1STRATEGY SERVERLESS ARCHITECTURE WORKSHOP

These are the work instructions that parallel the Serverless Architectures presentation materials

TABLE OF CONTENTS

Table of Contents

PURPOSE **1**

INTENDED AUDIENCE 1

SET UP **2**

TRAINING ACCOUNT ACCESS 2

LOG INTO THE TRAINING ACCOUNT 2

NOTE TAKING APPLICATION 2

CURL UTILITY 2

HANDS-ON INSTRUCTIONS **3**

LAMBDA 3

Concepts: *3*

Create a Lambda Function *4*

STEP FUNCTIONS 6

Concepts *6*

Create a State Machine *6*

API GATEWAY 7

Concepts *7*

Create an API *7*

SNS 11

Create an SNS Topic *11*

SQS 13

Concepts *13*

Create an SQS queue *13*

DYNAMODB 14

Create a DynamoDB Table *14*

RUNNING THE ENTIRE STATE MACHINE 16

PREPARED BY **18**

COMPANY INFORMATION **18**

PURPOSE

Purpose

The purpose of this training is to have users walk away with a working prototype of a Serverless Application on AWS using Lambda, Step Functions, API Gateway, SNS, SQS, and DynamoDB.

INTENDED AUDIENCE

The intended audience for training is people who are interested in understanding and building “Serverless” applications. To be successful, attendees should have some understanding of or experience with:

- Writing code
- Application Architecture
- General AWS Services (IAM)

Set Up

TRAINING ACCOUNT ACCESS

You should have received an email with credentials to the 1Strategy Training Account. If you haven't received these credentials, send an email to Training@1strategy.com.

LOG INTO THE TRAINING ACCOUNT

Log into the 1strategy training account by using the credentials provided before starting this lab.

NOTE TAKING APPLICATION

There are several points during this lab you will have to retain certain pieces of information, please have something for taking notes handy.

CURL UTILITY

This lab will require the use of the **curl** utility.

If you're using a Mac this can be accessed by using the **terminal**.

If you're using a PC or Chromebook, you can use a free online utility here: http://onlinecurl.com/ to execute the curl commands for the lab.

Hands-on Instructions

This hands-on portion is broken into 6 separate parts. Each part relates to one of the AWS services covered in the presentation material.

LAMBDA

Concepts:

handler_name(event, context):

The **handler** is the "main" function that will be run when the lambda function is invoked.

http://docs.aws.amazon.com/lambda/latest/dg/python-programming-model-handler-types.html

handler_name(event, context):

The **event** object– AWS Lambda uses this parameter to pass in event data to the handler. This parameter is usually of the Python **dict** type. It can also be list, **str**, **int**, **float**, or **NoneType** type.

http://docs.aws.amazon.com/lambda/latest/dg/python-programming-model-handler-types.html

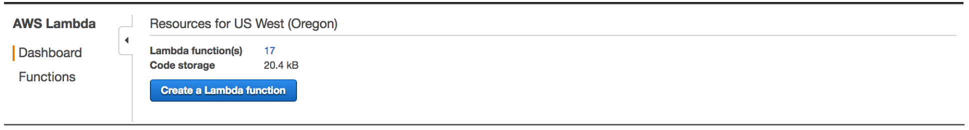
handler_name(event, context):

The **context** object – AWS Lambda uses this parameter to provide runtime information to your handler.

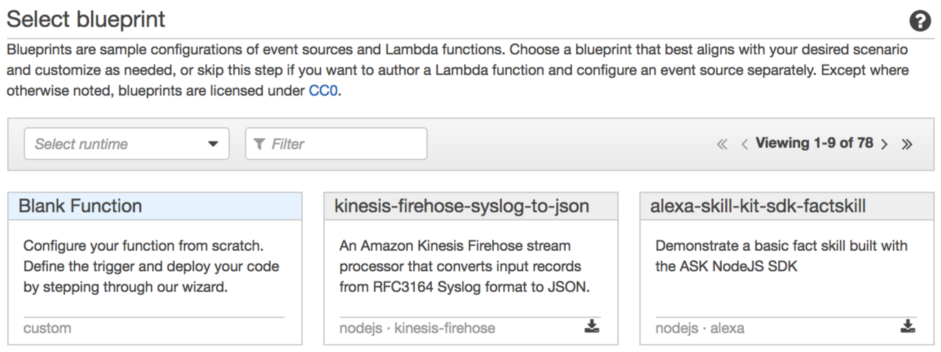
http://docs.aws.amazon.com/lambda/latest/dg/python-context-object.html

Create a Lambda Function

1. Navigate to https://us-west-2.console.aws.amazon.com/lambda/home?region=us-west-2
2. Click on **Create a Lambda Function**



3. Select the **Blank Function** box


Select blueprint

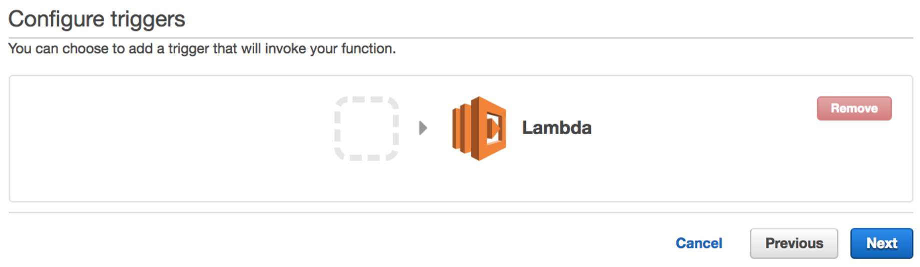
Blueprints are sample configurations of event sources and Lambda functions. Choose a blueprint that best aligns with your desired scenario and customize as needed, or skip this step if you want to author a Lambda function and configure an event source separately. Except where otherwise noted, blueprints are licensed under CC0.

