aws

개발자 가이드

Amazon Simple Workflow Service

API 버전 2012-01-25

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

Amazon Simple Workflow Service: 개발자 가이드

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

Amazon 상표 및 트레이드 드레스는 Amazon 외 제품 또는 서비스와 함께 사용되어서는 안되며, 고객에게 혼동을 일으키거나 Amazon 브랜드 이미지를 떨어뜨리고 폄하하는 방식으로 이용할 수 없습니다. Amazon이 소유하지 않은 기타 모든 상표는 Amazon 계열사, 관련 업체 또는 Amazon의 지원 업체 여부에 상관없이 해당 소유자의 자산입니다.

Table of Contents

Amazon Simple Workflow Service란 무엇인가요?

Amazon Simple Workflow Service(Amazon SWF)를 사용하면 병렬 또는 순차적 단계가 있는 백그라운드 작업을 빌드, 실행 및 확장할 수 있습니다. 분산 구성 요소 간에 작업을 조정하고 작업 상태를 추적할 수 있습니다.

Amazon SWF에서 작업은 애플리케이션의 구성 요소에 의해 수행되는 논리적 작업 단위를 나타냅니다. 여러 간 작업 조정에는 애플리케이션 흐름에서 작업 간 종속성, 예약 및 동시성 관리가 포함됩니다. Amazon SWF를 사용하면 진행 상황 추적 및 작업 상태 유지와 같은 기본 복잡성에 대한 걱정 없이 작업을 제어하고 조정할 수 있습니다.

Amazon SWF를 사용하는 경우 작업자를 구현하여 작업을 수행합니다. 작업자는 Amazon Elastic Compute Cloud(Amazon EC2)와 같은 클라우드 인프라 또는 자체 온프레미스에서 실행할 수 있습니다. 오래 실행되거나, 실패하거나, 제한 시간이 초과되거나, 재시작이 필요할 수 있는 작업 또는 다양한 처리량과 대기 시간을 통해 완료될 수 있는 작업을 생성할 수 있습니다. Amazon SWF는 작업을 저장하고 준비가 되면 작업자에게 할당하며, 진행 상황을 추적하고, 작업 완료 세부 정보를 포함하여 상태를 유지합니다.

작업을 조정하려면 Amazon SWF에서 최신 작업 상태를 가져오고 해당 상태를 사용하여 후속 작업을 시작하는 프로그램을 작성합니다. Amazon SWF는 애플리케이션의 실행 상태를 안정적으로 유지하므로 애플리케이션은 개별 구성 요소 장애에 대한 복원력이 뛰어납니다. Amazon SWF를 사용하면 애플리케이션 구성 요소를 독립적으로 빌드, 배포, 확장 및 수정할 수 있습니다.

> **기타 AWS 워크플로 서비스**
>
> 대부분의 사용 사례에서는 워크플로 및 오케스트레이션 요구 AWS Step Functions 사항을 고려하는 것이 좋습니다.
> Step Functions를 사용하면 상태 시스템이라고도 하는 워크플로를 생성하여 분산 애플리케이션을 구축하고, 프로세스를 자동화하고, 마이크로서비스를 오케스트레이션하고, 데이터 및 기계 학습 파이프라인을 생성할 수 있습니다. VS Code의 Step Functions 콘솔 또는 AWS 툴킷에서 그래픽 Workflow Studio를 사용하여 애플리케이션의 워크플로를 시각화, 편집, 테스트 및 디버깅할 수 있습니다.
> 자세한 기술 정보는 AWS Step Functions 개발자 안내서를 참조하세요.

Amazon SWF를 사용하여 워크플로 구성 요소 개발

분산 애플리케이션을 개발하려면 여러 구성 요소를 조정하고 원격 통신에 내재된 지연 시간과 불안정성을 처리해야 합니다.

Amazon Simple Workflow Service(Amazon SWF)를 사용하면 분산 구성 요소를 조정하고 안정적인 방식으로 실행 상태를 유지하기 위한 프로그래밍 모델 및 인프라를 제공하여 비동기식 분산 애플리케이션을 개발할 수 있습니다. Amazon SWF를 사용하면 차별화되는 애플리케이션 측면을 구축하는 데 마음껏 집중할 수 있습니다.

워크플로의 구성 요소

워크플로의 구성 요소 Amazon SWF의 기본 개념은 워크플로입니다. 워크플로는 활동을 조정하는 논리와 함께 목적을 수행하는 활동 세트입니다. 예를 들어 워크플로는 고객 주문을 수신하고 주문을 이행하는 데 필요한 모든 조치를 취할 수 있습니다.

각 워크플로는 워크플로의 범위를 제어하는 도메인이라는 리소스에서 실행됩니다. AWS 계정에는 여러 도메인이 있을 수 있고 각 도메인에는 여러 워크플로가 있을 수 있지만 다른 도메인의 워크플로는 상호 작용할 수 없습니다.

Amazon SWF 워크플로를 설계할 때 필요한 각 활동을 정의합니다. 그런 다음 Amazon SWF에 각 활동을 활동 유형으로 등록합니다. 이름, 버전 및 제한 시간 값을 제공합니다. 예를 들어, 고객은 24시간 이내에 주문이 배송될 것이라고 기대할 수 있습니다.

워크플로를 수행하는 과정에서 일부 활동은 입력을 변경하며 한 번 이상 수행해야 할 수 있습니다. 예를 들어, 고객 주문 워크플로에서 구매 품목을 처리하는 활동이 있을 수 있습니다. 고객이 여러 품목을 구매하는 경우 이 활동을 여러 번 실행해야 합니다. Amazon SWF에는 활동의 간접 호출을 나타내는 활동 작업이라는 개념이 있습니다. 이 예에서는 각 품목의 처리를 활동 작업 하나로 표현합니다.

활동 작업자는 활동 작업을 수신하고 수행하며 결과를 제공하는 프로그램입니다. 작업은 실제로 사람이 수행할 수 있습니다. 예를 들어 통계 분석가는 데이터 세트를 수신하고 데이터를 분석한 다음 분석을 다시 보낼 수 있습니다.

활동 작업과 이를 수행하는 활동 작업자는 동기식 또는 비동기식으로 실행할 수 있습니다. 작업자는 한 위치에서 실행하거나 잠재적으로 다른 지리적 리전에 있는 여러 컴퓨터에 배포할 수 있습니다. 여러 활동 작업자는 각기 다른 프로그래밍 언어로 작성되어 다른 운영 체제에서 실행될 수 있습니다. 예를 들어 한 활동 작업자는 아시아의 서버에서 실행되고 다른 활동 작업자는 북미의 모바일 디바이스에서 실행될 수 있습니다.

워크플로의 조정 로직은 결정자라고 하는 소프트웨어 프로그램에 들어 있습니다. 결정자는 활동 작업을 예약하고, 활동 작업자에게 입력을 제공하고, 워크플로가 진행되는 동안 도착하는 이벤트를 처리하고, 목표가 충족된 후 워크플로를 종료(또는 종료)합니다.

Amazon SWF 서비스의 역할은 결정자, 활동 작업자 및 기타 관련 엔터티(예: 워크플로 관리자) 간에 데이터가 교환되는 신뢰할 수 있는 중앙 허브의 기능을 하는 것입니다. 또한 Amazon SWF는 각 워크플로 실행 상태를 기록하므로 애플리케이션이 상태를 오래도록 안전하게 저장할 필요가 없습니다.

결정자는 Amazon SWF로부터 결정 작업을 수신한 다음 결정과 함께 다시 Amazon SWF에 응답해 워크플로에 지시합니다. 결정은 워크플로의 다음 단계인 작업 또는 작업 세트를 나타냅니다. 일반적인 결정은 활동 작업을 예약합니다. 결정은 타이머로 작업을 지연하고, 진행 중인 작업의 취소를 요청하고, 워크플로를 완료하는 데에도 사용할 수 있습니다.

활동 작업자와 결정자가 작업(각기 활동 작업 및 결정 작업)을 수신하는 메커니즘은 Amazon SWF 서비스를 폴링하는 것입니다.

Amazon SWF는 각 결정 작업과 함께 현재 워크플로 실행 내역을 포함해 워크플로의 상태를 결정자에게 알립니다. 워크플로 실행 내역은 이벤트로 구성되어 있는데, 여기서 이벤트는 워크플로 실행 상태의 중요한 변경을 나타냅니다. 이벤트의 예로는 작업 완료, 작업 제한 시간 또는 타이머 만료 등이 있습니다. 내역은 워크플로 진행 상황의 완벽하고 일관되며 신뢰할 수 있는 기록입니다.

Amazon SWF 액세스 제어는 AWS Identity and Access Management (IAM)을 사용하므로 AWS 리소스에 대한 액세스를 제어할 수 있습니다. 예를 들어, 사용자가 여러분의 계정에 액세스하도록 허용하지만 특정 도메인에 있는 특정 워크플로만 실행하도록 할 수 있습니다.

워크플로 실행

다음은 Amazon SWF에서 워크플로를 개발하고 실행하는 데 필요한 단계의 개요를 제공합니다.

1. 워크플로에서 처리 단계를 수행할 활동 작업자를 작성합니다.
2. 결정자를 작성하여 워크플로의 조정 로직을 처리합니다.
3. Amazon SWF에 활동 및 워크플로를 등록합니다.

   프로그래밍 방식으로 또는를 사용하여이 단계를 수행할 수 있습니다 AWS Management Console.
4. 활동 작업자와 결정자를 시작합니다.

   이러한 액터는 Amazon SWF 엔드포인트에 액세스할 수 있는 컴퓨팅 디바이스에서 실행할 수 있습니다. 예를 들어, 클라우드에서 컴퓨팅 인스턴스[예: 데이터 센터의 Amazon Elastic Compute

Cloud(Amazon EC2); 서버 또는 모바일 디바이스]를 사용해 결정자 또는 활동 작업자를 호스팅할 수 있습니다. 시작하면 결정자 및 활동 작업자는 작업을 위해 Amazon SWF 폴링을 시작해야 합니다.

5. 하나 이상의 워크플로 실행을 시작합니다.

   프로그래밍 방식으로 또는를 통해 워크플로를 시작할 수 있습니다 AWS Management Console.

   각 실행은 독립적으로 실행되며 각 실행에 고유한 입력 데이터 세트를 제공할 수 있습니다. 실행이 시작되면 Amazon SWF에서 최초의 결정 작업을 예약합니다. 이에 대한 응답으로 결정자는 활동 작업을 시작하는 결정을 생성하기 시작합니다. 결정자가 실행을 닫는 결정을 생성할 때까지 계속해서 실행됩니다.

6. 를 사용하여 워크플로 실행을 봅니다 AWS Management Console.

   실행 및 완료된 실행에 대한 전체 세부 정보를 필터링하고 볼 수 있습니다. 예를 들어 열린 실행을 선택하여 완료된 작업과 그 결과를 확인할 수 있습니다.

개발 환경 설정

에서 지원하는 모든 프로그래밍 언어로 Amazon SWF용를 개발할 수 있습니다 AWS. Java 개발자 AWS Flow Framework 의 경우 도 사용할 수 있습니다. 자세한 내용은 AWS Flow Framework 웹 사이트를 참조하고 for AWS Flow Framework Java 개발자 안내서를 참조하세요.

Amazon SWF는 지연 시간을 줄이고 요구 사항을 충족하는 위치에 데이터를 저장하기 위해 다양한 리전의 엔드포인트를 제공합니다.

Amazon SWF의 각 엔드포인트는 완전히 독립적입니다. 한 리전에 등록한 도메인, 워크플로 및 활동은 데이터 또는 속성을 다른 리전의 도메인, 워크플로 및 활동과 공유하지 않습니다.

Amazon SWF 도메인, 워크플로 또는 활동을 등록하면 등록한 리전 내에만 존재합니다. 예를 들어 서로 다른 두 리전SWF-Flows-1에 라는 도메인을 등록할 수 있지만 각각 완전히 독립적인 도메인 역할을 하는 데이터나 속성을 서로 공유하지 않습니다.

Amazon SWF 엔드포인트 목록은 리전 및 엔드포인트를 참조하십시오.

AWS SDKs 사용하여 개발

Amazon SWF는 Java, .NET, Node.js, PHP, Python 및 Ruby용 AWS SDKs에서 지원되므로 원하는 프로그래밍 언어로 Amazon SWF HTTP API를 편리하게 사용할 수 있습니다.

이러한 라이브러리에서 노출되는 API를 사용하여 결정자, 활동 작업자 또는 워크플로 시작자를 개발할 수 있습니다. 또한 이러한 라이브러리를 통해 가시성 작업을 사용하여 자체 Amazon SWF 모니터링 및 보고 도구를 개발할 수 있습니다.

SDKs를 AWS포함하여에서 애플리케이션을 개발하고 관리하기 위한 도구를 다운로드하려면 개발자 센터로 이동하십시오.

각 SDK의 Amazon SWF 작업에 대한 자세한 내용은 SDK의 언어별 참조 설명서를 참조하세요.

를 고려합니다. AWS Flow Framework

AWS Flow Framework 는 Amazon SWF에서 워크플로로 실행되는 분산 비동기 프로그램을 작성하기 위한 향상된 SDK입니다. 프레임워크는 Java 프로그래밍 언어에 사용할 수 있으며 복잡한 분산 프로그램을 작성하기 위한 클래스를 제공합니다.

를 사용하면 미리 구성된 유형을 AWS Flow Framework사용하여 워크플로 정의를 프로그램의 메서드에 직접 매핑할 수 있습니다. 는 예외 기반 오류 처리와 같은 표준 객체 지향 개념을 AWS Flow Framework 지원합니다. 로 작성된 프로그램은 원하는 편집기 또는 IDE 내에서 완전히 생성, 실행 및 디버깅할 AWS Flow Framework 수 있습니다. 자세한 내용은 AWS Flow Framework 웹 사이트를 참조하고 for AWS Flow Framework Java 개발자 안내서를 참조하세요.

Amazon SWF 시작하기

순차적으로 작동하는 네 가지 활동 세트로 구성된 다음 Amazon Simple Workflow Service 워크플로 애플리케이션을 시작할 수 있습니다. 이 자습서에서는 다음 주제도 다룹니다.

- 기본 및 execution-time 워크플로 및 활동 옵션 설정
- 결정 및 활동 작업을 위해 Amazon SWF 폴링
- Amazon SWF를 사용해 활동과 워크플로 간에 데이터 전달
- 사용자의 작업을 대기하고 활동 작업으로부터 하트비트를 Amazon SWF에 보고
- Amazon SNS를 사용하여 주제를 생성하고, 해당 주제를 구독하도록 사용자를 가입하고, 구독 엔드포인트에 메시지 게시

Amazon SWF와 Amazon Simple Notification Service(Amazon SNS)를 함께 사용하여 인간 작업자가 일부 작업을 수행한 다음 Amazon SWF와 통신하여 워크플로에서 다음 활동을 시작해야 하는 "인간 작업" 워크플로를 에뮬레이션할 수 있습니다.

Amazon SWF는 클라우드 기반 웹 서비스이기 때문에 Amazon SWF와의 통신은 인터넷 연결이 가능한 곳이면 어디서든 시작할 수 있습니다. 이러한 경우 Amazon SNS를 사용하여 이메일, SMS 문자 메시지 또는 둘 다를 통해 사용자와 통신합니다.

이 자습서에서는 AWS SDK for Ruby를 사용하여 Amazon SWF 및 Amazon SNS에 액세스하지만, Amazon SWF와의 간편한 조정 및 통신을 제공하는 Ruby AWS Flow Framework 용를 포함하여 다양한 개발 옵션을 사용할 수 있습니다.

> **Note**
>
> 이 자습서에서는를 사용하지 AWS SDK for Ruby만 AWS Flow Framework Java용를 사용하는 것이 좋습니다.

주제

- 구독 워크플로 자습서 파트 2: 워크플로 구현
- 구독 워크플로 자습서 파트 3: 활동 구현
- 구독 워크플로 자습서 파트 4: 활동 작업 Poller 구현
- 구독 워크플로 자습서: 워크플로 실행

워크플로 정보

개발할 워크플로는 다음 4가지 단계로 구성됩니다.

1. 사용자로부터 구독 주소(이메일 또는 SMS)를 얻습니다.
2. SNS 주제를 생성하고 해당 주제에 대해 제공된 엔드포인트를 구독합니다.
3. 사용자가 구독을 확인할 때까지 기다립니다.
4. 사용자가 확인하면 주제에 축하 메시지를 게시합니다.

이러한 단계에는 완전히 자동화된 활동(2 및 4단계)과 워크플로가 진행되기 전에 사용자가 활동에 일부 데이터를 제공할 때까지 워크플로가 대기해야 하는 활동(1 및 3단계)이 포함되어 있습니다.

각 단계에서는 이전 단계에서 생성된 데이터를 사용합니다. 즉, 주제를 구독하려면 엔드포인트가 있어야 하고, 확인을 대기하려면 주제 구독이 있어야 합니다. 이 자습서에서는 완료 시 활동 결과를 제공하는 방법과 예약 중인 작업에 입력을 전달하는 방법도 다룹니다. Amazon SWF는 활동과 워크플로 간의 조정 및 정보 전달을 처리하며, 그 반대의 경우도 처리합니다.

또한 키보드 입력 및 Amazon SNS를 둘 다 사용하여 Amazon SWF와 워크플로에 데이터를 제공하는 사람 간의 통신을 처리합니다. 실제로, 여러 가지 다양한 기법을 사용해 인간 사용자와 통신할 수 있지만 Amazon SNS에서는 이메일 또는 문자 메시지를 사용해 워크플로의 이벤트에 대해 사용자에게 알리는 아주 간단한 방법을 제공합니다.

사전 조건

이 자습서를 따라 수행하려면 다음이 필요합니다.

- Amazon Web Services 계정
- Ruby 인터프리터
- AWS SDK for Ruby

이미 설정한 경우에는 계속할 준비가 된 것입니다. 예제를 실행하지 않으려는 경우에도 자습서를 따를 수 있습니다.이 자습서의 대부분의 내용은 선택한 개발 옵션에 관계없이 Amazon SWF 및 Amazon SNS 사용에 적용됩니다.

자습서 단계

이 자습서는 다음 단계로 나뉩니다.

1. 구독 워크플로 자습서 1부:에서 Amazon SWF 사용 AWS SDK for Ruby
2. 구독 워크플로 자습서 파트 2: 워크플로 구현
3. 구독 워크플로 자습서 파트 3: 활동 구현
4. 구독 워크플로 자습서 파트 4: 활동 작업 Poller 구현
5. 구독 워크플로 자습서: 워크플로 실행

구독 워크플로 자습서 1부:에서 Amazon SWF 사용 AWS SDK for Ruby

주제

- 포함 AWS SDK for Ruby
- AWS 세션 구성
- Amazon SWF 도메인 등록
- 다음 단계

포함 AWS SDK for Ruby

`utils.rb`라는 파일을 생성하여 시작합니다. 이 파일의 코드는 필요한 경우 워크플로 및 활동 코드 둘 다에서 사용하는 Amazon SWF 도메인을 얻거나 생성하고 모든 클래스에 공통적인 코드를 삽입할 위치를 제공합니다.

먼저, SDK for Ruby에서 제공한 기능을 사용할 수 있도록 코드에 `aws-sdk-v1` 라이브러리를 포함해야 합니다.

```
require 'aws-sdk-v1'
```

이를 통해 AWS 네임스페이스에 액세스할 수 있습니다.이 네임스페이스는 AWS 자격 증명 및 리전과 같은 글로벌 세션 관련 값을 설정하는 기능을 제공하고 AWS 서비스 APIs에 대한 액세스도 제공합니다.

AWS 세션 구성

자격 AWS 증명(AWS 서비스에 액세스하는 데 필요)과 사용할 AWS 리전을 설정하여 AWS 세션을 구성합니다.

AWS SDK for Ruby에는 환경 변수(AWS_ACCESS_KEY_ID 및 AWS_SECRET_ACCESS_KEY)에서 자격 AWS 증명을 설정하거나 로 설정하여 자격 증명을 설정하는 여러 가지 방법이 있습니다`AWS.config`.AWS_ACCESS_KEY_ID AWS_SECRET_ACCESS_KEY 여기서는 후자의 방법을 사용하여 다음과 같은 YAML 구성 파일(`aws-config.txt`)에서 로드합니다.

```
---
:access_key_id: REPLACE_WITH_ACCESS_KEY_ID
:secret_access_key: REPLACE_WITH_SECRET_ACCESS_KEY
```

지금이 파일을 생성하여 REPLACE_WITH_로 시작하는 문자열을 AWS 액세스 키 ID 및 보안 액세스 키로 바꿉니다. AWS 액세스 키에 대한 자세한 내용은 Amazon Web Services 일반 참조의 보안 자격 증명을 얻으려면 어떻게 해야 합니까?를 참조하세요.

또한 사용할 AWS 리전을 설정해야 합니다. Amazon SNS를 통해 SMS(문자 서비스)를 사용하여 사용자 휴대폰으로 문자 메시지를 보낼 예정이므로 Amazon SNS가 지원하는 리전을 사용하고 있는지 확인해야 합니다. Amazon Simple Notification Service 개발자 안내서의 지원 리전 및 국가를 참조하십시오.

> **Note**
>
> us-east-1에 액세스할 수 없거나 SMS 메시징이 지원되는 데모를 실행해도 괜찮은 경우 원하는 모든 리전을 자유롭게 사용할 수 있습니다. 샘플에서 SMS 기능을 제거하고 이메일을 단독 엔드포인트로 사용하여 Amazon SNS 주제를 구독할 수 있습니다.
> SMS 메시지 전송에 대한 자세한 내용은 Amazon Simple Notification Service 개발자 안내서의 Amazon SNS를 사용한 SMS 알림 전송 및 수신을 참조하십시오.

이제 `utils.rb`에 일부 코드를 추가하여 구성 파일을 로드하고 사용자의 보안 인증을 가져온 다음 보안 인증과 리전을 `AWS.config`에 제공합니다.

```
require 'yaml'

# Load the user's credentials from a file, if it exists.
begin
  config_file = File.open('aws-config.txt') { |f| f.read }
rescue
  puts "No config file! Hope you set your AWS credentials in the environment..."
end

if config_file.nil?
  options = { }
else
  options = YAML.load(config_file)
end

# SMS Messaging (which can be used by Amazon SNS) is available only in the
# `us-east-1` region.
$SMS_REGION = 'us-east-1'
options[:region] = $SMS_REGION

# Now, set the options
AWS.config = options
```

Amazon SWF 도메인 등록

Amazon SWF를 사용하려면 도메인을 설정해야 합니다. 도메인은 워크플로 및 활동을 보관하는 명명된 엔터티입니다. 많은 Amazon SWF 도메인을 등록할 수 있지만 모두 AWS 계정 내에 고유한 이름이 있어야 하고 워크플로가 도메인 간에 상호 작용할 수 없습니다. 애플리케이션의 모든 워크플로와 활동이 서로 상호 작용하려면 동일한 도메인에 있어야 합니다.

애플리케이션 전체에서 동일한 도메인을 사용할 것이기 때문에 `init_domain`라는 `utils.rb`에서 Amazon SWF 도메인 SWFSampleDomain을 검색하는 함수를 생성합니다.

도메인을 등록하면 여러 워크플로 실행에 재사용할 수 있습니다. 그러나 이미 있는 도메인을 등록하려고 하면 오류가 발생하기 때문에 이 코드에서는 먼저 도메인이 있는지 확인한 다음 있는 경우 기존 도메인을 사용합니다. 도메인이 없으면 생성합니다.

SDK for Ruby에서 Amazon SWF 도메인을 사용하려면 도메인을 열거하고 등록하는 데 사용할 수 있는 DomainCollection을 반환하는 AWS::SimpleWorkflow.domains를 사용합니다.

- 도메인이 이미 등록되어 있는지 확인하려면 AWS::Simpleworkflow.domains.registered에서 제공하는 목록을 살펴봅니다.

- 새 도메인을 등록하려면 AWS::Simpleworkflow.domains.register를 사용합니다.

다음은 `utils.rb`의 `init_domain`에 대한 코드입니다.

```
# Registers the domain that the workflow will run in.
def init_domain
  domain_name = 'SWFSampleDomain'
  domain = nil
  swf = AWS::SimpleWorkflow.new

  # First, check to see if the domain already exists and is registered.
  swf.domains.registered.each do | d |
    if(d.name == domain_name)
      domain = d
      break
    end
  end

  if domain.nil?
    # Register the domain for one day.
    domain = swf.domains.create(
      domain_name, 1, { :description => "#{domain_name} domain" })
  end

  return domain
end
```

다음 단계

다음으로, 구독 워크플로 자습서 파트 2: 워크플로 구현 단원에서 워크플로 및 시작 코드를 생성합니다.

구독 워크플로 자습서 파트 2: 워크플로 구현

지금까지 예로 든 코드는 매우 일반적이었습니다. 워크플로가 수행하는 작업과 워크플로를 구현하는 데 필요한 활동을 정의하기 위해 시작할 부분입니다.

주제

- 워크플로 설계
- 워크플로 코드 설정
- 워크플로 등록

워크플로 설계

기억해 보면 워크플로에 대한 최초 아이디어는 다음 단계로 구성되어 있습니다.

1. 사용자로부터 구독 주소(이메일 또는 SMS)를 얻습니다.
2. SNS 주제를 생성하고 해당 주제에 대해 제공된 엔드포인트를 구독합니다.
3. 사용자가 구독을 확인할 때까지 기다립니다.
4. 사용자가 확인하면 주제에 축하 메시지를 게시합니다.

워크플로의 각 단계를 수행해야 하는 활동이라고 생각할 수 있습니다. 워크플로는 적절한 시점에 각 활동 예약과 활동 간 데이터 전송을 담당합니다.

이 워크플로의 경우 각 단계에 대해 개별 활동을 생성하고 자세한 이름을 지정합니다.

1. get_contact_activity
2. subscribe_topic_activity
3. wait_for_confirmation_activity
4. send_result_activity

이러한 활동은 순서대로 실행되며 각 단계의 데이터는 후속 단계에 사용됩니다.

모든 코드가 소스 파일 하나에 포함되어 있도록 애플리케이션을 설계할 수 있지만 이는 Amazon SWF가 설계된 방식과 반대로 실행됩니다. 워크플로의 경우 범위가 전체 인터넷에 걸쳐 있을 수 있기 때문에 애플리케이션을 개별 실행 파일 2개로 분할해 보겠습니다.

- `swf_sns_workflow.rb` - 워크플로와 워크플로 시작자를 포함합니다.
- `swf_sns_activities.rb` - 활동과 활동 시작자를 포함합니다.

워크플로와 활동 구현은 별도의 창, 별도의 컴퓨터 또는 전 세계의 여러 지역에서 실행할 수 있습니다. Amazon SWF는 워크플로 및 활동의 세부 정보를 추적하기 때문에 워크플로는 활동이 실행되는 위치와 상관 없이 활동 예약 및 활동의 데이터 전송을 조정할 수 있습니다.

워크플로 코드 설정

swf_sns_workflow.rb라는 파일을 생성하여 시작합니다. 이 파일에서 SampleWorkflow라는 클래스를 선언합니다. 다음은 클래스 선언 및 해당 클래스의 생성자인 initialize 메서드입니다.

```
require_relative 'utils.rb'

# SampleWorkflow - the main workflow for the SWF/SNS Sample
#
# See the file called `README.md` for a description of what this file does.
class SampleWorkflow

  attr_accessor :name

  def initialize(workflowId)

    # the domain to look for decision tasks in.
    @domain = init_domain

    # the task list is used to poll for decision tasks.
    @workflowId = workflowId

    # The list of activities to run, in order. These name/version hashes can be
    # passed directly to AWS::SimpleWorkflow::DecisionTask#schedule_activity_task.
    @activity_list = [
      { :name => 'get_contact_activity', :version => 'v1' },
      { :name => 'subscribe_topic_activity', :version => 'v1' },
      { :name => 'wait_for_confirmation_activity', :version => 'v1' },
      { :name => 'send_result_activity', :version => 'v1' },
    ].reverse! # reverse the order... we're treating this like a stack.

    register_workflow
  end
```

보시다시피 여기서는 다음 클래스 인스턴스 데이터를 기록합니다.

- domain - utils.rb의 init_domain에서 검색한 도메인 이름
- workflowId - initialize로 전달된 작업 목록
- activity_list - 실행할 활동의 이름 및 버전이 포함된 활동 목록

Amazon SWF는 도메인 이름, 활동 이름 및 활동 버전만 있으면 활동 유형을 충분히 식별할 수 있으므로 활동을 예약하기 위해 활동에 대해 이러한 데이터만 기록하면 됩니다.

작업 목록은 워크플로의 결정자 코드에서 결정 작업을 폴링하고 활동을 예약하는 데 사용합니다.

이 함수의 마지막에서는 아직 정의하지 않은 메서드인 `register_workflow`를 호출합니다. 이어서 이 메서드를 정의하겠습니다.

워크플로 등록

워크플로 유형을 사용하려면 먼저 등록해야 합니다. 활동 유형과 마찬가지로, 워크플로 유형은 도메인, 이름 및 버전별로 식별됩니다. 또한 도메인 및 활동 유형처럼 기존 워크플로 유형은 다시 등록할 수 없습니다. 워크플로 유형에 대한 일부 정보를 변경해야 하는 경우 워크플로에 새 버전을 제공해야 하는데 즉, 기본적으로 새로운 유형을 만들어야 합니다.

다음은 `register_workflow`에 대한 코드로, 이전 실행에서 등록한 기존 워크플로 유형을 검색하거나 아직 등록되지 않은 워크플로를 등록하는 데 사용됩니다.

```
# Registers the workflow
def register_workflow
  workflow_name = 'swf-sns-workflow'
  @workflow_type = nil

  # a default value...
  workflow_version = '1'

  # Check to see if this workflow type already exists. If so, use it.
  @domain.workflow_types.each do | a |
    if (a.name == workflow_name) && (a.version == workflow_version)
      @workflow_type = a
    end
  end

  if @workflow_type.nil?
    options =  {
      :default_child_policy => :terminate,
      :default_task_start_to_close_timeout => 3600,
      :default_execution_start_to_close_timeout => 24 * 3600 }

    puts "registering workflow: #{workflow_name}, #{workflow_version},
#{options.inspect}"
    @workflow_type = @domain.workflow_types.register(workflow_name, workflow_version,
options)
```

```
    end

    puts "** registered workflow: #{workflow_name}"
  end
```

먼저 도메인의 workflow_types 모음을 반복해 워크플로우의 이름과 버전이 이미 등록되어 있는지 확인합니다. 일치하는 항목을 찾으면 이미 등록되어 있는 그 워크플로 유형을 사용합니다.

일치하는 항목을 찾지 못하면 워크플로를 검색하던 동일한 `workflow_types` 컬렉션에서 register를 호출하여 'swf-sns-workflow', 버전 '1' 및 다음 옵션을 사용한 새로운 워크플로 유형이 등록됩니다.

```
    options =  {
      :default_child_policy => :terminate,
      :default_task_start_to_close_timeout => 3600,
      :default_execution_start_to_close_timeout => 24 * 3600 }
```

등록 중 전달된 옵션을 사용하여 해당 워크플로 유형의 기본 동작을 설정하며, 따라서 새 워크플로 실행을 시작할 때마다 이러한 값을 설정할 필요는 없습니다.

여기서는 작업 시작 시점부터 작업이 닫힌 시점까지 걸릴 수 있는 최대 시간(1시간), 워크플로 실행을 완료하는 데 걸릴 수 있는 최대 시간(24시간) 등 제한 시간 값 몇 개를 설정합니다. 이러한 시간 중 하나가 초과되면 작업 또는 워크플로가 시간 초과됩니다.

제한 시간 값에 대한 자세한 내용은 Amazon SWF 제한 시간 유형 단원을 참조하십시오.

결정 폴링

모든 워크플로 실행의 핵심은 결정자입니다. 결정자는 워크플로 자체의 실행 관리를 담당합니다. 결정자는 결정 작업을 수신한 다음 새 활동을 예약해 활동을 다시 시작하거나 워크플로 실행 상태를 완료, 취소됨 또는 실패로 설정해 결정 작업에 응답합니다.

결정자는 워크플로 실행의 작업 목록 이름을 사용해 응답할 결정 작업을 수신합니다. 결정 작업을 폴링하려면 도메인의 decision_tasks 컬렉션에 대해 폴링을 직접적으로 호출하여 사용 가능한 결정 작업을 반복합니다. 그런 다음 결정 작업의 new_events 모음에 대해 반복해 결정 작업에 새 이벤트가 있는지 확인할 수 있습니다.

반환된 이벤트는 AWS::SimpleWorkflow::HistoryEvent 객체이고 반환된 이벤트의 event_type 멤버를 사용해 이벤트 유형을 가져올 수 있습니다. 기록 이벤트 유형의 목록 및 설명은 Amazon Simple Workflow Service API 참조의 HistoryEvent를 참조하십시오.

다음은 결정 작업 Poller 로직의 시작 부분으로, 워크플로 클래스 `poll_for_decisions`의 새 메서드입니다.

```
def poll_for_decisions
  # first, poll for decision tasks...
  @domain.decision_tasks.poll(@workflowId) do | task |
    task.new_events.each do | event |
      case event.event_type
```

이제, 수신된 `event_type`을 기반으로 결정자의 실행을 분기합니다. 처음 수신할 수 있는 이벤트 유형은 WorkflowExecutionStarted입니다. 이 이벤트가 수신되면 Amazon SWF에서 결정자에게 워크플로 실행을 시작해야 한다는 신호를 보낸 것입니다. 폴링 중 수신한 작업에 대해 schedule_activity_task를 호출해 첫 번째 활동을 예약하는 것부터 시작합니다.

활동 목록에서 선언한 첫 번째 활동을 결정자에게 전달합니다. 활동 목록을 스택처럼 사용할 수 있도록 활동 목록을 반전시켰으므로 첫 번째 활동이 목록의 `last` 위치를 차지합니다. 정의한 "활동"은 이름 및 버전 번호로 구성된 맵인데, 활동이 이미 등록되어 있다고 가정하면 이러한 맵이 Amazon SWF에서 예약을 위해 활동을 식별하는 데 필요한 정보로 충분합니다.

```
        when 'WorkflowExecutionStarted'
          # schedule the last activity on the (reversed, remember?) list to
          # begin the workflow.
          puts "** scheduling activity task: #{@activity_list.last[:name]}"

          task.schedule_activity_task( @activity_list.last,
            { :workflowId => "#{@workflowId}-activities" } )
```

활동을 예약하면 Amazon SWF는 활동 예약 중 전달한 활동 작업 목록으로 활동 작업을 보내 해당 작업을 시작하라고 신호를 보냅니다. 구독 워크플로 자습서 파트 3: 활동 구현 단원의 활동 작업을 처리할 예정이지만 여기서는 이 작업을 실행하지는 않습니다. 여기서는 Amazon SWF에 해당 작업을 예약해야 한다고 알리기만 합니다.

처리해야 하는 다음 활동은 ActivityTaskCompleted 이벤트로, Amazon SWF에서 활동 작업의 활동 완료 응답을 수신한 경우 발생합니다.

```
        when 'ActivityTaskCompleted'
          # we are running the activities in strict sequential order, and
          # using the results of the previous activity as input for the next
          # activity.
          last_activity = @activity_list.pop
```

```
            if(@activity_list.empty?)
              puts "!! All activities complete! Sending complete_workflow_execution..."
              task.complete_workflow_execution
              return true;
            else
              # schedule the next activity, passing any results from the
              # previous activity. Results will be received in the activity
              # task.
              puts "** scheduling activity task: #{@activity_list.last[:name]}"
              if event.attributes.has_key?('result')
                task.schedule_activity_task(
                  @activity_list.last,
                  { :input => event.attributes[:result],
                    :workflowId => "#{@workflowId}-activities" } )
              else
                task.schedule_activity_task(
                  @activity_list.last, { :workflowId => "#{@workflowId}-activities" } )
              end
            end
```

선형 방식으로 작업을 실행하고 한 번에 하나의 활동만 실행하므로이 기회를 통해 `activity_list` 스택에서 완료된 작업을 팝업합니다. 그러면 목록이 비게 되고 워크플로가 완료되었음을 알게 됩니다. 이 경우에는 작업에 대해 complete_workflow_execution을 호출하여 워크플로가 완료되었음을 Amazon SWF에 알립니다.

목록에 아직 항목이 있는 경우 목록의 (마지막 위치에 있는) 다음 활동을 예약합니다. 그러나 이 경우에는 완료 시 이전 활동에서 Amazon SWF에 결과 데이터를 반환했는지 살펴볼 것입니다. 이러한 데이터는 이벤트 속성의 선택적 `result` 키에서 워크플로에 제공됩니다. 활동이 결과를 생성하면 활동 작업 목록과 함께 이러한 결과를 다음 예약된 활동에 `input` 옵션으로 전달합니다.

완료된 활동의 `result` 값을 검색하고 예약된 활동의 `input` 값을 설정하여 하나의 활동에서 다음 활동으로 데이터를 전달하거나 한 활동의 데이터를 사용해 활동의 결과를 바탕으로 결정자의 동작을 변경할 수 있습니다.

이 자습서의 용도에 맞춰 워크플로 동작을 정의할 때 이러한 두 가지 이벤트 유형이 가장 중요합니다. 그러나 활동이 ActivityTaskCompleted 이외의 이벤트를 생성할 수 있습니다. ActivityTaskTimedOut 및 ActivityTaskFailed 이벤트와 실행할 활동이 부족할 때 Amazon SWF가 `complete_workflow_execution` 직접 호출을 처리할 때 생성되는 WorkflowExecutionCompleted 이벤트에 대한 데모 핸들러 코드를 제공하여 결정자 코드를 마무리합니다.

```
          when 'ActivityTaskTimedOut'
```

```
            puts "!! Failing workflow execution! (timed out activity)"
            task.fail_workflow_execution
            return false

          when 'ActivityTaskFailed'
            puts "!! Failing workflow execution! (failed activity)"
            task.fail_workflow_execution
            return false

          when 'WorkflowExecutionCompleted'
            puts "## Yesss, workflow execution completed!"
            task.workflow_execution.terminate
            return false
        end
      end
    end
  end
```

워크플로 실행 시작

워크플로에서 폴링할 결정 작업이 생성되기 전에 워크플로 실행을 시작해야 합니다.

워크플로 실행을 시작하려면 등록된 워크플로 유형(AWS::SimpleWorkflow::WorkflowType)에서 start_execution을 직접적으로 호출하십시오. 클래스 생성자에서 검색한 `workflow_type` 인스턴스 멤버를 사용하기 위해 이러한 호출과 관련된 작은 래퍼를 정의할 것입니다.

```
  def start_execution
    workflow_execution = @workflow_type.start_execution( {
      :workflowId => @workflowId } )
    poll_for_decisions
  end
end
```

워크플로우가 실행되면 결정 이벤트가 시작되어 워크플로우의 작업 목록에 나타나는데, 결정 이벤트는 start_execution에서 워크플로우 실행 옵션으로 전달됩니다.

워크플로 유형 등록 시 제공된 옵션과 달리 `start_execution`에 전달된 옵션은 워크플로 유형의 일부로 간주되지 않습니다. 이러한 옵션은 워크플로 버전을 변경하지 않고 워크플로 실행당 자유롭게 변경할 수 있습니다.

파일을 실행할 때 워크플로 실행을 시작하려고 하므로 클래스를 인스턴스화한 다음 방금 정의한 `start_execution` 메서드를 호출하는 코드를 추가합니다.

```
if __FILE__ == $0
  require 'securerandom'

  # Use a different task list name every time we start a new workflow execution.
  #
  # This avoids issues if our pollers re-start before SWF considers them closed,
  # causing the pollers to get events from previously-run executions.
  workflowId = SecureRandom.uuid

  # Let the user start the activity worker first...

  puts ""
  puts "Amazon SWF Example"
  puts "------------------"
  puts ""
  puts "Start the activity worker, preferably in a separate command-line window, with"
  puts "the following command:"
  puts ""
  puts "> ruby swf_sns_activities.rb #{workflowId}-activities"
  puts ""
  puts "You can copy & paste it if you like, just don't copy the '>' character."
  puts ""
  puts "Press return when you're ready..."

  i = gets

  # Now, start the workflow.

  puts "Starting workflow execution."
  sample_workflow = SampleWorkflow.new(workflowId)
  sample_workflow.start_execution
end
```

작업 목록 이름이 충돌하지 않도록 SecureRandom.uuid를 사용해 작업 목록 이름으로 사용할 수 있는 랜덤 UUID를 생성하여 워크플로 실행마다 다른 작업 목록 이름이 사용되도록 할 것입니다.

> **Note**
>
> 작업 목록은 워크플로 실행에 대한 이벤트를 기록하는 데 사용됩니다. 따라서 같은 워크플로 유형의 여러 실행에 대해 동일한 작업 목록을 사용하면 이전 실행 중 생성된 이벤트를 수신할

수 있습니다. 워크플로 유형을 거의 연속으로 실행하는 경우에는 특히 더 그렇고, 새 코드를 시험해 보거나 테스트를 실행하는 경우가 종종 여기에 해당합니다.

이전 실행의 결과물을 처리해야 하는 문제를 방지하기 위해 각 실행마다 새 작업 목록을 사용할 수 있습니다. 워크플로 실행을 시작할 때 새 작업 목록을 지정하면 됩니다.

또한 여기에는 워크플로를 실행하는 사람(아마도 여러분)에게 지침을 제공하고 작업 목록의 "활동" 버전을 제공하는 코드도 있습니다. 결정자는 작업 목록 이름을 사용해 워크플로에 대한 활동을 예약하고 활동 구현 시 해당 작업 목록 이름에 대한 활동 이벤트를 수신해 예약된 활동이 시작된 시점과 활동 실행에 대한 업데이트가 제공된 시점을 파악합니다.

또한 코드는 자신이 워크플로 실행을 시작하기 전에 사용자가 활동 시작자를 실행하도록 대기합니다. 그러면 활동 시작자는 제공된 작업 목록에 나타나는 활동 작업이 시작할 때 응답할 준비가 됩니다.

다음 단계

이제 워크플로를 구현했습니다. 다음으로 구독 워크플로 자습서 파트 3: 활동 구현 단원에서 활동과 활동 시작자를 정의해 보겠습니다.

구독 워크플로 자습서 파트 3: 활동 구현

활동 코드에 일반적인 몇 가지 기능을 제공하는 기본 클래스부터 시작해 워크플로의 각 활동을 구현하려고 합니다.

주제

기본 활동 유형 정의

워크플로를 설계할 때 다음 활동을 식별했습니다.

- get_contact_activity
- subscribe_topic_activity
- wait_for_confirmation_activity
- send_result_activity

이제 이러한 각 활동을 구현합니다. 활동은 몇 가지 기능을 공유하므로 몇 가지 토대를 마련하고 공유할 수 있는 몇 가지 공통 코드를 만들어 보겠습니다. 이러한 공통 코드를 BasicActivity라고 하고 basic_activity.rb라는 새 파일에 정의해 보겠습니다.

다른 소스 파일에서처럼 utils.rb를 포함해 샘플 도메인을 설정하는 init_domain 함수에 액세스합니다.

```
require_relative 'utils.rb'
```

다음으로, 기본 활동 클래스와 각 활동에서 관심을 둘 몇 가지 공통 데이터를 선언합니다. 클래스의 속성에 활동의 AWS::SimpleWorkflow::ActivityType 인스턴스, 이름 및 결과를 저장합니다.

```
class BasicActivity

  attr_accessor :activity_type
  attr_accessor :name
  attr_accessor :results
```

이러한 속성은 활동 이름과 Amazon SWF에 활동을 등록할 때 사용할 선택 버전 및 옵션 맵을 사용하는 클래스의 initialize 메서드에 정의된 인스턴스 데이터에 액세스합니다.

```
  def initialize(name, version = 'v1', options = nil)

    @activity_type = nil
    @name = name
    @results = nil

    # get the domain to use for activity tasks.
    @domain = init_domain

    # Check to see if this activity type already exists.
    @domain.activity_types.each do | a |
      if (a.name == @name) && (a.version == version)
        @activity_type = a
      end
```

```
      end

      if @activity_type.nil?
        # If no options were specified, use some reasonable defaults.
        if options.nil?
          options = {
            # All timeouts are in seconds.
            :default_task_heartbeat_timeout => 900,
            :default_task_schedule_to_start_timeout => 120,
            :default_task_schedule_to_close_timeout => 3800,
            :default_task_start_to_close_timeout => 3600 }
        end
        @activity_type = @domain.activity_types.register(@name, version, options)
      end
    end
```

워크플로우 유형을 등록할 때처럼 활동 유형이 이미 등록된 경우에는 도메인의 activity_types 컬렉션을 살펴보고 활동 유형을 검색할 수 있습니다. 활동을 찾을 수 없는 경우에는 등록합니다.

또한 워크플로 유형처럼 등록 시 활동 유형과 함께 저장될 기본 옵션을 설정할 수 있습니다.

기본 활동으로 가져올 마지막 항목은 일관된 실행 방식입니다. 활동 작업을 가져오는 `do_activity` 메서드를 정의합니다. 표시된 것처럼 전달된 활동 작업을 사용해 `input` 인스턴스 속성을 통해 데이터를 수신할 수 있습니다.

```
    def do_activity(task)
      @results = task.input # may be nil
      return true
    end
  end
```

이 코드는 BasicActivity 클래스를 마무리합니다. 이제 이 코드를 사용해 활동 정의를 더욱 간단하고 일관되게 합니다.

GetContactActivity 정의

워크플로 실행 중 가장 먼저 실행되는 활동은 `get_contact_activity`로, 사용자의 Amazon SNS 주제 구독 정보를 검색합니다.

`get_contact_activity.rb`라는 새 파일을 생성하고 Amazon SWF에 전달할 문자열을 준비하는 데 사용할 `yaml`과 이 GetContactActivity 클래스의 기초로 사용할 `basic_activity.rb`가 모두 필요합니다.

```
require 'yaml'
require_relative 'basic_activity.rb'

# **GetContactActivity** provides a prompt for the user to enter contact
# information. When the user successfully enters contact information, the
# activity is complete.
class GetContactActivity < BasicActivity
```

활동 등록 코드를 BasicActivity에 넣기 때문에 GetContactActivity의 `initialize` 방법은 매우 간단합니다. 활동 이름 `get_contact_activity`를 사용해 기본 클래스 생성자를 호출하기만 하면 됩니다. 활동을 등록하는 데 이렇게 하기만 하면 됩니다.

```
    # initialize the activity
    def initialize
      super('get_contact_activity')
    end
```

이제 사용자의 이메일 및/또는 전화번호를 입력하라고 표시하는 `do_activity` 메서드를 정의합니다.

```
    def do_activity(task)
      puts ""
      puts "Please enter either an email address or SMS message (mobile phone) number
to"
      puts "receive SNS notifications. You can also enter both to use both address
types."
      puts ""
      puts "If you enter a phone number, it must be able to receive SMS messages, and
must"
      puts "be 11 digits (such as 12065550101 to represent the number
1-206-555-0101)."

      input_confirmed = false
      while !input_confirmed
        puts ""
        print "Email: "
        email = $stdin.gets.strip

        print "Phone: "
        phone = $stdin.gets.strip

        puts ""
        if (email == '') && (phone == '')
```

```
          print "You provided no subscription information. Quit? (y/n)"
           confirmation = $stdin.gets.strip.downcase
           if confirmation == 'y'
             return false
           end
        else
           puts "You entered:"
           puts "  email: #{email}"
           puts "  phone: #{phone}"
           print "\nIs this correct? (y/n): "
           confirmation = $stdin.gets.strip.downcase
           if confirmation == 'y'
             input_confirmed = true
           end
        end
      end

      # make sure that @results is a single string. YAML makes this easy.
      @results = { :email => email, :sms => phone }.to_yaml
      return true
    end
  end
```

do_activity 끝에서는 사용자로부터 검색한 이메일 및 전화번호를 가져와 맵에 배치한 다음 to_yaml을 사용해 전체 맵을 YAML 문자열로 변환합니다. 이렇게 변환하는 데에는 중요한 이유가 있는데, 활동 완료 시 Amazon SWF에 전달하는 모든 결과는 문자열 데이터여야 하기 때문입니다. 객체를 YAML 문자열로 변환한 다음 다시 객체로 쉽게 변환하는 Ruby의 기능은 다행스럽게도 이러한 용도에 잘 맞습니다.

get_contact_activity 구현을 마칩니다. 이러한 데이터는 다음에 subscribe_topic_activity 구현 시 사용됩니다.

SubscribeTopicActivity 정의

이제 Amazon SNS에 대해 좀 더 자세히 알아보고 get_contact_activity가 생성한 정보를 사용해 Amazon SNS 주제를 구독하도록 사용자를 등록하는 활동을 생성합니다.

subscribe_topic_activity.rb라는 새 파일을 생성하고, get_contact_activity에 사용한 것과 동일한 요구 사항을 추가하고, 클래스를 선언한 다음 initialize 메서드를 제공합니다.

```
  require 'yaml'
```

```
require_relative 'basic_activity.rb'

# **SubscribeTopicActivity** sends an SMS / email message to the user, asking for
# confirmation.  When this action has been taken, the activity is complete.
class SubscribeTopicActivity < BasicActivity

  def initialize
    super('subscribe_topic_activity')
  end
```

지금까지 활동을 설정 및 등록하는 코드를 준비했으므로 일부 코드를 추가하여 Amazon SNS 주제를 생성할 차례입니다. 주제를 생성하려면 AWS::SNS::Client 객체의 create_topic 메서드를 사용합니다.

전달된 Amazon SNS 클라이언트 객체를 가져오는 `create_topic` 메서드를 클래스에 추가합니다.

```
  def create_topic(sns_client)
    topic_arn = sns_client.create_topic(:name => 'SWF_Sample_Topic')[:topic_arn]

    if topic_arn != nil
      # For an SMS notification, setting `DisplayName` is *required*. Note that
      # only the *first 10 characters* of the DisplayName will be shown on the
      # SMS message sent to the user, so choose your DisplayName wisely!
      sns_client.set_topic_attributes( {
        :topic_arn => topic_arn,
        :attribute_name => 'DisplayName',
        :attribute_value => 'SWFSample' } )
    else
      @results = {
        :reason => "Couldn't create SNS topic", :detail => "" }.to_yaml
      return nil
    end

    return topic_arn
  end
```

주제의 Amazon 리소스 이름(ARN)이 있으면 클라이언트의 set_topic_attributes 메서드에 이 이름을 사용해 주제의 DisplayName을 설정합니다. 이 이름은 Amazon SNS에서 SMS 메시지를 전송하는 데 필요합니다.

마지막으로 `do_activity` 메서드를 정의합니다. 활동을 예약할 때 `input` 옵션을 통해 전달한 데이터를 모두 수집해 시작합니다. 앞서 언급한 것처럼 데이터는 `to_yaml`을 사용해 생성한 문자열로 전달해야 합니다. 데이터를 검색하면 `YAML.load`를 사용해 데이터를 Ruby 객체로 전환합니다.

다음은 입력 데이터를 검색하는 do_activity의 시작 부분입니다.

```
    def do_activity(task)
      activity_data = {
        :topic_arn => nil,
        :email => { :endpoint => nil, :subscription_arn => nil },
        :sms => { :endpoint => nil, :subscription_arn => nil },
      }

      if task.input != nil
        input = YAML.load(task.input)
        activity_data[:email][:endpoint] = input[:email]
        activity_data[:sms][:endpoint] = input[:sms]
      else
        @results = { :reason => "Didn't receive any input!", :detail => "" }.to_yaml
        puts("  #{@results.inspect}")
        return false
      end

      # Create an SNS client. This is used to interact with the service. Set the
      # region to $SMS_REGION, which is a region that supports SMS notifications
      # (defined in the file `utils.rb`).
      sns_client = AWS::SNS::Client.new(
        :config => AWS.config.with(:region => $SMS_REGION))
```

입력 데이터를 검색하지 못한 경우 수행할 작업이 별로 없으므로 활동을 실패로 처리합니다.

