Kitura Workshop Lab 1

Marek Sadowski, mwsadows@us.ibm.com

Lennart Frantzell alf@us.ibm.com

Date July 29, 2017

Introduction. 3

Phase 1: the **Swift sandbox**..3

Phase 2: Running Swift locally or in the Bluemix cloud...5

Step 1. 5

Step 2. 8

Appendix: 11

Introduction.

This Lab will take you on a brief journey with three stops:

1) It will start by introducing you to the Cloud-based Swift Sandbox, where you effortlessly will be able to write and test Swift code.

2) It will then teach you how to deploy a local Kitura server on the Mac.

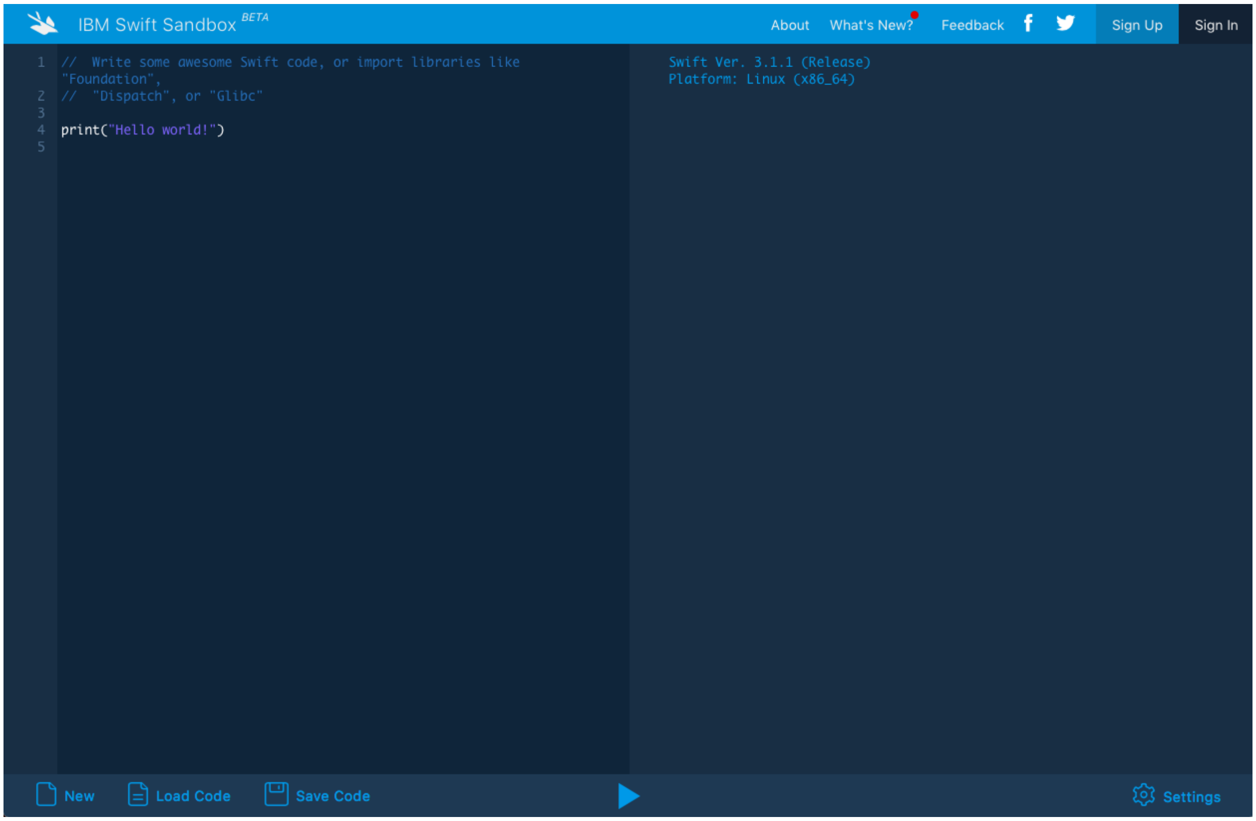
3) And finally, you'll learn how to run a Kitura server with Swift in the Cloud.

So, let's get started!

Phase 1: the **Swift sandbox**

https://swift.sandbox.bluemix.net/

After this Lab you will be able to test Swift language code snippets in Swift Sandbox, you will know how to deploy local Kitura server on MAC, and run the Kitura server side swift on a Cloud.



