

A background image showing a close-up of a business meeting. Two people are seated at a table, looking at documents. One person's hand is visible, holding a pen and pointing at a document. The documents contain charts and graphs. The scene is brightly lit, suggesting a professional office environment.

# Pivotal Cloud Foundry

# Table of Contents

---

- Getting started with Pivotal CF
  - Using Command Line
  - Web plug-in
  - Eclipse plug-in
  - Developing with Cloud Foundry

# Getting Started with Command Line Interface

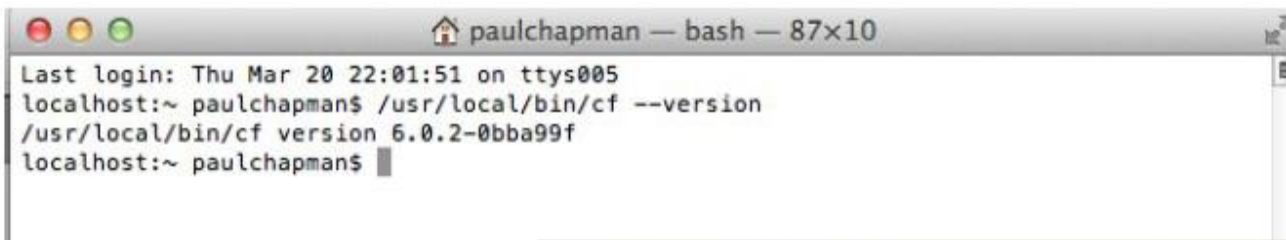
# The Command Line Interface

- Several interface options exists for Cloud Foundry
  - Command Line Interface (CLI)
  - Web-based Application Manager Console
  - Eclipse / STS plug-in
- Primary access is done via the CLI
  - Make sure you have it installed
    - Installation was covered in the “welcome” module

# Test the CLI Utility

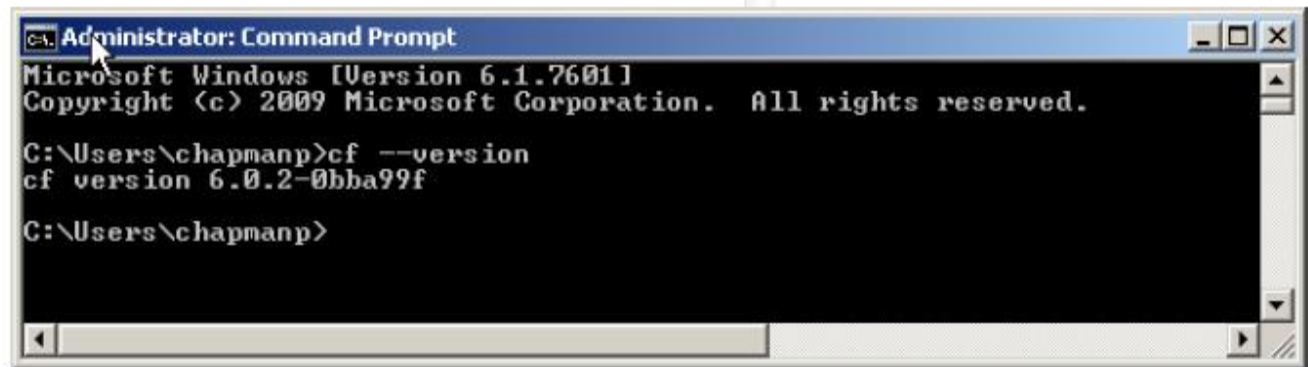
- It is called **cf**
  - Open a command/Shell window
  - At the prompt type : **cf --version**

- ◆ Version *must* be **6** or more
- ◆ *Must* remove any earlier (Ruby) version



A terminal window titled 'paulchapman — bash — 87x10'. The output shows the command '/usr/local/bin/cf --version' being executed, resulting in 'cf version 6.0.2-0bba99f'.

```
Last login: Thu Mar 20 22:01:51 on ttys005
localhost:~ paulchapman$ /usr/local/bin/cf --version
/usr/local/bin/cf version 6.0.2-0bba99f
localhost:~ paulchapman$
```

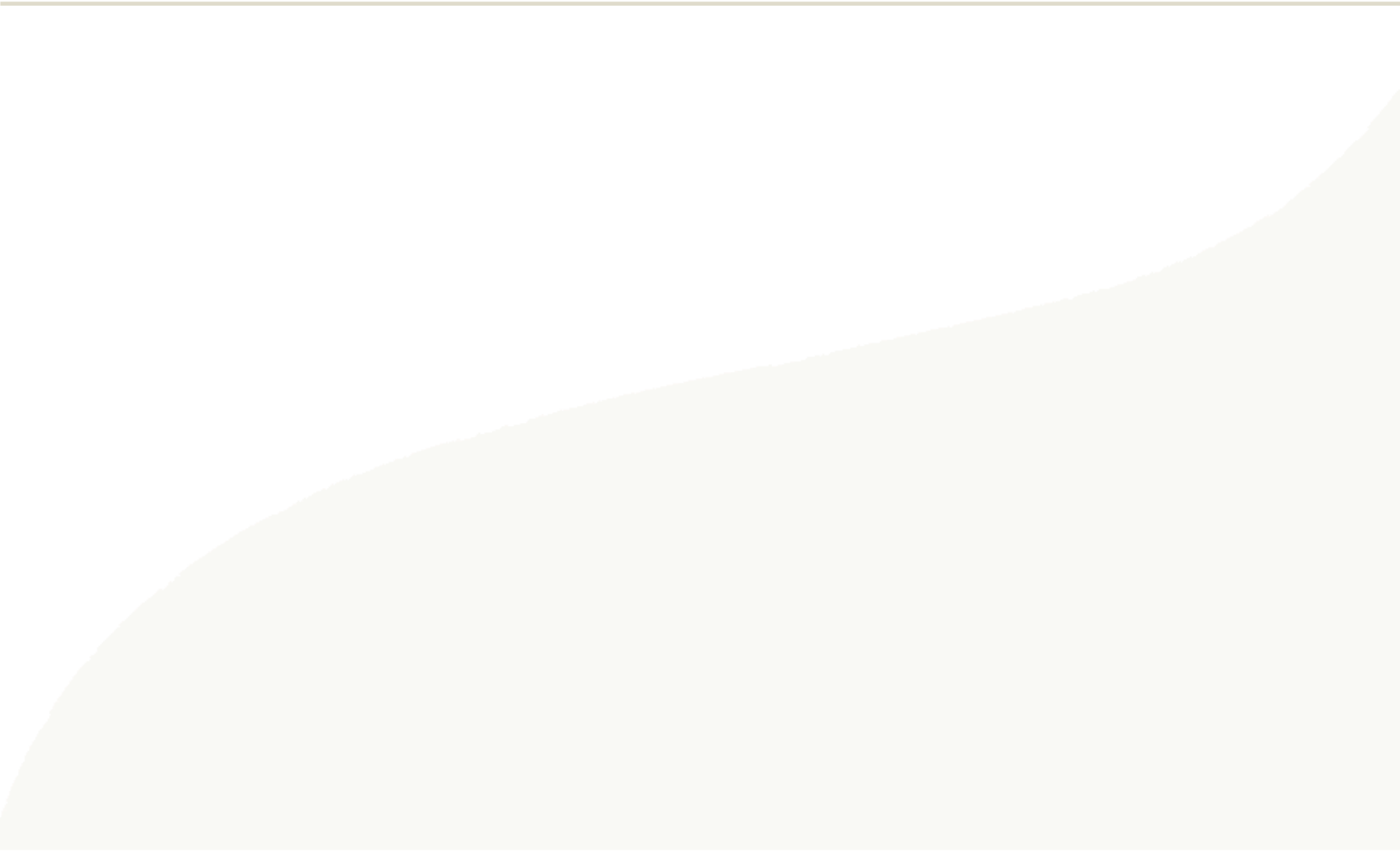


A Windows Command Prompt window titled 'Administrator: Command Prompt'. The output shows the command 'cf --version' being executed, resulting in 'cf version 6.0.2-0bba99f'.

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\chapmanp>cf --version
cf version 6.0.2-0bba99f

C:\Users\chapmanp>
```



# Getting Help

- Get help at any time via **cf help**
- Or **cf help <command>**

# DO NOW – Get Help on a Command

- Perform these steps on your computer:
  - Open a command prompt
  - Issue the **cf help** command
  - Get help on the login command: **cf help login**
- Answer these questions:
  - What option do you use to specify username?
  - Is specifying the password option encouraged?



# Login

# Login to Cloud Foundry

- Need to tell cf
  - What cloud foundry instance you are using
  - What your account details are
  - Use **cf login**

Color highlighting  
MacOS, Linux only

```
> cf login -a api.run.pivotal.io -u <username>
```

```
API endpoint: api.run.pivotal.io
```

```
Authenticating...
```

```
OK
```

```
Targeted org Cloud Foundry Course
```

```
Targeted space development
```

```
API endpoint: https://api.run.pivotal.io (API version: 2.0.0)
```

```
User: qzqz2020@yahoo.com.au
```

```
Org: Cloud Foundry Course
```

```
Space: development
```

*Will prompt for anything you  
don't specify  
No -p? Prompts for password*

# Cloud Foundry URLs

- To access CF you need to know 3 URLs

- API Endpoint

- Identifies your CF instance
    - Used to deploy applications, manage spaces, routes...
    - The `cf` utility makes *RESTful* requests to this URL

Actually to the Cloud Controller

- Apps Manager

- Application management dashboard
    - Pivotal CF only

- Apps Domain

- Used to access deployed applications
    - May be same as System Domain

# Cloud Foundry URLs

For simplicity, most examples  
In this session show PWS URLs

- System & App domains defined when CF was installed

- If using PWS

- System domain: **run.pivotal.io**
  - API Endpoint: **api.run.pivotal.io**
  - Apps Manager: **console.run.pivotal.io**
- Apps domain: **cfapps.io**



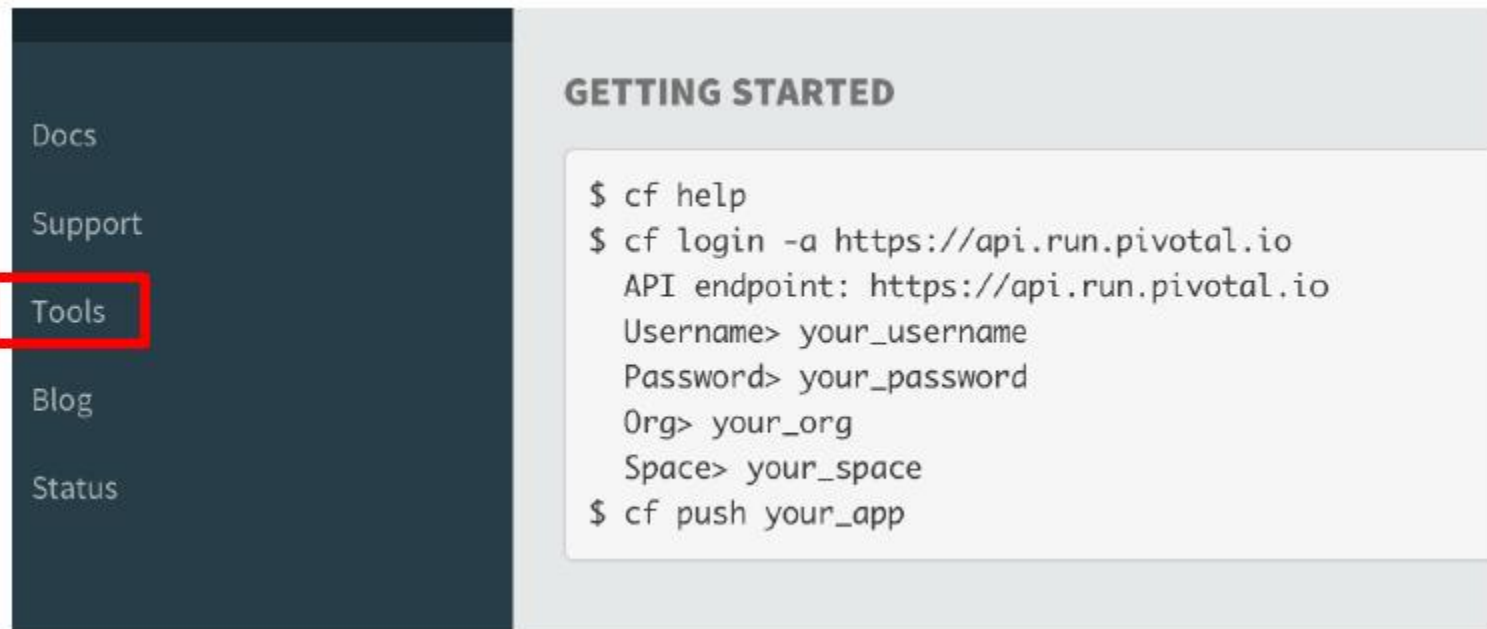
- Your own CF installation

- System domain: **<your-cf-system-domain>**
  - API Endpoint: **api. <your-cf-system-domain>**
  - Apps Manager: **console. <your-cf-system-domain>**
- Apps domain: **<your-cf-system-domain>**



# Finding the API Endpoint URL

- URL of Cloud Controller in our Cloud Foundry instance
  - On Apps Manager home-page on first login
  - Or click Tools
  - Shows how to run **cf login**, including the API Endpoint



# DO NOW – Login

- Perform these steps on your computer

- Login with **cf login** command
  - Specify CF instance using **-a <API-URL>**  
For PWS: **-a api.run.pivotal.io**
  - Specify email / password
  - If prompted, select an organization and space

```
$> cf login -a <API-URL> -u <your-email-or-username>  
API endpoint: api.run.pivotal.io  
...
```

- Firewall issues?

<http://docs.cloudfoundry.org/devguide/installcf/http-proxy.html>

# The .cf Directory

- **cf** creates a **.cf** directory in your home directory
  - Stores context, logs, crash reports ...
  - Remembers your CF API Endpoint
    - Don't need to specify **-a** option at next login