하지만 모든 것이 괜찮다고 가정하면 계속해서 do_activity 메서드를 작성하고,를 사용하여 Amazon SNS 클라이언트를 가져오고 AWS SDK for Ruby, 이를 create_topic 메서드에 전달하여 Amazon SNS 주제를 생성합니다.

```
      # Create the topic and get the ARN
      activity_data[:topic_arn] = create_topic(sns_client)

      if activity_data[:topic_arn].nil?
        return false
      end
```

여기서 짚고 넘어가야 할 부분이 몇 가지 있습니다.

- Amazon SNS 클라이언트의 리전을 설정하는 데 AWS.config.with를 사용합니다. SMS 메시지를 전송해야 하기 때문에 utils.rb에 선언한 SMS 지원 리전을 사용합니다.

- activity_data 맵에 주제의 ARN을 저장합니다. ARN은 워크플로 내 다음 활동으로 전달될 데이터의 일부입니다.

마지막으로, 이 활동은 전달된 엔드포인트(이메일 및 SMS)를 사용해 Amazon SNS 주제를 구독하도록 사용자를 가입합니다. 사용자가 엔드포인트를 둘 다 입력할 필요는 없지만 둘 중 하나는 필요합니다.

```
      # Subscribe the user to the topic, using either or both endpoints.
      [:email, :sms].each do | x |
        ep = activity_data[x][:endpoint]
        # don't try to subscribe an empty endpoint
        if (ep != nil && ep != "")
          response = sns_client.subscribe( {
            :topic_arn => activity_data[:topic_arn],
            :protocol => x.to_s, :endpoint => ep } )
          activity_data[x][:subscription_arn] = response[:subscription_arn]
        end
      end
```

AWS::SNS::Client.subscribe는 주제 ARN과 프로토콜(해당 엔드포인트에 대한 activity_data 맵 키로 지능적으로 위장함)을 가져옵니다.

마지막으로 다음 활동에 대한 정보를 YAML 형식으로 다시 패키징합니다. 그러면 해당 정보를 Amazon SWF로 다시 보낼 수 있습니다.

```
      # if at least one subscription arn is set, consider this a success.
      if (activity_data[:email][:subscription_arn] != nil) or (activity_data[:sms]
[:subscription_arn] != nil)
        @results = activity_data.to_yaml
      else
        @results = { :reason => "Couldn't subscribe to SNS topic", :detail =>
 "" }.to_yaml
        puts("  #{@results.inspect}")
        return false
      end
      return true
    end
  end
```

이것으로 subscribe_topic_activity 구현을 마칩니다. 다음으로, wait_for_confirmation_activity를 정의합니다.

WaitForConfirmationActivity 정의

사용자가 Amazon SNS 주제를 구독하도록 가입되면 사용자는 구독 요청을 확인해야 합니다. 이 경우 사용자가 이메일 또는 SMS 메시지로 확인할 때까지 대기합니다.

사용자가 구독을 확인하도록 대기하는 활동을 `wait_for_confirmation_activity`라고 하고 여기서 이 활동을 정의합니다. 시작하려면 `wait_for_confirmation_activity.rb`라는 새 파일을 생성해 이전 활동을 설정한 것처럼 설정합니다.

```
require 'yaml'
require_relative 'basic_activity.rb'

# **WaitForConfirmationActivity** waits for the user to confirm the SNS
# subscription.  When this action has been taken, the activity is complete. It
# might also time out...
class WaitForConfirmationActivity < BasicActivity

  # Initialize the class
  def initialize
    super('wait_for_confirmation_activity')
  end
```

다음으로, `do_activity` 메서드 정의를 시작하고 `subscription_data`라는 로컬 변수로 입력 데이터를 가져옵니다.

```
  def do_activity(task)
    if task.input.nil?
      @results = { :reason => "Didn't receive any input!", :detail => "" }.to_yaml
      return false
    end

    subscription_data = YAML.load(task.input)
```

이제 주제 ARN이 있으므로 AWS::SNS::Topic의 새 인스턴스를 생성해 주제를 검색하여 해당 주제에 ARN을 전달합니다.

```
    topic = AWS::SNS::Topic.new(subscription_data[:topic_arn])

    if topic.nil?
      @results = {
        :reason => "Couldn't get SWF topic ARN",
```

```
          :detail => "Topic ARN: #{topic.arn}" }.to_yaml
        return false
      end
```

이제, 엔드포인트 중 하나를 사용해 사용자가 구독을 확인했는지 살펴보기 위해 주제를 확인합니다. 활동을 성공으로 간주하기 위해서는 엔드포인트 하나만 확인되면 됩니다.

Amazon SNS 주제는 해당 주제에 대한 구독 목록을 유지하고, 구독의 ARN이 `PendingConfirmation` 외의 다른 값으로 설정되어 있는지 확인하여 사용자가 특정 구독을 확인했는지 여부를 알아볼 수 있습니다.

```
      # loop until we get some indication that a subscription was confirmed.
      subscription_confirmed = false
      while(!subscription_confirmed)
        topic.subscriptions.each do | sub |
          if subscription_data[sub.protocol.to_sym][:endpoint] == sub.endpoint
            # this is one of the endpoints we're interested in. Is it subscribed?
            if sub.arn != 'PendingConfirmation'
              subscription_data[sub.protocol.to_sym][:subscription_arn] = sub.arn
              puts "Topic subscription confirmed for (#{sub.protocol}:
#{sub.endpoint})"
              @results = subscription_data.to_yaml
              return true
            else
              puts "Topic subscription still pending for (#{sub.protocol}:
#{sub.endpoint})"
            end
          end
        end
```

구독에 대한 ARN을 얻을 경우 활동의 결과 데이터에 저장해 YAML로 변환하고 `do_activity`에서 true를 반환합니다. 그러면 활동이 성공적으로 완료되었다는 신호를 보냅니다.

구독이 확인될 때까지 기다리는 데 시간이 걸릴 수 있으므로 가끔 활동 작업`record_heartbeat`에서를 호출합니다. 그러면 Amazon SWF에 활동이 계속 처리 중이고 해당 활동을 사용해 활동 진행 상황에 대한 업데이트를 제공할 수 있음을 알리는 신호를 보냅니다(진행 상황을 보고할 수 있는 파일 처리와 같은 작업을 수행하는 경우).

```
        task.record_heartbeat!(
          { :details => "#{topic.num_subscriptions_confirmed} confirmed,
#{topic.num_subscriptions_pending} pending" })
```

```
        # sleep a bit.
        sleep(4.0)
      end
```

이것으로 while 루프를 마칩니다. 성공 없이 while 루프에서 빠져나오면 실패를 보고하고 do_activity 메서드를 마칩니다.

```
      if (subscription_confirmed == false)
        @results = {
          :reason => "No subscriptions could be confirmed",
          :detail => "#{topic.num_subscriptions_confirmed} confirmed,
#{topic.num_subscriptions_pending} pending" }.to_yaml
        return false
      end
    end
  end
```

이것으로 wait_for_confirmation_activity 구현을 마칩니다. 정의할 활동이 send_result_activity 하나만 남았습니다.

SendResultActivity 정의

워크플로를 여기까지 진행했으면 Amazon SNS 주제를 구독하도록 사용자를 성공적으로 가입했고 사용자는 구독을 확인한 것입니다.

마지막 활동 send_result_activity는 사용자가 구독한 주제와 구독을 확인한 엔드포인트를 사용해 사용자에게 성공적인 주제 구독 확인을 보냅니다.

send_result_activity.rb라는 새 파일을 생성해 지금까지 모든 활동을 설정한 것처럼 설정합니다.

```
require 'yaml'
require_relative 'basic_activity.rb'

# **SendResultActivity** sends the result of the activity to the screen, and, if
# the user successfully registered using SNS, to the user using the SNS contact
# information collected.
class SendResultActivity < BasicActivity

  def initialize
    super('send_result_activity')
```

```
    end
```

do_activity 메서드 역시 유사하게 시작되어 워크플로우에서 입력 데이터를 가져와 YAML에서 변환한 다음 주제 ARN을 사용해 AWS::SNS::Topic 인스턴스를 생성합니다.

```
    def do_activity(task)
      if task.input.nil?
        @results = { :reason => "Didn't receive any input!", :detail => "" }
        return false
      end

      input = YAML.load(task.input)

      # get the topic, so we publish a message to it.
      topic = AWS::SNS::Topic.new(input[:topic_arn])

      if topic.nil?
        @results = {
          :reason => "Couldn't get SWF topic",
          :detail => "Topic ARN: #{topic.arn}" }
        return false
      end
```

주제가 있으면 그 주제에 메시지를 게시합니다(화면에도 동일하게 반영).

```
      @results = "Thanks, you've successfully confirmed registration, and your
workflow is complete!"

      # send the message via SNS, and also print it on the screen.
      topic.publish(@results)
      puts(@results)

      return true
    end
  end
```

Amazon SNS 주제에 게시하면 해당 주제에 대해 존재하는 모든 구독 및 확인 엔드포인트에 제공한 메시지가 전송됩니다. 따라서 사용자가 이메일 및 SMS 번호 둘 다로 확인한 경우 사용자는 각 엔드포인트에 대해 하나씩 확인 메시지 2개를 수신합니다.

다음 단계

이것으로 `send_result_activity` 구현을 마칩니다. 이제, 활동 작업을 처리해 구독 워크플로 자습서 파트 4: 활동 작업 Poller 구현 단원에서의 응답으로 활동을 시작할 수 있는 활동 애플리케이션에서 이러한 모든 활동을 함께 연결합니다.

구독 워크플로 자습서 파트 4: 활동 작업 Poller 구현

Amazon SWF에서는 실행 중인 워크플로 실행의 활동 작업이 워크플로의 활동을 예약할 때 제공되는 활동 작업 목록에 나타납니다.

여기서는 이러한 워크플로 작업을 처리하는 기본 활동 폴러를 구현하고, Amazon SWF가 활동을 시작하기 위해 활동 작업 목록에 작업을 배치하면 이 폴러를 사용해 그 활동을 시작합니다.

시작하려면 `swf_sns_activities.rb`라는 새 파일을 생성합니다. 이 파일을 사용해 다음 작업을 수행합니다.

- 생성한 활동 클래스를 인스턴스화합니다.
- Amazon SWF에 각 활동을 등록합니다.
- 활동을 폴링하고, 활동 작업 목록에 활동 이름이 나타나면 각 활동에 대해 `do_activity`를 호출합니다.

`swf_sns_activities.rb`에서 다음 명령문을 추가하여 정의한 각 활동 클래스를 요구합니다.

```
require_relative 'get_contact_activity.rb'
require_relative 'subscribe_topic_activity.rb'
require_relative 'wait_for_confirmation_activity.rb'
require_relative 'send_result_activity.rb'
```

이제 클래스를 생성하고 초기화 코드 몇 개를 제공하겠습니다.

```
class ActivitiesPoller

  def initialize(domain, workflowId)
    @domain = domain
    @workflowId = workflowId
    @activities = {}

    # These are the activities we'll run
```

```
    activity_list = [
      GetContactActivity,
      SubscribeTopicActivity,
      WaitForConfirmationActivity,
      SendResultActivity ]

    activity_list.each do | activity_class |
      activity_obj = activity_class.new
      puts "** initialized and registered activity: #{activity_obj.name}"
      # add it to the hash
      @activities[activity_obj.name.to_sym] = activity_obj
    end
  end
```

전달된 도메인 및 작업 목록을 저장하는 것 이외에도 이 코드는 생성한 각 활동 클래스를 인스턴스화합니다. 연결된 활동을 클래스별로 등록하기 때문에(코드를 검토해야 하는 경우 `basic_activity.rb` 참조), 이렇게 하면 실행할 모든 활동을 Amazon SWF에 알려 주기에 충분합니다.

인스턴스화된 각 활동의 경우 활동 이름(예: `get_contact_activity`)을 키로 사용해 활동을 맵에 저장합니다. 따라서 다음에 정의할 활동 Poller 코드에서 해당 활동을 쉽게 찾을 수 있습니다.

`poll_for_activities`라는 새 메서드를 만들고 도메인이 보유하는 activity_tasks에 대해 폴링을 직접적으로 호출하여 활동 작업을 가져옵니다.

```
  def poll_for_activities
    @domain.activity_tasks.poll(@workflowId) do | task |
      activity_name = task.activity_type.name
```

작업의 activity_type 멤버에서 활동 이름을 가져올 수 있습니다. 다음으로, 이 작업과 연결된 활동 이름을 사용해 `do_activity`를 실행할 클래스를 조회해 작업에 전달합니다(이 클래스에는 활동으로 전송해야 하는 입력 데이터가 포함되어 있음).

```
      # find the task on the activities list, and run it.
      if @activities.key?(activity_name.to_sym)
        activity = @activities[activity_name.to_sym]
        puts "** Starting activity task: #{activity_name}"
        if activity.do_activity(task)
          puts "++ Activity task completed: #{activity_name}"
          task.complete!({ :result => activity.results })
          # if this is the final activity, stop polling.
          if activity_name == 'send_result_activity'
             return true
```

```
          end
        else
          puts "-- Activity task failed: #{activity_name}"
          task.fail!(
            { :reason => activity.results[:reason],
              :details => activity.results[:detail] } )
        end
      else
        puts "couldn't find key in @activities list: #{activity_name}"
        puts "contents: #{@activities.keys}"
      end
    end
  end
end
```

이 코드는 do_activity가 완료될 때까지 기다린 후 반환 코드에 따라 작업에 대해 complete! 또는 fail!을 호출합니다.

> **Note**
>
> 이 코드는 최종 활동이 시작되면 폴러에서 종료됩니다. 미션을 완료하고 모든 활동을 시작했기 때문입니다. 자체 Amazon SWF 코드에서 활동을 다시 실행할 수 있는 경우 활동 Poller를 무기한으로 실행하도록 할 수 있습니다.

ActivitiesPoller 클래스에 대한 코드의 끝입니다. 하지만 사용자가 명령줄에서 이 코드를 실행할 수 있도록 하기 위해 파일 끝에서 코드를 조금 더 추가할 것입니다.

```
if __FILE__ == $0
  if ARGV.count < 1
    puts "You must supply a task-list name to use!"
    exit
  end
  poller = ActivitiesPoller.new(init_domain, ARGV[0])
  poller.poll_for_activities
  puts "All done!"
end
```

사용자가 명령줄에서 이 파일을 실행하면(이 파일을 작업 목록에 첫 번째 인수로 전달) 이 코드는 Poller 클래스를 인스턴스화하고 활동 폴링을 시작합니다. (Poller가 최종 활동을 시작한 후) Poller가 완료되면 메시지를 인쇄한 다음 종료합니다.

여기까지가 활동 Poller에 대한 내용입니다. 이제는 코드를 실행하고 구독 워크플로 자습서: 워크플로 실행에서 코드가 어떻게 작동하는지 관찰하기만 하면 됩니다.

구독 워크플로 자습서: 워크플로 실행

지금까지 워크플로, 활동과 워크플로 및 활동 Poller 구현을 완료했으므로 워크플로를 실행할 준비가 되었습니다.

아직 제공하지 않은 경우 자습서 1AWS 세션 구성부의와 같이 `aws-config.txt` 파일에 AWS 액세스 키를 제공해야 합니다.

이제, 명령줄로 이동해 자습서 소스 파일이 있는 디렉터리로 변경합니다. 다음 파일이 필요합니다.

```
.
|-- aws-config.txt
|-- basic_activity.rb
|-- get_contact_activity.rb
|-- send_result_activity.rb
|-- subscribe_topic_activity.rb
|-- swf_sns_activities.rb
|-- swf_sns_workflow.rb
|-- utils.rb
`-- wait_for_confirmation_activity.rb
```

이제, 다음 명령을 사용해 워크플로를 시작합니다.

```
ruby swf_sns_workflow.rb
```

이 명령은 워크플로를 시작하고 복사해 별도의 명령줄 창에 붙여 넣을 수 있는 한 줄짜리 메시지를 출력해야 합니다(자습서 소스 파일을 복사한 경우에는 다른 컴퓨터에도 붙여 넣을 수 있음).

```
Amazon SWF Example
------------------

Start the activity worker, preferably in a separate command-line window, with
the following command:

> ruby swf_sns_activities.rb 87097e76-7c0c-41c7-817b-92527bb0ea85-activities

You can copy & paste it if you like, just don't copy the '>' character.
```

```
Press return when you're ready...
```

이 워크플로 코드는 별도의 창에서 활동 Poller를 시작할 때까지 대기합니다.

새 명령줄 창을 열고 소스 파일이 있는 디렉터리로 다시 변경한 다음 `swf_sns_workflow.rb` 파일에서 제공한 명령을 사용해 활동 Poller를 시작합니다. 예를 들어, 앞선 출력을 수신한 경우 다음과 같이 입력(또는 붙여넣기)합니다.

```
ruby swf_sns_activities.rb 87097e76-7c0c-41c7-817b-92527bb0ea85-activities
```

활동 Poller 실행을 시작하면 활동 Poller가 활동 등록에 대한 정보를 출력하기 시작합니다.

```
** initialized and registered activity: get_contact_activity
** initialized and registered activity: subscribe_topic_activity
** initialized and registered activity: wait_for_confirmation_activity
** initialized and registered activity: send_result_activity
```

이제 원래 명령줄 창으로 돌아가 Return을 눌러 워크플로 실행을 시작할 수 있습니다. 그러면 워크플로를 등록하고 첫 번째 활동을 예약합니다.

```
Starting workflow execution.
** registered workflow: swf-sns-workflow
** scheduling activity task: get_contact_activity
```

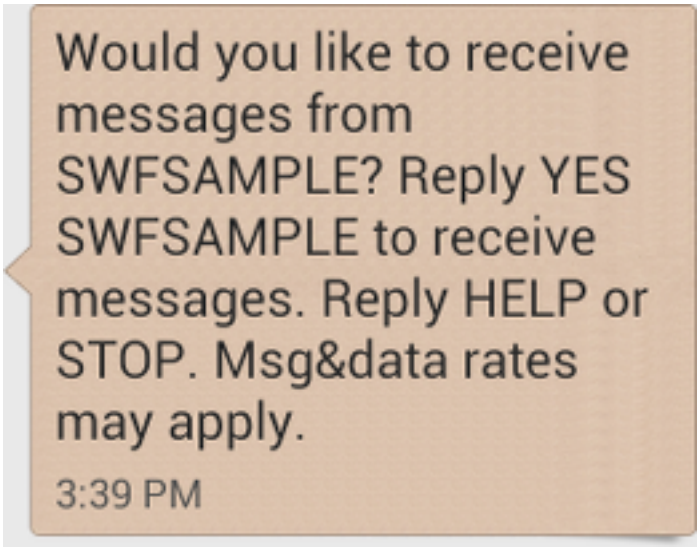
활동 Poller가 실행 중인 다른 창으로 돌아갑니다. 실행 중인 첫 번째 활동의 결과가 표시되어 이메일 및 SMS 전화번호를 입력할 수 있는 프롬프트를 제공합니다. 이러한 데이터 중 하나 또는 둘 다를 입력한 다음 텍스트 입력을 확인합니다.

```
activity task received: <AWS::SimpleWorkflow::ActivityTask>
** Starting activity task: get_contact_activity

Please enter either an email address or SMS message (mobile phone) number to
receive Amazon SNS notifications. You can also enter both to use both address types.

If you enter a phone number, it must be able to receive SMS messages, and must
be 11 digits (such as 12065550101 to represent the number 1-206-555-0101).

Email: me@example.com
Phone: 12065550101

You entered:
```

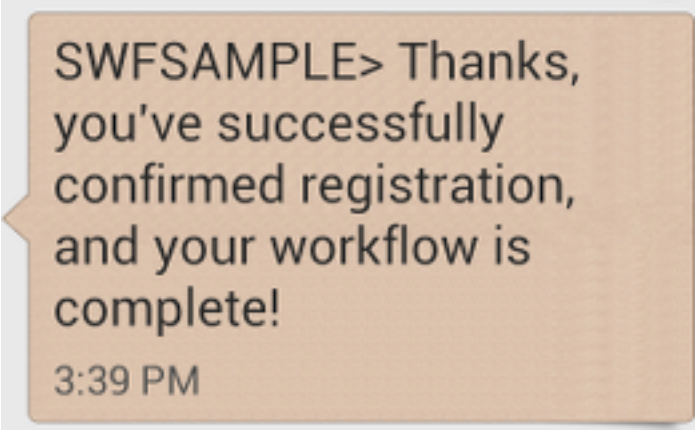
```
  email: me@example.com
  phone: 12065550101

Is this correct? (y/n): y
```

> **Note**
>
> 여기 제공된 전화번호를 가상의 번호로, 설명을 위해서만 사용됩니다. 실제로는 각자의 전화번호 및 이메일 주소를 사용하십시오!

이 정보를 입력하자마자 Amazon SNS로부터 주제 구독을 확인하라는 이메일 또는 문자 메시지가 도착해야 합니다. SMS 번호를 입력한 경우에는 휴대폰에 다음과 같은 메시지가 표시됩니다.



이 메시지에 `YES`라고 회신하면 `send_result_activity`로 제공한 응답을 받게 됩니다.



이러한 과정이 진행되는 동안 명령줄 창에서는 어떻게 진행되고 있는지 봤습니까? 워크플로 및 활동 Poller 둘 다 열심히 작업 중이었습니다.

워크플로 Poller에서는 다음과 같이 출력됩니다.

```
** scheduling activity task: subscribe_topic_activity
** scheduling activity task: wait_for_confirmation_activity
** scheduling activity task: send_result_activity
!! All activities complete! Sending complete_workflow_execution...
```

활동 Poller에서는 다음과 같이 출력되는데, 다른 명령줄 창에서 동시에 출력됩니다.

```
++ Activity task completed: get_contact_activity
** Starting activity task: subscribe_topic_activity
++ Activity task completed: subscribe_topic_activity
** Starting activity task: wait_for_confirmation_activity
Topic subscription still pending for (email: me@example.com)
Topic subscription confirmed for (sms: 12065550101)
++ Activity task completed: wait_for_confirmation_activity
** Starting activity task: send_result_activity
Thanks, you've successfully confirmed registration, and your workflow is complete!
++ Activity task completed: send_result_activity
All done!
```

축하합니다. 워크플로를 완료했고 이 자습서도 마치셨습니다!

제한 시간이 작동하는 방식을 보거나 다른 데이터를 입력하기 위해 이 워크플로를 다시 실행하고 싶을 수 있습니다. 토픽을 구독하면 구독을 해제할 때까지 계속 구독된 상태임을 명심하십시오. 주제 구독을 취소하기 전에 워크플로를 다시 실행하면에서 구독이 이미 확인되었음을 `wait_for_confirmation_activity` 확인하므로 자동으로 성공할 수 있습니다.

Amazon SNS 주제에서 구독을 해지하려면

- 문자 메시지에 부정적으로 응답합니다(STOP 보내기).
- 이메일에서 수신한 구독 해제 링크를 선택합니다.

이제 주제를 다시 구독할 준비가 되었습니다.

추가 정보

이 자습서에서는 많은 내용을 다루었지만 , AWS SDK for Ruby Amazon SWF 또는 Amazon SNS에 대해 더 자세히 알아볼 수 있습니다. 자세한 내용 및 더 많은 예를 보려면 각 항목에 해당하는 공식 문서를 참조하십시오.

- AWS SDK for Ruby 설명서
- Amazon Simple Notification Service 설명서
- Amazon Simple Workflow Service 설명서

Amazon SWF 콘솔에서 작업

Amazon SWF 콘솔은 워크플로 실행을 구성, 시작 및 관리하는 옵션을 제공합니다.

Amazon SWF 콘솔을 사용하여 다음을 수행할 수 있습니다.

- 워크플로 도메인을 등록합니다.
- 워크플로 유형 및 활동 유형을 등록합니다.
- 워크플로 실행을 시작, 보기, 신호, 취소, 종료 및 다시 시작합니다.

도메인 등록

워크플로는 워크플로의 범위를 제어하는 도메인이라는 AWS 리소스에서 실행됩니다. AWS 계정에는 여러 도메인이 있을 수 있고 각 도메인에는 여러 워크플로가 있을 수 있지만 다른 도메인의 워크플로는 상호 작용할 수 없습니다.

도메인 등록은 콘솔에서 처음 사용할 수 있는 유일한 기능입니다. 하나 이상의 도메인이 등록된 후 도메인에 대해 다음 작업을 수행할 수 있습니다.

- 워크플로 및 활동 유형을 등록합니다.
- 워크플로 실행을 시작합니다.
- 실행 중인 워크플로 실행을 취소 및 종료하고 이러한 워크플로 실행에 신호를 보냅니다.
- 닫힌 워크플로 실행을 다시 시작합니다.

도메인 사용 중지 및 사용 중지와 같은 도메인 관리 작업을 수행할 수도 있습니다.

도메인을 사용 중단한 이후에는 해당 도메인을 사용하여 새 워크플로 실행을 생성하거나 새 워크플로를 등록할 수 없습니다. 도메인을 더 이상 사용하지 않으면 도메인에 등록된 모든 활동 및 워크플로도 더 이상 사용되지 않습니다. 도메인 사용 중단 전에 시작된 실행은 계속 실행됩니다.

이전에 더 이상 사용되지 않는 도메인을 사용 중지한 후 도메인을 다시 사용하여 워크플로 유형을 등록하고 새 워크플로 실행을 시작할 수 있습니다.

이러한 도메인 관리 작업에 대한 자세한 내용은 DeprecateDomain 및 UndeprecateDomain을 참조하십시오.

워크플로 유형 등록

하나 이상의 도메인을 등록한 후 Amazon SWF 콘솔에서 워크플로 유형을 등록할 수 있습니다.

워크플로 유형은 목표를 수행하고 활동을 조정하는 로직을 포함하는 활동 유형 세트입니다. 워크플로 유형은 여러 컴퓨팅 디바이스에서 비동기적으로 실행할 수 있는 활동의 실행을 조정하고 관리하며 순차적 처리 방법과 병렬 처리 방법을 모두 제공합니다.

콘솔을 사용하여 Amazon SWF 워크플로 유형을 등록하려면

1. 워크플로를 등록하려는 도메인을 엽니다.
2. 등록, 워크플로 등록을 차례로 선택합니다.
3. 워크플로 등록 페이지에서 워크플로 이름 및 워크플로 버전을 입력합니다. 선택적으로 이 워크플로의 실행을 위한 결정 작업을 예약하는 데 사용할 기본 작업 목록을 지정할 수도 있습니다.
4. (선택)고급 옵션을 선택하여 워크플로에 대한 다음 세부 정보를 지정합니다.
   - 기본 작업 우선 순위 - 워크플로에 할당할 기본 작업 우선 순위입니다.
   - 기본 실행 시작-닫기 제한 시간 — 이 워크플로의 기본 최대 실행 기간입니다.
   - 기본 작업 시작-닫기 제한 시간 — 이 워크플로에 대한 결정 작업의 기본 최대 기간입니다.
   - 기본 하위 정책 - 하위 워크플로 실행에 사용할 기본 정책입니다.
   - 기본 Lambda 역할 - 이 워크플로에 연결된 기본 IAM 역할입니다.
5. 워크플로 등록을 선택합니다.

활동 유형 등록

활동은 워크플로 유형을 조정하고 실행하려는 작업입니다(예: 고객의 주문 확인, 신용 카드 청구 등). 활동이 수행되는 순서는 워크플로 유형의 조정 로직에 따라 결정됩니다.

하나 이상의 도메인이 등록된 후 활동 유형을 등록할 수 있습니다.

콘솔을 사용하여 Amazon SWF 활동 유형을 등록하려면

1. 활동을 등록하려는 도메인을 엽니다.
2. 등록, 활동 등록을 차례로 선택합니다.
3. 활동 등록 페이지에서 활동 이름 및 활동 버전을 입력합니다. 선택적으로 이 활동의 작업을 예약하는 데 사용할 기본 작업 목록을 지정할 수도 있습니다.

4. (선택)고급 옵션을 선택하여 활동에 대한 다음 세부 정보를 지정합니다.

   - 기본 작업 우선 순위 - 활동에 할당할 기본 작업 우선 순위입니다.
   - 기본 작업 예약-시작 제한 시간 — 이 활동의 작업이 작업자에게 할당되기 전에 대기할 수 있는 기본 최대 기간입니다.
   - 기본 작업 시작-닫기 제한 시간 — 작업자가 이 활동의 작업을 처리하는 데 소요될 수 있는 기본 최대 기간입니다.
   - 기본 작업 예약-닫기 제한 시간 — 이 활동의 작업에 대한 기본 최대 기간입니다.
   - 기본 작업 하트비트 제한 시간 — 이 유형의 작업을 처리하는 작업자가 RecordActivityTaskHeartbeat를 직접적으로 호출하여 진행 상황을 보고하기 전까지 걸리는 기본 최대 시간입니다.

5. 활동 등록을 선택합니다.

워크플로 시작

Amazon SWF 콘솔에서 워크플로 실행을 시작할 수 있습니다. 워크플로를 하나 이상 등록해야 워크플로 실행을 시작할 수 있습니다.

콘솔을 사용하여 워크플로 실행을 시작하려면

1. Amazon SWF 콘솔을 열고 왼쪽 탐색 창에서 도메인을 선택합니다.
2. 도메인 이름에서 워크플로를 선택합니다.
3. 워크플로 페이지에서 실행하려는 워크플로를 선택합니다.
4. 실행 시작을 선택합니다.
5. 실행 시작 페이지에서 워크플로 이름 및 실행 ID를 입력하여 이름으로 실행을 식별합니다. 선택적으로 이 워크플로 실행을 위해 생성된 결정 작업에 사용할 작업 목록을 지정할 수도 있습니다.
6. (선택)고급 옵션을 선택하여 워크플로 실행에 대한 다음 세부 정보를 지정합니다.

   - 작업 우선 순위 - 이 워크플로 실행에 사용할 작업 우선 순위입니다.
   - 실행 시작-닫기 제한 시간 - 이 워크플로 실행의 총 기간입니다.
   - 작업 시작-닫기 제한 시간 - 이 워크플로 실행에 대한 결정 작업의 최대 기간입니다.
   - 하위 정책 - TerminateWorkflowExecution 작업을 명시적으로 호출하거나 제한 시간이 만료되어 워크플로가 종료되는 경우 이 워크플로의 하위 워크플로 실행에 사용할 정책입니다.
   - Lambda 역할 -이 워크플로 실행에 연결할 IAM 역할입니다.

7. 실행 시작을 선택합니다.

워크플로 실행 관리

이름, 상태, ID 및 태그를 기준으로 워크플로 실행을 필터링할 수 있습니다. 활성 워크플로 실행에 입력과 함께 신호를 보낼 수 있습니다. 워크플로를 취소하거나 종료해야 하는 경우 Try-cancel 옵션을 사용할 수 있습니다. 취소하면 워크플로가 정리 작업을 수행한 다음 제대로 닫을 수 있기 때문에 워크플로 실행을 종료하는 것보다 취소하는 것이 좋습니다.

콘솔에서 현재 실행 중이거나 종료된 워크플로 실행을 관리할 수 있습니다.

워크플로 실행 관리 방법

1. 워크플로 실행을 관리하려면 도메인을 엽니다.
2. 실행 찾기를 선택합니다.
3. 워크플로 실행 페이지에서 속성을 기준으로 실행 필터링을 선택한 다음 속성에서 다음 필터 중 하나를 선택합니다.

선택	필터 적용 방법
워크플로	이 필터를 선택하면 특정 워크플로의 실행을 나열할 수 있습니다. 예를 들어 `fiction-books-order-workflow` 의 실행을 보려면 다음을 수행합니다. 1. 워크플로를 선택합니다. 2. 연산자에서 같음을 선택합니다. 3. 워크플로에서 fiction-books-order-workflow를 선택합니다. 4. (선택)필터 지우기를 선택하여 필터를 제거하고 새 실행 검색을 시작합니다.
상태	이 필터를 선택하면 특정 상태의 실행을 나열할 수 있습니다. 예를 들어, 상태가 실패인 실행을 보려면 다음을 수행합니다. 1. 상태를 선택합니다. 2. 연산자에서 같음을 선택합니다. 3. 상태에서 실패를 선택합니다.

선택	필터 적용 방법
	4. (선택)필터 지우기를 선택하여 필터를 제거하고 새 실행 검색을 시작합니다.
실행 ID	ID를 기반으로 워크플로 실행을 보려면 이 필터를 선택합니다. 예를 들어 ID `fiction-books-order-category1` 로 실행을 보려면 다음을 수행합니다. 1. 실행 ID를 선택합니다. 2. 연산자에서 같음을 선택합니다. 3. 실행 ID에서 소설-책-주문-카테고리1을 선택합니다. 4. (선택)필터 지우기를 선택하여 필터를 제거하고 새 실행 검색을 시작합니다.
태그	이 필터를 선택하면 특정 태그가 있는 실행을 나열할 수 있습니다. 예를 들어 `purchaseOrder` 상태의 실행을 보려면 다음을 수행합니다. 1. 태그를 선택합니다. 2. 연산자에서 같음을 선택합니다. 3. 태그에서 purchaseOrder를 선택합니다. 4. (선택)필터 지우기를 선택하여 필터를 제거하고 새 실행 검색을 시작합니다.

4. (선택)워크플로 실행을 나열하는 데 필요한 필터를 적용한 후 활성 실행에 대해 다음 작업을 수행할 수 있습니다.

   - 신호 - 이 옵션을 사용하여 실행 중인 워크플로 실행에 추가 데이터를 보낼 수 있습니다. 방법:

     1. 추가 데이터를 전송하려는 실행을 선택합니다.
     2. 신호를 선택한 다음 신호 실행 대화 상자에서 데이터를 지정합니다.
     3. 신호를 선택합니다.

   - 사용-취소 - 워크플로 실행을 취소하려면 이 옵션을 사용합니다. 워크플로 실행은 종료하기보다 취소하는 것이 좋습니다. 취소는 워크플로 실행에서 정리 작업을 수행한 다음 적당히 닫을 수 있는 기회가 됩니다.

     1. 취소할 실행을 선택합니다.
     2. 사용-취소를 선택합니다.

- 종료 - 이 옵션을 사용하면 워크플로 실행을 종료할 수 있습니다. 워크플로 실행은 종료하기보다 취소하는 것이 좋습니다.
  1. 종료할 실행을 선택합니다.
  2. 하위 정책의 경우 종료가 선택되어 있는지 확인합니다.
  3. (선택)실행 종료 이유 및 세부 정보를 지정합니다.
  4. 종료를 선택합니다.

5. (선택)다시 실행 - 종료된 워크플로 실행을 다시 실행하려면 이 옵션을 사용합니다.
   1. 워크플로 실행 목록에서 다시 실행할 닫힌 실행을 선택합니다. 닫힌 실행을 선택하면 다시 실행 버튼이 활성화됩니다. 다시 실행을 선택합니다.
   2. 다시 실행 페이지에서 워크플로 시작에 설명된 대로 워크플로 실행에 대한 세부 정보를 지정합니다.

Amazon SWF의 기본 워크플로 개념

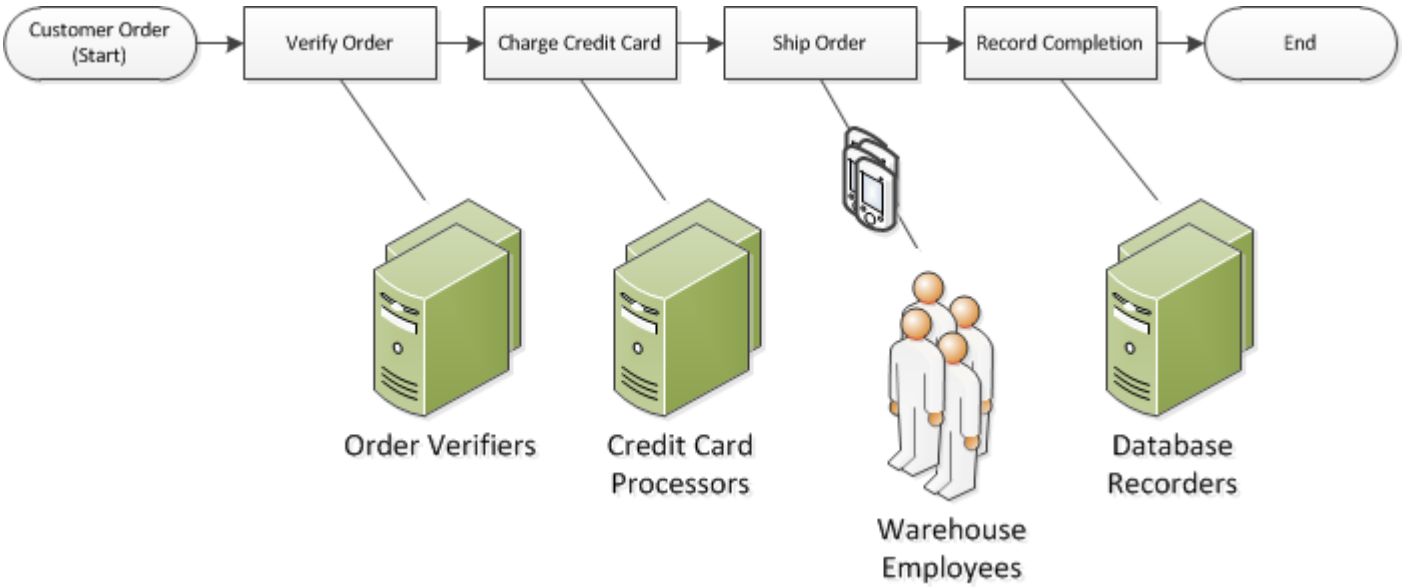
> **Note**
>
> 이 장의 개념은 Amazon Simple Workflow Service에 대해 간략하게 설명하고 주요 기능에 대해 설명합니다. 예제를 찾는 경우 섹션을 참조하세요Amazon SWF API 작업.

Amazon Simple Workflow Service(SWF)를 사용하면 분산된 비동기 애플리케이션을 워크플로로 구현할 수 있습니다. 워크플로는 여러 컴퓨팅 디바이스에서 비동기식으로 실행할 수 있고 순차 및 병렬 처리가 특징일 수 있는 활동의 실행을 조정 및 관리합니다.

워크플로를 설계하는 경우 애플리케이션을 분석해 구성 요소 작업을 식별합니다. Amazon SWF에서는 이러한 작업을 활동으로 나타냅니다. 활동이 수행되는 순서는 워크플로의 조정 로직에 따라 결정됩니다.

전자 상거래 애플리케이션의 워크플로 예제

다음 그림은 사람과 자동화된 프로세스를 모두 포함하는 전자 상거래 주문 처리 워크플로를 보여줍니다.



전자 상거래 애플리케이션 워크플로는 고객이 주문을 할 때 시작되며 다음 네 가지 작업을 포함합니다.

1. 주문을 확인합니다.
2. 주문이 유효한 경우 고객에게 비용을 청구합니다.

3. 결제되면 주문을 배송합니다.
4. 주문이 배송되면 주문 세부 정보를 저장합니다.

이 워크플로의 작업은 순차적으로 발생합니다. 즉, 먼저 주문이 확인되어야 신용카드로 비용을 청구하고, 먼저 신용카드에 비용이 청구되어야 주문을 배송하고, 먼저 주문이 배송되어야 주문을 기록합니다. 그러나 Amazon SWF는 분산된 프로세스를 지원하기 때문에 이러한 작업이 여러 위치에서 발생할 수 있습니다. 근본적으로 프로그래밍 방식의 작업인 경우에는 다른 프로그래밍 언어로 또는 다른 도구를 사용해 작업을 작성할 수도 있습니다.

Amazon SWF는 작업의 순차적 처리 외에 작업이 병렬 처리되는 워크플로도 지원합니다. 병렬 작업은 동시에 수행되며, 다른 애플리케이션 또는 인간 작업자가 독립적으로 수행할 수 있습니다. 병렬 작업 중 한 개 이상이 완료되면 워크플로에서 처리 방법을 결정합니다.

추가 개념

- Amazon SWF에서 워크플로 생성
- Amazon SWF에서 워크플로 실행
- Amazon SWF의 워크플로 기록
- Amazon SWF의 객체 식별자
- Amazon SWF의 도메인
- Amazon SWF의 액터
- Amazon SWF의 작업
- Amazon SWF의 작업 목록
- Amazon SWF의 워크플로 실행 종료
- Amazon SWF 워크플로의 수명 주기
- Amazon SWF의 작업에 대한 폴링

Amazon SWF에서 워크플로 생성

기본적인 순차적 워크플로를 생성하려면 다음 단계를 수행합니다.

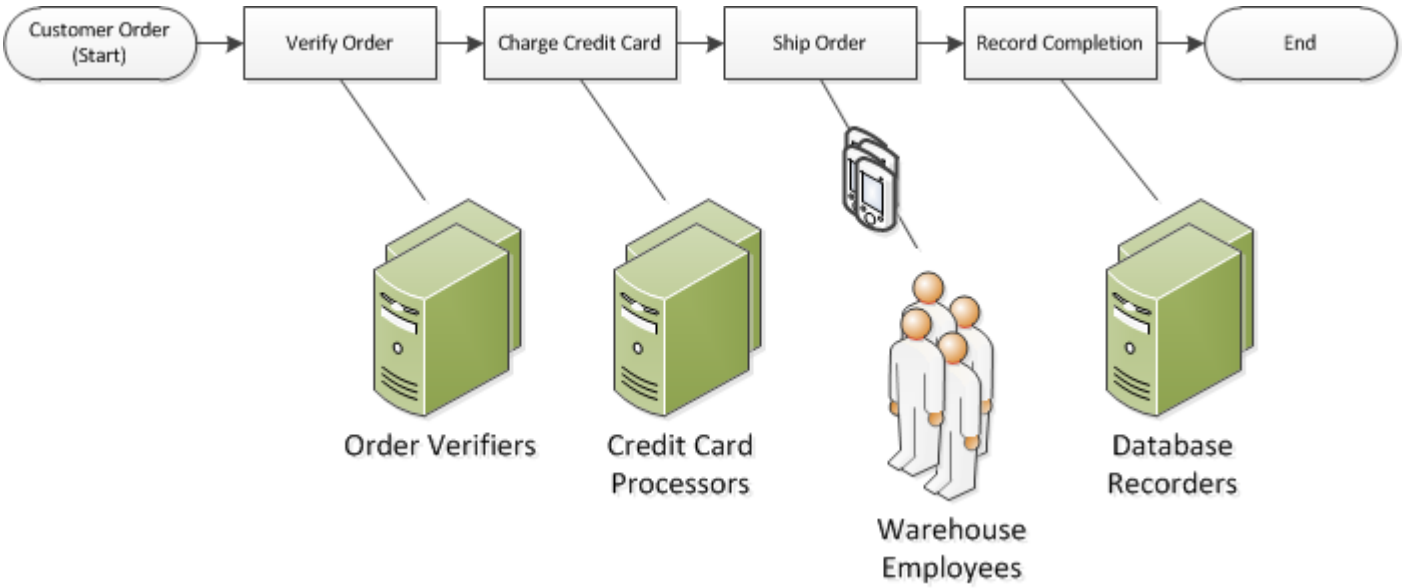
- 워크플로 모델링, 워크플로 유형 등록 및 활동 유형 등록
- 활동 작업을 수행할 활동 작업자 개발 및 시작
- 워크플로 내역을 사용해 다음에 수행할 작업을 결정하는 결정자 개발 및 시작
- 워크플로 시작자 즉, 워크플로 실행을 시작하는 애플리케이션 개발 및 시작

워크플로 및 워크플로의 활동 모델링

Amazon SWF를 사용하려면 애플리케이션의 논리적 단계를 활동으로 모델링합니다. 활동은 워크플로 내의 작업 또는 논리적 단계 하나를 나타냅니다. 예를 들어, 신용카드 인증은 신용카드 번호 및 다른 정보를 제공하고 승인 코드 또는 카드가 거부되었다는 메시지를 수신하는 작업이 관련된 활동입니다.

활동 정의 이외에 결정을 처리하는 조정 로직을 정의해야 합니다. 예를 들어 조정 로직은 신용카드가 승인 또는 거부되었는지 여부에 따라 다른 후속 조치 활동을 예약할 수 있습니다.

다음 그림은 4가지 활동(주문 확인, 신용카드 청구, 주문 발송 및 완료 기록)과 함께 순차적인 고객 주문 워크플로의 예를 보여줍니다.



Amazon SWF에서 워크플로 실행

조정 로직과 활동을 설계한 다음에는 Amazon SWF에 이러한 구성 요소를 워크플로로 등록하고 활동 유형을 등록합니다. 등록하는 동안 각 유형에 대한 이름, 버전 및 기본 구성 값을 지정합니다.

Amazon SWF는 등록된 워크플로 및 활동 유형만 사용할 수 있습니다. 이 전자 상거래 예에서는 CustomerOrder 워크플로 유형과 VerifyOrder, ChargeCreditCard, ShipOrder 및 RecordCompletion 활동 유형을 등록합니다.

워크플로 유형을 등록한 후에는 원하는 만큼 자주 실행할 수 있습니다. 워크플로 실행은 워크플로 실행 인스턴스입니다.

워크플로 실행은 임의의 프로세스 또는 애플리케이션이나 다른 워크플로 실행에 의해 시작할 수 있습니다. 이 전자 상거래 예에서 새 워크플로 실행은 각 고객 주문과 함께 시작됩니다. 워크플로를 시작하

는 애플리케이션 유형은 고객의 주문 방식에 따라 달라집니다. 워크플로는 웹 사이트 또는 모바일 애플리케이션을 통해 시작하거나 회사 내부 애플리케이션을 사용하는 고객 서비스 담당자가 시작할 수 있습니다.

Amazon SWF를 사용하면 `workflowId`라는 식별자를 워크플로 실행과 연결할 수 있으므로 기존 비즈니스 식별자를 워크플로에 통합할 수 있습니다. 이 전자 상거래 예에서 각 워크플로 실행은 고객 인보이스 번호로 식별할 수 있습니다.

Amazon SWF는 사용자가 제공하는 식별자 외에도 고유한 시스템 생성 식별자(`runId`)를 각 워크플로 실행과 연결합니다. Amazon SWF에서는 이 식별자를 사용하여 한 번에 한 번의 워크플로만 실행할 수 있습니다. 동일한 워크플로 유형으로 여러 워크플로 실행이 가능하지만 각 워크플로 실행에는 고유한 `runId`가 있습니다.

Amazon SWF의 워크플로 기록

Amazon SWF는 워크플로 기록에 모든 워크플로 실행의 진행 상황을 기록합니다. 워크플로 실행이 시작된 이후 발생한 모든 이벤트에 대한 상세하고 완전하며 일관된 레코드입니다.

이벤트는 새 활동이 예약되거나 실행 중인 활동이 완료되는 등 워크플로 실행 상태의 개별 변경을 나타냅니다. 워크플로 내역에는 워크플로 실행의 실행 상태에 변화를 일으킨 모든 이벤트가 포함됩니다(예: 활동이 예약됨 및 완료됨, 작업 시간 초과 및 신호).

워크플로 실행 상태를 변경하지 않는 작업은 일반적으로 워크플로 내역에 나타나지 않습니다. 예를 들어, 워크플로 내역에는 폴 시도 또는 시각적 작업의 사용은 표시되지 않습니다.

워크플로 내역에는 다음과 같은 몇 가지 주요 이점이 있습니다.

- 워크플로 실행에 대한 모든 정보가 워크플로 기록에 저장되므로 애플리케이션은 상태 비저장일 수 있습니다.
- 내역은 각 워크플로 실행에 대해 활동이 예약됨, 활동의 현재 상태 및 활동의 결과에 대한 기록을 제공합니다. 워크플로 실행은 이러한 정보를 사용해 다음 단계를 결정합니다.
- 내역은 실행 중인 워크플로 실행을 모니터링하고 완료된 워크플로 실행을 확인하는 데 사용할 수 있는 자세한 감사 추적을 제공합니다.

다음은 전자 상거래 워크플로 내역의 개념 보기입니다.

```
Invoice0001
```

```
Start Workflow Execution

Schedule Verify Order
Start Verify Order Activity
Complete Verify Order Activity

Schedule Charge Credit Card
Start Charge Credit Card Activity
Complete Charge Credit Card Activity

Schedule Ship Order
Start Ship Order Activity
```

이전 예에서는 주문이 배송 대기 중이었는데, 다음 예에서는 주문이 완료됩니다. 워크플로 내역은 누적되므로 다음과 같이 최신 이벤트가 추가됩니다.

```
Invoice0001

Start Workflow Execution

Schedule Verify Order
Start Verify Order Activity
Complete Verify Order Activity

Schedule Charge Credit Card
Start Charge Credit Card Activity
Complete Charge Credit Card Activity

Schedule Ship Order
Start Ship Order Activity

Complete Ship Order Activity

Schedule Record Order Completion
Start Record Order Completion Activity
Complete Record Order Completion Activity

Close Workflow
```

워크플로 실행 내역의 이벤트는 프로그래밍에서 JavaScript Object Notation(JSON) 객체로 표시됩니다. 내역 자체는 이러한 객체의 JSON 어레이입니다. 각 이벤트에는 다음 항목이 포함되어 있습니다.

- 유형(예: WorkflowExecutionStarted 또는 ActivityTaskCompleted)

- Unix 시간 형식의 타임스탬프
- 이벤트를 고유하게 식별하는 ID

또한 각 이벤트 유형에는 해당 유형에 알맞은 고유한 서술형 속성 집합이 포함되어 있습니다. 예를 들어, `ActivityTaskCompleted` 이벤트에는 활동 작업이 예약된 시간과 시작된 시점에 해당하는 이벤트의 ID가 포함된 속성과 결과 데이터를 포함하고 있는 속성이 들어 있습니다.

GetWorkflowExecutionHistory 작업을 사용하여 워크플로우 실행 내역의 현재 상태 사본을 얻을 수 있습니다. 또한 결정자는 Amazon SWF와 워크플로 결정자 간의 상호 작용 과정에서 주기적으로 내역 사본을 수신합니다.

아래는 JSON 형식으로 표시된 워크플로 실행 내역의 예 중 한 부분입니다.

```
[  {
      "eventId": 11,
      "eventTimestamp": 1326671603.102,
      "eventType": "WorkflowExecutionTimedOut",
      "workflowExecutionTimedOutEventAttributes": {
         "childPolicy": "TERMINATE",
         "timeoutType": "START_TO_CLOSE"
      }
   }, {
      "decisionTaskScheduledEventAttributes": {
         "startToCloseTimeout": "600",
         "taskList": {
            "name": "specialTaskList"
         }
      },
      "eventId": 10,
      "eventTimestamp": 1326670566.124,
      "eventType": "DecisionTaskScheduled"
   }, {
      "activityTaskTimedOutEventAttributes": {
         "details": "Waiting for confirmation",
         "scheduledEventId": 8,
         "startedEventId": 0,
         "timeoutType": "SCHEDULE_TO_START"
      },
      "eventId": 9,
      "eventTimestamp": 1326670566.124,
      "eventType": "ActivityTaskTimedOut"
   }, {
```

```
      "activityTaskScheduledEventAttributes": {
         "activityId": "verification-27",
         "activityType": {
            "name": "activityVerify",
            "version": "1.0"
         },
         "control": "digital music",
         "decisionTaskCompletedEventId": 7,
         "heartbeatTimeout": "120",
         "input": "5634-0056-4367-0923,12/12,437",
         "scheduleToCloseTimeout": "900",
         "scheduleToStartTimeout": "300",
         "startToCloseTimeout": "600",
         "taskList": {
            "name": "specialTaskList"
         }
      },
      "eventId": 8,
      "eventTimestamp": 1326670266.115,
      "eventType": "ActivityTaskScheduled"
   }, {
      "decisionTaskCompletedEventAttributes": {
         "executionContext": "Black Friday",
         "scheduledEventId": 5,
         "startedEventId": 6
      },
      "eventId": 7,
      "eventTimestamp": 1326670266.103,
      "eventType": "DecisionTaskCompleted"
   }, {
      "decisionTaskStartedEventAttributes": {
         "identity": "Decider01",
         "scheduledEventId": 5
      },
      "eventId": 6,
      "eventTimestamp": 1326670161.497,
      "eventType": "DecisionTaskStarted"
   }, {
      "decisionTaskScheduledEventAttributes": {
         "startToCloseTimeout": "600",
         "taskList": {
            "name": "specialTaskList"
         }
      },
```

```
      "eventId": 5,
      "eventTimestamp": 1326668752.66,
      "eventType": "DecisionTaskScheduled"
   }, {
      "decisionTaskTimedOutEventAttributes": {
         "scheduledEventId": 2,
         "startedEventId": 3,
         "timeoutType": "START_TO_CLOSE"
      },
      "eventId": 4,
      "eventTimestamp": 1326668752.66,
      "eventType": "DecisionTaskTimedOut"
   }, {
      "decisionTaskStartedEventAttributes": {
         "identity": "Decider01",
         "scheduledEventId": 2
      },
      "eventId": 3,
      "eventTimestamp": 1326668152.648,
      "eventType": "DecisionTaskStarted"
   }, {
      "decisionTaskScheduledEventAttributes": {
         "startToCloseTimeout": "600",
         "taskList": {
            "name": "specialTaskList"
         }
      },
      "eventId": 2,
      "eventTimestamp": 1326668003.094,
      "eventType": "DecisionTaskScheduled"
   }
]
```

워크플로 실행 기록에 표시될 수 있는 다양한 유형의 이벤트에 대한 자세한 목록은 Amazon Simple Workflow Service API 참조의 HistoryEvent 데이터 유형을 참조하십시오.

