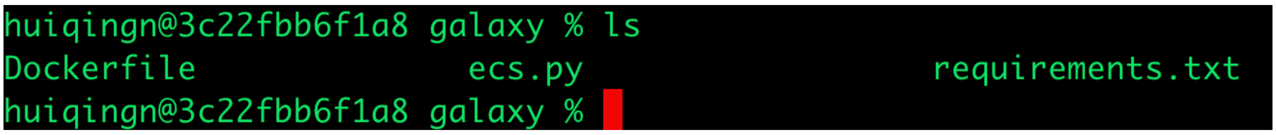
数据处理逻辑向ECS迁移

1.处理脚本示例

```python
import boto3
import os
import json

username = os.environ['MY_USER']
password = os.environ['MY_PASS']
tt = os.environ['MY_TASK_TOKEN']
print("Running with user: %s" % username)
print("Running with password: %s" % password)
print("Running with task token: %s" % tt)

client =
boto3.client('stepfunctions',aws_access_key_id="xxxxx",aws_secret_access_key="x
xxxx")
client.send_task_success(
        taskToken=tt,
        output=json.dumps({ "decision":"true"})
    )
```

2.制作镜像



```
huiqingn@3c22fbb6f1a8 galaxy % ls
Dockerfile                  ecs.py                      requirements.txt
huiqingn@3c22fbb6f1a8 galaxy % 
```

Dockerfile

```dockerfile
FROM python:3.6
RUN mkdir /code
WORKDIR /code
ADD . /code/
COPY ecs.py /code/
RUN pip install -r requirements.txt

EXPOSE 5000
CMD ["python", "/code/ecs.py"]`
```