```
localhost:dev$ ls -l ~/.cf
total 48
-rw-----  1 paulchapman  staff    2491 15 Aug 14:06 config.json
-rw-r--r--  1 paulchapman  staff  11737 29 Nov 2013 crash
drwxr-xr-x  3 paulchapman  staff    102 12 Sep 2013 logs
-rw-r--r--  1 paulchapman  staff     26 12 Sep 2013 target
-rw-r--r--  1 paulchapman  staff   2084 29 Nov 2013 tokens.yml
```

# DO NOW - .cf folder

- Perform these steps on your computer:
  - Find the **.cf** folder / directory on your computer:
    - You won't (yet) have all the files shown on previous slide
  - Open the **config.json** file, observe the contents



# Current Targets

- When your first login you see output like this
  - Notice it shows current organization and space
  - At any time, run **cf target** to get same information

```
API endpoint: https://api.run.pivotal.io (API version: 2.6.0)
User:        pchapman@pivotal.io
Org:         pivotedu
Space:       development
```

- By default your organization only has one space
  - Development
- **Note:** on PWS you are setup as your own organization

# Viewing Organization



## ■ Commands

- **cf orgs** All orgs for current user
- **cf org <org-name>** Shows specified org

```
>$ cf org pivotaledu
Getting info for org myorg as user@somedomain.com
OK

pivotaledu:
  domains:      cfapps.io
  quota:        paid (10240M memory limit, Unlimited instance
                 memory limit, 1000 routes, -1 services,
                 paid services allowed)
  spaces:       development, production, staging
  space quotas:
>$
```

# Managing Space



- To see all the spaces in an organization
  - **cf spaces**
- Create a new space (in current organization by default)
  - **cf create-space <space-name>**
  - **cf create-space <space-name> -o <org-name>**
- Use **target** command to change space ( or organization)
  - **cf target -s <space-name>**
  - **cf target -o <org-name>**

# Deploy An Application

# Deploy Using the CLI

- You need a deployable application
  - For example with Java: a jar or war
    - Ant, maven or Gradle built-tools can make it for us
    - Cloud Foundry doesn't care how you build your application
  - Other languages (Ruby, Node.js, etc ...) : the course will do

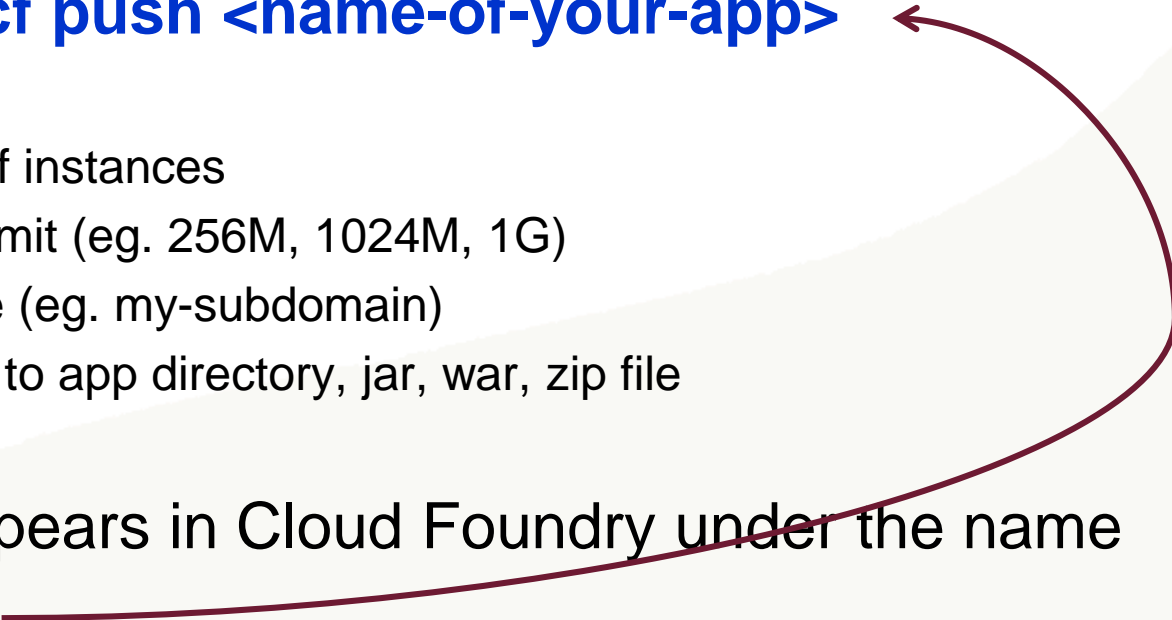
# The cf push Philosophy

- Onsi Fakhouri (Cloud Foundry PM)
  - Here is my source code
  - Run it on the cloud for me
  - I do not care how
- The architecture of CF is fascinating
  - And we will cover it
  - But ultimately irrelevant
- I just want to push an application
  - I no longer need to know: how that happens, how it is packaged or how it is run?



*Haiku*

# Deploy push to Cloud Foundry

- Deploy by running **cf push <name-of-your-app>**
    - Many options
      - -i Number of instances
      - -m Memory limit (eg. 256M, 1024M, 1G)
      - -n Hostname (eg. my-subdomain)
      - -p local path to app directory, jar, war, zip file
  - Your application appears in Cloud Foundry under the name you specify here
- 

# Domains and URLs

- Every CF instance is assigned a domain at installation
  - Known as the *Apps Domain*
  - For PWS this is **cfapps.io**
- When you deploy, your application gets a unique route (URL) to access it: **hostname + app domain name**
  - By default, hostname = application name
  - Make sure hostname is **unique**
    - cf push returns an HTTP 400 error if not
- PWS example:
  - **cf push spring-music ...**
  - gets route: **spring-music.cfapps.io**



# Example of Using cf push

*Specify unique sub-domain  
by adding numbers, initials ...*



## ■ Fully specified (recommended)

```
cf push spring-music -i 1
```

```
-m 512M
```

```
-n spring-music-678
```

```
-p build/libs/spring-music.war
```

- Deploys war file (specify path if needed)
- 1 instance, 512M memory
- Name: **spring-music**
  - Appears as **spring-music** in Cloud Foundry
- Hostname: **spring-music-678**
  - Creates URL (PWS): **spring-music-678.cfapps.io**

# What Happens?

- cf connects to Cloud Foundry using your credentials
- It 'pushes' your application to CF and tells it to deploy it
  - The whole application is uploaded – takes a while
  - CF “stages” your application
    - Recognizes Java WAR file, prepares a “droplet” with a JRE and Tomcat server
    - “Droplet” is deployed to a container and starts running
    - All requests to the deployed URL route to your application
- Whole process logged on screen
  - Next 3 slides

# What Happens – 1

```
cf push spring-music -n spring-music-678 -i 1 -m 512M
-p pre-built/spring-music.war
```

URL: spring-music-678.cfapps.io

```
> cf push spring-music -n spring-music-678 -p build/libs/spring-music.war -i 1 -m 512M
```

Updating app `spring-music` in org `your-org` / space `development` as `your-id@company.io`...

OK

Using route `spring-music-678.cfapps.io`

Uploading `spring-music`...

Uploading app files from: `pre-built/spring-music.war`

Uploading 574.8K, 95 files

Done uploading

OK

Starting app `spring-music` in org `your-org` / space `development` as `your-id@company.io`...

...

Next...

Updates CF metadata (app name, instances, memory)

Establish route

Uploads war

# What Happens – “Staging”

CF must prepare the app before its first run

```
...
Starting app spring-music in org your-org / space development as your-id@company.io...
OK
```

“Buildpack” selected and executed

```
-----> Downloaded app package (21M)
-----> Java Buildpack Version: v2.7.1 | https://github.com/cloudfoundry/java-buildpack#fee275a
-----> Downloading Open Jdk JRE 1.8.0_40 from
https://download.run.pivotal.io/openjdk/lucid/x86_64/openjdk-1.8.0_40.tar.gz (6.1s)
      Expanding Open Jdk JRE to .java-buildpack/open_jdk_jre (1.3s)
-----> Downloading Spring Auto Reconfiguration 1.7.0_RELEASE from
https://download.run.pivotal.io/auto-reconfiguration/auto-reconfiguration-1.7.0_RELEASE.jar (0.2s)
-----> Downloading Tomcat Instance 8.0.20 from
https://download.run.pivotal.io/tomcat/tomcat-8.0.20.tar.gz (1.1s)
      Expanding Tomcat to .java-buildpack/tomcat (0.1s)
-----> Downloading Tomcat Lifecycle Support 2.4.0_RELEASE from
https://download.run.pivotal.io/tomcat-lifecycle-support/tomcat-lifecycle-support-2.4.0_RELEASE.jar (0.0s)
-----> Uploading droplet (73M)
```

Buildpack configures Java

Reconfigure Spring for cloud environment

Buildpack obtains Tomcat

Next...

Buildpack creates “Droplet”

# What Happens – start

Cloud Foundry runs the  
“Droplet” on a “container”

```
...  
0 of 1 instances running, 1 starting  
0 of 1 instances running, 1 starting  
1 of 1 instances running
```

App started

OK

App `spring-music` was started using this command ``JAVA_HOME=$PWD/.java-buildpack/open_jdk_jre JAVA_OPTS="-Djava.io.tmpdir=$TMPDIR -XX:OnOutOfMemoryError=$PWD/.java-buildpack/open_jdk_jre/bin/killjava.sh -Xmx382293K -Xms382293K -XX:MaxMetaspaceSize=64M -XX:MetaspaceSize=64M -Xss995K -Daccess.logging.enabled=false -Dhttp.port=$PORT" $PWD/.java-buildpack/tomcat/bin/catalina.sh run``

Showing health and status for app `spring-music` in org `your-org` as `your-id@company.io`...

OK

```
requested state: started  
instances: 1/1  
usage: 512M x 1 instances  
urls: spring-music-678.cfapps.io  
last uploaded: Tue Mar 17 17:58:35 UTC 2015
```

Health Check

	state	since	cpu	memory	disk
#0	running	2015-03-17 01:59:35 PM	0.0%	474.4M of 512M	150.3M of 1G

Done! 1 application instance running on `spring-music-678.cfapps.io`

# Application State and Logs

- Run `cf apps`

```
> cf apps
```

```
Getting apps in org pivotaledu / space development as kkrueger@pivotal.io...
```

```
OK
```

name	requested state	instances	memory	disk	urls
spring-music	started	1/1	512M	1G	spring-music-678.cfapps.io

- `cf logs spring-music`

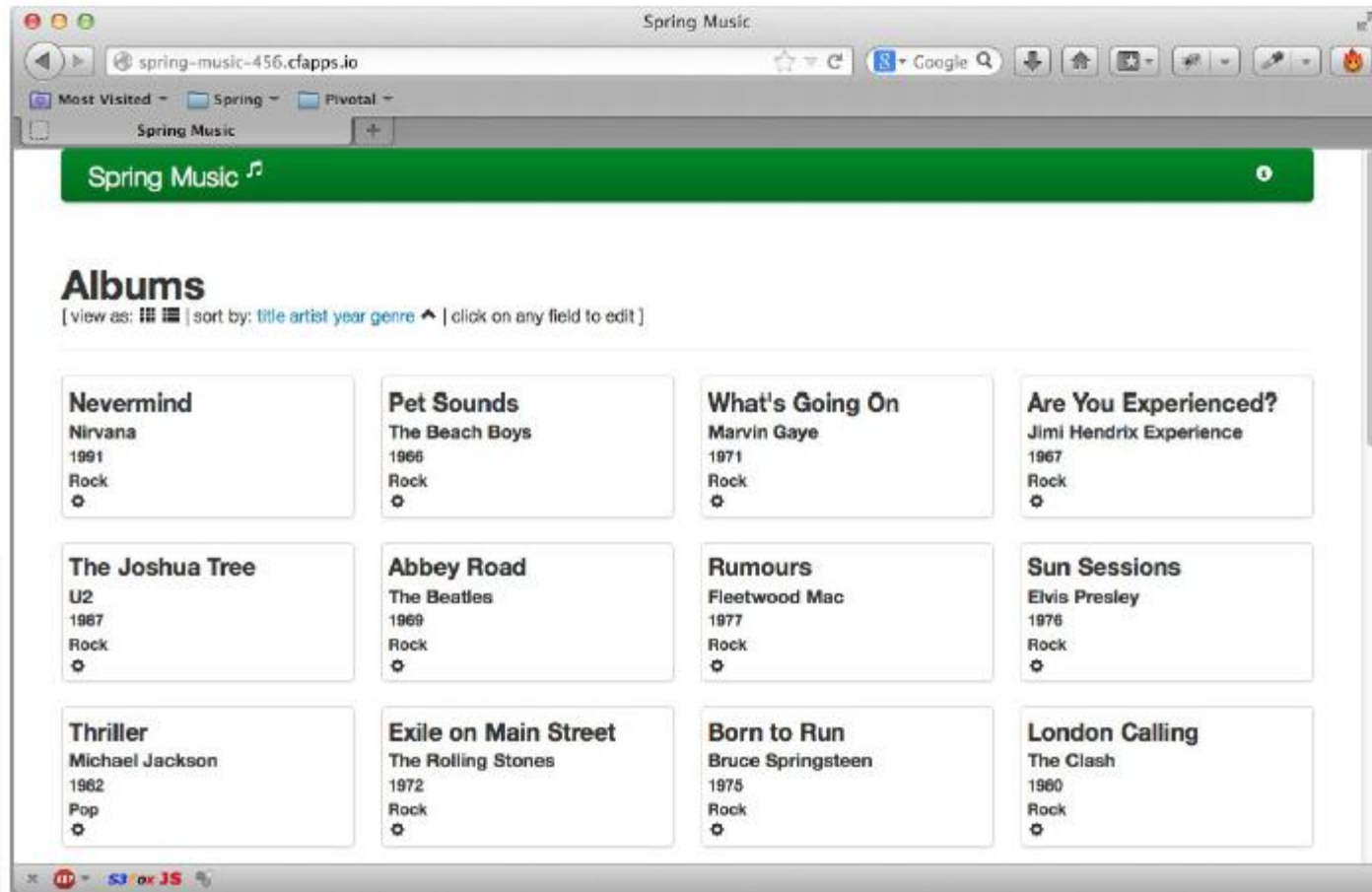
```
> cf logs spring-music
```

```
Connected, tailing logs for app spring-music in org pivotaledu / space development as kkrueger@gopivotal.com...
```