Amazon SWF는 실행이 종료된 후 구성 가능한 기간(일수) 동안 모든 워크플로 실행의 전체 내역을 저장합니다. 워크플로 내역 보존 기간이라고 하는 이 기간은 워크플로의 도메인을 등록할 때 지정합니다. 도메인에 대해서는 이 단원의 뒷부분에서 자세히 설명합니다.

Amazon SWF의 객체 식별자

다음 목록에서는 워크플로 실행과 같은 Amazon SWF 객체를 고유하게 식별하는 방법을 설명합니다.

- 워크플로 유형 – 도메인, 이름 및 버전별로 식별되는 등록된 워크플로 유형입니다. 워크플로 유형은 `RegisterWorkflowType` 호출에 지정됩니다.
- 활동 유형 – 도메인, 이름 및 버전별로 식별되는 등록된 활동 유형입니다. 활동 유형은 `RegisterActivityType` 호출에 지정됩니다.
- 결정 작업 및 활동 작업 – 각 결정 작업 및 활동 작업은 고유한 작업 토큰으로 식별됩니다. 작업 토큰은 Amazon SWF에서 생성하며 작업에 대한 다른 정보와 함께 `PollForDecisionTask` 또는 `PollForActivityTask`의 응답으로 반환됩니다. 작업 토큰은 주로 작업을 수신한 프로세스에서 사용하지만 해당 프로세스는 다른 프로세스로 토큰을 전달하고 작업의 완료 또는 실패를 보고할 수 있습니다.
- 워크플로 실행 – 도메인, 워크플로 ID 및 실행 ID로 워크플로의 실행 하나를 식별할 수 있습니다. 처음 두 개는 StartWorkflowExecution으로 전달할 파라미터입니다. 실행 ID는 `StartWorkflowExecution`에서 반환합니다.

Amazon SWF의 도메인

워크플로는 AWS 계정 내에서 Amazon SWF AWS 리소스의 범위를 지정하는 방법을 제공하는 도메인이라는 리소스에서 실행됩니다. 워크플로 유형 및 활동 유형과 같은 모든 워크플로 구성 요소를 도메인 안에 지정해야 합니다.

AWS 계정에는 여러 도메인이 있을 수 있고 각 도메인에는 여러 워크플로가 있을 수 있지만 다른 도메인의 워크플로는 상호 작용할 수 없습니다.

다른 워크플로 구성 요소를 설정하기 전에 새 워크플로를 설정하는 경우, 아직 도메인을 등록하지 않았다면 먼저 도메인부터 등록해야 합니다.

도메인을 등록하는 경우 워크플로 내역 보존 기간을 지정합니다. 보존 기간은 워크플로 실행이 완료된 후 Amazon SWF가 워크플로 실행에 대한 정보를 계속 보존하는 기간입니다.

도메인 등록은 콘솔에서 처음 사용할 수 있는 유일한 기능입니다. 하나 이상의 도메인이 등록된 후 도메인에 대해 다음 작업을 수행할 수 있습니다.

- 워크플로 및 활동 유형을 등록합니다.
- 워크플로 실행을 시작합니다.

- 실행 중인 워크플로 실행을 취소 및 종료하고 이러한 워크플로 실행에 신호를 보냅니다.
- 닫힌 워크플로 실행을 다시 시작합니다.

도메인 사용 중단 및 사용 중단과 같은 도메인 관리 작업을 수행할 수도 있습니다.

도메인을 사용 중단한 이후에는 해당 도메인을 사용하여 새 워크플로 실행을 생성하거나 새 워크플로를 등록할 수 없습니다. 도메인을 더 이상 사용하지 않으면 도메인에 등록된 모든 활동 및 워크플로도 더 이상 사용되지 않습니다. 도메인 사용 중단 전에 시작된 실행은 계속 실행됩니다.

이전에 더 이상 사용되지 않는 도메인을 사용 중지한 후 도메인을 다시 사용하여 워크플로 유형을 등록하고 새 워크플로 실행을 시작할 수 있습니다.

이러한 도메인 관리 작업에 대한 자세한 내용은 DeprecateDomain 및 UndeprecateDomain을 참조하십시오.

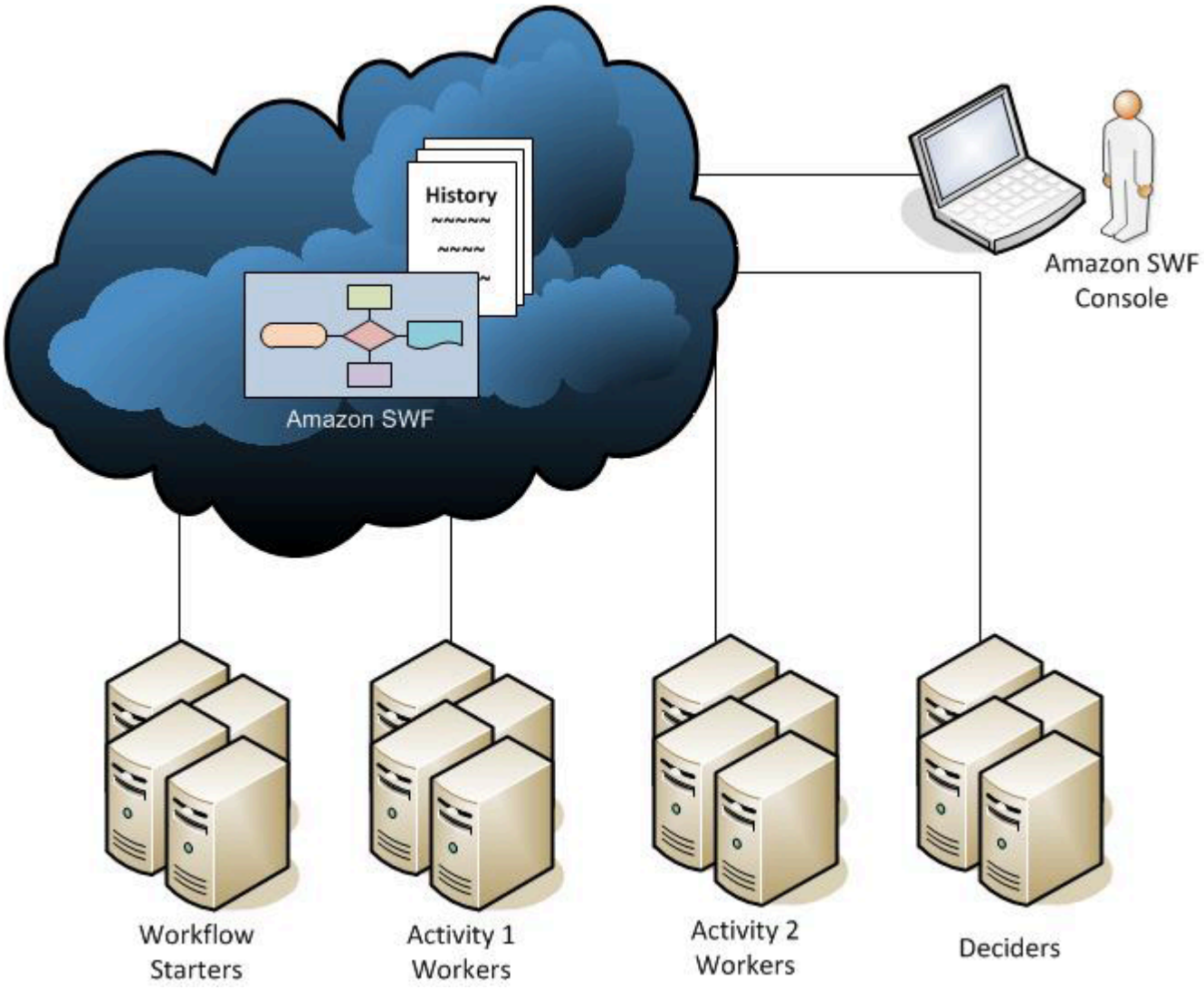
Amazon SWF의 액터

주제

Amazon SWF 액터란 무엇입니까?

작업 과정에서 Amazon SWF는 여러 가지 유형의 프로그래밍 방식 액터와 상호 작용합니다. 액터는 워크플로 시작자, 결정자 또는 활동 작업자일 수 있습니다. 이러한 액터는 API를 통해 Amazon SWF와 통신합니다. 이러한 액터는 모든 프로그래밍 언어로 개발할 수 있습니다.

다음 다이어그램은 Amazon SWF와 액터를 포함한 Amazon SWF 아키텍처를 보여 줍니다.



워크플로 시작자

워크플로 시작자는 워크플로 실행을 시작할 수 있는 애플리케이션입니다. 이 전자 상거래의 예에서는 워크플로 시작자가 고객이 주문을 하는 웹 사이트일 수 있습니다. 또 다른 워크플로 시작자로는 고객 서비스 담당자가 고객을 대신해 주문하는 모바일 애플리케이션 또는 시스템일 수 있습니다.

결정자

결정자는 워크플로의 조정 로직을 구현한 것입니다. 결정자는 워크플로 실행의 활동 작업 흐름을 제어합니다. 워크플로 실행 중 변경이 발생할 때마다(예: 작업 완료) 전체 워크플로 내역을 포함한 결정 작업이 결정자에게 전달됩니다. 결정자가 Amazon SWF에서 결정 작업을 수신하면 결정자는 워크플로 실행 내역을 분석해 워크플로 실행에서 다음으로 적절한 작업을 결정합니다. 결정자는 결정을 통해 이러한 단계를 Amazon SWF에 다시 전달합니다. 결정은 다양한 다음 작업을 나타낼 수 있는 Amazon

SWF 데이터 유형입니다. 가능한 결정의 목록은 Amazon Simple Workflow Service API 참조의 결정을 참조하십시오.

다음은 JSON 형식으로 작성된 결정의 예로, 이 형식은 Amazon SWF로 전달됩니다. 이 결정은 새 활동 작업을 예약합니다.

```
{
   "decisionType" : "ScheduleActivityTask",
   "scheduleActivityTaskDecisionAttributes" : {
      "activityType" : {
         "name" : "activityVerify",
         "version" : "1.0"
      },
      "activityId" : "verification-27",
      "control" : "digital music",
      "input" : "5634-0056-4367-0923,12/12,437",
      "scheduleToCloseTimeout" : "900",
      "taskList" : {
         "name": "specialTaskList"
      },
      "scheduleToStartTimeout" : "300",
      "startToCloseTimeout" : "600",
      "heartbeatTimeout" : "120"
   }
}
```

결정자는 워크플로 실행이 시작되는 경우 그리고 워크플로 실행에서 상태 변경이 발생할 때마다 결정 작업을 수신합니다. 결정자는 결정 작업을 수신하고 추가 결정으로 Amazon SWF에 응답해 결정자가 워크플로 실행이 완료되었다고 확인할 때까지 워크플로 실행을 계속해서 앞으로 진행합니다. 그런 다음 결정으로 응답해 워크플로 실행을 닫습니다. 워크플로 실행이 닫히면 Amazon SWF에서는 해당 실행에 대해 추가 작업을 예약하지 않습니다.

이 전자 상거래의 예에서 결정자는 각 단계가 제대로 수행되었는지 확인하고 다음 단계를 예약하거나 오류 상태를 관리합니다.

결정자는 컴퓨터 프로세스 또는 스레드 하나를 나타냅니다. 동일한 워크플로 유형의 작업을 여러 결정자가 처리할 수 있습니다.

활동 작업자

활동 작업자는 워크플로의 일부인 활동 작업을 수행하는 프로세스 또는 스레드입니다. 활동 작업은 애플리케이션에서 식별한 작업 중 하나를 나타냅니다.

워크플로우의 활동 작업을 사용하려면 Amazon SWF 콘솔 또는 RegisterActivityType 작업을 사용해 등록해야 합니다.

각 활동 작업자는 자신이 수행하기에 적절한 새 작업을 위해 Amazon SWF를 폴링합니다. 특정 작업은 특정 활동 작업자만 수행할 수 있습니다. 작업을 수신한 후 활동 작업자는 작업을 완료한 다음 해당 작업이 완료되었다고 Amazon SWF에 보고하고 결과를 제공합니다. 그런 다음 활동 작업자는 새 작업을 폴링합니다. 워크플로 실행과 연결된 활동 작업자는 이러한 방식으로 워크플로 실행 자체가 완료될 때까지 작업을 처리합니다. 이 전자 상거래 예에서 활동 작업자는 신용카드 처리 담당자 및 창고 담당 직원과 같은 사람이 사용하는 독립적인 프로세스 및 애플리케이션으로 프로세스의 개별 단계를 수행합니다.

활동 작업자는 컴퓨터 프로세스(또는 스레드) 하나를 나타냅니다. 동일한 활동 유형의 작업을 여러 활동 작업자가 처리할 수 있습니다.

액터 간에 데이터 교환

워크플로 실행이 시작되면 워크플로 실행에 입력 데이터를 제공할 수 있습니다. 마찬가지로, 입력 데이터는 활동 작업자가 활동 작업을 예약할 때 활동 작업자에게 제공할 수 있습니다. 활동 작업이 완료되면 활동 작업자는 Amazon SWF에 결과를 반환할 수 있습니다. 마찬가지로, 결정자는 실행이 완료되면 워크플로 실행의 결과를 보고할 수 있습니다. 각 액터는 사용자가 정의한 형식의 문자열을 통해 Amazon SWF에서 데이터를 전송하고 수신할 수 있습니다. 데이터의 크기 및 민감도에 따라 데이터를 직접 전달하거나 다른 시스템 또는 서비스(예: Amazon S3 또는 DynamoDB)에 저장된 데이터에 대한 포인터를 전달할 수 있습니다. 직접 전달된 데이터와 다른 데이터에 대한 포인터는 둘 다 워크플로 실행 내역에 기록되지만 Amazon SWF에서는 내역의 일부로 외부 저장소에서 데이터를 복사하거나 캐시하지 않습니다.

Amazon SWF에서는 입력 및 작업 결과를 비롯해 각 워크플로 실행의 전체 실행 상태를 유지관리하기 때문에 모든 액터는 상태 비저장일 수 있습니다. 따라서 워크플로 처리는 확장성이 매우 뛰어납니다. 시스템에 대한 로드가 증가하면 액터를 추가해 용량을 늘릴 수 있습니다.

Amazon SWF의 작업

Amazon SWF는 배정 작업(작업)을 활동 작업자 및 결정자에게 제공해 활동 작업자 및 결정자와 상호 작용합니다. Amazon SWF에는 다음 세 가지 유형의 작업이 있습니다.

- 활동 작업 – 활동 작업은 활동 작업자에게 인벤토리 확인 또는 신용카드에 청구 등과 같은 작업의 기능을 수행하도록 지시합니다. 활동 작업에는 활동 작업자가 활동 작업의 기능을 수행하는 데 필요한 모든 정보가 들어 있습니다.

- Lambda 작업 – Lambda 작업은 활동 작업과 유사하지만 일반적인 작업이 아니라 Lambda 함수를 실행합니다. Lambda 작업을 정의하는 방법에 대한 자세한 내용은 AWS Lambda Amazon SWF의 태스크 단원을 참조하십시오.
- 결정 작업 – 결정 작업은 결정자가 수행해야 할 다음 활동을 결정할 수 있도록 결정자에게 워크플로 실행의 상태가 변경되었음을 알립니다. 결정 작업에는 현재 워크플로 내역이 포함되어 있습니다.

Amazon SWF는 워크플로가 시작될 때 그리고 워크플로의 상태가 변경될 때마다(예: 활동 작업이 완료되는 경우) 결정 작업을 예약합니다. 각 결정 작업에는 전체 워크플로 실행 내역의 페이지 지정 보기가 포함되어 있습니다. 결정자는 워크플로 실행 내역을 분석해 워크플로 실행에서 다음에 발생해야 하는 작업을 지정하는 결정 세트와 함께 Amazon SWF에 다시 응답합니다. 기본적으로 모든 결정 작업은 결정자에게 워크플로를 평가해 Amazon SWF에 다시 지침을 줄 수 있는 기회를 제공합니다.

충돌하는 결정을 처리하는 일이 없도록 하기 위해 Amazon SWF는 정확하게 하나의 결정자에게 개별 결정 작업을 할당해 워크플로 실행에서 한 번에 하나의 결정 작업만 활성화되도록 합니다.

다음 표는 워크플로 및 결정자와 관련된 여러 구조 간의 관계를 보여줍니다.

논리적 설계	등록 유형	수행자	수신 및 수행	생성
워크플로	워크플로 유형	Decider	결정 작업	결정

활동 작업자가 활동 작업을 완료하면 Amazon SWF에 작업이 완료되었다고 보고하며, 여기에는 생성된 모든 관련 결과가 포함됩니다. Amazon SWF는 작업 완료를 나타내는 이벤트로 워크플로 실행 기록을 업데이트한 다음, 업데이트된 기록을 결정자에 전송하도록 결정 작업을 예약합니다.

Amazon SWF는 정확하게 하나의 활동 작업자에게 개별 활동 작업을 할당합니다. 작업이 할당되면 어떤 활동 작업자도 해당 작업을 신청하거나 수행할 수 없습니다.

다음 표는 활동과 관련된 여러 구조 간의 관계를 보여줍니다.

논리적 설계	등록 유형	수행자	수신 및 수행	생성
활동	Activity Type	Activity Worker	활동 작업	결과 데이터

Amazon SWF의 작업 목록

작업 목록을 사용하면 워크플로와 연결된 다양한 작업을 정리할 수 있습니다. 작업 목록은 동적 대기열과 유사하다고 생각하면 됩니다. Amazon SWF에서 작업을 예약할 때 대기열(작업 목록)을 지정하고 작업을 넣어 둘 수 있습니다. 마찬가지로, 작업을 위해 Amazon SWF를 폴링할 때도 작업을 가져올 대기열(작업 목록)을 지정할 수 있습니다.

작업 목록은 사용 사례의 필요에 따라 작업자에게 작업을 라우팅하는 유연한 메커니즘을 제공합니다. 작업 목록은 작업 목록을 등록하거나 작업을 통해 작업 목록을 명시적으로 생성할 필요가 없다는 점에서 동적입니다. 작업을 예약하기만 하면 지금까지 없던 작업 목록이 생깁니다.

활동 작업 및 결정 작업에 해당하는 별도의 목록이 있습니다. 작업은 항상 하나의 작업 목록에 대해서만 예약되고 목록 간에 작업은 공유되지 않습니다. 또한 활동 및 워크플로와 마찬가지로 작업 목록은 특정 AWS 리전 및 Amazon SWF 도메인으로 범위가 지정됩니다.

주제

결정 작업 목록

각 워크플로 실행은 특정한 결정 작업 목록과 연결됩니다. 워크플로우 유형이 등록되면 (RegisterWorkflowType 작업) 해당 워크플로 유형의 실행을 위한 기본 작업 목록을 지정할 수 있습니다. 워크플로 시작자가 워크플로 실행을 시작하면(`StartWorkflowExecution` 작업) 해당 워크플로 실행에 대해 다른 작업 목록을 지정할 수 있는 옵션이 있습니다.

결정자가 새 결정 작업을 폴링하면(`PollForDecisionTask` 작업) 결정자는 새 작업을 가져올 결정 작업 목록을 지정합니다. 한 결정자가 호출할 때마다 다른 작업 목록을 사용해 `PollForDecisionTask`를 여러 번 호출하면서 워크플로 실행을 여러 번 서비스할 수 있습니다. 이때 각 작업 목록은 특정 워크플로 실행에 고유합니다. 또한 결정자는 여러 가지 워크플로 실행을 위한 결정 작업을 담은 결정 작업 목록 하나를 폴링할 수 있습니다. 또한 여러 결정자가 모두 워크플로 실행을 위한 작업 목록을 폴링하도록 하여 그 워크플로 실행을 서비스하도록 할 수도 있습니다.

활동 작업 목록

활동 작업 목록 하나에 여러 활동 유형의 작업을 포함할 수 있습니다. 작업은 작업 목록 순서대로 예약됩니다. Amazon SWF는 최선을 다해 목록에서 작업을 순서대로 반환합니다. 그러나 목록의 작업을 순서대로 가져올 수 없는 상황도 있습니다.

활동 유형이 등록되면(RegisterActivityType 작업) 해당 활동 유형에 대한 기본 작업 목록을 지정할 수 있습니다. 기본적으로 이 유형의 활동 작업은 지정된 작업 목록에 대해 예약되지만 결정자가 활동 작업을 예약하는 경우(ScheduleActivityTask 결정) 결정자는 선택적으로 작업을 예약할 다른 작업 목록을 지정할 수 있습니다. 결정자가 작업 목록을 지정하지 않으면 기본 작업 목록이 사용됩니다. 따라서 작업의 속성에 따라 특정 작업 목록에 활동 작업을 배치할 수 있습니다. 예를 들어, 주어진 신용카드 유형에 대한 활동 작업의 모든 인스턴스를 특정 작업 목록에 배치할 수 있습니다.

작업 라우팅

활동 작업자가 새 작업을 폴링할 때(PollForActivityTask 작업) 작업을 가져올 활동 작업 목록을 지정할 수 있습니다. 그러면 활동 작업자는 해당 목록의 작업만 수락합니다. 이러한 방식으로 특정 작업을 특정 활동 작업자에게만 할당할 수 있습니다. 예를 들어, 고성능 컴퓨터 사용이 필요한 작업을 보관하는 작업 목록을 만들 수 있습니다. 적절한 하드웨어에서 실행 중인 활동 작업자만 해당 작업 목록을 폴링할 수 있습니다. 또 다른 예로, 특정 지리적 리전에 대한 작업 목록을 생성할 수 있습니다. 그런 다음 해당 리전에 배포된 작업자만 작업을 선택하도록 할 수 있습니다. 또는 우선순위가 높은 주문에 대한 작업 목록을 생성해 항상 해당 목록을 먼저 확인하도록 만들 수 있습니다.

이러한 방식으로 특정 작업을 특정 활동 작업자에게 할당하는 것을 작업 라우팅이라고 합니다. 작업 라우팅은 선택적입니다. 활동 작업 예약 시 작업 목록을 지정하지 않으면 작업은 기본 작업 목록에 자동으로 배치됩니다.

Amazon SWF의 워크플로 실행 종료

워크플로 실행을 시작하면 워크플로 실행이 열립니다. 열린 워크플로 실행은 완료됨, 취소됨, 실패 또는 시간 초과로 닫힙니다. 또한 새로운 실행으로 계속하거나 종료될 수 있습니다. 워크플로 실행은 결정자, 워크플로 관리자 또는 Amazon SWF가 닫을 수 있습니다.

결정자가 워크플로 활동이 완료되었는지 확인하면 `RespondDecisionTaskCompleted` 작업을 사용해 워크플로 실행을 완료됨으로 닫고 `CompleteWorkflowExecution` 결정을 전달해야 합니다.

또는 결정자가 취소됨 또는 실패로 워크플로 실행을 닫을 수 있습니다. 실행을 취소하려면 결정자는 `RespondDecisionTaskCompleted` 작업을 사용해 `CancelWorkflowExecution` 결정을 전달해야 합니다.

워크플로 실행이 일반적인 완료 영역을 벗어난 상태가 되면 결정자는 워크플로 실행을 실패시켜야 합니다. 실행을 실패시키려면 결정자는 `RespondDecisionTaskCompleted` 작업을 사용해 `FailWorkflowExecution` 결정을 전달해야 합니다.

Amazon SWF는 워크플로 실행을 모니터링해 사용자 지정 제한 시간 설정을 초과하지 않도록 합니다. 워크플로 실행 시간이 초과되면 Amazon SWF는 해당 워크플로 실행을 자동으로 닫습니다. 제한 시간 값에 대한 자세한 내용은 Amazon SWF 제한 시간 유형 단원을 참조하십시오.

결정자는 `RespondDecisionTaskCompleted` 작업을 사용하고 `ContinueAsNewWorkflowExecution` 결정을 전달해 실행을 닫은 다음 논리적으로 새 실행으로 계속 진행할 수 있습니다. 이것은 시간이 지남에 따라 내역이 매우 커질 수 있는 장기 워크플로 실행에 유용한 전략입니다.

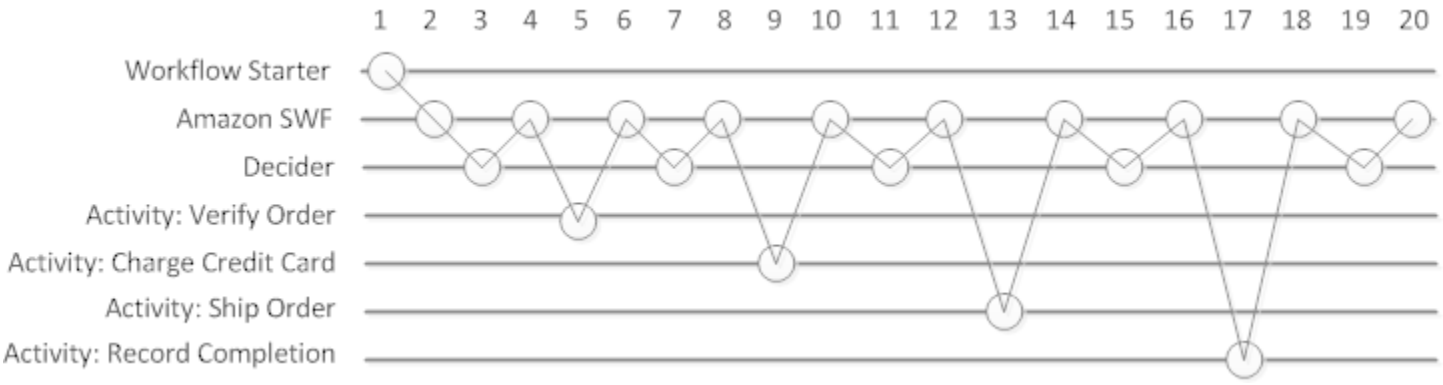
마지막으로, `TerminateWorkflowExecution` API를 사용하여 Amazon SWF 콘솔에서 직접 또는 프로그래밍 방식으로 워크플로우 실행을 종료할 수 있습니다. 종료는 워크플로 실행을 강제로 닫습니다. 취소하면 결정자가 워크플로 실행 닫기를 관리할 수 있기 때문에 종료보다는 취소가 좋습니다.

워크플로 실행이 서비스에 정의된 일정 한도를 초과하면 Amazon SWF는 실행을 종료합니다. 상위 워크플로가 종료되었고 해당하는 하위 정책에서 하위 워크플로 역시 종료하도록 지시하는 경우, Amazon SWF는 하위 워크플로를 종료합니다.

Amazon SWF 워크플로의 수명 주기

워크플로 실행 시작에서 완료에 이르기까지 Amazon SWF는 액터에게 적절한 작업(활동 작업 또는 결정 작업)을 할당해 액터와 상호 작용합니다.

다음 다이어그램은 구성 요소의 관점에서 주문 처리 워크플로 실행의 수명 주기를 보여줍니다.



워크플로 실행 수명 주기

다음 표에서는 앞선 이미지에 나온 각 작업에 대해 설명합니다.

설명	작업, 결정 또는 이벤트
1. 워크플로 시작자가 적절한 Amazon SWF 작업을 호출하여 주문에 대한 워크플로 실행을 시작하여 주문 정보를 제공합니다.	StartWorkflowExecution action.
2. Amazon SWF는 시작 워크플로 실행 요청을 수신한 다음 첫 번째 결정 작업을 예약합니다.	WorkflowExecutionStarted 이벤트 및 DecisionTaskScheduled 이벤트
3. 결정자가 Amazon SWF에서 작업을 수신하고, 내역을 검토한 다음 조정 로직을 적용해 이전 활동이 발생하지 않았는지 확인하고, 활동 작업자가 작업을 처리하는 데 필요한 정보를 사용해 주문 확인 활동을 예약하도록 결정하고 Amazon SWF에 결정을 반환합니다.	PollForDecisionTask 작업. RespondDecisionTaskCompleted 작업 및 ScheduleActivityTask 결정.
4. Amazon SWF에서 결정을 수신하고, 주문 확인 활동 작업을 예약하고, 작업이 완료 또는 시간 초과될 때까지 대기합니다.	ActivityTaskScheduled 이벤트를 트리거합니다
5. 주문 확인 활동을 수행할 수 있는 활동	PollForActivityTask 작업 및 RespondActivityTaskCompleted 작업

설명	작업, 결정 또는 이벤트
작업자가 작업을 수신하고, 수행한 다음 Amazon SWF에 결과를 반환합니다.	
6. Amazon SWF는 주문 확인 활동의 결과를 수신해 워크플로 내역에 추가하고, 결정 작업을 예약합니다.	ActivityTaskCompleted 이벤트 및 DecisionTaskScheduled 이벤트
7. 결정자가 Amazon SWF에서 작업을 수신하고, 내역을 검토한 다음 조정 로직을 적용해 활동 작업자가 작업을 처리하는 데 필요한 정보를 사용해 ChargeCreditCard 활동 작업을 예약하도록 결정하고 해당 결정을 Amazon SWF로 반환합니다.	PollForDecisionTask 작업. ScheduleActivityTask 결정을 통한 RespondDecisionTaskCompleted 작업.
8. Amazon SWF에서 결정을 수신하고, ChargeCreditCard 활동 작업을 예약하고, 해당 작업이 완료 또는 시간 초과될 때까지 대기합니다.	DecisionTaskCompleted 이벤트 및 ActivityTaskScheduled 이벤트

설명	작업, 결정 또는 이벤트
9. ChargeCreditCard 활동을 수행할 수 있는 활동 작업자가 작업을 수신하고, 수행한 다음 Amazon SWF에 결과를 반환합니다.	PollForActivityTask 및 RespondActivityTaskCompleted 작업
10. Amazon SWF에서는 ChargeCreditCard 활동 작업의 결과를 수신해 워크플로 내역에 추가하고, 결정 작업을 예약합니다.	ActivityTaskCompleted 이벤트 및 DecisionTaskScheduled 이벤트
11. 결정자가 Amazon SWF에서 작업을 수신하고, 내역을 검토한 다음 조정 로직을 적용해 활동 작업자가 작업을 수행하는 데 필요한 정보를 사용해 ShipOrder 활동 작업을 예약하도록 결정하고 해당 결정을 Amazon SWF로 반환합니다.	PollForDecisionTask 작업. ScheduleActivityTask 결정을 통한 RespondDecisionTaskCompleted.
12. Amazon SWF에서 결정을 수신하고, ShipOrder 활동 작업을 예약하고, 해당 작업이 완료 또는 시간 초과될 때까지 대기합니다.	DecisionTaskCompleted 이벤트 및 ActivityTaskScheduled 이벤트

설명	작업, 결정 또는 이벤트
13. ShipOrder 활동을 수행할 수 있는 활동 작업자가 작업을 수신하고, 수행한 다음 Amazon SWF에 결과를 반환합니다.	PollForActivityTask 작업 및 RespondActivityTaskCompleted 작업
14. Amazon SWF는 ShipOrder 활동 작업의 결과를 수신하여 워크플로 내역에 추가하고, 결정 작업을 예약합니다.	ActivityTaskCompleted 이벤트 및 DecisionTaskScheduled 이벤트
15. 결정자가 Amazon SWF에서 작업을 수신하고, 내역을 검토한 다음 조정 로직을 적용해 활동 작업자가 작업을 수행하는 데 필요한 정보를 사용해 RecordCompletion 활동 작업을 예약하도록 결정하고 해당 결정을 Amazon SWF로 반환합니다.	PollForDecisionTask 작업. ScheduleActivityTask 결정을 통한 RespondDecisionTaskCompleted 작업.
16. Amazon SWF에서 결정을 수신하고, RecordCompletion 활동 작업을 예약하고, 해당 작업이 완료 또는 시간 초과될 때까지 대기합니다.	DecisionTaskCompleted 이벤트 및 ActivityTaskScheduled 이벤트

설명	작업, 결정 또는 이벤트
17. RecordCompletion 활동을 수행할 수 있는 활동 작업자가 작업을 수신하고, 수행한 다음 Amazon SWF에 결과를 반환합니다.	PollForActivityTask 작업 및 RespondActivityTaskCompleted 작업
18. Amazon SWF는 RecordCompletion 활동 작업의 결과를 수신해 워크플로 내역에 추가하고, 결정 작업을 예약합니다.	ActivityTaskCompleted 이벤트 및 DecisionTaskScheduled 이벤트
19. 결정자가 Amazon SWF에서 작업을 수신하고, 내역을 검토한 다음 조정 로직을 적용해 워크플로 실행을 닫도록 결정하고 해당 결정을 결과와 함께 Amazon SWF로 반환합니다.	PollForDecisionTask 작업. CompleteWorkflowExecution 결정을 통한 RespondDecisionTaskCompleted 작업.
20. Amazon SWF는 워크플로 실행을 닫고 이후에 참조할 수 있도록 내역을 보관합니다.	WorkflowExecutionCompleted 이벤트를 트리거합니다.

Amazon SWF의 작업에 대한 폴링

결정자 및 활동 작업자는 긴 폴링을 사용하여 Amazon SWF와 통신합니다. 결정자 또는 활동 작업자가 주기적으로 Amazon SWF와 통신을 시작해 Amazon SWF에 작업을 수락할 수 있음을 알린 후 작업을 가져올 작업 목록을 지정합니다.

지정된 작업 목록에서 작업을 사용할 수 있으면 Amazon SWF가 그에 대한 응답으로 즉시 작업을 반환합니다. 사용할 수 있는 작업이 없을 경우에는 Amazon SWF가 TCP 연결을 60초 동안 개방된 상태로 유지합니다. 따라서 이 시간 동안 작업 사용이 가능하다면, 동일한 연결에서 작업을 반환할 수 있습니다. 60초 안에 사용 가능한 작업이 없으면 빈 응답을 반환하고 연결을 끊습니다. 빈 응답은 Task 구조로, 여기서 taskToken의 값이 빈 문자열입니다. 이 경우 결정자나 활동 작업자가 다시 폴링해야 합니다.

긴 폴링은 용량이 많은 작업 처리에 유용합니다. 결정자와 활동 작업자는 자체 용량을 관리할 수 있으며, 결정자와 활동 작업자가 방화벽 뒤에 있는 경우에 사용하기 편리합니다.

자세한 내용은 결정 작업 폴링 및 활동 작업 폴링 섹션을 참조하세요.

Amazon SWF의 고급 워크플로 개념

??? 단원에 나오는 전자 상거래의 예는 간소화된 워크플로 시나리오를 보여줍니다. 실제로, 사용자는 워크플로가 동시 작업(신용카드를 인증하는 동시에 주문 확인 이메일 보내기)을 수행하고, 주요 이벤트(모든 품목이 포장됨)를 기록하고, 변경 사항(품목 추가 또는 제거)으로 주문을 업데이트하고, 워크플로 실행의 일부로 더욱 높은 수준의 결정을 내리길 원합니다. 이 섹션에서는 워크플로를 구성하는 데 사용할 수 있는 고급 워크플로 개념을 설명합니다.

고급 개념

- 버전 관리
- 신호
- Amazon SWF의 하위 워크플로
- Amazon SWF의 마커
- Amazon SWF의 태그
- Amazon SWF를 사용하여 독점적 선택 구현
- Amazon SWF의 타이머
- Amazon SWF에서 활동 작업 취소

버전 관리

업무를 수행하다 보면 종종 동시에 실행 중인 동일한 워크플로 또는 활동을 다양한 구현 또는 변형해야 하는 경우가 있습니다. 예를 들어, 다른 워크플로가 프로덕션에 사용 중인 경우 워크플로의 새로운 구현을 테스트하려고 할 수 있습니다. 두 가지 다른 기능 세트를 사용해 기본 구현 및 프리미엄 구현처럼 두 가지 다른 구현을 실행하려고 할 수도 있습니다. 버전 관리를 통해서는 요구 사항에 부합하는 용도에 맞춰 워크플로 및 활동의 여러 구현을 동시에 실행할 수 있습니다.

워크플로와 활동 유형에는 등록 시 지정된 버전이 연결되어 있습니다. 버전은 자유로운 형식의 문자열로, 고유한 버전 관리 체계를 선택할 수 있습니다. 등록된 유형의 새 버전을 생성하려면 해당 유형을 같은 이름을 사용하되 다른 버전을 지정해 등록해야 합니다. 앞서 설명한 Amazon SWF의 작업 목록에서 버전 관리 구현에 대한 자세한 내용을 찾아볼 수 있습니다. 이미 진행 중인 지정된 유형의 워크플로 실행이 오래 실행 중인 상황과 새 기능 추가와 같이 워크플로를 수정해야 하는 상황을 고려합니다. 새 기능은 활동 유형 및 작업자의 새 버전과 새 결정자를 생성해 구현할 수 있습니다. 그런 다음 다른 작업 목록 세트를 사용해 새 워크플로 버전의 실행을 시작할 수 있습니다. 이러한 방식으로 동시에 실행 중인 여러 버전의 워크플로가 서로 영향을 미치지 않고 실행할 수 있습니다.

신호

신호를 통해서는 실행 중인 워크플로 실행에 정보를 삽입할 수 있습니다. 일부 시나리오에서는 실행 중인 워크플로 실행에 정보를 추가해 워크플로에 어떤 부분이 변경되었음을 알리거나 외부 이벤트에 대해 알릴 수 있습니다. 모든 프로세스에서 열린 워크플로 실행에 신호를 보낼 수 있습니다. 예를 들어 워크플로 실행 하나가 다른 워크플로에 신호를 보낼 수 있습니다.

> **Note**
>
> 열리지 않은 워크플로 실행에 신호를 보내려고 하면 `SignalWorkflowExecution` 실패로 이어지고 `UnknownResourceFault`가 발생합니다.

신호를 사용하려면 신호에 전달할 신호 이름과 데이터(있는 경우)를 정의하십시오. 그런 다음 내역에 있는 신호 이벤트(WorkflowExecutionSignaled)를 인식해 적절하게 처리하도록 결정자를 프로그래밍합니다. 프로세스가 워크플로 실행에 신호를 보내려는 경우, 대상 워크플로 실행에 대한 식별자, 신호 이름 및 신호 데이터를 지정하는 Amazon SWF(SignalWorkflowExecution 작업을 사용하거나 결정자의 경우 SignalExternalWorkflowExecution 결정을 사용)를 직접적으로 호출합니다. 그런 다음 Amazon SWF는 신호를 수신하여 대상 워크플로 실행 기록에 기록하고 이에 대한 결정 작업을 예약합니다. 결정자가 결정 작업을 수신하면 워크플로 실행 내역 내에서도 해당 신호를 수신합니다. 그런 다음 결정자는 신호 및 신호의 데이터를 기반으로 적절한 작업을 수행할 수 있습니다.

신호를 기다려야 하는 경우가 있습니다. 예를 들어, 사용자는 신호를 보내 주문을 취소할 수 있지만 주문 후 1시간 이내에만 주문을 취소할 수 있습니다. Amazon SWF에는 결정자가 서비스의 신호를 기다릴 수 있는 기본 요소가 없습니다. 결정자 자체에서 일시 중지 기능을 구현해야 합니다. 일시 중지하려면 결정자가 `StartTimer` 결정을 사용해 타이머를 시작해야 합니다. 결정자는 결정 작업을 계속 폴링하면서 이 타이머에 지정된 기간 동안 신호를 기다립니다. 결정 작업을 받은 결정자는 내역을 확인해 신호가 수신되었는지 아니면 타이머가 만료되었는지 확인해야 합니다. 신호가 수신되었다면 결정자는 타이머를 취소해야 합니다. 그렇지 않고 타이머가 만료되었다면 지정된 시간 안에 신호가 도착하지 않았다는 뜻입니다. 요약하자면, 특정 신호를 기다리려면 다음과 같이 합니다.

1. 결정자가 기다려야 하는 시간 길이에 해당하는 타이머를 생성합니다.
2. 결정 작업을 받으면 내역을 확인해 신호가 수신되었는지 아니면 타이머가 만료되었는지 확인합니다.
3. 신호가 수신되면 `CancelTimer` 결정을 사용해 타이머를 취소하고 신호를 처리합니다. 타이밍에 따라 내역에 `TimerFired` 및 `WorkflowExecutionSignaled` 이벤트가 둘 다 포함될 수 있습니다.

이러한 경우 내역에 표시된 이벤트의 상대적 순서에 따라 무엇이 먼저 발생했는지 확인할 수 있습니다.

4. 신호가 수신되기 전에 타이머가 만료되었다면 신호를 기다리던 결정자는 시간 초과됩니다. 실행에 실패하거나 사용 사례에 적합한 다른 로직은 무엇이든 수행할 수 있습니다.

워크플로를 취소해야 하는 경우(예: 고객이 주문 자체를 취소한 경우) 워크플로에 신호를 보내는 대신 `RequestCancelWorkflowExecution` 작업을 사용해야 합니다.

몇 가지 신호 적용 사례는 다음과 같습니다.

- 신호가 수신될 때까지 워크플로 실행의 진행을 일시 중지합니다(예: 재고 선적 대기).
- 결정자의 의사 결정 방식에 영향을 줄 수 있는 정보를 워크플로 실행에 제공합니다. 이것은 외부 이벤트의 영향을 받는 워크플로일 때 유용합니다(예: 폐점 후 재고 판매 종료).
- 변경이 예측되는 경우 워크플로 실행을 업데이트합니다(예: 주문 후 선적 이전에 주문 수량 변경).

다음 예에서는 워크플로 실행으로 주문 취소 신호가 전송됩니다.

```
https://swf.us-east-1.amazonaws.com
SignalWorkflowExecution
{"domain": "867530901",
 "workflowId": "20110927-T-1",
 "runId": "f5ebbac6-941c-4342-ad69-dfd2f8be6689",
 "signalName": "CancelOrder",
 "input": "order 3553"}
```

워크플로 실행이 신호를 수신하면 Amazon SWF는 다음과 유사한 성공적인 HTTP 응답을 반환합니다. Amazon SWF는 결정 작업을 생성하여 결정자에게 신호를 처리하도록 알립니다.

```
HTTP/1.1 200 OK
Content-Length: 0
Content-Type: application/json
x-amzn-RequestId: bf78ae15-3f0c-11e1-9914-a356b6ea8bdf
```

Amazon SWF의 하위 워크플로

복잡한 워크플로는 하위 워크플로를 사용해 더 작고 관리하기 쉽고 잠재적으로 재사용 가능한 구성 요소로 나눌 수 있습니다. 하위 워크플로는 다른 (상위) 워크플로 실행으로 시작되는 워크플로 실행입니

다. 하위 워크플로를 시작하기 위해 상위 워크플로 결정자는 `StartChildWorkflowExecution` 결정을 사용합니다. 이 결정으로 지정된 입력 데이터는 내역을 통해 하위 워크플로에 사용할 수 있습니다.

`StartChildWorkflowExecution` 결정의 속성 역시 하위 정책 즉, Amazon SWF가 하위 워크플로 실행 이전에 상위 워크플로 실행을 종료하는 상황을 처리해야 하는 방법을 지정합니다. 다음 세 가지 값이 가능합니다.

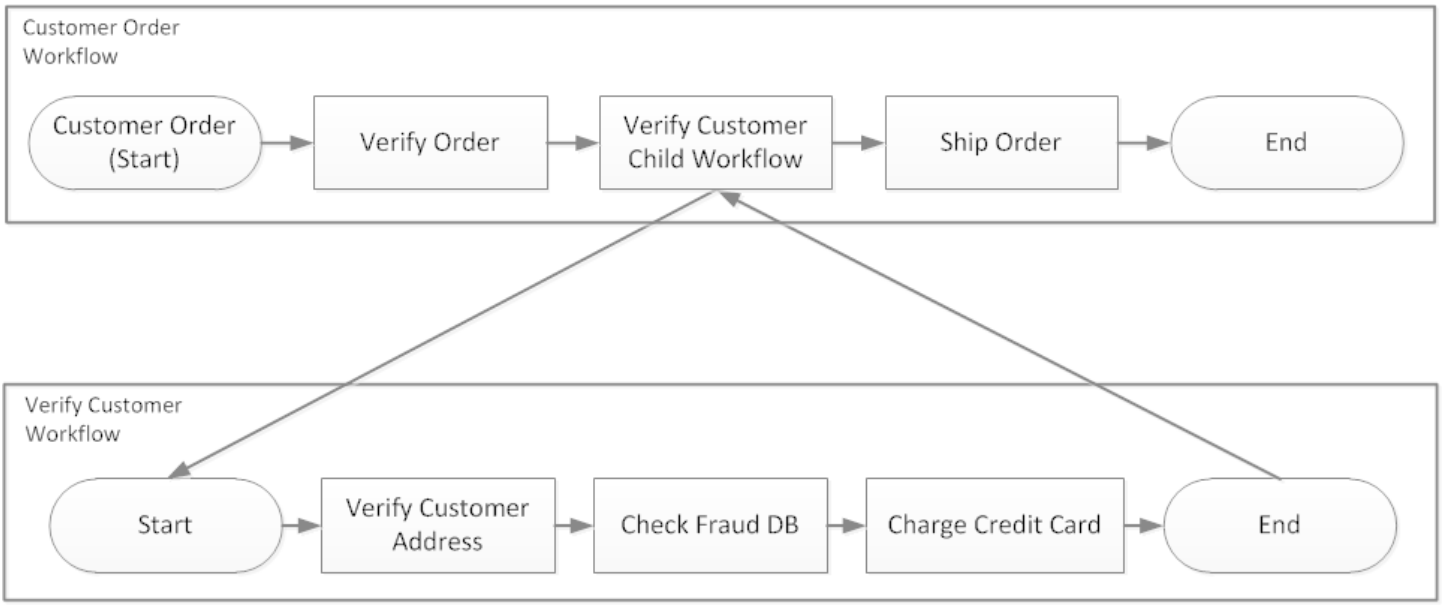
- TERMINATE: Amazon SWF가 하위 실행을 종료합니다.
- REQUEST_CANCEL: Amazon SWF가 하위 워크플로 실행 내역에 `WorkflowExecutionCancelRequested` 이벤트를 배치하여 하위 실행을 취소하려고 합니다.
- ABANDON: Amazon SWF에서 아무런 조치를 취하지 않고, 하위 실행이 계속 실행됩니다.

하위 워크플로 실행이 시작되면 마치 정규 실행처럼 실행됩니다. 완료되면 Amazon SWF가 상위 워크플로 실행의 워크플로 내역에 결과와 함께 완료를 기록합니다. 하위 워크플로의 예는 다음을 포함합니다.

- 여러 웹 사이트에서 워크플로가 사용하는 신용카드 처리 하위 워크플로
- 고객 이메일 주소를 확인하고, 옵트 아웃 목록을 확인하고, 이메일을 보내고, 이메일이 반송되거나 전송에 실패하지 않았음을 확인하는 이메일 하위 워크플로
- 연결, 설정, 트랜잭션 및 확인을 결합하는 데이터베이스 저장 및 검색 하위 워크플로
- 빌드, 패키징 및 확인을 결합하는 소스 코드 컴파일 하위 워크플로

전자 상거래의 예에서 신용카드에 비용 청구 활동을 하위 워크플로로 만들려고 할 수 있습니다. 이렇게 하려면 새 고객 확인 워크플로를 등록하고, 고객 주소 확인 및 사기 DB 확인 활동을 등록하고, 해당 작업에 대한 조정 로직을 정의할 수 있습니다. 그런 다음 고객 주문 워크플로의 결정자는 워크플로 유형을 지정하는 `StartChildWorkflowExecution` 결정을 예약해 고객 확인 하위 워크플로를 시작할 수 있습니다.

다음 그림은 고객 주소를 확인하고, 사기 데이터베이스를 확인하고, 신용카드에 비용을 청구하는 새로운 고객 확인 하위 워크플로를 포함한 고객 주문 워크플로를 보여줍니다.



여러 워크플로가 동일한 워크플로 유형을 사용해 하위 워크플로 실행을 생성할 수 있습니다. 예를 들어, 고객 확인 하위 워크플로는 조직의 다른 부분에서도 사용할 수 있습니다. 하위 워크플로에 대한 이벤트는 자체 워크플로 내역에 포함되며 상위 워크플로 내역에 포함되지 않습니다.

하위 워크플로는 결정자가 시작하는 단순한 워크플로 실행이므로 일반적인 독립형 워크플로 실행으로 시작할 수도 있습니다.

Amazon SWF의 마커

때로는 사용 사례에 고유한 워크플로 실행의 워크플로 내역에 정보를 기록하려고 할 수 있습니다. 마커를 통해 사용자 지정 용도 또는 시나리오별 용도에 사용할 수 있는 정보를 워크플로 실행 내역에 기록할 수 있습니다.

마커를 사용하기 위해 결정자는 RecordMarker 결정을 사용하고, 마커 이름을 지정하고, 결정에 원하는 데이터를 첨부하고, `RespondDecisionTaskCompleted` 작업을 사용하여 Amazon SWF에 알립니다. Amazon SWF는 요청을 수신하고, 워크플로 기록에 마커를 기록하고, 요청에서 다른 결정을 내립니다. 여기서 결정자는 워크플로 내역의 마커를 보고 사용자가 프로그래밍한 방식으로 사용할 수 있습니다.

마커를 기록하는 행위 자체로는 결정 작업이 시작되지 않습니다. 워크플로 실행이 멈추지 않도록 하려면 워크플로 실행을 계속하게 만드는 어떤 이벤트가 발생해야 합니다. 예를 들어 여기에는 결정자가 다른 활동 작업 예약, 워크플로 실행에서 신호 수신 또는 이전에 예약된 활동 작업 완료 등이 포함될 수 있습니다.

마커의 예는 다음을 포함합니다.

- recursive 워크플로의 루프 개수를 세는 카운터입니다.
- 활동의 결과를 기반으로 하는 워크플로 실행 진행 상황
- 이전의 워크플로 내역 이벤트에서 요약된 정보

전자 상거래의 예에서 매일 재고를 확인하고, 매번 마커의 개수를 증가시키는 활동을 추가할 수 있습니다. 그런 다음 고객에게 이메일을 보내거나 개수가 5개를 초과하는 경우 전체 내역을 검토할 필요 없이 관리자에게 알리는 결정 로직을 추가할 수 있습니다.