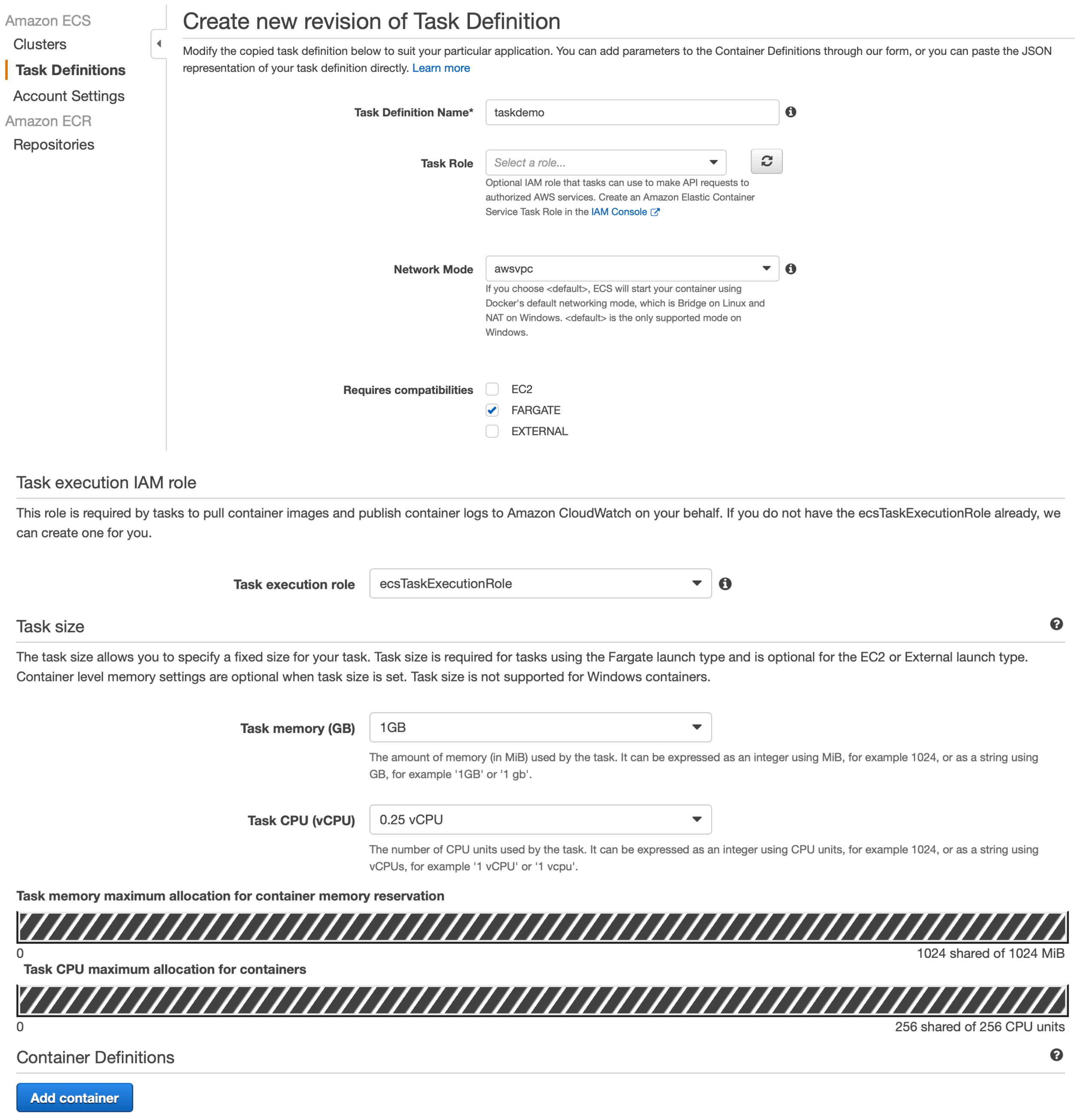
requirements.txt

```
boto3
```

3. 打包成镜像，上推到ECR镜像仓库

```
docker build -t mypython:v1 .
docker run -e MY_USER=huiqing -e MY_PASS=aaa mypython:v1
aws ecr get-login-password --region cn-north-1 | docker login --username AWS --
password-stdin xxx.dkr.ecr.cn-north-1.amazonaws.com.cn/galaxy
docker tag mypython:v1 xxx.dkr.ecr.cn-north-1.amazonaws.com.cn/galaxy:v2
docker push xxx.dkr.ecr.cn-north-1.amazonaws.com.cn/galaxy:latest
```

4.ECS 创建cluster，taskdefinition



Create new revision of Task Definition

Modify the copied task definition below to suit your particular application. You can add parameters to the Container Definitions through our form, or you can paste the JSON representation of your task definition directly. Learn more

Task Definition Name* taskdemo

Task Role Select a role...

Optional IAM role that tasks can use to make API requests to authorized AWS services. Create an Amazon Elastic Container Service Task Role in the IAM Console

Network Mode awsvpc

If you choose <default>, ECS will start your container using Docker's default networking mode, which is Bridge on Linux and NAT on Windows. <default> is the only supported mode on Windows.

Requires compatibilities
- [ ] EC2
- [x] FARGATE
- [ ] EXTERNAL

Task execution IAM role

This role is required by tasks to pull container images and publish container logs to Amazon CloudWatch on your behalf. If you do not have the ecsTaskExecutionRole already, we can create one for you.

Task execution role ecsTaskExecutionRole

Task size

The task size allows you to specify a fixed size for your task. Task size is required for tasks using the Fargate launch type and is optional for the EC2 or External launch type. Container level memory settings are optional when task size is set. Task size is not supported for Windows containers.

Task memory (GB) 1GB

The amount of memory (in MiB) used by the task. It can be expressed as an integer using MiB, for example 1024, or as a string using GB, for example '1GB' or '1 gb'.

Task CPU (vCPU) 0.25 vCPU

The number of CPU units used by the task. It can be expressed as an integer using CPU units, for example 1024, or as a string using vCPUs, for example '1 vCPU' or '1 vcpu'.

