亚马逊云科技

开发人员指南

Amazon Simple Workflow Service

API 版本 2012-01-25

Amazon Simple Workflow Service: 开发人员指南

Table of Contents

什么是 Amazon Simple Workflow Service？

借助亚马逊简单工作流服务 (Amazon SWF) Simple Workflow Service，您可以构建、运行和扩展具有并行或顺序步骤的后台作业。您可以跨分布式组件协调工作并跟踪任务状态。

在 Amazon SWF 中，任务代表由应用程序组件执行的逻辑工作单元。跨任务协调包括管理任务间的依赖关系、调度和应用程序流程中的并发性。借助 Amazon SWF，您可以控制和协调任务，而不必担心潜在的复杂性，例如跟踪进度和维护任务状态。

使用 Amazon SWF 时，您需要实施工作人员来执行任务。工作人员可以在云基础设施（例如亚马逊弹性计算云 (Amazon EC2)）上运行，也可以在您自己的本地运行。您可以创建长时间运行的任务，也可以创建可能失败、超时或需要重启的任务，还可以创建可能以不同的吞吐量和延迟完成的任务。Amazon SWF 存储任务并在任务准备就绪时将其分配给工作人员，跟踪进度并维护状态，包括任务完成的细节。

为了协调任务，您需要编写一个从 Amazon SWF 获取最新任务状态并使用该状态启动后续任务的程序。Amazon SWF 可以持久地保持应用程序的执行状态，因此您的应用程序可以抵御单个组件故障。借助 Amazon SWF，您可以独立构建、部署、扩展和修改应用程序组件。

> **其他 Amazon 工作流程服务**
>
> 对于大多数用例，我们建议考虑 Amazon Step Functions 您的工作流程和编排需求。借助 Step Functions，您可以创建工作流（也称为状态机），以构建分布式应用程序、自动执行流程、协调微服务以及创建数据和机器学习管道。在 Step Functions 的控制台或 VS Code 中的 Amazon 工具包中，您可以使用图形化的 Workflow Studio 来可视化、编辑、测试和调试应用程序的工作流程。
>
> 有关更多技术信息，请参阅Amazon Step Functions 开发人员指南。

使用 Amazon SWF 开发工作流程组件

开发分布式应用程序需要协调许多组件，并处理远程通信固有的延迟和不可靠性。

借助 Amazon Simple Workflow Service (Amazon SWF)，您可以通过提供编程模型和基础设施来协调分布式组件并以可靠的方式维护其执行状态，从而开发异步和分布式应用程序。借助 Amazon SWF，您可以专注于构建令应用程序与众不同的功能。

工作流程的组成部分

工作流程的组成部分Amazon SWF 中的基本概念是工作流程。工作流程是一组开展一些目标的活动，以及协作这些活动的逻辑。例如，工作流程可以接收客户订单，并采取任何必要的措施来履行订单。

每个工作流程都在名为域的资源中运行，该域控制着工作流程的范围。一个 Amazon 账户可以有多个域，每个域可以包含多个工作流，但不同域中的工作流不能交互。

在设计 Amazon SWF 工作流程时，您需要定义每项必需的活动。然后将每个活动作为一个活动类型在 Amazon SWF 中注册。您将提供名称、版本和超时值。例如，客户可能期望订单在 24 小时内发货。

工作流程执行过程中，有些活动可能需要执行多次，可能要用不同的输入。例如，在客户订购工作流程中，您可能要执行一个处理已购买项目的活动。如果客户购买了多件商品，则该活动必须运行多次。Amazon SWF 有活动任务的概念，表示对活动的一次调用。在我们的示例中，每一个项目的处理都可以用一个活动任务表示。

活动工作人员是一个接收活动任务、执行活动任务并提供结果的程序。该任务实际上可能由一个人执行。例如，统计分析师可能会收到一组数据，对数据进行分析，然后将分析结果发送回去。

活动任务以及执行这些任务的活动工作人员可以同步或异步运行。工作人员可以在一个地点运行，也可以分布在多台计算机上，可能位于不同的地理区域。不同的活动工作程序可以用不同的编程语言编写，且可运行在不同的操作系统上。例如，一个活动工作人员可能在亚洲的服务器上运行，而另一个活动工作人员可能在北美的移动设备上运行。

工作流程中的协作逻辑包含在被称为决策程序的软件程序中。决策者安排活动任务，向活动工作人员提供输入，处理在工作流程进行时到达的事件，并在目标实现后结束（或关闭）工作流程。

Amazon SWF 服务的角色可以充当一个可靠的中央枢纽，决策程序、活动工作线程和其他相关实体（如工作流管理人员）可以通过它来交换数据。Amazon SWF 还将维护每个工作流执行的状态，这样，您的应用程序就不必持久存储状态。

决策程序会接收 Amazon SWF 的决策任务并将决策结果反馈给 Amazon SWF，以此来管理工作流。决策代表一个或一组操作，它们是工作流程中的后续步骤。一般的决策为排定活动任务。决策还可用于通过计时器延迟任务、请求取消正在进行的任务以及完成工作流程。

活动工作线程和决策程序接收任务（分别为活动任务和决策任务）的机制是轮询 Amazon SWF 服务。

Amazon SWF 向决策程序告知工作流的状态，其中包括每个决策任务以及当前工作流的执行历史记录。工作流程执行历史由事件组成，其中事件代表工作流程执行状态的重要更改。事件的示例包括任务完成、任务超时或计时器过期。历史是工作流程进程的完整、一致且权威的记录。

Amazon SWF 访问控制使用 Amazon Identity and Access Management (IAM)，因此您可以控制对资源的访问权限。 Amazon 例如，您可以允许用于访问您的账户，但只运行其在特定的域中运行特定工作流程。

运行你的工作流程

以下内容概述了在 Amazon SWF 中开发和运行工作流程所需的步骤：

1. 编写活动工作人员来执行工作流程中的处理步骤。
2. 编写一个决策者来处理工作流程的协调逻辑。
3. 向 Amazon SWF 注册您的活动和工作流。

   您可以通过编程方式完成此步骤，也可以使用。 Amazon Web Services Management Console
4. 启动您的活动工作程序和决策程序。

   这些操作者能够在可访问 Amazon SWF 端点的任何计算设备上运行。例如，您可以使用云中的计算实例，例如亚马逊弹性计算云 (Amazon EC2)；数据中心的服务器；甚至是移动设备，来托管决策者或活动工作者。启动后，决策程序和活动工作线程应该轮询 Amazon SWF 以查找任务。
5. 启动工作流程的一个或多个执行。

   您可以通过编程方式启动工作流程，也可以通过。 Amazon Web Services Management Console

   每一个执行都独立运行，您可以为每个执行提供其自己的输入数据集。执行启动后，Amazon SWF 会计划初始决策任务。作为回应，您的决策者开始生成启动活动任务的决策。执行会继续，直到您的决策程序做出关闭执行的决策。
6. 使用查看工作流程执行情况 Amazon Web Services Management Console。

   您可以筛选和查看正在运行和已完成的执行的完整详细信息。例如，您可以选择打开的执行以查看哪些任务已完成以及其结果如何。

设置开发环境

您可以选择使用支持的任何一种编程语言为 Amazon SWF 进行开发。 Amazon对于 Java 开发人员 Amazon Flow Framework ，也可以使用。有关更多信息，请Amazon Flow Framework访问网站，并参阅Amazon Flow Framework 《Java 开发人员指南》。

为了减少延迟并将数据存储在符合您要求的位置，Amazon SWF 在不同区域提供了终端节点。

Amazon SWF 中的每个终端节点都是完全独立的。您在一个地区注册的任何域、工作流程和活动都不会与其他区域的域名、工作流程和活动共享数据或属性。

当您注册 Amazon SWF 域名、工作流程或活动时，该域名、工作流程或活动仅存在于您注册的区域内。例如，您可以注册一个名为两个不同区域`SWF-Flows-1`的域名，但它们彼此之间不会共享任何数据或属性——每个区域都充当一个完全独立的域。

有关 Amazon SWF 端点的列表，请参阅 Regions and Endpoints。

与之一起开发 Amazon SDKs

Amazon SDKs 适用于 Java、.NET、Node.js、PHP、Python 和 Ruby 的 Amazon SWF 支持，这为以你选择的编程语言使用 Amazon SWF HTTP API 提供了一种便捷的方式。

您可以使用这些库公开的 API 开发决策者、活动工作人员或工作流程启动者。而且，您可以通过这些库使用可视性操作，从而开发自己的 Amazon SWF 监控和报告工具。

要下载用于开发和管理应用程序的工具（包括） SDKs，请访问开发者中心。 Amazon

有关每个软件开发工具包中的 Amazon SWF 操作的详细信息，请参阅该软件开发工具包的特定语言参考文档。

考虑一下 Amazon Flow Framework

Amazon Flow Framework 是一款增强版 SDK，用于编写在 Amazon SWF 上作为工作流程运行的分布式异步程序。该框架可用于 Java 编程语言，并提供用于编写复杂分布式程序的类。

使用 Amazon Flow Framework，您可以使用预配置的类型将工作流程的定义直接映射到程序中的方法。 Amazon Flow Framework 支持标准的面向对象的概念，例如基于异常的错误处理。使用编写的程序 Amazon Flow Framework 可以完全在您的首选编辑器或 IDE 中创建、运行和调试。有关更多信息，请Amazon Flow Framework访问网站，并参阅Amazon Flow Framework 《Java 开发人员指南》。

亚马逊 SWF 入门

您可以开始使用以下 Amazon Simple Workflow Service 工作流程应用程序，该应用程序由按顺序运行的四个活动组成。本教程还涵盖以下主题：

- 设置默认和执行时间工作流程和活动选项。
- 轮询 Amazon SWF 以查找决策和活动任务。
- 通过 Amazon SWF 在活动和工作流之间传递数据。
- 等待人工任务并从活动任务向 Amazon SWF 报告检测信号。
- 使用 Amazon SNS 创建主题、让用户订阅主题并将消息发布到订阅的端点。

您可以同时使用 Amazon SWF 和亚马逊简单通知服务 (Amazon SNS) Simple Notification SERVICE 来模拟 “人工任务” 工作流程，即要求人类工作人员执行某些操作，然后与 Amazon SWF 通信以启动工作流程中的下一个活动。

由于 Amazon SWF 是一项基于云的 Web 服务，因此与 Amazon SWF 的通信可在任何有互联网连接的地方进行。在这种情况下，我们将使用 Amazon SNS 通过电子邮件和/或手机短信与用户通信。

本教程使用访问亚马逊 SWF 和 Amazon SNS，但有许多开发选项可供选择，包括 Amazon Flow Framework 适用于 Ruby 的，它可以更轻松地与 Amazon SWF 进行协调和沟通。适用于 Ruby 的 Amazon SDK

> **Note**
>
> 本教程使用了 适用于 Ruby 的 Amazon SDK，但我们建议您使用Amazon Flow Framework 适用于 Java 的。

主题

- 订阅工作流教程第 3 部分：实现活动
- 订阅工作流程教程第 4 部分：实现活动任务轮询器
- 订阅工作流程教程：运行工作流程

关于工作流程

我们将开发的工作流程由以下四个主要步骤组成：

1. 向用户获取订阅地址（电子邮件或手机短信）。
2. 创建 SNS 主题，然后让所提供的终端节点订阅该主题。
3. 等待用户确认订阅。
4. 如果用户确认，则向该主题发布一条祝贺消息。

这些步骤包括完全自动化的活动（第 2 步和第 4 步）和其他活动（第 1 步和第 3 步），后者要求工作流等待工作线程向活动提供某些数据，然后工作流才能前进。

每一步都依赖于上一步生成的数据 (必须先有终端节点，然后才能让其订阅主题，并且必须先有主题订阅，然后才能等待确认等)。本教程还将介绍如何在活动完成后提供结果，以及如何将输入传递给正在计划的任务。Amazon SWF 负责协调活动和工作流之间的信息传递（反之亦然）。

我们还使用键盘输入和 Amazon SNS 来处理 Amazon SWF 与向工作流提供数据的真人用户之间的通信。实际上，您可以使用多种不同的方法与真人用户通信，但 Amazon SNS 提供了一种非常简便的方式，可以使用电子邮件或手机短信向用户通知工作流中的事件。

先决条件

要按本教程进行操作，您需要以下各项：

- Amazon Web Services 账户
- Ruby 解释器
- 适用于 Ruby 的 Amazon SDK

如果已设置好这些，即准备就绪，可以继续。如果您不想运行该示例，仍然可以按照教程进行操作，无论您选择哪种开发选项，本教程中的大部分内容都适用于使用 Amazon SWF 和 Amazon SNS。

教程步骤

本教程分为以下几步：

1. 订阅工作流程教程第 1 部分：将 Amazon SWF 与 适用于 Ruby 的 Amazon SDK
2. 订阅工作流程教程第 2 部分：实现工作流程
3. 订阅工作流教程第 3 部分：实现活动
4. 订阅工作流程教程第 4 部分：实现活动任务轮询器
5. 订阅工作流程教程：运行工作流程

订阅工作流程教程第 1 部分：将 Amazon SWF 与 适用于 Ruby 的 Amazon SDK

主题

- 包括 适用于 Ruby 的 Amazon SDK
- 配置会 Amazon 话
- 注册 Amazon SWF 域
- 后续步骤

包括 适用于 Ruby 的 Amazon SDK

首先，创建一个名为 `utils.rb` 的文件。该文件中的代码将获取或创建（如有必要）工作流和活动代码所使用的 Amazon SWF 域，并提供一个位置来放置所有类通用的代码。

首先，我们需要在代码中包含 `aws-sdk-v1` 库，以便使用适用于 Ruby的 SDK 提供的功能。

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

这使我们能够访问 Amazon 命名空间，从而能够设置与会话相关的全局值，例如您的 Amazon 证书和区域，还可以访问该 Amazon 服务。 APIs

配置会 Amazon 话

我们将通过设置 Amazon 证书（访问 Amazon 服务所需的凭证）和要使用的 Amazon 区域来配置会 Amazon 话。

在 Ruby 的 Amazon SDK 中设置 Amazon 凭据的方法有很多：在环境变量（AWS_ACCESS_KEY_ID 和 _ACC AWS_SECRET ESS_KEY）中设置凭证，或者使用进行设置。AWS.config我们将使用后一种方法，从一个名为 aws-config.txt 的 YAML 配置文件中加载这些证书，该文件内容类似于此。

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

立即创建此文件，将以 REPLAC E_WITH_ 开头的字符串替换为您的 Amazon 访问密钥 ID 和私有访问密钥。有关您的 Amazon 访问密钥的信息，请参阅如何获取安全证书？ 在 Amazon Web Services 一般参考中。

我们还需要设置要使用的 Amazon 区域。由于我们将使用 Amazon SNS 的短信服务 (SMS) 向用户手机发送文本消息，因此需要确保我们使用的是 Amazon SNS 支持的区域。请参阅《Amazon Simple Notification Service Developer Guide》中的 Supported Regions and Countries。

> **Note**
>
> 如果您无权访问 us-east-1，或不在乎运行演示时是否启用了手机短信，则可随意使用任何地区。您可以从示例中移除 SMS 功能，使用电子邮件作为订阅 Amazon SNS 主题的唯一端点。有关发送 SMS 消息的更多信息，请参阅《Amazon Simple Notification Service Developer Guide》中的 Sending and Receiving SMS Notifications Using Amazon SNS。

现在，我们将在 utils.rb 中添加一些代码以加载配置文件、获取用户凭证，然后将凭证和区域同时提供给 AWS.config。

```
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 Amazon 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
```

注册 Amazon SWF 域

要使用 Amazon SWF，您需要设置一个域，它是一个命名实体，用于存放您的工作流和活动。您可以注册多个 Amazon SWF 域名，但它们在您的 Amazon 账户中都必须具有唯一的名称，并且工作流程不能跨域交互：您的应用程序的所有工作流程和活动都必须在同一个域中才能相互交互。

由于我们将在整个应用程序中使用相同的域，因此我们将在名为 Domain 中创建一个函数*init_domain*，用于检索*utils.rb*名为 Domain 的 Amazon SWF 域。SWFSample

注册域后，可将其重用于多个工作流程执行。但是，尝试注册已存在的域是错误行为，因此我们的代码首先将检查是否存在该域，如果可找到这个现有的域，则将使用它。如果无法找到该域，则我们将创建它。

要在适用于 Ruby 的 SDK 中使用 Amazon SWF 域名，请使用 w AWS::Simpleorkflow.domains，它返回可用于枚举和注册域名的：DomainCollection

- 要查看某个域名是否已经注册，您可以查看 AWS::Simpleworkflow.domains.registed 提供的列表。
- 要注册新域名，请使用 AWS::Simpleworkflow.domains.register。

以下是 `init_domain` 中 `utils.rb` 的代码。

```
# 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
```

后续步骤

接下来，您将在订阅工作流程教程第 2 部分：实现工作流程中创建工作流和启动程序代码。

订阅工作流程教程第 2 部分：实现工作流程

到目前为止，我们的代码都是比较通用的。在此部分，我们将开始实际定义工作流程执行的操作以及需要什么活动才能实现它。

主题

- 设计工作流程
- 设置工作流程代码
- 注册工作流程
- 轮询决策
- 启动工作流程执行
- 后续步骤

设计工作流程

回想一下，此工作流程的初步构想由以下步骤组成：

1. 向用户获取订阅地址（电子邮件或手机短信）。
2. 创建 SNS 主题，然后让所提供的终端节点订阅该主题。
3. 等待用户确认订阅。
4. 如果用户确认，则向该主题发布一条祝贺消息。

我们可将工作流程中的每步视为工作流程必须执行的一个活动。我们的工作流程 负责在适当的时间安排每个活动以及协调活动之间的数据传输。

对于此工作流程，我们将为其中每个步骤创建一个单独的活动，并为其提供描述性名称：

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

这些活动将按顺序执行，并且后续步骤中将使用每步中的数据。

我们可以将应用程序设计为将所有代码放在一个源文件中，但这与 Amazon SWF 的设计宗旨相悖。它适合的工作流程可跨越整个 Internet 范围，因此我们至少要将应用程序分为两个单独的执行文件：

- `swf_sns_workflow.rb` - 包含工作流程和工作流程启动者。
- `swf_sns_activities.rb` - 包含活动和活动启动者。

可在单独的时段、单独的计算机甚至世界上的不同地点运行工作流程和活动实现。由于 Amazon SWF 会跟踪工作流和活动的细节，因此无论活动在何处运行，工作流都能协调其计划和数据传输。

设置工作流程代码

我们首先将创建一个名为 `swf_sns_workflow.rb` 的文件。在此文件中，声明一个名为的类SampleWorkflow。以下是类声明及其构造函数 `initialize` 方法。

```
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
```

如您所见，我们保留以下类实例数据：

- domain - 从 init_domain 中的 utils.rb 检索的域名。
- workflowId - 任务列表传入到 initialize。
- activity_list - 活动列表，其中具有我们将运行的活动的名称和版本。

域名、活动名称和活动版本足以让 Amazon SWF 明确地识别活动类型，因此，这些就是我们要计划活动所需保存的所有数据。

工作流程的 decider 代码将使用任务列表轮询决策任务并安排活动。

在此函数的最后，我们调用一个尚未定义的方法：register_workflow。接下来，我们将定义此方法。

注册工作流程

要使用工作流程类型，必须先注册它。如同活动类型一样，工作流程类型按其域、名称和版本进行标识。此外，与域和活动类型一样，无法重新注册现有的工作流程类型。如果需要更改有关工作流程类型的任何内容，则必须为其提供新版本，基本上就是新建一个类型。

以下是 register_workflow 的代码，它用于检索我们在上次运行时注册的现有工作流程类型，如果尚未注册该工作流程，则注册它。

```
  # 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
```

首先，我们通过循环访问域的 workflow_types 集合，查看是否已注册工作流名称和版本。如果找到了匹配项，则将使用已注册的工作流程类型。

如果我们找不到匹配项，则会注册一个新的工作流程类型（通过在我们搜索工作流程的同一个workflow_types集合上调用 register），名称为 “swf-sns-workflow”，版本为 “1”，并包含以下选项。

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

注册期间传入的选项用于为工作流程类型设置默认行为，因此不需要在每次开始执行新工作流程时设置这些值。

在这里，我们只设置了一些超时值：从任务开始到结束时可用的最长时间（1 小时）以及工作流程执行完毕可用的最长时间（24 小时）。如果超出其中任意一个时间，则任务或工作流程将超时。

有关超时值的详细信息，请参阅Amazon SWF 超时类型 。

轮询决策

在每个工作流程执行的核心部分都有一个决策程序。决策程序的职责是管理工作流程自身的执行。决策程序接收决策任务，然后通过安排新活动、删除并重新启动活动，或通过将工作流程执行的状态设置为完成、已取消或失败，响应这些任务。

决策程序按照工作流程执行的任务列表 名称接收要响应的决策任务。要轮询决策任务，可在域的 decision_tasks 集合上调用 poll，以循环遍历可用的决策任务。然后，可通过循环访问决策任务的 new_events 集合，检查其中是否有新事件。

返回的事件是AWS::SimpleWorkflow::HistoryEvent对象，您可以使用返回事件的 e vent_type 成员来获取事件的类型。有关历史事件类型的列表和描述，请参阅HistoryEvent《亚马逊简单工作流程服务 API 参考》。

下面是决策任务轮询器逻辑的开头。我们的工作流程类中一个名为 `poll_for_decisions` 的新方法。

```
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
```

我们现在将根据收到的 `event_type` 划分决策的执行。我们可能收到的第一个是WorkflowExecutionStarted。收到该事件意味着 Amazon SWF 正在向决策程序发出信号，示意它应开始执行工作流。我们首先将通过对轮询时收到的任务调用 schedule_activity_task，安排第一个活动。

我们将在活动列表中声明的第一个活动传递给它，由于我们颠倒了列表，因此可将其用作堆栈，而该活动占据列表上的 `last` 位置。我们定义的“活动”只是由名称和版本号组成的映射，但这就是 Amazon SWF 识别活动以进行计划所需的全部信息（假定已注册该活动）。

```
        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" } )
```

当我们计划活动时，Amazon SWF 会向我们在计划时传入的活动任务列表发送一个活动任务，示意任务开始。我们将在 订阅工作流教程第 3 部分：实现活动 中处理活动任务，但值得注意的是，我们在此并未执行任务。我们仅需告知 Amazon SWF 应该计划该任务。

我们需要解决的下一个活动是ActivityTaskCompleted事件，该事件发生在 Amazon SWF 收到来自活动任务的活动已完成响应时。

```
          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
```

由于我们以线性方式执行任务，并且一次只有一个活动在执行，所以我们将借此机会从activity_list堆栈中弹出已完成的任务。如果这样导致列表变空，则表示我们的工作流程已完成。在这种情况下，我们通过调用任务的 complete_workflow_execution 向 Amazon SWF 发出工作流已完成的信号。

如果列表中仍有条目，则我们将安排列表上的下一活动（仍处于最后一个位置）。但是，这一次我们将查看上一活动在完成后是否向 Amazon SWF 返回了任何结果数据，这些数据将在事件属性中的可选 `result` 键中提供给工作流。如果该活动产生了结果，则我们将其作为 `input` 选项传递给所安排的下一活动以及活动任务列表。

通过检索完成的活动的 `result` 值以及通过设置安排的活动的 `input` 值，我们可将数据从一个活动传递到下一个，或可使用活动中的数据，根据活动得到的结果更改决策程序中的行为。

就本教程而言，在定义工作流程的行为时，这两种事件类型最为重要。但是，活动可以生成除之外的事件ActivityTaskCompleted。我们将通过为ActivityTaskTimedOut和ActivityTaskFailed事件以及事件提供演示处理程序代码来总结决策程序代码，这些代码将在 WorkflowExecutionCompletedAmazon SWF 处理`complete_workflow_execution`我们用完要运行的活动时发出的调用时生成。

```
          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
```

启动工作流程执行

在将为要轮询的工作流程生成任何决策任务之前，我们需要启动工作流程执行。

要启动工作流程执行，请在注册的工作流程类型 () AWS::SimpleWorkflow::WorkflowType上调用 start_e xecution。我们将在此周围定义一个小型包装器，以利用在类构造函数中检索的 `workflow_type` 实例成员。

```
  def start_execution
```

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

一旦执行工作流，工作流的任务列表（它作为工作流执行选项传入 start_execution）上即开始显示决策事件。

与注册工作流程类型时提供的选项不同，不将传递给 start_execution 的选项视为工作流程类型的一部分。无需更改工作流程的版本，即可随意在每次执行工作流程时更改这些选项。

因为我们希望工作流程在运行文件时开始执行，所以添加一些实例化类的代码，然后调用我们刚才定义start_execution的方法。

```
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
```

为避免任务列表命名发生任何冲突，我们将使用 SecureRandom.uuid 生成可用作任务列表名称的随机 UUID，确保将不同的任务列表名称用于每次工作流程执行。

> **Note**
>
> 任务列表用于记录有关工作流程执行的事件，因此，如果将同一任务列表用于多次执行同一工作流程类型，则可能获得在上一次执行期间生成的事件，当多次执行间隔较小（试验新代码或运行测试时经常这样）时尤为如此。

为了避免必须处理以前执行中的项目的问题，可对每次执行使用新任务列表，在开始工作流程执行时指定该列表。

此处还有一些代码，可向运行代码的人员（很可能是您）提供说明以及提供任务列表的“活动”版本。决策程序使用此任务列表名称为工作流程安排活动，而活动实现将对此任务列表名称侦听活动事件以了解何时开始安排的活动并提供有关活动执行的最新消息。

这段代码还等待用户开始运行活动启动程序，然后再 开始工作流程执行，因此在所提供的任务列表上开始出现活动任务时，活动启动程序将准备好作出响应。

后续步骤

您已实现工作流程。接下来，将在订阅工作流教程第 3 部分：实现活动中定义活动和活动启动程序。

订阅工作流教程第 3 部分：实现活动

我们现在将实现工作流中的每个活动，首先是基类，它为活动代码提供某些共有功能。

主题

- 定义基本活动类型
- 定义 GetContactActivity
- 定义 SubscribeTopicActivity

- 定义 WaitForConfirmationActivity
- 定义 SendResultActivity
- 后续步骤

定义基本活动类型

设计工作流时，我们指定了以下活动：

- `get_contact_activity`
- `subscribe_topic_activity`
- `wait_for_confirmation_activity`
- `send_result_activity`

我们现在将实现这些活动中的每个。因为我们的活动将共享一些功能，所以让我们做一些基础工作，创建一些他们可以共享的常用代码。我们将对其进行调用 BasicActivity，并在名为的新文件中对其进行定义`basic_activity.rb`。

如同其他源文件一样，我们将加入 `utils.rb` 以使用 `init_domain` 函数设置示例域。

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

接下来，我们将声明基本活动类和我们在每项活动中都将感兴趣的一些共有数据。我们将把活动的实AWS::SimpleWorkflow::ActivityType例、名称和结果保存到类的属性中。

```
class BasicActivity

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

这些属性可访问在类的 `initialize` 方法中定义的实例数据，该方法需要一个活动名称、一个可选版本以及向 Amazon SWF 注册活动时使用的选项映射。

```
  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
```

与工作流类型注册一样，如果已注册某个活动类型，则可通过查看域的 activity_types 集合，检索该类型。如果找不到该活动，则将注册该活动。

此外，与工作流类型一样，可设置在注册活动类型时与其存储在一起的默认选项。

我们的基本活动最后得到的是运行它的一致方式。我们将定义一个 do_activity 方法，该方法采用活动任务。如下所示，我们可使用传入的活动任务通过其 input 实例属性接收数据。

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

这结束了这BasicActivity堂课。现在，我们将用它使我们的活动变得简单一致。

定义 GetContactActivity

工作流执行期间运行的第一个活动是 get_contact_activity，它会检索用户的 Amazon SNS 主题订阅信息。

创建一个名为的新文件get_contact_activity.rb，并要求两者兼而有之yaml，我们将使用它来准备传递给 Amazon SWF 的字符串basic_activity.rb，并使用它作为本GetContactActivity类的基础。

```
  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
```

因为我们输入了活动注册码 BasicActivity，所以的initialize方法GetContactActivity非常简单。我们仅仅用活动名称 get_contact_activity 调用基类构造函数。只需此项即可注册我们的活动。

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

现在我们将定义 do_activity 方法，它提示输入用户的电子邮件和/或电话号码。

```
    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
```

在 do_activity 的结尾处，我们获得从用户检索的电子邮件和电话号码，将其放入映射中，然后使用 to_yaml 将整个映射转换为 YAML 字符串。这样做有一个重要原因：当您完成活动后，传递给 Amazon SWF 的任何结果都必须仅为字符串数据。Ruby 可轻松地将对象转换为 YAML 字符串，然后再转换回对象，这一点非常适合此用途。

这是 get_contact_activity 执行的结束。此数据将在接下来的 subscribe_topic_activity 执行中使用。

定义 SubscribeTopicActivity

现在，我们将深入探讨 Amazon SNS 并创建一个活动，该活动使用 get_contact_activity 生成的信息让用户订阅 Amazon SNS 主题。

新建一个名为 subscribe_topic_activity.rb 的文件，添加我们用于 get_contact_activity 的相同要求，声明您的类，然后提供其 initialize 方法。

```
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
```

现在，我们已经有了用于设置和注册活动的代码。下面，我们将添加一些代码来创建 Amazon SNS 主题。为此，我们将使用AWS::SNS::Client对象的 create_topic 方法。

将 create_topic 方法添加到您的类，该方法采用传入的 Amazon SNS 客户端对象。

```
  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
```

一旦我们有了主题的亚马逊资源名称 (ARN)，我们就可以将其与亚马逊 SNS 客户端的 set_topic_attributes 方法一起使用来设置主题 DisplayName，这是使用亚马逊 SNS 发送短信所必需的。

最后，我们将定义 do_activity 方法。首先将收集在安排该活动时通过 input 选项传递的任何数据。如前所述，必须以字符串（我们使用 to_yaml 创建了它）形式传递此项。在检索它时，我们将使用 YAML.load 将数据转换为 Ruby 对象。

以下是 do_activity 的开头，我们在此处检索输入数据。

```
    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))
```

如果我们未收到任何输入，则无计可施，因此我们只好将活动视为失败。

但是，假设一切正常，我们将继续填写do_activity方法，获取带有的 Amazon SNS 客户端 适用于 Ruby 的 Amazon SDK，然后将其传递给我们创建 Amazon SNS 主题create_topic的方法。

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

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

在此，有几点值得注意：

- 我们使用 AWS.config.with 为 Amazon SNS 客户端设置区域。由于我们要发送手机短信，因此我们使用在 utils.rb 中声明的支持手机短信的地区。
- 将该主题的 ARN 保存在 activity_data 映射中。这是将传递给我们的工作流程中下一活动的部分数据。

最后，此活动使用传入的端点（电子邮件和手机短信）让用户订阅 Amazon SNS 主题。我们不要求用户同时输入两个 终端节点，但是，我们至少需要一个。

```
      # 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.subsc ribe 采用主题 ARN，即协议（我们巧妙地将其*activity_data*伪装成相应端点的地图密钥）。

最后，我们以 YAML 格式重新打包下一活动的信息，以便将其发送回 Amazon SWF。

```
      # 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
```

至此，即完成了 subscribe_topic_activity 的执行。接下来，我们将定义 wait_for_confirmation_activity。

定义 WaitForConfirmationActivity

用户订阅 Amazon SNS 主题后，仍需确认订阅请求。在这种情况下，我们将等待用户通过电子邮件或手机短信进行确认。

等待用户确认订阅的活动称为 `wait_for_confirmation_activity`，下面我们将定义它。首先，新建一个名为 `wait_for_confirmation_activity.rb` 的文件，并像设置以前的活动那样设置它。

```
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
```

接下来，我们将开始定义 `do_activity` 方法，然后检索向名为 `subscription_data` 的本地变量中输入的任何数据。

```
  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)
```

现在我们有了主题 ARN，我们可以通过创建新的实例来检索主题，AWS::SNS::Topic然后将 ARN 传递给它。

```
    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
```

现在，我们将检查该主题以了解用户是否已使用某个终端节点确认了订阅。我们只需要一个终端节点得到确认，即将活动视为成功。

Amazon SNS 主题维护该主题的订阅列表，我们可以通过检查订阅的 ARN 是否被设置为 `PendingConfirmation` 以外的值来检查用户是否已确认特定订阅。

```
      # 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
```

如果获取订阅的 ARN，则将其保存在活动的结果数据中，将其转换为 YAML，然后从 `do_activity` 返回 true，这表示活动成功完成。

由于等待订阅确认可能需要一段时间，因此我们偶尔会调用`record_heartbeat`活动任务。这将向 Amazon SWF 发出信号，表明活动仍在处理中，还可用于提供有关活动进度的最新信息（如果您正在进行某些可报告进度的操作，如处理文件）。

```
        task.record_heartbeat!(
          { :details => "#{topic.num_subscriptions_confirmed} confirmed,
#{topic.num_subscriptions_pending} pending" })
        # sleep a bit.
        sleep(4.0)
      end
```

至此，我们的 while 循环结束。如果我们由于某种原因未成功即退出 while 循环，则我们将报告失败并结束 do_activity 方法。

```
      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
```

至此，wait_for_confirmation_activity 的实现结束。我们还剩下一个活动要定义：send_result_activity。

定义 SendResultActivity

如果工作流进行到这一步，说明我们已成功让用户订阅 Amazon SNS 主题，且用户已确认订阅。

我们的最后一个活动 send_result_activity，使用用户订阅的主题和用户确认订阅的终端节点，向用户发送成功主题订阅的确认。

新建一个名为 send_result_activity.rb 的文件，并像设置至今为止的所有活动那样设置它。

```
  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
```

我们的do_activity方法也以类似的方式开始，即从工作流程中获取输入数据，从 YAML 中进行转换，然后使用主题 ARN 创建AWS::SNS::Topic实例。

```
    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
```

具有主题后，我们将向其发布一条消息（并且将其回显到屏幕上）。

```
      @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
```

发布到 Amazon SNS 主题会将您提供的消息发送到该主题已存在的所有订阅和确认的端点。因此，如果用户同时 通过电子邮件和手机短信进行确认，则用户将收到两条确认消息，每个终端节点一条。

后续步骤

这将完成 `send_result_activity` 实现。现在，将在处理活动任务的活动应用程序中将所有这些活动联系在一起，并可启动活动作为回应，如订阅工作流程教程第 4 部分：实现活动任务轮询器所述。

订阅工作流程教程第 4 部分：实现活动任务轮询器

在 Amazon SWF 中，运行工作流执行的活动任务会显示在活动任务列表中，该列表在您计划工作流中的活动时提供。

我们将实施一个基本的活动轮询器来处理工作流的这些任务，并在 Amazon SWF 将任务放入活动任务列表以启动活动时使用它来启动我们的活动。

首先，新建一个名为 swf_sns_activities.rb 的文件。我们将使用它：

- 将我们创建的活动类实例化。
- 将每个活动注册到 Amazon SWF。
- 轮询活动，并在其名称出现在活动任务列表上时对每个活动调用 do_activity。

在 swf_sns_activities.rb 中，添加以下语句以需要我们定义的每个活动类。

```
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'
```

现在，我们将创建类并提供一些初始化代码。

```
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
```

除了保存传入的域 和任务列表 以外，此代码还将我们创建的每个活动类实例化。由于每个类都会注册与其关联的活动（如需查看代码，请参阅 basic_activity.rb），这足以让 Amazon SWF 知道我们将运行的所有活动。

对于每个实例化的活动，我们都将其存储在使用活动名称（如 get_contact_activity）作为键的映射上，因此我们可在活动轮询器代码（接下来我们将定义这段代码）中轻松查找这些活动。

创建一个名为 poll_for_activities 的新方法，并调用域托管的 activity_tasks 上的 poll 来获取活动任务。

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

我们可从任务的 activity_type 成员获取活动名称。接下来，我们将使用与此任务关联的活动名称查找要对其运行 do_activity 的类，并向其传递此任务（其中包括应传输到活动的任何输入数据）。

```
      # 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
```

这段代码仅等待 do_activity 的完成，然后根据返回代码对此任务调用 complete! 或 fail!。

> **Note**
>
> 启动最终活动后，此代码将从轮询器中退出，因为它已完成任务并启动了所有活动。在您自己的 Amazon SWF 代码中，如果您的活动可能再次运行，您可能需要使活动轮询器无限期运行。