다음 예에서 결정자는 결정 작업을 완료하고 `RecordMarker` 결정이 포함된 `RespondDecisionTaskCompleted` 작업으로 응답합니다.

```
https://swf.us-east-1.amazonaws.com
RespondDecisionTaskCompleted
{
  "taskToken":"12342e17-80f6-FAKE-TASK-TOKEN32f0223",
  "decisions":[{
          "decisionType":"RecordMarker",
          "recordMarkerDecisionAttributes":{
              "markerName":"customer elected special shipping offer"
          }
      },
  ]
}
```

Amazon SWF가 마커를 성공적으로 기록하면 다음과 유사한 성공적인 HTTP 응답을 반환합니다.

```
HTTP/1.1 200 OK
Content-Length: 0
Content-Type: application/json
x-amzn-RequestId: 6c0373ce-074c-11e1-9083-8318c48dee96
```

Amazon SWF의 태그

Amazon SWF는 워크플로 실행 태깅 작업을 지원합니다. 이 기능은 리소스가 많을 때 특히 유용합니다.

Amazon SWF는 최대 5개 태그를 사용해 워크플로 실행에 태그를 지정할 수 있습니다. 각 태그는 자유 형식의 문자열로 길이는 최대 256자입니다. 태그를 사용하려는 경우 워크플로 실행 시작 시 태그를 할

당해야 합니다. 워크플로 실행을 시작한 후에는 워크플로 실행에 태그를 추가할 수 없고 워크플로 실행에 할당된 태그는 편집하거나 제거할 수 없습니다.

IAM은 태그를 기반으로 Amazon SWF 도메인에 대한 액세스를 제어하는 작업을 지원합니다. 태그를 기반으로 액세스를 제어하려면 IAM 정책의 조건 요소에 태그 정보를 제공하십시오.

태그 관리

AWS SDKs를 사용하거나 Amazon SWF API와 직접 상호 작용하여 Amazon Simple Workflow Service 태그를 관리합니다. API를 사용하면 도메인을 등록할 때 태그를 추가하고, 기존 도메인에 대한 태그를 나열하며, 기존 도메인에 대한 태그를 추가하거나 삭제할 수 있습니다.

> **Note**
>
> 리소스당 태그는 50개로 제한됩니다. Amazon SWF의 일반 계정 할당량 섹션을 참조하세요

- `RegisterDomain`
- `ListTagsForResource`
- `TagResource`
- `UntagResource`

자세한 내용은 Amazon SWF API 작업 및 Amazon Simple Workflow Service API 참조를 참조하십시오.

워크플로 실행에 태그 지정

Amazon SWF를 사용하면 태그를 워크플로 실행과 연결한 다음 이러한 태그를 기반으로 워크플로 실행을 쿼리할 수 있습니다. 가시성 작업을 사용할 때 listi를 필터링할 수 있습니다. 실행에 할당하는 태그를 신중하게 선택하면 태그를 사용하여 의미 있는 목록을 제공할 수 있습니다.

예를 들어, 주문 처리 센터를 여러 개 운영 중이라고 가정해 보겠습니다. 태그를 사용하면 특정 이행 센터에서 발생하는 프로세스를 나열할 수 있습니다. 또는 고객이 다른 유형의 미디어 파일을 변환하는 경우 태그는 비디오, 오디오 및 이미지 파일을 변환할 때 다른 프로세스를 나타낼 수 있습니다.

`StartWorkflowExecution` 작업, `StartChildWorkflowExecution` 결정 또는 `ContinueAsNewWorkflowExecution` 결정을 사용하여 실행을 시작할 때 워크플로 실행에 태그를 최대 5개까지 연결할 수 있습니다. 가시성 작업을 사용하여 워크플로 실행을 나열하거나 계산하는 경우 태그를 기반으로 결과를 필터링할 수 있습니다.

태그 지정을 사용하려면

1. 태그 지정 전략을 세웁니다. 비즈니스 요구 사항에 대해 생각해 보고 의미 있는 태그 목록을 만듭니다. 어떤 실행에 어떤 태그가 필요한지 결정합니다. 실행에 태그를 최대 5개까지 할당할 수 있긴 하지만 태그 라이브러리에는 태그가 무제한으로 포함될 수 있습니다. 각 태그는 최대 256자의 모든 문자열 값일 수 있으므로 태그는 거의 모든 비즈니스 개념을 설명할 수 있습니다.
2. 실행을 생성할 때 최대 5개 태그를 사용해 실행에 태그를 지정합니다.
3. `ListOpenWorkflowExecutions`, `ListClosedWorkflowExecutions`, `CountOpenWorkflowExecutions` 및 `CountClosedWorkflowExecutions` 작업으로 tagFilter 파라미터를 지정하여 특정 태그로 태그가 지정된 실행을 나열하거나 계산합니다. 작업은 지정된 태그를 기반으로 실행을 필터링합니다.

워크플로 실행에 태그를 연결하면 해당 태그는 실행에 영구히 연결되어 제거할 수 없습니다.

`tagFilter` 파라미터에는 `ListWorkflowExecutions`를 사용해 태그를 하나만 지정할 수 있습니다. 또한 태그 일치는 대/소문자를 구분하고 정확하게 일치하는 항목만 결과를 반환합니다.

다음과 같이 태그가 지정된 실행 2개를 이미 설정했다고 가정해 보겠습니다.

실행 이름	할당된 태그
실행-1	Consumer, 2011-February
실행-2	Wholesale, 2011-March

Consumer 태그에 대해 `ListOpenWorkflowExecutions`에서 반환하는 실행 목록을 필터링할 수 있습니다. `oldestDate` 및 `latestDate` 값은 Unix Time 값으로 지정됩니다.

```
https://swf.us-east-1.amazonaws.com
  RespondDecisionTaskCompleted
  {
    "domain":"867530901",
    "startTimeFilter":{
        "oldestDate":1262332800,
        "latestDate":1325348400
    },
    "tagFilter":{
      "tag":"Consumer"
```

```
        }
    }
```

태그를 사용하여 도메인에 대한 액세스 제어

IAM에서 Amazon SWF 도메인과 연결된 태그를 참조하여 Amazon Simple Workflow Service 도메인에 대한 액세스를 제어할 수 있습니다.

예를 들어 키가 있는 태그`environment`와 다음 조건이 있는 값을 포함하는 Amazon SWF 도메인을 제한`production`할 수 있습니다.

```
"Condition": {
    "StringEquals": {"aws:ResourceTag/environment": "production"}
}
```

자세한 내용은 다음을 참조하세요.

- IAM 태그를 사용한 액세스 제어
- 태그 기반 정책

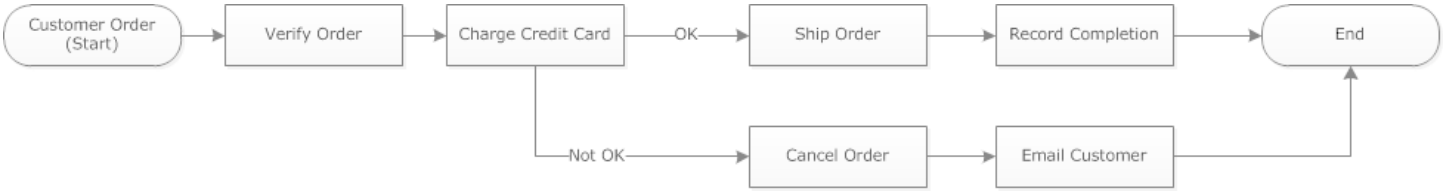
Amazon SWF를 사용하여 독점적 선택 구현

일부 시나리오에서는 이전 활동의 결과에 따라 여러 활동 세트를 예약하려고 할 수 있습니다. 전용 선택 패턴을 사용하면 애플리케이션의 복잡한 요구 사항을 충족하는 유연한 워크플로를 생성할 수 있습니다.

Amazon SWF에는 특정 배타적 선택 작업이 없습니다. 독점적인 선택을 구현하려면 결정자 로직을 작성하여 이전 활동의 결과를 기반으로 결정을 내려야 합니다. 독점적 선택의 적용 사례는 다음과 같습니다.

- 이전 활동의 결과가 실패인 경우 정리 활동 수행
- 고객이 기본 또는 고급 요금제를 구입했는지 여부에 따라 다른 활동 예약
- 고객의 주문 내역에 따라 다른 고객 인증 활동 수행

전자 상거래의 예에서 독점적 선택을 사용하면 신용카드 청구 결과에 따라 주문을 배송 또는 취소할 수 있습니다. 다음 그림에서 결정자는 신용카드가 성공적으로 청구된 경우 주문 발송 및 완료 기록 활동 작업을 예약합니다. 그렇지 않으면 주문 취소 및 고객에게 이메일 보내기 활동 작업을 예약합니다.



신용카드에 성공적으로 청구되면 결정자가 `ShipOrder` 활동을 예약합니다. 그렇지 않으면 결정자는 `CancelOrder` 활동을 예약합니다.

이 경우 내역을 해석하고 신용카드에 성공적으로 청구되었는지 여부를 확인하도록 결정자를 프로그래밍합니다. 이렇게 하려면 다음과 유사한 로직이 있어야 합니다.

```
IF lastEvent = "WorkflowExecutionStarted"
 addToDecisions ScheduleActivityTask(ActivityType = "VerifyOrderActivity")

ELSIF lastEvent = "ActivityTaskCompleted"
     AND ActivityType = "VerifyOrderActivity"
 addToDecisions ScheduleActivityTask(ActivityType = "ChargeCreditCardActivity")

#Successful Credit Card Charge Activities
ELSIF lastEvent = "ActivityTaskCompleted"
      AND ActivityType = "ChargeCreditCardActivity"
  addToDecisions ScheduleActivityTask(ActivityType = "ShipOrderActivity")

ELSIF lastEvent = "ActivityTaskCompleted"
     AND ActivityType = "ShipOrderActivity"
  addToDecisions ScheduleActivityTask(ActivityType = "RecordOrderCompletionActivity")

ELSIF lastEvent = "ActivityTaskCompleted"
     AND ActivityType = "RecordOrderCompletionActivity"
 addToDecisions CompleteWorkflowExecution

#Unsuccessful Credit Card Charge Activities
ELSIF lastEvent = "ActivityTaskFailed"
      AND ActivityType = "ChargeCreditCardActivity"
  addToDecisions ScheduleActivityTask(ActivityType = "CancelOrderActivity")

ELSIF lastEvent = "ActivityTaskCompleted"
     AND ActivityType = "CancelOrderActivity"
 addToDecisions ScheduleActivityTask(ActivityType = "EmailCustomerActivity")

ELSIF lastEvent = "ActivityTaskCompleted"
     AND ActivityType = "EmailCustomerActivity"
```

```
 addToDecisions CompleteWorkflowExecution

ENDIF
```

신용카드에 성공적으로 청구되면 결정자는 RespondDecisionTaskCompleted로 응답해 ShipOrder 활동을 예약해야 합니다.

```
https://swf.us-east-1.amazonaws.com
RespondDecisionTaskCompleted
{
  "taskToken": "12342e17-80f6-FAKE-TASK-TOKEN32f0223",
  "decisions":[
      {
          "decisionType":"ScheduleActivityTask",
          "scheduleActivityTaskDecisionAttributes":{
              "control":"OPTIONAL_DATA_FOR_DECIDER",
              "activityType":{
                  "name":"ShipOrder",
                  "version":"2.4"
              },
              "activityId":"3e2e6e55-e7c4-fee-deed-aa815722b7be",
              "scheduleToCloseTimeout":"3600",
              "taskList":{
                  "name":"SHIPPING"
              },
              "scheduleToStartTimeout":"600",
              "startToCloseTimeout":"3600",
              "heartbeatTimeout":"300",
              "input": "123 Main Street, Anytown, United States"
          }
      }
  ]
}
```

신용카드에 성공적으로 청구되지 않으면 결정자는 RespondDecisionTaskCompleted로 응답해 CancelOrder 활동을 예약해야 합니다.

```
https://swf.us-east-1.amazonaws.com
RespondDecisionTaskCompleted
{
  "taskToken": "12342e17-80f6-FAKE-TASK-TOKEN32f0223",
  "decisions":[
```

```
      {
          "decisionType":"ScheduleActivityTask",
          "scheduleActivityTaskDecisionAttributes":{
              "control":"OPTIONAL_DATA_FOR_DECIDER",
              "activityType":{
                  "name":"CancelOrder",
                  "version":"2.4"
              },
              "activityId":"3e2e6e55-e7c4-fee-deed-aa815722b7be",
              "scheduleToCloseTimeout":"3600",
              "taskList":{
                  "name":"CANCELLATIONS"
              },
              "scheduleToStartTimeout":"600",
              "startToCloseTimeout":"3600",
              "heartbeatTimeout":"300",
              "input": "Out of Stock"
          }
      }
  ]
}
```

Amazon SWF에서 RespondDecisionTaskCompleted 작업의 데이터를 확인할 수 있는 경우 Amazon SWF는 다음과 유사한 성공적인 HTTP 응답을 반환합니다.

```
HTTP/1.1 200 OK
Content-Length: 11
Content-Type: application/json
x-amzn-RequestId: 93cec6f7-0747-11e1-b533-79b402604df1
```

Amazon SWF의 타이머

타이머를 사용하면 일정 시간이 경과하면 결정자에게 알릴 수 있습니다.

결정 작업에 응답할 때 결정자는 StartTimer 결정을 사용해 응답할 수 있는 옵션이 있습니다. 이 결정은 경과 후 타이머가 만료되어야 하는 기간을 지정합니다. 지정된 시간이 경과되면 Amazon SWF는 워크플로 실행 내역에 TimerFired 이벤트를 추가하여 결정 작업을 예약합니다. 그런 다음 결정자는 이 정보를 사용해 추가 결정을 알립니다. 타이머의 일반적인 적용 사례 하나는 활동 작업 실행을 지연하는 것입니다. 예를 들어 고객이 품목 배송을 지연시키길 원할 수 있습니다.

Amazon SWF에서 활동 작업 취소

활동 작업 취소는 결정자에게 더 이상 수행할 필요가 없는 활동을 종료하도록 알립니다. Amazon SWF는 협력적 취소 메커니즘을 사용하며 실행 중인 활동 작업을 강제로 중단하지 않습니다. 취소 요청을 처리하도록 활동 작업자를 프로그래밍해야 합니다.

결정자는 결정 작업을 처리하는 중에 활동 작업 취소를 결정할 수 있습니다. 활동 작업을 취소하려면 결정자는 `RespondDecisionTaskCompleted` 작업을 `RequestCancelActivityTask` 결정으로 사용합니다.

활동 작업자가 아직 인수하지 않은 활동 작업은 서비스에서 취소합니다. 활동 작업자가 언제든지 작업을 인수할 수 있는 잠재적 경합 상태도 있습니다. 작업이 이미 활동 작업자에게 할당된 경우 활동 작업자는 해당 작업을 취소하라는 요청을 받습니다.

이 예에서는 워크플로 실행으로 신호를 보내 주문을 취소합니다.

```
https://swf.us-east-1.amazonaws.com
SignalWorkflowExecution
{"domain": "867530901",
 "workflowId": "20110927-T-1",
 "runId": "9ba33198-4b18-4792-9c15-7181fb3a8852",
 "signalName": "CancelOrder",
 "input": "order 3553"}
```

워크플로 실행이 신호를 수신하면 Amazon SWF는 다음과 유사한 성공적인 HTTP 응답을 반환합니다. Amazon SWF는 결정 작업을 생성하여 결정자에게 신호를 처리하도록 알립니다.

```
HTTP/1.1 200 OK
Content-Length: 0
Content-Type: application/json
x-amzn-RequestId: 6c0373ce-074c-11e1-9083-8318c48dee96
```

결정자가 결정 작업을 처리하고 내역에서 신호를 확인한 경우 결정자는 활동 ID가 `ShipOrderActivity0001`인 대기 중인 활동을 취소하려고 합니다. 활동 ID는 예약 활동 작업 이벤트의 워크플로 내역에 제공됩니다.

```
https://swf.us-east-1.amazonaws.com
RespondDecisionTaskCompleted
{
```

```
  "taskToken":"12342e17-80f6-FAKE-TASK-TOKEN32f0223",
  "decisions":[{
          "decisionType":"RequestCancelActivityTask",
          "RequestCancelActivityTaskDecisionAttributes":{
              "ActivityID":"ShipOrderActivity0001"
          }
      }
  ]
}
```

Amazon SWF가 취소 요청을 성공적으로 수신하면 다음과 유사한 성공적인 HTTP 응답을 반환합니다.

```
HTTP/1.1 200 OK
Content-Length: 0
Content-Type: application/json
x-amzn-RequestId: 6c0373ce-074c-11e1-9083-8318c48dee96
```

취소 시도는 내역에 `ActivityTaskCancelRequested` 이벤트로 기록됩니다.

작업이 성공적으로 취소되면 (`ActivityTaskCanceled`이벤트로 표시됨) 워크플로 실행 종료와 같이 결정자가 작업 취소에 뒤따라야 하는 적절한 단계를 취하도록 프로그래밍합니다.

활동 작업을 취소할 수 없는 경우(예: 작업이 완료, 실패 또는 취소하는 대신 시간이 초과된 경우) 결정자는 활동 결과를 수락하거나 사용 사례에 따라 필요한 정리 또는 완화 조치를 취해야 합니다.

활동 작업자가 이미 활동 작업을 인수한 경우 취소 요청은 작업-하트비트 메커니즘을 통해 전송됩니다. 활동 작업자는 정기적으로 `RecordActivityTaskHeartbeat`를 사용하여 Amazon SWF에 작업이 계속 진행 중임을 보고할 수 있습니다.

장시간 실행 작업에는 하트비트를 사용하는 것이 좋지만, 활동 작업자는 그렇게 하지 않아도 됩니다. 작업을 취소하려면 정기 하트비트를 기록해야 합니다. 작업자가 하트비트를 사용하지 않는 경우 작업을 취소할 수 없습니다.

결정자가 작업 취소를 요청한 경우 Amazon SWF는 `cancelRequest` 객체의 값을 true로 설정합니다. `cancelRequest` 객체는 `RecordActivityTaskHeartbeat`에 대한 응답으로 서비스에서 반환하는 `ActivityTaskStatus` 객체의 일부입니다.

Amazon SWF는 취소 요청된 활동 작업의 완료를 방해하지 않습니다. 따라서 취소 요청을 처리하는 방법은 활동에 따라 달라집니다. 요구 사항에 따라 활동 작업을 취소하거나 취소 요청을 무시하도록 활동 작업자를 프로그래밍하십시오.

활동 작업이 취소되었을 때 활동 작업자가 이를 표시하게 하려면 RespondActivityTaskCanceled로 응답하도록 프로그래밍합니다. 활동 작업자가 작업을 완료하게 하려면 표준 RespondActivityTaskCompleted로 응답하도록 프로그래밍합니다.

RespondActivityTaskCompleted 또는 RespondActivityTaskCanceled 요청을 받은 Amazon SWF는 워크플로 실행 내역을 업데이트하고 결정 작업을 예약해 결정자에게 알립니다.

결정 작업을 처리하고 추가 결정이 있으면 표시하도록 결정자를 프로그래밍합니다. 활동 작업이 취소되면 워크플로 실행을 계속하거나 닫는 데 필요한 작업을 수행하도록 결정자를 프로그래밍합니다. 활동 작업이 취소되지 않은 경우, 결과를 수락 또는 무시하거나 필요한 정리 단계를 예약하도록 결정자를 프로그래밍합니다.

Amazon Simple Workflow Service 보안

이 단원에서는 Amazon Simple Workflow Service 보안 및 인증에 대해 설명합니다.

주제

Amazon SWF는 IAM을 사용하여 다른 AWS 서비스 및 리소스에 대한 액세스를 제어합니다. IAM의 작동 방식에 대한 개요는 IAM 사용 설명서의 액세스 관리 개요를 참조하세요. 보안 인증에 대한 개요는 Amazon Web Services 일반 참조의 AWS 보안 인증을 참조하십시오.

Amazon Simple Workflow Service 데이터 보호

AWS 공동 책임 모델 Amazon Simple Workflow Service의 데이터 보호에 적용됩니다. 이 모델에 설명된 대로 AWS 는 모든를 실행하는 글로벌 인프라를 보호할 책임이 있습니다 AWS 클라우드. 사용자는 인프라에서 호스팅되는 콘텐츠를 관리해야 합니다. 사용하는 AWS 서비스 의 보안 구성과 관리 태스크에 대한 책임도 사용자에게 있습니다. 데이터 프라이버시에 대한 자세한 내용은 데이터 프라이버시 FAQ를 참조하세요. 유럽의 데이터 보호에 대한 자세한 내용은 AWS 보안 블로그의 AWS 공동 책임 모델 및 GDPR 블로그 게시물을 참조하세요.

데이터 보호를 위해 자격 증명을 보호하고 AWS 계정 AWS IAM Identity Center 또는 AWS Identity and Access Management (IAM)를 사용하여 개별 사용자를 설정하는 것이 좋습니다. 이렇게 하면 개별 사용자에게 자신의 직무를 충실히 이행하는 데 필요한 권한만 부여됩니다. 또한 다음과 같은 방법으로 데이터를 보호하는 것이 좋습니다.

- 각 계정에 다중 인증(MFA)을 사용하세요.
- SSL/TLS를 사용하여 AWS 리소스와 통신합니다. TLS 1.2는 필수이며 TLS 1.3을 권장합니다.

- 를 사용하여 API 및 사용자 활동 로깅을 설정합니다 AWS CloudTrail. CloudTrail 추적을 사용하여 AWS 활동을 캡처하는 방법에 대한 자세한 내용은 AWS CloudTrail 사용 설명서의 CloudTrail 추적 작업을 참조하세요.
- 내의 모든 기본 보안 제어와 함께 AWS 암호화 솔루션을 사용합니다 AWS 서비스.
- Amazon S3에 저장된 민감한 데이터를 검색하고 보호하는 데 도움이 되는 Amazon Macie와 같은 고급 관리형 보안 서비스를 사용하세요.
- 명령줄 인터페이스 또는 API를 AWS 통해에 액세스할 때 FIPS 140-3 검증 암호화 모듈이 필요한 경우 FIPS 엔드포인트를 사용합니다. 사용 가능한 FIPS 엔드포인트에 대한 자세한 내용은 Federal Information Processing Standard(FIPS) 140-3을 참조하세요.

고객의 이메일 주소와 같은 기밀 정보나 중요한 정보는 태그나 이름 필드와 같은 자유 형식 텍스트 필드에 입력하지 않는 것이 좋습니다. 여기에는 Amazon SWF 또는 기타 AWS 서비스 에서 콘솔 AWS CLI, API 또는 AWS SDKs를 사용하여 작업하는 경우가 포함됩니다. 이름에 사용되는 태그 또는 자유 형식 텍스트 필드에 입력하는 모든 데이터는 청구 또는 진단 로그에 사용될 수 있습니다. 외부 서버에 URL을 제공할 때 해당 서버에 대한 요청을 검증하기 위해 자격 증명을 URL에 포함해서는 안 됩니다.

Amazon Simple Workflow Service 암호화

저장 시 암호화

Amazon SWF는 저장 데이터를 항상 암호화합니다. Amazon Simple Workflow Service의 데이터는 투명 서버 측 암호화를 사용하여 저장 상태로 암호화됩니다. 이를 사용하면 중요한 데이터 보호와 관련된 운영 부담 및 복잡성을 줄일 수 있습니다. 저장 시 암호화를 사용하면 암호화 규정 준수 및 규제 요구 사항을 충족하는 보안에 민감한 애플리케이션을 구축할 수 있습니다.

전송 중 암호화

Amazon SWF와 기타 서비스 간에 전달되는 모든 데이터는 전송 계층 보안(TLS)을 사용하여 암호화됩니다.

Amazon Simple Workflow Service의 Identity and Access Management

Amazon SWF에 액세스하려면가 요청을 인증하는 데 사용할 AWS 수 있는 자격 증명이 필요합니다. 이러한 자격 증명에는 다른 AWS 리소스에서 이벤트 데이터를 검색하는 등 AWS 리소스에 액세스할 수 있는 권한이 있어야 합니다. 다음 섹션에서는 AWS Identity and Access Management (IAM) 및

Amazon SWF를 사용하여 리소스에 대한 액세스를 제어함으로써 리소스를 보호하는 방법에 대해 자세히 설명합니다.

AWS Identity and Access Management (IAM)는 관리자가 AWS 리소스에 대한 액세스를 안전하게 제어하는 데 도움이 AWS 서비스 되는 입니다. IAM 관리자는 어떤 사용자가 Amazon SWF 리소스를 사용할 수 있도록 인증(로그인)되고 권한이 부여(권한 있음)될 수 있는지 제어합니다. IAM은 추가 비용 없이 사용할 수 AWS 서비스 있는 입니다.

주제

대상

AWS Identity and Access Management (IAM)를 사용하는 방법은 역할에 따라 다릅니다.

- 서비스 사용자 - 기능에 액세스할 수 없는 경우 관리자에게 권한 요청(참조Amazon Simple Workflow Service ID 및 액세스 문제 해결)
- 서비스 관리자 - 사용자 액세스 결정 및 권한 요청 제출(Amazon Simple Workflow Service가 IAM으로 작동하는 방식 참조)
- IAM 관리자 - 액세스를 관리하기 위한 정책 작성(Amazon Simple Workflow Service의 자격 증명 기반 정책 예 참조)

ID를 통한 인증

인증은 AWS 자격 증명으로에 로그인하는 방법입니다. AWS 계정 루트 사용자, IAM 사용자 또는 IAM 역할을 수임하여 인증되어야 합니다.

AWS IAM Identity Center (IAM Identity Center), Single Sign-On 인증 또는 Google/Facebook 자격 증명과 같은 자격 증명 소스의 자격 증명을 사용하여 페더레이션 자격 증명으로 로그인할 수 있습니다. 로그인하는 방법에 대한 자세한 내용은 AWS 로그인 사용 설명서의 AWS 계정에 로그인하는 방법 섹션을 참조하세요.

프로그래밍 방식 액세스를 위해는 요청에 암호화 방식으로 서명할 수 있는 SDK 및 CLI를 AWS 제공합니다. 자세한 내용은 IAM 사용 설명서의 API 요청용AWS Signature Version 4 섹션을 참조하세요.

AWS 계정 루트 사용자

를 생성할 때 모든 AWS 서비스 및 리소스에 대한 완전한 액세스 권한이 있는 AWS 계정 theroot 사용자라는 하나의 로그인 자격 증명으로 AWS 계정시작합니다. 일상적인 태스크에 루트 사용자를 사용하지 않을 것을 강력히 권장합니다. 루트 사용자 자격 증명이 필요한 작업은 IAM 사용 설명서의 루트 사용자 자격 증명이 필요한 작업 섹션을 참조하세요.

페더레이션 ID

가장 좋은 방법은 인간 사용자에게 자격 증명 공급자와의 페더레이션을 사용하여 임시 자격 증명을 AWS 서비스 사용하여에 액세스하도록 요구하는 것입니다.

페더레이션 자격 증명은 엔터프라이즈 디렉터리, 웹 자격 증명 공급자 또는 자격 증명 소스의 자격 증명을 AWS 서비스 사용하여에 Directory Service 액세스하는 사용자입니다. 페더레이션 ID는 임시 자격 증명을 제공하는 역할을 수임합니다.

중앙 집중식 액세스 관리를 위해 AWS IAM Identity Center를 추천합니다. 자세한 정보는 AWS IAM Identity Center 사용 설명서의 What is IAM Identity Center?를 참조하세요.

IAM 사용자 및 그룹

IAM 사용자는 단일 개인 또는 애플리케이션에 대한 특정 권한을 가진 ID입니다. 장기 자격 증명이 있는 IAM 사용자 대신 임시 자격 증명을 사용하는 것이 좋습니다. 자세한 내용은 IAM 사용 설명서의 자격 증명 공급자와의 페더레이션을 사용하여 임시 자격 증명을 AWS 사용하여에 액세스하도록 인간 사용자에게 요구하기를 참조하세요.

IAM 그룹은 IAM 사용자 모음을 지정하고 대규모 사용자 집합에 대한 관리 권한을 더 쉽게 만듭니다. 자세한 내용은 IAM 사용 설명서의 IAM 사용자 사용 사례 섹션을 참조하세요.

IAM 역할

IAM 역할은 임시 자격 증명을 제공하는 특정 권한이 있는 자격 증명입니다. 사용자에서 IAM 역할(콘솔)로 전환하거나 또는 API 작업을 호출하여 역할을 수임할 수 있습니다. AWS CLI AWS 자세한 내용은 IAM 사용 설명서의 역할 수임 방법을 참조하세요.

IAM 역할은 페더레이션 사용자 액세스, 임시 IAM 사용자 권한, 교차 계정 액세스, 교차 서비스 액세스 및 Amazon EC2에서 실행되는 애플리케이션에 유용합니다. 자세한 내용은 IAM 사용 설명서의 교차 계정 리소스 액세스를 참조하세요.

정책을 사용하여 액세스 관리

정책을 AWS 생성하고 자격 증명 또는 리소스에 연결하여 AWS 에서 액세스를 제어합니다. 정책은 자격 증명 또는 리소스와 연결될 때 권한을 정의합니다.는 보안 주체가 요청할 때 이러한 정책을 AWS 평가합니다. 대부분의 정책은에 JSON 문서 AWS 로 저장됩니다. JSON 정책 문서에 대한 자세한 내용은 IAM 사용 설명서의 JSON 정책 개요 섹션을 참조하세요.

정책을 사용하여 관리자는 어떤 보안 주체가 어떤 리소스에 대해 어떤 조건에서 작업을 수행할 수 있는지 정의하여 누가 무엇을 액세스할 수 있는지 지정합니다.

기본적으로 사용자 및 역할에는 어떠한 권한도 없습니다. IAM 관리자는 IAM 정책을 생성하고 사용자가 수임할 수 있는 역할에 추가합니다. IAM 정책은 작업을 수행하기 위해 사용하는 방법과 관계없이 작업에 대한 권한을 정의합니다.

ID 기반 정책

ID 기반 정책은 ID(사용자, 사용자 그룹 또는 역할)에 연결하는 JSON 권한 정책 문서입니다. 이러한 정책은 자격 증명이 수행할 수 있는 작업, 대상 리소스 및 이에 관한 조건을 제어합니다. ID 기반 정책

을 생성하는 방법을 알아보려면 IAM 사용 설명서에서 고객 관리형 정책으로 사용자 지정 IAM 권한 정의를 참조하세요.

ID 기반 정책은 인라인 정책(단일 ID에 직접 포함) 또는 관리형 정책(여러 ID에 연결된 독립 실행형 정책)일 수 있습니다. 관리형 정책 또는 인라인 정책을 선택하는 방법을 알아보려면 IAM 사용 설명서의 관리형 정책 및 인라인 정책 중에서 선택 섹션을 참조하세요.

리소스 기반 정책

리소스 기반 정책은 리소스에 연결하는 JSON 정책 설명서입니다. 예를 들어 IAM 역할 신뢰 정책 및 Amazon S3 버킷 정책이 있습니다. 리소스 기반 정책을 지원하는 서비스에서 서비스 관리자는 이러한 정책을 사용하여 특정 리소스에 대한 액세스를 통제할 수 있습니다. 리소스 기반 정책에서 보안 주체를 지정해야 합니다.

리소스 기반 정책은 해당 서비스에 있는 인라인 정책입니다. 리소스 기반 정책에서는 IAM의 AWS 관리형 정책을 사용할 수 없습니다.

기타 정책 유형

AWS 는 보다 일반적인 정책 유형에서 부여한 최대 권한을 설정할 수 있는 추가 정책 유형을 지원합니다.

- 권한 경계 - ID 기반 정책에서 IAM 엔터티에 부여할 수 있는 최대 권한을 설정합니다. 자세한 정보는 IAM 사용 설명서의 IAM 엔터티의 권한 범위를 참조하세요.
- 서비스 제어 정책(SCP) - AWS Organizations내 조직 또는 조직 단위에 대한 최대 권한을 지정합니다. 자세한 내용은AWS Organizations 사용 설명서의 서비스 제어 정책을 참조하세요.
- 리소스 제어 정책(RCP) – 계정의 리소스에 사용할 수 있는 최대 권한을 설정합니다. 자세한 내용은 AWS Organizations 사용 설명서의 리소스 제어 정책(RCP)을 참조하세요.
- 세션 정책 – 역할 또는 페더레이션 사용자에 대해 임시 세션을 프로그래밍 방식으로 생성할 때 파라미터로 전달하는 고급 정책입니다. 자세한 내용은 IAM 사용 설명서의 세션 정책을 참조하세요.

여러 정책 유형

여러 정책 유형이 요청에 적용되는 경우, 결과 권한은 이해하기가 더 복잡합니다. 에서 여러 정책 유형이 관련될 때 요청을 허용할지 여부를 AWS 결정하는 방법을 알아보려면 IAM 사용 설명서의 정책 평가 로직을 참조하세요.

액세스 통제

요청을 인증하는 데 유효한 보안 인증이 있더라도 권한이 없다면 Amazon SWF 리소스를 생성하거나 액세스할 수 없습니다. 예를 들어 Amazon SWF 규칙과 연결된 AWS Lambda Amazon Simple Notification Service(Amazon SNS) 및 Amazon Simple Queue Service(Amazon SQS) 대상을 호출할 수 있는 권한이 있어야 합니다.

다음 섹션에서는 Amazon SWF에 대한 권한을 관리하는 방법을 설명합니다. 먼저 개요를 읽어보면 도움이 됩니다.

- 기본 원칙
- Amazon SWF IAM 정책
- Amazon SWF에 대한 정책 작성

Amazon SWF의 정책 작업

정책 작업 지원: 예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

JSON 정책의 `Action`요소는 정책에서 액세스를 허용하거나 거부하는 데 사용할 수 있는 작업을 설명합니다. 연결된 작업을 수행할 수 있는 권한을 부여하기 위한 정책에 작업을 포함하세요.

Amazon SWF 작업의 목록을 보려면 서비스 권한 부여 참조의 Amazon Simple Workflow Service에서 정의한 작업을 참조하세요.

Amazon SWF의 정책 작업은 작업 앞에 다음 접두사를 사용합니다.

```
swf
```

단일 문에서 여러 작업을 지정하려면 쉼표로 구분합니다.

```
"Action": [
      "swf:action1",
      "swf:action2"
         ]
```

Amazon SWF 자격 증명 기반 정책 예제를 보려면 Amazon Simple Workflow Service의 자격 증명 기반 정책 예 섹션을 참조하세요.

Amazon SWF의 정책 리소스

정책 리소스 지원: 예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

`Resource` JSON 정책 요소는 작업이 적용되는 하나 이상의 객체를 지정합니다. 모범 사례에 따라 Amazon 리소스 이름(ARN)을 사용하여 리소스를 지정합니다. 리소스 수준 권한을 지원하지 않는 작업의 경우, 와일드카드(*)를 사용하여 해당 문이 모든 리소스에 적용됨을 나타냅니다.

```
"Resource": "*"
```

Amazon SWF 리소스 유형 및 해당 ARN의 목록을 보려면 서비스 권한 부여 참조의 Amazon Simple Workflow Service에서 정의한 작업을 참조하세요. 각 리소스의 ARN을 지정할 수 있는 작업을 알아보려면 Amazon Simple Workflow Service에서 정의한 리소스를 참조하세요.

Amazon SWF 자격 증명 기반 정책 예제를 보려면 Amazon Simple Workflow Service의 자격 증명 기반 정책 예 섹션을 참조하세요.

Amazon SWF의 정책 조건 키

서비스별 정책 조건 키 지원: 예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

`Condition` 요소는 정의된 기준에 따라 문이 실행되는 시기를 지정합니다. 같음(equals) 또는 미만(less than)과 같은 조건 연산자를 사용하여 정책의 조건을 요청의 값과 일치시키는 조건식을 생성할 수 있습니다. 모든 AWS 전역 조건 키를 보려면 IAM 사용 설명서의 AWS 전역 조건 컨텍스트 키를 참조하세요.

Amazon SWF 조건 키 목록을 보려면 서비스 권한 부여 참조의 Amazon Simple Workflow Service에 사용되는 조건 키를 참조하세요. 조건 키를 사용할 수 있는 작업과 리소스를 알아보려면 Amazon Simple Workflow Service에서 정의한 리소스를 참조하세요.

Amazon SWF 자격 증명 기반 정책 예제를 보려면 Amazon Simple Workflow Service의 자격 증명 기반 정책 예 섹션을 참조하세요.

Amazon SWF의 ACL

ACL 지원: 아니요

액세스 제어 목록(ACL)은 어떤 보안 주체(계정 멤버, 사용자 또는 역할)가 리소스에 액세스할 수 있는 권한을 가지고 있는지를 제어합니다. ACL은 JSON 정책 문서 형식을 사용하지 않지만 리소스 기반 정책과 유사합니다.

Amazon SWF의 ABAC

ABAC 지원(정책의 태그): 부분적

속성 기반 액세스 제어(ABAC)는 태그라고 불리는 속성을 기반으로 권한을 정의하는 권한 부여 전략입니다. IAM 엔터티 및 AWS 리소스에 태그를 연결한 다음 보안 주체의 태그가 리소스의 태그와 일치할 때 작업을 허용하는 ABAC 정책을 설계할 수 있습니다.

태그에 근거하여 액세스를 제어하려면 `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`또는 `aws:TagKeys`조건 키를 사용하여 정책의 조건 요소에 태그 정보를 제공합니다.

서비스가 모든 리소스 유형에 대해 세 가지 조건 키를 모두 지원하는 경우, 값은 서비스에 대해 예입니다. 서비스가 일부 리소스 유형에 대해서만 세 가지 조건 키를 모두 지원하는 경우, 값은 부분적입니다.

ABAC에 대한 자세한 내용은 IAM 사용 설명서의 ABAC 권한 부여를 통한 권한 정의를 참조하세요. ABAC 설정 단계가 포함된 자습서를 보려면 IAM 사용 설명서의 속성 기반 액세스 제어(ABAC) 사용을 참조하세요.

Amazon SWF에서 임시 보안 인증 사용

임시 자격 증명 지원: 예

임시 자격 증명은 AWS 리소스에 대한 단기 액세스를 제공하며 페더레이션 또는 전환 역할을 사용할 때 자동으로 생성됩니다. 장기 액세스 키를 사용하는 대신 임시 자격 증명을 동적으로 생성하는 것이 AWS 좋습니다. 자세한 내용은 IAM 사용 설명서의 IAM의 임시 보안 자격 증명 및 IAM으로 작업하는 AWS 서비스 섹션을 참조하세요.

Amazon SWF에 대한 교차 서비스 보안 주체 권한

전달 액세스 세션(FAS) 지원: 예

전달 액세스 세션(FAS)은를 호출하는 보안 주체의 권한을 다운스트림 서비스에 AWS 서비스 대한 요청과 AWS 서비스함께 사용합니다. FAS 요청 시 정책 세부 정보는 전달 액세스 세션을 참조하세요.

Amazon SWF의 서비스 역할

서비스 역할 지원: 예

서비스 역할은 서비스가 사용자를 대신하여 작업을 수행하는 것으로 가정하는 IAM 역할입니다. IAM 관리자는 IAM 내에서 서비스 역할을 생성, 수정 및 삭제할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 AWS 서비스 AWS에 권한을 위임할 역할 생성을 참조하세요.

> **Warning**
>
> 서비스 역할에 대한 권한을 변경하면 Amazon SWF 기능이 중단될 수 있습니다. Amazon SWF에서 관련 지침을 제공하는 경우에만 서비스 역할을 편집합니다.

Amazon SWF의 서비스 연결 역할

서비스 연결 역할 지원: 아니요

서비스 연결 역할은에 연결된 서비스 역할의 한 유형입니다 AWS 서비스. 서비스는 사용자를 대신하여 작업을 수행하기 위해 역할을 수임할 수 있습니다. 서비스 연결 역할은에 표시 AWS 계정 되며 서비스가 소유합니다. IAM 관리자는 서비스 연결 역할의 권한을 볼 수 있지만 편집은 할 수 없습니다.

서비스 연결 역할 생성 또는 관리에 대한 자세한 내용은 IAM으로 작업하는AWS 서비스를 참조하세요. 서비스 연결 역할 열에서 Yes가 포함된 서비스를 테이블에서 찾습니다. 해당 서비스에 대한 서비스 연결 역할 설명서를 보려면 예(Yes) 링크를 선택합니다.

Amazon SWF의 자격 증명 기반 정책

ID 기반 정책 지원: 예

ID 기반 정책은 IAM 사용자, 사용자 그룹 또는 역할과 같은 ID에 연결할 수 있는 JSON 권한 정책 문서입니다. 이러한 정책은 사용자 및 역할이 어떤 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는지를 제어합니다. 자격 증명 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서에서 고객 관리형 정책으로 사용자 지정 IAM 권한 정의를 참조하세요.

IAM ID 기반 정책을 사용하면 허용되거나 거부되는 작업과 리소스뿐 아니라 작업이 허용되거나 거부되는 조건을 지정할 수 있습니다. JSON 정책에서 사용할 수 있는 모든 요소에 대해 알아보려면 IAM 사용 설명서의 IAM JSON 정책 요소 참조를 참조하세요.

Amazon SWF의 자격 증명 기반 정책 예

Amazon SWF 자격 증명 기반 정책 예제를 보려면 Amazon Simple Workflow Service의 자격 증명 기반 정책 예 섹션을 참조하세요.

Amazon SWF 내의 리소스 기반 정책

리소스 기반 정책 지원: 아니요

리소스 기반 정책은 리소스에 연결하는 JSON 정책 설명서입니다. 리소스 기반 정책의 예제는 IAM 역할 신뢰 정책과 Amazon S3 버킷 정책입니다. 리소스 기반 정책을 지원하는 서비스에서 서비스 관리자는 이러한 정책을 사용하여 특정 리소스에 대한 액세스를 통제할 수 있습니다. 정책이 연결된 리소스의 경우 정책은 지정된 보안 주체가 해당 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는지를 정의합니다. 리소스 기반 정책에서 보안 주체를 지정해야 합니다. 보안 주체에는 계정, 사용자, 역할, 페더레이션 사용자 또는가 포함될 수 있습니다 AWS 서비스.

교차 계정 액세스를 활성화하려는 경우, 전체 계정이나 다른 계정의 IAM 개체를 리소스 기반 정책의 보안 주체로 지정할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 IAM에서 교차 계정 리소스 액세스를 참조하세요.

Amazon Simple Workflow Service가 IAM으로 작동하는 방식

IAM을 사용하여 Amazon SWF에 대한 액세스를 관리하기 전에 Amazon SWF에서 사용할 수 있는 IAM 기능에 대해 알아봅니다.

Amazon Simple Workflow Service와 함께 사용할 수 있는 IAM 기능

IAM 특성	Amazon SWF 지원
자격 증명 기반 정책	예
리소스 기반 정책	아니요
정책 작업	예
정책 리소스	예
정책 조건 키(서비스별)	예
ACL	아니요

IAM 특성	Amazon SWF 지원
ABAC(정책 내 태그)	부분적
임시 자격 증명	예
엔터티 권한	예
서비스 역할	예
서비스 연결 역할	아니요

Amazon SWF 및 기타 AWS 서비스가 대부분의 IAM 기능과 작동하는 방식을 개괄적으로 알아보려면 IAM 사용 설명서의 AWS IAM으로 작업하는 서비스를 참조하세요.

Amazon Simple Workflow Service의 자격 증명 기반 정책 예

기본적으로 사용자 및 역할은 Amazon SWF 리소스를 생성하거나 수정할 수 있는 권한이 없습니다. 사용자에게 사용자가 필요한 리소스에서 작업을 수행할 권한을 부여하려면 IAM 관리자가 IAM 정책을 생성하면 됩니다.

이러한 예제 JSON 정책 문서를 사용하여 IAM ID 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서의 IAM 정책 생성(콘솔)을 참조하세요.

각 리소스 유형에 대한 ARN 형식을 포함하여 Amazon SWF에서 정의한 작업 및 리소스 유형에 대한 자세한 내용은 서비스 권한 부여 참조에서 Amazon Simple Workflow Service에 대한 작업, 리소스 및 조건 키를 참조하세요.

주제

- 정책 모범 사례
- Amazon SWF 콘솔 사용
- 사용자가 자신의 고유한 권한을 볼 수 있도록 허용

정책 모범 사례

ID 기반 정책에 따라 계정에서 사용자가 Amazon SWF 리소스를 생성, 액세스 또는 삭제할 수 있는지 여부가 결정됩니다. 이 작업으로 인해 AWS 계정에 비용이 발생할 수 있습니다. ID 기반 정책을 생성하거나 편집할 때는 다음 지침과 권장 사항을 따르세요.

- AWS 관리형 정책을 시작하고 최소 권한으로 전환 - 사용자 및 워크로드에 권한 부여를 시작하려면 많은 일반적인 사용 사례에 대한 권한을 부여하는 AWS 관리형 정책을 사용합니다. 에서 사용할 수 있습니다 AWS 계정. 사용 사례에 맞는 AWS 고객 관리형 정책을 정의하여 권한을 추가로 줄이는 것이 좋습니다. 자세한 내용은 IAM 사용 설명서의 AWS 관리형 정책 또는 AWS 직무에 대한 관리형 정책을 참조하세요.
- 최소 권한 적용 – IAM 정책을 사용하여 권한을 설정하는 경우, 작업을 수행하는 데 필요한 권한만 부여합니다. 이렇게 하려면 최소 권한으로 알려진 특정 조건에서 특정 리소스에 대해 수행할 수 있는 작업을 정의합니다. IAM을 사용하여 권한을 적용하는 방법에 대한 자세한 정보는 IAM 사용 설명서에 있는 IAM의 정책 및 권한을 참조하세요.
- IAM 정책의 조건을 사용하여 액세스 추가 제한 – 정책에 조건을 추가하여 작업 및 리소스에 대한 액세스를 제한할 수 있습니다. 예를 들어, SSL을 사용하여 모든 요청을 전송해야 한다고 지정하는 정책 조건을 작성할 수 있습니다. AWS 서비스와 같은 특정를 통해 사용되는 경우 조건을 사용하여 서비스 작업에 대한 액세스 권한을 부여할 수도 있습니다 CloudFormation. 자세한 내용은 IAM 사용 설명서의 IAM JSON 정책 요소: 조건을 참조하세요.
- IAM Access Analyzer를 통해 IAM 정책을 확인하여 안전하고 기능적인 권한 보장 - IAM Access Analyzer에서는 IAM 정책 언어(JSON)와 모범 사례가 정책에서 준수되도록 새로운 및 기존 정책을 확인합니다. IAM Access Analyzer는 100개 이상의 정책 확인 항목과 실행 가능한 추천을 제공하여 안전하고 기능적인 정책을 작성하도록 돕습니다. 자세한 내용은 IAM 사용 설명서의 IAM Access Analyzer에서 정책 검증을 참조하세요.
- 다중 인증(MFA) 필요 -에서 IAM 사용자 또는 루트 사용자가 필요한 시나리오가 있는 경우 추가 보안을 위해 MFA를 AWS 계정켭니다. API 작업을 직접적으로 호출할 때 MFA가 필요하면 정책에 MFA 조건을 추가합니다. 자세한 내용은 IAM 사용 설명서의 MFA를 통한 보안 API 액세스를 참조하세요.

IAM의 모범 사례에 대한 자세한 내용은 IAM 사용 설명서의 IAM의 보안 모범 사례를 참조하세요.

Amazon SWF 콘솔 사용

Amazon Simple Workflow Service 콘솔에 액세스하려면 최소한의 권한 집합이 있어야 합니다. 이러한 권한은에서 Amazon SWF 리소스에 대한 세부 정보를 나열하고 볼 수 있도록 허용해야 합니다 AWS 계정. 최소 필수 권한보다 더 제한적인 ID 기반 정책을 생성하는 경우, 콘솔이 해당 정책에 연결된 엔티티(사용자 또는 역할)에 대해 의도대로 작동하지 않습니다.

AWS CLI 또는 AWS API만 호출하는 사용자에게 최소 콘솔 권한을 허용할 필요는 없습니다. 대신, 수행하려는 API 작업과 일치하는 작업에만 액세스할 수 있도록 합니다.

사용자와 역할이 여전히 Amazon SWF 콘솔을 사용할 수 있도록 하려면 Amazon SWF *ConsoleAccess* 또는 *ReadOnly* AWS 관리형 정책도 엔터티에 연결합니다. 자세한 내용은 IAM 사용 설명서의 사용자에게 권한 추가를 참조하세요.

사용자가 자신의 고유한 권한을 볼 수 있도록 허용

이 예제는 IAM 사용자가 자신의 사용자 ID에 연결된 인라인 및 관리형 정책을 볼 수 있도록 허용하는 정책을 생성하는 방법을 보여줍니다. 이 정책에는 콘솔에서 또는 AWS CLI 또는 AWS API를 사용하여 프로그래밍 방식으로이 작업을 완료할 수 있는 권한이 포함됩니다.

```
{
    "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": "*"
        }
    ]
}
```

기본 원칙

Amazon SWF 액세스 제어는 주로 다음 두 가지 권한 유형을 기반으로 합니다.

- 리소스 권한: 사용자가 액세스할 수 있는 Amazon SWF 리소스를 지정합니다.

  리소스 권한은 도메인에 대해서만 표현할 수 있습니다.
- API 권한: 사용자가 직접적으로 호출할 수 있는 Amazon SWF 작업을 지정합니다.

가장 간단한 접근 방식은 모든 도메인에서 Amazon SWF 작업을 직접적으로 호출하여 전체 계정 액세스를 허용하거나 액세스를 완전히 거부하는 것입니다. 그러나 IAM에서는 액세스 제어에 대해 보다 세부적인 접근 방식을 지원하며, 이는 일반적으로 더욱 유용합니다. 예를 들면, 다음과 같이 할 수 있습니다.

- 지정된 도메인에 한해 모든 Amazon SWF 작업을 제한 없이 직접적으로 호출할 수 있도록 허용합니다. 이러한 정책을 사용해 개발 중인 워크플로 애플리케이션이 "샌드박스" 도메인 내에서만 모든 작업을 사용할 수 있도록 허용할 수 있습니다.
- 사용자가 모든 도메인에 액세스하도록 허용하되 API를 사용하는 방법을 제한합니다. 이러한 정책을 사용해 "감사자" 애플리케이션이 모든 도메인에서 API를 호출하도록 허용하되 읽기 액세스만 허용합니다.
- 사용자가 특정 도메인에서 제한된 작업 세트만 호출하도록 허용합니다. 이러한 정책을 사용해 워크플로 시작자가 지정된 도메인 내에서 `StartWorkflowExecution` 작업만 호출하도록 허용할 수 있습니다.

Amazon SWF 액세스 제어는 다음 원칙을 기반으로 합니다.

- 액세스 제어 결정은 IAM 정책만을 기반으로 하며, 모든 정책 감사 및 조작은 IAM을 통해 완료됩니다.
- 액세스 제어 모델은 기본 거부 정책을 사용하여 명시적으로 허용되지 않은 모든 액세스가 거부됩니다.
- 워크플로의 액터에 적절한 IAM 정책을 연결하여 Amazon SWF 리소스에 대한 액세스를 제어합니다.
- 리소스 권한은 도메인에 대해서만 표현할 수 있습니다.
- 하나 이상의 파라미터에 조건을 적용해 일부 작업의 사용을 추가로 제한할 수 있습니다.
- RespondDecisionTaskCompleted 사용 권한을 부여할 때 해당 작업에 포함된 결정 목록에 대한 권한도 표현할 수 있습니다.

각 결정에는 정규 API 호출과 매우 유사한 파라미터가 하나 이상 있습니다. 정책을 가급적 판독 가능하게 하려면 일부 파라미터에 대한 조건 적용을 비롯해 정책이 실제 API 호출인 것처럼 결정에 대한 권한을 표현할 수 있습니다. 이러한 권한 유형을 의사 API 권한이라고 합니다.

조건을 사용해 제한할 수 있는 정규 및 의사 API 파라미터에 대한 간단한 설명은 API 요약 단원을 참조하십시오.