Task memory maximum allocation for container memory reservation

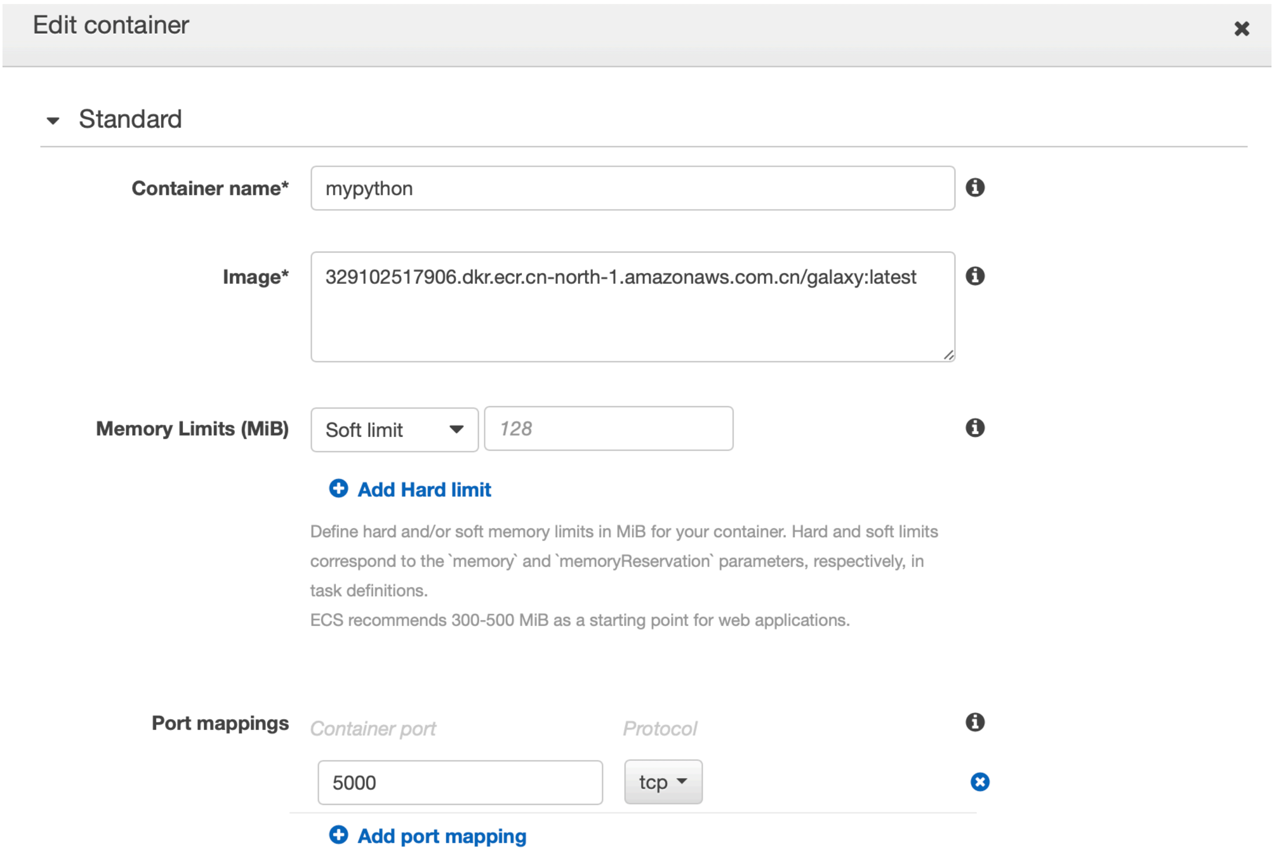
0 1024 shared of 1024 MiB

Task CPU maximum allocation for containers

0 256 shared of 256 CPU units

Container Definitions

进入add container：



Edit container

Standard

Container name* mypython

Image* 329102517906.dkr.ecr.cn-north-1.amazonaws.com.cn/galaxy:latest

Memory Limits (MiB) Soft limit 128

Add Hard limit

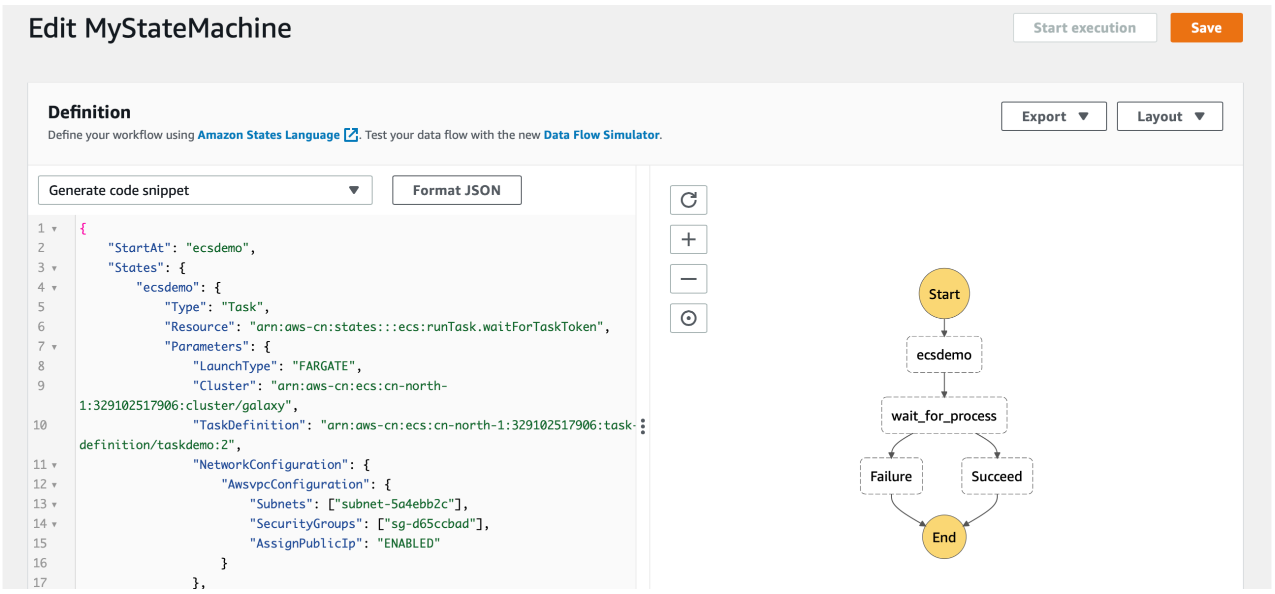
Define hard and/or soft memory limits in MiB for your container. Hard and soft limits correspond to the `memory` and `memoryReservation` parameters, respectively, in task definitions.
ECS recommends 300-500 MiB as a starting point for web applications.

