Predix

Introduction to Cloud Foundry for Predix

Student Lab Guide

September 2015

GE Digital

Predix

Getting Started

This guide provides step-by-step instructions for lab exercises. Each lab corresponds to a topic covered in class and provides students with hands-on experience developing applications on the Predix platform.

Course Prerequisites:

- Request a Cloud Foundry account
- Request a GitHub account
- Install the most recent DevBox version

Start the VM in Oracle VirtualBox

- Login with:

  User name: **predix**

  Password: **predix**

> ***Note:*** All lab exercises will be completed in your DevBox.

Set Your Environment

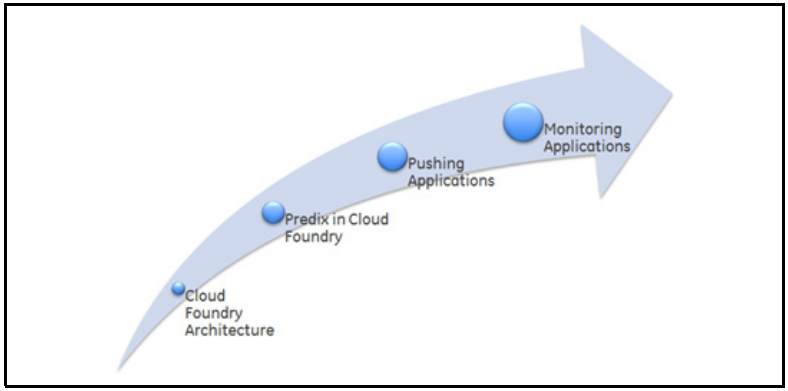
- Open a Terminal (icon on the Desktop)
- Run the **cloudfoundry_update.sh** script
  - A new **predix/PredixApps/predix/training_Labs/CloudFoundryLabs** directory structure is created

Lab 1: Getting Started with Cloud Foundry

Learning Objectives

By the end of the lab, you will be able to use the Cloud Foundry CLI (Command Line Interface) tool to:

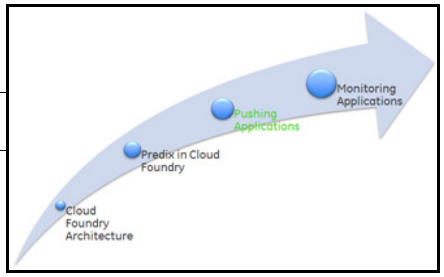
- Log into Cloud Foundry
- Add a service
- Monitor a service



Lab Exercises

Directions

Complete the exercises that follow.



Exercise 1: Creating a Service

Overview

In this exercise, you will log into Cloud Foundry and create a postgreSQL service instance.

This service will be bound to an application that you will deploy in a later lab. The application (microservice) needs an instance of this database in order to store and retrieve data.

The labs are designed to support novice as well as advanced users.

Steps

1. Log into Cloud Foundry (CF).

- Double-click the Terminal window icon on your desktop to open a Terminal window

- Run the following command

```
cf login
```

Tip: To run or execute a command in a Terminal window, type the command, and then press **Enter**. Note that the CLI is ***case-sensitive***.

Note: the system responds with your API endpoint location

- Enter your email address and press **Enter**
- At the Password prompt, enter **`P@ssword1`**
- Enter the number of the targeted space that your instructor provides

```
[predix@localhost spring-music]$ cf login
API endpoint: https://api.grc-apps.svc.ice.ge.com

Email> georgia.smith@ge.com

Password>
Authenticating...
OK

Targeted org predix-adoption

Select a space (or press enter to skip):
1. training1
2. training2
3. training3

Space>
```

The Terminal displays the API endpoint, user, and organization and space into which you have logged.

```
API endpoint:   https://api.grc-apps.svc.ice.ge.com (API version: 2.28.0)
User:           georgia.smith@ge.com
Org:            Predix-adoption
Space:          training2
```