我们ActivitiesPoller类的代码到此结束，但我们将在文件末尾再添加一点代码，以允许用户从命令行运行它。

```
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
```

如果用户在命令行下运行该文件（向其传递活动任务列表作为第一个参数），则此代码将轮询器类实例化，并启动它，轮询活动。轮询器结束后（在它启动最后一个活动后），我们打印一条消息即退出。

活动轮询器就此结束。接下来，您只需运行代码，观察其如何发挥作用，如订阅工作流程教程：运行工作流程所述。

订阅工作流程教程：运行工作流程

既然已完成工作流程、活动以及工作流程和活动轮询器的实现，那么我们即准备就绪，可运行工作流程。

如果您还没有这样做，则需要在aws-config.txt文件中提供 Amazon 访问密钥，如本教程配置会 Amazon 话的第 1 部分所示。

现在，转到命令行，然后转到本教程源文件所在的目录。您应有以下文件：

```
.
```

```
|-- 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
```

现在，使用以下命令启动工作流程。

```
ruby swf_sns_workflow.rb
```

此命令将开始工作流程，并应显示一条消息，其中有一行，您可将该行复制并粘贴到单独的命令行窗口（甚至其他计算机，如果已将本教程源文件复制到它上面）中。

```
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...
```

工作流程代码将耐心等待您在单独的窗口中启动活动轮询器。

打开一个新的命令行窗口，再次转到源文件所在的目录，然后使用 swf_sns_workflow.rb 文件提供的命令启动活动轮询器。例如，如果收到了前面的输出，则要键入（粘贴）以下内容。

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

一旦开始运行活动轮询器，它即开始输出有关活动注册的信息。

```
** 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
```

现在可返回原有的命令行窗口，然后按回车键以启动工作流程执行。它将注册工作流程并安排第一个活动。

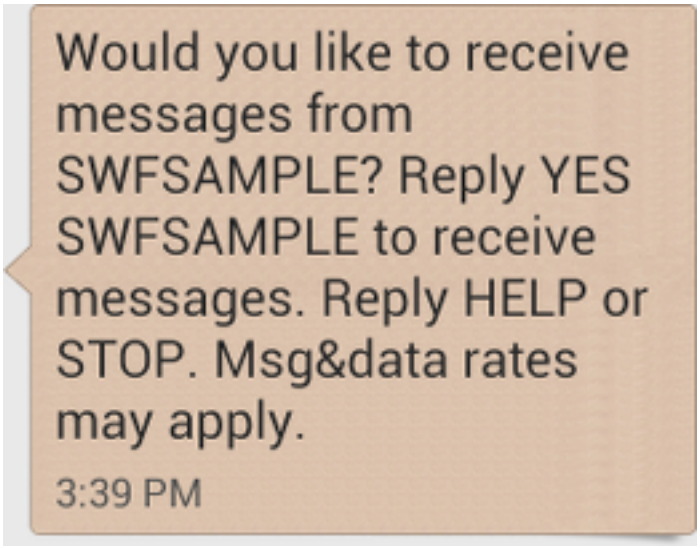
```
Starting workflow execution.
** registered workflow: swf-sns-workflow
** scheduling activity task: get_contact_activity
```

返回另一窗口，活动轮询器运行的位置。此时显示所运行的第一个活动的结果，提示您输入电子邮件或手机短信号码。输入其中一项或两项数据，然后确认所输入的文本。

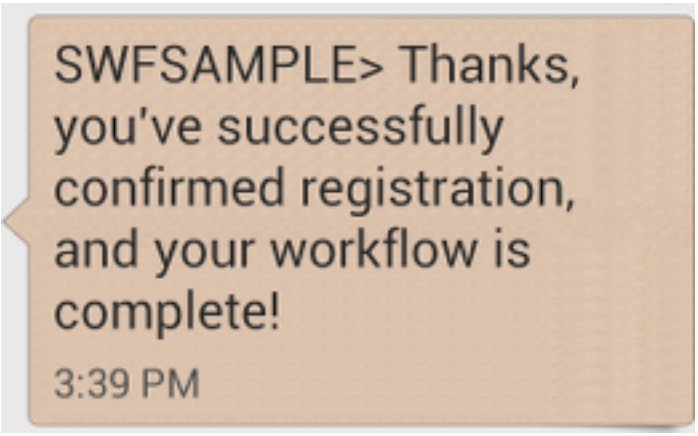
```
activity task received: <AWS::SimpleWorkflow::ActivityTask>
** Starting activity task: get_contact_activity

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**
>
> 此处提供的电话号码为虚构，仅作说明用途。请在此处使用您自己的手机号码和电子邮件地址！

输入此信息后不久，您应该会收到一封来自 Amazon SNS 的电子邮件或短信，要求您确认主题订阅。如果输入了手机短信号码，则将看到手机上显示类似如下内容。



如果向此短信回复 YES，则将获得在 send_result_activity 中提供的响应。



在发生所有这些情况时，您看到命令行窗口中发生了什么情况？ 工作流程和活动轮询器均在努力工作。

以下是来自工作流程轮询器的输出。

```
** 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...
```

以下是来自活动轮询器的输出，同一时间在另一命令行窗口中发生。

```
++ 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!
```

恭喜，您的工作流程已完成，而本教程也已完成！

您可能要再次重新运行工作流程以了解超时如何发挥作用或输入其他数据。请记住，订阅主题后，即已订阅，直到取消订阅为止。在取消订阅主题之前重新运行工作流程可能会自动成功，因为他们`wait_for_confirmation_activity`会看到您的订阅已经得到确认。

从 Amazon SNS 主题取消订阅

- 向该手机短信回复拒绝（发送 STOP）。
- 选择在电子邮件中收到的取消订阅链接。

现已准备就绪，可重新订阅该主题。

我从这里可以继续进行哪些内容？

本教程已经涵盖了很多领域，但是关于亚马逊 SWF 或 Amazon SNS 适用于 Ruby 的 Amazon SDK，你还有很多东西可以学习。有关详细信息和更多示例，请参阅以下各项的官方文档：

- 适用于 Ruby 的 Amazon SDK 文档
- Amazon Simple Notification Service Documentation
- Amazon Simple Workflow Service Documentation

在 Amazon SWF 控制台中工作

Amazon SWF 控制台提供了配置、启动和管理工作流程执行的选项。

使用 Amazon SWF 控制台，您可以：

- 注册工作流程域。
- 注册工作流程类型和活动类型。
- 启动、查看、发出信号、取消、终止和重启工作流程执行。

注册域

工作流程在名为域的 Amazon 资源中运行，该域控制着工作流程的范围。一个 Amazon 账户可以有多个域，每个域可以包含多个工作流，但不同域中的工作流不能交互。

域名注册是控制台最初提供的唯一功能。注册至少一个域名后，您可以对该域执行以下操作：

- 注册工作流程和活动类型。
- 启动工作流程执行。
- 取消、终止并发送信号至正在运行的工作流程执行中。
- 重新启动已关闭的工作流程执行。

您还可以执行域名管理操作，例如弃用和取消删除域名。

启用域之后，无法将其用于创建新工作流执行或注册新工作流。弃用域名也会弃用该域中注册的所有活动和工作流。在域被弃用之前启动的执行将继续运行。

取消先前已弃用的域名后，您可以继续使用该域来注册工作流程类型并开始新的工作流程执行。

有关这些域管理操作的更多信息，请参阅DeprecateDomain和UndeprecateDomain。

注册工作流程类型

注册至少一个域后，您可以在 Amazon SWF 控制台中注册工作流程类型。

工作流类型是一组活动类型，它们执行目标并包含协调活动的逻辑。工作流类型协调和管理可在多个计算设备上异步运行的活动的执行，并具有顺序和并行处理方法。

使用控制台注册 Amazon SWF 工作流程类型

1. 打开您要在其中注册工作流的域。
2. 选择注册，然后选择注册工作流。
3. 在注册工作流页面上，输入工作流名称和工作流版本。或者，您也可以指定默认任务列表，用于为此工作流执行计划决策任务。
4. （可选）选择高级选项，为您的工作流指定以下详细信息：
   - 默认任务优先级 – 分配给工作流的默认任务优先级。
   - 默认执行启动到关闭超时 – 此工作流执行的默认最长持续时间。
   - 默认任务启动到关闭超时 – 此工作流的决策任务的默认最长持续时间。
   - 默认子策略 – 用于子工作流执行的默认策略。
   - 默认 Lambda 角色 – 附加到此工作流的默认 IAM 角色。
5. 选择注册工作流。

注册活动类型

活动是您希望您的工作流程类型协调和执行的任务（例如：验证客户的订单、从信用卡中扣款等）。活动的执行顺序由工作流类型的协调逻辑决定。

注册至少一个域名后，您就可以注册活动类型。

使用控制台注册 Amazon SWF 活动类型

1. 打开您要在其中注册活动的域。
2. 选择注册，然后选择注册活动。
3. 在注册活动页面上，输入活动名称和活动版本。或者，您也可以指定默认任务列表，用于计划此活动的任务。
4. （可选）选择高级选项，为您的活动指定以下详细信息：
   - 默认任务优先级 – 分配给活动的默认任务优先级。
   - 默认任务计划到启动超时 – 此活动的任务在分配给工作线程之前可以等待的默认最长时间。
   - 默认任务启动到关闭超时 – 工作线程处理此活动的任务所需的默认最长时间。
   - 默认任务计划到关闭超时 – 此活动的任务的默认最长持续时间。

- 默认任务心跳超时-处理此类任务的工作器必须通过调用RecordActivityTaskHeartbeat来报告进度的默认最长时间。

5. 选择注册活动。

启动工作流程

您可以从 Amazon SWF 控制台启动工作流执行。除非注册了至少一个工作流类型，否则您不能启动工作流执行。

使用控制台启动工作流程执行

1. 打开 Amazon SWF 控制台，在左侧导航窗格中选择域。
2. 在域名下方，选择工作流。
3. 在工作流页面上，选择要执行的工作流。
4. 选择启动执行。
5. 在开始执行页面上，输入工作流名称和执行 ID，按名称标识您的执行。或者，您也可以指定一个任务列表，用于为该工作流执行生成的决策任务。
6. （可选）选择高级选项，为您的工作流执行指定以下详细信息：

   - 任务优先级 – 执行此工作流时要使用的任务优先级。
   - 执行启动到关闭超时 – 此工作流执行的总持续时间。
   - 任务启动到关闭超时 – 此工作流执行决策任务的最长持续时间。
   - 子策略-此工作流程执行因显式调用TerminateWorkflowExecution操作或超时已过期而终止时，用于执行该工作流程的子工作流程的策略。
   - Lambda 角色 — 要附加到此工作流程执行的 IAM 角色。

7. 选择启动执行。

管理工作流程执行

您可以按名称、状态、ID 和标签筛选工作流程执行。您可以将带有输入的信号发送到活动的工作流程执行中。如果您需要取消或终止工作流程，可以使用 “尝试取消” 选项。取消比终止工作流程执行更可取，因为取消可以让工作流程有机会执行任何清理任务，然后正确关闭。

在控制台中，您可以管理当前正在运行和/或已关闭的工作流程执行。

管理您的工作流执行

1. 打开一个域来管理其工作流执行。
2. 选择查找执行。
3. 在工作流执行页面上，选择按属性筛选执行，然后在属性下选择以下筛选条件之一：

选择	应用此筛选条件
Workflow (工作流程)	选择此筛选条件可列出特定工作流的执行情况。例如，要查看 `fiction-books-order-workflow` 的执行情况，请执行以下操作： 1. 选择工作流。 2. 在运算符下，选择 equals。 3. 在 “工作流程” 下，选择fiction-books-order-workflow。 4. （可选）选择清除筛选条件，删除筛选条件并开始新的执行搜索。
状态	选择此筛选条件可列出具有特定状态的执行。例如，要查看处于失败状态的执行，请执行以下操作： 1. 选择一种状态： 2. 在运算符下，选择 equals。 3. 在状态下，选择失败。 4. （可选）选择清除筛选条件，删除筛选条件并开始新的执行搜索。
执行 ID	选择此筛选条件可根据工作流的 ID 查看工作流执行情况。例如，要查看 ID 为 `fiction-books-order-category1` 的工作流的执行情况，请执行以下操作： 1. 选择执行 ID。 2. 在运算符下，选择 equals。 3. 在 “执行” 下 IDs，选择 fiction-books-order-category1。 4. （可选）选择清除筛选条件，删除筛选条件并开始新的执行搜索。
Tag	选择此筛选条件可列出具有特定标签的执行。例如，要查看处于 `purchaseOrder` 状态的执行，请执行以下操作：

选择	应用此筛选条件
	1. 选择标签。 2. 在运算符下，选择 equals。 3. 在标签下，选择 purchaseOrder。 4. （可选）选择清除筛选条件，删除筛选条件并开始新的执行搜索。

4. （可选）应用所需筛选条件列出工作流执行后，您可以对活跃执行执行以下操作：

   - 信号 – 使用此选项向正在运行的工作流执行发送其他数据。要实现此目的，应按照以下步骤进行：

     1. 选择要向其发送其他数据的执行。
     2. 选择信号，然后在信号执行对话框中指定数据。
     3. 选择信号。

   - 尝试取消 – 使用此选项尝试取消工作流执行。最好的做法是取消工作流执行而不是终止它。取消可使工作流执行有机会执行任何清理任务然后正确关闭任务。

     1. 选择要取消的执行。
     2. 选择尝试取消。

   - 终止 – 使用此选项终止工作流执行。请注意，最好的做法是取消工作流执行而不是终止它。

     1. 选择要终止的执行。
     2. 对于子策略，请确保选中终止。
     3. （可选）指定终止执行的原因和详细信息。
     4. 选择终止。

5. （可选）重新运行 – 使用此选项重新运行已关闭的工作流执行。

   1. 在工作流执行列表中，选择要重新运行的已关闭执行。在=选择已关闭的执行时，系统会启用重新运行按钮。选择重新运行。
   2. 在重新运行执行页面上，指定工作流执行的详细信息，如 启动工作流程 中所述。

Amazon SWF 中的基本工作流程概念

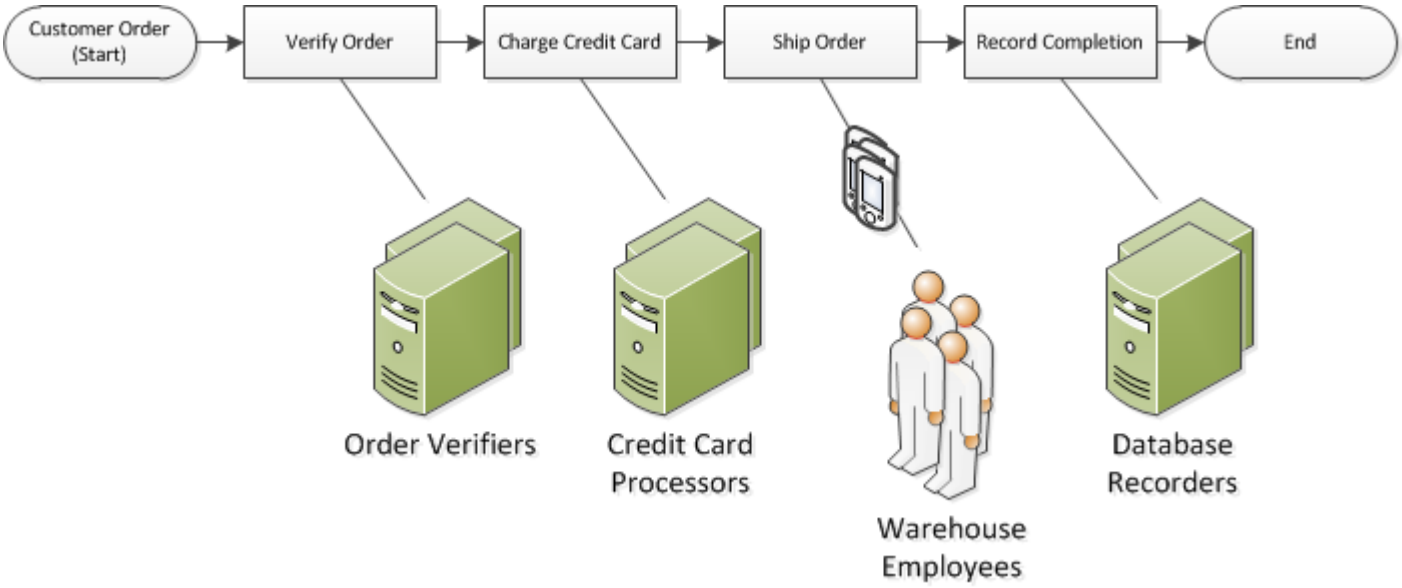
> **Note**
>
> 本章中的概念概述了 Amazon Simple Workflow Service 并介绍了其主要功能。如果您正在寻找示例，请参阅与亚马逊 SWF 合作 APIs。

使用 Amazon Simple Workflow Service (Amazon SWF)，您可以将分布式异步应用程序作为工作流来实现。工作流程对可以跨多个计算设备异步运行并可进行顺序和并行处理的活动的执行进行协调和管理。

在设计工作流程时，需要对应用程序进行分析以识别其组件任务。在 Amazon SWF 中，这些任务由活动表示。活动的执行顺序由工作流程的协作逻辑决定。

电子商务应用程序的工作流程示例

下图显示了涉及人员和自动化流程的电子商务订单处理工作流程：



电子商务应用程序工作流程从客户下单时开始，包括四项任务：

1. 验证订单。
2. 如果订单有效，要求客户付款。
3. 如果付款完成，则按订单发货。

4. 如果发货完成，则保存订单详情。

此工作流程中的任务是顺序 任务：用信用卡付款之前必须验证订单；按订单发货之前必须用信用卡成功付款；而在记录订单之前订单必须已发货。即便如此，这些任务也可以在不同的位置执行，因为 Amazon SWF 支持分布式流程。如果任务本质上可编程，则还可以用不同的编程语言或不同的工具编写这些任务。

除了顺序处理任务外，Amazon SWF 还支持并行处理任务的工作流。并行任务是同时执行的，并且可由不同的应用程序或工作程序来完成。工作流程会作出有关在一个或更多并行任务执行完之后如何继续操作的决策。

其他概念

- 在 Amazon SWF 中创建工作流程
- 在 Amazon SWF 中运行工作流程
- 亚马逊 SWF 中的工作流程历史记录
- 亚马逊 SWF 中的对象标识符
- 亚马逊 SWF 中的域名
- 亚马逊 SWF 中的演员
- 亚马逊 SWF 中的任务
- 亚马逊 SWF 中的任务列表
- 关闭 Amazon SWF 中的工作流程执行
- 亚马逊 SWF 工作流程的生命周期
- 在 Amazon SWF 中轮询任务

在 Amazon SWF 中创建工作流程

创建基本顺序工作流程涉及以下阶段。

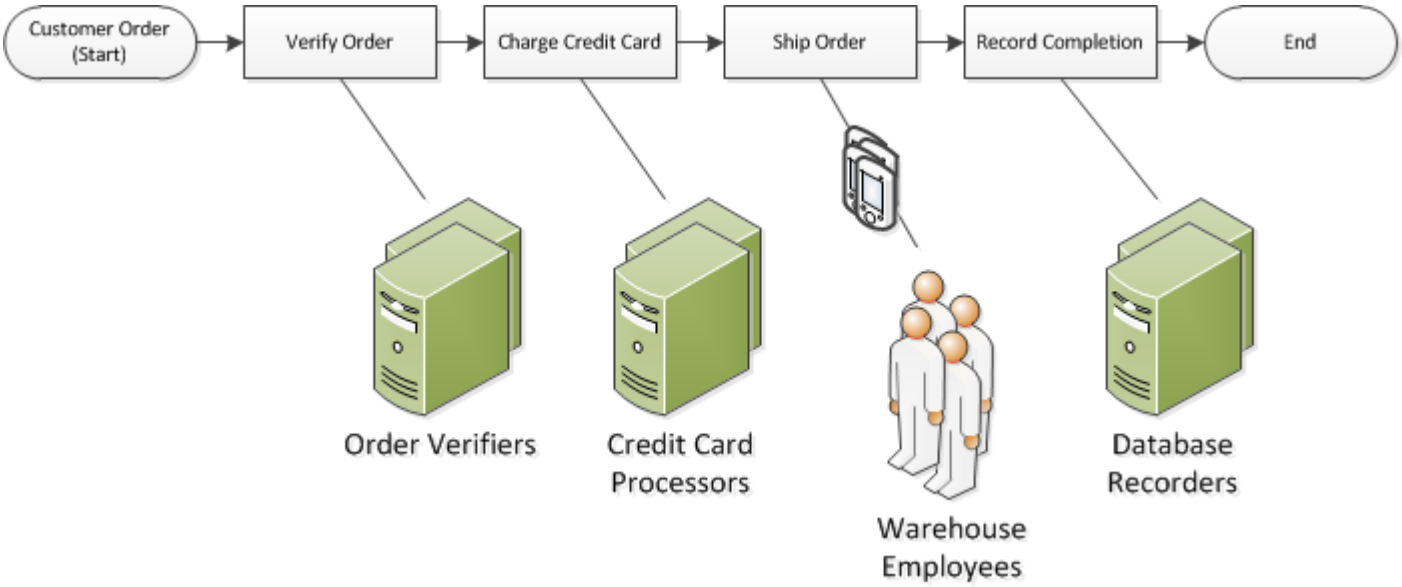
- 建立工作流程模型，注册其类型以及注册其活动类型
- 开发和启动执行活动任务的活动工作程序
- 开发和启动使用工作流程历史决定下一步操作的决策程序
- 开始和启动工作流程启动程序，即启动工作流程执行的应用程序

建立工作流程及其活动的模型

要使用 Amazon SWF，请在应用程序中将逻辑步骤作为活动进行建模。活动表示一个逻辑步骤或工作流程中的一个任务。例如，授权信用卡就是一项活动，其中涉及提供信号卡号码和其它信息、接收信用卡拒绝的授权代码或消息。

除了定义活动之外，您还需要定义处理决策点的协作逻辑。例如，协作逻辑可根据信用卡是被授权还是被拒绝来排定不同的后续活动。

下图显示的是顺序客户订单工作流程的示例，共有四个活动（Verify Order、Charge Credit Card、Ship Order 和 Record Completion）。



在 Amazon SWF 中运行工作流程

完成协调逻辑和活动的设计后，您可在 Amazon SWF 中将这些组件注册为工作流和活动类型。在注册期间，您可以为每种类型指定名称、版本和默认配置值。

只有经过注册的工作流和活动类型才能用于 Amazon SWF。在电子商务示例中，您将注册 CustomerOrder 工作流程类型以及 VerifyOrder ChargeCreditCard、 ShipOrder、和 RecordCompletion活动类型。

注册完您的工作流程类型后，您可以按经常使用的方式运行它。工作流程执行是正在运行的工作流程实例。

工作流程执行可从任何过程或应用程序，甚至是另一个工作流程开始执行。在电子商务示例中，新工作流程执行从每一个客户订单开始。启动工作流程的应用程序类型取决于客户下订单的方式。工作流程可由网站或移动应用程序或使用公司内部应用程序的客户服务代表启动。

使用 Amazon SWF，您可以将名为 `workflowId` 的标识符与您的工作流执行关联起来，这样您就可以将现有业务标识符集成到工作流中。在电子商务示例中，可以使用客户发票编号标识每个工作流程执行。

除了您提供的标识符外，Amazon SWF 还会将系统生成的唯一标识符（即 `runId`）与每个工作流执行关联起来。Amazon SWF 只允许在任何给定时间运行一个具有此标识符的工作流执行；尽管您可以执行多个具有相同工作流类型的工作流，但每个工作流执行都有不同的 `runId`。

亚马逊 SWF 中的工作流程历史记录

Amazon SWF 在工作流程历史记录中记录每个工作流程的执行进度，这是自工作流程执行开始以来发生的每个事件的详细、完整且一致的记录。

事件表示工作流程执行状态的离散变化，例如计划中的新活动或正在完成的正在运行的活动。工作流历史中包含每个导致工作流执行状态更改的事件，如已排定和完成的活动、任务超时和信号。

工作流历史记录中通常不会出现不更改工作流执行状态的操作。例如，工作流历史记录中不显示轮询尝试次数或可见性操作的使用。

工作流历史有几个主要优势：

- 应用程序可以是无状态的，因为有关工作流程执行的所有信息都存储在其工作流程历史记录中。
- 对于每个工作流执行，其历史记录中都包含被排定的活动、其当前状态和结果。工作流执行使用此信息确定下面要执行的步骤。
- 历史记录提供具体的审查跟踪，您可以用它监控正在运行的工作流执行并验证已完成的工作流执行。

以下所示为电子商务工作流历史的概念视图：

```
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
```

前述示例中，订单正等待发货。以下示例中，订单已完成。由于工作流历史是累积形成的，会附加较新的事件：

```
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
```

在编程方面，工作流程执行历史中的事件以 JavaScript 对象表示法 (JSON) 对象的形式表示。历史记录本身是上述数据元的 JSON 阵列。每个事件都有以下内容：

- 一种类型，例如WorkflowExecutionStarted或 ActivityTaskCompleted
- Unix 时间格式的时间戳
- 唯一标识事件用 ID

此外，每种类型的事件都有一组适用于该类型的独特描述性属性。例如，该ActivityTaskCompleted事件的属性包含与活动任务的计划时间和启动时间相对应的事件，还有一个 IDs 用于保存结果数据的属性。

您可以使用GetWorkflowExecutionHistory操作获取工作流程执行历史记录当前状态的副本。此外，作为 Amazon SWF 与工作流决策程序之间交互的一部分，决策程序会定期接收历史记录的副本。

以下部分是 JSON 格式的示例工作流执行历史。

```
[  {
      "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"
   }
]
```

有关工作流程执行历史中可能出现的不同类型事件的详细列表，请参阅《Amazon Simple Workflow Service API 参考》中的HistoryEvent数据类型。

Amazon SWF 会在执行结束后将所有工作流执行的完整历史记录存储可配置的天数。该时期在您为您的工作流注册域时指定，它被称为工作流历史保存期限。本章节的后面部分将更详细地讨论域。

亚马逊 SWF 中的对象标识符

下面的列表描述了如何对 Amazon SWF 对象（如工作流执行）进行唯一标识。

- 工作流类型 – 注册的工作流类型通过其域、名称和版本标识。工作流程类型是在对 `RegisterWorkflowType` 的调用中指定的。
- 活动类型 – 注册的活动类型通过其域、名称和版本标识。活动类型是在对 `RegisterActivityType` 的调用中指定的。

- 决策任务和活动任务 – 每个决策任务和活动任务都通过唯一的任务令牌标识。任务令牌由 Amazon SWF 生成，在 PollForDecisionTask 或 PollForActivityTask 响应中返回，且包含有关该任务的其他信息。虽然令牌最常用于接收任务的过程，但该过程可以将令牌传给另一个过程，然后这个过程会报告任务完成或失败。
- 工作流执行 – 工作流执行通过域、工作流 ID 和运行 ID 标识。前两个是传递给的参数StartWorkflowExecution。运行 ID 由 StartWorkflowExecution 返回。

亚马逊 SWF 中的域名

工作流程在名为域的 Amazon 资源中运行，该域提供了一种在您的账户中限定 Amazon SWF 资源范围 Amazon 的方法。工作流的所有组成部分，如工作流类型和活动类型，都必须指定在一个域中。

一个 Amazon 账户可以有多个域，每个域可以包含多个工作流，但不同域中的工作流不能交互。

设置新工作流时，您需要在设置任何其它工作流组成部分之前注册域，如果您此前没有注册。

注册域时，您需要指定工作流程历史保留期。保留期是指在工作流程执行完成后，Amazon SWF 将继续保留有关工作流程执行的信息的时间长度。

域名注册是控制台最初提供的唯一功能。注册至少一个域名后，您可以对该域执行以下操作：

- 注册工作流程和活动类型。
- 启动工作流程执行。
- 取消、终止并发送信号至正在运行的工作流程执行中。
- 重新启动已关闭的工作流程执行。

您还可以执行域名管理操作，例如弃用和取消删除域名。

启用域之后，无法将其用于创建新工作流执行或注册新工作流。弃用域名也会弃用该域中注册的所有活动和工作流。在域被弃用之前启动的执行将继续运行。

取消先前已弃用的域名后，您可以继续使用该域来注册工作流程类型并开始新的工作流程执行。

有关这些域管理操作的更多信息，请参阅DeprecateDomain和UndeprecateDomain。

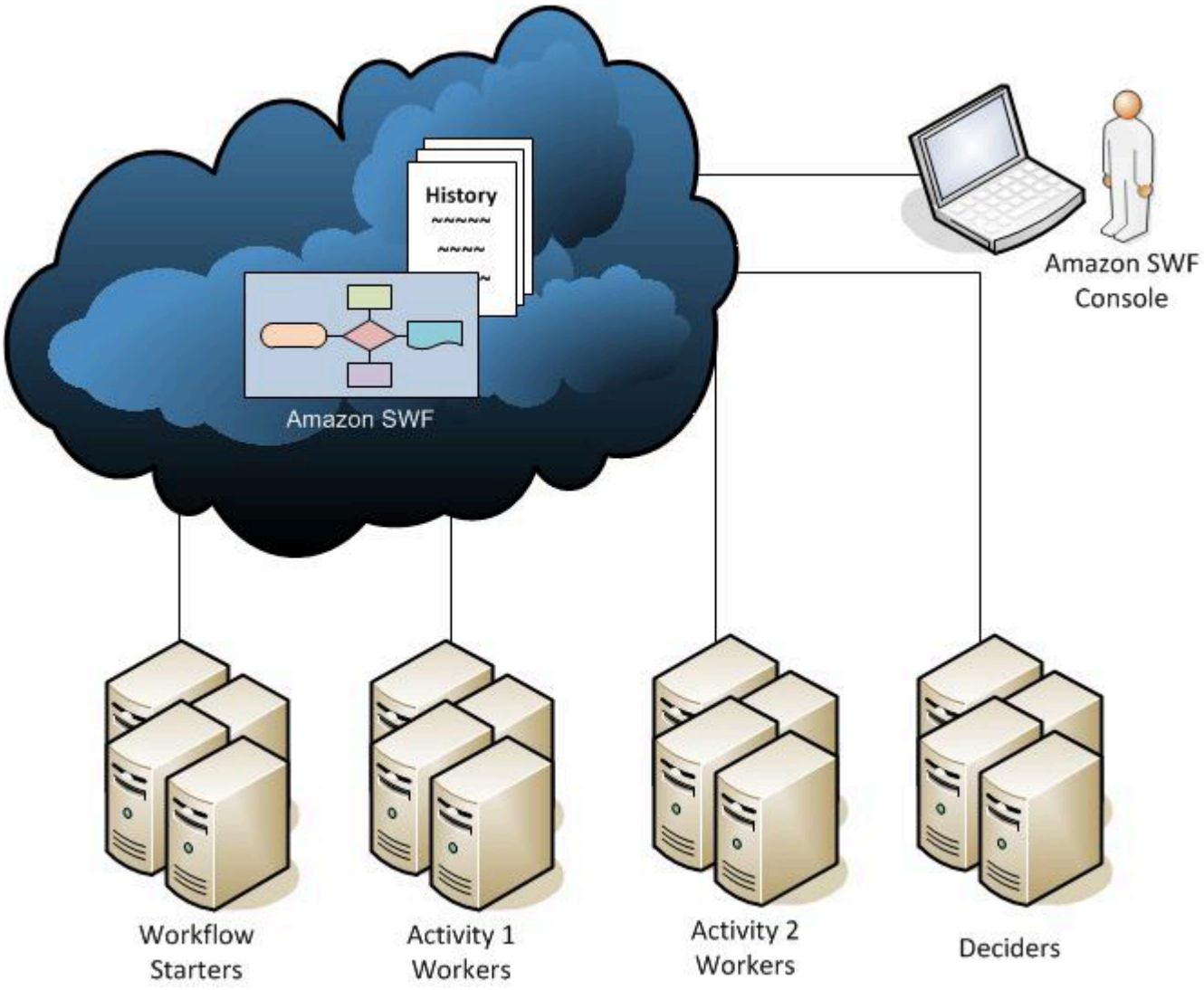
亚马逊 SWF 中的演员

主题

什么是 Amazon SWF 中的操作者

在运行过程中，Amazon SWF 会与许多不同类型的程序操作者进行交互。参与者可以是工作流程启动程序、决策程序或活动工作程序。这些操作者通过其 API 与 Amazon SWF 通信。您可以用任何编程语言开发这些参与者。

下图显示了 Amazon SWF 架构，其中包括 Amazon SWF 及其操作者。



工作流程启动程序

工作流程启动程序是能启动工作流程执行的任何应用程序。在电子商务示例中，一个工作流程启动程序可以是客户下订单所在网站。而另一个工作流程启动程序可以是客户服务代表代表客户下订单时所用移动应用程序或系统。

决策程序

决策程序是工作流程写作逻辑的执行。决策程序控制工作流程执行中的活动任务流程。只要工作流程执行过程中发生更改 (如任务完成)，都会向决策程序传递包含整个工作流程历史记录的决策任务。当决策程序从 Amazon SWF 接收决策任务时，它会分析工作流执行历史记录，以确定工作流执行过程中的下一个相应步骤。决策程序使用决策将这些步骤传递回 Amazon SWF。决策是一种 Amazon SWF 数据类型，用于代表接下来的各种操作。要查看可能的决策列表，请参阅《Amazon Simple Workflow Service API Reference》中的 Decision。

下面是 JSON 格式的决策示例，该决策通过这种格式传送到 Amazon SWF。此决策会排定新活动任务。

