QUICK LAB 3

NO INFRASTRUCTURE, JUST CODE. SEE THE SIMPLICITY OF SERVERLESS.

Create, build, and run three serverless functions as a sequence

INTRODUCTION

This lab walks you through the steps required to create, build and run a Serverless application using IBM Cloud Functions. **Serverless computing** refers to a model where the existence of servers is entirely abstracted away. Even though servers exist, developers are relieved from the need to care about their operation. They are relieved from the need to worry about low-level infrastructural and operational details such as scalability, high-availability, infrastructure-security, and other details. Serverless computing is essentially about reducing maintenance efforts to allow developers to quickly focus on developing code that adds value.

Serverless computing simplifies developing cloud-native applications, especially microservice-oriented solutions that decompose complex applications into small and independent modules that can be easily exchanged. Some promising solutions like Apache OpenWhisk have recently emerged that ease development approaches used in the serverless model. **IBM Cloud Functions** is a Function-as-a-Service (FaaS) platform on IBM Cloud, built using the Apache OpenWhisk open source project, that allows you to execute code in response to an event.

With a granular pricing model that works at any scale, you get exactly the resources you need – not more, not less – and you are charged only for running code.

IBM Cloud Functions provides:

- Support for multiple languages, including JavaScript, Python, Swift, and Java
- Support for running custom logic through Docker containers
- The ability to focus more on value-adding business logic, and less on low-level infrastructural and operational details.

- The ability to easily chain together microservices to form workflows via composition.

In this lab, you'll create three functions for doing some data manipulation of a string, and then tie them together in a sequence.

Sequence of actions
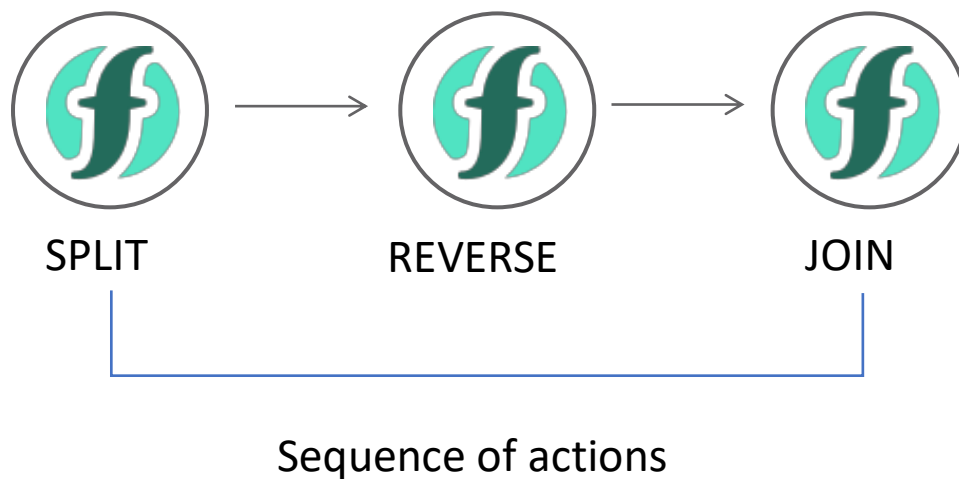
PREREQUISITES FOR THIS LAB

- You will need an IBM Cloud Account. Either use your existing account, or create a new account by accessing the following link: https://ibm.biz/Bd2Sfe

CREATE THE FIRST ACTION

There are two main options to get started with Cloud Functions. Both allow you to work with Cloud Function's basic entities by creating, updating, and deleting actions, triggers, rules and sequences.

The CLI (command line interface) allows you to perform these basic operations from your shell. The IBM Cloud Functions UI (user interface), allows you to perform

the same operations from your browser. During this lab we will use the UI to learn how to work with Cloud Functions.

1. Select the hamburger menu in the IBM Cloud header.

   ☰ IBM **Cloud**

   # Dashboard

   Resource summary
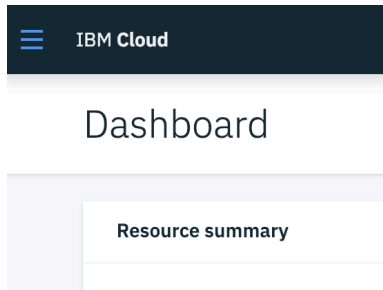
2. Then click on **Functions** to access the IBM Cloud Functions development experience on IBM Cloud.

   DevOps

   Finance

   Functions

   Integrate
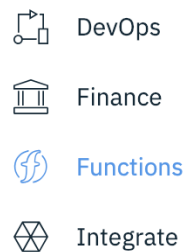
3. The Cloud Functions UI is comprised of the following sections in the left hand side menu bar.