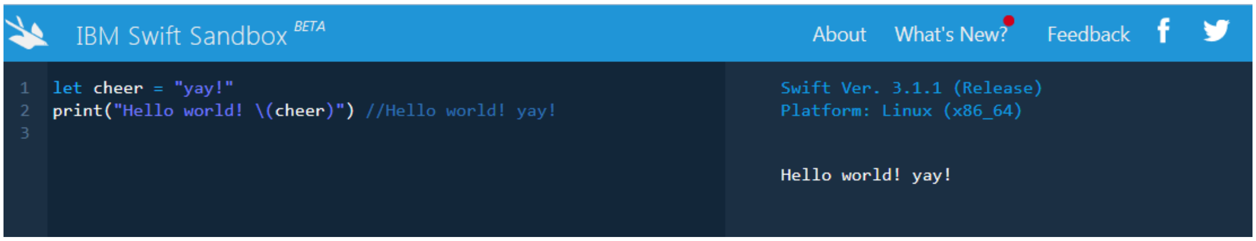
```
//  Write some awesome Swift code, or import libraries like
"Foundation",
//  "Dispatch", or "Glibc"

print("Hello world!")
```

```
Swift Ver. 3.1.1 (Release)
Platform: Linux (x86_64)
```

Enter the following lines of the code:

let cheer = "yay!"
print("Hello world! \(cheer)") //Hello world! yay!

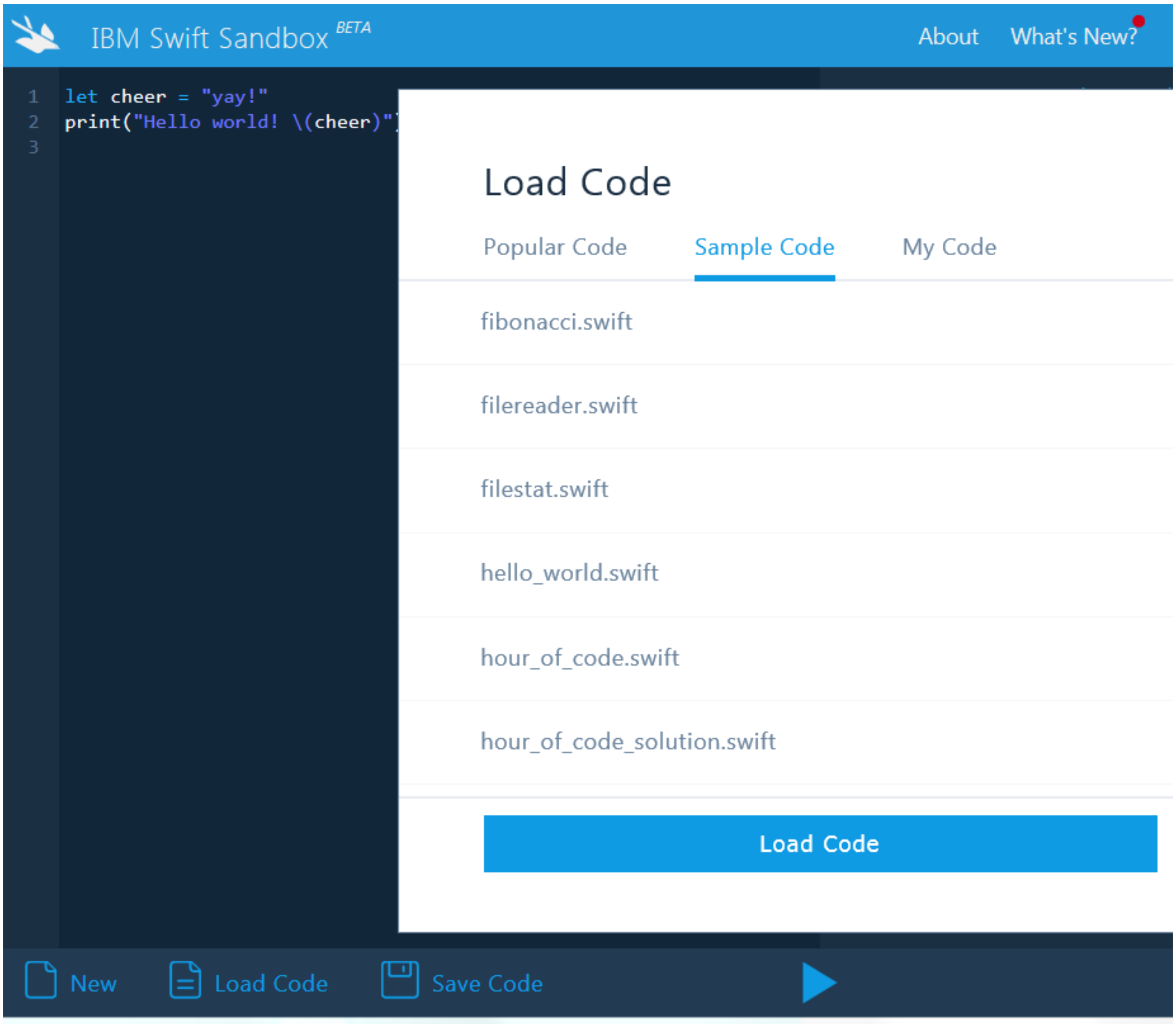
Press play button in the middle of the action bar at the bottom of the page.
You should see the result on the right side of the screen:
Hello world! yay!



```
let cheer = "yay!"
print("Hello world! \(cheer)") //Hello world! yay!
```

```
Swift Ver. 3.1.1 (Release)
Platform: Linux (x86_64)


Hello world! yay!
```