Blank Function: Configure your function from scratch. Define the trigger and deploy your code by stepping through our wizard. custom

kinesis-firehose-syslog-to-json: An Amazon Kinesis Firehose stream processor that converts input records from RFC3164 Syslog format to JSON. nodejs · kinesis-firehose

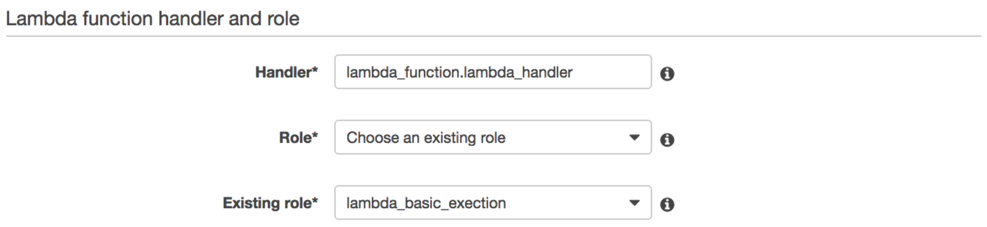
alexa-skill-kit-sdk-factskill: Demonstrate a basic fact skill built with the ASK NodeJS SDK. nodejs · alexa

4. Press **Next**


Configure triggers

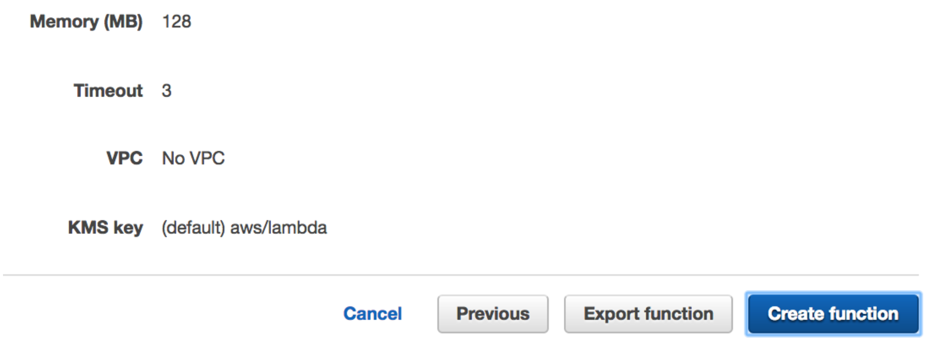
You can choose to add a trigger that will invoke your function.

5. Enter **your-name-lambda** as the name of the lambda function:
6. Select **Python 2.7** as the Runtime.
   Note: We will be using Python examples in this workshop, however if you are sufficiently comfortable with Node.JS, it can be used as well.
7. Select **serverless_lambda_basic_execution** as the Existing Role.


Lambda function handler and role

Handler*	lambda_function.lambda_handler
Role*	Choose an existing role
Existing role*	lambda_basic_exection

8. Leave the rest of the parameters as their default values and press **Next** then press **Create Function**



Memory (MB) 128

Timeout 3

VPC No VPC

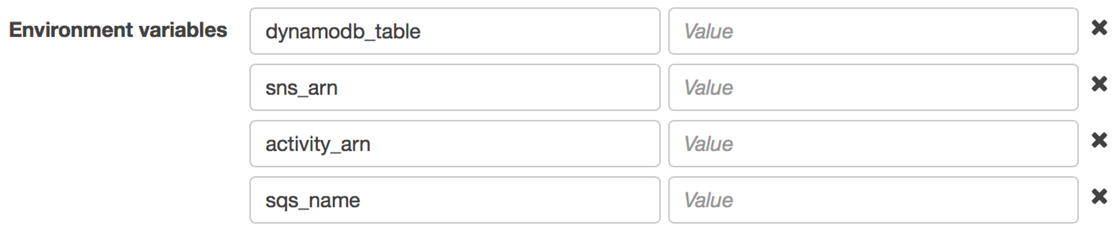
KMS key (default) aws/lambda

9. The Lambda function you just created will be used to pass the other AWS resources created in this lab (SQS, SNS, Activity, DynamoDB) as parameters into your Step Functions state machine (next session of this lab).
10. Enter the below code into your lambda function (if you copy and paste, double check your quotes to make sure they're not the fancy ones that Word uses).

```
import os
def lambda_handler(event, context):
        event['activity_arn'] = os.environ['activity_arn']
        event['sns_arn'] = os.environ['sns_arn']
        event['sqs_name'] = os.environ['sqs_name']
        event['dynamodb_table'] = os.environ['dynamodb_table']

        event['commander'] = {
                "rank":"<REPLACE ME>",
                "name":"<REPLACE ME>"
                }
        return event
```

