Bluemix Hands-On Workshop

Section 4- DevOps Services

Version:	4
Last modification date:	6-Oct-14
Owner:	IBM Ecosystem Development

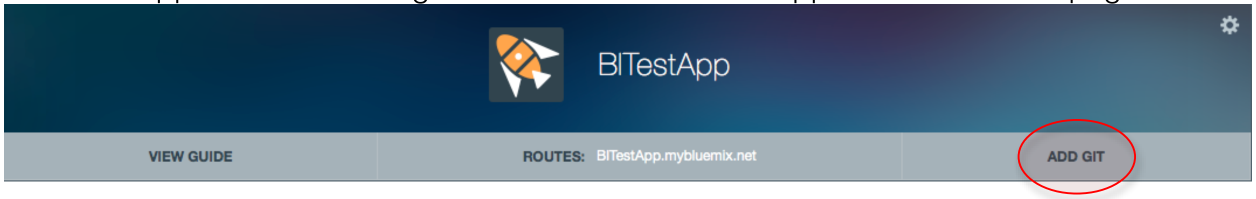
Table of Contents

Exercise 4.a - Bluemix integration with DevOps Services

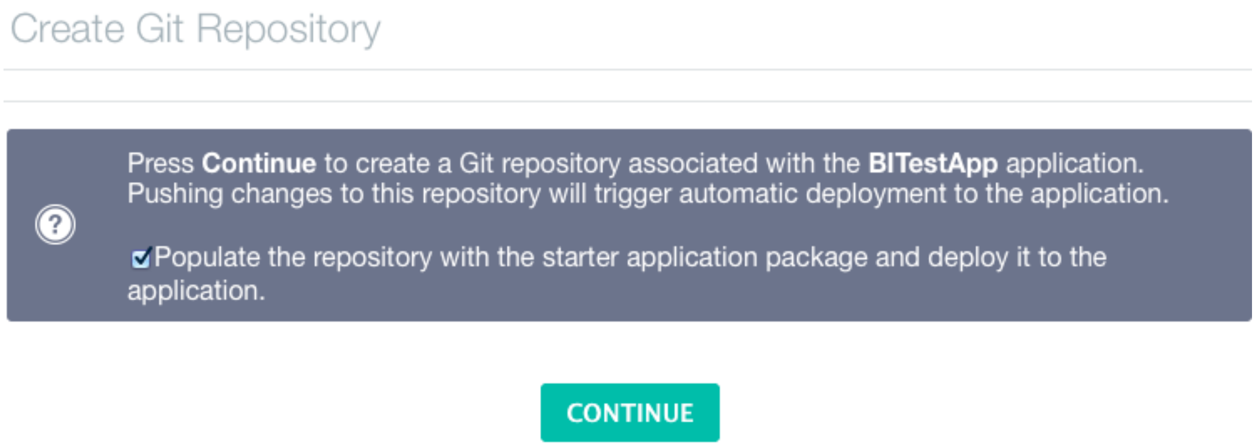
This exercise looks at how Bluemix and DevOps Services work together

Log into Bluemix http://bluemix.net and deploy the Java Cache Web Starter boilerplate.

Once the application is running select 'ADD GIT' on the Application Overview page



add your DevOps Service details and select 'Sign In'. Ensure you leave the option to populate the repository selected then select 'CONTINUE'

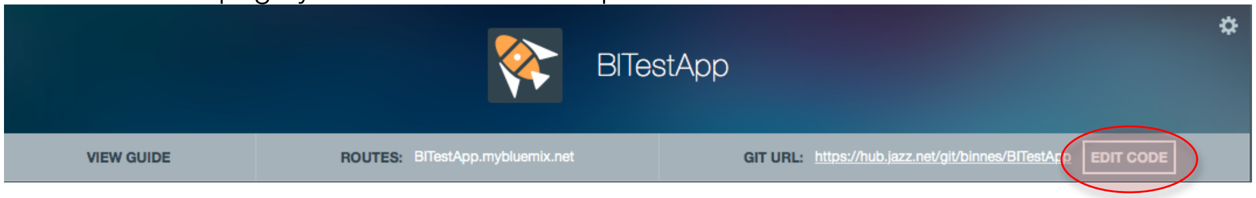


Create Git Repository

Press **Continue** to create a Git repository associated with the **BITestApp** application. Pushing changes to this repository will trigger automatic deployment to the application.

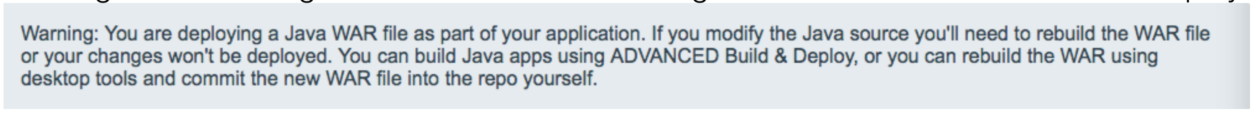
☑ Populate the repository with the starter application package and deploy it to the application.

In the overview page you will now see the option to 'EDIT CODE'


BITestApp
VIEW GUIDE
ROUTES: BITestApp.mybluemix.net
GIT URL: https://hub.jazz.net/git/binnes/BITestApp
EDIT CODE

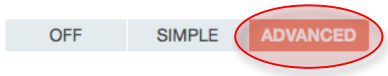
select 'EDIT CODE' – you should be taken into DevOps Services with the sample application code imported and ready to work with.

