]
      },
      "Id": "7b555310-e7f8-4cd2-bb3d-dcec37f3d90e"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.98260498046875,
      "Text": "USA",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.08771833777427673,
          "Height": 0.05706935003399849,
          "Left": 0.6889894604682922,
          "Top": 0.3258342146873474
        },
        "Polygon": [
          {
```

```
            "X": 0.6889894604682922,
            "Y": 0.3258342146873474
          },
          {
            "X": 0.7767078280448914,
            "Y": 0.3258342146873474
          },
          {
            "X": 0.7767078280448914,
            "Y": 0.3829035460948944
          },
          {
            "X": 0.6889894604682922,
            "Y": 0.3829035460948944
          }
        ]
      },
      "Id": "b479c24d-448d-40ef-9ed5-36a6ef08e5c7"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.9583969116211,
      "Text": "Mailing",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.06291338801383972,
          "Height": 0.03957144916057587,
          "Left": 0.03068041242659092,
          "Top": 0.43351811170578003
        },
        "Polygon": [
          {
            "X": 0.03068041242659092,
            "Y": 0.43351811170578003
          },
          {
            "X": 0.09359379857778549,
            "Y": 0.43351811170578003
          },
          {
            "X": 0.09359379857778549,
            "Y": 0.4730895459651947
          },
```

```
        {
          "X": 0.03068041242659092,
          "Y": 0.4730895459651947
        }
      ]
    },
    "Id": "d7261cdc-6ac5-4711-903c-4598fe94952d"
  },
  {
    "BlockType": "WORD",
    "Confidence": 99.87476348876953,
    "Text": "Address:",
    "TextType": "PRINTED",
    "Geometry": {
      "BoundingBox": {
        "Width": 0.07364854216575623,
        "Height": 0.03147412836551666,
        "Left": 0.0954652726650238,
        "Top": 0.43450701236724854
      },
      "Polygon": [
        {
          "X": 0.0954652726650238,
          "Y": 0.43450701236724854
        },
        {
          "X": 0.16911381483078003,
          "Y": 0.43450701236724854
        },
        {
          "X": 0.16911381483078003,
          "Y": 0.465981125831604
        },
        {
          "X": 0.0954652726650238,
          "Y": 0.465981125831604
        }
      ]
    },
    "Id": "287f80c3-6db2-4dd7-90ec-5f017c80aa31"
  },
  {
    "BlockType": "WORD",
    "Confidence": 99.94071960449219,
```

```
      "Text": "same",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.04640670120716095,
          "Height": 0.026415130123496056,
          "Left": 0.17156922817230225,
          "Top": 0.44010937213897705
        },
        "Polygon": [
          {
            "X": 0.17156922817230225,
            "Y": 0.44010937213897705
          },
          {
            "X": 0.2179759293794632,
            "Y": 0.44010937213897705
          },
          {
            "X": 0.2179759293794632,
            "Y": 0.46652451157569885
          },
          {
            "X": 0.17156922817230225,
            "Y": 0.46652451157569885
          }
        ]
      },
      "Id": "ce31c3ad-b51e-4068-be64-5fc9794bc1bc"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.76510620117188,
      "Text": "as",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.02041218988597393,
          "Height": 0.025104399770498276,
          "Left": 0.2207803726196289,
          "Top": 0.44124215841293335
        },
        "Polygon": [
          {
```

```
            "X": 0.2207803726196289,
            "Y": 0.44124215841293335
          },
          {
            "X": 0.24119256436824799,
            "Y": 0.44124215841293335
          },
          {
            "X": 0.24119256436824799,
            "Y": 0.4663465619087219
          },
          {
            "X": 0.2207803726196289,
            "Y": 0.4663465619087219
          }
        ]
      },
      "Id": "e96eb92c-6774-4d6f-8f4a-68a7618d4c66"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.9301528930664,
      "Text": "above",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.05268359184265137,
          "Height": 0.03216424956917763,
          "Left": 0.24375422298908234,
          "Top": 0.4354657828807831
        },
        "Polygon": [
          {
            "X": 0.24375422298908234,
            "Y": 0.4354657828807831
          },
          {
            "X": 0.2964377999305725,
            "Y": 0.4354657828807831
          },
          {
            "X": 0.2964377999305725,
            "Y": 0.4676300287246704
          },
```

```
        {
          "X": 0.24375422298908234,
          "Y": 0.4676300287246704
        }
      ]
    },
    "Id": "88b85c05-427a-4d4f-8cc4-3667234e8364"
  },
  {
    "BlockType": "WORD",
    "Confidence": 85.3905029296875,
    "Text": "Previous",
    "TextType": "PRINTED",
    "Geometry": {
      "BoundingBox": {
        "Width": 0.09860499948263168,
        "Height": 0.04000622034072876,
        "Left": 0.3194798231124878,
        "Top": 0.5194430351257324
      },
      "Polygon": [
        {
          "X": 0.3194798231124878,
          "Y": 0.5194430351257324
        },
        {
          "X": 0.4180848002433777,
          "Y": 0.5194430351257324
        },
        {
          "X": 0.4180848002433777,
          "Y": 0.5594492554664612
        },
        {
          "X": 0.3194798231124878,
          "Y": 0.5594492554664612
        }
      ]
    },
    "Id": "8b324501-bf38-4ce9-9777-6514b7ade760"
  },
  {
    "BlockType": "WORD",
    "Confidence": 99.14524841308594,
```

```
      "Text": "Employment",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.14039960503578186,
          "Height": 0.04645847901701927,
          "Left": 0.4214291274547577,
          "Top": 0.5219109654426575
        },
        "Polygon": [
          {
            "X": 0.4214291274547577,
            "Y": 0.5219109654426575
          },
          {
            "X": 0.5618287324905396,
            "Y": 0.5219109654426575
          },
          {
            "X": 0.5618287324905396,
            "Y": 0.568369448184967
          },
          {
            "X": 0.4214291274547577,
            "Y": 0.568369448184967
          }
        ]
      },
      "Id": "b0cea99a-5045-464d-ac8a-a63ab0470995"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.48454284667969,
      "Text": "History",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.08361124992370605,
          "Height": 0.05192042887210846,
          "Left": 0.5668527483940125,
          "Top": 0.5172380208969116
        },
        "Polygon": [
          {
```

```
            "X": 0.5668527483940125,
            "Y": 0.5172380208969116
          },
          {
            "X": 0.6504639983177185,
            "Y": 0.5172380208969116
          },
          {
            "X": 0.6504639983177185,
            "Y": 0.5691584348678589
          },
          {
            "X": 0.5668527483940125,
            "Y": 0.5691584348678589
          }
        ]
      },
      "Id": "b92a6ee5-ca59-44dc-9c47-534c133b11e7"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.78699493408203,
      "Text": "Start",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.041341401636600494,
          "Height": 0.030926469713449478,
          "Left": 0.034429505467414856,
          "Top": 0.6124123334884644
        },
        "Polygon": [
          {
            "X": 0.034429505467414856,
            "Y": 0.6124123334884644
          },
          {
            "X": 0.07577090710401535,
            "Y": 0.6124123334884644
          },
          {
            "X": 0.07577090710401535,
            "Y": 0.6433387994766235
          },
```

```
          {
            "X": 0.034429505467414856,
            "Y": 0.6433387994766235
          }
        ]
      },
      "Id": "ffe8b8e0-df59-4ac5-9aba-6b54b7c51b45"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.55198669433594,
      "Text": "Date",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.03923053666949272,
          "Height": 0.03072454035282135,
          "Left": 0.07830137014389038,
          "Top": 0.6123942136764526
        },
        "Polygon": [
          {
            "X": 0.07830137014389038,
            "Y": 0.6123942136764526
          },
          {
            "X": 0.1175319105386734,
            "Y": 0.6123942136764526
          },
          {
            "X": 0.1175319105386734,
            "Y": 0.6431187391281128
          },
          {
            "X": 0.07830137014389038,
            "Y": 0.6431187391281128
          }
        ]
      },
      "Id": "91e582cd-9871-4e9c-93cc-848baa426338"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.8897705078125,
```

```
      "Text": "End",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.03212086856365204,
          "Height": 0.03193363919854164,
          "Left": 0.14846202731132507,
          "Top": 0.6120467782020569
        },
        "Polygon": [
          {
            "X": 0.14846202731132507,
            "Y": 0.6120467782020569
          },
          {
            "X": 0.1805828958749771,
            "Y": 0.6120467782020569
          },
          {
            "X": 0.1805828958749771,
            "Y": 0.6439804434776306
          },
          {
            "X": 0.14846202731132507,
            "Y": 0.6439804434776306
          }
        ]
      },
      "Id": "7c97b56b-699f-49b0-93f4-98e6d90b107c"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.8445816040039,
      "Text": "Date",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.03987143933773041,
          "Height": 0.03142518177628517,
          "Left": 0.1844055950641632,
          "Top": 0.612853467464447
        },
        "Polygon": [
          {
```

```
            "X": 0.1844055950641632,
            "Y": 0.612853467464447
          },
          {
            "X": 0.22427703440189362,
            "Y": 0.612853467464447
          },
          {
            "X": 0.22427703440189362,
            "Y": 0.6442786455154419
          },
          {
            "X": 0.1844055950641632,
            "Y": 0.6442786455154419
          }
        ]
      },
      "Id": "7af04e27-0c15-447e-a569-b30edb99a133"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.9652328491211,
      "Text": "Employer",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.08150768280029297,
          "Height": 0.0392492301762104,
          "Left": 0.2647075653076172,
          "Top": 0.6140711903572083
        },
        "Polygon": [
          {
            "X": 0.2647075653076172,
            "Y": 0.6140711903572083
          },
          {
            "X": 0.34621524810791016,
            "Y": 0.6140711903572083
          },
          {
            "X": 0.34621524810791016,
            "Y": 0.6533204317092896
          },
```

```
          {
            "X": 0.2647075653076172,
            "Y": 0.6533204317092896
          }
        ]
      },
      "Id": "a9bfeb55-75cd-47cd-b953-728e602a3564"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.94273376464844,
      "Text": "Name",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.05018233880400658,
          "Height": 0.03248906135559082,
          "Left": 0.34925445914268494,
          "Top": 0.6162016987800598
        },
        "Polygon": [
          {
            "X": 0.34925445914268494,
            "Y": 0.6162016987800598
          },
          {
            "X": 0.3994368016719818,
            "Y": 0.6162016987800598
          },
          {
            "X": 0.3994368016719818,
            "Y": 0.6486907601356506
          },
          {
            "X": 0.34925445914268494,
            "Y": 0.6486907601356506
          }
        ]
      },
      "Id": "9f0f9c06-d02c-4b07-bb39-7ade70be2c1b"
    },
    {
      "BlockType": "WORD",
      "Confidence": 98.85071563720703,
```

```
      "Text": "Position",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.07007700204849243,
          "Height": 0.03255689889192581,
          "Left": 0.49973347783088684,
          "Top": 0.6164342164993286
        },
        "Polygon": [
          {
            "X": 0.49973347783088684,
            "Y": 0.6164342164993286
          },
          {
            "X": 0.5698104500770569,
            "Y": 0.6164342164993286
          },
          {
            "X": 0.5698104500770569,
            "Y": 0.6489911079406738
          },
          {
            "X": 0.49973347783088684,
            "Y": 0.6489911079406738
          }
        ]
      },
      "Id": "6d5edf02-845c-40e0-9514-e56d0d652ae0"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.86096954345703,
      "Text": "Held",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.04017873853445053,
          "Height": 0.03292537108063698,
          "Left": 0.5734874606132507,
          "Top": 0.614840030670166
        },
        "Polygon": [
          {
```

```
            "X": 0.5734874606132507,
            "Y": 0.614840030670166
          },
          {
            "X": 0.6136662364006042,
            "Y": 0.614840030670166
          },
          {
            "X": 0.6136662364006042,
            "Y": 0.6477653980255127
          },
          {
            "X": 0.5734874606132507,
            "Y": 0.6477653980255127
          }
        ]
      },
      "Id": "3297ab59-b237-45fb-ae60-a108f0c95ac2"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.97740936279297,
      "Text": "Reason",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.06497219949960709,
          "Height": 0.03248770162463188,
          "Left": 0.7430596351623535,
          "Top": 0.6136704087257385
        },
        "Polygon": [
          {
            "X": 0.7430596351623535,
            "Y": 0.6136704087257385
          },
          {
            "X": 0.8080317974090576,
            "Y": 0.6136704087257385
          },
          {
            "X": 0.8080317974090576,
            "Y": 0.6461580991744995
          },
```

```
        {
          "X": 0.7430596351623535,
          "Y": 0.6461580991744995
        }
      ]
    },
    "Id": "f4b8cf26-d2da-4a76-8345-69562de3cc11"
  },
  {
    "BlockType": "WORD",
    "Confidence": 99.98371887207031,
    "Text": "for",
    "TextType": "PRINTED",
    "Geometry": {
      "BoundingBox": {
        "Width": 0.029645200818777084,
        "Height": 0.03462234139442444,
        "Left": 0.8108851909637451,
        "Top": 0.6117717623710632
      },
      "Polygon": [
        {
          "X": 0.8108851909637451,
          "Y": 0.6117717623710632
        },
        {
          "X": 0.8405303955078125,
          "Y": 0.6117717623710632
        },
        {
          "X": 0.8405303955078125,
          "Y": 0.6463940739631653
        },
        {
          "X": 0.8108851909637451,
          "Y": 0.6463940739631653
        }
      ]
    },
    "Id": "386d4a63-1194-4c0e-a18d-4d074a0b1f93"
  },
  {
    "BlockType": "WORD",
    "Confidence": 99.98424530029297,
```

```
      "Text": "leaving",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.06517849862575531,
          "Height": 0.040626998990774155,
          "Left": 0.8430007100105286,
          "Top": 0.6116235852241516
        },
        "Polygon": [
          {
            "X": 0.8430007100105286,
            "Y": 0.6116235852241516
          },
          {
            "X": 0.9081792235374451,
            "Y": 0.6116235852241516
          },
          {
            "X": 0.9081792235374451,
            "Y": 0.6522505879402161
          },
          {
            "X": 0.8430007100105286,
            "Y": 0.6522505879402161
          }
        ]
      },
      "Id": "a8622541-1896-4d54-8d10-7da2c800ec5c"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.77413177490234,
      "Text": "1/15/2009",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.08799663186073303,
          "Height": 0.03832906112074852,
          "Left": 0.03175082430243492,
          "Top": 0.691371738910675
        },
        "Polygon": [
          {
```

```
            "X": 0.03175082430243492,
            "Y": 0.691371738910675
          },
          {
            "X": 0.11974745243787766,
            "Y": 0.691371738910675
          },
          {
            "X": 0.11974745243787766,
            "Y": 0.7297008037567139
          },
          {
            "X": 0.03175082430243492,
            "Y": 0.7297008037567139
          }
        ]
      },
      "Id": "da7a6482-0964-49a4-bc7d-56942ff3b4e1"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.72286224365234,
      "Text": "6/30/2011",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.08843102306127548,
          "Height": 0.03991425037384033,
          "Left": 0.14642837643623352,
          "Top": 0.6919752955436707
        },
        "Polygon": [
          {
            "X": 0.14642837643623352,
            "Y": 0.6919752955436707
          },
          {
            "X": 0.2348593920469284,
            "Y": 0.6919752955436707
          },
          {
            "X": 0.2348593920469284,
            "Y": 0.731889545917511
          },
```

```
          {
            "X": 0.14642837643623352,
            "Y": 0.731889545917511
          }
        ]
      },
      "Id": "5a8da66a-ecce-4ee9-a765-a46d6cdc6cde"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.92295837402344,
      "Text": "Any",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.034067559987306595,
          "Height": 0.037968240678310394,
          "Left": 0.2626699209213257,
          "Top": 0.6972727179527283
        },
        "Polygon": [
          {
            "X": 0.2626699209213257,
            "Y": 0.6972727179527283
          },
          {
            "X": 0.2967374622821808,
            "Y": 0.6972727179527283
          },
          {
            "X": 0.2967374622821808,
            "Y": 0.7352409362792969
          },
          {
            "X": 0.2626699209213257,
            "Y": 0.7352409362792969
          }
        ]
      },
      "Id": "77749c2b-aa7f-450e-8dd2-62bcaf253ba2"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.81578063964844,
```

```
      "Text": "Company",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.08160992711782455,
          "Height": 0.03890080004930496,
          "Left": 0.29906952381134033,
          "Top": 0.6978086829185486
        },
        "Polygon": [
          {
            "X": 0.29906952381134033,
            "Y": 0.6978086829185486
          },
          {
            "X": 0.3806794583797455,
            "Y": 0.6978086829185486
          },
          {
            "X": 0.3806794583797455,
            "Y": 0.736709475517273
          },
          {
            "X": 0.29906952381134033,
            "Y": 0.736709475517273
          }
        ]
      },
      "Id": "713bad19-158d-4e3e-b01f-f5707ddb04e5"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.37964630126953,
      "Text": "Assistant",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.0789310410618782,
          "Height": 0.03139699995517731,
          "Left": 0.49814170598983765,
          "Top": 0.7005078196525574
        },
        "Polygon": [
          {
```

```
            "X": 0.49814170598983765,
            "Y": 0.7005078196525574
          },
          {
            "X": 0.5770727396011353,
            "Y": 0.7005078196525574
          },
          {
            "X": 0.5770727396011353,
            "Y": 0.7319048047065735
          },
          {
            "X": 0.49814170598983765,
            "Y": 0.7319048047065735
          }
        ]
      },
      "Id": "989944f9-f684-4714-87d8-9ad9a321d65c"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.784912109375,
      "Text": "baker",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.050264399498701096,
          "Height": 0.03237773850560188,
          "Left": 0.5806865096092224,
          "Top": 0.699238657951355
        },
        "Polygon": [
          {
            "X": 0.5806865096092224,
            "Y": 0.699238657951355
          },
          {
            "X": 0.630950927734375,
            "Y": 0.699238657951355
          },
          {
            "X": 0.630950927734375,
            "Y": 0.7316163778305054
          },
```

```
        {
          "X": 0.5806865096092224,
          "Y": 0.7316163778305054
        }
      ]
    },
    "Id": "ae82e2aa-1601-4e0c-8340-1db7ad0c9a31"
  },
  {
    "BlockType": "WORD",
    "Confidence": 99.96180725097656,
    "Text": "relocated",
    "TextType": "PRINTED",
    "Geometry": {
      "BoundingBox": {
        "Width": 0.08668994158506393,
        "Height": 0.03330250084400177,
        "Left": 0.7426905632019043,
        "Top": 0.6974037289619446
      },
      "Polygon": [
        {
          "X": 0.7426905632019043,
          "Y": 0.6974037289619446
        },
        {
          "X": 0.8293805122375488,
          "Y": 0.6974037289619446
        },
        {
          "X": 0.8293805122375488,
          "Y": 0.7307062149047852
        },
        {
          "X": 0.7426905632019043,
          "Y": 0.7307062149047852
        }
      ]
    },
    "Id": "a9cf9a8c-fdaa-413e-9346-5a28a98aebdb"
  },
  {
    "BlockType": "WORD",
    "Confidence": 99.98190307617188,
```

```
      "Text": "7/1/2011",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.09747002273797989,
          "Height": 0.07067439705133438,
          "Left": 0.028500309213995934,
          "Top": 0.7745237946510315
        },
        "Polygon": [
          {
            "X": 0.028500309213995934,
            "Y": 0.7745237946510315
          },
          {
            "X": 0.12597033381462097,
            "Y": 0.7745237946510315
          },
          {
            "X": 0.12597033381462097,
            "Y": 0.8451982140541077
          },
          {
            "X": 0.028500309213995934,
            "Y": 0.8451982140541077
          }
        ]
      },
      "Id": "0f711065-1872-442a-ba6d-8fababaa452a"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.98418426513672,
      "Text": "8/10/2013",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.10664612054824829,
          "Height": 0.06439515948295593,
          "Left": 0.14159755408763885,
          "Top": 0.7791688442230225
        },
        "Polygon": [
          {
```

```
            "X": 0.14159755408763885,
            "Y": 0.7791688442230225
          },
          {
            "X": 0.24824367463588715,
            "Y": 0.7791688442230225
          },
          {
            "X": 0.24824367463588715,
            "Y": 0.843563973903656
          },
          {
            "X": 0.14159755408763885,
            "Y": 0.843563973903656
          }
        ]
      },
      "Id": "a92d8eef-db28-45ba-801a-5da0f589d277"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.97722625732422,
      "Text": "Example",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.12127546221017838,
          "Height": 0.05682983994483948,
          "Left": 0.26764172315597534,
          "Top": 0.794414758682251
        },
        "Polygon": [
          {
            "X": 0.26764172315597534,
            "Y": 0.794414758682251
          },
          {
            "X": 0.3889172077178955,
            "Y": 0.794414758682251
          },
          {
            "X": 0.3889172077178955,
            "Y": 0.8512446284294128
          },
```

```
        {
          "X": 0.26764172315597534,
          "Y": 0.8512446284294128
        }
      ]
    },
    "Id": "d6962efb-34ab-4ffb-9f2f-5f263e813558"
  },
  {
    "BlockType": "WORD",
    "Confidence": 99.98429870605469,
    "Text": "Corp.",
    "TextType": "HANDWRITING",
    "Geometry": {
      "BoundingBox": {
        "Width": 0.07650306820869446,
        "Height": 0.05481306090950966,
        "Left": 0.4026312530040741,
        "Top": 0.7980174422264099
      },
      "Polygon": [
        {
          "X": 0.4026312530040741,
          "Y": 0.7980174422264099
        },
        {
          "X": 0.47913432121276855,
          "Y": 0.7980174422264099
        },
        {
          "X": 0.47913432121276855,
          "Y": 0.8528305292129517
        },
        {
          "X": 0.4026312530040741,
          "Y": 0.8528305292129517
        }
      ]
    },
    "Id": "1876c8ea-d3e8-4c39-870e-47512b3b5080"
  },
  {
    "BlockType": "WORD",
    "Confidence": 99.91166687011719,
```

```
      "Text": "Baker",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.09931197017431259,
          "Height": 0.06008723005652428,
          "Left": 0.5098910331726074,
          "Top": 0.787897527217865
        },
        "Polygon": [
          {
            "X": 0.5098910331726074,
            "Y": 0.787897527217865
          },
          {
            "X": 0.609203040599823,
            "Y": 0.787897527217865
          },
          {
            "X": 0.609203040599823,
            "Y": 0.8479847311973572
          },
          {
            "X": 0.5098910331726074,
            "Y": 0.8479847311973572
          }
        ]
      },
      "Id": "00adeaef-ed57-44eb-b8a9-503575236d62"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.98870849609375,
      "Text": "better",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.10782185196876526,
          "Height": 0.06207133084535599,
          "Left": 0.7428008317947388,
          "Top": 0.7928366661071777
        },
        "Polygon": [
          {
```

```
            "X": 0.7428008317947388,
            "Y": 0.7928366661071777
          },
          {
            "X": 0.8506226539611816,
            "Y": 0.7928366661071777
          },
          {
            "X": 0.8506226539611816,
            "Y": 0.8549079895019531
          },
          {
            "X": 0.7428008317947388,
            "Y": 0.8549079895019531
          }
        ]
      },
      "Id": "c0fc9a58-7a4b-4f69-bafd-2cff32be2665"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.8883285522461,
      "Text": "opp.",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.07421936094760895,
          "Height": 0.058906231075525284,
          "Left": 0.8577775359153748,
          "Top": 0.8038780689239502
        },
        "Polygon": [
          {
            "X": 0.8577775359153748,
            "Y": 0.8038780689239502
          },
          {
            "X": 0.9319969415664673,
            "Y": 0.8038780689239502
          },
          {
            "X": 0.9319969415664673,
            "Y": 0.8627843260765076
          },
```

```
        {
          "X": 0.8577775359153748,
          "Y": 0.8627843260765076
        }
      ]
    },
    "Id": "bf6dc8ee-2fb3-4b6c-aee4-31e96912a2d8"
  },
  {
    "BlockType": "WORD",
    "Confidence": 99.92573547363281,
    "Text": "8/15/2013",
    "TextType": "HANDWRITING",
    "Geometry": {
      "BoundingBox": {
        "Width": 0.10257463902235031,
        "Height": 0.05412459000945091,
        "Left": 0.027909137308597565,
        "Top": 0.8608770370483398
      },
      "Polygon": [
        {
          "X": 0.027909137308597565,
          "Y": 0.8608770370483398
        },
        {
          "X": 0.13048377633094788,
          "Y": 0.8608770370483398
        },
        {
          "X": 0.13048377633094788,
          "Y": 0.915001630783081
        },
        {
          "X": 0.027909137308597565,
          "Y": 0.915001630783081
        }
      ]
    },
    "Id": "5384f860-f857-4a94-9438-9dfa20eed1c6"
  },
  {
    "BlockType": "WORD",
    "Confidence": 99.99625396728516,
```

```
      "Text": "Present",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.09982697665691376,
          "Height": 0.06888339668512344,
          "Left": 0.1420602649450302,
          "Top": 0.8511748909950256
        },
        "Polygon": [
          {
            "X": 0.1420602649450302,
            "Y": 0.8511748909950256
          },
          {
            "X": 0.24188724160194397,
            "Y": 0.8511748909950256
          },
          {
            "X": 0.24188724160194397,
            "Y": 0.9200583100318909
          },
          {
            "X": 0.1420602649450302,
            "Y": 0.9200583100318909
          }
        ]
      },
      "Id": "0bb96ed6-b2e6-4da4-90b3-b85561bbd89d"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.9826431274414,
      "Text": "AnyCompany",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.18611273169517517,
          "Height": 0.08581399917602539,
          "Left": 0.2615866959095001,
          "Top": 0.869536280632019
        },
        "Polygon": [
          {
```

```
            "X": 0.2615866959095001,
            "Y": 0.869536280632019
          },
          {
            "X": 0.4476994276046753,
            "Y": 0.869536280632019
          },
          {
            "X": 0.4476994276046753,
            "Y": 0.9553502798080444
          },
          {
            "X": 0.2615866959095001,
            "Y": 0.9553502798080444
          }
        ]
      },
      "Id": "25343360-d906-440a-88b7-92eb89e95949"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.99523162841797,
      "Text": "head",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.07429949939250946,
          "Height": 0.05485520139336586,
          "Left": 0.49359121918678284,
          "Top": 0.8714361190795898
        },
        "Polygon": [
          {
            "X": 0.49359121918678284,
            "Y": 0.8714361190795898
          },
          {
            "X": 0.5678907036781311,
            "Y": 0.8714361190795898
          },
          {
            "X": 0.5678907036781311,
            "Y": 0.926291286945343
          },
```

```
        {
          "X": 0.49359121918678284,
          "Y": 0.926291286945343
        }
      ]
    },
    "Id": "0ef3c194-8322-4575-94f1-82819ee57e3a"
  },
  {
    "BlockType": "WORD",
    "Confidence": 99.99574279785156,
    "Text": "baker",
    "TextType": "HANDWRITING",
    "Geometry": {
      "BoundingBox": {
        "Width": 0.1019822508096695,
        "Height": 0.05615599825978279,
        "Left": 0.585354208946228,
        "Top": 0.8702592849731445
      },
      "Polygon": [
        {
          "X": 0.585354208946228,
          "Y": 0.8702592849731445
        },
        {
          "X": 0.6873364448547363,
          "Y": 0.8702592849731445
        },
        {
          "X": 0.6873364448547363,
          "Y": 0.9264153242111206
        },
        {
          "X": 0.585354208946228,
          "Y": 0.9264153242111206
        }
      ]
    },
    "Id": "d296acd9-3e9a-4985-95f8-f863614f2c46"
  },
  {
    "BlockType": "WORD",
    "Confidence": 99.9880599975586,
```

```
      "Text": "N/A,",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.08230073750019073,
          "Height": 0.06588289886713028,
          "Left": 0.7411766648292542,
          "Top": 0.8722732067108154
        },
        "Polygon": [
          {
            "X": 0.7411766648292542,
            "Y": 0.8722732067108154
          },
          {
            "X": 0.8234773874282837,
            "Y": 0.8722732067108154
          },
          {
            "X": 0.8234773874282837,
            "Y": 0.9381561279296875
          },
          {
            "X": 0.7411766648292542,
            "Y": 0.9381561279296875
          }
        ]
      },
      "Id": "195cfb5b-ae06-4203-8520-4e4b0a73b5ce"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.97914123535156,
      "Text": "current",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.12791454792022705,
          "Height": 0.04768490046262741,
          "Left": 0.8387037515640259,
          "Top": 0.8843405842781067
        },
        "Polygon": [
          {
```

```
            "X": 0.8387037515640259,
            "Y": 0.8843405842781067
          },
          {
            "X": 0.9666182994842529,
            "Y": 0.8843405842781067
          },
          {
            "X": 0.9666182994842529,
            "Y": 0.9320254921913147
          },
          {
            "X": 0.8387037515640259,
            "Y": 0.9320254921913147
          }
        ]
      },
      "Id": "549ef3f9-3a13-4b77-bc25-fb2e0996839a"
    }
  ],
  "DetectDocumentTextModelVersion": "1.0",
  "ResponseMetadata": {
    "RequestId": "337129e6-3af7-4014-842b-f6484e82cbf6",
    "HTTPStatusCode": 200,
    "HTTPHeaders": {
      "x-amzn-requestid": "337129e6-3af7-4014-842b-f6484e82cbf6",
      "content-type": "application/x-amz-json-1.1",
      "content-length": "45675",
      "date": "Mon, 09 Nov 2020 23:54:38 GMT"
    },
    "RetryAttempts": 0
  }
}
}
```


Détection du texte de document avec Amazon Textract

Pour détecter du texte dans un document, vous utilisez leDetectDocumentTextet transmettez un fichier de document en entrée.`DetectDocumentText`renvoie une structure JSON contenant des lignes et des mots de texte détecté, l'emplacement du texte dans le document et les relations entre le texte détecté. Pour plus d'informations, consultez Détection de texte.

Vous pouvez fournir un document d'entrée sous la forme d'un tableau d'octets d'image (octets d'image encodés en base64) ou en tant qu'objet Amazon S3. Dans cette procédure, vous chargez un fichier image dans votre compartiment S3 et spécifiez le nom du fichier.

Pour détecter du texte dans un document (API)

1. Si vous ne l'avez pas déjà fait :

   a. Créer ou mettre à jour un utilisateur IAM avec`AmazonTextractFullAccess`et`AmazonS3ReadOnlyAccess`Autorisations. Pour plus d'informations, consultez Étape 1 : Configuration d'un compte AWS et création d'un utilisateur IAM.

   b. Installez et configurez l'AWS CLI et les kits SDK AWS. Pour plus d'informations, consultez Étape 2 : Configuration de l'AWS CLIetAWSKits SDK.

2. Chargez un document dans votre compartiment S3.

   Pour obtenir des instructions, consultezChargement d'objets dans Amazon S3dans leManuel de l'utilisateur Amazon Simple Storage Service.

3. Utilisez les exemples suivants pour appeler l'opération `DetectDocumentText`.

   Java

   L'exemple de code suivant affiche le document et les zones autour des lignes de texte détecté.

   Dans la fonction`main`, remplacez les valeurs de`bucket`et`document`avec les noms du compartiment Amazon S3 et du document utilisés à l'étape 2.

```
//Calls DetectDocumentText.
//Loads document from S3 bucket. Displays the document and bounding boxes around
 detected lines/words of text.
package com.amazonaws.samples;

import java.awt.*;
import java.awt.image.BufferedImage;
import java.util.List;
import javax.imageio.ImageIO;
import javax.swing.*;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
```

```
import com.amazonaws.services.s3.model.S3ObjectInputStream;
import com.amazonaws.client.builder.AwsClientBuilder.EndpointConfiguration;
import com.amazonaws.services.textract.AmazonTextract;
import com.amazonaws.services.textract.AmazonTextractClientBuilder;
import com.amazonaws.services.textract.model.Block;
import com.amazonaws.services.textract.model.BoundingBox;
import com.amazonaws.services.textract.model.DetectDocumentTextRequest;
import com.amazonaws.services.textract.model.DetectDocumentTextResult;
import com.amazonaws.services.textract.model.Document;
import com.amazonaws.services.textract.model.S3Object;
import com.amazonaws.services.textract.model.Point;
import com.amazonaws.services.textract.model.Relationship;

public class DocumentText extends JPanel {

    private static final long serialVersionUID = 1L;

    BufferedImage image;
    DetectDocumentTextResult result;

    public DocumentText(DetectDocumentTextResult documentResult, BufferedImage
 bufImage) throws Exception {
        super();

        result = documentResult; // Results of text detection.
        image = bufImage; // The image containing the document.

    }

    // Draws the image and text bounding box.
    public void paintComponent(Graphics g) {

        int height = image.getHeight(this);
        int width = image.getWidth(this);

        Graphics2D g2d = (Graphics2D) g; // Create a Java2D version of g.

        // Draw the image.
        g2d.drawImage(image, 0, 0, image.getWidth(this) , image.getHeight(this),
 this);

        // Iterate through blocks and display polygons around lines of detected
 text.
        List<Block> blocks = result.getBlocks();
```

```
        for (Block block : blocks) {
            DisplayBlockInfo(block);
            if ((block.getBlockType()).equals("LINE")) {
                ShowPolygon(height, width, block.getGeometry().getPolygon(),
g2d);
                /*
                  ShowBoundingBox(height, width,
block.getGeometry().getBoundingBox(), g2d);
                 */
            } else { // its a word, so just show vertical lines.
                ShowPolygonVerticals(height, width,
block.getGeometry().getPolygon(), g2d);
            }
        }
    }

    // Show bounding box at supplied location.
    private void ShowBoundingBox(int imageHeight, int imageWidth, BoundingBox
box, Graphics2D g2d) {

        float left = imageWidth * box.getLeft();
        float top = imageHeight * box.getTop();

        // Display bounding box.
        g2d.setColor(new Color(0, 212, 0));
        g2d.drawRect(Math.round(left), Math.round(top),
                Math.round(imageWidth * box.getWidth()), Math.round(imageHeight
* box.getHeight()));

    }

    // Shows polygon at supplied location
    private void ShowPolygon(int imageHeight, int imageWidth, List<Point>
points, Graphics2D g2d) {

        g2d.setColor(new Color(0, 0, 0));
        Polygon polygon = new Polygon();

        // Construct polygon and display
        for (Point point : points) {
            polygon.addPoint((Math.round(point.getX() * imageWidth)),
                    Math.round(point.getY() * imageHeight));
        }
        g2d.drawPolygon(polygon);
```

```
   }

   // Draws only the vertical lines in the supplied polygon.
   private void ShowPolygonVerticals(int imageHeight, int imageWidth,
List<Point> points, Graphics2D g2d) {

       g2d.setColor(new Color(0, 212, 0));
       Object[] parry = points.toArray();
       g2d.setStroke(new BasicStroke(2));

       g2d.drawLine(Math.round(((Point) parry[0]).getX() * imageWidth),
               Math.round(((Point) parry[0]).getY() * imageHeight),
Math.round(((Point) parry[3]).getX() * imageWidth),
               Math.round(((Point) parry[3]).getY() * imageHeight));

       g2d.setColor(new Color(255, 0, 0));
       g2d.drawLine(Math.round(((Point) parry[1]).getX() * imageWidth),
               Math.round(((Point) parry[1]).getY() * imageHeight),
Math.round(((Point) parry[2]).getX() * imageWidth),
               Math.round(((Point) parry[2]).getY() * imageHeight));

   }
   //Displays information from a block returned by text detection and text
analysis
   private void DisplayBlockInfo(Block block) {
       System.out.println("Block Id : " + block.getId());
       if (block.getText()!=null)
           System.out.println("    Detected text: " + block.getText());
       System.out.println("    Type: " + block.getBlockType());

       if (block.getBlockType().equals("PAGE") !=true) {
           System.out.println("    Confidence: " +
block.getConfidence().toString());
       }
       if(block.getBlockType().equals("CELL"))
       {
           System.out.println("    Cell information:");
           System.out.println("        Column: " + block.getColumnIndex());
           System.out.println("        Row: " + block.getRowIndex());
           System.out.println("        Column span: " + block.getColumnSpan());
           System.out.println("        Row span: " + block.getRowSpan());

       }
```

```
        System.out.println("    Relationships");
        List<Relationship> relationships=block.getRelationships();
        if(relationships!=null) {
            for (Relationship relationship : relationships) {
                System.out.println("        Type: " + relationship.getType());
                System.out.println("        IDs: " +
 relationship.getIds().toString());
            }
        } else {
            System.out.println("        No related Blocks");
        }

        System.out.println("    Geometry");
        System.out.println("        Bounding Box: " +
 block.getGeometry().getBoundingBox().toString());
        System.out.println("        Polygon: " +
 block.getGeometry().getPolygon().toString());

        List<String> entityTypes = block.getEntityTypes();

        System.out.println("    Entity Types");
        if(entityTypes!=null) {
            for (String entityType : entityTypes) {
                System.out.println("        Entity Type: " + entityType);
            }
        } else {
            System.out.println("        No entity type");
        }
        if(block.getPage()!=null)
            System.out.println("    Page: " + block.getPage());
        System.out.println();
    }

    public static void main(String arg[]) throws Exception {

        // The S3 bucket and document
        String document = "";
        String bucket = "";


        AmazonS3 s3client = AmazonS3ClientBuilder.standard()
                .withEndpointConfiguration(
                        new EndpointConfiguration("https://
s3.amazonaws.com","us-east-1"))
```

```
                .build();


        // Get the document from S3
        com.amazonaws.services.s3.model.S3Object s3object =
 s3client.getObject(bucket, document);
        S3ObjectInputStream inputStream = s3object.getObjectContent();
        BufferedImage image = ImageIO.read(inputStream);

        // Call DetectDocumentText
        EndpointConfiguration endpoint = new EndpointConfiguration(
                "https://textract.us-east-1.amazonaws.com", "us-east-1");
        AmazonTextract client = AmazonTextractClientBuilder.standard()
                .withEndpointConfiguration(endpoint).build();


        DetectDocumentTextRequest request = new DetectDocumentTextRequest()
            .withDocument(new Document().withS3Object(new
 S3Object().withName(document).withBucket(bucket)));

        DetectDocumentTextResult result = client.detectDocumentText(request);

        // Create frame and panel.
        JFrame frame = new JFrame("RotateImage");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        DocumentText panel = new DocumentText(result, image);
        panel.setPreferredSize(new Dimension(image.getWidth() ,
 image.getHeight() ));
        frame.setContentPane(panel);
        frame.pack();
        frame.setVisible(true);

    }
}
```

AWS CLI

Cette commande de l'AWS CLI affiche la sortie JSON pour l'opération `detect-document-text` de l'interface de ligne de commande (CLI).

Remplacez les valeurs de`Bucket`et`Name`avec les noms du compartiment Amazon S3 et du document utilisés à l'étape 2.

```
aws textract detect-document-text \
 --document '{"S3Object":{"Bucket":"bucket","Name":"document"}}'
```

Python

L'exemple de code suivant affiche le document et les zones autour des lignes de texte détectées.

Dans la fonction`main`, remplacez les valeurs de`bucket`et`document`avec les noms du compartiment Amazon S3 et du document utilisés à l'étape 2.

```
#Detects text in a document stored in an S3 bucket. Display polygon box around
 text and angled text
import boto3
import io
from io import BytesIO
import sys

import psutil
import time

import math
from PIL import Image, ImageDraw, ImageFont


# Displays information about a block returned by text detection and text
 analysis
def DisplayBlockInformation(block):
    print('Id: {}'.format(block['Id']))
    if 'Text' in block:
        print('    Detected: ' + block['Text'])
    print('    Type: ' + block['BlockType'])

    if 'Confidence' in block:
        print('    Confidence: ' + "{:.2f}".format(block['Confidence']) + "%")

    if block['BlockType'] == 'CELL':
        print("    Cell information")
        print("        Column: " + str(block['ColumnIndex']))
        print("        Row: " + str(block['RowIndex']))
        print("        ColumnSpan: " + str(block['ColumnSpan']))
        print("        RowSpan: " + str(block['RowSpan']))
```

```
    if 'Relationships' in block:
        print('    Relationships: {}'.format(block['Relationships']))
    print('    Geometry: ')
    print('        Bounding Box: {}'.format(block['Geometry']['BoundingBox']))
    print('        Polygon: {}'.format(block['Geometry']['Polygon']))

    if block['BlockType'] == "KEY_VALUE_SET":
        print ('    Entity Type: ' + block['EntityTypes'][0])
    if 'Page' in block:
        print('Page: ' + block['Page'])
    print()

def process_text_detection(bucket, document):


    #Get the document from S3
    s3_connection = boto3.resource('s3')

    s3_object = s3_connection.Object(bucket,document)
    s3_response = s3_object.get()

    stream = io.BytesIO(s3_response['Body'].read())
    image=Image.open(stream)


    # Detect text in the document

    client = boto3.client('textract')
    #process using image bytes
    #image_binary = stream.getvalue()
    #response = client.detect_document_text(Document={'Bytes': image_binary})

    #process using S3 object
    response = client.detect_document_text(
        Document={'S3Object': {'Bucket': bucket, 'Name': document}})

    #Get the text blocks
    blocks=response['Blocks']
    width, height =image.size
    draw = ImageDraw.Draw(image)
    print ('Detected Document Text')

    # Create image showing bounding box/polygon the detected lines/text
```

```
    for block in blocks:
            print('Type: ' + block['BlockType'])
            if block['BlockType'] != 'PAGE':
                print('Detected: ' + block['Text'])
                print('Confidence: ' + "{:.2f}".format(block['Confidence']) +
"%")

            print('Id: {}'.format(block['Id']))
            if 'Relationships' in block:
                print('Relationships: {}'.format(block['Relationships']))
            print('Bounding Box: {}'.format(block['Geometry']['BoundingBox']))
            print('Polygon: {}'.format(block['Geometry']['Polygon']))
            print()
            draw=ImageDraw.Draw(image)
            # Draw WORD - Green -  start of word, red - end of word
            if block['BlockType'] == "WORD":
                draw.line([(width * block['Geometry']['Polygon'][0]['X'],
                height * block['Geometry']['Polygon'][0]['Y']),
                (width * block['Geometry']['Polygon'][3]['X'],
                height * block['Geometry']['Polygon'][3]['Y'])],fill='green',
                width=2)

                draw.line([(width * block['Geometry']['Polygon'][1]['X'],
                height * block['Geometry']['Polygon'][1]['Y']),
                (width * block['Geometry']['Polygon'][2]['X'],
                height * block['Geometry']['Polygon'][2]['Y'])],
                fill='red',
                width=2)


            # Draw box around entire LINE
            if block['BlockType'] == "LINE":
                points=[]

                for polygon in block['Geometry']['Polygon']:
                    points.append((width * polygon['X'], height * polygon['Y']))

                draw.polygon((points), outline='black')

                # Uncomment to draw bounding box
                #box=block['Geometry']['BoundingBox']
                #left = width * box['Left']
                #top = height * box['Top']
```

```
                #draw.rectangle([left,top, left + (width * box['Width']), top
 +(height * box['Height'])],outline='black')


    # Display the image
    image.show()
    # display image for 10 seconds


    return len(blocks)

def main():

    bucket = ''
    document = ''
    block_count=process_text_detection(bucket,document)
    print("Blocks detected: " + str(block_count))

if __name__ == "__main__":
    main()
```

Node.js

L'exemple de code Node.js suivant affiche le document et les cases autour des lignes de texte détectées, affichant une image des résultats dans le répertoire à partir de lequel vous exécutez le code. Il utilise le`image-size`et`images`paquets.

Dans la fonction`main`, remplacez les valeurs de`bucket`et`document`avec les noms du compartiment Amazon S3 et du document utilisés à l'étape 2. Remplacez la valeur de`regionConfig`avec le nom de la région dans laquelle se trouve votre compte.

```
async function main(){

// Import AWS
const AWS = require("aws-sdk")
// Use Image-Size to get
const sizeOf = require('image-size');
// Image tool to draw buffers
const images = require("images");

// Create a canvas and get the context
```

```
const { createCanvas } = require('canvas')
const canvas = createCanvas(200, 200)
const ctx = canvas.getContext('2d')

// Set variables
const bucket = 'bucket-name' // the s3 bucket name
const photo  = 'image-name' // the name of file
const regionConfig = 'region'

// Set region if needed
AWS.config.update({region:regionConfig});

// Connect to Textract
const client = new AWS.Textract();
// Connect to S3 to display image
const s3 = new AWS.S3();

// Define paramaters
const params = {
  Document: {
    S3Object: {
      Bucket: bucket,
      Name: photo
    },
  },
}

// Function to display image
async function getImage(){
  const imageData =  s3.getObject(
    {
        Bucket: bucket,
        Key: photo
      }

  ).promise();
  return imageData;
}

// get image
var imageData = await getImage()

// Get the height, width of the image
const dimensions = sizeOf(imageData.Body)
```

```
const width = dimensions.width
const height = dimensions.height
console.log(imageData.Body)
console.log(width, height)

canvas.width = width;
canvas.height = height;

try{
  // Call API and log response
  const res = await client.detectDocumentText(params).promise();
  var image = images(imageData.Body).size(width, height)
  //console.log the type of block, text, text type, and confidence
  res.Blocks.forEach(block => {
    console.log(`Block Type: ${block.BlockType}`),
    console.log(`Text: ${block.Text}`)
    console.log(`TextType: ${block.TextType}`)
    console.log(`Confidence: ${block.Confidence}`)

    // Draw box around detected text using polygons
    ctx.strokeStyle = 'rgba(0,0,0,0.5)';
    ctx.beginPath();
    block.Geometry.Polygon.forEach(({X, Y}) =>
    ctx.lineTo(width * X - 10, height * Y - 10)
    );
    ctx.closePath();
    ctx.stroke();
    console.log("-----")
  })

  // render image
  var buffer = canvas.toBuffer("image/png");
  image.draw(images(buffer), 10, 10)
  image.save("output-image.jpg");

} catch (err){
console.error(err);}

}

main()
```

4. Exécutez l'exemple. Les exemples Python et Java affichent l'image du document. Une boîte noire entoure chaque ligne de texte détectée. Une ligne verticale verte est le début d'un mot

détecté. Une ligne verticale rouge est la fin d'un mot détecté. LeAWS CLIl'exemple de l'affiche uniquement la sortie JSON pour le`DetectDocumentText`.

Analyse du texte du document avec Amazon Textract

Pour analyser du texte dans un document, vous utilisez leAnalyzeDocumentet transmettez un fichier de document en entrée.`AnalyzeDocument`renvoie une structure JSON qui contient le texte analysé. Pour plus d'informations, consultez Analyse des documents.

Vous pouvez fournir un document d'entrée sous la forme d'un tableau d'octets d'image (octets d'image encodés en base64) ou en tant qu'objet Amazon S3. Dans cette procédure, vous chargez un fichier image dans votre compartiment S3 et spécifiez le nom du fichier.

Pour analyser du texte dans un document (API)

1. Si vous ne l'avez pas déjà fait :

   a. Créer ou mettre à jour un utilisateur IAM avec`AmazonTextractFullAccess`et`AmazonS3ReadOnlyAccess`Autorisations. Pour plus d'informations, consultez Étape 1 : Configuration d'un compte AWS et création d'un utilisateur IAM.

   b. Installez et configurez l'AWS CLI et les kits SDK AWS. Pour plus d'informations, consultez Étape 2 : Configuration de l'AWS CLIetAWSKits SDK.

2. Chargez une image qui contient un document dans votre compartiment S3.

   Pour obtenir des instructions, consultezChargement d'objets dans Amazon S3dans leManuel de l'utilisateur Amazon Simple Storage Service.

3. Utilisez les exemples suivants pour appeler l'opération `AnalyzeDocument`.

   Java

   L'exemple de code suivant affiche le document et les boîtes autour des éléments détectés.

   Dans la fonction`main`, remplacez les valeurs de`bucket`et`document`avec les noms du compartiment Amazon S3 et de l'image de document utilisés à l'étape 2.

```
//Loads document from S3 bucket. Displays the document and polygon around
 detected lines of text.
```

```
package com.amazonaws.samples;

import java.awt.*;
import java.awt.image.BufferedImage;
import java.util.List;
import javax.imageio.ImageIO;
import javax.swing.*;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.S3ObjectInputStream;
import com.amazonaws.services.textract.AmazonTextract;
import com.amazonaws.services.textract.AmazonTextractClientBuilder;
import com.amazonaws.services.textract.model.AnalyzeDocumentRequest;
import com.amazonaws.services.textract.model.AnalyzeDocumentResult;
import com.amazonaws.services.textract.model.Block;
import com.amazonaws.services.textract.model.BoundingBox;
import com.amazonaws.services.textract.model.Document;
import com.amazonaws.services.textract.model.S3Object;
import com.amazonaws.services.textract.model.Point;
import com.amazonaws.services.textract.model.Relationship;
import com.amazonaws.client.builder.AwsClientBuilder.EndpointConfiguration;

public class AnalyzeDocument extends JPanel {

    private static final long serialVersionUID = 1L;

    BufferedImage image;

    AnalyzeDocumentResult result;

    public AnalyzeDocument(AnalyzeDocumentResult documentResult, BufferedImage
 bufImage) throws Exception {
        super();

        result = documentResult; // Results of text detection.
        image = bufImage; // The image containing the document.

    }

    // Draws the image and text bounding box.
    public void paintComponent(Graphics g) {

        int height = image.getHeight(this);
        int width = image.getWidth(this);
```

```
        Graphics2D g2d = (Graphics2D) g; // Create a Java2D version of g.

        // Draw the image.
        g2d.drawImage(image, 0, 0, image.getWidth(this), image.getHeight(this),
this);

        // Iterate through blocks and display bounding boxes around everything.

        List<Block> blocks = result.getBlocks();
        for (Block block : blocks) {
            DisplayBlockInfo(block);
            switch(block.getBlockType()) {

            case "KEY_VALUE_SET":
                if (block.getEntityTypes().contains("KEY")){
                    ShowBoundingBox(height, width,
block.getGeometry().getBoundingBox(), g2d, new Color(255,0,0));
                }
                else {  //VALUE
                    ShowBoundingBox(height, width,
block.getGeometry().getBoundingBox(), g2d, new Color(0,255,0));
                }
                break;
            case "TABLE":
                ShowBoundingBox(height, width,
block.getGeometry().getBoundingBox(), g2d, new Color(0,0,255));
                break;
            case "CELL":
                ShowBoundingBox(height, width,
block.getGeometry().getBoundingBox(), g2d, new Color(255,255,0));
                break;
            case "SELECTION_ELEMENT":
                if (block.getSelectionStatus().equals("SELECTED"))
                    ShowSelectedElement(height, width,
block.getGeometry().getBoundingBox(), g2d, new Color(0,0,255));
                break;
            default:
                //PAGE, LINE & WORD
                //ShowBoundingBox(height, width,
block.getGeometry().getBoundingBox(), g2d, new Color(200,200,0));
            }
        }
```

```
         // uncomment to show polygon around all blocks
         //ShowPolygon(height,width,block.getGeometry().getPolygon(),g2d);


   }

   // Show bounding box at supplied location.
   private void ShowBoundingBox(int imageHeight, int imageWidth, BoundingBox
box, Graphics2D g2d, Color color) {

       float left = imageWidth * box.getLeft();
       float top = imageHeight * box.getTop();

       // Display bounding box.
       g2d.setColor(color);
       g2d.drawRect(Math.round(left), Math.round(top),
               Math.round(imageWidth * box.getWidth()), Math.round(imageHeight
* box.getHeight()));

   }
   private void ShowSelectedElement(int imageHeight, int imageWidth,
BoundingBox box, Graphics2D g2d, Color color) {

       float left = imageWidth * box.getLeft();
       float top = imageHeight * box.getTop();

       // Display bounding box.
       g2d.setColor(color);
       g2d.fillRect(Math.round(left), Math.round(top),
               Math.round(imageWidth * box.getWidth()), Math.round(imageHeight
* box.getHeight()));

   }

   // Shows polygon at supplied location
   private void ShowPolygon(int imageHeight, int imageWidth, List<Point>
points, Graphics2D g2d) {

       g2d.setColor(new Color(0, 0, 0));
       Polygon polygon = new Polygon();

       // Construct polygon and display
       for (Point point : points) {
           polygon.addPoint((Math.round(point.getX() * imageWidth)),
```

```
                    Math.round(point.getY() * imageHeight));
        }
        g2d.drawPolygon(polygon);
    }
    //Displays information from a block returned by text detection and text
analysis
    private void DisplayBlockInfo(Block block) {
        System.out.println("Block Id : " + block.getId());
        if (block.getText()!=null)
            System.out.println("    Detected text: " + block.getText());
        System.out.println("    Type: " + block.getBlockType());

        if (block.getBlockType().equals("PAGE") !=true) {
            System.out.println("    Confidence: " +
block.getConfidence().toString());
        }
        if(block.getBlockType().equals("CELL"))
        {
            System.out.println("    Cell information:");
            System.out.println("        Column: " + block.getColumnIndex());
            System.out.println("        Row: " + block.getRowIndex());
            System.out.println("        Column span: " + block.getColumnSpan());
            System.out.println("        Row span: " + block.getRowSpan());

        }

        System.out.println("    Relationships");
        List<Relationship> relationships=block.getRelationships();
        if(relationships!=null) {
            for (Relationship relationship : relationships) {
                System.out.println("        Type: " + relationship.getType());
                System.out.println("        IDs: " +
relationship.getIds().toString());
            }
        } else {
            System.out.println("        No related Blocks");
        }

        System.out.println("    Geometry");
        System.out.println("        Bounding Box: " +
block.getGeometry().getBoundingBox().toString());
        System.out.println("        Polygon: " +
block.getGeometry().getPolygon().toString());
```

```
        List<String> entityTypes = block.getEntityTypes();

        System.out.println("    Entity Types");
        if(entityTypes!=null) {
            for (String entityType : entityTypes) {
                System.out.println("        Entity Type: " + entityType);
            }
        } else {
            System.out.println("        No entity type");
        }

        if(block.getBlockType().equals("SELECTION_ELEMENT")) {
            System.out.print("    Selection element detected: ");
            if (block.getSelectionStatus().equals("SELECTED")){
                System.out.println("Selected");
            }else {
                System.out.println(" Not selected");
            }
        }

        if(block.getPage()!=null)
            System.out.println("    Page: " + block.getPage());
        System.out.println();
    }

    public static void main(String arg[]) throws Exception {

        // The S3 bucket and document
        String document = "";
        String bucket = "";

        AmazonS3 s3client = AmazonS3ClientBuilder.standard()
                .withEndpointConfiguration(
                        new EndpointConfiguration("https://
s3.amazonaws.com","us-east-1"))
                .build();


        // Get the document from S3
        com.amazonaws.services.s3.model.S3Object s3object =
 s3client.getObject(bucket, document);
        S3ObjectInputStream inputStream = s3object.getObjectContent();
        BufferedImage image = ImageIO.read(inputStream);
```

```
        // Call AnalyzeDocument
        EndpointConfiguration endpoint = new EndpointConfiguration(
                "https://textract.us-east-1.amazonaws.com", "us-east-1");
        AmazonTextract client = AmazonTextractClientBuilder.standard()
                .withEndpointConfiguration(endpoint).build();


        AnalyzeDocumentRequest request = new AnalyzeDocumentRequest()
                .withFeatureTypes("TABLES","FORMS")
                 .withDocument(new Document().
                       withS3Object(new
 S3Object().withName(document).withBucket(bucket)));


        AnalyzeDocumentResult result = client.analyzeDocument(request);

        // Create frame and panel.
        JFrame frame = new JFrame("RotateImage");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        AnalyzeDocument panel = new AnalyzeDocument(result, image);
        panel.setPreferredSize(new Dimension(image.getWidth(),
 image.getHeight()));
        frame.setContentPane(panel);
        frame.pack();
        frame.setVisible(true);

    }
}
```

AWS CLI

Cette commande de l'AWS CLI affiche la sortie JSON pour l'opération `detect-document-text` de l'interface de ligne de commande (CLI).

Remplacez les valeurs de`Bucket`et`Name`avec les noms du compartiment Amazon S3 et du document utilisés à l'étape 2.