```
2014-06-07T23:01:47.68-0400 [RTR]      OUT spring-music-678.cfapps.io -  
[08/06/2014:03:01:47 +0000] "GET /assets/js/status.js HTTP/1.1" 200 844 "http://spring-music-678.cfapps.io/" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_7_5) AppleWebKit/537.73.11 (KHTML, like Gecko) Version/6.1.1 Safari/537.73.11"  
10.10.66.34:64401 vcap_request_id:73037523-63ef-498f-6cd8-d3b48fe69e84  
response_time:0.003693009 app_id:314f0434-d2c9-446c-ab4a-6c310878ca80  
2014-06-07T23:01:48.47-0400 [RTR]      OUT spring-music-678.cfapps.io -  
[08/06/2014:03:01:48 +0000] "GET /assets/templates/header.html HTTP/1.1" 200 1060  
"http://spring-music-678.cfapps.io/" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_7_5) AppleWebKit/537.73.11 (KHTML, like Gecko) Version/6.1.1 Safari/537.73.11"  
10.10.66.34:64324 vcap_request_id:39fbb3f2-46fb-4bd7-78d6-8994fafade9f  
response_time:0.004132254 app_id:314f0434-d2c9-446c-ab4a-6c310878ca80
```

# See the Application Running

- Open a browser window to [spring-music-678.cfapps.io](http://spring-music-678.cfapps.io)





# Configuring a Deployed Application

- Change the number of instances
  - **cf scale <app> -i <new-value>**
  - Two instances: **cf scale spring-music -i 2**
  - New instances added, or some existing instances stopped
- Change the memory allocation
  - **cf scale <app> -m <new-value>**
  - 1024M: **cf scale spring-music -m 1024M**
  - Requires a restart to take effect



# Stopping and Starting

- cf stop
  - Sends SIGTERM message to application
  - Sends SIGKILL 10 seconds later if still running
- cf start
  - Starts existing application
- cf restart
  - cf stop followed by cf start
- cf restage
  - Repeats the staging process, and starts the app.
  - Useful when environment variables / bound services change
    - \* (Covered later)

# Adding / Removing Routes



- Add a new domain mapping
  - **cf map-route <app> <domain> -n <hostname>**
  - **cf map-route spring-music cfapps.io -n mymusic**
    - **mymusic.cfapps.io** **also** maps to spring-music
- Remove mapping
  - **cf unmap-route <app> <domain> -n <hostname>**
  - **cf unmap-route spring-music cfapps.io -n spring-music-678**
    - **Spring-music-678.cfapps.io** **no longer** maps to spring-music

# Cleaning Up Unused Routes

- Routes tend to accumulate over time
  - Application in other Orgs / Spaces cannot use these routes
- Find all other routes used in a space:
  - **cf routes**
- Remove route:
  - **cf delete-route**
- Very Useful! Remove unused routes:
  - **cf delete-orphaned-routes**

# Managing Application Instances

# Apps Manager

- Login to Cloud Foundry using your web-browser
  - Pivotal Web Services : <http://run.pivotal.io>
    - Your Cloud Foundry instance URL will be different
    - console.<your-cf-domain>
  - Use the username and password you registered with
  - Our new application should show green in the Apps Manager
- Next slide

***Note:*** only pivotal CF comes with the apps manager  
Open Source Cloud Foundry does not

# Apps Manager Home Page

- At a glance view of all your applications
  - Shows current space

The screenshot shows the Pivotal Web Services Apps Manager interface. On the left is a dark sidebar with the Pivotal logo and 'Web Services' text. Below this, it shows 'ORG' as 'pivotaledu' and 'SPACES' with a list: 'development' (selected), 'production', 'staging', and 'Marketplace'. The main area has a header 'pivotaledu > development' and a 'SPACE' button labeled 'development'. Below this is an 'APPLICATIONS' section with a 'LEARN MORE' link. It contains a table with columns: STATUS, APP, INSTANCES, and MEMORY. The table lists two applications: 'classfeedback-dev' (status: STOPPED, 1 instance, 1GB memory) and 'spring-music' (status: 100%, 1 instance, 512MB memory). Annotations with red arrows point to: the 'development' space button, the 'staging' space in the sidebar, the 'spring-music' application row, and the application URL 'spring-music-678.cfapp...'.

Click application name to see its dashboard (next slide)

Click to select different space

URL of your application

STATUS	APP	INSTANCES	MEMORY
STOPPED	classfeedback-dev classfeedback-dev.cfapp...	1	1GB
100%	spring-music spring-music-678.cfapp...	1	512MB

# Application Dashboard

The screenshot shows the Application Dashboard for an application named 'smkk'. The dashboard is divided into several sections:

- Configuration:** Includes a 'Scale App' button, an 'INSTANCES' field set to 1, a 'MEMORY LIMIT' field set to 512 MB, and a 'DISK LIMIT' field set to 1024 MB. A red arrow points to the 'Scale App' button and the 'INSTANCES' field with the text 'Change instances, memory'.
- Status:** A table showing instance statistics. A red arrow points to the table with the text 'Instance Statistics'.
- Activity:** A tabbed interface with 'Services', 'Env Variables', and 'Routes'. A red arrow points to the 'Delete App' button with the text 'Logs and Events'.
- Events:** A section showing recent events, including 'started app'.
- Recent Logs:** A section showing recent logs, including timestamps and log messages.

Annotations with red arrows point to the following elements:

- Change instances, memory (points to the 'Scale App' button and the 'INSTANCES' field)
- Application state (points to the 'smkk' application icon)
- Instance Statistics (points to the 'STATUS' table)
- Logs and Events (points to the 'Delete App' button)

Change instances, memory

Application state

Instance Statistics

Logs and Events

# Change Mapped URL

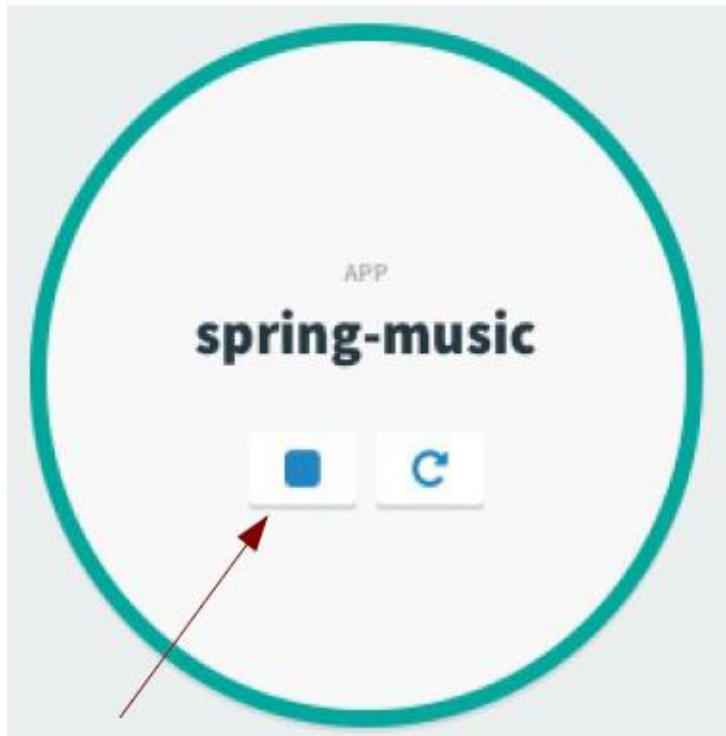
- Enter new domain (1)
  - Remember to make it Unique
- Click MAP URL (2)
- To remove a mapping (3)
  - Click UNMAP URL at far right of same line

The screenshot shows a web interface titled "ROUTES". It contains a form for managing domain mappings. The form has a text input field labeled "New URL" with a red arrow and the number "1" pointing to it. To the right of this field is a dropdown menu showing "cfapps.io" with a red arrow and the number "2" pointing to it. To the right of the dropdown is a blue button labeled "MAP URL" with a red arrow and the number "2" pointing to it. To the right of the "MAP URL" button is a blue button labeled "Unmap URL" with a red arrow and the number "3" pointing to it. Above the form, the text "spring-music-678.cfapps.io" is displayed.



# Stopping and Starting

- Just click the square to stop
- Click play to start



Click to Stop



Click to Start

# Monitoring Instances

- The very bottom panel shows all your instances
  - Provides statistics
  - Updated live (slight time – lag)

INSTANCES					
INSTANCE	CPU	MEMORY	DISK	UPTIME	STATE
1	0%	312MB	121MB		Running
0	0%	314MB	121MB	31min	Running

# Deploying via Eclipse or Spring Tool Suite

# Setting up Eclipse / Spring Tool Suite

- Select Help → Eclipse Marketplace
- Search for Cloud Foundry plugin and install

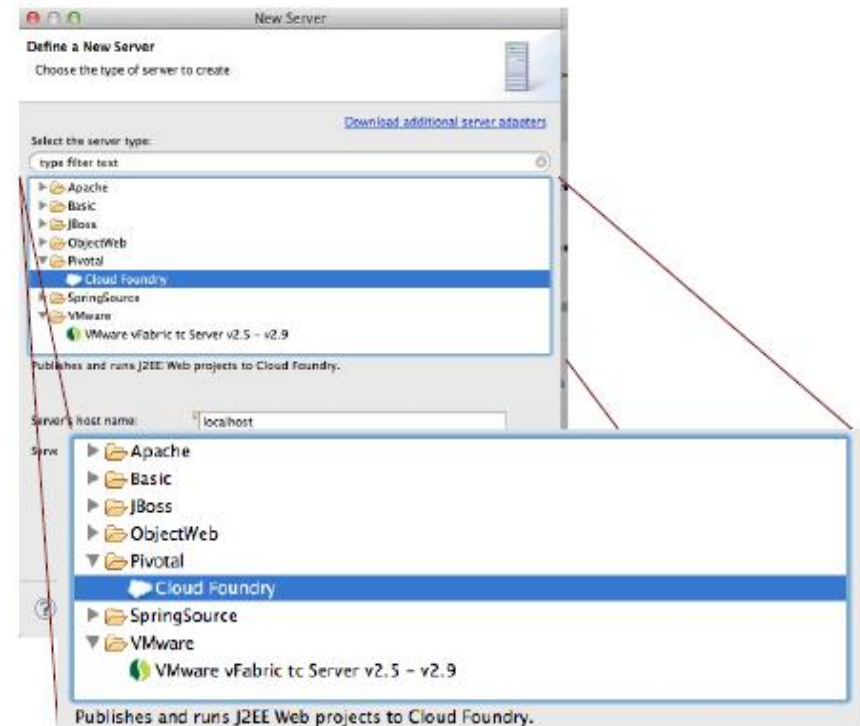
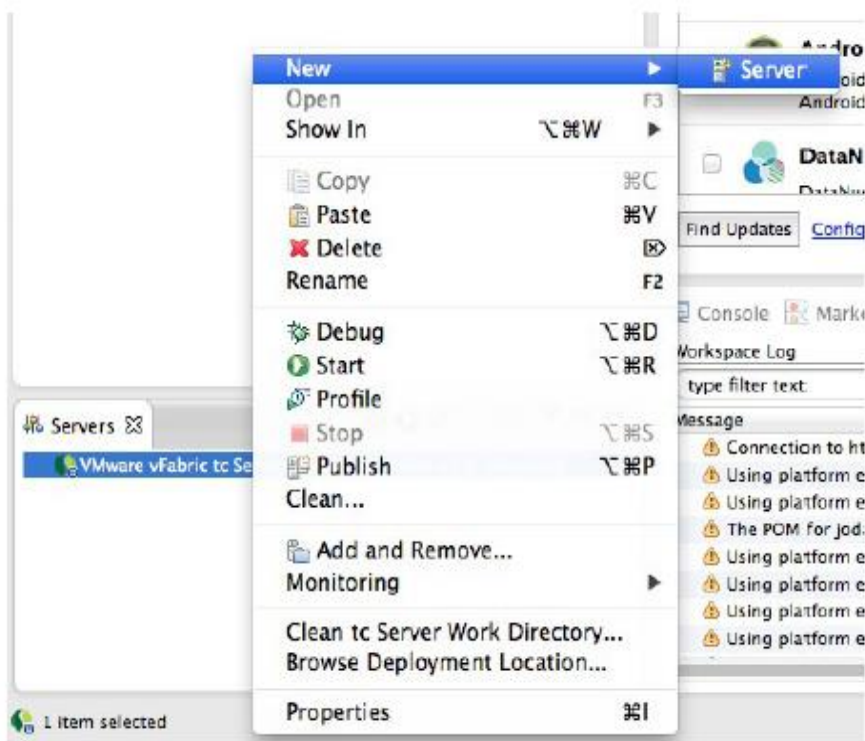


Do this now...

- The install takes time to run.

# Setting up a New Server

- In white-area of Servers panel, right click New → Server
- In popup, under Pivotal select Cloud Foundry



# Fill in Details of your CF Account

- Fill in registration dialog
  - Use Manage Cloud to specify a different URL
    - Another public PaaS or for your private cloud
  - Note there is a Signup button here.

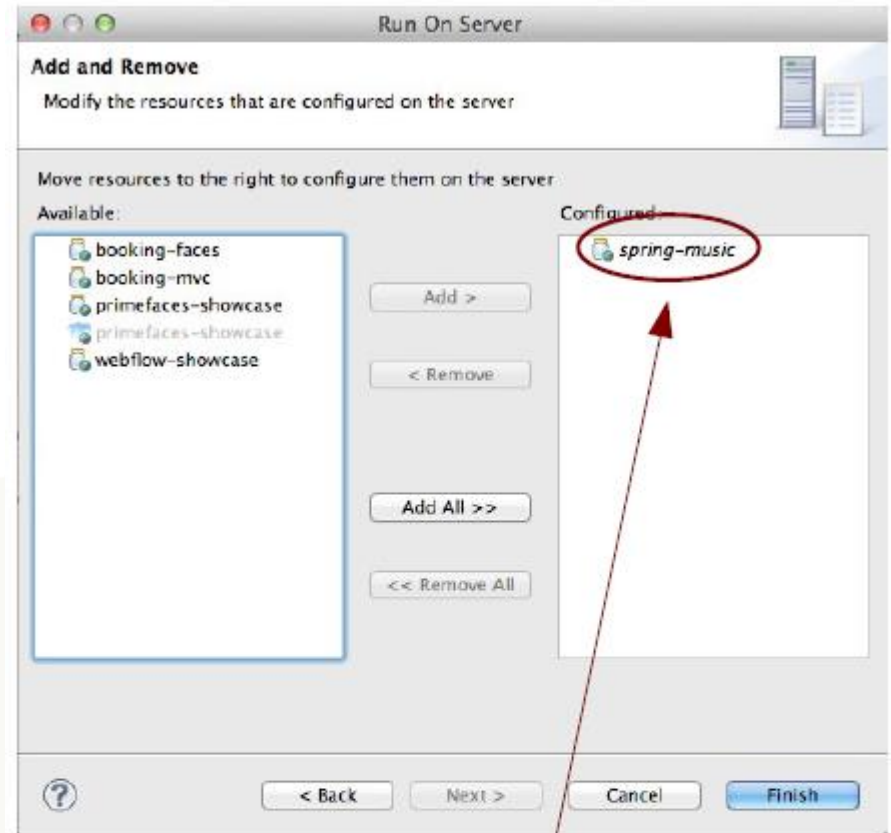
*Same email and password  
That you signed-up with*

*Click Finish when done*

The screenshot shows a 'New Server' dialog box with a 'Cloud Foundry Account' section. It includes fields for 'Email', 'Password', and 'URL'. The 'URL' field is set to 'Pivotal Cloud Foundry Hosted Developer Edition - https://api.run.pivotal.io'. There are buttons for 'Validate Account', 'Register Account...', and 'Pivotal CF Signup'. At the bottom right, there are navigation buttons: '< Back', 'Next >', 'Cancel', and 'Finish'. The 'Finish' button is circled in red. A red arrow points from the text 'Click Finish when done' to the 'Finish' button. Another red arrow points from the text 'Same email and password That you signed-up with' to the 'Email' and 'Password' fields.

# Deployment

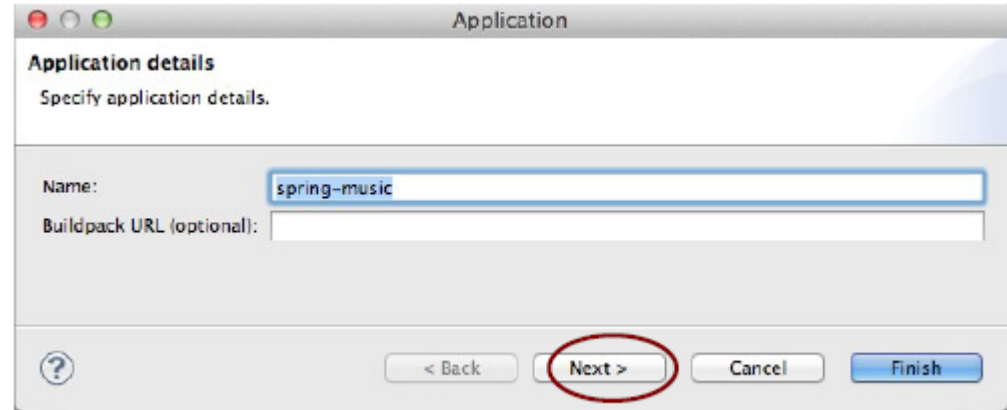
- We are now ready to deploy an application
- Select a project in Eclipse
  - Right click and select
    - Run As ... → Run on server
    - *Just like any other server*
  - Select Cloud Foundry server
  - Click Next
  - In next dialog make sure your project is in the RHS list
    - Just as you would normally
  - Click Finish to deploy



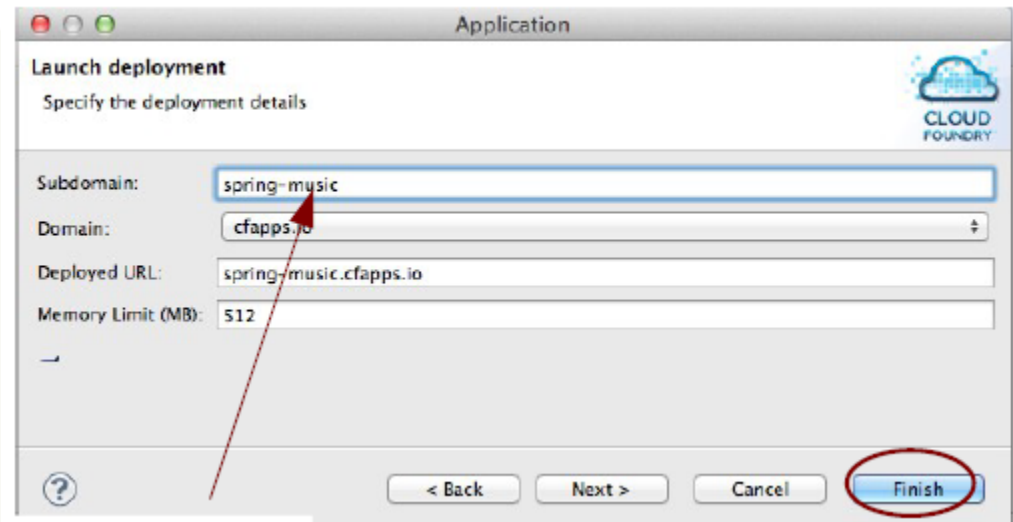
**Now things get different ...**

# Application details

- Two dialogs appear
  - Application details ... just click next for now
  - Launch deployment... Pick a unique URL
    - All apps deploy to <appname>.cfapps.io
  - For now, just click Finish to deploy



The 'Application details' dialog box is titled 'Application' and 'Application details'. It contains a text field for 'Name' with the value 'spring-music' and an empty text field for 'Buildpack URL (optional)'. At the bottom, there are four buttons: '< Back', 'Next >', 'Cancel', and 'Finish'. The 'Next >' button is circled in red.



The 'Launch deployment' dialog box is titled 'Application' and 'Launch deployment'. It contains a 'Subdomain' text field with 'spring-music', a 'Domain' dropdown menu with 'cfapps.io', a 'Deployed URL' text field with 'spring-music.cfapps.io', and a 'Memory Limit (MB)' text field with '512'. At the bottom, there are four buttons: '< Back', 'Next >', 'Cancel', and 'Finish'. The 'Finish' button is circled in red. A red arrow points from the 'Subdomain' field to the 'Finish' button.

*Will this be unique? Change Sub-domain to make sure*

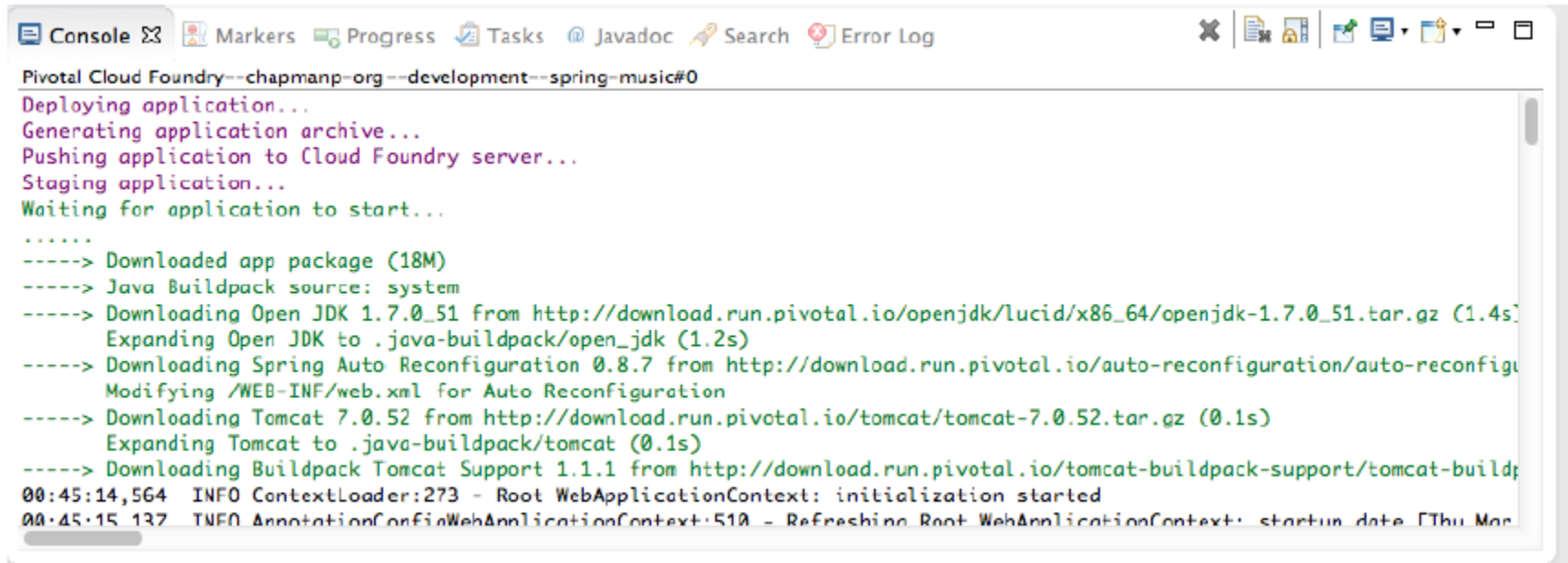


# What just happened?

- Eclipse Connected to Cloud Foundry using your credentials
- It 'pushed' your application to CF and told it to deploy it
  - The whole application is uploaded – takes a while
  - CF “staged” your application
    - Recognized Java/ WAR, prepared a “droplet” containing JRE and Tomcat server
  - “Droplet” was deployed to a container and began running
  - All requests to the Deployed URL route to your application
- Same process as when using the CLI

# Watching it Run

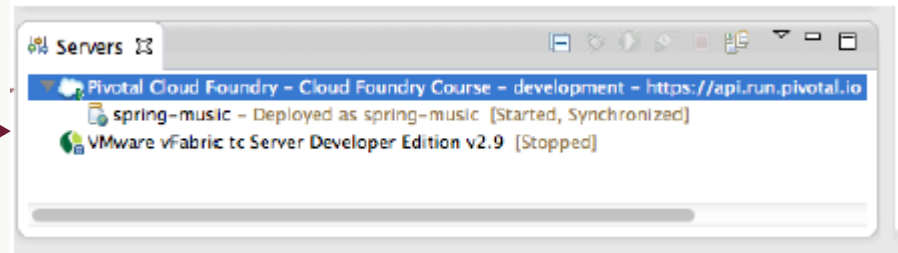
## ■ In the Console View