11. Update the <REPLACE ME> items
12. Update the environment variables of your function as pictured below (you can leave the values blank for now, we will be updating them through the lab):



Environment variables		
	dynamodb_table	Value
	sns_arn	Value
	activity_arn	Value
	sqs_name	Value

13. Press **Save**

14. Copy the ARN of the lambda (arn:aws:lambda:us-west-2:281782457076:function:**your-name-lambda**) to your note taking application
15. To confirm our function works, we can test it by pressing on **Save and Test**
16. This is the of end the Lambda Section

STEP FUNCTIONS

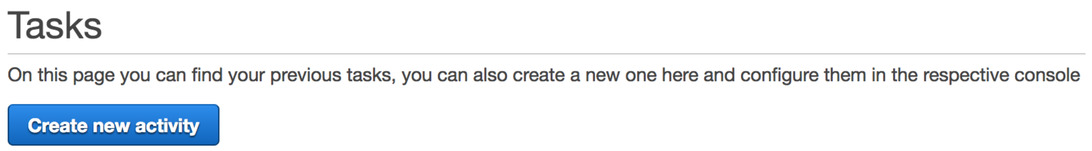
Concepts

A **State Machine** in AWS is an orchestration tool for lambda functions. It allows for complicated functionality (retries, decision trees, etc) while still allowing for the application to remain as loosely coupled as possible

An **Activity** in Step Function is an endpoint where worker nodes can receive instructions to perform tasks that require resource not natively supported by step functions (e.g. some compute resource other than a lambda function). Once the worker node has completed the task, it can send a success or failure notification back to the state machine via the activity.

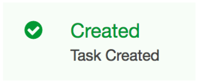
Create a State Machine

1. Navigate to https://us-west-2.console.aws.amazon.com/states/home?region=us-west-2#/tasks
2. Click **Create New Activity**, name the activity **your-name-activity**


Tasks

On this page you can find your previous tasks, you can also create a new one here and configure them in the respective console

3. Press **Create Activity**
4. A green confirmation message should appear

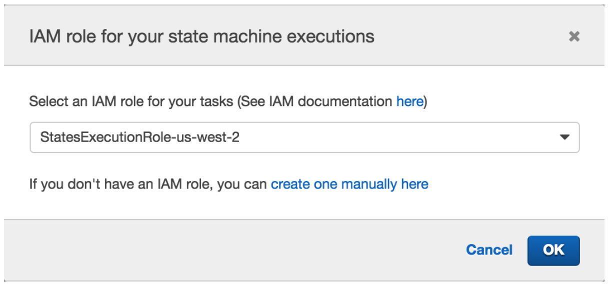

Created

Task Created

5. Copy the Activity ARN (arn:aws:states:us-west-2:281782457076:activity:**your-name-activity**) to your note taking application
6. Navigate to https://us-west-2.console.aws.amazon.com/states/home?region=us-west-2#/
7. Press **Create state machine**
8. Take time to explore the sample state machines
9. Under **Give your state machine a name,** enter **your-name-state-machine**
10. Copy the SERVERLESS STATE MACHINE JSON from https://github.com/1Strategy/workshop-serverless/blob/master/serverless_state_machine_attendee.json
11. Paste the JSON into **CODE** section
12. In the JSON, find the <REPLACE ME> and insert the ARNs for your lambda function (line 7) and activity (line 157) from your notes
13. Press the **circular arrows** next to **Preview** for a visualization of the state machine

14. Press **Create state machine**

15. Select the **StateExecutionRole-us-west-2** as the IAM role then press **Ok**



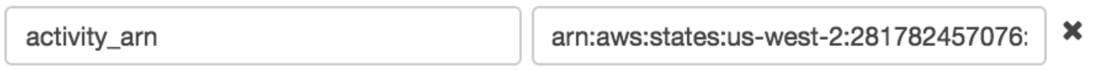
IAM role for your state machine executions

Select an IAM role for your tasks (See IAM documentation here)

StatesExecutionRole-us-west-2

If you don't have an IAM role, you can create one manually here

16. Click on **Dashboard**
17. Find and Copy the State Machine ARN (arn:aws:states:us-west-2:281782457076:stateMachine:**your-name-state-machine**) to your note taking application
18. Now that you've created your activity, update your lambda function from the Lambda section with an environment variable named: **activity_arn** with the value of the ARN from the activity you created in this section



activity_arn

arn:aws:states:us-west-2:281782457076:

19. Save the lambda function
20. This is the end of the Step Function section

API GATEWAY

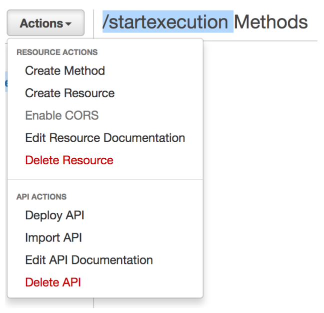
Concepts

A **resource** is a part of a URL www.example.com/resource

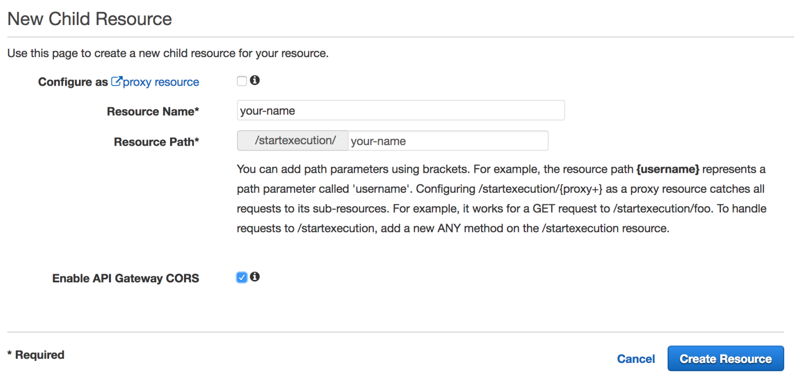
A **method** is an HTTP verb that tells a URL endpoint what action the client is taking and how the server should respond (GET, PUT, POST, DELETE, etc.)

Create an API

1. Navigate to https://us-west-2.console.aws.amazon.com/apigateway/home?region=us-west-2#/apis/
2. Select **ServerlessAPI**
3. Click on **/startexecution**
4. Click on **Actions** then **Create Resource**



5. Enter **your-name** as **Resource Name** and check the **Enable API Gateway CORS** then press **Create Resource**



New Child Resource

Use this page to create a new child resource for your resource.

Configure as proxy resource ☐

Resource Name* your-name

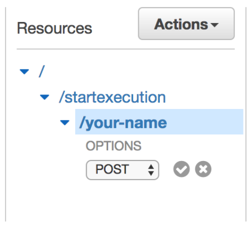
Resource Path* /startexecution/ your-name

You can add path parameters using brackets. For example, the resource path **{username}** represents a path parameter called 'username'. Configuring /startexecution/{proxy+} as a proxy resource catches all requests to its sub-resources. For example, it works for a GET request to /startexecution/foo. To handle requests to /startexecution, add a new ANY method on the /startexecution resource.

Enable API Gateway CORS ☑

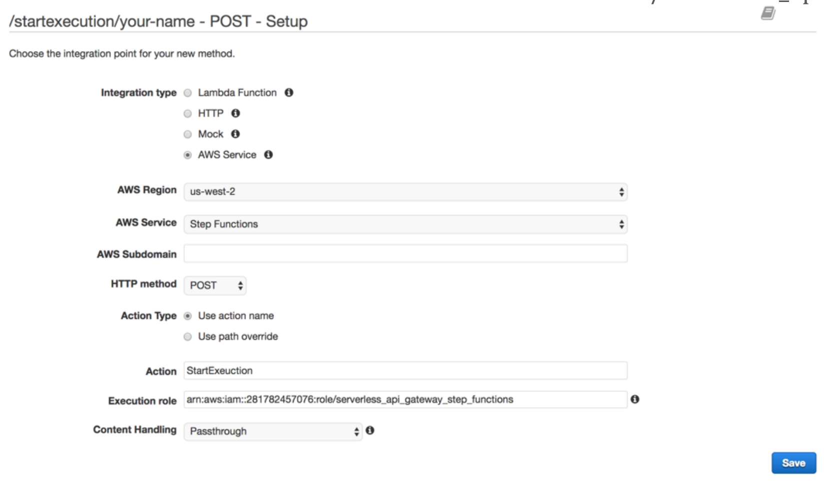
* Required

6. Click on **/your-name** then select **Actions** and **Create Method**. Select **POST** then press the **Checkmark** to the right of the drop-down menu



7. Click the **radio button** next to **AWS Service**

8. Set **AWS Region** as **us-west-2**
9. Set **AWS Service** to **Step Functions**
10. Set **HTTP Method** to **POST**
11. Set **Action** to **StartExecution**
12. Set **Execution role** to: arn:aws:iam::281782457076:role/serverless_api_gateway_step_functions



/startexecution/your-name - POST - Setup

Choose the integration point for your new method.

Integration type: Lambda Function / HTTP / Mock / [✓ AWS Service]

AWS Region: us-west-2

AWS Service: Step Functions

AWS Subdomain:

HTTP method: POST

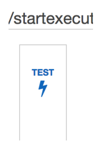
Action Type: [✓ Use action name] / Use path override

Action: StartExecution

Execution role: arn:aws:iam::281782457076:role/serverless_api_gateway_step_functions

Content Handling: Passthrough

13. Press **Save**
14. To test your API, Click on **Test**

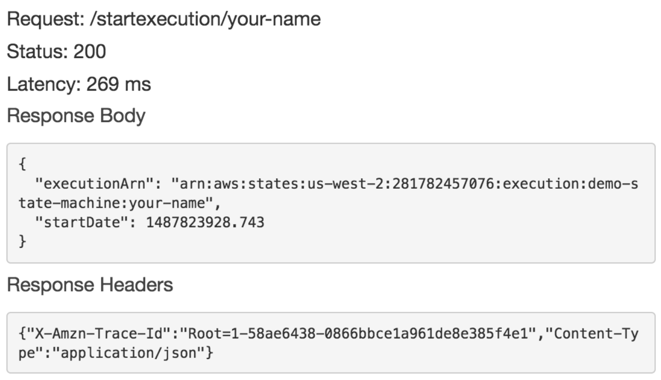


15. Enter the below information in the **Request Body**

```
{
        "input": "{}",
        "name": "SomeName", "stateMachineArn":
        "arn:aws:states:us-west-2:281782457076:stateMachine:demo-state-machine"
}
```