```
{
   "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"
   }
}
```

决策程序会在工作流程执行启动时以及每当工作流程执行状态发生更改时接收决策任务。决策程序通过接收决策任务和使用更多决策响应 Amazon SWF 来不断推进工作流执行，直到决策程序确定工作流执

行完成为止。然后，它会回应决策，以关闭工作流程执行。工作流执行关闭后，Amazon SWF 不会为该执行安排其他任务。

在电子商务示例中，决策程序会决定每个步骤是否都适当执行，然后回排定下一步或管理任何错误条件。

决策程序代表的是单个计算过程或线程。多个决策程序可为同一种工作流程类型处理任务。

活动工作程序

活动工作程序是一种过程或线程，用于执行活动任务这一工作流程组成部分。活动任务表示的是您在应用程序中标识的任务之一。

要在工作流程中使用活动任务，必须使用 Amazon SWF 控制台或操作对其进行注册。RegisterActivityType

每个活动工作人员都会轮询 Amazon SWF 以获取适合该活动工作线程执行的新任务；某些任务只能由特定活动工作线程执行。接收任务后，活动工作线程处理该任务直至完成，然后向 Amazon SWF 报告该任务已完成并提供结果。然后，活动工作程序会轮询新任务。与活动流程执行相关的活动工作程序会用此方式继续处理任务，直到工作流程执行本身已完成。在电子商务示例中，活动工作程序是信用卡处理人员和物流人员等执行过程中单独步骤的人所使用的独立过程和应用程序。

活动工作程序表示单个计算过程（或线程）。多个活动工作程序可处理同一种活动类型的任务。

参与者之间的数据交换

工作流程执行启动时会提供输入数据到工作流程执行。同样地，当活动工作程序排定活动任务时会提供输入数据给工作程序。活动任务完成后，活动工作线程可将结果返回到 Amazon SWF。同样地，决策程序会在执行完成时报告工作流程执行的结果。每个操作者都可以通过用户自定义形式的字符串将数据发送到 Amazon SWF 或从 Amazon SWF 接收数据。根据数据的大小和敏感性，您可以直接传递数据，或传递指向存储在其他系统或服务上的数据的指针（如 Amazon S3 或 DynamoDB）。直接传递的数据和指向其他数据存储的指针都会记录在工作流执行历史记录中；但是，Amazon SWF 不会从外部存储复制或缓存任何数据作为历史记录的一部分。

由于 Amazon SWF 会维护每个工作流执行的完整执行状态（包括任务的输入和结果），所有操作者都可能是无状态的。因此，工作流程处理具有高度可扩展。随着系统负载增加，您可以简单增加更多参与者来提高容量。

亚马逊 SWF 中的任务

Amazon SWF 通过为活动工作线程和决策程序提供工作（称为“任务”）来与它们交互。Amazon SWF 中有三种类型的任务：

- 活动任务 – 活动任务指示活动工作线程执行其功能，如检查库存或信用卡计费。活动任务中包含活动工作程序执行其功能需要的所有信息。
- Lambda 任务 – Lambda 任务与活动任务类似，但它执行 Lambda 函数而不是传统的 Amazon SWF 活动。有关如何定义 Lambda 任务的更多信息，请参阅 Amazon Lambda 亚马逊 SWF 中的任务。
- 决策任务 – 决策任务向决策程序告知工作流执行状态已更改，以便决策程序确定下一个需要执行的活动。决策任务包含当前工作流程历史。

Amazon SWF 会在工作流启动时以及工作流状态发生变化时（如活动任务完成时）调度决策任务。每个决策任务中都包含整个工作流程执行历史标有页码的视图。决策程序会分析工作流执行历史，并向 Amazon SWF 回应一组决策，这些决策指定了工作流执行中接下来应完成的操作。实质上，每个决策任务都为决策程序提供了评估工作流以及向 Amazon SWF 提供指示的机会。

为确保处理的决策没有冲突，Amazon SWF 会将每个决策任务分配给一个决策程序，且一次只允许一个决策任务在工作流执行中处于活动状态。

下表显示的是与工作流程相关的不同构建与决策程序之间的关系。

逻辑设计	注册为	执行人	接收和执行	生成
工作流	工作流类型	决策程序	决策任务	决策

活动工作线程完成活动任务后，会向 Amazon SWF 报告该任务已经完成，并包括生成的任何相关结果。Amazon SWF 会使用一个表示任务已完成的事件来更新工作流执行历史记录，然后调度一个决策任务，以将更新的历史记录传输给决策程序。

Amazon SWF 会将每个活动任务分配给一个活动工作线程。一旦任务分配后，其他活动工作程序就不能认领或执行该任务。

下图所示为与活动相关的不同构建之间的关系。

逻辑设计	注册为	执行人	接收和执行	生成
活动	活动类型	活动工作程序	个活动任务	结果数据

亚马逊 SWF 中的任务列表

任务列表提供与工作流程相关的各种任务的组织方法。可以认为任务列表与动态队列相似。在 Amazon SWF 中安排任务时，您可以指定容纳任务的队列（任务列表）。同样，当您轮询 Amazon SWF 以获取任务时，您可以指定从哪个队列（任务列表）获取任务。

任务列表提供了灵活的机制，以在您的使用案例需要时将任务路由至工作程序。任务列表是动态的，在列表中，您不需要注册任务列表或通过一个操作明确创建列表：如果列表不存在，只需要排定任务就可创建任务列表。

活动任务和决策任务的列表是分开的。始终只在一个任务列表上排定任务；不能跨过列表共享任务。此外，与活动和工作流程一样，任务列表的范围仅限于特定 Amazon 区域和 Amazon SWF 域。

主题

- 决策任务列表
- 活动任务列表
- 任务路由

决策任务列表

每个工作流程执行都与一个特定的决策任务列表相关。注册工作流程类型（RegisterWorkflowType操作）后，您可以为该工作流程类型的执行指定默认任务列表。当工作流程启动程序启动工作流程执行时（`StartWorkflowExecution` 操作），它可以选择为该工作流程执行指定不同的任务列表。

当决策程序轮询新决策任务（`PollForDecisionTask` 操作）时，决策程序可以指定决策任务列表以从中获取任务。一个决策程序可以通过多次调用 `PollForDecisionTask`、在每一个调用中使用不同的任务列表，以服务多个工作流程执行，在调用中，每个任务列表都指定了一个特定的工作流程执行。或者，决策程序可以轮询为多个工作流程执行提供决策任务的一个决策任务列表。您还可以使多个决策程序一起轮询一个工作流程执行的任务列表，从而使它们全部服务该工作流程执行。

活动任务列表

一个活动任务列表中可包含不同活动类型的任务。任务按顺序排列在任务列表中。Amazon SWF 会尽最大努力按顺序返回列表中的任务。某些情况下，任务可能不会按序出现在列表中。

注册活动类型（RegisterActivityType操作）后，您可以为该活动类型指定默认任务列表。默认情况下，此类活动任务将在指定的任务列表中进行安排；但是，当决策者计划活动任务（ScheduleActivityTask决策）时，决策者可以选择指定不同的任务列表来安排任务。如果决策程序没有指定任务列表，则使用默认任务列表。因此，您可以根据任务属性将活动任务置放在特定任务列表中。例如，您可以将给定信用卡类型的活动任务的所有实例置放到特定任务列表中。

任务路由

当活动工作人员轮询新任务（PollForActivityTask操作）时，它可以指定要从中提取的活动任务列表。如果指定了列表，活动工作程序将只能从该列表中接受任务。用此方法，您可以确保特定任务只会分配给特定的活动工作程序。例如，您可以创建任务列表以保存需要使用高性能计算机的任务。只有运行适当硬件的活动工作程序才能轮询该任务列表。另一个示例是，为某个特定地理区域创建一个任务列表。然后，您可以确保只有在该地区部署的工作程序才能接收这些任务。或者，您可以为高优先级订单创建一个任务列表，并始终优先检查该列表。

用此方法将特定任务分配给特定活动工作程序的操作被称为任务路由。任务路由是可选操作；如果没有在安排活动任务时指定任务列表，则任务自动置于默认任务列表中。

关闭 Amazon SWF 中的工作流程执行

启动工作流程执行后，该工作流程即处于打开状态。开启的工作流程执行可以在完成、取消、失败或超时的情况下终止。它还可以继续作为新执行，或可以被终止。工作流执行可由决策程序、管理该工作流的人员或 Amazon SWF 来关闭。

如果决策程序确定工作流程活动已关闭，它应该使用 `RespondDecisionTaskCompleted` 操作并传出 `CompleteWorkflowExecution` 决策终止完成状态的工作流程执行。

或者，决策程序还可以终止取消或失败状态下的工作流程执行。若要取消执行，决策程序应使用 `RespondDecisionTaskCompleted` 操作并传出 `CancelWorkflowExecution` 决策。

如果工作流程执行进入到正常完成范围之外的状态，决策程序应舍弃该工作流程执行。若要舍弃执行，决策程序应使用 `RespondDecisionTaskCompleted` 操作并传出 `FailWorkflowExecution` 决策。

Amazon SWF 会监控工作流的执行，确保它们不会超过用户指定的任何超时设置。如果工作流执行超时，Amazon SWF 会自动将其关闭。有关超时值的更多信息，请参阅 Amazon SWF 超时类型 一节。

决策程序可能还要终止执行，并使用 RespondDecisionTaskCompleted 操作和传出 ContinueAsNewWorkflowExecution 决策来将其在逻辑上继续以新执行进行操作。这对于长时间运行的工作流程执行是一个非常有用的策略，因为这种工作流程执行的历史可能会随着时间的推移变得很大。

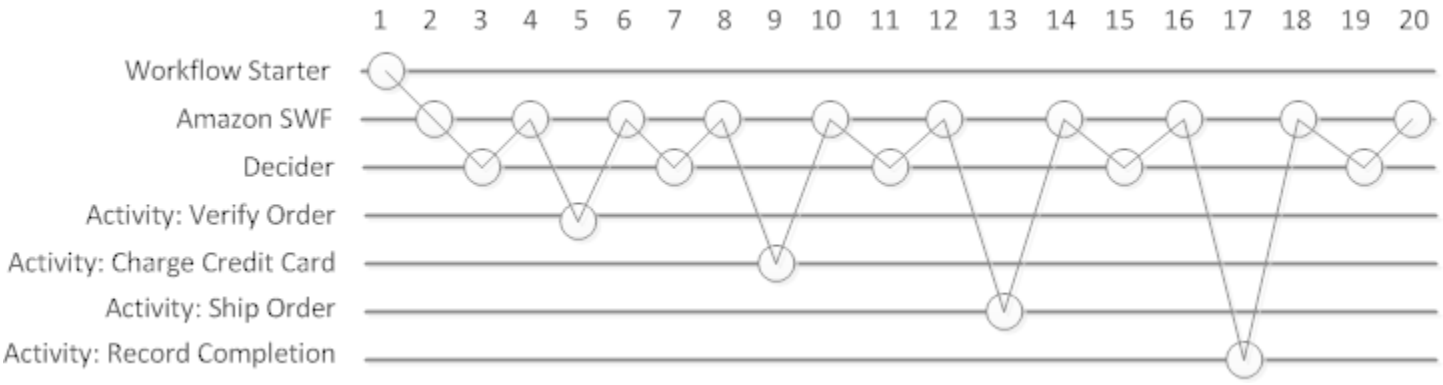
最后，您可以直接通过 Amazon SWF 控制台或使用 TerminateWorkflowExecution API 以编程方式终止工作流执行。终止操作会迫使工作流程执行关闭。取消优先于终止，因为您的决策程序可以管理工作流程执行的终止。

如果执行超过某些服务定义的限制，Amazon SWF 会终止该工作流的执行。如果父工作流已终止且适用的子策略指示子工作流也应终止，则 Amazon SWF 会终止子工作流。

亚马逊 SWF 工作流程的生命周期

从工作流执行开始到完成的过程中，Amazon SWF 通过向操作者分配适当的任务（活动任务或决策任务）来与其进行交互。

下图从工作流程上所执行组件的角度显示了订单处理工作流程执行的生命周期。



工作流程执行生命周期

下表解释了上图中的每个任务。

描述	操作、决策或事件
1. 工作流启动程序调用适当的 Amazon	StartWorkflowExecution action。

描述	操作、决策或事件
SWF 操作来启动订单的工作流执行，从而提供订单信息。	
2. Amazon SWF 接收启动工作流执行请求，然后调度第一个决策任务。	WorkflowExecutionStarted 事件和 DecisionTaskScheduled 事件。
3. 决策程序从 Amazon SWF 接收任务，查看历史记录，应用协调逻辑以确定之前未发生任何活动，然后做出安排 Verify Order 活动的决策，并提供活动工作线程处理任务所需的信息，最后将决策返回给 Amazon SWF。	PollForDecisionTask 操作。RespondDecisionTaskCompleted 操作和 ScheduleActivityTask 决策。
4. Amazon SWF 接收决策，安排 Verify Order 活动任务，并等待该活动任务完成或超时。	ActivityTaskScheduled 事件
5. 能够执行 Verify Order 活动的活动工作线程接收并执行该任务，将结果返回给 Amazon SWF。	PollForActivityTask 操作和 RespondActivityTaskCompleted 操作。

描述	操作、决策或事件
6. Amazon SWF 接收 Verify Order 活动的结果，将结果添加到工作流历史记录，并安排决策任务。	ActivityTaskCompleted 事件和 DecisionTaskScheduled 事件。
7. 决策者从 Amazon SWF 接收任务，查看历史记录，应用协调逻辑，使用活动工作人员处理任务所需的信息做出安排 ChargeCreditCard活动任务的决定，然后将决策返回给 Amazon SWF。	PollForDecisionTask 操作。RespondDecisionTaskCompleted 操作，带 ScheduleActivityTask 决策。
8. Amazon SWF 收到决策，安排 ChargeCreditCard 活动任务，然后等待其完成或超时。	DecisionTaskCompleted 事件和 ActivityTaskScheduled 事件。
9. 可以执行活动的活动工作人员接收并执行任务，然后将结果返回给 Amazon SWF。 ChargeCreditCard	PollForActivityTask 和 RespondActivityTaskCompleted 操作。

描述	操作、决策或事件
10. Amazon SWF 接收 ChargeCreditCard 活动任务的结果，将其添加到工作流程历史记录中，并安排决策任务。	ActivityTaskCompleted 事件和 DecisionTaskScheduled 事件。
11. 决策者从 Amazon SWF 接收任务，查看历史记录，应用协调逻辑，使用活动工作人员执行任务所需的信息做出安排 ShipOrder 活动任务的决定，然后将决策返回给 Amazon SWF。	PollForDecisionTask 操作。RespondDecisionTaskCompleted，带 ScheduleActivityTask 决策。
12. Amazon SWF 收到决策，安排 ShipOrder 活动任务，然后等待其完成或超时。	DecisionTaskCompleted 事件和 ActivityTaskScheduled 事件。
13. 可以执行活动的活动工作人员接收并执行任务，然后将结果返回给 Amazon SWF。 ShipOrder	PollForActivityTask 操作和 RespondActivityTaskCompleted 操作。
14. Amazon SWF 接收 ShipOrder 活动任务的结果，将其添加到工作流程历史记录中，并安排决策任务。	ActivityTaskCompleted 事件和 DecisionTaskScheduled 事件。

描述	操作、决策或事件
15. 决策者从 Amazon SWF 接收任务，查看历史记录，应用协调逻辑，使用活动工作人员执行任务所需的信息做出安排 RecordCompletion活动任务的决定，然后将决策返回给 Amazon SWF。	PollForDecisionTask 操作。RespondDecisionTaskCompleted 操作，带 ScheduleActivityTask 决策。
16. Amazon SWF 收到决策，安排 RecordCompletion 活动任务，然后等待其完成或超时。	DecisionTaskCompleted 事件和 ActivityTaskScheduled 事件。
17. 可以执行活动的活动工作人员接收并执行任务，然后将结果返回给 Amazon SWF。 RecordCompletion	PollForActivityTask 操作和 RespondActivityTaskCompleted 操作。
18。Amazon SWF 接收 RecordCompletion 活动任务的结果，将其添加到工作流程历史记录中，并安排决策任务。	ActivityTaskCompleted 事件和 DecisionTaskScheduled 事件。

描述	操作、决策或事件
19. 决策程序从 Amazon SWF 接收任务，检查历史记录，应用协调逻辑，做出决策以关闭工作流执行，并将决策与所有结果一起返回给 Amazon SWF。	PollForDecisionTask 操作。RespondDecisionTaskCompleted 操作，带 CompleteWorkflowExecution 决策。
20。Amazon SWF 关闭工作流执行，并将历史记录归档以供将来参考。	WorkflowExecutionCompleted event。

在 Amazon SWF 中轮询任务

决策程序和活动工作线程使用长轮询与 Amazon SWF 进行通信。决策程序或活动工作线程会定期启动与 Amazon SWF 的通信，通知 Amazon SWF 它可以接受任务，然后指定用于获取任务的任务列表。

如果任务位于指定任务列表中，Amazon SWF 会立即在响应中返回该任务。如果没有提供任务，Amazon SWF 将保持 TCP 连接打开最长 60 秒，这样，如果任务在此时间内变为可用，就可以在同一连接中返回该任务。如果 60 秒内没有提供任务，则会返回空响应并结束连接。（空响应为 Task 结构，其中的 taskToken 值为空字符串。）如果发生这种情况，决策程序或活动工作线程应重新轮询。

长时间轮询对大容量任务处理有效。决策程序和活动工作线程可管理自己的容量，当决策程序和活动工作线程处于防火墙后时使用方便。

有关更多信息，请参阅轮询决策任务和轮询活动任务。

Amazon SWF 中的高级工作流程概念

??? 章节中的电子商务示例表示的是简单的工作流场景。事实上，您可能希望您的工作流执行并行任务（在核准信用卡的同时发送订单确认）、记录主要事件（所有项目都组到一起）、更新订单更改（增加或删除项目）以及进行其它更多高级决定作为工作流执行的一部分。本节介绍可用于构建工作流程的高级工作流程概念。

高级概念

- 版本控制
- 信号
- Amazon SWF 中的子工作流程
- 亚马逊 SWF 中的标记
- 亚马逊 SWF 中的标签
- 使用 Amazon SWF 实现独家选择
- 亚马逊 SWF 中的计时器
- 取消亚马逊 SWF 中的活动任务

版本控制

业务通常需要您对同一工作流采用不同的实现方式或变化，或者需要同时运行多个活动。例如，您可能想要在另一个工作流处于生产中时测试新执行的工作流。您还可能想要用两种不同的功能集运行两个不同的执行，如基本和高级执行。您可以通过版本控制同时执行多个工作流和活动，以满足您的要求。

工作流和活动类型都有一个与之相关的版本，在注册时指定。版本是自由格式的字符串，您可以选择您自己的版本控制方案。为了能够创建某个已注册类型的新版本，您应该用同一名称和不同版本进行注册。前面所述的 亚马逊 SWF 中的任务列表 可进一步帮助您执行版本控制。考虑这样一个情形，您的给定类型长期运行工作流执行正在进程中，并考虑需要您修改工作流（如增加新功能）的情况。您可以通过创建新版本的活动类型和工作线程以及新决策程序。然后，您可以使用不同组的任务列表启动新工作流版本的执行。通过这种方式，您可以同时运行不同版本工作流的执行，而不会相互造成影响。

信号

信号可使您将信息插入正在运行的工作流执行中。在某些应用场景中，建议您将信息增加到正在运行的工作流执行中，使其了解有些情况发生了改变或向其通知外部事件。任何过程都能发送信号到开启的工作流执行中。例如，一个工作流执行可向另一个发送信号。

> **Note**
>
> 尝试将信号发送到未打开的工作流程执行会导致 SignalWorkflowExecution 失败，出现 UnknownResourceFault 错误。

要使用信号，请定义信号名称和要传递给信号的数据（如果有）。然后，对决策者进行编程，使其识别历史记录中的信号事件 (WorkflowExecutionSignaled) 并对其进行适当的处理。当流程想要发出工作流程执行信号时，它会调用 Amazon SWF（使用SignalWorkflowExecution操作，或者如果是决策者，则使用SignalExternalWorkflowExecution决策），指定目标工作流程执行的标识符、信号名称和信号数据。然后，Amazon SWF 会接收信号，将其记录在目标工作流执行的历史记录中，并为其安排决策任务。当决策程序收到决策任务时，它还会接收到工作流执行历史内部的信号。然后，决策程序可根据信号及其数据采取适当操作。

有时，您可能需要等待信号。例如，用户可以通过发送信号取消订单，但只能在下单后一小时内取消订单。Amazon SWF 没有允许决策程序等待服务信号的基本功能。暂停功能需要决策程序自行执行。决策程序应使用 StartTimer 决策启动计时器来执行暂停功能，通过计时器指定决策程序在继续轮询决策任务时等待信号的持续时间。当决策程序收到决策任务时，它应该检查历史记录以查看信号是否已收到或计时器是否已启动。如果已收到信号，则决策程序应取消计时器。然而，如果未收到信号，计时器已启动，则表示信号未在指定时间内到达。简而言之，请执行以下操作来等待特定信号。

1. 为决策程序应等待的时间量创建计时器。
2. 收到决策任务时，检查历史记录以查看信号是否已到达或计时器是否已启动。
3. 如果信号已到达，请使用 CancelTimer 决策来取消计时器并处理该信号。历史记录中可能包含 TimerFired 和 WorkflowExecutionSignaled 事件，这取决于定时情况。在这种情况下，您可以依据历史记录中事件的相对顺序来确定首先发生的事件。
4. 收到信号前，如果计时器已启动，则决策程序等待信号超时。您可以放弃执行或执行适合您的使用案例的其它任何逻辑。

对于应取消工作流的情况（例如，订单本身已被客户取消）应使用 RequestCancelWorkflowExecution 操作，而不是向工作流发送信号。

以下是信号的一些应用情况：

- 暂停工作流执行的进展，直到收到信号（例如，等待库存发货）。
- 向工作流执行提供可能会影响决策程序做决策之逻辑的信息。这对受外部事件（例如，尝试在闭市后完成股票销售）影响的工作流很有用。

- 在您期望发生更改时（例如，在下订单后发货之前更改订单数量）更新工作流执行。

在下面的示例中，工作流执行收到了取消订单的信号。

```
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"}
```

如果工作流执行收到信号，Amazon SWF 将返回一个成功的 HTTP 响应，类似于如下内容。Amazon SWF 将生成一个决策任务，通知决策程序处理信号。

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

Amazon SWF 中的子工作流程

复杂的工作流可使用子工作流分解为更小、更可管理且有可能被重用的组件。子工作流是被另一个（上层）工作流执行启动的工作流执行。要启动子工作流，上层工作流的决策程序应使用 `StartChildWorkflowExecution` 决策。用此决策指定的输入数据可通过其历史记录提供给子工作流。

`StartChildWorkflowExecution` 决策的属性还会指定子策略，即 Amazon SWF 对父工作流执行在子工作流执行之前终止这种情况的处理方式。可能存在以下三种值：

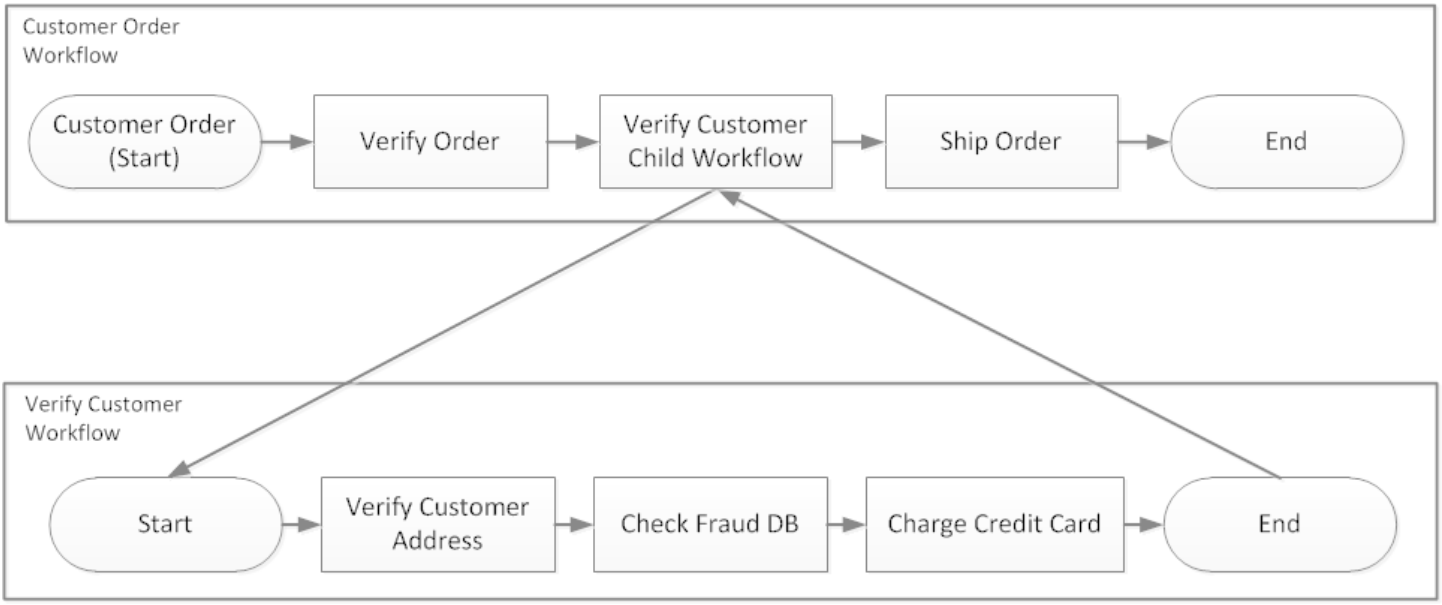
- TERMINATE；Amazon SWF 将终止子执行。
- REQUEST_CANCEL：Amazon SWF 将通过在子工作流执行历史中放置 `WorkflowExecutionCancelRequested` 事件来尝试取消子执行。
- ABANDON：Amazon SWF 将不采取任何操作；子执行将继续运行。

子工作流执行启动后，其运行方式类似于常规执行。执行完成时，Amazon SWF 会将完成情况及其结果记录在父工作流执行的工作流历史中。子工作流的示例包括以下内容：

- 在不同网站中被工作流所使用的处理子工作流的信号卡
- 用于验证客户电子邮件地址、检查退出列表、发送电子邮件以及验证电子邮件未被退回或发送失败的电子邮件子工作流。
- 将连接、设置、交易和验证进行组合的数据库存储和检索子工作流。
- 将构建、包装和验证相结合的源代码编译子工作流。

在电子商务示例中，建议您将 Charge Credit Card 活动设为子工作流。为了实现这一操作，您可以注册一个新的 Verify Customer 工作流、注册 Verify Customer Address 和 Check Fraud DB 活动，并定义任务的协同逻辑。然后，Customer Order 工作流的决策程序可通过排定 `StartChildWorkflowExecution` 决策来指定此工作流类型，从而启动 Verify Customer 子工作流。

下图所示为客户订单工作流，其中包括一个新的 Verify Customer 子工作流，用于检验客户地址、欺诈数据库和用信用卡付费。



多个工作流可使用同一种工作流类型创建子工作流执行。例如，Verify Customer 子工作流还能用在机构的其它部分。子工作流的事件包含在其自己的工作流历史中，而不包含在上层工作流历史中。

由于子工作流仅是由决策程序启动的工作流执行，所以它们还可作为正常的独立工作流执行启动。

亚马逊 SWF 中的标记

有时，建议您将信息记录在您的使用案例之特定工作流执行的工作流历史中。您可以通过标记将信息记录在工作流执行历史中，从而使您能将其用于任何自定义和适合特定情景的目的。

要使用标记，决策者需要使用 RecordMarker 决策，命名标记，将所需的数据附加到决策中，然后使用操作通知 Amazon SWF。RespondDecisionTaskCompletedAmazon SWF 将接收请求，在工作流历史记录中记录标记，并在请求中做出任何其他决策。从那一刻其，决策程序就能在工作流历史中查看标记并通过您编程所用任何方式使用该标记。

记录标记本身不会启动决策任务。为了防止工作流执行被卡住，必须执行一些继续工作流执行的操作。举例来说，这些操作可能包括决策程序排定另一个活动任务、接收信号的工作流执行或之前排定的活动任务完成。

以下为标记示例：

- 用于对递归工作流中的回路数进行计数的计数器。
- 根据活动结果进行的工作流执行的进度。
- 从较早的工作流历史事件总结的信息。

在电子商务示例中，您可以增加按日检查库存以及每次递增标记计数的活动。然后，您可以增加决策逻辑，以在计数超出五时发送电子邮件给客户或通知管理人员，无需审查整个历史记录。

在以下示例中，决策程序完成了一个决策任务并对包含 RecordMarker 决策的 RespondDecisionTaskCompleted 操作做出响应。

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

成功记录该标记后，Amazon SWF 将返回一个成功的 HTTP 响应，类似于如下内容。

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

亚马逊 SWF 中的标签

Amazon SWF 支持对工作流执行加标签。这在您拥有许多资源时尤其有用。

Amazon SWF 支持为一个工作流执行添加最多加五个标签。每个标签都是自由格式的字符串，长度最多为 256 个字符。如果您想使用标签，您必须在启动工作流执行时分配标签。工作流程执行启动之后不能向其增加标签，也不能编辑或删除已分配给工作流执行的标签。

IAM 支持根据标签控制对 Amazon SWF 域的访问。要基于标签控制访问，请在 IAM 策略的条件元素中提供有关您的标签的信息。

管理标签

使用 Amazon SDKs 或直接与 Amazon SWF API 交互来管理亚马逊简单工作流程服务标签。通过使用 API，您可以在注册域时添加标签，列出现有域的标签，以及添加或删除现有域的标签。

> **Note**
>
> 每个资源最多有 50 个标签。请参阅 Amazon SWF 的一般账户限额。

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

有关更多信息，请参阅 与亚马逊 SWF 合作 APIs 以及 Amazon Simple Workflow Service API Reference。

标记工作流程执行

借助 Amazon SWF，您可以将标签与工作流程执行关联起来，然后根据这些标签查询工作流程执行情况。当你使用可见性操作时，你可以筛选列表。通过仔细选择分配给执行的标签，您可以使用它们来提供有意义的清单。

举例来说，假设您在运行多个订单履行中心。使用标签，您可以列出特定运营中心发生的流程。或者，如果客户正在转换不同类型的媒体文件，则标签可能表示转换视频、音频和图像文件时的不同过程。

在您使用 StartWorkflowExecution 操作、StartChildWorkflowExecution 决策或者 ContinueAsNewWorkflowExecution 决策开始执行时，您最多可以将五个标签与工作流程执行关联起来。当您使用可见性操作列出或计算工作流程执行次数时，您可以根据标签筛选结果。

若要使用加标签

1. 修改加标签策略。考虑您的业务要求并创建一列对您有意义的标签。决定哪种执行获得哪种标签。虽然一种执行最多能分配五个标签，但您的标签库中可以有任何数量的标签。由于每种标签可以是长度最多为 256 个字符的任何字符串值，标签几乎可以描述所有的业务理念。
2. 在您创建执行时，给执行最多加上五个标签。
3. 用 、、ListOpenWorkflowExecutions 和 ListClosedWorkflowExecutions 操作指定 CountOpenWorkflowExecutionstagFilterCountClosedWorkflowExecutions 参数，从而列出或计数加有特定标签的执行。该操作将根据指定的标签筛选执行。

将标签与工作流执行关联时，标签与该执行永久关联，且不可删除。

您可以在带 ListWorkflowExecutions 的 tagFilter 参数中只指定一个标签。同时，标签匹配区分大小写，只有完全匹配才能返回结果。

假设您已经建立了两个加有以下标签的执行。

执行名称	分配的标签
第一次执行	消费者，2011 年 2 月
第二次执行	批发，2011 年 3 月

您可以筛选由消费者标签上的 ListOpenWorkflowExecutions 返回的执行列表。oldestDate 和 latestDate 值被指定为 Unix Time 值。

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

```
    "tagFilter":{
      "tag":"Consumer"
      }
  }
```

使用标签控制对域名的访问权限

您可以通过在 IAM 中引用与 Amazon SWF 域关联的标签来控制对 Amazon Simple Workflow Service 域的访问。例如，您可以限制包含密钥为、值为`production`：的标签的域名 `environment`

```
{
      "Version": "2012-10-17",
      "Statement": [
          {
              "Effect": "Deny",
              "Action": "swf:*",
              "Resource": "arn:aws:swf:*:123456789012:/domain/*",
              "Condition": {
                  "StringEquals": {"aws:ResourceTag/environment": "production"}
              }
          }
      ]
  }
```

此策略将`Deny`访问任何已标记为 `environment/production` 的域。

有关更多信息，请参阅：

- 使用 IAM 标签控制访问
- 基于标签的策略

使用 Amazon SWF 实现独家选择

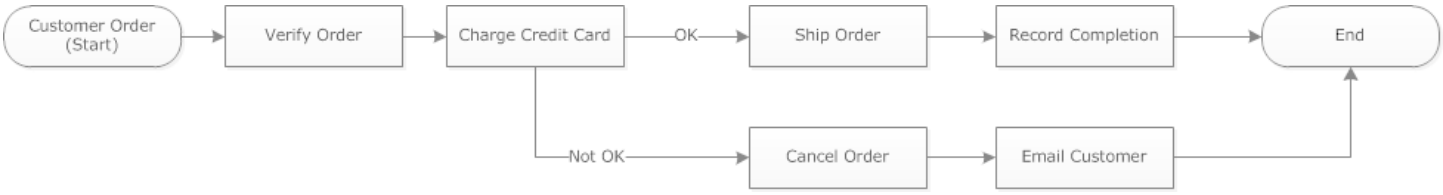
在某些应用场景中，建议您根据前一个活动的结果排定不同组活动。使用独家选择模式，您可以创建灵活的工作流程，以满足应用程序的复杂要求。

Amazon SWF 没有特定的独家选择行动。要实现排他性选择，您必须编写决策者逻辑，以便根据先前活动的结果做出决策。以下为适用于排他选择的某些应用程序：

- 如果前一个活动的结果不成功，执行清除活动

- 根据客户购买的是基本还是高级计划来排定不同的活动
- 根据客户订单历史执行不同的客户身份验证活动

在电子商务示例中，您可以使用排他选择根据信用卡付费结果给订单发货或取消订单。在下图中，如果信用卡付费成功，决策程序将排定 Ship Order 和 Record Completion 活动任务。否则，它将排定 Cancel Order 和 Email Customer 活动任务。



如果信用卡成功付费，决策程序将排定 `ShipOrder` 活动。否则，决策程序将排定 `CancelOrder` 活动。

在这种情况下，给决策程序编程以解析历史记录并决定信用卡是否付费成功。要实现此操作，您要有与下列内容相类似的逻辑

```
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
```

如果信用卡付费成功，决策程序应响应 RespondDecisionTaskCompleted 以排定 ShipOrder 活动。

```
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"
          }
      }
  ]
}
```

如果信用卡未付费成功，决策程序应响应 RespondDecisionTaskCompleted 以排定 CancelOrder 活动。

```
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"
          }
      }
  ]
}
```

如果 Amazon SWF 能够验证 RespondDecisionTaskCompleted 操作中的数据，Amazon SWF 将返回与下面类似的成功 HTTP 响应。

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

亚马逊 SWF 中的计时器

使用计时器，您可以在经过一定时间后通知您的决策者。

响应决策任务时，决策程序可以选择响应 StartTimer 决策。此决策将指定时间量，在此之后，计时器应启动。指定时间过后，Amazon SWF 将向工作流执行历史记录添加一个 TimerFired 事件，并安

排决策任务。然后，决策程序将使用此信息通知进一步决策。计时器的一项常见应用为延迟活动任务的执行。例如，客户可能想延迟交付项目。

取消亚马逊 SWF 中的活动任务

取消活动任务会通知决策者结束不再需要执行的活动。Amazon SWF 使用合作取消机制，不会强行中断正在运行的活动任务。您必须给您的活动工作程序编程，以处理取消请求。

决策程序可决定在活动任务正在处理决策任务时取消该活动任务。要取消活动任务，决策程序可使用带 `RequestCancelActivityTask` 决策的 `RespondDecisionTaskCompleted` 操作。

如果活动工作程序尚未获得活动任务，则服务将取消任务。请注意，存在一种潜在的竞态条件，活动工作程序可在此条件中随时获取任务。如果任务已分配给活动工作程序，则该活动工作程序将被请求取消任务。

在这个示例中，将会发送取消订单信号到工作流程执行。