```
Pivotal Cloud Foundry--chapman-org--development--spring-music#0
Deploying application...
Generating application archive...
Pushing application to Cloud Foundry server...
Staging application...
Waiting for application to start...
.....
-----> Downloaded app package (18M)
-----> Java Buildpack source: system
-----> Downloading Open JDK 1.7.0_51 from http://download.run.pivotal.io/openjdk/lucid/x86_64/openjdk-1.7.0_51.tar.gz (1.4s)
Expanding Open JDK to .java-buildpack/open_jdk (1.2s)
-----> Downloading Spring Auto Reconfiguration 0.8.7 from http://download.run.pivotal.io/auto-reconfiguration/auto-reconfigu
Modifying /WEB-INF/web.xml for Auto Reconfiguration
-----> Downloading Tomcat 7.0.52 from http://download.run.pivotal.io/tomcat/tomcat-7.0.52.tar.gz (0.1s)
Expanding Tomcat to .java-buildpack/tomcat (0.1s)
-----> Downloading Buildpack Tomcat Support 1.1.1 from http://download.run.pivotal.io/tomcat-buildpack-support/tomcat-buildp
00:45:14,564 INFO ContextLoader:273 - Root WebApplicationContext: initialization started
00:45:15,137 INFO AnnotationConfigWebApplicationContext:510 - Refreshing Root WebApplicationContext: startup date [Thu Mar
```

## ■ In the Dashboard

- Double click



# Overview Dashboard Tab

Pivotal Cloud Foundry

Spring Music

Pivotal Cloud Foundry

General Information

Specify the host name and other common settings.

Server name:

Host name:

Runtime Environment:

Account Information

Email:

Password:

URL:

Organization:

Space:

Clone Server...

Change Password...

Validate Account

Pivotal CF Signup

Server Status

Pivotal Cloud Foundry: Connected

Connect

Disconnect

Publishing (manually only)

Timeouts

Overview

Applications and Services

# Application Tab

The screenshot shows the Pivotal Cloud Foundry interface for the 'Spring Music' application. The 'Applications' tab is active, displaying a list of applications with 'spring-music' selected. A red arrow points to this selection with the text 'Select app first'. The right-hand pane shows the 'General' configuration for the selected application. A red arrow points to the 'Name' field, which contains 'spring-music (Started)', with the text 'Application status here'. Another red arrow points to the 'Mapped URLs' field, which contains 'spring-music-123.cfapps.io', with the text 'Click URL once running'. Below the 'General' section, there are buttons for 'Start', 'Stop', 'Restart', and 'Update and Restart'. The 'Instances' section shows a table with one instance running. The 'Services' section is empty.

**General**

Name: spring-music (Started)

Mapped URLs: [spring-music-123.cfapps.io](http://spring-music-123.cfapps.io)

Instances: 1

Manifest: [Save](#)

**General (Application Restart Required)**

Memory Limit (MB): 512 [Set](#)

Environment Variables: [Edit...](#)

**Application Operations**

[Start](#) [Stop](#) [Restart](#) [Update and Restart](#)

**Application Services**

Name	Vendor	Plan	Version
------	--------	------	---------

**Instances**

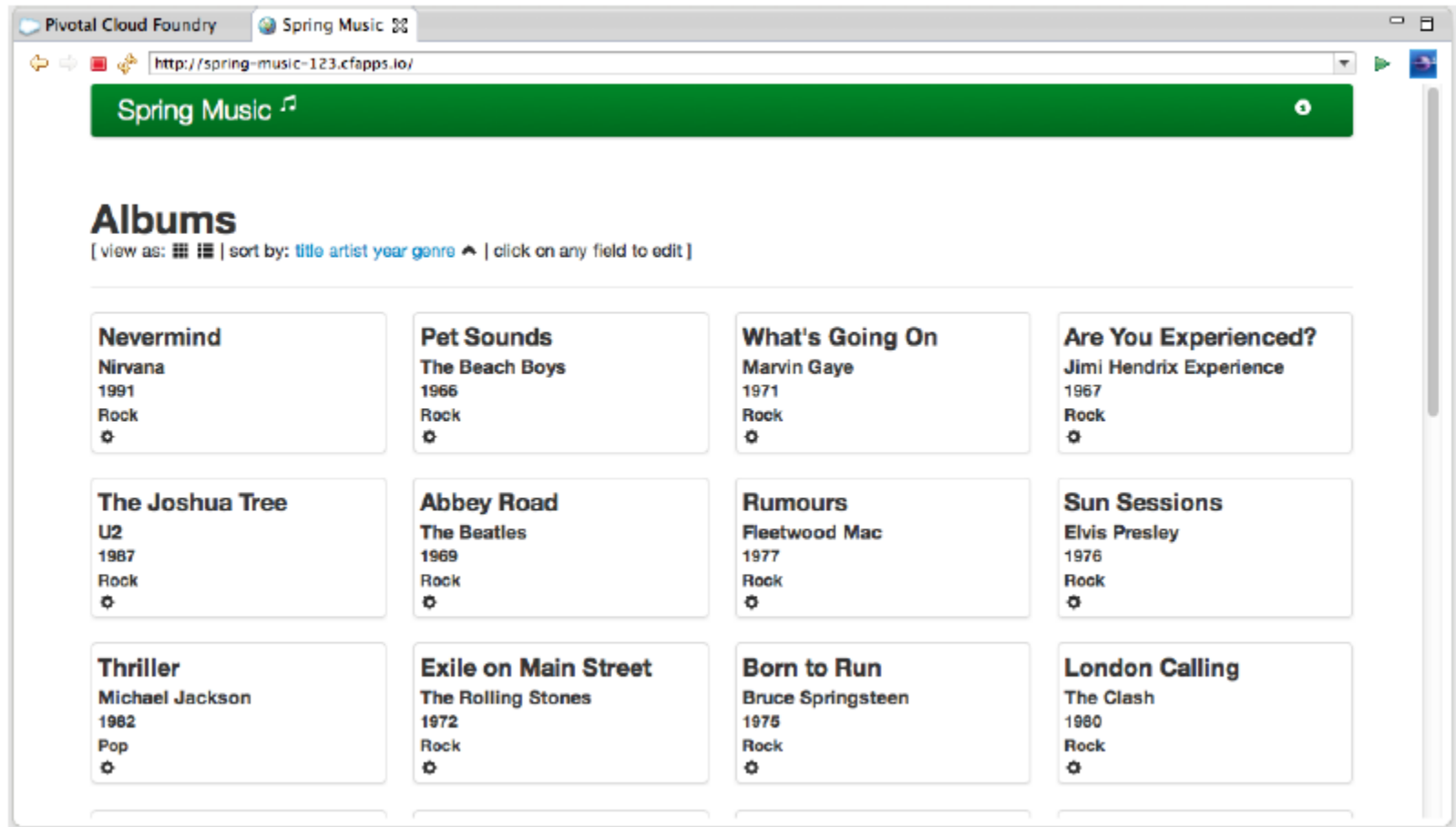
ID	Host	CPU	Memory	Disk	Uptime
0	10.10.81.12	7.7237...	440388M (5...	132M (1024M)	0h:14m:55s

Show deployed files in [Remote Systems View](#).