Amazon SWF IAM 정책

IAM 정책에는 정책을 정의하는 요소 세트가 포함된 `Statement` 요소가 하나 이상 포함되어 있습니다. 전체 요소 목록 및 정책 구성 방법에 대한 일반적인 설명은 액세스 정책 언어를 참조하십시오. Amazon SWF 액세스 제어는 다음 요소를 기반으로 합니다.

Effect

(필수)명령문 `deny` 또는 `allow`의 효과.

> **Note**
> 액세스를 명시적으로 허용해야 합니다. 기본적으로 IAM은 액세스를 거부합니다.

Resource

(필수) 명령문이 적용되는 리소스, 즉 사용자가 상호 작용할 수 있는 AWS 서비스의 엔터티입니다.

리소스 권한은 도메인에 대해서만 표현할 수 있습니다. 예를 들어, 정책이 계정 내 특정 도메인에 대한 액세스만 허용할 수 있습니다. 도메인에 대한 권한을 표현하려면 `Resource`를 도메인의 Amazon 리소스 이름(ARN)으로 설정합니다. 이 이름의 형식은 "arn:aws:swf:*Region*:*AccountID*:/domain/*DomainName*"입니다. *##*은 AWS 리전이고, *AccountID*는 대시가 없는 계정 ID이며, *DomainName*은 도메인 이름입니다.

작업

(필수) 명령문이 적용되는 작업으로, *serviceId*:*action* 형식을 사용해 지칭합니다. Amazon SWF의 경우 *serviceID*를 `swf`로 설정합니다. 예를 들어, `swf:StartWorkflowExecution`은 StartWorkflowExecution 작업을 가리키며 워크플로우를 시작하도록 허용된 사용자를 제어하는 데 사용됩니다.

RespondDecisionTaskCompleted 사용 권한을 부여하는 경우, 포함된 결정 목록에 대한 액세스도 `Action`으로 제어해 의사 API에 대한 권한을 표현할 수 있습니다. IAM은 기본적으로 액세스를 거부하기 때문에 결정자의 결정은 명시적으로 허용해야 합니다. 그렇지 않으면 결정이 수락되지 않습니다. * 값을 사용해 모든 결정을 허용할 수 있습니다.

조건

(선택 사항) 하나 이상의 작업 파라미터에 대한 제약을 표현합니다. 이러한 제약은 허용된 값을 제한합니다.

Amazon SWF 작업은 일반적으로 범위가 넓으며, IAM 조건을 사용하여 범위를 줄일 수 있습니다. 예를 들어, PollForActivityTask 작업이 액세스하도록 허용된 작업 목록을 제한하려면 `Condition`을 포함하고 `swf:taskList.name` 키를 사용해 허용 가능한 목록을 지정합니다.

다음 엔터티에 대한 제약을 표현할 수 있습니다.

- 워크플로 유형. 이름 및 버전에는 별도의 키가 있습니다.
- 활동 유형. 이름 및 버전에는 별도의 키가 있습니다.
- 작업 목록
- Tags. 일부 작업의 경우 태그를 여러 개 지정할 수 있습니다. 이러한 경우 각 태그에는 별도의 키가 있습니다.

> **Note**
>
> Amazon SWF의 경우 값은 모든 문자열이므로, 문자열 연산자(예: 지정된 문자열로 파라미터를 제한하는 `StringEquals`)를 사용하여 파라미터를 제한합니다. 그러나 정규 문자열 비교 연산자(예: `StringEquals`)에는 모두 파라미터를 포함하라는 요청이 필요합니다. 파라미터를 명시적으로 포함하지 않고 기본값(예: 유형 등록 중 제공한 기본 작업 목록)이 없으면 액세스가 거부됩니다.
> 조건은 선택 사항으로 취급하는 것이 일반적으로 유용합니다. 그래야 연결된 파라미터를 반드시 포함하지 않고 작업을 호출할 수 있습니다. 예를 들어, 결정자가 RespondDecisionTaskCompleted 결정 세트를 지정하도록 허용하지만 특정 호출에 대해서는 결정 중 하나만 지정하도록 허용하려고 할 수 있습니다. 이러한 경우에는 `StringEqualsIfExists` 연산자를 사용해 적절한 파라미터를 제한합니다. 그러면 파라미터가 조건을 충족하는 경우 액세스를 허용하지만 파라미터가 없는 경우에는 액세스를 거부하지 않습니다.

제한 가능한 파라미터와 연결된 키의 전체 목록은 API 요약 단원을 참조하십시오.

다음 단원에서는 Amazon SWF 정책을 구성하는 방법의 예를 제공합니다. 자세한 내용은 문자열 조건을 참조하십시오.

Amazon SWF에 대한 정책 작성

워크플로는 활동, 결정자 등 여러 액터로 구성됩니다. 적절한 IAM 정책을 연결하여 각 액터에 대한 액세스를 제어할 수 있습니다.

다음 작업을 수행하면 모든 리전에서 액터에게 전체 계정 액세스 권한이 부여됩니다.

- 작업: `swf:*`
- 리소스: `arn:aws:swf:*:123456789012:/domain/*`

와일드카드를 사용해 여러 리소스, 작업 또는 리전을 값 하나로 나타낼 수 있습니다.

- `Resource` 값의 첫 번째 와일드카드(*)는 리소스 권한이 모든 리전에 적용됨을 나타냅니다.

  권한을 단일 리전으로 제한하려면 와일드카드를 적절한 리전 문자열(예: us-east-1)로 바꿉니다.
- `Resource` 값의 두 번째 와일드카드(*)는 액터가 지정된 리전에 있는 계정의 모든 도메인에 액세스하도록 허용합니다.
- `Action` 값의 와일드카드(*)는 액터가 모든 Amazon SWF 작업을 직접적으로 호출하도록 허용합니다.

와일드카드를 사용하는 방법에 대한 자세한 내용은 요소 설명을 참조하십시오.

도메인 권한

부서의 워크플로를 특정 도메인으로 제한하려면 액터가 특정 부서에 대해서만 모든 작업을 호출하도록 허용하는 권한을 부여할 수 있습니다.

둘 이상의 도메인에 대한 액터 액세스를 실행하려면 각 도메인에 대한 권한을 문 목록으로 표현합니다.

- 작업: `swf:*`
- 리소스: `arn:aws:swf:*:123456789012:/domain/department1`
- 리소스: `arn:aws:swf:*:123456789012:/domain/department2`

액터가 `department1` 및 `department2` 도메인에서 Amazon SWF 작업을 사용하도록 허용할 수 있습니다. 또한 경우에 따라 와일드카드를 사용해 여러 도메인을 나타낼 수도 있습니다.

API 권한 및 제약

Action 요소에서 작업을 지정하여 액터가 사용할 수 있는 작업을 제어합니다.

다음 작업에서는 액터가 워크플로를 시작하기 StartWorkflowExecution 위해서만를 호출할 수 있습니다. 다른 작업은 사용할 수 없습니다.

- 작업: swf:StartWorkflowExecution

조건

요소를 사용하여 작업의 허용 가능한 파라미터 값을 선택적으로 제한할 수 Condition 있습니다.

액터가 시작할 수 있는 워크플로를 제한하려면 다음과 같이 하나 이상의 StartWorkflowExecution 파라미터 값을 제한합니다.

```
"Condition" : {
   "StringEquals" : {
      "swf:workflowType.name" : "workflow1",
      "swf:workflowType.version" : "version2"
    }
}
```

이전 제약 조건이 있는 액터는 version2 만 실행할 수 workflow1 있으며 두 파라미터가 모두 요청에 포함되어야 합니다.

다음과 같이 StringEqualsIfExists 연산자를 사용해 파라미터를 요청에 포함하지 않고 제한할 수 있습니다.

```
"Condition" : {
   "StringEqualsIfExists" : { "swf:taskList.name" : "task_list_name" }
}
```

이전 정책이 있는 액터는 워크플로 실행을 시작할 때 선택적으로 작업 목록을 지정할 수 있습니다.

일부 작업의 태그 목록을 제한할 수 있습니다. 각 태그에는 별도의 키가 있으므로 swf:tagList.member.0를 사용하여 목록의 첫 번째 태그를 제한swf:tagList.member.1하고 목록의 두 번째 태그를 최대 5개까지 제한합니다.

태그 목록을 제한하는 방법에 주의해야 합니다. 예를 들어 다음 조건은 권장되지 않습니다.

다음 조건은 some_ok_tag 또는 중 하나를 선택적으로 지정할 수 있으므로 권장되지 않습니다another_ok_tag. 그러나 조건은 태그 목록의 첫 번째 요소만 제한합니다. 이 목록에는 조건이 swf:tagList.member.1, swf:tagList.member.2등에 조건을 적용하지 않기 때문에 모두 허용되는 임의의 값을 가진 추가 요소가 있을 수 있습니다.

```
// Example to illustrate an insecure Condition
"Condition" : {
   "StringEqualsIfExists" : {
      "swf:tagList.member.0" : "some_ok_tag", "another_ok_tag"
   }
}
```

이전 문제를 해결하는 한 가지 방법은 태그 목록 사용을 허용하지 않는 것입니다.

다음 정책은 목록에 요소가 하나만 있도록 해 some_ok_tag 또는 another_ok_tag만 허용하도록 합니다.

```
"Condition" : {
   "StringEqualsIfExists" : {
      "swf:tagList.member.0" : "some_ok_tag", "another_ok_tag"
    },
    "Null" : { "swf:tagList.member.1" : "true" }
}
```

의사 API 권한 및 제약

에서 사용할 수 있는 결정을 제한하려면 먼저 액터가를 호출하도록 허용RespondDecisionTaskCompleted해야 합니다RespondDecisionTaskCompleted. 그런 다음 다음과 같이 일반 API와 동일한 구문을 사용하여 적절한 의사 API 멤버에 대한 권한을 표현합니다.

- 문 1

  리소스: arn:aws:swf:*:123456789012:/domain/*

  작업: swf:RespondDecisionTaskCompleted
- 문 2

  리소스: *

  작업: swf:ScheduleActivityTask

조건: "StringEquals" : { "swf:activityType.name" : "SomeActivityType" }

첫 번째는 액터가를 호출하도록 Statement 허용합니다RespondDecisionTaskCompleted. 두 번째 문을 통해 액터는 ScheduleActivityTask 결정을 사용하여 Amazon SWF에 활동 작업을 예약하도록 지시할 수 있습니다. 모든 결정을 허용하려면 "swf:ScheduleActivityTask"를 "swf:*"로 바꿉니다.

조건 연산자를 사용해 정규 API처럼 파라미터를 제한할 수 있습니다. 이전 예제의 StringEquals 연산자는 RespondDecisionTaskCompleted가 SomeActivityType 활동에 대한 활동 작업을 예약할 수 있도록 Condition 허용하며 해당 작업을 예약해야 합니다. 파라미터 값을 사용하도록 RespondDecisionTaskCompleted를 허용하지만 반드시 사용할 필요는 없도록 하려면 대신 StringEqualsIfExists 연산자를 사용할 수 있습니다.

AWS 관리형 정책: SimpleWorkflowFullAccess

SimpleWorkflowFullAccess 정책을 IAM ID에 연결할 수 있습니다.

이 정책은 Amazon SWF 구성 서비스에 대한 전체 액세스 권한을 제공합니다.

IAM 정책에 대한 서비스 모델 제한 사항

IAM 정책을 생성할 때 서비스 모델 제약을 고려해야 합니다. 구문상 유효한 IAM 정책을 생성했으나 해당 정책이 잘못된 Amazon SWF 요청을 나타낼 수 있습니다. 이 경우, 액세스 제어 관점에서 허용되는 요청에 실패할 수 있는데 이는 잘못된 요청이기 때문입니다.

예를 들어 Amazon SWF 서비스 모델은 typeFilter 및 tagFilter 파라미터를 동일한 ListOpenWorkflowExecutions 요청에 사용하도록 허용하지 않습니다. 다음 조건은 서비스가 잘못된 요청ValidationException으로 발생시켜 거부할 호출을 허용합니다.

```
"Condition" : {
   "StringEquals" : {
      "swf:typeFilter.name" : "workflow_name",
      "swf:typeFilter.version" : "workflow_version",
      "swf:tagFilter.tag" : "some_tag"
    }
}
```

API 요약

이 단원에서는 IAM 정책을 사용해 액터가 각 API 및 의사 API를 사용하여 Amazon SWF 리소스에 액세스하는 방식을 제어할 수 있는 방법을 간략하게 설명합니다.

- RegisterDomain 및 ListDomains를 제외한 모든 작업에서 도메인 리소스에 대한 권한을 표시해 계정의 도메인 일부 또는 전부에 대한 액세스를 허용 또는 거부할 수 있습니다.
- 정규 API 멤버에 대한 권한을 허용하거나 거부할 수 있고, RespondDecisionTaskCompleted 호출 권한을 부여하여 의사 API 멤버에 대한 권한을 허용 또는 거부할 수 있습니다.
- 조건을 사용해 일부 파라미터의 허용 가능한 값을 제한할 수 있습니다.

다음 단원에서는 정규 및 의사 API의 각 멤버에 대해 제한할 수 있는 파라미터를 나열하고, 연결된 키를 제공하고, 도메인 액세스를 제어할 수 있는 방법에 대한 제한 사항을 설명합니다.

일반 API

이 단원에서는 정규 API 멤버를 나열하고, 제한할 수 있는 파라미터와 연결된 키에 대해 간략하게 설명합니다. 또한 도메인 액세스를 제어할 수 있는 방법에 대한 제한 사항을 설명합니다.

CountClosedWorkflowExecutions

- tagFilter.tag – 문자열 제약. 키는 swf:tagFilter.tag입니다.
- typeFilter.name – 문자열 제약. 키는 swf:typeFilter.name입니다.
- typeFilter.version – 문자열 제약. 키는 swf:typeFilter.version입니다.

> **Note**
>
> CountClosedWorkflowExecutions에서는 typeFilter와 tagFilter를 함께 사용할 수 없습니다.

CountOpenWorkflowExecutions

- tagFilter.tag – 문자열 제약. 키는 swf:tagFilter.tag입니다.
- typeFilter.name – 문자열 제약. 키는 swf:typeFilter.name입니다.
- typeFilter.version – 문자열 제약. 키는 swf:typeFilter.version입니다.

> **Note**
>
> CountOpenWorkflowExecutions에서는 typeFilter와 tagFilter를 함께 사용할 수 없습니다.

CountPendingActivityTasks

- taskList.name – 문자열 제약. 키는 swf:taskList.name입니다.

CountPendingDecisionTasks

- taskList.name – 문자열 제약. 키는 swf:taskList.name입니다.

DeleteActivityType

- activityType.name – 문자열 제약. 키는 swf:activityType.name입니다.
- activityType.version – 문자열 제약. 키는 swf:activityType.version입니다.

DeprecateActivityType

- activityType.name – 문자열 제약. 키는 swf:activityType.name입니다.
- activityType.version – 문자열 제약. 키는 swf:activityType.version입니다.

DeprecateDomain

- 이 작업의 파라미터는 제한할 수 없습니다.

DeleteWorkflowType

- workflowType.name – 문자열 제약. 키는 swf:workflowType.name입니다.
- workflowType.version – 문자열 제약. 키는 swf:workflowType.version입니다.

DeprecateWorkflowType

- workflowType.name – 문자열 제약. 키는 swf:workflowType.name입니다.
- workflowType.version – 문자열 제약. 키는 swf:workflowType.version입니다.

DescribeActivityType

- activityType.name – 문자열 제약. 키는 swf:activityType.name입니다.
- activityType.version – 문자열 제약. 키는 swf:activityType.version입니다.

DescribeDomain

- 이 작업의 파라미터는 제한할 수 없습니다.

DescribeWorkflowExecution

- 이 작업의 파라미터는 제한할 수 없습니다.

DescribeWorkflowType

- workflowType.name – 문자열 제약. 키는 swf:workflowType.name입니다.
- workflowType.version – 문자열 제약. 키는 swf:workflowType.version입니다.

GetWorkflowExecutionHistory

- 이 작업의 파라미터는 제한할 수 없습니다.

ListActivityTypes

- 이 작업의 파라미터는 제한할 수 없습니다.

ListClosedWorkflowExecutions

- tagFilter.tag – 문자열 제약. 키는 swf:tagFilter.tag입니다.
- typeFilter.name – 문자열 제약. 키는 swf:typeFilter.name입니다.
- typeFilter.version – 문자열 제약. 키는 swf:typeFilter.version입니다.

> **Note**
>
> ListClosedWorkflowExecutions에서는 typeFilter와 tagFilter를 함께 사용할 수 없습니다.

ListDomains

- 이 작업의 파라미터는 제한할 수 없습니다.

ListOpenWorkflowExecutions

- tagFilter.tag – 문자열 제약. 키는 swf:tagFilter.tag입니다.
- typeFilter.name – 문자열 제약. 키는 swf:typeFilter.name입니다.
- typeFilter.version – 문자열 제약. 키는 swf:typeFilter.version입니다.

> **Note**
>
> ListOpenWorkflowExecutions에서는 typeFilter와 tagFilter를 함께 사용할 수 없습니다.

ListWorkflowTypes

- 이 작업의 파라미터는 제한할 수 없습니다.

PollForActivityTask

- taskList.name – 문자열 제약. 키는 swf:taskList.name입니다.

PollForDecisionTask

- taskList.name – 문자열 제약. 키는 swf:taskList.name입니다.

RecordActivityTaskHeartbeat

- 이 작업의 파라미터는 제한할 수 없습니다.

RegisterActivityType

- defaultTaskList.name – 문자열 제약. 키는 swf:defaultTaskList.name입니다.
- name – 문자열 제약. 키는 swf:name입니다.
- version – 문자열 제약. 키는 swf:version입니다.

RegisterDomain

- name – 등록 중인 도메인의 이름으로, 이 작업의 리소스로 사용할 수 있습니다.

RegisterWorkflowType

- defaultTaskList.name – 문자열 제약. 키는 swf:defaultTaskList.name입니다.
- name – 문자열 제약. 키는 swf:name입니다.
- version – 문자열 제약. 키는 swf:version입니다.

RequestCancelWorkflowExecution

- 이 작업의 파라미터는 제한할 수 없습니다.

RespondActivityTaskCanceled

- 이 작업의 파라미터는 제한할 수 없습니다.

RespondActivityTaskCompleted

- 이 작업의 파라미터는 제한할 수 없습니다.

RespondActivityTaskFailed

- 이 작업의 파라미터는 제한할 수 없습니다.

RespondDecisionTaskCompleted

- decisions.member.N – 의사 API 권한을 통해 간접적으로 제한됩니다. 자세한 내용은 의사 API을 참조하세요.

SignalWorkflowExecution

- 이 작업의 파라미터는 제한할 수 없습니다.

StartWorkflowExecution

- tagList.member.0 – 문자열 제약. 키는 swf:tagList.member.0입니다.

- `tagList.member.1` – 문자열 제약. 키는 `swf:tagList.member.1`입니다.
- `tagList.member.2` – 문자열 제약. 키는 `swf:tagList.member.2`입니다.
- `tagList.member.3` – 문자열 제약. 키는 `swf:tagList.member.3`입니다.
- `tagList.member.4` – 문자열 제약. 키는 `swf:tagList.member.4`입니다.
- `taskList.name` – 문자열 제약. 키는 `swf:taskList.name`입니다.
- `workflowType.name` – 문자열 제약. 키는 `swf:workflowType.name`입니다.
- `workflowType.version` – 문자열 제약. 키는 `swf:workflowType.version`입니다.

> **Note**
>
> 태그는 다섯 개까지 제한할 수 있습니다.

`TerminateWorkflowExecution`

- 이 작업의 파라미터는 제한할 수 없습니다.

의사 API

이 단원에는 `RespondDecisionTaskCompleted`에 포함된 결정을 나타내는 의사 API의 멤버 목록이 나옵니다. `RespondDecisionTaskCompleted`를 사용하도록 권한을 부여한 경우 정책은 정규 API와 동일한 방식으로 의사 API의 멤버에 대한 권한을 표현할 수 있습니다. 하나 이상의 파라미터에 대해 조건을 설정해 의사 API의 일부 멤버를 추가로 제한할 수 있습니다. 이 단원에서는 의사 API 멤버 목록을 나열하고, 제한할 수 있는 파라미터와 연결된 키에 대해 간략하게 설명합니다.

> **Note**
>
> `aws:SourceIP`, `aws:UserAgent` 및 `aws:SecureTransport` 키는 의사 API에 사용할 수 없습니다. 의도한 보안 정책에서 이러한 키가 의사 API에 대한 액세스를 제어하도록 요구하면 `RespondDecisionTaskCompleted` 작업과 함께 해당 키를 사용할 수 있습니다.

`CancelTimer`

- 이 작업의 파라미터는 제한할 수 없습니다.

`CancelWorkflowExecution`

- 이 작업의 파라미터는 제한할 수 없습니다.

`CompleteWorkflowExecution`

- 이 작업의 파라미터는 제한할 수 없습니다.

`ContinueAsNewWorkflowExecution`

- `tagList.member.0` – 문자열 제약. 키는 `swf:tagList.member.0`입니다.
- `tagList.member.1` – 문자열 제약. 키는 `swf:tagList.member.1`입니다.
- `tagList.member.2` – 문자열 제약. 키는 `swf:tagList.member.2`입니다.
- `tagList.member.3` – 문자열 제약. 키는 `swf:tagList.member.3`입니다.
- `tagList.member.4` – 문자열 제약. 키는 `swf:tagList.member.4`입니다.
- `taskList.name` – 문자열 제약. 키는 `swf:taskList.name`입니다.
- `workflowTypeVersion` – 문자열 제약. 키는 `swf:workflowTypeVersion`입니다.

> **Note**
>
> 태그는 다섯 개까지 제한할 수 있습니다.

`FailWorkflowExecution`

- 이 작업의 파라미터는 제한할 수 없습니다.

`RecordMarker`

- 이 작업의 파라미터는 제한할 수 없습니다.

`RequestCancelActivityTask`

- 이 작업의 파라미터는 제한할 수 없습니다.

`RequestCancelExternalWorkflowExecution`

- 이 작업의 파라미터는 제한할 수 없습니다.

`ScheduleActivityTask`

- `activityType.name` – 문자열 제약. 키는 `swf:activityType.name`입니다.
- `activityType.version` – 문자열 제약. 키는 `swf:activityType.version`입니다.
- `taskList.name` – 문자열 제약. 키는 `swf:taskList.name`입니다.

`SignalExternalWorkflowExecution`

- 이 작업의 파라미터는 제한할 수 없습니다.

`StartChildWorkflowExecution`

- `tagList.member.0` – 문자열 제약. 키는 `swf:tagList.member.0`입니다.
- `tagList.member.1` – 문자열 제약. 키는 `swf:tagList.member.1`입니다.
- `tagList.member.2` – 문자열 제약. 키는 `swf:tagList.member.2`입니다.
- `tagList.member.3` – 문자열 제약. 키는 `swf:tagList.member.3`입니다.
- `tagList.member.4` – 문자열 제약. 키는 `swf:tagList.member.4`입니다.
- `taskList.name` – 문자열 제약. 키는 `swf:taskList.name`입니다.
- `workflowType.name` – 문자열 제약. 키는 `swf:workflowType.name`입니다.
- `workflowType.version` – 문자열 제약. 키는 `swf:workflowType.version`입니다.

> **Note**
>
> 태그는 다섯 개까지 제한할 수 있습니다.

`StartTimer`

- 이 작업의 파라미터는 제한할 수 없습니다.

태그 기반 정책

Amazon SWF는 태그를 기반으로 하는 정책을 지원합니다. 예를 들어 키가 있는 태그environment와 다음 조건이 있는 값이 포함된 Amazon SWF 도메인을 제한production할 수 있습니다.

```
"Condition": {
    "StringEquals": {"aws:ResourceTag/environment": "production"}
}
```

태그 지정에 대한 자세한 내용은 다음을 참조하세요.

- Amazon SWF의 태그
- IAM 태그를 사용한 액세스 제어

Amazon SWF용 Amazon VPC 엔드포인트

> **Note**
>
> AWS PrivateLink 지원은 현재 AWS 일급 보안 암호 - 동부, AWS 비밀 리전 및 중국 리전에서만 사용할 수 있습니다.

Amazon Virtual Private Cloud(VPC)를 사용하여 AWS 리소스를 호스팅하는 경우 Amazon VPC와 Amazon Simple Workflow Service 워크플로 간에 연결을 설정할 수 있습니다. 퍼블릭 인터넷을 사용하지 않고도 Amazon SWF 워크플로에서 이 연결을 사용할 수 있습니다.

Amazon VPC를 사용하면 사용자 지정 가상 네트워크에서 AWS 리소스를 시작할 수 있습니다. VPC를 사용하여 IP 주소 범위, 서브넷, 라우팅 테이블, 네트워크 게이트웨이 등의 네트워크 설정을 제어할 수 있습니다. VPC에 대한 자세한 내용은 Amazon VPC 사용 설명서를 참조하세요.

Amazon VPC를 Amazon SWF에 연결하려면 먼저 VPC를 다른 AWS 서비스에 연결할 수 있는 인터페이스 VPC 엔드포인트를 정의해야 합니다. 이 엔드포인트를 이용하면 인터넷 게이트웨이나 NAT(네트워크 주소 변환) 인스턴스 또는 VPN 연결 없이도 안정적이고 확장 가능하게 연결됩니다. 자세한 내용은 Amazon VPC 사용 설명서의 인터페이스 VPC 엔드포인트(AWS PrivateLink)를 참조하십시오.

엔드포인트 만들기

AWS Management Console, AWS Command Line Interface (AWS CLI), AWS SDK, Amazon SWF API 또는를 사용하여 VPC에서 Amazon SWF 엔드포인트를 생성할 수 있습니다 CloudFormation.

Amazon VPC 콘솔 또는 AWS CLI를 사용한 엔드포인트 생성 및 구성에 대한 자세한 내용은 Amazon VPC 사용 설명서의 인터페이스 엔드포인트 생성을 참조하십시오.

> **Note**
>
> 엔드포인트를 생성할 때 VPC를 연결할 서비스로 Amazon SWF를 지정해야 합니다. Amazon VPC 콘솔에서 서비스 이름은 AWS 리전에 따라 다릅니다. 예를 들어, AWS 일급 보안 암호 - 동부 리전에서 Amazon SWF의 서비스 이름은 com.amazonaws.us-iso-east-1.swf입니다.

를 사용하여 엔드포인트를 생성하고 구성하는 방법에 대한 자세한 내용은 CloudFormation 사용 설명서의 AWS::EC2::VPCEndpoint 리소스를 CloudFormation참조하세요.

Amazon VPC 엔드포인트 정책

Amazon SWF에 대한 연결 액세스를 제어하려면 Amazon VPC 엔드포인트를 생성하는 동안 AWS Identity and Access Management (IAM) 엔드포인트 정책을 연결할 수 있습니다. 엔드포인트 정책 여러 개를 연결하여 복잡한 IAM 규칙을 만들 수 있습니다. 자세한 내용은 다음을 참조하십시오.

- Amazon SWF용 Amazon Virtual Private Cloud 엔드포인트 정책
- VPC 엔드포인트로 서비스에 대한 액세스 제어

Amazon SWF용 Amazon Virtual Private Cloud 엔드포인트 정책

Amazon SWF에 대한 Amazon VPC 엔드포인트 정책을 생성하여 다음을 지정할 수 있습니다.

- 작업을 수행할 수 있는 위탁자.
- 수행할 수 있는 작업.
- 작업을 수행할 수 있는 리소스.

다음 예시에서는 정책에 특정 IAM 역할을 추가합니다.

```
"Principal": {
   "AWS": "arn:aws:iam::123456789012:role/MyRole"
}
```

- 엔드포인트 정책 생성에 대한 자세한 내용은 VPC 엔드포인트로 서비스에 대한 액세스 제어를 참조하십시오.

- IAM을 사용하여 AWS 및 Amazon SWF 리소스에 대한 액세스를 제어하는 방법에 대한 자세한 내용은 섹션을 참조하세요Amazon Simple Workflow Service의 Identity and Access Management.

Amazon Simple Workflow Service ID 및 액세스 문제 해결

다음 정보를 사용하여 Amazon SWF 및 IAM으로 작업할 때 발생할 수 있는 일반적인 문제를 진단하고 수정할 수 있습니다.

주제

- Amazon SWF에서 작업을 수행할 권한이 없음
- iam:PassRole을 수행하도록 인증되지 않음
- 내 외부의 사람이 내 Amazon SWF 리소스 AWS 계정 에 액세스하도록 허용하고 싶습니다.

Amazon SWF에서 작업을 수행할 권한이 없음

작업을 수행할 수 있는 권한이 없다는 오류가 수신되면 작업을 수행할 수 있도록 정책을 업데이트해야 합니다.

다음 예제 오류는 `mateojackson` 사용자가 콘솔을 사용하여 가상 *`my-example-widget`* 리소스에 대한 세부 정보를 보려고 하지만 가상 `swf:`*`GetWidget`* 권한이 없을 때 발생합니다.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 swf:GetWidget on resource: my-example-widget
```

이 경우 Mateo의 정책은 `swf:`*`GetWidget`* 작업을 사용하여 *`my-example-widget`* 리소스에 액세스하도록 허용하도록 업데이트해야 합니다.

도움이 필요한 경우 AWS 관리자에게 문의하세요. 관리자는 로그인 자격 증명을 제공한 사람입니다.

iam:PassRole을 수행하도록 인증되지 않음

`iam:PassRole` 작업을 수행할 수 있는 권한이 없다는 오류가 수신되면 Amazon SWF에 역할을 전달할 수 있도록 정책을 업데이트해야 합니다.

일부 AWS 서비스 에서는 새 서비스 역할 또는 서비스 연결 역할을 생성하는 대신 기존 역할을 해당 서비스에 전달할 수 있습니다. 이렇게 하려면 역할을 서비스에 전달할 권한이 있어야 합니다.

다음 예제 오류는 marymajor라는 IAM 사용자가 콘솔을 사용하여 Amazon SWF에서 작업을 수행하려고 하는 경우에 발생합니다. 하지만 작업을 수행하려면 서비스 역할이 부여한 권한이 서비스에 있어야 합니다. Mary는 서비스에 역할을 전달할 권한이 없습니다.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

이 경우, Mary가 iam:PassRole작업을 수행할 수 있도록 Mary의 정책을 업데이트해야 합니다.

도움이 필요한 경우 AWS 관리자에게 문의하세요. 관리자는 로그인 자격 증명을 제공한 사람입니다.

내 외부의 사람이 내 Amazon SWF 리소스 AWS 계정 에 액세스하도록 허용하고 싶습니다.

다른 계정의 사용자 또는 조직 외부의 사람이 리소스에 액세스할 때 사용할 수 있는 역할을 생성할 수 있습니다. 역할을 수임할 신뢰할 수 있는 사람을 지정할 수 있습니다. 리소스 기반 정책 또는 액세스 제어 목록(ACL)을 지원하는 서비스의 경우, 이러한 정책을 사용하여 다른 사람에게 리소스에 대한 액세스 권한을 부여할 수 있습니다.

자세한 내용은 다음을 참조하세요.

- Amazon SWF에서 이러한 기능을 지원하는지 여부를 알아보려면 Amazon Simple Workflow Service가 IAM으로 작동하는 방식 단원을 참조하세요.
- 소유 AWS 계정 한의 리소스에 대한 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용 설명서의 소유한 다른의 IAM 사용자에게 액세스 권한 제공을 참조 AWS 계정 하세요.
- 타사에 리소스에 대한 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용 설명서의 타사가 AWS 계정 소유한에 대한 액세스 권한 제공을 AWS 계정참조하세요.
- ID 페더레이션을 통해 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용 설명서의 외부에서 인증된 사용자에게 액세스 권한 제공(ID 페더레이션)을 참조하세요.
- 크로스 계정 액세스에 대한 역할과 리소스 기반 정책 사용의 차이점을 알아보려면 IAM 사용 설명서의 IAM의 크로스 계정 리소스 액세스를 참조하세요.

로깅 및 모니터링

이 단원에서는 Amazon SWF 로깅 및 모니터링에 대해 설명합니다.

주제

- CloudWatch에 대한 Amazon SWF 지표

- 를 사용하여 CloudWatch에 대한 Amazon SWF 지표 보기 AWS Management Console
- 를 사용하여 API 호출 기록 AWS CloudTrail
- Amazon SWF용 EventBridge 실행 상태 변경
- Amazon Simple Workflow Service AWS 사용자 알림 에서 사용

CloudWatch에 대한 Amazon SWF 지표

Amazon SWF는 이제 CloudWatch에 대한 지표를 제공합니다. 이것으로 워크플로 및 활동을 추적하고, 선택한 임계값에 대한 경보를 설정할 수 있습니다. 를 사용하여 지표를 볼 수 있습니다 AWS Management Console. 자세한 내용은 를 사용하여 CloudWatch에 대한 Amazon SWF 지표 보기 AWS Management Console 단원을 참조하십시오.

주제

- Amazon SWF 지표에 대한 단위 보고
- API 및 결정 이벤트 측정치
- Amazon SWF 지표
- Amazon SWF 비 ASCII 리소스 이름 및 CloudWatch 차원

Amazon SWF 지표에 대한 단위 보고

시간 간격을 보고하는 지표

일부 CloudWatch에 대한 Amazon SWF 지표는 시간 간격이며, 항상 밀리초로 측정됩니다. CloudWatch 단위는 `Time`으로 보고됩니다. 이들 지표는 보통 워크플로 실행의 단계에 해당되며, 사용자는 이에 대해 워크플로 및 활동 제한 시간을 설정하고 비슷한 이름을 명명할 수 있습니다.

예를 들어 `DecisionTaskStartToCloseTime` 지표는 실행 시작 이후에 의사 결정 작업이 완료될 때까지 소요된 시간을 측정하는데, 같은 시간에 대해 `DecisionTaskStartToCloseTimeout` 값을 설정할 수 있습니다.

이러한 각 워크플로 단계의 다이어그램의 경우 워크플로 및 활동 수명 주기에서 해당 단계가 발생한 시점을 알아보려면 Amazon SWF 제한 시간 유형 단원을 참조하십시오.

개수를 보고하는 지표

CloudWatch에 대한 Amazon SWF 지표 중 일부는 결과를 개수로 보고합니다. 예를 들어 `WorkflowsCanceled`는 결과를 1 또는 0으로 기록하여 워크플로의 취소 여부를 표시합니다. 0 값은

해당 지표가 보고되지 않았음을 나타내는 것이 아니라, 단순히 해당 지표에 설명된 조건이 발생하지 않았음을 나타냅니다.

CloudWatch에 대한 Amazon SWF 지표 중 일부는 CloudWatch `Count`를 초당 개수로 보고합니다. 예를 들어 CloudWatch에서 `Count`로 보고되는 `ProvisionedRefillRate`은 초당 요청 `Count` 비율을 나타냅니다.

수 지표에서 최소값과 최대값은 항상 0 아니면 1이 되지만, 평균은 0과 1 사이의 값이 됩니다.

API 및 결정 이벤트 측정치

CloudWatch에서 API 및 결정 이벤트를 둘 다 모니터링하여 사용량 및 용량을 파악할 수 있습니다. Amazon SWF의 기본 워크플로 개념 섹션의 결정자 및 Amazon Simple Workflow Service API 참조의 결정 주제를 참조하십시오.

또한 Amazon SWF 제한에 가까워지는 경우에도 이러한 제한을 모니터링하여 경보를 보낼 수 있습니다. 이러한 제한과 각 기본 설정에 대한 설명은 Amazon SWF 제한 할당량 단원을 참조하십시오. 이러한 제한은 잘못된 워크플로가 시스템 리소스를 과도하게 사용하지 않도록 방지하기 위해 마련되었습니다. 제한에 대한 증가를 요청하려면 ??? 단원을 참조하십시오.

API 또는 결정 이벤트 용량의 약 60%에서 CloudWatch 경보를 구성하는 것이 가장 좋습니다. 그러면 워크플로를 조정하거나 Amazon SWF 제한이 활성화되기 전에 서비스 제한 증가를 요청할 수 있습니다. 통화의 간헐성에 따라 서비스 제한에 근접할 때 알리도록 여러 경보를 구성할 수 있습니다.

- 트래픽이 크게 증가하는 경우 `ProvisionedBucketSize` 제한의 60%에서 경보를 설정합니다.
- 통화가 상대적으로 일정한 비율을 유지하는 경우 관련 API 및 결정 이벤트에 대한 `ProvisionedRefillRate` 제한의 60%에서 경보를 설정합니다.

Amazon SWF 지표

Amazon SWF에 사용할 수 있는 지표는 다음과 같습니다.

지표	설명
`DecisionTaskScheduleToStartTime`	의사결정 작업의 예약 시간과 작업자가 그 작업을 선택하여 시작한 시간 사이의 간격(밀리초) CloudWatch 단위: `Time`

지표	설명
	차원: Domain, WorkflowTypeName, WorkflowTypeVersion 유효한 통계: Average, Minimum, Maximum
DecisionTaskStartToCloseTime	결정 작업 시작 시간과 닫힌 시간 사이의 시간 간격(밀리초) CloudWatch 단위: Time 차원: Domain, WorkflowTypeName, WorkflowTypeVersion 유효한 통계: Average, Minimum, Maximum
DecisionTasksCompleted	완료된 의사결정 작업 수 CloudWatch 단위: Count 차원: Domain, WorkflowTypeName, WorkflowTypeVersion 유효한 통계: Sum
PendingTasks	특정 작업 목록에서 1분 간격으로 확인된 보류 중인 작업의 수. CloudWatch 단위: Count 차원: Domain, TaskListName 유효한 통계: Sum
StartedDecisionTasksTimedOutOnClose	시작했지만 종료 시간이 초과된 의사결정 작업 수 CloudWatch 단위: Count 차원: Domain, WorkflowTypeName, WorkflowTypeVersion 유효한 통계: Sum

지표	설명
WorkflowStartToCloseTime	워크플로가 시작된 시간과 닫힌 시간 사이의 시간 간격(밀리초) CloudWatch 단위: Time 차원: Domain, WorkflowTypeName, WorkflowTypeVersion 유효한 통계: Average, Minimum, Maximum
WorkflowsCanceled	취소된 워크플로 수 CloudWatch 단위: Count 차원: Domain, WorkflowTypeName, WorkflowTypeVersion 유효한 통계: Sum
WorkflowsCompleted	완료된 워크플로 수 CloudWatch 단위: Count 차원: Domain, WorkflowTypeName, WorkflowTypeVersion 유효한 통계: Sum
WorkflowsContinuedAsNew	신규로 계속된 워크플로 수 CloudWatch 단위: Count 차원: Domain, WorkflowTypeName, WorkflowTypeVersion 유효한 통계: Sum

지표	설명
WorkflowsFailed	실패한 워크플로 수 CloudWatch 단위: Count 차원: Domain, WorkflowTypeName, WorkflowTypeVersion 유효한 통계: Sum
WorkflowsTerminated	종료된 워크플로 수 CloudWatch 단위: Count 차원: Cause, Domain, WorkflowTypeName, WorkflowTypeVersion 유효한 통계: Sum
WorkflowsTimedOut	어떤 이유든지 시간이 초과된 워크플로 수 CloudWatch 단위: Count 차원: Domain, WorkflowTypeName, WorkflowTypeVersion 유효한 통계: Sum
ActivityTaskScheduleToCloseTime	활동 예약 시간과 닫힌 시간 사이의 시간 간격(밀리초) CloudWatch 단위: Time 차원: Domain, ActivityTypeName, ActivityTypeVersion 유효한 통계: Average, Minimum, Maximum

지표	설명
ActivityTaskScheduleToStartTime	활동 작업 예약 시간과 시작 시간 사이의 시간 간격(밀리초) CloudWatch 단위: Time 차원: Domain, ActivityTypeName, ActivityTypeVersion 유효한 통계: Average, Minimum, Maximum
ActivityTaskStartToCloseTime	활동 작업 시작 시간과 닫힌 시간 사이의 시간 간격(밀리초) CloudWatch 단위: Time 차원: Domain, ActivityTypeName, ActivityTypeVersion 유효한 통계: Average, Minimum, Maximum
ActivityTasksCanceled	취소된 활동 작업 수 CloudWatch 단위: Count 차원: Domain, ActivityTypeName, ActivityTypeVersion 유효한 통계: Sum
ActivityTasksCompleted	완료된 활동 작업 수 CloudWatch 단위: Count 차원: Domain, ActivityTypeName, ActivityTypeVersion 유효한 통계: Sum

지표	설명
ActivityTasksFailed	실패한 활동 작업 수 CloudWatch 단위: Count 차원: Domain, ActivityTypeName, ActivityTypeVersion 유효한 통계: Sum
ScheduledActivityTasksTimedOutOnClose	예약되었지만 종료 시간이 초과된 활동 작업 수 CloudWatch 단위: Count 차원: Domain, ActivityTypeName, ActivityTypeVersion 유효한 통계: Sum
ScheduledActivityTasksTimedOutOnStart	예약되었지만 시작 시간이 초과된 활동 작업 수 CloudWatch 단위: Count 차원: Domain, ActivityTypeName, ActivityTypeVersion 유효한 통계: Sum
StartedActivityTasksTimedOutOnClose	시작하였지만 종료 시간이 초과된 활동 작업 수 CloudWatch 단위: Count 차원: Domain, ActivityTypeName, ActivityTypeVersion 유효한 통계: Sum

지표	설명
StartedActivityTasksTimedOutOnHeartbeat	시작하였지만 하트비트 제한 시간으로 인해 시간이 초과된 활동 작업 수 CloudWatch 단위: Count 차원: Domain, ActivityTypeName, ActivityTypeVersion 유효한 통계: Sum
ThrottledEvents	조절된 요청 수 CloudWatch 단위: Count 차원: APIName, DecisionName, ThrottlingScope 유효한 통계: Sum
ProvisionedBucketSize	초당 사용 가능한 요청 수 차원: APIName, DecisionName 유효한 통계: Minimum
ConsumedCapacity	초당 요청 수 CloudWatch 단위: Count 차원: APIName, DecisionName 유효한 통계: Sum
ConsumedLimit	소비된 일반 한도 양 차원: GeneralLimitType

지표	설명
ProvisionedRefillRate	버킷에 허용되는 초당 요청 수 차원: APIName, DecisionName 유효한 통계: Minimum
ProvisionedLimit	계정에 프로비저닝되는 일반 한도 양 차원: GeneralLimitType

차원	설명
Domain	워크플로 또는 활동이 실행 중인 Amazon SWF 도메인에 대한 데이터를 필터링합니다.
ActivityTypeName	활동 유형 이름에 대한 데이터를 필터링합니다.
ActivityTypeVersion	활동 유형 버전에 대한 데이터를 필터링합니다.
WorkflowTypeName	워크플로 실행의 워크플로 유형 이름에 대한 데이터를 필터링합니다.
WorkflowTypeVersion	워크플로 실행의 워크플로 유형 버전에 대한 데이터를 필터링합니다.
APIName	지정한 API 이름의 API에 대한 데이터를 필터링합니다.
DecisionName	지정한 결정 이름에 대한 데이터를 필터링합니다.
TaskListName	지정한 작업 목록 이름에 대한 데이터를 필터링합니다.
TaskListClassification	작업 목록의 분류에 대한 데이터를 필터링합니다. 결정 작업 목록의 값은 "D"이고 활동 작업 목록의 값은 "A"입니다.
ThrottlingScope	지정된 제한 범위로 데이터를 필터링합니다. 값은 계정 수준 할당량을 초과할 경우 "계정"이고 워크플로 수준 할당량을 초과할 경우 "워크플로"입니다.

Amazon SWF 비 ASCII 리소스 이름 및 CloudWatch 차원

Amazon SWF는 TaskList 및 DomainName과 같은 리소스 이름에 비 ASCII 문자를 허용합니다. 하지만 CloudWatch 지표의 차원 값에는 인쇄 가능한 ASCII 문자만 포함될 수 있습니다. Amazon SWF가 CloudWatch 요구 사항과 호환되는 차원 값을 사용할 수 있도록 이러한 요구 사항을 충족하지 않는 Amazon SWF 리소스 이름은 변환되며 다음과 같이 체크섬이 추가됩니다.

- 비 ASCII 문자는 모두 ?로 대체됩니다.
- 입력 문자열이나 변환된 문자열은 필요한 경우 잘립니다. 이렇게 하면 체크섬이 추가될 때 새 문자열 길이가 CloudWatch의 최대값을 초과하지 않게 됩니다.
- ASCII가 아닌 문자는 로 변환되므로 변환 전에 다른 ?일부 CloudWatch 지표 차원 값은 변환 후 동일한 것으로 보일 수 있습니다. 구분하기 쉽도록 밑줄(_) 뒤에 원래 리소스 이름의 SHA256 체크섬 처음 16자가 리소스 이름에 추가됩니다.

변환 예제:

- `test àpple`는 `test ?pple_82cc5b8e3a771d12`로 변환됩니다
- `àòà`는 `???_2fec5edbb2c05c22`로 변환됩니다.
- TaskList 이름 `àpplé`과 `âpplè`은 모두 `?ppl?`로 변환되며 동일합니다. 체크섬을 추가하면 고유 값 `?ppl?_f39a36df9d85a69d` 및 `?ppl?_da3efb4f11dd0f7f`가 반환됩니다.

> **Tip**
>
> SHA256 체크섬을 직접 생성할 수 있습니다. 예를 들어, `shasum` 명령줄 도구를 사용하려면 다음을 수행합니다.
>
> echo -n "<the original resource name>" | shasum -a 256 | cut -c1-16

를 사용하여 CloudWatch에 대한 Amazon SWF 지표 보기 AWS Management Console

Amazon CloudWatch는 Amazon SWF 워크플로 및 활동에 대해 확인 가능한 여러 지표를 제공합니다. AWS Management Console을 사용하여 Amazon SWF 워크플로 실행에 대한 지표를 보고 경보를 설정할 수 있습니다. 계속하려면 콘솔에 로그인한 상태여야 합니다.

사용 가능한 각 측정치에 대한 설명은 CloudWatch에 대한 Amazon SWF 지표 단원을 참조하십시오.

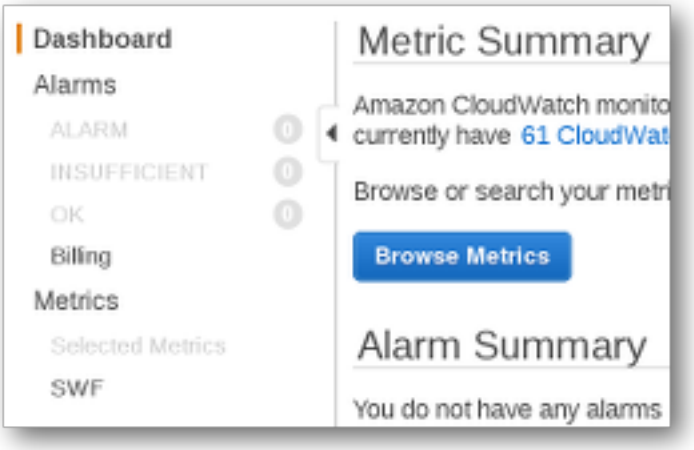
주제

- 지표 보기
- 경보 설정

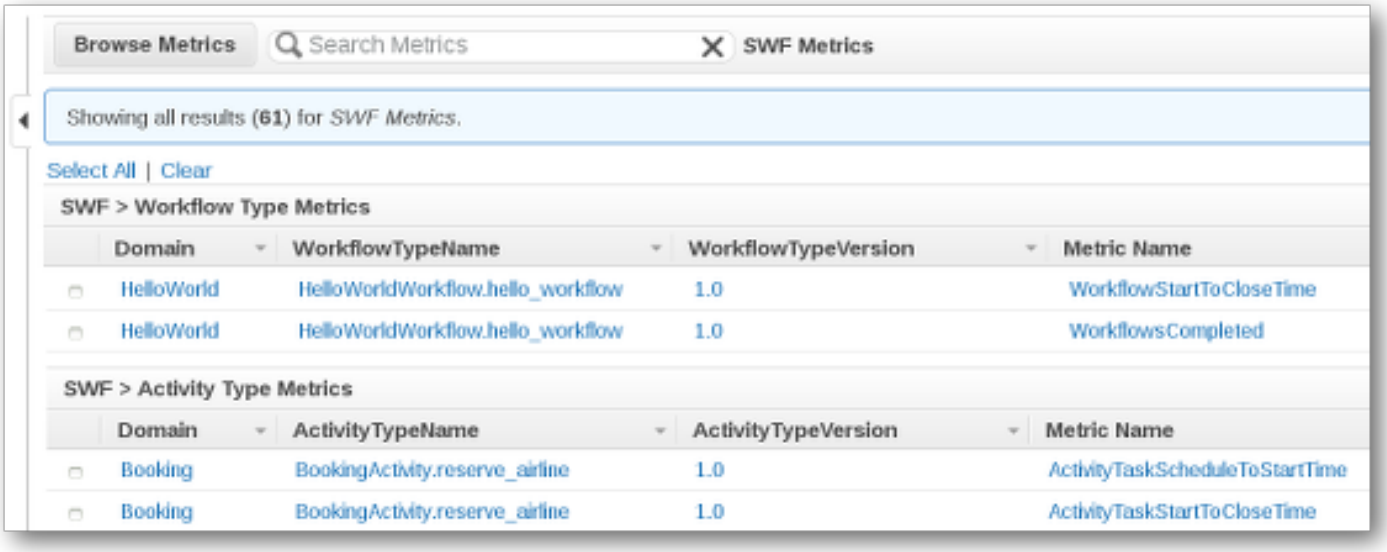
지표 보기

Amazon SWF에 대한 지표를 보려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/cloudwatch/ CloudWatch 콘솔을 엽니다.
2. 탐색 창의 [Metrics]에서 [SWF]를 선택합니다.



최근에 임의의 워크플로 실행을 실행한 경우 두 가지 측정치 목록 즉, [Workflow Type Metrics] 및 [Activity Type Metrics]가 표시됩니다.



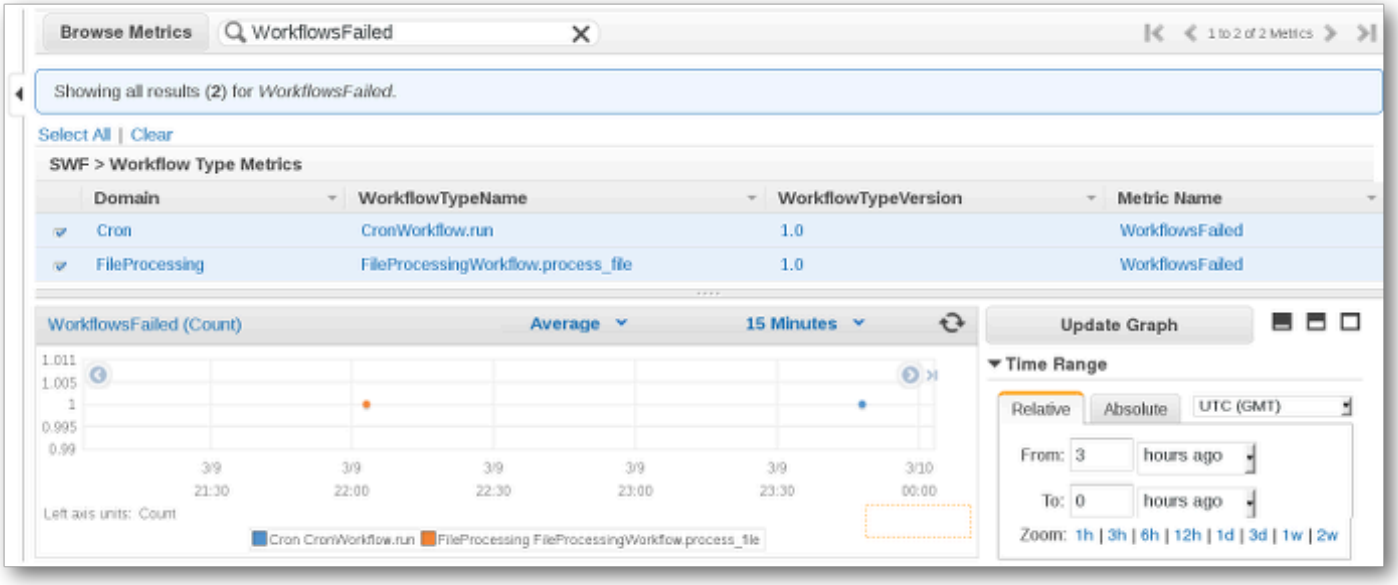
> **Note**
>
> 처음에는 [Workflow Type Metrics]만 보일 수 있습니다. 하지만 아래로 스크롤하면 [Activity Type Metrics]가 보입니다.

측정치는 한 번에 최대 50개의 최신 항목이 표시되고, 워크플로 및 활동 측정치가 구분되어 있습니다.

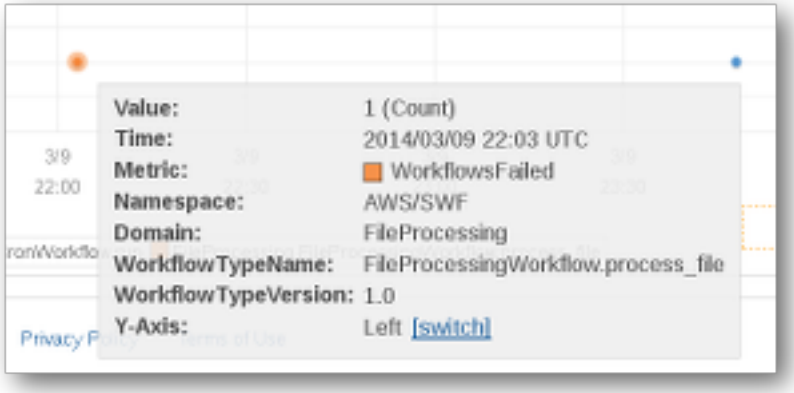
목록의 각 열 위에서 대화형 머리글을 사용하면 제공된 차원을 사용해 측정치를 정렬할 수 있습니다. 워크플로의 경우 차원은 도메인, WorkflowTypeName, WorkflowTypeVersion 및 메트릭 이름입니다. 활동의 경우 차원은 도메인, ActivityTypeName, ActivityTypeVersion 및 메트릭 이름입니다.

메트릭의 버전 유형은 CloudWatch에 대한 Amazon SWF 지표 단원을 참조하십시오.

목록의 측정치 행 옆에 있는 상자를 선택해 측정치에 대한 그래프를 보고 그래프 보기 오른쪽에 있는 [Time Range] 컨트롤을 사용해 그래프 파라미터를 변경할 수 있습니다.



그래프 상의 점 위에 커서를 올려 놓으면 그래프에 있는 점에 대한 자세한 내용이 표시됩니다. 점의 차원에 대한 세부 정보가 표시됩니다.



이러한 CloudWatch 지표 작업에 대한 자세한 정보는 Amazon CloudWatch 사용자 설명서의 지표 보기, 그래프 작성 및 게시를 참조하십시오.

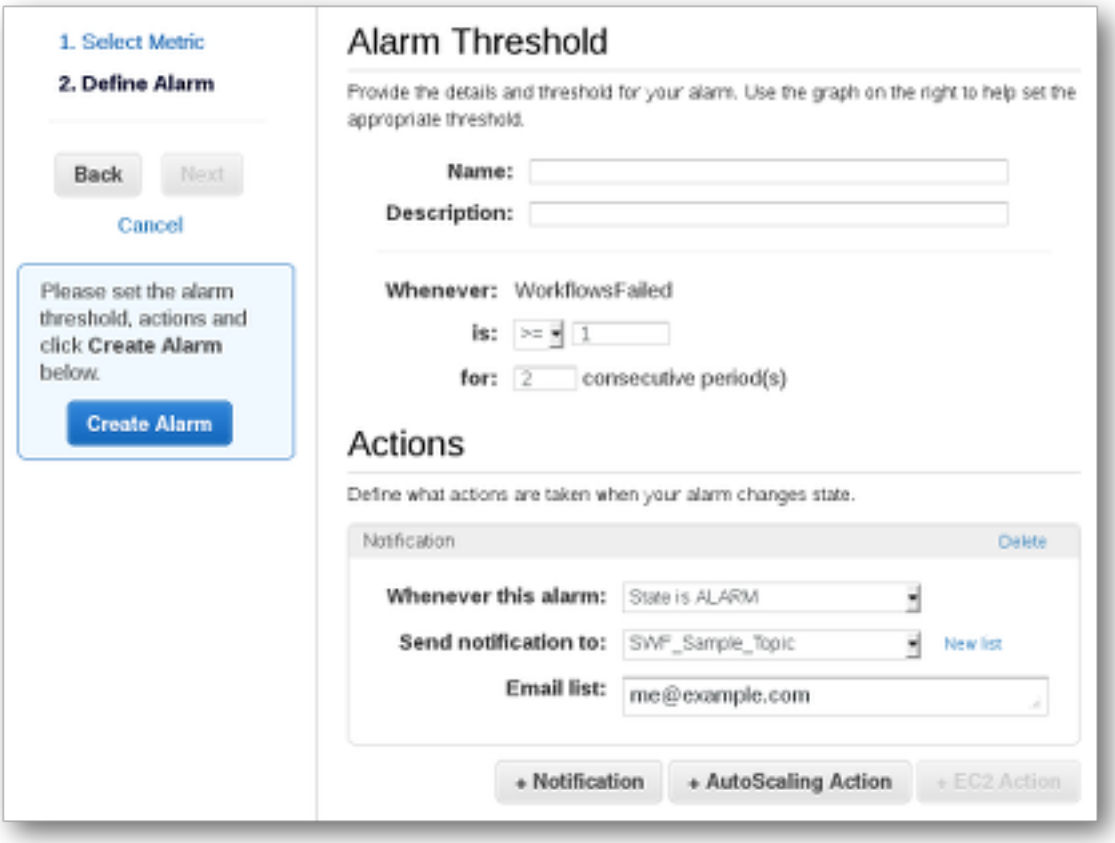
경보 설정

CloudWatch 경보를 사용하여 경보 임계값에 도달한 경우 알리는 등의 작업을 수행할 수 있습니다. 예를 들어, `WorkflowsFailed` 측정치가 특정 임계값을 초과할 때 SNS 주제에 알림을 보내거나 이메일을 보내는 경보를 설정할 수 있습니다.

측정치에 대한 경보를 설정하려면

1. 해당하는 상자를 선택해 측정치 하나를 선택합니다.

2. 그래프 오른쪽에 있는 [Tools] 컨트롤에서 [Create Alarm]을 선택합니다.
3. [Define Alarm] 화면에서 경보 임계값, 기간 파라미터 및 수행할 작업을 입력합니다.



CloudWatch 경보 설정 및 사용에 대한 자세한 내용은 Amazon CloudWatch 사용자 설명서의 Amazon CloudWatch 경보 생성을 참조하십시오.

를 사용하여 API 호출 기록 AWS CloudTrail

Amazon Simple Workflow Service는 사용자AWS CloudTrail, 역할 또는가 수행한 작업에 대한 레코드를 제공하는 서비스인와 통합됩니다 AWS 서비스. CloudTrail은 Amazon SWF에 대한 모든 API 직접 호출을 이벤트로 캡처합니다. 캡처되는 호출에는 Amazon SWF 콘솔로부터의 호출과 Amazon SWF API 작업에 대한 코드 호출이 포함됩니다. CloudTrail에서 수집한 정보를 사용하여 Amazon SWF에 수행된 요청, 요청이 수행된 IP 주소, 요청이 수행된 시간 및 추가 세부 정보를 확인할 수 있습니다.

모든 이벤트 또는 로그 항목에는 요청을 생성했던 사용자에 대한 정보가 포함됩니다. 자격 증명을 이용하면 다음을 쉽게 판단할 수 있습니다.

- 요청을 루트 사용자로 했는지 사용자 보안 인증으로 했는지 여부.

- IAM Identity Center 사용자를 대신하여 요청이 이루어졌는지 여부입니다.
- 역할 또는 페더레이션 사용자에 대한 임시 자격 증명을 사용하여 요청이 생성되었는지 여부.
- 다른 AWS 서비스에서 요청했는지 여부.

CloudTrail은 계정을 생성할 AWS 계정 때에서 활성화되며 CloudTrail 이벤트 기록에 자동으로 액세스할 수 있습니다. CloudTrail 이벤트 기록은 지난 90일 간 AWS 리전의 관리 이벤트에 대해 보기, 검색 및 다운로드가 가능하고, 수정이 불가능한 레코드를 제공합니다. 자세한 설명은 AWS CloudTrail 사용 설명서의 CloudTrail 이벤트 기록 작업을 참조하세요. Event history(이벤트 기록) 보기는 CloudTrail 요금이 부과되지 않습니다.

AWS 계정 지난 90일 동안 이벤트를 지속적으로 기록하려면 추적 또는 CloudTrail Lake 이벤트 데이터 스토어를 생성합니다.

CloudTrail 추적

CloudTrail은 추적을 사용하여 Amazon S3 버킷으로 로그 파일을 전송할 수 있습니다. 를 사용하여 생성된 모든 추적 AWS Management Console 은 다중 리전입니다. AWS CLI를 사용하여 단일 리전 또는 다중 리전 추적을 생성할 수 있습니다. 계정 AWS 리전 의 모든에서 활동을 캡처하므로 다중 리전 추적을 생성하는 것이 좋습니다. 단일 리전 추적을 생성하는 경우 추적의 AWS 리전에 로깅된 이벤트만 볼 수 있습니다. 추적에 대한 자세한 내용은 AWS CloudTrail 사용 설명서의 Creating a trail for your AWS 계정 및 Creating a trail for an organization을 참조하세요.

CloudTrail에서 추적을 생성하여 진행 중인 관리 이벤트의 사본 하나를 Amazon S3 버킷으로 무료로 전송할 수는 있지만, Amazon S3 스토리지 요금이 부과됩니다. CloudTrail 요금에 대한 자세한 내용은 AWS CloudTrail 요금을 참조하세요. Amazon S3 요금에 대한 자세한 내용은 Amazon S3 요금을 참조하세요.

CloudTrail Lake 이벤트 데이터 스토어

CloudTrail Lake를 사용하면 이벤트에 대해 SQL 기반 쿼리를 실행할 수 있습니다. CloudTrail Lake는 행 기반 JSON 형식의 기존 이벤트를 Apache ORC 형식으로 변환합니다. ORC는 빠른 데이터 검색에 최적화된 열 기반 스토리지 형식입니다. 이벤트는 이벤트 데이터 스토어로 집계되며, 이벤트 데이터 스토어는 고급 이벤트 선택기를 적용하여 선택한 기준을 기반으로 하는 변경 불가능한 이벤트 컬렉션입니다. 이벤트 데이터 스토어에 적용하는 선택기는 어떤 이벤트가 지속되고 쿼리할 수 있는지 제어합니다. CloudTrail Lake에 대한 자세한 내용은 AWS CloudTrail 사용 설명서의 AWS CloudTrail Lake 작업을 참조하세요.

CloudTrail Lake 이벤트 데이터 스토어 및 쿼리에는 비용이 발생합니다. 이벤트 데이터 스토어를 생성할 때 이벤트 데이터 스토어에 사용할 요금 옵션을 선택합니다. 요금 옵션에 따라 이벤트 모으기

및 저장 비용과 이벤트 데이터 스토어의 기본 및 최대 보존 기간이 결정됩니다. CloudTrail 요금에 대한 자세한 내용은 AWS CloudTrail 요금을 참조하세요.

CloudTrail의 데이터 이벤트

데이터 이벤트는 리소스 기반 또는 리소스에서 수행된 리소스 작업에 대한 정보를 제공합니다(예: Amazon S3 객체 읽기 또는 쓰기). 이를 데이터 영역 작업이라고도 합니다. 데이터 이벤트가 대량 활동인 경우도 있습니다. 기본적으로 CloudTrail은 데이터 이벤트를 로깅하지 않습니다. CloudTrail 이벤트 기록은 데이터 이벤트를 기록하지 않습니다.

데이터 이벤트에는 추가 요금이 적용됩니다. CloudTrail 요금에 대한 자세한 내용은 AWS CloudTrail 요금을 참조하세요.

CloudTrail 콘솔 AWS CLI또는 CloudTrail API 작업을 사용하여 Amazon SWF 리소스 유형에 대한 데이터 이벤트를 로깅할 수 있습니다. 데이터 이벤트를 로깅하는 방법에 대한 자세한 내용은 AWS CloudTrail 사용 설명서의 Logging data events with the AWS Management Console 및 Logging data events with the AWS Command Line Interface를 참조하세요.

다음 표에는 데이터 이벤트를 로깅할 수 있는 Amazon SWF 리소스 유형이 나열되어 있습니다. 데이터 이벤트 유형 열에는 CloudTrail 콘솔의 데이터 이벤트 유형 목록에서 선택할 값이 표시됩니다. resources.type 값 열에는 AWS CLI 또는 CloudTrail APIs를 사용하여 고급 이벤트 선택기를 구성할 때 지정하는 `resources.type` 값이 표시됩니다. CloudTrail에 로깅되는 데이터 API 열에는 리소스 유형에 대해 CloudTrail에 로깅된 API 호출이 표시됩니다.

`eventName`, `readOnly` 및 `resources.ARN` 필드를 필터링하여 중요한 이벤트만 로깅하도록 고급 이벤트 선택기를 구성할 수 있습니다. 이러한 필드에 대한 자세한 내용은 AWS CloudTrail API 참조의 AdvancedFieldSelector 섹션을 참조하세요.

데이터 이벤트 유형	resources.type 값	CloudTrail에 로깅되는 데이터 API
SWF 도메인	`AWS::SWF::Domain`	워크플로 이벤트 • CountClosedWorkflowExecutions • CountOpenWorkflowExecutions • DescribeWorkflowExecution

데이터 이벤트 유형	resources.type 값	CloudTrail에 로깅되는 데이터 API
		• ListClosedWorkflowExecutions • ListOpenWorkflowExecutions • GetWorkflowExecutionHistory • RequestCancelWorkflowExecution • SignalWorkflowExecution • StartWorkflowExecution • TerminateWorkflowExecution 작업 이벤트 • CountPendingActivityTasks • PollForDecisionTask • PollForActivityTask • RecordActivityTaskHeartbeat • RespondActivityTaskCanceled • RespondActivityTaskCompleted • RespondActivityTaskFailed • RespondDecisionTaskCompleted 결정 이벤트 • CancelTimer

데이터 이벤트 유형	resources.type 값	CloudTrail에 로깅되는 데이터 API
		• CancelWorkflowExecution • CompleteWorkflowExecution • ContinueAsNewWorkflowExecution • FailWorkflowExecution • RecordMarker • RequestCancelActivityTask • RequestCancelExternalWorkflowExecution • ScheduleActivityTask • ScheduleLambdaFunction • SignalExternalWorkflowExecution • StartChildWorkflowExecution • StartTimer

> **CloudTrail 이벤트 및 RespondDecisionTaskCompleted**
>
> RespondDecisionTaskCompleted 작업은 요청 페이로드의 결정 목록을 가져옵니다. 완료된 호출은 각 결정에 대해 하나씩, API 호출 자체에 대해 하나씩 N+1 CloudTrail 데이터 이벤트를 내보냅니다. 데이터 이벤트와 API 이벤트는 모두 동일한 요청 ID를 갖습니다.

CloudTrail의 관리 이벤트

관리 이벤트는의 리소스에서 수행되는 관리 작업에 대한 정보를 제공합니다 AWS 계정. 이를 컨트롤 플레인 작업이라고도 합니다. 기본적으로 CloudTrail은 관리 이벤트를 로깅합니다.

Amazon Simple Workflow Service는 다음과 같은 컨트롤 플레인 작업을 관리 이벤트로 CloudTrail에 기록합니다.

도메인 이벤트

- RegisterDomain
- DescribeDomain
- ListDomains
- DeprecateDomain
- UndeprecateDomain

활동 이벤트

- RegisterActivityType
- DescribeActivityType
- ListActivityTypes
- DeprecateActivityType
- UndeprecateActivityType
- DeleteActivityType

WorkflowType 이벤트

- RegisterWorkflowType
- DescribeWorkflowType
- ListWorkflowTypes
- DeprecateWorkflowType
- UndeprecateWorkflowType
- DeleteWorkflowType

태그 이벤트

- TagResource
- UntagResource
- ListTagsforResource

예제 이벤트

이벤트는 모든 소스로부터의 단일 요청을 나타내며 요청된 API 작업, 작업 날짜와 시간, 요청 파라미터 등에 대한 정보가 들어 있습니다. CloudTrail 로그 파일은 퍼블릭 API 직접 호출의 주문 스택 추적이 아니므로 이벤트가 특정 순서로 표시되지 않습니다.

다음 예제는 `CountClosedWorkflowExecutions` 작업을 시연하는 CloudTrail 이벤트를 보여줍니다.


