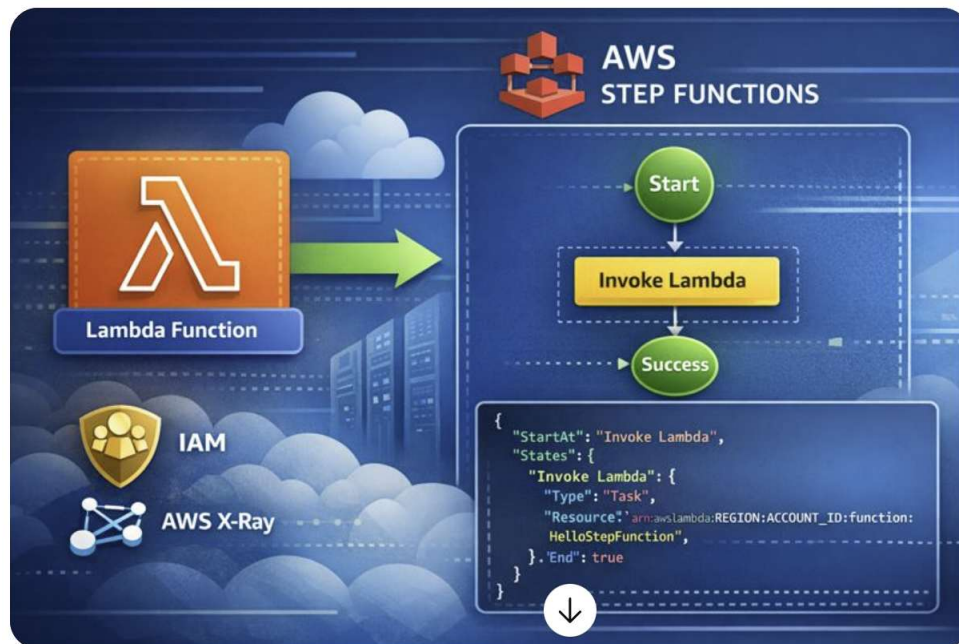


AWS Hands-on Lab | Lambda (Python) & Step Functions



Réalisation d'un lab très simple pour comprendre le fonctionnement des **AWS Step Functions** et leur intégration avec **AWS Lambda**.

Objectif :

- Créer une fonction Lambda minimale, codée en Python
- Orchestrer son exécution avec une **State Machine (Standard)**
- Comprendre le workflow visuel, le code JSON généré automatiquement et les rôles IAM associés
- Tester l'exécution et analyser le résultat

Services utilisés :

- AWS Lambda (Python 3.11)
- AWS Step Functions (Standard)
- IAM
- AWS X-Ray (permissions)

Sommaire :

1 : Créer une fonction Lambda simple	2
2 : Créer un "state machine" (Step function)	2

3 Créer la Step Function.....	4
4 : Tester la Step Function.....	6

1 : Créer une fonction Lambda simple

Dans la console AWS, Aller sur AWS Lambda → Create Function → Author from scratch.

Nom : HelloStepFunction.

Runtime : Python 3.11 (ou Node.js si tu préfères).

Dans le code Lambda, mettons quelque chose de minimal :

```
def lambda_handler(event, context):  
    return {  
        "statusCode": 200,  
        "message": "Hello from Step Function!"  
    }
```

Déployer la fonction.

≡

Lambda

> Functions

Functions (2)

Last fetched 1/16/2026, 8:1

🔍

Search by attributes or search by keyword

<input type="checkbox"/>	Function name	▼	Description	▼	Package type	▼	Runtime
<input type="checkbox"/>	Hellostepfunction		-		Zip		Python 3.14

2 : Créer un “state machine” (Step function)

Dans la console AWS, Aller sur Step Functions → Create state machine → Standard et “create from blank”

Nom : SimpleLab.

Choisir : Design your workflow visually.

☰

[Step Functions](#)

> State machines

State machines (2)

🔍

Search for state machines

Any type

▼

View execution counts

🕒

View details

Edit

Copy to new

Delete

Create state machine

	Name	Type	Creation date	Status
<input type="radio"/>	SimpleLab	Standard	Jan 16, 2026, 20:51:48 (UTC-05:00)	Active
<input type="radio"/>	MyState	Standard	Jan 16, 2026, 20:43:08 (UTC-05:00)	Active

Create state machine

Create from blank

Create your own workflow from scratch.

Create from template

Choose a workflow template that matches your use case.

State machine name

State machine name can be edited in the Config section of Workflow Studio before creating your state machine. After creation, the name cannot be changed.

MyStateMachine

Generate

Must be 1-80 characters. Can use alphanumeric characters, dashes, and underscores.

State machine type

State machine type can be edited in the Config section of Workflow Studio before creating your state machine. After creation, the type cannot be changed.

