Using Separate Namespaces to Test IBM Cloud Functions

Just like any other programming, it's important to test your Cloud Functions before they are deployed into a live environment. This short document provides some information on how to best go about testing Cloud Functions on IBM Cloud.

Cloud Functions are deployed into namespaces. Think of a Namespace as a separate area into which you can deploy and run your functions, as well as control who has access to them.

A good practice is to have separate Namespaces for the different stages of your code. For example, you might have the following Namespaces:

Developer-Private – This would be a 'sandpit' Namespace, perhaps one per developer
Developer-all - This would be a true development Namespace, where code exists in the latest 'development' stage
Staging – this would be the Staging / System Test Namespace, where code is in a stable, test environment
Production – this would be the stable Production Namespace

The idea here is that code would be 'promoted' up the stack. For example, a developer could use their own 'private' Namespace to develop new Cloud Functions and privately test them, before 'releasing' into the general developer space, where they can then be more widely tested alongside other functions. When a function is deemed ready for more formal testing, it can be deployed into the System-Test namespace and once past that point, it can be promoted to Production.

Of course, this can be as formal or informal as necessary, but it allows you to develop and adapt code without affecting the main production code-base, all backed up by toolchains and code repositories such as Git.

Creating Namespaces
There are two ways to create a namespace – either through the IBM Cloud console or via the *ibmcloud* CLI tool.

To create via the console:

- From the Cloud Functions console namespace menu, select **Create Namespace**.
- Enter a display name for the namespace and a short description, such as the actions or packages that you plan to create in this namespace.
- Choose the resource group where you want to create the namespace and a location to deploy the namespace resource.
- Click **Create**.

And using the CLI:

- First, select the IAM resource group that the Namespace will reside in:
  `ibmcloud target -g <resource_group_name>`

- Then, create the namespace:
  `ibmcloud fn namespace create <namespace_name> [--description <"description of your namespace">]`

We'll concentrate on using the CLI for the remainder of this paper.

For example, you might want to create the following two namespaces:

```
ibmcloud fn namespace create staging

ibmcloud fn namespace create production
```

To see what namespaces you already have and are available, run this command:

```
ibmcloud fn namespace list
```

Targeting a Namespace
Having several Namespaces means that developers may need to move between Namesapaces when developing, deploying and running their code. To do this, they will need to run the following CLI command:

```
ibmcloud fn property set --namespace <namespace_name>
```

So, to target the namespace named 'staging' you would enter:

```
ibmcloud fn property set --namespace staging
```

Deploying or Undeploying Code
Once a developer is targeted at the correct namespace, they can then deploy or undeploy the code that they have been working on into that namespace. As a simple example, they can run this command:

```
ibmcloud fn deploy --manifest folder/manifest.yaml
```

This will deploy the function into the Namespace, where it can then be tested.

If the testing fails or the code needs to be removed from the namespace for some other reason, then the code can be simply undeployed as follows:

```
ibmcloud fn undeploy --manifest folder/manifest.yaml
```

Testing with Local Log Output
To see the logs generated by testing, a developer can start activation polling. This will display the ActivationID and and further details can be seen via an `'activation get':`

```
ibmcloud fn activation poll
```

Then, invoke an action in another terminal window, for example:

```
ibmcloud fn action invoke /DoctorCloud/samples/helloWorld
--param payload Bob
```

You might then see the following activation log:

```
   Activation: helloWorld (7331f9b9e2044d85afd219b12c0f1491)
2016-02-11T16:46:56.842065025Z stdout: hello bob!
```

You can then see further details of this log via:

```
ibmcloud fn activation get <activation_ID>
```

i.e.

```
ibmcloud fn activation get 7331f9b9e2044d85afd219b12c0f1491
```

Once the code is tested and ready to be deployed into the next Namespace, all the developer needs to do is issue the *ibmcloud fn property set –namespace <namespace>* command to point at the correct namespace and then issue the *ibmcloud fn deploy –manifest folder/manifest.yaml* command to deploy the code.

More information
Thanks for reading this short guide. For more information, see the IBM Cloud documentation, including:

Installing the CLI and Plug-in: https://cloud.ibm.com/docs/openwhisk?topic=cloud-functions-cli_install

Managing Namespaces: https://cloud.ibm.com/docs/openwhisk?topic=cloud-functions-namespaces

Testing Serverless Apps: https://cloud.ibm.com/docs/openwhisk?topic=cloud-functions-test

Cloud Functions CLI: https://cloud.ibm.com/docs/openwhisk?topic=cloud-functions-cli-plugin-functions-cli