```
{
    "eventVersion": "1.09",
    "userIdentity": {
      "type": "AssumedRole",
      "principalId": "1234567890abcdef02345:admin",
      "arn": "arn:aws:sts::111122223333:assumed-role/Admin/admin",
      "accountId": "111122223333",
      "accessKeyId": "abcdef01234567890abc",
      "sessionContext": {
        "sessionIssuer": {
          "type": "Role",
          "principalId": "1234567890abcdef02345",
          "arn": "arn:aws:iam::111122223333:role/Admin",
          "accountId": "111122223333",
          "userName": "Admin"
        },
        "attributes": {
          "creationDate": "2023-11-23T16:37:38Z",
          "mfaAuthenticated": "false"
        }
      }
    },
    "eventTime": "2023-11-23T17:52:46Z",
    "eventSource": "swf.amazonaws.com",
    "eventName": "CountClosedWorkflowExecutions",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "198.51.100.42",
    "userAgent": "aws-internal/3 aws-sdk-java/1.11.42",
    "requestParameters": {
      "domain": "nsg-domain",
      "closeTimeFilter": {
        "oldestDate": "Nov 23, 2023 5:52:46 PM",
        "latestDate": "Nov 23, 2023 5:52:46 PM"
      }
```

```
    },
    "responseElements": null,
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEbbbbb",
    "readOnly": true,
    "resources": [
      {
        "accountId": "111122223333",
        "type": "AWS::SWF::Domain",
        "ARN": "arn:aws:swf:us-east-1:111122223333:/domain/nsg-domain"
      }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": false,
    "recipientAccountId": "111122223333",
    "eventCategory": "Data",
    "tlsDetails": {
      "clientProvidedHostHeader": "swf.example.amazondomains.com"
    }
  }
```


CloudTrail 레코드 콘텐츠에 대한 자세한 내용은 AWS CloudTrail 사용 설명서의 CloudTrail record contents를 참조하세요.

Amazon SWF용 EventBridge 실행 상태 변경

Amazon EventBridge를 사용하여 AWS 리소스의 상태 변경 또는 이벤트에 응답합니다. Amazon SWF가 이벤트를 출력하면 항상 계정의 기본 EventBridge 이벤트 버스로 이동합니다. 이벤트에 대한 규칙을 생성하고, 기본 이벤트 버스에 연결하고, EventBridge가 규칙과 일치하는 이벤트를 수신할 때 수행할 대상 작업을 지정할 수 있습니다. 이러한 방식으로 `GetWorkflowExecutionHistory` API를 사용하여 지속적으로 폴링하지 않고도 워크플로를 모니터링할 수 있습니다. 워크플로 실행의 변경 사항에 따라 EventBridge 대상을 사용하여 AWS Lambda 함수를 호출하고 Amazon Simple Notification Service(Amazon SNS) 주제에 메시지를 게시하는 등의 작업을 수행할 수 있습니다.

`DescribeWorkflowExecution`를 사용하여 실행 상태 변경 이벤트의 전체 내용을 볼 수 있습니다.

자세한 내용은 Amazon EventBridge 사용 설명서를 참조하세요.

EventBridge 이벤트

기록 이벤트 유형에는 실행 상태 변경이 포함됩니다. 각 이벤트의 `detail` 섹션에는 최소한 다음 파라미터가 포함되어 있습니다.

- eventId: GetWorkflowExecutionHistory에 표시된 이벤트 ID입니다.
- workflowExecutionDetail: 이벤트가 발생했을 때의 워크플로 상태입니다.
- eventType: 기록 이벤트 유형은 다음 중 하나입니다.
  - ActivityTaskCanceled
  - ActivityTaskFailed
  - ActivityTaskTimedOut
  - WorkflowExecutionCanceled
  - WorkflowExecutionCompleted
  - WorkflowExecutionFailed
  - WorkflowExecutionStarted
  - WorkflowExecutionTerminated
  - WorkflowExecutionTimedOut
  - WorkflowExecutionContinuedAsNew
  - CancelTimerFailed
  - CancelWorkflowExecutionFailed
  - ChildWorkflowExecutionFailed
  - ChildWorkflowExecutionTimedOut
  - CompleteWorkflowExecutionFailed
  - ContinueAsNewWorkflowExecutionFailed
  - DecisionTaskTimedOut
  - FailWorkflowExecutionFailed
  - RecordMarkerFailed
  - RequestCancelActivityTaskFailed
  - RequestCancelExternalWorkflowExecutionFailed
  - ScheduleActivityTaskFailed
  - SignalExternalWorkflowExecutionFailed
  - StartActivityTaskFailed
  - StartChildWorkflowExecutionFailed
  - StartTimerFailed
  - TimerCanceled

- LambdaFunctionFailed
- LambdaFunctionTimedOut
- StartLambdaFunctionFailed
- ScheduleLambdaFunctionFailed

Amazon SWF 이벤트 예제

다음은 이벤트를 EventBridge로 보내는 Amazon SWF의 예제입니다.

주제

- 실행 시작
- 실행 완료
- 실행 실패
- 실행 시간 초과
- 실행 종료

각각의 경우, 이벤트 데이터의 `detail` 섹션은 `DescribeWorkflowExecution` API와 동일한 정보를 제공합니다. `executionStatus` 필드는 이벤트가 송신된 시점의 실행 상태(`OPEN` 또는 `CLOSED`)를 나타냅니다.

실행 시작

```
{
  "version": "0",
  "id": "444444444444",
  "detail-type": "Simple Workflow Execution State Change",
  "source": "aws.swf",
  "account": "444444444444",
  "time": "2020-05-08T15:57:38Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:swf:us-east-1:444444444444:/domain/SimpleWorkflowUserSimulator"
  ],
  "detail": {
    "eventId": 1,
    "eventType": "WorkflowExecutionStarted",
    "workflowExecutionDetail": {
      "executionInfo": {
```

```
        "execution": {
          "workflowId": "123456789012",
          "runId": "AKIAIOSFODNN7EXAMPLE"
        },
        "workflowType": {
          "name": "SimpleWorkflowUserSimulator",
          "version": "myWorkflow"
        },
        "startTimestamp": 1588953458484,
        "closeTimestamp": null,
        "executionStatus": "OPEN",
        "closeStatus": null,
        "parent": null,
        "parentExecutionArn": null,
        "tagList": null,
        "cancelRequested": false
      },
      "executionConfiguration": {
        "taskStartToCloseTimeout": "60",
        "executionStartToCloseTimeout": "1000",
        "taskList": {
          "name": "444444444444"
        },
        "taskPriority": null,
        "childPolicy": "ABANDON",
        "lambdaRole": "arn:aws:iam::444444444444:role/BasicSWFLambdaExecution"
      },
      "openCounts": {
        "openActivityTasks": 0,
        "openDecisionTasks": 1,
        "openTimers": 0,
        "openChildWorkflowExecutions": 0,
        "openLambdaFunctions": 0
      },
      "latestActivityTaskTimestamp": null,
    }
  }
}
```


실행 완료


```
{
  "version": "0",
```

```
  "id": "1111-2222-3333",
  "detail-type": "Simple Workflow Execution State Change",
  "source": "aws.swf",
  "account": "444455556666",
  "time": "2020-05-08T15:57:39Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:swf:us-east-1:444455556666:/domain/SimpleWorkflowUserSimulator"
  ],
  "detail": {
    "eventId": 35,
    "eventType": "WorkflowExecutionCompleted",
    "workflowExecutionDetail": {
      "executionInfo": {
        "execution": {
          "workflowId": "1234-5678-9012",
          "runId": "777788889999"
        },
        "workflowType": {
          "name": "SimpleWorkflowUserSimulator",
          "version": "myWorkflow"
        },
        "startTimestamp": 1588953458820,
        "closeTimestamp": 1588953459448,
        "executionStatus": "CLOSED",
        "closeStatus": "COMPLETED",
        "parent": null,
        "parentExecutionArn": null,
        "tagList": null,
        "cancelRequested": false
      },
      "executionConfiguration": {
        "taskStartToCloseTimeout": "60",
        "executionStartToCloseTimeout": "1000",
        "taskList": {
          "name": "1111-1111-1111"
        },
        "taskPriority": null,
        "childPolicy": "ABANDON",
        "lambdaRole": "arn:aws:iam::444455556666:role/BasicSWFLambdaExecution"
      },
      "openCounts": {
        "openActivityTasks": 0,
        "openDecisionTasks": 0,
```

```
        "openTimers": 0,
        "openChildWorkflowExecutions": 0,
        "openLambdaFunctions": 0
      },
      "latestActivityTaskTimestamp": 1588953459402,
    }
  }
}
```

실행 실패

```
{
  "version": "0",
  "id": "1111-2222-3333",
  "detail-type": "Simple Workflow Execution State Change",
  "source": "aws.swf",
  "account": "444455556666",
  "time": "2020-05-08T15:57:38Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:swf:us-east-1:444455556666:/domain/SimpleWorkflowUserSimulator"
  ],
  "detail": {
    "eventId": 11,
    "eventType": "WorkflowExecutionFailed",
    "workflowExecutionDetail": {
      "executionInfo": {
        "execution": {
          "workflowId": "1234-5678-9012",
          "runId": "777788889999"
        },
        "workflowType": {
          "name": "SimpleWorkflowUserSimulator",
          "version": "myWorkflow"
        },
        "startTimestamp": 1588953158481,
        "closeTimestamp": 1588953458560,
        "executionStatus": "CLOSED",
        "closeStatus": "FAILED",
        "parent": null,
        "parentExecutionArn": null,
        "tagList": null,
        "cancelRequested": false
```

```
      },
      "executionConfiguration": {
        "taskStartToCloseTimeout": "60",
        "executionStartToCloseTimeout": "1000",
        "taskList": {
          "name": "1111-1111-1111"
        },
        "taskPriority": null,
        "childPolicy": "ABANDON",
        "lambdaRole": "arn:aws:iam::444455556666:role/BasicSWFLambdaExecution"
      },
      "openCounts": {
        "openActivityTasks": 0,
        "openDecisionTasks": 0,
        "openTimers": 0,
        "openChildWorkflowExecutions": 0,
        "openLambdaFunctions": 0
      },
      "latestActivityTaskTimestamp": null,
    }
  }
}
```

실행 시간 초과

```
{
  "version": "0",
  "id": "1111-2222-3333",
  "detail-type": "Simple Workflow Execution State Change",
  "source": "aws.swf",
  "account": "444455556666",
  "time": "2020-05-05T17:26:30Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:swf:us-east-1:444455556666:/domain/SimpleWorkflowUserSimulator"
  ],
  "detail": {
    "eventId": 6,
    "eventType": "WorkflowExecutionTimedOut",
    "workflowExecutionDetail": {
      "executionInfo": {
        "execution": {
          "workflowId": "1234-5678-9012",
```

```
          "runId": "777788889999"
        },
        "workflowType": {
          "name": "SimpleWorkflowUserSimulator",
          "version": "myWorkflow"
        },
        "startTimestamp": 1588698073748,
        "closeTimestamp": 1588699590745,
        "executionStatus": "CLOSED",
        "closeStatus": "TIMED_OUT",
        "parent": null,
        "parentExecutionArn": null,
        "tagList": null,
        "cancelRequested": false
      },
      "executionConfiguration": {
        "taskStartToCloseTimeout": "60",
        "executionStartToCloseTimeout": "1000",
        "taskList": {
          "name": "1111-1111-1111"
        },
        "taskPriority": null,
        "childPolicy": "ABANDON",
        "lambdaRole": "arn:aws:iam::444455556666:role/BasicSWFLambdaExecution"
      },
      "openCounts": {
        "openActivityTasks": 1,
        "openDecisionTasks": 0,
        "openTimers": 0,
        "openChildWorkflowExecutions": 0,
        "openLambdaFunctions": 0
      },
      "latestActivityTaskTimestamp": 1588699585802,
    }
  }
}
```

실행 종료

```
{
  "version": "0",
  "id": "1111-2222-3333",
  "detail-type": "Simple Workflow Execution State Change",
```

```
  "source": "aws.swf",
  "account": "444455556666",
  "time": "2020-05-08T22:37:26Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:swf:us-east-1:444455556666:/domain/canary"
  ],
  "detail": {
    "eventId": 48,
    "eventType": "WorkflowExecutionTerminated",
    "workflowExecutionDetail": {
      "executionInfo": {
        "execution": {
          "workflowId": "1234-5678-9012",
          "runId": "777788889999"
        },
        "workflowType": {
          "name": "1111-1111-1111",
          "version": "1.3"
        },
        "startTimestamp": 1588977445279,
        "closeTimestamp": 1588977446062,
        "executionStatus": "CLOSED",
        "closeStatus": "TERMINATED",
        "parent": null,
        "parentExecutionArn": null,
        "tagList": null,
        "cancelRequested": false
      },
      "executionConfiguration": {
        "taskStartToCloseTimeout": "60",
        "executionStartToCloseTimeout": "120",
        "taskList": {
          "name": "1111-1111-1111-2222-2222-2222"
        },
        "taskPriority": null,
        "childPolicy": "TERMINATE",
        "lambdaRole": null
      },
      "openCounts": {
        "openActivityTasks": 0,
        "openDecisionTasks": 1,
        "openTimers": 0,
        "openChildWorkflowExecutions": 0,
```

```
        "openLambdaFunctions": 0
      },
      "latestActivityTaskTimestamp": 1588977445882,
    }
  }
}
```

Amazon Simple Workflow Service AWS 사용자 알림 에서 사용

AWS 사용자 알림을 사용하여 Amazon Simple Workflow Service 이벤트에 대한 알림을 받을 전송 채널을 설정할 수 있습니다. 이벤트가 지정한 규칙과 일치하면 알림을 받습니다. 이메일, 채팅 애플리케이션의 Amazon Q Developer 채팅 알림 또는 AWS Console Mobile Application 푸시 알림을 비롯한 여러 채널을 통해 이벤트에 대한 알림을 받을 수 있습니다. 콘솔 알림 센터에서도 알림을 볼 수 있습니다. 사용자 알림 은 집계를 지원하므로 특정 이벤트 중에 받는 알림 수를 줄일 수 있습니다.

Amazon Simple Workflow Service에 대한 규정 준수 확인

타사 감사자는 여러 AWS 규정 준수 프로그램의 일환으로 Amazon Simple Workflow Service의 보안 및 규정 준수를 평가합니다. 여기에는 SOC, PCI, FedRAMP, HIPAA 등이 포함됩니다.

특정 규정 준수 프로그램 범위의 AWS 서비스 목록은 규정 준수 프로그램 AWS 제공 범위 내 서비스규정 준수 프로그램 . 일반 정보는 AWS 규정 준수 프로그램.

를 사용하여 타사 감사 보고서를 다운로드할 수 있습니다 AWS Artifact. 자세한 내용은 AWS Artifact에서 보고서 다운로드.

Amazon SWF 사용 시 규정 준수 책임은 데이터의 민감도, 회사의 규정 준수 목표 및 관련 법률과 규정에 따라 결정됩니다. AWS 는 규정 준수를 지원할 다음과 같은 리소스를 제공합니다.

- 보안 및 규정 준수 퀵 스타트 가이드 -이 배포 가이드에서는 아키텍처 고려 사항에 대해 설명하고 보안 및 규정 준수 중심 기준 환경을 배포하기 위한 단계를 제공합니다 AWS.
- HIPAA 보안 및 규정 준수를 위한 설계 백서 -이 백서에서는 기업이 AWS 를 사용하여 HIPAA 준수 애플리케이션을 생성하는 방법을 설명합니다.
- AWS 규정 준수 리소스 -이 워크북 및 가이드 모음은 산업 및 위치에 적용될 수 있습니다.
- AWS Config 개발자 안내서의 규칙을 사용하여 리소스 평가 -이 AWS Config 서비스는 리소스 구성이 내부 관행, 업계 지침 및 규정을 얼마나 잘 준수하는지 평가합니다.
- AWS Security Hub CSPM -이 AWS 서비스는 보안 업계 표준 및 모범 사례 준수 여부를 확인하는 데 도움이 AWS 되는 내 보안 상태에 대한 포괄적인 보기를 제공합니다.

Amazon Simple Workflow Service 복원성

AWS 글로벌 인프라는 AWS 리전 및 가용 영역을 중심으로 구축됩니다. AWS 리전은 물리적으로 분리되고 격리된 여러 가용 영역을 제공하며,이 가용 영역은 지연 시간이 짧고 처리량이 높으며 중복성이 높은 네트워킹과 연결됩니다. 가용 영역을 사용하면 중단 없이 영역 간에 자동으로 장애 극복 조치가 이루어지는 애플리케이션 및 데이터베이스를 설계하고 운영할 수 있습니다. 가용 영역은 기존의 단일 또는 다중 데이터 센터 인프라보다 가용성, 내결함성, 확장성이 뛰어납니다.

AWS 리전 및 가용 영역에 대한 자세한 내용은 AWS 글로벌 인프라를 참조하세요.

AWS 글로벌 인프라 외에도 Amazon SWF는 데이터 복원력 및 백업 요구 사항을 지원하는 몇 가지 기능을 제공합니다.

Amazon Simple Workflow Service의 인프라 보안

관리형 서비스인는 AWS 글로벌 네트워크 보안으로 보호됩니다. AWS 보안 서비스 및가 인프라를 AWS 보호하는 방법에 대한 자세한 내용은 AWS 클라우드 보안을 참조하세요. 인프라 보안 모범 사례를 사용하여 환경을 설계하려면 보안 원칙 AWS Well-Architected Framework의 인프라 보호를 참조하세요 AWS .

AWS 에서 게시한 API 호출을 사용하여 네트워크를 통해에 액세스합니다. 고객은 다음을 지원해야 합니다.

- Transport Layer Security(TLS) TLS 1.2는 필수이며 TLS 1.3을 권장합니다.
- DHE(Ephemeral Diffie-Hellman) 또는 ECDHE(Elliptic Curve Ephemeral Diffie-Hellman)와 같은 완전 전송 보안(PFS)이 포함된 암호 제품군 Java 7 이상의 최신 시스템은 대부분 이러한 모드를 지원합니다.

이러한 API 태스크는 어떤 네트워크 위치에서든 직접적으로 호출할 수 있지만, Amazon SWF는 소스 IP 주소에 따른 제한 사항을 포함할 수 있는 리소스 기반 액세스 정책을 지원합니다. Amazon SWF 정책을 사용하여 특정 Amazon Virtual Private Cloud(VPC) 엔드포인트 또는 특정 VPC에서 액세스를 제어할 수도 있습니다. 이렇게 하면 네트워크 내의 특정 VPC에서만 지정된 Amazon SWF 리소스에 대한 AWS 네트워크 액세스가 효과적으로 격리됩니다.

Amazon Simple Workflow Service의 구성 및 취약성 분석

구성 및 IT 제어는 AWS 와 고객 간의 공동 책임입니다. 자세한 내용은 AWS 공동 책임 모델을 참조하세요.

Amazon Simple Workflow Service AWS CLI 에서 사용

Amazon Simple Workflow Service의 많은 기능은 AWS CLI에서 액세스할 수 있습니다. 는 AWS Management Console 에서 Amazon SWF를 사용하거나 경우에 따라 Amazon SWF API 및를 사용한 프로그래밍에 대한 대안을 AWS CLI 제공합니다 AWS Flow Framework.

예를 들어 AWS CLI 를 사용하여 새 워크플로 유형을 등록할 수 있습니다.

```
aws swf register-workflow-type --domain MyDomain --name "MySimpleWorkflow" --workflow-
version "v1"
```

또한 등록된 워크플로 유형을 나열할 수도 있습니다.

```
aws swf list-workflow-types --domain MyDomain --registration-status REGISTERED
```

다음은 JSON으로 작성된 기본 출력의 예입니다.

```
{
    "typeInfos": [
        {
            "status": "REGISTERED",
            "creationDate": 1377471607.752,
            "workflowType": {
                "version": "v1",
                "name": "MySimpleWorkflow"
            }
        },
        {
            "status": "REGISTERED",
            "creationDate": 1371454149.598,
            "description": "MyDomain subscribe workflow",
            "workflowType": {
                "version": "v3",
                "name": "subscribe"
            }
        }
    ]
}
```

의 Amazon SWF 명령은 워크플로 실행을 시작 및 관리하고, 활동 작업을 폴링하고, 작업 하트비트를 기록하는 등의 기능을 AWS CLI 제공합니다. 사용 가능한 인수에 대한 설명과 사용법을 보여주는 예제

가 포함된 전체 Amazon SWF 명령 목록을 보려면 AWS CLI 명령 참조의 Amazon SWF 명령을 참조하십시오.

AWS CLI 명령은 Amazon SWF API를 면밀히 따르므로 AWS CLI 를 사용하여 기본 Amazon SWF API에 대해 알아볼 수 있습니다. 기존 API 지식을 활용하여 프로토타입 코드를 만들거나 명령줄에서 Amazon SWF 작업을 수행할 수도 있습니다.

에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서를 AWS CLI참조하세요.

Amazon SWF API 작업

에 설명된 AWS SDKs를 사용하는 것 외에도 HTTP API를 직접 사용할 AWS SDKs 사용하여 개발수 있습니다.

이 API를 사용하려면 도메인, 워크플로우 및 활동에 사용할 리전에 맞는 SWF 엔드포인트로 HTTP 요청을 전송합니다. Amazon SWF의 HTTP 요청에 대한 자세한 내용은 Amazon SWF에 대한 HTTP 요청 만들기 단원을 참조하십시오.

이 단원에서는 HTTP API를 사용해 Amazon SWF에서 워크플로를 개발하는 과정을 기본적으로 소개합니다. 타이머 사용, CloudTrail을 사용한 로깅, 워크플로에 태그 지정과 같은 고급 기능에 대해서는 Amazon SWF의 기본 워크플로 개념 단원에서 설명합니다.

주제

Amazon SWF에 대한 HTTP 요청 만들기

AWS SDKs 중 하나를 사용하지 않는 경우 POST 요청 방법을 사용하여 HTTP를 통해 Amazon Simple Workflow Service(Amazon SWF) 작업을 수행할 수 있습니다. POST 메서드를 사용하는 경우, 요청의 헤더에 작업을 지정하고 요청 본문에 JSON 형식으로 작업 데이터를 입력해야 합니다.

HTTP 헤더 콘텐츠

Amazon SWF는 HTTP 요청의 헤더에 다음 정보를 제공해야 합니다.

- `host` Amazon SWF 엔드포인트
- `x-amz-date` HTTP `Date` 헤더 또는 AWS `x-amz-date` header에 타임스탬프를 제공해야 합니다(일부 HTTP 클라이언트 라이브러리에서는 `Date` 헤더를 설정할 수 없음). `x-amz-date` 헤더가 있으면 요청 인증 시 모든 `Date` 헤더가 무시됩니다.

  HTTP/1.1 RFC에 지정된 다음 3개 형식 중 하나로 날짜를 지정해야 합니다.

  - Sun, 06 Nov 1994 08:49:37 GMT (RFC 822, RFC 1123 이후)
  - Sunday, 06-Nov-94 08:49:37 GMT(RFC 1036에 의해 폐기된 RFC 850)
  - Sun Nov 6 08:49:37 1994 (ANSI C's asctime() 형식)
- `x-amzn-authorization`: 다음 형식의 서명된 요청 파라미터입니다.

  ```
  AWS3 AWSAccessKeyId=####,Algorithm=HmacSHA256, [,SignedHeaders=Header1;Header2;...]
  Signature=S(StringToSign)
  ```

  `AWS3` - 요청에 서명하는 데 사용되는 인증 버전을 나타내는 AWS 구현별 태그입니다(현재 Amazon SWF의 경우이 값은 항상 임`AWS3`).

  `AWSAccessKeyId` - AWS 액세스 키 ID입니다.

  `Algorithm` – 서명할 문자열의 HMAC-SHA 값(예: `HmacSHA256` 또는 `HmacSHA1`)을 생성하는 데 사용되는 알고리즘입니다.

  `Signature` – Base64[알고리즘(StringToSign, SigningKey)]. 자세한 내용은 Amazon SWF에 대한 HMAC-SHA 서명 계산 단원을 참조하십시오.

  `SignedHeaders` – (선택)이 헤더가 있는 경우, 규정화된 HttpHeaders 계산에 사용되는 모든 HTTP 헤더 목록을 포함해야 합니다. 세미콜론 문자(;)(ASCII 문자 59) 하나를 목록 값의 구분 기호로 사용해야 합니다.
- `x-amz-target` – 요청 및 데이터 작업의 대상 서비스로 다음과 같은 형식을 띕니다.

  `com.amazonaws.swf.service.model.SimpleWorkflowService.` + *`<action>`*

  예: `com.amazonaws.swf.service.model.SimpleWorkflowService.RegisterDomain`

- content-type – JSON 및 application/json; charset=UTF-8 같은 문자 세트를 지정하는 데 필요한 유형입니다.

다음은 도메인을 만들기 위한 HTTP 요청의 예제 헤더입니다.

```
POST http://swf.us-east-1.amazonaws.com/ HTTP/1.1
Host: swf.us-east-1.amazonaws.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.2.25) Gecko/20111212
 Firefox/3.6.25 ( .NET CLR 3.5.30729; .NET4.0E)
Accept: application/json, text/javascript, */*
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Content-Type: application/json; charset=UTF-8
X-Requested-With: XMLHttpRequest
X-Amz-Date: Fri, 13 Jan 2012 18:42:12 GMT
X-Amz-Target: com.amazonaws.swf.service.model.SimpleWorkflowService.RegisterDomain
Content-Encoding: amz-1.0
X-Amzn-Authorization: AWS3
 AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE,Algorithm=HmacSHA256,SignedHeaders=Host;X-Amz-
Date;X-Amz-Target;Content-Encoding,Signature=tzjkF55lxAxPhzp/BRGFYQRQRq6CqrM254dTDE/
EncI=
Referer: http://swf.us-east-1.amazonaws.com/explorer/index.html
Content-Length: 91
Pragma: no-cache
Cache-Control: no-cache

{"name": "867530902",
 "description": "music",
 "workflowExecutionRetentionPeriodInDays": "60"}
```

다음은 해당하는 HTTP 응답의 예입니다.

```
HTTP/1.1 200 OK
Content-Length: 0
Content-Type: application/json
x-amzn-RequestId: 4ec4ac3f-3e16-11e1-9b11-7182192d0b57
```

HTTP 본문

HTTP 요청의 본문에는 HTTP 요청의 헤더에 지정한 작업의 데이터가 포함되며, JSON 데이터 형식을 사용하여 데이터 값과 데이터 구조를 동시에 표현합니다. 괄호 표기를 사용하여 요소를 다른 요소 안에 중첩할 수 있습니다. 예를 들어, 다음은 Unix Time 표기법을 사용하여 지정된 두 시점 사이에 시작된 모든 워크플로 실행을 나열하라는 요청을 보여줍니다.

```
{
 "domain": "867530901",
 "startTimeFilter":
 {
   "oldestDate": 1325376070,
  "latestDate": 1356998399
 },
 "tagFilter":
 {
   "tag": "music purchase"
 }
}
```

예제 Amazon SWF JSON 요청 및 응답

다음 예제는 이전에 생성한 도메인을 설명하기 위한 Amazon SWF에 대한 요청을 보여줍니다. 그런 다음에는 Amazon SWF 응답을 보여줍니다.

HTTP POST 요청

```
POST http://swf.us-east-1.amazonaws.com/ HTTP/1.1
Host: swf.us-east-1.amazonaws.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.2.25) Gecko/20111212
 Firefox/3.6.25 ( .NET CLR 3.5.30729; .NET4.0E)
Accept: application/json, text/javascript, */*
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Content-Type: application/json; charset=UTF-8
X-Requested-With: XMLHttpRequest
X-Amz-Date: Sun, 15 Jan 2012 03:13:33 GMT
X-Amz-Target: com.amazonaws.swf.service.model.SimpleWorkflowService.DescribeDomain
```

```
Content-Encoding: amz-1.0
X-Amzn-Authorization: AWS3
 AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE,Algorithm=HmacSHA256,SignedHeaders=Host;X-Amz-
Date;X-Amz-Target;Content-
Encoding,Signature=IFJtq3M366CHqMlTpyqYqd9z0ChCoKDC5SCJBsLifu4=
Referer: http://swf.us-east-1.amazonaws.com/explorer/index.html
Content-Length: 21
Pragma: no-cache
Cache-Control: no-cache

{"name": "867530901"}
```

Amazon SWF 응답

```
HTTP/1.1 200 OK
Content-Length: 137
Content-Type: application/json
x-amzn-RequestId: e86a6779-3f26-11e1-9a27-0760db01a4a8

{"configuration":
  {"workflowExecutionRetentionPeriodInDays": "60"},
 "domainInfo":
  {"description": "music",
   "name": "867530901",
   "status": "REGISTERED"}
}
```

프로토콜(HTTP/1.1) 다음에는 상태 코드(200)가 따라 옵니다. 코드 값 200은 작업 성공을 나타냅니다.

Amazon SWF는 null 값을 직렬화하지 않습니다. JSON 구문 분석기가 요청에 대한 null 값을 직렬화하도록 설정된 경우 Amazon SWF에서는 이러한 값을 무시합니다.

Amazon SWF에 대한 HMAC-SHA 서명 계산

Amazon SWF에 대한 모든 요청은 인증해야 합니다. AWS SDKs는 요청에 자동으로 서명하고 토큰 기반 인증을 관리합니다. 그러나 자체 HTTP POST 요청을 작성하려는 경우 요청 인증의 일부로 HTTP POST Header 콘텐츠에 대한 x-amzn-authorization 값을 생성해야 합니다.

헤더 형식 지정에 대한 자세한 내용은 HTTP 헤더 콘텐츠 단원을 참조하십시오. AWS 버전 3 서명 AWS SDK for Java 구현은 AWSSigner.java 클래스를 참조하세요.

요청 서명 생성

HMAC-SHA 요청 서명을 생성하려면 AWS 자격 증명(액세스 키 ID 및 보안 키)을 얻어야 합니다.

> **Important**
>
> SHA1 또는 SHA256을 사용하여 요청에 서명할 수 있습니다. 그러나 서명 프로세스 전체에서 동일한 방법을 사용해야 합니다. 선택한 메서드는 HTTP 헤더의 `Algorithm` 이름 값과 일치해야 합니다.

요청 서명을 생성하려면

1. HTTP 요청 헤더의 표준 양식을 생성합니다. HTTP 헤더의 표준 양식에는 다음이 포함됩니다.

   - `host`
   - `x-amz-`로 시작되는 모든 헤더 요소

   포함된 헤더에 대한 자세한 내용은 HTTP 헤더 콘텐츠 단원을 참조하십시오.

   a. 각 헤더 이름-값 페어의 경우 헤더 이름(헤더 값 아님)을 소문자로 변환합니다.

   b. 헤더 이름 맵을 쉼표로 구분된 헤더 값으로 작성합니다.

   ```
   x-amz-example: value1
   x-amz-example: value2  =>  x-amz-example:value1,value2
   ```

   자세한 내용은 RFC 2616의 섹션 4.2를 참조하십시오.

   c. 각 헤더 이름-값 페어의 경우 이름-값 페어를 `headerName:headerValue` 형식의 문자열로 변환합니다. 콜론 양쪽에 공백이 없도록 하고, `headerName` 및 `headerValue` 모두 시작 및 끝부분에서 공백을 잘라냅니다.

   ```
   x-amz-example1:value1,value2
   x-amz-example2:value3
   ```

   d. 마지막 문자열을 포함하여 변환된 각 문자열 뒤에 새 줄(`U+000A`)을 삽입합니다.

   e. 변환된 문자열 컬렉션을 헤더 이름에 따라 영문자순으로 정렬합니다.

2. 다음 항목이 포함된 서명할 문자열 값을 생성합니다.

- 줄 1: HTTP 메서드(POST), 뒤에서 줄바꿈합니다.
- 줄 2: 요청 URI(/), 뒤에서 줄바꿈합니다.
- 줄 3: 빈 문자열, 뒤에서 줄바꿈합니다.

> **Note**
>
> 일반적으로 여기에 쿼리 문자열이 나타나지만 Amazon SWF는 쿼리 문자열을 사용하지 않습니다.

- 줄 4-n: 1단계에서 계산한 정규화된 요청 헤더를 나타내는 문자열로, 뒤에서 줄바꿈합니다. 이 새 줄 때문에 HTTP 요청의 헤더와 본문 사이에 빈 줄이 생깁니다. 자세한 내용은 RFC 2616를 참조하십시오.
- 요청 본문 뒤에는 줄바꿈하지 않습니다.

3. 서명할 문자열 값의 SHA256 또는 SHA1 다이제스트를 계산합니다. 프로세스 전체에서 동일한 SHA 메서드를 사용합니다.
4. 이전 단계에서 생성된 값의 SHA256 또는 SHA1 다이제스트(사용한 방법에 따라 다름)를 사용하고 `GetSessionToken` API 작업을 사용하여 AWS Security Token Service의 임시 보안 액세스 키를 사용하여 HMAC-SHA를 계산하고 Base64-encode합니다.

   > **Note**
   >
   > Amazon SWF는 Base64로 인코딩된 HMAC-SHA 값 끝에 등호(=)가 있어야 합니다. Base64 인코딩 루틴에 등호가 추가되어 있지 않으면 값의 끝에 등호를 추가합니다.

   Amazon SWF 및 기타 AWS 서비스에서 임시 보안 자격 증명을 사용하는 방법에 대한 자세한 내용은 AWS IAM 사용 설명서의 IAM으로 작업하는 서비스를 참조하세요.
5. Amazon SWF에 대한 HTTP 요청의 `x-amzn-authorization` 헤더에 `Signature` 이름에 대한 값으로 결과 값을 배치합니다.
6. Amazon SWF는 요청을 확인하고 지정된 작업을 수행합니다.

범주별 Amazon SWF 작업 목록

이 단원에는 Amazon SWF 애플리케이션 프로그래밍 인터페이스(API)의 Amazon SWF 작업에 대한 참조 항목이 나열되어 있습니다. 참조 항목은 기능 범주별로 나열됩니다.

영문자순 작업 목록은 Amazon Simple Workflow Service API 참조를 참조하십시오.

주제

- 활동 관련 작업
- 결정자 관련 작업
- 워크플로 실행 관련 작업
- 관리 관련 작업
- 가시성 작업

활동 관련 작업

활동 작업자는 PollForActivityTask를 사용하여 새 활동 작업을 가져옵니다. 작업자는 Amazon SWF에서 활동 태스크를 수신한 후 태스크를 수행하고 성공한 경우 RespondActivityTaskCompleted, 실패한 경우 RespondActivityTaskFailed를 사용하여 응답합니다.

활동 작업자가 수행하는 작업은 다음과 같습니다.

- PollForActivityTask
- RespondActivityTaskCompleted
- RespondActivityTaskFailed
- RespondActivityTaskCanceled
- RecordActivityTaskHeartbeat

결정자 관련 작업

결정자는 PollForDecisionTask를 사용하여 의사 결정 작업을 가져옵니다. 결정자는 Amazon SWF에서 의사 결정 태스크를 수신한 후 워크플로 실행 기록을 검사하고 다음에 수행할 작업을 결정합니다. 그리고 RespondDecisionTaskCompleted를 호출하여 의사 결정 작업을 완료하고 0개 이상의 다음 의사 결정을 제공합니다.

결정자가 수행하는 작업은 다음과 같습니다.

- PollForDecisionTask
- RespondDecisionTaskCompleted

워크플로 실행 관련 작업

워크플로 실행에서 작동하는 작업은 다음과 같습니다.

- RequestCancelWorkflowExecution
- StartWorkflowExecution
- SignalWorkflowExecution
- TerminateWorkflowExecution

관리 관련 작업

Amazon SWF 콘솔에서 관리 태스크를 수행할 수 있지만 이 섹션의 활동을 사용하여 기능을 자동화하거나 고유한 관리 도구를 개발할 수 있습니다.

활동 관리

- RegisterActivityType
- DeprecateActivityType
- UndeprecateActivityType
- DeleteActivityType

워크플로우 관리

- RegisterWorkflowType
- DeprecateWorkflowType
- UndeprecateWorkflowType
- DeleteWorkflowType

도메인 관리

다음 작업을 사용하면 Amazon SWF 도메인을 등록 및 사용 중지할 수 있습니다.

- RegisterDomain
- DeprecateDomain
- UndeprecateDomain

도메인 관리 작업에 대한 예 및 자세한 내용은 Amazon SWF에 도메인 등록 단원을 참조하십시오.

워크플로 실행 관리

- RequestCancelWorkflowExecution
- TerminateWorkflowExecution

가시성 작업

Amazon SWF 콘솔에서 가시성 태스크를 수행할 수 있지만 이 섹션의 활동을 사용하여 고유한 콘솔 또는 관리 도구를 개발할 수 있습니다.

활동 가시성

- ListActivityTypes
- DescribeActivityType

워크플로우 가시성

- ListWorkflowTypes
- DescribeWorkflowType

워크플로우 실행 가시성

- DescribeWorkflowExecution
- ListOpenWorkflowExecutions
- ListClosedWorkflowExecutions
- CountOpenWorkflowExecutions
- CountClosedWorkflowExecutions
- GetWorkflowExecutionHistory

도메인 가시성

- ListDomains
- DescribeDomain

작업 목록 가시성

- CountPendingActivityTasks
- CountPendingDecisionTasks

Amazon SWF에 도메인 등록

워크플로, 활동 유형 및 워크플로 실행 자체는 모두 도메인으로 범위가 지정됩니다. 도메인은 유형, 실행 및 작업 목록을 동일한 계정 내의 다른 항목과 격리합니다.

를 사용하거나 Amazon SWF API의 RegisterDomain 작업을 AWS Management Console 사용하여 도메인을 등록할 수 있습니다. 다음 예에서는 API를 사용합니다.