```
aws textract analyze-document \
 --document '{"S3Object":{"Bucket":"bucket","Name":"document"}}' \
 --feature-types '["TABLES","FORMS"]'
```

Python

L'exemple de code suivant affiche le document et les boîtes autour des éléments détectés.

Dans la fonction`main`, remplacez les valeurs de`bucket`et`document`avec les noms du compartiment Amazon S3 et du document utilisés à l'étape 2.

```
#Analyzes text in a document stored in an S3 bucket. Display polygon box around
 text and angled text
import boto3
import io
from io import BytesIO
import sys

import math
from PIL import Image, ImageDraw, ImageFont

def ShowBoundingBox(draw,box,width,height,boxColor):

    left = width * box['Left']
    top = height * box['Top']
    draw.rectangle([left,top, left + (width * box['Width']), top +(height *
 box['Height'])],outline=boxColor)

def ShowSelectedElement(draw,box,width,height,boxColor):

    left = width * box['Left']
    top = height * box['Top']
    draw.rectangle([left,top, left + (width * box['Width']), top +(height *
 box['Height'])],fill=boxColor)

# Displays information about a block returned by text detection and text
 analysis
def DisplayBlockInformation(block):
    print('Id: {}'.format(block['Id']))
    if 'Text' in block:
        print('    Detected: ' + block['Text'])
    print('    Type: ' + block['BlockType'])

    if 'Confidence' in block:
        print('    Confidence: ' + "{:.2f}".format(block['Confidence']) + "%")
```

```
    if block['BlockType'] == 'CELL':
        print("    Cell information")
        print("        Column:" + str(block['ColumnIndex']))
        print("        Row:" + str(block['RowIndex']))
        print("        Column Span:" + str(block['ColumnSpan']))
        print("        RowSpan:" + str(block['ColumnSpan']))

    if 'Relationships' in block:
        print('    Relationships: {}'.format(block['Relationships']))
    print('    Geometry: ')
    print('        Bounding Box: {}'.format(block['Geometry']['BoundingBox']))
    print('        Polygon: {}'.format(block['Geometry']['Polygon']))

    if block['BlockType'] == "KEY_VALUE_SET":
        print ('    Entity Type: ' + block['EntityTypes'][0])

    if block['BlockType'] == 'SELECTION_ELEMENT':
        print('    Selection element detected: ', end='')

        if block['SelectionStatus'] =='SELECTED':
            print('Selected')
        else:
            print('Not selected')

    if 'Page' in block:
        print('Page: ' + block['Page'])
    print()

def process_text_analysis(bucket, document):

    #Get the document from S3
    s3_connection = boto3.resource('s3')

    s3_object = s3_connection.Object(bucket,document)
    s3_response = s3_object.get()

    stream = io.BytesIO(s3_response['Body'].read())
    image=Image.open(stream)

    # Analyze the document
    client = boto3.client('textract')

    image_binary = stream.getvalue()
    response = client.analyze_document(Document={'Bytes': image_binary},
```

```
        FeatureTypes=["TABLES", "FORMS"])

    ### Alternatively, process using S3 object ###
    #response = client.analyze_document(
    #    Document={'S3Object': {'Bucket': bucket, 'Name': document}},
    #    FeatureTypes=["TABLES", "FORMS"])

    ### To use a local file ###
    # with open("pathToFile", 'rb') as img_file:
        ### To display image using PIL ###
    #    image = Image.open()
        ### Read bytes ###
    #    img_bytes = img_file.read()
    #    response = client.analyze_document(Document={'Bytes': img_bytes},
 FeatureTypes=["TABLES", "FORMS"])


    #Get the text blocks
    blocks=response['Blocks']
    width, height =image.size
    draw = ImageDraw.Draw(image)
    print ('Detected Document Text')

    # Create image showing bounding box/polygon the detected lines/text
    for block in blocks:

        DisplayBlockInformation(block)

        draw=ImageDraw.Draw(image)
        if block['BlockType'] == "KEY_VALUE_SET":
            if block['EntityTypes'][0] == "KEY":
                ShowBoundingBox(draw, block['Geometry']
['BoundingBox'],width,height,'red')
            else:
                ShowBoundingBox(draw, block['Geometry']
['BoundingBox'],width,height,'green')

        if block['BlockType'] == 'TABLE':
            ShowBoundingBox(draw, block['Geometry']['BoundingBox'],width,height,
 'blue')

        if block['BlockType'] == 'CELL':
            ShowBoundingBox(draw, block['Geometry']['BoundingBox'],width,height,
 'yellow')
```

```
        if block['BlockType'] == 'SELECTION_ELEMENT':
            if block['SelectionStatus'] =='SELECTED':
                ShowSelectedElement(draw, block['Geometry']
['BoundingBox'],width,height, 'blue')

            #uncomment to draw polygon for all Blocks
            #points=[]
            #for polygon in block['Geometry']['Polygon']:
            #    points.append((width * polygon['X'], height * polygon['Y']))
            #draw.polygon((points), outline='blue')

    # Display the image
    image.show()
    return len(blocks)


def main():

    bucket = ''
    document = ''
    block_count=process_text_analysis(bucket,document)
    print("Blocks detected: " + str(block_count))

if __name__ == "__main__":
    main()
```

Node.js

L'exemple de code suivant affiche le document et les boîtes autour des éléments détectés.

Dans le code ci-dessous, remplacez les valeurs de`bucket`et`photo`avec les noms du compartiment Amazon S3 et du document utilisés à l'étape 2. Remplacez la valeur de`region`avec la région associée à votre compte.

```
// Import required AWS SDK clients and commands for Node.js
import { AnalyzeDocumentCommand } from  "@aws-sdk/client-textract";
import  { TextractClient } from "@aws-sdk/client-textract";

// Set the AWS Region.
const REGION = "region"; //e.g. "us-east-1"
// Create SNS service object.
const textractClient = new TextractClient({ region: REGION });
```

```
const bucket = 'buckets'
const photo = 'photo'

// Set params
const params = {
    Document: {
      S3Object: {
        Bucket: bucket,
        Name: photo
      },
    },
    FeatureTypes: ['TABLES', 'FORMS'],
  }

const displayBlockInfo = async (response) => {
    try {
        response.Blocks.forEach(block => {
            console.log(`ID: ${block.Id}`)
            console.log(`Block Type: ${block.BlockType}`)
            if ("Text" in block && block.Text !== undefined){
                console.log(`Text: ${block.Text}`)
            }
            else{}
            if ("Confidence" in block && block.Confidence !== undefined){
                console.log(`Confidence: ${block.Confidence}`)
            }
            else{}
            if (block.BlockType == 'CELL'){
                console.log("Cell info:")
                console.log(`   Column Index - ${block.ColumnIndex}`)
                console.log(`   Row - ${block.RowIndex}`)
                console.log(`   Column Span - ${block.ColumnSpan}`)
                console.log(`   Row Span - ${block.RowSpan}`)
            }
            if ("Relationships" in block && block.Relationships !== undefined){
                console.log(block.Relationships)
                console.log("Geometry:")
                console.log(`   Bounding Box -
 ${JSON.stringify(block.Geometry.BoundingBox)}`)
                console.log(`   Polygon -
 ${JSON.stringify(block.Geometry.Polygon)}`)
            }
            console.log("-----")
        });
```

```
      } catch (err) {
        console.log("Error", err);
      }
}

const analyze_document_text = async () => {
    try {
        const analyzeDoc = new AnalyzeDocumentCommand(params);
        const response = await textractClient.send(analyzeDoc);
        //console.log(response)
        displayBlockInfo(response)
        return response; // For unit tests.
      } catch (err) {
        console.log("Error", err);
      }
}

analyze_document_text()
```

4. Exécutez l'exemple. Les exemples Python et Java affichent l'image du document avec les zones de sélection colorées suivantes :

   - Red — Objets KEY Block
   - Vert — objets Value Block
   - Blue — Objets BLOC TABLE
   - Yellow — objets CELL Block

   Les éléments de sélection sélectionnés sont remplis de bleu.

   LeAWS CLIl'exemple de l'affiche uniquement la sortie JSON pour le`AnalyzeDocument`.

Analyse des factures et des reçus avec Amazon Textract

Pour analyser les documents de facture et de réception, vous utilisez l'API AnalyzeExpense et transmettez un fichier de document en entrée.`AnalyzeExpense`est une opération synchrone qui renvoie une structure JSON contenant le texte analysé. Pour plus d'informations, consultez Analyse des factures et des reçus.

Pour analyser les factures et les réceptions de manière asynchrone, utilisez`StartExpenseAnalysis`pour commencer à traiter un fichier de document d'entrée et utiliser`GetExpenseAnalysis`pour obtenir les résultats.

Vous pouvez fournir un document d'entrée sous la forme d'un tableau d'octets d'image (octets d'image encodés en base64) ou en tant qu'objet Amazon S3. Dans cette procédure, vous chargez un fichier image dans votre compartiment S3 et spécifiez le nom du fichier.

Pour analyser une facture ou un reçu (API)

1. Si vous ne l'avez pas déjà fait :

   a. Créer ou mettre à jour un utilisateur IAM avec`AmazonTextractFullAccess`et`AmazonS3ReadOnlyAccess`Autorisations. Pour plus d'informations, consultez Étape 1 : Configuration d'un compte AWS et création d'un utilisateur IAM.

   b. Installez et configurez l'AWS CLI et les kits SDK AWS. Pour plus d'informations, consultez Étape 2 : Configuration de l'AWS CLIetAWSKits SDK.

2. Chargez une image qui contient un document dans votre compartiment S3.

   Pour obtenir des instructions, consultezChargement d'objets dans Amazon S3dans leManuel de l'utilisateur Amazon Simple Storage Service.

3. Utilisez les exemples suivants pour appeler l'opération `AnalyzeExpense`.

   CLI

```
  aws textract analyze-expense --document '{"S3Object": {"Bucket": "bucket
 name","Name":
"object name"}}'
```

   Python

```
import boto3
import io
from PIL import Image, ImageDraw

def draw_bounding_box(key, val, width, height, draw):
    # If a key is Geometry, draw the bounding box info in it
```

```
    if "Geometry" in key:
        # Draw bounding box information
        box = val["BoundingBox"]
        left = width * box['Left']
        top = height * box['Top']
        draw.rectangle([left, top, left + (width * box['Width']), top + (height
 * box['Height'])],
                       outline='black')

# Takes a field as an argument and prints out the detected labels and values
def print_labels_and_values(field):
    # Only if labels are detected and returned
    if "LabelDetection" in field:
        print("Summary Label Detection - Confidence: {}".format(
            str(field.get("LabelDetection")["Confidence"])) + ", "
              + "Summary Values: {}".format(str(field.get("LabelDetection")
["Text"])))
        print(field.get("LabelDetection")["Geometry"])
    else:
        print("Label Detection - No labels returned.")
    if "ValueDetection" in field:
        print("Summary Value Detection - Confidence: {}".format(
            str(field.get("ValueDetection")["Confidence"])) + ", "
              + "Summary Values: {}".format(str(field.get("ValueDetection")
["Text"])))
        print(field.get("ValueDetection")["Geometry"])
    else:
        print("Value Detection - No values returned")

def process_text_detection(bucket, document):
    # Get the document from S3
    s3_connection = boto3.resource('s3')
    s3_object = s3_connection.Object(bucket, document)
    s3_response = s3_object.get()

    # opening binary stream using an in-memory bytes buffer
    stream = io.BytesIO(s3_response['Body'].read())

    # loading stream into image
    image = Image.open(stream)

    # Detect text in the document
    client = boto3.client('textract', region_name="us-east-1")
```

```
    # process using S3 object
    response = client.analyze_expense(
        Document={'S3Object': {'Bucket': bucket, 'Name': document}})

    # Set width and height to display image and draw bounding boxes
    # Create drawing object
    width, height = image.size
    draw = ImageDraw.Draw(image)

    for expense_doc in response["ExpenseDocuments"]:
        for line_item_group in expense_doc["LineItemGroups"]:
            for line_items in line_item_group["LineItems"]:
                for expense_fields in line_items["LineItemExpenseFields"]:
                    print_labels_and_values(expense_fields)
                    print()

        print("Summary:")
        for summary_field in expense_doc["SummaryFields"]:
            print_labels_and_values(summary_field)
            print()

        #For draw bounding boxes
        for line_item_group in expense_doc["LineItemGroups"]:
            for line_items in line_item_group["LineItems"]:
                for expense_fields in line_items["LineItemExpenseFields"]:
                    for key, val in expense_fields["ValueDetection"].items():
                        if "Geometry" in key:
                            draw_bounding_box(key, val, width, height, draw)

        for label in expense_doc["SummaryFields"]:
            if "LabelDetection" in label:
                for key, val in label["LabelDetection"].items():
                    draw_bounding_box(key, val, width, height, draw)

    # Display the image
    image.show()

def main():
    bucket = 'Bucket-Name'
    document = 'Document-Name'
    process_text_detection(bucket, document)

if __name__ == "__main__":
    main()
```

Java

```
package com.amazonaws.samples;

import java.awt.*;
import java.awt.image.BufferedImage;
import java.io.ByteArrayInputStream;
import java.io.IOException;
import java.util.List;
import java.util.concurrent.CompletableFuture;
import javax.imageio.ImageIO;
import javax.swing.*;
import software.amazon.awssdk.auth.credentials.AwsBasicCredentials;
import software.amazon.awssdk.auth.credentials.StaticCredentialsProvider;
import software.amazon.awssdk.core.ResponseBytes;
import software.amazon.awssdk.core.async.AsyncResponseTransformer;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.*;
import software.amazon.awssdk.services.s3.model.GetObjectRequest;
import software.amazon.awssdk.services.s3.model.GetObjectResponse;
import software.amazon.awssdk.services.textract.TextractClient;
import software.amazon.awssdk.services.textract.model.AnalyzeExpenseRequest;
import software.amazon.awssdk.services.textract.model.AnalyzeExpenseResponse;
import software.amazon.awssdk.services.textract.model.BoundingBox;
import software.amazon.awssdk.services.textract.model.Document;
import software.amazon.awssdk.services.textract.model.ExpenseDocument;
import software.amazon.awssdk.services.textract.model.ExpenseField;
import software.amazon.awssdk.services.textract.model.LineItemFields;
import software.amazon.awssdk.services.textract.model.LineItemGroup;
import software.amazon.awssdk.services.textract.model.S3Object;
import software.amazon.awssdk.services.textract.model.Point;

/**
 *
 * Demo code to parse Textract AnalyzeExpense API
 *
 */
public class TextractAnalyzeExpenseSample extends JPanel {
```

```
private static final long serialVersionUID = 1L;

BufferedImage image;
static AnalyzeExpenseResponse result;

public TextractAnalyzeExpenseSample(AnalyzeExpenseResponse documentResult,
BufferedImage bufImage) throws Exception {
 super();

 result = documentResult; // Results of analyzeexpense summaryfields and
lineitemgroups detection.
 image = bufImage; // The image containing the document.

}

// Draws the image and text bounding box.
public void paintComponent(Graphics g) {

 Graphics2D g2d = (Graphics2D) g; // Create a Java2D version of g.

 // Draw the image.
 g2d.drawImage(image, 0, 0, image.getWidth(this), image.getHeight(this), this);

 // Iterate through summaryfields and lineitemgroups and display boundedboxes
around lines of detected label and value.
 List<ExpenseDocument> expenseDocuments = result.expenseDocuments();
 for (ExpenseDocument expenseDocument : expenseDocuments) {

  if (expenseDocument.hasSummaryFields()) {
   DisplayAnalyzeExpenseSummaryInfo(expenseDocument);
   List<ExpenseField> summaryfields = expenseDocument.summaryFields();
   for (ExpenseField summaryfield : summaryfields) {

    if (summaryfield.valueDetection() != null) {
     ShowBoundingBox(image.getHeight(this), image.getWidth(this),
       summaryfield.valueDetection().geometry().boundingBox(), g2d, new
Color(0, 0, 0));
    }

    if (summaryfield.labelDetection() != null) {

     ShowBoundingBox(image.getHeight(this), image.getWidth(this),
       summaryfield.labelDetection().geometry().boundingBox(), g2d, new
Color(0, 0, 0));
```

```
    }

   }

  }

  if (expenseDocument.hasLineItemGroups()) {
   DisplayAnalyzeExpenseLineItemGroupsInfo(expenseDocument);

   List<LineItemGroup> lineitemgroups = expenseDocument.lineItemGroups();

   for (LineItemGroup lineitemgroup : lineitemgroups) {

    if (lineitemgroup.hasLineItems()) {

     List<LineItemFields> lineItems = lineitemgroup.lineItems();
     for (LineItemFields lineitemfield : lineItems) {

      if (lineitemfield.hasLineItemExpenseFields()) {

       List<ExpenseField> expensefields =
lineitemfield.lineItemExpenseFields();
       for (ExpenseField expensefield : expensefields) {

        if (expensefield.valueDetection() != null) {
         ShowBoundingBox(image.getHeight(this), image.getWidth(this),
           expensefield.valueDetection().geometry().boundingBox(), g2d,
           new Color(0, 0, 0));
        }

        if (expensefield.labelDetection() != null) {
         ShowBoundingBox(image.getHeight(this), image.getWidth(this),
           expensefield.labelDetection().geometry().boundingBox(), g2d,
           new Color(0, 0, 0));
        }

       }
      }

     }

    }
```

```
    }

   }
  }

}

// Show bounding box at supplied location.
private void ShowBoundingBox(float imageHeight, float imageWidth, BoundingBox
box, Graphics2D g2d, Color color) {

 float left = imageWidth * box.left();
 float top = imageHeight * box.top();

 // Display bounding box.
 g2d.setColor(color);
 g2d.drawRect(Math.round(left), Math.round(top), Math.round(imageWidth *
box.width()),
    Math.round(imageHeight * box.height()));

}

private void ShowSelectedElement(float imageHeight, float imageWidth,
BoundingBox box, Graphics2D g2d,
  Color color) {

 float left = (float) imageWidth * (float) box.left();
 float top = (float) imageHeight * (float) box.top();
 System.out.println(left);
 System.out.println(top);

 // Display bounding box.
 g2d.setColor(color);
 g2d.fillRect(Math.round(left), Math.round(top), Math.round(imageWidth *
box.width()),
    Math.round(imageHeight * box.height()));

}

// Shows polygon at supplied location
private void ShowPolygon(int imageHeight, int imageWidth, List<Point> points,
Graphics2D g2d) {

 g2d.setColor(new Color(0, 0, 0));
```

```
 Polygon polygon = new Polygon();

 // Construct polygon and display
 for (Point point : points) {
  polygon.addPoint((Math.round(point.x() * imageWidth)), Math.round(point.y() *
imageHeight));
 }
 g2d.drawPolygon(polygon);
}

private void DisplayAnalyzeExpenseSummaryInfo(ExpenseDocument expensedocument)
{
 System.out.println(" ExpenseId : " + expensedocument.expenseIndex());
 System.out.println("    Expense Summary information:");
 if (expensedocument.hasSummaryFields()) {

  List<ExpenseField> summaryfields = expensedocument.summaryFields();

  for (ExpenseField summaryfield : summaryfields) {

   System.out.println("    Page: " + summaryfield.pageNumber());
   if (summaryfield.type() != null) {

    System.out.println("    Expense Summary Field Type:" +
summaryfield.type().text());

   }
   if (summaryfield.labelDetection() != null) {

    System.out.println("    Expense Summary Field Label:" +
summaryfield.labelDetection().text());
    System.out.println("    Geometry");
    System.out.println("        Bounding Box: "
      + summaryfield.labelDetection().geometry().boundingBox().toString());
    System.out.println(
      "        Polygon: " +
summaryfield.labelDetection().geometry().polygon().toString());

   }
   if (summaryfield.valueDetection() != null) {
    System.out.println("    Expense Summary Field Value:" +
summaryfield.valueDetection().text());
    System.out.println("    Geometry");
    System.out.println("        Bounding Box: "
```

```
      + summaryfield.valueDetection().geometry().boundingBox().toString());
    System.out.println(
      "        Polygon: " +
summaryfield.valueDetection().geometry().polygon().toString());

   }

  }

 }

}

private void DisplayAnalyzeExpenseLineItemGroupsInfo(ExpenseDocument
expensedocument) {

 System.out.println(" ExpenseId : " + expensedocument.expenseIndex());
 System.out.println("    Expense LineItemGroups information:");

 if (expensedocument.hasLineItemGroups()) {

  List<LineItemGroup> lineitemgroups = expensedocument.lineItemGroups();

  for (LineItemGroup lineitemgroup : lineitemgroups) {

   System.out.println("    Expense LineItemGroupsIndexID :" +
lineitemgroup.lineItemGroupIndex());

   if (lineitemgroup.hasLineItems()) {

    List<LineItemFields> lineItems = lineitemgroup.lineItems();

    for (LineItemFields lineitemfield : lineItems) {

     if (lineitemfield.hasLineItemExpenseFields()) {

      List<ExpenseField> expensefields = lineitemfield.lineItemExpenseFields();
      for (ExpenseField expensefield : expensefields) {

       if (expensefield.type() != null) {
        System.out.println("    Expense LineItem Field Type:" +
expensefield.type().text());

       }
```

```
       if (expensefield.valueDetection() != null) {
        System.out.println(
          "    Expense Summary Field Value:" +
expensefield.valueDetection().text());
        System.out.println("    Geometry");
        System.out.println("        Bounding Box: "
          + expensefield.valueDetection().geometry().boundingBox().toString());
        System.out.println("        Polygon: "
          + expensefield.valueDetection().geometry().polygon().toString());

       }

       if (expensefield.labelDetection() != null) {
        System.out.println(
          "    Expense LineItem Field Label:" +
expensefield.labelDetection().text());
        System.out.println("    Geometry");
        System.out.println("        Bounding Box: "
          + expensefield.labelDetection().geometry().boundingBox().toString());
        System.out.println("        Polygon: "
          + expensefield.labelDetection().geometry().polygon().toString());
       }

      }
     }

    }

   }
  }
 }

}

public static void main(String arg[]) throws Exception {

 // Creates a default async client with credentials and AWS Region loaded from
 // the
 // environment


 S3AsyncClient client =
S3AsyncClient.builder().region(Region.US_EAST_1).build();
```

```
  System.out.println("Creating the S3 Client");

  // Start the call to Amazon S3, not blocking to wait for the result
  CompletableFuture<ResponseBytes<GetObjectResponse>> responseFuture =
 client.getObject(
    GetObjectRequest.builder().bucket("textractanalyzeexpense").key("input/
sample-receipt.jpg").build(),
    AsyncResponseTransformer.toBytes());

  System.out.println("Successfully read the object");

  // When future is complete (either successfully or in error), handle the
  // response
  CompletableFuture<ResponseBytes<GetObjectResponse>> operationCompleteFuture =
 responseFuture
    .whenComplete((getObjectResponse, exception) -> {
     if (getObjectResponse != null) {
      // At this point, the file my-file.out has been created with the data
      // from S3; let's just print the object version
      // Convert this into Async call and remove the below block from here and
 put it
      // outside


      TextractClient textractclient =
 TextractClient.builder().region(Region.US_EAST_1).build();

      AnalyzeExpenseRequest request = AnalyzeExpenseRequest.builder()
        .document(
          Document.builder().s3Object(S3Object.builder().name("YOURObjectName")
            .bucket("YOURBucket").build()).build())
        .build();

      AnalyzeExpenseResponse result = textractclient.analyzeExpense(request);

      System.out.print(result.toString());

      ByteArrayInputStream bais = new
 ByteArrayInputStream(getObjectResponse.asByteArray());
      try {
       BufferedImage image = ImageIO.read(bais);
       System.out.println("Successfully read the image");
       JFrame frame = new JFrame("Expense Image");
```

```
       frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
       TextractAnalyzeExpense panel = new TextractAnalyzeExpense(result, image);
       panel.setPreferredSize(new Dimension(image.getWidth(),
 image.getHeight()));
       frame.setContentPane(panel);
       frame.pack();
       frame.setVisible(true);
      } catch (IOException e) {
       throw new RuntimeException(e);
      } catch (Exception e) {
       // TODO Auto-generated catch block
       e.printStackTrace();
      }
     } else {
      // Handle the error
      exception.printStackTrace();
     }
    });

  // We could do other work while waiting for the AWS call to complete in
  // the background, but we'll just wait for "whenComplete" to finish instead
  operationCompleteFuture.join();

 }
}
```

Node.Js

```
                    // Import required AWS SDK clients and commands for Node.js
import { AnalyzeExpenseCommand } from  "@aws-sdk/client-textract";
import  { TextractClient } from "@aws-sdk/client-textract";

// Set the AWS Region.
const REGION = "region"; //e.g. "us-east-1"
// Create SNS service object.
const textractClient = new TextractClient({ region: REGION });

const bucket = 'bucket'
const photo = 'photo'
```

```
// Set params
const params = {
    Document: {
      S3Object: {
        Bucket: bucket,
        Name: photo
      },
    },
  }

const process_text_detection = async () => {
    try {
        const aExpense = new AnalyzeExpenseCommand(params);
        const response = await textractClient.send(aExpense);
        //console.log(response)
        response.ExpenseDocuments.forEach(doc => {
            doc.LineItemGroups.forEach(items => {
                items.LineItems.forEach(fields => {
                    fields.LineItemExpenseFields.forEach(expenseFields =>{
                        console.log(expenseFields)
                    })
                }
                )}
                )
            }
        )
        return response; // For unit tests.
      } catch (err) {
        console.log("Error", err);
      }
}

process_text_detection()
```

4. Vous obtiendrez ainsi la sortie JSON pour`AnalyzeExpense`.

Analyse de la documentation d'identité avec Amazon Textract

Pour analyser les documents d'identité, vous utilisez l'API AnalyzeID et transmettez un fichier de document en entrée.`AnalyzeID`renvoie une structure JSON qui contient le texte analysé. Pour plus d'informations, consultez Analyse des documents d'identité.

Vous pouvez fournir un document d'entrée sous la forme d'un tableau d'octets d'image (octets d'image encodés en base64) ou en tant qu'objet Amazon S3. Dans cette procédure, vous chargez un fichier image dans votre compartiment S3 et spécifiez le nom du fichier.

Pour analyser un document d'identité (API)

1. Si vous ne l'avez pas déjà fait :

   a. Créer ou mettre à jour un utilisateur IAM avec`AmazonTextractFullAccess`et`AmazonS3ReadOnlyAccess`Autorisations. Pour plus d'informations, consultez Étape 1 : Configuration d'un compte AWS et création d'un utilisateur IAM.

   b. Installez et configurez l'AWS CLI et les kits SDK AWS. Pour plus d'informations, consultez Étape 2 : Configuration de l'AWS CLIetAWSKits SDK.

2. Chargez une image qui contient un document dans votre compartiment S3.

   Pour obtenir des instructions, consultezChargement d'objets dans Amazon S3dans leManuel de l'utilisateur Amazon Simple Storage Service.

3. Utilisez les exemples suivants pour appeler l'opération `AnalyzeID`.

   CLI

   L'exemple suivant prend en charge un fichier d'entrée à partir d'un compartiment S3 et exécute le`AnalyzeID`Opération sur elle. Dans le code ci-dessous, remplacez la valeur de*seau*avec le nom de votre compartiment S3, la valeur de*fichier*avec le nom du fichier dans votre compartiment et la valeur de*région*avec le nom du`region`associé à votre compte.

   ```
   aws textract analyze-id --document-pages '[{"S3Object":
   {"Bucket":"bucket","Name":"name"}}]' --region region
   ```

   Vous pouvez également appeler l'API à l'avant et à l'arrière d'un permis de conduire en ajoutant un autre objet S3 à l'entrée.

```
aws textract analyze-id --document-pages '[{"S3Object":
{"Bucket":"bucket","Name":"name front"}}, {"S3Object":
{"Bucket":"bucket","Name":"name back"}}]' --region us-east-1
```

Si vous accédez à l'interface de ligne de commande sur un appareil Windows, utilisez des guillemets doubles au lieu de guillemets simples et échappez aux guillemets doubles internes par une barre oblique inverse (c'est-à-dire \) pour résoudre les erreurs d'analyseur que vous pourriez rencontrer. Pour un exemple, veuillez consulter ci-dessous :

```
aws textract analyze-id --document-pages "[{\"S3Object\":{\"Bucket\":\"bucket\",
\"Name\":\"name\"}}]" --region region
```

Python

L'exemple suivant prend en charge un fichier d'entrée à partir d'un compartiment S3 et exécute le`AnalyzeID`en renvoyant les paires clé-valeur détectées. Dans le code ci-dessous, remplacez la valeur de*bucket_name*avec le nom de votre compartiment S3, la valeur de*file_name*avec le nom du fichier dans votre compartiment et la valeur de*région*avec le nom du`region`associé à votre compte.

```
import boto3

bucket_name = "bucket-name"
file_name = "file-name"
region = "region-name"

def analyze_id(region, bucket_name, file_name):

    textract_client = boto3.client('textract', region_name=region)
    response = textract_client.analyze_id(DocumentPages=[{"S3Object":
{"Bucket":bucket_name,"Name":file_name}}])

    for doc_fields in response['IdentityDocuments']:
        for id_field in doc_fields['IdentityDocumentFields']:
            for key, val in id_field.items():
                if "Type" in str(key):
                    print("Type: " + str(val['Text']))
            for key, val in id_field.items():
                if "ValueDetection" in str(key):
                    print("Value Detection: " + str(val['Text']))
            print()
```

```
analyze_id(region, bucket_name, file_name)
```

Java

L'exemple suivant prend en charge un fichier d'entrée à partir d'un compartiment S3 et exécute le`AnalyzeID`opération sur elle, renvoyant les données détectées. Dans la fonction principale, remplacez les valeurs des`s3bucket`et`sourceDoc`avec les noms du compartiment Amazon S3 et de l'image de document utilisés à l'étape 2.

```
/*
   Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
   SPDX-License-Identifier: Apache-2.0
*/

package com.amazonaws.samples;

import com.amazonaws.regions.Regions;
import com.amazonaws.services.textract.AmazonTextractClient;
import com.amazonaws.services.textract.AmazonTextractClientBuilder;
import com.amazonaws.services.textract.model.*;
import java.util.ArrayList;
import java.util.List;

public class AnalyzeIdentityDocument {

    public static void main(String[] args) {

        final String USAGE = "\n" +
                "Usage:\n" +
                "    <s3bucket><sourceDoc> \n\n" +
                "Where:\n" +
                "    s3bucket - the Amazon S3 bucket where the document is
 located. \n" +
                "    sourceDoc - the name of the document. \n";

        if (args.length != 1) {
            System.out.println(USAGE);
            System.exit(1);
        }

        String s3bucket = "bucket-name"; //args[0];
        String sourceDoc = "sourcedoc-name";  //args[1];
```

```
       AmazonTextractClient textractClient = (AmazonTextractClient)
AmazonTextractClientBuilder.standard()
               .withRegion(Regions.US_EAST_1)
               .build();

       getDocDetails(textractClient, s3bucket, sourceDoc);
   }

   public static void getDocDetails(AmazonTextractClient textractClient, String
s3bucket, String sourceDoc ) {

      try {

           S3Object s3 = new S3Object();
           s3.setBucket(s3bucket);
           s3.setName(sourceDoc);

           com.amazonaws.services.textract.model.Document myDoc = new
com.amazonaws.services.textract.model.Document();
           myDoc.setS3Object(s3);

           List<Document> list1 = new ArrayList();
           list1.add(myDoc);

           AnalyzeIDRequest idRequest = new AnalyzeIDRequest();
           idRequest.setDocumentPages(list1);

           AnalyzeIDResult result = textractClient.analyzeID(idRequest);
           List<IdentityDocument> docs =  result.getIdentityDocuments();
           for (IdentityDocument doc: docs) {

               List<IdentityDocumentField>idFields =
doc.getIdentityDocumentFields();
               for (IdentityDocumentField field: idFields) {
                   System.out.println("Field type is "+
field.getType().getText());
                   System.out.println("Field value is "+
field.getValueDetection().getText());
               }
           }

      } catch (Exception e) {
           e.printStackTrace();
      }
```

```
    }
}
```

4. Vous obtiendrez ainsi la sortie JSON pour`AnalyzeID`.

Traitement de documents avec des opérations asynchrones

Amazon Textract peut détecter et analyser du texte dans des documents multipages au format PDF ou TIFF. Cela inclut les factures et les reçus. Le traitement de documents multipages est une opération asynchrone. Le traitement asynchrone de documents est utile pour le traitement de documents volumineux sur plusieurs pages. Par exemple, un fichier PDF de plus de 1 000 pages prend un certain temps à traiter. Le traitement asynchrone du fichier PDF permet à votre application d'effectuer d'autres tâches pendant qu'elle attend la fin du processus.

Cette section explique comment utiliser Amazon Textract pour détecter et analyser de manière asynchrone du texte sur un document multipage ou une seule page. Les documents multipages doivent être au format PDF ou TIFF. Les documents d'une page traités avec des opérations asynchrones peuvent être au format JPEG, PNG, TIFF ou PDF.

Vous pouvez utiliser les opérations asynchrone Amazon Textract aux fins suivantes :

- Détection de texte : vous pouvez détecter des lignes et des mots sur un document multipage. Les opérations asynchrones sont les suivantes :StartDocumentTextDetectionetGetDocumentTextDetection. Pour plus d'informations, consultez Détection de texte.
- Analyse de texte : vous pouvez identifier les relations entre le texte détecté sur un document multipage. Les opérations asynchrones sont les suivantes :StartDocumentAnalysisetGetDocumentAnalysis. Pour plus d'informations, consultez Analyse des documents.
- Analyse des dépenses : vous pouvez identifier les relations de données sur des factures et des reçus multipages. Amazon Textract traite chaque facture ou page de réception d'un document de plusieurs pages comme un reçu individuel ou une facture. Il ne conserve pas le contexte d'une page à une autre d'un document multi-pages. Les opérations asynchrones sont les suivantes :StartExpenseAnalysisetGetExpenseAnalysis. Pour plus d'informations, consultez Analyse des factures et des reçus.

Rubriques

Appel d'opérations asynchrones Amazon Textract

Amazon Textract fournit une API asynchrone que vous pouvez utiliser pour traiter des documents multipages au format PDF ou TIFF. Vous pouvez également utiliser des opérations asynchrones pour traiter des documents d'une seule page au format JPEG, PNG, TIFF ou PDF.

Les informations de cette rubrique utilisent des opérations de détection de texte pour montrer comment utiliser les opérations asynchrones Amazon Textract. La même approche fonctionne avec les opérations d'analyse de texte dethe section called “StartDocumentAnalysis”etthe section called “GetDocumentAnalysis”. Il fonctionne également de la même manière avecthe section called “StartExpenseAnalysis”etthe section called “GetExpenseAnalysis”.

Pour voir un exemple, consultez Détection ou analyse de texte dans un document multipage.

Amazon Textract traite de manière asynchrone un document stocké dans un compartiment Amazon S3. Vous commencez le traitement en appelant un`Start`opération, telle queStartDocumentTextDetection. Le statut d'achèvement de la demande est publié dans une rubrique Amazon Simple Notification Service (Amazon SNS). Pour obtenir le statut d'achèvement dans la rubrique Amazon SNS, vous pouvez utiliser une file d'attente Amazon Simple Queue Service (Amazon SQS) ou uneAWS Lambda. Après avoir obtenu le statut d'achèvement, vous appelez une opération `Get` telle que GetDocumentTextDetection pour obtenir le résultat de la demande.

Les résultats des appels asynchrones sont chiffrés et stockés pendant 7 jours dans un compartiment appartenant à Amazon Textract par défaut, sauf si vous spécifiez un compartiment Amazon S3 à l'aide d'une opération`OutputConfig`argument.

Le tableau suivant présente les opérations Démarrer et Obtenir correspondantes pour les différents types de traitement asynchrone pris en charge par Amazon Textract :

Opérations d'API Start/Get pour Amazon Textract Asynchronous Operations

Type de traitement	API de démarrage	Obtenez l'API
Détection de texte	StartDocumentTextDetection	GetDocumentTextDetection
Analyse de texte	StartDocumentAnalysis	GetDocumentAnalysis
Analyse des dépenses	Commencer l'analyse des dépenses	Obtenez une analyse des dépenses

Pour un exemple qui utiliseAWS Lambdafonctions, consultezTraitement de documents à grande échelle avec Amazon Textract.

Le schéma suivant illustre le processus de détection de texte dans une image de document stockée dans un compartiment Amazon S3. Dans ce schéma, une file d'attente Amazon SQS obtient le statut d'achèvement à partir de la rubrique Amazon SNS.

Le processus affiché par le diagramme précédent est le même pour l'analyse du texte et des factures/reçus. Vous commencez à analyser du texte en appelantthe section called "StartDocumentAnalysis"et commencez à analyser les factures/reçus en appelantthe section called "StartExpenseAnalysis"Vous obtenez les résultats en appelantthe section called "GetDocumentAnalysis"outhe section called "GetExpenseAnalysis"respectivement.

Détection de texte

Vous lancez une demande de détection de texte Amazon Textract en appelantStartDocumentTextDetection. L'exemple suivant est une demande JSON transmise par `StartDocumentTextDetection`.

```
{
    "DocumentLocation": {
        "S3Object": {
            "Bucket": "bucket",
            "Name": "image.pdf"
        }
    },
    "ClientRequestToken": "DocumentDetectionToken",
    "NotificationChannel": {
        "SNSTopicArn": "arn:aws:sns:us-east-1:nnnnnnnnnn:topic",
        "RoleArn": "arn:aws:iam::nnnnnnnnnn:role/roleTopic"
    },
    "JobTag": "Receipt"
}
```

Le paramètre d'entrée`DocumentLocation`fournit le nom du fichier de document et le compartiment Amazon S3 dont ce fichier peut être extrait.`NotificationChannel`contient l'Amazon Resource Name (ARN) de la rubrique Amazon SNS que Amazon Textract informe lorsque la demande de détection de texte est terminée. La rubrique Amazon SNS doit se trouver dans la même région AWS que le point de terminaison Amazon Textract que vous appelez.`NotificationChannel`contient

également l'ARN pour un rôle qui permet à Amazon Textract de publier dans la rubrique Amazon SNS. Vous attribuez des autorisations de publication Amazon Textract à vos rubriques Amazon SNS en créant un rôle de service IAM. Pour plus d'informations, consultez Configuration d'Amazon Textract pour les opérations asynchrones.

Vous pouvez également spécifier un paramètre d'entrée en option,`JobTag`, qui vous permet d'identifier la tâche ou les groupes de tâches dans le statut d'achèvement publié dans la rubrique Amazon SNS. Par exemple, vous pouvez utiliser`JobTag`pour identifier le type de document traité, tel qu'un formulaire fiscal ou un reçu.

Afin d'éviter toute duplication accidentelle des tâches d'analyse, vous pouvez, si vous le souhaitez, fournir un jeton idempotent, `ClientRequestToken`. Si vous fournissez une valeur pour`ClientRequestToken`, le`Start`renvoie la même opération`JobId`pour plusieurs appels identiques à la`Start`opération, telle que`StartDocumentTextDetection`. Un jeton `ClientRequestToken` a une durée de vie de 7 jours. Au delà de 7 jours, vous pouvez le réutiliser. Si vous réutilisez le jeton pendant sa durée de vie, ce qui suit se produit :

- Si vous réutilisez le jeton avec la même opération `Start` et les mêmes paramètres d'entrée, le même `JobId` est renvoyé. La tâche n'est pas exécutée à nouveau et Amazon Textract n'envoie pas de statut d'achèvement à la rubrique Amazon SNS enregistrée.
- Si vous réutilisez le jeton avec la même opération `Start` et une modification mineure du paramètre d'entrée, vous obtenez une exception `idempotentparametermismatchexception` (code de statut HTTP : 400).
- Si vous réutilisez le jeton avec une autre opération `Start`, l'opération aboutit.

Un autre paramètre optionnel disponible est :`OutputConfig`, qui vous permet de régler l'endroit où votre sortie sera placée. Par défaut, Amazon Textract stocke les résultats en interne et n'est accessible que par les opérations Obtenir l'API. avec`OutputConfig`activé, vous pouvez définir le nom du compartiment vers lequel la sortie sera envoyée et le préfixe de fichier des résultats, dans lequel vous pouvez télécharger vos résultats. De plus, vous pouvez définir le`KMSKeyID`d'une clé gérée par le client pour chiffrer votre sortie. Sans ce jeu de paramètres, Amazon Textract chiffrera côté serveur à l'aide de la commandeClé gérée par AWSpour Amazon S3

> **Note**
>
> Avant d'utiliser ce paramètre, assurez-vous que vous disposez de l'autorisation PutObject pour le compartiment de sortie. En outre, assurez-vous que vous disposez des autorisations

Decrypt, ReEncrypt, GenerateDataKey et DescribeKey pour leAWS KMSsi vous décidez de l'utiliser.

La réponse à l'opération `StartDocumentTextDetection` est un identifiant de tâche (`JobId`). Utiliser`JobId`pour suivre les demandes et obtenir les résultats d'analyse une fois qu'Amazon Textract a publié le statut d'achèvement dans la rubrique Amazon SNS. Voici un exemple :

```
{"JobId":"270c1cc5e1d0ea2fbc59d97cb69a72a5495da75851976b14a1784ca90fc180e3"}
```

Si vous commencez trop de tâches simultanément, appelez à`StartDocumentTextDetection`RAISE UN`LimitExceededException`exception (code de statut HTTP : 400) jusqu'à ce que le nombre de tâches exécutées simultanément soit inférieur à la limite de service Amazon Textract.

Si vous constatez que les exceptions LimitExceededException sont générées lors des périodes d'activité, vous devez envisager d'utiliser une file d'attente Amazon SQS afin de gérer les demandes entrantes. ContacterAWSSupport si vous constatez que le nombre moyen de demandes simultanées ne peut pas être géré par une file d'attente Amazon SQS et que vous continuez à recevoir`LimitExceededException`exceptions.

Obtention du statut d'achèvement d'une demande d'analyse Amazon Textract

Amazon Textract envoie une notification d'achèvement d'analyse à la rubrique Amazon SNS enregistrée. La notification comprend l'identifiant de la tâche et le statut d'achèvement de l'opération dans une chaîne JSON. Une demande de détection de texte réussie comporte un`SUCCEEDED`État. Par exemple, le résultat suivant illustre la réussite du traitement d'une tâche de détection de texte.

```
{
    "JobId": "642492aea78a86a40665555dc375ee97bc963f342b29cd05030f19bd8fd1bc5f",
    "Status": "SUCCEEDED",
    "API": "StartDocumentTextDetection",
    "JobTag": "Receipt",
    "Timestamp": 1543599965969,
    "DocumentLocation": {
        "S3ObjectName": "document",
        "S3Bucket": "bucket"
    }
```

```
}
```

Pour plus d'informations, consultez Notification des résultats Amazon Textract.

Pour obtenir les informations de statut publiées dans la rubrique Amazon SNS par Amazon Textract, utilisez l'une des options suivantes :

- AWS Lambda— vous pouvez abonner unAWS Lambdaque vous écrivez dans une rubrique Amazon SNS. La fonction est appelée quand Amazon Textract informe la rubrique Amazon SNS que la demande est terminée. Utilisez une fonction Lambda si vous souhaitez que le code côté serveur traite les résultats d'une demande de détection de texte. Par exemple, vous pouvez utiliser du code côté serveur pour annoter l'image ou créer un rapport sur le texte détecté avant de renvoyer les informations vers une application cliente.
- Amazon SQS— Vous pouvez abonner une file d'attente Amazon SQS à une rubrique Amazon SNS. Vous pouvez ensuite interroger la file d'attente Amazon SQS pour extraire le statut d'achèvement publié par Amazon Textract lorsqu'une demande de détection de texte se termine. Pour plus d'informations, consultez Détection ou analyse de texte dans un document multipage. Utilisez une file d'attente Amazon SQS si vous souhaitez appeler des opérations Amazon Textract uniquement à partir d'une application cliente.

> **Important**
>
> Nous vous déconseillons d'obtenir le statut d'achèvement de la demande en appelant de manière répétée le Amazon Textract`Get`. Cela est dû au fait qu'Amazon Textract limite le`Get`si de trop nombreuses demandes sont lancées. Si vous traitez plusieurs documents simultanément, il est plus simple et plus efficace de surveiller la création d'une notification d'achèvement dans une file d'attente SQS que d'interroger Amazon Textract pour obtenir le statut de chaque tâche individuellement.

Obtention des résultats de détection de texte Amazon Textract

Pour obtenir les résultats d'une demande de détection de texte, assurez-vous tout d'abord que le statut d'achèvement extrait de la rubrique Amazon SNS est`SUCCEEDED`. Ensuite, appelez `GetDocumentTextDetection`, qui transmet la valeur `JobId` renvoyée par `StartDocumentTextDetection`. Le format JSON de la demande est similaire à l'exemple suivant :

```
{
    "JobId": "270c1cc5e1d0ea2fbc59d97cb69a72a5495da75851976b14a1784ca90fc180e3",
    "MaxResults": 10,
    "SortBy": "TIMESTAMP"
}
```

`JobId`est l'identificateur de l'opération de détection de texte. Étant donné que la détection de texte peut générer de grandes quantités de données, utilisez`MaxResults`pour spécifier le nombre maximal de résultats à renvoyer en une seule fois`Get`. La valeur par défaut de`MaxResults`est de 1 000. Si vous spécifiez une valeur supérieure à 1 000, seuls 1 000 résultats sont renvoyés. Si l'opération ne renvoie pas tous les résultats, un jeton de pagination pour la page suivante est renvoyé. Pour obtenir la page de résultats suivante, spécifiez le jeton dans la page de résultats`NextToken` Paramètre .

> **Note**
>
> Amazon Textract conserve les résultats d'opérations asynchrone pendant 7 jours. Passé ce délai, vous ne pouvez pas récupérer les résultats.

Le`GetDocumentTextDetection`JSON d'une réponse d'opération est similaire à ce qui suit. Le nombre total de pages détectées est renvoyé dans`DocumentMetadata`. Le texte détecté est renvoyé dans le`Blocks`tableau. Pour obtenir des informations sur`Block`objets, voirObjets de réponse Détection de texte et analyse de documents.