Try to **Load code** (see the menu on the bottom - try *fibonacci* number search), and update the code to print a different number
print(fibonacci(12)) //144 (choose number smaller than 42)



```
let cheer = "yay!"
print("Hello world! \(cheer)"
```

Phase 2: Running Swift locally or in the Bluemix cloud

Visit page: kitura.io
Either try it locally (if you have the right environment - ie. MacOS with XCode 8.3 - follow the next step 1, otherwise immediately try to run in the cloud step 2.

Step 1.

http://www.kitura.io/en/starter/settingup.html - check your XCode 8.1
open terminal and write the command:

```
$ mkdir myFirstKituraProject
$ cd myFirstKituraProject/
$ swift package init --type executable
```

The last command creates the directories and the following files:

```
myFirstKituraProject
├── Package.swift
├── Sources
│       └── main.swift
└── Tests
```

Use your favorite editor to change the contents of the Package.swift:

```
import PackageDescription
let package = Package(
   name: "myFirstKituraProject",
   dependencies: [
      .Package(url: "https://github.com/IBM-Swift/Kitura.git", majorVersion: 1, minor: 7)
   ])
```

Change the file Sources/main.swift

```
import Kitura
// Create a new routerlet router = Router()
// Handle HTTP GET requests to /
router.get("/") {
   request, response, next in
   response.send("Hello, World!")
   next()}
// Add an HTTP server and connect it to the router
Kitura.addHTTPServer(onPort: 8080, with: router)
// Start the Kitura runloop (this call never returns)
Kitura.run()
```

Compile the application:

From the root of the application build the project

```
$ swift build
```

The command would download the repository and linked libraries.