Note: The state machine in this request is for testing purposes only, you will be using your own state-machine later in the workshop

16. Press the **Test** button at the bottom of the screen
17. The follow output should appear



Request: /startexecution/your-name

Status: 200

Latency: 269 ms

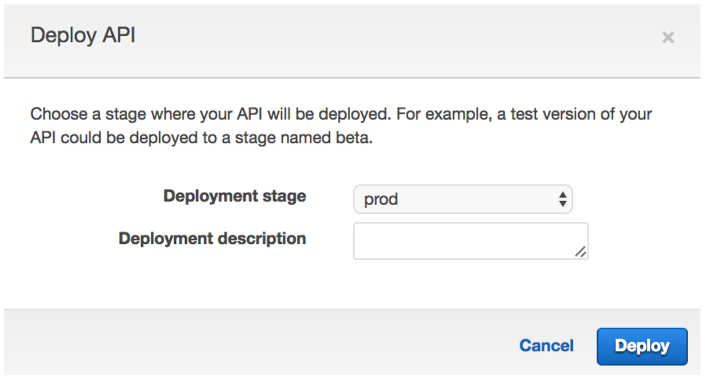
Response Body

```
{
  "executionArn": "arn:aws:states:us-west-2:281782457076:execution:demo-s
tate-machine:your-name",
  "startDate": 1487823928.743
}
```

Response Headers

```
{"X-Amzn-Trace-Id":"Root=1-58ae6438-0866bbce1a961de8e385f4e1","Content-Ty
pe":"application/json"}
```

18. If you don't see the success message, please complain loudly to the instructor
19. Press **Actions** then **Deploy API**



Deploy API

Choose a stage where your API will be deployed. For example, a test version of your API could be deployed to a stage named beta.

Deployment stage prod

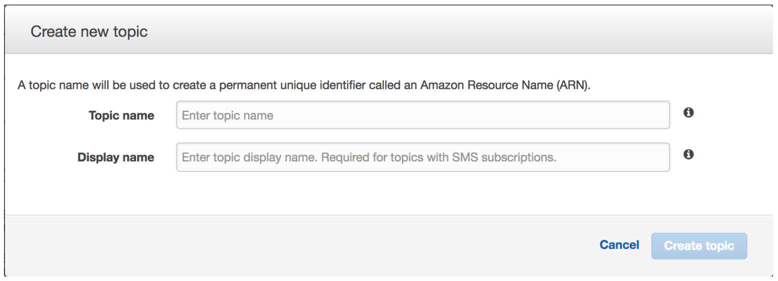
Deployment description

20. This is the end of the API Gateway Portion

SNS

Create an SNS Topic

1. Navigate to: https://us-west-2.console.aws.amazon.com/sns/v2/home?region=us-west-2#/home
2. Click on **Create Topic**
3. Enter **your-name-topic** as the **Topic name** and **Serverless** as the **Display name** then press **Create Topic**.



Create new topic

A topic name will be used to create a permanent unique identifier called an Amazon Resource Name (ARN).

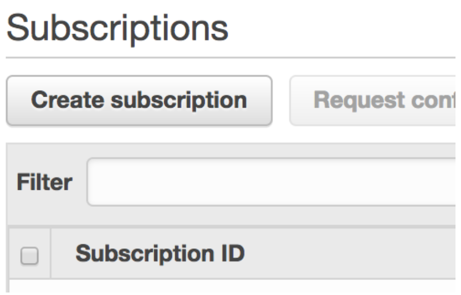
Topic name: Enter topic name

Display name: Enter topic display name. Required for topics with SMS subscriptions.

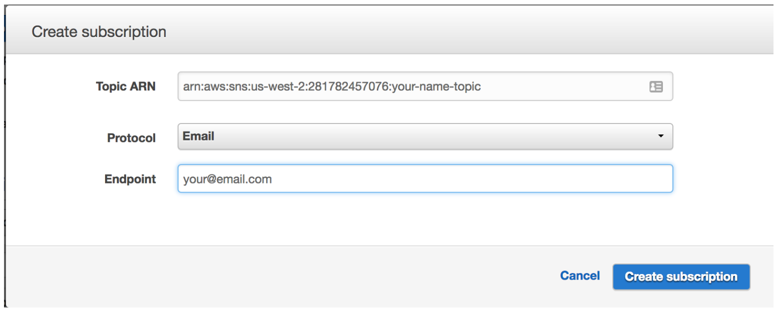
4. Click on the **ARN** of the newly created topic