   Functions

   Getting Started ⌄

   Actions

   Triggers

   Monitor

   Logs ↗

   APIs
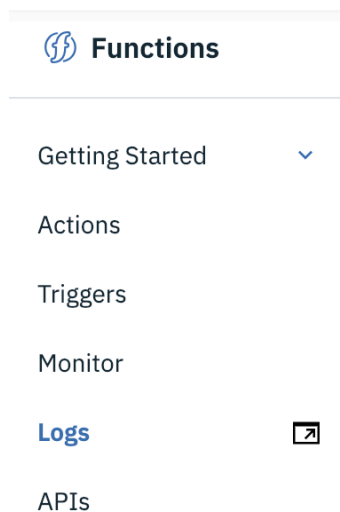
a. Actions – The actions section lists all actions you have created prior. An action is a small piece of code that can be explicitly invoked or set to automatically run in response to an event.

b. Triggers – A trigger is a declaration that you want to react to a certain type of event, whether from a user or by an event source. A trigger can be fired or activated. Triggers can be associated with actions, so that when the trigger is fired the action is run.

c. Monitor – This section shows you information about your actions and their activity, including an activity summary and timeline.

d. Logs – The logs section takes you to the IBM Cloud Logging service, which provides you with the ability to collect, analyze, and build dashboards for your logs.

e. APIs – The APIs section allows you to set up an API Gateway and API management for IBM Cloud Functions.

4. Start creating your first action by selecting the **Start Creating** button in the center of the UI, which opens the Create page. Then select the **Create Action** button.

Create

Quickstart Templates
Get started quickly using one of the Templates. A number of use cases are available, from a hello world action to invoking functions from Cloudant or Message Hub events.

Create Action
Actions contain your function code and are invoked by events or REST API calls.

Create Sequence
Sequences invoke Actions in a linear order, passing parameters from one to the next.

Create Trigger
Triggers receive events from outside IBM Cloud Functions and invoke all connected Actions.

Install Packages
Installing Packages installs reusable Actions into your namespace.

5. Specify the **Action Name**, **split**, by entering it into the text field, and then select Node.js 10 as the runtime. Leave everything else as-is and click the **Create** button at the bottom of the screen.

Action Name

split

Enclosing Package ⓘ

(Default Package)   ▾   Create Package

Runtime ⓘ

Node.js 10   ▾

Looking for Java, .NET or Docker? **Java**, **.NET** and **Docker** Actions can be created with the **CLI**

Cancel   Previous   Create

6. This opens a cloud-based code editor that you can use to create and extend your actions. There should already be some hello world code in the action.

7. Replace the code with the following code for the action:

```
function main(params) {

  var text = params.text || ""

  var words = text.split(' ')

  return { words: words }

}
```

8. Click **Save.** You can test that this code works by clicking **Change Input**, and providing the following json as the input parameter to the action:

   `{"text":"these are just some words"}`

9. Click **Invoke**, and you should see that the input words have been split into an array.


```
Activations              Collapse ⬆    Clear  🗑

∨  ✔  split        52 ms        1/17/2019, 23:02:51

   Activation ID:
   d62ea47e46104d06aea47e46104d0648

   Results:

   {
     "words": [
       "these",
       "are",
       "just",
       "some",
       "words"
     ]
   }

   Logs:

   []
```

CREATE THE SECOND AND THIRD ACTION

The first action split the words apart by using space as a delimiter. The second action will reverse the text within each word.

1. Click **/ Actions /** in the breadcrumb in the upper left of the page to go to the dashboard.


```
ns   /   Actions   /   spli
```

2. Click **Create** to open the Create page, and then select **Create Action.**

Create

Quickstart Templates
Get started quickly using one of the Templates. A number of use cases are available, from a hello world action to invoking functions from Cloudant or Message Hub events.

Create Action
Actions contain your function code and are invoked by events or REST API calls.

Create Sequence
Sequences invoke Actions in a linear order, passing parameters from one to the next.

Create Trigger
Triggers receive events from outside IBM Cloud Functions and invoke all connected Actions.

Install Packages
Installing Packages installs reusable Actions into your namespace.

3. Again, specify your action name. This time, **reverse.** Click **Create.**

Create Action

Actions contain your function code and are invoked by events or REST API calls.

