# 什么是AWS Step Functions

* AWS Step Functions 是一项 Web 服务，能够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。您可通过能执行任务（离散函数）的各单独组件构建应用程序，能够快速扩展和更改应用程序。
* Step Functions 提供可靠的方法来协调组件和逐步执行应用程序的方法。Step Functions 提供图形控制台，以一系列步骤展示应用程序的组件。
* 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够按照预期顺序执行。Step Functions 记录每个步骤的状态，这样在出现错误时，您能够迅速诊断并调试问题。
* Step Functions 管理操作和底层基础设施，确保应用程序在任何规模下均可用。
* 可在 AWS 云、在自己的服务器或者任何可访问 AWS 的系统上运行自己的任务。可以使用 [Step Functions 控制台](https://console.aws.amazon.com/states/home?region=us-east-1#/)、AWS 开发工具包或 HTTP API 访问和使用 Step Functions。

# 标准与快速工作流

* 标准工作流非常适合长时间运行、持久且可审计的工作流。它们最多可以运行 1 年，可以使用 [Step Functions API](https://docs.aws.amazon.com/step-functions/latest/apireference) 检索完整的执行历史记录，最长在执行完成后 90 天。
* 标准工作流采用最多一次模型，除非您在 ASL 中指定了 Retry 行为，否则您的任务和状态永远不会多次执行。这使得它们适合编排非幂等操作，例如启动 Amazon EMR 集群或处理付款。标准工作流执行会根据处理的状态转换次数计费。
* 快速工作流是适用于大批量事件处理工作负载，例如 IoT 数据提取、流数据处理和转换以及移动应用程序后端。它们最多可以运行五分钟。快速工作流使用至少一次模型，在这种模型中可能会运行多次执行。这使得它们非常适合编排幂等操作，例如转换输入数据和通过 PUT 存储到 Amazon DynamoDB 中。快速工作流的执行按执行次数、执行持续时间和消耗的内存计费。
* 标准和快速工作流可以自动启动以响应事件，例如通过 Amazon API Gateway 进行的 HTTP 请求（大规模完全托管的 API）、IoT 规则以及 Amazon EventBridge 中超过 140 个事件源。

# 状态

* 各个状态可以根据其输入做出决定，执行操作并将输出传递给其他状态。在 AWS Step Functions 中，您可以使用 Amazon 状态语言 定义工作流。Step Functions 控制台提供了该状态机的图形表示，以帮助可视化您的应用程序逻辑。
* 状态是您状态机中的元素。状态通过其*名称* 来引用，这可以是任意字符串，但在整个状态机的范围内必须唯一。
* 在状态机中，状态可以执行各种函数：
* 在状态机中执行一些工作（[任务](https://docs.aws.amazon.com/zh_cn/step-functions/latest/dg/amazon-states-language-task-state.html)状态）
* 在执行分支之间进行选择 ([Choice](https://docs.aws.amazon.com/zh_cn/step-functions/latest/dg/amazon-states-language-choice-state.html) 状态)
* 停止执行，返回失败或成功 ([Fail](https://docs.aws.amazon.com/zh_cn/step-functions/latest/dg/amazon-states-language-fail-state.html) 或 [Succeed](https://docs.aws.amazon.com/zh_cn/step-functions/latest/dg/amazon-states-language-succeed-state.html) 状态)
* 单纯将其输入传递到其输出或者注入一些固定数据 ([Pass](https://docs.aws.amazon.com/zh_cn/step-functions/latest/dg/amazon-states-language-pass-state.html) 状态)
* 提供一定时间量的延迟或直至指定时间/日期 ([Wait](https://docs.aws.amazon.com/zh_cn/step-functions/latest/dg/amazon-states-language-wait-state.html) 状态)
* 开始执行的并行分支 ([Parallel](https://docs.aws.amazon.com/zh_cn/step-functions/latest/dg/amazon-states-language-parallel-state.html) 状态)
* 动态迭代步骤（[映射](https://docs.aws.amazon.com/zh_cn/step-functions/latest/dg/amazon-states-language-map-state.html)状态）

手机屏幕截图

描述已自动生成

* 每个状态必须有一个 Type 字段，指示状态是什么类型。
* 每个状态可以有可选的 Comment 字段，以人类可读的格式存放有关状态的评论或说明。
* 每个状态 (除了 Succeed 或 Fail 状态) 需要一个 Next 字段，或者也可以通过指定 End 字段成为最终状态

# 状态语言

* 用于定义状态机的基于 JSON 的结构化语言，状态机是一个[状态](https://docs.aws.amazon.com/zh_cn/step-functions/latest/dg/concepts-states.html)集合，可以执行工作（Task状态），确定哪些状态转换为下一个状态（Choice状态），在出错的情况下停止执行（Fail状态）等等。
* 状态机结构 JSON文本定义