```
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"}
```

如果工作流执行收到信号，Amazon SWF 将返回类似于以下内容的请求。Amazon SWF 将生成一个决策任务，通知决策程序处理信号。

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

当决策程序处理决策任务并查看历史信号时，决策程序会尝试取消具有 `ShipOrderActivity0001` 活动 ID 的未解决活动。活动 ID 在排定活动任务事件的工作流程历史中有提供。

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

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

如果 Amazon SWF 成功接收取消请求，将会返回与下面类似的成功 HTTP 响应：

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

取消尝试以 ActivityTaskCancelRequested 事件的形式记录在历史中。

如果任务成功取消（如 ActivityTaskCanceled 事件所示），您需要对决策程序进行编程，使其在任务取消后采取相应的步骤，例如关闭工作流执行。

如果活动任务无法取消（例如，任务完成、失败或超时而不是取消），则决定程序应接受活动结果，或执行使用案例所需的任何清理或缓解措施。

如果活动工作程序已获得活动任务，则取消请求将通过任务检测信号机制发送。活动工作线程可定期使用 RecordActivityTaskHeartbeat 向 Amazon SWF 报告任务仍在进行中。

请注意，虽然建议执行长期运行的任务，但不需要活动工作程序检测信号。任务取消需要记录定期检测信号；如果工作程序不检测信号，则不能取消任务。

如果决策程序请求取消任务，则 Amazon SWF 会将 cancelRequest 对象的值设置为 true。cancelRequest 数据元是 ActivityTaskStatus 数据元的一部分，由服务响应 RecordActivityTaskHeartbeat 而返回。

Amazon SWF 不会阻止已请求取消的活动任务的成功完成；如何处理取消请求取决于活动。根据您的要求，对活动工作程序进行编程，以取消活动任务或忽略取消请求。

如果您希望活动工作程序指出该活动任务的工作已取消，请对其进行编程以响应 RespondActivityTaskCanceled。如果您希望活动工作程序完成任务，请对其进行编程以响应标准 RespondActivityTaskCompleted。

当 Amazon SWF 收到 RespondActivityTaskCompleted 或 RespondActivityTaskCanceled 请求时，会更新工作流执行历史并安排决策任务以通知决策程序。

对决策程序进行编程以处理决策任务和返回任何附加决策。如果活动任务被成功取消，请对决策程序进行编程以执行需要继续的任务或关闭工作流程执行。如果活动任务未成功取消，请对决策程序进行编程以接受结果、忽略结果或安排任何必要的清除。

Amazon Simple Workflow Service 中的安全性

本部分提供了有关 Amazon Simple Workflow Service 安全性和身份验证的信息。

主题

Amazon SWF 使用 IAM 来控制对其他 Amazon 服务和资源的访问。有关 IAM 工作原理的概述，请参阅《IAM 用户指南》https://docs.amazonaws.cn/IAM/latest/UserGuide/introduction_access-management.html中的访问管理概述。有关安全凭证的概述，请参阅 中的Amazon 安全凭证Amazon Web Services 一般参考。

Amazon Simple Workflow Service 中的数据保护

分 Amazon 分担责任模型适用于亚马逊简单工作流程服务中的数据保护。如本模型所述 Amazon ，负责保护运行所有内容的全球基础架构 Amazon Web Services 云。您负责维护对托管在此基础结构上的内容的控制。您还负责您所使用的 Amazon Web Services 服务 的安全配置和管理任务。有关数据隐私的更多信息，请参阅数据隐私常见问题。

出于数据保护目的，我们建议您保护 Amazon Web Services 账户 凭证并使用 Amazon IAM Identity Center 或 Amazon Identity and Access Management (IAM) 设置个人用户。这样，每个用户只获得履行其工作职责所需的权限。还建议您通过以下方式保护数据：

- 对每个账户使用多重身份验证（MFA）。
- 使用 SSL/TLS 与资源通信。 Amazon 我们要求使用 TLS 1.2，建议使用 TLS 1.3。
- 使用设置 API 和用户活动日志 Amazon CloudTrail。有关使用 CloudTrail 跟踪捕获 Amazon 活动的信息，请参阅《Amazon CloudTrail 用户指南》中的使用跟 CloudTrail 踪。
- 使用 Amazon 加密解决方案以及其中的所有默认安全控件 Amazon Web Services 服务。

- 使用高级托管安全服务（例如 Amazon Macie），它有助于发现和保护存储在 Amazon S3 中的敏感数据。
- 如果您在 Amazon 通过命令行界面或 API 进行访问时需要经过 FIPS 140-3 验证的加密模块，请使用 FIPS 端点。有关可用的 FIPS 端点的更多信息，请参阅《美国联邦信息处理标准（FIPS）第 140-3 版》。

强烈建议您切勿将机密信息或敏感信息（如您客户的电子邮件地址）放入标签或自由格式文本字段（如名称字段）。这包括您 Amazon Web Services 服务 使用控制台、API 或与 Amazon SWF 或其他机构合作的情况 Amazon CLI。 Amazon SDKs在用于名称的标签或自由格式文本字段中输入的任何数据都可能会用于计费或诊断日志。如果您向外部服务器提供网址，强烈建议您不要在网址中包含凭证信息来验证对该服务器的请求。

Amazon Simple Workflow Service 中的加密

静态加密

Amazon SWF 始终加密您的静态数据。Amazon Simple Workflow Service 中的静态数据使用透明的服务器端进行加密。这样可以帮助减少在保护敏感数据时涉及的操作负担和复杂性。通过静态加密，您可以构建符合加密合规性和法规要求的安全敏感型应用程序

传输中加密

Amazon SWF 和其他服务之间传递的所有数据均使用传输层安全性协议 (TLS) 进行加密。

Amazon Simple Workflow Service 中的 Identity and Access Management

访问 Amazon SWF 需要 Amazon 能够用来对您的请求进行身份验证的凭证。这些凭证必须具有访问 Amazon 资源的权限，例如从其他 Amazon 资源检索事件数据。下面几个部分提供了详细信息来说明如何使用 Amazon Identity and Access Management (IAM) 和 Amazon SWF 控制谁能访问您的资源，从而对这些资源进行保护。

Amazon Identity and Access Management (IAM) Amazon Web Services 服务 可帮助管理员安全地控制对 Amazon 资源的访问权限。IAM 管理员控制谁可以通过身份验证（登录）和获得授权（具有权限）来使用 Amazon SWF 资源。您可以使用 IAM Amazon Web Services 服务 ，无需支付额外费用。

主题

受众

您的使用方式 Amazon Identity and Access Management (IAM) 会有所不同，具体取决于您在 Amazon SWF 中所做的工作。

服务用户 – 如果您使用 Amazon SWF 服务来完成任务，您的管理员会为您提供所需的凭证和权限。随着您使用更多 Amazon SWF 功能来完成工作，您可能需要额外的权限。了解如何管理访问权限有

助于您向管理员请求适合的权限。如果您无法访问 Amazon SWF 中的功能，请参阅 Amazon Simple Workflow Service 身份和访问权限故障排除。

服务管理员 – 如果您在公司负责管理 Amazon SWF 资源，您可能对 Amazon SWF 具有完全访问权限。您有责任确定您的服务用户应访问哪些 Amazon SWF 功能和资源。然后，您必须向 IAM 管理员提交请求以更改服务用户的权限。请查看该页面上的信息以了解 IAM 的基本概念。要了解有关您的公司如何将 IAM 与 Amazon SWF 搭配使用的更多信息，请参阅 Amazon Simple Workflow Service works 如何与 IAM 结合使用。

IAM 管理员 – 如果您是 IAM 管理员，您可能需要详细了解如何编写策略以管理对 Amazon SWF 的访问。要查看您可以在 IAM 中使用的 Amazon SWF 基于身份的策略示例，请参阅 适用于 Amazon Simple Workflow Service 的基于身份的策略示例。

使用身份进行身份验证

身份验证是您 Amazon 使用身份凭证登录的方式。您必须以 IAM 用户身份或通过担 Amazon Web Services 账户根用户任 IAM 角色进行身份验证（登录 Amazon）。

如果您 Amazon 以编程方式访问，则会 Amazon 提供软件开发套件 (SDK) 和命令行接口 (CLI)，以便使用您的凭据对请求进行加密签名。如果您不使用 Amazon 工具，则必须自己签署请求。有关使用推荐的方法自行签署请求的更多信息，请参阅《IAM 用户指南》中的用于签署 API 请求的Amazon 签名版本 4。

无论使用何种身份验证方法，您都可能需要提供其他安全信息。例如， Amazon 建议您使用多重身份验证 (MFA) 来提高账户的安全性。要了解更多信息，请参阅《IAM 用户指南》中的 IAM 中的Amazon 多重身份验证。

Amazon Web Services 账户 root 用户

创建时 Amazon Web Services 账户，首先要有一个登录身份，该身份可以完全访问账户中的所有资源 Amazon Web Services 服务 和资源。此身份被称为 Amazon Web Services 账户 root 用户，使用您创建账户时使用的电子邮件地址和密码登录即可访问该身份。强烈建议您不要使用根用户执行日常任务。保护好根用户凭证，并使用这些凭证来执行仅根用户可以执行的任务。有关要求您以根用户身份登录的任务的完整列表，请参阅 IAM 用户指南中的需要根用户凭证的任务。

联合身份

作为最佳实践，要求人类用户（包括需要管理员访问权限的用户）使用与身份提供商的联合身份验证 Amazon Web Services 服务 通过临时证书进行访问。

联合身份是指您的企业用户目录、Web 身份提供商、Identity C 或者任何使用 Amazon Web Services 服务 通过身份源提供的凭据进行访问的用户。 Amazon Directory Service当联合身份访问时 Amazon Web Services 账户，他们将扮演角色，角色提供临时证书。

IAM 用户和群组

I AM 用户是您 Amazon Web Services 账户 内部对个人或应用程序具有特定权限的身份。在可能的情况下，我们建议使用临时凭证，而不是创建具有长期凭证（如密码和访问密钥）的 IAM 用户。但是，如果您有一些特定的使用场景需要长期凭证以及 IAM 用户，建议您轮换访问密钥。有关更多信息，请参阅《IAM 用户指南》中的对于需要长期凭证的用例，应在需要时更新访问密钥。

IAM 组是一个指定一组 IAM 用户的身份。您不能使用组的身份登录。您可以使用组来一次性为多个用户指定权限。如果有大量用户，使用组可以更轻松地管理用户权限。例如，您可以拥有一个名为的群组，IAMAdmins并向该群组授予管理 IAM 资源的权限。

用户与角色不同。用户唯一地与某个人员或应用程序关联，而角色旨在让需要它的任何人代入。用户具有永久的长期凭证，而角色提供临时凭证。要了解更多信息，请参阅《IAM 用户指南》中的 IAM 用户的使用案例。

IAM 角色

I AM 角色是您内部具有特定权限 Amazon Web Services 账户 的身份。它类似于 IAM 用户，但与特定人员不关联。要在中临时担任 IAM 角色 Amazon Web Services Management Console，您可以从用户切换到 IAM 角色（控制台）。您可以通过调用 Amazon CLI 或 Amazon API 操作或使用自定义 URL 来代入角色。有关使用角色的方法的更多信息，请参阅《IAM 用户指南》中的代入角色的方法。

具有临时凭证的 IAM 角色在以下情况下很有用：

- 联合用户访问：要向联合身份分配权限，请创建角色并为角色定义权限。当联合身份进行身份验证时，该身份将与角色相关联并被授予由此角色定义的权限。有关用于联合身份验证的角色的信息，请参阅《IAM 用户指南》中的针对第三方身份提供商创建角色（联合身份验证）。
- 临时 IAM 用户权限：IAM 用户可代入 IAM 用户或角色，以暂时获得针对特定任务的不同权限。
- 跨账户存取：您可以使用 IAM 角色以允许不同账户中的某个人（可信主体）访问您的账户中的资源。角色是授予跨账户访问权限的主要方式。但是，对于某些资源 Amazon Web Services 服务，您可以将策略直接附加到资源（而不是使用角色作为代理）。要了解用于跨账户访问的角色和基于资源的策略之间的差别，请参阅 IAM 用户指南中的 IAM 中的跨账户资源访问。
- 跨服务访问 — 有些 Amazon Web Services 服务 使用其他 Amazon Web Services 服务服务中的功能。例如，当您在服务中拨打电话时，该服务通常会在 Amazon 中运行应用程序 EC2 或在 Amazon

S3 中存储对象。服务可能会使用发出调用的主体的权限、使用服务角色或使用服务相关角色来执行此操作。

- 转发访问会话 (FAS) — 当您使用 IAM 用户或角色在中执行操作时 Amazon，您被视为委托人。使用某些服务时，您可能会执行一个操作，然后此操作在其他服务中启动另一个操作。FAS 使用调用委托人的权限以及 Amazon Web Services 服务 向下游服务发出请求的请求。 Amazon Web Services 服务只有当服务收到需要与其他 Amazon Web Services 服务 或资源交互才能完成的请求时，才会发出 FAS 请求。在这种情况下，您必须具有执行这两项操作的权限。有关发出 FAS 请求时的策略详情，请参阅转发访问会话。
- 服务角色 - 服务角色是服务代表您在您的账户中执行操作而分派的 IAM 角色。IAM 管理员可以在 IAM 中创建、修改和删除服务角色。有关更多信息，请参阅《IAM 用户指南》中的创建向 Amazon Web Services 服务委派权限的角色。
- 服务相关角色-服务相关角色是一种与服务相关联的服务角色。 Amazon Web Services 服务服务可以代入代表您执行操作的角色。服务相关角色出现在您的中 Amazon Web Services 账户 ，并且归服务所有。IAM 管理员可以查看但不能编辑服务相关角色的权限。

- 在 A@@ mazon 上运行的应用程序 EC2 — 您可以使用 IAM 角色管理在 EC2 实例上运行并发出 Amazon CLI 或 Amazon API 请求的应用程序的临时证书。这比在 EC2 实例中存储访问密钥更可取。要为 EC2 实例分配 Amazon 角色并使其可供其所有应用程序使用，您需要创建一个附加到该实例的实例配置文件。实例配置文件包含角色并允许在 EC2 实例上运行的程序获得临时证书。有关更多信息，请参阅 IAM 用户指南中的使用 IAM 角色向在 A mazon EC2 实例上运行的应用程序授予权限。

使用策略管理访问

您可以 Amazon 通过创建策略并将其附加到 Amazon 身份或资源来控制中的访问权限。策略是其中的一个对象 Amazon ，当与身份或资源关联时，它会定义其权限。 Amazon 在委托人（用户、root 用户或角色会话）发出请求时评估这些策略。策略中的权限确定是允许还是拒绝请求。大多数策略都以 JSON 文档的 Amazon 形式存储在中。有关 JSON 策略文档的结构和内容的更多信息，请参阅 IAM 用户指南中的 JSON 策略概览。

管理员可以使用 Amazon JSON 策略来指定谁有权访问什么。也就是说，哪个主体可以对什么资源执行操作，以及在什么条件下执行。

默认情况下，用户和角色没有权限。要授予用户对所需资源执行操作的权限，IAM 管理员可以创建 IAM 策略。管理员随后可以向角色添加 IAM 策略，用户可以代入角色。

IAM 策略定义操作的权限，无关乎您使用哪种方法执行操作。例如，假设您有一个允许 `iam:GetRole` 操作的策略。拥有该策略的用户可以从 Amazon Web Services Management Console Amazon CLI、或 Amazon API 获取角色信息。

基于身份的策略

基于身份的策略是可附加到身份（如 IAM 用户、用户组或角色）的 JSON 权限策略文档。这些策略控制用户和角色可在何种条件下对哪些资源执行哪些操作。要了解如何创建基于身份的策略，请参阅《IAM 用户指南》中的使用客户托管策略定义自定义 IAM 权限。

基于身份的策略可以进一步归类为内联策略或托管式策略。内联策略直接嵌入单个用户、组或角色中。托管策略是独立的策略，您可以将其附加到中的多个用户、群组和角色 Amazon Web Services 账户。托管策略包括 Amazon 托管策略和客户托管策略。要了解如何在托管策略和内联策略之间进行选择，请参阅《IAM 用户指南》中的在托管策略与内联策略之间进行选择。

基于资源的策略

基于资源的策略是附加到资源的 JSON 策略文档。基于资源的策略的示例包括 IAM 角色信任策略和 Amazon S3 存储桶策略。在支持基于资源的策略的服务中，服务管理员可以使用它们来控制对特定资源的访问。对于在其中附加策略的资源，策略定义指定主体可以对该资源执行哪些操作以及在什么条件下执行。您必须在基于资源的策略中指定主体。委托人可以包括账户、用户、角色、联合用户或 Amazon Web Services 服务。

基于资源的策略是位于该服务中的内联策略。您不能在基于资源的策略中使用 IAM 中的 Amazon 托管策略。

访问控制列表 (ACLs)

访问控制列表 (ACLs) 控制哪些委托人（账户成员、用户或角色）有权访问资源。 ACLs 与基于资源的策略类似，尽管它们不使用 JSON 策略文档格式。

Amazon S3 和 Amazon VPC 就是支持的服务示例 ACLs。 Amazon WAF要了解更多信息 ACLs，请参阅《亚马逊简单存储服务开发者指南》中的访问控制列表 (ACL) 概述。

其他策略类型

Amazon 支持其他不太常见的策略类型。这些策略类型可以设置更常用的策略类型向您授予的最大权限。

- 权限边界：权限边界是一个高级特征，用于设置基于身份的策略可以为 IAM 实体（IAM 用户或角色）授予的最大权限。您可为实体设置权限边界。这些结果权限是实体基于身份的策略及其权限边界

的交集。在 `Principal` 中指定用户或角色的基于资源的策略不受权限边界限制。任一项策略中的显式拒绝将覆盖允许。有关权限边界的更多信息，请参阅IAM 用户指南中的 IAM 实体的权限边界。

- 服务控制策略 (SCPs)- SCPs 是指定组织或组织单位 (OU) 的最大权限的 JSON 策略 Amazon Organizations。 Amazon Organizations 是一项用于对您的企业拥有的多 Amazon Web Services 账户 项进行分组和集中管理的服务。如果您启用组织中的所有功能，则可以将服务控制策略 (SCPs) 应用于您的任何或所有帐户。SCP 限制成员账户中的实体（包括每个 Amazon Web Services 账户根用户实体）的权限。有关 Organization SCPs s 和的更多信息，请参阅《Amazon Organizations 用户指南》中的服务控制策略。
- 资源控制策略 (RCPs) — RCPs 是 JSON 策略，您可以使用它来设置账户中资源的最大可用权限，而无需更新附加到您拥有的每个资源的 IAM 策略。RCP 限制成员账户中资源的权限，并可能影响身份（包括身份）的有效权限 Amazon Web Services 账户根用户，无论这些身份是否属于您的组织。有关 Organizations 的更多信息 RCPs，包括 Amazon Web Services 服务 该支持的列表 RCPs，请参阅《Amazon Organizations 用户指南》中的资源控制策略 (RCPs)。
- 会话策略：会话策略是当您以编程方式为角色或联合用户创建临时会话时作为参数传递的高级策略。结果会话的权限是用户或角色的基于身份的策略和会话策略的交集。权限也可以来自基于资源的策略。任一项策略中的显式拒绝将覆盖允许。有关更多信息，请参阅 IAM 用户指南中的会话策略。

多个策略类型

当多个类型的策略应用于一个请求时，生成的权限更加复杂和难以理解。要了解在涉及多种策略类型时如何 Amazon 确定是否允许请求，请参阅 IAM 用户指南中的策略评估逻辑。

访问控制

您可以使用有效的凭证来对自己的请求进行身份验证，但您还必须拥有权限才能创建或访问 Amazon SWF 资源。例如，您必须有权调用 Amazon Lambda与您的亚马逊 SWF 规则关联的亚马逊简单通知服务 (Amazon SNS) Service 和亚马逊简单队列服务 (Amazon SQS) 目标。

下面几部分介绍了如何管理 Amazon SWF 的权限。我们建议您先阅读概述。

- 基本原理
- Amazon SWF IAM 策略
- Amazon SWF 策略示例

Amazon SWF 的策略操作

支持策略操作：是

管理员可以使用 Amazon JSON 策略来指定谁有权访问什么。也就是说，哪个主体可以对什么资源执行操作，以及在什么条件下执行。

JSON 策略的 `Action` 元素描述可用于在策略中允许或拒绝访问的操作。策略操作通常与关联的 Amazon API 操作同名。有一些例外情况，例如没有匹配 API 操作的仅限权限 操作。还有一些操作需要在策略中执行多个操作。这些附加操作称为相关操作。

在策略中包含操作以授予执行关联操作的权限。

要查看 Amazon SWF 操作的列表，请参阅《Service Authorization Reference》中的 Resources Defined by Amazon Simple Workflow Service。

Amazon SWF 中的策略操作在操作前使用以下前缀：

```
swf
```

要在单个语句中指定多项操作，请使用逗号将它们隔开。

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

要查看 Amazon SWF 基于身份的策略的示例，请参阅 适用于 Amazon Simple Workflow Service 的基于身份的策略示例。

Amazon SWF 的策略资源

支持策略资源：是

管理员可以使用 Amazon JSON 策略来指定谁有权访问什么。也就是说，哪个主体可以对什么资源执行操作，以及在什么条件下执行。

`Resource` JSON 策略元素指定要向其应用操作的一个或多个对象。语句必须包含 `Resource` 或 `NotResource` 元素。作为最佳实践，请使用其 Amazon 资源名称（ARN）指定资源。对于支持特定资源类型（称为资源级权限）的操作，您可以执行此操作。

对于不支持资源级权限的操作（如列出操作），请使用通配符（*）指示语句应用于所有资源。

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

要查看 Amazon SWF 资源类型及其列表 ARNs，请参阅服务授权参考中的亚马逊简单工作流程服务定义的操作。要了解您可以在哪些操作中指定每个资源的 ARN，请参阅 Resources Defined by Amazon Simple Workflow Service。

要查看 Amazon SWF 基于身份的策略的示例，请参阅 适用于 Amazon Simple Workflow Service 的基于身份的策略示例。

Amazon SWF 的策略条件键

支持特定于服务的策略条件键：是

管理员可以使用 Amazon JSON 策略来指定谁有权访问什么。也就是说，哪个主体可以对什么资源执行操作，以及在什么条件下执行。

在 Condition 元素（或 Condition 块）中，可以指定语句生效的条件。Condition 元素是可选的。您可以创建使用条件运算符（例如，等于或小于）的条件表达式，以使策略中的条件与请求中的值相匹配。

如果您在一个语句中指定多个 Condition 元素，或在单个 Condition 元素中指定多个键，则 Amazon 使用逻辑 AND 运算评估它们。如果您为单个条件键指定多个值，则使用逻辑OR运算来 Amazon 评估条件。在授予语句的权限之前必须满足所有的条件。

在指定条件时，您也可以使用占位符变量。例如，只有在使用 IAM 用户名标记 IAM 用户时，您才能为其授予访问资源的权限。有关更多信息，请参阅《IAM 用户指南》中的 IAM 策略元素：变量和标签。

Amazon 支持全局条件密钥和特定于服务的条件密钥。要查看所有 Amazon 全局条件键，请参阅 IAM 用户指南中的Amazon 全局条件上下文密钥。

要查看 Amazon SWF 条件键的列表，请参阅《Service Authorization Reference》中的 Condition Keys for Amazon Simple Workflow Service。要了解您可以对哪些操作和资源使用条件键，请参阅 Resources Defined by Amazon Simple Workflow Service。

要查看 Amazon SWF 基于身份的策略的示例，请参阅 适用于 Amazon Simple Workflow Service 的基于身份的策略示例。

ACLs 在亚马逊 SWF 中

支持 ACLs：否

访问控制列表 (ACLs) 控制哪些委托人（账户成员、用户或角色）有权访问资源。 ACLs 与基于资源的策略类似，尽管它们不使用 JSON 策略文档格式。

ABAC 与 Amazon SWF

支持 ABAC（策略中的标签）：部分支持

基于属性的访问控制（ABAC）是一种授权策略，该策略基于属性来定义权限。在中 Amazon，这些属性称为标签。您可以向 IAM 实体（用户或角色）和许多 Amazon 资源附加标签。标记实体和资源是 ABAC 的第一步。然后设计 ABAC 策略，以在主体的标签与他们尝试访问的资源标签匹配时允许操作。

ABAC 在快速增长的环境中非常有用，并在策略管理变得繁琐的情况下可以提供帮助。

要基于标签控制访问，您需要使用 `aws:ResourceTag/key-name`、`aws:RequestTag/key-name` 或 `aws:TagKeys` 条件键在策略的条件元素中提供标签信息。

如果某个服务对于每种资源类型都支持所有这三个条件键，则对于该服务，该值为是。如果某个服务仅对于部分资源类型支持所有这三个条件键，则该值为部分。

有关 ABAC 的更多信息，请参阅《IAM 用户指南》中的使用 ABAC 授权定义权限。要查看设置 ABAC 步骤的教程，请参阅《IAM 用户指南》中的使用基于属性的访问权限控制（ABAC）。

将临时凭证用于 Amazon SWF

支持临时凭证：是

当你使用临时证书登录时，有些 Amazon Web Services 服务 不起作用。有关更多信息，包括哪些 Amazon Web Services 服务 适用于临时证书，请参阅 IAM 用户指南中的Amazon Web Services 服务与 IAM 配合使用的信息。

如果您使用除用户名和密码之外的任何方法登录，则 Amazon Web Services Management Console 使用的是临时证书。例如，当您 Amazon 使用公司的单点登录 (SSO) 链接进行访问时，该过程会自动创建临时证书。当您以用户身份登录控制台，然后切换角色时，您还会自动创建临时凭证。有关切换角色的更多信息，请参阅《IAM 用户指南》中的从用户切换到 IAM 角色（控制台）。

您可以使用 Amazon CLI 或 Amazon API 手动创建临时证书。然后，您可以使用这些临时证书进行访问 Amazon。 Amazon 建议您动态生成临时证书，而不是使用长期访问密钥。有关更多信息，请参阅 IAM 中的临时安全凭证。

Amazon SWF 的跨服务主体权限

支持转发访问会话（FAS）：是

当您使用 IAM 用户或角色在中执行操作时 Amazon，您被视为委托人。使用某些服务时，您可能会执行一个操作，然后此操作在其他服务中启动另一个操作。FAS 使用调用委托人的权限以及 Amazon Web Services 服务 向下游服务发出请求的请求。 Amazon Web Services 服务只有当服务收到需要与其他 Amazon Web Services 服务 或资源交互才能完成的请求时，才会发出 FAS 请求。在这种情况下，您必须具有执行这两项操作的权限。有关发出 FAS 请求时的策略详情，请参阅转发访问会话。

Amazon SWF 的服务角色

支持服务角色：是

服务角色是由一项服务担任、代表您执行操作的 IAM 角色。IAM 管理员可以在 IAM 中创建、修改和删除服务角色。有关更多信息，请参阅《IAM 用户指南》中的创建向 Amazon Web Services 服务委派权限的角色。

> **Warning**
>
> 更改服务角色的权限可能会破坏 Amazon SWF 的功能。仅当 Amazon SWF 提供相关指导时才编辑服务角色。

Amazon SWF 的服务相关角色

支持服务相关角色：否

服务相关角色是一种与服务相关联的 Amazon Web Services 服务服务角色。服务可以代入代表您执行操作的角色。服务相关角色出现在您的中 Amazon Web Services 账户 ，并且归服务所有。IAM 管理员可以查看但不能编辑服务相关角色的权限。

有关创建或管理服务相关角色的详细信息，请参阅能够与 IAM 搭配使用的Amazon 服务。在表中查找服务相关角色列中包含 Yes 的表。选择是链接以查看该服务的服务相关角色文档。

Amazon SWF 基于身份的策略

支持基于身份的策略：是

基于身份的策略是可附加到身份（如 IAM 用户、用户组或角色）的 JSON 权限策略文档。这些策略控制用户和角色可在何种条件下对哪些资源执行哪些操作。要了解如何创建基于身份的策略，请参阅《IAM 用户指南》中的使用客户管理型策略定义自定义 IAM 权限。

通过使用 IAM 基于身份的策略，您可以指定允许或拒绝的操作和资源以及允许或拒绝操作的条件。您无法在基于身份的策略中指定主体，因为它适用于其附加的用户或角色。要了解可在 JSON 策略中使用的所有元素，请参阅《IAM 用户指南》中的 IAM JSON 策略元素引用。

Amazon SWF 基于身份的策略示例

要查看 Amazon SWF 基于身份的策略的示例，请参阅 适用于 Amazon Simple Workflow Service 的基于身份的策略示例。

Amazon SWF 基于资源的策略

支持基于资源的策略：否

基于资源的策略是附加到资源的 JSON 策略文档。基于资源的策略的示例包括 IAM 角色信任策略和 Amazon S3 存储桶策略。在支持基于资源的策略的服务中，服务管理员可以使用它们来控制对特定资源的访问。对于在其中附加策略的资源，策略定义指定主体可以对该资源执行哪些操作以及在什么条件下执行。您必须在基于资源的策略中指定主体。委托人可以包括账户、用户、角色、联合用户或 Amazon Web Services 服务。

要启用跨账户访问，您可以将整个账户或其他账户中的 IAM 实体指定为基于资源的策略中的主体。将跨账户主体添加到基于资源的策略只是建立信任关系工作的一半而已。当委托人和资源处于不同位置时 Amazon Web Services 账户，可信账户中的 IAM 管理员还必须向委托人实体（用户或角色）授予访问资源的权限。他们通过将基于身份的策略附加到实体以授予权限。但是，如果基于资源的策略向同一个账户中的主体授予访问权限，则不需要额外的基于身份的策略。有关更多信息，请参阅《IAM 用户指南》中的 IAM 中的跨账户资源访问。

Amazon Simple Workflow Service works 如何与 IAM 结合使用

在使用 IAM 管理对 Amazon SWF 的访问权限之前，您应该了解哪些 IAM 功能可用于 Amazon SWF。

可与 Amazon Simple Queue Service 结合使用的 IAM 功能

IAM 特征	Amazon SWF 支持
基于身份的策略	是
基于资源的策略	否
策略操作	是

IAM 特征	Amazon SWF 支持
策略资源	是
策略条件键（特定于服务）	是
ACLs	否
ABAC（策略中的标签）	部分
临时凭证	是
主体权限	是
服务角色	是
服务相关角色	否

要全面了解 Amazon SWF 和其他 Amazon 服务如何与大多数 IAM 功能配合使用，请参阅 IAM 用户指南中与 IAM 配合使用的Amazon 服务。

适用于 Amazon Simple Workflow Service 的基于身份的策略示例

原定设置情况下，用户和角色没有创建或修改 Amazon SWF 资源的权限。他们也无法使用 Amazon Web Services Management Console、 Amazon Command Line Interface (Amazon CLI) 或 Amazon API 执行任务。要授予用户对所需资源执行操作的权限，IAM 管理员可以创建 IAM 策略。管理员随后可以向角色添加 IAM 策略，用户可以代入角色。

要了解如何使用这些示例 JSON 策略文档创建基于 IAM 身份的策略，请参阅《IAM 用户指南》中的创建 IAM 策略（控制台）。

有关 Amazon SWF 定义的操作和资源类型（包括每种资源类型的格式）的详细信息，请参阅服务授权参考中的亚马逊简单工作流程服务的操作、资源和条件密钥。 ARNs

主题

- 策略最佳实践
- 使用 Amazon SWF 控制台
- 允许用户查看他们自己的权限

策略最佳实践

基于身份的策略确定某个人是否可以创建、访问或删除您账户中的 Amazon SWF 资源。这些操作可能会使 Amazon Web Services 账户产生成本。创建或编辑基于身份的策略时，请遵循以下指南和建议：

- 开始使用 Amazon 托管策略并转向最低权限权限 — 要开始向用户和工作负载授予权限，请使用为许多常见用例授予权限的Amazon 托管策略。它们在你的版本中可用 Amazon Web Services 账户。我们建议您通过定义针对您的用例的 Amazon 客户托管策略来进一步减少权限。有关更多信息，请参阅《IAM 用户指南》中的 Amazon 托管式策略或工作职能的Amazon 托管式策略。
- 应用最低权限：在使用 IAM 策略设置权限时，请仅授予执行任务所需的权限。为此，您可以定义在特定条件下可以对特定资源执行的操作，也称为最低权限许可。有关使用 IAM 应用权限的更多信息，请参阅《IAM 用户指南》中的 IAM 中的策略和权限。
- 使用 IAM 策略中的条件进一步限制访问权限：您可以向策略添加条件来限制对操作和资源的访问。例如，您可以编写策略条件来指定必须使用 SSL 发送所有请求。如果服务操作是通过特定的方式使用的，则也可以使用条件来授予对服务操作的访问权限 Amazon Web Services 服务，例如 Amazon CloudFormation。有关更多信息，请参阅《IAM 用户指南》中的 IAM JSON 策略元素：条件。
- 使用 IAM Access Analyzer 验证您的 IAM 策略，以确保权限的安全性和功能性 – IAM Access Analyzer 会验证新策略和现有策略，以确保策略符合 IAM 策略语言（JSON）和 IAM 最佳实践。IAM Access Analyzer 提供 100 多项策略检查和可操作的建议，以帮助您制定安全且功能性强的策略。有关更多信息，请参阅《IAM 用户指南》中的使用 IAM Access Analyzer 验证策略。
- 需要多重身份验证 (MFA)-如果 Amazon Web Services 账户您的场景需要 IAM 用户或根用户，请启用 MFA 以提高安全性。若要在调用 API 操作时需要 MFA，请将 MFA 条件添加到您的策略中。有关更多信息，请参阅《IAM 用户指南》中的使用 MFA 保护 API 访问。

有关 IAM 中的最佳实操的更多信息，请参阅《IAM 用户指南》中的 IAM 中的安全最佳实践。

使用 Amazon SWF 控制台

要访问 Amazon Simple Workflow Service 控制台，您必须具有一组最低权限。这些权限必须允许您列出和查看有关您的 Amazon SWF 资源的详细信息。 Amazon Web Services 账户如果创建比必需的最低权限更为严格的基于身份的策略，对于附加了该策略的实体（用户或角色），控制台将无法按预期正常运行。

对于仅调用 Amazon CLI 或 Amazon API 的用户，您无需为其设置最低控制台权限。相反，只允许访问与其尝试执行的 API 操作相匹配的操作。

为确保用户和角色仍然可以使用 Amazon SWF 控制台，还需要将 Amazon *ConsoleAccess* SWF *ReadOnly* Amazon 或托管策略附加到实体。有关更多信息，请参阅《IAM 用户指南》中的为用户添加权限。

允许用户查看他们自己的权限

该示例说明了您如何创建策略，以允许 IAM 用户查看附加到其用户身份的内联和托管式策略。此策略包括在控制台上或使用 Amazon CLI 或 Amazon API 以编程方式完成此操作的权限。

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

基本原理

Amazon SWF 访问控制主要基于两种类型的权限：

- 资源权限：用户可以访问何种 Amazon SWF 资源。

  您可以只针对域授予资源权限。
- API 权限：用户可以调用何种 Amazon SWF 操作。

最简单的方法是授予完全账户访问权限（在任何域中调用任何 Amazon SWF 操作）或者完全拒绝访问。但是，IAM 支持更精细的访问控制方法，这种方法通常更有用。例如，您可以：

- 允许用户不受限制地调用任何 Amazon SWF 操作，但只能在指定域中调用。您可以使用这样一个策略允许开发中的工作流应用程序使用任何操作，但只能是“沙盒”域。
- 允许用户访问任何域，但限制其使用 API 的方法。您可以使用这样一个策略允许“审核员”应用程序在任何域中调用 API，但只允许读取访问。
- 允许用户在特定域中只调用一组有限的操作。您可以使用这样一个策略允许工作流启动程序只在特定域中调用 `StartWorkflowExecution` 操作。

Amazon SWF 访问控制基于以下原则：

- 访问控制决策仅基于 IAM 策略；所有策略审计和操作均通过 IAM 完成。
- 访问控制模型使用 deny-by-default策略；任何未明确允许的访问都将被拒绝。
- 您可以通过向工作流的操作者附加适当的 IAM 策略来控制对 Amazon SWF 资源的访问。
- 资源权限只能针对域授予。
- 您可以通过将条件应用到一个或多个参数上来进一步限制某些操作的使用。
- 如果您授予使用权限 RespondDecisionTaskCompleted，则可以对该操作中包含的决策列表表示权限。

  每个决策都由一个或多个参数，就像常规 API 调用那样。为了使策略可读性强一点，您可以授予决策权限，就像它们是实际的 API 调用那样，还包括将条件应用于一些参数。这些类型的权限被称为伪 API 权限。

有关能用条件进行限制的常规和伪 API 参数的摘要，请参阅 API 摘要。

Amazon SWF IAM 策略

IAM 策略包含一个或多个 Statement 元素，每个元素都包含一组定义该策略的元素。有关元素的完整列表以及如何构建策略的一般性讨论，请参阅 The Access Policy Language。Amazon SWF 访问控制基于以下元素：

效果

(必需) 该语句的效果：deny 或 allow。

> **Note**
>
> 您必须显式允许访问；默认情况下，IAM 会拒绝访问。

资源

（必需）该语句适用的资源（ Amazon 服务中用户可以与之交互的实体）。

您可以只针对域授予资源权限。例如，策略只允许对您的账户中特定域进行访问。要表达对域名的权限，Resource请将其设置为该域的亚马逊资源名称 (ARN)，其格式为 “arn: aws: swf::: / domain/”。*Region* *AccountID* *DomainName* *Region*是 Amazon 区域，*AccountID*是没有破折号的账户 ID，*DomainName*是域名。

操作

（必填）该语句适用的操作，您可以使用以下格式来引用该操作：*serviceId*: *action*。对于亚马逊 SWF，请设置为*serviceID*。swf例如，swf:StartWorkflowExecution指的是StartWorkflowExecution操作，用于控制允许哪些用户启动工作流程。

如果您授予使用权限 RespondDecisionTaskCompleted，则还可以使用来表达伪 API 的权限，从而控制Action对包含的决策列表的访问权限。由于 IAM 在默认情况下拒绝访问，您必须显式允许决策程序的决策，否则决策将不被接受。您可以使用 * 值允许所有决策。

状况

(可选) 表示一个或多个操作参数的约束条件，以限制允许的值。

Amazon SWF 操作的范围通常较广，您可以使用 IAM 条件来缩小范围。例如，要限制允许该PollForActivityTask操作访问的任务列表，您可以添加Condition并使用swf:taskList.name密钥指定允许列表。

您可以表达下列实体的约束条件。

- 工作流类型。名称和版本具有单独密钥。
- 活动类型。名称和版本具有单独密钥。
- 任务列表。
- Tags. 您可以为某些操作指定多个标签。在此情况下，每个标签都有一个单独的密钥。

> **Note**
>
> 对于 Amazon SWF，所有值都是字符串，因此您可以使用字符串运算符（如 `StringEquals`）来限制参数，将参数限制为指定的字符串。但是，`StringEquals` 等常规字符串比较运算符需要全部请求以包含参数。如果不显式包含参数，且类型注册过程中未提供默认值 (如默认任务列表)，则访问将被拒绝。
>
> 将条件视为可选项通常很有用，这样一来，您可以调用操作，而无需包含相关参数。例如，您可能希望允许决策者指定一组RespondDecisionTaskCompleted决策，但也允许它为任何特定调用仅指定其中一个决策。这种情况下，可使用 `StringEqualsIfExists` 运算符限制适当的参数，在参数满足条件时允许访问，但不会在参数不存在时拒绝访问。

有关可限制参数的完整列表和相关密钥，请参阅 API 摘要。

下一部分将举例说明如何构建 Amazon SWF 策略。有关详细信息，请参阅字符串条件。

Amazon SWF 策略示例

一个工作流由多个操作者组成，包括活动、决策程序等。您可以通过附加适当的 IAM 策略控制每个操作者的访问权限。本章节提供了一些示例。以下所示为最简单的案例：