Standard

Durable workflows for ETL, ML, e-commerce and automation. Standard workflows can run for up to 1 year, and history is stored in Step Functions for auditing and playback. Supported by a feature-rich console debugger. Recommended for new users. [Learn more](#)

Express

Low cost, high scale workflows for streaming data processing and microservice APIs. Express workflows can run for up to 5 minutes, and history can be streamed to CloudWatch Logs. [Learn more](#)

Exit

Continue

Cliquer sur ‘continuer’

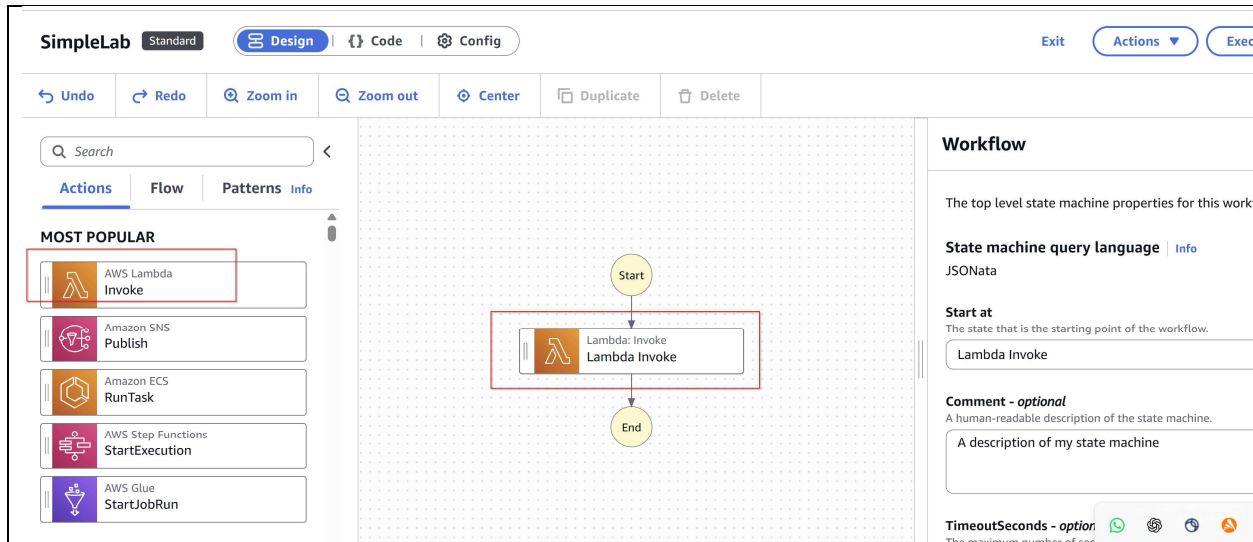
Ajouter un état Task :

Resource : votre fonction Lambda HelloStepFunction.

Pour cela faire glisser la fonction du pannel de gauche, intitulée : ‘AWS Lambda invoke’ vers le graphique dans le champs : ‘drag first state here’. Le ‘first state’ correspond au premier ‘état’ qui est : ‘Lambda Invoke’.

The screenshot shows the AWS Step Functions console interface. At the top, there's a header with 'test4' and tabs for 'Design', 'Code', and 'Config'. Below the header is a toolbar with 'Undo', 'Redo', 'Zoom in', 'Zoom out', 'Center', 'Duplicate', and 'Delete' buttons. The main workspace is a grid where a workflow is being designed. The workflow starts with a yellow circle labeled 'Start', followed by a dashed box labeled 'Drag first state here', and ends with a yellow circle labeled 'End'. On the left, there's a sidebar with a search bar and tabs for 'Actions', 'Flow', 'Patterns', and 'Info'. Under 'Actions', there's a list of 'MOST POPULAR' actions: 'AWS Lambda Invoke', 'Amazon SNS Publish', 'Amazon ECS RunTask', 'AWS Step Functions StartExecution', and 'AWS Glue StartJobRun'. On the right, there's a 'Workflow' panel showing details about the state machine, including its name and type.

Cela remplit automatiquement le panel, intitulé ‘workflow sur la droite’ avec les données de la fonction lambda. Voir le graphique ci-dessous.



Exemple de définition JSON très simple :

```
{
  "Comment": "A simple Step Function",
  "StartAt": "HelloTask",
  "States": {
    "HelloTask": {
      "Type": "Task",
      "Resource": "arn:aws:lambda:REGION:ACCOUNT_ID:function:HelloStepFunction",
      "End": true
    }
  }
}
```

On n'utilisera pas ce code Json , mais on utilisera a la place le code Json , décrit au paragraphe suivant.

3 Créer la Step Function.

L'outil AWS Step function workflow , génère du code Json automatiquement, que l'on peut voir dans la fenêtre "code". Le code Json appelle (invoke) la fonction lambda créée précédemment.