Port mappings

Container port	Protocol
5000	tcp

Add port mapping

5. 创建stepfunction



Edit MyStateMachine

Definition

Define your workflow using Amazon States Language. Test your data flow with the new Data Flow Simulator.

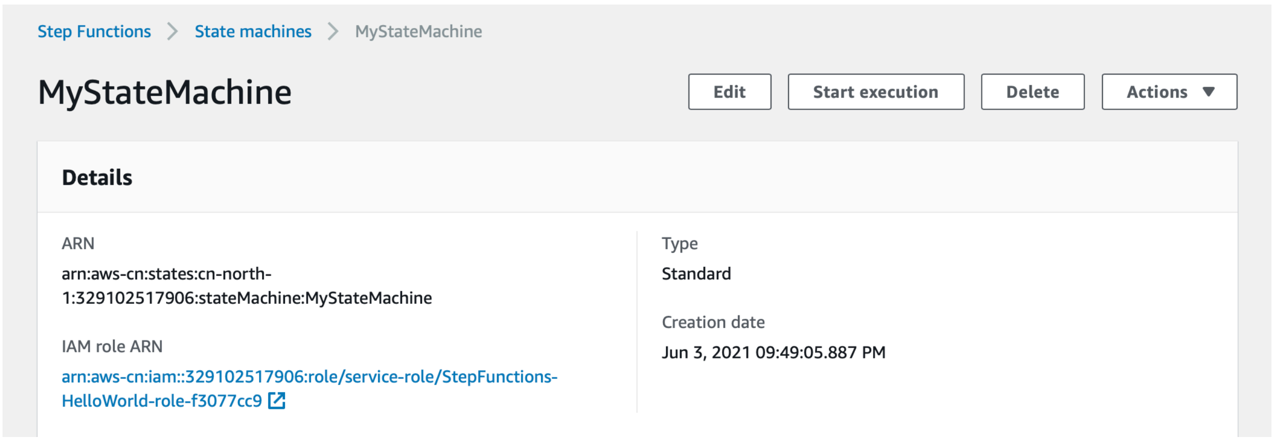
```
{
    "StartAt": "ecsdemo",
    "States": {
        "ecsdemo": {
            "Type": "Task",
            "Resource": "arn:aws-cn:states:::ecs:runTask.waitForTaskToken",
            "Parameters": {
                "LaunchType": "FARGATE",
                "Cluster": "arn:aws-cn:ecs:cn-north-
1:329102517906:cluster/galaxy",
                "TaskDefinition": "arn:aws-cn:ecs:cn-north-1:329102517906:task-
definition/taskdemo:2",
                "NetworkConfiguration": {
                    "AwsvpcConfiguration": {
                        "Subnets": ["subnet-5a4ebb2c"],
                        "SecurityGroups": ["sg-d65ccbad"],
                        "AssignPublicIp": "ENABLED"
                    }
                },
```

stepfunction创建脚本参考：

```
{
    "StartAt": "ecsdemo",
    "States": {
        "ecsdemo": {
            "Type": "Task",
            "Resource": "arn:aws-cn:states:::ecs:runTask.waitForTaskToken",
            "Parameters": {
                "LaunchType": "FARGATE",
                "Cluster": "arn:aws-cn:ecs:cn-north-
1:329102517906:cluster/galaxy",
                "TaskDefinition": "arn:aws-cn:ecs:cn-north-1:329102517906:task-
definition/taskdemo:2",
                "NetworkConfiguration": {
                    "AwsvpcConfiguration": {
                        "Subnets": ["subnet-5a4ebb2c"],
                        "SecurityGroups": ["sg-d65ccbad"],
                        "AssignPublicIp": "ENABLED"
                    }
                },
                "Overrides": {
                    "ContainerOverrides": [{
                        "Name": "mypython",
                        "Command": ["python3", "/code/ecs.py"],
                        "Environment": [{
                            "Name": "MY_PASS",
                            "Value.$": "$.my_pass"
                        }, {
                            "Name": "MY_USER",
                            "Value.$": "$.my_user"
                        },{
                          "Name":"MY_TASK_TOKEN",
                          "Value.$":"$$.Task.Token"
                        }]
                    }]
                }
            },
            "Retry": [{
                "ErrorEquals": ["States.TaskFailed"],
                "IntervalSeconds": 3,
                "MaxAttempts": 2,
                "BackoffRate": 1.5
            }],
            "Next": "wait_for_process"
        },
      "wait_for_process": {
            "Type": "Choice",
            "Choices": [
                {
                    "Variable": "$.decision",
                    "StringEquals": "true",
                    "Next": "Succeed"
                },
                {
                    "Variable": "$.decision",
                    "StringEquals": "false",
                    "Next": "Failure"
                }
```

```
              ]
            },
          "Failure": {
          "Type": "Fail",
          "Cause": "Invalid response.",
          "Error": "CheckFailed"
        },
        "Succeed": {
          "Type": "Succeed"
        }
        }
    }
```