1.Comment 可读的说明性文本（可选）

2.StartAt 一个字符串，必须与某个状态对象的名称完全匹配

3. TimeoutSeconds执行失败会显示States.Timeout（可选）

4.states 一个对象，包含逗号分隔的状态合集

状态机是由其包含的状态以及状态之间的关系决定的

* Type 状态类型

**Pass状态**，将输入传递到输出，构造和调试状态机时很有用

**Result状态**，处理传递到下一个状态的虚拟任务输出，按照ResultPath字段进行筛选

**ResultPath状态，**指定 (输入中) 用于放置执行 Result 中虚拟任务的“输出”的位置。输入将进一步按照 OutputPath 字段（如果存在）指定的内容进行筛选，然后再用作状态输出。

Parameters状态，创建将作为输入传递的键值对集合。值可以是静态的，也可以从具有路径的输入中选择。

* Next 当前状态完成以后指向的下一个状态名称,如果指向choice，则会允许多个状态转换
* End 如果设置为True,那么此分支会结束，一个状态只能有一个Next和End,某些类型不能出现Eed，比如Choice类型
* Comment给人看的描述语句
* InputPath 一个路径，用于传递到状态任务进行处理的状态输入的一部分，如果省略则其值为$，表示指定整个输入
* OutputPath 一个路径用于要传递到状态输入的一部分，如果省略则为$，表示指定整个输入

# Task状态（任务状态）——工作单元

* 任务通过使用活动、AWS Lambda 函数或将参数传递给其他服务的 API 操作来执行工作。
* Step Function可以直接从任务状态调用Lambda函数—原生云任务，运行于Step Function
* 活动Activity由程序代码组成，它们等待运算符执行操作或者提供输入。可以在 Amazon EC2 上、在 Amazon ECS 上甚至在移动设备上托管活动。Activity用 GetActivityTask 和 SendTaskSuccess、SendTaskFail-ure 以及 SendTaskHeartbeat API 操作轮询 Step Functions。Amazon 状态语言通过将状态的类型设置为 Task 并且向任务提供活动或 Lambda 函数的 Amazon 资源名称 (ARN) 来呈现任务。
* **常用字段**（除了常用的状态字段外Task特有的一些）
* Resource (必需)
* 一个 URI，尤其是唯一标识要执行的特定任务的 ARN。
* Parameters（可选）
* 用于将信息传递给所连接资源的 API 操作。参数可以混合使用静态 JSON 和 [JsonPath](https://github.com/json-path/JsonPath" \t "_blank)。
* ResultPath（可选）
* 指定（输入中）用于放置 Resource 中所指定任务的执行结果的位置。接下来，输入将按照 OutputPath 字段（如果存在）指定的内容进行筛选，然后再用作状态输出。有关更多信息，请参阅[路径](https://docs.aws.amazon.com/zh_cn/step-functions/latest/dg/concepts-input-output-filtering.html)。
* Retry（可选）
* 一个称为重试器的对象的数组，定义在状态遇到运行时错误时的重试策略。
* Catch (可选)
* 一个称为捕获器的对象的数组，用于定义回退状态。如果状态遇到运行时错误并且其重试策略已耗尽或者未定义，则执行该状态。
* TimeoutSeconds（可选）
* 如果任务运行时间超过了指定的秒数，则该状态将失败，并返回 States.Timeout 错误名称。必须为非零正整数。如果未提供，则默认值为 99999999。任务启动后开始计数，例如，当 ActivityStarted 或 LambdaFunctionStarted 记录在 Execution event history (执行事件历史记录) 中。
* HeartbeatSeconds（可选）
* 如果任务检测信号之间的时间超过了指定秒数，则该状态将失败，并返回 States.Timeout 错误名称。必须为非零正整数，并小于在 TimeoutSeconds 字段中指定的秒数。如果未提供，则默认值为 99999999。对于活动，当 GetActivityTask 收到令牌并且 ActivityStarted 记录在执行事件历史记录中时开始计数。
* Task 状态必须将 End 字段设置为 true（如果状态结束执行），或者必须在 Next 字段中提供一个状态（该状态将在 Task 状态完成时运行）。

# 在Tsak中指定资源 ARN格式

Resource 字段的 ARN 是使用以下模式指定的。

arn:*partition*:*service*:*region*:*account*:*task\_type*:*name*

不支持跨分区、区域或账户引用 ARN,在该模式中：

* partition 是要使用的 AWS Step Functions 分区，最常用的是 aws。
* service 指示用于执行任务的 AWS 服务，并且为以下之一：
  + states，表示一个[活动](https://docs.aws.amazon.com/zh_cn/step-functions/latest/dg/amazon-states-language-task-state.html" \l "amazon-states-language-task-state-activity)。
  + lambda，表示一个 [Lambda 函数](https://docs.aws.amazon.com/zh_cn/step-functions/latest/dg/amazon-states-language-task-state.html#amazon-states-language-task-state-lambda)。
* region 是在其中创建 Step Functions 活动或状态机类型或 Lambda 函数的 [AWS 区域](https://docs.aws.amazon.com/general/latest/gr/rande.html)。
* account 是您的 AWS 账户 ID。
* task\_type 是要运行的任务的类型。它为以下值之一：
  + activity – 一个[活动](https://docs.aws.amazon.com/zh_cn/step-functions/latest/dg/amazon-states-language-task-state.html" \l "amazon-states-language-task-state-activity)。
  + function – 一个 [Lambda 函数](https://docs.aws.amazon.com/zh_cn/step-functions/latest/dg/amazon-states-language-task-state.html#amazon-states-language-task-state-lambda)。
  + *servicename* – 支持的连接服务的名称
* name 是已注册的资源名称（活动名称、Lambda 函数名称或服务 API 操作）