```
{
    "DocumentMetadata": {
        "Pages": 1
    },
    "JobStatus": "SUCCEEDED",
    "Blocks": [
        {
            "BlockType": "PAGE",
            "Geometry": {
                "BoundingBox": {
                    "Width": 1.0,
                    "Height": 1.0,
                    "Left": 0.0,
                    "Top": 0.0
                },
```

```
                "Polygon": [
                    {
                        "X": 0.0,
                        "Y": 0.0
                    },
                    {
                        "X": 1.0,
                        "Y": 0.0
                    },
                    {
                        "X": 1.0,
                        "Y": 1.0
                    },
                    {
                        "X": 0.0,
                        "Y": 1.0
                    }
                ]
            },
            "Id": "64533157-c47e-401a-930e-7ca1bb3ac3fa",
            "Relationships": [
                {
                    "Type": "CHILD",
                    "Ids": [
                        "4297834d-dcb1-413b-8908-3b96866ebbb5",
                        "1d85ba24-2877-4d09-b8b2-393833d769e9",
                        "193e9c47-fd87-475a-ba09-3fda210d8784",
                        "bd8aeb62-961b-4b47-b78a-e4ed9eeecd0f"
                    ]
                }
            ],
            "Page": 1
        },
        {
            "BlockType": "LINE",
            "Confidence": 53.301639556884766,
            "Text": "ellooworio",
            "Geometry": {
                "BoundingBox": {
                    "Width": 0.9999999403953552,
                    "Height": 0.5365243554115295,
                    "Left": 0.0,
                    "Top": 0.46347561478614807
                },
```

```
                "Polygon": [
                    {
                        "X": 0.0,
                        "Y": 0.46347561478614807
                    },
                    {
                        "X": 0.9999999403953552,
                        "Y": 0.46347561478614807
                    },
                    {
                        "X": 0.9999999403953552,
                        "Y": 1.0
                    },
                    {
                        "X": 0.0,
                        "Y": 1.0
                    }
                ]
            },
            "Id": "4297834d-dcb1-413b-8908-3b96866ebbb5",
            "Relationships": [
                {
                    "Type": "CHILD",
                    "Ids": [
                        "170c3eb9-5155-4bec-8c44-173bba537e70"
                    ]
                }
            ],
            "Page": 1
        },
        {
            "BlockType": "LINE",
            "Confidence": 89.15632629394531,
            "Text": "He llo,",
            "Geometry": {
                "BoundingBox": {
                    "Width": 0.33642634749412537,
                    "Height": 0.49159330129623413,
                    "Left": 0.13885067403316498,
                    "Top": 0.17169663310050964
                },
                "Polygon": [
                    {
                        "X": 0.13885067403316498,
```

```
                    "Y": 0.17169663310050964
                },
                {
                    "X": 0.47527703642845154,
                    "Y": 0.17169663310050964
                },
                {
                    "X": 0.47527703642845154,
                    "Y": 0.6632899641990662
                },
                {
                    "X": 0.13885067403316498,
                    "Y": 0.6632899641990662
                }
            ]
        },
        "Id": "1d85ba24-2877-4d09-b8b2-393833d769e9",
        "Relationships": [
            {
                "Type": "CHILD",
                "Ids": [
                    "516ae823-3bab-4f9a-9d74-ad7150d128ab",
                    "6bcf4ea8-bbe8-4686-91be-b98dd63bc6a6"
                ]
            }
        ],
        "Page": 1
    },
    {
        "BlockType": "LINE",
        "Confidence": 82.44834899902344,
        "Text": "worlo",
        "Geometry": {
            "BoundingBox": {
                "Width": 0.33182239532470703,
                "Height": 0.3766750991344452,
                "Left": 0.5091826915740967,
                "Top": 0.23131252825260162
            },
            "Polygon": [
                {
                    "X": 0.5091826915740967,
                    "Y": 0.23131252825260162
                },
```

```
                    {
                        "X": 0.8410050868988037,
                        "Y": 0.23131252825260162
                    },
                    {
                        "X": 0.8410050868988037,
                        "Y": 0.607987642288208
                    },
                    {
                        "X": 0.5091826915740967,
                        "Y": 0.607987642288208
                    }
                ]
            },
            "Id": "193e9c47-fd87-475a-ba09-3fda210d8784",
            "Relationships": [
                {
                    "Type": "CHILD",
                    "Ids": [
                        "ed135c3b-35dd-4085-8f00-26aedab0125f"
                    ]
                }
            ],
            "Page": 1
        },
        {
            "BlockType": "LINE",
            "Confidence": 88.50325775146484,
            "Text": "world",
            "Geometry": {
                "BoundingBox": {
                    "Width": 0.35004907846450806,
                    "Height": 0.19635874032974243,
                    "Left": 0.527581512928009,
                    "Top": 0.30100569128990173
                },
                "Polygon": [
                    {
                        "X": 0.527581512928009,
                        "Y": 0.30100569128990173
                    },
                    {
                        "X": 0.8776305913925171,
                        "Y": 0.30100569128990173
```

```
                    },
                    {
                        "X": 0.8776305913925171,
                        "Y": 0.49736443161964417
                    },
                    {
                        "X": 0.527581512928009,
                        "Y": 0.49736443161964417
                    }
                ]
            },
            "Id": "bd8aeb62-961b-4b47-b78a-e4ed9eeecd0f",
            "Relationships": [
                {
                    "Type": "CHILD",
                    "Ids": [
                        "9e28834d-798e-4a62-8862-a837dfd895a6"
                    ]
                }
            ],
            "Page": 1
        },
        {
            "BlockType": "WORD",
            "Confidence": 53.301639556884766,
            "Text": "ellooworio",
            "Geometry": {
                "BoundingBox": {
                    "Width": 1.0,
                    "Height": 0.5365243554115295,
                    "Left": 0.0,
                    "Top": 0.46347561478614807
                },
                "Polygon": [
                    {
                        "X": 0.0,
                        "Y": 0.46347561478614807
                    },
                    {
                        "X": 1.0,
                        "Y": 0.46347561478614807
                    },
                    {
                        "X": 1.0,
```

```
                    "Y": 1.0
                },
                {
                    "X": 0.0,
                    "Y": 1.0
                }
            ]
        },
        "Id": "170c3eb9-5155-4bec-8c44-173bba537e70",
        "Page": 1
    },
    {
        "BlockType": "WORD",
        "Confidence": 88.46246337890625,
        "Text": "He",
        "Geometry": {
            "BoundingBox": {
                "Width": 0.15350718796253204,
                "Height": 0.29955607652664185,
                "Left": 0.13885067403316498,
                "Top": 0.21856294572353363
            },
            "Polygon": [
                {
                    "X": 0.13885067403316498,
                    "Y": 0.21856294572353363
                },
                {
                    "X": 0.292357861995697,
                    "Y": 0.21856294572353363
                },
                {
                    "X": 0.292357861995697,
                    "Y": 0.5181190371513367
                },
                {
                    "X": 0.13885067403316498,
                    "Y": 0.5181190371513367
                }
            ]
        },
        "Id": "516ae823-3bab-4f9a-9d74-ad7150d128ab",
        "Page": 1
    },
```

```
        {
            "BlockType": "WORD",
            "Confidence": 89.8501968383789,
            "Text": "llo,",
            "Geometry": {
                "BoundingBox": {
                    "Width": 0.17724157869815826,
                    "Height": 0.49159327149391174,
                    "Left": 0.2980354428291321,
                    "Top": 0.17169663310050964
                },
                "Polygon": [
                    {
                        "X": 0.2980354428291321,
                        "Y": 0.17169663310050964
                    },
                    {
                        "X": 0.47527703642845154,
                        "Y": 0.17169663310050964
                    },
                    {
                        "X": 0.47527703642845154,
                        "Y": 0.6632899045944214
                    },
                    {
                        "X": 0.2980354428291321,
                        "Y": 0.6632899045944214
                    }
                ]
            },
            "Id": "6bcf4ea8-bbe8-4686-91be-b98dd63bc6a6",
            "Page": 1
        },
        {
            "BlockType": "WORD",
            "Confidence": 82.44834899902344,
            "Text": "worlo",
            "Geometry": {
                "BoundingBox": {
                    "Width": 0.33182239532470703,
                    "Height": 0.3766750991344452,
                    "Left": 0.5091826915740967,
                    "Top": 0.23131252825260162
                },
```

```
                "Polygon": [
                    {
                        "X": 0.5091826915740967,
                        "Y": 0.23131252825260162
                    },
                    {
                        "X": 0.8410050868988037,
                        "Y": 0.23131252825260162
                    },
                    {
                        "X": 0.8410050868988037,
                        "Y": 0.607987642288208
                    },
                    {
                        "X": 0.5091826915740967,
                        "Y": 0.607987642288208
                    }
                ]
            },
            "Id": "ed135c3b-35dd-4085-8f00-26aedab0125f",
            "Page": 1
        },
        {
            "BlockType": "WORD",
            "Confidence": 88.50325775146484,
            "Text": "world",
            "Geometry": {
                "BoundingBox": {
                    "Width": 0.35004907846450806,
                    "Height": 0.19635874032974243,
                    "Left": 0.527581512928009,
                    "Top": 0.30100569128990173
                },
                "Polygon": [
                    {
                        "X": 0.527581512928009,
                        "Y": 0.30100569128990173
                    },
                    {
                        "X": 0.8776305913925171,
                        "Y": 0.30100569128990173
                    },
                    {
                        "X": 0.8776305913925171,
```

```
                        "Y": 0.49736443161964417
                    },
                    {
                        "X": 0.527581512928009,
                        "Y": 0.49736443161964417
                    }
                ]
            },
            "Id": "9e28834d-798e-4a62-8862-a837dfd895a6",
            "Page": 1
        }
    ]
}
```


Configuration d'Amazon Textract pour les opérations asynchrones

Les procédures suivantes vous montrent comment configurer Amazon Textract pour qu'il soit utilisé avec une rubrique Amazon Simple Notification Service (Amazon SNS) et une file d'attente Amazon Simple Queue Service (Amazon SQS).

> **Note**
>
> Si vous utilisez ces instructions pour configurer l'Détection ou analyse de texte dans un document multipagePar exemple, vous n'avez pas besoin de suivre les étapes 3 à 6. L'exemple inclut le code permettant de créer et de configurer la rubrique Amazon SNS et la file d'attente Amazon SQS.

Pour configurer Amazon Textract

1. Configurez uneAWSpour accéder à Amazon Textract. Pour plus d'informations, consultez Étape 1 : Configuration d'un compte AWS et création d'un utilisateur IAM.

   Assurez-vous que l'utilisateur possède au moins les autorisations suivantes :

   - AmazonTextractFullAccess
   - AmazonS3ReadOnlyAccess
   - AmazonSNSFullAccess
   - AmazonSQSFullAccess

2. Installez et configurez le kit SDK AWS requis. Pour plus d'informations, consultez Étape 2 : Configuration de l'AWS CLIetAWSKits SDK.
3. Créer une rubrique Amazon SNS. Préfixez le nom de rubrique avecExtrait Amazon. Notez l'Amazon Resource Name (ARN) de la rubrique. Assurez-vous que la rubrique se trouve dans la même région que la rubriqueAWSpoint de terminaison que vous utilisez avec votre compte AWS.
4. Créez une file d'attente standard Amazon SQSen utilisant leConsole Amazon SQS. Notez l'ARN de la file d'attente.
5. Abonnez la file d'attente à la rubrique que vous avez créée à l'étape 3.
6. Autoriser la rubrique Amazon SNS à envoyer des messages à la file d'attente Amazon SQS.
7. Créez un rôle de service IAM pour accorder à Amazon Textract l'autorisation d'accès à vos rubriques Amazon SNS. Notez l'Amazon Resource Name (ARN) du rôle de service. Pour plus d'informations, consultez Donner à Amazon Textract l'autorisation d'accès à votre rubrique Amazon SNS.
8. Ajoutez la stratégie en ligne ci-dessousà l'utilisateur IAM que vous avez créé à l'étape 1.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "MySid",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "Service role ARN from step 7"
        }
    ]
}
```

   Attribuez un nom à la stratégie en ligne.
9. Vous pouvez désormais exécuter les exemples dansDétection ou analyse de texte dans un document multipage.

Donner à Amazon Textract l'autorisation d'accès à votre rubrique Amazon SNS

Amazon Textract doit être autorisé à envoyer un message à votre rubrique Amazon SNS lorsqu'une opération asynchrone est terminée. Vous utilisez un rôle de service IAM pour accorder à Amazon Textract l'autorisation d'accès à la rubrique Amazon SNS.

Lorsque vous créez la rubrique Amazon SNS, vous devez ajouter le nom de la rubrique avec**`AmazonTextract`**—par exemple,**`AmazonTextractMyTopicName`**.

1. Connectez-vous à la console IAM (https://console.aws.amazon.com/iam).
2. Dans le panneau de navigation, choisissez Rôles.
3. Sélectionnez Créer un rôle.
4. Pour Select type of trusted entity (Sélectionner le type d'entité de confiance), choisissez Service AWS.
5. PourChoisissez le service qui utilisera ce rôle, choisissezTextract.
6. Choisissez Next (Suivant) Permissions (Autorisations).
7. Vérifiez que leAmazonTextractServiceRolea été incluse dans la liste des stratégies attachées. Pour afficher la stratégie dans la liste, tapez une partie du nom de la stratégie dans la listeStratégies de filtre.
8. Choisissez Next (Suivant) Tags (Balises).
9. Vous n'avez pas besoin d'ajouter de tags, choisissez doncSuivant: Review (Examiner).
10. Dans la section Vérification, pour role name (Nom du rôle), entrez un nom pour le rôle (par exemple, `TextractRole`). DansDescription du rôle, mettez à jour la description du rôle, puis choisissezCréation d'un rôle.
11. Choisissez le nouveau rôle pour ouvrir la page des détails du rôle.
12. Dans le Summary (Récapitulatif), copiez la valeur de l'ARN de rôle et enregistrez-la.
13. Choisissez Trust Relationships.
14. ChoisissezModifier la relation d'approbation, et veiller à ce que la politique de confiance se présente comme suit.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Principal": {
        "Service": "textract.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

15. Choisissez Update Trust Policy (Mettre à jour la politique d'approbation).

Détection ou analyse de texte dans un document multipage

Cette procédure montre comment détecter ou analyser du texte dans un document multipage à l'aide des opérations de détection Amazon Textract, un document stocké dans un compartiment Amazon S3, une rubrique Amazon SNS et une file d'attente Amazon SQS. Le traitement de documents multipages est une opération asynchrone. Pour plus d'informations, consultez Appel d'opérations asynchrones Amazon Textract.

Vous pouvez choisir le type de traitement que le code doit effectuer : détection de texte, analyse de texte ou analyse des dépenses.

Les résultats du traitement sont renvoyés dans un tableau dethe section called “Block”objets, qui sont différents en fonction du type de traitement que vous utilisez.

Pour détecter du texte dans des documents multipages ou les analyser, procédez comme suit :

1. Créez la rubrique Amazon SNS et la file d'attente Amazon SQS.
2. Abonnez la file d'attente à la rubrique.
3. Autorisez la rubrique à envoyer des messages à la file d'attente.
4. Commencez à traiter le document. Utilisez l'opération appropriée pour le type d'analyse que vous avez choisi :
   - StartDocumentTextDetectionpour les tâches de détection de texte.
   - StartDocumentAnalysispour les tâches d'analyse de texte.
   - StartExpenseAnalysispour les tâches d'analyse des dépenses.
5. Obtenir le statut d'achèvement à partir de la file d'attente Amazon SQS. L'exemple de code suit l'identificateur de la tâche (`JobId`) qui est retourné par le`Start`. Il obtient uniquement les résultats pour les identifiants de tâche correspondants lus à partir du statut d'achèvement. Cela est

important si d'autres applications utilisent la même file d'attente et la même rubrique. Par souci de simplicité, l'exemple supprime les tâches qui ne correspondent pas. Envisagez d'ajouter les tâches supprimées à une file d'attente de lettres mortes Amazon SQS pour un examen plus approfondi.

6. Obtenez et affichez les résultats du traitement en appelant l'opération appropriée pour le type d'analyse que vous avez choisi :
   - GetDocumentTextDetectionpour les tâches de détection de texte.
   - GetDocumentAnalysispour les tâches d'analyse de texte.
   - GetExpenseAnalysispour les tâches d'analyse des dépenses.
7. Supprimez la rubrique Amazon SNS et la file d'attente Amazon SQS.

Exécution d'opérations asynchrones

L'exemple de code pour cette procédure est fourni en Java, Python et le fichierAWS CLI. Avant de commencer, installez leAWSSDK. Pour plus d'informations, consultez Étape 2 : Configuration de l'AWS CLIetAWSKits SDK.

Pour détecter ou analyser du texte dans un document multipage

1. Configurez l'accès des utilisateurs à Amazon Textract et configurez l'accès Amazon Textract à Amazon SNS. Pour plus d'informations, consultez Configuration d'Amazon Textract pour les opérations asynchrones. Pour suivre cette procédure, vous devez disposer d'un fichier de documents multipages au format PDF. Ignorez les étapes 3 à 6, car l'exemple de code crée et configure la rubrique Amazon SNS et la file d'attente Amazon SQS. S'il est complexetDans l'exemple de l'interface de ligne de commande, vous n'avez pas besoin de configurer une file d'attente SQS.
2. Chargez un fichier de documents multipages au format PDF ou TIFF dans votre compartiment Amazon S3. (Les documents d'une page au format JPEG, PNG, TIFF ou PDF peuvent également être traités).

   Pour obtenir des instructions, consultezChargement d'objets dans Amazon S3dans leManuel de l'utilisateur Amazon Simple Storage Service.
3. Utilisez ce qui suitAWS SDK pour Java, kit SDK for Python (Boto3) ouAWS CLIcode permettant de détecter du texte ou d'analyser du texte dans un document multipage. Dans`main`fonction :

   - Remplacez la valeur de`roleArn`avec l'ARN du rôle IAM que vous avez enregistré dansDonner à Amazon Textract l'autorisation d'accès à votre rubrique Amazon SNS.

- Remplacez les valeurs de`bucket`et`document`avec le nom du fichier de compartiment et le nom du document que vous avez spécifiés à l'étape 2.
- Remplacez la valeur de l'`type`paramètre d'entrée du paramètre`ProcessDocument`avec le type de traitement que vous souhaitez effectuer. Utiliser`ProcessType.DETECTION`pour détecter du texte. Utiliser`ProcessType.ANALYSIS`pour analyser du texte.
- Pour l'exemple Python, remplacez la valeur de`region_name`avec la région dans laquelle votre client opère.

PourAWS CLIPar exemple, procédez comme suit :

- Lorsque vous appelezStartDocumentTextDetection, remplacez la valeur de`bucket-name`avec le nom de votre compartiment S3 et remplacez`file-name`Avec le nom du fichier que vous avez spécifié à l'étape 2. Spécifiez la région de votre compartiment en remplaçant`region-name`avec le nom de votre région. Notez que l'exemple CLI n'utilise pas SQS.
- Lorsque vous appelezGetDocumentTextDetectionremplacer`job-id-number`avec le`job-id`renvoyé parStartDocumentTextDetection. Spécifiez la région de votre compartiment en remplaçant`region-name`avec le nom de votre région.

Java

```
package com.amazonaws.samples;

import java.util.Arrays;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import com.amazonaws.auth.policy.Condition;
import com.amazonaws.auth.policy.Policy;
import com.amazonaws.auth.policy.Principal;
import com.amazonaws.auth.policy.Resource;
import com.amazonaws.auth.policy.Statement;
import com.amazonaws.auth.policy.Statement.Effect;
import com.amazonaws.auth.policy.actions.SQSActions;
import com.amazonaws.services.sns.AmazonSNS;
import com.amazonaws.services.sns.AmazonSNSClientBuilder;
import com.amazonaws.services.sns.model.CreateTopicRequest;
import com.amazonaws.services.sns.model.CreateTopicResult;
```

```
import com.amazonaws.services.sqs.AmazonSQS;
import com.amazonaws.services.sqs.AmazonSQSClientBuilder;
import com.amazonaws.services.sqs.model.CreateQueueRequest;
import com.amazonaws.services.sqs.model.Message;
import com.amazonaws.services.sqs.model.QueueAttributeName;
import com.amazonaws.services.sqs.model.SetQueueAttributesRequest;
import com.amazonaws.services.textract.AmazonTextract;
import com.amazonaws.services.textract.AmazonTextractClientBuilder;
import com.amazonaws.services.textract.model.Block;
import com.amazonaws.services.textract.model.DocumentLocation;
import com.amazonaws.services.textract.model.DocumentMetadata;
import com.amazonaws.services.textract.model.GetDocumentAnalysisRequest;
import com.amazonaws.services.textract.model.GetDocumentAnalysisResult;
import com.amazonaws.services.textract.model.GetDocumentTextDetectionRequest;
import com.amazonaws.services.textract.model.GetDocumentTextDetectionResult;
import com.amazonaws.services.textract.model.NotificationChannel;
import com.amazonaws.services.textract.model.Relationship;
import com.amazonaws.services.textract.model.S3Object;
import com.amazonaws.services.textract.model.StartDocumentAnalysisRequest;
import com.amazonaws.services.textract.model.StartDocumentAnalysisResult;
import com.amazonaws.services.textract.model.StartDocumentTextDetectionRequest;
import com.amazonaws.services.textract.model.StartDocumentTextDetectionResult;
import com.fasterxml.jackson.databind.JsonNode;
import com.fasterxml.jackson.databind.ObjectMapper;;
public class DocumentProcessor {

    private static String sqsQueueName=null;
    private static String snsTopicName=null;
    private static String snsTopicArn = null;
    private static String roleArn= null;
    private static String sqsQueueUrl = null;
    private static String sqsQueueArn = null;
    private static String startJobId = null;
    private static String bucket = null;
    private static String document = null;
    private static AmazonSQS sqs=null;
    private static AmazonSNS sns=null;
    private static AmazonTextract textract = null;

    public enum ProcessType {
        DETECTION,ANALYSIS
    }

    public static void main(String[] args) throws Exception {
```

```
        String document = "document";
        String bucket = "bucket";
        String roleArn="role";

        sns = AmazonSNSClientBuilder.defaultClient();
        sqs= AmazonSQSClientBuilder.defaultClient();
        textract=AmazonTextractClientBuilder.defaultClient();

        CreateTopicandQueue();
        ProcessDocument(bucket,document,roleArn,ProcessType.DETECTION);
        DeleteTopicandQueue();
        System.out.println("Done!");


    }
    // Creates an SNS topic and SQS queue. The queue is subscribed to the
topic.
    static void CreateTopicandQueue()
    {
        //create a new SNS topic
        snsTopicName="AmazonTextractTopic" +
Long.toString(System.currentTimeMillis());
        CreateTopicRequest createTopicRequest = new
CreateTopicRequest(snsTopicName);
        CreateTopicResult createTopicResult =
sns.createTopic(createTopicRequest);
        snsTopicArn=createTopicResult.getTopicArn();

        //Create a new SQS Queue
        sqsQueueName="AmazonTextractQueue" +
Long.toString(System.currentTimeMillis());
        final CreateQueueRequest createQueueRequest = new
CreateQueueRequest(sqsQueueName);
        sqsQueueUrl = sqs.createQueue(createQueueRequest).getQueueUrl();
        sqsQueueArn = sqs.getQueueAttributes(sqsQueueUrl,
Arrays.asList("QueueArn")).getAttributes().get("QueueArn");

        //Subscribe SQS queue to SNS topic
        String sqsSubscriptionArn = sns.subscribe(snsTopicArn, "sqs",
sqsQueueArn).getSubscriptionArn();

        // Authorize queue
          Policy policy = new Policy().withStatements(
```

```
                    new Statement(Effect.Allow)
                    .withPrincipals(Principal.AllUsers)
                    .withActions(SQSActions.SendMessage)
                    .withResources(new Resource(sqsQueueArn))
                    .withConditions(new
Condition().withType("ArnEquals").withConditionKey("aws:SourceArn").withValues(snsTopic
                    );


            Map queueAttributes = new HashMap();
            queueAttributes.put(QueueAttributeName.Policy.toString(),
policy.toJson());
            sqs.setQueueAttributes(new SetQueueAttributesRequest(sqsQueueUrl,
queueAttributes));


           System.out.println("Topic arn: " + snsTopicArn);
           System.out.println("Queue arn: " + sqsQueueArn);
           System.out.println("Queue url: " + sqsQueueUrl);
           System.out.println("Queue sub arn: " + sqsSubscriptionArn );
       }
   static void DeleteTopicandQueue()
   {
       if (sqs !=null) {
           sqs.deleteQueue(sqsQueueUrl);
           System.out.println("SQS queue deleted");
       }

       if (sns!=null) {
           sns.deleteTopic(snsTopicArn);
           System.out.println("SNS topic deleted");
       }
   }

   //Starts the processing of the input document.
   static void ProcessDocument(String inBucket, String inDocument, String
inRoleArn, ProcessType type) throws Exception
   {
       bucket=inBucket;
       document=inDocument;
       roleArn=inRoleArn;

       switch(type)
       {
```

```
            case DETECTION:
                StartDocumentTextDetection(bucket, document);
                System.out.println("Processing type: Detection");
                break;
            case ANALYSIS:
                StartDocumentAnalysis(bucket,document);
                System.out.println("Processing type: Analysis");
                break;
            default:
                System.out.println("Invalid processing type. Choose Detection or
Analysis");
                throw new Exception("Invalid processing type");

        }

        System.out.println("Waiting for job: " + startJobId);
        //Poll queue for messages
        List<Message> messages=null;
        int dotLine=0;
        boolean jobFound=false;

        //loop until the job status is published. Ignore other messages in
queue.
        do{
            messages = sqs.receiveMessage(sqsQueueUrl).getMessages();
            if (dotLine++<40){
                System.out.print(".");
            }else{
                System.out.println();
                dotLine=0;
            }

            if (!messages.isEmpty()) {
                //Loop through messages received.
                for (Message message: messages) {
                    String notification = message.getBody();

                    // Get status and job id from notification.
                    ObjectMapper mapper = new ObjectMapper();
                    JsonNode jsonMessageTree = mapper.readTree(notification);
                    JsonNode messageBodyText = jsonMessageTree.get("Message");
                    ObjectMapper operationResultMapper = new ObjectMapper();
                    JsonNode jsonResultTree =
operationResultMapper.readTree(messageBodyText.textValue());
```

```
                    JsonNode operationJobId = jsonResultTree.get("JobId");
                    JsonNode operationStatus = jsonResultTree.get("Status");
                    System.out.println("Job found was " + operationJobId);
                    // Found job. Get the results and display.
                    if(operationJobId.asText().equals(startJobId)){
                        jobFound=true;
                        System.out.println("Job id: " + operationJobId );
                        System.out.println("Status : " +
operationStatus.toString());
                        if (operationStatus.asText().equals("SUCCEEDED")){
                            switch(type)
                            {
                                case DETECTION:
                                    GetDocumentTextDetectionResults();
                                    break;
                                case ANALYSIS:
                                    GetDocumentAnalysisResults();
                                    break;
                                default:
                                    System.out.println("Invalid processing type.
Choose Detection or Analysis");
                                    throw new Exception("Invalid processing
type");

                            }
                        }
                        else{
                            System.out.println("Document analysis failed");
                        }


sqs.deleteMessage(sqsQueueUrl,message.getReceiptHandle());
                    }

                    else{
                        System.out.println("Job received was not job " +
startJobId);
                        //Delete unknown message. Consider moving message to
dead letter queue

sqs.deleteMessage(sqsQueueUrl,message.getReceiptHandle());
                    }
                }
            }
```

```
            else {
                Thread.sleep(5000);
            }
        } while (!jobFound);

        System.out.println("Finished processing document");
    }

    private static void StartDocumentTextDetection(String bucket, String
document) throws Exception{

        //Create notification channel
        NotificationChannel channel= new NotificationChannel()
                .withSNSTopicArn(snsTopicArn)
                .withRoleArn(roleArn);

        StartDocumentTextDetectionRequest req = new
StartDocumentTextDetectionRequest()
                .withDocumentLocation(new DocumentLocation()
                    .withS3Object(new S3Object()
                        .withBucket(bucket)
                        .withName(document)))
                .withJobTag("DetectingText")
                .withNotificationChannel(channel);

        StartDocumentTextDetectionResult startDocumentTextDetectionResult =
textract.startDocumentTextDetection(req);
        startJobId=startDocumentTextDetectionResult.getJobId();
    }

 //Gets the results of processing started by StartDocumentTextDetection
    private static void GetDocumentTextDetectionResults() throws Exception{
        int maxResults=1000;
        String paginationToken=null;
        GetDocumentTextDetectionResult response=null;
        Boolean finished=false;

        while (finished==false)
        {
            GetDocumentTextDetectionRequest documentTextDetectionRequest= new
GetDocumentTextDetectionRequest()
                    .withJobId(startJobId)
                    .withMaxResults(maxResults)
                    .withNextToken(paginationToken);
```

```
            response =
textract.getDocumentTextDetection(documentTextDetectionRequest);
            DocumentMetadata documentMetaData=response.getDocumentMetadata();

            System.out.println("Pages: " +
documentMetaData.getPages().toString());

            //Show blocks information
            List<Block> blocks= response.getBlocks();
            for (Block block : blocks) {
                DisplayBlockInfo(block);
            }
            paginationToken=response.getNextToken();
            if (paginationToken==null)
                finished=true;

        }

    }

    private static void StartDocumentAnalysis(String bucket, String document)
throws Exception{
        //Create notification channel
        NotificationChannel channel= new NotificationChannel()
                .withSNSTopicArn(snsTopicArn)
                .withRoleArn(roleArn);

        StartDocumentAnalysisRequest req = new StartDocumentAnalysisRequest()
                .withFeatureTypes("TABLES","FORMS")
                .withDocumentLocation(new DocumentLocation()
                    .withS3Object(new S3Object()
                        .withBucket(bucket)
                        .withName(document)))
                .withJobTag("AnalyzingText")
                .withNotificationChannel(channel);

        StartDocumentAnalysisResult startDocumentAnalysisResult =
textract.startDocumentAnalysis(req);
        startJobId=startDocumentAnalysisResult.getJobId();
    }
    //Gets the results of processing started by StartDocumentAnalysis
    private static void GetDocumentAnalysisResults() throws Exception{

        int maxResults=1000;
```

```
        String paginationToken=null;
        GetDocumentAnalysisResult response=null;
        Boolean finished=false;

        //loops until pagination token is null
        while (finished==false)
        {
            GetDocumentAnalysisRequest documentAnalysisRequest= new
GetDocumentAnalysisRequest()
                    .withJobId(startJobId)
                    .withMaxResults(maxResults)
                    .withNextToken(paginationToken);

            response = textract.getDocumentAnalysis(documentAnalysisRequest);

            DocumentMetadata documentMetaData=response.getDocumentMetadata();

            System.out.println("Pages: " +
documentMetaData.getPages().toString());

            //Show blocks, confidence and detection times
            List<Block> blocks= response.getBlocks();

            for (Block block : blocks) {
                DisplayBlockInfo(block);
            }
            paginationToken=response.getNextToken();
            if (paginationToken==null)
                finished=true;
        }

    }
    //Displays Block information for text detection and text analysis
    private static void DisplayBlockInfo(Block block) {
        System.out.println("Block Id : " + block.getId());
        if (block.getText()!=null)
            System.out.println("\tDetected text: " + block.getText());
        System.out.println("\tType: " + block.getBlockType());

        if (block.getBlockType().equals("PAGE") !=true) {
            System.out.println("\tConfidence: " +
block.getConfidence().toString());
        }
        if(block.getBlockType().equals("CELL"))
```

```
        {
            System.out.println("\tCell information:");
            System.out.println("\t\tColumn: " + block.getColumnIndex());
            System.out.println("\t\tRow: " + block.getRowIndex());
            System.out.println("\t\tColumn span: " + block.getColumnSpan());
            System.out.println("\t\tRow span: " + block.getRowSpan());

        }

        System.out.println("\tRelationships");
        List<Relationship> relationships=block.getRelationships();
        if(relationships!=null) {
            for (Relationship relationship : relationships) {
                System.out.println("\t\tType: " + relationship.getType());
                System.out.println("\t\tIDs: " +
relationship.getIds().toString());
            }
        } else {
            System.out.println("\t\tNo related Blocks");
        }

        System.out.println("\tGeometry");
        System.out.println("\t\tBounding Box: " +
block.getGeometry().getBoundingBox().toString());
        System.out.println("\t\tPolygon: " +
block.getGeometry().getPolygon().toString());

        List<String> entityTypes = block.getEntityTypes();

        System.out.println("\tEntity Types");
        if(entityTypes!=null) {
            for (String entityType : entityTypes) {
                System.out.println("\t\tEntity Type: " + entityType);
            }
        } else {
            System.out.println("\t\tNo entity type");
        }

        if(block.getBlockType().equals("SELECTION_ELEMENT")) {
            System.out.print("    Selection element detected: ");
            if (block.getSelectionStatus().equals("SELECTED")){
                System.out.println("Selected");
            }else {
                System.out.println(" Not selected");
```

```
            }
        }
        if(block.getPage()!=null)
            System.out.println("\tPage: " + block.getPage());
        System.out.println();
    }
}
```

AWS CLI

CetteAWS CLIlance la détection asynchrone de texte dans un document spécifié. Elle renvoie un objet .`job-id`qui peuvent être utilisés pour récupérer les résultats de la détection.

```
aws textract start-document-text-detection --document-location
"{\"S3Object\":{\"Bucket\":\"bucket-name\",\"Name\":\"file-name\"}}" --
region region-name
```

CetteAWS CLIrenvoie les résultats d'une opération asynchrone Amazon Textract lorsqu'elle est fournie avec un`job-id`.

```
aws textract get-document-text-detection --region region-name --job-id job-id-
number
```

Si vous accédez à l'interface de ligne de commande sur un appareil Windows, utilisez des guillemets doubles au lieu de guillemets simples et échappez aux guillemets doubles internes par une barre oblique inverse (c'est-à-dire \) pour résoudre les erreurs d'analyseur que vous pourriez rencontrer. Pour un exemple, consultez ci-dessous

```
aws textract start-document-text-detection --document-location "{\"S3Object\":
{\"Bucket\":\"bucket\",\"Name\":\"document\"}}" --region region-name
```

Python

```
import boto3
import json
import sys
import time


class ProcessType:
```

```
    DETECTION = 1
    ANALYSIS = 2


class DocumentProcessor:
    jobId = ''
    region_name = ''

    roleArn = ''
    bucket = ''
    document = ''

    sqsQueueUrl = ''
    snsTopicArn = ''
    processType = ''

    def __init__(self, role, bucket, document, region):
        self.roleArn = role
        self.bucket = bucket
        self.document = document
        self.region_name = region

        self.textract = boto3.client('textract', region_name=self.region_name)
        self.sqs = boto3.client('sqs')
        self.sns = boto3.client('sns')

    def ProcessDocument(self, type):
        jobFound = False

        self.processType = type
        validType = False

        # Determine which type of processing to perform
        if self.processType == ProcessType.DETECTION:
            response = self.textract.start_document_text_detection(
                DocumentLocation={'S3Object': {'Bucket': self.bucket, 'Name':
 self.document}},
                NotificationChannel={'RoleArn': self.roleArn, 'SNSTopicArn':
 self.snsTopicArn})
            print('Processing type: Detection')
            validType = True

        if self.processType == ProcessType.ANALYSIS:
            response = self.textract.start_document_analysis(
```

```
                DocumentLocation={'S3Object': {'Bucket': self.bucket, 'Name':
self.document}},
                FeatureTypes=["TABLES", "FORMS"],
                NotificationChannel={'RoleArn': self.roleArn, 'SNSTopicArn':
self.snsTopicArn})
            print('Processing type: Analysis')
            validType = True

        if validType == False:
            print("Invalid processing type. Choose Detection or Analysis.")
            return

        print('Start Job Id: ' + response['JobId'])
        dotLine = 0
        while jobFound == False:
            sqsResponse = self.sqs.receive_message(QueueUrl=self.sqsQueueUrl,
MessageAttributeNames=['ALL'],
                                                   MaxNumberOfMessages=10)

            if sqsResponse:

                if 'Messages' not in sqsResponse:
                    if dotLine < 40:
                        print('.', end='')
                        dotLine = dotLine + 1
                    else:
                        print()
                        dotLine = 0
                    sys.stdout.flush()
                    time.sleep(5)
                    continue

                for message in sqsResponse['Messages']:
                    notification = json.loads(message['Body'])
                    textMessage = json.loads(notification['Message'])
                    print(textMessage['JobId'])
                    print(textMessage['Status'])
                    if str(textMessage['JobId']) == response['JobId']:
                        print('Matching Job Found:' + textMessage['JobId'])
                        jobFound = True
                        self.GetResults(textMessage['JobId'])
                        self.sqs.delete_message(QueueUrl=self.sqsQueueUrl,

ReceiptHandle=message['ReceiptHandle'])
```

```
                    else:
                        print("Job didn't match:" +
                              str(textMessage['JobId']) + ' : ' +
 str(response['JobId']))
                    # Delete the unknown message. Consider sending to dead
 letter queue
                    self.sqs.delete_message(QueueUrl=self.sqsQueueUrl,

 ReceiptHandle=message['ReceiptHandle'])

        print('Done!')

    def CreateTopicandQueue(self):

        millis = str(int(round(time.time() * 1000)))

        # Create SNS topic
        snsTopicName = "AmazonTextractTopic" + millis

        topicResponse = self.sns.create_topic(Name=snsTopicName)
        self.snsTopicArn = topicResponse['TopicArn']

        # create SQS queue
        sqsQueueName = "AmazonTextractQueue" + millis
        self.sqs.create_queue(QueueName=sqsQueueName)
        self.sqsQueueUrl = self.sqs.get_queue_url(QueueName=sqsQueueName)
['QueueUrl']

        attribs = self.sqs.get_queue_attributes(QueueUrl=self.sqsQueueUrl,
                                                AttributeNames=['QueueArn'])
['Attributes']

        sqsQueueArn = attribs['QueueArn']

        # Subscribe SQS queue to SNS topic
        self.sns.subscribe(
            TopicArn=self.snsTopicArn,
            Protocol='sqs',
            Endpoint=sqsQueueArn)

        # Authorize SNS to write SQS queue
        policy = """{{
  "Version":"2012-10-17",
  "Statement":[
```

```
    {{
      "Sid":"MyPolicy",
      "Effect":"Allow",
      "Principal" : {{"AWS" : "*"}},
      "Action":"SQS:SendMessage",
      "Resource": "{}",
      "Condition":{{
        "ArnEquals":{{
          "aws:SourceArn": "{}"
        }}
      }}
    }}
  ]
}}""".format(sqsQueueArn, self.snsTopicArn)

        response = self.sqs.set_queue_attributes(
            QueueUrl=self.sqsQueueUrl,
            Attributes={
                'Policy': policy
            })

    def DeleteTopicandQueue(self):
        self.sqs.delete_queue(QueueUrl=self.sqsQueueUrl)
        self.sns.delete_topic(TopicArn=self.snsTopicArn)

    # Display information about a block
    def DisplayBlockInfo(self, block):

        print("Block Id: " + block['Id'])
        print("Type: " + block['BlockType'])
        if 'EntityTypes' in block:
            print('EntityTypes: {}'.format(block['EntityTypes']))

        if 'Text' in block:
            print("Text: " + block['Text'])

        if block['BlockType'] != 'PAGE':
            print("Confidence: " + "{:.2f}".format(block['Confidence']) + "%")

        print('Page: {}'.format(block['Page']))

        if block['BlockType'] == 'CELL':
            print('Cell Information')
            print('\tColumn: {} '.format(block['ColumnIndex']))
```

```
            print('\tRow: {}'.format(block['RowIndex']))
            print('\tColumn span: {} '.format(block['ColumnSpan']))
            print('\tRow span: {}'.format(block['RowSpan']))

            if 'Relationships' in block:
                print('\tRelationships: {}'.format(block['Relationships']))

        print('Geometry')
        print('\tBounding Box: {}'.format(block['Geometry']['BoundingBox']))
        print('\tPolygon: {}'.format(block['Geometry']['Polygon']))

        if block['BlockType'] == 'SELECTION_ELEMENT':
            print('    Selection element detected: ', end='')
            if block['SelectionStatus'] == 'SELECTED':
                print('Selected')
            else:
                print('Not selected')

    def GetResults(self, jobId):
        maxResults = 1000
        paginationToken = None
        finished = False

        while finished == False:

            response = None

            if self.processType == ProcessType.ANALYSIS:
                if paginationToken == None:
                    response = self.textract.get_document_analysis(JobId=jobId,

MaxResults=maxResults)
                else:
                    response = self.textract.get_document_analysis(JobId=jobId,

MaxResults=maxResults,

NextToken=paginationToken)

            if self.processType == ProcessType.DETECTION:
                if paginationToken == None:
                    response =
self.textract.get_document_text_detection(JobId=jobId,
```

```
MaxResults=maxResults)
                else:
                    response =
self.textract.get_document_text_detection(JobId=jobId,

MaxResults=maxResults,

NextToken=paginationToken)

            blocks = response['Blocks']
            print('Detected Document Text')
            print('Pages: {}'.format(response['DocumentMetadata']['Pages']))

            # Display block information
            for block in blocks:
                self.DisplayBlockInfo(block)
                print()
                print()

            if 'NextToken' in response:
                paginationToken = response['NextToken']
            else:
                finished = True

    def GetResultsDocumentAnalysis(self, jobId):
        maxResults = 1000
        paginationToken = None
        finished = False

        while finished == False:

            response = None
            if paginationToken == None:
                response = self.textract.get_document_analysis(JobId=jobId,

MaxResults=maxResults)
            else:
                response = self.textract.get_document_analysis(JobId=jobId,

MaxResults=maxResults,

NextToken=paginationToken)
```

```
                # Get the text blocks
            blocks = response['Blocks']
            print('Analyzed Document Text')
            print('Pages: {}'.format(response['DocumentMetadata']['Pages']))
            # Display block information
            for block in blocks:
                self.DisplayBlockInfo(block)
                print()
                print()

                if 'NextToken' in response:
                    paginationToken = response['NextToken']
                else:
                    finished = True


def main():
    roleArn = ''
    bucket = ''
    document = ''
    region_name = ''

    analyzer = DocumentProcessor(roleArn, bucket, document, region_name)
    analyzer.CreateTopicandQueue()
    analyzer.ProcessDocument(ProcessType.DETECTION)
    analyzer.DeleteTopicandQueue()


if __name__ == "__main__":
    main()
```

Node.JS

Dans cet exemple, remplacez la valeur de`roleArn`avec l'ARN du rôle IAM que vous avez enregistré dansDonner à Amazon Textract l'autorisation d'accès à votre rubrique Amazon SNS. Remplacez les valeurs de`bucket`et`document`Avec le nom du fichier de compartiment et le nom du document que vous avez spécifiés à l'étape 2 ci-dessus. Remplacez la valeur de`processType`avec le type de traitement que vous souhaitez utiliser sur le document d'entrée. Enfin, remplacez la valeur de`REGION`Avec la région dans laquelle votre client opère.

```
 // snippet-start:[sqs.JavaScript.queues.createQueueV3]
// Import required AWS SDK clients and commands for Node.js
```

```
import { CreateQueueCommand, GetQueueAttributesCommand, GetQueueUrlCommand,
    SetQueueAttributesCommand, DeleteQueueCommand, ReceiveMessageCommand,
 DeleteMessageCommand } from  "@aws-sdk/client-sqs";
  import {CreateTopicCommand, SubscribeCommand, DeleteTopicCommand } from "@aws-
sdk/client-sns";
  import  { SQSClient } from "@aws-sdk/client-sqs";
  import  { SNSClient } from "@aws-sdk/client-sns";
  import  { TextractClient, StartDocumentTextDetectionCommand,
 StartDocumentAnalysisCommand, GetDocumentAnalysisCommand,
 GetDocumentTextDetectionCommand, DocumentMetadata } from "@aws-sdk/client-
textract";
  import { stdout } from "process";

  // Set the AWS Region.
  const REGION = "us-east-1"; //e.g. "us-east-1"
  // Create SNS service object.
  const sqsClient = new SQSClient({ region: REGION });
  const snsClient = new SNSClient({ region: REGION });
  const textractClient = new TextractClient({ region: REGION });

  // Set bucket and video variables
  const bucket = "bucket-name";

  const documentName = "document-name";
  const roleArn = "role-arn"
  const processType = "DETECTION"
  var startJobId = ""

  var ts = Date.now();
  const snsTopicName = "AmazonTextractExample" + ts;
  const snsTopicParams = {Name: snsTopicName}
  const sqsQueueName = "AmazonTextractQueue-" + ts;

  // Set the parameters
  const sqsParams = {
    QueueName: sqsQueueName, //SQS_QUEUE_URL
    Attributes: {
      DelaySeconds: "60", // Number of seconds delay.
      MessageRetentionPeriod: "86400", // Number of seconds delay.
    },
  };

  // Process a document based on operation type
```

```
 const processDocumment = async (type, bucket, videoName, roleArn, sqsQueueUrl,
snsTopicArn) =>
   {
   try
   {
       // Set job found and success status to false initially
     var jobFound = false
     var succeeded = false
     var dotLine = 0
     var processType = type
     var validType = false

     if (processType == "DETECTION"){
       var response = await textractClient.send(new
StartDocumentTextDetectionCommand({DocumentLocation:{S3Object:{Bucket:bucket,
Name:videoName}},
         NotificationChannel:{RoleArn: roleArn, SNSTopicArn: snsTopicArn}}))
       console.log("Processing type: Detection")
       validType = true
     }

     if (processType == "ANALYSIS"){
       var response = await textractClient.send(new
StartDocumentAnalysisCommand({DocumentLocation:{S3Object:{Bucket:bucket,
Name:videoName}},
         NotificationChannel:{RoleArn: roleArn, SNSTopicArn: snsTopicArn}}))
       console.log("Processing type: Analysis")
       validType = true
     }

     if (validType == false){
         console.log("Invalid processing type. Choose Detection or Analysis.")
         return
     }
   // while not found, continue to poll for response
   console.log(`Start Job ID: ${response.JobId}`)
   while (jobFound == false){
     var sqsReceivedResponse = await sqsClient.send(new
ReceiveMessageCommand({QueueUrl:sqsQueueUrl,
       MaxNumberOfMessages:'ALL', MaxNumberOfMessages:10}));
     if (sqsReceivedResponse){
       var responseString = JSON.stringify(sqsReceivedResponse)
       if (!responseString.includes('Body')){
         if (dotLine < 40) {
```

```
            console.log('.')
            dotLine = dotLine + 1
          }else {
            console.log('')
            dotLine = 0
          };
          stdout.write('', () => {
            console.log('');
          });
          await new Promise(resolve => setTimeout(resolve, 5000));
          continue
        }
      }

        // Once job found, log Job ID and return true if status is succeeded
        for (var message of sqsReceivedResponse.Messages){
            console.log("Retrieved messages:")
            var notification = JSON.parse(message.Body)
            var rekMessage = JSON.parse(notification.Message)
            var messageJobId = rekMessage.JobId
            if (String(rekMessage.JobId).includes(String(startJobId))){
                console.log('Matching job found:')
                console.log(rekMessage.JobId)
                jobFound = true
                // GET RESUlTS FUNCTION HERE
                var operationResults = await GetResults(processType,
rekMessage.JobId)
                //GET RESULTS FUMCTION HERE
                console.log(rekMessage.Status)
            if (String(rekMessage.Status).includes(String("SUCCEEDED"))){
                succeeded = true
                console.log("Job processing succeeded.")
                var sqsDeleteMessage = await sqsClient.send(new
DeleteMessageCommand({QueueUrl:sqsQueueUrl,
ReceiptHandle:message.ReceiptHandle}));
            }
            }else{
            console.log("Provided Job ID did not match returned ID.")
            var sqsDeleteMessage = await sqsClient.send(new
DeleteMessageCommand({QueueUrl:sqsQueueUrl,
ReceiptHandle:message.ReceiptHandle}));
            }
        }
```

```
   console.log("Done!")
   }
   }catch (err) {
       console.log("Error", err);
     }
 }

 // Create the SNS topic and SQS Queue
 const createTopicandQueue = async () => {
   try {
     // Create SNS topic
     const topicResponse = await snsClient.send(new
CreateTopicCommand(snsTopicParams));
     const topicArn = topicResponse.TopicArn
     console.log("Success", topicResponse);
     // Create SQS Queue
     const sqsResponse = await sqsClient.send(new
CreateQueueCommand(sqsParams));
     console.log("Success", sqsResponse);
     const sqsQueueCommand = await sqsClient.send(new
GetQueueUrlCommand({QueueName: sqsQueueName}))
     const sqsQueueUrl = sqsQueueCommand.QueueUrl
     const attribsResponse = await sqsClient.send(new
GetQueueAttributesCommand({QueueUrl: sqsQueueUrl, AttributeNames:
['QueueArn']}))
     const attribs = attribsResponse.Attributes
     console.log(attribs)
     const queueArn = attribs.QueueArn
     // subscribe SQS queue to SNS topic
     const subscribed = await snsClient.send(new SubscribeCommand({TopicArn:
topicArn, Protocol:'sqs', Endpoint: queueArn}))
     const policy = {
       Version: "2012-10-17",
       Statement: [
         {
           Sid: "MyPolicy",
           Effect: "Allow",
           Principal: {AWS: "*"},
           Action: "SQS:SendMessage",
           Resource: queueArn,
           Condition: {
             ArnEquals: {
               'aws:SourceArn': topicArn
             }
```

```
            }
          }
        ]
      };

      const response = sqsClient.send(new SetQueueAttributesCommand({QueueUrl:
sqsQueueUrl, Attributes: {Policy: JSON.stringify(policy)}}))
      console.log(response)
      console.log(sqsQueueUrl, topicArn)
      return [sqsQueueUrl, topicArn]

    } catch (err) {
      console.log("Error", err);

    }
 }

 const deleteTopicAndQueue = async (sqsQueueUrlArg, snsTopicArnArg) => {
    const deleteQueue = await sqsClient.send(new DeleteQueueCommand({QueueUrl:
sqsQueueUrlArg}));
    const deleteTopic = await snsClient.send(new DeleteTopicCommand({TopicArn:
snsTopicArnArg}));
    console.log("Successfully deleted.")
 }

 const displayBlockInfo = async (block) => {
    console.log(`Block ID: ${block.Id}`)
    console.log(`Block Type: ${block.BlockType}`)
    if (String(block).includes(String("EntityTypes"))){
        console.log(`EntityTypes: ${block.EntityTypes}`)
    }
    if (String(block).includes(String("Text"))){
        console.log(`EntityTypes: ${block.Text}`)
    }
    if (!String(block.BlockType).includes('PAGE')){
        console.log(`Confidence: ${block.Confidence}`)
    }
    console.log(`Page: ${block.Page}`)
    if (String(block.BlockType).includes("CELL")){
        console.log("Cell Information")
        console.log(`Column: ${block.ColumnIndex}`)
        console.log(`Row: ${block.RowIndex}`)
        console.log(`Column Span: ${block.ColumnSpan}`)
        console.log(`Row Span: ${block.RowSpan}`)
```

```
        if (String(block).includes("Relationships")){
            console.log(`Relationships: ${block.Relationships}`)
        }
    }

    console.log("Geometry")
    console.log(`Bounding Box: ${JSON.stringify(block.Geometry.BoundingBox)}`)
    console.log(`Polygon: ${JSON.stringify(block.Geometry.Polygon)}`)

    if (String(block.BlockType).includes('SELECTION_ELEMENT')){
      console.log('Selection Element detected:')
      if (String(block.SelectionStatus).includes('SELECTED')){
        console.log('Selected')
      } else {
        console.log('Not Selected')
      }

    }
 }

 const GetResults = async (processType, JobID) => {

    var maxResults = 1000
    var paginationToken = null
    var finished = false

    while (finished == false){
      var response = null
      if (processType == 'ANALYSIS'){
        if (paginationToken == null){
          response = textractClient.send(new
GetDocumentAnalysisCommand({JobId:JobID, MaxResults:maxResults}))

        }else{
          response = textractClient.send(new
GetDocumentAnalysisCommand({JobId:JobID, MaxResults:maxResults,
NextToken:paginationToken}))
        }
      }

      if(processType == 'DETECTION'){
        if (paginationToken == null){
          response = textractClient.send(new
GetDocumentTextDetectionCommand({JobId:JobID, MaxResults:maxResults}))
```

```
      }else{
        response = textractClient.send(new
GetDocumentTextDetectionCommand({JobId:JobID, MaxResults:maxResults,
NextToken:paginationToken}))
      }
    }

    await new Promise(resolve => setTimeout(resolve, 5000));
    console.log("Detected Documented Text")
    console.log(response)
    //console.log(Object.keys(response))
    console.log(typeof(response))
    var blocks = (await response).Blocks
    console.log(blocks)
    console.log(typeof(blocks))
    var docMetadata = (await response).DocumentMetadata
    var blockString = JSON.stringify(blocks)
    var parsed = JSON.parse(JSON.stringify(blocks))
    console.log(Object.keys(blocks))
    console.log(`Pages: ${docMetadata.Pages}`)
    blocks.forEach((block)=> {
      displayBlockInfo(block)
      console.log()
      console.log()
    })

    //console.log(blocks[0].BlockType)
    //console.log(blocks[1].BlockType)


    if(String(response).includes("NextToken")){
      paginationToken = response.NextToken
    }else{
      finished = true
    }
  }

}


// DELETE TOPIC AND QUEUE
const main = async () => {
  var sqsAndTopic = await createTopicandQueue();
```

```
    var process = await processDocumment(processType, bucket, documentName,
 roleArn, sqsAndTopic[0], sqsAndTopic[1])
    var deleteResults = await deleteTopicAndQueue(sqsAndTopic[0],
 sqsAndTopic[1])
  }

main()
```

4. Exécutez le code. L'opération peut prendre un certain temps pour s'exécuter. Lorsqu'elle est terminée, une liste de blocs pour le texte détecté ou analysé s'affiche.

Notification des résultats Amazon Textract

Amazon Textract publie les résultats d'une demande d'analyse Amazon Textract, y compris le statut d'achèvement, dans une rubrique Amazon Simple Notification Service (Amazon SNS). Pour obtenir la notification à partir d'une rubrique Amazon SNS, utilisez une file d'attente Amazon SQS ou une rubriqueAWS Lambda. Pour plus d'informations, consultez Appel d'opérations asynchrones Amazon Textract. Pour voir un exemple, consultez Détection ou analyse de texte dans un document multipage.

Les résultats ont le format JSON suivant :

```
{
  "JobId": "String",
  "Status": "String",
  "API": "String",
  "JobTag": "String",
  "Timestamp": Number,
  "DocumentLocation": {
    "S3ObjectName": "String",
    "S3Bucket": "String"
  }
}
```

Ce tableau décrit les différents paramètres dans une réponse Amazon Textract.

Paramètre	Description
JobId	Identificateur unique qu'Amazon Textract attribue à la tâche. Il correspond à l'identifiant

Paramètre	Description
	de tâche renvoyé par un fichier`Start`opération, telle queStartDocumentTextDetection.
État	Statut de la tâche. Les valeurs valides sont Succès, Echec ou Erreur.
API	Opération Amazon Textract utilisée pour analyser le document d'entrée, par exempleStartDocumentTextDetectionouStartDocumentAnalysis.
JobTag	L'identifiant spécifié par l'utilisateur pour la tâche. Vous spécifiez`JobTag`lors d'un appel au`Start`opération, telle queStartDocumentTextDetection.
Horodatage	Horodatage Unix qui indique quand la tâche s'est terminée, renvoyée en millisecondes.
Emplacement du document	Détails sur le document qui a été traité. Comprend le nom du fichier et le compartiment Amazon S3 dans lequel le fichier est stocké.

Gestion des appels restreintés et des connexions abandonnées

Une opération Amazon Textract peut échouer si vous dépassez le nombre maximal de transactions par seconde (TPS), ce qui entraîne une limitation du service pour votre application ou lorsque votre connexion est interrompue. Par exemple, si vous passez trop d'appels à des opérations Amazon Textract en peu de temps, il limite vos appels et envoie un`ProvisionedThroughputExceededException`erreur dans la réponse de l'opération. Pour obtenir des informations sur les quotas TPS Amazon Textract, veuillez consulterQuotas Amazon Textract.

Vous pouvez gérer la limitation et les connexions abandonnées en réessayant automatiquement l'opération. Vous pouvez spécifier le nombre de nouvelles tentatives en incluant le`Config`paramètre lorsque vous créez le client Amazon Textract. Nous recommandons un nombre de nouvelles tentatives de 5. LeAWSLe kit SDK tente à nouveau une opération autant de fois qu'indiqué avant d'échouer et de lever une exception. Pour plus d'informations, consultez la page Nouvelles tentatives après erreur et interruptions exponentielles dans AWS.