SimpleLab
Design
Code
Config

Undo
Redo
Format
Copy
Commands
View docs
Zoom in
Zoom out
Center

```

1 {
2   "Comment": "A description of my state machine",
3   "StartAt": "Lambda Invoke",
4   "States": {
5     "Lambda Invoke": {
6       "Type": "Task",
7       "Resource": "arn:aws:states:::lambda:invoke",
8       "Output": "{% $states.result.Payload %}",
9       "Arguments": {
10        "FunctionName": "arn:aws:lambda:ca-central-1:123456789012:func:HelloStepFunction",
11        "Payload": "{% $states.input %}"
12      },
13      "Retry": [
14        {
15          "ErrorEquals": [
16            "Lambda.ServiceException",
17            "Lambda.AWSLambdaException",
18            "Lambda.SdkClientException",
19            "Lambda.TooManyRequestsException"
20          ],
21          "IntervalSeconds": 1,
22          "MaxAttempts": 3,
23          "BackoffRate": 2,
24          "JitterStrategy": "FULL"
25        }
26      ],
27      "End": true
28    }
29  }
30 }

```

```

graph TD
    Start((Start)) --> LambdaInvoke[Lambda: Invoke  
Lambda Invoke]
    LambdaInvoke --> End((End))

```

Dans l'URL :

"arn:aws:lambda:ca-central-1: **ACCOUNT_ID**:function:HelloStepFunction:\$LATEST"

A la place de **ACCOUNT_ID** , on aura l'ID de votre compte AWS

Cliquer sur le bouton ‘create’

Lorsqu'on clique sur ‘create’ , une autre fenêtre s’affiche qui demande confirmation pour ajouter des rôles et permissions pour les rôles, afin que AWS Step Function puisse exécuter la fonction Lambda et aussi avoir le droit d'utiliser le service AWS X-Ray pur le debugging.

Confirm role creation

ⓘ An execution role will be created with full permissions.
A new execution role named *StepFunctions-SimpleLab-role-ch8uti6wg* will be created. All required permissions for the actions specified in your state machine will be auto-generated.

Role permissions

Service	Action(s)	Status	Documentation links
AWS Lambda	lambda:Invoke	✔ Policy will be generated to perform the action for specified Lambda resources only	Call Lambda with Step Functions Lambda policies for Step Functions
AWS X-Ray	xray:PutTraceSegments xray:PutTelemetryRecords xray:GetSamplingRules xray:GetSamplingTargets	✔ Policies will be generated for X-Ray tracing	X-Ray policies for Step Functions

Cancel
View role configuration
Confirm

Cliquer sur ‘confirmation ‘

On peut aussi cliquer sur le bouton ‘view rôle configuration’ afin de voir le détail des rôles.

4 : Tester la Step Function

Cliquer sur Start execution.

La fenetre suivante s’Affiche :

The screenshot shows the 'Start execution' dialog box. It has a title bar 'Start execution' with a close button. Below the title, there is a 'Name' field containing the ID '6c4a8c56-0dc9-4d5e-a17d-83b5898e7605'. A note below the field states: 'Must be 1-80 characters. Can use alphanumeric characters, dashes, or underscores.' Underneath is the 'Input - optional' section with the instruction 'Enter input values for this execution in JSON format'. There are three buttons: 'Format JSON', 'Export', and 'Import'. A text area contains a JSON snippet:

```
1 {  
  "Comment": "Insert your JSON here"  
}
```

 At the bottom left, there is a link 'Start execution with latest revision' and a checkbox 'Open in a new browser tab'. At the bottom right, there are two buttons: 'Cancel' and 'Start execution'.

Pour ce premier test , laisser entrée : {} (vide).

Résultat attendu :

```
{  
  "statusCode": 200,  
  "message": "Hello from Step Function!"  
}
```

The screenshot shows the AWS Step Functions console. The breadcrumb navigation at the top is 'Step Functions > State machines > SimpleLab > Execution: 6c4a8c56-0dc9-4d5e-a17d-83b5898e7605'. The main heading is 'Execution: 6c4a8c56-0dc9-4d5e-a17d-83b5898e7605' with an 'Edit state machine' button. Below the heading are three tabs: 'Details', 'Execution input and output' (which is selected), and 'Definition'. Under the 'Execution input and output' tab, there are two sections: 'State input' and 'State output'. The 'State input' section shows a table with one row:

1	{}
---	----

. The 'State output' section shows a table with one row:

1	{ "statusCode": 200, "message": "Hello from Step Function!" }
---	--

. At the bottom, there are two buttons: 'Graph view' and 'Table view'.

Vérifier l’état des boutons sur le graphique :

Le status “vert” indique le succès de l’opération