```
https://swf.us-east-1.amazonaws.com
RegisterDomain
{
  "name" : "867530901",
  "description" : "music",
  "workflowExecutionRetentionPeriodInDays" : "60"
}
```

파라미터는 JavaScript Object Notation(JSON) 형식으로 지정됩니다. 여기서 보존 기간은 60일로 설정됩니다. 보존 기간 동안 워크플로 실행에 대한 모든 정보는 AWS Management Console 또는 Amazon SWF API를 사용하는 가시성 작업을 통해 확인할 수 있습니다.

도메인을 등록한 후에는 워크플로 유형과 해당 워크플로에서 사용하는 활동 유형을 등록해야 합니다. 등록된 도메인 이름이 워크플로 및 활동 유형을 등록하기 위한 필수 정보의 일부이기 때문에 도메인을 먼저 등록해야 합니다.

참고

Amazon Simple Workflow Service API 참조의 RegisterDomain

Amazon SWF에서 제한 시간 값 설정

주제

- 제한 시간 값에 대한 할당량

제한 시간 값에 대한 할당량

제한 시간 값은 항상 초 단위로 선언되며 모든 워크플로 또는 활동의 최대 실행 제한인 1년(31536000초)까지 임의의 초 단위로 설정할 수 있습니다. 특수 값인 NONE은 제한 시간 파라미터를 "제한 시간 없음" 또는 무한대로 설정하는 데 사용할 수 있지만 1년의 최대 제한은 계속해서 적용됩니다.

워크플로 실행 및 결정 작업 제한 시간

워크플로 유형을 등록할 때 워크플로 및 결정 작업에 대한 제한 시간 값을 설정할 수 있습니다. 예:

```
https://swf.us-east-1.amazonaws.com
RegisterWorkflowType
{
  "domain": "867530901",
  "name": "customerOrderWorkflow",
  "version": "1.0",
  "description": "Handle customer orders",
  "defaultTaskStartToCloseTimeout": "600",
  "defaultExecutionStartToCloseTimeout": "3600",
  "defaultTaskList": { "name": "mainTaskList" },
  "defaultChildPolicy": "TERMINATE"
}
```

워크플로 유형 등록은 defaultTaskStartToCloseTimeout을 600초(10분)로 설정하고 defaultExecutionStartToCloseTimeout은 3600초(1시간)로 설정합니다.

워크플로 유형 등록에 대한 자세한 내용은 Amazon SWF에 워크플로 유형 등록 및 Amazon Simple Workflow Service API 참조의 RegisterWorkflowType을 참조하십시오.

executionStartToCloseTimeout 를 지정해 defaultExecutionStartToCloseTimeout에 대해 설정된 값을 재정의할 수 있습니다.

활동 작업의 제한 시간

활동 유형을 등록할 때 활동 작업에 대한 제한 시간 값을 설정할 수 있습니다. 예:

```
https://swf.us-east-1.amazonaws.com
RegisterActivityType
{
  "domain": "867530901",
  "name": "activityVerify",
  "version": "1.0",
  "description": "Verify the customer credit",
  "defaultTaskStartToCloseTimeout": "600",
  "defaultTaskHeartbeatTimeout": "120",
  "defaultTaskList": { "name": "mainTaskList" },
  "defaultTaskScheduleToStartTimeout": "1800",
  "defaultTaskScheduleToCloseTimeout": "5400"
}
```

이 활동 유형 등록은 defaultTaskStartToCloseTimeout을 600초(10분)으로, defaultTaskHeartbeatTimeout은 120초(2분)으로, defaultTaskScheduleToStartTimeout은 1800초(30분)로, defaultTaskScheduleToCloseTimeout은 5400초(1.5시간)로 설정합니다.

활동 유형 등록에 대한 자세한 내용은 Amazon SWF에 활동 유형 등록 및 Amazon Simple Workflow Service API 참조의 RegisterActivityType을 참조하십시오.

활동 작업을 예약할 때 taskStartToCloseTimeout를 지정하여 defaultTaskStartToCloseTimeout에 대해 설정된 값을 재정의할 수 있습니다.

참고

Amazon SWF 제한 시간 유형

Amazon SWF에 워크플로 유형 등록

이 단원에서 설명하는 예에서는 Amazon SWF API를 사용하여 워크플로 유형을 등록합니다. 등록 중 지정한 이름 및 버전이 워크플로 유형에 대한 고유한 식별자를 구성합니다. 지정된 도메인은 RegisterDomain API 작업을 사용해 이미 등록되어 있어야 합니다.

다음 예에서 시간 제한 파라미터는 초 단위로 지정된 기간 값입니다. defaultTaskStartToCloseTimeout 파라미터의 경우 제한 시간이 없음을 나타내는 기간 지정자 NONE을 사용할 수 있습니다. 그러나 defaultExecutionStartToCloseTimeout에 대해서는 NONE 값을 지정할 수 없습니다. 워크플로 실행을 실행할 수 있는 최대 시

간 제한은 1년입니다. 이 제한을 초과하면 항상 워크플로 실행이 시간 초과됩니다. `defaultExecutionStartToCloseTimeout`에 대해 1년보다 긴 값을 지정하면 등록에 실패합니다.

```
https://swf.us-east-1.amazonaws.com
RegisterWorkflowType
{
  "domain" : "867530901",
  "name" : "customerOrderWorkflow",
  "version" : "1.0",
  "description" : "Handle customer orders",
  "defaultTaskStartToCloseTimeout" : "600",
  "defaultExecutionStartToCloseTimeout" : "3600",
  "defaultTaskList" : { "name": "mainTaskList" },
  "defaultChildPolicy" : "TERMINATE"
}
```

참고

Amazon Simple Workflow Service API 참조의 RegisterWorkflowType

Amazon SWF에 활동 유형 등록

다음 예에서는 Amazon SWF API를 사용하여 활동 유형을 등록합니다. 등록 중 지정한 이름 및 버전이 도메인 내에서 활동 유형에 대한 고유한 식별자를 구성합니다. 지정된 도메인은 `RegisterDomain` 작업을 사용해 이미 등록되어 있어야 합니다.

이 예에서 시간 제한 파라미터는 초 단위로 지정된 기간 값입니다. 제한 시간이 없음을 나타내는 기간 지정자 `NONE`을 사용할 수 있습니다.

```
https://swf.us-east-1.amazonaws.com
RegisterActivityType
{
  "domain" : "867530901",
  "name" : "activityVerify",
  "version" : "1.0",
  "description" : "Verify the customer credit",
  "defaultTaskStartToCloseTimeout" : "600",
  "defaultTaskHeartbeatTimeout" : "120",
  "defaultTaskList" : { "name" : "mainTaskList" },
  "defaultTaskScheduleToStartTimeout" : "1800",
```

```
  "defaultTaskScheduleToCloseTimeout" : "5400"
}
```

참고

Amazon Simple Workflow Service API 참조의 RegisterActivityType

AWS Lambda Amazon SWF의 태스크

주제

- 정보 AWS Lambda
- Lambda 작업 사용의 이점 및 제한 사항
- 워크플로에서 Lambda 작업 사용

정보 AWS Lambda

AWS Lambda 는 사용자 지정 코드 또는 Amazon S3, DynamoDB, Amazon Kinesis, Amazon SNS, Amazon Cognito와 같은 다양한 서비스에서 생성된 이벤트에 대한 응답으로 코드를 실행하는 완전관리형 컴퓨팅 AWS 서비스입니다. Lambda에 대한 자세한 내용은 AWS Lambda 개발자 안내서를 참조하세요.

Amazon Simple Workflow Service는 Lambda 작업을 제공하므로 기존 Amazon SWF 활동 대신 또는 이러한 활동과 함께 Lambda 함수를 실행할 수 있습니다.

> **Important**
>
> Amazon SWF가 사용자를 대신하여 실행하는 Lambda 실행(요청)에 대해 AWS 계정에 요금이 부과됩니다. Lambda 요금에 대한 자세한 내용은 https://aws.amazon.com/lambda/pricing/ 참조하십시오.

Lambda 작업 사용의 이점 및 제한 사항

일반적인 Amazon SWF 활동 대신 Lambda 작업을 사용하면 다음과 같은 여러 가지 이점이 있습니다.

- Lambda 작업은 Amazon SWF 활동 유형처럼 등록하거나 버전을 관리할 필요가 없습니다.

- 워크플로에 이미 정의해 둔 기존 Lambda 함수는 어느 것이나 사용할 수 있습니다.
- Lambda 함수는 Amazon SWF에서 직접 호출하기 때문에 일반적인 활동에 대해 수행해야 하는 것처럼 작업을 실행할 작업자 프로그램을 구현할 필요가 없습니다.
- Lambda에서는 함수 실행을 추적 및 분석할 수 있도록 측정치 및 로그를 제공합니다.

또한 Lambda 작업과 관련해서는 반드시 알고 있어야 하는 여러 가지 제한 사항이 있습니다.

- Lambda 태스크는 Lambda에 대한 지원을 제공하는 AWS 리전에서만 실행할 수 있습니다. 현재 Lambda를 지원하는 리전에 대한 자세한 내용은 Amazon Web Services 일반 참조의 Lambda 리전 및 엔드포인트를 참조하십시오.
- Lambda 작업은 현재 기본 SWF HTTP API 및 Java AWS Flow Framework 용 에서만 지원됩니다. 현재 for AWS Flow Framework Ruby에서는 Lambda 작업을 지원하지 않습니다.

워크플로에서 Lambda 작업 사용

Amazon SWF 워크플로에서 Lambda 작업을 사용하려면 다음을 수행해야 합니다.

1. IAM 역할을 설정하고 Amazon SWF에 Lambda 함수 간접 호출 권한을 제공합니다.
2. 워크플로에 IAM 역할을 연결합니다.
3. 워크플로 실행 중 Lambda 함수를 호출합니다.

IAM 역할 설정

Amazon SWF에서 Lambda 함수를 간접적으로 호출하려면 Amazon SWF에서 Lambda에 대한 액세스를 제공하는 IAM 역할을 제공해야 합니다. 다음 작업 중 하나를 수행할 수 있습니다.

- 사전 정의된 역할인 AWSLambdaRole을 선택해 워크플로에 계정과 연결된 모든 Lambda 함수를 간접적으로 호출할 수 있는 권한을 부여합니다.
- 자체 정책 및 연결된 역할을 정의해 Amazon 리소스 이름(ARN)으로 지정된 특정 Lambda 함수를 간접적으로 호출하는 워크플로 권한을 부여합니다.

IAM 역할에 대한 권한 제한

리소스 신뢰 정책의 `SourceArn` 및 `SourceAccount` 컨텍스트 키를 사용하여 Amazon SWF에 제공하는 IAM 역할에 대한 권한을 제한할 수 있습니다. 이러한 키는 지정된 도메인 ARN에 속하는 Amazon Simple Workflow Service 실행에서만 사용하도록 IAM 정책의 사용을 제한합니다. 두 글로벌 조건 컨텍

스트 키를 모두 사용하는 경우 aws:SourceAccount 값과 aws:SourceArn 값에서 참조되는 계정은 동일한 정책 문에서 사용할 경우 동일한 계정 ID를 사용해야 합니다.

다음 예제에서 SourceArn 컨텍스트 키는 계정에 속한 Amazon Simple Workflow Service 실행에서만 IAM 서비스 역할을 사용하도록 제한합니다someDomain123456789012.

- 문 1

  보안 주체: "Service": "swf.amazonaws.com"

  작업: sts:AssumeRole

```
"Condition": {
   "ArnLike": {
     "aws:SourceArn": "arn:aws:swf:*:123456789012:/domain/someDomain"
   }
}
```

다음 예제에서 SourceAccount 컨텍스트 키는 계정의 Amazon Simple Workflow Service 실행에서만 IAM 서비스 역할을 사용하도록 제한합니다123456789012.

```
"Condition": {
   "StringLike": {
     "aws:SourceAccount": "123456789012"
   }
}
```

Amazon SWF에 Lambda 역할을 간접 호출하기 위한 액세스 권한 제공

사전 정의된 역할인 AWSLambdaRole을 사용해 Amazon SWF 워크플로에 계정과 연결된 모든 Lambda 함수를 간접적으로 호출할 수 있는 권한을 부여합니다.

AWSLambdaRole을 사용해 Amazon SWF에 Lambda 함수를 간접적으로 호출할 수 있는 액세스 권한을 부여하려면

1. Amazon IAM 콘솔을 엽니다.
2. [Roles]를 선택한 다음 [Create New Role]을 선택합니다.
3. swf-lambda와 같이 역할에 이름을 지정한 다음 [Next Step]을 선택합니다.
4. AWS 서비스 역할에서 Amazon SWF를 선택하고 다음 단계를 선택합니다.

5. 정책 연결 화면의 목록에서 AWSLambdaRole을 선택합니다.
6. 역할을 검토한 후 [Next Step]을 선택하고 [Create Role]을 선택합니다.

특정 Lambda 함수를 간접 호출할 액세스 권한을 제공하는 IAM 역할 정의

워크플로에서 특정 Lambda 함수를 간접적으로 호출하는 액세스 권한을 제공하려는 경우 자체 IAM 정책을 정의해야 합니다.

특정 Lambda 함수에 대한 액세스 권한을 부여하는 IAM 정책을 생성하려면

1. Amazon IAM 콘솔을 엽니다.
2. [Policies]를 선택한 다음 [Create Policy]를 선택합니다.
3. AWS 관리형 정책 복사를 선택하고 목록에서 AWSLambdaRole을 선택합니다. 정책이 자동으로 생성됩니다. 경우에 따라 필요에 맞춰 정책 이름 및 설명을 편집합니다.
4. 정책 문서의 리소스 필드에 Lambda 함수의 ARN을 추가합니다. 예:

   - 리소스: `arn:aws:lambda:us-east-1:111122223333:function:hello_lambda_function`

   > **Note**
   >
   > IAM 역할에서 리소스를 지정하는 방법에 대한 전체 설명은 IAM 사용의 IAM 정책 개요를 참조하십시오.

5. [Create policy]를 선택하여 정책 생성을 마칩니다.

그런 다음 새 IAM 역할을 생성할 때 이 정책을 선택하고 해당 역할을 사용해 Amazon SWF 워크플로에 간접 호출 액세스 권한을 부여합니다. 이 절차는 AWSLambdaRole 정책을 사용해 역할을 생성하는 것과 매우 유사합니다. 대신 역할을 생성할 때 고유한 정책을 선택합니다.

Lambda 정책을 사용하여 Amazon SWF 역할을 생성하려면

1. Amazon IAM 콘솔을 엽니다.
2. [Roles]를 선택한 다음 [Create New Role]을 선택합니다.
3. `swf-lambda-function`와 같이 역할에 이름을 지정한 다음 [Next Step]을 선택합니다.
4. AWS 서비스 역할에서 Amazon SWF를 선택하고 다음 단계를 선택합니다.

5. 정책 연결 화면의 목록에서 Lambda 함수 관련 정책을 선택합니다.
6. 역할을 검토한 후 [Next Step]을 선택하고 [Create Role]을 선택합니다.

워크플로에 IAM 역할 연결

IAM 역할을 정의하면 해당 역할을 워크플로에 연결해야 합니다. 그러면 워크플로에서는 이 역할을 사용해 Amazon SWF에 액세스 권한을 제공한 Lambda 함수를 직접적으로 호출합니다.

역할을 워크플로에 연결할 수 있는 지점에는 다음 2개가 있습니다.

- 워크플로 유형 등록 중. 해당 워크플로 유형을 실행할 때마다 이 역할을 기본 Lambda 역할로 사용할 수 있습니다.
- 워크플로 실행 시작 시. 역할이 워크플로 실행 중(및 전체 실행 중)에만 사용됩니다.

워크플로 유형에 대해 기본 Lambda 역할을 제공하려면

- RegisterWorkflowType을 호출하는 경우 defaultLambdaRole 필드를 정의한 역할의 ARN으로 설정합니다.

워크플로 실행 중 사용할 Lambda 역할을 제공하려면

- StartWorkflowExecution을 호출하는 경우 lambdaRole 필드를 정의한 역할의 ARN으로 설정합니다.

> **Note**
>
> RegisterWorkflowType 또는 StartWorkflowExecution을 호출하는 계정에 주어진 역할을 사용할 권한이 없는 경우 호출에 실패하고 OperationNotPermittedFault가 발생합니다.

Amazon SWF 워크플로에서 Lambda 함수 직접 호출

ScheduleLambdaFunctionDecisionAttributes 데이터 유형을 사용해 워크플로 실행 중 호출할 Lambda 함수를 식별합니다.

RespondDecisionTaskCompleted 호출 중 결정 목록에 ScheduleLambdaFunctionDecisionAttributes를 제공합니다. 예:

```
{
  "decisions": [{
    "ScheduleLambdaFunctionDecisionAttributes": {
      "id": "lambdaTaskId",
      "name": "myLambdaFunctionName",
      "input": "inputToLambdaFunction",
      "startToCloseTimeout": "30"
    },
  }],
}
```

다음 파라미터를 설정합니다.

- Lambda 작업 식별자인 id. 1~256자의 문자열로, :(콜론), /(슬래시), |(세로 막대), 제어 문자(\u0000 ~ \u001f 및 \u007f ~ \u009f), 리터럴 문자열 `arn`은 사용할 수 없습니다.
- name: Lambda 함수의 이름. Amazon SWF 워크플로는 Lambda 함수를 직접적으로 호출할 액세스 권한을 제공하는 IAM 역할과 함께 제공되어야 합니다. 제공된 이름은 Lambda 호출 작업에서처럼 FunctionName 파라미터에 대한 제약을 따라야 합니다.
- input: 함수에 대한 선택적 입력 데이터. 설정하면 이 파라미터는 Lambda 호출 작업에서처럼 ClientContext 파라미터에 대한 제약을 따라야 합니다.
- startToCloseTimeout: 작업에 실패하고 시간 초과 예외가 발생하기 전에 함수 실행에 걸릴 수 있는 선택적 최대 기간(초). 값 NONE을 사용하면 기간을 무제한으로 지정할 수 있습니다.

자세한 내용은 AWS Lambda 작업 구현을 참조하세요.

Amazon SWF에서 활동 작업자 개발

활동 작업자는 하나 이상의 작업 유형을 구현합니다. 활동 작업자는 Amazon SWF와 통신하여 활동 작업을 수신하고 수행합니다. 동일한 활동 유형의 활동 작업을 수행하는 활동 작업자가 여러 개 있을 수 있습니다.

결정자가 활동 작업을 예약하면 활동 작업자는 Amazon SWF에서 그 활동 작업을 사용할 수 있습니다. 결정자는 활동 작업을 예약할 때 활동 작업자가 활동 작업을 수행하는 데 필요한 데이터(사용자가 판단)를 제공합니다. Amazon SWF는 활동 작업에 이러한 데이터를 삽입하여 활동 작업자에게 보냅니다.

활동 작업자는 사용자가 관리합니다. 활동 작업자는 모든 언어로 작성이 가능하고, API를 통해 Amazon SWF와 통신할 수 있는 한 어디에서나 작업자를 실행할 수 있습니다. 활동 작업을 수행하는

데 필요한 모든 정보를 Amazon SWF가 제공하기 때문에 모든 활동 작업자는 상태 비저장일 수 있습니다. 상태 비저장 방식에서는 워크플로의 확장 가능성이 매우 커지므로, 증가한 필요 용량을 처리하려면 활동 작업자를 더 추가하기만 하면 됩니다.

이 단원에서는 활동 작업자를 구현하는 방법을 설명합니다. 활동 작업자는 반복적으로 다음 작업을 수행해야 합니다.

1. 활동 작업에 대해 Amazon SWF를 폴링합니다.
2. 작업 수행을 시작합니다.
3. 작업이 오래 실행되는 경우 Amazon SWF에 정기적으로 하트비트를 보고합니다.
4. Amazon SWF에 작업이 완료 또는 실패했는지 보고하고 결과를 반환합니다.

주제

활동 작업 폴링

활동 작업을 수행하기 위해 각 활동 작업자는 `PollForActivityTask` 작업을 정기적으로 호출해 Amazon SWF를 폴링해야 합니다.

다음 예에서 활동 작업자 `ChargeCreditCardWorker01`은 작업 목록 `ChargeCreditCard-v0.1`에서 작업을 폴링합니다. 사용 가능한 활동 작업이 없으면 60초 후 Amazon SWF가 빈 응답을 다시 보냅니다. 빈 응답은 `Task` 구조로, 여기서 `taskToken`의 값이 빈 문자열입니다.

```
https://swf.us-east-1.amazonaws.com
PollForActivityTask
{
  "domain" : "867530901",
  "taskList" : { "name": "ChargeCreditCard-v0.1" },
  "identity" : "ChargeCreditCardWorker01"
}
```

활동 작업을 사용할 수 있게 되면 Amazon SWF는 작업을 활동 작업자에게 반환합니다. 이 작업에는 결정자가 활동을 예약할 때 지정한 데이터가 들어 있습니다.

활동 작업자가 이 활동 작업을 받으면 작업을 수행할 준비가 된 것입니다. 다음 단원에서는 활동 작업 수행에 대한 정보를 제공합니다.

활동 작업 수행

활동 작업을 수신하면 활동 작업자가 작업을 수행할 준비가 된 것입니다.

활동 작업을 수행하려면

1. 작업의 입력 필드에 있는 내용을 해석하도록 활동 작업자를 프로그래밍합니다. 이 필드에는 작업 예약 시 결정자가 지정한 데이터가 들어 있습니다.
2. 데이터 처리 및 로직 실행을 시작하도록 활동 작업자를 프로그래밍합니다.

다음 섹션에서는 오래 실행되는 활동에 대한 상태 업데이트를 Amazon SWF에 제공하도록 활동 작업자를 프로그래밍하는 방법에 대해 설명합니다.

활동 작업 하트비트 보고

활동 유형에 하트비트 제한 시간이 등록되면 활동 작업자는 하트비트 제한 시간이 초과되기 전에 하트비트를 기록해야 합니다. 활동 작업이 제한 시간 안에 하트비트를 제공하지 못하면 작업 시간이 초과되고, Amazon SWF는 작업을 닫고 새 결정 작업을 예약해 결정자에게 시간 초과 사실을 알립니다. 그러면 결정자는 활동 작업을 다시 예약하거나 다른 조치를 취할 수 있습니다.

시간 초과 후 활동 작업자가 `RespondActivityTaskCompleted` 호출 등을 통해 Amazon SWF에 접속하려고 하면 Amazon SWF는 `UnknownResource` 장애를 반환합니다.

이 단원에서는 활동 하트비트를 제공하는 방법을 설명합니다.

활동 작업 하트비트를 기록하려면 `RecordActivityTaskHeartbeat` 작업을 호출하도록 활동 작업자를 프로그래밍합니다. 또한 이 작업은 애플리케이션에 대한 작업 진행 상황을 정량화하기 위해 자유 형식의 데이터를 저장하는 데 사용할 수 있는 문자열 필드를 제공합니다.

이 예에서 활동 작업자는 Amazon SWF에 하트비트를 보고하고 세부 정보 필드를 사용해 활동 작업이 40% 완료되었음을 보고합니다. 하트비트를 보고하려면 활동 작업자가 활동 작업의 작업 토큰을 지정해야 합니다.

```
https://swf.us-east-1.amazonaws.com
```

```
RecordActivityTaskHeartbeat
{
  "taskToken" : "12342e17-80f6-FAKE-TASK-TOKEN32f0223",
  "details" : "40"
}
```

이 작업은 워크플로 실행 내역에 이벤트를 생성하지 않지만 작업 시간이 초과되면 워크플로 실행 내역에 ActivityTaskTimedOut 이벤트가 포함됩니다. 이 이벤트에는 활동 작업자가 생성한 마지막 하트비트의 정보가 들어 있습니다.

활동 작업 완료 또는 실패

작업 실행 후 활동 작업자는 활동 작업을 완료했는지 아니면 작업에 실패했는지 보고해야 합니다.

활동 작업 완료

활동 작업을 완료하려면 활동 작업을 성공적으로 완료한 후 작업 토큰을 지정하는 RespondActivityTaskCompleted 작업을 호출하도록 활동 작업자를 프로그래밍해야 합니다.

이 예에서 활동 작업자는 작업이 성공적으로 완료되었음을 나타냅니다.

```
https://swf.us-east-1.amazonaws.com
RespondActivityTaskCompleted
{
  "taskToken": "12342e17-80f6-FAKE-TASK-TOKEN32f0223",
  "results": "40"
}
```

활동이 완료되면 Amazon SWF는 활동이 연결된 워크플로 실행에 대해 새 결정 작업을 예약합니다.

수중에 있는 작업을 완료하면 다른 활동 작업을 폴링하도록 활동 작업자를 프로그래밍합니다. 이렇게 하면 활동 작업자가 작업을 계속해서 폴링 및 완료하는 루프가 생성됩니다.

StartToCloseTimeout 기간 이내에 활동이 응답하지 않거나 ScheduleToCloseTimeout이 만료되면 Amazon SWF는 활동 작업 기간을 초과하고 결정 작업을 예약합니다. 따라서 결정자가 작업 다시 예약 등과 같은 적절한 조치를 취할 수 있습니다.

예를 들어, Amazon EC2 인스턴스가 활동 작업을 실행 중인데 작업이 완료되기 전에 인스턴스가 실패하면 결정자는 워크플로 실행 내역에 시간 초과 이벤트를 수신합니다. 활동 작업이 하트비트를 사용하는 경우 결정자는 Amazon EC2 인스턴스 실패 후 작업이 다음 하트비트 전달에 실패할 때 이 이벤트를

수신합니다. 그렇지 않은 경우 결정자는 전체 제한 시간 값 중 하나에 도달하기 전에 활동 작업을 완료하는 데 실패한 경우 이 이벤트를 수신합니다. 실패한 작업을 다시 할당하거나 다른 조치를 취하는 것은 결정자의 몫입니다.

활동 작업 실패

활동 작업자가 어떤 이유로 활동 작업을 수행할 수 없는데 Amazon SWF와 계속해서 통신할 수 있는 경우 작업에 실패하도록 활성 작업자를 프로그래밍할 수 있습니다.

활동 작업에 실패하도록 활동 작업자를 프로그래밍하려면 작업의 토큰을 지정하는 `RespondActivityTaskFailed` 작업을 호출하도록 활동 작업자를 프로그래밍합니다.

```
https://swf.us-east-1.amazonaws.com
RespondActivityTaskFailed
{
  "taskToken" : "12342e17-80f6-FAKE-TASK-TOKEN32f0223",
  "reason" : "CC-Invalid",
  "details" : "Credit Card Number Checksum Failed"
}
```

개발자는 사유 및 세부 정보 필드에 저장할 값을 정의합니다. 이러한 값은 자유 형식 문자열인데, 애플리케이션에서 지원하는 모든 오류 코드 규칙을 사용할 수 있습니다. Amazon SWF는 이러한 값을 처리하지 않습니다. 하지만 Amazon SWF는 콘솔에 이러한 값을 표시할 수는 있습니다.

활동 작업에 실패하면 Amazon SWF는 활동 작업이 연결된 워크플로 실행을 위한 결정 작업을 예약해 결정자에게 실패 사실을 알립니다. 실패의 특성에 따라 실패한 활동을 활동 다시 예약 또는 워크플로 실행 실패 등으로 처리하도록 결정자를 프로그래밍합니다.

활동 작업자 시작

활동 작업자를 시작하려면 활동 작업자 플랫폼에서 사용할 수 있는 실행 파일로 로직을 패키징합니다. 예를 들어, Linux 및 Windows 서버 양쪽 모두에서 실행되는 Java 실행 파일로 활동 코드를 패키징할 수 있습니다.

시작된 작업자는 작업을 폴링하기 시작합니다. 그러나 결정자가 활동 작업을 예약할 때까지는 작업 없이 폴링 시간이 초과되고, 그래도 작업자는 폴링을 계속합니다.

폴은 아웃바운드 요청이기 때문에 Amazon SWF 엔드포인트에 액세스할 수 있는 모든 네트워크에서 활동 작업자를 실행할 수 있습니다.

원하는 수만큼의 활동 작업자를 시작할 수 있습니다. 결정자가 활동 작업을 예약하면 Amazon SWF는 활동 작업을 폴링 활동 작업자에게 자동으로 분배합니다.

Amazon SWF에서 결정자 개발

결정자는 워크플로 실행 중 실행되는 워크플로 유형의 조정 로직을 구현하는 것입니다. 한 가지 워크플로 유형에 대해 여러 결정자를 실행할 수 있습니다.

워크플로 실행의 실행 상태는 워크플로 기록에 저장되므로 결정자는 상태 비저장 상태가 될 수 있습니다. Amazon SWF는 워크플로 실행 기록을 유지 관리하고 이를 각 의사 결정 작업과 함께 결정자에게 제공합니다. 따라서 필요에 따라 결정자를 동적으로 추가 및 제거할 수 있어 워크플로 처리의 확장 가능성이 매우 커집니다. 시스템에 대한 로드가 증가하면 결정자를 추가해 늘어난 용량을 처리할 수 있습니다. 그러나 워크플로 실행 중 결정 작업은 하나만 열려 있을 수 있습니다.

워크플로 실행에 대한 상태가 변경될 때마다 Amazon SWF는 결정 작업을 예약합니다. 결정자는 결정 작업을 수신할 때마다 다음 작업을 수행합니다.

- 결정 작업과 함께 제공된 워크플로 실행 내역 해석
- 워크플로 실행 내역을 기반으로 조정 로직 적용 및 다음으로 수행할 작업 결정. 각 결정은 결정 구조로 표현됩니다.
- 결정 작업 완료 및 Amazon SWF에 결정 목록 제공

이 단원에서는 결정자를 개발하는 방법을 설명하고, 여기에는 다음 작업이 관련됩니다.

- 결정 작업을 폴링하도록 결정자 프로그래밍
- 워크플로 실행 내역을 해석하고 결정을 하도록 결정자 프로그래밍
- 결정 작업에 대해 응답하도록 결정자 프로그래밍

이 단원의 예에서는 전자 상거래의 예 워크플로에 대한 결정자를 프로그래밍할 수 있는 방법을 보여줍니다.

원하는 언어로 결정자를 구현하고, 서비스 API를 통해 Amazon SWF와 통신할 수 있는 한 어디에서나 이를 실행할 수 있습니다.

주제

- 결정 작업 폴링
- 조정 로직 적용
- 결정에 응답
- 워크플로 실행 닫기
- 결정자 시작

조정 로직 정의

결정자를 개발할 때 해야 할 첫 번째 작업은 조정 로직을 정의하는 것입니다. 전자 상거래의 예에서 이전 활동 완료 후 수행될 각 활동을 예약하는 조정 로직은 다음과 같을 수 있습니다.

```
IF lastEvent = "StartWorkflowInstance"
 addToDecisions ScheduleVerifyOrderActivity

ELSIF lastEvent = "CompleteVerifyOrderActivity"
 addToDecisions ScheduleChargeCreditCardActivity

ELSIF lastEvent = "CompleteChargeCreditCardActivity"
 addToDecisions ScheduleCompleteShipOrderActivity

ELSIF lastEvent = "CompleteShipOrderActivity"
 addToDecisions ScheduleRecordOrderCompletion

ELSIF lastEvent = "CompleteRecordOrderCompletion"
 addToDecisions CloseWorkflow

ENDIF
```

결정자는 워크플로 실행 내역에 조정 로직을 적용해 `RespondDecisionTaskCompleted` 작업을 사용해 결정 작업 완료 시 결정 목록을 생성합니다.

결정 작업 폴링

각 결정자는 결정 작업을 폴링합니다. 결정 작업에는 결정자가 활동 작업 예약 등과 같은 결정을 생성하는 데 사용하는 정보가 들어 있습니다. 결정 작업을 폴링하기 위해 결정자는 `PollForDecisionTask` 작업을 사용합니다.

이 예에서 결정자는 결정 작업을 폴링해 `customerOrderWorkflow-0.1` 작업 목록을 지정합니다.

```
https://swf.us-east-1.amazonaws.com
PollForDecisionTask
{
  "domain": "867530901",
  "taskList": {"name": "customerOrderWorkflow-v0.1"},
  "identity": "Decider01",
  "maximumPageSize": 50,
  "reverseOrder": true
}
```

지정된 작업 목록에서 결정 작업을 사용할 수 있으면 Amazon SWF에서 해당 작업을 즉시 반환합니다. 사용 가능한 결정 작업이 없는 경우 Amazon SWF는 최대 60초 동안 연결을 열린 상태로 유지하고 작업을 사용할 수 있게 되면 즉시 반환합니다. 사용 가능한 작업이 없는 경우 Amazon SWF는 빈 응답을 반환합니다. 빈 응답은 Task 구조로, 여기서 taskToken의 값이 빈 문자열입니다. 결정자가 빈 응답을 수신하면 다른 작업을 폴링하도록 결정자를 프로그래밍해야 합니다.

결정 작업을 사용할 수 있는 경우 Amazon SWF는 결정 작업과 워크플로 실행 내역의 페이지를 매긴 보기가 포함된 응답을 반환합니다.

이 예에서 최신 이벤트 유형은 워크플로 실행이 시작되었으며 입력 요소에 첫 번째 작업을 수행하는 데 필요한 정보가 포함되어 있음을 나타냅니다.

```
{
  "events": [
    {
      "decisionTaskStartedEventAttributes": {
        "identity": "Decider01",
        "scheduledEventId": 2
      },
      "eventId": 3,
      "eventTimestamp": 1326593394.566,
      "eventType": "DecisionTaskStarted"
    }, {
      "decisionTaskScheduledEventAttributes": {
        "startToCloseTimeout": "600",
        "taskList": { "name": "specialTaskList" }
      },
      "eventId": 2,
      "eventTimestamp": 1326592619.474,
      "eventType": "DecisionTaskScheduled"
    }, {
      "eventId": 1,
```

```
      "eventTimestamp": 1326592619.474,
      "eventType": "WorkflowExecutionStarted",
      "workflowExecutionStartedEventAttributes": {
        "childPolicy" : "TERMINATE",
        "executionStartToCloseTimeout" : "3600",
        "input" : "data-used-decider-for-first-task",
        "parentInitiatedEventId": 0,
        "tagList" : ["music purchase", "digital", "ricoh-the-dog"],
        "taskList": { "name": "specialTaskList" },
        "taskStartToCloseTimeout": "600",
        "workflowType": {
          "name": "customerOrderWorkflow",
          "version": "1.0"
        }
      }
    }
  ],
  ...
}
```

워크플로 실행 내역을 수신하면 결정자는 내역을 해석하고 조정 로직에 따라 결정을 내립니다.

워크플로 실행 하나의 워크플로 내역 이벤트 수가 많은 경우, 결과가 여러 페이지로 나뉘어 반환될 수 있습니다. 후속 페이지를 검색하려면 초기 호출에서 반환된 nextPageToken을 사용하여 추가로 PollForDecisionTask을 직접적으로 호출합니다. 참고: nextPageToken을 사용하여 GetWorkflowExecutionHistory을 직접적으로 호출하지 마십시오. 그 대신 PollForDecisionTask를 다시 호출하십시오.

조정 로직 적용

결정 작업을 받은 결정자가 워크플로 실행 내역을 해석해 지금까지 어떤 일이 발생했는지 확인하도록 결정자를 프로그래밍합니다. 결정자는 이를 바탕으로 결정 목록을 생성해야 합니다.

전자 상거래의 예에서는 워크플로 내역의 마지막 이벤트만 고려하므로 다음 로직을 정의합니다.

```
IF lastEvent = "StartWorkflowInstance"
 addToDecisions ScheduleVerifyOrderActivity

ELSIF lastEvent = "CompleteVerifyOrderActivity"
 addToDecisions ScheduleChargeCreditCardActivity

ELSIF lastEvent = "CompleteChargeCreditCardActivity"
```

```
 addToDecisions ScheduleCompleteShipOrderActivity

ELSIF lastEvent = "CompleteShipOrderActivity"
 addToDecisions ScheduleRecordOrderCompletion

ELSIF lastEvent = "CompleteRecordOrderCompletion"
 addToDecisions CloseWorkflow

ENDIF
```

lastEvent가 CompleteVerifyOrderActivity이면 결정 목록에 ScheduleChargeCreditCardActivity 활동을 추가합니다.

결정자가 어떤 결정을 내릴지 결정하면 Amazon SWF에 적절한 결정으로 응답할 수 있습니다.

결정에 응답

워크플로 내역을 해석하고 결정 목록을 생성하면 결정자는 해당 결정으로 다시 Amazon SWF에 응답할 준비가 된 것입니다.

프로그램 실행 내역에서 필요한 데이터를 추출한 다음 워크플로에 적절한 다음 작업을 지정하도록 결정자를 프로그래밍합니다. 결정자는 RespondDecisionTaskCompleted 작업을 사용하여 이러한 결정을 Amazon SWF로 다시 전송합니다. 사용 가능한 결정 유형 목록은 Amazon Simple Workflow Service API 참조를 참조하십시오.

이 전자 상거래의 예에서 결정자가 자신이 생성한 결정 세트로 응답할 때 워크플로 실행 내역의 신용카드 입력을 포함합니다. 그러면 활동 작업자는 활동 작업을 수행하는 데 필요한 정보를 갖게 됩니다.

워크플로 실행의 모든 활동이 완료되면 결정자는 워크플로 실행을 닫습니다.

```
https://swf.us-east-1.amazonaws.com
RespondDecisionTaskCompleted
{
  "taskToken" : "12342e17-80f6-FAKE-TASK-TOKEN32f0223",
  "decisions" : [
    {
      "decisionType" :"ScheduleActivityTask",
      "scheduleActivityTaskDecisionAttributes" : {
        "control" :"OPTIONAL_DATA_FOR_DECIDER",
        "activityType" : {
          "name" :"ScheduleChargeCreditCardActivity",
```

```
          "version" :"1.1"
        },
        "activityId" :"3e2e6e55-e7c4-beef-feed-aa815722b7be",
        "scheduleToCloseTimeout" :"360",
        "taskList" : { "name" :"CC_TASKS" },
        "scheduleToStartTimeout" :"60",
        "startToCloseTimeout" :"300",
        "heartbeatTimeout" :"60",
        "input" : "4321-0001-0002-1234: 0212 : 234"
      }
    }
  ]
}
```

워크플로 실행 닫기

결정자가 비즈니스 프로세스가 완료되었음을 확인하면 즉, 수행할 활동이 더 이상 없음을 확인하면 결정자는 워크플로 실행을 닫는 결정을 생성합니다.

워크플로 실행을 닫으려면 워크플로 내역의 이벤트를 해석해 지금까지 실행에서 발생한 일을 확인해 워크플로 실행을 닫아야 하는지 살펴보도록 결정자를 프로그래밍합니다.

워크플로가 성공적으로 완료되면 `CompleteWorkflowExecution` 결정으로 `RespondDecisionTaskCompleted`를 호출해 워크플로 실행을 닫습니다. 또는 `FailWorkflowExecution` 결정을 사용해 잘못된 실행을 실패로 처리할 수 있습니다.

이 전자 상거래의 예에서 결정자는 내역을 검토하고 조정 로직을 기반으로 결정 목록에 워크플로 실행을 닫는 결정을 추가하고, 워크플로 닫기 결정으로 `RespondDecisionTaskCompleted` 작업을 시작합니다.

> **Note**
>
> 워크플로 실행 닫기에 실패하는 몇 가지 경우가 있습니다. 예를 들어 결정자가 워크플로 실행을 닫는 중 신호를 수신하면 닫기 결정에 실패합니다. 이러한 가능성을 처리하기 위해 결정자는 계속해서 결정 작업을 폴링해야 합니다. 또한 다음 결정 작업을 수신하는 결정자가 실행 종료를 방해한 이벤트(이 경우에는 신호)에 응답하는지 확인합니다.

또한 워크플로 실행의 취소를 지원할 수도 있습니다. 이러한 기능은 오래 실행 중인 워크플로에 특히 유용할 수 있습니다. 취소를 지원하려면 결정자가 내역의 `WorkflowExecutionCancelRequested`

이벤트를 처리해야 합니다. 이 이벤트는 실행 취소가 요청되었음을 나타냅니다. 결정자는 진행 중인 활동 작업 취소와 CancelWorkflowExecution 결정으로 RespondDecisionTaskCompleted 작업을 호출해 워크플로 닫기 등과 같은 적절한 정리 작업을 수행해야 합니다.

다음 예에서는 RespondDecisionTaskCompleted를 호출해 현재 워크플로 실행이 취소되었음을 지정합니다.

```
https://swf.us-east-1.amazonaws.com
RespondDecisionTaskCompleted
{
  "taskToken" : "12342e17-80f6-FAKE-TASK-TOKEN32f0223",
  "decisions" : [
    {
      "decisionType":"CancelWorkflowExecution",
      "CancelWorkflowExecutionAttributes":{
        "Details": "Customer canceled order"
      }
    }
  ]
}
```

Amazon SWF는 워크플로 실행 닫기 또는 취소 결정이 결정자가 보낸 마지막 결정인지 확인합니다. 즉, 워크플로를 닫는 결정 이후에는 결정이 들어 있는 결정 세트가 있으면 잘못된 것입니다.

결정자 시작

결정자 개발을 완료하면 하나 이상의 결정자를 시작할 준비가 된 것입니다.

결정자를 시작하려면 결정자 플랫폼에서 사용할 수 있는 실행 파일로 조정 로직을 패키징합니다. 예를 들어, Linux 및 Windows 컴퓨터 양쪽 모두에서 실행할 수 있는 Java 실행 파일로 결정자 코드를 패키징할 수 있습니다.

시작되면 결정자는 작업을 위해 Amazon SWF를 폴링하기 시작해야 합니다. 사용자가 워크플로 실행을 시작하고 Amazon SWF에서 결정 작업을 예약할 때까지 이러한 폴링은 시간 초과되어 빈 응답을 받습니다. 빈 응답은 Task 구조로, 여기서 taskToken의 값이 빈 문자열입니다. 결정자는 계속해서 폴링해야 합니다.

Amazon SWF는 항상 하나의 워크플로 실행에 대해 결정 작업을 하나만 활성화할 수 있도록 합니다. 따라서 결정 충돌과 같은 문제가 방지됩니다. 또한 Amazon SWF는 실행 중인 결정자 수에 상관없이 결정자 하나에는 결정 작업이 하나만 할당되도록 합니다.

결정자가 다른 결정 작업을 처리하는 동안 결정 작업 하나를 발생시키는 어떤 일이 일어나면 Amazon SWF는 현재 작업이 완료될 때까지 새 작업을 대기열에 추가합니다. 현재 작업이 완료되면 Amazon SWF는 새 결정 작업을 사용 가능하도록 만듭니다. 또한 결정 작업은 일괄 처리됩니다. 즉, 결정자가 결정 작업을 처리하는 동안 여러 활동이 완료되면 Amazon SWF는 여러 작업 완료를 고려하는 새로운 결정 작업을 하나만 생성합니다. 그러나 워크플로 실행 내역에는 각 작업 완료에 대한 개별 이벤트가 수신됩니다.

폴링은 아웃바운드 요청이기 때문에 Amazon SWF 엔드포인트에 액세스할 수 있는 모든 네트워크에서 결정자를 실행할 수 있습니다.

워크플로 실행을 진행하도록 하려면 결정자가 하나 이상 실행 중이어야 합니다. 원하는 만큼 결정자를 실행할 수 있습니다. Amazon SWF는 동일한 작업 목록에 대한 여러 결정자 폴링을 지원합니다.

Amazon SWF에서 워크플로 시작

어떤 애플리케이션에서든 `StartWorkflowExecution` 작업을 사용해 등록된 워크플로 유형의 워크플로 실행을 시작할 수 있습니다. 실행을 시작할 때 `workflowId`라는 식별자를 실행과 연결합니다. 애플리케이션에 알맞은 문자열을 `workflowId`로 사용할 수 있습니다(예: 주문 처리 애플리케이션의 주문 번호). 동일한 도메인 내에서 복수의 열린 워크플로 실행에 대해 동일한 `workflowId`를 사용할 수 없습니다. 예를 들어, `workflowId` `Customer Order 01`로 두 가지 워크플로 실행을 시작하면 두 번째 워크플로 실행이 시작되지 않고 요청은 실패합니다. 하지만 종료된 실행의 `workflowId`를 다시 사용할 수 있습니다. 또한 Amazon SWF는 시스템에서 생성한 고유한 식별자인 `runId`를 각 워크플로 실행과 연결합니다.

워크플로 및 활동 유형이 등록되면 `StartWorkflowExecution` 작업을 호출해 워크플로를 시작합니다. `input` 파라미터의 값은 워크플로를 시작하는 애플리케이션에서 지정한 임의의 문자열일 수 있습니다. `executionStartToCloseTimeout`은 워크플로 실행이 시작되어 닫힐 때까지 걸릴 수 있는 초 단위 시간입니다. 이 제한을 초과하면 워크플로 실행이 시간 초과됩니다. Amazon SWF의 다른 제한 시간 파라미터 중 일부와 달리 이 제한 시간에 대해서는 `NONE` 값을 지정할 수 없습니다. 워크플로 실행을 실행할 수 있는 최대 시간 제한은 1년입니다. 마찬가지로, taskStartToCloseTimeout은 워크플로 실행과 연결된 결정 작업이 시간 초과되기 이전에 실행될 수 있는 초 단위 시간입니다.

```
https://swf.us-east-1.amazonaws.com
StartWorkflowExecution
{
  "domain" : "867530901",
  "workflowId" : "20110927-T-1",
  "workflowType" : {
    "name" : "customerOrderWorkflow", "version" : "1.1"
```

```
  },
  "taskList" : { "name" : "specialTaskList" },
  "input" : "arbitrary-string-that-is-meaningful-to-the-workflow",
  "executionStartToCloseTimeout" : "1800",
  "tagList" : [ "music purchase", "digital", "ricoh-the-dog" ],
  "taskStartToCloseTimeout" : "1800",
  "childPolicy" : "TERMINATE"
}
```

`StartWorkflowExecution` 작업에 성공하면 Amazon SWF는 워크플로 실행을 위한 `runId`를 반환합니다. 워크플로 실행의 `runId`는 특정 리전 내에서 고유합니다. 나중에 Amazon SWF 직접 호출에서 워크플로 실행을 지정해야 하는 경우를 대비해 `runId`를 저장해 둡니다. 예를 들어, 이후에 워크플로 실행으로 신호를 보내야 하는 경우 `runId`를 사용할 수 있습니다.

```
{"runId": "9ba33198-4b18-4792-9c15-7181fb3a8852"}
```

Amazon SWF에서 작업 우선 순위 설정

기본적으로 작업 목록의 작업은 도착 시간에 따라 제공됨: 가능한 한 먼저 예약된 작업이 일반적으로 먼저 실행됩니다. 선택적 작업 우선 순위를 설정해 특정 작업에 우선 순위를 부여할 수 있음: Amazon SWF는 작업 목록에서 우선 순위가 높은 작업을 우선 순위가 낮은 작업보다 먼저 제공하려고 합니다.

> **Note**
>
> 일반적으로 먼저 예약된 작업이 먼저 실행되지만 이러한 순서가 반드시 보장되는 것은 아닙니다.

워크플로 및 활동 둘 다에 대해 작업 우선 순위를 설정할 수 있습니다. 워크플로의 작업 우선 순위는 워크플로가 예약한 활동 작업의 우선 순위에 영향을 미치지 않고 워크플로가 시작한 하위 워크플로에도 영향을 미치지 않습니다. 활동 또는 워크플로의 기본 우선 순위는 등록 중(사용자 또는 Amazon SWF가) 설정하지만 활동을 예약하거나 워크플로 실행을 시작하는 동안 재정의하지 않는 한 항상 등록된 작업 우선 순위가 사용됩니다.

작업 우선 순위 값의 범위는 "-2147483648" ~ "2147483647"일 수 있으며 숫자가 클수록 우선 순위가 높음을 나타냅니다. 활동 또는 워크플로에 대해 작업 우선 순위를 설정하지 않으면 우선 순위 0이 할당됩니다.

주제

워크플로의 작업 우선 순위 설정

워크플로를 등록 또는 시작할 때 워크플로의 작업 우선 순위를 설정할 수 있습니다. 워크플로 실행 설정 시 재정의되지 않는 한 워크플로 유형 등록 시 설정한 작업 우선 순위가 해당 형의 워크플로 실행에 대해 기본값으로 사용됩니다.

기본 작업 우선 순위로 워크플로 유형을 등록하려면 RegisterWorkflowType 작업을 사용할 때 defaultTaskPriority 옵션을 설정하십시오.

```
{
  "domain": "867530901",
  "name": "expeditedOrderWorkflow",
  "version": "1.0",
  "description": "Expedited customer orders workflow",
  "defaultTaskStartToCloseTimeout": "600",
  "defaultExecutionStartToCloseTimeout": "3600",
  "defaultTaskList": {"name": "mainTaskList"},
  "defaultTaskPriority": "10",
  "defaultChildPolicy": "TERMINATE"
}
```

StartWorkflowExecution으로 워크플로 실행을 시작하는 경우, 워크플로우 유형의 등록된 작업 우선 순위를 재정의할 수 있습니다.