> **Note**
>
> Les nouvelles tentatives automatiques fonctionnent à la fois pour les opérations synchrone et asynchrone. Avant de spécifier des tentatives automatiques, assurez-vous que vous disposez de la version la plus récente du kit AWS SDK. Pour plus d'informations, consultez Étape 2 : Configuration de l'AWS CLIetAWSKits SDK.

L'exemple suivant montre comment réessayer automatiquement les opérations Amazon Textract lorsque vous traitez plusieurs documents.

Prérequis

- Si vous ne l'avez pas déjà fait :

  a. Créer ou mettre à jour un utilisateur IAM avec`AmazonTextractFullAccess`et`AmazonS3ReadOnlyAccess`Autorisations. Pour plus d'informations, consultez Étape 1 : Configuration d'un compte AWS et création d'un utilisateur IAM.

b. Installez et configurez l'AWS CLI et les kits SDK AWS. Pour plus d'informations, consultez Étape 2 : Configuration de l'AWS CLIetAWSKits SDK.

Pour relancer automatiquement les opérations

1. Chargez plusieurs images de documents dans votre compartiment S3 pour exécuter l'exemple Synchrone. Chargez un document de plusieurs pages dans votre compartiment S3 et exécutez`StartDocumentTextDetection`dessus pour exécuter l'exemple asynchrone.

   Pour obtenir des instructions, consultezChargement d'objets dans Amazon S3dans leGuide de l'utilisateur Amazon Simple Storage Service.

2. Les exemples suivants montrent comment utiliser le`Config`pour relancer automatiquement une opération. L'exemple synchrone appelle le`DetectDocumentText`, tandis que l'exemple asynchrone appelle l'option`GetDocumentTextDetection`.

   Sync Example

   Utilisez les exemples suivants pour appeler le`DetectDocumentText`opération sur les documents de votre compartiment Amazon S3. Dans`main`, modifiez la valeur de`bucket`dans votre compartiment S3. Modifiez la valeur de`documents`aux noms des images de document que vous avez téléchargées à l'étape 2.

```
import boto3
from botocore.client import Config
# Documents

def process_multiple_documents(bucket, documents):

    config = Config(retries = dict(max_attempts = 5))

    # Amazon Textract client
    textract = boto3.client('textract', config=config)

    for documentName in documents:

        print("\nProcessing:
 {}\n=========================================".format(documentName))

        # Call Amazon Textract
        response = textract.detect_document_text(
```

```
            Document={
                'S3Object': {
                    'Bucket': bucket,
                    'Name': documentName
                }
            })

        # Print detected text
        for item in response["Blocks"]:
            if item["BlockType"] == "LINE":
                print ('\033[94m' +  item["Text"] + '\033[0m')


def main():
    bucket = ""
    documents = ["document-image-1.png",
    "document-image-2.png", "document-image-3.png",
    "document-image-4.png", "document-image-5.png" ]
    process_multiple_documents(bucket, documents)



if __name__ == "__main__":
    main()
```

Async Example

Utilisez les exemples suivants pour appeler l'opération `GetDocumentTextDetection`. Il suppose que vous avez déjà appelé`StartDocumentTextDetection`sur les documents de votre compartiment Amazon S3 et obtenu un`JobId`. Dans`main`, modifiez la valeur de`bucket`à votre compartiment S3 et à la valeur de`roleArn`à l'arn affecté à votre rôle Textract. Vous devez également modifier la valeur de`document`au nom de votre document multi-pages dans votre compartiment Amazon S3. Enfin, remplacez la valeur de`region_name`avec le nom de votre région et fournissez le`GetResults`avec le nom de votre`jobId`.

```
import boto3
from botocore.client import Config

class DocumentProcessor:
    jobId = ''
    region_name = ''
```

```
    roleArn = ''
    bucket = ''
    document = ''

    sqsQueueUrl = ''
    snsTopicArn = ''
    processType = ''

    def __init__(self, role, bucket, document, region):
        self.roleArn = role
        self.bucket = bucket
        self.document = document
        self.region_name = region
        self.config = Config(retries = dict(max_attempts = 5))

        self.textract = boto3.client('textract', region_name=self.region_name,
 config=self.config)
        self.sqs = boto3.client('sqs')
        self.sns = boto3.client('sns')

# Display information about a block
    def DisplayBlockInfo(self, block):

        print("Block Id: " + block['Id'])
        print("Type: " + block['BlockType'])
        if 'EntityTypes' in block:
            print('EntityTypes: {}'.format(block['EntityTypes']))

        if 'Text' in block:
            print("Text: " + block['Text'])

        if block['BlockType'] != 'PAGE':
            print("Confidence: " + "{:.2f}".format(block['Confidence']) + "%")

        print('Page: {}'.format(block['Page']))

        if block['BlockType'] == 'CELL':
            print('Cell Information')
            print('\tColumn: {} '.format(block['ColumnIndex']))
            print('\tRow: {}'.format(block['RowIndex']))
            print('\tColumn span: {} '.format(block['ColumnSpan']))
            print('\tRow span: {}'.format(block['RowSpan']))
```

```
            if 'Relationships' in block:
                print('\tRelationships: {}'.format(block['Relationships']))

        print('Geometry')
        print('\tBounding Box: {}'.format(block['Geometry']['BoundingBox']))
        print('\tPolygon: {}'.format(block['Geometry']['Polygon']))

        if block['BlockType'] == 'SELECTION_ELEMENT':
            print('    Selection element detected: ', end='')
            if block['SelectionStatus'] == 'SELECTED':
                print('Selected')
            else:
                print('Not selected')

    def GetResults(self, jobId):
        maxResults = 1000
        paginationToken = None
        finished = False

        while finished == False:

            response = None

            if paginationToken == None:
                response =
self.textract.get_document_text_detection(JobId=jobId,

MaxResults=maxResults)
            else:
                response =
self.textract.get_document_text_detection(JobId=jobId,

MaxResults=maxResults,

NextToken=paginationToken)

            blocks = response['Blocks']
            print('Detected Document Text')
            print('Pages: {}'.format(response['DocumentMetadata']['Pages']))

            # Display block information
            for block in blocks:
                self.DisplayBlockInfo(block)
                print()
```

```
                print()

            if 'NextToken' in response:
                paginationToken = response['NextToken']
            else:
                finished = True

def main():
    roleArn = 'role-arn'
    bucket = 'bucket-name'
    document = 'document-name'
    region_name = 'region-name'
    analyzer = DocumentProcessor(roleArn, bucket, document, region_name)
    analyzer.GetResults("job-id")

if __name__ == "__main__":
    main()
```

Bonnes pratiques pour Amazon Textract

Amazon Textract utilise l'apprentissage automatique pour lire des documents comme le ferait une personne. Il extrait du texte, des tableaux et des formulaires de documents. Utilisez les bonnes pratiques suivantes pour obtenir les meilleurs résultats à partir de vos documents.

Fournir un document de saisie optimal

Voici une liste de plusieurs façons d'optimiser vos documents d'entrée pour obtenir de meilleurs résultats.

- Assurez-vous que le texte de votre document est dans une langue prise en charge par Amazon Textract. Actuellement, Amazon Textract prend en charge l'anglais, l'espagnol, l'allemand, l'italien, le français et le portugais.
- Fournissez une image de haute qualité, idéalement au moins 150 ppp.
- Si votre document est déjà dans l'un des formats de fichier pris en charge par Amazon Textract (PDF, TIFF, JPEG et PNG), ne convertissez pas ou ne sous-échantillonnez pas le document avant de le télécharger Amazon Textract.

Pour obtenir les meilleurs résultats lors de l'extraction de texte à partir de tableaux dans des documents, veillez à ce que :

- Les tableaux de votre document sont visuellement séparés des éléments environnants de la page. Par exemple, le tableau n'est pas superposé sur une image ou un motif complexe.
- Le texte contenu dans le tableau est droit. Par exemple, le texte n'est pas pivoté par rapport à d'autres textes de la page.

Lorsque vous extrayez du texte à partir de tableaux, des résultats peuvent être incohérents lorsque :

- Cellules de tableau fusionnées couvrant plusieurs colonnes.
- Tableaux avec des cellules, des lignes ou des colonnes différentes des autres parties d'une même table.

Nous vous recommandons d'utiliserdétection de textecomme solution de contournement.

Utiliser les scores de fiabilité

Vous devez prendre en compte les scores de confiance renvoyés par les opérations de l'API Amazon Textract et la sensibilité de leur cas d'utilisation. Un score de fiabilité est un nombre compris entre 0 et 100 qui indique la probabilité qu'une prédiction donnée soit correcte. Il vous aide à prendre des décisions éclairées sur la façon dont vous utilisez les résultats.

Dans les applications sensibles aux erreurs de détection (faux positifs), appliquez un seuil de confiance minimal. L'application doit rejeter les résultats inférieurs à ce seuil ou signaler que les situations nécessitent un niveau plus élevé d'examen humain.

Le seuil optimal dépend de l'application. À des fins d'archivage, telles que la documentation de notes manuscrites, elle peut atteindre 50 %. Les processus métier impliquant des décisions financières peuvent nécessiter des seuils de 90 % ou plus.

Envisager d'utiliser la vérification humaine

Envisagez également d'intégrer la revue humaine dans vos flux de travail. Cela est particulièrement important pour les applications sensibles, telles que les processus métier impliquant des décisions financières.

Tutoriels

the section called “Block”les objets renvoyés par les opérations Amazon Textract contiennent les résultats des opérations de détection de texte et d'analyse de texte, telles quethe section called “AnalyzeDocument”. Les didacticiels Python suivants montrent les différentes façons dont vous pouvez utiliser les objets Block. Par exemple, vous pouvez exporter les informations de table dans un fichier CSV (valeurs séparées par une virgule).

Les didacticiels utilisent des opérations synchrone Amazon Textract qui renvoient tous les résultats. Si vous souhaitez utiliser des opérations asynchrones telles quethe section called “StartDocumentAnalysis”, vous devez modifier l'exemple de code pour prendre en charge plusieurs lots de données renvoyées`Block`objets. Pour utiliser l'exemple des opérations asynchrones, assurez-vous d'avoir suivi les instructions données à l'adresseConfiguration d'Amazon Textract pour les opérations asynchrones.

Pour obtenir des exemples qui vous montrent d'autres façons d'utiliser Amazon Textract, consultezExemples de code supplémentaires.

Rubriques

- Prérequis
- Extraction de paires clé-valeur à partir d'un document de formulaire
- Exportation de tables dans un fichier CSV
- Création d'unAWS LambdaFonction
- Exemples de code supplémentaires

Prérequis

Avant de pouvoir exécuter les exemples désignés dans cette section, vous devez configurer votre environnement.

Pour configurer votre environnement

1. Créer ou mettre à jour un utilisateur IAM avec`AmazonTextractFullAccess`autorisations. Pour plus d'informations, consultez Étape 1 : Configuration d'un compte AWS et création d'un utilisateur IAM.

2. Installez et configurez l'AWS CLI et les kits SDK AWS. Pour plus d'informations, consultez Étape 2 : Configuration de l'AWS CLIetAWSKits SDK.

Extraction de paires clé-valeur à partir d'un document de formulaire

L'exemple Python suivant montre comment extraire des paires clé-valeur dans des documents de formulaire à partir dethe section called “Block”objets stockés dans une carte. Les objets Block sont renvoyés à partir d'un appel àthe section called “AnalyzeDocument”. Pour plus d'informations, consultez Données de formulaire (paires clé-valeur).

Vous utilisez les fonctions suivantes :

- `get_kv_map`— AppelsAnalyzeDocument, et stocke les objets KEY et VALUE BLOCK dans une carte.
- `get_kv_relationshipetfind_value_block`: construit les relations clé-valeur à partir de la carte.

Pour extraire les paires clé-valeur d'un document de formulaire

1. Configurez votre environnement. Pour plus d'informations, consultez Prérequis.
2. Enregistrez l'exemple de code suivant dans un fichier nommétextract_python_kv_parser.py.

```
import boto3
import sys
import re
import json


def get_kv_map(file_name):

    with open(file_name, 'rb') as file:
        img_test = file.read()
        bytes_test = bytearray(img_test)
        print('Image loaded', file_name)

    # process using image bytes
    client = boto3.client('textract')
    response = client.analyze_document(Document={'Bytes': bytes_test},
 FeatureTypes=['FORMS'])
```

```
    # Get the text blocks
    blocks=response['Blocks']


    # get key and value maps
    key_map = {}
    value_map = {}
    block_map = {}
    for block in blocks:
        block_id = block['Id']
        block_map[block_id] = block
        if block['BlockType'] == "KEY_VALUE_SET":
            if 'KEY' in block['EntityTypes']:
                key_map[block_id] = block
            else:
                value_map[block_id] = block

    return key_map, value_map, block_map


def get_kv_relationship(key_map, value_map, block_map):
    kvs = {}
    for block_id, key_block in key_map.items():
        value_block = find_value_block(key_block, value_map)
        key = get_text(key_block, block_map)
        val = get_text(value_block, block_map)
        kvs[key] = val
    return kvs


def find_value_block(key_block, value_map):
    for relationship in key_block['Relationships']:
        if relationship['Type'] == 'VALUE':
            for value_id in relationship['Ids']:
                value_block = value_map[value_id]
    return value_block


def get_text(result, blocks_map):
    text = ''
    if 'Relationships' in result:
        for relationship in result['Relationships']:
            if relationship['Type'] == 'CHILD':
```

```
                for child_id in relationship['Ids']:
                    word = blocks_map[child_id]
                    if word['BlockType'] == 'WORD':
                        text += word['Text'] + ' '
                    if word['BlockType'] == 'SELECTION_ELEMENT':
                        if word['SelectionStatus'] == 'SELECTED':
                            text += 'X '


    return text


def print_kvs(kvs):
    for key, value in kvs.items():
        print(key, ":", value)


def search_value(kvs, search_key):
    for key, value in kvs.items():
        if re.search(search_key, key, re.IGNORECASE):
            return value

def main(file_name):

    key_map, value_map, block_map = get_kv_map(file_name)

    # Get Key Value relationship
    kvs = get_kv_relationship(key_map, value_map, block_map)
    print("\n\n== FOUND KEY : VALUE pairs ===\n")
    print_kvs(kvs)

    # Start searching a key value
    while input('\n Do you want to search a value for a key? (enter "n" for exit)
 ') != 'n':
        search_key = input('\n Enter a search key:')
        print('The value is:', search_value(kvs, search_key))

if __name__ == "__main__":
    file_name = sys.argv[1]
    main(file_name)
```

3. A partir d'une invite de commande, entrez la commande suivante. Remplacez`file`avec le fichier image du document que vous souhaitez analyser.

```
textract_python_kv_parser.py file
```

4. Lorsque vous y êtes invité, entrez une clé qui se trouve dans le document d'entrée. Si le code détecte la clé, il affiche la valeur de la clé.

Exportation de tables dans un fichier CSV

Ces exemples Python montrent comment exporter des tables à partir d'une image d'un document dans un fichier CSV (valeurs séparées par une virgule).

L'exemple d'analyse synchrone de documents recueille des informations de table à partir d'un appel àthe section called “AnalyzeDocument”. L'exemple d'analyse de documents asynchrone fait un appel àthe section called “StartDocumentAnalysis”puis récupère les résultats dethe section called “GetDocumentAnalysis”comme`Block`objets.

Les informations de table sont renvoyées sous formethe section called “Block”objets d'un appel àthe section called “AnalyzeDocument”. Pour plus d'informations, consultez Tables. Le`Block`sont stockés dans une structure cartographique utilisée pour exporter les données de la table dans un fichier CSV.

Synchronous

Dans cet exemple, vous allez utiliser les fonctions suivantes :

- `get_table_csv_results`— AppelsAnalyzeDocument, et crée une carte des tables détectées dans le document. Crée une représentation CSV de toutes les tables détectées.
- `generate_table_csv`: génère le fichier CSV pour une table individuelle.
- `get_rows_columns_map`: récupère les lignes et les colonnes de la carte.
- `get_text`: récupère le texte d'une cellule.

Pour exporter des tables dans un fichier CSV

1. Configurez votre environnement. Pour plus d'informations, consultez Prérequis.
2. Enregistrez l'exemple de code suivant dans un fichier nommétextract_python_table_parser.py.

```
import webbrowser, os
import json
```

```
import boto3
import io
from io import BytesIO
import sys
from pprint import pprint


def get_rows_columns_map(table_result, blocks_map):
    rows = {}
    for relationship in table_result['Relationships']:
        if relationship['Type'] == 'CHILD':
            for child_id in relationship['Ids']:
                cell = blocks_map[child_id]
                if cell['BlockType'] == 'CELL':
                    row_index = cell['RowIndex']
                    col_index = cell['ColumnIndex']
                    if row_index not in rows:
                        # create new row
                        rows[row_index] = {}

                    # get the text value
                    rows[row_index][col_index] = get_text(cell, blocks_map)
    return rows


def get_text(result, blocks_map):
    text = ''
    if 'Relationships' in result:
        for relationship in result['Relationships']:
            if relationship['Type'] == 'CHILD':
                for child_id in relationship['Ids']:
                    word = blocks_map[child_id]
                    if word['BlockType'] == 'WORD':
                        text += word['Text'] + ' '
                    if word['BlockType'] == 'SELECTION_ELEMENT':
                        if word['SelectionStatus'] =='SELECTED':
                            text +=  'X '
    return text


def get_table_csv_results(file_name):

    with open(file_name, 'rb') as file:
        img_test = file.read()
```

```
        bytes_test = bytearray(img_test)
        print('Image loaded', file_name)

    # process using image bytes
    # get the results
    client = boto3.client('textract')

    response = client.analyze_document(Document={'Bytes': bytes_test},
 FeatureTypes=['TABLES'])

    # Get the text blocks
    blocks=response['Blocks']
    pprint(blocks)

    blocks_map = {}
    table_blocks = []
    for block in blocks:
        blocks_map[block['Id']] = block
        if block['BlockType'] == "TABLE":
            table_blocks.append(block)

    if len(table_blocks) <= 0:
        return "<b> NO Table FOUND </b>"

    csv = ''
    for index, table in enumerate(table_blocks):
        csv += generate_table_csv(table, blocks_map, index +1)
        csv += '\n\n'

    return csv

def generate_table_csv(table_result, blocks_map, table_index):
    rows = get_rows_columns_map(table_result, blocks_map)

    table_id = 'Table_' + str(table_index)

    # get cells.
    csv = 'Table: {0}\n\n'.format(table_id)

    for row_index, cols in rows.items():

        for col_index, text in cols.items():
            csv += '{}'.format(text) + ","
        csv += '\n'
```

```
    csv += '\n\n\n'
    return csv

def main(file_name):
    table_csv = get_table_csv_results(file_name)

    output_file = 'output.csv'

    # replace content
    with open(output_file, "wt") as fout:
        fout.write(table_csv)

    # show the results
    print('CSV OUTPUT FILE: ', output_file)


if __name__ == "__main__":
    file_name = sys.argv[1]
    main(file_name)
```

3. A partir d'une invite de commande, entrez la commande suivante. Remplacez`file`avec le nom du fichier image que vous souhaitez analyser.

```
python textract_python_table_parser.py file
```

Lorsque vous exécutez cet exemple, la sortie CSV est enregistrée dans un fichier nommé`output.csv`.

Asynchronous

Dans cet exemple, vous allez utiliser deux scripts différents. Le premier script démarre le processus d'analyse asynchrone des documents avec`StartDocumentAnalysis`et obtient le`Block`informations renvoyées par`GetDocumentAnalysis`. Le second script prend le texte renvoyé`Block`informations pour chaque page, formate les données sous forme de tableau et enregistre les tables dans un fichier CSV.

Pour exporter des tables dans un fichier CSV

1. Configurez votre environnement. Pour plus d'informations, consultez Prérequis.

2. Assurez-vous d'avoir suivi les instructions données à l'adresseConfiguration d'Amazon Textract pour les opérations asynchrones. Le processus documenté sur cette page vous permet d'envoyer et de recevoir des messages sur l'état d'achèvement des tâches asynchrones.

3. Dans l'exemple de code suivant, remplacez la valeur de`roleArn`avec l'Arn attribué au rôle que vous avez créé à l'étape 2. Remplacez la valeur de`bucket`avec le nom du compartiment S3 contenant votre document. Remplacez la valeur de`document`par le nom du document dans votre compartiment S3. Remplacez la valeur de`region_name`par le nom de la région de votre compartiment.

   Enregistrez l'exemple de code suivant dans un fichier nomméstart_doc_analysis_for_table_extraction.py..

```
import boto3
import time

class DocumentProcessor:

    jobId = ''
    region_name = ''

    roleArn = ''
    bucket = ''
    document = ''

    sqsQueueUrl = ''
    snsTopicArn = ''
    processType = ''

    def __init__(self, role, bucket, document, region):
        self.roleArn = role
        self.bucket = bucket
        self.document = document
        self.region_name = region

        self.textract = boto3.client('textract', region_name=self.region_name)
        self.sqs = boto3.client('sqs')
        self.sns = boto3.client('sns')

    def ProcessDocument(self):
```

```
        jobFound = False

        response =
 self.textract.start_document_analysis(DocumentLocation={'S3Object': {'Bucket':
 self.bucket, 'Name': self.document}},
                FeatureTypes=["TABLES", "FORMS"],
 NotificationChannel={'RoleArn': self.roleArn, 'SNSTopicArn':
 self.snsTopicArn})
        print('Processing type: Analysis')

        print('Start Job Id: ' + response['JobId'])

        print('Done!')

    def CreateTopicandQueue(self):

        millis = str(int(round(time.time() * 1000)))

        # Create SNS topic
        snsTopicName = "AmazonTextractTopic" + millis

        topicResponse = self.sns.create_topic(Name=snsTopicName)
        self.snsTopicArn = topicResponse['TopicArn']

        # create SQS queue
        sqsQueueName = "AmazonTextractQueue" + millis
        self.sqs.create_queue(QueueName=sqsQueueName)
        self.sqsQueueUrl = self.sqs.get_queue_url(QueueName=sqsQueueName)
['QueueUrl']

        attribs = self.sqs.get_queue_attributes(QueueUrl=self.sqsQueueUrl,
                                                AttributeNames=['QueueArn'])
['Attributes']

        sqsQueueArn = attribs['QueueArn']

        # Subscribe SQS queue to SNS topic
        self.sns.subscribe(TopicArn=self.snsTopicArn, Protocol='sqs',
 Endpoint=sqsQueueArn)

        # Authorize SNS to write SQS queue
        policy = """{{
      "Version":"2012-10-17",
      "Statement":[
```

```
        {{
          "Sid":"MyPolicy",
          "Effect":"Allow",
          "Principal" : {{"AWS" : "*"}},
          "Action":"SQS:SendMessage",
          "Resource": "{}",
          "Condition":{{
            "ArnEquals":{{
              "aws:SourceArn": "{}"
            }}
          }}
        }}
      ]
    }}""".format(sqsQueueArn, self.snsTopicArn)

        response = self.sqs.set_queue_attributes(
            QueueUrl=self.sqsQueueUrl,
            Attributes={
                'Policy': policy
            })

def main():
    roleArn = 'role-arn'
    bucket = 'bucket-name'
    document = 'document-name'
    region_name = 'region-name'

    analyzer = DocumentProcessor(roleArn, bucket, document, region_name)
    analyzer.CreateTopicandQueue()
    analyzer.ProcessDocument()

if __name__ == "__main__":
    main()
```

4. Exécutez le code. Le code va imprimer un JobId. Copiez ce JobId vers le bas.
5. Attendez la fin du traitement de votre tâche et, une fois qu'il est terminé, copiez le code suivant dans un fichier nomméget_doc_analysis_for_table_extraction.py. Remplacez la valeur de`jobId`avec l'ID de Job que vous avez copié précédemment. Remplacez la valeur de`region_name`avec le nom de la région associée à votre rôle Textract. Remplacez la valeur de`file_name`par le nom que vous souhaitez donner au CSV de sortie.

```
import boto3
```

```
from pprint import pprint

jobId = 'job-id'
region_name = 'region-name'
file_name = "output-file-name.csv"

textract = boto3.client('textract', region_name=region_name)

# Display information about a block
def DisplayBlockInfo(block):
    print("Block Id: " + block['Id'])
    print("Type: " + block['BlockType'])
    if 'EntityTypes' in block:
        print('EntityTypes: {}'.format(block['EntityTypes']))

    if 'Text' in block:
        print("Text: " + block['Text'])

    if block['BlockType'] != 'PAGE':
        print("Confidence: " + "{:.2f}".format(block['Confidence']) + "%")

def GetResults(jobId, file_name):
    maxResults = 1000
    paginationToken = None
    finished = False

    while finished == False:

        response = None

        if paginationToken == None:
            response = textract.get_document_analysis(JobId=jobId,
 MaxResults=maxResults)
        else:
            response = textract.get_document_analysis(JobId=jobId,
 MaxResults=maxResults,

 NextToken=paginationToken)

        blocks = response['Blocks']
        table_csv = get_table_csv_results(blocks)
        output_file = file_name
        # replace content
        with open(output_file, "at") as fout:
```

```
            fout.write(table_csv)
        # show the results
        print('Detected Document Text')
        print('Pages: {}'.format(response['DocumentMetadata']['Pages']))
        print('OUTPUT TO CSV FILE: ', output_file)

        # Display block information
        for block in blocks:
            DisplayBlockInfo(block)
            print()
            print()

        if 'NextToken' in response:
            paginationToken = response['NextToken']
        else:
            finished = True


def get_rows_columns_map(table_result, blocks_map):
    rows = {}
    for relationship in table_result['Relationships']:
        if relationship['Type'] == 'CHILD':
            for child_id in relationship['Ids']:
                try:
                    cell = blocks_map[child_id]
                    if cell['BlockType'] == 'CELL':
                        row_index = cell['RowIndex']
                        col_index = cell['ColumnIndex']
                        if row_index not in rows:
                            # create new row
                            rows[row_index] = {}

                        # get the text value
                        rows[row_index][col_index] = get_text(cell, blocks_map)
                except KeyError:
                    print("Error extracting Table data - {}:".format(KeyError))
                    pass
    return rows


def get_text(result, blocks_map):
    text = ''
    if 'Relationships' in result:
        for relationship in result['Relationships']:
```

```
            if relationship['Type'] == 'CHILD':
                for child_id in relationship['Ids']:
                    try:
                        word = blocks_map[child_id]
                        if word['BlockType'] == 'WORD':
                            text += word['Text'] + ' '
                        if word['BlockType'] == 'SELECTION_ELEMENT':
                            if word['SelectionStatus'] == 'SELECTED':
                                text += 'X '
                    except KeyError:
                        print("Error extracting Table data -
 {}:".format(KeyError))

    return text


def get_table_csv_results(blocks):

    pprint(blocks)

    blocks_map = {}
    table_blocks = []
    for block in blocks:
        blocks_map[block['Id']] = block
        if block['BlockType'] == "TABLE":
            table_blocks.append(block)

    if len(table_blocks) <= 0:
        return "<b> NO Table FOUND </b>"

    csv = ''
    for index, table in enumerate(table_blocks):
        csv += generate_table_csv(table, blocks_map, index + 1)
        csv += '\n\n'

    return csv


def generate_table_csv(table_result, blocks_map, table_index):
    rows = get_rows_columns_map(table_result, blocks_map)

    table_id = 'Table_' + str(table_index)

    # get cells.
```

```
    csv = 'Table: {0}\n\n'.format(table_id)

    for row_index, cols in rows.items():

        for col_index, text in cols.items():
            csv += '{}'.format(text) + ","
        csv += '\n'

    csv += '\n\n\n'
    return csv

response_blocks = GetResults(jobId, file_name)
```

6. Exécutez le code.

   Une fois que vous avez obtenu vos résultats, veillez à supprimer les ressources SNS et SQS associées, sinon vous risquez d'accumuler des frais pour elles.

Création d'unAWS LambdaFonction

Vous pouvez appeler les opérations de l'API Amazon Textract depuis unAWS Lambda. Les instructions ci-dessous montrent comment créer une fonction Lambda en Python qui appellethe section called “DetectDocumentText”. Il renvoie une liste dethe section called “Block”objets. Pour exécuter cet exemple, vous devez disposer d'un compartiment Amazon S3 contenant un document au format PNG ou JPEG. Pour créer la fonction, vous utilisez la console.

Pour obtenir un exemple qui utilise les fonctions Lambda pour traiter des documents à grande échelle, voirTraitement de documents à grande échelle avec Amazon Textract.

Pour appeler l'opération DetectDocumentText à partir d'une fonction Lambda :

Étape 1 : Créer un package de déploiement Lambda

1. Ouvrez une fenêtre de commande.
2. Saisissez les commandes suivantes pour créer un package de déploiement avec la version la plus récente duAWSSDK.

```
pip install boto3 --target python/.
zip boto3-layer.zip -r python/
```

Étape 2 : Création d'une fonction Lambda

1. Connectez-vous à la AWS Management Console et ouvrez la console AWS Lambda à l'adresse https://console.aws.amazon.com/lambda/.
2. Sélectionnez Créer une fonction.
3. Spécifiez les paramètres suivants.
   - Choisissez Créer à partir de zéro.
   - Pour Nom de la fonction, entrez un nom.
   - PourRuntime (Exécution), choisissezPython 3.7ouPython 3.6.
   - PourChoisir ou créer un rôle d'exécution, choisissezCréer un nouveau rôle avec les autorisations Lambda de base.
4. ChoisissezCréer une fonctionpour créer la fonction Lambda.
5. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
6. Dans le volet de navigation, choisissezRôles.
7. Dans la liste des ressources, choisissez le rôle IAM que Lambda a créé pour vous. Le nom du rôle commence par le nom de votre fonction Lambda.
8. Cliquez sur l'ongletAutorisations, puis choisissezAttacher des stratégies.
9. Sélectionnez les stratégies AmazonTextractFullAccess et AmazonS3ReadOnlyAccess.
10. Tâche de sélectionAttacher une politique.

Pour de plus amples informations, veuillez consulterCréer une fonction Lambda avec la console

Étape 3 : Créer et ajouter une couche

1. Ouvrez la console AWS Lambda à l'adresse https://console.aws.amazon.com/lambda/.
2. Choisissez Layers dans le volet de navigation.
3. Sélectionnez Créer un calque.
4. PourNom, entrez un nom.
5. Pour Description, entrez une description.
6. PourType d'entrée de code, choisissezCharger le fichier .zipet sélectionnezCharger.

7. Dans la boîte de dialogue, sélectionnez le fichier zip (boto3-layer.zip), le zip que vous avez créé dansÉtape 1 : Créer un package de déploiement Lambda.
8. PourRuntimes compatibles, choisissez la version du moteur d'exécution que vous avez choisie dansÉtape 2 : Création d'une fonction Lambda.
9. ChoisissezCréerpour créer la couche.
10. Choisissez l'icône de menu du volet de navigation.
11. Dans le volet de navigation, choisissez Fonctions.
12. Dans la liste des ressources, sélectionnez la fonction que vous avez créée dansÉtape 2 : Création d'une fonction Lambda.
13. ChoisissezConfigurationet dans leDesigner, choisissezCouches(sous le nom de votre fonction Lambda).
14. DansCouches, choisissezAjout d'une couche.
15. ChoisissezSélectionnez dans la liste des couches compatibles avec l'environnement d'exécution.
16. DansCouches compatibles, sélectionnez laNometVersiondu calque que vous avez créé à l'étape 3.
17. Choisissez Add (Ajouter).

Étape 4 : Ajouter du code Python à la fonction

1. DansDesigner, choisissez votre fonction.
2. Dans l'éditeur de code, ajoutez les éléments suivants au fichierlambda_function.py. Modifiez les valeurs des`bucket`et`document`à votre compartiment et à votre document.

```
import json
import boto3

def lambda_handler(event, context):

    bucket="bucket"
    document="document"
    client = boto3.client('textract')



    #process using S3 object
    response = client.detect_document_text(
        Document={'S3Object': {'Bucket': bucket, 'Name': document}})
```

```
    #Get the text blocks
    blocks=response['Blocks']

    return {
        'statusCode': 200,
        'body': json.dumps(blocks)
    }
```

3. ChoisissezEnregistrerpour enregistrer votre fonction Lambda.

Étape 5 : Testez votre Lambda

1. Tâche de sélectionTest.
2. Saisir une valeur pourNom de l'événement.
3. Sélectionnez Create (Créer) Application Load Balancer request count per target.
4. La sortie, une liste dethe section called “Block”, apparaît dans le volet des résultats de l'exécution.

Si l'icôneAWS Lambdarenvoie une erreur de délai d'attente, un appel d'opération de l'API Amazon Textract peut en être la cause. Pour plus d'informations sur la prolongation du délai d'expiration d'uneAWS Lambdafonction, voirConfiguration d'une fonction AWS Lambda.

Pour plus d'informations sur l'appel d'une fonction Lambda à partir de votre code, consultezInvoquerAWS LambdaFonctions.

Exemples de code supplémentaires

Le tableau suivant fournit des liens vers d'autres exemples de code Amazon Textract.

Exemple	Description
Exemples de Code Amazon Textract	Affichez différentes façons d'utiliser Amazon Textract.

Exemple	Description
Traitement de documents à grande échelle avec Amazon Textract	Affiche une architecture de référence sans serveur qui traite les documents à grande échelle.
Analyseur Amazon Textract	Montre comment analyser la fonctionthe section called “Block”objets renvoyés par les opérations Amazon Textract.
Exemples de code de documentation Amazon Textract	Exemples de code utilisés dans ce guide.
Extracteur	Indique comment convertir la sortie Amazon Textract en plusieurs formats.
Générez des documents PDF interrogeables avec Amazon Textract	Indique comment créer un document PDF interrogeable à partir de différents types de documents d'entrée tels que des images au format JPG/PNG et des documents PDF numérisés.

Exemples de code pour Amazon Textract

Les exemples de code suivants montrent comment utiliser Amazon Textract avecAWSkit de développement logiciel (SDK).

Les exemples sont répartis dans les catégories suivantes :

Actions

Des extraits de code qui vous montrent comment appeler des fonctions de service individuelles.

Exemples de services croisés

Exemples d'applications fonctionnant sur plusieursAWSServices .

Pour obtenir la liste complète desAWSGuides de développement SDK et exemples de code, voirUtilisation d'Amazon Textract avec unAWSKIT DE DÉVELOPPEMENT LOGICIEL. Cette rubrique inclut également des informations sur le démarrage et des détails sur les versions précédentes du SDK.

Exemples de code

- Actions pour Amazon Textract
  - Analyser un document à l'aide d'Amazon Textract et d'unAWSKIT DE DÉVELOPPEMENT LOGICIEL
  - Détecter du texte dans un document à l'aide d'Amazon Textract et d'unAWSKIT DE DÉVELOPPEMENT LOGICIEL
  - Obtenez des données sur une tâche d'analyse de documents Amazon Textract à l'aide d'unAWSKIT DE DÉVELOPPEMENT LOGICIEL
  - Commencez l'analyse asynchrone d'un document à l'aide d'Amazon Textract et d'unAWSKIT DE DÉVELOPPEMENT LOGICIEL
  - Démarrez la détection de texte asynchrone à l'aide d'Amazon Textract et d'unAWSKIT DE DÉVELOPPEMENT LOGICIEL
- Exemples de services pour Amazon Textract
  - Créer une application Amazon Textract Explorer
  - Détecter les entités dans du texte extrait d'une image à l'aide d'unAWSKIT DE DÉVELOPPEMENT LOGICIEL

Actions pour Amazon Textract

Les exemples de code suivants montrent comment effectuer des actions Amazon Textract individuelles avecAWSKits SDK. Ces extraits appellent l'API Amazon Textract et ne sont pas destinés à être exécutés isolément. Chaque exemple inclut un lien vers GitHub, où vous trouverez des instructions sur la configuration et l'exécution du code en contexte.

Les exemples suivants incluent uniquement les actions les plus couramment utilisées. Pour obtenir la liste complète, consultezRéférence d'API Amazon Textract.

Exemples

- Analyser un document à l'aide d'Amazon Textract et d'unAWSKIT DE DÉVELOPPEMENT LOGICIEL
- Détecter du texte dans un document à l'aide d'Amazon Textract et d'unAWSKIT DE DÉVELOPPEMENT LOGICIEL
- Obtenez des données sur une tâche d'analyse de documents Amazon Textract à l'aide d'unAWSKIT DE DÉVELOPPEMENT LOGICIEL
- Commencez l'analyse asynchrone d'un document à l'aide d'Amazon Textract et d'unAWSKIT DE DÉVELOPPEMENT LOGICIEL
- Démarrez la détection de texte asynchrone à l'aide d'Amazon Textract et d'unAWSKIT DE DÉVELOPPEMENT LOGICIEL

Analyser un document à l'aide d'Amazon Textract et d'unAWSKIT DE DÉVELOPPEMENT LOGICIEL

Les exemples de code suivants montrent comment analyser un document à l'aide d'Amazon Textract.

Java

SDK pour Java 2.x

```
   public static void analyzeDoc(TextractClient textractClient, String
sourceDoc) {

       try {
           InputStream sourceStream = new FileInputStream(new File(sourceDoc));
```

```
            SdkBytes sourceBytes = SdkBytes.fromInputStream(sourceStream);

            // Get the input Document object as bytes
            Document myDoc = Document.builder()
                    .bytes(sourceBytes)
                    .build();

            List<FeatureType> featureTypes = new ArrayList<FeatureType>();
            featureTypes.add(FeatureType.FORMS);
            featureTypes.add(FeatureType.TABLES);

            AnalyzeDocumentRequest analyzeDocumentRequest =
AnalyzeDocumentRequest.builder()
                    .featureTypes(featureTypes)
                    .document(myDoc)
                    .build();

            AnalyzeDocumentResponse analyzeDocument =
textractClient.analyzeDocument(analyzeDocumentRequest);
            List<Block> docInfo = analyzeDocument.blocks();
            Iterator<Block> blockIterator = docInfo.iterator();

            while(blockIterator.hasNext()) {
                Block block = blockIterator.next();
                System.out.println("The block type is "
+block.blockType().toString());
            }

       } catch (TextractException | FileNotFoundException e) {

            System.err.println(e.getMessage());
            System.exit(1);
       }
   }
```

- Trouvez des instructions et plus de code sur GitHub.
- Pour plus d'informations sur l'API, consultezAnalyzeDocumentdansAWS SDK for Java 2.xAPI Reference.

Python

SDK pour Python (Boto3)

```
class TextractWrapper:
    """Encapsulates Textract functions."""
    def __init__(self, textract_client, s3_resource, sqs_resource):
        """
        :param textract_client: A Boto3 Textract client.
        :param s3_resource: A Boto3 Amazon S3 resource.
        :param sqs_resource: A Boto3 Amazon SQS resource.
        """
        self.textract_client = textract_client
        self.s3_resource = s3_resource
        self.sqs_resource = sqs_resource

    def analyze_file(
            self, feature_types, *, document_file_name=None,
 document_bytes=None):
        """
        Detects text and additional elements, such as forms or tables, in a local
 image
        file or from in-memory byte data.
        The image must be in PNG or JPG format.

        :param feature_types: The types of additional document features to
 detect.
        :param document_file_name: The name of a document image file.
        :param document_bytes: In-memory byte data of a document image.
        :return: The response from Amazon Textract, including a list of blocks
                 that describe elements detected in the image.
        """
        if document_file_name is not None:
            with open(document_file_name, 'rb') as document_file:
                document_bytes = document_file.read()
        try:
            response = self.textract_client.analyze_document(
                Document={'Bytes': document_bytes}, FeatureTypes=feature_types)
            logger.info(
                "Detected %s blocks.", len(response['Blocks']))
        except ClientError:
            logger.exception("Couldn't detect text.")
            raise
```

```
        else:
            return response
```

- Trouvez des instructions et plus de code sur GitHub.
- Pour plus d'informations sur l'API, consultezAnalyzeDocumentdansAWSRéférence d'API SDK for Python (Boto3).

Pour obtenir la liste complète desAWSGuides de développement SDK et exemples de code, voirUtilisation d'Amazon Textract avec unAWSKIT DE DÉVELOPPEMENT LOGICIEL. Cette rubrique inclut également des informations sur le démarrage et des détails sur les versions précédentes du SDK.

Détecter du texte dans un document à l'aide d'Amazon Textract et d'unAWSKIT DE DÉVELOPPEMENT LOGICIEL

Les exemples de code suivants montrent comment détecter du texte dans un document à l'aide d'Amazon Textract.

Java

SDK pour Java 2.x

Détecte le texte d'un document d'entrée.

```
   public static void detectDocText(TextractClient textractClient,String
sourceDoc) {

       try {

           InputStream sourceStream = new FileInputStream(new File(sourceDoc));
           SdkBytes sourceBytes = SdkBytes.fromInputStream(sourceStream);

           // Get the input Document object as bytes
           Document myDoc = Document.builder()
                   .bytes(sourceBytes)
                   .build();

           DetectDocumentTextRequest detectDocumentTextRequest =
DetectDocumentTextRequest.builder()
```

```
                    .document(myDoc)
                    .build();

            // Invoke the Detect operation
            DetectDocumentTextResponse textResponse =
textractClient.detectDocumentText(detectDocumentTextRequest);

            List<Block> docInfo = textResponse.blocks();

            Iterator<Block> blockIterator = docInfo.iterator();

            while(blockIterator.hasNext()) {
                Block block = blockIterator.next();
                System.out.println("The block type is "
+block.blockType().toString());
            }

            DocumentMetadata documentMetadata = textResponse.documentMetadata();
            System.out.println("The number of pages in the document is "
+documentMetadata.pages());

        } catch (TextractException | FileNotFoundException e) {

            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

Détecte du texte d'un document situé dans un compartiment Amazon S3.

```
    public static void detectDocTextS3 (TextractClient textractClient, String
bucketName, String docName) {

        try {
            S3Object s3Object = S3Object.builder()
                    .bucket(bucketName)
                    .name(docName)
                    .build();

            // Create a Document object and reference the s3Object instance
            Document myDoc = Document.builder()
                    .s3Object(s3Object)
```

```
                    .build();

            // Create a DetectDocumentTextRequest object
            DetectDocumentTextRequest detectDocumentTextRequest =
DetectDocumentTextRequest.builder()
                    .document(myDoc)
                    .build();

            // Invoke the detectDocumentText method
            DetectDocumentTextResponse textResponse =
textractClient.detectDocumentText(detectDocumentTextRequest);

            List<Block> docInfo = textResponse.blocks();

            Iterator<Block> blockIterator = docInfo.iterator();

            while(blockIterator.hasNext()) {
                Block block = blockIterator.next();
                System.out.println("The block type is "
+block.blockType().toString());
            }

            DocumentMetadata documentMetadata = textResponse.documentMetadata();
            System.out.println("The number of pages in the document is "
+documentMetadata.pages());

        } catch (TextractException e) {

            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

- Trouvez des instructions et plus de code sur GitHub.
- Pour plus d'informations sur l'API, consultezDetectDocumentTextdansAWS SDK for Java 2.xAPI Reference.

Python

SDK pour Python (Boto3)

```
class TextractWrapper:
    """Encapsulates Textract functions."""
    def __init__(self, textract_client, s3_resource, sqs_resource):
        """
        :param textract_client: A Boto3 Textract client.
        :param s3_resource: A Boto3 Amazon S3 resource.
        :param sqs_resource: A Boto3 Amazon SQS resource.
        """
        self.textract_client = textract_client
        self.s3_resource = s3_resource
        self.sqs_resource = sqs_resource

    def detect_file_text(self, *, document_file_name=None, document_bytes=None):
        """
        Detects text elements in a local image file or from in-memory byte data.
        The image must be in PNG or JPG format.

        :param document_file_name: The name of a document image file.
        :param document_bytes: In-memory byte data of a document image.
        :return: The response from Amazon Textract, including a list of blocks
                 that describe elements detected in the image.
        """
        if document_file_name is not None:
            with open(document_file_name, 'rb') as document_file:
                document_bytes = document_file.read()
        try:
            response = self.textract_client.detect_document_text(
                Document={'Bytes': document_bytes})
            logger.info(
                "Detected %s blocks.", len(response['Blocks']))
        except ClientError:
            logger.exception("Couldn't detect text.")
            raise
        else:
            return response
```

- Trouvez des instructions et plus de code sur GitHub.
- Pour plus d'informations sur l'API, consultezDetectDocumentTextdansAWSRéférence d'API SDK for Python (Boto3).

Pour obtenir la liste complète desAWSGuides de développement SDK et exemples de code, voirUtilisation d'Amazon Textract avec unAWSKIT DE DÉVELOPPEMENT LOGICIEL. Cette rubrique inclut également des informations sur le démarrage et des détails sur les versions précédentes du SDK.

Obtenez des données sur une tâche d'analyse de documents Amazon Textract à l'aide d'unAWSKIT DE DÉVELOPPEMENT LOGICIEL

L'exemple de code suivant montre comment obtenir des données sur une tâche d'analyse de document Amazon Textract.

Python

SDK pour Python (Boto3)

```
class TextractWrapper:
    """Encapsulates Textract functions."""
    def __init__(self, textract_client, s3_resource, sqs_resource):
        """
        :param textract_client: A Boto3 Textract client.
        :param s3_resource: A Boto3 Amazon S3 resource.
        :param sqs_resource: A Boto3 Amazon SQS resource.
        """
        self.textract_client = textract_client
        self.s3_resource = s3_resource
        self.sqs_resource = sqs_resource

    def get_analysis_job(self, job_id):
        """
        Gets data for a previously started detection job that includes additional
        elements.

        :param job_id: The ID of the job to retrieve.
        :return: The job data, including a list of blocks that describe elements
                 detected in the image.
        """
        try:
            response = self.textract_client.get_document_analysis(
                JobId=job_id)
            job_status = response['JobStatus']
            logger.info("Job %s status is %s.", job_id, job_status)
```

```
        except ClientError:
            logger.exception("Couldn't get data for job %s.", job_id)
            raise
        else:
            return response
```

- Trouvez des instructions et plus de code sur GitHub.
- Pour plus d'informations sur l'API, consultezGetDocumentAnalysisdansAWSRéférence d'API SDK for Python (Boto3).

Pour obtenir la liste complète desAWSGuides de développement SDK et exemples de code, voirUtilisation d'Amazon Textract avec unAWSKIT DE DÉVELOPPEMENT LOGICIEL. Cette rubrique inclut également des informations sur le démarrage et des détails sur les versions précédentes du SDK.

Commencez l'analyse asynchrone d'un document à l'aide d'Amazon Textract et d'unAWSKIT DE DÉVELOPPEMENT LOGICIEL

Les exemples de code suivants montrent comment démarrer l'analyse asynchrone d'un document à l'aide d'Amazon Textract.

Java

SDK pour Java 2.x

```
   public static String startDocAnalysisS3 (TextractClient textractClient,
String bucketName, String docName) {

        try {

            List<FeatureType> myList = new ArrayList<FeatureType>();
            myList.add(FeatureType.TABLES);
            myList.add(FeatureType.FORMS);

            S3Object s3Object = S3Object.builder()
                    .bucket(bucketName)
                    .name(docName)
                    .build();
```

```
            DocumentLocation location = DocumentLocation.builder()
                    .s3Object(s3Object)
                    .build();

            StartDocumentAnalysisRequest documentAnalysisRequest =
StartDocumentAnalysisRequest.builder()
                    .documentLocation(location)
                    .featureTypes(myList)
                    .build();

            StartDocumentAnalysisResponse response =
textractClient.startDocumentAnalysis(documentAnalysisRequest);

            // Get the job ID
            String jobId = response.jobId();
            return jobId;

        } catch (TextractException e) {

            System.err.println(e.getMessage());
            System.exit(1);
        }
        return "" ;
    }

    private static String getJobResults(TextractClient textractClient, String
jobId) {

        boolean finished = false;
        int index = 0 ;
        String status = "" ;

       try {
        while (!finished) {
            GetDocumentAnalysisRequest analysisRequest =
GetDocumentAnalysisRequest.builder()
                    .jobId(jobId)
                    .maxResults(1000)
                    .build();

            GetDocumentAnalysisResponse response =
textractClient.getDocumentAnalysis(analysisRequest);
            status = response.jobStatus().toString();
```

```
            if (status.compareTo("SUCCEEDED") == 0)
                finished = true;
            else {
                System.out.println(index + " status is: " + status);
                Thread.sleep(1000);
            }
            index++ ;
        }
        return status;

       } catch( InterruptedException e) {
           System.out.println(e.getMessage());
           System.exit(1);
       }
       return "";
    }
```

- Trouvez des instructions et plus de code sur GitHub.
- Pour plus d'informations sur l'API, consultezStartDocumentAnalysisdansAWS SDK for Java 2.xAPI Reference.

Python

SDK pour Python (Boto3)

Démarrez une tâche asynchrone pour analyser un document.

```
class TextractWrapper:
    """Encapsulates Textract functions."""
    def __init__(self, textract_client, s3_resource, sqs_resource):
        """
        :param textract_client: A Boto3 Textract client.
        :param s3_resource: A Boto3 Amazon S3 resource.
        :param sqs_resource: A Boto3 Amazon SQS resource.
        """
        self.textract_client = textract_client
        self.s3_resource = s3_resource
        self.sqs_resource = sqs_resource

    def start_analysis_job(
            self, bucket_name, document_file_name, feature_types, sns_topic_arn,
```

```
            sns_role_arn):
        """
        Starts an asynchronous job to detect text and additional elements, such
as
        forms or tables, in an image stored in an Amazon S3 bucket. Textract
publishes
        a notification to the specified Amazon SNS topic when the job completes.
        The image must be in PNG, JPG, or PDF format.

        :param bucket_name: The name of the Amazon S3 bucket that contains the
image.
        :param document_file_name: The name of the document image stored in
Amazon S3.
        :param feature_types: The types of additional document features to
detect.
        :param sns_topic_arn: The Amazon Resource Name (ARN) of an Amazon SNS
topic
                              where job completion notification is published.
        :param sns_role_arn: The ARN of an AWS Identity and Access Management
(IAM)
                             role that can be assumed by Textract and grants
permission
                             to publish to the Amazon SNS topic.
        :return: The ID of the job.
        """
        try:
            response = self.textract_client.start_document_analysis(
                DocumentLocation={
                    'S3Object': {'Bucket': bucket_name, 'Name':
document_file_name}},
                NotificationChannel={
                    'SNSTopicArn': sns_topic_arn, 'RoleArn': sns_role_arn},
                FeatureTypes=feature_types)
            job_id = response['JobId']
            logger.info(
                "Started text analysis job %s on %s.", job_id,
document_file_name)
        except ClientError:
            logger.exception("Couldn't analyze text in %s.", document_file_name)
            raise
        else:
            return job_id
```

- Trouvez des instructions et plus de code sur GitHub.
- Pour plus d'informations sur l'API, consultezStartDocumentAnalysisdansAWSRéférence d'API SDK for Python (Boto3).

Pour obtenir la liste complète desAWSGuides de développement SDK et exemples de code, voirUtilisation d'Amazon Textract avec unAWSKIT DE DÉVELOPPEMENT LOGICIEL. Cette rubrique inclut également des informations sur le démarrage et des détails sur les versions précédentes du SDK.

Démarrez la détection de texte asynchrone à l'aide d'Amazon Textract et d'unAWSKIT DE DÉVELOPPEMENT LOGICIEL

L'exemple de code suivant montre comment démarrer la détection de texte asynchrone dans un document à l'aide d'Amazon Textract.

Python

SDK pour Python (Boto3)

Démarre une tâche asynchrone pour détecter du texte dans un document.

```
class TextractWrapper:
    """Encapsulates Textract functions."""
    def __init__(self, textract_client, s3_resource, sqs_resource):
        """
        :param textract_client: A Boto3 Textract client.
        :param s3_resource: A Boto3 Amazon S3 resource.
        :param sqs_resource: A Boto3 Amazon SQS resource.
        """
        self.textract_client = textract_client
        self.s3_resource = s3_resource
        self.sqs_resource = sqs_resource

    def start_detection_job(
            self, bucket_name, document_file_name, sns_topic_arn, sns_role_arn):
        """
        Starts an asynchronous job to detect text elements in an image stored in
 an
        Amazon S3 bucket. Textract publishes a notification to the specified
 Amazon SNS
        topic when the job completes.
```

```
        The image must be in PNG, JPG, or PDF format.

        :param bucket_name: The name of the Amazon S3 bucket that contains the
image.
        :param document_file_name: The name of the document image stored in
Amazon S3.
        :param sns_topic_arn: The Amazon Resource Name (ARN) of an Amazon SNS
topic
                              where the job completion notification is published.
        :param sns_role_arn: The ARN of an AWS Identity and Access Management
(IAM)
                             role that can be assumed by Textract and grants
permission
                             to publish to the Amazon SNS topic.
        :return: The ID of the job.
        """
        try:
            response = self.textract_client.start_document_text_detection(
                DocumentLocation={
                    'S3Object': {'Bucket': bucket_name, 'Name':
document_file_name}},
                NotificationChannel={
                    'SNSTopicArn': sns_topic_arn, 'RoleArn': sns_role_arn})
            job_id = response['JobId']
            logger.info(
                "Started text detection job %s on %s.", job_id,
document_file_name)
        except ClientError:
            logger.exception("Couldn't detect text in %s.", document_file_name)
            raise
        else:
            return job_id
```

- Trouvez des instructions et plus de code sur GitHub.
- Pour plus d'informations sur l'API, consultezStartDocumentTextDetectiondansAWSRéférence d'API SDK for Python (Boto3).