```
{
   "Version": "2012-10-17",
   "Statement" : [ {
      "Effect" : "Allow",
      "Action" : "swf:*",
      "Resource" : "arn:aws:swf:*:123456789012:/domain/*"
   } ]
}
```

如果您将此策略附加到参与者，它对所有地区的账户都有访问权。您可以使用通配符利用一个值来表示多个资源、操作或地区。

- Resource 值中的第一个通配符 (*) 表示资源权限适用于所有区域。要将权限限制在单个区域，请使用适当的区域字符串替换通配符，如 us-east-1。
- Resource 值中的第二个通配符 (*) 让操作者可以访问指定区域中的任何账户域。
- Action 值中的通配符 (*) 让操作者可以调用任何 Amazon SWF 操作。

有关通配符使用方法的详细信息，请参阅元素描述

以下章节显示的是以更精细方法授予权限的策略的示例。

域权限

如果您想将部门工作流限制到一个特定域中，您可以使用与下述类似的方法：

```
{
   "Version": "2012-10-17",
   "Statement": [ {
      "Effect" : "Allow",
      "Action" : "swf:*",
      "Resource" : "arn:aws:swf:*:123456789012:/domain/department1"
   } ]
}
```

如果您附加此策略到参与者，它可以调用任何操作，但只能针对 department1 域。

如果您希望参与者有权访问多个域，您可以针对每个域单独授予权限，如下所述：

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect" : "Allow",
         "Action" : "swf:*",
         "Resource" : "arn:aws:swf:*:123456789012:/domain/department1"
      }, {
         "Effect" : "Allow",
         "Action" : "swf:*",
         "Resource" : "arn:aws:swf:*:123456789012:/domain/department2"
      }
   ]
}
```

如果将此策略附加到一个操作者，它就可以使用 department1 和 department2 域中的任何 Amazon SWF 操作。有时候，您还可以使用通配符表示多个域。

API 权限和约束条件

您控制用户可以对 Action 操作元素使用哪些操作。您可以选择通过使用 Condition 元素限制操作的可允许参数值。

如果您想将参与者限制为只进行特定操作，您可以使用与下述类似的方法：

```
{
   "Version": "2012-10-17",
   "Statement": [ {
      "Effect" : "Allow",
      "Action" : "swf:StartWorkflowExecution",
      "Resource" : "arn:aws:swf:*:123456789012:/domain/department2"
   } ]
}
```

如果将此策略附加到操作者，它能调用 StartWorkflowExecution 在 department2 域中启动工作流。它不能在任何其他域中使用任何其他操作或启动工作流。

您可以进一步限制参与者通过限制一个或多个 StartWorkflowExecution 参数值启动的工作流范围，如下所述：

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect" : "Allow",
         "Action" : "swf:StartWorkflowExecution",
         "Resource" : "arn:aws:swf:*:123456789012:/domain/department1",
         "Condition" : {
            "StringEquals" : {
               "swf:workflowType.name" : "workflow1",
               "swf:workflowType.version" : "version2"
            }
         }
      }
   ]
}
```

此策略限制 StartWorkflowExecution 操作的 name 和 version 参数。如果将此策略附加到操作者，它只能在 department1 域中运行 workflow1 的 version2，且两个参数都必须包含在请求中。

您可以使用 StringEqualsIfExists 运算符限制操作，无需将其包含在请求中，如下所述：

```
{
   "Version": "2012-10-17",
   "Statement" : [ {
      "Effect" : "Allow",
      "Action" : "swf:StartWorkflowExecution",
      "Resource" : "arn:aws:swf:*:123456789012:/domain/some_domain",
      "Condition" : {
         "StringEqualsIfExists" : { "swf:taskList.name" : "task_list_name" }
      }
   } ]
}
```

参与者可通过此策略在启动工作流执行时选择性地指定任务列表。

您可以限制某些操作的标签列表。在此情况下，每个标签都有一个单独的密钥，所以您会使用 swf:tagList.member.0 限制列表中的第一个标签，使用 swf:tagList.member.1 限制列表中的第二个标签，以此类推，最多能限制 5 个。但是，对于标签列表的限制方法，您必须要谨慎。关于实例，下面是非推荐策略的示例：

```
{
   "Version": "2012-10-17",
   "Statement" : [ {
      "Effect" : "Allow",
      "Action" : "swf:StartWorkflowExecution",
      "Resource" : "arn:aws:swf:*:123456789012:/domain/some_domain",
      "Condition" : {
         "StringEqualsIfExists" : {
            "swf:tagList.member.0" : "some_ok_tag", "another_ok_tag"
         }
      }
   } ]
}
```

此策略允许您选择指定 some_ok_tag 或 another_ok_tag。但是，此策略只会限制标签列表的第一个元素。列表可以有包含都会被允许的任意值的其他元素，因为此策略不会对 swf:tagList.member.1、swf:tagList.member.2 等等应用任何条件。

解决此问题的一个方法是禁止使用标签列表。以下策略要求列表中只能包含一个元素，从而确保只允许有 some_ok_tag 或 another_ok_tag。

```
{
   "Version": "2012-10-17",
   "Statement" : [ {
      "Effect" : "Allow",
      "Action" : "swf:StartWorkflowExecution",
      "Resource" : "arn:aws:swf:*:123456789012:/domain/some_domain",
      "Condition" : {
         "StringEqualsIfExists" : {
            "swf:tagList.member.0" : "some_ok_tag", "another_ok_tag"
         },
         "Null" : { "swf:tagList.member.1" : "true" }
      }
   } ]
}
```

伪 API 权限和约束条件

如果您想限制提供给 RespondDecisionTaskCompleted 的决策，您首先必须允许参与者调用 RespondDecisionTaskCompleted。然后，您可以使用常规 API 的句法，授予适当伪 API 成员权限，如下所示：

```
{
   "Version": "2012-10-17",
   "Statement" : [
      {
         "Resource" : "arn:aws:swf:*:123456789012:/domain/*",
         "Action" : "swf:RespondDecisionTaskCompleted",
         "Effect" : "Allow"
      }, {
         "Resource" : "*",
         "Action" : "swf:ScheduleActivityTask",
         "Effect" : "Allow",
         "Condition" : {
            "StringEquals" : { "swf:activityType.name" : "SomeActivityType" }
         }
```

```
      }
   ]
}
```

如果您将此策略附加到参与者，第一个 Statement 元素将允许参与者调用 RespondDecisionTaskCompleted。第二个元素允许操作者使用 ScheduleActivityTask 决策来指导 Amazon SWF 计划活动任务。要允许所有决定，请将 “swf:ScheduleActivityTask” 替换为 “swf:*”。

您可以使用 Condition 运算符像使用常规 API 一样地限制参数。此 Condition 的 StringEquals 运算符允许 RespondDecisionTaskCompleted 为 SomeActivityType 活动安排一个活动任务，并且它必须安排该任务。如果您想要允许 RespondDecisionTaskCompleted 使用一个参数值但又不需要它这样做，您可以替代使用 StringEqualsIfExists 运算符。

Amazon 托管策略： SimpleWorkflowFullAccess

您可以将 SimpleWorkflowFullAccess 策略附加到 IAM 身份。

该策略提供对 Amazon SWF 配置服务的完全访问权限。

权限详细信息

该策略包含以下权限。

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

IAM 策略的服务模型限制

在创建 IAM 策略时，您必须考虑服务模型限制条件。创建一个代表无效 Amazon SWF 请求的在语法上有效的 IAM 策略是有可能的；但在访问控制方面得到允许的请求仍然会因为是无效请求而失败。

对于实例，ListOpenWorkflowExecutions不 建议对 使用以下策略：

```
{
   "Version": "2012-10-17",
   "Statement" : [ {
      "Effect" : "Allow",
      "Action" : "swf:ListOpenWorkflowExecutions",
      "Resource" : "arn:aws:swf:*:123456789012:/domain/domain_name",
      "Condition" : {
         "StringEquals" : {
            "swf:typeFilter.name" : "workflow_name",
            "swf:typeFilter.version" : "workflow_version",
            "swf:tagFilter.tag" : "some_tag"
         }
      }
   } ]
}
```

Amazon SWF 服务模型不允许在同一个 ListOpenWorkflowExecutions 请求中使用 typeFilter 和 tagFilter 参数。因此，该策略允许通过引发无效请求 ValidationException 来拒绝服务的调用。

API 摘要

本部分简要介绍了如何使用 IAM 策略来控制操作者如何使用每个 API 和伪 API 访问 Amazon SWF 资源。

- 对于除 RegisterDomain 和 ListDomains 之外的所有操作，您可以通过授予域资源权限的方式允许或拒绝对任何或所有账户域的访问。
- 您可以允许或拒绝任何常规 API 成员的权限，如果您授权调用 RespondDecisionTaskCompleted，还可以允许或拒绝任何伪 API 成员的权限。
- 您可以使用 Condition 限制某些参数的可允许值。

以下章节列出了可针对每个常规和伪 API 成员限制的参数，并提供了相关密钥，指出您可以根据其控制域访问的任何限制条件。

常规 API

本章节列出了常规 API 成员，并简要描述了可进行限制的参数和相关密钥。它还指出了您可以根据其控制域访问的任何限制条件。

CountClosedWorkflowExecutions

- tagFilter.tag – 字符串约束。密钥是 swf:tagFilter.tag
- typeFilter.name – 字符串约束。键是 swf:typeFilter.name。
- typeFilter.version – 字符串约束。键是 swf:typeFilter.version。

> **Note**
>
> CountClosedWorkflowExecutions 要求 typeFilter 和 tagFilter 相互排斥。

CountOpenWorkflowExecutions

- tagFilter.tag – 字符串约束。密钥是 swf:tagFilter.tag
- typeFilter.name – 字符串约束。键是 swf:typeFilter.name。
- typeFilter.version – 字符串约束。键是 swf:typeFilter.version。

> **Note**
>
> CountOpenWorkflowExecutions 要求 typeFilter 和 tagFilter 相互排斥。

CountPendingActivityTasks

- taskList.name – 字符串约束。键是 swf:taskList.name。

CountPendingDecisionTasks

- taskList.name – 字符串约束。键是 swf:taskList.name。

DeleteActivityType

- activityType.name – 字符串约束。键是 swf:activityType.name。
- activityType.version – 字符串约束。键是 swf:activityType.version。

DeprecateActivityType

- activityType.name – 字符串约束。键是 swf:activityType.name。
- activityType.version – 字符串约束。键是 swf:activityType.version。

DeprecateDomain

- 您不能限制此操作的参数。

DeleteWorkflowType

- workflowType.name – 字符串约束。键是 swf:workflowType.name。
- workflowType.version – 字符串约束。键是 swf:workflowType.version。

DeprecateWorkflowType

- workflowType.name – 字符串约束。键是 swf:workflowType.name。
- workflowType.version – 字符串约束。键是 swf:workflowType.version。

DescribeActivityType

- activityType.name – 字符串约束。键是 swf:activityType.name。
- activityType.version – 字符串约束。键是 swf:activityType.version。

DescribeDomain

- 您不能限制此操作的参数。

DescribeWorkflowExecution

- 您不能限制此操作的参数。

DescribeWorkflowType

- workflowType.name – 字符串约束。键是 swf:workflowType.name。
- workflowType.version – 字符串约束。键是 swf:workflowType.version。

GetWorkflowExecutionHistory

- 您不能限制此操作的参数。

ListActivityTypes

- 您不能限制此操作的参数。

ListClosedWorkflowExecutions

- tagFilter.tag – 字符串约束。密钥是 swf:tagFilter.tag
- typeFilter.name – 字符串约束。键是 swf:typeFilter.name。
- typeFilter.version – 字符串约束。键是 swf:typeFilter.version。

> **Note**
>
> ListClosedWorkflowExecutions 要求 typeFilter 和 tagFilter 相互排斥。

ListDomains

- 您不能限制此操作的参数。

ListOpenWorkflowExecutions

- tagFilter.tag – 字符串约束。密钥是 swf:tagFilter.tag
- typeFilter.name – 字符串约束。键是 swf:typeFilter.name。
- typeFilter.version – 字符串约束。键是 swf:typeFilter.version。

> **Note**
>
> ListOpenWorkflowExecutions 要求 typeFilter 和 tagFilter 相互排斥。

ListWorkflowTypes

- 您不能限制此操作的参数。

PollForActivityTask

- taskList.name – 字符串约束。键是 swf:taskList.name。

PollForDecisionTask

- taskList.name – 字符串约束。键是 swf:taskList.name。

RecordActivityTaskHeartbeat

- 您不能限制此操作的参数。

RegisterActivityType

- defaultTaskList.name – 字符串约束。键是 swf:defaultTaskList.name。
- name – 字符串约束。键是 swf:name。
- version – 字符串约束。键是 swf:version。

RegisterDomain

- name – 注册的域名可作为此操作的资源。

RegisterWorkflowType

- defaultTaskList.name – 字符串约束。键是 swf:defaultTaskList.name。
- name – 字符串约束。键是 swf:name。
- version – 字符串约束。键是 swf:version。

RequestCancelWorkflowExecution

- 您不能限制此操作的参数。

RespondActivityTaskCanceled

- 您不能限制此操作的参数。

RespondActivityTaskCompleted

- 您不能限制此操作的参数。

RespondActivityTaskFailed

- 您不能限制此操作的参数。

RespondDecisionTaskCompleted

- decisions.member.N – 通过伪 API 权限间接限制。有关详细信息，请参阅伪 API。

SignalWorkflowExecution

- 您不能限制此操作的参数。

StartWorkflowExecution

- tagList.member.0 – 字符串约束。密钥是 swf:tagList.member.0
- tagList.member.1 – 字符串约束。密钥是 swf:tagList.member.1
- tagList.member.2 – 字符串约束。密钥是 swf:tagList.member.2
- tagList.member.3 – 字符串约束。密钥是 swf:tagList.member.3
- tagList.member.4 – 字符串约束。密钥是 swf:tagList.member.4
- taskList.name – 字符串约束。键是 swf:taskList.name。
- workflowType.name – 字符串约束。键是 swf:workflowType.name。
- workflowType.version – 字符串约束。键是 swf:workflowType.version。

> **Note**
>
> 限制的标签不能超过五个。

TerminateWorkflowExecution

- 您不能限制此操作的参数。

伪 API

本章节列出了表示 RespondDecisionTaskCompleted 中所包含决策的伪 API 成员。如果您已授权使用 RespondDecisionTaskCompleted，您的策略会用与常规 API 相同的方法来授予此 API 成员权限。您可以通过设置一个或多个参数的条件来限制伪 API 的某些成员。本章节列出了伪 API 成员，并简要描述了可进行限制的参数与相关密钥。

> **Note**
>
> aws:SourceIP、aws:UserAgent 和 aws:SecureTransport 密钥不可用于伪 API。如果您的预期安全策略需要这些密钥对伪 API 进行访问控制，您可以将其用于 RespondDecisionTaskCompleted 操作。

CancelTimer

- 您不能限制此操作的参数。

CancelWorkflowExecution

- 您不能限制此操作的参数。

CompleteWorkflowExecution

- 您不能限制此操作的参数。

ContinueAsNewWorkflowExecution

- tagList.member.0 – 字符串约束。密钥是 swf:tagList.member.0
- tagList.member.1 – 字符串约束。密钥是 swf:tagList.member.1
- tagList.member.2 – 字符串约束。密钥是 swf:tagList.member.2
- tagList.member.3 – 字符串约束。密钥是 swf:tagList.member.3
- tagList.member.4 – 字符串约束。密钥是 swf:tagList.member.4
- taskList.name – 字符串约束。键是 swf:taskList.name。

- workflowTypeVersion – 字符串约束。键是 swf:workflowTypeVersion。

> **Note**
>
> 限制的标签不能超过五个。

FailWorkflowExecution

- 您不能限制此操作的参数。

RecordMarker

- 您不能限制此操作的参数。

RequestCancelActivityTask

- 您不能限制此操作的参数。

RequestCancelExternalWorkflowExecution

- 您不能限制此操作的参数。

ScheduleActivityTask

- activityType.name – 字符串约束。键是 swf:activityType.name。
- activityType.version – 字符串约束。键是 swf:activityType.version。
- taskList.name – 字符串约束。键是 swf:taskList.name。

SignalExternalWorkflowExecution

- 您不能限制此操作的参数。

StartChildWorkflowExecution

- tagList.member.0 – 字符串约束。密钥是 swf:tagList.member.0

- tagList.member.1 – 字符串约束。密钥是 swf:tagList.member.1
- tagList.member.2 – 字符串约束。密钥是 swf:tagList.member.2
- tagList.member.3 – 字符串约束。密钥是 swf:tagList.member.3
- tagList.member.4 – 字符串约束。密钥是 swf:tagList.member.4
- taskList.name – 字符串约束。键是 swf:taskList.name。
- workflowType.name – 字符串约束。键是 swf:workflowType.name。
- workflowType.version – 字符串约束。键是 swf:workflowType.version。

> **Note**
>
> 限制的标签不能超过五个。

StartTimer

- 您不能限制此操作的参数。

基于标签的策略

Amazon SWF 支持基于标签的策略。例如，您可以限制包含具有 environment 键和 production 值的标签的 Amazon SWF 域：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "swf:*",
            "Resource": "arn:aws:swf:*:123456789012:/domain/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/environment": "production"}
            }
        }
    ]
}
```

此策略将Deny访问任何已标记为 environment/production 的域。

有关标记的更多信息，请参阅：

- 亚马逊 SWF 中的标签
- 使用 IAM 标签控制访问

适用于 Amazon S3 的 Amazon VPC 端点

> **Note**
>
> Amazon PrivateLink 目前仅在 “绝密-东 Amazon 部”、“ Amazon 秘密区域” 和 “中国” 区域提供支持。

如果您使用亚马逊虚拟私有云（ Amazon VPC ）托管 Amazon 资源，则可以在您的亚马逊 VPC 和亚马逊简单工作流程服务工作流程之间建立连接。您可以将此连接用于您的 Amazon SWF 工作流，而无需穿越公共互联网。

Amazon VPC 允许您在自定义虚拟网络中启动 Amazon 资源。可以使用 VPC 控制您的网络设置，例如 IP 地址范围、子网、路由表和网络网关。有关更多信息 VPCs，请参阅 Amazon VPC 用户指南。

要将 Amazon VPC 连接到 Amazon SWF，您必须先定义一个接口 VPC 端点，该端点可让您将 VPC 连接到其他 Amazon Web Services 服务服务。该端点提供了可靠且可扩展的连接，无需互联网网关、网络地址转换 (NAT) 实例或 VPN 连接。有关更多信息，请参阅 Amazon VPC 用户指南中的接口 VPC 端点 (Amazon PrivateLink)。

创建端点

您可以使用 Amazon Web Services Management Console、 Amazon Command Line Interface (Amazon CLI)、软件开发工具包、Amazon SWF API 或在您的 VPC 中创建 Amazon Amazon SWF 终端节点。 Amazon CloudFormation

有关使用 Amazon VPC 控制台或 Amazon CLI创建和配置端点的信息，请参阅 Amazon VPC 用户指南中的创建接口端点。

> **Note**
>
> 在创建端点时，请将 Amazon SNS 指定为 VPC 要连接的服务。在 Amazon VPC 控制台中，服务名称因 Amazon 区域而异。例如，在 Amazon 绝密——东部区域，亚马逊 SWF 的服务名称为 com.amazonaws。 us-iso-east-1.swf。

有关使用创建和配置终端节点的信息 Amazon CloudFormation，请参阅Amazon CloudFormation 用户指南中的 AWSEC2::: VPCEndpoint 资源。

Amazon VPC 端点策略

要控制对 Amazon SWF 的连接访问权限，您可以在创建 Amazon VPC 终端节点时附加 Amazon Identity and Access Management (IAM) 终端节点策略。您可以通过附加多个端点策略来创建复杂的 IAM 规则。有关更多信息，请参阅：

- Amazon Virtual Private Cloud Endpoint Policies for Amazon SWF
- 使用 VPC 端点控制对服务的访问