6. 测试stepfunction



MyStateMachine

Details

ARN
arn:aws-cn:states:cn-north-1:329102517906:stateMachine:MyStateMachine

IAM role ARN
arn:aws-cn:iam::329102517906:role/service-role/StepFunctions-HelloWorld-role-f3077cc9

Type
Standard

Creation date
Jun 3, 2021 09:49:05.887 PM

start execution：输入参数

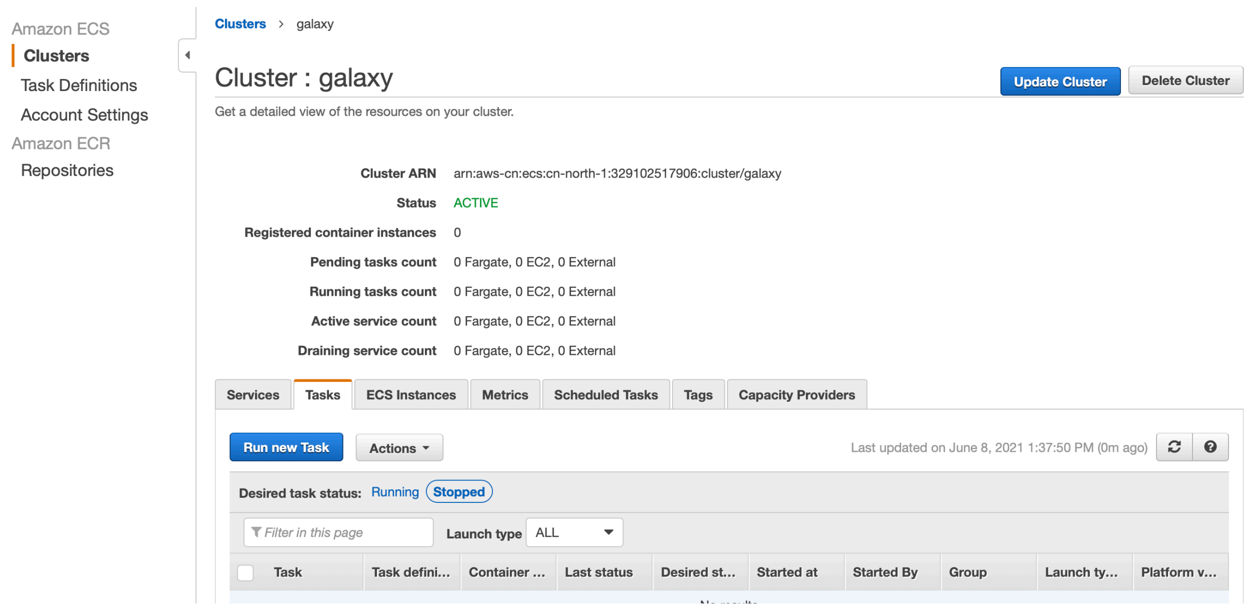
{

  "my_user": "sharonNi",

  "my_pass": "1234"

}

查看stepfunction，ecs task是否启动成功，并进入ecs task查看执行状态，进入可看到输出日志。



Amazon ECS
Clusters
Task Definitions
Account Settings
Amazon ECR
Repositories

Clusters › galaxy

Cluster : galaxy

Get a detailed view of the resources on your cluster.