Pour obtenir la liste complète desAWSGuides de développement SDK et exemples de code, voirUtilisation d'Amazon Textract avec unAWSKIT DE DÉVELOPPEMENT LOGICIEL. Cette rubrique inclut également des informations sur le démarrage et des détails sur les versions précédentes du SDK.

Exemples de services pour Amazon Textract

Les exemples d'applications suivants utilisentAWSSDK pour combiner Amazon Textract avec d'autresAWSServices . Chaque exemple inclut un lien vers GitHub, où vous pouvez trouver des instructions sur la configuration et l'exécution de l'application.

Exemples

- Créer une application Amazon Textract Explorer
- Détecter les entités dans du texte extrait d'une image à l'aide d'unAWSKIT DE DÉVELOPPEMENT LOGICIEL

Créer une application Amazon Textract Explorer

Les exemples de code suivants expliquent comment explorer la sortie Amazon Textract via une application interactive.

JavaScript

SDK pour JavaScript V3

Indique comment utiliser l'outilAWS SDK pour JavaScript pour créer une application React qui utilise Amazon Textract pour extraire des données d'une image de document et les afficher dans une page Web interactive. Cet exemple s'exécute dans un navigateur Web et nécessite une identité Amazon Cognito authentifiée pour les informations d'identification. Il utilise Amazon Simple Storage Service (Amazon S3) pour le stockage et, pour les notifications, il interroge une file d'attente Amazon Simple Queue Service (Amazon SQS) abonnée à une rubrique Amazon Simple Notification Service (Amazon SNS).

Pour obtenir un code source complet et des instructions sur la configuration et l'exécution, consultez l'exemple complet sur GitHub.

Les services utilisés dans cet exemple

- Amazon Cognito Identity
- Amazon S3
- Amazon SNS
- Amazon SQS
- Amazon Textract

Python

SDK pour Python (Boto3)

Indique comment utiliser l'outil AWS SDK pour Python (Boto3) avec Amazon Textract pour détecter le texte, le formulaire et les éléments de tableau dans une image de document. L'image d'entrée et la sortie d'Amazon Textract sont affichées dans une application Tkinter qui vous permet d'explorer les éléments détectés.

- Soumettez une image de document à Amazon Textract et explorez la sortie des éléments détectés.
- Soumettez des images directement à Amazon Textract ou via un compartiment Amazon Simple Storage Service (Amazon S3).
- Utilisez des API asynchrones pour démarrer une tâche qui publie une notification dans une rubrique Amazon Simple Notification Service (Amazon SNS) lorsque le travail est terminé.
- Interrogez un service Amazon Simple Queue Service (Amazon SQS) pour obtenir un message de fin de tâche et affichez les résultats.

Pour obtenir un code source complet et des instructions sur la configuration et l'exécution, consultez l'exemple complet sur GitHub.

Les services utilisés dans cet exemple

- Amazon S3
- Amazon SNS
- Amazon SQS
- Amazon Textract

Pour obtenir la liste complète desAWSGuides de développement SDK et exemples de code, voirUtilisation d'Amazon Textract avec unAWSKIT DE DÉVELOPPEMENT LOGICIEL. Cette rubrique inclut également des informations sur le démarrage et des détails sur les versions précédentes du SDK.

Détecter les entités dans du texte extrait d'une image à l'aide d'unAWSKIT DE DÉVELOPPEMENT LOGICIEL

L'exemple de code suivant montre comment utiliser Amazon Comprehend pour détecter des entités dans du texte extrait par Amazon Textract à partir d'une image stockée dans Amazon S3.

Python

SDK pour Python (Boto3)

Indique comment utiliser laAWS SDK pour Python (Boto3)dans un bloc-notes Jupyter pour détecter les entités du texte extrait d'une image. Cet exemple utilise Amazon Textract pour extraire du texte d'une image stockée dans Amazon Simple Storage Service (Amazon S3) et Amazon Comprehend pour détecter les entités dans le texte extrait.

Cet exemple est un bloc-notes Jupyter et doit être exécuté dans un environnement pouvant héberger des ordinateurs portables. Pour obtenir des instructions sur l'exécution de l'exemple à l'aide d'Amazon SageMaker, consultez.Textract and Comprehend Notebook. iPy NB.

Pour obtenir un code source complet et des instructions sur la configuration et l'exécution, consultez l'exemple complet sur GitHub.

Les services utilisés dans cet exemple

- Amazon Comprehend
- Amazon S3
- Amazon Textract

Pour obtenir la liste complète desAWSGuides de développement SDK et exemples de code, voirUtilisation d'Amazon Textract avec unAWSKIT DE DÉVELOPPEMENT LOGICIEL. Cette rubrique inclut également des informations sur le démarrage et des détails sur les versions précédentes du SDK.

Utilisation d'Amazon Augmented AI pour ajouter un avis humain à la sortie Amazon Textract

Amazon Augmented AI (Amazon A2I) est un service d'apprentissage automatique (ML) qui facilite la création de workflows pour la vérification humaine des analyses ML.

Amazon Textract est intégré à Amazon A2I. Vous pouvez l'utiliser pour acheminer les résultats de l'analyse de documents dont le score de confiance est faible vers les réviseurs humains.

Vous pouvez utiliser Amazon Textract`AnalyzeDocument`API pour extraire des données des formulaires et de la console Amazon A2I. Vous pouvez spécifier les conditions dans lesquelles Amazon A2I achemine les prévisions vers les réviseurs. Vous définissez des conditions en fonction du seuil de confiance des clés de formulaire importantes. Par exemple, vous pouvez envoyer un document à un réviseur humain si la cléNomou sa valeur associéeJane Donea été détecté avec une faible confiance.

Rubriques

- Principaux concepts d'Amazon A2I
- Commencez à utiliser Amazon A2I

Principaux concepts d'Amazon A2I

Examinez les termes suivants pour vous familiariser avec les concepts principaux d'Amazon A2I.

Conditions d'activation de la vérification humaine

Vous pouvez utiliser Amazon A2Iconditions d'activationpour spécifier quand un document est envoyé aux humains pour examen et le contenu du formulaire que les travailleurs sont invités à réviser.

Par exemple, vous pouvez définir une condition d'activation pour qu'Amazon Textract achemine les formulaires vers Amazon A2I lorsqu'une clé importante est détectée avec une faible confiance, commeNuméro de téléphone. Dans cet exemple, les évaluateurs humains sont invités à revoir leNuméro de téléphonechamp et valeur associés détectés par Amazon Textract.

Vous pouvez utiliser les conditions d'activation suivantes pour spécifier quand les formulaires sont envoyés aux humains pour révision :

- Déclenchement d'une vérification humaine pour des clés de formulaire spécifiques en fonction de l'indice de confiance de la clé de formulaire. Les vérificateurs humains sont chargés de passer en revue ces clés de formulaire et les valeurs associées.
- Déclenchement d'une vérification humaine lorsque des clés de formulaire spécifiques sont manquantes. Les réviseurs humains sont invités à identifier ces clés de formulaire et les valeurs associées.
- Déclenchement d'une vérification humaine pour toutes les clés de formulaire identifiées par Amazon Textract avec des indices de confiance situés dans une plage spécifiée.
- Envoi aléatoire d'un échantillon de formulaires aux collaborateurs humains pour vérification. Les vérificateurs humains sont chargés de passer en revue toutes les clés de formulaire et toutes les valeurs détectées par Amazon Textract.

Lorsque votre condition d'activation dépend des indices de confiance des clés de formulaire, vous pouvez utiliser deux types de seuils de confiance prédictive pour déclencher une vérification humaine :

- Confiance d'identification : l'indice de confiance des paires clé-valeur détectées dans un formulaire.
- Confiance à— Indice de confiance du texte contenu dans une paire clé-valeur d'un formulaire.

Si vous spécifiez un seuil de confiance, Amazon A2I achemine uniquement les prévisions qui se situent dans le seuil vers les réviseurs humains. Vous pouvez ajuster ces seuils à tout moment pour obtenir le juste équilibre entre précision et rentabilité. Cela peut vous aider à mettre en œuvre des audits pour surveiller régulièrement l'exactitude des prévisions.

> **Note**
>
> Vous pouvez également personnaliser les conditions dans lesquelles les documents sont envoyés aux humains pour révision à l'aide du type de tâche personnalisée Amazon A2I. Avec ce type de tâche, vous spécifiez les conditions d'une évaluation humaine directement dans votre application. Pour plus d'informations, consultezUtiliser Amazon Augmented AI avec des types de tâches personnalisésdans le Manuel du développeur Amazon SageMaker.

Workflow de révision humaine (définition de flux)

Vous utilisez unflux de vérification humaine, également appelédéfinition de flux, pour spécifier les ressources utilisées pour créer votre flux de travail de révision humaine et pour spécifier vos conditions d'activation.

Les ressources que vous spécifiez sont les suivantes :

- Un rôle IAM avec autorisation d'appeler des opérations d'API Amazon A2I
- Compartiment Amazon S3 dans lequel vous souhaitez stocker la sortie de l'avis humain.
- Votre humainéquipe de travail
- UNModèle de tâche de travailqui comprend des instructions et des exemples pour aider les travailleurs à terminer la tâche de révision

Vous utilisez également le flux de travail de révision humaine pour spécifier les conditions d'activation. Pour plus d'informations, consultez Conditions d'activation de la vérification humaine.

Vous pouvez utiliser un flux de travail de vérification humaine unique pour créer plusieursBoucles humaines.

Vous pouvez créer un flux de vérification humaine dans la console SageMaker ou avec l'API SageMaker. Pour plus d’informations, veuillez consulter Créer un flux de vérification humaine.

Modèle de tâche de travailleur

Vous utilisez unModèle de tâche de travailpour créer une interface utilisateur de travail utilisée pour vos tâches de révision humaines.

L'interface utilisateur du travailleur affiche vos documents et les instructions du travailleur. Elle fournit également des outils que les employés utilisent pour effectuer vos tâches.

Vous pouvez utiliser la console SageMaker pour configurer votre modèle de tâche de travail lorsque vous créez un flux de vérification humaine. Pour plus d’informations, veuillez consulter Créer un flux de vérification humaine.

Équipe de travail

UNéquipe de travailest un groupe de travailleurs humains à qui vous envoyez vos tâches de vérification humaine.

Lorsque vous créez un flux de vérification humaine, vous spécifiez une équipe de travail unique.

Avec Amazon A2I, vous pouvez utiliser un pool d'évaluateurs au sein de votre propre organisation. Vous pouvez également accéder à la main-d'œuvre composée de plus de 500 000 entrepreneurs indépendants qui effectuent déjà des tâches d'apprentissage automatique via Amazon Mechanical Turk. Une autre option consiste à recourir à des vendeurs de main-d'œuvre présélectionnés par AWS pour garantir la qualité et le respect des procédures de sécurité.

Amazon A2I fournit également aux réviseurs une interface Web comprenant toutes les instructions et tous les outils dont ils ont besoin pour effectuer leurs tâches de révision.

Pour chaque type de main-d'œuvre (privée, fournisseur et Mechanical Turk), vous pouvez créer plusieurs équipes de travail. Vous pouvez utiliser chaque équipe de travail dans plusieurs flux de vérification humaine. Pour savoir comment créer une main-d'œuvre et des équipes de travail, consultezCréation et gestion de mains-d'œuvresdans le Manuel du développeur Amazon SageMaker.

> **Important**
>
> Cliquez suricipour voir les programmes de conformité qui s'appliquent à Amazon Augmented AI. Si vous utilisez Amazon Augmented AI conjointement avec d'autres services AWS (comme Amazon Rekognition et Amazon Textract), veuillez noter qu'Amazon Augmented AI peut ne pas être concerné par les mêmes programmes de conformité que ces autres services. Vous êtes responsable de la façon dont vous utilisez Amazon Augmented AI. Il vous incombe notamment de comprendre comment le service traitera ou stocke les données des clients, ainsi que l'impact sur la conformité de votre environnement de données. Vous devriez discuter de vos objectifs de charge de travail avec votre équipe de compte AWS. Ils peuvent vous aider à évaluer si le service convient au cas d'utilisation et à l'architecture que vous proposez.
> Actuellement, Amazon Augmented AI est conforme à la norme PCI, à l'exception des cas publics et fournisseurs.
> Pour plus d'informations sur la conformité Amazon Augmented AI HIPAA, cliquez surici.

Boucles humaines

Vous utilisez une boucle humaine pour créer une tâche de vérification humaine.

Vous attribuez une tâche de révision humaine à un travailleur de votre équipe de travailleurs humains. Le travailleur est invité à examiner les paires clé-valeur détectées par Amazon Textract dans votre document d'entrée que vous avez spécifié dans vos conditions d'activation.

Par exemple, supposons qu'une image se situe entre cinquante et soixante pour cent de confiance qu'elle contient une pomme. Cela correspond à votre seuil de confiance pour l'examen humain et est envoyé à un travailleur. Le travailleur vérifie la présence d'une pomme dans le document, en le marquant comme contenant ou non une pomme. Amazon A2I renvoie ensuite le document dans le flux de travail.

Lorsque vous appelez Amazon Textract`AnalyzeDocument`et spécifiez un flux de travail de révision humaine (définition de flux) et un nom de boucle humaine, une tâche de révision humaine est créée lorsque les conditions d'activation spécifiées dans votre flux de travail de révision humaine sont remplies. Ces tâches sont créées à l'aide des ressources que vous spécifiez dans votre flux de travail de révision humaine.

Commencez à utiliser Amazon A2I

Les étapes suivantes vous aident à intégrer Amazon A2I dans une tâche d'analyse de documents d'une seule page Amazon Textract. Vous effectuez les actions suivantes :

1. Créez un flux de travail de révision humaine à l'aide de la console Amazon A2I (recommandée pour les nouveaux utilisateurs) ou de l'API Amazon A2I.
2. Pour analyser un formulaire et inclure un examen humain si nécessaire, utilisez le`AnalyzeDocument`et spécifiez l'Amazon Resource Name (ARN) du flux de vérification humaine. La réponse vous indique si une vérification humaine est nécessaire.
3. Surveillez votre boucle humaine à l'aide de la console et de l'API Amazon A2I.
4. Passez en revue les résultats de la révision humaine dans un compartiment Amazon S3 où les résultats sont envoyés.

Pour configurer une instance de notebook SageMaker et utiliser un exemple de bloc-notes, voirDémonstration de bout en bout sur l'utilisation d'Amazon Textract et d'Augmented AIdans leAmazon SageMaker Developer Guide

> **Note**
>
> Cette section explique comment créer un flux de vérification humaine pour le type de tâche Amazon A2I, Amazon Textract. Pour personnaliser davantage l'intégration Amazon A2I et Amazon Textract, vous pouvez utiliser le type de tâche personnalisée Amazon A2I. Avec cette option, vous fournissez un modèle de tâche de travail personnalisé et vous spécifiez les conditions dans lesquelles un document est envoyé pour vérification humaine directement

dans votre application. Pour plus d'informations, consultezUtiliser Amazon Augmented AI avec des types de tâches personnalisésdans leAmazon SageMaker Developer Guide.

Rubriques

- Créer un flux de vérification humaine
- Analyse du document
- Surveillance de la boucle humaine
- Afficher les données de sortie et les mesures de travail

Créer un flux de vérification humaine

Vous pouvez créer un flux de vérification humaine à l'aide de la console Amazon A2I (recommandée pour les nouveaux utilisateurs) ou d'Amazon A2I.`CreateFlowDefinition`.

Rubriques

- Créer un flux de vérification humaine (console)
- Créer un flux de vérification humaine (API)

Créer un flux de vérification humaine (console)

Vous pouvez soit compléter cet exemple à l'aide de votre propre document dans Amazon S3, soit télécharger[cet exemple de document](#)et placez-le dans votre compartiment Amazon S3.

Vérifiez que votre compartiment S3 est dans le même compartimentAWSRégion dans laquelle vous utilisez Amazon Textract. Pour créer un compartiment, consultezCréer un compartimentdans leManuel de l'utilisateur Amazon Simple Storage Service Console.

> **Note**
>
> La console Amazon A2I est intégrée à la console SageMaker. Pour utiliser la console, vous devez avoir des autorisations pour accéder à la console SageMaker et pour créer une équipe de travail. Pour commencer, vous pouvez utiliser leAmazonSageMakerFullAccessStratégie gérée IAM qui inclut toutes les autorisations nécessaires pour effectuer la plupart des actions dans SageMaker. Pour de plus amples informations, veuillez consulterIdentity and Access Management pour Amazon SageMakerdans leAmazon SageMaker Developer Guide.

Rubriques

Étape 1 : Créer une équipe de travail (console)

Tout d'abord, créez une équipe de travail dans la console Amazon A2I et ajoutez-vous en tant qu'employé afin de pouvoir prévisualiser la tâche de vérification humaine dans le portail des employés. Les membres de l'équipe de travail peuvent examiner différentes tâches et documents qui leur sont affectés.

Pour créer une main-d'œuvre privée à l'aide d'e-mails de travailleurs (console)

1. Ouvrez la console Amazon SageMaker à l'adresse https://console.aws.amazon.com/sagemaker/.
2. Dans le volet de navigation, sousGround Truth, choisissezMain-d'œuvre d'étiquetage.
3. Choisissez Private (Privée), puis Create private team (Créer une équipe privée).
4. Choisissez Invite new workers by email (Inviter les nouveaux employés par e-mail).
5. Pour cet exemple, saisissez votre adresse e-mail et celle de toutes les autres personnes dont vous souhaitez qu'elles puissent prévisualiser le portail d'employé. Collez ou tapez une liste de 50 adresses e-mail au maximum, séparées par des virgules, dans la pageAdresses e-mail.
6. Saisissez un nom d'organisation et une adresse e-mail de contact.
7. Choisissez Create private team (Créer une équipe privée).

Si vous vous ajoutez à une équipe de travail privée, vous recevez un e-mail de `no-reply@verificationemail.com` contenant des informations de connexion. Utilisez le lien figurant dans cet e-mail pour réinitialiser votre mot de passe et vous connecter au portail des employés. C'est là que vos tâches de vérification humaine apparaîtront après que vous appelez.`AnalyzeDocument`.

Étape 2 : Créer un flux de vérification humaine (console)

Dans cette étape, vous créez un flux de vérification humaine Amazon Textract.

Pour créer un flux de travail de vérification humaine (console)

1. Ouvrez la console Amazon A2I à l'adressehttps://console.aws.amazon.com/a2ipour accéder à la sectionWorkflows de révision humaine.
2. ChoisissezCréation d'un flux de vérification humaine.

3. PourNom, entrez un nom de flux de travail.
4. PourCompartiment S3, choisissez le compartiment dans lequel vous souhaitez qu'Amazon A2I stocke les résultats de vos tâches de vérification humaine. Si vous ne choisissez pas de compartiment, modifiez-le pour entrer le nom du compartiment.
5. UNDERRôle IAM, sélectionnezCréer un rôle. Une fenêtre apparaît avec le titreCréer un rôle IAM. Cette fenêtre permet de spécifier les compartiments Amazon S3 auxquels vous souhaitez que ce rôle ait accès. Si vous ne sélectionnez pasCompartiment S3, spécifiez le compartiment de sortie que vous avez spécifié à l'étape 4 et le compartiment qui contient votre document d'entrée.
6. PourType de tâche, choisissezTextract - Extraction de paire clé-valeur.
7. DansExtraction de formulaire Amazon Textract - Conditions d'appel d'un avis humain, spécifiez les conditions d'activation. Nous vous recommandons de définir un seuil de confiance élevé pour au moins une clé de votre document afin de déclencher une révision humaine afin de pouvoir prévisualiser une tâche de travail dans le portail de travail.

   Si vous avez utilisé l'exemple de document fourni dans cette procédure pas à pas, spécifiez les conditions d'activation comme suit :

   a. ChoisissezDéclenchement d'une vérification humaine pour des clés de formulaire spécifiques en fonction de l'indice de confiance de la clé de formulaire ou lorsque des clés de formulaire spécifiques sont manquantes.
   b. Pour Key name (Nom de la clé), saisissez **`Mail Address`**.
   c. Définir la propriétéIdentification de confianceseuil entre0et99.
   d. Définir la propriétéqualification de confianceseuil entre0et99.
   e. ChoisissezDéclenchement d'une vérification humaine pour toutes les clés de formulaire identifiées par Amazon Textract avec des indices de confiance situés dans une plage spécifique.
   f. Pour**`identification confidence`**, choisissez un nombre entre 0 et 90.
   g. Pour**`qualification confidence`**, choisissez un nombre entre 0 et 90.

   Cela déclenche une vérification humaine si Amazon Textract renvoie un indice de confiance inférieur à 99 pourAdresse postaleet sa valeur, ou s'il renvoie un score de confiance inférieur à 90 pour toute paire clé-valeur détectée dans le document.
8. Pour Worker task template creation (Création d'un modèle de tâche d'employé), sélectionnez Create from a default template (Créer à partir d'un modèle par défaut).

9. PourNom du modèle, entrez un nom descriptif.
10. PourDescription de la tâche, ajoutez quelque chose de similaire à ce qui suit :

    ```
    Read the instructions and review the document.
    ```

11. PourTravailleurschoisirPrivé.
12. Dans le menu, choisissez l'équipe privée que vous avez créée.
13. Sélectionnez Create (Créer).

Une fois que votre flux de vérification humaine est créé, il apparaît dans le tableau de la pageWorkflows de révision humaine. Lorsque leÉtatestActif, copiez et enregistrez l'ARN du flux de travail.

Créer un flux de vérification humaine (API)

Vous pouvez créer un flux de travail de vérification humaine ou undéfinition de flux, en utilisant Amazon A2I,`CreateFlowDefinition`.

Pour cet exemple, vous pouvez soit utiliser votre propre document dans Amazon S3, soit téléchargercet exemple de documentet rangez-le dans votre compartiment S3.

Vérifiez que votre compartiment Amazon S3 est dans le même compartimentAWSRégion que vous prévoyez d'utiliser pour appelerAnalyzeDocument. Pour créer un compartiment, suivez les instructions de la section Create a Bucket (Créer un compartiment) dans le Guide de l'utilisateur de la console Amazon Simple Storage Service.

Prérequis

Pour utiliser l'API Amazon A2I pour créer un flux de vérification humaine, vous devez remplir les conditions préalables suivantes :

- Configurez un rôle IAM avec l'autorisation d'appeler à la fois les opérations Amazon A2I et Amazon Textract API. Pour commencer, vous pouvez associer les stratégies AWS, AmazonAugmentedAIFullAccess et AmazonTextractFullAccess à un rôle IAM. Enregistrez le rôle Amazon Resources Name (ARN), car vous en aurez besoin ultérieurement.

  Pour obtenir des autorisations plus granulaires lors de l'utilisation d'Amazon Textract, voirExemples de stratégies basées sur l'identité Amazon Textract. Pour Amazon A2I, consultezAutorisations et sécurité dans Amazon Augmented AIdans leAmazon SageMaker Developer Guide.

- Créez une équipe de travail privée et enregistrez l'ARN de l'équipe de travail. Si vous êtes un nouvel utilisateur d'Amazon A2I, suivez les instructions deÉtape 1 : Créer une équipe de travail (console).
- Créez un modèle de tâche de collaborateur. Suivez les instructions de la sectionCréer un modèle de tâche d'employépour créer un modèle à l'aide de la console Amazon A2I. Lorsque vous créez le modèle, choisissezExtraction sous forme de textepourType de modèle. Dans le modèle, remplacez`s3_arn`avec l'ARN Amazon S3 de votre document. Ajouter des instructions supplémentaires pour le travailleur dans`<full-instructions header="Instructions"></full-instructions>`.

  Si vous souhaitez prévisualiser votre modèle, assurez-vous que votre rôle IAM dispose des autorisations décrites à la sectionActivation des aperçus du modèle de tâche de travail.

  Une fois que vous avez créé votre modèle, enregistrez l'ARN du modèle de tâche de travail.

Vous utilisez les ressources que vous avez créées dansPrérequispour configurer votre`CreateFlowDefinition`onde la demande. Dans cette demande, vous spécifiez également des conditions d'activation au format JSON. Pour savoir comment configurer vos conditions d'activation, consultezUtilisation du schéma JSON pour les conditions d'activation de boucle humaine avec Amazon Textract.

Création d'un flux de vérification humaine (kit AWS SDK for Python (Boto3))

Pour utiliser cet exemple, remplacez la*rouge*du texte avec vos spécifications et ressources.

Tout d'abord, encodez vos conditions d'activation dans un objet JSON à l'aide du code suivant. Cela déclenche une vérification humaine si Amazon Textract renvoie un indice de confiance inférieur à 99 pourAdresse postaleet sa valeur, ou s'il renvoie un score de confiance inférieur à 90 pour toute paire clé-valeur détectée dans le document. Si vous utilisez l'exemple de document fourni dans cet exemple, ces conditions d'activation créent une tâche de révision humaine.

```
import json

humanLoopActivationConditions = json.dumps("{
                "Conditions": [
                    {
                        "ConditionType": "ImportantFormKeyConfidenceCheck",
                        "ConditionParameters": {
                            "ImportantFormKey": "Mail Address",
```

```
                        "KeyValueBlockConfidenceLessThan": 99,
                        "WordBlockConfidenceLessThan": 99
                    }
                },
                {
                    "ConditionType": "ImportantFormKeyConfidenceCheck",
                    "ConditionParameters": {
                        "ImportantFormKey": "*",
                        "KeyValueBlockConfidenceLessThan": 90,
                        "WordBlockConfidenceLessThan": 90
                    }
                }
            ]
        }"
)
```

Utiliser`humanLoopActivationConditions`pour configurer le`create_flow_definition`de la demande. L'exemple suivant utilise le kit SDK for Python (Boto3) pour appeler[create_flow_definition](https://docs.aws.amazon.com/sagemaker/latest/dg/API_CreateFlowDefinition.html)dans la région AWS us-west-2. Il spécifie l'utilisation d'une équipe de travail privée.

```
response = client.create_flow_definition(
    FlowDefinitionName='string',
    HumanLoopRequestSource={
         'AwsManagedHumanLoopRequestSource': "AWS/Textract/AnalyzeDocument/Forms/V1"
    },
    HumanLoopActivationConfig={
        'HumanLoopActivationConditionsConfig': {
            'HumanLoopActivationConditions': humanLoopActivationConditions
        }
    },
    HumanLoopConfig={
        'WorkteamArn': "arn:aws:sagemaker:us-west-2:111122223333:workteam/private-
crowd/work-team-name",
        'HumanTaskUiArn': "arn:aws:sagemaker:us-west-2:111122223333:human-task-
ui/worker-task-template-name",
        'TaskTitle': "Add a task title",
        'TaskDescription': "Describe your task",
        'TaskCount': 1,
        'TaskAvailabilityLifetimeInSeconds': 3600,
        'TaskTimeLimitInSeconds': 86400,
        'TaskKeywords': ["Document Review", "Content Review"]
        }
```

```
    },
    OutputConfig={
        'S3OutputPath': "s3://DOC-EXAMPLE-BUCKET/prefix/",
    },
    RoleArn="arn:aws:iam::111122223333:role/role-name"
)
```

Analyse du document

Pour intégrer Amazon A2I dans un flux de travail d'analyse de documents Amazon Textract, vous devez configurerHumanLoopConfigdans leAnalyzeDocument.

DansHumanLoopConfig, vous spécifiez votre ARN de travail de vérification humaine (définition de flux) dansFlowDefinitionArn, et donnez un nom à votre boucle humaine dansHumanLoopName.

Analyze the Document (AWS SDK for Python (Boto3))

L'exemple suivant utilise le kit SDK for Python (Boto3) pour appeleranalyze_documentdans us-west-2. Remplacez le*rouge, italique*texte avec vos ressources. Pour de plus amples informations, veuillez consulter analyze_document dans la référence d'API du kit AWS SDK for Python (Boto).

```
client.analyze_document(Document={'S3Object': {"Bucket": "DOC-EXAMPLE-BUCKET",
 "Name": "document-name.png"}},
        HumanLoopConfig={"FlowDefinitionArn":"arn:aws:sagemaker:us-
west-2:111122223333:flow-definition/flow-definition-name",
                         "HumanLoopName":"human-loop-name",
                         "DataAttributes":{"ContentClassifiers":
["FreeOfPersonallyIdentifiableInformation"|"FreeOfAdultContent",]}},
        FeatureTypes=["FORMS"])
```

Analyze the Document (AWS CLI)

L'exemple suivant utilise laAWSCLI pour appeleranalyze_document. Ces exemples sont compatibles avecAWSVersion 2 de la CLI. La première est la syntaxe abrégée, la seconde dans la syntaxe JSON. Pour de plus amples informations, veuillez consulter analyze-document dans la référence AWS CLIcommande.

```
aws textract analyze-document \
     --document '{"S3Object":{"Bucket":"bucket_name","Name":"file_name"}}' \
     --human-loop-config
 HumanLoopName="test",FlowDefinitionArn="arn:aws:sagemaker:eu-west-1:xyz:flow-
definition/
hl_name",DataAttributes='{ContentClassifiers=["FreeOfPersonallyIdentifiableInformation","Fre
     --feature-types '["FORMS"]'
```

```
aws textract analyze-document \
     --document '{"S3Object":{"Bucket":"bucket_name","Name":"file_name"}}' \
     --human-loop-config \
          '{"HumanLoopName":"test","FlowDefinitionArn":"arn:aws:sagemaker:eu-
west-1:xyz:flow-definition/hl_name","DataAttributes": {"ContentClassifiers":
["FreeOfPersonallyIdentifiableInformation","FreeOfAdultContent"]}}' \
     --feature-types '["FORMS"]'
```

> **Note**
>
> Évitez les espaces blancs dans votre paramètre —human-loop-config, car cela peut entraîner des problèmes de traitement pour votre code.

La réponse à cette demande contientSortie d'activation Human Loop, qui indique si une boucle humaine a été créée et si c'est le cas, pourquoi. Si une boucle humaine a été créée, cet objet contient également le`HumanLoopArn`.

Pour plus d'informations sur et sur des exemples d'utilisation de la`AnalyzeDocument`opération, voirAnalyse du texte du document avec Amazon Textract.

Surveillance de la boucle humaine

Vous pouvez afficher des détails sur votre boucle humaine et arrêter une boucle humaine active en cas d'erreur à l'aide de la console Amazon A2I et de l'API.

Afficher les détails de la boucle humaine

Vous pouvez voir l'état de votre boucle humaine dans la console Amazon A2I et à l'aide de la commandeAmazon A2I Runtime.

Pour trouver des détails sur votre boucle humaine (console)

1. Ouvrez la console Amazon A2I à l'adressehttps://console.aws.amazon.com/a2ipour accéder à la sectionWorkflows de révision humaine.
2. Choisissez le flux de vérification humaine que vous avez également utilisé pour configurer.`HumanLoopConfig`dans`AnalyzeDocument`.
3. DansBoucles humaines, choisissez la boucle humaine dont vous souhaitez afficher les détails.

Pour en savoir plus sur votre boucle humaine (API) :

Utiliser Amazon A2I`DescribeHumanLoop`. Spécifiez le nom de la boucle humaine que vous avez utilisé pour appeler`AnalyzeDocument`.

L'exemple suivant : le kit SDK pour Python (Boto3) appelle`describe_human_loop`.

```
response = client.describe_human_loop(HumanLoopName="human-loop-name")
```

Arrêtez une boucle humaine

Une fois qu'une boucle humaine a démarré, vous pouvez l'arrêter à l'aide de la console et de l'API Amazon A2I.

Pour arrêter votre boucle humaine (console)

1. Ouvrez la console Amazon A2I à l'adressehttps://console.aws.amazon.com/a2ipour accéder à la sectionWorkflows de révision humaine.
2. Choisissez le flux de travail de vérification humaine que vous avez utilisé pour configurer`HumanLoopConfig`dans le`AnalyzeDocument`.
3. DansBoucles humaines, choisissez la boucle humaine que vous souhaitez arrêter.
4. Choisissez Stop (Arrêter).

Pour arrêter votre boucle humaine (API)

Utiliser Amazon A2I`StopHumanLoop`. Spécifiez le nom de la boucle humaine que vous avez utilisée pour appeler`AnalyzeDocument`.

Exemple de kit SDK for Python (Boto3)`stop_human_loop`.

```
response = client.stop_human_loop(HumanLoopName="human-loop-name")
```

Afficher les données de sortie et les mesures de travail

Lorsqu'une tâche de vérification humaine est effectuée par un collaborateur, Amazon A2I stocke vos données de sortie dans le compartiment Amazon S3 que vous avez spécifié dans votre flux de vérification humaine.

Si vous utilisez une main-d'œuvre privée, vos données de sortie contiennent des métadonnées de travail que vous pouvez utiliser pour suivre l'activité de chaque travailleur.

Rechercher les données de sortie dans Amazon S3

Amazon A2I utilise le nom de votre flux de travail de révision humaine comme préfixe du nom du fichier qui stocke les données de sortie des boucles humaines créées à l'aide de ce flux de travail de révision humaine.

Le chemin d'accès à une sortie de boucle humaine utilise le schéma suivant dans lequel*YYYY*/*MM*/*DD*/*hh*/*mm*/*ss*représente la date de création de la boucle humaine avec l'année (YYYY), mois (MM), et jour (DD) et le temps de création avec heure (hh), minute (mm), et le second (ss).

```
s3://output-bucket-specified-in-flow-definition/flow-definition-
name/YYYY/MM/DD/hh/mm/ss/human-loop-name/output.json
```

Pour voir la sortie d'une boucle humaine, utilisez la console Amazon A2I.

Pour voir la sortie de la boucle humaine

1. Ouvrez la console Amazon A2I à l'adressehttps://console.aws.amazon.com/a2ipour accéder à la sectionWorkflows de révision humaine.
2. Choisissez le flux de travail de vérification humaine que vous utilisez pour configurerHumanLoopConfigdansAnalyzeDocument.
3. DansBoucles humaines, choisissez la boucle humaine dont vous souhaitez consulter la sortie.
4. UNDEREmplacement de sortie, choisissez le lien vers les données de sortie.

Suivi de l'activité des employés privés

Lorsque vous utilisez une main-d'œuvre privée pour des tâches de révision humaine, les données de sortie incluent les informations suivantes sur le travailleur qui a terminé la révision :

- La valeur `workerId`.
- Dans `workerMetadata`:
  - `identityProviderType`— Service utilisé pour gérer la main-d'œuvre privée.
  - `issuer`— Le groupe d'utilisateurs Amazon Cognito ou l'émetteur du fournisseur d'identité OIDC (IdP) associé à l'équipe de travail affectée à cette tâche de vérification humaine.
  - `sub`— Identifiant unique associé au collaborateur. Si vous avez créé une main-d'œuvre à l'aide d'Amazon Cognito, vous pouvez extraire des détails sur cet employé (par ex., son nom ou son nom d'utilisateur) à l'aide de cet ID à l'aide d'Amazon Cognito. Pour savoir comment procéder, veuillez consulter Managing and Searching for User Accounts (Gestion et recherche de comptes utilisateur) dans le Guide du développeur Amazon Cognito.

Voici un exemple de la sortie que vous pouvez voir si vous avez utilisé Amazon Cognito pour créer une main-d'œuvre privée.

```
"workerId": "a12b3cdefg4h5i67",
            "workerMetadata": {
                "identityData": {
                    "identityProviderType": "Cognito",
                    "issuer": "https://cognito-idp.aws-region.amazonaws.com/aws-
region_123456789",
                    "sub": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee"
```

Voici un exemple de la sortie que vous pouvez voir si vous avez utilisé votre propre IdP OIDC pour créer une main-d'œuvre privée :

```
"workerId": "a12b3cdefg4h5i67",
            "workerMetadata": {
                "identityData": {
                    "identityProviderType": "Oidc",
                    "issuer": "https://example-oidc-ipd.com/adfs",
                    "sub": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee"
```

Pour en savoir plus sur l'utilisation de main-d'œuvre privée, consultezUtilisation d'une main-d'œuvre privéedans leAmazon SageMaker Developer Guide.

Sécurité dans Amazon Textract

Chez AWS, la sécurité dans le cloud est notre priorité numéro 1. En tant que client AWS, vous bénéficiez d'un centre de données et d'une architecture réseau conçus pour répondre aux exigences des organisations les plus pointilleuses en termes de sécurité.

Consultez les rubriques suivantes pour apprendre à sécuriser vos ressources Amazon Textract.

Rubriques

Protection des données dans Amazon Textract

Amazon Textract est conforme à laAWS modèle de responsabilité partagée, qui inclut les réglementations et directives relatives à la protection des données.AWSest responsable de la protection de l'infrastructure mondiale qui exécute tous lesAWSServices .AWSgère le contrôle sur les données hébergées sur cette infrastructure, y compris les contrôles de configuration de la sécurité pour la gestion du contenu des clients et des données personnelles.AWSclients etAPNles partenaires, agissant en tant que contrôleurs de données ou processeurs de données, sont responsables de toutes les données personnelles qu'ils mettent dans leAWScloud.

À des fins de protection des données, nous vous recommandons de protéger les autorisations du compte AWS et de configurer les comptes d'utilisateur individuels avec Gestion des identités et des accès AWS (IAM). Cela permet de garantir que chaque utilisateur se voit attribuer uniquement les autorisations nécessaires pour les besoins de ses tâches. Nous vous recommandons également de sécuriser vos données comme indiqué ci-dessous :

- Utilisez l'authentification multi-facteur (MFA) avec chaque compte.
- Utilisez SSL/TLS pour communiquer avec les ressources AWS.
- Configurez une API et la journalisation des activités utilisateur avec AWS CloudTrail.
- Utilisez des solutions de chiffrement AWS, ainsi que tous les contrôles de sécurité par défaut au sein des services AWS.
- Utilisez des services de sécurité gérés avancés tels qu'Amazon Macie, qui contribuent à la découverte et à la sécurisation des données personnelles stockées dans Amazon S3.

Nous vous recommandons vivement de ne jamais placer d'informations identifiables sensibles, telles que les numéros de compte de vos clients, dans des champs de formulaire comme Nom. Cela s'applique également lorsque vous travaillez avec Amazon Textract ou d'autresAWSservices utilisant la console, l'API,AWS CLI, ouAWSKits SDK. Toutes les données que vous entrez dans Amazon Textract ou d'autres services peuvent être récupérées, afin d'être insérées dans des journaux de diagnostic. Lorsque vous fournissez une URL à un serveur externe, n'incluez pas les informations d'identification non chiffrées dans l'URL pour valider votre demande adressée au serveur.

Pour en savoir plus sur la protection des données, consultez le billet de blog Modèle de responsabilité partagée AWS et RGPD sur le Blog sur la sécurité d'AWS.

Chiffrement dans Amazon Textract

Le chiffrement des données fait référence à la protection des données en transit et au repos. Vous pouvez protéger vos données à l'aide des clés gérées Amazon S3 ouAWS KMS keyau repos, parallèlement à la sécurité standard de la couche de transport pendant le transport.

Chiffrement au repos

La principale méthode de chiffrement des données dans Amazon Textract est le chiffrement côté serveur. Les documents d'entrée transmis depuis des compartiments Amazon S3 sont chiffrés par Amazon S3 et déchiffrés lorsque vous y accédez. Tant que vous authentifiez votre demande et que vous avez des autorisations d'accès, il n'y a aucune différence dans la manière dont vous accédez aux objets chiffrés ou déchiffrés. Par exemple, si vous partagez vos objets en utilisant une URL pré-signée, cette URL fonctionne de la même manière pour les objets chiffrés et déchiffrés. En outre, lorsque vous répertoriez des objets dans votre compartiment,`ListL`'API renvoie la liste de tous les objets, qu'ils soient chiffrés ou non.

Amazon Textract utilise deux méthodes de chiffrement côté serveur qui s'excluent mutuellement.

Chiffrement côté serveur avec des clés gérées par Amazon S3 (SSE-S3)

Lorsque vous utilisez le chiffrement côté serveur avec des clés gérées par Amazon S3 (SSE-S3), chaque objet est chiffré à l'aide d'une clé unique. Comme protection supplémentaire, cette méthode chiffre la clé elle-même à l'aide d'une clé principale dont il effectue une rotation régulière. Le chiffrement côté serveur Amazon S3 utilise l'un des chiffrements par bloc les plus puissants qui existent, Advanced Encryption Standard 256 bits (AES-256), pour chiffrer vos données. Pour plus d'informations, consultez Protection des données à l'aide du chiffrement côté serveur avec les clés de chiffrement Amazon S3 (SSE-S3).

Chiffrement côté serveur avec des clés KMS stockées dans AWS Key Management Service (SSE-KMS)

Le chiffrement côté serveur avec des clés KMS stockées dans AWS Key Management Service (SSE-KMS) est similaire à SSE-S3, avec certains avantages supplémentaires mais aussi certains frais supplémentaires d'utilisation de ce service. Il existe des autorisations distinctes pour l'utilisation d'une clé KMS qui renforce la protection contre tout accès non autorisé à vos objets dans Amazon S3. SSE-KMS vous fournit également un suivi d'audit indiquant quand votre clé KMS a été utilisée et par qui. En outre, vous pouvez créer et gérer des clés KMS ou utiliserClés gérées par AWSqui sont propres à vous-même, à votre service et à votre région. Pour de plus amples informations, veuillez consulterProtection des données à l'aide d'un chiffrement côté serveuravec les clés KMS stockées dans AWS Key Management Service (SSE-KMS).

Chiffrement en transit

Pour les données en transit, Amazon Textract utilise le protocole TLS (Transport Layer Security) pour chiffrer les données envoyées entre le service et l'agent. En outre, Amazon Textract utilise des points de terminaison VPC pour envoyer des données entre les différents microservices utilisés lors du traitement d'un document par Amazon Textract.

Confidentialité du trafic inter-réseaux

Amazon Textract communique exclusivement via les points de terminaison HTTPS, qui sont pris en charge dans toutes les régions prises en charge par Amazon Textract

Identity and Access Management pour Amazon Textract

Gestion des identités et des accès AWS (IAM) est un AWS service qui aide un administrateur à contrôler en toute sécurité l'accès aux AWS ressources. Les administrateurs IAM contrôlent qui peut

êtreauthentifié(connecté) etautorisé(possédant les autorisations) pour utiliser les ressources Amazon Textract. IAM est un AWS service que vous pouvez utiliser sans frais supplémentaires.

Rubriques

Public ciblé

Comment utilisez-vousGestion des identités et des accès AWS(IAM) diffère selon la tâche que vous accomplissez dans Amazon Textract.

Utilisateur du service— Si vous utilisez le service Amazon Textract pour effectuer votre tâche, votre administrateur vous fournit les informations d'identification et les autorisations dont vous avez besoin. Vous pourrez avoir besoin d'autorisations supplémentaires si vous utilisez davantage de fonctionnalités Amazon Textract. Comprendre la gestion des accès peut vous aider à demander à votre administrateur les autorisations appropriées. Si vous ne pouvez pas accéder à une fonctionnalité dans Amazon Textract, consultezRésolution des problèmes liés à Identity and Access Amazon Textract.

administrateur de service— Si vous êtes le responsable des ressources Amazon Textract de votre entreprise, vous bénéficiez probablement d'un accès total à Amazon Textract. C'est à vous de déterminer les fonctionnalités et les ressources Amazon Textract auxquelles vos employés pourront accéder. Vous devrez ensuite soumettre les demandes à votre administrateur IAM pour modifier les autorisations des utilisateurs de votre service. Consultez les informations sur cette page pour comprendre les concepts de base d'IAM. Pour en savoir plus sur la façon dont votre entreprise peut utiliser IAM avec Amazon Textract, veuillez consulter.Fonctionnement d'Amazon Textract avec IAM.

Administrateur IAMSi vous êtes un administrateur IAM, vous souhaiterez peut-être obtenir des détails sur la façon dont vous pouvez écrire des stratégies pour gérer l'accès à Amazon Textract. Pour afficher des exemples de stratégies basées sur l'identité Amazon Textract que vous pouvez utiliser dans IAM, consultez.Exemples de stratégies basées sur l'identité Amazon Textract.

Authentification avec des identités

L'authentification correspond au processus par lequel vous vous connectez à AWS via vos informations d'identification. Pour de plus amples informations sur la connexion à l'aide de la AWS Management Console, veuillez consulter Connexion à la AWS Management Console en tant qu'utilisateur IAM ou utilisateur racine dans le Guide de l'utilisateur IAM.

Vous devez vous authentifier (être connecté à AWS) en tant qu'utilisateur racine du Compte AWS, utilisateur IAM ou en endossant un rôle IAM. Vous pouvez également utiliser l'authentification de connexion unique de votre entreprise ou vous connecter par le biais de Google ou de Facebook. Dans ces cas, votre administrateur aura précédemment configuré une fédération d'identités avec des rôles IAM. Lorsque vous accédez à AWS avec des informations d'identification d'une autre entreprise, vous assumez indirectement un rôle.

Pour vous connecter directement à la AWS Management Console, utilisez votre mot de passe avec votre adresse e-mail d'utilisateur racine ou votre nom d'utilisateur IAM. Vous pouvez accéder à AWS par programmation avec vos clés d'accès d'utilisateur IAM ou racine. AWS fournit un kit SDK et des outils de ligne de commande pour signer de manière chiffrée votre demande avec vos informations d'identification. Si vous n'utilisez pas les outils AWS, vous devez signer la requête vous-même. Pour ce faire, utilisez Signature Version 4, un protocole permettant d'authentifier les demandes d'API entrantes. Pour plus d'informations sur l'authentification des demandes, consultez Processus de signature de la version 4 dans les Références générales AWS.

Quelle que soit la méthode d'authentification que vous utilisez, vous devrez peut-être également fournir des informations de sécurité supplémentaires. Par exemple, AWS vous recommande d'utiliser l'authentification multi-facteur (MFA) pour améliorer la sécurité de votre compte. Pour en savoir plus, veuillez consulter Utilisation de l'Authentification multi-facteur (MFA) dans AWS du Guide de l'utilisateur IAM.

Utilisateur racine Compte AWS

Lorsque vous créez un Compte AWS, vous commencez avec une seule identité de connexion disposant d'un accès complet à tous les services et ressources AWS du compte. Cette identité est appelée Compte AWS utilisateur racine Vous pouvez y accéder en vous connectant à l'aide de l'adresse e-mail et du mot de passe que vous avez utilisés pour créer le compte. Il est vivement recommandé de ne pas utiliser l'utilisateur racine pour vos tâches quotidiennes, y compris pour les tâches administratives. Respectez plutôt la bonne pratique qui consiste à avoir recours à l'utilisateur racine uniquement pour créer le premier utilisateur IAM. Ensuite, mettez en sécurité les informations

d'identification de l'utilisateur racine et utilisez-les uniquement pour effectuer certaines tâches de gestion des comptes et des services.

Utilisateurs et groupes IAM

Un utilisateur IAM est une identité dans votre Compte AWS qui dispose d'autorisations spécifiques pour une seule personne ou application. Un utilisateur IAM peut disposer d'informations d'identification à long terme, comme un nom d'utilisateur et un mot de passe ou un ensemble de clés d'accès. Pour découvrir comment générer des clés d'accès, consultez Gestion des clés d'accès pour les utilisateurs IAM dans le guide de l'utilisateur IAM. Lorsque vous générez des clés d'accès pour un utilisateur IAM, veillez à afficher et enregistrer la paire de clés de manière sécurisée. Vous ne pourrez plus récupérer la clé d'accès secrète à l'avenir. Au lieu de cela, vous devrez générer une nouvelle paire de clés d'accès.

Un groupe IAM est une identité qui concerne un ensemble d'utilisateurs IAM. Vous ne pouvez pas vous connecter en tant que groupe. Vous pouvez utiliser les groupes pour spécifier des autorisations pour plusieurs utilisateurs à la fois. Les groupes permettent de gérer plus facilement les autorisations pour de grands ensembles d'utilisateurs. Par exemple, vous pouvez avoir un groupe nommé IAMAdmins et accorder à ce groupe les autorisations d'administrer des ressources IAM.

Les utilisateurs sont différents des rôles. Un utilisateur est associé de manière unique à une personne ou une application, alors qu'un rôle est conçu pour être endossé par tout utilisateur qui en a besoin. Les utilisateurs disposent d'informations d'identification permanentes, mais les rôles fournissent des informations d'identification temporaires. Pour en savoir plus, consultez Quand créer un utilisateur IAM (au lieu d'un rôle) dans le Guide de l'utilisateur IAM.

Rôles IAM

Un rôle IAM est une entité au sein de votre Compte AWS qui dispose d'autorisations spécifiques. Le concept ressemble à celui d'utilisateur IAM, mais le rôle IAM n'est pas associé à une personne en particulier. Vous pouvez temporairement endosser un rôle IAM dans la AWS Management Console en changeant de rôle. Vous pouvez obtenir un rôle en appelant une opération d'API AWS CLI ou AWS à l'aide d'une URL personnalisée. Pour en savoir plus sur les méthodes d'utilisation des rôles, consultez Utilisation des rôles IAM dans le guide de l'utilisateur IAM.

Les rôles IAM avec des informations d'identification temporaires sont utiles dans les cas suivants :

- Autorisations utilisateur IAM temporaires : un utilisateur IAM peut endosser un rôle IAM pour accepter différentes autorisations temporaires concernant une tâche spécifique.

- Accès par des utilisateurs fédérés – Au lieu de créer un utilisateur IAM, vous pouvez utiliser des identités existantes provenant d'Directory Service, de votre répertoire d'utilisateurs d'entreprise ou d'un fournisseur d'identité web. On parle alors d'utilisateurs fédérés. AWS attribue un rôle à un utilisateur fédéré lorsque l'accès est demandé via un fournisseur d'identité. Pour en savoir plus sur les utilisateurs fédérés, consultez Utilisateurs fédérés et rôles dans le guide de l'utilisateur IAM.
- Accès comptes multiples : vous pouvez utiliser un rôle IAM pour permettre à un utilisateur (principal de confiance) d'un compte différent d'accéder aux ressources de votre compte. Les rôles constituent le principal moyen d'accorder l'accès entre plusieurs comptes. Toutefois, certains services AWS vous permettent d'attacher une stratégie directement à une ressource (au lieu d'utiliser un rôle en tant que proxy). Pour en savoir plus sur la différence entre les rôles et les politiques basées sur les ressources pour l'accès comptes multiples, veuillez consulter Différence entre les rôles IAM et les politiques basées sur les ressources dans le Guide de l'utilisateur IAM.
- Accès inter-services : certains services AWS utilisent des fonctionnalités dans d'autres services AWS. Par exemple, lorsque vous effectuez un appel dans un service, il est courant pour ce service d'exécuter des applications dans Amazon EC2 ou de stocker des objets dans Amazon S3. Un service peut le faire en utilisant les autorisations d'appel du principal, un rôle de service ou un rôle lié au service.
  - Autorisations principales – Lorsque vous utilisez un utilisateur ou un rôle IAM afin d'effectuer des actions dans AWS, vous êtes considéré comme principal. Les politiques accordent des autorisations au principal. Lorsque vous utilisez certains services, vous pouvez effectuer une action qui déclenche une autre action dans un autre service. Dans ce cas, vous devez disposer d'autorisations nécessaires pour effectuer les deux actions. Pour savoir si une action nécessite d'autres actions supplémentaires dans une stratégie, veuillez consulter.Actions, ressources et clés de condition pour Amazon Textractdans leRéférence de l'autorisation de service.
  - Rôle de service : il s'agit d'un rôle IAM attribué à un service afin de réaliser des actions en votre nom. Un administrateur IAM peut créer, modifier et supprimer une fonction du service à partir d'IAM. Pour plus d'informations, consultez Création d'un rôle pour la délégation d'autorisations à un service AWS dans le Guide de l'utilisateur IAM.
  - Rôle lié au service – Un rôle lié au service est un type de rôle de service lié à un service AWS. Le service peut assumer le rôle afin d'effectuer une action en votre nom. Les rôles liés à un service s'affichent dans votre compte IAM et sont la propriété du service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.
- Applications s'exécutant sur Amazon EC2 : vous pouvez utiliser un rôle IAM pour gérer des informations d'identification temporaires pour les applications s'exécutant sur une instance EC2 et effectuant des requêtes d'API AWS CLI ou AWS. Cette solution est préférable au stockage

des clés d'accès au sein de l'instance EC2. Pour attribuer un rôle AWS à une instance EC2 et le rendre disponible à toutes les applications associées, vous pouvez créer un profil d'instance attaché à l'instance. Un profil d'instance contient le rôle et permet aux programmes qui s'exécutent sur l'instance EC2 d'obtenir des informations d'identification temporaires. Pour de plus amples informations, veuillez consulter Utilisation d'un rôle IAM pour accorder des autorisations à des applications s'exécutant sur des instances Amazon EC2 dans le Guide de l'utilisateur IAM.