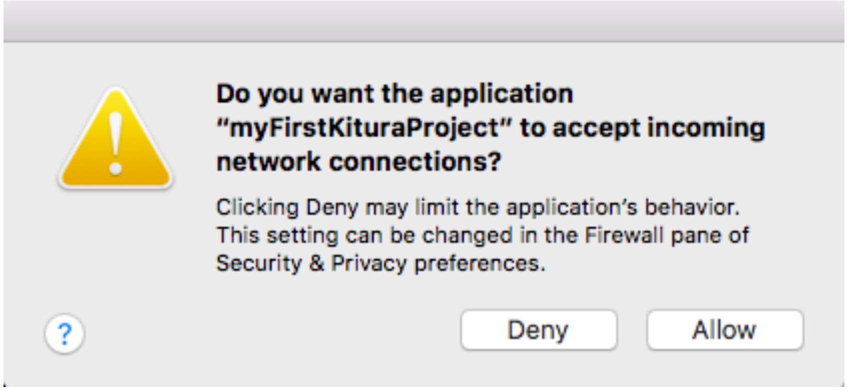
```
Fetching https://github.com/IBM-Swift/Kitura.git
Fetching https://github.com/IBM-Swift/Kitura-net.git
Fetching https://github.com/IBM-Swift/SwiftyJSON.git
Fetching https://github.com/IBM-Swift/Kitura-TemplateEngine.git
Fetching https://github.com/IBM-Swift/LoggerAPI.git
Fetching https://github.com/IBM-Swift/BlueSocket.git
Fetching https://github.com/IBM-Swift/CCurl.git
Fetching https://github.com/IBM-Swift/BlueSSLService.git
Cloning https://github.com/IBM-Swift/BlueSocket.git
Resolving https://github.com/IBM-Swift/BlueSocket.git at 0.12.56
Cloning https://github.com/IBM-Swift/LoggerAPI.git
Resolving https://github.com/IBM-Swift/LoggerAPI.git at 1.7.0
Cloning https://github.com/IBM-Swift/SwiftyJSON.git
Resolving https://github.com/IBM-Swift/SwiftyJSON.git at 16.0.1
Cloning https://github.com/IBM-Swift/Kitura-TemplateEngine.git
Resolving https://github.com/IBM-Swift/Kitura-TemplateEngine.git at 1.7.1
Cloning https://github.com/IBM-Swift/Kitura.git
Resolving https://github.com/IBM-Swift/Kitura.git at 1.7.6
```

```
Cloning https://github.com/IBM-Swift/BlueSSLService.git
Resolving https://github.com/IBM-Swift/BlueSSLService.git at 0.12.45
Cloning https://github.com/IBM-Swift/Kitura-net.git
Resolving https://github.com/IBM-Swift/Kitura-net.git at 1.7.15
Cloning https://github.com/IBM-Swift/CCurl.git
Resolving https://github.com/IBM-Swift/CCurl.git at 0.4.0
2017-07-25 15:53:20.336 xcodebuild[23857:746845] [MT] DVTToolchain: Failed to load toolchain:
<DVTFilePath:0x7fd78ac48a40:'/Library/Developer/Toolchains/swift-DEVELOPMENT-SNAPSHOT-2016-05-09-a.xctoolchain'>:
Error Domain=DVTToolchainErrorDomain Code=6
"Info.plist:OverrideBuildSettings:{SWIFT_LINK_OBJC_RUNTIME,SWIFT_DISABLE_REQUIRED_ARCLITE,ENABLE_BITCODE} must
contain a string or array of strings" UserInfo={NSFilePath=/Library/Developer/Toolchains/swift-DEVELOPMENT-SNAPSHOT-
2016-05-09-a.xctoolchain,
NSLocalizedDescription=Info.plist:OverrideBuildSettings:{SWIFT_LINK_OBJC_RUNTIME,SWIFT_DISABLE_REQUIRED_ARCLITE,EN
ABLE_BITCODE} must contain a string or array of strings}
2017-07-25 15:53:22.429 xcodebuild[23864:746885] [MT] DVTToolchain: Failed to load toolchain:
<DVTFilePath:0x7ff35171c170:'/Library/Developer/Toolchains/swift-DEVELOPMENT-SNAPSHOT-2016-05-09-a.xctoolchain'>:
Error Domain=DVTToolchainErrorDomain Code=6
"Info.plist:OverrideBuildSettings:{SWIFT_LINK_OBJC_RUNTIME,SWIFT_DISABLE_REQUIRED_ARCLITE,ENABLE_BITCODE} must
contain a string or array of strings" UserInfo={NSFilePath=/Library/Developer/Toolchains/swift-DEVELOPMENT-SNAPSHOT-
2016-05-09-a.xctoolchain,
NSLocalizedDescription=Info.plist:OverrideBuildSettings:{SWIFT_LINK_OBJC_RUNTIME,SWIFT_DISABLE_REQUIRED_ARCLITE,EN
ABLE_BITCODE} must contain a string or array of strings}
Compile Swift Module 'Socket' (3 sources)
Compile Swift Module 'SwiftyJSON' (2 sources)
Compile Swift Module 'LoggerAPI' (1 sources)
Compile Swift Module 'KituraTemplateEngine' (1 sources)
Compile Swift Module 'SSLService' (1 sources)
Compile CHTTPParser utils.c
Compile CHTTPParser http_parser.c
Linking CHTTPParser
Compile Swift Module 'KituraNet' (35 sources)
Compile Swift Module 'Kitura' (44 sources)
Compile Swift Module 'myFirstKituraProject' (1 sources)
Linking ./.build/debug/myFirstKituraProject
```

Finally you can run locally Kitura server:

```
$ .build/debug/myFirstKituraProject
```

Allow for accepting the connections when asked:



Do you want the application "myFirstKituraProject" to accept incoming network connections?