2. Create a new postgreSQL service instance in your space.

- Run the command **`cf m`** to list the services available in the Cloud Foundry Marketplace. The services are listed, along with each purchase plan type and description

```
service                        plans              description
business-operations-dev        Free               Upgrade your service
business-operations-sysint     Free               Upgrade your service
logstash                       free               Logstash 1.4 service
stc-analytics-catalog          beta-plan          STC Analytics Catalo
stc-analytics-runtime          beta-plan          STC Analytics Runtim
stc-asset                      beta-plan          Service to model and
stc-monitoring                 beta-plan          Manage and monitor a
stc-postgresql-2               free               PostgreSQL 9.3 servi
stc-redis-6                    shared-vm          Redis service to pro
stc-time-series                beta-plan          Predix Time-Series S
time-series-sandbox            Development Plan   Time-Series Service
```

- Enter the command to create your own service instance with the following syntax:

  `cf create-service <service> <plan> <yourname-svc-instance>`

  Example: **`cf create-service stc-postgresql-2 free billpostgres`**

```
[predix@localhost ~]$ cf create-service stc-postgresql-2 free alpostgresql
Creating service alpostgresql in org predix-adoption / space training2 as gr
etchen.rivas@ge.com...
OK
[predix@localhost ~]$
```

Exercise 2: Deploying an Application

Overview

In this exercise, you will deploy an application to Cloud Foundry from the command line interface (CLI).

Steps

1. Change to the spring-music directory in the Terminal window.

- To determine your current directory, enter **`pwd`** and press **Enter**

```
[predix@localhost ~]$ pwd
/predix
[predix@localhost ~]$
```

- If you are not in the home directory (`/predix`), run the command: **`cd`** and then **`pwd`** again to confirm you are in the root directory
- Change to the `spring-music` directory by running this command:
  `cd PredixApps/training_labs/CloudFoundryLabs/spring-music`

```
[predix@localhost ~]$ cd PredixApps/training_labs
/CloudFoundryLabs/spring-music
[predix@localhost spring-music]$ pwd
/predix/PredixApps/training_labs/CloudFoundryLabs
/spring-music
[predix@localhost spring-music]$
```

- Run the command **`cf push`** to publish the application instance of the web application The command fails because you have not provided the parameters it needs.

```
FAILED
Error: Manifest file is not found in the current directory, please
provide either an app name or manifest
[predix@localhost spring-music]$
```

- Run the command **`cf push -h`** and read through the information presented

> ***Tip:*** **`-h`** is for help. Any command used with `-h` tells the CLI to return help information for the command. You will see the command definition, its syntax, and its options (parameters).

- Run the **`cf push`** command with the following syntax to correctly publish your application

  Note: You should be in the spring-music directory, which contains the pre-built sub-directory with the **W**eb application **AR**chive (.war), or wrapper file with all of the application files required for deployment.

  `cf push <firstname>spring-music -p pre-built/spring-music.war`

  Your application should successfully publish to Cloud Foundry.

```
Showing health and status for app gretchenspring-music in org predi
x-adoption / space training2 as gretchen.rivas@ge.com...
OK

requested state: started
instances: 1/1
usage: 512M x 1 instances
urls: gretchenspring-music.grc-apps.svc.ice.ge.com
last uploaded: Thu Sep 24 18:49:21 UTC 2015

     state     since                    cpu    memory           dis
k           details
#0   running   2015-09-24 11:50:23 AM   0.0%   476.9M of 512M   150
.6M of 1G
[predix@localhost spring-music]$
```

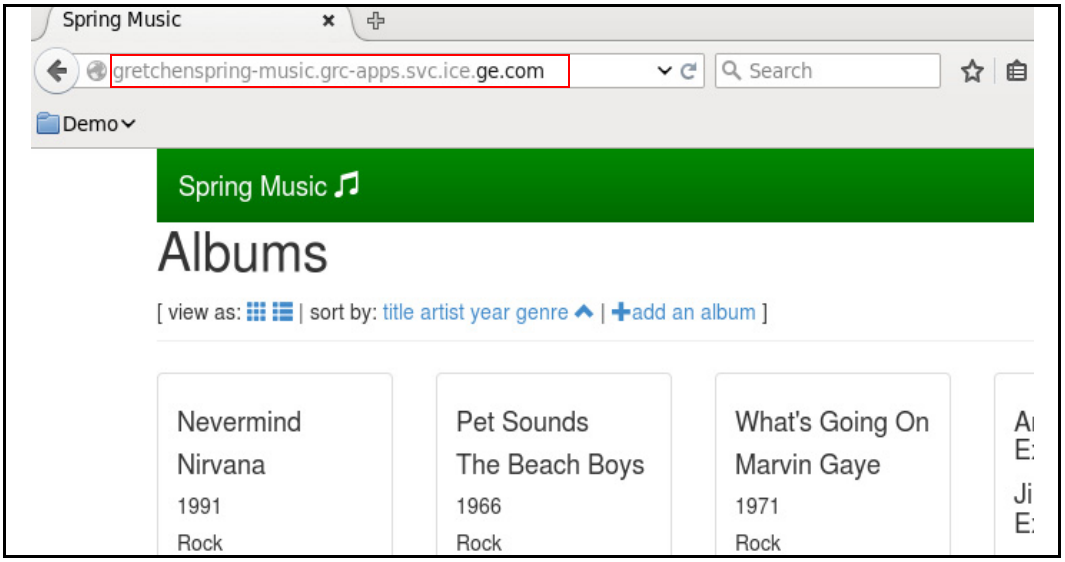
1. Test your application in a web browser