**Note URL:** added -123 to make it *unique*

# See your Application Running in Eclipse

- Eclipse pops up a browser window open at your URL
  - Or use the browser of your choice



# Managing Application Instances

# Cloud Foundry Dashboard

**General**

Name: spring-music [Started]

Mapped URLs: [spring-music-123.cfapps.io](#) 1

Instances: 2 2

Manifest: Save

**General (Application Restart Required)**

Memory Limit (MB): 512 Set 3

Environment Variables: Edit...

**Application Operations**

Start Stop Restart Update and Restart 4

**Application Services**

Name	Vendor	Plan	Version
------	--------	------	---------

**Instances**

ID	Host	CPU	Memory	Disk	Uptime
0	10.10.81.12	6.7973...	440460M (5...	132M (1024M)	3h:33m:57s
1	10.10.81.9	7.5758...	442472M (5...	132M (1024M)	0h:2m:20s

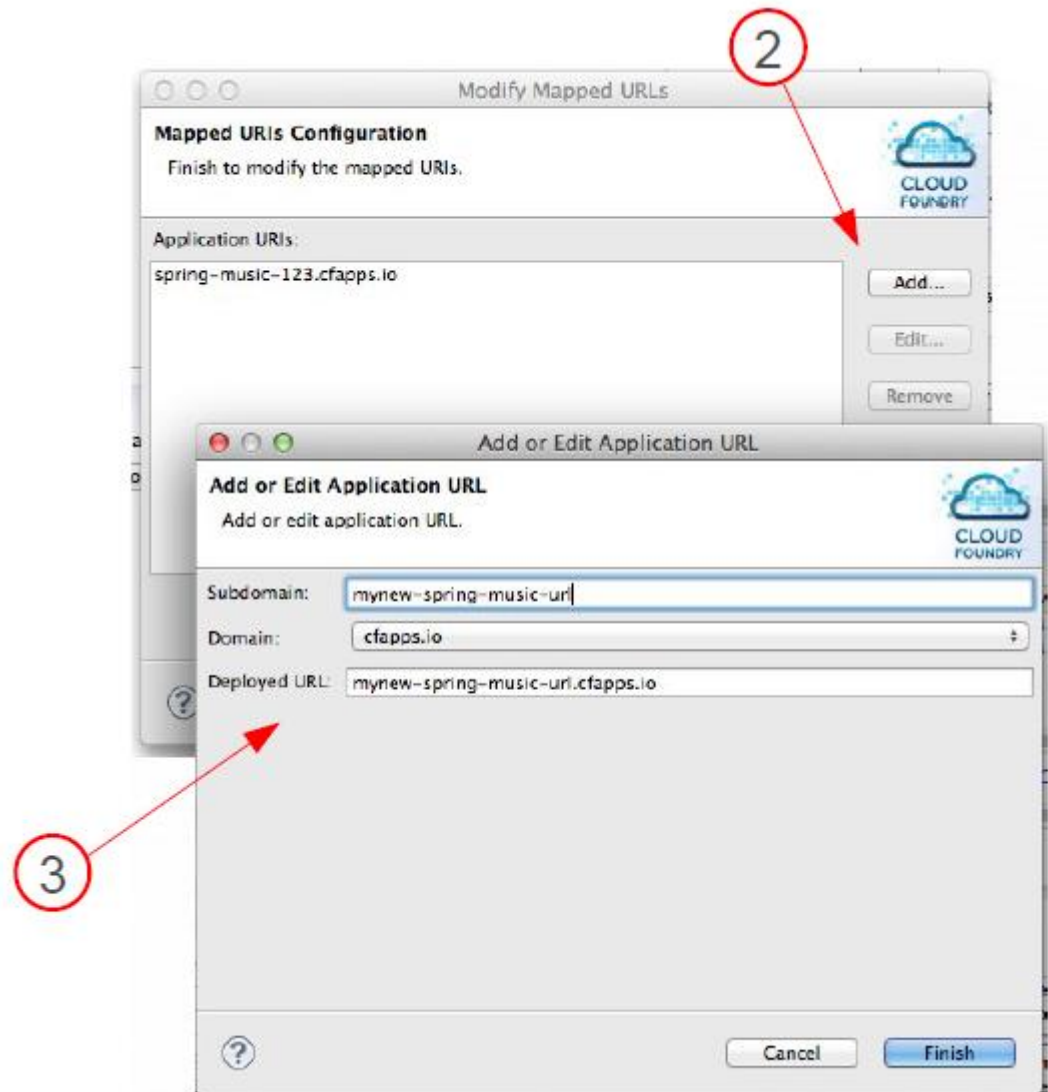
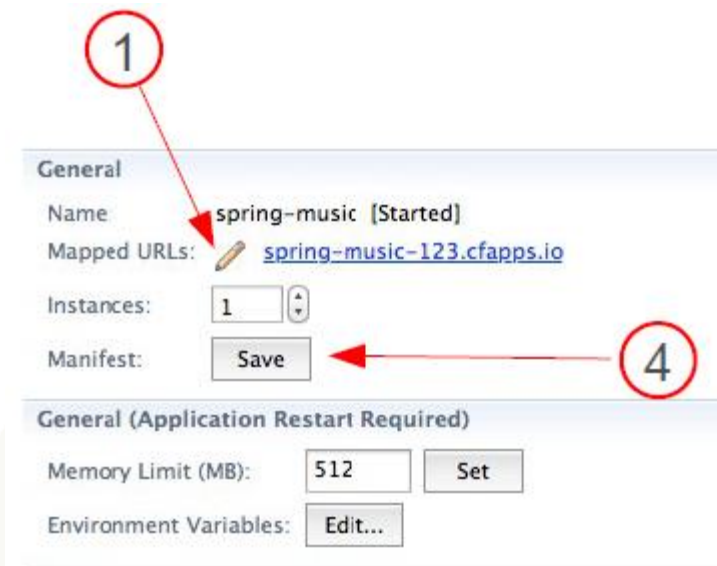
5

Show deployed files in [Remote Systems View](#).

- The right-side panel of Applications and Services tab
  - Below General
- Control your application
  1. Change mapped URL
  2. Add/remove instances
  3. Change memory
  4. Stop/start
  5. Monitor instances

# Change Mapped URL

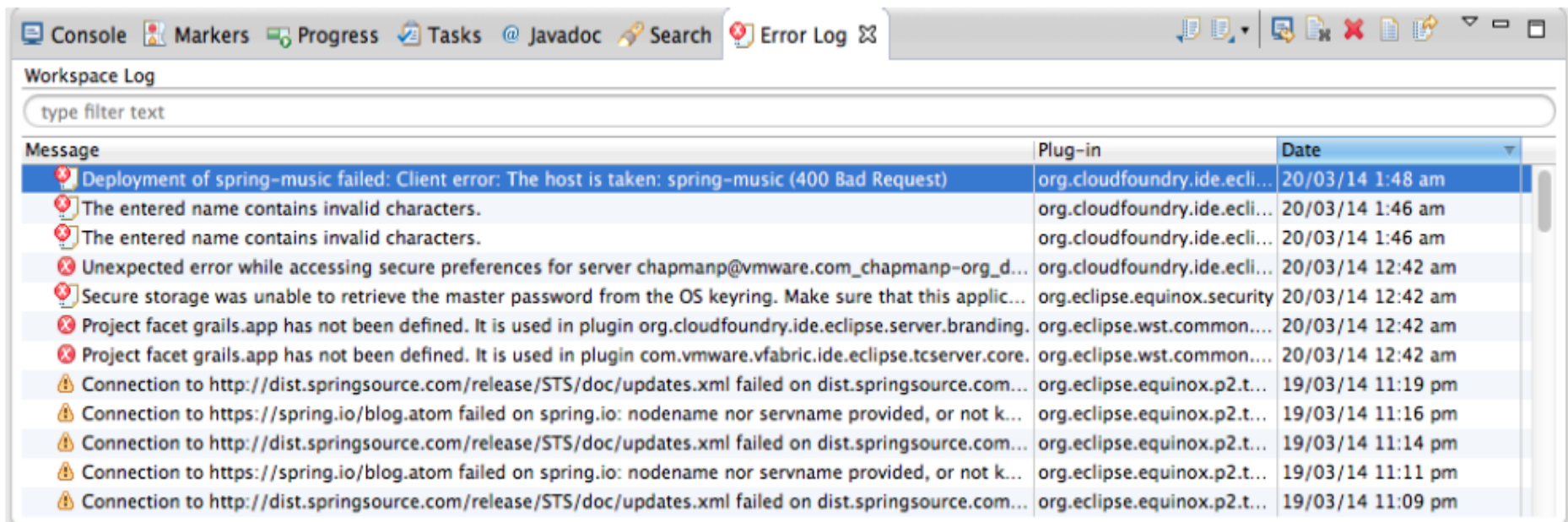
- Click pencil(edit) icon
- Can add, edit or remove URLs
- Save when done





# Choosing Your URL

- All applications mapped to cfapps.io domain
- Your URL must be unique
  - Get a Bad Request 400 if you try to use an existing URL



The screenshot shows the Eclipse IDE's Error Log window. The toolbar includes icons for Console, Markers, Progress, Tasks, Javadoc, Search, and Error Log. The 'Workspace Log' section has a filter text input. The error log table contains the following entries:

Message	Plug-in	Date
Deployment of spring-music failed: Client error: The host is taken: spring-music (400 Bad Request)	org.cloudfoundry.ide.ecli...	20/03/14 1:48 am
The entered name contains invalid characters.	org.cloudfoundry.ide.ecli...	20/03/14 1:46 am
The entered name contains invalid characters.	org.cloudfoundry.ide.ecli...	20/03/14 1:46 am
Unexpected error while accessing secure preferences for server chapmanp@vmware.com_chapmanp-org_d...	org.cloudfoundry.ide.ecli...	20/03/14 12:42 am
Secure storage was unable to retrieve the master password from the OS keyring. Make sure that this applic...	org.eclipse.equinox.security	20/03/14 12:42 am
Project facet grails.app has not been defined. It is used in plugin org.cloudfoundry.ide.eclipse.server.branding.	org.eclipse.wst.common....	20/03/14 12:42 am
Project facet grails.app has not been defined. It is used in plugin com.vmware.vfabric.ide.eclipse.tcserver.core.	org.eclipse.wst.common....	20/03/14 12:42 am
Connection to http://dist.springsource.com/release/STS/doc/updates.xml failed on dist.springsource.com...	org.eclipse.equinox.p2.t...	19/03/14 11:19 pm
Connection to https://spring.io/blog.atom failed on spring.io: nodename nor servname provided, or not k...	org.eclipse.equinox.p2.t...	19/03/14 11:16 pm
Connection to http://dist.springsource.com/release/STS/doc/updates.xml failed on dist.springsource.com...	org.eclipse.equinox.p2.t...	19/03/14 11:14 pm