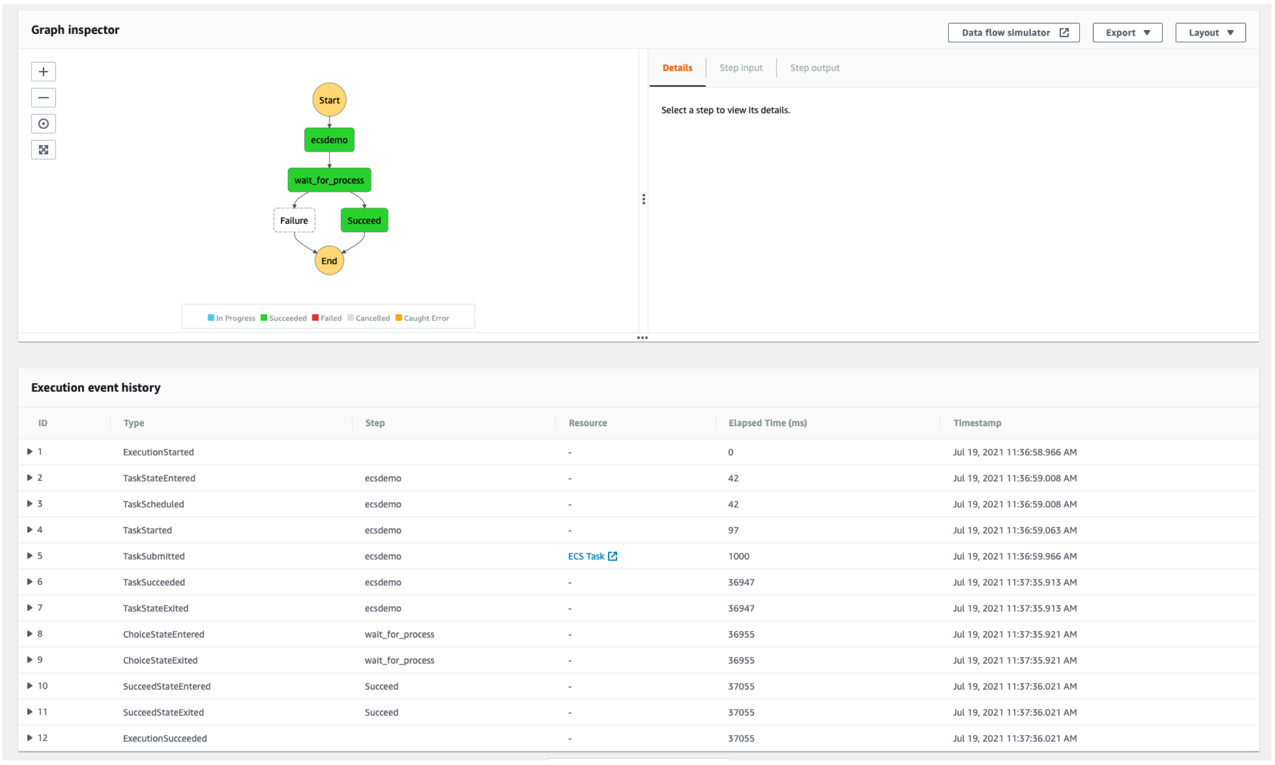
Cluster ARN	arn:aws-cn:ecs:cn-north-1:329102517906:cluster/galaxy
Status	ACTIVE
Registered container instances	0
Pending tasks count	0 Fargate, 0 EC2, 0 External
Running tasks count	0 Fargate, 0 EC2, 0 External
Active service count	0 Fargate, 0 EC2, 0 External
Draining service count	0 Fargate, 0 EC2, 0 External

Last updated on June 8, 2021 1:37:50 PM (0m ago)

Desired task status: Running [✓ Stopped]

Filter in this page Launch type ALL

Task	Task defini...	Container ...	Last status	Desired st...	Started at	Started By	Group	Launch ty...	Platform v...



Select a step to view its details.

Execution event history

ID	Type	Step	Resource	Elapsed Time (ms)	Timestamp
▶ 1	ExecutionStarted		-	0	Jul 19, 2021 11:36:58.966 AM
▶ 2	TaskStateEntered	ecsdemo	-	42	Jul 19, 2021 11:36:59.008 AM
▶ 3	TaskScheduled	ecsdemo	-	42	Jul 19, 2021 11:36:59.008 AM
▶ 4	TaskStarted	ecsdemo	-	97	Jul 19, 2021 11:36:59.063 AM
▶ 5	TaskSubmitted	ecsdemo	ECS Task	1000	Jul 19, 2021 11:36:59.966 AM
▶ 6	TaskSucceeded	ecsdemo	-	36947	Jul 19, 2021 11:37:35.913 AM
▶ 7	TaskStateExited	ecsdemo	-	36947	Jul 19, 2021 11:37:35.913 AM
▶ 8	ChoiceStateEntered	wait_for_process	-	36955	Jul 19, 2021 11:37:35.921 AM
▶ 9	ChoiceStateExited	wait_for_process	-	36955	Jul 19, 2021 11:37:35.921 AM
▶ 10	SucceedStateEntered	Succeed	-	37055	Jul 19, 2021 11:37:36.021 AM
▶ 11	SucceedStateExited	Succeed	-	37055	Jul 19, 2021 11:37:36.021 AM
▶ 12	ExecutionSucceeded		-	37055	Jul 19, 2021 11:37:36.021 AM