- Enter the command **`cf a`** to find the URL of your application

```
name                                requested state   instances   memor
y   disk   urls
GR-spring-music                     stopped           0/1         1G
    1G     gr-spring-music.grc-apps.svc.ice.ge.com
gretchenspring-music                started           1/1         512M
    1G     gretchenspring-music.grc-apps.svc.ice.ge.com
predix-alarmservice-GR              started           1/1         1G
    1G     predix-alarmservice-gr.grc-apps.svc.ice.ge.com
predix-alarmservice-hiroshioi       started           1/1         1G
    1G     predix-alarmservice-hiroshioi.grc-apps.svc.ice.ge.com
predix-alarmservice-PeterC          started           1/1         1G
    1G     predix-alarmservice-peterc.grc-apps.svc.ice.ge.com
```

- Copy your application URL (located in the URL column of the output)
- Open the Firefox Web browser, and paste your application URL

  The Spring Music application displays in your browser.



Exercise 3: Using a Manifest File to Deploy an Application

Overview

In this exercise you will edit a manifest file and use it to deploy your application instance. The manifest file includes a list of parameters that indicate how the solution should be deployed. Some of the parameters are required, and some are optional. You can also provide these parameters in the command line, but a manifest file is usually used to reduce the complexity of the command, and to save the information for later reuse.

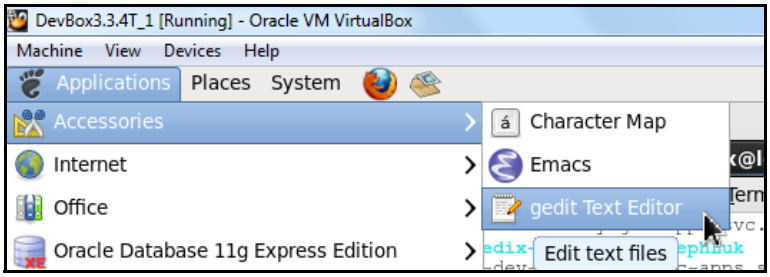
Steps

1. Add deployment parameters to the manifest file.

- Navigate to the following directory in the CLI:

`predix/PredixApps/training_labs/CloudFoundryLabs/cf-spring-mvc-demo`

- Open the gedit text editor from the Applications menu at the top left of the DevBox
  - From the Applications menu, select *Accessories>gedit Text Editor*



- Enter **Ctrl + O** and browse to the **`manifest.yml`** file in the following folder

`predix/PredixApps/training_labs/CloudFoundryLabs/cv-spring-mvc-demo`

- Edit the manifest file as follows
  - Change the application name to the name of the application (microservice) you created in Exercise 2
    - To verify the correct name, enter **`cf a`** command and locate your application (microservice) name