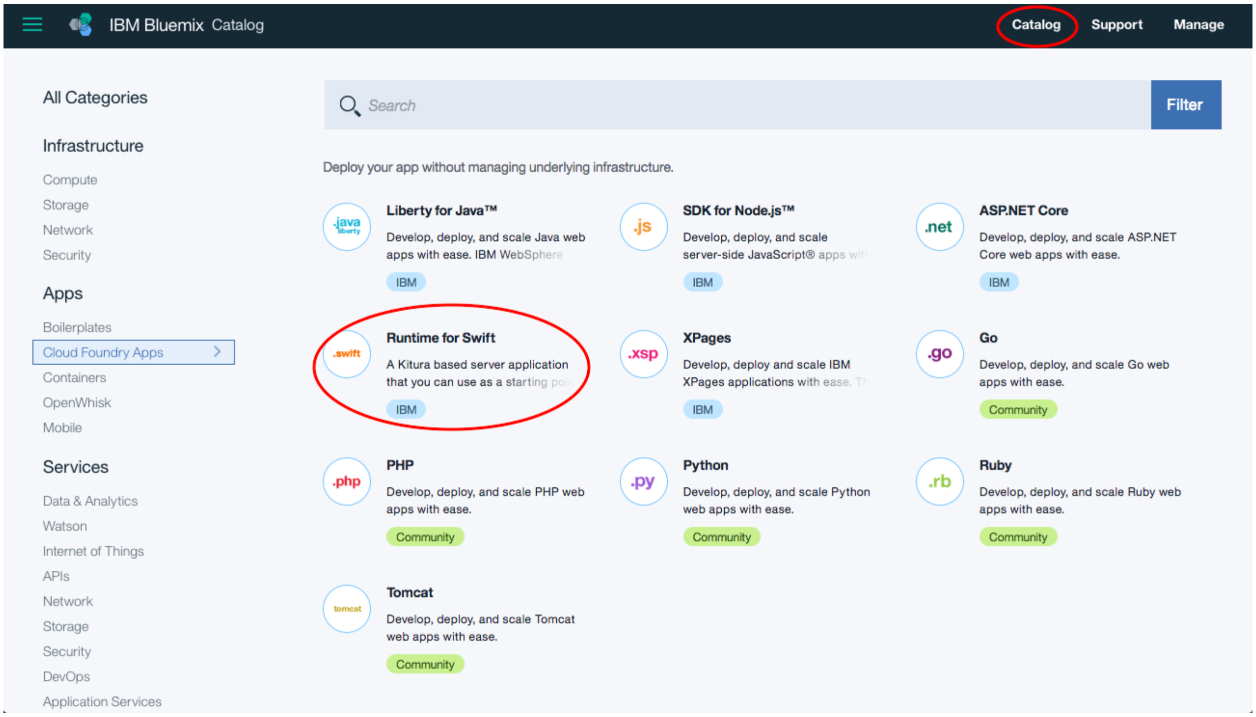
Clicking Deny may limit the application's behavior. This setting can be changed in the Firewall pane of Security & Privacy preferences.

Open your browser and point to the URL on which Kitura listens:
http://localhost:8080

To stop the server press control-C in the terminal window

Step 2.

Open Bluemix.net (sign up if you need to - it is free for first 30 days, it has also a free tier). From the catalog choose the Cloud Foundry Apps, and Runtime for Swift



Deploy your app without managing underlying infrastructure.