```
{
  "childPolicy": "TERMINATE",
  "domain": "867530901",
  "executionStartToCloseTimeout": "1800",
  "input": "arbitrary-string-that-is-meaningful-to-the-workflow",
  "tagList": ["music purchase", "digital", "ricoh-the-dog"],
  "taskList": {"name": "specialTaskList"},
  "taskPriority": "-20",
  "taskStartToCloseTimeout": "600",
  "workflowId": "20110927-T-1",
  "workflowType": {"name": "customerOrderWorkflow", "version": "1.0"},
}
```

또한 하위 워크플로우를 시작하거나 워크플로우를 새로 계속 진행할 때도(예: RespondDecisionTaskCompleted로 결정에 응답하는 경우) 등록된 작업 우선 순위를 재정의할 수 있습니다.

하위 워크플로의 작업 우선 순위를 설정하려면 `startChildWorkflowExecutionDecisionAttributes`에 값을 제공합니다.

```
{
  "taskToken": "AAAAKgAAAAEAAAAAAAAA...",
  "decisions": [
    {
      "decisionType": "StartChildWorkflowExecution",
      "startChildWorkflowExecutionDecisionAttributes": {
        "childPolicy": "TERMINATE",
        "control": "digital music",
        "executionStartToCloseTimeout": "900",
        "input": "201412-Smith-011x",
        "taskList": {"name": "specialTaskList"},
        "taskPriority": "5",
        "taskStartToCloseTimeout": "600",
        "workflowId": "verification-workflow",
        "workflowType": {
          "name": "MyChildWorkflow",
          "version": "1.0"
        }
      }
    }
  ]
}
```

워크플로를 새 워크플로로 계속 실행하는 경우 `continueAsNewWorkflowExecutionDecisionAttributes`에 작업 우선 순위를 설정합니다.

```
{
  "taskToken": "AAAAKgAAAAEAAAAAAAAA...",
  "decisions": [
    {
      "decisionType": "ContinueAsNewWorkflowExecution",
      "continueAsNewWorkflowExecutionDecisionAttributes": {
        "childPolicy": "TERMINATE",
        "executionStartToCloseTimeout": "1800",
        "input": "5634-0056-4367-0923,12/12,437",
```

```
        "taskList": {"name": "specialTaskList"},
        "taskStartToCloseTimeout": "600",
        "taskPriority": "100",
        "workflowTypeVersion": "1.0"
      }
    }
  ]
}
```

활동의 작업 우선 순위 설정

활동을 등록하거나 예약할 때 활동에 대한 작업 우선 순위를 설정할 수 있습니다. 활동 예약 재정의하지 않는 한 작업 유형 등록 시 설정한 작업 우선 순위가 활동 실행 시 기본 우선 순위로 사용됩니다.

활동 유형을 등록할 때 작업 우선 순위를 지정하려면 RegisterActivityType 작업을 사용할 때 defaultTaskPriority 옵션을 설정하십시오.

```
{
  "defaultTaskHeartbeatTimeout": "120",
  "defaultTaskList": {"name": "mainTaskList"},
  "defaultTaskPriority": "10",
  "defaultTaskScheduleToCloseTimeout": "900",
  "defaultTaskScheduleToStartTimeout": "300",
  "defaultTaskStartToCloseTimeout": "600",
  "description": "Verify the customer credit card",
  "domain": "867530901",
  "name": "activityVerify",
  "version": "1.0"
}
```

작업 우선 순위가 있는 작업을 예약하려면 RespondDecisionTaskCompleted 작업이 있는 활동을 예약할 때 taskPriority 옵션을 사용하십시오.

```
{
  "taskToken": "AAAAKgAAAAEAAAAAAAAAA...",
  "decisions": [
    {
      "decisionType": "ScheduleActivityTask",
      "scheduleActivityTaskDecisionAttributes": {
        "activityId": "verify-account",
        "activityType": {
            "name": "activityVerify",
```

```
            "version": "1.0"
        },
        "control": "digital music",
        "input": "abab-101",
        "taskList": {"name": "mainTaskList"},
        "taskPriority": "15"
      }
    }
  ]
}
```

작업 우선 순위 정보를 반환하는 작업

다음 Amazon SWF 작업에서 설정된 작업 우선 순위(또는 설정된 기본 작업 우선 순위)에 대한 정보를 얻을 수 있습니다.

- DescribeActivityType은 응답의 `configuration` 섹션에 있는 defaultTaskPriority 활동 유형을 반환합니다.
- DescribeWorkflowExecution은 응답의 `executionConfiguration` 섹션에 있는 taskPriority 워크플로 실행을 반환합니다.
- DescribeWorkflowType은 응답의 `configuration` 섹션에 있는 defaultTaskPriority 워크플로 유형을 반환합니다.
- GetWorkflowExecutionHistory 및 PollForDecisionTask는 응답의 `activityTaskScheduledEventAttributes`, `decisionTaskScheduledEventAttributes`, `workflowExecutionContinuedAsNewEventAttributes` 및 `workflowExecutionStartedEventAttributes` 섹션에 작업 우선 순위 정보를 제공합니다.

Amazon SWF의 오류 처리

워크플로 실행 과정 중에는 여러 가지 유형의 오류가 발생할 수 있습니다.

주제

- 유효성 검사 오류
- 작업 또는 결정 실행 오류
- 시간 초과
- 사용자 코드로 인해 발생한 오류

- 워크플로 실행 닫기와 관련된 오류

유효성 검사 오류

유효성 검사 오류는 Amazon SWF에 대한 요청의 형식이 제대로 지정되지 않았거나 요청에 잘못된 데이터가 포함되어 있어 요청에 실패하는 경우 발생합니다. 이러한 경우 요청은 DescribeDomain과 같은 작업 또는 StartTimer와 같은 결정일 수 있습니다. 요청이 작업인 경우 Amazon SWF에서는 응답에 오류 코드를 반환합니다. 실패의 원인이 된 요청의 측면에 대한 정보를 제공할 수 있으므로 오류 코드를 확인합니다. 예를 들어, 요청과 함께 전달된 하나 이상의 인수가 유효하지 않을 수 있습니다. 일반적인 오류 코드 목록을 보려면 Amazon Simple Workflow Service API 참조의 작업 주제를 참조하십시오.

실패한 요청이 결정인 경우 워크플로 실행 내역에 적절한 이벤트가 나열됩니다. 예를 들어, StartTimer 결정에 실패하면 내역에 StartTimerFailed 이벤트가 표시됩니다. 결정자는 PollForDecisionTask 또는 GetWorkflowExecutionHistory에 대한 응답으로 내역을 수신할 때 이러한 이벤트가 있는지 확인해야 합니다. 아래는 결정의 형식이 잘못 지정되었거나 결정에 잘못된 데이터가 포함된 경우 발생할 수 있는 가능한 결정 실패 이벤트 목록입니다.

작업 또는 결정 실행 오류

요청의 형식이 적절하게 지정되었더라도 Amazon SWF에서 요청을 수행하려고 할 때 오류가 발생할 수 있습니다. 이러한 경우 내역의 다음 이벤트 중 하나가 오류가 발생했음을 나타냅니다. 이벤트의 reason 필드를 살펴보고 실패의 원인을 확인합니다.

- CancelTimerFailed
- RequestCancelActivityTaskFailed
- RequestCancelExternalWorkflowExecutionFailed
- ScheduleActivityTaskFailed
- SignalExternalWorkflowExecutionFailed
- StartChildWorkflowExecutionFailed
- StartTimerFailed

시간 초과

결정자, 활동 작업자 및 워크플로우 실행은 모두 제한 시간의 한계 안에서 작동합니다. 이 유형의 오류에서는 작업 또는 하위 워크플로의 시간이 초과됩니다. 내역에 시간 초과를 설명하는 이벤트가 표시됩

니다. 결정자는 예를 들어 작업을 다시 예약하거나 하위 워크플로를 다시 시작해 해당 이벤트를 처리해야 합니다. 제한 시간에 대한 자세한 정보는 Amazon SWF 제한 시간 유형 단원을 참조하십시오.

- ActivityTaskTimedOut
- ChildWorkflowExecutionTimedOut
- DecisionTaskTimedOut
- WorkflowExecutionTimedOut

사용자 코드로 인해 발생한 오류

이 유형의 오류 상태의 예는 활동 작업 실패와 하위 워크플로 실패입니다. 제한 시간 오류와 마찬가지로 Amazon SWF에서는 워크플로 실행 내역에 적절한 이벤트를 추가합니다. 결정자는 가능한 경우 작업을 다시 예약하거나 하위 워크플로를 다시 시작해 해당 이벤트를 처리해야 합니다.

- ActivityTaskFailed
- ChildWorkflowExecutionFailed

워크플로 실행 닫기와 관련된 오류

결정자는 보류 중인 결정 작업이 있는 워크플로를 닫으려고 하는 경우 다음 이벤트를 확인할 수도 있습니다.

- FailWorkflowExecutionFailed
- CompleteWorkFlowExecutionFailed
- ContinueAsNewWorkflowExecutionFailed
- CancelWorkflowExecutionFailed

위에 나열된 이벤트에 대한 자세한 내용은 Amazon SWF API 참조의 기록 이벤트를 참조하세요.

Amazon SWF 할당량

Amazon SWF는 계정당 도메인 수, 워크플로 실행 내역 크기 등과 같은 특정 워크플로 파라미터에 대해 크기 할당량을 지정합니다. 이러한 할당량은 잘못된 워크플로가 시스템의 리소스를 전부 사용해 버리는 일을 방지하기 위해 설계되었으나, 하드 제한은 아닙니다. 애플리케이션이 이러한 할당량을 종종 초과하는 경우에는 서비스 할당량 증가를 요청할 수 있습니다.

내용

Amazon SWF의 일반 계정 할당량

- 등록된 최대 도메인 수 – 100개

  이 할당량에는 등록된 도메인과 더 이상 사용하지 않는 도메인이 모두 포함됩니다.
- 워크플로 및 활동 유형의 최대 개수 – 도메인당 10,000개

  이 할당량에는 등록된 유형과 더 이상 사용하지 않는 유형이 모두 포함됩니다.
- API 직접 호출 할당량 – 부정기적인 급증 시기 외에도, 매우 짧은 기간에 다수의 API 작업이 발생하는 경우 애플리케이션을 조정할 수 있습니다.
- 최대 요청 크기 – 요청당 1MB

  Amazon SWF API 요청당 전체 데이터 크기로, 요청 헤더 및 그 밖의 모든 연결된 요청 데이터를 포함합니다.
- Count API에 대한 잘린 응답 – 내부 할당량에 도달했으며 응답이 최대 수가 아니었음을 나타냅니다.

  일부 쿼리가 전체 응답을 반환하기 전에 내부적으로 상기의 1MB 할당량에 도달하게 됩니다. 다음은 최대 수 대신 잘린 응답을 반환할 수 있습니다.

- CountClosedWorkflowExecutions
- CountOpenWorkflowExecutions
- CountPendingActivityTasks
- CountPendingDecisionTasks

  이들 각각은 `truncated` 응답이 true로 설정될 경우 그 수가 최대량보다 적습니다. 이 내부 할당량은 높일 수 없습니다.
- 최대 태그 수 – 리소스당 50개 태그

  50개를 초과하여 태그를 추가하려고 하면 400 오류(`TooManyTagsFault`)가 발생합니다.

워크플로 실행 할당량

- 열려 있는 최대 워크플로 실행 수 – 도메인당 100,000개

  이 개수에는 하위 워크플로 실행이 포함됩니다.
- 최대 워크플로 실행 기간 – 1년 이 수는 하드 할당량이며 변경할 수 없습니다.
- 최대 워크플로 실행 내역 크기 – 이벤트 25,000개 이 수는 하드 할당량이며 변경할 수 없습니다.

  내역의 이벤트 개수가 10,000개를 넘지 않도록 각 워크플로를 구성하는 것이 가장 좋습니다. 결정자가 워크플로 기록을 가져와야 하므로 기록이 작을수록 결정자가 더 빨리 완료할 수 있습니다. Flow Framework를 사용하는 경우에는 ContinueAsNew를 사용해 워크플로를 새 내역으로 계속 실행할 수 있습니다.
- 최대 하위 워크플로 실행 수 – 워크플로 실행당 1,000개

이러한 할당량을 초과해야 하는 사용 사례에서는 Amazon SWF의 기능으로 하위 워크플로 실행을 사용해 애플리케이션을 구성하고 실행을 계속할 수 있습니다. 그래도 할당량 증가가 필요한 경우에는 할당량 증가 요청 단원을 참조하십시오.

작업 실행에 대한 할당량

- 작업 목록당 최대 폴러 수 – 작업 목록당 1,000개

  특정 작업 목록을 동시에 폴링할 수 있는 Poller의 최대 개수는 1,000개입니다. 1,000개를 초과하면 `LimitExceededException`을 받게 됩니다.

> **Note**
>
> 최대값은 1,000개이지만 이 할당량에 도달하기 전에 `LimitExceededException` 오류가 발생할 수 있습니다. 이 오류가 발생했다고 해서 작업이 지연되고 있는 것은 아닙니다. 대신 작업 목록에 있는 유휴 폴러의 수가 최대라는 뜻입니다. Amazon SWF는 클라이언트와 서버 측 모두에서 리소스를 절약하기 위해 이 제한을 설정합니다. 제한을 설정하면 너무 많은 수의 폴러가 불필요하게 대기하는 것을 방지할 수 있습니다. 작업 목록을 여러 개 사용해 폴링을 분산하여 `LimitExceededException` 오류를 줄일 수 있습니다.

- 초당 예약된 최대 작업 수 – 작업 목록당 2,000개

  특정 작업 목록에서 초당 최대 2,000개의 작업을 예약할 수 있습니다. 2,000개를 초과하면 `ScheduleActivityTask` 결정이 실패하고 `ACTIVITY_CREATION_RATE_EXCEEDED` 오류가 발생합니다.

  > **Note**
  >
  > 최대값은 2,000개이지만 이 할당량에 도달하기 전에 `ACTIVITY_CREATION_RATE_EXCEEDED` 오류가 발생할 수 있습니다. 이러한 오류를 줄이려면 작업 목록을 여러 개 사용해 로드를 분산하십시오.

- 최대 작업 실행 시간 – 1년(워크플로 최대 실행 시간에 따라 제한됨)

  활동 작업 실행이 너무 오래 걸리는 경우 제한 시간 이벤트가 발생하도록 활동 제한 시간을 구성할 수 있습니다.

- SWF에서 작업을 대기열에 보유할 수 있는 최대 기간 – 1년(워크플로 실행 시간 할당량에 따라 제한됨)

  활동 등록 중에 활동 작업 실행의 특정 단계가 너무 오래 걸리는 경우 제한 시간 이벤트가 발생하도록 기본 활동 제한 시간을 구성할 수 있습니다. 또한 결정자 코드에서 활동 작업을 예약하는 경우 기본 활동 제한 시간을 재정의할 수도 있습니다.

- 열려 있는 최대 활동 작업 수 – 워크플로 실행당 1,000개

  이 할당량에는 일정이 지정된 활동 작업과 작업자가 처리 중인 활동 작업이 모두 포함됩니다.

- 열려 있는 최대 타이머 수 – 워크플로 실행당 1,000개
- 최대 입력/결과 데이터 크기 – 32,768자

이 할당량은 활동 또는 워크플로 실행 결과 데이터, 입력 데이터(활동 작업 또는 워크플로 실행을 예약 시 입력 데이터) 및 워크플로 실행 신호와 함께 전송된 입력에 적용됩니다.

- 결정 작업 응답의 최대 결정 개수 – 경우에 따라 다름

  최대 API 요청 크기 1MB의 할당량이 있기 때문에 `RespondDecisionTaskCompleted`에 대한 직접 호출 하나에서 반환되는 결정 수는 각 결정에서 사용하는 데이터의 크기에 따라 제한됩니다. 여기에는 예약된 활동 작업 또는 워크플로우 실행에 제공된 입력 데이터의 크기가 포함됩니다.

Amazon SWF 제한 할당량

앞서 설명한 Service Quotas 외에도 Amazon SWF API 호출 및 결정 이벤트는 토큰 버킷 체계를 사용하여 서비스 대역폭을 유지하기 위해 조절됩니다. 요청량이 여기 나열된 양을 계속해서 초과하는 경우 제한 할당량 증가를 요청할 수 있습니다.

제한 및 결정 할당량은 모든 리전에서 동일합니다.

모든 리전에 대한 제한 할당량

다음 할당량은 개별 계정 수준에서 적용됩니다. 다음 할당량에 대한 증가를 요청할 수 있습니다. 이를 위한 자세한 방법은 할당량 증가 요청을 참조하세요.

API 이름	버킷 크기	초당 다시 채우기 속도
`CountClosedWorkflowExecutions`	2000	6
`CountOpenWorkflowExecutions`	2000	6
`CountPendingActivityTasks`	200	6
`CountPendingDecisionTasks`	200	6
`DeleteActivityType`	200	6
`DeleteWorkflowType`	200	6
`DeprecateActivityType`	200	6
`DeprecateDomain`	100	6

API 이름	버킷 크기	초당 다시 채우기 속도
DeprecateWorkflowType	200	6
DescribeActivityType	2000	6
DescribeDomain	200	6
DescribeWorkflowExecution	2000	6
DescribeWorkflowType	2000	6
GetWorkflowExecutionHistory	2000	60
ListActivityTypes	200	6
ListClosedWorkflowExecutions	200	6
ListDomains	100	6
ListOpenWorkflowExecutions	200	48
ListTagsForResource	50	30
ListWorkflowTypes	200	6
PollForActivityTask	2000	200
PollForDecisionTask	2000	200
RecordActivityTaskHeartbeat	2000	160
RegisterActivityType	200	60
RegisterDomain	100	6
RegisterWorkflowType	200	60
RequestCancelWorkflowExecution	2000	30
RespondActivityTaskCanceled	2000	200

API 이름	버킷 크기	초당 다시 채우기 속도
RespondActivityTaskCompleted	2000	200
RespondActivityTaskFailed	2000	200
RespondDecisionTaskCompleted	2000	200
SignalWorkflowExecution	2000	30
StartWorkflowExecution	2000	200
TagResource	50	30
TerminateWorkflowExecution	2000	60
UndeprecateActivityType	200	6
UndeprecateDomain	100	6
UndeprecateWorkflowType	200	6
UntagResource	50	30

모든 리전에 대한 결정 할당량

다음 할당량은 개별 계정 수준에서 적용됩니다. 다음 할당량에 대한 증가를 요청할 수 있습니다. 이를 위한 자세한 방법은 할당량 증가 요청을 참조하세요.

API 이름	버킷 크기	초당 다시 채우기 속도
RequestCancelExternalWorkflowExecution	1200	120
ScheduleActivityTask	1000	200
SignalExternalWorkflowExecution	1200	120
StartChildWorkflowExecution	500	12

API 이름	버킷 크기	초당 다시 채우기 속도
StartTimer	2000	200

워크플로우 수준 할당량

다음 할당량은 워크플로우 수준에서 적용되며 증가할 수 없습니다.

API 이름	버킷 크기	초당 다시 채우기 속도
GetWorkflowExecutionHistory	400	200
SignalWorkflowExecution	1000	1000
RecordActivityTaskHeartbeat	1000	1000
RequestCancelWorkflowExecution	200	200

할당량 증가 요청

자세한 내용은AWS 일반 참조의 AWS Service Quotas를 참조하세요.

Amazon SWF에 대한 추가 리소스 및 참조 정보

이 장에서는 Amazon SWF로 워크플로 개발 시 유용한 추가 리소스 및 참조 정보를 제공합니다.

주제

- Amazon SWF 제한 시간 유형
- Amazon Simple Workflow Service 엔드포인트
- Amazon Simple Workflow Service에 대한 추가 설명서
- Amazon Simple Workflow Service용 웹 리소스
- Ruby Flow의 마이그레이션 옵션

Amazon SWF 제한 시간 유형

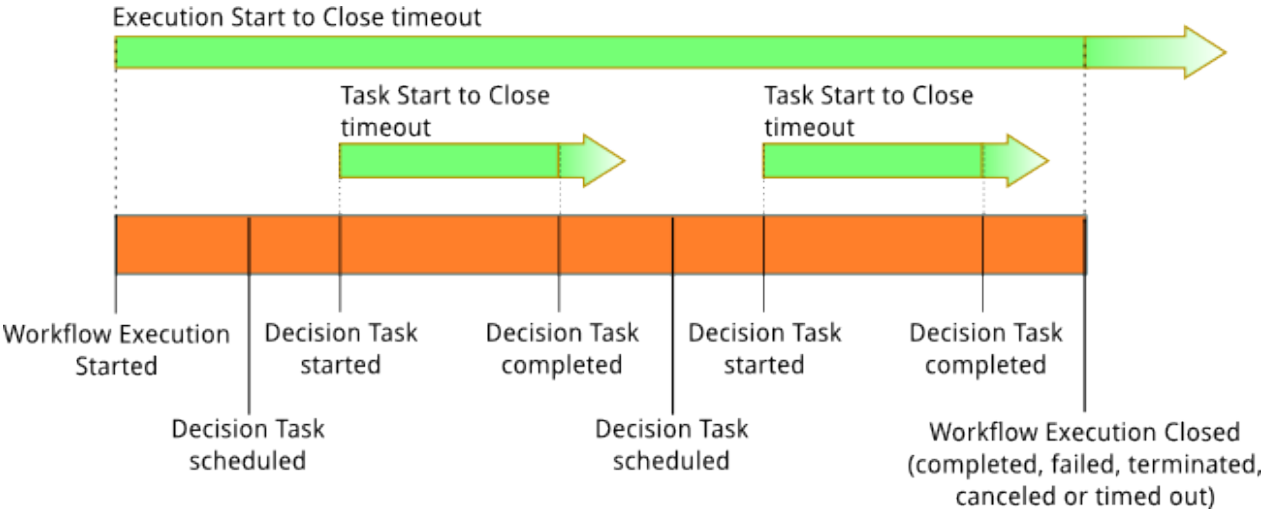
워크플로 실행이 올바르게 실행되도록 Amazon SWF를 사용하여 다양한 유형의 제한 시간을 설정할 수 있습니다. 워크플로 전체를 얼마나 오래 실행할 수 있는지 지정하는 제한 시간도 있고, 활동 작업이 작업자에게 할당되기까지 걸리는 시간 및 예약 시점으로부터 완료되기까지 걸리는 시간을 지정하는 제한 시간도 있습니다. Amazon SWF API의 모든 제한 시간은 초 단위로 지정됩니다. Amazon SWF는 `NONE` 문자열을 제한 시간 값으로도 지원하며 이는 제한 시간이 없음을 나타냅니다.

결정 작업 및 활동 작업과 관련된 제한 시간의 경우, Amazon SWF는 워크플로 실행 내역에 이벤트를 추가합니다. 이러한 이벤트의 속성은 발생한 제한 시간의 유형과 해당하는 결정 작업 또는 활동 작업에 대한 정보를 제공합니다. Amazon SWF는 결정 작업을 예약합니다. 새 결정 작업을 받은 결정자는 내역에 표시된 제한 시간 이벤트를 보고 RespondDecisionTaskCompleted 작업을 호출해 적절한 조치를 취합니다.

작업은 예약 시점부터 닫힐 때까지 열린 상태로 간주됩니다. 따라서 작업자가 작업을 처리하는 동안에는 열려 있는 상태로 보고됩니다. 작업자가 작업을 완료됨, 취소됨 또는 실패로 보고하면 작업이 닫힙니다. 제한 시간으로 인해 Amazon SWF에서 작업을 종료할 수도 있습니다.

워크플로 및 결정 작업의 제한 시간

다음 다이어그램은 워크플로 및 결정 제한 시간이 워크플로의 수명과 어떤 관계인지 보여줍니다.

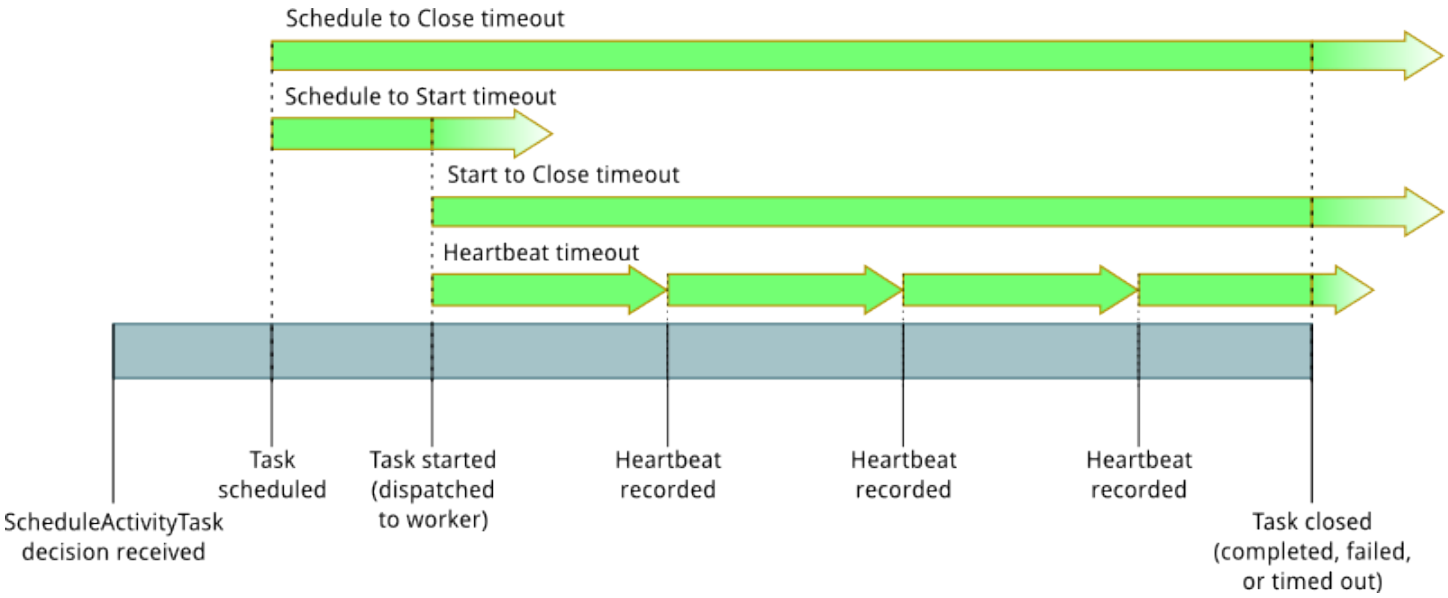


워크플로 및 결정 작업과 관련된 제한 시간 유형은 다음 두 가지입니다.

- 워크플로 시작-닫기(**`timeoutType: START_TO_CLOSE`**) – 이 제한 시간은 워크플로 실행이 완료되는 데 걸리는 최대 시간을 지정합니다. 워크플로 등록 과정에서 기본값으로 설정되지만 워크플로가 시작될 때 다른 값으로 재정의할 수 있습니다. 이 제한 시간을 초과하면 Amazon SWF는 워크플로 실행을 종료하고 워크플로 실행 기록에 WorkflowExecutionTimedOut 유형의 이벤트를 추가합니다. `timeoutType`을 비롯한 이벤트 속성으로 이 워크플로 실행에 적용되는 `childPolicy`를 지정합니다. 하위 정책은 상위 워크플로 실행 시간이 초과되거나 그렇지 않고 종료되면 하위 워크플로 실행을 처리하는 방법을 지정합니다. 예를 들어, `childPolicy`를 TERMINATE로 설정하면 하위 워크플로 실행이 종료됩니다. 워크플로 실행이 시간 초과되면 가시성 호출 이외의 조치를 취할 수 없습니다.
- 결정 작업 시작-닫기(**`timeoutType: START_TO_CLOSE`**) – 이 제한 시간은 해당 결정자가 결정 작업을 완료하는 데 소요할 수 있는 최대 시간을 지정합니다. 이 값은 워크플로 유형 등록 중 설정합니다. 이 제한 시간을 초과하면 워크플로 실행 내역에 해당 작업이 시간 초과된 것으로 표시되고, Amazon SWF는 워크플로 내역에 DecisionTaskTimedOut 유형의 이벤트를 추가합니다. 이벤트 속성에는 결정 작업이 예약되었을 때 해당하는 이벤트의 ID(`scheduledEventId`)와 결정 작업이 시작되었을 때 해당하는 이벤트의 ID(`startedEventId`)가 포함됩니다. Amazon SWF는 이벤트를 추가하는 한편 새 결정 작업을 예약해 결정자에게 이 결정 작업이 시간 초과되었음을 알립니다. 시간 초과가 발생한 후 `RespondDecisionTaskCompleted`를 사용해 시간 초과된 결정 작업을 완료하려고 하면 실패합니다.

활동 작업의 제한 시간

다음 다이어그램은 제한 시간이 활동 작업의 수명과 어떻게 관련이 있는지를 보여줍니다.



여기에는 활동 작업과 관련된 제한 시간 유형이 4개 있습니다.

- 활동 작업 시작-닫기(**timeoutType: START_TO_CLOSE**) – 이 제한 시간은 활동 작업자가 작업을 수신한 후 작업을 처리하기 위해 보낼 수 있는 최대 시간을 지정합니다. RespondActivityTaskCanceled, RespondActivityTaskCompleted 및 RespondActivityTaskFailed를 사용해 시간 초과된 활동 작업을 닫으려고 하면 실패합니다.
- 활동 작업 하트비트(**timeoutType: HEARTBEAT**) – 이 제한 시간은 RecordActivityTaskHeartbeat 작업을 통해 진행 상황을 제공하기 전에 작업을 실행할 수 있는 최대 시간을 지정합니다.
- 활동 작업 예약-시작(**timeoutType: SCHEDULE_TO_START**) – 활동 작업을 수행할 작업자가 없을 때 Amazon SWF는 이 제한 시간만큼 기다렸다가 활동 작업을 시간 초과로 처리합니다. 시간이 초과로 만료된 작업은 다른 작업자에게 할당되지 않습니다.
- 활동 작업 예약-닫기(**timeoutType: SCHEDULE_TO_CLOSE**) – 이 제한 시간은 예약 시간부터 완료될 때까지 걸릴 수 있는 기간을 지정합니다. 이 값은 작업 예약-시작 제한 시간과 작업 시작-닫기 제한 시간의 합보다 크면 안 됩니다.

> **Note**
>
> 각 제한 시간 유형에는 기본값이 있는데, 일반적으로 NONE(무한정)으로 설정되어 있습니다. 그러나 활동 실행의 최대 시간은 1년으로 제한됩니다.

활동 유형 등록 중 활동에 대한 기본값을 설정하지만 활동 작업을 예약할 때 새 값으로 기본값을 재정의할 수 있습니다. 이러한 제한 시간 중 하나가 발생하면 Amazon SWF는 워크플로 기록에 ActivityTaskTimedOut 유형의 이벤트를 추가합니다. 이 이벤트의 `timeoutType` 값 속성은 어떤 제한 시간이 발생했는지 지정합니다. 각 제한 시간의 경우 `timeoutType`의 값은 괄호 안에 표시됩니다. 또한 이벤트 속성에는 활동 작업이 예약되었을 때 해당하는 이벤트의 ID(`scheduledEventId`)와 결정 작업이 시작되었을 때 해당하는 이벤트의 ID(`startedEventId`)가 포함됩니다. Amazon SWF는 이벤트를 추가하는 한편 새 결정 작업을 예약해 결정자에게 시간 초과가 발생했음을 알립니다.

Amazon Simple Workflow Service 엔드포인트

현재 Amazon SWF 리전 및 엔드포인트 목록은 다른 서비스의 엔드포인트와 함께 Amazon Web Services 일반 참조에서 확인할 수 있습니다.

Amazon SWF 도메인과 관련된 모든 워크플로 및 활동이 서로 통신하기 위해서는 동일한 리전 내에 있어야 합니다. 또한 리전 내의 등록된 모든 도메인, 워크플로 및 활동은 다른 리전에는 없습니다. 예를 들어, us-east-1 및 us-west-2 리전 둘 다에서 "MySampleDomain"이라는 도메인을 생성하면 해당 도메인은 별도의 도메인으로 존재합니다. 즉, 도메인과 연결된 워크플로, 작업 목록, 활동 또는 데이터 중 어떤 것도 리전 간에 공유되지 않습니다.

워크플로에서 Amazon EC2 인스턴스와 같은 다른 AWS 리소스를 사용하는 경우 이러한 리소스도 Amazon SWF 리소스와 동일한 리전에 있어야 합니다. 유일한 예외는 Amazon S3 및 IAM과 같이 여러 리전에 걸쳐 있는 서비스입니다. 이러한 서비스에는 해당 서비스를 지원하는 모든 리전에 있는 워크플로에서 액세스할 수 있습니다.

Amazon Simple Workflow Service에 대한 추가 설명서

이 개발자 안내서 외에도 다음 문서가 유용합니다.

Amazon Simple Workflow Service API 참조

Amazon Simple Workflow Service API 참조에는 작업, 요청 및 응답 구조와 오류 코드를 비롯해 Amazon SWF HTTP API에 대한 자세한 정보가 나와 있습니다.

AWS Flow Framework 설명서

AWS Flow Framework는 Amazon SWF로 워크플로우 및 활동을 관리하는 분산된 비동기 애플리케이션의 구현 프로세스를 간소화하는 프로그래밍 프레임워크입니다. 따라서 워크플로우 로직을 구현하는 데 집중할 수 있습니다.

각 AWS Flow Framework 는 설계된 언어로 관용적으로 작동하도록 설계되었으므로 선택한 언어로 자연스럽게 작업하여 Amazon SWF의 모든 이점을 갖춘 워크플로를 구현할 수 있습니다.

Java용 AWS Flow Framework가 있습니다. AWS Flow Framework for Java 개발자 안내서에서는 for Java를 얻고 설정하고 사용하는 방법에 AWS Flow Framework 대한 정보를 제공합니다.

AWS SDK 설명서

AWS 소프트웨어 개발 키트(SDKs)는 다양한 프로그래밍 언어로 Amazon SWF에 대한 액세스를 제공합니다. SDK는 HTTP API를 밀접하게 따르지만 일부 Amazon SWF 기능의 경우 언어별 프로그래밍 인터페이스도 제공합니다. 다음 링크를 방문하면 각 SDK에 대한 자세한 정보를 찾을 수 있습니다.

> **Note**
>
> 여기에는 작성 당시 Amazon SWF를 지원한 SDK만 나열되어 있습니다. 사용 가능한 AWS SDKs. https://aws.amazon.com/tools/

Java

는 AWS 인프라 서비스를 위한 Java API를 AWS SDK for Java 제공합니다.

사용 가능한 설명서를 보려면 AWS SDK for Java 설명서 페이지를 참조하십시오. 또한 다음 링크를 따라 SDK 참조의 Amazon SWF 섹션으로 바로 이동할 수도 있습니다.

- `Class: AmazonSimpleWorkflowClient`
- `Class: AmazonSimpleWorkflowAsyncClient`
- `Interface: AmazonSimpleWorkflow`
- `Interface: AmazonSimpleWorkflowAsync`

JavaScript

를 AWS SDK for JavaScript 통해 개발자는 브라우저 또는 서버의 Node.js 애플리케이션 내부에서 사용할 수 있는 간단하고 easy-to-use API를 사용하여 서비스를 사용하는 AWS 라이브러리 또는 애플리케이션을 구축할 수 있습니다.

사용 가능한 설명서를 보려면 AWS SDK for JavaScript 설명서 페이지를 참조하십시오. 다음 링크를 따라 SDK 참조의 Amazon SWF 섹션으로 바로 이동할 수도 있습니다.

- `Class: AWS.SimpleWorkflow`

.NET

는 Visual Studio 프로젝트 템플릿, AWS .NET 라이브러리, C# 코드 샘플 및 설명서가 포함된 다운로드 가능한 단일 패키지 AWS SDK for .NET 입니다. 를 AWS SDK for .NET 사용하면 Windows 개발자가 Amazon SWF 및 기타 서비스를 위한 .NET 애플리케이션을 더 쉽게 구축할 수 있습니다.

사용 가능한 설명서를 보려면 AWS SDK for .NET 설명서 페이지를 참조하십시오. 또한 다음 링크를 따라 SDK 참조의 Amazon SWF 섹션으로 바로 이동할 수도 있습니다.

- `Namespace: Amazon.SimpleWorkflow`
- `Namespace: Amazon.SimpleWorkflow.Model`

PHP

는 Amazon SWF에 대한 PHP 프로그래밍 인터페이스를 AWS SDK for PHP 제공합니다.

사용 가능한 설명서를 보려면 AWS SDK for PHP 설명서 페이지를 참조하십시오. 다음 링크를 따라 SDK 참조의 Amazon SWF 섹션으로 바로 이동할 수도 있습니다.

- `Class: SwfClient`

Python

는 Amazon SWF에 대한 Python 프로그래밍 인터페이스를 AWS SDK for Python (Boto) 제공합니다.

사용 가능한 설명서를 보려면 boto: Amazon Web Services에 대한 Python 인터페이스 페이지를 참조하십시오. 또한 다음 링크를 따라 설명서의 Amazon SWF 섹션으로 바로 이동할 수도 있습니다.

- Amazon SWF 자습서
- Amazon SWF 참조

Ruby

는 Amazon SWF에 대한 Ruby 프로그래밍 인터페이스를 AWS SDK for Ruby 제공합니다.

사용 가능한 설명서를 보려면 AWS SDK for Ruby 설명서 페이지를 참조하십시오. 다음 링크를 따라 SDK 참조의 Amazon SWF 섹션으로 바로 이동할 수도 있습니다.

- 클래스: AWS::SimpleWorkflow

AWS CLI 설명서

AWS Command Line Interface (AWS CLI)는 AWS 서비스를 관리하기 위한 통합 도구입니다. 다운로드 및 구성할 도구를 하나만 사용하면 명령줄에서 여러 AWS 서비스를 제어하고 스크립트를 통해 자동화할 수 있습니다.

에 대한 자세한 내용은 AWS Command Line Interface 페이지를 AWS CLI참조하십시오.

Amazon SWF에 사용할 수 있는 명령에 대한 개요는 AWS CLI 명령 참조의 swf를 참조하십시오.

Amazon Simple Workflow Service용 웹 리소스

Amazon SWF에 대해 자세히 알아보고 서비스 사용 및 워크플로 개발에 도움을 얻는 데 활용할 수 있는 다양한 웹 리소스가 있습니다.

Amazon SWF 포럼

Amazon SWF 포럼은 Amazon의 다른 Amazon SWF 개발자 및 Amazon SWF 개발 팀원과 질문 및 답변을 통해 의사소통할 수 있는 장소를 제공합니다.

포럼: Amazon Simple Workflow Service에서 포럼을 방문할 수 있습니다.

Amazon SWF FAQ

Amazon SWF FAQ는 일반 사용 사례, Amazon SWF와 다른 서비스 간의 차이점 등에 대한 개요를 비롯해 Amazon SWF에 대해 자주 묻는 질문에 대한 답변을 제공합니다.

Amazon SWF FAQ에서 FAQ에 액세스할 수 있습니다.

Amazon SWF 비디오

YouTube의 Amazon Web Services 채널에서는 Amazon SWF를 비롯해 모든 Amazon Web Services에 대한 동영상 교육을 제공합니다. Amazon SWF 관련 비디오의 전체 목록을 보려면 Amazon Web Services의 Simple Workflow쿼리를 사용합니다.

Ruby Flow의 마이그레이션 옵션

AWS Flow Framework for Ruby는 더 이상 활성 개발 중이 아닙니다. 기존 코드는 계속 사용 가능하지만 새 기능이나 버전은 제공되지 않습니다. 이 단원에서는 Amazon SWF를 계속 사용하기 위한 사용법과 마이그레이션 옵션 및 Amazon SWF를 마이그레이션하는 방법을 설명합니다.

옵션	설명
Ruby Flow Framework 사용	현재 Ruby Flow Framework는 계속 사용할 수 있습니다. 아무 작업 없이 코드는 그대로 계속 작동합니다. 가까운 시일 내에 AWS Flow Framework for Ruby에서 마이그레이션할 계획입니다.
Java Flow Framework로 마이그레이션	Java Flow Framework는 계속 개발 중이며 새 기능과 업데이트가 계속 제공됩니다.
Step Functions로 마이그레이션	Step Functions를 사용하면 상태 시스템을 통해 제어되는 시각적 워크플로를 사용하여 분산 애플리케이션의 구성 요소를 조정할 수 있습니다.
Flow Framework 없이 SWF API 직접 사용	Ruby Flow Framework 대신 Ruby에서 SWF API를 직접 사용할 수 있습니다.

Ruby 또는 Java에 대해 Flow Framework가 제공하는 이점을 활용하여 워크플로 로직에 초점을 맞출 수 있습니다. 이 프레임워크는 커뮤니케이션 및 조정에 대한 많은 세부 정보를 다루며 일부 복잡성을 추상화합니다. Java Flow Framework로 마이그레이션하여 동일한 수준의 추상화를 유지하거나, Amazon SWF SDK를 직접 사용할 수 있습니다.

Ruby Flow Framework 사용

Ruby AWS Flow Framework 용는 현재와 같이 단기적으로 계속 작동합니다. AWS Flow Framework for Ruby에 작성된 워크플로가 있으면 계속 작동합니다. 업데이트, 지원 또는 보안 수정 없이 가까운 시일 내에 AWS Flow Framework for Ruby를 마이그레이션하기 위한 확실한 계획을 세우는 것이 좋습니다.

Java Flow Framework로 마이그레이션

Java AWS Flow Framework 용는 활성 개발 상태로 유지됩니다. 개념적으로 AWS Flow Framework for Java는 AWS Flow Framework for Ruby와 비슷합니다. 워크플로 로직에 계속 집중할 수 있으며 프레임워크는 결정자 로직을 관리하는 데 도움이 되고 Amazon SWF의 다른 측면을 더 쉽게 관리할 수 있습니다.

- AWS Flow Framework for Java
- AWS Flow Framework Java API 참조용

Step Functions로 마이그레이션

AWS Step Functions 는 Amazon SWF와 유사하지만 워크플로 로직이 상태 시스템에 의해 제어되는 서비스를 제공합니다. Step Functions은 시각적 워크플로우를 사용해 분산 애플리케이션 및 마이크로서비스의 구성 요소를 손쉽게 조정하도록 해주는 웹 서비스입니다. 각각 기능 또는 작업을 수행하는 개별 구성 요소를 사용하여 애플리케이션을 구축하면 애플리케이션을 빠르게 확장하거나 변경할 수 있습니다. Step Functions는 구성 요소를 조정하고 애플리케이션 기능을 단계별로 실행할 수 있는 안정적인 방법을 제공합니다. 그래픽 화면의 콘솔에서 애플리케이션의 구성 요소를 일련의 단계로 시각화할 수 있습니다. 자동으로 각 단계를 트리거 및 추적하고 오류가 발생할 경우 재시도하므로 애플리케이션이 항상 의도대로 정상적으로 실행됩니다. Step Functions는 각 단계의 상태를 기록합니다. 따라서 무언가 잘못된 경우 빠르게 문제를 진단하고 디버깅할 수 있습니다.

Step Functions에서는 선언적 JSON으로 작성되고 Amazon States Language를 사용하여 정의된 상태 시스템을 사용하여 작업 조정을 관리합니다. 상태 머신을 사용하면 애플리케이션 로직을 제어하기 위해 결정자 프로그램을 작성하고 유지 관리할 필요가 없습니다. Step Functions는 시각적 워크플로를 사용하여 애플리케이션 구성 요소를 조정하는 직관적이고 생산적이며 민첩한 접근 방식을 제공합니다. 모든 새 애플리케이션에 AWS Step Functions 를 사용하는 것을 고려해야 하며, Step Functions는 현재 AWS Flow Framework for Ruby에서 구현한 워크플로에 대해 로 마이그레이션할 수 있는 우수한 플랫폼을 제공합니다.

Ruby 언어 스킬을 계속 이용하면서 작업을 Step Functions로 마이그레이션할 수 있도록 Step Functions는 예제 Ruby 활동 작업자를 제공합니다. 이 예제는 활동 작업자를 구현하기 위한 모범 사례를 사용하며 작업 로직을 Step Functions로 마이그레이션하기 위한 템플릿으로 사용할 수 있습니다. 자세한 내용은 AWS Step Functions 개발자 안내서의 Ruby 활동 작업자 예제 주제를 참조하십시오.

> **Note**
>
> 많은 고객의 경우 AWS Flow Framework for Ruby에서 Step Functions로 마이그레이션하는 것이 가장 좋습니다. 그러나 해당 신호가 프로세스에 개입해야 하거나 결과를 상위로 반환하는 하위 프로세스를 시작해야 하는 경우 Amazon SWF API를 직접 사용하거나 AWS Flow Framework for Java로 마이그레이션하는 것이 좋습니다.

에 대한 자세한 내용은 다음을 AWS Step Functions참조하세요.

- AWS Step Functions 개발자 안내서
- AWS Step Functions API Reference
- AWS Step Functions 명령줄 참조

Amazon SWF API 직접 사용하기

AWS Flow Framework for Ruby는 Amazon SWF의 일부 복잡성을 관리하지만 Amazon SWF API를 직접 사용할 수도 있습니다. API를 직접 사용하면 진행 상황을 추적하고 상태를 유지 관리하는 것과 같이 기저의 복잡한 문제에 대한 염려 없이 구현 작업과 조정을 완벽하게 제어하는 워크플로를 만들 수 있습니다.

- Amazon Simple Workflow Service 개발자 안내서
- Amazon Simple Workflow Service API 참조

문서 이력

다음 표에서는 Amazon Simple Workflow Service 개발자 안내서의 최근 릴리스가 나온 이후에 이 설명서에서 변경된 중요 사항에 대해 설명합니다.

변경 사항	설명	변경 날짜
설명서 전용 업데이트	Amazon SWF에는 이제에서 알림의 중앙 위치 역할을 AWS 서비스 하는 AWS 사용자 AWS 알림에 대한 섹션이 포함되어 있습니다 AWS Management Console. 자세한 내용은 Amazon Simple Workflow Service AWS 사용자 알림 에서 사용 단원을 참조하십시오.	2023년 5월 4일
업데이트	이제 Amazon SWF는 SWF 워크플로와 실행 관련 작업을 관리할 수 있는 새로운 콘솔 환경을 제공합니다. 자세한 내용은 Amazon SWF 콘솔 자습서를 참조하십시오.	2022년 9월 12일
업데이트	`Maximum tasks scheduled per second`를 포함하도록 작업 실행에 대한 할당량 섹션을 업데이트하고 CloudWatch에서 비 ASCII 리소스 이름 사용 정보를 포함하도록 CloudWatch에 대한 Amazon SWF 지표 페이지를 업데이트했습니다.	2021년 5월 12일
새로운 특성	Amazon Simple Workflow Service에서는 이제 Amazon EventBridge를 지원합니다. 자세한 내용은 다음을 참조하세요. • Amazon SWF용 EventBridge • EventBridge 사용자 설명서	2020년 12월 18일

새로운 특성	Amazon Simple Workflow Service는 태그를 사용한 IAM 권한을 지원합니다. 추가 정보는 다음을 참조하세요.
	• Amazon SWF의 태그
	  • 태그 관리
	  • 워크플로 실행에 태그 지정
	  • 태그를 사용하여 도메인에 대한 액세스 제어
		2019년 6월 20일

변경 사항	설명	변경 날짜
	• TagResource • UntagResource • ListTagsForResource • RegisterDomain	
새로운 특성	이제 유럽(스톡홀름) 리전에서 Amazon Simple Workflow Service를 사용할 수 있습니다.	2018년 12월 12일
업데이트	CloudTrail 통합에 대한 Amazon Simple Workflow Service 주제를 개선했습니다. 를 사용하여 API 호출 기록 AWS CloudTrail을(를) 참조하세요.	2018년 8월 7일
업데이트	CloudWatch의 새로운 PendingTasks 지표에 대한 정보를 추가했습니다. 자세한 내용은 Amazon SWF 지표 단원을 참조하십시오.	2018년 18월 6일
업데이트	코드 샘플의 구문 강조 기능 개선	2018년 3월 29일
업데이트	Ruby Flow 사용자를 위해 해당 플랫폼에서 마이그레이션하기 위한 옵션을 설명하는 단원이 추가되었습니다. 자세한 내용은 Ruby Flow의 마이그레이션 옵션 단원을 참조하십시오.	2018년 3월 9일
업데이트	고급 개념 단원에 대한 탐색 구조가 개선되었습니다. Amazon SWF의 고급 워크플로 개념을(를) 참조하세요.	2018년 2월 19일
업데이트	CloudWatch 지표 설명서에 유효한 통계 정보를 추가하여 내용을 개선했습니다. CloudWatch에 대한 Amazon SWF 지표을(를) 참조하세요.	2017년 12월 4일
업데이트	문서 구조를 개선하기 위해 TOC가 변경되었습니다. API 및 결정 이벤트 측정치 단원에 대한 새로운 정보가 추가되었습니다.	2017년 11월 9일
업데이트	모든 리전에 대한 조절 제한을 포함하도록 Amazon SWF 할당량 단원이 업데이트되었습니다.	2017년 10월 18일

변경 사항	설명	변경 날짜
업데이트	Amazon SWF 시작하기 단원에서 `activity_list` 와의 혼동을 피하기 위해 `task_list` 가 `workflowId` 로 변경되었습니다.	2017년 7월 25일
업데이트	이 가이드 전체의 코드 샘플을 정리했습니다.	2017년 6월 5일
업데이트	이 가이드의 구성 및 내용을 간소화하고 개선했습니다.	2017년 5월 19일
업데이트	업데이트 내용 및 링크를 수정했습니다.	2017년 5월 16일
업데이트	업데이트 내용 및 링크를 수정했습니다.	2016년 10월 1일
Lambda 작업 지원	워크플로 내 기존 활동 작업 이외에 Lambda 작업을 지정할 수 있습니다. 자세한 내용은 AWS Lambda Amazon SWF의 태스크 단원을 참조하십시오.	2015년 7월 21일
작업 우선 순위 설정 지원	Amazon SWF는 이제 작업 목록에 대한 작업 우선순위를 설정할 수 있어 우선순위가 낮은 작업에 앞서 우선순위가 높은 작업을 전송하려고 시도합니다. 워크플로 및 활동에 대한 작업 우선 순위 설정 방법은 Amazon SWF에서 작업 우선 순위 설정 단원을 참조하십시오.	2014년 12월 17일
업데이트	CloudTrail을 사용하여 Amazon SWF API 호출을 로깅하는 방법을 설명하는 새로운 항목 추가: 를 사용하여 API 호출 기록 AWS CloudTrail.	2014년 5월 8일
업데이트	Amazon SWF에 대한 CloudWatch 지표와 관련된 새로운 항목 2개가 추가되었습니다. CloudWatch에 대한 Amazon SWF 지표 단원에서는 지원되는 지표 목록 및 설명을 제공하고, 를 사용하여 CloudWatch에 대한 Amazon SWF 지표 보기 AWS Management Console 단원에서는 AWS Management Console을 사용하여 지표를 보고 경보를 설정하는 방법을 설명합니다.	2014년 4월 28일

변경 사항	설명	변경 날짜
업데이트	새로운 단원(Amazon SWF에 대한 추가 리소스 및 참조 정보)이 추가되었습니다. 이 단원에서는 몇 가지 서비스 참조 정보를 제공하고 Amazon SWF 개발자를 위해 추가 설명서, 샘플, 코드 및 기타 웹 리소스에 대한 정보를 제공합니다.	2014년 3월 19일
업데이트	워크플로 자습서가 추가되었습니다. Amazon SWF 시작하기을(를) 참조하세요.	2013년 10월 25일
업데이트	AWS CLI 정보 및 예가 추가되었습니다.	2013년 8월 26일
업데이트	내용이 업데이트 및 수정되었습니다.	2013년 8월 1일
업데이트	IAM을 사용하여 액세스를 제어하는 방법을 설명하는 문서가 업데이트되었습니다.	2013년 2월 22일
최초 릴리스	이 설명서는 Amazon Simple Workflow Service 개발자 안내서의 최초 릴리스입니다.	2012년 10월 16일

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.