Amazon Virtual Private Cloud Endpoint Policies for Amazon SWF

您可以为 Amazon SWF 创建 Amazon VPC 端点策略，并在其中指定以下内容：

- 可执行操作的主体。
- 可执行的操作。
- 可对其执行操作的资源。

以下示例显示了一个 Amazon VPC 端点策略，该策略允许特定 IAM 角色在单个域上执行所有 Amazon SWF 操作。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "swf:*",
      "Resource": "arn:aws:swf:*:123456789012:/domain/myDomain",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:role/MyRole"
      }
    }
  ]
}
```

- 有关创建端点策略的更多信息，请参阅 Controlling Access to Services with VPC Endpoints。

- 有关如何使用 IAM 控制对您 Amazon 和 Amazon SWF 资源的访问权限的信息，请参阅。Amazon Simple Workflow Service 中的 Identity and Access Management

Amazon Simple Workflow Service 身份和访问权限故障排除

使用以下信息帮助您诊断和修复在使用 Amazon SWF 和 IAM 时可能遇到的常见问题。

主题

- 我无权在 Amazon SWF 中执行操作
- 我无权执行 iam：PassRole
- 我想允许我以外的人访问我 Amazon Web Services 账户 的 Amazon SWF 资源

我无权在 Amazon SWF 中执行操作

如果您收到一个错误，指明您无权执行某个操作，则必须更新策略以允许您执行该操作。

当 `mateojackson` 用户尝试使用控制台查看有关虚构 *`my-example-widget`* 资源的详细信息，但不拥有虚构 `swf:`*`GetWidget`* 权限时，会发生以下示例错误。

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

在此情况下，Mateo 的策略必须更新以允许其使用 `swf:`*`GetWidget`* 操作访问 *`my-example-widget`* 资源。

如果您需要帮助，请联系您的 Amazon 管理员。您的管理员是提供登录凭证的人。

我无权执行 iam：PassRole

如果您收到错误，指明您无权执行 `iam:PassRole` 操作，则必须更新策略以允许您将角色传递给 Amazon SWF。

有些 Amazon Web Services 服务 允许您将现有角色传递给该服务，而不是创建新的服务角色或服务相关角色。为此，您必须具有将角色传递到服务的权限。

当名为 `marymajor` 的 IAM 用户尝试使用控制台在 Amazon SWF 中执行操作时，会发生以下示例错误。但是，服务必须具有服务角色所授予的权限才可执行此操作。Mary 不具有将角色传递到服务的权限。

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

在这种情况下，必须更新 Mary 的策略以允许她执行 `iam:PassRole` 操作。

如果您需要帮助，请联系您的 Amazon 管理员。您的管理员是提供登录凭证的人。

我想允许我以外的人访问我 Amazon Web Services 账户 的 Amazon SWF 资源

您可以创建一个角色，以便其他账户中的用户或您组织外的人员可以使用该角色来访问您的资源。您可以指定谁值得信赖，可以代入角色。对于支持基于资源的策略或访问控制列表 (ACLs) 的服务，您可以使用这些策略向人们授予访问您的资源的权限。

要了解更多信息，请参阅以下内容：

- 要了解 Amazon SWF 是否支持这些功能，请参阅 Amazon Simple Workflow Service works 如何与 IAM 结合使用。
- 要了解如何提供对您拥有的资源的访问权限 Amazon Web Services 账户 ，请参阅 IAM 用户指南中的向您拥有 Amazon Web Services 账户 的另一个 IAM 用户提供访问权限。
- 要了解如何向第三方提供对您的资源的访问权限 Amazon Web Services 账户，请参阅 IAM 用户指南中的向第三方提供访问权限。 Amazon Web Services 账户
- 要了解如何通过身份联合验证提供访问权限，请参阅《IAM 用户指南》中的为经过外部身份验证的用户（身份联合验证）提供访问权限。
- 要了解使用角色和基于资源的策略进行跨账户访问之间的差别，请参阅《IAM 用户指南》中的 IAM 中的跨账户资源访问。

日志记录和监控

此部分提供了有关对 Amazon SNS 主题进行日志记录和监控的信息。

主题

- 的亚马逊 SWF 指标 CloudWatch
- 查看亚马逊 SWF 指标以使用 CloudWatch Amazon Web Services Management Console
- 使用录制 API 调用 Amazon CloudTrail
- EventBridge 亚马逊 SWF 的执行状态已更改

- Amazon 用户通知服务 与 Amazon 简单工作流程服务配合使用

的亚马逊 SWF 指标 CloudWatch

Amazon SWF 现在提供了指标 CloudWatch ，您可以使用这些指标来跟踪您的工作流程和活动，并根据您选择的阈值设置警报。您可以使用查看指标 Amazon Web Services Management Console。有关更多信息，请参阅 查看亚马逊 SWF 指标以使用 CloudWatch Amazon Web Services Management Console。

主题

- Amazon SWF 指标的报告单位
- API 和决策事件指标
- Amazon SWF 指标
- Amazon SWF 非 ASCII 资源名称和尺寸 CloudWatch

Amazon SWF 指标的报告单位

报告时间间隔的指标

Amazon SWF 的某些指标 CloudWatch 是时间间隔，始终以毫秒为单位。该 CloudWatch 单位被报告为`Time`。这些指标通常对应于可以设置工作流程和活动超时的工作流程执行的各阶段，并具有类似的名称。

例如，`DecisionTaskStartToCloseTime` 指标度量决策任务开始执行之后直到完成所用的时间，该时间就是可为其设置 `DecisionTaskStartToCloseTimeout` 值的时间段。

要获得各工作流程阶段的总示意图和了解其在工作流程和活动生命周期内何时发生，请参阅 Amazon SWF 超时类型 。

报告计数的指标

一些用于将结果 CloudWatch 报告为计数的 Amazon SWF 指标。例如，`WorkflowsCanceled` 将结果记录为 1 或 0，指示是否已取消工作流。值 0 并不表示未报告该指标，仅表示该指标描述的条件未出现。

该报告的某些 Amazon SWF 指标为 CloudWatch 每 CloudWatch 秒计数。`Count`例如`ProvisionedRefillRate`，报告为 a `Count` in CloudWatch，表示每秒请求`Count`的速率。

对于计数指标，最小值和最大值将始终是 0 或 1，但平均值将是 0 到 1 范围内的一个值。

API 和决策事件指标

您可以在中监控 API 和决策事件 CloudWatch ，以深入了解您的使用情况和容量。请参阅 Amazon SWF 中的基本工作流程概念 部分中的 deciders，以及 Amazon Simple Workflow Service API Reference 中的 Decision 主题。

您还可以监控这些限制，以便在接近 Amazon SWF 节流限制时发出警报。请参阅 Amazon SWF 节流限额，以了解这些限制及其默认设置的说明。这些限制旨在防止不正确的工作流占用过多的系统资源。要请求提高您的限制，请参阅：???。

作为最佳实践，您应在 API 或决策事件容量的 60% 左右配置 CloudWatch 警报。这让您可以在启用 Amazon SWF 节流之前调整工作流或请求增加服务限制。根据您的调用的 burstiness，您可以配置不同警报，以便在接近您的服务限制时发出通知：

- 如果您的流量有明显的峰值，请将警报设置为 `ProvisionedBucketSize` 限制的 60%。
- 如果您的调用具有相对稳定的速率，请将相关 API 和决策事件的警报设置为 `ProvisionedRefillRate` 限制的 60%。

Amazon SWF 指标

Amazon SWF 提供以下指标：

指标	描述
`DecisionTaskScheduleToStartTime`	计划决策任务的时刻与工作人员选取并启动该决策任务的时刻之间的时间间隔，以毫秒为单位。 CloudWatch 单位：`Time` 维度：`Domain, WorkflowTypeName, WorkflowTypeVersion` 有效统计数据：`Average, Minimum, Maximum`
`DecisionTaskStartToCloseTime`	启动活动任务的时刻与决策任务结束的时刻之间的时间间隔，以毫秒为单位。 CloudWatch 单位：`Time`

指标	描述
	维度：Domain, WorkflowTypeName, WorkflowTypeVersion 有效统计数据：Average, Minimum, Maximum
DecisionTasksCompleted	已完成的决策任务的计数。 CloudWatch 单位：Count 维度：Domain, WorkflowTypeName, WorkflowTypeVersion 有效统计数据：Sum
PendingTasks	针对某个特定任务列表在 1 分钟间隔内待办任务的计数。 CloudWatch 单位：Count 维度：Domain, TaskListName 有效统计数据：Sum
StartedDecisionTasksTimedOutOnClose	已启动但在结束时已超时的决策任务的计数。 CloudWatch 单位：Count 维度：Domain, WorkflowTypeName, WorkflowTypeVersion 有效统计数据：Sum
WorkflowStartToCloseTime	启动工作流的时刻与工作流结束的时刻之间的时间，以毫秒为单位。 CloudWatch 单位：Time 维度：Domain, WorkflowTypeName, WorkflowTypeVersion 有效统计数据：Average, Minimum, Maximum

指标	描述
WorkflowsCanceled	取消的工作流程的计数。 CloudWatch 单位：Count 维度：Domain, WorkflowTypeName, WorkflowTypeVersion 有效统计数据：Sum
WorkflowsCompleted	完成的工作流程的计数。 CloudWatch 单位：Count 维度：Domain, WorkflowTypeName, WorkflowTypeVersion 有效统计数据：Sum
WorkflowsContinuedAsNew	继续作为新工作流程的工作流程计数。 CloudWatch 单位：Count 维度：Domain, WorkflowTypeName, WorkflowTypeVersion 有效统计数据：Sum
WorkflowsFailed	失败的工作流的计数。 CloudWatch 单位：Count 维度：Domain, WorkflowTypeName, WorkflowTypeVersion 有效统计数据：Sum

指标	描述
WorkflowsTerminated	终止的工作流的计数。 CloudWatch 单位：Count 维度：Cause, Domain, WorkflowTypeName, WorkflowTypeVersion 有效统计数据：Sum
WorkflowsTimedOut	出于任何原因而超时的工作流程的计数。 CloudWatch 单位：Count 维度：Domain, WorkflowTypeName, WorkflowTypeVersion 有效统计数据：Sum
ActivityTaskScheduleToCloseTime	计划活动的时刻与该活动结束的时刻之间的时间间隔，以毫秒为单位。 CloudWatch 单位：Time 维度：Domain, ActivityTypeName, ActivityTypeVersion 有效统计数据：Average, Minimum, Maximum
ActivityTaskScheduleToStartTime	计划活动任务的时刻与该活动任务启动的时刻之间的时间间隔，以毫秒为单位。 CloudWatch 单位：Time 维度：Domain, ActivityTypeName, ActivityTypeVersion 有效统计数据：Average, Minimum, Maximum

指标	描述
ActivityTaskStartToCloseTime	启动活动任务的时刻与活动任务结束的时刻之间的时间间隔，以毫秒为单位。 CloudWatch 单位：Time 维度：Domain, ActivityTypeName, ActivityTypeVersion 有效统计数据：Average, Minimum, Maximum
ActivityTasksCanceled	取消的活动任务的计数。 CloudWatch 单位：Count 维度：Domain, ActivityTypeName, ActivityTypeVersion 有效统计数据：Sum
ActivityTasksCompleted	已完成的活动任务的计数。 CloudWatch 单位：Count 维度：Domain, ActivityTypeName, ActivityTypeVersion 有效统计数据：Sum
ActivityTasksFailed	失败的活动任务的计数。 CloudWatch 单位：Count 维度：Domain, ActivityTypeName, ActivityTypeVersion 有效统计数据：Sum

指标	描述
ScheduledActivityTasksTimedOutOnClose	已计划但在结束时已超时的活动任务的计数。 CloudWatch 单位：Count 维度：Domain, ActivityTypeName, ActivityTypeVersion 有效统计数据：Sum
ScheduledActivityTasksTimedOutOnStart	已计划但在启动时已超时的活动任务的计数。 CloudWatch 单位：Count 维度：Domain, ActivityTypeName, ActivityTypeVersion 有效统计数据：Sum
StartedActivityTasksTimedOutOnClose	已启动但在结束时已超时的活动任务的计数。 CloudWatch 单位：Count 维度：Domain, ActivityTypeName, ActivityTypeVersion 有效统计数据：Sum
StartedActivityTasksTimedOutOnHeartbeat	已启动但因检测信号超时而发生超时的活动任务的计数。 CloudWatch 单位：Count 维度：Domain, ActivityTypeName, ActivityTypeVersion 有效统计数据：Sum

指标	描述
ThrottledEvents	已被限制的请求的计数。 CloudWatch 单位：Count 维度：APIName, DecisionName, ThrottlingScope 有效统计数据：Sum
ProvisionedBucketSize	每秒可用请求的计数。 维度：APIName, DecisionName 有效统计数据：Minimum
ConsumedCapacity	每秒请求的计数。 CloudWatch 单位：Count 维度：APIName, DecisionName 有效统计数据：Sum
ConsumedLimit	已使用的一般限额。 维度：GeneralLimitType
ProvisionedRefillRate	每秒允许进入存储桶中的请求的计数。 维度：APIName, DecisionName 有效统计数据：Minimum
ProvisionedLimit	为账户预置的一般限额。 维度：GeneralLimitType

维度	描述
Domain	按照工作流或活动在其中运行的 Amazon SWF 域来筛选数据。

维度	描述
ActivityTypeName	按照活动类型的名称来筛选数据。
ActivityTypeVersion	按照活动类型的版本来筛选数据。
WorkflowTypeName	按照此工作流执行的工作流类型的名称来筛选数据。
WorkflowTypeVersion	按照此工作流执行的工作流类型的版本来筛选数据。
APIName	按照指定 API 名称的 API 来筛选数据。
DecisionName	按照指定决策名称来筛选数据。
TaskListName	按照指定任务列表名称来筛选数据。
TaskListClassification	按照任务列表的分类来筛选数据。决定任务列表的值为“D”，活动任务列表的值为“A”。
ThrottlingScope	将数据筛选到指定的限制范围。超过账户级别配额时值为 “账户”，超过工作流级别配额时值为 “工作流程”。

Amazon SWF 非 ASCII 资源名称和尺寸 CloudWatch

Amazon SWF 允许在资源名称中使用非 ASCII 字符，例如和。 TaskList DomainName但是，CloudWatch 指标的维度值只能包含可打印的 ASCII 字符。为确保 Amazon SWF 使用符合要求的维度值，不符合这些CloudWatch 要求的 Amazon SWF 资源名称会被转换并附加校验和，如下所示：

- 任何非 ASCII 字符都将替换为 ？
- 如有必要，输入字符串或转换后的字符串将被截断。这样可以确保在附加校验和时，新的字符串长度不会超过最大值。 CloudWatch
- 由于任何非 ASCII 字符都会转换为？，因此转换前有所不同的某些 CloudWatch 指标维度值在转换后可能看起来相同。为了帮助区分它们，在资源名称后面附加了一个下划线 (_)，后跟原始资源名称的 SHA256 校验和的前 16 个字符。

转换示例：

- `test àpple` 将转换为 `test ?pple_82cc5b8e3a771d12`

- àòà 将转换为 ???_2fec5edbb2c05c22
- TaskList 名称àpplé和都âpplè将转换为?ppl?，并且将完全相同。附加校验和会返回不同的值，即 ?ppl?_f39a36df9d85a69d 和 ?ppl?_da3efb4f11dd0f7f。

> **Tip**
>
> 您可以生成自己的 SHA256 校验和。例如，要使用 shasum 命令行工具，请执行以下操作：echo -n "<the original resource name>" | shasum -a 256 | cut -c1-16

查看亚马逊 SWF 指标以使用 CloudWatch Amazon Web Services Management Console

亚马逊 CloudWatch 为亚马逊 SWF 工作流程和活动提供了许多可查看的指标。您可以使用 Amazon Web Services Management Console查看这些指标并为 Amazon SWF 工作流的执行设置警报。您必须登录控制台才能继续。

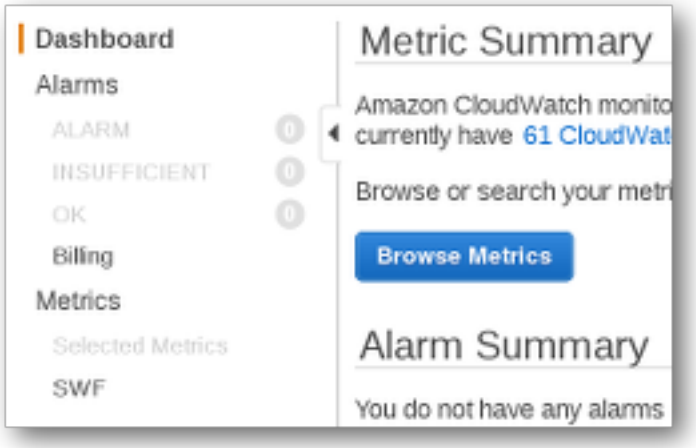
有关每一个可用指标的说明，请参阅的亚马逊 SWF 指标 CloudWatch。

主题

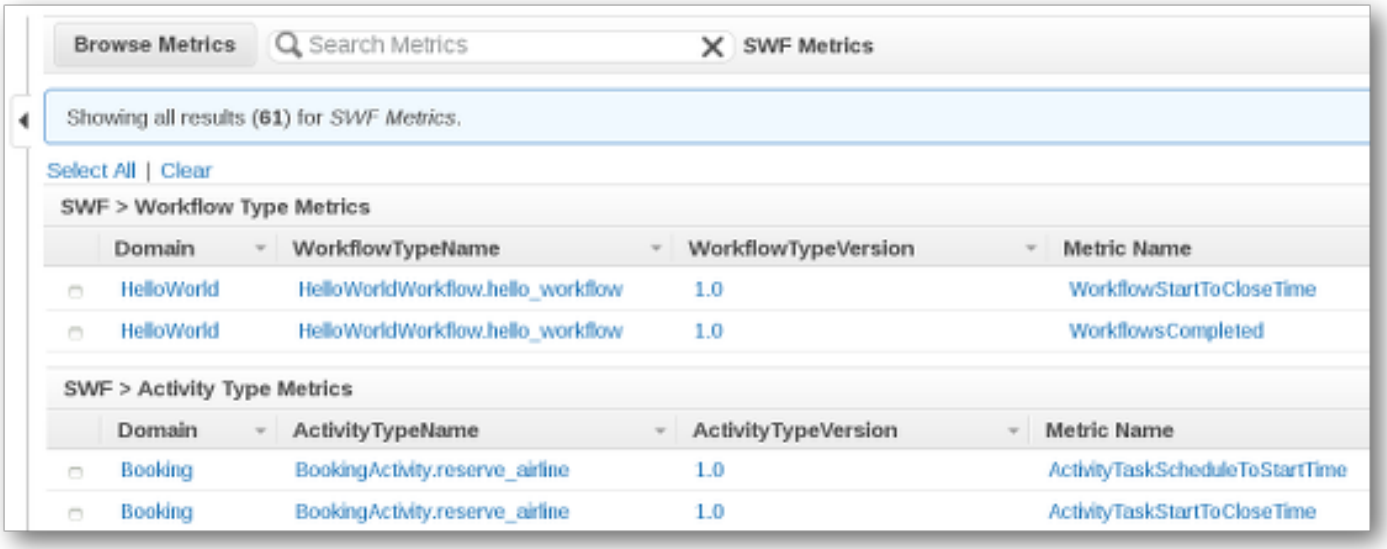
查看 指标

查看您的 Amazon SWF 指标

1. 登录 Amazon Web Services Management Console 并打开 CloudWatch 控制台，网址为https://console.aws.amazon.com/cloudwatch/。
2. 在导航窗格的 Metrics 下，选择 SWF。



如果最近运行过任何工作流程执行，您会看到两个指标列表：Workflow Type Metrics (工作流程类型指) 和 Activity Type Metrics (活动类型指标)。



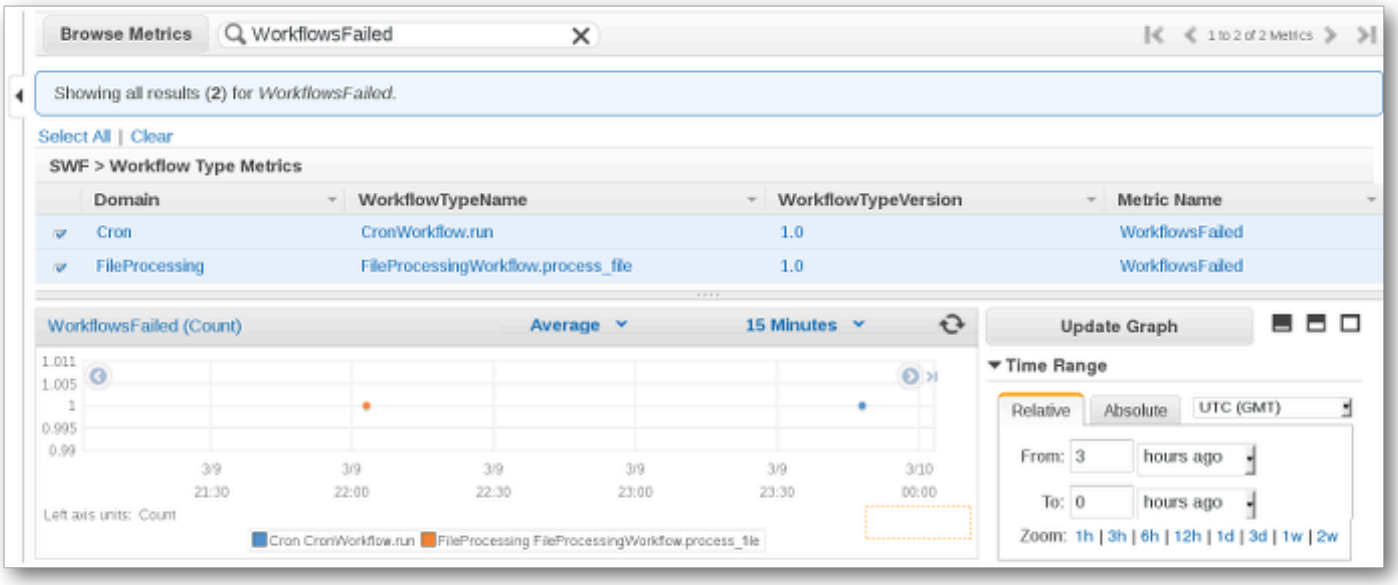
> **Note**
>
> 一开始您可能只看到 Workflow Type Metrics (工作流程类型指标)；实际上 Activity Type Metrics (活动类型指标)也显示在相同视图中，可能需要向下滚动才能看到。

同时最多显示 50 个最新指标，按工作流程和活动指标分开显示。

您可以使用提供的任何维度，按列表中每一列上方的交互式标题对指标排序。对于工作流程，维度包括域WorkflowTypeNameWorkflowTypeVersion、和指标名称。对于活动，维度包括域ActivityTypeNameActivityTypeVersion、和指标名称。

的亚马逊 SWF 指标 CloudWatch中介绍了各指标类型。

您可以选择列表中指标行旁边的框来查看指标图，还可使用图形视图右侧的 Time Range 控件来更改图形参数。



要查看图形中任何点的详细信息，请将光标放在该图形点上。此时将显示该点的维度的详细信息。

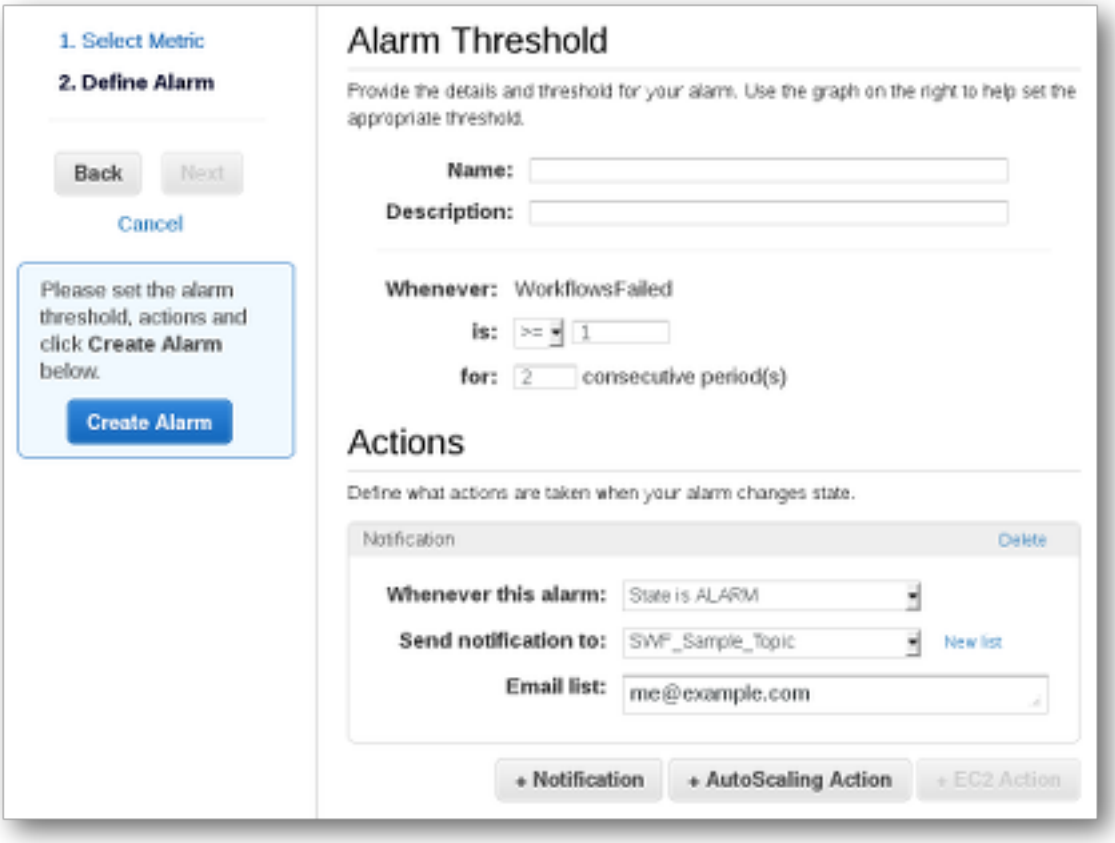
有关使用 CloudWatch 指标的更多信息，请参阅 Amazon CloudWatch 用户指南中的查看、绘制和发布指标。

设置警报

您可以使用 CloudWatch 警报来执行操作，例如在达到警报阈值时通知您。例如，您可以设置警报，在 `WorkflowsFailed` 指标升高到超过特定阈值时向 SNS 主题发送通知，或发送电子邮件。

针对您的任何指标设置警报

1. 选择指标对应的框即可选择该指标。
2. 在图形右边的 Tools 控件中，选择 Create Alarm。
3. 在 Define Alarm (定义警报) 屏幕中，输入警报阈值、句点参数和要执行的操作。



有关设置和使用 CloudWatch 警报的更多信息，请参阅亚马逊 CloudWatch 用户指南中的创建亚马逊 CloudWatch 警报。

使用录制 API 调用 Amazon CloudTrail

Amazon Simple Workflow Service 与Amazon CloudTrail一项服务集成，该服务提供用户、角色或角色所执行操作的记录 Amazon Web Services 服务。 CloudTrail 将 Amazon SWF 的所有 API 调用捕获为事件。捕获的调用包括来自 Amazon SWF 控制台的调用和对 Amazon SWF API 操作的代码调用。使用收集的信息 CloudTrail，您可以确定向 Amazon SWF 发出的请求、发出请求的 IP 地址、发出请求的时间以及其他详细信息。

每个事件或日志条目都包含有关生成请求的人员信息。身份信息有助于您确定以下内容：

- 请求是使用根用户凭证还是用户凭证发出的。
- 请求是否代表 IAM Identity Center 用户发出。
- 请求是使用角色还是联合用户的临时安全凭证发出的。
- 请求是否由其他 Amazon Web Services 服务发出。

CloudTrail 在您创建账户 Amazon Web Services 账户 时在您的账户中处于活动状态，并且您自动可以访问 CloudTrail 活动历史记录。 CloudTrail 事件历史记录提供了过去 90 天中记录的管理事件的可查看、可搜索、可下载且不可变的记录。 Amazon Web Services 区域有关更多信息，请参阅《Amazon CloudTrail 用户指南》中的 “使用 CloudTrail 事件历史记录”。查看活动历史记录不 CloudTrail收取任何费用。

要持续记录 Amazon Web Services 账户 过去 90 天内的事件，请创建跟踪或 CloudTrailLake 事件数据存储。

CloudTrail 步道

跟踪允许 CloudTrail 将日志文件传输到 Amazon S3 存储桶。使用创建的所有跟踪 Amazon Web Services Management Console 都是多区域的。您可以通过使用 Amazon CLI创建单区域或多区域跟踪。建议创建多区域跟踪，因为您可以捕获账户 Amazon Web Services 区域 中的所有活动。如果您创建单区域跟踪，则只能查看跟踪的 Amazon Web Services 区域中记录的事件。有关跟踪的更多信息，请参阅《Amazon CloudTrail 用户指南》中的为您的 Amazon Web Services 账户创建跟踪和为组织创建跟踪。

通过创建跟踪，您可以免费将正在进行的管理事件的一份副本传送到您的 Amazon S3 存储桶，但是，会收取 Amazon S3 存储费用。 CloudTrail 有关 CloudTrail 定价的更多信息，请参阅Amazon CloudTrail 定价。有关 Amazon S3 定价的信息，请参阅 Amazon S3 定价。

CloudTrail 湖泊事件数据存储

CloudTrail L@@ ak e 允许你对自己的事件运行基于 SQL 的查询。 CloudTrail Lake 将基于行的 JSON 格式的现有事件转换为 Apache ORC 格式。ORC 是一种针对快速检索数据进行优化的列式存储格式。事件将被聚合到事件数据存储中，它是基于您通过应用高级事件选择器选择的条件的不可变的事件集合。应用于事件数据存储的选择器用于控制哪些事件持续存在并可供您查询。有关 CloudTrail Lake 的更多信息，请参阅《Amazon CloudTrail 用户指南》中的 “使用 Amazon CloudTrail Lake”。

CloudTrail 湖泊事件数据存储和查询会产生费用。创建事件数据存储时，您可以选择要用于事件数据存储的定价选项。定价选项决定了摄取和存储事件的成本，以及事件数据存储的默认和最长保留期。有关 CloudTrail 定价的更多信息，请参阅Amazon CloudTrail 定价。

中的数据事件 CloudTrail

数据事件可提供对资源或在资源中所执行资源操作（例如，读取或写入 Amazon S3 对象）的相关信息。这些也称为数据层面操作。数据事件通常是高容量活动。默认情况下， CloudTrail 不记录数据事件。 CloudTrail 事件历史记录不记录数据事件。

记录数据事件将收取额外费用。有关 CloudTrail 定价的更多信息，请参阅Amazon CloudTrail 定价。

您可以使用 CloudTrail 控制台或 CloudTrail API 操作记录 Amazon SWF 资源类型的数据事件。Amazon CLI有关如何记录数据事件的更多信息，请参阅《Amazon CloudTrail 用户指南》中的使用 Amazon Web Services Management Console记录数据事件和使用 Amazon Command Line Interface 记录数据事件。

下表列出了您可以记录数据事件的 Amazon SWF 资源类型。数据事件类型列显示要从 CloudTrail 控制台上的数据事件类型列表中选择的值。res ources.type 值列显示该`resources.type`值，您将在使用或配置高级事件选择器时指定该值。 Amazon CLI CloudTrail APIs“ APIs 记录到的数据 CloudTrail” 列显示了 CloudTrail 针对该资源类型记录的 API 调用。

您可以将高级事件选择器配置为在 `eventName`、`readOnly` 和 `resources.ARN` 字段上进行筛选，从而仅记录那些对您很重要的事件。有关这些字段的更多信息，请参阅 AdvancedFieldSelector《Amazon CloudTrail API 参考》中的 。

数据事件类型	resources.type 值	数据 APIs 已记录到 CloudTrail
SWF 域名	`AWS::SWF::Domain`	工作流程事件 • CountClosedWorkflowExecutions • CountOpenWorkflowExecutions • DescribeWorkflowExecution • ListClosedWorkflowExecutions • ListOpenWorkflowExecutions • GetWorkflowExecutionHistory • RequestCancelWorkflowExecution • SignalWorkflowExecution • StartWorkflowExecution • TerminateWorkflowExecution

数据事件类型	resources.type 值	数据 APIs 已记录到 CloudTrail
		任务事件 • CountPendingActivityTasks • PollForDecisionTask • PollForActivityTask • RecordActivityTaskHeartbeat • RespondActivityTaskCanceled • RespondActivityTaskCompleted • RespondActivityTaskFailed • RespondDecisionTaskCompleted 决策事件 • CancelTimer • CancelWorkflowExecution • CompleteWorkflowExecution • ContinueAsNewWorkflowExecution • FailWorkflowExecution • RecordMarker • RequestCancelActivityTask • RequestCancelExternalWorkflowExecution • ScheduleActivityTask • ScheduleLambdaFunction • SignalExternalWorkflowExecution

数据事件类型	resources.type 值	数据 APIs 已记录到 CloudTrail
		• StartChildWorkflow Execution • StartTimer

> **CloudTrail 活动和 RespondDecisionTaskCompleted**
>
> 这些区域有：RespondDecisionTaskCompletedaction 在请求有效载荷中获取决策列表。完成的调用将发出 N+1 个 CloudTrail 数据事件，每个决策一个，API 调用本身一个。数据事件和 API 事件都将具有相同的请求 ID。

中的管理活动 CloudTrail

管理事件提供有关对中的资源执行的管理操作的信息 Amazon Web Services 账户。这些也称为控制面板操作。默认情况下， CloudTrail 记录管理事件。

Amazon Simple Workflow Service 将以下控制平面操作记录 CloudTrail 为管理事件。

域名事件

- RegisterDomain
- DescribeDomain
- ListDomains
- DeprecateDomain
- UndeprecateDomain

活动事件

- RegisterActivityType
- DescribeActivityType
- ListActivityTypes
- DeprecateActivityType
- UndeprecateActivityType

- DeleteActivityType

WorkflowType 事件

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

标签：活动

- TagResource
- UntagResource
- ListTagsforResource

事件示例

事件代表来自任何来源的单个请求，包括有关所请求的 API 操作、操作的日期和时间、请求参数等的信息。 CloudTrail 日志文件不是公共 API 调用的有序堆栈跟踪，因此事件不会按任何特定顺序出现。

以下示例显示了一个演示该`CountClosedWorkflowExecutions`操作 CloudTrail 的事件。

```
{
    "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"
  }
}
```


有关 CloudTrail 录音内容的信息，请参阅《Amazon CloudTrail 用户指南》中的CloudTrail录制内容。

EventBridge 亚马逊 SWF 的执行状态已更改

您可以使用 Amazon EventBridge 来响应 Amazon 资源中的状态变化或事件。当 Amazon SWF 发出事件时，它始终会转到您账户的默认 EventBridge 事件总线。您可以为事件创建规则，将其与默认事件总线相关联，并指定在 EventBridge 收到与该规则匹配的事件时要采取的目标操作。这样一来，您可以监控您的工作流，而不必持续不断地使用 `GetWorkflowExecutionHistory` API 进行轮询。根据工作流程执行的变化，您可以使用 EventBridge 目标来调用 Amazon Lambda 函数、向亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 主题发布消息等。

您可以使用 `DescribeWorkflowExecution` 查看执行状态更改事件的全部内容。

有关更多信息，请参阅 Amazon EventBridge 用户指南。

EventBridge 事件

历史记录事件类型包含执行状态的更改。每个事件的 `detail` 部分至少包含以下参数：

- `eventId`: 显示的事件 ID GetWorkflowExecutionHistory。
- `workflowExecutionDetail`：发出事件时工作流的状态。
- `eventType`：历史记录事件类型，以下任一类型：
  - `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 事件示例

以下是 Amazon SWF 向发送事件的示例： EventBridge

主题

- 执行已启动
- 执行已完成
- 执行已失败
- 执行超时
- 执行已终止

在每种情况下，事件数据中的 detail 部分都提供与 DescribeWorkflowExecution API 相同的信息。executionStatus 字段表示事件发送时的执行状态，即 OPEN 或 CLOSED。

执行已启动


```
{
  "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,
    }
  }
}
```

执行已完成

```
{
  "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,
    }
  }
}
```

执行已失败

```
{
  "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,
    }
  }
}
```

执行超时


```
{
  "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,
    }
  }
}
```

执行已终止

```
{
  "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 用户通知服务 与 Amazon 简单工作流程服务配合使用

您可以使用 Amazon 用户通知服务 来设置传送渠道，以获得有关 Amazon Simple Workflow Service 事件的通知。当事件与指定的规则匹配时，会收到通知。您可以通过多种渠道接收事件通知，包括电子邮件、聊天应用程序中的 Amazon Q D eveloper 聊天通知或Amazon Console Mobile Application推送通知。您还可以在控制台通知中心查看通知。 用户通知服务 支持聚合，这可以减少在具体事件期间收到的通知数量。

Amazon Simple Workflow Service 合规性验证

作为多项 Amazon 合规性计划的一部分，第三方审计员将评估 Amazon Simple Workflow Service 的安全性和合规性。其中包括 SOC、PCI、FedRAMP、HIPAA 及其他。

有关特定合规计划范围内的 Amazon 服务列表，请参阅合规计划划分的范围内的Amazon 服务。有关一般信息，请参阅合规计划。

您可以使用下载第三方审计报告 Amazon Artifact。有关更多信息，请参阅在 Artifact 中 tif Amazon act 中下载报告。

您在使用 Amazon SWF 时的合规性责任取决于您数据的敏感度、您公司的合规性目标以及适用的法律法规。 Amazon 提供以下资源来帮助满足合规性要求：

- 安全与合规性快速入门指南安全与合规性快速入门指南 — 这些部署指南讨论了架构注意事项，并提供了在上部署以安全性和合规性为重点的基准环境的步骤。 Amazon
- HIPAA 安全与合规架构白皮书 — 本白皮书描述了公司如何使用来 Amazon 创建符合 HIPAA 标准的应用程序。
- 合规资源 — 此工作簿和指南集可能适用于您的行业和所在地区。
- 使用Amazon Config 开发人员指南中的规则评估资源 — 该 Amazon Config 服务评估您的资源配置在多大程度上符合内部实践、行业准则和法规。
- Amazon Security Hub— 此 Amazon 服务可全面了解您的安全状态 Amazon ，帮助您检查是否符合安全行业标准和最佳实践。

Amazon Simple Workflow Service 中的恢复能力

Amazon 全球基础设施是围绕 Amazon 区域和可用区构建的。 Amazon 区域提供多个物理隔离和隔离的可用区，这些可用区通过低延迟、高吞吐量和高度冗余的网络相连。利用可用区，您可以设计和操作在可用区之间无中断地自动实现失效转移的应用程序和数据库。与传统的单个或多个数据中心基础结构相比，可用区具有更高的可用性、容错性和可扩展性。

有关 Amazon 区域和可用区的更多信息，请参阅Amazon 全球基础设施。

除了 Amazon 全球基础设施外，Amazon SWF 还提供多项功能来帮助支持您的数据弹性和备份需求。

Amazon Simple Workflow Service 中的基础设施安全性

作为一项托管服务，受 Amazon 全球网络安全的保护。有关 Amazon 安全服务以及如何 Amazon 保护基础设施的信息，请参阅Amazon 云安全。要使用基础设施安全的最佳实践来设计您的 Amazon 环境，请参阅 S Amazon ecurity Pillar Well-Architected Fram ework 中的基础设施保护。

您可以使用 Amazon 已发布的 API 调用通过网络进行访问。客户端必须支持以下内容：

- 传输层安全性协议（TLS）。我们要求使用 TLS 1.2，建议使用 TLS 1.3。
- 具有完全向前保密（PFS）的密码套件，例如 DHE（临时 Diffie-Hellman）或 ECDHE（临时椭圆曲线 Diffie-Hellman）。大多数现代系统（如 Java 7 及更高版本）都支持这些模式。

此外，必须使用访问密钥 ID 和与 IAM 主体关联的秘密访问密钥来对请求进行签名。或者，您可以使用 Amazon Security Token Service（Amazon STS）生成临时安全凭证来对请求进行签名。

您可以从任何网络位置调用这些 API 操作，但 Amazon SWF 不支持基于资源的访问策略，其中可能包含基于源 IP 地址的限制。您还可以使用 Amazon SWF 策略来控制来自特定亚马逊虚拟私有云（亚马逊 VPC）终端节点或特定终端节点的访问。 VPCs实际上，这可以将对给定 Amazon SWF 资源的网络访问与网络中的 Amazon 特定 VPC 隔离开来。

Amazon Simple Workflow Service 中的配置和漏洞分析

配置和 IT 控制由您（我们的客户）共同 Amazon 负责。有关更多信息，请参阅责任 Amazon 共担模型。

Amazon CLI 与 Amazon 一起使用简单工作流程服务

Amazon Simple Workflow Service 的许多功能都可以通过 Amazon CLI 访问。为使用 Amazon SWF Amazon CLI 提供了另一种选择， Amazon Web Services Management Console 或者在某些情况下，也可以使用 Amazon SWF API 和进行编程。 Amazon Flow Framework

例如，您可以使用 Amazon CLI 来注册新的工作流程类型：

```
aws swf register-workflow-type --domain MyDomain --name "MySimpleWorkflow" --workflow-version "v1"
```

您还可以列出已注册的工作流类型：

```
aws swf list-workflow-types --domain MyDomain --registration-status REGISTERED
```

以下显示了 JSON 中默认输出的示例：

```
{
    "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"
            }
        }
    ]
}
```

中的 Amazon SWF 命令 Amazon CLI 提供了启动和管理工作流程执行、轮询活动任务、记录任务心跳等功能！有关 Amazon SWF 命令的完整列表、可用参数的说明及其使用方法示例，请参阅 Amazon CLI Command Reference 中的 Amazon SWF 命令。

这些 Amazon CLI 命令严格遵循亚马逊 SWF API，因此您可以使用 Amazon CLI 来了解底层的亚马逊 SWF API。您还可以使用现有的 API 知识创建代码原型或在命令行上执行 Amazon SWF 操作。

要了解更多信息 Amazon CLI，请参阅《Amazon Command Line Interface 用户指南》。

与亚马逊 SWF 合作 APIs

除了使用中描述的 Amazon SDKs 与之一起开发 Amazon SDKs，您还可以直接使用 HTTP API。

要使用该 API，您可将 HTTP 请求发送到 SWF 终端节点，该终端节点与您要为域、工作流和活动使用的区域匹配。有关发出针对 Amazon SWF 的 HTTP 请求的更多信息，请参阅 向 Amazon SWF 发出 HTTP 请求。

本部分提供了有关通过 HTTP API 使用 Amazon SWF 开发工作流的基本信息。本节提供了更多高级功能，例如使用计时器、记录工作流程 CloudTrail 和标记工作流程。Amazon SWF 中的基本工作流程概念

主题

向 Amazon SWF 发出 HTTP 请求

如果您不使用其中一个 Amazon SDKs，则可以使用 POST 请求方法通过 HTTP 执行亚马逊简单工作流服务 (Amazon SWF) 操作。POST 方法要求您在请求标头中指定操作并在请求正文中以 JSON 格式提供操作数据。

HTTP 标头内容

Amazon SWF 要求在 HTTP 请求标头中包含以下信息：

- host Amazon SWF 端点。
- x-amz-date您必须在 HTTP Date 标头或 Amazon x-amz-date header（某些 HTTP 客户端库不允许您设置Date标头）中提供时间戳。当 x-amz-date 标头呈现时，系统在验证请求身份时会忽略任何 Date 标头。

  必须利用以下三种格式中的一种来指定数据，如 HTTP/1.1 RFC 中所规定：

  - 格林威治时间 1994 年 11 月 6 日，星期日 08:49:37（RFC 822，由 RFC 1123 更新）
  - 格林威治时间 1994 年 11 月 6 日，星期日 08:49:37（RFC 850，由 RFC 1036 废弃）
  - 1994 年 11 月 6 日 08:49:37，星期日（ANSI C 的 asctime() 格式）
- x-amzn-authorization已签名的请求参数格式如下：

```
AWS3 AWSAccessKeyId=####,Algorithm=HmacSHA256, [,SignedHeaders=Header1;Header2;...]
Signature=S(StringToSign)
```

  AWS3— 这是一个 Amazon 特定于实现的标签，表示用于签署请求的身份验证版本（目前，对于 Amazon SWF，此值始终为）。AWS3

  AWSAccessKeyId— 您的 Amazon 访问密钥 ID。

  Algorithm— 用于创建的 HMAC-SHA 值的算法 string-to-sign，例如或。HmacSHA256 HmacSHA1

  Signature— Base64（算法（StringToSign, SigningKey））。有关详细信息，请参阅 计算 Amazon SWF 的 HMAC-SHA 签名

  SignedHeaders—（可选）如果存在，则必须包含规范 HttpHeaders 化计算中使用的所有 HTTP 标头的列表。必须用一个分号字符 (;)（ASCII 字符 59）分隔列表值。
- x-amz-target – 请求的目标服务和数据操作，格式如下：

  com.amazonaws.swf.service.model.SimpleWorkflowService. + *<action>*

  例如，com.amazonaws.swf.service.model.SimpleWorkflowService.RegisterDomain
- content-type – 类型需要将 JSON 和字符集指定为 application/json; charset=UTF-8

以下示例为创建域所用的 HTTP 请求的标头。

```
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"}
```

此处为对应 HTTP 响应的示例。

```
HTTP/1.1 200 OK
Content-Length: 0
Content-Type: application/json
x-amzn-RequestId: 4ec4ac3f-3e16-11e1-9b11-7182192d0b57
```

HTTP 正文内容

HTTP 请求的正文包含 HTTP 请求标头中指定的操作数据。使用 JSON 数据格式可以同时传递数据值和数据结构。元素可通过括号嵌套在其它元素内。例如，下面显示了一个请求，该请求使用 Unix 时间注释列出在两个指定时间点之间开始的所有工作流执行。

```
{
 "domain": "867530901",
 "startTimeFilter":
```

```
 {
   "oldestDate": 1325376070,
  "latestDate": 1356998399
 },
 "tagFilter":
 {
   "tag": "music purchase"
 }
}
```

Amazon SWF JSON 请求和响应示例

下面的示例显示了一个对 Amazon SWF 的请求，用于获取对我们在前面创建的域的描述。然后，它显示了 Amazon SWF 响应。

HTTP POST 请求

```
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 响应

```
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"}
}
```

注意协议 (HTTP/1.1) 后面有一个状态代码 (200)。200 的代码值表示操作成功。

Amazon SWF 不对空值执行序列化操作。如果您的 JSON 分析程序设置为对请求的空值执行序列化，则 Amazon SWF 会忽略这些值。

计算 Amazon SWF 的 HMAC-SHA 签名

发给 Amazon SWF 的每个请求都必须进行身份验证。会 Amazon SDKs 自动签署您的请求并管理基于令牌的身份验证。但是，如果要写入自己的 HTTP POST 请求，在对请求进行身份验证时，您需要为 HTTP POST `Header` 内容创建 `x-amzn-authorization` 值。

有关对标头进行格式设置的详细信息，请参阅 HTTP 标头内容。有关 Amazon 版本 3 签名的 适用于 Java 的 Amazon SDK 实现，请参阅 AWSSigner.java 类。

创建请求签名

在创建 HMAC-SHA 请求签名之前，必须获取 Amazon 凭证（访问密钥 ID 和私有密钥）。

> **Important**
>
> 您可以使用 SHA1 或 SHA256 来签署您的请求。但是，请务必在整个签名过程中使用同一方法。您选择的方法必须与 HTTP 标头中的 `Algorithm` 名称的值匹配。

创建请求签名

1. 创建一个标准化的 HTTP 请求标头。HTTP 标头规范形式包括以下内容：

- host
- 以 x-amz- 为开头的任何标头元素

有关所包含的标头的详细信息，请参阅 HTTP 标头内容。

a. 对于每一个标头名值对，将标头名称转换为小写字母 (但不是标头值)。

b. 将标头名称映射到逗号分隔的标头值。

```
x-amz-example: value1
x-amz-example: value2  =>  x-amz-example:value1,value2
```

有关更多信息，请参阅 RFC 2616 的第 4.2 节。

c. 对于每个标头名称-值对，将其转换成 headerName:headerValue 格式的字符串。从 headerName 和 headerValue 的开头和结尾删除所有空白，冒号两边不留空格。

```
x-amz-example1:value1,value2
x-amz-example2:value3
```

d. 在转换好的每一个字符串 (包括最后一个字符串) 后插入一个换行符 (U+000A)。

e. 按照字母顺序用标头名称给转换好的字符串集合分类。

2. 创建包含以下项目的 string-to-sign值：

- 第 1 行：HTTP 方法 (POST)，其后是换行符。
- 第 2 行：请求 URI (/)，其后是换行符。
- 第 3 行：空字符串，其后是换行符。

> **Note**
>
> 通常，这里会显示查询字符串，但 Amazon SWF 不使用查询字符串。

- 第 4–n 行：表示步骤 1 中计算的规范请求标头的字符串，后跟换行符。该换行符在 HTTP 请求的标头与正文之间创建一个空白行。有关更多信息，请参见 RFC 2616。
- 请求正文，不 跟换行符。

3. 计算 string-to-sign值的 SHA256 或 SHA1 摘要。在整个过程中采用同一种 SHA 方法。

4. 使用 API 操作使用上一步生成的值和来自 Amazon 安全令牌服务的临时私有访问密钥的 SHA1 摘要（取决于您使用的方法）计算并对 HMAC-SHA 进行编码。 SHA256 `GetSessionToken`

   > **Note**
   >
   > Amazon SWF 希望在进行 Base64 编码的 HMAC-SHA 值末尾添加一个等号 (=)。如果 Base64 编码程序不包括附加等号，请在值的结尾附加一个等号。

   有关在 Amazon SWF 和其他 Amazon 服务中使用临时安全证书的更多信息，请参阅 IAM 用户指南中的与 IAM 配合使用的Amazon 服务。
5. 将生成的值作为 `Signature` 名称的值放入向 Amazon SWF 发送的 HTTP 请求的 `x-amzn-authorization` 标头中。
6. Amazon SWF 会验证该请求并执行指定操作。

按类别列出 Amazon SWF 操作

本部分列出了 Amazon SWF 应用程序编程接口 (API) 中 Amazon SWF 操作的参考主题。其中按功能类别 列出这些主题。

有关按字母顺序排列的操作列表，请参阅 Amazon Simple Workflow Service API Reference。

主题

- 与活动相关的操作
- 与决策程序相关的操作
- 与工作流执行相关的操作
- 与管理相关的操作
- 可见性操作

与活动相关的操作

活动工作者使用 `PollForActivityTask` 获取新活动任务。工作线程从 Amazon SWF 收到活动任务后即执行该任务，如果成功，则使用 `RespondActivityTaskCompleted` 进行响应，如果失败，则使用 `RespondActivityTaskFailed` 进行响应。

以下是活动工作线程执行的操作。

- PollForActivityTask
- RespondActivityTaskCompleted
- RespondActivityTaskFailed
- RespondActivityTaskCanceled
- RecordActivityTaskHeartbeat

与决策程序相关的操作

决策者使用 PollForDecisionTask 获取决策任务。决策者从 Amazon SWF 收到决策任务后，检查其工作流程执行历史记录并决定接下来要做什么。它调用 RespondDecisionTaskCompleted 以完成该决策任务，并提供零个或多个后续决策。

以下是由决策程序执行的操作。

- PollForDecisionTask
- RespondDecisionTaskCompleted

与工作流执行相关的操作

对工作流可执行以下操作。

- RequestCancelWorkflowExecution
- StartWorkflowExecution
- SignalWorkflowExecution
- TerminateWorkflowExecution

与管理相关的操作

尽管可以从 Amazon SWF 控制台中执行管理任务，但您还可以使用本部分中操作自动执行各种功能或构建您自己的管理工具。

活动管理

- RegisterActivityType
- DeprecateActivityType

- UndeprecateActivityType
- DeleteActivityType

工作流程管理

- RegisterWorkflowType
- DeprecateWorkflowType
- UndeprecateWorkflowType
- DeleteWorkflowType

域管理

通过这些操作，您可以注册和启用 Amazon SWF 域。

- RegisterDomain
- DeprecateDomain
- UndeprecateDomain

有关这些域管理操作的详细信息和示例，请参阅使用 Amazon SWF 注册域。

工作流执行管理

- RequestCancelWorkflowExecution
- TerminateWorkflowExecution

可见性操作

尽管可以从 Amazon SWF 控制台中执行可见性操作，但您还可以使用本部分中的命令构建您自己的控制台或管理工具。

活动可见性

- ListActivityTypes
- DescribeActivityType

工作流程可见性

- ListWorkflowTypes
- DescribeWorkflowType

工作流程执行可见性

- DescribeWorkflowExecution
- ListOpenWorkflowExecutions
- ListClosedWorkflowExecutions
- CountOpenWorkflowExecutions
- CountClosedWorkflowExecutions
- GetWorkflowExecutionHistory

域可见性

- ListDomains
- DescribeDomain

任务列表可见性

- CountPendingActivityTasks
- CountPendingDecisionTasks

使用 Amazon SWF 注册域

您的工作流程和活动类型以及工作流程执行本身都囊括在域 范围之内。域将一组类型、执行和任务列表与同一账户内的其它内容隔开。

您可以使用 Amazon Web Services Management Console 或使用 Amazon SWF API 中的`RegisterDomain`操作来注册域名。以下为使用 API 的示例。

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