Liberty for Java™
Develop, deploy, and scale Java web apps with ease. IBM WebSphere
IBM

SDK for Node.js™
Develop, deploy, and scale server-side JavaScript® apps wi
IBM

ASP.NET Core
Develop, deploy, and scale ASP.NET Core web apps with ease.
IBM

Runtime for Swift
A Kitura based server application that you can use as a starting poi
IBM

XPages
Develop, deploy and scale IBM XPages applications with ease. Th
IBM

Go
Develop, deploy, and scale Go web apps with ease.
Community

PHP
Develop, deploy, and scale PHP web apps with ease.
Community

Python
Develop, deploy, and scale Python web apps with ease.
Community

Ruby
Develop, deploy, and scale Ruby web apps with ease.
Community

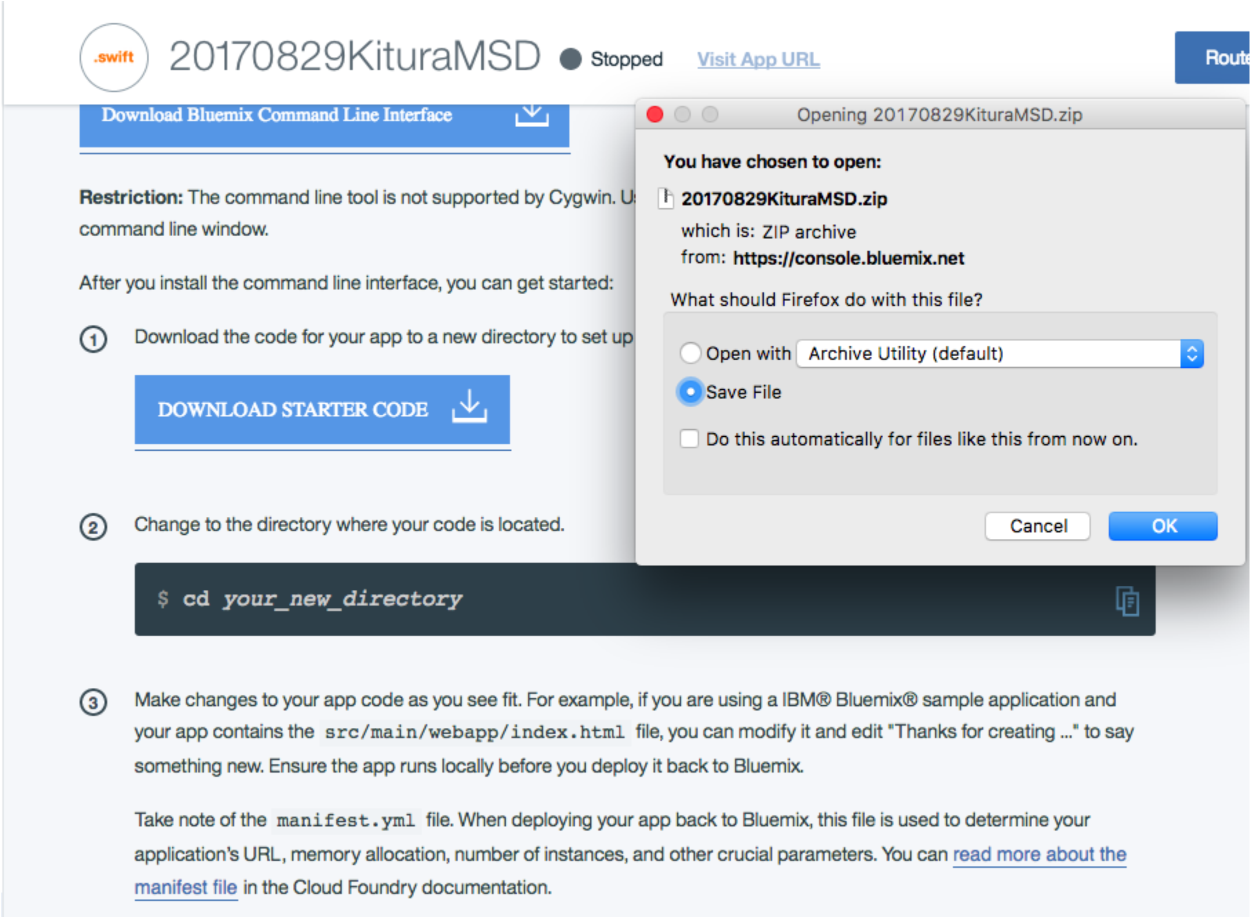
Tomcat
Develop, deploy, and scale Tomcat web apps with ease.
Community

Give the application a unique name (it is a public resource!)