Pour savoir dans quel cas utiliser des rôles ou des utilisateurs IAM, veuillez consulter Quand créer un rôle IAM (au lieu d'un utilisateur) dans le Guide de l'utilisateur IAM.

Gestion des accès à l'aide de politiques

Vous contrôler les accès dans AWS en créant des stratégies et en les attachant à des identités ou à des ressources AWS. Une politique est un objet dans AWS qui, lorsqu'il est associé à une identité ou à une ressource, définit ses autorisations. Vous pouvez vous connecter en tant qu'utilisateur racine ou IAM ou vous pouvez endosser un rôle IAM. Lorsque vous effectuez ensuite une demande, AWS évalue les stratégies relatives basées sur l'identité ou les ressources. Les autorisations dans les politiques déterminent si la demande est autorisée ou refusée. La plupart des politiques sont stockées dans AWS en tant que documents JSON. Pour de plus amples informations sur la structure et le contenu des documents de politique JSON, veuillez consulter Présentation des politiques JSON dans le Guide de l'utilisateur IAM.

Les administrateurs peuvent utiliser les stratégies JSON AWS pour spécifier qui a accès à quoi. Cela signifie : quel principal peut effectuer des actions sur quel type de ressources et dans quelles conditions.

Chaque entité IAM (utilisateur ou rôle) démarre sans autorisation. En d'autres termes, par défaut, les utilisateurs ne peuvent rien faire, pas même changer leurs propres mots de passe. Pour autoriser un utilisateur à effectuer une opération, un administrateur doit associer une politique d'autorisations à ce dernier. Il peut également ajouter l'utilisateur à un groupe disposant des autorisations prévues. Lorsqu'un administrateur accorde des autorisations à un groupe, tous les utilisateurs de ce groupe se voient octroyer ces autorisations.

Les politiques IAM définissent les autorisations d'une action, quelle que soit la méthode que vous utilisez pour exécuter l'opération. Par exemple, supposons que vous disposiez d'une stratégie qui autorise l'action `iam:GetRole`. Un utilisateur avec cette stratégie peut obtenir des informations utilisateur à partir de l'AWS Management Console, de l'AWS CLI ou de l'API AWS.

Politiques basées sur l'identité

Les politiques basées sur l'identité sont des documents de politique d'autorisations JSON que vous pouvez attacher à une identité telle qu'un utilisateur, un groupe d'utilisateurs ou un rôle IAM. Ces politiques contrôlent quel type d'actions des utilisateurs et des rôles peuvent exécuter, sur quelles ressources et dans quelles conditions. Pour découvrir comment créer une politique basée sur l'identité, veuillez consulter Création de politiques IAM dans le Guide de l'utilisateur IAM.

Les politiques basées sur l'identité peuvent être classées comme étant des politiques en ligne ou des politiques gérées. Les politiques en ligne sont intégrées directement à un utilisateur, groupe ou rôle. Les politiques gérées sont des politiques autonomes que vous pouvez lier à plusieurs utilisateurs, groupes et rôles de votre compte.Compte AWS. Les stratégies gérées incluent les stratégies gérées par AWS et les stratégies gérées par le client. Pour découvrir comment choisir entre une politique gérée et une politique en ligne, veuillez consulter Choix entre les politiques gérées et les politiques en ligne dans le Guide de l'utilisateur IAM.

Politiques basées sur une ressource

Les politiques basées sur les ressources sont des documents de politique JSON que vous attachez à une ressource. Des politiques basées sur les ressources sont, par exemple, les politiques de confiance de rôle IAM et des politiques de compartiment Amazon S3. Dans les services qui sont compatibles avec les politiques basées sur les ressources, les administrateurs de service peuvent les utiliser pour contrôler l'accès à une ressource spécifique. Pour la ressource dans laquelle se trouve la politique, cette dernière définit quel type d'actions un principal spécifié peut effectuer sur cette ressource et dans quelles conditions. Vous devez spécifier un principal dans une politique basée sur les ressources. Les mandataires peuvent inclure des comptes, des utilisateurs, des rôles, des utilisateurs fédérés ou des services AWS.

Les politiques basées sur les ressources sont des politiques en ligne situées dans ce service. Vous ne pouvez pas utiliser les politiques gérées AWS depuis IAM dans une politique basée sur une ressource.

Listes de contrôle d'accès (ACL)

Les listes de contrôle d'accès (ACL) vérifie quels principaux (membres de compte, utilisateurs ou rôles) ont l'autorisation d'accéder à une ressource. Les listes de contrôle d'accès sont similaires aux politiques basées sur les ressources, bien qu'elles n'utilisent pas le format de document de politique JSON.

Amazon S3, AWS WAF et Amazon VPC sont des exemples de services prenant en charge les ACL. Pour en savoir plus sur les listes de contrôle d'accès, veuillez consulter Présentation des listes de contrôle d'accès (ACL) dans le Guide du développeur Amazon Simple Storage Service.

Autres types de politique

AWS prend en charge d'autres types de stratégies moins courantes. Ces types de politiques peuvent définir le nombre maximum d'autorisations qui vous sont accordées par des types de politiques plus courants.

- Limite d'autorisations – Une limite d'autorisations est une fonction avancée dans laquelle vous définissez le nombre maximal d'autorisations qu'une politique basée sur l'identité peut accorder à une entité IAM (utilisateur ou rôle IAM). Vous pouvez définir une limite d'autorisations pour une entité. Les autorisations obtenues représentent la combinaison des politiques basées sur l'identité de l'entité et de ses limites d'autorisations. Les stratégies basées sur les ressources qui spécifient l'utilisateur ou le rôle dans le champ `Principal` ne sont pas limitées par les limites d'autorisations. Un refus explicite dans l'une de ces politiques remplace l'autorisation. Pour de plus amples informations sur les limites d'autorisations, veuillez consulter Limites d'autorisations pour des entités IAM dans le Guide de l'utilisateur IAM.
- Politiques de contrôle des services (SCP) : les SCP sont des politiques JSON qui spécifient le nombre maximal d'autorisations pour une organisation ou une unité d'organisation (OU) dans AWS Organizations. AWS Organizations est un service qui vous permet de regrouper et de gérer de façon centralisée plusieurs Comptes AWS détenus par votre entreprise. Si vous activez toutes les fonctions d'une organisation, vous pouvez appliquer les politiques de contrôle de service (SCP) à l'un ou à l'ensemble de vos comptes. Les politiques de contrôle des services (SCP) limitent les autorisations pour les entités dans les comptes membres, y compris chaque utilisateur racine de compte Compte AWS. Pour plus d'informations sur les organisations et les SCP, veuillez consulter Fonctionnement des SCP dans le Guide de l'utilisateur AWS Organizations.
- Politiques de session – Les politiques de session sont des politiques avancées que vous passez en tant que paramètre lorsque vous programmez afin de créer une session temporaire pour un rôle ou un utilisateur fédéré. Les autorisations de la session obtenue sont une combinaison des politiques basées sur l'identité de l'utilisateur ou du rôle et des politiques de session. Les autorisations peuvent également provenir d'une politique basée sur les ressources. Un refus explicite dans l'une de ces politiques remplace l'autorisation. Pour de plus amples informations, consultez politiques de séance dans le Guide de l'utilisateur IAM.

Plusieurs types de politique

Lorsque plusieurs types de politiques s'appliquent à la requête, les autorisations obtenues sont plus compliquées à comprendre. Pour découvrir la façon dont AWS détermine s'il convient d'autoriser une demande en présence de plusieurs types de politiques, veuillez consulter Logique d'évaluation de politiques dans le Guide de l'utilisateur IAM.

Fonctionnement d'Amazon Textract avec IAM

Avant d'utiliser IAM pour gérer l'accès à Amazon Textract, vous devez comprendre quelles sont les fonctions IAM qui peuvent être utilisées dans cette Amazon Textract. Pour obtenir une vue globale de la façon dont Amazon Textract et autresAWSservices fonctionnent avec IAM, consultezAWSServices qui fonctionnent avec IAMdans leIAM User Guide.

Rubriques

- Stratégies basées sur l'identité Amazon Textract
- Stratégies basées sur des ressources Amazon Textract
- Autorisation basée sur les balises Amazon Textract
- Rôles IAM Textract

Stratégies basées sur l'identité Amazon Textract

Avec les stratégies IAM basées sur l’identité, vous pouvez spécifier des actions et ressources autorisées ou refusées et les conditions dans lesquelles les actions sont autorisées ou refusées. Amazon Textract est compatible avec des actions, des ressources et des clés de condition spécifiques. Pour en savoir plus sur tous les éléments que vous utilisez dans une politique JSON, veuillez consulter Références des éléments de politique JSON IAM dans le Guide de l'utilisateur IAM.

Actions

Les administrateurs peuvent utiliser les stratégies JSON AWS pour spécifier qui a accès à quoi. Cela signifie : quel principal peut effectuer des actions sur quel type de ressources et dans quelles conditions.

L'élément `Action` d'une stratégie JSON décrit les actions que vous pouvez utiliser pour autoriser ou refuser l'accès à une stratégie. Les actions de stratégie possèdent généralement le même nom que l'opération d'API AWS associée. Il existe quelques exceptions, telles que les actions avec

autorisations uniquement qui n'ont pas d'opération API correspondante. Certaines opérations nécessitent également plusieurs actions dans une politique. Ces actions supplémentaires sont nommées actions dépendantes.

Intégration d'actions dans une politique afin d'accorder l'autorisation d'exécuter les opérations associées.

Les actions asynchrones dans Amazon Textract nécessitent deux autorisations d'action, l'une pour les actions Démarrer et l'autre pour les actions Obtenir. De plus, si vous utilisez un compartiment Amazon S3 pour transmettre des documents, vous devrez accorder l'accès en lecture à votre compte.

Dans Amazon Textract, toutes les actions de stratégie commencent par : `textract:`. Par exemple, pour accorder à une personne l'autorisation d'exécuter une opération Amazon Textract avec Amazon Textract.`AnalyzeDocument`, vous incluez le`textract:AnalyzeDocument`action dans leur politique. Les déclarations de politique doivent inclure un élément `Action` ou `NotAction`. Amazon Textract définit son propre ensemble d'actions qui décrivent les tâches que vous pouvez effectuer avec ce service.

Pour spécifier plusieurs actions dans une seule déclaration, séparez-les par des virgules comme suit.

```
"Action": [
      "textract:action1",
      "textract:action2"
```

Vous pouvez aussi préciser plusieurs actions à l’aide de caractères génériques (*). Par exemple, pour préciser toutes les actions qui commencent par le mot `Describe`, incluez l’action suivante.

```
"Action": "textract:Describe*"
```

Pour obtenir la liste des actions Amazon Textract, consultezActions définies par Amazon Textractdans leIAM User Guide.

Ressources

Les administrateurs peuvent utiliser les stratégies JSON AWS pour spécifier qui a accès à quoi. Cela indique quel principal peut exécuter des actions, sur quel type de ressources et dans quelles conditions.

L'élément de stratégie JSON `Resource` indique le ou les objets pour lesquels l'action s'applique. Les instructions doivent inclure un élément `Resource` ou `NotResource`. Il est recommandé de définir une ressource à l'aide de son Amazon Resource Name (ARN). Vous pouvez le faire pour des actions qui prennent en charge un type de ressource spécifique, connu sous la dénomination autorisations de niveau ressource.

Pour les actions qui ne sont pas compatibles avec les autorisations de niveau ressource, telles que les opérations de liste, utilisez un caractère générique (*) afin d'indiquer que l'instruction s'applique à toutes les ressources.

```
"Resource": "*"
```

Amazon Textract ne prend pas en charge la spécification des ARN de ressource dans une politique.

Clés de condition

Les administrateurs peuvent utiliser les stratégies JSON AWS pour spécifier qui a accès à quoi. Cela indique quel principal peut exécuter des actions, sur quel type de ressources et dans quelles conditions.

L'élément `Condition` (ou le bloc `Condition`) vous permet de spécifier des conditions lorsqu'une instruction est appliquée. L'élément `Condition` est facultatif. Vous pouvez créer des expressions conditionnelles qui utilisent des opérateurs de condition, tels que les signes égal ou inférieur à, pour faire correspondre la condition de la politique aux valeurs de la demande.

Si vous spécifiez plusieurs éléments `Condition` dans une instruction, ou plusieurs clés dans un seul élément `Condition`, AWS les évalue à l'aide d'une opération `AND` logique. Si vous spécifiez plusieurs valeurs pour une seule clé de condition, AWS évalue la condition à l'aide d'une opération `OR` logique. Toutes les conditions doivent être remplies avant que les autorisations associées à l'instruction ne soient accordées.

Vous pouvez aussi utiliser des variables d'espace réservé quand vous spécifiez des conditions. Par exemple, vous pouvez accorder à un utilisateur IAM l'autorisation d'accéder à une ressource uniquement si elle est balisée avec son nom d'utilisateur IAM. Pour de plus amples d'informations, veuillez consulter Éléments d'une politique IAM : variables et balises dans le Guide de l'utilisateur IAM.

AWS prend en charge les clés de condition globales et les clés de condition spécifiques à un service. Pour afficher toutes les clés de condition globales AWS, veuillez consulter la rubrique Clés de contexte de condition globale AWS dans le Guide de l'utilisateur IAM.

Amazon Textract ne fournit pas de clés de condition spécifiques au service, mais prend en charge l'utilisation de certaines clés de condition globales. Pour obtenir une liste de tousAWSclés de condition globales, consultezAWSClés de contexte de condition globaledans leIAM User Guide.

Exemples

Pour voir des exemples de stratégies Amazon Textract basées sur l'identité, consultezExemples de stratégies basées sur l'identité Amazon Textract.

Stratégies basées sur des ressources Amazon Textract

Amazon Textract ne prend pas en charge les stratégies basées sur les ressources.

Autorisation basée sur les balises Amazon Textract

Amazon Textract ne prend pas en charge le balisage des ressources ni le contrôle d'accès basé sur des balises.

Rôles IAM Textract

Un rôle IAM est une entité au sein de votre compte AWS qui dispose d'autorisations spécifiques.

Utilisation d'informations d'identification temporaires avec Amazon Textract

Vous pouvez utiliser des informations d'identification temporaires pour vous connecter à l'aide de la fédération, endosser un rôle IAM ou encore pour endosser un rôle entre comptes. Vous obtenez des informations d'identification de sécurité temporaires en appelant des opérations d'API AWS STS comme AssumeRole ou GetFederationToken.

Amazon Textract est compatible avec l'utilisation des informations d'identification temporaires

Rôles liés à un service

Les rôles liés à un service permettent aux services AWS d'accéder à des ressources dans d'autres services pour effectuer une action en votre nom. Les rôles liés à un service s'affichent dans votre compte IAM et sont la propriété du service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.

Amazon Textract ne prend pas en charge les rôles liés à un service.

> **Note**
>
> Amazon Textract ne prenant pas en charge les rôles liés à un service, il ne prend pas en charge les mandataires de service AWS. Pour de plus amples informations sur les principaux de service, consultezPrincipaux de service AWSdans leIAM User Guide

Rôles de service

Cette fonction permet à un service d'endosser une fonction du service en votre nom. Ce rôle autorise le service à accéder à des ressources d'autres services pour effectuer une action en votre nom. Les rôles de service s'affichent dans votre compte IAM et sont la propriété du compte. Cela signifie qu'un administrateur IAM peut modifier les autorisations associées à ce rôle. Toutefois, une telle action peut perturber le bon fonctionnement du service.

Amazon Textract prend en charge les rôles de service.

Exemples de stratégies basées sur l'identité Amazon Textract

Par défaut, les utilisateurs et les rôles IAM ne sont pas autorisés à créer ou à modifier les ressources Amazon Textract. Ils ne peuvent pas non plus exécuter des tâches à l'aide de AWS Management Console, AWS CLI ou de l'API AWS. Un administrateur IAM doit créer des politiques IAM autorisant les utilisateurs et les rôles à exécuter des opérations d'API spécifiques sur les ressources spécifiées dont ils ont besoin. Il doit ensuite attacher ces politiques aux utilisateurs ou aux groupes IAM ayant besoin de ces autorisations.

Pour savoir comment créer une stratégie IAM basée sur l'identité à l'aide de ces exemples de documents de stratégie JSON, veuillez consulter Création de stratégies dans l'onglet JSON dans le Guide de l'utilisateur IAM.

Rubriques

- Bonnes pratiques en matière de politiques
- Autoriser les utilisateurs à afficher leurs propres autorisations
- Accès aux opérations synchrone dans Amazon Textract
- Accès aux opérations asynchrones dans Amazon Textract

Bonnes pratiques en matière de politiques

Les politiques basées sur l'identité sont très puissantes. Elles déterminent si une personne peut créer, consulter ou supprimer des ressources Amazon Textract dans votre compte. Ces actions peuvent entraîner des frais pour votre Compte AWS. Lorsque vous créez ou modifiez des politiques basées sur l'identité, suivez ces instructions et recommandations :

- Commencez à utiliserAWSpolitiques gérées— Pour commencer à utiliser Amazon Textract rapidement, utilisezAWSgérées pour accorder à vos employés les autorisations dont ils ont besoin. Ces politiques sont déjà disponibles dans votre compte et sont gérées et mises à jour par AWS. Pour plus d'informations, veuillez consulter Démarrer avec les autorisations à l'aide des politiques gérées par AWS dans le Guide de l'utilisateur IAM.
- Accorder le moindre privilège – Lorsque vous créez des politiques personnalisées, accordez uniquement les autorisations nécessaires à l'exécution d'une tâche. Commencez avec un minimum d'autorisations et accordez-en d'autres si nécessaire. Cette méthode est plus sûre que de commencer avec des autorisations trop permissives et d'essayer de les restreindre plus tard. Pour en savoir plus, consultez Accorder le moindre privilège possible dans le Guide de l'utilisateur IAM.
- Activer la MFA pour les opérations confidentielles : pour plus de sécurité, demandez aux utilisateurs IAM d'utiliser l'Authentification multi-facteur (MFA) pour accéder à des ressources ou à des opérations d'API confidentielles. Pour en savoir plus, consultez Utilisation de l'authentification multi-facteur (MFA) dans AWS dans le Guide de l'utilisateur IAM.
- Utiliser des conditions de politique pour davantage de sécurité – Dans la mesure du possible, définissez les conditions dans lesquelles vos politiques basées sur l'identité autorisent l'accès à une ressource. Par exemple, vous pouvez rédiger les conditions pour spécifier une plage d'adresses IP autorisées d'où peut provenir une demande. Vous pouvez également écrire des conditions pour autoriser les requêtes uniquement à une date ou dans une plage de temps spécifiée, ou pour imposer l'utilisation de SSL ou de MFA. Pour de plus amples informations, veuillez consulterÉléments de politique JSON IAM : Conditiondans leIAM User Guide.

Autoriser les utilisateurs à afficher leurs propres autorisations

Cet exemple montre comment créer une politique qui permet aux utilisateurs IAM d'afficher les politiques en ligne et gérées attachées à leur identité d'utilisateur. Cette politique inclut les autorisations nécessaires pour réaliser cette action sur la console ou par programmation à l'aide du AWS CLI ou de AWS l'API.

```
{
```

```
    "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": "*"
        }
    ]
}
```

Accès aux opérations synchrone dans Amazon Textract

Cet exemple de stratégie accorde l'accès aux actions synchrone dans Amazon Textract à un utilisateur IAM sur votreAWS.

```
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "textract:DetectDocumentText",
```

```
                "textract:AnalyzeDocument"
            ],
            "Resource": "*"
        }
    ]
```

Accès aux opérations asynchrones dans Amazon Textract

L'exemple de stratégie suivant indique à un utilisateur IAM sur votreAWSaccès au compte à toutes les opérations asynchrones utilisées dans Amazon Textract.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "textract:StartDocumentTextDetection",
                "textract:StartDocumentAnalysis",
                "textract:GetDocumentTextDetection",
                "textract:GetDocumentAnalysis"
            ],
            "Resource": "*"
        }
    ]
}
```

Résolution des problèmes liés à Identity and Access Amazon Textract

Utilisez les informations suivantes pour identifier et résoudre les problèmes courants que vous pouvez rencontrer lorsque vous travaillez avec Amazon Textract et IAM.

Rubriques

- Je ne suis pas autorisé à exécuter une action dans Amazon Textract
- Je ne suis pas autorisé à exécuter iam:PassRole
- Je veux afficher mes clés d'accès
- Je suis administrateur et je veux autoriser d'autres utilisateurs à accéder à Amazon Textract
- Je veux autoriser des personnes extérieures à monAWSCompte pour accéder aux ressources My Amazon Textract

Je ne suis pas autorisé à exécuter une action dans Amazon Textract

Si AWS Management Console indique que vous n'êtes pas autorisé à exécuter une action, vous devez contacter votre administrateur pour obtenir de l'aide. Votre administrateur est la personne qui vous a fourni votre nom d'utilisateur et votre mot de passe.

L'exemple d'erreur suivant se produit lorsque le`mateojackson`L'utilisateur IAM tente d'exécuter`DetectDocumentText`sur une image de test mais n'a pas`textract:DetectDocumentText`Autorisations.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 textract:DetectDocumentText on resource: textimage.png
```

Dans ce cas, Mateo demande à son administrateur de mettre à jour ses politiques pour lui permettre d'accéder à la ressource `textimage.png` à l'aide de l'action `textract:DetectDocumentText`.

Je ne suis pas autorisé à exécuter iam:PassRole

Si vous recevez un message d'erreur selon lequel vous n'êtes pas autorisé à exécuter l'action `iam:PassRole`, vous devez contacter votre administrateur pour obtenir de l'aide. Votre administrateur est la personne qui vous a fourni votre nom d'utilisateur et votre mot de passe. Demandez à cette personne de mettre à jour vos stratégies afin de vous permettre de transmettre un rôle à Amazon Textract.

Certains services AWS vous permettent de transmettre un rôle existant à ce service, au lieu de créer un nouveau rôle de service ou rôle lié à un service. Pour ce faire, un utilisateur doit disposer des autorisations nécessaires pour transmettre le rôle au service.

L'exemple d'erreur suivant se produit lorsqu'un utilisateur IAM nommé`marymajor`essaie d'utiliser la console pour exécuter une action dans Amazon Textract. Toutefois, l'action nécessite que le service ait des autorisations accordées par une fonction du service. Mary ne dispose pas des autorisations nécessaires pour transférer le rôle au service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

Dans ce cas, Mary demande à son administrateur de mettre à jour ses stratégies pour lui permettre d'exécuter l'action `iam:PassRole`.

Je veux afficher mes clés d'accès

Une fois les clés d'accès utilisateur IAM créées, vous pouvez afficher votre ID de clé d'accès à tout moment. Toutefois, vous ne pouvez pas revoir votre clé d'accès secrète. Si vous perdez votre clé d'accès secrète, vous devez créer une nouvelle paire de clés.

Les clés d'accès se composent de deux parties : un ID de clé d'accès (par exemple, `AKIAIOSFODNN7EXAMPLE`) et une clé d'accès secrète (par exemple, `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`). À l'instar d'un nom d'utilisateur et un mot de passe, vous devez utiliser à la fois l'ID de clé d'accès et la clé d'accès secrète pour authentifier vos demandes. Gérez vos clés d'accès de manière aussi sécurisée que votre nom d'utilisateur et votre mot de passe.

> **Important**
>
> Ne communiquez pas vos clés d'accès à un tiers, même pour qu'il vous aide à trouver votre ID utilisateur canonique. En effet, vous lui accorderiez ainsi un accès permanent à votre compte.

Lorsque vous créez une paire de clé d'accès, enregistrez l'ID de clé d'accès et la clé d'accès secrète dans un emplacement sécurisé. La clé d'accès secrète est accessible uniquement au moment de sa création. Si vous perdez votre clé d'accès secrète, vous devez ajouter de nouvelles clés d'accès pour votre utilisateur IAM. Vous pouvez avoir un maximum de deux clés d'accès. Si vous en avez déjà deux, vous devez supprimer une paire de clés avant d'en créer une nouvelle. Pour afficher les instructions, veuillez consulter Gestion des clés d'accès dans le Guide de l'utilisateur IAM.

Je suis administrateur et je veux autoriser d'autres utilisateurs à accéder à Amazon Textract

Pour permettre à d'autres utilisateurs d'accéder à Amazon Textract, vous devez créer une entité IAM (utilisateur ou rôle) pour la personne ou l'application nécessitant un accès. Ils utiliseront les informations d'identification de cette entité pour accéder à AWS. Vous devez ensuite associer une stratégie à l'entité qui va lui accorder les autorisations nécessaires dans Amazon Textract.

Pour démarrer immédiatement, veuillez consulter Création de votre premier groupe et utilisateur délégué IAM dans le Guide de l'utilisateur IAM.

Je veux autoriser des personnes extérieures à monAWSCompte pour accéder aux ressources My Amazon Textract

Vous pouvez créer un rôle que les utilisateurs provenant d'autres comptes ou les personnes extérieures à votre organisation pourront utiliser pour accéder à vos ressources. Vous pouvez spécifier la personne à qui vous souhaitez confier le rôle. Pour les services qui prennent en charge les politiques basées sur les ressources ou les listes de contrôle d'accès (ACL), vous pouvez utiliser ces politiques pour donner l'accès à vos ressources.

Pour en savoir plus, consultez les éléments suivants :

- Pour savoir si Amazon Textract est compatible avec ces fonctionnalités, consultezFonctionnement d'Amazon Textract avec IAM.
- Pour savoir comment octroyer l'accès à vos ressources à des Comptes AWS dont vous êtes propriétaire, veuillez consulter la section Fournir l'accès à un utilisateur IAM dans un autre Compte AWS que vous possédez dans le Guide de l'utilisateur IAM.
- Pour savoir comment octroyer l'accès à vos ressources à des Comptes AWS tiers, veuillez consulter Fournir l'accès aux Comptes AWS appartenant à des tiers dans le Guide de l'utilisateur IAM.
- Pour savoir comment fournir un accès par le biais de la fédération d'identité, veuillez consulter Fournir un accès à des utilisateurs authentifiés en externe (fédération d'identité) dans le Guide de l'utilisateur IAM.
- Pour découvrir quelle est la différence entre l'utilisation des rôles et l'utilisation des politiques basées sur les ressources pour l'accès entre comptes, veuillez consulter Différence entre les rôles IAM et les politiques basées sur les ressources dans le Guide de l'utilisateur IAM.

Journalisation et surveillance

Pour surveiller Amazon Textract, utilisez Amazon CloudWatch. Cette section fournit des informations sur la façon de configurer la surveillance pour Amazon Textract. Il fournit également du contenu de référence pour les mesures Amazon Textract.

Rubriques

- Surveillance de Amazon Textract
- Métriques CloudWatch pour Amazon Textract.

Surveillance de Amazon Textract

Avec CloudWatch, vous pouvez obtenir des métriques relatives à des opérations individuelles Amazon Textract ou globales pour votre compte. Vous pouvez utiliser les métriques pour suivre l'état de votre solution basée sur Amazon Texact, et configurer des alarmes pour vous notifier lorsqu'une ou plusieurs métriques dépassent un seuil défini. Par exemple, vous pouvez afficher des métriques pour connaître le nombre d'erreurs serveur qui se sont produites. Vous pouvez également consulter les métriques pour obtenir le nombre de fois où une opération Amazon Textract spécifique a réussi. Pour afficher les métriques, vous pouvez utiliserAmazon CloudWatch, leAWS CLI, ou leAPI CloudWatch.

Utiliser CloudWatch pour Amazon Textract

Pour utiliser les métriques, vous devez spécifier les informations suivantes :

- La dimension de métrique ou l'absence de dimension. Une dimension est une paire nom-valeur qui vous aide à identifier une métrique de façon unique. Amazon Textract possède une dimension, nomméeOpération. Elle fournit les métriques d'une opération spécifique. Si vous ne spécifiez pas de dimension, la portée de la métrique englobe toutes les opérations Amazon Textract au sein de votre compte.
- Le nom de la métrique, par exemple `UserErrorCount`.

Vous pouvez obtenir des données de surveillance pour Amazon Textract à l'aide duAWS Management Console, leAWS CLI, ou l'API CloudWatch. Vous pouvez également utiliser l'API CloudWatch via l'un des kits (SDK) Amazon AWS ou via les outils de l'API CloudWatch. La console affiche un ensemble de graphiques basés sur les données brutes obtenues avec l'API CloudWatch. En fonction de vos besoins, vous pouvez utiliser les graphiques affichés dans la console ou extraits de l'API.

La liste suivante présente certaines utilisations courantes des métriques. Voici quelques suggestions pour vous aider à démarrer, qui ne forment pas une liste exhaustive.

Comment... ?	Métriques pertinentes
Comment savoir si mon application a atteint le nombre maximal de demandes par seconde ?	Surveillez la statistique `Sum` de la métrique `ThrottledCount` .

Comment... ?	Métriques pertinentes
Comment surveiller les erreurs de demande ?	Utilisez la statistique `Sum` de la métrique `UserErrorCount` .
Comment obtenir le nombre total de demandes ?	Utilisez la statistique `SampleCount` de la métrique `ResponseTime` . Celle-ci inclut les demandes qui se sont traduites par une erreur. Si vous voulez ne voir que les appels d'opération ayant réussi, utilisez la métrique `SuccessfulRequestCount` .
Comment puis-je surveiller la latence des appels de l'opération Amazon Textract ?	Utilisez la métrique `ResponseTime` .

Vous devez avoir les autorisations CloudWatch appropriées pour surveiller Amazon Textract avec CloudWatch. Pour plus d'informations, consultez Authentification et contrôle d'accès pour Amazon CloudWatch Logs.

Accéder aux métriques Amazon Textract

Les exemples suivants montrent comment accéder aux métriques Amazon Textract à l'aide de la console CloudWatch, l'AWS CLI, et l'API CloudWatch.

Pour consulter les métriques (console)

1. Ouvrez la console CloudWatch à l'adresse https://console.aws.amazon.com/ CloudWatch/.
2. ChoisissezMétriques, choisissez leToutes les métriques, puis choisissezAmazon Textract.
3. ChoisissezPar opération, puis choisissez une métrique.

   Par exemple, choisissez leStartDocumentAnalysismesure le nombre de fois où l'analyse de documents asynchrone a été lancée.
4. Choisissez une valeur pour la plage de dates. Nombre de métriques affichées dans le graphique.

Pour afficher des métriques de succès**`StartDocumentAnalysis`**appels d'opération effectués sur une période de temps (CLI)

- Ouvrez l'AWS CLI et entrez la commande suivante :

```
aws cloudwatch get-metric-statistics \
    --metric-name SuccessfulRequestCount \
    --start-time 2019-02-01T00:00:00Z \
    --period 3600 \
    --end-time 2019-03-01T00:00:00Z \
    --namespace AWS/Textract \
    --dimensions Name=Operation,Value=StartDocumentAnalysis \
    --statistics Sum
```

  Cet exemple illustre les appels réussis de l'opération `StartDocumentAnalysis` effectués sur une période de temps. Pour de plus amples informations, veuillez consulter get-metric-statistics.

Pour accéder aux métriques (API CloudWatch)

- Appelez `GetMetricStatistics`. Pour plus d'informations, consultez le .Référence d'API Amazon CloudWatch.

Créer une alarme

Vous pouvez créer une alarme CloudWatch qui envoie un message Amazon Simple Notification Service (Amazon SNS) quand l'alarme change d'état. Une alarme surveille une seule métrique pendant la période que vous spécifiez. Elle réalise une ou plusieurs actions en fonction de la valeur de la métrique par rapport à un seuil donné sur un certain nombre de périodes. L'action est une notification envoyée à une rubrique Amazon SNS ou à une stratégie Auto Scaling.

Les alarmes appellent les actions pour les changements d'état soutenus uniquement. Les alarmes CloudWatch ne déclenchent pas d'actions simplement parce qu'elles se trouvent dans un état particulier. L'état doit avoir changé et avoir été maintenu pendant un nombre de périodes spécifié.

Pour définir une alarme (console)

1. Connectez-vous à la AWS Management Console et ouvrez la console CloudWatch à l'adresse https://console.aws.amazon.com/cloudwatch/.

2. Dans le volet de navigation, choisissezAlarmes, et choisissezCréer une alarme. Cela ouvre leAssistant de création d'alarmes.
3. Choisissez Select metric (Sélectionner une métrique).
4. DansToutes les mesures, choisissezTextract.
5. ChoisissezPar opération, puis choisissez une métrique.

   Par exemple, choisissezStartDocumentAnalysispour définir une alarme pour un nombre maximal d'opérations d'analyse de documents asynchrone.
6. Sélectionnez l'onglet Graphed metrics (Graphiques des métriques).
7. Pour Statistics (Statistique), choisissez Sum (Somme).
8. Choisissez Select metric (Sélectionner une métrique).
9. Remplissez les champs Nom et Description. Pour Lorsque, choisissez **>=** et entrez une valeur maximale de votre choix.
10. Si vous voulez que CloudWatch vous envoie un e-mail quand l'état de l'alarme est atteint, pourÀ chaque fois que cette alarme :, choisissezL'état est ALARME. Pour envoyer des alarmes à une rubrique Amazon SNS existante, pourEnvoyer la notification à :, choisissez une rubrique SNS existante. Pour définir les noms et adresses de messagerie d'une nouvelle liste d'abonnement par e-mail, choisissez.Nouvelle liste. CloudWatch enregistre la liste et l'affiche dans le champ de telle sorte que vous puissiez l'utiliser pour définir de futures alarmes.

    > **Note**
    >
    > Si vous utilisezNouvelle listePour créer une rubrique Amazon SNS, les adresses e-mail doivent être vérifiées avant que les destinataires prévus ne reçoivent les notifications. Amazon SNS n'envoie les e-mails que lorsque l'alarme passe à l'état d'alarme. Si ce changement d'état de l'alarme se produit avant la vérification des adresses e-mail, les destinataires prévus ne reçoivent pas de notification.
11. Choisissez Create Alarm (Créer l'alarme).

Pour définir une alarme (AWS CLI)

- Ouvrez l'AWS CLI et entrez la commande suivante. Modifier la valeur du`alarm-actions`pour faire référence à une rubrique Amazon SNS que vous avez précédemment créée.

```
aws cloudwatch put-metric-alarm \
```

```
    --alarm-name StartDocumentAnalysisUserErrors \
    --alarm-description "Alarm when more than 10 StartDocumentAnalysys user errors
occur within 5 minutes" \
    --metric-name UserErrorCount \
    --namespace AWS/Textract \
    --statistic Sum \
    --period 300 \
    --threshold 10 \
    --comparison-operator GreaterThanThreshold \
    --evaluation-periods 1 \
    --unit Count \
    --dimensions Name=Operation,Value=StartDocumentAnalysis \
    --alarm-actions arn:aws:sns:us-east-1:111111111111:alarmtopic
```

Cet exemple montre comment créer une alarme lorsque plus de 10 erreurs d'utilisateur se produisent dans un délai de 5 minutes pour les appels à`StartDocumentAnalysis`. Pour de plus amples informations, veuillez consulter put-metric-alarm.

Pour définir une alarme (API CloudWatch)

- Appelez `PutMetricAlarm`. Pour de plus amples informations, veuillez consulterRéférence d'API Amazon CloudWatch.

Métriques CloudWatch pour Amazon Textract.

Cette section contient des informations sur les métriques Amazon CloudWatch et leOpérationdimension disponible pour Amazon Textract.

Vous pouvez également afficher une vue de groupe des métriques Amazon Textract à partir de la console Amazon Textract.

Métriques CloudWatch pour Amazon Textract.

Le tableau suivant récapitule les métriques Amazon Textract.

Métrique	Description
SuccessfulRequestCount	Nombre de requêtes réussies. La plage de codes de réponse d'une demande réussie est comprise entre 200 et 299.

Métrique	Description
	Unité : Nombre Statistiques valides : `Sum,Average`
ThrottledCount	Nombre de demandes limitées. Amazon Textract limite une demande quand il reçoit plus de demandes que la limite de transactions par seconde définie pour votre compte. Si cette limite est souvent franchie, vous pouvez demander une augmentation de la limite. Pour demander une augmentation, consultez Limites de service AWS. Unité : Nombre Statistiques valides : `Sum,Average`
ResponseTime	Durée en millisecondes pour qu'Amazon Textract calcule la réponse. Unités: 1. Nombre de statistiques `Data Samples` 2. Millisecondes pour les statistiques `Average` Statistiques valides : `Data Samples,Average` **Note** Le`ResponseTime` la mesure n'est pas incluse dans le volet de mesures Amazon Textract.
ServerErrorCount	Nombre d'erreurs de serveur. La plage des codes de réponse d'une erreur de serveur est comprise entre 500 et 599. Unité : Nombre Statistiques valides : `Sum,Average`

Métrique	Description
UserErrorCount	Nombre d'erreurs d'utilisateur (paramètres non valides, image non valide, absence d'autorisation, etc.). La plage des codes de réponse d'une erreur d'utilisateur est comprise entre 400 et 499. Unité : Nombre Statistiques valides : `Sum,Average`

CloudWatch Dimension pour Amazon Textract

Pour extraire les métriques spécifiques à une opération, utilisez l'espace de noms `AWS/Textract` et fournissez une dimension d'opération. Pour obtenir plus d'informations sur les dimensions, consultezDimensionsdans leGuide de l'utilisateur Amazon CloudWatch..

Journalisation des appels d'API Amazon Textract avecAWS CloudTrail

Amazon Textract est intégré àAWS CloudTrail, un service qui enregistre les actions réalisées par un utilisateur, un rôle ou unAWSservice dans Amazon Textract. CloudTrail capture tous les appels d'API pour Amazon Textract en tant qu'événements. Ces captures incluent les appels de la console Amazon Textract et les appels de code vers les opérations d'API Amazon Textract.

Si vous créez un journal d'activité, vous pouvez activer la livraison continue des événements CloudTrail dans un compartiment Amazon S3, y compris pour les événements relatifs à Amazon Textract. Si vous ne configurez pas de journal d'activité, vous pouvez toujours afficher les événements les plus récents dans la console CloudTrail dans Historique des événements. Avec les informations collectées par CloudTrail, vous pouvez déterminer la requête qui a été envoyée à Amazon Textract, l'adresse IP à partir de laquelle la requête a été exécutée, l'auteur et la date de la requête, ainsi que d'autres détails.

Pour en savoir plus sur CloudTrail, consultez le AWS CloudTrailGuide de l'utilisateur .

Informations Amazon Textract dans CloudTrail

CloudTrail est activé dans votre compte AWS lors de la création de ce dernier. Lorsqu'une activité a lieu dans Amazon Textract, cette activité est enregistrée dans un événement CloudTrail avec

d'autresAWSévénements de service dansHistorique des événements. Vous pouvez afficher, rechercher et télécharger les événements récents dans votre compte AWS. Pour de plus amples informations, veuillez consulter Affichage des événements avec l'historique des événements CloudTrail.

Pour un registre permanent des événements dans votreAWScompte, y compris les événements pour Amazon Textract, créez un journal de suivi. Un journal d'activité permet à CloudTrail de distribuer les fichiers journaux vers Amazon S3 bucket. Par défaut, lorsque vous créez un journal d'activité dans la console, il s'applique à toutes les régions AWS. Le journal d'activité consigne les événements de toutes les régions dans la partition AWS et livre les fichiers journaux dans le compartiment Amazon S3 de votre choix. De plus, vous pouvez configurer d'autresAWSpour analyser plus en profondeur les données d'événement collectées dans les journaux CloudTrail et agir sur celles-ci. Pour en savoir plus, consultez les ressources suivantes :

- Présentation de la création d'un journal d'activité
- Intégrations et services pris en charge par CloudTrail
- Configuration des Notifications de Amazon SNS pour CloudTrail
- Réception des fichiers journaux CloudTrail de plusieurs régions et Réception des fichiers journaux CloudTrail de plusieurs comptes

Toutes les opérations Amazon Textract sont enregistrées par CloudTrail et sont documentées dans le.API Reference. À titre d'exemple, les appels vers les actions `DetectDocumentText`, `AnalyzeDocument` et `GetDocumentText` génèrent des entrées dans les fichiers journaux CloudTrail.

Chaque événement ou entrée du journal contient des informations sur la personne qui a généré la demande. Les informations relatives à l'identité permettent de déterminer les éléments suivants :

- Si la demande a été effectuée avec les informations d'identification utilisateur racine ou Gestion des identités et des accès AWS (IAM).
- Si la demande a été effectuée avec les informations d'identification de sécurité temporaires d'un rôle ou d'un utilisateur fédéré.
- Si la requête a été effectuée par un autre service AWS.

Pour plus d'informations, veuillez consulter l'élément userIdentity CloudTrail.

Paramètres de demande et champs de réponse qui ne sont pas consignés

Pour des raisons de confidentialité, certains paramètres de demande et champs de réponse ne sont pas consignés, par exemple, des octets d'image de demande ou des informations de zone de sélection de réponse. Les noms de compartiments Amazon S3 et les noms de fichiers fournis dans les paramètres de demande sont fournis dans les entrées de journal CloudTrail. Aucune information sur les octets d'image transmis dans une demande n'est fournie dans un journal CloudTrail. Le tableau suivant présente les paramètres d'entrée et les paramètres de réponse qui ne sont pas consignés pour chaque opération Amazon Textract.

Opération	Paramètres de demande	Champs de réponse
AnalyzeDocument	`Bytes`	Tous
DetectDocumentText	`Bytes`	Tous
StartDocumentAnalysis	Aucun	Aucun
GetDocumentAnalysis	Aucun	Tous
StartDocumentTextDetection	Aucun	Aucun
GetDocumentTextDetection	Aucun	Tous

Présentation des entrées des fichiers journaux Amazon Textract

Un journal de suivi est une configuration qui permet d'envoyer des événements sous forme de fichiers journaux à un compartiment Amazon S3 que vous spécifiez. Les fichiers journaux CloudTrail peuvent contenir une ou plusieurs entrées. Une entrée de journal représente une demande individuelle à partir d'une source quelconque et comprend des informations sur l'opération demandée, y compris la date et l'heure de l'opération, les paramètres de la demande, etc. Les fichiers journaux CloudTrail ne constituent pas une série ordonnée retraçant les appels d'API publics. Ils ne suivent aucun ordre précis.

L'exemple suivant montre une entrée de journal CloudTrail qui illustre l'opération `AnalyzeDocument`. Les octets d'image de l'entrée`document`et les résultats d'analyse (`responseElements`) ne sont pas enregistrés.

```
{
```

```
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111111111111:user/janedoe",
        "accountId": "111111111111",
        "accessKeyId": "AIDACKCEVSQ6C2EXAMPLE",
        "userName": "janedoe"
    },
    "eventTime": "2019-04-03T23:56:31Z",
    "eventSource": "textract.amazonaws.com",
    "eventName": "AnalyzeDocument",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "198.51.100.0",
    "userAgent": "",
    "requestParameters": {
        "document": {},
        "featureTypes": [
            "TABLES"
        ]
    },
    "responseElements": null,
    "requestID": "e387676b-d1f0-4ea7-85d6-f5a344052dce",
    "eventID": "c5db79ce-e4ea-4401-8517-784481d559f7",
    "eventType": "AwsApiCall",
    "recipientAccountId": "111111111111"
}
```

L'exemple suivant montre une entrée de journal CloudTrail pour`StartDocumentAnalysis`. L'entrée de journal inclut le nom du compartiment Amazon S3 et le nom du fichier image dans`documentLocation`. Le journal inclut également la réponse de l'opération.

```
{
    "Records": [
        {
            "eventVersion": "1.05",
            "userIdentity": {
                "type": "IAMUser",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::111111111111:user/janedoe",
                "accountId": "11111111111",
                "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
                "userName": "janedoe"
```

```
            },
            "eventTime": "2019-04-04T01:42:24Z",
            "eventSource": "textract.amazonaws.com",
            "eventName": "StartDocumentAnalysis",
            "awsRegion": "us-east-1",
            "sourceIPAddress": "198.51.100.0",
            "userAgent": "",
            "requestParameters": {
                "documentLocation": {
                    "s3Object": {
                        "bucket": "bucket",
                        "name": "document.png"
                    }
                },
                "featureTypes": [
                    "TABLES"
                ]
            },
            "responseElements": {
                "jobId":
 "f3c718b444fa603d5d625ab967008f4b620d4650c9db8ca1cae01ef7efe51373"
            },
            "requestID": "9ae352e8-9de1-41ad-b77b-85aa348c2e82",
            "eventID": "f741bca0-c3cb-4805-82ea-baf76439deef",
            "eventType": "AwsApiCall",
            "recipientAccountId": "111111111111"
        }

    ]
}
```

Validation de conformité pour Amazon Textract

Des auditeurs tiers évaluent la sécurité et la conformité d'Amazon Textract dans le cadre de plusieursAWSprogrammes de conformité. Il s'agit notamment des normes HIPAA, SOC, ISO et PCI.

> **Note**
>
> Si vous traitez des données via le service Ttract soumis à la conformité PCI DSS, vous devez désactiver votre compte en contactant AWS Support et en suivant le processus qui vous est fourni.

Pour obtenir la liste des services AWS relevant de programmes de conformité spécifiques, consultez les Services AWS relevant de programmes de conformité. Pour obtenir des renseignements généraux, consultez Programmes de conformité AWS.

Vous pouvez télécharger les rapports de l'audit externe avec AWS Artifact. Pour de plus amples informations, veuillez consulter Téléchargement de rapports dans AWS Artifact.

Votre responsabilité de conformité lors de l'utilisation de Amazon Textract est déterminée par la sensibilité de vos données, les objectifs de conformité de votre entreprise, ainsi que par la législation et la réglementation applicables.AWSfournit les ressources suivantes pour faciliter la conformité :

- Guides de Quick Start (démarrage rapide) de la sécurité et de la conformité. Ces guides de déploiement traitent des considérations architecturales et fournissent des étapes pour déployer des environnements de base axés sur la sécurité et la conformité sur AWS.
- Livre blanc sur l'architecture pour la sécurité et la conformité HIPAA – Le livre blanc décrit comment les entreprises peuvent utiliser AWS pour créer des applications conformes à HIPAA.
- Ressources de conformité AWS – Cet ensemble de manuels et de guides peut s'appliquer à votre secteur d'activité et à votre emplacement.
- Évaluation des ressources à l'aide de règles dans le Guide du développeur AWS Config : le service AWS Config évalue dans quelle mesure vos configurations de ressources sont conformes aux pratiques internes, aux directives sectorielles et aux réglementations.
- AWS Security Hub CSPM— CeAWSfournit une vue complète de votre état de sécurité dansAWS. Le hub de sécurité vous aide à vérifier votre conformité avec les normes et les bonnes pratiques du secteur de la sécurité.

Résilience dans Amazon Textract

L'infrastructure mondiale AWS s'articule autours de régions et de zones de disponibilité AWS. AWS Les Régions fournissent plusieurs zones de disponibilité physiquement séparées et isolées, reliées par un réseau à latence faible, à haut débit et hautement redondant. Avec les zones de disponibilité, vous pouvez concevoir et exploiter des applications et des bases de données qui basculent automatiquement d'une zone à l'autre sans interruption. Les zones de disponibilité sont plus hautement disponibles, tolérantes aux pannes et évolutives que les infrastructures traditionnelles à un ou plusieurs centres de données.

Pour plus d'informations sur les régions et les zones de disponibilité AWS, consultez AWS Infrastructure mondiale.

> **Note**
>
> Le transfert de données entre régions n'est pas autorisé en raison du Règlement général sur la protection des données (GDPR).

Sécurité de l'infrastructure dans Amazon Textract

Amazon Textract est un service géré, protégé parAWSprocédures de sécurité du réseau mondial décrites dansAmazon Web Services : Présentation des processus de sécuritélivre blanc.

Vous utilisezAWSAppels d'API publiés pour accéder à Amazon Textract via le réseau. Les clients doivent supporter le protocole TLS (Sécurité de la couche transport) 1.0 ou une version ultérieure. Nous recommandons TLS 1.2 ou version ultérieure. Les clients doivent aussi prendre en charge les suites de chiffrement PFS (Perfect Forward Secrecy) comme Ephemeral Diffie-Hellman (DHE) ou Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). La plupart des systèmes modernes tels que Java 7 et les versions ultérieures prennent en charge ces modes.

En outre, les demandes doivent être signées à l'aide d'un ID de clé d'accès et d'une clé d'accès secrète associée à un principal IAM. Vous pouvez également utiliser AWS Security Token Service (AWS STS) pour générer des informations d'identification de sécurité temporaires et signer les demandes.

Configuration et analyse des vulnérabilités dans Amazon Textract

La configuration et les contrôles informatiques sont une responsabilité partagée entre AWS et vous, notre client. Pour de plus amples informations, veuillez consulter Modèle de responsabilité partagée AWS.

Amazon Textract et points de terminaison d'un VPC (AWS PrivateLink)

Vous pouvez établir une connexion privée entre votre VPC et Amazon Textract en créant unPoint de terminaison d'un VPC d'interface. Les points de terminaison d'interface sont alimentés parAWS PrivateLink, technologie qui vous permet d'accéder en privé aux API Amazon Textract sans passerelle Internet, périphérique NAT, connexion VPN ou connexion AWS Direct Connect. Les

instances de votre VPC ne nécessitent pas d'adresses IP publiques pour communiquer avec les API Amazon Textract. Le trafic entre votre VPC et Amazon Textract ne quitte pas le réseau AWS.

Chaque point de terminaison d'interface est représenté par une ou plusieurs interfaces réseau Elastic dans vos sous-réseaux.

Pour de plus amples informations, veuillez consulter Points de terminaison d'un VPC d'interface (AWS PrivateLink) dans le Guide de l'utilisateur Amazon VPC.

Considérations relatives aux points de terminaison de VPC Amazon Textract

Avant de configurer un point de terminaison de VPC d'interface pour Amazon Textract, assurez-vous de vérifier.Propriétés et limites des points de terminaison d'interfacedans leAmazon VPC User Guide.

Amazon Textract est compatible avec l'exécution d'appels en direction de toutes ses actions d'API à partir de votre VPC.

Création d'un point de terminaison de VPC d'interface pour Amazon Textract

Vous pouvez créer un point de terminaison de VPC pour le service Amazon Textract à l'aide de la console Amazon VPC ou d'AWS Command Line Interface(AWS CLI). Pour de plus amples informations, veuillez consulter Création d'un point de terminaison d'interface dans le Guide de l'utilisateur Amazon VPC.

Créez un point de terminaison de VPC pour Amazon Textract à l'aide du nom de service suivant :

- com.amazonaws.*région*.textract : permet de créer un point de terminaison pour la plupart des opérations Amazon Textract.
- com.amazonaws.*région*.textract-fips - Permet de créer un point de terminaison pour Amazon Textract conforme avec la norme FIPS (Federal Information Processing Standard) 140-2 du gouvernement américain.

Si vous activez le DNS privé pour le point de terminaison, vous pouvez faire des demandes d'API à Amazon Textract à l'aide de son nom DNS par défaut pour la région, par exemple,`textract.us-east-1.amazonaws.com`.

Pour plus d'informations, consultez Accès à un service via un point de terminaison d'interface dans le guide de l'utilisateur Amazon VPC.

Création d'une stratégie de point de terminaison de VPC pour Amazon Textract

Vous pouvez attacher une stratégie de point de terminaison à votre point de terminaison de VPC qui contrôle l'accès à Amazon Textract. La politique spécifie les informations suivantes :

- Le principal qui peut exécuter des actions.
- Les actions qui peuvent être effectuées.
- Les ressources sur lesquelles les actions peuvent être exécutées.

Pour plus d'informations, veuillez consulter Contrôle de l'accès aux services avec points de terminaison d'un VPC dans le Amazon VPC Guide de l'utilisateur.

Exemple : Stratégie de point de terminaison de VPC pour les actions Amazon Textract

Voici un exemple de stratégie de point de terminaison pour Amazon Textract. Lorsqu'elle est attachée à un point de terminaison, cette stratégie accorde l'accès aux actions Amazon Textract répertoriées pour tous les mandataires sur toutes les ressources.

Cet exemple de stratégie autorise l'accès aux opérations uniquement.`DetectDocumentTextetAnalyzeDocument`. Les utilisateurs peuvent toujours appeler des opérations Amazon Textract à partir de l'extérieur du point de terminaison de VPC.

```
   "Statement":[
      {
         "Principal":"*",
         "Effect":"Allow",
         "Action":[
            "textract:DetectDocumentText",
            "textract:AnalyzeDocument",
         ],
         "Resource":"*"
      }
   ]
}
```

Référence API

Cette section fournit la documentation concernant les opérations API Amazon Textract.

Rubriques

Actions

Les actions suivantes sont prises en charge :

- AnalyzeDocument
- AnalyzeExpense
- AnalyzeID
- DetectDocumentText
- GetDocumentAnalysis
- GetDocumentTextDetection
- GetExpenseAnalysis
- StartDocumentAnalysis
- StartDocumentTextDetection
- StartExpenseAnalysis

AnalyzeDocument

Analyse un document d'entrée afin de rechercher des relations entre les éléments détectés.