Stepfunctions 插入Lambda

1. 创建dynamodb table

名为galaxydb

2. 编写stepfunctions脚本

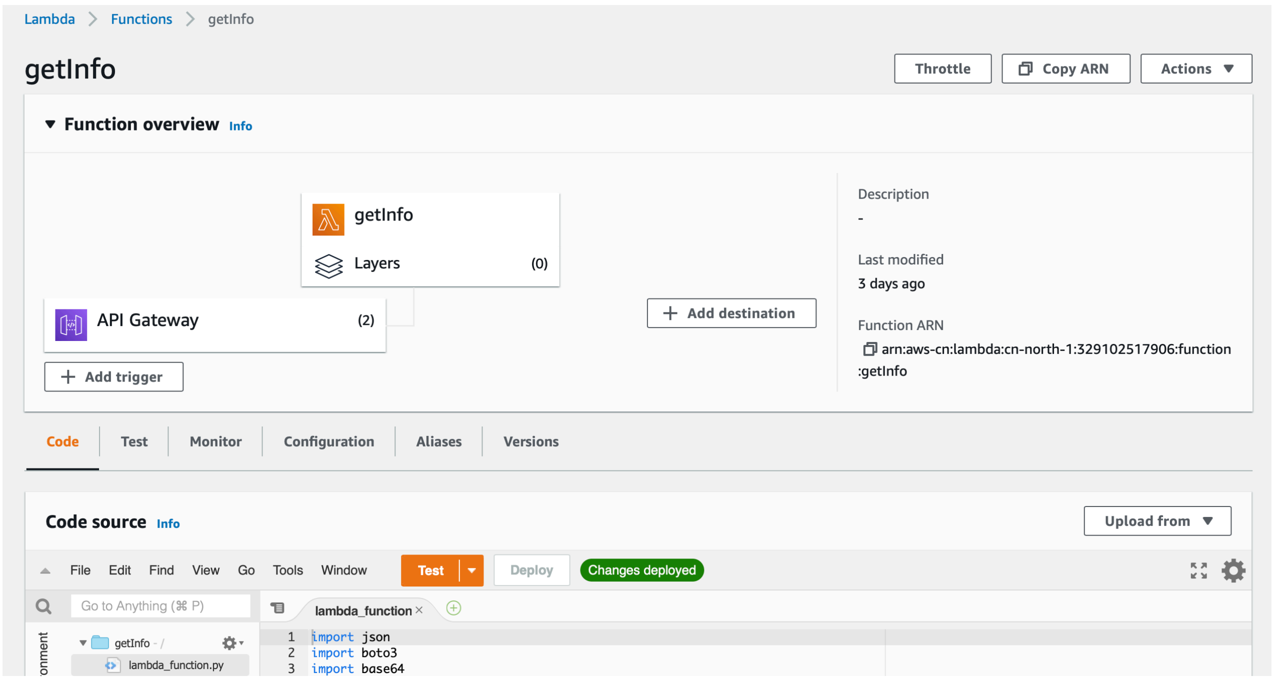
```
{
  "Comment": "A Hello World example of the Amazon States Language using Pass
states",
  "StartAt": "Put item into DynamoDB",
  "States": {
    "Put item into DynamoDB": {
      "Type": "Task",
      "Resource": "arn:aws-cn:states:::dynamodb:putItem",
      "Parameters": {
        "TableName": "galaxydb",
        "Item": {
          "uuid": {"S.$": "$.uuid"},
          "dbid": {"S.$": "$.dbid"},
          "userid": {"S.$": "$.userid"}
        }
      },
      "End": true
    }

}
}
```

2. 测试

使用如下参数，uuid改用唯一值，

```
{
    "uuid": "003",
    "dbid":"kingdee",
    "userid":"galaxy"
}
```

APIGW + Lambda 检查处理状态


Lambda > Functions > getInfo

getInfo

Throttle | Copy ARN | Actions

Function overview Info

getInfo
Layers (0)
API Gateway (2)
+ Add trigger
+ Add destination

Description
-

Last modified
3 days ago

Function ARN
arn:aws-cn:lambda:cn-north-1:329102517906:function:getInfo

Code | Test | Monitor | Configuration | Aliases | Versions

Code source Info

Upload from

File Edit Find View Go Tools Window Test Deploy Changes deployed

Go to Anything (⌘ P)

getInfo - /
lambda_function.py

lambda_function

```
import json
import boto3
import base64
```

1. Lambda函数编写：

以lambda方式直接调用，进行测试，输入json格式的参数:

{
  "key1": "001"
}

```
import json
import boto3
import base64
from botocore.exceptions import ClientError

def lambda_handler(event, context):
    print(event)
    uuid = event["key1"]

    dynamodb = boto3.resource('dynamodb')
    table = dynamodb.Table('galaxydb')

    #检查处理状态
    response = table.get_item( Key={
            'uuid': uuid
        }
    )

    return {
        'statusCode': 200,
        'body': json.dumps(response['Item'].get('userid'))
    }
```

event的内容：

{'key1': '001'}

通过APIGW方式调用，输入--data的string参数：

curl https://mlexr1c5dj.execute-api.cn-north-1.amazonaws.com.cn/default/getInfo --data '{"key1": "001"}'