arn:aws:sns:us-west-2:281782457076:your-name-topic

5. Press **Create Subscription**



6. Change the **Protocol** to **Email**. Enter your email address into **Endpoint** and press **Create subscription**



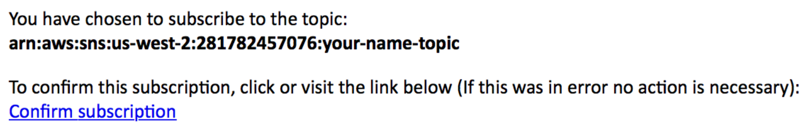
Create subscription

Topic ARN: arn:aws:sns:us-west-2:281782457076:your-name-topic

Protocol: Email

Endpoint: your@email.com

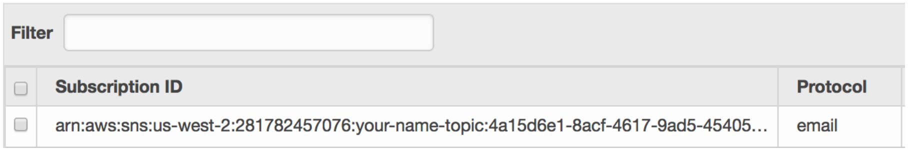
7. You should get an email that will prompt you to confirm your subscription. Click on the **Confirm subscription** link



You have chosen to subscribe to the topic:
arn:aws:sns:us-west-2:281782457076:your-name-topic

To confirm this subscription, click or visit the link below (If this was in error no action is necessary):
Confirm subscription

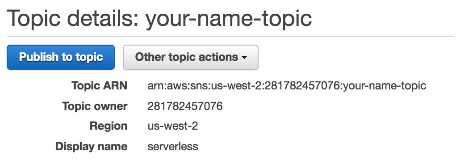
8. Copy the ARN of your SNS topic to your notes
9. Navigate back to your topic https://us-west-2.console.aws.amazon.com/sns/v2/home?region=us-west-2#/topics/ and if you see the below image, you now have successfully subscribed to your topic



Filter

	Subscription ID	Protocol
☐	arn:aws:sns:us-west-2:281782457076:your-name-topic:4a15d6e1-8acf-4617-9ad5-45405…	email

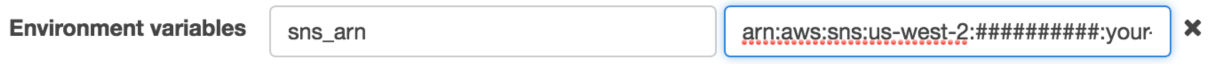
10. To send yourself a sample message, press **Publish to topic**.



Topic details: your-name-topic

Topic ARN	arn:aws:sns:us-west-2:281782457076:your-name-topic
Topic owner	281782457076
Region	us-west-2
Display name	serverless

11. Enter a **Subject** and **Message** then press **Publish message**. You should receive your message in your email
12. Now that you've created your SNS topic, update your lambda function from the Lambda section with an environment variable named: **sns_arn** with the value of the ARN from the SNS topic you just created



Environment variables | sns_arn | arn:aws:sns:us-west-2:##########:your

13. Save the lambda function
14. This is the end of the SNS section

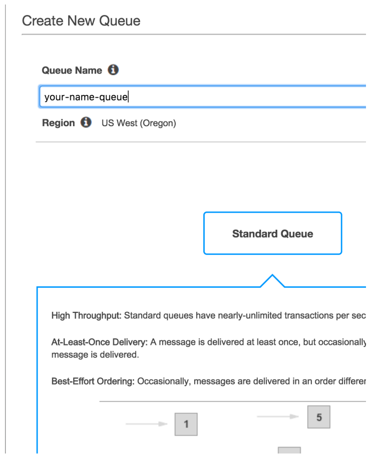
HANDS-ON INSTRUCTIONS

SQS

Concepts

Create an SQS queue

1. Navigate to https://console.aws.amazon.com/sqs/home?region=us-west-2
2. Press **Create new queue**
3. Enter **your-name-queue** under **Queue Name** and select **Standard Queue** then press **Quick create queue** at the bottom of the page



Create New Queue

Queue Name: your-name-queue

Region: US West (Oregon)

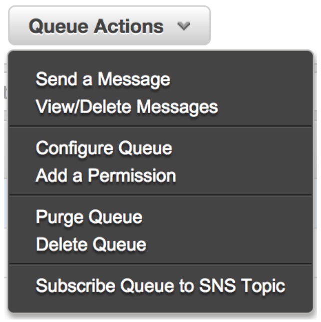
Standard Queue

High Throughput: Standard queues have nearly-unlimited transactions per sec

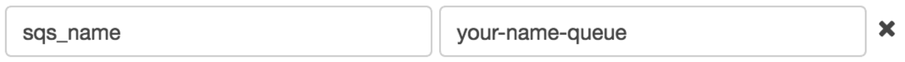
At-Least-Once Delivery: A message is delivered at least once, but occasionally message is delivered.