Les types d'informations renvoyées sont les suivants :

- Données de formulaire (paires clé-valeur). Les informations associées sont renvoyées dans deuxBlockobjets, chacun de typeKEY_VALUE_SET: une cléBlockobjet et une VALEURBlockobjet. Par exemple,Name : Ana Silva Carolinecontient une clé et une valeur. Name :est la clé. Ana Silva Carolineest la valeur.
- Données de cellules de table et de table. UNE TABLEBlockcontient des informations sur une table détectée. UNE CELLULEBlockest renvoyé pour chaque cellule d'un tableau.
- Lignes et mots de texte. UNE LIGNEBlockl'objet contient un ou plusieurs motsBlockobjets. Toutes les lignes et tous les mots détectés dans le document sont renvoyés (y compris le texte qui n'a pas de relation avec la valeur deFeatureTypes).

Les éléments de sélection tels que les cases à cocher et les boutons d'option (boutons radio) peuvent être détectés dans les données de formulaire et dans les tableaux. UN ÉLÉMENT SÉLECTION_Blockcontient des informations sur un élément de sélection, y compris l'état de la sélection.

Vous pouvez choisir le type d'analyse à effectuer en spécifiant leFeatureTypesliste.

La sortie est renvoyée dans une liste deBlockobjets.

AnalyzeDocumentest une opération synchrone. Pour analyser des documents de manière asynchrone, utilisezStartDocumentAnalysis.

Pour de plus amples informations, veuillez consulterAnalyse du texte du document.

Syntaxe de la demande

```
{
   "Document": {
      "Bytes": blob,
      "S3Object": {
         "Bucket": "string",
         "Name": "string",
         "Version": "string"
      }
```

```
    },
    "FeatureTypes": [ "string" ],
    "HumanLoopConfig": {
       "DataAttributes": {
          "ContentClassifiers": [ "string" ]
       },
       "FlowDefinitionArn": "string",
       "HumanLoopName": "string"
    }
}
```

Paramètres de demande

Cette demande accepte les données suivantes au format JSON.

Document

Le document d'entrée sous forme d'octets codés en base64 ou d'un objet Amazon S3. Si vous utilisez l'interface de ligne de commande AWS pour appeler les opérations Amazon Textract, vous ne pouvez pas transmettre d'octets d'image. Le document doit être une image au format JPEG, PNG, PDF ou TIFF.

Si vous utilisez un kit SDK AWS pour appeler Amazon Textract, il est possible que vous n'ayez pas besoin de coder en base64 octets d'image transmis à l'aide de l'option`Bytes`.

Type : objet Document

Obligatoire Oui

FeatureTypes

Liste des types d'analyses à effectuer. Ajoutez TABLES à la liste pour renvoyer des informations sur les tables détectées dans le document d'entrée. Ajoutez FORMS pour renvoyer les données de formulaire détectées. Pour effectuer les deux types d'analyse, ajoutez TABLES et FORMS à`FeatureTypes`. Toutes les lignes et tous les mots détectés dans le document sont inclus dans la réponse (y compris le texte qui n'est pas lié à la valeur de`FeatureTypes`).

Type : Tableau de chaînes

Valeurs valides : `TABLES | FORMS`

Obligatoire Oui

HumanLoopConfig

Définit la configuration du flux de travail humain dans la boucle pour analyser les documents.

Type : objet HumanLoopConfig

Obligatoire Non

Syntaxe de la réponse


```
{
   "AnalyzeDocumentModelVersion": "string",
   "Blocks": [
      {
         "BlockType": "string",
         "ColumnIndex": number,
         "ColumnSpan": number,
         "Confidence": number,
         "EntityTypes": [ "string" ],
         "Geometry": {
            "BoundingBox": {
               "Height": number,
               "Left": number,
               "Top": number,
               "Width": number
            },
            "Polygon": [
               {
                  "X": number,
                  "Y": number
               }
            ]
         },
         "Id": "string",
         "Page": number,
         "Relationships": [
            {
               "Ids": [ "string" ],
               "Type": "string"
            }
         ],
         "RowIndex": number,
         "RowSpan": number,
```

```
            "SelectionStatus": "string",
            "Text": "string",
            "TextType": "string"
         }
      ],
      "DocumentMetadata": {
         "Pages": number
      },
      "HumanLoopActivationOutput": {
         "HumanLoopActivationConditionsEvaluationResults": "string",
         "HumanLoopActivationReasons": [ "string" ],
         "HumanLoopArn": "string"
      }
}
```

Eléments de réponse

Si l'action aboutit, le service renvoie une réponse HTTP 200.

Les données suivantes sont renvoyées au format JSON par le service.

AnalyzeDocumentModelVersion

Version du modèle utilisée pour analyser le document.

Type : Chaîne

Blocks

Les éléments détectés et analysés par`AnalyzeDocument`.

Type : Tableau deBlockobjets

DocumentMetadata

Métadonnées concernant le document analysé. Par exemple, le nombre de pages est le nombre de pages.

Type : objet DocumentMetadata

HumanLoopActivationOutput

Affiche les résultats de l'évaluation humaine dans la boucle.

Type : objet HumanLoopActivationOutput

Erreurs

AccessDeniedException

Vous n'êtes pas autorisé à effectuer l'action. Utilisez l'Amazon Resource Name (ARN) d'un utilisateur ou d'un rôle IAM autorisé pour effectuer l'opération.

HTTP Status Code : 400

BadDocumentException

Amazon Textract n'est pas en mesure de lire le document. Pour plus d'informations sur les limites de documents dans Amazon Textract, voirLimites strictes dans Amazon Textract.

HTTP Status Code : 400

DocumentTooLargeException

Le document ne peut pas être traité car il est trop volumineux. Taille maximale du document pour les opérations synchrone 10 Mo. La taille maximale du document pour les opérations asynchrones est de 500 Mo pour les fichiers PDF.

HTTP Status Code : 400

HumanLoopQuotaExceededException

Indique que vous avez dépassé le nombre maximum d'humains actifs dans les flux de travail en boucle disponibles

HTTP Status Code : 400

InternalServerError

Amazon Textract a rencontré un problème de service. Renouvelez votre appel.

HTTP Status Code : 500

InvalidParameterException

Un paramètre d'entrée a enfreint une contrainte. Par exemple, dans les opérations synchrone, un`InvalidParameterException`exception se produit lorsque aucune des options`S3Object`ou`Bytes`les valeurs sont fournies dans le`Document`paramètre de demande. Validez votre paramètre avant d'appeler à nouveau l'opération d'API.

HTTP Status Code : 400

InvalidS3ObjectException

Amazon Textract n'est pas en mesure d'accéder à l'objet S3 spécifié dans la demande. Pour plus d'informations,Configuration de l'accès à Amazon S3Pour plus d'informations sur le dépannage, consultezRésolutions des problèmes liés à Amazon S3

HTTP Status Code : 400

ProvisionedThroughputExceededException

Le nombre de demandes dépasse votre limite de débit. Si vous avez besoin d'augmenter cette limite, contactez Amazon Textract.

HTTP Status Code : 400

ThrottlingException

Amazon Textract est temporairement dans l'impossibilité de traiter la demande. Renouvelez votre appel.

HTTP Status Code : 500

UnsupportedDocumentException

Le format du document d'entrée n'est pas pris en charge. Les documents utilisés pour les opérations peuvent être au format PNG, JPEG, PDF ou TIFF.

HTTP Status Code : 400

Voir aussi

Pour plus d’informations sur l’utilisation de cette API dans l’un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Interface de ligne de commande AWS
- AWS SDK pour .NET
- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- Kit AWS SDK pour JavaScript
- AWS SDK pour PHP V3

- AWS SDK pour Python
- AWSSDK pour Ruby V3

AnalyzeExpense

`AnalyzeExpense`analyse de manière synchrone un document d'entrée pour détecter les relations financières entre le texte.

Les informations sont renvoyées sous la forme`ExpenseDocuments`et se sont séparés comme suit.

- `LineItemGroups`- Un ensemble de données contenant`LineItems`qui stockent des informations sur les lignes de texte, telles qu'un article acheté et son prix sur un reçu.
- `SummaryFields`- Contient toutes les autres informations d'un reçu, telles que les informations d'en-tête ou le nom du fournisseur.

Syntaxe de la demande

```
{
   "Document": {
      "Bytes": blob,
      "S3Object": {
         "Bucket": "string",
         "Name": "string",
         "Version": "string"
      }
   }
}
```

Paramètres de demande

Cette demande accepte les données suivantes au format JSON.

Document

Le document d'entrée, soit en octets, soit en tant qu'objet S3.

Pour transmettre des octets d'image à une opération d'API Amazon Textract, utilisez la commande`Bytes`propriété. Par exemple, utilisez la commande`Bytes`pour transmettre un document chargé à partir d'un système de fichiers local. Octets d'image transmis à l'aide de l'option`Bytes`doit être codée en base64. Il est possible que votre code n'ait pas besoin de coder des octets de fichiers de documents si vous utilisez un kit SDK AWS pour appeler les opérations de l'API Amazon Textract.

Pour transmettre des images stockées dans un compartiment S3 à une opération d'API Amazon Textract, utilisez la commande`S3Object`propriété. Il n'est pas nécessaire d'encoder en base64 les documents stockés dans un compartiment S3.

La région AWS du compartiment S3 contenant l'objet S3 doit correspondre à la région AWS que vous utilisez pour les opérations Amazon Textract.

Si vous utilisez l'AWS CLI pour appeler des opérations Amazon Textract, la transmission d'octets d'image à l'aide de la propriété Bytes n'est pas prise en charge. Vous devez d'abord charger le document dans un compartiment Amazon S3, puis appeler l'opération à l'aide de la propriété S3Object.

Pour qu'Amazon Textract traite un objet S3, l'utilisateur doit disposer des autorisations permettant d'accéder à l'objet S3.

Type : objet Document

Regatoire : Oui

Syntaxe de la réponse

```
{
   "DocumentMetadata": {
      "Pages": number
   },
   "ExpenseDocuments": [
      {
         "ExpenseIndex": number,
         "LineItemGroups": [
            {
               "LineItemGroupIndex": number,
               "LineItems": [
                  {
                     "LineItemExpenseFields": [
                        {
                           "LabelDetection": {
                              "Confidence": number,
                              "Geometry": {
                                 "BoundingBox": {
                                    "Height": number,
                                    "Left": number,
                                    "Top": number,
```

```
                                    "Width": number
                                },
                                "Polygon": [
                                    {
                                        "X": number,
                                        "Y": number
                                    }
                                ]
                            },
                            "Text": "string"
                        },
                        "PageNumber": number,
                        "Type": {
                            "Confidence": number,
                            "Text": "string"
                        },
                        "ValueDetection": {
                            "Confidence": number,
                            "Geometry": {
                                "BoundingBox": {
                                    "Height": number,
                                    "Left": number,
                                    "Top": number,
                                    "Width": number
                                },
                                "Polygon": [
                                    {
                                        "X": number,
                                        "Y": number
                                    }
                                ]
                            },
                            "Text": "string"
                        }
                    }
                ]
            }
        ]
    }
],
"SummaryFields": [
    {
        "LabelDetection": {
            "Confidence": number,
```

```
                    "Geometry": {
                       "BoundingBox": {
                          "Height": number,
                          "Left": number,
                          "Top": number,
                          "Width": number
                       },
                       "Polygon": [
                          {
                             "X": number,
                             "Y": number
                          }
                       ]
                    },
                    "Text": "string"
                 },
                 "PageNumber": number,
                 "Type": {
                    "Confidence": number,
                    "Text": "string"
                 },
                 "ValueDetection": {
                    "Confidence": number,
                    "Geometry": {
                       "BoundingBox": {
                          "Height": number,
                          "Left": number,
                          "Top": number,
                          "Width": number
                       },
                       "Polygon": [
                          {
                             "X": number,
                             "Y": number
                          }
                       ]
                    },
                    "Text": "string"
                 }
              }
           ]
        }
     ]
```

```
}
```

Eléments de réponse

Si l'action aboutit, le service renvoie une réponse HTTP 200.

Les données suivantes sont renvoyées au format JSON par le service.

DocumentMetadata

Informations sur le document d'entrée.

Type : objet DocumentMetadata

ExpenseDocuments

Les dépenses détectées par Amazon Textract.

Type : Tableau deExpenseDocumentobjets

Erreurs

AccessDeniedException

Vous n'êtes pas autorisé à effectuer l'action. Utilisez l'Amazon Resource Name (ARN) d'un utilisateur ou d'un rôle IAM autorisé pour effectuer l'opération.

Code d'état HTTP : 400

BadDocumentException

Amazon Textract n'est pas en mesure de lire le document. Pour plus d'informations sur les limites de documents dans Amazon Textract, voirLimites strictes dans Amazon Textract.

Code d'état HTTP : 400

DocumentTooLargeException

Le document ne peut pas être traité car il est trop volumineux. Taille maximale du document pour les opérations synchrone 10 Mo. La taille maximale du document pour les opérations asynchrones est de 500 Mo pour les fichiers PDF.

Code d'état HTTP : 400

InternalServerError

Amazon Textract a rencontré un problème de service. Renouvelez votre appel.

Code d'état HTTP : 500

InvalidParameterException

Un paramètre d'entrée a enfreint une contrainte. Par exemple, dans les opérations synchrone, un`InvalidParameterException`exception se produit lorsque aucune des options`S3Object`ou`Bytes`les valeurs sont fournies dans le`Document`paramètre de demande. Validez votre paramètre avant d'appeler à nouveau l'opération d'API.

Code d'état HTTP : 400

InvalidS3ObjectException

Amazon Textract n'est pas en mesure d'accéder à l'objet S3 spécifié dans la demande. Pour plus d'informations, consultez la pageConfiguration de l'accès à Amazon S3Pour plus d'informations sur le dépannage, consultezRésolutions des problèmes liés à Amazon S3

Code d'état HTTP : 400

ProvisionedThroughputExceededException

Le nombre de demandes dépasse votre limite de débit. Si vous avez besoin d'augmenter cette limite, contactez Amazon Textract.

Code d'état HTTP : 400

ThrottlingException

Amazon Textract est temporairement dans l'impossibilité de traiter la demande. Renouvelez votre appel.

Code d'état HTTP : 500

UnsupportedDocumentException

Le format du document d'entrée n'est pas prise en charge. Les documents utilisés pour les opérations peuvent être au format PNG, JPEG, PDF ou TIFF.

Code d'état HTTP : 400

Voir aussi

Pour plus d'informations sur l'utilisation de cette API dans l'un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Interface de ligne de commande AWS
- AWS SDK pour .NET
- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- Kit AWS SDK pour JavaScript
- AWS SDK pour PHP V3
- AWS SDK pour Python
- AWSSDK pour Ruby V3

AnalyzeID

Analyse les documents d'identité à la recherche d'informations pertinentes. Ces informations sont extraites et renvoyées sous la formeIdentityDocumentFields, qui enregistre à la fois le champ normalisé et la valeur du texte extrait. Contrairement aux autres opérations Amazon Textract,AnalyzeIDne renvoie aucune donnée de géométrie.

Syntaxe de la demande

```
{
   "DocumentPages": [
      {
         "Bytes": blob,
         "S3Object": {
            "Bucket": "string",
            "Name": "string",
            "Version": "string"
         }
      }
   ]
}
```

Paramètres de demande

Cette demande accepte les données suivantes au format JSON.

DocumentPages

Le document transmis à AnalyzeID.

Type : Tableau deDocumentobjets

Membres du tableau : Nombre minimum de 1 élément. Nombre maximum de 2 éléments.

Nécessaire : Oui

Syntaxe de la réponse

```
{
   "AnalyzeIDModelVersion": "string",
   "DocumentMetadata": {
```

```
      "Pages": number
   },
   "IdentityDocuments": [
      {
         "DocumentIndex": number,
         "IdentityDocumentFields": [
            {
               "Type": {
                  "Confidence": number,
                  "NormalizedValue": {
                     "Value": "string",
                     "ValueType": "string"
                  },
                  "Text": "string"
               },
               "ValueDetection": {
                  "Confidence": number,
                  "NormalizedValue": {
                     "Value": "string",
                     "ValueType": "string"
                  },
                  "Text": "string"
               }
            }
         ]
      }
   ]
}
```

Eléments de réponse

Si l'action aboutit, le service renvoie une réponse HTTP 200.

Les données suivantes sont renvoyées au format JSON par le service.

AnalyzeIDModelVersion

La version de l'API AnalyzeIdentity utilisée pour traiter des documents.

Type : Chaîne

DocumentMetadata

Informations sur le document d'entrée.

Type : objet DocumentMetadata

IdentityDocuments

Liste des documents traités par AnalyzeID. Comprend un numéro indiquant leur place dans la liste et la structure de réponse du document.

Type : Tableau deIdentityDocumentobjets

Erreurs

AccessDeniedException

Vous n'êtes pas autorisé à effectuer l'action. Utilisez l'Amazon Resource Name (ARN) d'un utilisateur ou d'un rôle IAM autorisé pour effectuer l'opération.

HTTP Status Code : 400

BadDocumentException

Amazon Textract n'est pas en mesure de lire le document. Pour plus d'informations sur les limites de documents dans Amazon Textract, voirLimites strictes dans Amazon Textract.

HTTP Status Code : 400

DocumentTooLargeException

Le document ne peut pas être traité car il est trop volumineux. Taille maximale du document pour les opérations synchrone 10 Mo. La taille maximale du document pour les opérations asynchrones est de 500 Mo pour les fichiers PDF.

HTTP Status Code : 400

InternalServerError

Amazon Textract a rencontré un problème de service. Renouvelez votre appel.

HTTP Status Code : 500

InvalidParameterException

Un paramètre d'entrée a enfreint une contrainte. Par exemple, dans les opérations synchrone, un`InvalidParameterException`exception se produit lorsque aucune des options`S3Object`ou`Bytes`les valeurs sont fournies dans le`Document`paramètre de demande. Validez votre paramètre avant d'appeler à nouveau l'opération d'API.

HTTP Status Code : 400

InvalidS3ObjectException

Amazon Textract est dans l'impossibilité d'accéder à l'objet S3 spécifié dans la demande. Pour plus d'informations,Configuration de l'accès à Amazon S3Pour plus d'informations sur le dépannage, consultezRésolutions des problèmes liés à Amazon S3

HTTP Status Code : 400

ProvisionedThroughputExceededException

Le nombre de demandes dépasse votre limite de débit. Si vous avez besoin d'augmenter cette limite, contactez Amazon Textract.

HTTP Status Code : 400

ThrottlingException

Amazon Textract est temporairement dans l'impossibilité de traiter la demande. Renouvelez votre appel.

HTTP Status Code : 500

UnsupportedDocumentException

Le format du document d'entrée n'est pas pris en charge. Les documents utilisés pour les opérations peuvent être au format PNG, JPEG, PDF ou TIFF.

HTTP Status Code : 400

Voir aussi

Pour plus d’informations sur l’utilisation de cette API dans l’un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Interface de ligne de commande AWS
- AWS SDK pour .NET
- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- Kit AWS SDK pour JavaScript

- AWS SDK pour PHP V3
- AWS SDK pour Python
- AWSSDK pour Ruby V3

DetectDocumentText

Détecte le texte dans le document d'entrée. Amazon Textract peut détecter les lignes de texte et les mots qui constituent une ligne de texte. Le document d'entrée doit être une image au format JPEG, PNG, PDF ou TIFF.`DetectDocumentText`renvoie le texte détecté dans un tableau deBlockobjets.

Chaque page de document est associée`Block`de type PAGE. Chaque PAGE`Block`objet est le parent de LINE`Block`objets qui représentent les lignes de texte détecté sur une page. UNE LIGNE`Block`objet est un parent pour chaque mot qui constitue la ligne. Les mots sont représentés par`Block`objets de type WORD.

`DetectDocumentText`est une opération synchrone. Pour analyser des documents de manière asynchrone, utilisezStartDocumentTextDetection.

Pour de plus amples informations, veuillez consulterDétection de texte de document.

Syntaxe de la demande

```
{
   "Document": {
      "Bytes": blob,
      "S3Object": {
         "Bucket": "string",
         "Name": "string",
         "Version": "string"
      }
   }
}
```

Paramètres de demande

Cette demande accepte les données suivantes au format JSON.

Document

Le document d'entrée sous forme d'octets codés en base64 ou d'un objet Amazon S3. Si vous utilisez l'interface de ligne de commande AWS pour appeler les opérations Amazon Textract, vous ne pouvez pas transmettre d'octets d'image. Le document doit être une image au format JPEG ou PNG.

Si vous utilisez un kit SDK AWS pour appeler Amazon Textract, il est possible que vous n'ayez pas besoin de coder en base64 octets d'image transmis à l'aide de l'option`Bytes`.

Type : objet Document

Obligatoire Oui

Syntaxe de la réponse

```
{
   "Blocks": [
      {
         "BlockType": "string",
         "ColumnIndex": number,
         "ColumnSpan": number,
         "Confidence": number,
         "EntityTypes": [ "string" ],
         "Geometry": {
            "BoundingBox": {
               "Height": number,
               "Left": number,
               "Top": number,
               "Width": number
            },
            "Polygon": [
               {
                  "X": number,
                  "Y": number
               }
            ]
         },
         "Id": "string",
         "Page": number,
         "Relationships": [
            {
               "Ids": [ "string" ],
               "Type": "string"
            }
         ],
         "RowIndex": number,
         "RowSpan": number,
         "SelectionStatus": "string",
         "Text": "string",
         "TextType": "string"
      }
   ],
```

```
   "DetectDocumentTextModelVersion": "string",
   "DocumentMetadata": {
      "Pages": number
   }
}
```

Eléments de réponse

Si l'action aboutit, le service renvoie une réponse HTTP 200.

Les données suivantes sont renvoyées au format JSON par le service.

Blocks

Tableau d'élémentsBlockobjets contenant le texte détecté dans le document.

Type : Tableau deBlockobjets

DetectDocumentTextModelVersion

Type : Chaîne

DocumentMetadata

Métadonnées concernant le document. Il contient le nombre de pages détectées dans le document.

Type : objet DocumentMetadata

Erreurs

AccessDeniedException

Vous n'êtes pas autorisé à effectuer l'action. Utilisez l'Amazon Resource Name (ARN) d'un utilisateur ou d'un rôle IAM autorisé pour effectuer l'opération.

HTTP Status Code : 400

BadDocumentException

Amazon Textract n'est pas en mesure de lire le document. Pour plus d'informations sur les limites de documents dans Amazon Textract, voirLimites strictes dans Amazon Textract.

HTTP Status Code : 400

DocumentTooLargeException

Le document ne peut pas être traité car il est trop volumineux. Taille maximale du document pour les opérations synchrone 10 Mo. La taille maximale du document pour les opérations asynchrones est de 500 Mo pour les fichiers PDF.

HTTP Status Code : 400

InternalServerError

Amazon Textract a rencontré un problème de service. Renouvelez votre appel.

HTTP Status Code : 500

InvalidParameterException

Un paramètre d'entrée a enfreint une contrainte. Par exemple, dans les opérations synchrone, un`InvalidParameterException`exception se produit lorsque aucune des options`S3Object`ou`Bytes`les valeurs sont fournies dans le`Document`paramètre de demande. Validez votre paramètre avant d'appeler à nouveau l'opération d'API.

HTTP Status Code : 400

InvalidS3ObjectException

Amazon Textract n'est pas en mesure d'accéder à l'objet S3 spécifié dans la demande. Pour plus d'informations,Configuration de l'accès à Amazon S3Pour plus d'informations sur le dépannage, consultezRésolutions des problèmes liés à Amazon S3

HTTP Status Code : 400

ProvisionedThroughputExceededException

Le nombre de demandes dépasse votre limite de débit. Si vous avez besoin d'augmenter cette limite, contactez Amazon Textract.

HTTP Status Code : 400

ThrottlingException

Amazon Textract est temporairement dans l'impossibilité de traiter la demande. Renouvelez votre appel.

HTTP Status Code : 500

UnsupportedDocumentException

Le format du document d'entrée n'est pas pris en charge. Les documents utilisés pour les opérations peuvent être au format PNG, JPEG, PDF ou TIFF.

HTTP Status Code : 400

Voir aussi

Pour plus d’informations sur l’utilisation de cette API dans l’un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Interface de ligne de commande AWS
- AWS SDK pour .NET
- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- Kit AWS SDK pour JavaScript
- AWS SDK pour PHP V3
- AWS SDK pour Python
- AWSSDK pour Ruby V3

GetDocumentAnalysis

Obtient les résultats d'une opération asynchrone Amazon Textract qui analyse du texte dans un document.

Vous commencez l'analyse de texte asynchrone en appelantStartDocumentAnalysis, qui renvoie un identifiant de tâche (`JobId`). Lorsque l'opération d'analyse de texte est terminée, Amazon Textract publie un état d'achèvement dans la rubrique Amazon Simple Notification Service (Amazon SNS) enregistrée lors de l'appel initial à`StartDocumentAnalysis`. Pour obtenir les résultats de l'opération de détection de texte, vérifiez d'abord que la valeur d'état publiée sur la rubrique Amazon SNS est`SUCCEEDED`. Si c'est le cas, appelez`GetDocumentAnalysis`, et transmettez l'identificateur de la tâche (`JobId`) depuis l'appel initial à`StartDocumentAnalysis`.

`GetDocumentAnalysis`renvoie un tableau deBlockobjets. Les types d'informations suivants sont renvoyés :

- Données de formulaire (paires clé-valeur). Les informations associées sont renvoyées dans deuxBlockobjets, chacun de type`KEY_VALUE_SET`: une clé`Block`objet et une VALEUR`Block`objet. Par exemple,`Name : Ana Silva Carolina`contient une clé et une valeur. `Name :`est la clé. `Ana Silva Carolina`est la valeur.
- Données de cellules de table et de tableau. TABLE`Block`contient des informations sur une table détectée. UNE CELLULE`Block`est renvoyé pour chaque cellule d'un tableau.
- Lignes et mots de texte. UNE LIGNE`Block`l'objet contient un ou plusieurs mots`Block`objets. Toutes les lignes et tous les mots détectés dans le document sont renvoyés (y compris le texte qui n'a pas de relation avec la valeur du`StartDocumentAnalysis FeatureTypes`paramètre d'entrée).

Les éléments de sélection tels que les cases à cocher et les boutons d'option (boutons radio) peuvent être détectés dans les données de formulaire et dans les tableaux. UN ÉLÉMENT SÉLECTION_`Block`contient des informations sur un élément de sélection, y compris l'état de la sélection.

Utilisation de l'`MaxResults`pour limiter le nombre de blocs renvoyés. S'il y a plus de résultats que ceux spécifiés dans`MaxResults`, la valeur de`NextToken`dans la réponse d'opération contient un jeton de pagination permettant d'obtenir l'ensemble suivant de résultats. Pour obtenir la page de résultats suivante, appelez`GetDocumentAnalysis`, puis remplissez-le`NextToken`paramètre request avec la valeur du jeton renvoyée par l'appel précédent à`GetDocumentAnalysis`.

Pour de plus amples informations, veuillez consulterAnalyse de texte de document.

Syntaxe de la demande

```
{
   "JobId": "string",
   "MaxResults": number,
   "NextToken": "string"
}
```

Paramètres de demande

Cette demande accepte les données suivantes au format JSON.

JobId

Identifiant unique du travail de détection de texte. LeJobIdest renvoyée parStartDocumentAnalysis. UNJobIdn'est valide que pendant 7 jours.

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximale de 64.

Modèle : ^[a-zA-Z0-9-_]+$

Obligatoire Oui

MaxResults

Nombre maximal de résultats à renvoyer par appel paginé. La valeur la plus élevée que vous pouvez spécifier est 1 000. Si vous spécifiez une valeur supérieure à 1 000, seuls 1 000 résultats sont renvoyés au maximum. La valeur par défaut est 1,000.

Type : Entier

Plage valide : Valeur minimale est 1.

Obligatoire Non

NextToken

Si la réponse précédente était incomplète (car il y a plus de blocs à récupérer), Amazon Textract renvoie un jeton de pagination dans la réponse. Vous pouvez utiliser ce jeton de pagination pour récupérer l'ensemble suivant de blocs.

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximale de 255.

Modèle : `.*\S.*`

Obligatoire Non

Syntaxe de la réponse

```
{
   "AnalyzeDocumentModelVersion": "string",
   "Blocks": [
      {
         "BlockType": "string",
         "ColumnIndex": number,
         "ColumnSpan": number,
         "Confidence": number,
         "EntityTypes": [ "string" ],
         "Geometry": {
            "BoundingBox": {
               "Height": number,
               "Left": number,
               "Top": number,
               "Width": number
            },
            "Polygon": [
               {
                  "X": number,
                  "Y": number
               }
            ]
         },
         "Id": "string",
         "Page": number,
         "Relationships": [
            {
               "Ids": [ "string" ],
               "Type": "string"
            }
         ],
         "RowIndex": number,
         "RowSpan": number,
```

```
            "SelectionStatus": "string",
            "Text": "string",
            "TextType": "string"
         }
      ],
      "DocumentMetadata": {
         "Pages": number
      },
      "JobStatus": "string",
      "NextToken": "string",
      "StatusMessage": "string",
      "Warnings": [
         {
            "ErrorCode": "string",
            "Pages": [ number ]
         }
      ]
}
```

Eléments de réponse

Si l'action aboutit, le service renvoie une réponse HTTP 200.

Les données suivantes sont renvoyées au format JSON par le service.

AnalyzeDocumentModelVersion

Type : Chaîne

Blocks

Les résultats de l'opération d'analyse de texte.

Type : Grappe deBlockobjets

DocumentMetadata

Informations sur un document traité par Amazon Textract.`DocumentMetadata`est renvoyé sur chaque page de réponses paginées provenant d'une opération vidéo Amazon Textract.

Type : objet DocumentMetadata

JobStatus

Statut actuel du travail de détection de texte.

Type : Chaîne

Valeurs valides : `IN_PROGRESS | SUCCEEDED | FAILED | PARTIAL_SUCCESS`

NextToken

Si la réponse est tronquée, Amazon Textract renvoie ce jeton. Vous pouvez utiliser ce jeton dans la demande suivante pour récupérer l'ensemble suivant de résultats de détection de texte.

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximale de 255.

Modèle : `.*\S.*`

StatusMessage

Renvoie si la tâche de détection n'a pas pu être exécutée. Contient une explication de l'erreur survenue.

Type : Chaîne

Warnings

Liste des avertissements survenus pendant l'opération d'analyse de documents.

Type : Grappe deWarningobjets

Erreurs

AccessDeniedException

Vous n'êtes pas autorisé à effectuer l'action. Utilisez l'Amazon Resource Name (ARN) d'un utilisateur ou d'un rôle IAM autorisé pour effectuer l'opération.

Code d'état HTTP : 400

InternalServerError

Amazon Textract a rencontré un problème de service. Renouvelez votre appel.

Code d'état HTTP : 500

InvalidJobIdException

Un identifiant de tâche non valide a été transmis àGetDocumentAnalysisou àGetDocumentAnalysis.

Code d'état HTTP : 400

InvalidKMSKeyException

Indique que vous ne disposez pas d'autorisations de déchiffrement avec la clé KMS entrée ou que la clé KMS n'a pas été saisie correctement.

Code d'état HTTP : 400

InvalidParameterException

Un paramètre d'entrée a enfreint une contrainte. Par exemple, dans les opérations synchrone, un`InvalidParameterException`exception se produit lorsque aucune des options`S3Object`ou`Bytes`les valeurs sont fournies dans le`Document`paramètre de demande. Validez votre paramètre avant d'appeler à nouveau l'opération d'API.

Code d'état HTTP : 400

InvalidS3ObjectException

Amazon Textract ne peut pas accéder à l'objet S3 spécifié dans la demande. Pour plus d'informations,Configurer l'accès à Amazon S3Pour plus d'informations sur le dépannage, consultezRésolutions des problèmes liés à Amazon S3

Code d'état HTTP : 400

ProvisionedThroughputExceededException

Le nombre de demandes dépasse votre limite de débit. Si vous avez besoin d'augmenter cette limite, contactez Amazon Textract.

Code d'état HTTP : 400

ThrottlingException

Amazon Textract est temporairement dans l'impossibilité de traiter la demande. Renouvelez votre appel.

Code d'état HTTP : 500

Voir aussi

Pour plus d’informations sur l’utilisation de cette API dans l’un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Interface de ligne de commande AWS
- AWS SDK pour .NET
- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- Kit AWS SDK pour JavaScript
- AWS SDK pour PHP V3
- AWS SDK pour Python
- AWSSDK pour Ruby V3

GetDocumentTextDetection

Obtient les résultats d'une opération asynchrone Amazon Textract qui détecte du texte dans un document. Amazon Textract peut détecter les lignes de texte et les mots qui constituent une ligne de texte.

Vous commencez la détection de texte asynchrone en appelantStartDocumentTextDetection, qui renvoie un identifiant de tâche (`JobId`). Lorsque l'opération de détection de texte est terminée, Amazon Textract publie un état d'achèvement dans la rubrique Amazon Simple Notification Service (Amazon SNS) enregistrée lors de l'appel initial à`StartDocumentTextDetection`. Pour obtenir les résultats de l'opération de détection de texte, vérifiez d'abord que la valeur d'état publiée sur la rubrique Amazon SNS est`SUCCEEDED`. Si c'est le cas, appelez`GetDocumentTextDetection`, et transmettez l'identificateur de la tâche (`JobId`) depuis l'appel initial à`StartDocumentTextDetection`.

`GetDocumentTextDetection`renvoie un tableau deBlockobjets.

Chaque page de document est associée`Block`de type PAGE. Chaque PAGE`Block`objet est le parent de LINE`Block`objets qui représentent les lignes de texte détecté sur une page. UNE LIGNE`Block`objet est un parent pour chaque mot qui constitue la ligne. Les mots sont représentés par`Block`objets de type WORD.

Utilisez le paramètre MaxResults pour limiter le nombre de blocs renvoyés. S'il y a plus de résultats que ceux spécifiés dans`MaxResults`, la valeur de`NextToken`dans la réponse d'opération contient un jeton de pagination permettant d'obtenir l'ensemble suivant de résultats. Pour obtenir la page de résultats suivante, appelez`GetDocumentTextDetection`, et remplissez-le`NextToken`paramètre request avec la valeur du jeton renvoyée par l'appel précédent à`GetDocumentTextDetection`.

Pour de plus amples informations, veuillez consulterDétection de texte.

Syntaxe de la demande

```
{
   "JobId": "string",
   "MaxResults": number,
   "NextToken": "string"
}
```

Paramètres de demande

Cette demande accepte les données suivantes au format JSON.

JobId

Identifiant unique du travail de détection de texte. Le`JobId`est renvoyée par`StartDocumentTextDetection`. UN`JobId`n'est valide que pendant 7 jours.

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximale de 64.

Modèle : `^[a-zA-Z0-9-_]+$`

Obligatoire Oui

MaxResults

Nombre maximal de résultats à renvoyer par appel paginé. La valeur la plus élevée que vous pouvez spécifier est 1 000. Si vous spécifiez une valeur supérieure à 1 000, un maximum de 1 000 résultats est renvoyé. La valeur par défaut est 1,000.

Type : Entier

Plage valide : Valeur minimale est 1.

Obligatoire Non

NextToken

Si la réponse précédente était incomplète (car il y a plus de blocs à récupérer), Amazon Textract renvoie un jeton de pagination dans la réponse. Vous pouvez utiliser ce jeton de pagination pour récupérer l'ensemble suivant de blocs.

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximale de 255.

Modèle : `.*\S.*`

Obligatoire Non

Syntaxe de la réponse

```
{
   "Blocks": [
      {
         "BlockType": "string",
         "ColumnIndex": number,
         "ColumnSpan": number,
         "Confidence": number,
         "EntityTypes": [ "string" ],
         "Geometry": {
            "BoundingBox": {
               "Height": number,
               "Left": number,
               "Top": number,
               "Width": number
            },
            "Polygon": [
               {
                  "X": number,
                  "Y": number
               }
            ]
         },
         "Id": "string",
         "Page": number,
         "Relationships": [
            {
               "Ids": [ "string" ],
               "Type": "string"
            }
         ],
         "RowIndex": number,
         "RowSpan": number,
         "SelectionStatus": "string",
         "Text": "string",
         "TextType": "string"
      }
   ],
   "DetectDocumentTextModelVersion": "string",
   "DocumentMetadata": {
      "Pages": number
   },
```

```
   "JobStatus": "string",
   "NextToken": "string",
   "StatusMessage": "string",
   "Warnings": [
      {
         "ErrorCode": "string",
         "Pages": [ number ]
      }
   ]
}
```

Eléments de réponse

Si l'action aboutit, le service renvoie une réponse HTTP 200.

Les données suivantes sont renvoyées au format JSON par le service.

Blocks

Les résultats de l'opération de détection de texte.

Type : Grappes deBlockobjets

DetectDocumentTextModelVersion

Type : Chaîne

DocumentMetadata

Informations sur un document traité par Amazon Textract.`DocumentMetadata`est renvoyé sur chaque page de réponses paginées provenant d'une opération vidéo Amazon Textract.

Type : objet DocumentMetadata

JobStatus

Statut actuel du travail de détection de texte.

Type : Chaîne

Valeurs valides : `IN_PROGRESS | SUCCEEDED | FAILED | PARTIAL_SUCCESS`

NextToken

Si la réponse est tronquée, Amazon Textract renvoie ce jeton. Vous pouvez utiliser ce jeton dans la demande suivante pour récupérer l'ensemble suivant de résultats de détection de texte.

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximale de 255.

Modèle : `.*\S.*`

StatusMessage

Renvoie si la tâche de détection n'a pas pu être exécutée. Contient une explication de l'erreur survenue.

Type : Chaîne

Warnings

Liste des avertissements survenus pendant l'opération de détection de texte pour le document.

Type : Grappes deWarningobjets

Erreurs

AccessDeniedException

Vous n'êtes pas autorisé à effectuer l'action. Utilisez l'Amazon Resource Name (ARN) d'un utilisateur ou d'un rôle IAM autorisé pour effectuer l'opération.

HTTP Status Code : 400

InternalServerError

Amazon Textract a rencontré un problème de service. Renouvelez votre appel.

HTTP Status Code : 500

InvalidJobIdException

Un identifiant de tâche non valide a été transmis àGetDocumentAnalysisou àGetDocumentAnalysis.

HTTP Status Code : 400

InvalidKMSKeyException

Indique que vous ne disposez pas d'autorisations de déchiffrement avec la clé KMS entrée ou que la clé KMS n'a pas été saisie correctement.

HTTP Status Code : 400

InvalidParameterException

Un paramètre d'entrée a enfreint une contrainte. Par exemple, dans les opérations synchrone, un`InvalidParameterException`exception se produit lorsque aucune des options`S3Object`ou`Bytes`les valeurs sont fournies dans le`Document`paramètre de demande. Validez votre paramètre avant d'appeler à nouveau l'opération d'API.

HTTP Status Code : 400

InvalidS3ObjectException

Amazon Textract ne peut pas accéder à l'objet S3 spécifié dans la demande. Pour plus d'informations,Configurer l'accès à Amazon S3Pour plus d'informations sur le dépannage, consultezRésolutions des problèmes liés à Amazon S3

HTTP Status Code : 400

ProvisionedThroughputExceededException

Le nombre de demandes dépasse votre limite de débit. Si vous avez besoin d'augmenter cette limite, contactez Amazon Textract.

HTTP Status Code : 400

ThrottlingException

Amazon Textract est temporairement dans l'impossibilité de traiter la demande. Renouvelez votre appel.

HTTP Status Code : 500

Voir aussi

Pour plus d’informations sur l’utilisation de cette API dans l’un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Interface de ligne de commande AWS
- AWS SDK pour .NET
- Kit AWS SDK pour C++
- Kit AWS SDK pour Go

- AWSSDK pour Java V2
- Kit AWS SDK pour JavaScript
- AWS SDK pour PHP V3
- AWS SDK pour Python
- AWSSDK pour Ruby V3

GetExpenseAnalysis

Obtient les résultats d'une opération asynchrone Amazon Textract qui analyse les factures et les reçus. Amazon Textract trouve les informations de contact, les articles achetés et le nom du fournisseur, à partir des factures d'entrée et des reçus.

Vous commencez une analyse asynchrone des factures et des reçus en appelantStartExpenseAnalysis, qui renvoie un identifiant de tâche (`JobId`). Une fois l'analyse de la facturation/de la réception terminée, Amazon Textract publie l'état d'achèvement dans la rubrique Amazon Simple Notification Service (Amazon SNS). Cette rubrique doit être enregistrée lors de l'appel initial à`StartExpenseAnalysis`. Pour obtenir les résultats de l'analyse de facturation/réception, assurez-vous tout d'abord que la valeur d'état publiée sur la rubrique Amazon SNS est`SUCCEEDED`. Si c'est le cas, appelez`GetExpenseAnalysis`, et transmettez l'identificateur de la tâche (`JobId`) depuis l'appel initial à`StartExpenseAnalysis`.

Utilisez le paramètre MaxResults pour limiter le nombre de blocs renvoyés. S'il y a plus de résultats que ceux spécifiés dans`MaxResults`, la valeur de`NextToken`dans la réponse d'opération contient un jeton de pagination permettant d'obtenir l'ensemble de résultats suivant. Pour obtenir la page de résultats suivante, appelez`GetExpenseAnalysis`, et remplissez-le`NextToken`paramètre request avec la valeur du jeton renvoyée par l'appel précédent à`GetExpenseAnalysis`.

Pour de plus amples informations, veuillez consulterAnalyse des factures et des reçus.

Syntaxe de la demande

```
{
   "JobId": "string",
   "MaxResults": number,
   "NextToken": "string"
}
```

Paramètres de demande

Cette demande accepte les données suivantes au format JSON.

JobId

Identifiant unique du travail de détection de texte. Le`JobId`est renvoyé par`StartExpenseAnalysis`. UN`JobId`n'est valide que pendant 7 jours.

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximale de 64.

Modèle : `^[a-zA-Z0-9-_]+$`

Obligatoire Oui

MaxResults

Nombre maximal de résultats à renvoyer par appel paginé. La valeur la plus élevée que vous pouvez spécifier est 20. Si vous spécifiez une valeur supérieure à 20, seuls 20 résultats sont renvoyés au maximum. La valeur par défaut est 20.

Type : Entier

Plage valide : Valeur minimale est 1.

Obligatoire Non

NextToken

Si la réponse précédente était incomplète (car il y a plus de blocs à récupérer), Amazon Textract renvoie un jeton de pagination dans la réponse. Vous pouvez utiliser ce jeton de pagination pour récupérer l'ensemble de blocs suivant.

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximale de 255.

Modèle : `.*\S.*`

Obligatoire Non

Syntaxe de la réponse

```
{
   "AnalyzeExpenseModelVersion": "string",
   "DocumentMetadata": {
      "Pages": number
   },
   "ExpenseDocuments": [
      {
         "ExpenseIndex": number,
         "LineItemGroups": [
            {
               "LineItemGroupIndex": number,
```

```
            "LineItems": [
               {
                  "LineItemExpenseFields": [
                     {
                        "LabelDetection": {
                           "Confidence": number,
                           "Geometry": {
                              "BoundingBox": {
                                 "Height": number,
                                 "Left": number,
                                 "Top": number,
                                 "Width": number
                              },
                              "Polygon": [
                                 {
                                    "X": number,
                                    "Y": number
                                 }
                              ]
                           },
                           "Text": "string"
                        },
                        "PageNumber": number,
                        "Type": {
                           "Confidence": number,
                           "Text": "string"
                        },
                        "ValueDetection": {
                           "Confidence": number,
                           "Geometry": {
                              "BoundingBox": {
                                 "Height": number,
                                 "Left": number,
                                 "Top": number,
                                 "Width": number
                              },
                              "Polygon": [
                                 {
                                    "X": number,
                                    "Y": number
                                 }
                              ]
                           },
                           "Text": "string"
```

```
                        }
                     }
                  ]
               }
            ]
         }
      ],
      "SummaryFields": [
         {
            "LabelDetection": {
               "Confidence": number,
               "Geometry": {
                  "BoundingBox": {
                     "Height": number,
                     "Left": number,
                     "Top": number,
                     "Width": number
                  },
                  "Polygon": [
                     {
                        "X": number,
                        "Y": number
                     }
                  ]
               },
               "Text": "string"
            },
            "PageNumber": number,
            "Type": {
               "Confidence": number,
               "Text": "string"
            },
            "ValueDetection": {
               "Confidence": number,
               "Geometry": {
                  "BoundingBox": {
                     "Height": number,
                     "Left": number,
                     "Top": number,
                     "Width": number
                  },
                  "Polygon": [
                     {
                        "X": number,
                     }
                  ]
               },
               "Text": "string"
            },
            "PageNumber": number,
            "Type": {
               "Confidence": number,
               "Text": "string"
            },
            "ValueDetection": {
               "Confidence": number,
               "Geometry": {
                  "BoundingBox": {
                     "Height": number,
                     "Left": number,
                     "Top": number,
                     "Width": number
                  },
                  "Polygon": [
                     {
                        "X": number,
```

```
                        "Y": number
                     }
                  ]
               },
               "Text": "string"
            }
         }
      ]
   }
   ],
   "JobStatus": "string",
   "NextToken": "string",
   "StatusMessage": "string",
   "Warnings": [
      {
         "ErrorCode": "string",
         "Pages": [ number ]
      }
   ]
}
```

Eléments de réponse

Si l'action aboutit, le service renvoie une réponse HTTP 200.

Les données suivantes sont renvoyées au format JSON par le service.

AnalyzeExpenseModelVersion

La version modèle actuelle d'AnalyzeExpense.

Type : Chaîne

DocumentMetadata

Informations sur un document traité par Amazon Textract.`DocumentMetadata`est renvoyé sur chaque page de réponses paginées provenant d'une opération Amazon Textract.

Type : objet DocumentMetadata

ExpenseDocuments

Les dépenses détectées par Amazon Textract.

Type : Tableau deExpenseDocumentobjets

JobStatus

Statut actuel de la tâche de détection de texte.

Type : Chaîne

Valeurs valides : `IN_PROGRESS | SUCCEEDED | FAILED | PARTIAL_SUCCESS`

NextToken

Si la réponse est tronquée, Amazon Textract renvoie ce jeton. Vous pouvez utiliser ce jeton dans la demande suivante pour récupérer l'ensemble suivant de résultats de détection de texte.

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximale de 255.

Modèle : `.*\S.*`

StatusMessage

Renvoie si le travail de détection n'a pas pu être exécuté. Contient une explication de l'erreur survenue.

Type : Chaîne

Warnings

Liste des avertissements survenus pendant l'opération de détection de texte pour le document.

Type : Tableau deWarningobjets

Erreurs

AccessDeniedException

Vous n'êtes pas autorisé à effectuer l'action. Utilisez l'Amazon Resource Name (ARN) d'un utilisateur ou d'un rôle IAM autorisé pour effectuer l'opération.

HTTP Status Code : 400

InternalServerError

Amazon Textract a rencontré un problème de service. Renouvelez votre appel.

HTTP Status Code : 500

InvalidJobIdException

Un identifiant de tâche non valide a été transmis àGetDocumentAnalysisou àGetDocumentAnalysis.

HTTP Status Code : 400

InvalidKMSKeyException

Indique que vous ne disposez pas d'autorisations de déchiffrement avec la clé KMS entrée ou que la clé KMS n'a pas été saisie correctement.

HTTP Status Code : 400

InvalidParameterException

Un paramètre d'entrée a enfreint une contrainte. Par exemple, dans les opérations synchrone, un`InvalidParameterException`exception se produit lorsque aucune des options`S3Object`ou`Bytes`les valeurs sont fournies dans le`Document`paramètre de demande. Validez votre paramètre avant d'appeler à nouveau l'opération d'API.

HTTP Status Code : 400

InvalidS3ObjectException

Amazon Textract n'est pas en mesure d'accéder à l'objet S3 spécifié dans la demande. Pour plus d'informations,Configurer l'accès à Amazon S3Pour plus d'informations sur le dépannage, consultezRésolutions des problèmes liés à Amazon S3

HTTP Status Code : 400

ProvisionedThroughputExceededException

Le nombre de demandes dépasse votre limite de débit. Si vous avez besoin d'augmenter cette limite, contactez Amazon Textract.

HTTP Status Code : 400

ThrottlingException

Amazon Textract est temporairement dans l'impossibilité de traiter la demande. Renouvelez votre appel.

HTTP Status Code : 500

Voir aussi

Pour plus d'informations sur l'utilisation de cette API dans l'un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Interface de ligne de commande AWS
- AWS SDK pour .NET
- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- Kit AWS SDK pour JavaScript
- AWS SDK pour PHP V3
- AWS SDK pour Python
- AWSSDK pour Ruby V3

StartDocumentAnalysis

Lance l'analyse asynchrone d'un document d'entrée pour les relations entre des éléments détectés tels que des paires de clé-valeur, des tableaux et des éléments de sélection.

`StartDocumentAnalysis`peut analyser du texte dans des documents au format JPEG, PNG, TIFF et PDF. Les documents sont stockés dans un compartiment Amazon S3. UtiliserDocumentLocationpour spécifier le nom du compartiment et le nom du fichier du document.

`StartDocumentAnalysis`renvoie un identifiant de tâche (`JobId`) que vous utilisez pour obtenir les résultats de l'opération. Lorsque l'analyse de texte est terminée, Amazon Textract publie un état d'achèvement dans la rubrique Amazon Simple Notification Service (Amazon SNS) que vous spécifiez dans`NotificationChannel`. Pour obtenir les résultats de l'opération d'analyse de texte, vérifiez d'abord que la valeur d'état publiée sur la rubrique Amazon SNS est`SUCCEEDED`. Si c'est le cas, appelezGetDocumentAnalysis, et transmettez l'identificateur de la tâche (`JobId`) depuis l'appel initial à`StartDocumentAnalysis`.

Pour de plus amples informations, veuillez consulterAnalyse de texte de document.

Syntaxe de la demande

```
{
   "ClientRequestToken": "string",
   "DocumentLocation": {
      "S3Object": {
         "Bucket": "string",
         "Name": "string",
         "Version": "string"
      }
   },
   "FeatureTypes": [ "string" ],
   "JobTag": "string",
   "KMSKeyId": "string",
   "NotificationChannel": {
      "RoleArn": "string",
      "SNSTopicArn": "string"
   },
   "OutputConfig": {
      "S3Bucket": "string",
      "S3Prefix": "string"
   }
```

```
}
```

Paramètres de demande

Cette demande accepte les données suivantes au format JSON.

ClientRequestToken

Le jeton idempotent que vous utilisez pour identifier la demande de démarrage. Si vous utilisez le même jeton avec plusieurs`StartDocumentAnalysis`demandes, les mêmes`JobId`est renvoyé. Utiliser`ClientRequestToken`pour éviter que le même travail ne soit lancé accidentellement plus d'une fois. Pour de plus amples informations, veuillez consulterAppel d'opérations asynchrones Amazon Textract.

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximale de 64.

Modèle : `^[a-zA-Z0-9-_]+$`

Obligatoire Non

DocumentLocation

Emplacement du document à traiter.

Type : objet DocumentLocation

Obligatoire Oui

FeatureTypes

Liste des types d'analyses à effectuer. Ajoutez TABLES à la liste pour renvoyer des informations sur les tables détectées dans le document d'entrée. Ajoutez FORMS pour renvoyer les données de formulaire détectées. Pour effectuer les deux types d'analyse, ajoutez TABLES et FORMS à`FeatureTypes`. Toutes les lignes et tous les mots détectés dans le document sont inclus dans la réponse (y compris le texte qui n'est pas lié à la valeur de`FeatureTypes`).

Type : Tableau de chaînes

Valeurs valides : `TABLES | FORMS`

Obligatoire Oui

JobTag

Identifiant que vous spécifiez inclus dans la notification de fin publiée sur la rubrique Amazon SNS. Par exemple, vous pouvez utiliserJobTagpour identifier le type de document auquel correspond la notification d'achèvement (tel qu'un formulaire fiscal ou un reçu).

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximale de 64.

Modèle : `[a-zA-Z0-9_.\-:]+`

Obligatoire Non

KMSKeyId

Clé KMS utilisée pour chiffrer les résultats de l'inférence. Cela peut être au format Key ID ou Key Alias. Lorsqu'une clé KMS est fournie, la clé KMS est utilisée pour le chiffrement côté serveur des objets du compartiment client. Lorsque ce paramètre n'est pas activé, le résultat est chiffré côté serveur, à l'aide de SSE-S3.

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximale de 2048.

Modèle : `^[A-Za-z0-9][A-Za-z0-9:_/+=,@.-]{0,2048}$`

Obligatoire Non

NotificationChannel

L'ARN de rubrique Amazon SNS sur lequel vous souhaitez qu'Amazon Textract publie l'état d'achèvement de l'opération.