```
  "name" : "867530901",
  "description" : "music",
  "workflowExecutionRetentionPeriodInDays" : "60"
}
```

这些参数以 JavaScript 对象表示法 (JSON) 格式指定。此处，保留期设置为 60 天。在保留期内，有关工作流程执行的所有信息均可使用 Amazon Web Services Management Console 或 Amazon SWF API 通过可见性操作获得。

注册域后，您应该注册工作流程类型和该工作流程所用的活动类型。您需要首先注册域，因为注册的域名是注册工作流程和活动类型时所需要信息的一部分。

另请参阅

《Amazon Simple Workflow Service API Reference》中的 RegisterDomain

在 Amazon SWF 中设置超时值

主题

超时值配额

超时值始终以秒为单位声明，可设置为任何秒数，最长可达一年（31536000 秒），这是任何工作流或活动的最大执行时限。特殊值 NONE 用于将超时参数设置为“无超时”或无限，但仍适用一年的最大限制。

工作流程执行和决策任务超时

您可以在注册工作流程类型时设置工作流和决策任务的超时值。例如：

```
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"
}
```

此工作流程类型注册会将 defaultTaskStartToCloseTimeout 设置为 600 秒（10 分钟），并将 defaultExecutionStartToCloseTimeout 设置为 3600 秒（1 小时）。

有关工作流类型注册的更多信息，请参阅 使用 Amazon SWF 注册工作流类型 和 Amazon Simple Workflow Service API Reference 中的 RegisterWorkflowType。

您可以覆盖为 defaultExecutionStartToCloseTimeout 设置的值，方法是指定 executionStartToCloseTimeout i。

活动任务超时

您可以注册活动类型时设置活动任务的超时值。例如：

```
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"
}
```

此活动类型注册将 defaultTaskStartToCloseTimeout 设置为 600 秒（10 分钟），将 defaultTaskHeartbeatTimeout 设置为 120 秒（2 分钟），将 defaultTaskScheduleToStartTimeout 设置为 1800 秒（30 分钟），并将 defaultTaskScheduleToCloseTimeout 设置为 5400 秒（1.5 小时）。

有关活动类型注册的更多信息，请参阅 使用 Amazon SWF 注册活动类型 和 Amazon Simple Workflow Service API Reference 中的 RegisterActivityType。

您可以覆盖为 defaultTaskStartToCloseTimeout 设置的值，方法是在安排活动任务时指定 taskStartToCloseTimeout。

另请参阅

Amazon SWF 超时类型

使用 Amazon SWF 注册工作流类型

本部分讨论的示例使用 Amazon SWF API 注册了一个工作流类型。您在注册过程中指定的名称和版本会形成工作流程类型唯一的标识符。指定的域必须已使用 RegisterDomain API 操作注册。

下列示例中的超时参数为持续时间值，以秒为单位。对于 defaultTaskStartToCloseTimeout 参数，您可以使用持续时间说明符 NONE 指示无超时。但是，不能将 defaultExecutionStartToCloseTimeout 的值指定为 NONE；工作流程执行可运行的最大时长限制是一年。超出此限制都会导致工作流程执行超时。如果您将 defaultExecutionStartToCloseTimeout 的值指定为大于一年，注册将会失败。

```
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"
}
```

另请参阅

《Amazon Simple Workflow Service API Reference》中的 RegisterWorkflowType

使用 Amazon SWF 注册活动类型

下面的示例使用 Amazon SWF API 注册了一个活动类型。您在注册期间指定的名称和版本形成域内活动类型的唯一标识符。指定的域必须使用 RegisterDomain 操作进行了注册。

本示例中的超时参数为持续时间值，以秒为单位。您可以使用持续时间说明符 NONE 指示无超时。

```
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"
}
```

另请参阅

《Amazon Simple Workflow Service API Reference》中的 RegisterActivityType

Amazon Lambda 亚马逊 SWF 中的任务

主题

- 关于 Amazon Lambda
- 使用 Lambda 任务的优势和限制
- 在您的工作流中使用 Lambda 任务

关于 Amazon Lambda

Amazon Lambda 是一项完全托管的计算服务，它运行您的代码以响应由自定义代码或各种 Amazon 服务（例如亚马逊 S3、DynamoDB、Amazon Kinesis、Amazon SNS 和 Amazon Cognito）生成的事件。有关 Lambda 的更多信息，请参阅 Amazon Lambda 开发人员指南。

Amazon Simple Workflow Service 提供了一项 Lambda 任务，以便您可以运行 Lambda 函数来代替传统的 Amazon SWF 活动，或与此类活动一起运行。

> **Important**
>
> 对于亚马逊 SWF 代表您执行的 Lambda 执行（请求），将向您的 Amazon 账户收费。有关 Lambda 定价的详细信息，请参阅https://aws.amazon.com/lambda/定价/。

使用 Lambda 任务的优势和限制

使用 Lambda 任务替代传统 Amazon SWF 活动具有许多优势：

- Lambda 任务不需要像 Amazon SWF 活动类型一样注册或版本化。
- 您可以使用已在工作流中定义的任何现有 Lambda 函数。
- Lambda 函数由 Amazon SWF 直接调用，无需像传统活动那样，需要实现工作线程程序才能执行。
- Lambda 为您提供指标和日志，用于跟踪和分析函数的执行情况。

您还应了解 Lambda 任务有很多限制：

- Lambda 任务只能在支持 Lambda 的 Amazon 地区运行。要详细了解当前支持 Lambda 的区域，请参阅 Amazon Web Services General Reference中的 Lambda Regions and Endpoints。
- 目前，只有基本 SWF HTTP API 和适用于 Java 的 SWF HTTP API 支持 Lambda 任务。 Amazon Flow Framework Ruby 版中目前不支持 Lambda 任务 Amazon Flow Framework 。

在您的工作流中使用 Lambda 任务

要在 Amazon SWF 工作流中使用 Lambda 任务，您需要：

1. 设置 IAM 角色，为 Amazon SWF 提供调用 Lambda 函数的权限。
2. 将 IAM 角色附加到工作流。
3. 在工作流执行期间调用 Lambda 函数。

设置 IAM 角色

在从 Amazon SWF 调用 Lambda 函数之前，您必须先提供一个 IAM 角色，用于从 Amazon SWF 访问 Lambda。您可以：

- 选择一个预定义的角色，即AWSLambda角色，以授予您的工作流程调用与您的账户关联的任何 Lambda 函数的权限。
- 定义您自己的策略和关联角色，以授予工作流程调用特定 Lambda 函数的权限，这些函数由其 Amazon 资源名称 () ARNs 指定。

限制 IAM 角色的访问权限

您可以使用资源信任策略中的 SourceArn 和 SourceAccount 上下文密钥来限制提供给 Amazon SWF 的 IAM 角色的访问权限。这些密钥会限制 IAM 策略的使用，使其只能在属于指定域 ARN 的 Amazon Simple Workflow Service 执行中使用。如果您同时使用两个全局条件上下文密钥，则在同一策略语句中使用 aws:SourceAccount 值和 aws:SourceArn 值中引用的账户时，必须使用相同的账户 ID。

在下面的信任策略示例中，我们使用 SourceArn 上下文密钥将 IAM 服务角色限制为只能在属于账户 123456789012 中的 someDomain 的 Amazon Simple Workflow Service 执行中使用。

```
{
  "Version": "2012-10-17",
  "Statement": [
     {
       "Sid": "",
       "Effect": "Allow",
       "Principal": {
         "Service": "swf.amazonaws.com"
       },
       "Action": "sts:AssumeRole",
       "Condition": {
         "ArnLike": {
           "aws:SourceArn": "arn:aws:swf:*:123456789012:/domain/someDomain"
       }
      }
    }
  ]
}
```

在下面的信任策略示例中，我们使用 SourceAccount 上下文密钥将 IAM 服务角色限制为只能在属于账户 123456789012 的 Amazon Simple Workflow Service 执行中使用。

```
{
  "Version": "2012-10-17",
  "Statement": [
     {
       "Sid": "",
       "Effect": "Allow",
       "Principal": {
         "Service": "swf.amazonaws.com"
       },
```

```
        "Action": "sts:AssumeRole",
        "Condition": {
          "StringLike": {
            "aws:SourceAccount": "123456789012"
        }
       }
     }
   ]
}
```

为 Amazon SWF 提供调用任何 Lambda 角色的访问权限

您可以使用预定义的角色（角色）使您的 Amazon SWF 工作流程能够调用与您的账户关联的任何 Lambda 函数。AWSLambda

使用 AWSLambda角色向 Amazon SWF 授予调用 Lambda 函数的权限

1. 打开 Amazon IAM 控制台。
2. 选择 Roles，然后选择 Create New Role。
3. 提供角色名称 (如 `swf-lambda`)，然后选择 Next Step。
4. 在 Amazon 服务角色下，选择 Amazon SWF，然后选择下一步。
5. 在 “附加策略” 屏幕上，从列表中选择 “AWSLambda角色”。
6. 检查角色之后，选择 Next Step，然后选择 Create Role。

定义 IAM 角色以提供调用特定 Lambda 函数的访问权限

如果要提供从工作流调用特定 Lambda 函数的访问权限，您需要定义自己的 IAM 策略。

创建 IAM 策略以提供对特定 Lambda 函数的访问权限

1. 打开 Amazon IAM 控制台。
2. 选择 Policies，然后选择 Create Policy。
3. 选择 “复制 Amazon 托管策略”，然后从列表中选择 “AWSLambda角色”。随即将生成策略。根据需要编辑策略的名称和描述。
4. 在策略文档的资源字段中，添加您的 Lambda 函数的 ARN。例如：

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

```
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "lambda:InvokeFunction"
      ],
      "Resource": [
        "arn:aws:lambda:us-east-1:111111000000:function:hello_lambda_function"
      ]
    }
  ]
}
```

> **Note**
>
> 有关如何在 IAM 角色中指定资源的完整说明，请参阅《Using IAM》中的 Overview of IAM Policies。

5. 选择 Create Policy 完成策略创建。

您随后可以在创建新的 IAM 角色时选择该策略，并使用该角色提供对 Amazon SWF 工作流的调用访问权限。此过程与使用角色策略创建角色非常相似。相反，请在创建角色时选择自己的策略。AWSLambda

使用 Lambda 策略创建 Amazon SWF 角色

1. 打开 Amazon IAM 控制台。
2. 选择 Roles，然后选择 Create New Role。
3. 提供角色名称 (如 `swf-lambda-function`)，然后选择 Next Step。
4. 在 Amazon 服务角色下，选择 Amazon SWF，然后选择下一步。
5. 在附加策略屏幕上，从列表中选择特定于 Lambda 函数的策略。
6. 检查角色之后，选择 Next Step，然后选择 Create Role。

将 IAM 角色附加到工作流

定义 IAM 角色后，您需要将其附加到工作流，用于调用允许 Amazon SWF 访问的 Lambda 函数。

可以将角色附加到工作流的位置有两个：

- 在工作流类型注册时。随后，在每次执行该工作流类型时，可以将此角色用作默认 Lambda 角色。
- 在启动工作流执行时。此角色将仅在此工作流执行期间（且在整个执行过程中）使用。

为工作流类型提供默认 Lambda 角色

- 调用时 RegisterWorkflowType，将该 defaultLambdaRole 字段设置为您定义的角色的 ARN。

提供工作流执行期间使用的 Lambda 角色

- 调用时 StartWorkflowExecution，将 LambdaRole 字段设置为您定义的角色的 ARN。

> **Note**
>
> 如果调用 RegisterWorkflowType 或的账户 StartWorkflowExecution 无权使用给定角色，则调用将失败，并显示 OperationNotPermittedFault。

从 Amazon SWF 工作流调用 Lambda 函数

您可以使用 ScheduleLambdaFunctionDecisionAttributes 数据类型来标识在工作流程执行期间要调用的 Lambda 函数。

在致电时 RespondDecisionTaskCompleted，请 ScheduleLambdaFunctionDecisionAttributes 向您的决策清单提供一个。例如：

```
{
  "decisions": [{
    "ScheduleLambdaFunctionDecisionAttributes": {
      "id": "lambdaTaskId",
      "name": "myLambdaFunctionName",
      "input": "inputToLambdaFunction",
      "startToCloseTimeout": "30"
    },
  }],
}
```

设置以下参数：

- ID 是 Lambda 任务的标识符。此值必须为 1-256 个字符的字符串，且不能包含 :（冒号）、/（斜杠）、|（竖线）等字符，也不能保护任何控制字符（\u0000 - \u001f 和 \u007f - \u009f）及文字字符串 `arn`。
- name 是 Lambda 函数的名称。您必须为 Amazon SWF 工作流提供一个允许其调用 Lambda 函数的 IAM 角色。提供的名称必须遵循 FunctionName 参数的约束，例如 Lambda Invoke 操作中的限制。
- input 是函数的可选输入数据。如果已设置，则必须遵循 ClientContext 参数的约束条件，例如 Lambda Invoke 操作中的限制。
- startToClose超时，可选的最大时间（以秒为单位），该函数在任务因超时异常而失败之前可以执行该函数。可以使用 NONE 值指定无期限。

有关更多信息，请参阅实现 Amazon Lambda 任务

在 Amazon SWF 开发活动工作线程

活动工作线程会执行一个或多个活动类型。活动工作线程与 Amazon SWF 通信，以接收并执行活动任务。您可以有一队多个活动工作线程执行具有同一种活动类型的活动任务。

当决策程序安排活动任务时，Amazon SWF 会将活动任务提供给活动工作线程。当决策程序安排活动任务时，会提供活动工作线程执行活动任务所需的数据（此数据由您决定）。Amazon SWF 会将这些数据插入到活动任务中，然后再将其发送给活动工作线程。

活动工作线程由您来管理。可以用任何语言编写活动工作线程。工作线程可在任何地点运行，只要它能通过 API 与 Amazon SWF 通信。Amazon SWF会 提供执行活动任务所需的所有信息，因此，所有活动工作线程都可以是无状态的。无状态性使您的工作流可以高度扩展；处理增长的容量需求并简单增加更多活动工作线程。

本章节说明了如何执行活动工作线程。活动工作线程应重复执行下列操作。

1. 轮询 Amazon SWF 以获取活动任务。
2. 开始执行任务。
3. 如果任务长时间运行，请定期向 Amazon SWF 报告检测信号。
4. 报告任务已完成或失败，并将结果返回到 Amazon SWF。

主题

- 报告活动任务检测信号
- 完成活动任务或活动任务失败
- 启动活动工作线程

轮询活动任务

为执行活动任务，每个活动工作线程都必须定期调用 PollForActivityTask 操作以轮询 Amazon SWF。

以下示例所示为，活动工作线程 ChargeCreditCardWorker01 轮询任务列表上的任务 ChargeCreditCard-v0.1。如果没有可用的活动任务，在 60 秒后，Amazon SWF 会发回一个空响应。空响应是一个 Task 结构，在该结构中，taskToken 值为空字符串。

```
https://swf.us-east-1.amazonaws.com
PollForActivityTask
{
  "domain" : "867530901",
  "taskList" : { "name": "ChargeCreditCard-v0.1" },
  "identity" : "ChargeCreditCardWorker01"
}
```

如果活动任务变为可用，则 Amazon SWF 会将其返回给活动工作线程。任务中包含决策程序在其排定活动时指定的数据。

活动工作线程接收活动任务后就已作好执行工作的准备。下一节提供有关执行活动任务的信息。

执行活动任务

接收活动任务后，活动工作线程就做好了执行任何的准备。

要执行活动任务

1. 给您的活动工作线程编程以说明任务输入字段中的内容。该字段中包含决策程序在排定任务时指定的数据。
2. 给活动工作线程编程以开始处理数据并执行您的逻辑。

下一部分说明了如何对活动工作线程进行编程，以将长时间运行的活动的状态更新提供给 Amazon SWF。

报告活动任务检测信号

如果在活动类型中注册了检测信号超时，则活动工作线程必须在超出检测信号超时之前记录检测信号。如果活动任务没有在超时时间内提供检测信号，任务就会超时，Amazon SWF 将关闭该任务并安排新的决策任务，以将超时情况通知决策程序。然后，决策程序可重新排定活动任务或采取另一个操作。

如果活动工作线程在超时后尝试联系 Amazon SWF（例如调用 RespondActivityTaskCompleted），Amazon SWF 将返回 UnknownResource 故障。

本章描述了提供活动检测信号的方法。

要记录活动任务检测信号，给您的活动工作线程编程以调用 RecordActivityTaskHeartbeat 操作。此操作还会提供字符串字段，您可以在该字段中存储自由格式的数据，从而量化以任何方式作用于您的应用程序的过程。

在此示例中，活动任务工作线程向 Amazon SWF 检测信号，并使用详细信息字段报告活动任务已完成 40%。要报告检测信号，活动工作线程必须指定活动任务的任务令牌。

```
https://swf.us-east-1.amazonaws.com
RecordActivityTaskHeartbeat
{
  "taskToken" : "12342e17-80f6-FAKE-TASK-TOKEN32f0223",
  "details" : "40"
}
```

此操作本身不会在工作流执行历史记录中创建时间；但是，如果任务超时，工作流执行历史记录中将包含 ActivityTaskTimedOut 事件，该事件中包含活动工作线程生成的最后一个检测信号的信息。

完成活动任务或活动任务失败

执行完任务后，活动工作线程应报告活动任务完成或失败。

完成活动任务

若要完成活动任务，请给活动工作线程编程，以使其在成功完成活动任务时调用 RespondActivityTaskCompleted 操作，从而指定任务令牌。

在此示例中，活动工作线程指示任务已成功完成。

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

```
  "taskToken": "12342e17-80f6-FAKE-TASK-TOKEN32f0223",
  "results": "40"
}
```

活动完成时，Amazon SWF 会针对与该活动关联的工作流执行安排新决策任务。

给活动工作线程编程，以在其即将完成任务后轮询另一个活动任务。此操作会创建一个循环，在该循环中，活动工作线程会继续轮询并完成任务。

如果活动在该StartToCloseTimeout时间段内没有响应，或者ScheduleToCloseTimeout已超过该时间段，则 Amazon SWF 会超时活动任务并安排决策任务。这会使决策程序采取适当操作，如重新排定任务。

例如，如果 Amazon EC2 实例正在执行活动任务，而该实例在任务完成之前失败，则决策者会在工作流程执行历史中收到超时事件。如果活动任务使用心跳，则在 Ama EC2 zon 实例失败后任务未能传送下一个心跳时，决策者会收到该事件。否则，决策程序会在活动任务触及整个超时值中的一个前完成失败时最终接收事件。然后，由决策程序决定重新分配任务或采取某些其它操作。

活动任务失败

如果活动工作线程出于某种原因无法执行活动任务，但仍能与 Amazon SWF 通信，您可以对其进行编程，使该任务失败。

若要对活动工作线程编程以使某个活动任务失败，请对活动工作线程编程以调用指定任务令牌的 `RespondActivityTaskFailed` 操作。

```
https://swf.us-east-1.amazonaws.com
RespondActivityTaskFailed
{
  "taskToken" : "12342e17-80f6-FAKE-TASK-TOKEN32f0223",
  "reason" : "CC-Invalid",
  "details" : "Credit Card Number Checksum Failed"
}
```

作为开发人员，您要规定存储在原因和详情字段中的值。这些值是自由格式的字符串；您可以使用适用于您应用程序的任何错误代码约定。Amazon SWF 不处理这些值。但是，Amazon SWF 可在控制台中显示这些值。

当活动任务失败时，Amazon SWF 会为与该活动任务相关联的工作流执行安排一个决策任务，以通知决策程序任务失败。给您的决策程序编程以处理失败的活动，如通过重新排定活动或使工作流执行失败，这取决于失败的性质。

启动活动工作线程

要启动活动工作线程，将您的逻辑打包成可执行，这样您就可以将其用于您的活动工作线程平台上。例如，您可以将您的活动代码打包成 Java 可执行，这样您就可以将其用于 Linux 和 Windows 服务器上。

启动后，您的工作线程即会开始轮询任务。在决策程序排定活动任务之前，虽然这些轮询超时未轮询到任务，但您的工作线程只会继续轮询。

由于轮询是出站请求，活动工作线程可以在任何有权访问 Amazon SWF 端点的网站上运行。

您想要启动多少活动工作线程就可以启动多少。在决策程序安排活动任务时，Amazon SWF 会将活动任务自动分配给轮询活动工作线程。

在 Amazon SWF 中开发决策者

决策程序指的是工作流程类型协作逻辑的执行，在您执行工作流程期间运行。您可以针对一种工作流程类型运行多个决策程序。

工作流执行的执行状态存储在其工作流历史记录中，因此决策程序可以是无状态的。Amazon SWF 会维护工作流执行历史记录，并将其提供给每项决策任务的决策程序。这会使您视需要动态增加和删除决策程序，从而使您的工作流程处理过程具有高度可扩展性。随着系统负载的增长，您只需增加更多决策程序来处理增加的容量。但是，请注意，任何时候，对于给定的工作流程执行，都只能由一个决策任务处于开启状态。

每当工作流执行的状态发生更改时，Amazon SWF 都会安排一个决策任务。每当决策程序收到决策任务时，它都会执行以下操作：

- 解析与决策任务一并提供的工作流程执行历史
- 根据工作流程执行历史采用协作逻辑，并作出下一步操作的决策。每一个决策都由一个 Decision 结构表示
- 完成该决策任务，并向 Amazon SWF 提供一个决策列表。

本章表述了开发决策程序的方法，其中包括：

- 给您的决策程序编程以轮询决策任务
- 给您的决策程序编程以解析工作流程执行历史并作出决策
- 给您的决策程序编程以响应决策任务。

本章节中的示例显示的是，您针对电子商务示例工作流程给决策程序编程的方法。

您可以使用所需的任何语言来实现决策程序并在任何位置运行，只要该决策程序可通过其服务 API 与 Amazon SWF 通信。

主题

定义协作逻辑

开发决策程序的第一步是定义协作逻辑。在电子商务示例中，在前一个活动完成后排定每个活动的协作逻辑可能与以下内容相似：

```
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
```

决策程序将协作逻辑应用于工作流程执行历史，并在使用 `RespondDecisionTaskCompleted` 操作完成决策任务时创建决策列表。

轮询决策任务

每个决策程序都要轮询决策任务。决策任务中包含决策程序用于生成决策（如排定活动任务）的信息。要轮询决策任务，决策程序应使用 PollForDecisionTask 操作。

在此示例中，决策程序轮询决策任务，以指定 customerOrderWorkflow-0.1 任务列表。

```
https://swf.us-east-1.amazonaws.com
PollForDecisionTask
{
  "domain": "867530901",
  "taskList": {"name": "customerOrderWorkflow-v0.1"},
  "identity": "Decider01",
  "maximumPageSize": 50,
  "reverseOrder": true
}
```

如果指定的任务列表中有可用的决策任务，Amazon SWF 会立即返回该任务。如果没有决策任务可用，Amazon SWF 会保持连接打开长达 60 秒，并在任务可用时立即将其返回。如果没有任务可用，Amazon SWF 会返回一个空响应。空响应为 Task 结构，其中的 taskToken 值为空字符串。确保给您的决策程序编程以在它收到空响应时轮询另一个任务。

如果决策任务可用，Amazon SWF 会返回一个包含决策任务和工作流执行历史分页视图的响应。

在此示例中，最近事件类型指的是启动的工作流程执行，输入元素中包含执行第一个任务所需的信息。

```
{
  "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"
        }
      }
    }
  ],
  ...
}
```

收到工作流程执行历史后，决策程序会对历史进行解析并根据其协作逻辑做出决策。

由于一个工作流程执行的工作流程历史事件数量可能会很大，返回的结果可能会被拆分跨页显示。要检索后续页面，请PollForDecisionTask使用初始调用nextPageToken返回的进行其他调用。请注意，你不能GetWorkflowExecutionHistory用这个来电nextPageToken。请重新调用 PollForDecisionTask 替代之。

应用协作逻辑

在决策程序收到决策任务后，给任务编程以解析工作流程执行历史，从而确定目前为止发生的情况。决策程序应该基于这个情况生成决策列表。

在电子商务示例中，我们只关心工作流程历史中最近的事件，所有我们要定义以下逻辑。

```
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
```

如果 lastEvent 为 `CompleteVerifyOrderActivity`，您应增加 `ScheduleChargeCreditCardActivity` 活动到决策列表。

在决策程序确定要做出的决策后，可以用适当的决策来响应 Amazon SWF。

响应决策

在解读工作流历史记录并生成决策列表后，决策程序就可以将这些决策返回 Amazon SWF。

给您的决策程序编程，以从工作流程执行历史中提取出它所需要的数据，然后创建决策以指定工作流程的下一步适当操作。决策程序使用 `RespondDecisionTaskCompleted` 操作将这些决策发送回 Amazon SWF。有关可用决策类型的列表，请参阅《Amazon Simple Workflow Service API Reference》。

在电子商务示例中，当决策程序回应其生成的决策集时，还会包括工作流程执行历史中的信用卡输入。然后，活动工作程序会获得其需要的信息来执行活动任务。

当工作流程执行中的所有活动均完成时，决策程序会关闭工作流程执行。

```
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"
      }
    }
  ]
}
```

关闭工作流程执行

当决策程序确定业务过程完成也就是没有更多要执行的活动时，决策程序会生成关闭工作流程执行的决策。

若要关闭工作流程执行，请给您的决策程序编程以解析工作流程历史中的事件，从而确定目前为止执行中发生的情况并查看工作流程执行是否应关闭。

如果工作流程已成功完成，则用 `CompleteWorkflowExecution` 决策来调用 `RespondDecisionTaskCompleted` 以关闭工作流程执行。或者，您可以使用 `FailWorkflowExecution` 决策放弃错误的执行。

在电子商务实例中，决策程序审查历史记录并根据协作逻辑增加关闭工作流程执行的决策到其决策列表中，并且用关闭工作流程决策启动 `RespondDecisionTaskCompleted` 操作。

> **Note**
>
> 有些情况下关闭工作流程执行的操作会失败。例如，如果在决策程序正关闭工作流程执行时收到信号，关闭决策会失败。要处理这种可能性，请确保决策程序继续轮询决策任务。此外，请确保接收下一个决策任务的决策程序对阻止执行结束的事件（在本例中为信号）做出响应。

您还可以支持取消工作流程执行。这尤其对长时间运行的工作流程有用。若要支持取消，决策程序应处理历史中的 `WorkflowExecutionCancelRequested` 事件。此事件表示已请求取消执行。您的决策程序应执行适当的清除操作，如取消持续进行的活动任务以及用 `CancelWorkflowExecution` 决策来调用 `RespondDecisionTaskCompleted` 操作以关闭工作流程。

以下示例显示的是，调用 `RespondDecisionTaskCompleted` 以指定当前工作流程执行已取消。

```
https://swf.us-east-1.amazonaws.com
RespondDecisionTaskCompleted
{
  "taskToken" : "12342e17-80f6-FAKE-TASK-TOKEN32f0223",
  "decisions" : [
    {
      "decisionType":"CancelWorkflowExecution",
      "CancelWorkflowExecutionAttributes":{
        "Details": "Customer canceled order"
      }
    }
  ]
}
```

Amazon SWF 会进行检查，确保关闭和取消工作流执行的决策是决策程序发送的最后一个决策。也就是说，拥有在关闭工作流程之后还有决策的这样一组决策无效。

启动决策程序

完成决策程序开发之后，您就做好了启动一个或多个决策程序的准备。

要启动决策程序，请将您的协作逻辑打包成可执行，以使您能够将其用于您的决策程序平台上。例如，您可以将您的决策程序代码打包为 Java 可执行，以使您能够将其运行于 Linux 和 Windows 计算机上。

启动后，决策程序应开始轮询 Amazon SWF 以获取任务。在您启动工作流执行且 Amazon SWF 安排决策任务之前，这些轮询将会超时并获得空响应。空响应为 `Task` 结构，其中的 `taskToken` 值为空字符串。您的决策程序应只需继续轮询。

Amazon SWF 可确保在任何时候，一个工作流的执行只能有一个决策任务处于活动状态。这会防止决策冲突之类的问题。此外，Amazon SWF 还可确保将单个决策任务分配给单个决策程序，无论正在运行的决策程序有多少。

如果在决定程序处理另一个决策任务时发生了生成决策任务的情况，Amazon SWF 会将新任务排队，直到当前任务完成。当前任务完成后，Amazon SWF 会使新决策任务可用。此外，决策任务是分批进行的，也就是说，如果在决策程序处理决策任务的同时有多个活动完成，Amazon SWF 将只创建一个新决策任务，以应对多个任务完成。但是，每个任务完成都会收到工作流程执行历史中的一个单独事件。

由于轮询是出站请求，决策程序可以在任何可以访问 Amazon SWF 端点的网络上运行。

为了推进工作流程执行，必须运行一个或多个决策程序。您可以根据需要启动任意数量的决策程序。Amazon SWF 支持多个决策程序对同一任务列表进行轮询。

在 Amazon SWF 中启动工作流程

您可以利用 StartWorkflowExecution 操作从任何应用程序中启动已注册工作流程类型的工作流程执行。启动执行时，您将被称为 workflowId 标识符与执行相关联。workflowId 可以是适用于您的应用程序的任何字符串，如订单中处理应用程序的订单号码。同一个 workflowId 不能用于同一个域中多个已开启工作流程执行。例如，如果您用workflowId Customer Order 01 启动两个工作流程执行，则第二个工作流程执行不会启动且请求会失败。不过，您可以重复使用已关闭执行的 workflowId。Amazon SWF 还会将系统生成的唯一标识符（称为 runId）与每个工作流执行关联起来。

注册完工作流程和活动类型之后，通过调用 StartWorkflowExecution 操作启动工作流程。input 参数的值可以是启动工作流程所用应用程序指定的任何字符串。executionStartToCloseTimeout 是工作流程执行从启动到关闭所消耗的时长，以秒为单位。超出这个范围会导致工作流程执行超时。与 Amazon SWF 中的其他一些超时参数不同，该超时值不能指定为 NONE；工作流执行可运行的最大时长限制是一年。同样，taskStartToCloseTimeout是与该工作流程执行相关的决策任务在超时之前可能花费的时间（以秒为单位）。

```
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"
}
```

如果 StartWorkflowExecution 操作成功，Amazon SWF 将返回工作流执行的 runId。工作流程执行的 runId 在特定区域内是唯一的。请保存 runId 以备日后需要在 Amazon SWF 调用中指定此工作流执行时使用。例如，如果您以后需要发送信号至工作流程执行中，您将使用 runId。

```
{"runId": "9ba33198-4b18-4792-9c15-7181fb3a8852"}
```

在 Amazon SWF 中设置任务优先级

默认情况下，任务列表上的任务将基于其到达时间进行交付：首先安排的任务通常会尽可能首先运行。通过设置可选的任务优先级，您可以设定特定任务的优先级：Amazon SWF 会尝试先交付任务列表上优先级较高的任务，然后再交付优先级较低的任务。

> **Note**
>
> 先安排的任务通常先运行，但是也不一定。

您可以同时为工作流和活动设置任务优先级。工作流的任务优先级既不影响其安排的任何活动的任务优先级，也不影响其开始的任何子工作流。活动或工作流的默认优先级是由您或 Amazon SWF 在注册过程中设置的，除非在安排活动或启动工作流执行时覆盖了注册的任务优先级，否则应始终使用默认优先级。

任务优先级值的范围可为“-2147483648”到“2147483647”，数字越大优先级越高。如果您未为活动或工作流设置任务优先级，则将为其分配零（“0”）优先级。

主题

- 设置工作流的任务优先级
- 设置活动的任务优先级
- 返回任务优先级信息的操作

设置工作流的任务优先级

在您注册或开始工作流时，可以设置其任务优先级。注册工作流类型时设置的任务优先级将用作执行该类型的任何工作流的默认优先级，除非该优先级在开始执行工作流时被覆盖。

要注册具有默认任务优先级的工作流类型，请在使用RegisterWorkflowType操作时设置以下defaultTaskPriority选项：

```
{
  "domain": "867530901",
  "name": "expeditedOrderWorkflow",
```

```
  "version": "1.0",
  "description": "Expedited customer orders workflow",
  "defaultTaskStartToCloseTimeout": "600",
  "defaultExecutionStartToCloseTimeout": "3600",
  "defaultTaskList": {"name": "mainTaskList"},
  "defaultTaskPriority": "10",
  "defaultChildPolicy": "TERMINATE"
}
```

使用以下命令开始执行工作流程时，您可以覆盖工作流程类型的注册任务优先级 StartWorkflowExecution：

```
{
  "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"},
}
```

您还可以在启动子工作流程或以新方式继续工作流程时（例如使用回应决策时）覆盖已注册的任务优先级RespondDecisionTaskCompleted。

要设置子工作流的任务优先级，请在 `startChildWorkflowExecutionDecisionAttributes` 中提供值：

```
{
  "taskToken": "AAAAKgAAAAEAAAAAAAAAA...",
  "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"
        }
      }
    }
  ]
}
```

在将一个工作流作为新的工作流继续执行时，请在 `continueAsNewWorkflowExecutionDecisionAttributes` 中设置任务优先级：

```
{
  "taskToken": "AAAAKgAAAAEAAAAAAAAAA...",
  "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"
      }
    }
  ]
}
```

设置活动的任务优先级

您可以在注册或安排活动时设置该活动的任务优先级。注册活动类型时设置的任务优先级将用作运行活动时的默认优先级，除非该优先级在安排活动时被覆盖。

要在注册活动类型时设置任务优先级，请在使用RegisterActivityType操作时设置以下defaultTaskPriority选项：

```
{
```

```
  "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"
}
```

要计划具有任务优先级的任务，请在安排带有以下操作的活动时使用 task Priority 选项：RespondDecisionTaskCompleted