Best-Effort Ordering: Occasionally, messages are delivered in an order differer

4. Spend some time exploring (e.g. Select **your-name-queue** then press **Queue Actions** and **Send a message**. View your message by pressing **Queue Actions** and **View/Delete Messages**)



5. Once you're done exploring clear your test messages. Select **Queue Actions, View/Delete Messages,** select your test messages, and then **Purge Messages.**
6. Update your lambda function from the Lambda section with an environment variable named: **sqs_name** with the value of the **your-name-queue** that was just created



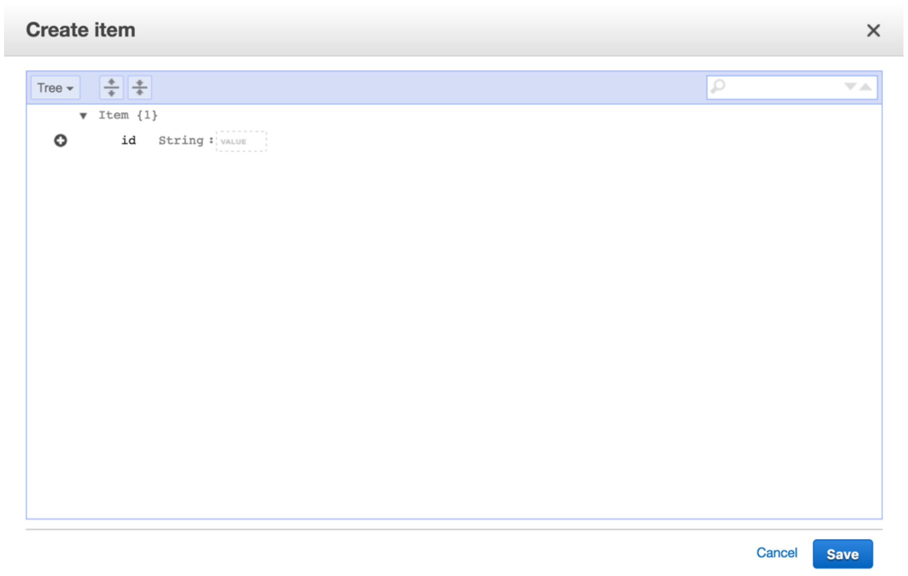
sqs_name | your-name-queue

7. This is the end of the SQS section

DYNAMODB

Create a DynamoDB Table

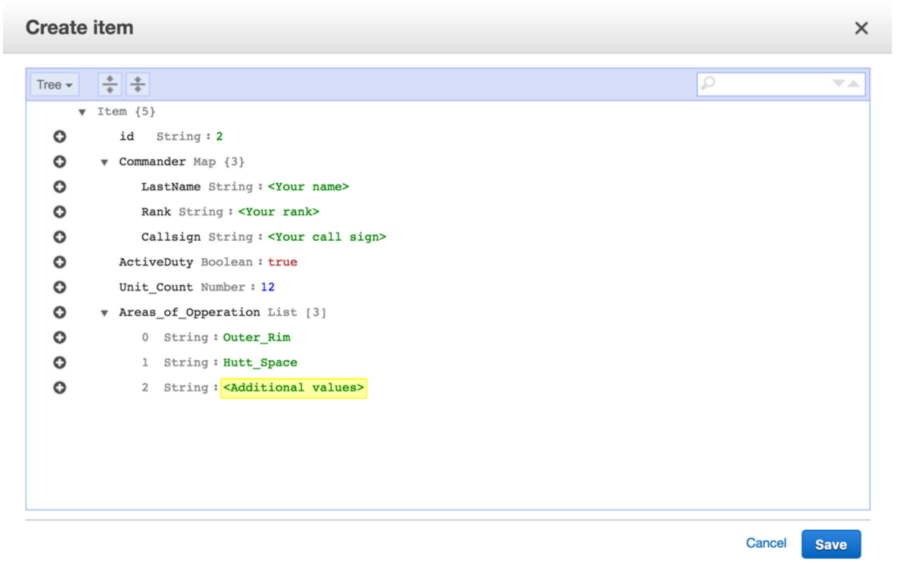
1. Navigate to https://us-west-2.console.aws.amazon.com/dynamodb/home?region=us-west-2
2. Press **Create table**
3. Enter **your-name-table** under **Table Name**
4. Enter **id** under **Partition key**
5. Ensure that the **String** option is selected in the data type field. This table should not need a **sort key**.
6. Leave the **Use default settings** radio button checked
7. The table should take 2 – 5 minutes to provision
8. With your table selected from the **Tables** tab, click on the **Items** tab
9. Click **Create item.** You should see the following window popup



```
▾ Item {1}
   id   String : VALUE
```

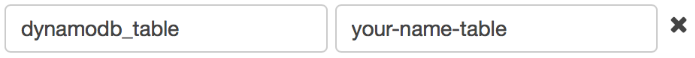
10. Type a **number** in the **value** field (the number you input will be stored as a string type)
11. Click the
12. When the drop-down menu appears, select the **Append** option
13. You will see another drop-down menu with a list of data types that you can add.
14. Choose the **Map** value
15. You have now inserted a blank Map object into this DynamoDB table. Now we will populate this map object with attributes.
16. Enter **Commander** in the **FIELD** box for the Map object
17. Append a **String** type item to the map

18. In the **FIELD** box type **Rank** and input your rank in the **VALUE** box
19. Append two more **String** types to the **Map** object, one for your last name and one for your call sign
20. Click the ⊕ next to the **Map** object to append items after the map (instead of appending attributes within the map)
21. Append a **Boolean** item type with **FIELD**: **ActiveDuty**, and **VALUE**: **true**
22. Append a **Number** type with **FIELD**: **Unit_Count** and **VALUE**: **12**
23. Append a **List** type with **FIELD**: **Areas_of_Opperation**
24. Append a few **String** attributes into the **List** item
25. Your **Create item** window should look something like this



```
▾ Item {5}
⊕     id   String : 2
⊕   ▾ Commander Map {3}
⊕       LastName String : <Your name>
⊕       Rank String : <Your rank>
⊕       Callsign String : <Your call sign>
⊕     ActiveDuty Boolean : true
⊕     Unit_Count Number : 12
⊕   ▾ Areas_of_Opperation List [3]
⊕       0  String : Outer_Rim
⊕       1  String : Hutt_Space
⊕       2  String : <Additional values>
```