And click create

You will modify the files locally and push them to the cloud with cf push command. The steps are the following:

1. setup environment (install CF/Bluemix command) and download the example from this repo



20170829KituraMSD ● Stopped Visit App URL

Download Bluemix Command Line Interface

Restriction: The command line tool is not supported by Cygwin. U
command line window.

After you install the command line interface, you can get started:

1. Download the code for your app to a new directory to set up

DOWNLOAD STARTER CODE

2. Change to the directory where your code is located.

```
$ cd your_new_directory
```

3. Make changes to your app code as you see fit. For example, if you are using a IBM® Bluemix® sample application and your app contains the `src/main/webapp/index.html` file, you can modify it and edit "Thanks for creating ..." to say something new. Ensure the app runs locally before you deploy it back to Bluemix.

Take note of the `manifest.yml` file. When deploying your app back to Bluemix, this file is used to determine your application's URL, memory allocation, number of instances, and other crucial parameters. You can read more about the manifest file in the Cloud Foundry documentation.

Opening 20170829KituraMSD.zip

You have chosen to open:

20170829KituraMSD.zip

which is: ZIP archive
from: **https://console.bluemix.net**

What should Firefox do with this file?

○ Open with Archive Utility (default)
● Save File
□ Do this automatically for files like this from now on.

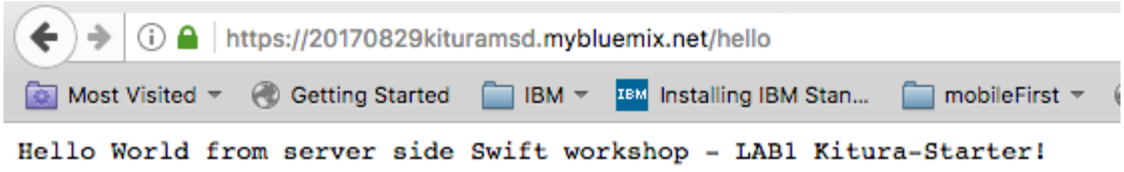
1. make changes in the downloaded app
- observe the contents of the manifest.yml - this is the core definition of the services binding of the cloud app (which services are bound to the app, what is the name, what are the parameters of the service container, etc).
- make changes to the app - update the contents of the file: Sources/Controller/Controller.swift
public func getHello
to the following code:

```
/**
  * Handler for getting a text/plain response.
  */
  public func getHello(request: RouterRequest, response: RouterResponse, next: @escaping () -> Void) throws {
    Log.debug("GET - /hello route handler...")
    response.headers["Content-Type"] = "text/plain; charset=utf-8"
    try response.status(.OK).send("Hello World from server side Swift workshop - LAB1 Kitura-Starter!").end()
  }
```

- follow the instructions to redeploy the app to the cloud

```
$ cf push
Using manifest file /Users/mareksadowski/Downloads/20170829KituraMSD(1)/manifest.yml

Updating app 20170829KituraMSD in org mwsadows@us.ibm.com / space dev as
mwsadows@us.ibm.com...
OK
………….
Uploading complete
Destroying container
Successfully destroyed container

0 of 1 instances running, 1 starting
1 of 1 instances running

App started


OK

App 20170829KituraMSD was started using this command `Kitura-Starter`

Showing health and status for app 20170829KituraMSD in org mwsadows@us.ibm.com / space dev
as mwsadows@us.ibm.com...
OK

requested state: started
instances: 1/1
usage: 256M x 1 instances
urls: 20170829KituraMSD.mybluemix.net
last uploaded: Fri Jul 28 17:58:27 UTC 2017
stack: cflinuxfs2
buildpack: swift_buildpack

     state     since                  cpu    memory      disk      details
#0   running   2017-07-28 11:01:37 AM   0.0%   0 of 256M   0 of 1G

Note: the full deploy messages appear in the Appendix.
```

test the changes at the given address: https://(your assigned address to Kitura).mybluemix.net/hello



```
Hello World from server side Swift workshop - LAB1 Kitura-Starter!
```

This is the end of the lab.