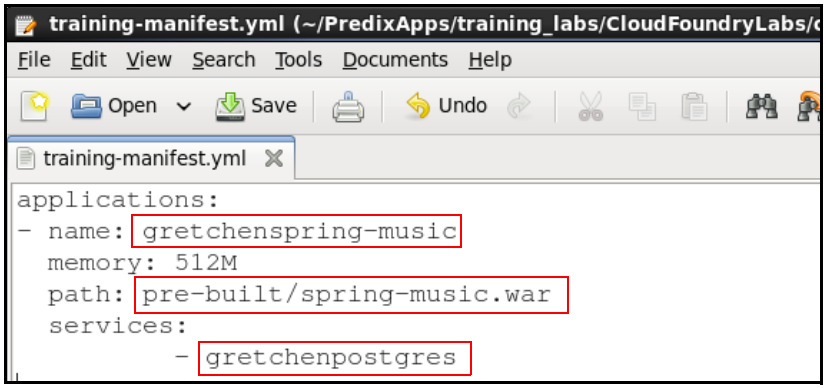
```
[predix@localhost cf-spring-mvc-demo]$ cf a
Getting apps in org predix-adoption / space training2 as gretchen.riv
as@ge.com...
OK

name                              requested state   instances   memory
  disk   urls
GR-spring-music                   stopped           0/1         1G
  1G     gr-spring-music.grc-apps.svc.ice.ge.com
gretchenspring-music              started           1/1         1G
  1G     gretchenspring-music.grc-apps.svc.ice.ge.com
inception-clock2                  started           1/1         1G
  1G     inception-clock2.grc-apps.svc.ice.ge.com
```

  - Change the path to **`pre-built/spring-music.war`**
  - Change the service listed to (keep the hyphen and space in the file)

    `- <your service created in lab 1>`

    This binds the postgresSQL service to your application (microservice)
  - To find your service name, enter **`cf s`** (services) in the CLI and look for your postgresSQL service
  - Save the file to the following folder **`predix/PredixApps/training_labs/CloudFoundryLabs/spring-music`**

Your file should read as follows (but with your individual service and application names)



2. Deploy the application to Cloud Foundry and confirm it is running.

- From the `spring-music` directory, deploy your spring-music service using the **`cf push`** command
  - Navigate to the spring-music directory in the CLI (you can check your directory by using the **`pwd`** command)
  - Run the **`cf push`** command as follows:

    `cf push <your application name> -p pre-built/spring-music.war`
  - Copy the application URL into the web browser
  - Run the **`cf a`** command to see the application URL

Exercise 4: Managing your Environment

Overview

In this exercise you will scale, stop, and delete your application instance.

Steps

1. Stop the application in Cloud Foundry.

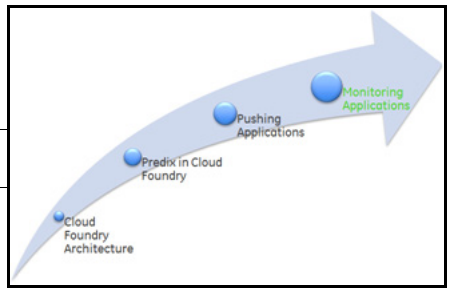
- In the Terminal, run **`cf stop < your-application-name>`**
  Example: **`cf stop bill-spring-music`**
  The application stops

2. Scale your application up to 3 instances.

- Run **`cf scale <your application name> -i 3`**
- Enter the command to scale your application back down to 1 instance

3. Delete the application.

- Enter **`cf delete <your application name> -r`**
  Note: This deletes the application and any orphaned routes from Cloud Foundry.



Exercise 5: Monitoring your Application

Overview

In this exercise you will bind a monitoring service to your application and re-deploy it.

Steps

1. Bind your application to a CF monitoring service instance.

- Enter **`cf m`** in the CLI to list the services available in the CF Marketplace
- Note the monitoring service listed

```
stc-analytics-catalog   beta-plan    STC Analytics Catalog
stc-analytics-runtime   beta-plan    STC Analytics Runtime
stc-asset               beta-plan    Service to model and manage industrial assets

stc-monitoring          beta-plan    Manage and monitor apps. By New Relic
stc-postgresql-2        free         PostgreSQL 9.3 service for application develo
```

Note: Your instructor has created a service instance and will provide the name to you

- Edit the training-manifest.yml file to include the monitoring service

```
*training-manifest.yml
applications:
- name: yourname-spring-music
  memory: 512M
  path: pre-built/spring-music.war
  services:
          - yourname-postgres
          - monitoring-service
```

- Save the file to the same location as before:
  `/predix/PredixApps/training_labs/CloudFoundryLabs/spring-music`
- Run the **`cf push`** command to re-deploy as follows:

  `cf push <your application name> -f training-manifest.yml`

  This binds your application to the monitoring service as well as to your postgresSQL service