```
{
  "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"
      }
    }
  ]
}
```

返回任务优先级信息的操作

您可以从以下 Amazon SWF 操作中获取有关设置任务优先级（或设置默认任务优先级）的信息：

- DescribeActivityType返回响应defaultTaskPriority`configuration`部分中活动类型的。
- DescribeWorkflowExecution返回响应`executionConfiguration`部分中工作流程执行的任务优先级。

- DescribeWorkflowType返回响应defaultTaskPriorityconfiguration部分中工作流程类型的。
- GetWorkflowExecutionHistory并在响应的activityTaskScheduledEventAttributes、decisionTaskScheduledEventAttributeswor和workflowExecutionStartedEventAttributes部分中PollForDecisionTask提供任务优先级信息。

处理 Amazon SWF 中的错误

工作流执行过程中会发生很多不同类型的错误。

主题

- 验证错误
- 制定操作或决策中发生的错误
- 超时
- 用户代码导致的错误
- 与关闭工作流执行相关的错误。

验证错误

当发送至 Amazon SWF 的请求因格式不正确或包含无效数据而失败时，就会发生验证错误。在此背景下，请求可以是 DescribeDomain 之类的操作或 StartTimer 之类的决策。如果该请求是一个操作，则 Amazon SWF 会在响应中返回错误代码。检查此错误代码，因为它可能会提供有关请求的哪方面导致失败的信息。例如，请求传输的一个或多个参数可能无效。有关常见错误代码的列表，请参阅《Amazon Simple Workflow Service API Reference》中的操作主题。

如果失败的请求是一个决策，则将会在工作流执行历史中列出适当的事件。例如，如果 StartTimer 决策失败，您会在历史中看到 StartTimerFailed 事件。决策程序应在其接收历史时检查上述事件以响应 PollForDecisionTask 或 GetWorkflowExecutionHistory。下面是在决策格式不正确或包含无效数据时可能发生的决策失败事件的列表。

制定操作或决策中发生的错误

即使请求的格式正确，也可能会在 Amazon SWF 尝试执行请求时发生错误。在这种情况下，历史中的以下事件中会有一个指示错误已发生。查看事件的 reason 字段以确定失败原因。

- CancelTimerFailed

- RequestCancelActivityTaskFailed
- RequestCancelExternalWorkflowExecutionFailed
- ScheduleActivityTaskFailed
- SignalExternalWorkflowExecutionFailed
- StartChildWorkflowExecutionFailed
- StartTimerFailed

超时

决策程序、活动工作线程和工作流执行都会在超时时间限制内运行。在这种类型的错误中，任务或子工作流超时。描述超时的事件会出现在历史记录中。决策程序应通过重新排定任务或重新启动子工作流等方法处理此事件。有关超时的更多信息，请参阅 Amazon SWF 超时类型

- ActivityTaskTimedOut
- ChildWorkflowExecutionTimedOut
- DecisionTaskTimedOut
- WorkflowExecutionTimedOut

用户代码导致的错误

这种错误条件类型的示例为活动任务失败和子工作流失败。与超时错误一样，Amazon SWF 会将相应事件添加到工作流执行历史中。决策程序应合理地重新排定任务或重新启动子工作流以处理此事件。

- ActivityTaskFailed
- ChildWorkflowExecutionFailed

与关闭工作流执行相关的错误。

如果决策程序尝试关闭具有待办决策任务的工作流，它们还可以查看以下事件。

- FailWorkflowExecutionFailed
- CompleteWorkFlowExecutionFailed
- ContinueAsNewWorkflowExecutionFailed
- CancelWorkflowExecutionFailed

有关上述任何事件的更多信息，请参阅《Amazon SWF API Reference》中的 History Event。

Amazon SWF 限额

Amazon SWF 为某些工作流参数的大小设置了限额，例如每个账户的域数量和工作流执行历史记录的大小。这些限额旨在防止错误的工作流消耗系统的所有资源，但并不是硬性限制。如果发现自己的应用程序经常超出这些限额，您可以申请提高服务限额。

内容

Amazon SWF 的一般账户限额

- 最大注册域数 – 100

  此限额包括已注册和已弃用的域。
- 最大工作流和活动类型数 – 每个域 10,000 个

  此限额包括已注册和已弃用的类型。
- API 调用限额 – 除了少见的峰值外，如果应用程序在很短时间内调用了大量 API，也可能会被节流。
- 最大请求大小 – 每个请求 1MB

  这是每个 Amazon SWF API 请求的总数据大小，包括请求标头以及所有其他关联的请求数据。
- C@@ ount 的响应被截断 APIs — 表示已达到内部配额且响应未达到全部计数。

  在返回完整响应之前，某些查询将达到上述 1 MB 的内部限额。以下 API 可以返回截断的响应，而不是完整的计数。
  - CountClosedWorkflowExecutions
  - CountOpenWorkflowExecutions

- CountPendingActivityTasks
- CountPendingDecisionTasks

对于上述每一个 API，如果 `truncated` 响应设置为 true，计数将小于完整数量。此内部限额无法提高。

- 最大标签数 – 每个资源 50 个标签。

  尝试添加超过 50 个标签会引发 400 错误 `TooManyTagsFault`。

工作流执行限额

- 最大已开启工作流执行数 – 每个域 100000 个

  此计数包括子工作流执行。
- 最长工作流执行时间 – 1 年 这是硬性限额，无法更改。
- 最大工作流执行历史记录数 – 25000 个事件 这是硬性限额，无法更改。

  最佳实践是使构造的每个工作流的历史记录不会增长到超过 10000 个事件。由于决策者必须获取工作流程历史记录，因此较小的历史记录可以让决策者更快地完成任务。如果使用 Flow Fram ework，则可以使用 ContinueAsNew 以全新的历史记录继续工作流程。
- 开启的子工作流执行的最大数量 – 每个工作流执行 1000 个

如果您的使用案例需要超出这些限额，您可以使用 Amazon SWF 提供的功能继续执行，并使用子工作流执行来构建应用程序。如果您发现仍需要提高限额，请参阅 请求提高限额。

任务执行限额

- 每个任务列表最大轮询数 – 每个任务列表 1000 个

  同时最多可有 1000 个轮询程序对一个特定任务列表进行轮询。如果超过 1000，会引发 `LimitExceededException`。

> **Note**
>
> 虽然最大限额是 1000，但您可能会在达到此限额之前就遇到 `LimitExceededException` 错误。此错误并不意味着任务被延迟。相反，这意味着您在任务列表中拥有最大数量的

闲置轮询器。Amazon SWF 设置此限额是为了节省客户端和服务器端的资源。设置限制可以防止过多轮询器进行不必要的等待。您可以使用多个任务列表分配轮询，以减少 `LimitExceededException` 错误。

- 每秒安排的最大任务数 – 每个任务列表 2000 个

  在特定任务列表中，您每秒最多可安排 2000 个任务。如果超过 2000，您的 `ScheduleActivityTask` 决策将因 `ACTIVITY_CREATION_RATE_EXCEEDED` 错误而失败。

  > **Note**
  >
  > 虽然最大限额是 2000，但您可能会在达到此限额之前就遇到 `ACTIVITY_CREATION_RATE_EXCEEDED` 错误。要减少这些错误，您可以使用多个任务列表来分配负载。

- 最长任务执行时间 – 1 年（受工作流最长执行时间限制）

  可以配置活动超时，以便在活动任务执行的特定阶段耗时太长时引发超时事件。

- SWF 在队列中保留任务的最长时间 – 1 年（受工作流执行时间限额限制）

  可以配置活动注册过程中的默认活动超时，以便在活动任务执行的特定阶段耗时太长时引发超时事件。还可以在决策程序代码中排定活动任务时，覆盖默认活动超时。

- 最大已开启活动任务数 – 每个工作流执行 1000 个。

  此限额包括已安排的活动任务和工作线程正在处理的活动任务。

- 最大已开启定时器数 – 每个工作流执行 1000 个

- 最大输入/结果数据大小 – 32768 个字符

  此限额会影响活动或工作流执行结果数据、安排活动任务或工作流执行时的输入数据，以及使用工作流执行信号发送的输入。

- 决策任务响应中的最大决策数 – 变化

  由于最大 API 请求大小有 1MB 的限额，因此单次调用 `RespondDecisionTaskCompleted` 所返回的决策数量将根据每个决策所使用的数据大小（包括提供给安排活动任务或工作流执行的任何输入数据的大小）而受到限制。

Amazon SWF 节流限额

除了上述服务限额外，某些 Amazon SWF API 调用和决策事件海会使用令牌存储桶方案进行节流，以保持服务带宽。如果请求速率持续超出此处列出的速率，您可以请求提高节流限额。

所有区域的节流和决策限额都是相同的。

所有区域的节流限额

以下限额适用于个人账户。此外，您也不能请求提高以下限额。有关此操作的信息，请参阅 请求提高限额。

API 名称	桶大小	每秒的重填速率
CountClosedWorkflowExecutions	2000	6
CountOpenWorkflowExecutions	2000	6
CountPendingActivityTasks	200	6
CountPendingDecisionTasks	200	6
DeleteActivityType	200	6
DeleteWorkflowType	200	6
DeprecateActivityType	200	6
DeprecateDomain	100	6
DeprecateWorkflowType	200	6
DescribeActivityType	2000	6
DescribeDomain	200	6
DescribeWorkflowExecution	2000	6
DescribeWorkflowType	2000	6
GetWorkflowExecutionHistory	2000	60

API 名称	桶大小	每秒的重填速率
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
RespondActivityTaskCompleted	2000	200
RespondActivityTaskFailed	2000	200
RespondDecisionTaskCompleted	2000	200
SignalWorkflowExecution	2000	30
StartWorkflowExecution	2000	200
TagResource	50	30

API 名称	桶大小	每秒的重填速率
TerminateWorkflowExecution	2000	60
UndeprecateActivityType	200	6
UndeprecateDomain	100	6
UndeprecateWorkflowType	200	6
UntagResource	50	30

所有区域的决策限额

以下限额适用于个人账户。此外，您也不能请求提高以下限额。有关此操作的信息，请参阅 请求提高限额。

API 名称	桶大小	每秒的重填速率
RequestCancelExternalWorkflowExecution	1200	120
ScheduleActivityTask	1000	200
SignalExternalWorkflowExecution	1200	120
StartChildWorkflowExecution	500	12
StartTimer	2000	200

工作流级别限额

以下限额适用于工作流级别，不可提高。

API 名称	桶大小	每秒的重填速率
GetWorkflowExecutionHistory	400	200

API 名称	桶大小	每秒的重填速率
SignalWorkflowExecution	1000	1000
RecordActivityTaskHeartbeat	1000	1000
RequestCancelWorkflowExecution	200	200

请求提高限额

有关更多信息，请参阅 Amazon Web Services 一般参考 中的 Amazon 服务限额。

亚马逊 SWF 的其他资源和参考信息

本章提供了使用 Amazon SWF 开发工作流时需要的其他资源和参考信息。

主题

Amazon SWF 超时类型

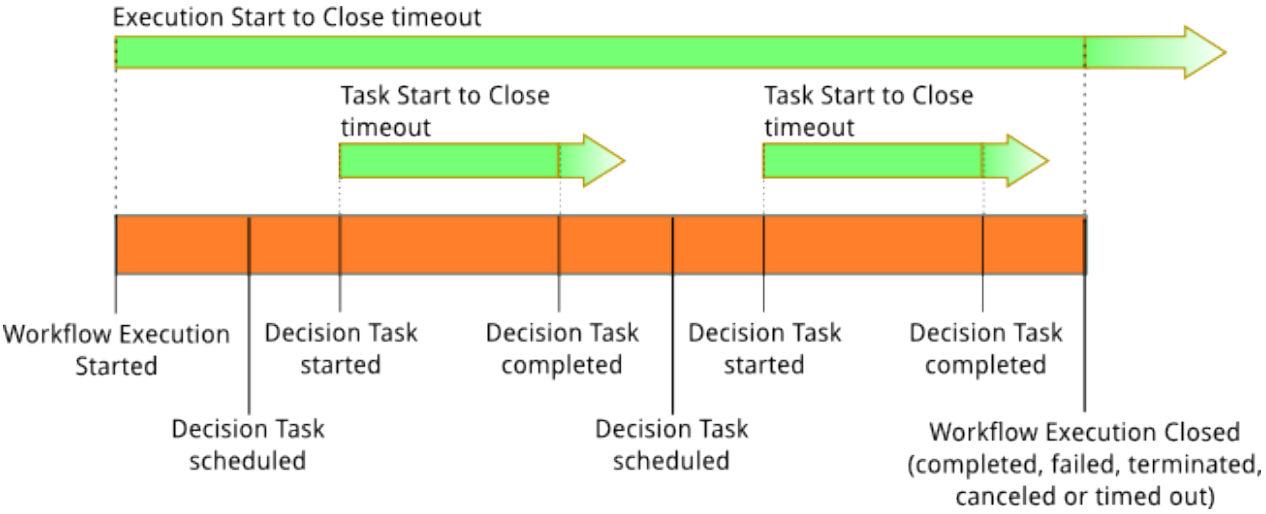
为确保工作流程执行正常运行，您可以使用 Amazon SWF 设置不同类型的超时。一些超时指定工作流程的总运行时长。其它超时指定活动任务在被分配给工作程序之前所花费的时间以及从排定到完成所花费的时间。Amazon SWF API 中的所有超时都以秒为单位。Amazon SWF 还支持将字符串 NONE 作为超时值，表示没有超时。

对于与决策任务和活动任务相关的超时，Amazon SWF 会在工作流执行历史中添加一个事件。事件的属性提供了所发生超时类型的信息以及以及受影响的决策任务或活动任务。Amazon SWF 还会计划决策任务。当决策者收到新的决策任务时，它将在历史记录中看到超时事件，并通过调用操作来采取适当的RespondDecisionTaskCompleted行动。

任务被视为从排定时开始到其关闭为止都处于开启状态。因此，当工作程序处理任务时，任务被报告为开启状态。当工作程序将任务状态报告为已完成、已取消或失败时，任务关闭。Amazon SWF 也可能会因超时而关闭任务。

工作流程和决策任务中的超时

下图显示工作流程和决策超时如何与工作流程的生命周期相关：

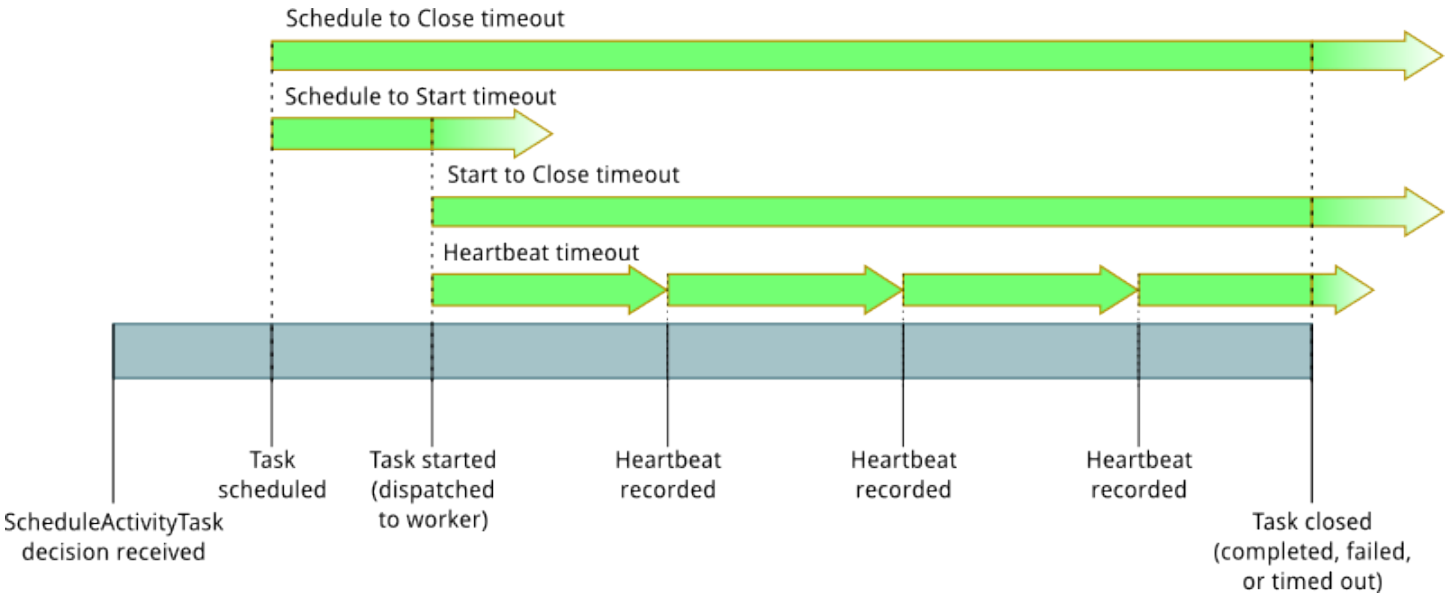


与工作流程和决策任务相关的超时类型有两个：

- 工作流启动到关闭 (**`timeoutType: START_TO_CLOSE`**) - 该超时指定了完成工作流执行所需的最长时间。工作流程注册期间，这一超时被设置为默认值，但当工作流程启动时，可用其它值覆盖该默认值。如果超过此超时时间，Amazon SWF 将关闭工作流程执行并在工作流程执行历史中WorkflowExecutionTimedOut添加类型的事件。除了 `timeoutType` 之外，事件属性还会指定对此工作流程执行有效的 `childPolicy`。子策略指定上级工作流程执行超时或终止时子工作流程执行的处理方法。例如，如果 `childPolicy` 被设置为 TERMINATE，则子工作流程执行将被终止。一旦工作流程执行超时，则不能对其执行可视性调用之外的其他任何操作。
- 决策任务启动到关闭 (**`timeoutType: START_TO_CLOSE`**) – 该超时指定了相应决策程序完成决策任务所需的最长时间。该超时在工作流程类型注册期间设置。如果超过此超时时间，则任务将在工作流程执行历史中标记为超时，Amazon SWF 会在工作流程历史记录中添加DecisionTaskTimedOut类型的事件。事件属性将包括与该决策任务 IDs 的计划时间 (`scheduledEventId`) 和启动时间 () 相对应的事件。`startedEventId`除了添加事件外，Amazon SWF 还会计划新决策任务以警告决策程序此决策任务已超时。此超时发生后，使用 `RespondDecisionTaskCompleted` 完成超时决策任务的尝试将失败。

活动任务中的超时

下图显示超时如何与活动任务的生命周期相关：



与活动任务相关的超时类型有四个：

- 活动任务启动到关闭 (**timeoutType: START_TO_CLOSE**) – 该超时指定了活动工作线在接收到任务后处理任务所需的最长时间。尝试使用RespondActivityTaskCanceled、RespondActivityTaskCompleted和关闭超时活动任务RespondActivityTaskFailed将失败。
- 活动任务检测信号 (**timeoutType: HEARTBEAT**) – 该超时指定了任务在通过 操作提供其进程前可以运行的最长时间。
- 活动任务计划到开始 (**timeoutType: SCHEDULE_TO_START**) – 该超时指定了在没有工作线执行活动任务时 Amazon SWF 在活动任务超时前等待的时间。一旦超时，过期的任务就不能分配给另一工作程序。
- 活动任务安排到关闭 (**timeoutType: SCHEDULE_TO_CLOSE**) – 该超时指定了任务从计划到完成所需的时间。作为最佳实践，此值不应大于任务 schedule-to-start超时和任务超时之和。 start-to-close

> **Note**
>
> 每一个超时类型都有默认值，一般设置为 NONE (无限)。但是任何活动执行的最长时间均被限制为一年。

您在活动类型注册期间设置这些活动的默认值，但当您排定活动任务时您可以用新值覆盖默认值。当其中一个超时发生时，Amazon SWF 将在工作流程历史记录中添加一个ActivityTaskTimedOut类型的事件。此事件的 `timeoutType` 值属性将指定发生了何种超时。对于其中每一个超时，`timeoutType`

的值都显示在括号中。事件属性还将包括与活动任务的 IDs 计划时间 (`scheduledEventId`) 和启动时间 () 相对应的事件。`startedEventId`除了添加事件外，Amazon SWF 还会计划新决策任务以警告决策程序已发生超时。

Amazon Simple Workflow Service 端点

Amazon Web Services 一般参考 中提供了当前 Amazon SWF 区域和端点的列表，以及其他服务的端点。

Amazon SWF 域及所有相关工作流和活动必须在同一区域中才能互相通信。此外，一个区域中的任何已注册域、工作流程和活动不存在于其他区域中。例如，如果您在 us-east-1 和 us-west- 2 中都创建了一个名为 MySampleDomain “” 的域，则它们将作为单独的域存在：与您的域名关联的所有工作流程、任务列表、活动或数据都不会跨区域共享。

如果您在工作流程中使用其他 Amazon 资源，例如 Amazon EC2 实例，则这些资源也必须与您的 Amazon SWF 资源位于同一区域。唯一的例外是跨区域的服务，如 Amazon S3 和 IAM。在任何支持这些服务的区域中，您都可以从工作流程访问这些服务。

Amazon Simple Workflow Service 的其他文档

除了本开发人员指南之外，以下文档也很有用。

Amazon Simple Workflow Service API Reference

Amazon Simple Workflow Service API Reference 提供了有关 Amazon SWF HTTP API 的详细信息，包括操作、请求和响应结构以及错误代码。

Amazon Flow Framework 文档

Amazon Flow Framework 是一个编程框架，能够简化使用 Amazon SWF 管理其工作流和活动的分布式异步应用程序的实现过程，以便您能够集中精力实现您的工作流逻辑。

每种语言 Amazon Flow Framework 都按照其设计语言的惯用语进行设计，因此您可以自然地使用自己选择的语言来实现具有 Amazon SWF 所有优势的工作流程。

有一个适用于 Java 的 Amazon 流程框架。Amazon Flow Framework 适用于 Java 的开发人员指南提供了有关如何获取、设置和使用 Amazon Flow Framework 适用于 Java 的信息。

Amazon 软件开发工具包文档

Amazon 软件开发套件 (SDKs) 提供多种不同编程语言的 Amazon SWF 访问权限。它们严格 SDKs 遵循 HTTP API，但也为某些 Amazon SWF 功能提供了特定语言的编程接口。您可以访问以下链接了解有关每个软件开发工具包的更多信息。

> **Note**
>
> SDKs 此处仅列出了在撰写本文时支持 Amazon SWF 的产品。要查看可用工具的完整列表 Amazon SDKs，请访问亚马逊 Web Services 工具页面。

Java

为 Amazon 基础设施服务 适用于 Java 的 Amazon SDK 提供了 Java API。

要查看可用文档，请参阅 适用于 Java 的 Amazon SDK Documentation 页面。您也可以通过访问以下链接直接转到 SDK 参考中的 Amazon SWF 部分：

- `Class: AmazonSimpleWorkflowClient`
- `Class: AmazonSimpleWorkflowAsyncClient`
- `Interface: AmazonSimpleWorkflow`
- `Interface: AmazonSimpleWorkflowAsync`

JavaScript

适用于 JavaScript 的 Amazon SDK 允许开发人员使用浏览器或服务器上的 Node.js 应用程序内部提供的简单的 easy-to-use API 来构建利用 Amazon 服务的库或应用程序。

要查看可用文档，请参阅 适用于 JavaScript 的 Amazon SDK Documentation 页面。您也可以通过访问此链接直接转到 SDK 参考中的 Amazon SWF 部分：

- `Class: AWS.SimpleWorkflow`

.NET

适用于 .NET 的 Amazon SDK 是一个可下载的单一软件包，其中包括 Visual Studio 项目模板、Amazon .NET 库、C# 代码示例和文档。 适用于 .NET 的 Amazon SDK 这使得 Windows 开发人员可以更轻松地为 Amazon SWF 和其他服务构建.NET 应用程序。

要查看可用文档，请参阅 适用于 .NET 的 Amazon SDK Documentation 页面。您也可以通过访问以下链接直接转到 SDK 参考中的 Amazon SWF 部分：

- Namespace: Amazon.SimpleWorkflow
- Namespace: Amazon.SimpleWorkflow.Model

PHP

为 Amazon SWF Amazon SDK for PHP 提供了 PHP 编程接口。

要查看可用文档，请参阅 Amazon SDK for PHP Documentation 页面。您也可以通过访问此链接直接转到 SDK 参考中的 Amazon SWF 部分：

- Class: SwfClient

Python

为亚马逊 SWF Amazon SDK for Python (Boto) 提供了 Python 编程接口。

要查看可用文档，请参阅 boto: A Python interface to Amazon Web Services 页面。您也可以通过访问以下链接直接转到文档中的 Amazon SWF 部分：

- Amazon SWF Tutorial
- Amazon SWF Reference

Ruby

为 Amazon SWF 适用于 Ruby 的 Amazon SDK 提供了 Ruby 编程接口。

要查看可用文档，请参阅 适用于 Ruby 的 Amazon SDK Documentation 页面。您也可以通过访问此链接直接转到 SDK 参考中的 Amazon SWF 部分：

- 类别： AWS::Simple工作流程

Amazon CLI 文档

Amazon Command Line Interface (Amazon CLI) 是用于管理 Amazon 服务的统一工具。只需下载和配置一个工具，您就可以从命令行控制多项 Amazon 服务，并通过脚本自动执行这些服务。

有关更多信息 Amazon CLI，请参阅Amazon Command Line Interface页面。

有关 Amazon SWF 可用命令的概述，请参阅《Amazon CLI Command Reference》中 的 swf。

Amazon Simple Workflow Service 的 Web 资源

您可以使用多种 Web 资源来了解有关 Amazon SWF 的更多信息，或者获取有关使用该服务和开发工作流方面的帮助。

Amazon SWF 论坛

Amazon SWF 论坛为您提供了一个与其 Amazon SWF 开发人员和 Amazon SWF 开发团队成员交流的场所，以便您提出问题并获得答案。

您可以通过访问该论坛，地址为：Forum: Amazon Simple Workflow Service。

Amazon SWF 常见问题

“Amazon SWF 常见问题”提供了有关 Amazon SWF 的常见问题的答案，包括常见使用案例的概述、Amazon SWF 与其他服务的区别等。

您可在此处访问常见问题：Amazon SWF 常见问题。

Amazon SWF 视频

上的 Amazon Web Services 频道为包括亚马逊 SWF 在内的所有亚马逊网络服务 YouTube提供视频培训。要查看亚马逊 SWF 相关视频的完整列表，请使用以下查询：亚马逊 Web Services 中的简单工作流程

Ruby Flow 的迁移选项

fo Amazon Flow Framework r Ruby 的版本已不再处于积极开发阶段。现有代码仍将无限期继续运行，但不会再有新功能或新版本。本主题将介绍继续使用 Amazon SWF 的使用和迁移选项，以及有关如何迁移到 Step Functions 的信息。

选项	描述
继续使用 Ruby Flow Framework	目前，Ruby Flow Framework 将继续使用。如果您不执行任何操作，您的代码将继续按现有方式运行。计划在不久的将来迁移出 Amazon Flow Framework 适用于 Ruby 的。
迁移到 Java Flow Framework	Java Flow Framework 仍在进行积极开发，并将继续收到新功能和更新。
迁移到 Step Functions	Step Functions 可通过由状态机控制的可视化工作流程，来协调分布式应用程序的组件。
直接使用 SWF API，无需 Flow Framework	您可以直接使用 SWF API 继续在 Ruby 中工作，而无需使用 Ruby Flow Framework。

无论对于 Ruby 还是 Java，Flow Framework 提供的优势均在于，可让您专注于工作流程逻辑。该架构可处理大多数通信和协作的详细信息，并将某些复杂问题抽象化。您可以通过迁移到 Java Flow Framework 来继续使用相同级别的抽象，也可以直接与 Amazon SWF SDK 交互。

继续使用 Ruby Flow Framework

在 Amazon Flow Framework 短期内，for Ruby 将继续像现在一样发挥作用。如果您在 Ruby 中编写了 Amazon Flow Framework 工作流程，那么这些工作流程将继续起作用。在没有更新、支持或安全修复的情况下，您最好制定明确的计划，以便在不久的将来从适用于 Ruby 的 Amazon Flow Framework 迁移出来。

迁移到 Java Flow Framework

Amazon Flow Framework 适用于 Java 的将继续积极开发中。从概念上讲，for Java 与 Ruby 类似：您仍然可以专注于工作流程逻辑，该框架将有助于管理您的决策者逻辑，并将使 Amazon SWF 的其他方面更易于管理。 Amazon Flow Framework Amazon Flow Framework

- Amazon Flow Framework for Java
- Amazon Flow Framework 适用于 Java API 参考

迁移到 Step Functions

Amazon Step Functions 提供的服务与 Amazon SWF 类似，但您的工作流程逻辑由状态机控制。借助 Step Functions，您可以使用可视化工作流来协调分布式应用程序和微服务的组件。您可通过能执行离散函数（或称为任务）的各单独组件构建应用程序，这样您能够快速扩展和更改应用程序。Step Functions 提供了一种可靠的方法来协调组件并逐步执行应用程序的函数。图形控制台可将应用程序的组件直观地展示为一系列步骤。它可以自动触发和跟踪各个步骤，并在出现错误时重试，因此您的应用程序每次都能按照预期顺序执行。Step Functions 会记录每个步骤的状态，这样在出现错误时，您就能够迅速诊断并调试问题。

在 Step Functions 中，您可以使用声明性 JSON 编写的状态机来管理各项任务的协调，该状态机是使用 Amazon States Language 定义的。通过使用状态机，您不必编写和维护决定程序来控制您的应用程序逻辑。Step Functions 提供了一种直观、高效且敏捷的方法来使用可视化工作流协调应用程序组件。你应该考虑在所有新应用程序中使用，Step Functions Amazon Step Functions 为你目前在 for Ruby 中实现的工作流程提供了一个很好的迁移平台。 Amazon Flow Framework

为了帮助您将任务迁移到 Step Functions，同时继续利用 Ruby 语言技能，Step Functions 提供了一个 Ruby 活动工作线程示例。该示例采用了实现活动工作线程的最佳实践，可用作将任务逻辑迁移到

Step Functions 的模板。有关更多信息，请参阅《Amazon Step Functions Developer Guide》https://docs.amazonaws.cn/step-functions/latest/dg/中的 Example Activity Worker in Ruby。

> **Note**
>
> 对于许多客户来说，从 for Ruby 迁移到 Step Amazon Flow Framework Functions 是最佳选择。但是，如果您要求信号干预您的进程，或者需要启动将结果返回给父进程的子进程，请考虑直接使用 Amazon SWF API，或者迁移到 for Java Amazon Flow Framework 的。

有关更多信息 Amazon Step Functions，请参阅：

- Amazon Step Functions 开发人员指南
- Amazon Step Functions API 引用
- Amazon Step Functions 命令行参考

直接使用 Amazon SWF API

虽然 f Amazon Flow Framework or Ruby 可以管理亚马逊 SWF 的某些复杂性，但你也可以直接使用亚马逊 SWF API。直接使用 API 可通过构建工作流程使您完全控制任务的执行和协作，无需担心跟踪任务进度和维持任务状态等底层复杂性。

- Amazon Simple Workflow Service Developer Guide
- Amazon Simple Workflow Service API Reference

文档历史记录

下表说明了自 Amazon Simple Workflow Service Developer Guide 上次发布以来对该文档所做的重要更改。

更改	描述	更改日期
仅文档更新	Amazon SWF 现在包含一个关于 Amazon 用户通知的部分 Amazon Web Services 服务 ，该部分充当中 Amazon 通知的中心位置。 Amazon Web Services Management Console有关更多信息，请参阅 Amazon 用户通知服务 与 Amazon 简单工作流程服务配合使用。	2023 年 5 月 4 日
新特征	Amazon SWF 现已在中国地区 Amazon PrivateLink 提供支持。有关更多信息，请参阅 适用于 Amazon S3 的 Amazon VPC 端点。	2023 年 3 月 31 日
更新	更新了任务执行限额部分以包括`Maximum tasks scheduled per second`，并更新了的亚马逊 SWF 指标 CloudWatch页面，以包含有关使用非 ASCII 资源名称的信息。 CloudWatch	2021 年 5 月 12 日
新特征	亚马逊简单工作流程服务现在支持亚马逊 EventBridge。有关更多信息，请参阅： • EventBridge 适用于亚马逊 SWF • EventBridge 用户指南	2020 年 12 月 18 日

新特征	Amazon Simple Workflow Service 支持使用标签的 IAM 权限。有关更多信息，请参阅下列内容。
	• 亚马逊 SWF 中的标签
	  • 管理标签
	  • 标记工作流程执行
	  • 使用标签控制对域名的访问权限
	• `TagResource`
	• `UntagResource`
		2019 年 6 月 20 日

更改	描述	更改日期
	• ListTagsForResource • RegisterDomain	
新特征	Amazon Simple Workflow Service 现已在欧洲地区（斯德哥尔摩）区域推出。	2018 年 12 月 12 日
更新	改进了 Amazon 简单工作流程服务中关于 CloudTrail 集成的主题。请参阅使用录制 API 调用 Amazon CloudTrail。	2018 年 8 月 7 日
更新	添加了有关新PendingTasks 指标的信息 CloudWatch。有关更多信息，请参阅 Amazon SWF 指标。	2018 年 18 月 6 日
更新	在代码示例中高亮显示的改进语法。	2018 年 3 月 29 日
更新	添加主题，介绍 Ruby Flow 用户从该平台进行迁移的选项。有关更多信息，请参阅 Ruby Flow 的迁移选项。	2018 年 3 月 9 日
更新	改进了高级概念主题的导航。请参阅Amazon SWF 中的高级工作流程概念。	2018 年 2 月 19 日
更新	通过添加有效的统计信息改进了 CloudWatch 指标文档。请参阅的亚马逊 SWF 指标 CloudWatch。	2017 年 12 月 4 日
更新	更改了 TOC 以改进文档结构。添加了有关 API 和决策事件指标 的新信息。	2017 年 11 月 9 日
更新	将 Amazon SWF 限额一节更新为包括对所有区域的限制。	2017 年 10 月 18 日
更新	亚马逊 SWF 入门中已将 task_list 更改为 workflowId ，以免与 activity_list 混淆。	2017 年 25 月 7 日
更新	清除了本指南中的代码示例。	2017 年 6 月 5 日
更新	简化并改进了本指南的组织结构和内容。	2017 年 5 月 19 日

更改	描述	更改日期
更新	更新和链接补丁。	2017 年 5 月 16 日
更新	更新和链接补丁。	2016 年 10 月 1 日
Lambda 任务支持	除了传统活动任务之外，您还可以在工作流中指定 Lambda 任务。有关更多信息，请参阅 Amazon Lambda 亚马逊 SWF 中的任务。	2015 年 7 月 21 日
支持设置任务优先级	Amazon SWF 现在支持在任务列表中设置任务优先级，并尝试先交付优先级较高的任务，然后再交付优先级较低的任务。在 Amazon SWF 中设置任务优先级 提供了有关如何设置工作流和活动的任务优先级的信息。	2014 年 12 月 17 日
更新	添加了一个新主题，介绍如何使用以下方式记录 Amazon SWF API 调用 CloudTrail：。使用录制 API 调用 Amazon CloudTrail	2014 年 5 月 8 日
更新	添加了两个与 Amazon SWF CloudWatch 指标相关的新主题：的亚马逊 SWF 指标 CloudWatch，它提供了支持的指标的列表和查看亚马逊 SWF 指标以使用 CloudWatch Amazon Web Services Management Console描述；以及提供了有关如何使用查看指标和设置警报的信息。 Amazon Web Services Management Console	2014 年 4 月 28 日
更新	增加了新的部分：亚马逊 SWF 的其他资源和参考信息。这部分提供了一些服务参考信息，并为 Amazon SWF 开发人员提供了有关其他文档、示例、代码和其他 Web 资源的信息。	2014 年 3 月 19 日
更新	添加了工作流程教程。请参阅亚马逊 SWF 入门。	2013 年 10 月 25 日
更新	增加了 Amazon CLI 信息和示例。	2013 年 8 月 26 日

更改	描述	更改日期
更新	更新和补丁。	2013 年 8 月 1 日
更新	更新了该文档以说明如何使用 IAM 进行访问控制。	2013 年 2 月 22 日
首次发布	本指南是《Amazon Simple Workflow Service Developer Guide》的第一个版本。	2012 年 10 月 16 日

本文属于机器翻译版本。若本译文内容与英语原文存在差异，则一律以英文原文为准。