Appendix:

Messages you can expect when deploying to the cloud.

```
Using manifest file /Users/mareksadowski/Downloads/20170829KituraMSD(1)/manifest.yml

Updating app 20170829KituraMSD in org mwsadows@us.ibm.com / space dev as
mwsadows@us.ibm.com...
OK

Using route 20170829KituraMSD.mybluemix.net
Uploading 20170829KituraMSD...
Uploading app files from: /Users/mareksadowski/Downloads/20170829KituraMSD(1)
Uploading 30.6K, 21 files
Done uploading
OK

Starting app 20170829KituraMSD in org mwsadows@us.ibm.com / space dev as
mwsadows@us.ibm.com...
Downloading swift_buildpack...
Downloaded swift_buildpack
Creating container
Successfully created container
Downloading app package...
Downloaded app package (29.5K)
Downloading build artifacts cache...
Downloaded build artifacts cache (20.4M)
Staging...
-----> Buildpack version 2.0.7
-----> Default supported Swift version is 3.1.1
-----> Configure for apt-get installs...
-----> Copying deb files to installation folder...
-----> Writing profile script...
-----> No Aptfile found.
```

```
-----> Getting swift-3.1.1
      Cached swift-3.1.1
-----> Unpacking swift-3.1.1.tar.gz
-----> Getting clang-4.0.0
      Cached clang-4.0.0
-----> Unpacking clang-4.0.0.tar.xz
-----> .ssh directory and config file not found.
-----> Loading from cache:
-----> - .build
-----> Fetching Swift packages and parsing Package.swift files...
-----> Additional packages to download: libcurl4-openssl-dev openssl libssl-dev
-----> libcurl4-openssl-dev is already installed.
-----> openssl is already installed.
-----> libssl-dev is already installed.
-----> No additional packages to download.
-----> Skipping installation of App Management (debug)
-----> Installing system level dependencies...
-----> Building Package...
-----> Build config: release
      Compile Swift Module 'KituraTemplateEngine' (1 sources)
      Compile Swift Module 'SwiftyJSON' (2 sources)
      Compile Swift Module 'Signals' (1 sources)
      Compile Swift Module 'Socket' (3 sources)
      Compile Swift Module 'LoggerAPI' (1 sources)
      Compile Swift Module 'HeliumLogger' (2 sources)
      Compile Swift Module 'Configuration' (6 sources)
      Compile Swift Module 'Health' (3 sources)
      Compile Swift Module 'TestProgram' (1 sources)
      Compile Swift Module 'CloudFoundryEnv' (6 sources)
      Linking ./.build/release/TestProgram
      Compile Swift Module 'SSLService' (1 sources)
      Compile CHTTPParser utils.c
      Compile CHTTPParser http_parser.c
      Linking CHTTPParser
      Compile Swift Module 'KituraNet' (34 sources)
      Compile Swift Module 'CloudFoundryConfig' (2 sources)
      Compile Swift Module 'Kitura' (43 sources)
      Compile Swift Module 'CloudFoundryDeploymentTracker' (1 sources)
      Compile Swift Module 'Controller' (1 sources)
      Compile Swift Module 'Kitura_Starter' (1 sources)
      Linking ./.build/release/Kitura-Starter
-----> Copying dynamic libraries
-----> Copying binaries to 'bin'
-----> Clearing previous swift cache
-----> Saving cache (default):
-----> - .build
```

```
-----> Optimizing contents of cache folder...
No start command specified by buildpack or via Procfile.
App will not start unless a command is provided at runtime.
Exit status 0
Staging complete
Uploading droplet, build artifacts cache...
Uploading build artifacts cache...
Uploading droplet...
Uploaded build artifacts cache (20.4M)
Uploaded droplet (91.7M)
Uploading complete
Destroying container
Successfully destroyed container

0 of 1 instances running, 1 starting
1 of 1 instances running

App started


OK

App 20170829KituraMSD was started using this command `Kitura-Starter`

Showing health and status for app 20170829KituraMSD in org mwsadows@us.ibm.com / space dev
as mwsadows@us.ibm.com...
OK

requested state: started
instances: 1/1
usage: 256M x 1 instances
urls: 20170829KituraMSD.mybluemix.net
last uploaded: Fri Jul 28 17:58:27 UTC 2017
stack: cflinuxfs2
buildpack: swift_buildpack

     state     since                    cpu    memory      disk      details
#0   running   2017-07-28 11:01:37 AM   0.0%   0 of 256M   0 of 1G
```