Learn more about Actions

Learn more about Packages

Action Name

reverse

Enclosing Package ⓘ

(Default Package) ▾   **Create Package**

Runtime ⓘ

Node.js 10 ▾

Looking for Java, .NET or Docker? **Java**, **.NET** and **Docker** Actions can be created with the **CLI**
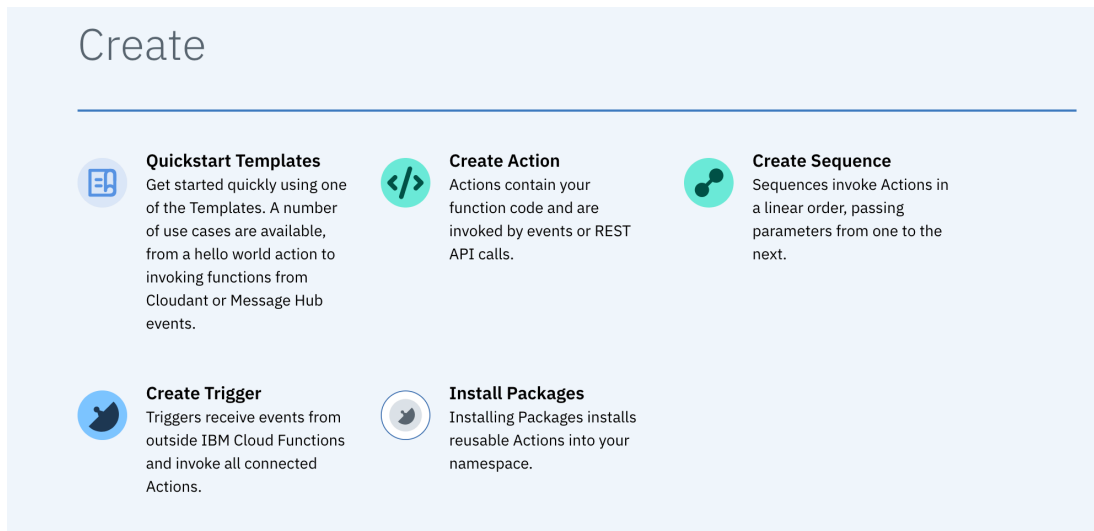
4. Copy, and then paste the action code into the code editor, and click **Save:**

```
function main(params) {

  var words = params.words || []

  var reversed = words.map(word => word.split("").reverse().join(""))

  return { words: reversed }

}
```

5. Let's create the third action in our sequence. Click **/ Actions /** in the breadcrumb in the upper left of the page to go to the dashboard.

6. Click **Create** to open the Create page, and then select **Create Action.**

7. Again, specify your action name. This time, **join.** The join action will rejoin the words into a string, with spaces in-between each reversed word. Click **Create.**

8. Copy, and then paste your action code into the code editor, and click **Save:**

```javascript
function main(params) {

  var words = params.words || []

  var text = words.join(' ')

  return { text: text }

}
```

CREATE THE SEQUENCE TO TIE THE ACTIONS TOGETHER

Sequences are strings of actions. The output of one action in a sequence would be the input for the next action in the sequence, which would provide an output that can be used in the next action and so on. Sequences are treated by IBM Cloud Functions as a special kind of action, which means a sequence behaves just like a normal action. Sequences can be created, invoked, and managed just as an action would be.

Sequences can be useful as an alternative to one action manually invoking another. If an action manually calls another, the system will charge you for both actions, since the first action is waiting on the second to complete. In a sequence, one action runs at a time, and you are only charged for one action running at a time. Let's tie our actions into a sequence.

1. Click **Enclosing Sequences** in the left side menu, and then click **Add to Sequence.** We don't have a sequence created, so let's create a new one. We need to name it something, for example data-manipulation-sequence.

2. Once the sequence is named, click **Create & Add.**

Add To Sequence

Sequences invoke Actions in a linear order, passing parameters from one to the next.