Type : objet NotificationChannel

Obligatoire Non

OutputConfig

Définit si la sortie sera envoyée à un compartiment défini par le client. Par défaut, Amazon Textract enregistre les résultats en interne pour que l'opération GetDocumentAnalysis puisse y accéder.

Type : objet OutputConfig

Obligatoire Non

Syntaxe de la réponse

```
{
   "JobId": "string"
}
```

Eléments de réponse

Si l'action aboutit, le service renvoie une réponse HTTP 200.

Les données suivantes sont renvoyées au format JSON par le service.

JobId

Identificateur du document de tâche de détection de texte de document. Utiliser`JobId`pour identifier la tâche lors d'un appel ultérieur à`GetDocumentAnalysis`. UN`JobId`n'est valide que pendant 7 jours.

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximale de 64.

Modèle : `^[a-zA-Z0-9-_]+$`

Erreurs

AccessDeniedException

Vous n'êtes pas autorisé à effectuer l'action. Utilisez l'Amazon Resource Name (ARN) d'un utilisateur ou d'un rôle IAM autorisé pour effectuer l'opération.

Code d'état HTTP : 400

BadDocumentException

Amazon Textract n'est pas en mesure de lire le document. Pour plus d'informations sur les limites de documents dans Amazon Textract, voirLimites strictes dans Amazon Textract.

Code d'état HTTP : 400

DocumentTooLargeException

Le document ne peut pas être traité car il est trop volumineux. Taille maximale du document pour les opérations synchrone 10 Mo. La taille maximale du document pour les opérations asynchrones est de 500 Mo pour les fichiers PDF.

Code d'état HTTP : 400

IdempotentParameterMismatchException

UNClientRequestTokenLe paramètre d'entrée a été réutilisé avec une opération, mais au moins un des autres paramètres d'entrée est différent de l'appel précédent à l'opération.

Code d'état HTTP : 400

InternalServerError

Amazon Textract a rencontré un problème de service. Renouvelez votre appel.

Code d'état HTTP : 500

InvalidKMSKeyException

Indique que vous ne disposez pas d'autorisations de déchiffrement avec la clé KMS entrée ou que la clé KMS n'a pas été saisie correctement.

Code d'état HTTP : 400

InvalidParameterException

Un paramètre d'entrée a enfreint une contrainte. Par exemple, dans les opérations synchrone, un`InvalidParameterException`exception se produit lorsque aucune des options`S3Object`ou`Bytes`les valeurs sont fournies dans le`Document`paramètre de demande. Validez votre paramètre avant d'appeler à nouveau l'opération d'API.

Code d'état HTTP : 400

InvalidS3ObjectException

Amazon Textract n'est pas en mesure d'accéder à l'objet S3 spécifié dans la demande. Pour plus d'informations,Configurer l'accès à Amazon S3Pour plus d'informations sur le dépannage, consultezRésolutions des problèmes liés à Amazon S3

Code d'état HTTP : 400

LimitExceededException

Une limite de service Amazon Textract a été dépassée. Par exemple, si vous démarrez un trop grand nombre de tâches asynchrones simultanément, des appels pour démarrer des opérations (`StartDocumentTextDetection`, par exemple) génère une exception LimitExceededException (code de statut HTTP : 400) jusqu'à ce que le nombre de tâches exécutées simultanément soit inférieur à la limite de service Amazon Textract.

Code d'état HTTP : 400

ProvisionedThroughputExceededException

Le nombre de demandes dépasse votre limite de débit. Si vous avez besoin d'augmenter cette limite, contactez Amazon Textract.

Code d'état HTTP : 400

ThrottlingException

Amazon Textract est temporairement dans l'impossibilité de traiter la demande. Renouvelez votre appel.

Code d'état HTTP : 500

UnsupportedDocumentException

Le format du document d'entrée n'est pas pris en charge. Les documents utilisés pour les opérations peuvent être au format PNG, JPEG, PDF ou TIFF.

Code d'état HTTP : 400

Voir aussi

Pour plus d’informations sur l’utilisation de cette API dans l’un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Interface de ligne de commande AWS
- AWS SDK pour .NET
- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2

- Kit AWS SDK pour JavaScript
- AWS SDK pour PHP V3
- AWS SDK pour Python
- AWSSDK pour Ruby V3

StartDocumentTextDetection

Démarre la détection asynchrone de texte dans un document. Amazon Textract peut détecter les lignes de texte et les mots qui constituent une ligne de texte.

`StartDocumentTextDetection`peut analyser du texte dans des documents au format JPEG, PNG, TIFF et PDF. Les documents sont stockés dans un compartiment Amazon S3. UtiliserDocumentLocationpour spécifier le nom du compartiment et le nom du fichier du document.

`StartTextDetection`renvoie un identifiant de tâche (`JobId`) que vous utilisez pour obtenir les résultats de l'opération. Lorsque la détection de texte est terminée, Amazon Textract publie un état d'achèvement dans la rubrique Amazon Simple Notification Service (Amazon SNS) que vous avez spécifiée dans`NotificationChannel`. Pour obtenir les résultats de l'opération de détection de texte, vérifiez d'abord que la valeur d'état publiée sur la rubrique Amazon SNS est`SUCCEEDED`. Si c'est le cas, appelezGetDocumentTextDetection, et transmettez l'identificateur de la tâche (`JobId`) depuis l'appel initial à`StartDocumentTextDetection`.

Pour de plus amples informations, veuillez consulterDétection de texte.

Syntaxe de la demande

```
{
   "ClientRequestToken": "string",
   "DocumentLocation": {
      "S3Object": {
         "Bucket": "string",
         "Name": "string",
         "Version": "string"
      }
   },
   "JobTag": "string",
   "KMSKeyId": "string",
   "NotificationChannel": {
      "RoleArn": "string",
      "SNSTopicArn": "string"
   },
   "OutputConfig": {
      "S3Bucket": "string",
      "S3Prefix": "string"
   }
}
```

Paramètres de demande

Cette demande accepte les données suivantes au format JSON.

ClientRequestToken

Jeton idempotent utilisé pour identifier la demande de démarrage. Si vous utilisez le même jeton avec plusieurs`StartDocumentTextDetection`demandes, les mêmes`JobId`est renvoyé. Utiliser`ClientRequestToken`pour éviter que le même travail ne soit lancé accidentellement plus d'une fois. Pour de plus amples informations, veuillez consulterAppel d'opérations asynchrones Amazon Textract.

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximale de 64.

Modèle : `^[a-zA-Z0-9-_]+$`

Obligatoire Non

DocumentLocation

Emplacement du document à traiter.

Type : objet DocumentLocation

Obligatoire Oui

JobTag

Identifiant que vous spécifiez inclus dans la notification de fin publiée sur la rubrique Amazon SNS. Par exemple, vous pouvez utiliser`JobTag`pour identifier le type de document auquel correspond la notification d'achèvement (tel qu'un formulaire fiscal ou un reçu).

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximale de 64.

Modèle : `[a-zA-Z0-9_.\-:]+`

Obligatoire Non

KMSKeyId

La clé KMS utilisée pour chiffrer les résultats de l'inférence. Cela peut être au format Key ID ou Key Alias. Lorsqu'une clé KMS est fournie, la clé KMS est utilisée pour le chiffrement côté serveur

des objets du compartiment client. Lorsque ce paramètre n'est pas activé, le résultat est chiffré côté serveur, à l'aide de SSE-S3.

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximale de 2048.

Modèle : `^[A-Za-z0-9][A-Za-z0-9:_/+=,@.-]{0,2048}$`

Obligatoire Non

NotificationChannel

L'ARN de rubrique Amazon SNS sur lequel vous souhaitez qu'Amazon Textract publie l'état d'achèvement de l'opération.

Type : objet NotificationChannel

Obligatoire Non

OutputConfig

Définit si la sortie sera envoyée à un compartiment défini par le client. Par défaut, Amazon Textract enregistre les résultats en interne pour pouvoir être consultés avec l'opération GetDocumentTextDetection.

Type : objet OutputConfig

Obligatoire Non

Syntaxe de la réponse

```
{
   "JobId": "string"
}
```

Eléments de réponse

Si l'action aboutit, le service renvoie une réponse HTTP 200.

Les données suivantes sont renvoyées au format JSON par le service.

JobId

Identificateur de la tâche de détection de texte pour le document. UtiliserJobIdpour identifier la tâche lors d'un appel ultérieur àGetDocumentTextDetection. UNJobIdn'est valide que pendant 7 jours.

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximale de 64.

Modèle : ^[a-zA-Z0-9-_]+$

Erreurs

AccessDeniedException

Vous n'êtes pas autorisé à effectuer l'action. Utilisez l'Amazon Resource Name (ARN) d'un utilisateur ou d'un rôle IAM autorisé pour effectuer l'opération.

HTTP Status Code : 400

BadDocumentException

Amazon Textract n'est pas en mesure de lire le document. Pour plus d'informations sur les limites de documents dans Amazon Textract, voirLimites strictes dans Amazon Textract.

HTTP Status Code : 400

DocumentTooLargeException

Le document ne peut pas être traité car il est trop volumineux. Taille maximale du document pour les opérations synchrone 10 Mo. La taille maximale du document pour les opérations asynchrones est de 500 Mo pour les fichiers PDF.

HTTP Status Code : 400

IdempotentParameterMismatchException

UNClientRequestTokenLe paramètre d'entrée a été réutilisé avec une opération, mais au moins un des autres paramètres d'entrée est différent de l'appel précédent à l'opération.

HTTP Status Code : 400

InternalServerError

Amazon Textract a rencontré un problème de service. Renouvelez votre appel.

HTTP Status Code : 500

InvalidKMSKeyException

Indique que vous ne disposez pas d'autorisations de déchiffrement avec la clé KMS entrée ou que la clé KMS n'a pas été saisie correctement.

HTTP Status Code : 400

InvalidParameterException

Un paramètre d'entrée a enfreint une contrainte. Par exemple, dans les opérations synchrone, un`InvalidParameterException`exception se produit lorsque aucune des options`S3Object`ou`Bytes`les valeurs sont fournies dans le`Document`paramètre de demande. Validez votre paramètre avant d'appeler à nouveau l'opération d'API.

HTTP Status Code : 400

InvalidS3ObjectException

Amazon Textract n'est pas en mesure d'accéder à l'objet S3 spécifié dans la demande. Pour plus d'informations,Configuration de l'accès à Amazon S3Pour plus d'informations sur le dépannage, consultezRésolutions des problèmes liés à Amazon S3

HTTP Status Code : 400

LimitExceededException

Une limite de service Amazon Textract a été dépassée. Par exemple, si vous démarrez un trop grand nombre de tâches asynchrones simultanément, des appels pour démarrer des opérations (`StartDocumentTextDetection`, par exemple) génère une exception LimitExceededException (code de statut HTTP : 400) jusqu'à ce que le nombre de tâches exécutées simultanément soit inférieur à la limite de service Amazon Textract.

HTTP Status Code : 400

ProvisionedThroughputExceededException

Le nombre de demandes dépasse votre limite de débit. Si vous avez besoin d'augmenter cette limite, contactez Amazon Textract.

HTTP Status Code : 400

ThrottlingException

Amazon Textract est temporairement dans l'impossibilité de traiter la demande. Renouvelez votre appel.

HTTP Status Code : 500

UnsupportedDocumentException

Le format du document d'entrée n'est pas pris en charge. Les documents utilisés pour les opérations peuvent être au format PNG, JPEG, PDF ou TIFF.

HTTP Status Code : 400

Voir aussi

Pour plus d’informations sur l’utilisation de cette API dans l’un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Interface de ligne de commande AWS
- AWS SDK pour .NET
- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- Kit AWS SDK pour JavaScript
- AWS SDK pour PHP V3
- AWS SDK pour Python
- AWSSDK pour Ruby V3

StartExpenseAnalysis

Lance l'analyse asynchrone des factures ou des reçus pour des données telles que les informations de contact, les articles achetés et les noms des fournisseurs.

`StartExpenseAnalysis`peut analyser du texte dans des documents au format JPEG, PNG et PDF. Les documents doivent être stockés dans un compartiment Amazon S3. Utilisation de l'DocumentLocationpour spécifier le nom de votre compartiment S3 et le nom du document dans ce compartiment S3.

`StartExpenseAnalysis`renvoie un identifiant de tâche (`JobId`) que vous allez fournir à`GetExpenseAnalysis`pour récupérer les résultats de l'opération. Lorsque l'analyse des factures et des reçus d'entrée est terminée, Amazon Textract publie un état d'achèvement dans la rubrique Amazon Simple Notification Service (Amazon SNS) que vous fournissez au`NotificationChannel`. Pour obtenir les résultats de l'opération d'analyse des factures et des reçus, assurez-vous que la valeur d'état publiée sur la rubrique Amazon SNS est`SUCCEEDED`. Si c'est le cas, appelezGetExpenseAnalysis, et transmettez l'identificateur de la tâche (`JobId`) qui a été renvoyé par votre appel à`StartExpenseAnalysis`.

Pour de plus amples informations, veuillez consulterAnalyse des factures et des reçus.

Syntaxe de la demande

```
{
   "ClientRequestToken": "string",
   "DocumentLocation": {
      "S3Object": {
         "Bucket": "string",
         "Name": "string",
         "Version": "string"
      }
   },
   "JobTag": "string",
   "KMSKeyId": "string",
   "NotificationChannel": {
      "RoleArn": "string",
      "SNSTopicArn": "string"
   },
   "OutputConfig": {
      "S3Bucket": "string",
      "S3Prefix": "string"
```

```
    }
}
```

Paramètres de demande

Cette demande accepte les données suivantes au format JSON.

ClientRequestToken

Jeton idempotent utilisé pour identifier la demande de démarrage. Si vous utilisez le même jeton avec plusieurs`StartDocumentTextDetection`demandes, les mêmes`JobId`est renvoyé. Utiliser`ClientRequestToken`pour éviter que le même travail ne soit lancé accidentellement plus d'une fois. Pour de plus amples informations, veuillez consulterAppel d'opérations asynchrones Amazon Textract

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximale de 64.

Modèle : `^[a-zA-Z0-9-_]+$`

Obligatoire Non

DocumentLocation

Emplacement du document à traiter.

Type : objet DocumentLocation

Obligatoire Oui

JobTag

Identifiant que vous spécifiez inclus dans la notification de fin publiée sur la rubrique Amazon SNS. Par exemple, vous pouvez utiliser`JobTag`pour identifier le type de document auquel correspond la notification d'achèvement (tel qu'un formulaire fiscal ou un reçu).

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximale de 64.

Modèle : `[a-zA-Z0-9_.\-:]+`

Obligatoire Non

KMSKeyId

Clé KMS utilisée pour chiffrer les résultats de l'inférence. Cela peut être au format Key ID ou Key Alias. Lorsqu'une clé KMS est fournie, la clé KMS est utilisée pour le chiffrement côté serveur des objets du compartiment client. Lorsque ce paramètre n'est pas activé, le résultat est chiffré côté serveur, à l'aide de SSE-S3.

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximale de 2048.

Modèle : `^[A-Za-z0-9][A-Za-z0-9:_/+=,@.-]{0,2048}$`

Obligatoire Non

NotificationChannel

L'ARN de rubrique Amazon SNS sur lequel vous souhaitez qu'Amazon Textract publie l'état d'achèvement de l'opération.

Type : objet NotificationChannel

Obligatoire Non

OutputConfig

Définit si la sortie sera envoyée à un compartiment défini par le client. Par défaut, Amazon Textract enregistre les résultats en interne pour qu'ils soient accessibles par le`GetExpenseAnalysis`.

Type : objet OutputConfig

Obligatoire Non

Syntaxe de la réponse

```
{
   "JobId": "string"
}
```

Eléments de réponse

Si l'action aboutit, le service renvoie une réponse HTTP 200.

Les données suivantes sont renvoyées au format JSON par le service.

JobId

Identifiant unique de la tâche de détection de texte. Le`JobId`est renvoyé par`StartExpenseAnalysis`. UN`JobId`n'est valide que pendant 7 jours.

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximale de 64.

Modèle : `^[a-zA-Z0-9-_]+$`

Erreurs

AccessDeniedException

Vous n'êtes pas autorisé à effectuer l'action. Utilisez l'Amazon Resource Name (ARN) d'un utilisateur ou d'un rôle IAM autorisé pour effectuer l'opération.

HTTP Status Code : 400

BadDocumentException

Amazon Textract n'est pas en mesure de lire le document. Pour plus d'informations sur les limites de documents dans Amazon Textract, voirLimites strictes dans Amazon Textract.

HTTP Status Code : 400

DocumentTooLargeException

Le document ne peut pas être traité car il est trop volumineux. Taille maximale du document pour les opérations synchrone 10 Mo. La taille maximale du document pour les opérations asynchrones est de 500 Mo pour les fichiers PDF.

HTTP Status Code : 400

IdempotentParameterMismatchException

UN`ClientRequestToken`Le paramètre d'entrée a été réutilisé avec une opération, mais au moins un des autres paramètres d'entrée est différent de l'appel précédent à l'opération.

HTTP Status Code : 400

InternalServerError

Amazon Textract a rencontré un problème de service. Renouvelez votre appel.

HTTP Status Code : 500

InvalidKMSKeyException

Indique que vous ne disposez pas d'autorisations de déchiffrement avec la clé KMS entrée ou que la clé KMS n'a pas été saisie correctement.

HTTP Status Code : 400

InvalidParameterException

Un paramètre d'entrée a enfreint une contrainte. Par exemple, dans les opérations synchrone, un`InvalidParameterException`exception se produit lorsque aucune des options`S3Object`ou`Bytes`les valeurs sont fournies dans le`Document`paramètre de demande. Validez votre paramètre avant d'appeler à nouveau l'opération d'API.

HTTP Status Code : 400

InvalidS3ObjectException

Amazon Textract n'est pas en mesure d'accéder à l'objet S3 spécifié dans la demande. Pour plus d'informations,Configuration de l'accès à Amazon S3Pour plus d'informations sur le dépannage, consultezRésolutions des problèmes liés à Amazon S3

HTTP Status Code : 400

LimitExceededException

Une limite de service Amazon Textract a été dépassée. Par exemple, si vous démarrez un trop grand nombre de tâches asynchrones simultanément, des appels pour démarrer des opérations (`StartDocumentTextDetection`, par exemple) génère une exception LimitExceededException (code de statut HTTP : 400) jusqu'à ce que le nombre de tâches exécutées simultanément soit inférieur à la limite de service Amazon Textract.

HTTP Status Code : 400

ProvisionedThroughputExceededException

Le nombre de demandes dépasse votre limite de débit. Si vous avez besoin d'augmenter cette limite, contactez Amazon Textract.

HTTP Status Code : 400

ThrottlingException

Amazon Textract est temporairement dans l'impossibilité de traiter la demande. Renouvelez votre appel.

HTTP Status Code : 500

UnsupportedDocumentException

Le format du document d'entrée n'est pas pris en charge. Les documents utilisés pour les opérations peuvent être au format PNG, JPEG, PDF ou TIFF.

HTTP Status Code : 400

Voir aussi

Pour plus d'informations sur l'utilisation de cette API dans l'un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Interface de ligne de commande AWS
- AWS SDK pour .NET
- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- Kit AWS SDK pour JavaScript
- AWS SDK pour PHP V3
- AWS SDK pour Python
- AWSSDK pour Ruby V3

Types de données

Les types de données suivants sont pris en charge :

- AnalyzeIDDetections
- Block
- BoundingBox

- Document
- DocumentLocation
- DocumentMetadata
- ExpenseDetection
- ExpenseDocument
- ExpenseField
- ExpenseType
- Geometry
- HumanLoopActivationOutput
- HumanLoopConfig
- HumanLoopDataAttributes
- IdentityDocument
- IdentityDocumentField
- LineItemFields
- LineItemGroup
- NormalizedValue
- NotificationChannel
- OutputConfig
- Point
- Relationship
- S3Object
- Warning

AnalyzeIDDetections

Utilisé pour contenir les informations détectées par une opération AnalyzeID.

Table des matières

Confidence

Le score de confiance du texte détecté.

Type : Float

Plage valide : La valeur minimale est fixée à 0. Valeur maximale fixée à 100.

Obligatoire Non

NormalizedValue

Retourné uniquement pour les dates, renvoie le type de valeur détectée et la date écrite de manière plus lisible par machine.

Type : objet NormalizedValue

Obligatoire Non

Text

Texte du champ normalisé ou de la valeur associée.

Type : Chaîne

Obligatoire Oui

Voir aussi

Pour plus d'informations sur l'utilisation de cette API dans l'un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- AWSSDK pour Ruby V3

Block

UN`Block`représente les éléments reconnus dans un document au sein d'un groupe de pixels proches les uns des autres. Les informations renvoyées dans un`Block`dépend du type d'opération. Dans la détection de texte pour les documents (par exemple)DetectDocumentText), vous obtenez des informations sur les mots et les lignes de texte détectés. Dans l'analyse de texte (par exemple)AnalyzeDocument), vous pouvez également obtenir des informations sur les champs, les tables et les éléments de sélection détectés dans le document.

Tableau d'éléments`Block`objets sont renvoyés par des opérations synchrone et asynchrone. Dans les opérations synchrone, telles queDetectDocumentText, tableau de`Block`object représente l'ensemble des résultats. Dans les opérations asynchrones, telles queGetDocumentAnalysis, la baie est renvoyée sur une ou plusieurs réponses.

Pour de plus amples informations, veuillez consulterFonctionnement d'Amazon Textract.

Table des matières

BlockType

Type d'élément de texte reconnu. Dans les opérations de détection de texte, les types suivants sont renvoyés :

- PAGE- Contient une liste des lignes`Block`objets détectés sur une page de document.
- MOT- Un mot détecté sur une page de document. Un mot est constitué d'un ou plusieurs caractères latins de base ISO non séparés par des espaces.
- LIGNE- Une chaîne de mots contigus délimités par des tabulations qui sont détectés sur une page de document.

Dans les opérations d'analyse de texte, les types suivants sont renvoyés :

- PAGE- Contient une liste des enfants`Block`objets détectés sur une page de document.
- KEY_VALUE_SET- Stocke la CLÉ et la VALEUR`Block`objets pour le texte lié détecté sur une page de document. Utilisation de l'`EntityType`pour déterminer si un objet KEY_VALUE_SET est une clé`Block`objet ou VALUE`Block`objet.
- MOT- Un mot détecté sur une page de document. Un mot est constitué d'un ou plusieurs caractères latins de base ISO non séparés par des espaces.
- LIGNE- Une chaîne de mots contigus délimités par des tabulations qui sont détectés sur une page de document.

- TABLE- Tableau détecté sur une page de document. Un tableau est constitué d'informations basées sur une grille comportant au moins deux lignes ou colonnes, avec une plage de cellules d'une ligne et d'une colonne chacune.
- CELLULE- Une cellule dans une table détectée. La cellule est le parent du bloc contenant le texte de la cellule.
- SELECTION_ELEMENT- Un élément de sélection tel qu'un bouton d'option (bouton radio) ou une case à cocher détectée sur une page de document. Utilisez la valeur de`SelectionStatus`pour déterminer le statut de l'élément de sélection.

Type : Chaîne

Valeurs valides : `KEY_VALUE_SET | PAGE | LINE | WORD | TABLE | CELL | SELECTION_ELEMENT`

Obligatoire Non

ColumnIndex

Colonne dans laquelle une cellule de tableau apparaît. La position de la première colonne est 1.`ColumnIndex`n'est pas renvoyé par`DetectDocumentText`et`GetDocumentTextDetection`.

Type : Entier

Plage valide : La valeur minimale est 0.

Obligatoire Non

ColumnSpan

Nombre de colonnes couvrant une cellule de tableau. Actuellement, cette valeur est toujours 1, même si le nombre de colonnes échelonnées est supérieur à 1.`ColumnSpan`n'est pas renvoyé par`DetectDocumentText`et`GetDocumentTextDetection`.

Type : Entier

Plage valide : La valeur minimale est 0.

Obligatoire Non

Confidence

Le score de confiance d'Amazon Textract dans la précision du texte reconnu et la précision de la géométrie pointe autour du texte reconnu.

Type : Float

Plage valide : La valeur minimale est 0. Valeur maximale fixée à 100.

Obligatoire Non

EntityTypes

Type d'entité. Les éléments suivants peuvent être renvoyés :

- CLÉ- Identifiant d'un champ du document.
- VALEUR- Le texte du champ.

`EntityTypes`n'est pas renvoyé par`DetectDocumentTextetGetDocumentTextDetection`.

Type : Tableau de chaînes

Valeurs valides : `KEY | VALUE`

Obligatoire Non

Geometry

L'emplacement du texte reconnu sur l'image. Il comprend un cadre de sélection grossier aligné sur l'axe qui entoure le texte, et un polygone à grain fin pour des informations spatiales plus précises.

Type : objet Geometry

Obligatoire Non

Id

Identificateur du texte reconnu. L'identifiant n'est unique que pour une seule opération.

Type : Chaîne

Modèle : `.*\S.*`

Obligatoire Non

Page

Page sur laquelle un bloc a été détecté.`Page`est renvoyé par des opérations asynchrones. Les valeurs de page supérieures à 1 ne sont renvoyées que pour les documents multipages au format PDF ou TIFF. Une image numérisée (JPEG/PNG), même si elle contient plusieurs pages de

document, est considérée comme un document d'une seule page. Pour`Page`est toujours 1. Les opérations synchrone ne reviennent pas`Page`car chaque document d'entrée est considéré comme un document d'une seule page.

Type : Entier

Plage valide : La valeur minimale est 0.

Obligatoire Non

Relationships

Liste des blocs enfants du bloc actuel. Par exemple, un objet LINE comporte des blocs enfants pour chaque bloc WORD faisant partie de la ligne de texte. Il n'y a pas d'objets Relationship dans la liste pour les relations qui n'existent pas, par exemple lorsque le bloc actuel ne comporte pas de blocs enfants. La taille de la liste peut être la suivante :

- 0 - Le bloc ne comporte pas de blocs enfants.
- 1 - Le bloc comporte des blocs enfants.

Type : Tableau de[Relationship](#)objets

Obligatoire Non

RowIndex

Ligne dans laquelle se trouve une cellule de tableau. La position de la première ligne est 1.`RowIndex`n'est pas renvoyé par`DetectDocumentText`et`GetDocumentTextDetection`.

Type : Entier

Plage valide : La valeur minimale est 0.

Obligatoire Non

RowSpan

Nombre de lignes couvrant une cellule de tableau. Actuellement, cette valeur est toujours 1, même si le nombre de lignes étendues est supérieur à 1.`RowSpan`n'est pas renvoyé par`DetectDocumentText`et`GetDocumentTextDetection`.

Type : Entier

Plage valide : La valeur minimale est 0.

Obligatoire Non

SelectionStatus

Statut de sélection d'un élément de sélection, tel qu'un bouton d'option ou une case à cocher.

Type : Chaîne

Valeurs valides : `SELECTED | NOT_SELECTED`

Obligatoire Non

Text

Le mot ou la ligne de texte reconnu par Amazon Textract.

Type : Chaîne

Obligatoire Non

TextType

Type de texte détecté par Amazon Textract. Peut vérifier s'il y a du texte manuscrit et du texte imprimé.

Type : Chaîne

Valeurs valides : `HANDWRITING | PRINTED`

Obligatoire Non

Voir aussi

Pour plus d'informations sur l'utilisation de cette API dans l'un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- AWSSDK pour Ruby V3

BoundingBox

Le cadre de sélection autour de la page, du texte, de la paire clé-valeur, du tableau, de la cellule de tableau ou de l'élément de sélection détectés sur une page de document. Le`left`(coordonnée X) et`top`(coordonnée y) représentent les coordonnée représentant les côtés supérieur et gauche du cadre de délimitation. Notez que le coin supérieur gauche de l'image est l'origine (0,0).

Le`top`et`left`les valeurs renvoyées sont des ratios de la taille globale de la page du document. Par exemple, si l'image d'entrée a une résolution de 700 x 200 pixels, et que la coordonnée supérieure gauche du cadre de délimitation est de 350 x 50 pixels, l'API renvoie une valeur `left` de 0,5 (350/700) et une valeur `top` de 0,25 (50/200).

Le`width`et`height`Les valeurs représentent les dimensions du cadre de délimitation sous forme de ratio de la dimension de page de document globale. Par exemple, si la taille de la page du document est de 700 x 200 pixels et que la largeur du cadre de sélection est de 70 pixels, la largeur renvoyée est de 0,1.

Table des matières

Height

Height Hauteur du cadre de délimitation sous forme de ratio de la hauteur d'image globale.

Type : Float

Obligatoire Non

Left

Left Coordonnée gauche du cadre de délimitation sous forme de ratio de la largeur d'image globale.

Type : Float

Obligatoire Non

Top

Top Coordonnée supérieure du cadre de délimitation sous forme de ratio de la hauteur d'image globale.

Type : Float

Obligatoire Non

Width

Width Largeur du cadre de délimitation sous forme de ratio de la largeur d'image globale.

Type : Float

Obligatoire Non

Voir aussi

Pour plus d’informations sur l’utilisation de cette API dans l’un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- AWSSDK pour Ruby V3

Document

Le document d'entrée, soit en octets, soit en tant qu'objet S3.

Pour transmettre des octets d'image à une opération d'API Amazon Textract, utilisez la commande`Bytes`propriété. Par exemple, vous utiliseriez la commande`Bytes`pour transmettre un document chargé à partir d'un système de fichiers local. Octets d'image transmis à l'aide de la`Bytes`doit être codée en base64. Il est possible que votre code n'ait pas besoin de coder des octets de fichiers de documents si vous utilisez un kit SDK AWS pour appeler les opérations de l'API Amazon Textract.

Pour transmettre des images stockées dans un compartiment S3 à une opération d'API Amazon Textract, utilisez la commande`S3Object`propriété. Il n'est pas nécessaire d'encoder en base64 les documents stockés dans un compartiment S3.

La région AWS du compartiment S3 contenant l'objet S3 doit correspondre à la région AWS que vous utilisez pour les opérations Amazon Textract.

Si vous utilisez l'AWS CLI pour appeler les opérations Amazon Textract, la transmission d'octets d'image à l'aide de la propriété Bytes n'est pas prise en charge. Vous devez d'abord charger le document dans un compartiment Amazon S3, puis appeler l'opération à l'aide de la propriété S3Object.

Pour qu'Amazon Textract traite un objet S3, l'utilisateur doit disposer des autorisations permettant d'accéder à l'objet S3.

Table des matières

Bytes

Un blob d'octets de document codés en base64. La taille maximale d'un document fourni dans un blob d'octets est de 5 Mo. Les octets de document doivent être au format PNG ou JPEG.

Si vous utilisez un kit SDK AWS pour appeler Amazon Textract, il est possible que vous n'ayez pas besoin de coder en base64 octets d'image transmis à l'aide de l'option`Bytes`.

Type : Objet de données binaires encodées en base64

Contraintes de longueur : Longueur minimale de 1. Longueur maximale de 10485760.

Obligatoire Non

S3Object

Identifie un objet S3 en tant que source de document. La taille maximale d'un document stocké dans un compartiment S3 est de 5 Mo.

Type : objet S3Object

Obligatoire Non

Voir aussi

Pour plus d'informations sur l'utilisation de cette API dans l'un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- AWSSDK pour Ruby V3

DocumentLocation

Compartiment Amazon S3 qui contient le document à traiter. Il est utilisé par des opérations asynchrones telles queStartDocumentTextDetection.

Le document d'entrée peut être un fichier image au format JPEG ou PNG. Il peut également s'agir d'un fichier au format PDF.

Table des matières

S3Object

Compartiment Amazon S3 qui contient le document d'entrée.

Type : objet S3Object

Obligatoire Non

Voir aussi

Pour plus d’informations sur l’utilisation de cette API dans l’un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- AWSSDK pour Ruby V3

DocumentMetadata

Informations sur le document d'entrée.

Table des matières

Pages

Nombre de pages détectées dans le document.

Type : Entier

Plage valide : La valeur minimale est 0.

Obligatoire Non

Voir aussi

Pour plus d’informations sur l’utilisation de cette API dans l’un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- AWSSDK pour Ruby V3

ExpenseDetection

Objet utilisé pour stocker des informations sur la valeur ou l'étiquette détectée par Amazon Textract.

Table des matières

Confidence

La confiance dans la détection, en pourcentage

Type : Float

Plage valide : La valeur minimale est 0. Valeur maximale fixée à 100.

Obligatoire Non

Geometry

Informations sur l'emplacement des éléments suivants sur une page de document : page détectée, texte, paires de clé-valeur, des tableaux, des cellules de tableau et des éléments de sélection.

Type : objet Geometry

Obligatoire Non

Text

Le mot ou la ligne de texte reconnu par Amazon Textract

Type : Chaîne

Obligatoire Non

Voir aussi

Pour plus d'informations sur l'utilisation de cette API dans l'un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2

- AWSSDK pour Ruby V3

ExpenseDocument

La structure contenant toutes les informations renvoyées par AnalyzeExpense

Table des matières

ExpenseIndex

Désigne la facture ou le reçu du document dont proviennent les informations. Le premier document sera 1, le deuxième document, et ainsi de suite.

Type : Entier

Plage valide : La valeur minimale est fixée à 0.

Obligatoire Non

LineItemGroups

Informations détectées sur chaque table d'un document, divisées en`LineItems`.

Type : Tableau deLineItemGroupobjets

Obligatoire Non

SummaryFields

Toutes les informations trouvées en dehors d'une table par Amazon Textract.

Type : Tableau deExpenseFieldobjets

Obligatoire Non

Voir aussi

Pour plus d’informations sur l’utilisation de cette API dans l’un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- AWSSDK pour Ruby V3

ExpenseField

Ventilation des informations détectées, divisées en catégories Type, LabelDetection et ValueDetection

Table des matières

LabelDetection

L'étiquette explicitement indiquée d'un élément détecté.

Type : objet ExpenseDetection

Obligatoire Non

PageNumber

Numéro de page sur lequel la valeur a été détectée.

Type : Entier

Plage valide : La valeur minimale est fixée à 0.

Obligatoire Non

Type

Étiquette implicite d'un élément détecté. Présent aux côtés de LabelDetection pour les éléments explicites.

Type : objet ExpenseType

Obligatoire Non

ValueDetection

La valeur d'un élément détecté. Présent dans des éléments explicites et implicites.

Type : objet ExpenseDetection

Obligatoire Non

Voir aussi

Pour plus d’informations sur l’utilisation de cette API dans l’un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- AWSSDK pour Ruby V3

ExpenseType

Objet utilisé pour stocker des informations sur le type détecté par Amazon Textract.

Table des matières

Confidence

La confiance de la précision, en pourcentage.

Type : Float

Plage valide : La valeur minimale est fixée à 0. Valeur maximale fixée à 100.

Obligatoire Non

Text

Le mot ou la ligne de texte détecté par Amazon Textract.

Type : Chaîne

Obligatoire Non

Voir aussi

Pour plus d'informations sur l'utilisation de cette API dans l'un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- AWSSDK pour Ruby V3

Geometry

Informations sur l'emplacement des éléments suivants sur une page de document : page détectée, texte, paires de clé-valeur, des tableaux, des cellules de tableau et des éléments de sélection.

Table des matières

BoundingBox

Représentation grossière alignée sur l'axe de l'emplacement de l'élément reconnu sur la page de document.

Type : objet BoundingBox

Obligatoire Non

Polygon

Dans le cadre de sélection, un polygone à grains fins autour de l'élément reconnu.

Type : Tableau dePointobjets

Obligatoire Non

Voir aussi

Pour plus d'informations sur l'utilisation de cette API dans l'un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- AWSSDK pour Ruby V3

HumanLoopActivationOutput

Affiche les résultats de l'évaluation humaine dans la boucle. S'il n'y a pas de HumanLooParn, l'entrée n'a pas déclenché de révision humaine.

Table des matières

HumanLoopActivationConditionsEvaluationResults

Affiche le résultat des évaluations de l'état, y compris les conditions qui ont activé un examen humain.

Type : Chaîne

Contraintes de longueur : Longueur maximale de 10 240.

Obligatoire Non

HumanLoopActivationReasons

Démontre si et pourquoi un examen humain était nécessaire.

Type : Tableau de chaînes

Membres du tableau : Nombre minimum de 1 élément.

Obligatoire Non

HumanLoopArn

Amazon Resource Name (ARN) de HumanLoop créé.

Type : Chaîne

Contraintes de longueur : Longueur maximum de 256.

Obligatoire Non

Voir aussi

Pour plus d’informations sur l’utilisation de cette API dans l’un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++

- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- AWSSDK pour Ruby V3

HumanLoopConfig

Configure le flux de travail de révision humaine vers lequel le document sera envoyé si l'une des conditions est remplie. Vous pouvez également définir certains attributs de l'image avant de les réviser.

Table des matières

DataAttributes

Définit les attributs des données d'entrée.

Type : objet HumanLoopDataAttributes

Obligatoire Non

FlowDefinitionArn

Amazon Resource Name (ARN) de la définition de flux.

Type : Chaîne

Contraintes de longueur : Longueur maximum de 256.

Obligatoire Oui

HumanLoopName

Nom du flux de travail humain utilisé pour cette image. Cela doit être unique au sein d'une région.

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximum de 63.

Modèle : `^[a-z0-9](-*[a-z0-9])*`

Obligatoire Oui

Voir aussi

Pour plus d'informations sur l'utilisation de cette API dans l'un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++

- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- AWSSDK pour Ruby V3

HumanLoopDataAttributes

Permet de définir les attributs de l'image. Actuellement, vous pouvez déclarer une image comme étant exempte d'informations personnelles identifiables et de contenus pour adultes.

Table des matières

ContentClassifiers

Définit si l'image de saisie est exempte d'informations personnelles identifiables ou de contenu pour adultes.

Type : Tableau de chaînes

Membres de tableau : Nombre maximal de 256 éléments.

Valeurs valides : `FreeOfPersonallyIdentifiableInformation | FreeOfAdultContent`

Obligatoire Non

Voir aussi

Pour plus d’informations sur l’utilisation de cette API dans l’un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- AWSSDK pour Ruby V3

IdentityDocument

Structure qui répertorie chaque document traité dans une opération AnalyzeID.

Table des matières

DocumentIndex

Indique le placement d'un document dans la liste IdentityDocument. Le premier document porte la mention 1, le second 2 et ainsi de suite.

Type : Entier

Plage valide : La valeur minimale est fixée à 0.

Obligatoire Non

IdentityDocumentFields

Structure utilisée pour enregistrer des informations extraites de documents d'identité. Contient à la fois un champ normalisé et une valeur du texte extrait.

Type : Tableau deIdentityDocumentFieldobjets

Obligatoire Non

Voir aussi

Pour plus d’informations sur l’utilisation de cette API dans l’un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- AWSSDK pour Ruby V3

IdentityDocumentField

Structure contenant à la fois le type normalisé des informations extraites et le texte qui y est associé. Ils sont extraits respectivement sous forme de type et de valeur.

Table des matières

Type

Utilisé pour contenir les informations détectées par une opération AnalyzeID.

Type : objet AnalyzeIDDetections

Obligatoire Non

ValueDetection

Utilisé pour contenir les informations détectées par une opération AnalyzeID.

Type : objet AnalyzeIDDetections

Obligatoire Non

Voir aussi

Pour plus d'informations sur l'utilisation de cette API dans l'un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- AWSSDK pour Ruby V3

LineItemFields

Structure qui contient des informations sur les différentes lignes présentes dans les tableaux d'un document.

Table des matières

LineItemExpenseFields

ExpenseFields permet d'afficher des informations provenant de lignes détectées sur une table.

Type : Tableau deExpenseFieldobjets

Obligatoire Non

Voir aussi

Pour plus d’informations sur l’utilisation de cette API dans l’un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- AWSSDK pour Ruby V3

LineItemGroup

Un regroupement de tables contenant des éléments LineItems, chaque table étant identifiée par le`LineItemGroupIndex`.

Table des matières

LineItemGroupIndex

Numéro utilisé pour identifier une table spécifique dans un document. La première table rencontrée aura un LineItemGroupIndex de 1, le second 2, etc.

Type : Entier

Plage valide : La valeur minimale est fixée à 0.

Obligatoire Non

LineItems

La ventilation des informations sur une ligne particulière d'un tableau.

Type : Tableau deLineItemFieldsobjets

Obligatoire Non

Voir aussi

Pour plus d’informations sur l’utilisation de cette API dans l’un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- AWSSDK pour Ruby V3

NormalizedValue

Contient des informations relatives aux dates d'un document, y compris le type de valeur et la valeur.

Table des matières

Value

La valeur de la date, écrite sous la forme Year-Mois-Daythour:Minute:Second.

Type : Chaîne

Obligatoire Non

ValueType

Type normalisé de la valeur détectée. Dans ce cas, DATE.

Type : Chaîne

Valeurs valides : `DATE`

Obligatoire Non

Voir aussi

Pour plus d'informations sur l'utilisation de cette API dans l'un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- AWSSDK pour Ruby V3

NotificationChannel

Amazon Simple Notification Service (Amazon SNS) dans laquelle Amazon Textract publie l'état d'achèvement d'une opération de document asynchrone, telle queStartDocumentTextDetection.

Table des matières

RoleArn

Amazon Resource Name (ARN) d'un rôle IAM qui accorde des autorisations de publication Amazon Textract à la rubrique Amazon SNS.

Type : Chaîne

Contraintes de longueur : Longueur minimale de 20. Longueur maximale de 2048.

Modèle : `arn:([a-z\d-]+):iam::\d{12}:role/?[a-zA-Z_0-9+=,.@\-_/]+`

Obligatoire Oui

SNSTopicArn

Rubrique Amazon SNS sur laquelle Amazon Textract publie l'état d'achèvement.

Type : Chaîne

Contraintes de longueur : Longueur minimale de 20. Longueur maximum de 1024.

Modèle : `(^arn:([a-z\d-]+):sns:[a-zA-Z\d-]{1,20}:\w{12}:.+$)`

Obligatoire Oui

Voir aussi

Pour plus d'informations sur l'utilisation de cette API dans l'un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- AWSSDK pour Ruby V3

OutputConfig

Définit si votre sortie sera envoyée ou non à un compartiment créé par l'utilisateur. Permet de définir le nom du compartiment et le préfixe du fichier de sortie.

`OutputConfig`est un paramètre facultatif qui vous permet de régler l'endroit où votre sortie sera placée. Par défaut, Amazon Textract stocke les résultats en interne et n'est accessible que par les opérations Obtenir l'API. Lorsque OutputConfig est activé, vous pouvez définir le nom du compartiment vers lequel la sortie sera envoyée et le préfixe de fichier des résultats dans lequel vous pouvez télécharger vos résultats. De plus, vous pouvez définir la`KMSKeyID`d'une clé principale de client (CMK) pour chiffrer votre sortie. Sans ce jeu de paramètres, Amazon Textract chiffrera côté serveur à l'aide du CMK géré par AWS pour Amazon S3.

Le déchiffrement du contenu client est nécessaire au traitement des documents par Amazon Textract. Si votre compte est désactivé en vertu d'une politique de désactivation des services IA, tout le contenu client non chiffré est immédiatement et définitivement supprimé une fois que le contenu client a été traité par le service. Aucune copie de la sortie n'est conservée par Amazon Textract. Pour plus d'informations sur la façon de désactiver, consultezGestion de la politique de désactivation des services IA.

Pour plus d'informations sur la confidentialité des données, consultezFAQ sur la confidentialité des données.

Table des matières

S3Bucket

Le nom du compartiment vers lequel votre sortie va aller.

Type : Chaîne

Contraintes de longueur : Longueur minimale de 3. Longueur maximale de 255.

Modèle : `[0-9A-Za-z\.\-_]*`

Requise : Oui

S3Prefix

Le préfixe de la clé d'objet dans laquelle la sortie sera enregistrée. Lorsqu'il n'est pas activé, le préfixe sera « textract_output ».

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximum de 1024.

Modèle : `.*\S.*`

Requise : Non

Voir aussi

Pour plus d'informations sur l'utilisation de cette API dans l'un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- AWSSDK pour Ruby V3

Point

Les coordonnées X et Y d'un point sur une page de document. Les valeurs X et Y renvoyées sont des ratios de la taille globale de la page du document. Par exemple, si le document en entrée est de 700 x 200 et que l'opération renvoie X=0,5 et Y=0,25, le point se situe à la coordonnée (350,50) pixels sur la page du document.

Tableau d'éléments`Point`objets,`Polygon`est retourné comme partie intégrante de l'Geometryobjet renvoyé dans unBlockobjet. UN`Polygon`objet représente un polygone à grains fins autour du texte détecté, une paire clé-valeur, un tableau, une cellule de tableau ou un élément de sélection.

Table des matières

X

La valeur de la coordonnée X d'un point sur un`Polygon`.

Type : Float

Obligatoire Non

Y

La valeur de la coordonnée Y pour un point sur un`Polygon`.

Type : Float

Obligatoire Non

Voir aussi

Pour plus d’informations sur l’utilisation de cette API dans l’un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- AWSSDK pour Ruby V3

Relationship

Informations sur la façon dont les blocs sont liés les uns aux autres. UNBlockobjet contient 0 ou plusRelationobjets dans une liste,Relationships. Pour plus d'informations, consultez Block.

LeTypefournit le type de relation pour tous les blocs de laIDstableau.

Table des matières

Ids

Un tableau d'identifiants pour les blocs associés. Vous pouvez accéder au type de relation à partir duTypeélément.

Type : Tableau de chaînes

Modèle : `.*\S.*`

Obligatoire Non

Type

Type de relation entre les blocs du tableau ID et le bloc actuel. La relation peut êtreVALUEouCHILD. Une relation de type VALUE est une liste qui contient l'ID du bloc VALUE associé à la CLÉ d'une paire clé-valeur. Une relation de type CHILD est une liste d'ID qui identifie les blocs WORD dans le cas des lignes, des blocs de cellules dans le cas des tables, et des blocs WORD dans le cas des éléments de sélection.

Type : Chaîne

Valeurs valides : `VALUE | CHILD | COMPLEX_FEATURES`

Obligatoire Non

Voir aussi

Pour plus d’informations sur l’utilisation de cette API dans l’un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++
- Kit AWS SDK pour Go

- AWSSDK pour Java V2
- AWSSDK pour Ruby V3

S3Object

Le nom du compartiment S3 et le nom de fichier qui identifient le document.

La région AWS du compartiment S3 qui contient le document doit correspondre à la région que vous utilisez pour les opérations Amazon Textract.

Pour qu'Amazon Textract traite un fichier dans un compartiment S3, l'utilisateur doit être autorisé à accéder au compartiment et au fichier S3.

Table des matières

Bucket

Nom du compartiment S3. Notez que le caractère # n'est pas valide dans le nom de fichier.

Type : Chaîne

Contraintes de longueur : Longueur minimale de 3. Longueur maximale de 255.

Modèle : `[0-9A-Za-z\.\-_]*`

Obligatoire Non

Name

Nom de fichier du document d'entrée. Les opérations synchrone peuvent utiliser des fichiers image au format JPEG ou PNG. Les opérations asynchrones prennent également en charge les fichiers au format PDF et TIFF.

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximum de 1024.

Modèle : `.*\S.*`

Obligatoire Non

Version

Si le compartiment est activé pour la gestion des versions, vous pouvez spécifier la version de l'objet.

Type : Chaîne

Contraintes de longueur : Longueur minimale de 1. Longueur maximum de 1024.

Modèle : `.*\S.*`

Obligatoire Non

Voir aussi

Pour plus d'informations sur l'utilisation de cette API dans l'un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- AWSSDK pour Ruby V3

Warning

Avertissement concernant un problème survenu lors de l'analyse de texte asynchrone (StartDocumentAnalysis) ou détection de texte de document asynchrone (StartDocumentTextDetection).

Table des matières

ErrorCode

Code d'erreur correspondant à l'avertissement.

Type : Chaîne

Obligatoire Non

Pages

Liste des pages auxquelles l'avertissement s'applique.

Type : Tableau de nombres entiers

Plage valide : La valeur minimale est fixée à 0.

Obligatoire Non

Voir aussi

Pour plus d'informations sur l'utilisation de cette API dans l'un des kits SDK AWS spécifiques au langage, consultez les ressources suivantes :

- Kit AWS SDK pour C++
- Kit AWS SDK pour Go
- AWSSDK pour Java V2
- AWSSDK pour Ruby V3

Limites strictes dans Amazon Textract

Vous trouverez ci-dessous la liste des limites strictes d'Amazon Textract, qui ne peuvent pas être modifiées. Pour de plus amples informations sur les limitations d'emplacement et les limites que vous pouvez modifier, veuillez consulterPoints de terminaison et quotas Amazon Textract. Pour plus d'informations sur les limites que vous pouvez modifier, consultez Limites de service AWS. Pour modifier une limite, consultez Création d'une demande.

Amazon Textract

Limite	Description
Formats de fichier acceptés	Les opérations prennent en charge les fichiers JPEG, PNG, PDF et TIFF. (Les images encodées au format JPEG 2000 dans les PDF sont prises en charge).
Limites de taille de fichier et de nombre de pages	Pour les opérations synchrone, les fichiers JPEG, PNG, PDF et TIFF ont une taille limite de 10 Mo. Les fichiers PDF et TIFF ont également une limite d'une page. Pour les opérations asynchron es, les fichiers JPEG et PNG ont une taille limite de 10 Mo. Les fichiers PDF et TIFF ont une limite de 500 Mo. Les fichiers PDF et TIFF ont une limite de 3 000 pages.
Limites propres au PDF	La hauteur et la largeur maximales sont de 40 pouces et de 2880 points. Les PDF ne peuvent pas être protégés par mot de passe. Les fichiers PDF peuvent contenir des images au format JPEG 2000.
Rotation du document et taille de l'image	Amazon Textract prend en charge toutes les rotations de documents dans le plan, par exemple une rotation de 45 degrés dans le plan. Amazon Textract prend en charge les images dont la résolution est inférieure ou égale à 10000 pixels de tous les côtés.

Limite	Description
Alignement du texte	Le texte peut être aligné horizontalement dans le document. Amazon Textract ne prend pas en charge l'alignement vertical du texte dans le document.
Langages	Amazon Textract prend en charge la détection de texte anglais, français, allemand, italien, portugais et espagnol. Amazon Textract ne renvoie pas la langue détectée dans sa sortie.
Taille caractères	La hauteur minimale du texte à détecter est de 15 pixels. À 150 ppp, cette police serait identique à une police à 8 points.
Type caractères	Amazon Textract prend en charge la reconnaissance manuscrite et imprimée des caractères.
Caractères	Amazon Textract détecte les caractères suivants : • a-z • A-Z • 0-9 • ä Ö Ö ü ü ç é é ê ê î ô ô û Ã Ã Ã Ã Ã Ã Ã è ù ù ë ï ü ò ò Ã Ã ÃÕ Õ • ! « # $% '& () * +, -./: ; = ? @ [\] ^ _ `{|} ~ > <° € £ ¥ ¥ ß ¿¡€ ¥ ¥ ¥ ø ø œ©®™ § ¹ ² ³ '
Limites propres à AnalyzeID	AnalyzeID ne prend en charge que les passeports américains et les permis de conduire américains.

Historique du document pour Amazon Textract

Le tableau ci-après décrit les modifications importantes dans chaque édition duAmazon Textract Developer Guide. Pour receConsultez les notifications des mises à jour de cette documentation, abonnez-vous à un flux RSS.

- Dernière mise à jour de la documentation : 29 mai 2019

update-history-change	update-history-description	update-history-date
Intégration d'exemples de code à partir deAWSExemples de code Docs SDK Repo GitHub	Le guide Amazon Textract contient désormais des exemples de code supplémentaires. La section Exemples précédents a été renommée Tutoriels.	30 janvier 2022
AnalyzeExpense Ajouté	Amazon Textract prend désormais en charge l'analyse des documents de facture et de réception à l'aide de l'API AnalyzeExpense. Cette fonctionnalité est uniquement disponible dans nos régions Asie-Pacifique (Mumbai), Asie-Pacifique (Séoul), Asie-Pacifique (Singapour), Asie-Pacifique (Sydney), UE (Londres), USA Est (Virginie du Nord), USA Est (Oregon), USA Ouest (Californie du Nord) et USA Ouest (Oregon).	26 juillet 2021
Augmented AI Support	Amazon Textract prend désormais en charge Amazon	3 décembre 2019

	Augmented AI pour la mise en œuvre de la revue humaine.	
Nouveau guide et service	Amazon Textract est maintenant disponible pour une utilisation générale.	29 mai 2019
Support des éléments de sélection	Amazon Textract peut désormais détecter les éléments de sélection (boutons radio et cases à cocher).	24 avril 2019
Version d'Amazon Textract	Il s'agit de la première version de la documentation pour Amazon Textract.	28 novembre 2018

Glossaire AWS

Pour connaître la terminologie la plus récente d'AWS, consultez le glossaire AWS dans la Référence générale d'AWS.

Les traductions sont fournies par des outils de traduction automatique. En cas de conflit entre le contenu d'une traduction et celui de la version originale en anglais, la version anglaise prévaudra.