26. Click **Save**
27. You should now see your item in the **Items** tab of your table
28. Take a few minutes to explore the other tabs before moving on
29. When you are finished looking around, make sure your lambda function has the environment variable named: **dynamodb_table** with the value of the **your-name-table** that you just created



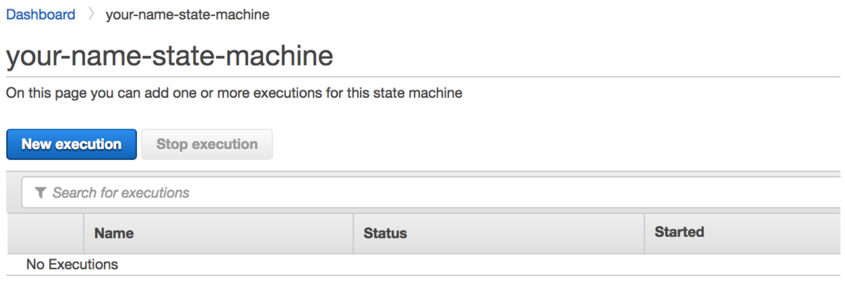
dynamodb_table	your-name-table

30. This is the end of the DynamoDB section

RUNNING THE ENTIRE STATE MACHINE

Now onto the fun part, now that we have all of the elements in place, we're going to destroy our planet.

1. Navigate to your state machine in the console https://us-west-2.console.aws.amazon.com/states/home?region=us-west-2#/



your-name-state-machine

On this page you can add one or more executions for this state machine

	Name	Status	Started
No Executions			

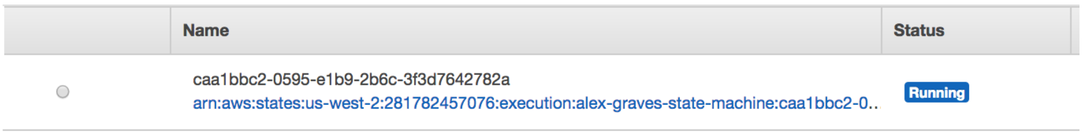
2. Open a Terminal and run the following command

```
curl -X POST -d '{ "input": "{}",   "name": "ATestOfThisBattleStation", "stateMachineArn":
"arn:aws:states:us-west-2:281782457076:stateMachine:your-name-state-machine"}' \
https://00dlxb9yr1.execute-api.us-west-2.amazonaws.com/prod/startexecution/your-name
```

3. You should get a response in the terminal that looks like this:

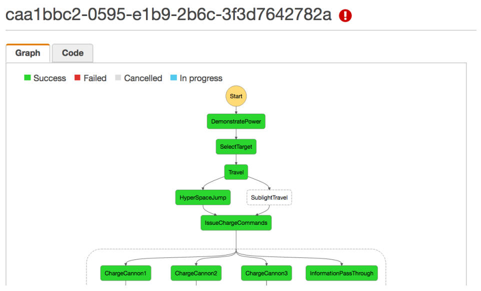
```
{
 "executionArn": "arn:aws:states:us-west-2:281782457076:execution:demo-state-machine:SomeName",
 "startDate": 1487912974.749
}
```

4. Click on the **Refresh** button for your state machine and an execution should appear with the **status** of **Running**



	Name	Status
○	caa1bbc2-0595-e1b9-2b6c-3f3d7642782a arn:aws:states:us-west-2:281782457076:execution:alex-graves-state-machine:caa1bbc2-0..	Running

5. Click on the **ARN** of the running execution
6. From here you can watch the execution progress



caa1bbc2-0595-e1b9-2b6c-3f3d7642782a

7. The bottom of the page will show the execution of the state machine step by step including failures and retries of states
8. If the all of the states in the execution are successful (all the states are green), you should get an email with **your-name** and **rank** as well as a long stream of characters which represent a **firing code**
9. If there is an error in a state, that state will turn red and can check the **input** and **output** of failed step to troubleshoot the issue
10. Once you receive the email copy the **firing code** in the **terminal** execution the code below

```
curl https://serverless.1strategy.com/enter-firing-code --data "<Firing Code Here>"
```

    Note: Alternatively, you can navigate https://serverless.1strategy.com and enter your firing code there.

11. If you do not execute the above command in under 120 seconds, the state machine will time out
12. In either case, you should get a notification in your email
13. This concludes this workshop

PREPARED BY

Prepared By

ALEX GRAVES AWS BIG DATA SOLUTIONS ARCHITECT	JUSTIN IRAVANI AWS SOLUTIONS ARCHITECT
alex@1strategy.com	Justin.Iravani@1strategy.com

Company Information

1Strategy
3025 112th Ave SE #200

Bellevue, WA 98004

www.1strategy.com