Connection to https://spring.io/blog.atom failed on spring.io: nodename nor servname provided, or not k...	org.eclipse.equinox.p2.t...	19/03/14 11:11 pm
Connection to http://dist.springsource.com/release/STS/doc/updates.xml failed on dist.springsource.com...	org.eclipse.equinox.p2.t...	19/03/14 11:09 pm

# Instances

- By default one instance of your application runs up
  - Typically a Tomcat server
- To handle large loads you need multiple servers
  - Known as instances
  - Run behind load balancer
- How many instances do I need?
  - Design issue – covered later
- Modify as shown

**General**

Name: spring-music [Started]

Mapped URLs: [spring-music-123.cfapps.io](http://spring-music-123.cfapps.io)

Instances: 1

Manifest:

**General (Application Restart Required)**

Memory Limit (MB): 512

Environment Variables:

# Memory Allocation

- Define how much memory our process gets to run in
  - 512 is the default
    - Good for a typical application under test (1 or 2 users)
  - How much memory do I need in production
    - Another good question for later!
- Easy to configure ...
  - But the server has to be started

The screenshot displays the 'General' configuration tab for an application named 'spring-music'. The 'Name' field is 'spring-music [Started]' and the 'Mapped URLs' field is 'spring-music-123.cfapps.io'. The 'Instances' field is set to '1'. The 'Manifest' field has a 'Save' button. Below this, the 'General (Application Restart Required)' section shows the 'Memory Limit (MB)' set to '512' with a 'Set' button and 'Environment Variables' with an 'Edit...' button. A red arrow points from the text 'But the server has to be started' to the 'Instances' field, labeled with a circled '1'. Another red arrow points from the 'Set' button to the 'Memory Limit (MB)' field, labeled with a circled '2'.

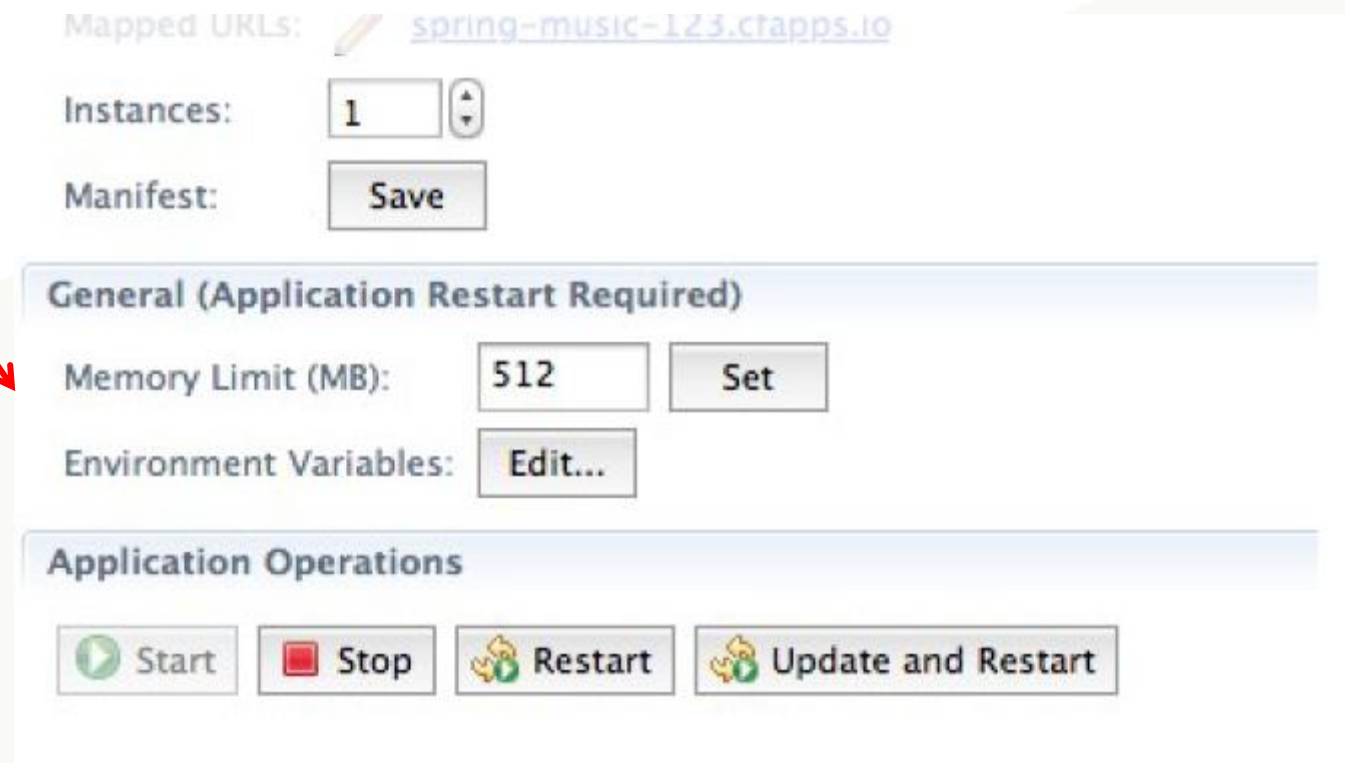
General	
Name:	spring-music [Started]
Mapped URLs:	<a href="#">spring-music-123.cfapps.io</a>
Instances:	1
Manifest:	<button>Save</button>


  


General (Application Restart Required)	
Memory Limit (MB):	512 <button>Set</button>
Environment Variables:	<button>Edit...</button>

# Stopping and Starting

- Normally this happens in the Servers view
  - Those buttons are grayed out
- Instead use the Dashboard



Mapped URLs:  [spring-music-123.cfapps.io](http://spring-music-123.cfapps.io)

Instances:  

Manifest:

**General (Application Restart Required)**

Memory Limit (MB):

Environment Variables:

**Application Operations**

A red arrow points from the text 'Instead use the Dashboard' to the 'Start' button in the 'Application Operations' section.

## Monitoring Instances

- The very bottom panel shows all your instances
  - Provides statistics
  - Not real-time
- To refresh
  - Click refresh icon on the application list

Application Services

Name	Vendor	Plan	Version

▼ Instances

ID	Host	CPU	Memory	Disk	Uptime
0	10.10.81.12	9.7046974537506...	440484M (5...	132M (1024M)	4h:44m:48s
1	10.10.81.9	7.5637518159724...	442484M (5...	132M (1024M)	1h:13m:11s

Show deployed files in [Remote Systems View](#).

# Summary: Cloud Foundry Dashboard

- In the Application and Services tab