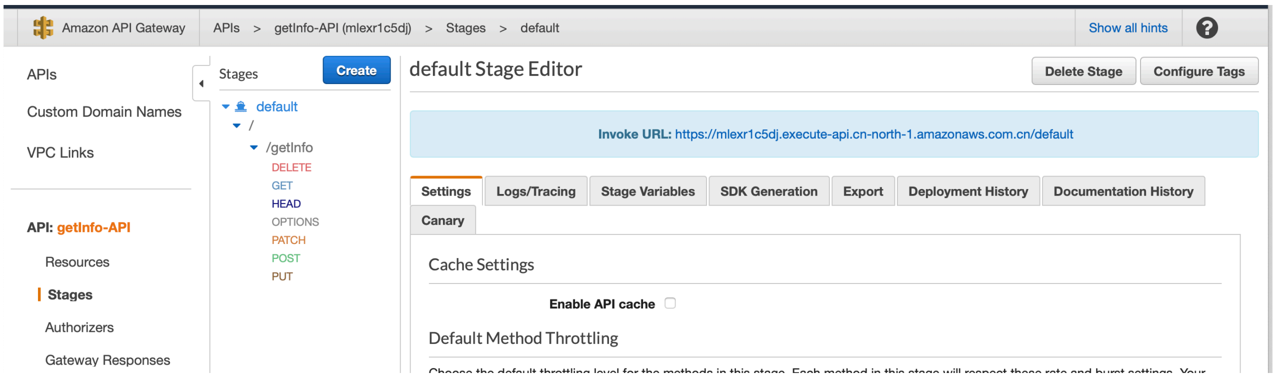
```
import json
import boto3
import base64
from botocore.exceptions import ClientError

def lambda_handler(event, context):
    print(event)
    print(event['body'])
    inputdata =json.loads(event['body'])

    uuid = inputdata["key1"]

    dynamodb = boto3.resource('dynamodb')
    table = dynamodb.Table('galaxydb')

    #检查处理状态
    response = table.get_item( Key={
            'uuid': uuid
        }
    )

    return {
        'statusCode': 200,
        'body': json.dumps(response['Item'].get('userid'))
    }
```

apigw调用时，event的内容如下

```
{'resource': '/getInfo', 'path': '/getInfo', 'httpMethod': 'POST', 'headers':
{'accept': '*/*', 'content-type': 'application/x-www-form-urlencoded', 'Host':
'mlexr1c5dj.execute-api.cn-north-1.amazonaws.com.cn', 'User-Agent':
'curl/7.64.1', 'X-Amzn-Trace-Id': 'Root=1-60f4f0d9-32b5185531f4947e097ae323',
'X-Forwarded-For': '54.222.45.2', 'X-Forwarded-Port': '443', 'X-Forwarded-
Proto': 'https'}, 'multiValueHeaders': {'accept': ['*/*'], 'content-type':
['application/x-www-form-urlencoded'], 'Host': ['mlexr1c5dj.execute-api.cn-
north-1.amazonaws.com.cn'], 'User-Agent': ['curl/7.64.1'], 'X-Amzn-Trace-Id':
['Root=1-60f4f0d9-32b5185531f4947e097ae323'], 'X-Forwarded-For':
['54.222.45.2'], 'X-Forwarded-Port': ['443'], 'X-Forwarded-Proto': ['https']},
'queryStringParameters': None, 'multiValueQueryStringParameters': None,
'pathParameters': None, 'stageVariables': None, 'requestContext':
{'resourceId': 'pdwr1z', 'resourcePath': '/getInfo', 'httpMethod': 'POST',
'extendedRequestId': 'CsqR9F7aBTIFqJQ=', 'requestTime': '19/Jul/2021:03:26:17
+0000', 'path': '/default/getInfo', 'accountId': '329102517906', 'protocol':
'HTTP/1.1', 'stage': 'default', 'domainPrefix': 'mlexr1c5dj',
'requestTimeEpoch': 1626665177191, 'requestId': 'c48f22b0-d855-4c42-8686-
f2a64309e55a', 'identity': {'cognitoIdentityPoolId': None, 'accountId': None,
'cognitoIdentityId': None, 'caller': None, 'sourceIp': '54.222.45.2',
'principalOrgId': None, 'accessKey': None, 'cognitoAuthenticationType': None,
'cognitoAuthenticationProvider': None, 'userArn': None, 'userAgent':
'curl/7.64.1', 'user': None}, 'domainName': 'mlexr1c5dj.execute-api.cn-north-
1.amazonaws.com.cn', 'apiId': 'mlexr1c5dj'}, 'body': '{\\"key1\\": \\"001\\"}',
'isBase64Encoded': False}
```

2. 创建apigw

在该lambda上创建trigger，选择stage进行发布，获得APIGW的URI



default Stage Editor

Invoke URL: https://mlexr1c5dj.execute-api.cn-north-1.amazonaws.com.cn/default

Cache Settings

Enable API cache ☐

Default Method Throttling