Learn more about Sequences

| Create New | Select Existing (1) |

Action Name

data-manipulation-sequence

Enclosing Package ⓘ

(Default Package)   ▾   Create Package

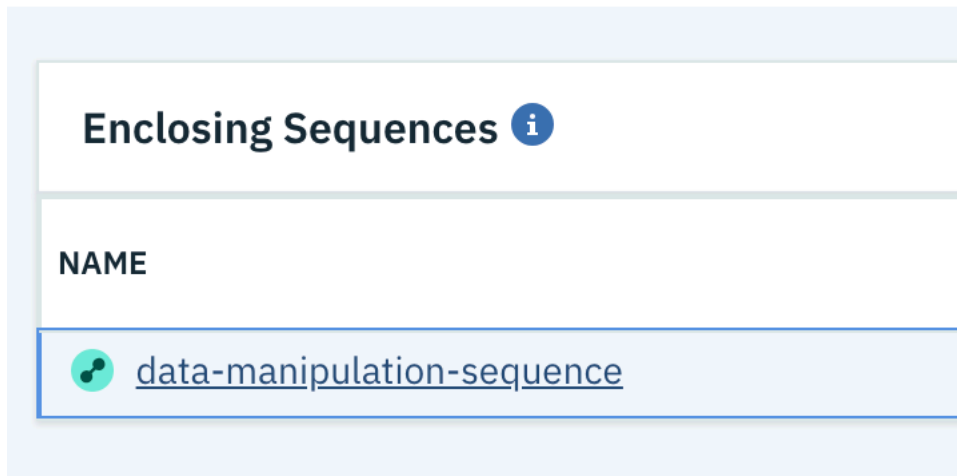**Runtime** ⓘ

Sequence   ▾

Cancel   Create & Add

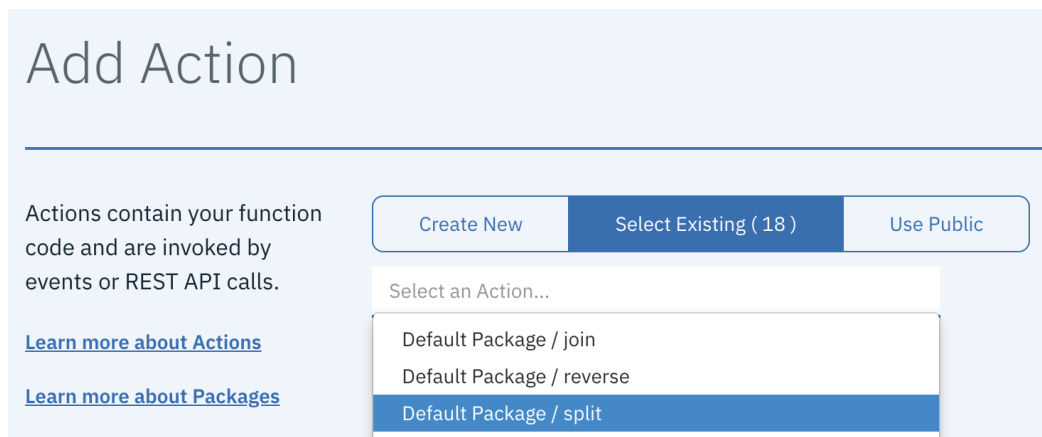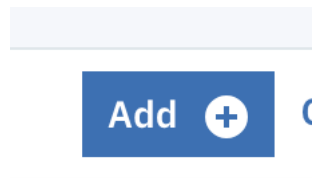3. Click the **sequence name** to further edit the sequence.

4. Click the **Add +** button to add more actions to this sequence.

5. Select the **Select Existing** option, and then find the **split** action in the drop down. Once selected, click **Add** in the bottom right.

6. Repeat the steps for **reverse.** Click the **Add +** button, choose **Select Existing**, and then find the **reverse** action in the drop down. Once selected, click **Add** in the bottom right.

7. Using the arrows, rearrange your sequence actions until they are in order: **split, reverse, join,** and then click **Save.**

Sequence Actions

ACTIONS	PACKAGE
⇕ split	Default Package
⇕ reverse	Default Package
⇕ join	Default Package

8. Let's try it out. We can provide the sequence with input, just like we can an action. Click **Change Input**, and then provide the following json, and then click **Apply**:

`{"text":"These are some words"}`

Click **Invoke,** and you should see the following output:

Activations

✓ data-manipulation-sequence

Activation ID:
224e09c566974d968e09c566970d96d8

Results:
```
{
  "text": "esehT era emos sdrow"
}
```

Logs:
```
[
  "b06c80bbd9364ff4ac80bbd936fff435",
  "04a38e934ad54bb5a38e934ad51bb563",
  "edafe151326247c6afe151326227c698"
]
```

Did you notice that the logs include 3 numbers? Those are activation IDs for each of the individual actions within the sequence. Activation IDs are a unique identifier that actions generate when they are run. These can be used to debug in case something goes wrong with a particular action in the sequence.

9. Try another input, by clicking **Change Input**, and then **Invoke.**

```
{"text":" A malayalam pup did kayak"}
```

CONCLUSION

Congratulations! You have completed this lab. You have successfully created three Node.js actions, and tied them together into a sequence to do some data manipulation on strings – all from within a browser! Feel free to reach out should you have any questions.