  - Configure your application (below general on right-side)
  - Options
    - Modify mapped URL
    - Change number of instances
    - Change the amount of memory allocated
    - Start and stop the application
    - Monitor instances
  - You may have noticed we missed two options (later)
    - Add or remove services
    - Set environment variables

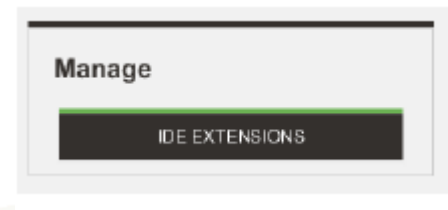
# Appendices

- Describe installation of Cloud Foundry plug-in into
  - Appendix A : STS
  - Appendix B : Standard Eclipse
- For full details see:
  - <http://docs.cloudfoundry.org/devguide/deploy-apps/sts.html>



# Appendix A: Installing CF Plug in into STS

- Click Spring leaf icon in STS
  - Displays dashboard
- At bottom right under Manage
  - Click “IDE Extensions”

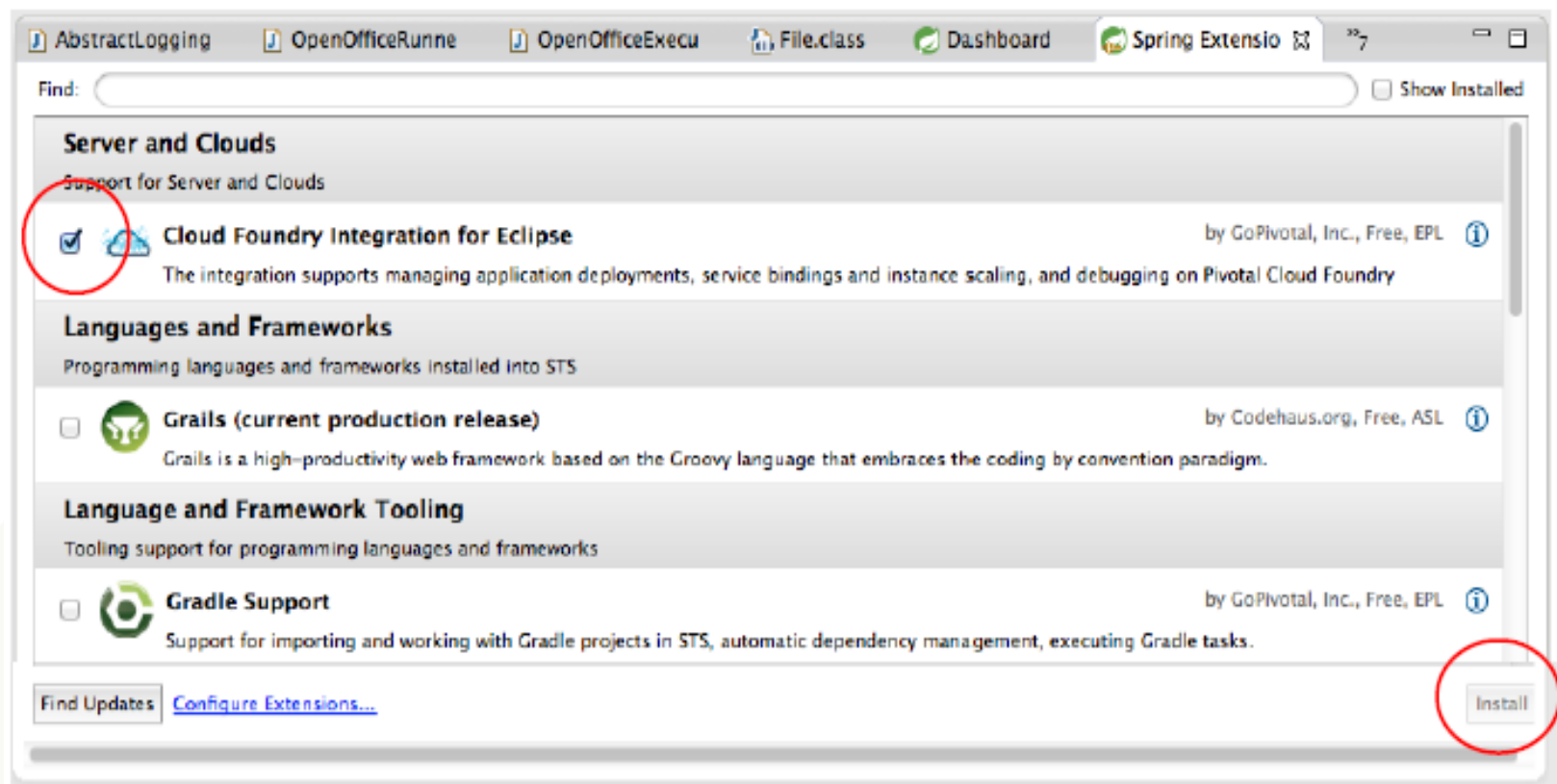


See : <http://docs.cloudfoundry.org/devguide/deploy-apps/sts.html#install-to-sts>



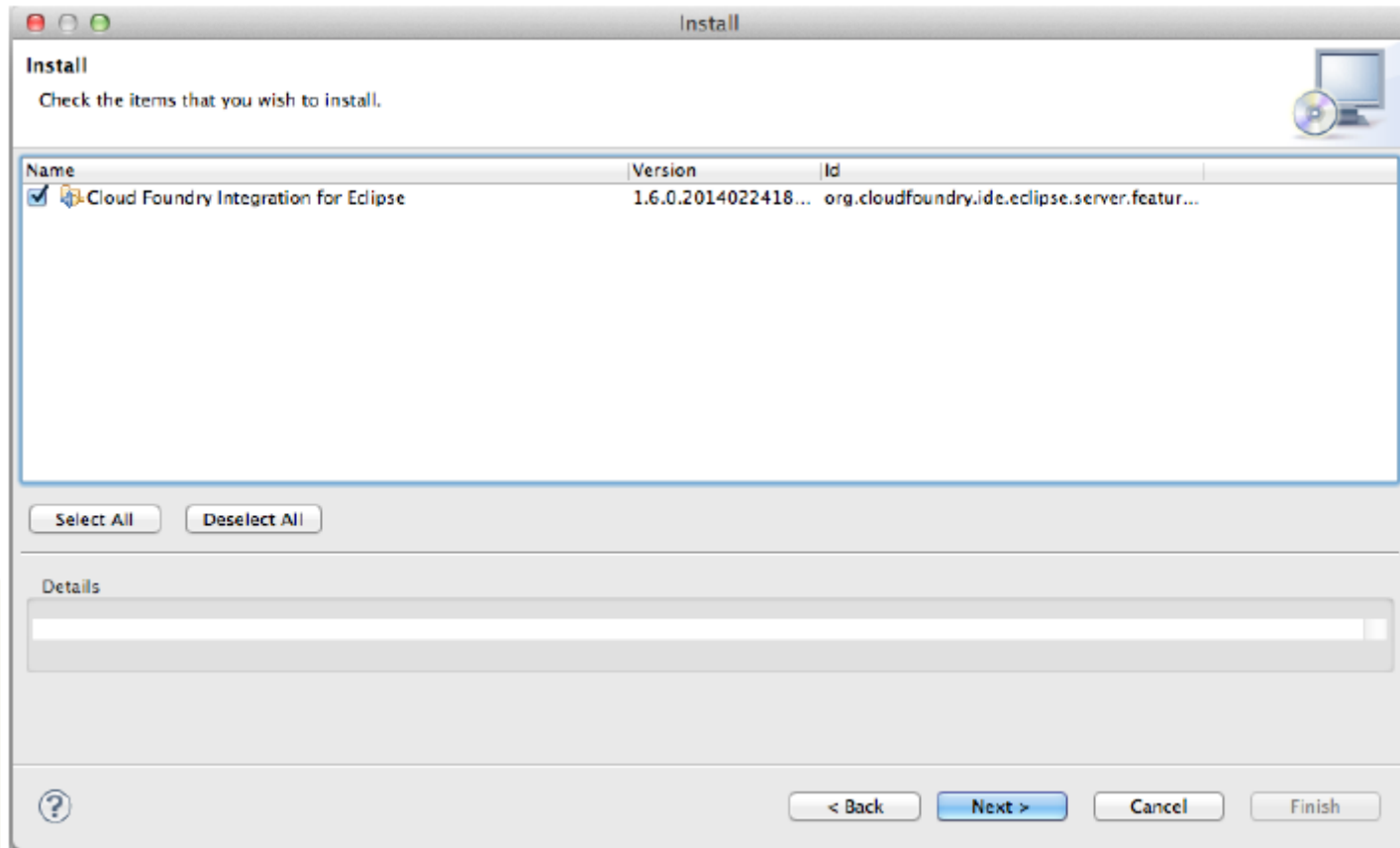
# See Cloud Foundry Integration for Eclipse

- Select checkbox and click install button
  - If not listed, enter “cloud foundry” in Find and hit enter



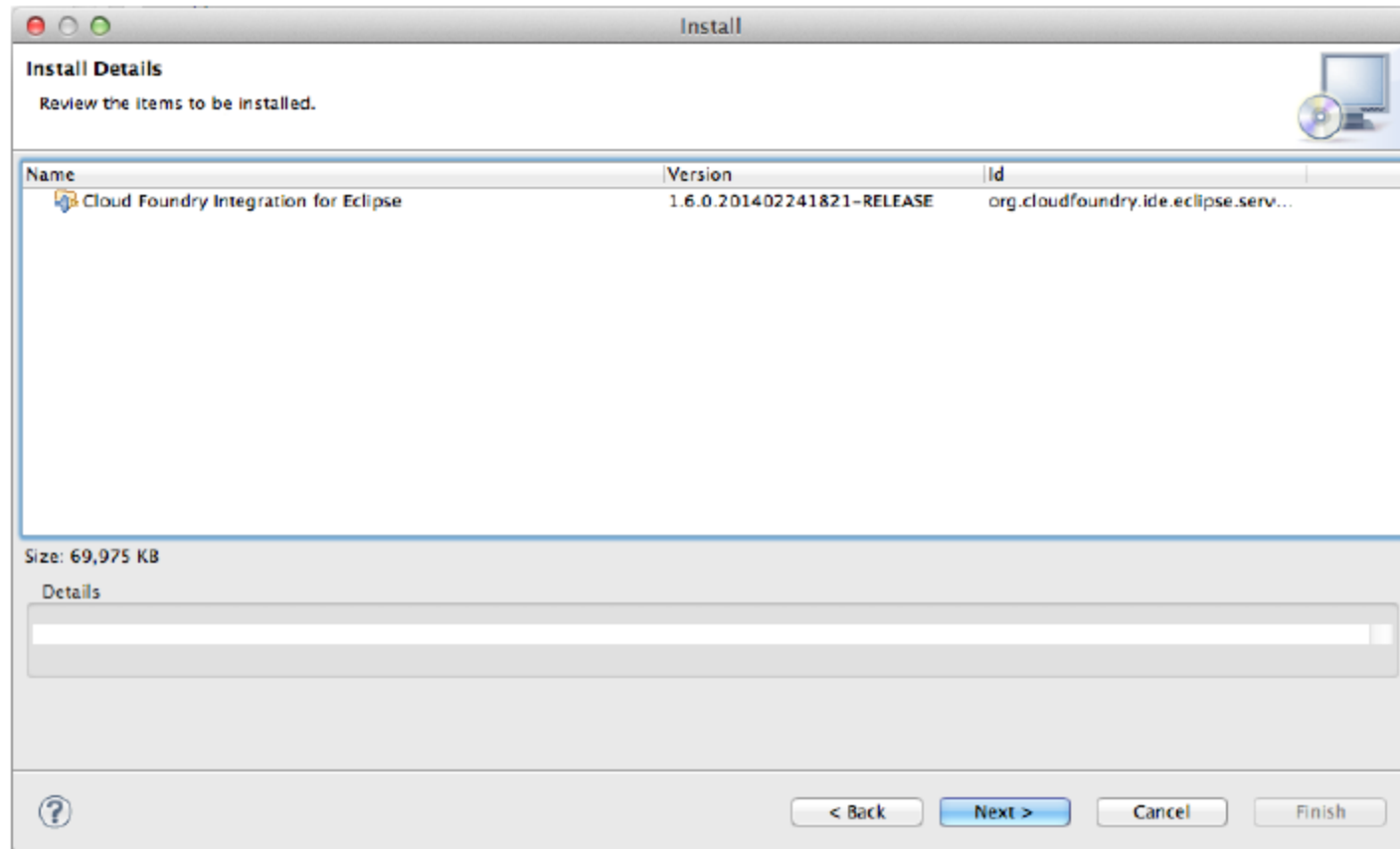
# Runs up a wizard

- Click next



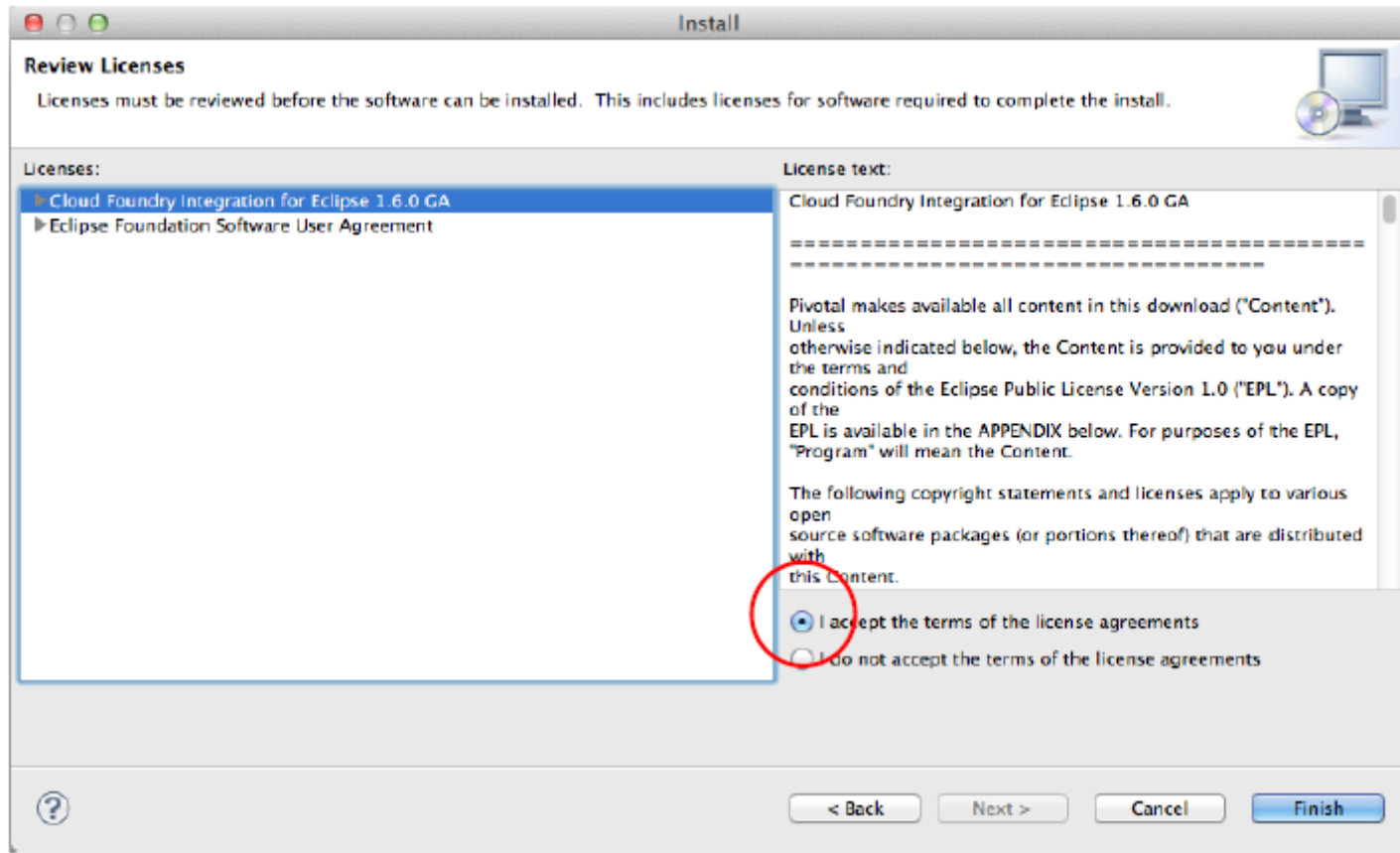
# Wizard – Step 2

- Click next again



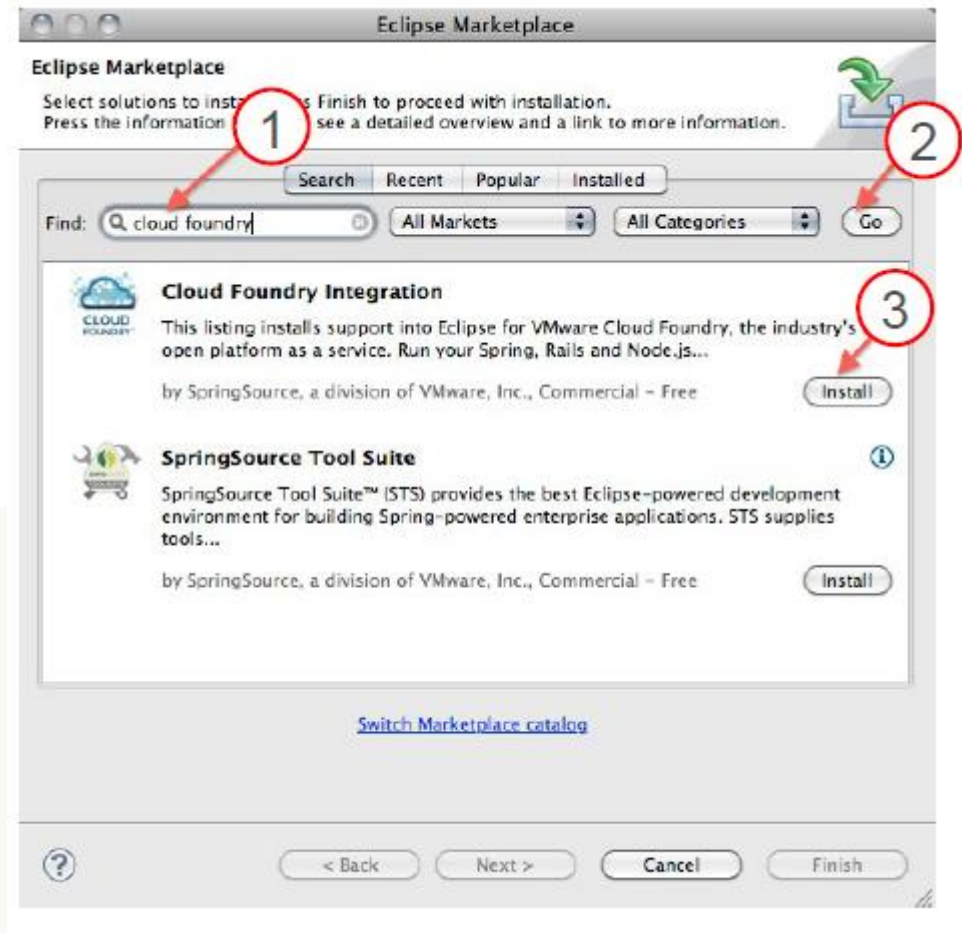
# Wizard – Step 3

- Accept license agreement, and the installer will run



# Appendix B : Installing into Eclipse

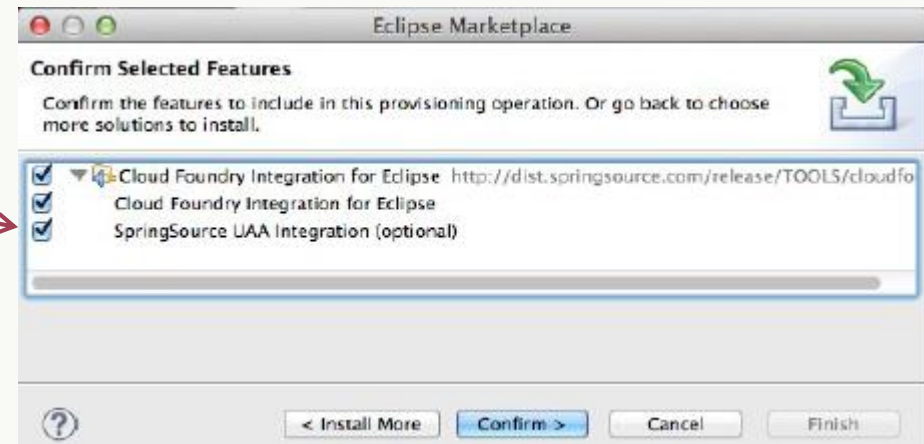
- For Help menu
  - Eclipse Marketplace
- Enter “Cloud Foundry” into find box
- Click “Go”
- Click “Cloud Foundry Integration” *Install* button



See : <http://docs.cloudfoundry.org/devguide/deploy-apps/sts.html#install-to-eclipse>

# Confirm Selected Features

- Popup window lists what will be installed
  - “Cloud Foundry Integration for Eclipse”
  - “SpringSource UAA Integration” (optional)
    - Reports tool usage data, anonymously
    - Helps us track usage of free software
    - Deselect to stop plug in usage statistics being sent
  - Click Confirm.



# Last Few Steps

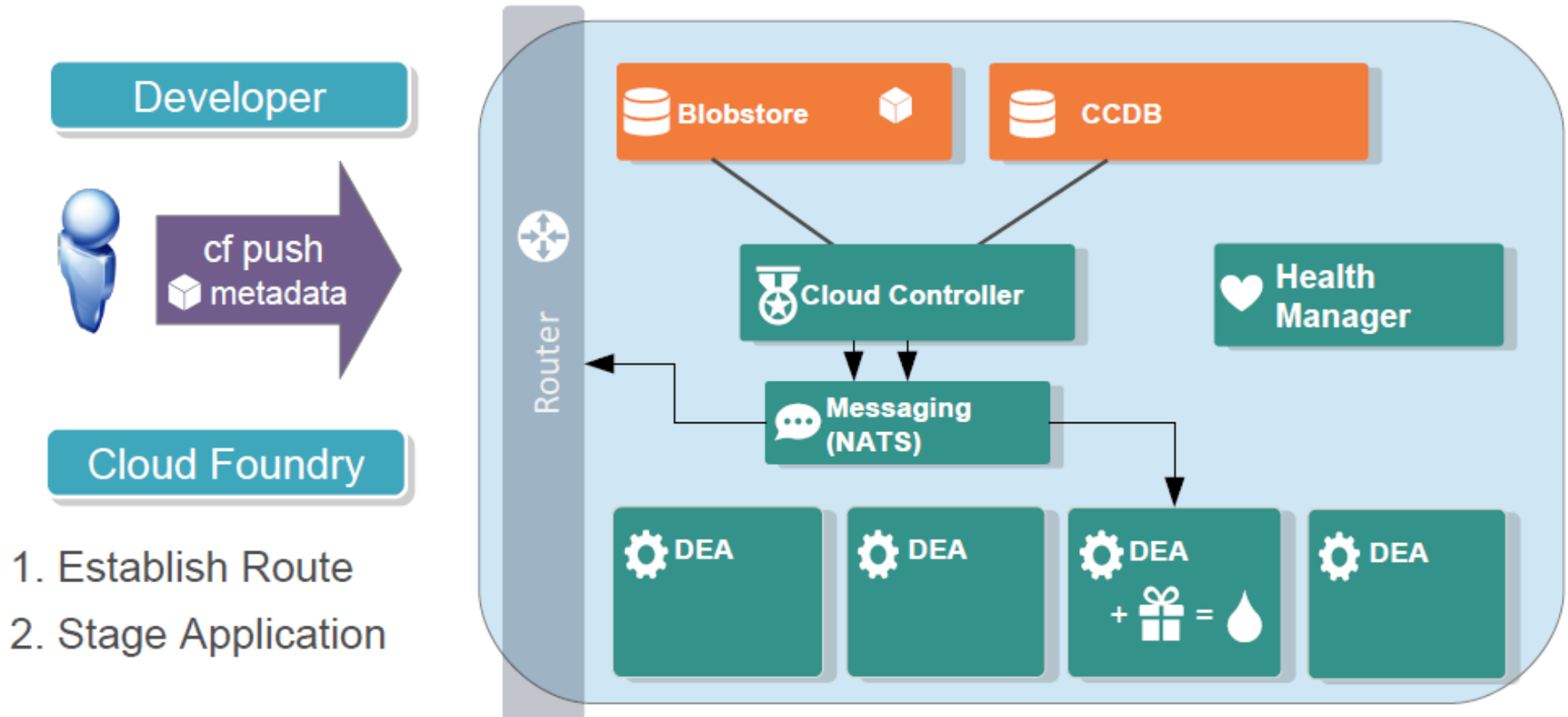
- Accept the license agreement
- Click finish
- Installer runs (takes a while)
- Eventually you are asked to restart Eclipse



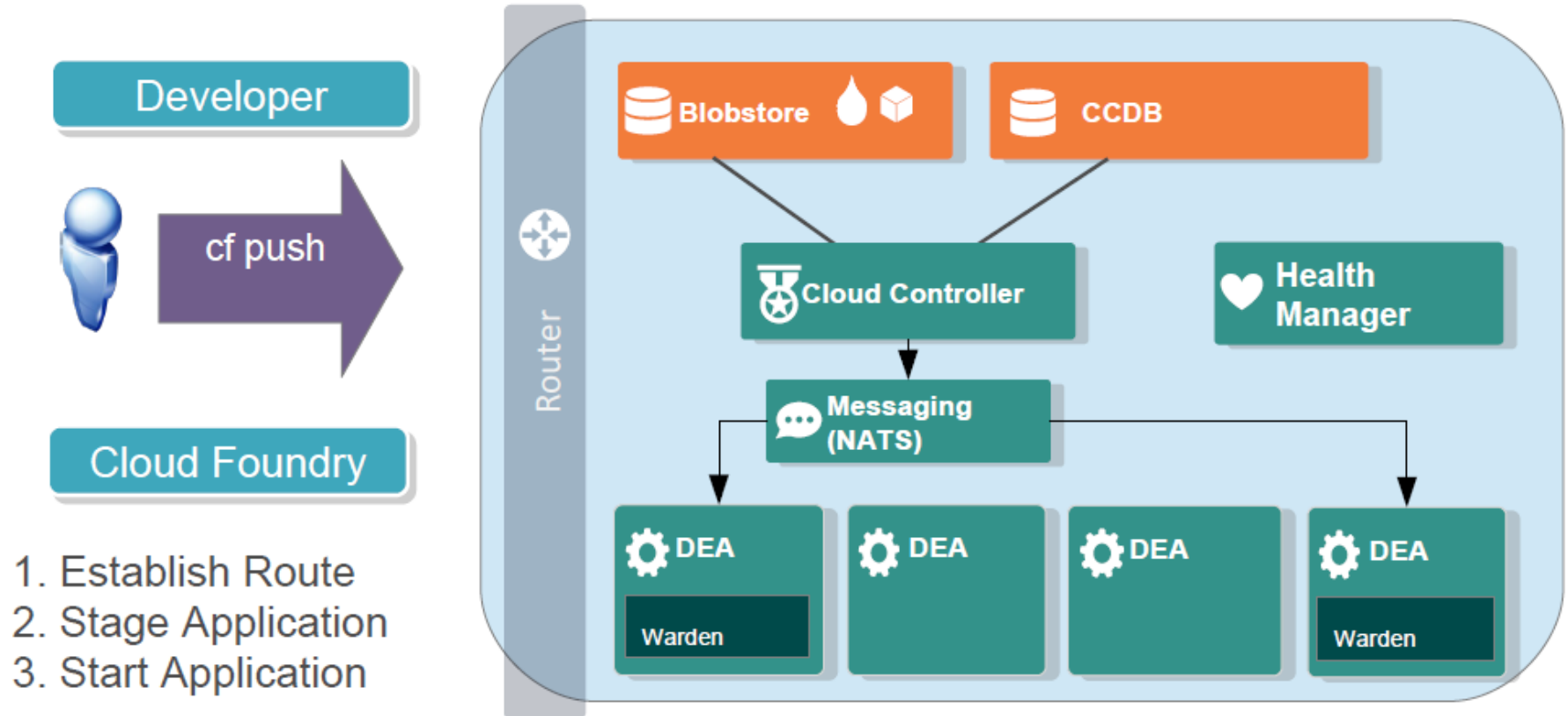
# How Deployment Happens



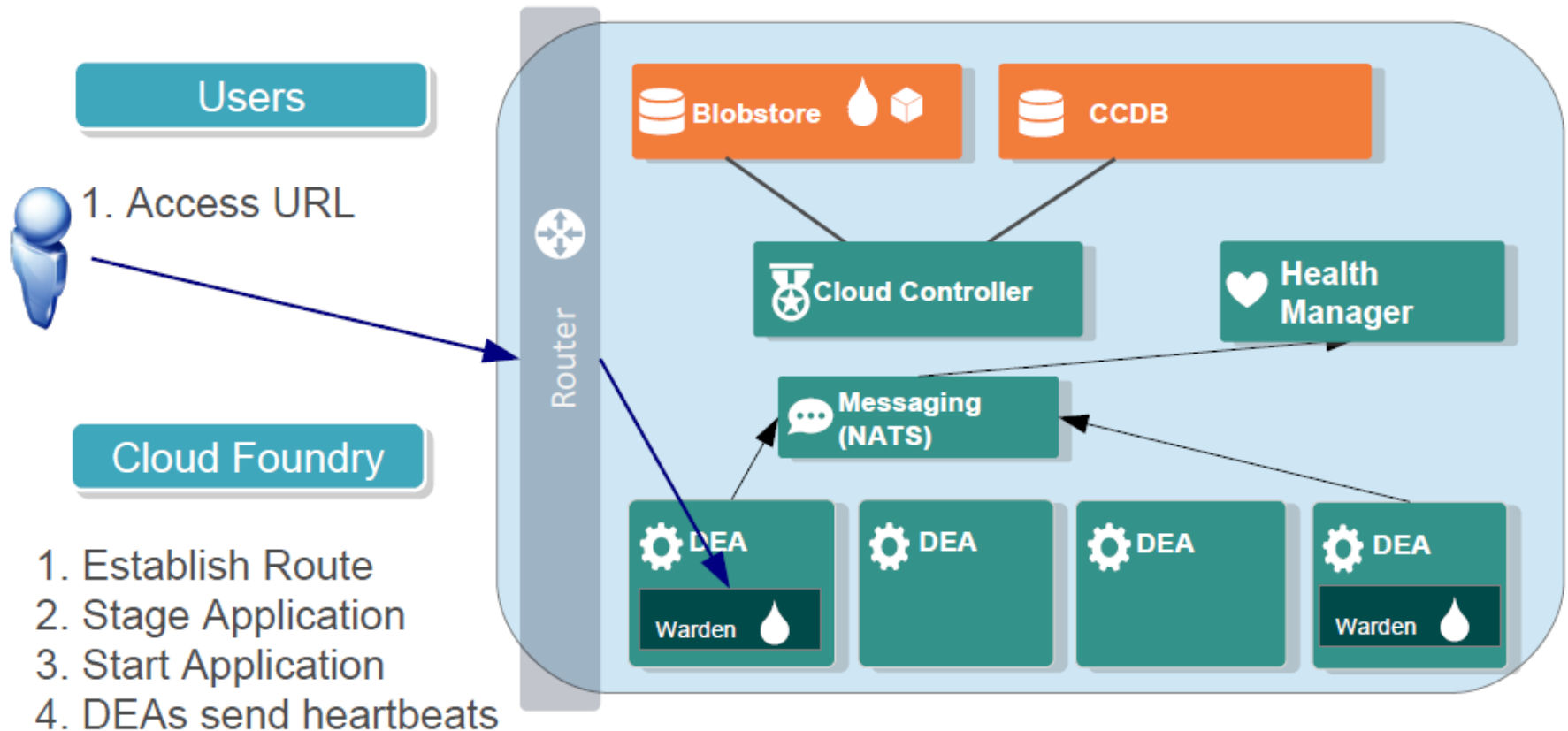
# Deploying App to Cloud Foundry (Push & Staging)



# Deploying App to Cloud Foundry (Starting)

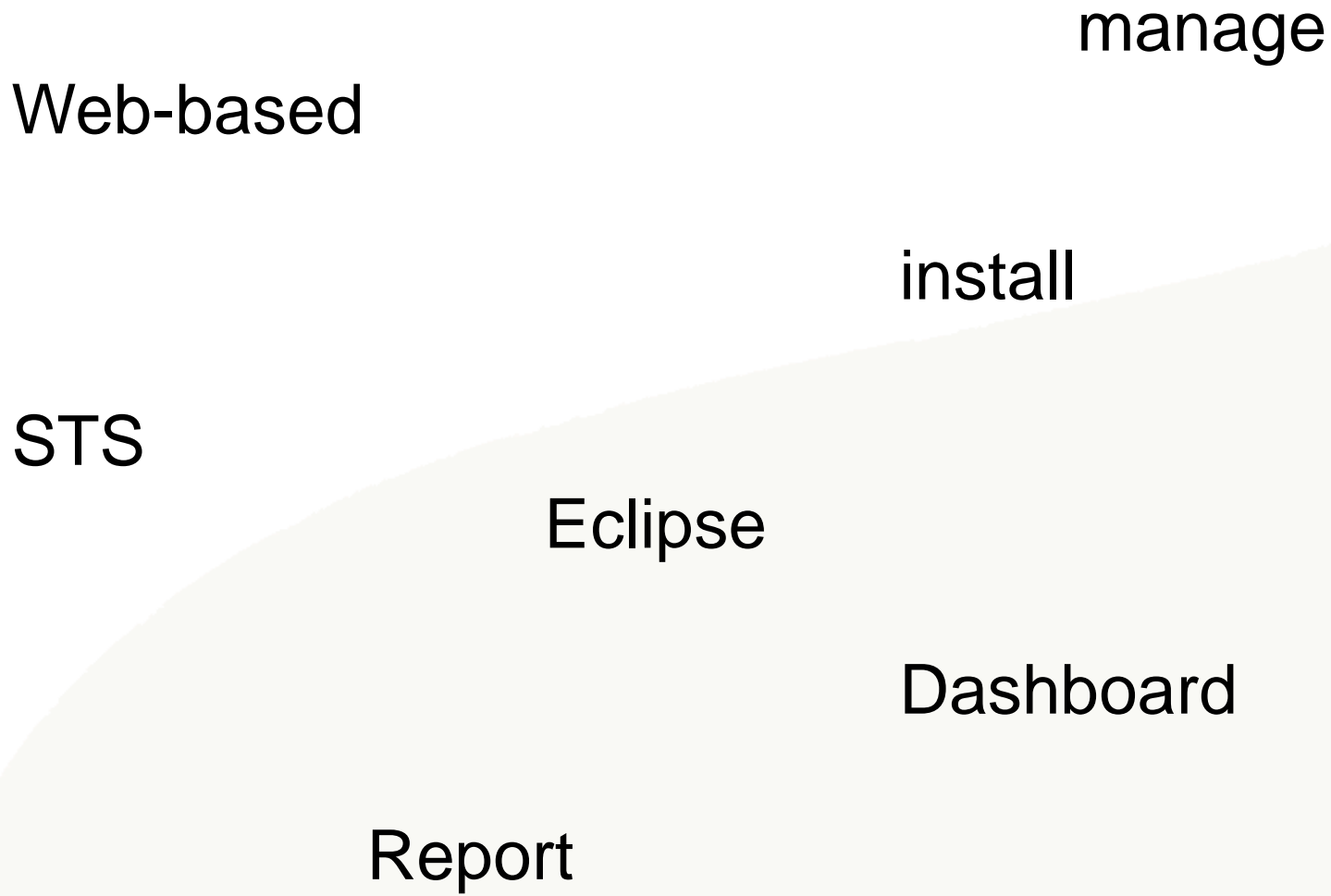


# Deploying App to Cloud Foundry (Health Management & Running)



# Recap

---



People matter, results count.



## About Capgemini

With more than 130,000 people in 44 countries, Capgemini is one of the world's foremost providers of consulting, technology and outsourcing services. The Group reported 2012 global revenues of EUR 10.3 billion.

Together with its clients, Capgemini creates and delivers business and technology solutions that fit their needs and drive the results they want. A deeply multicultural organization, Capgemini has developed its own way of working, the Collaborative Business Experience™, and draws on Rightshore®, its worldwide delivery model.



[www.capgemini.com](http://www.capgemini.com)