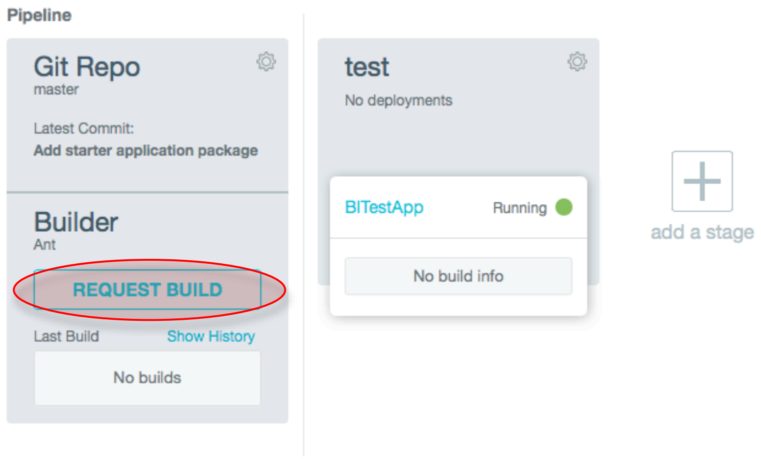
Select 'BUILD & DEPLOY' by default the SIMPLE option is chosen, but you will see a warning about building Java WAR files and needing to use the ADVANCED Build & Deploy



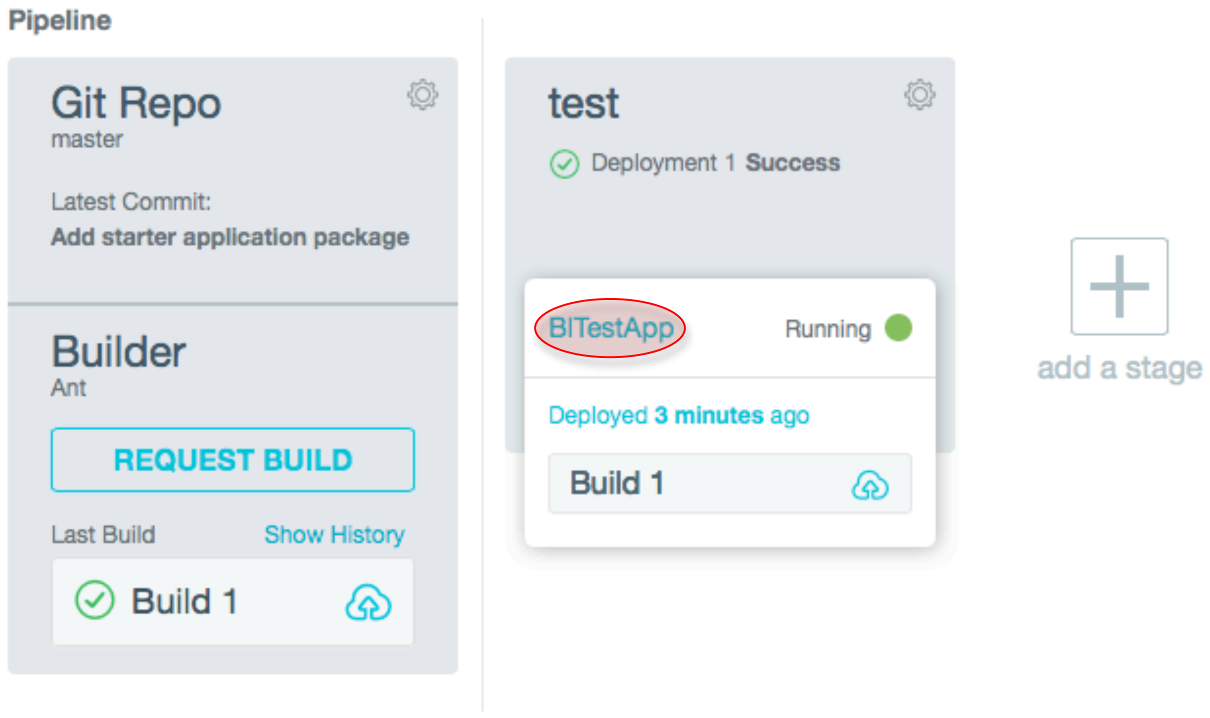
Warning: You are deploying a Java WAR file as part of your application. If you modify the Java source you'll need to rebuild the WAR file or your changes won't be deployed. You can build Java apps using ADVANCED Build & Deploy, or you can rebuild the WAR using desktop tools and commit the new WAR file into the repo yourself.

Select 'ADVANCED' you will see the project will be setup to use Ant to build the project



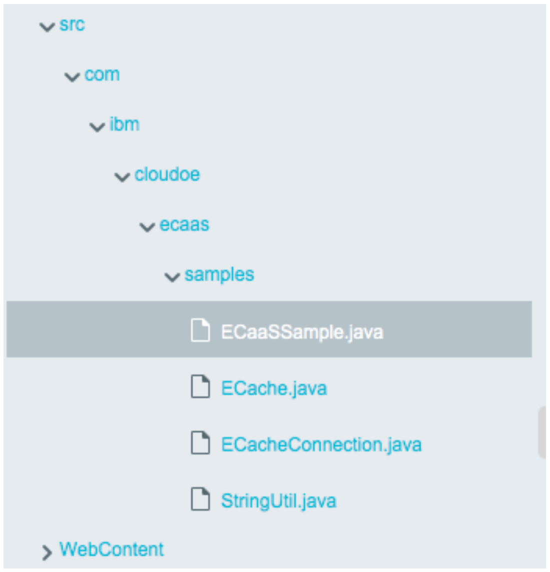


You can now modify the code and deploy to Bluemix from DevOps Services. To test this select the 'REQUEST BUILD' in the Builder section of the pipeline. You will see the Build Queued, then the build run and finally complete. Once the build has complete a deploy operation should be queued, then run. You should see the deployment running then Success – to launch the application you can click on the app name

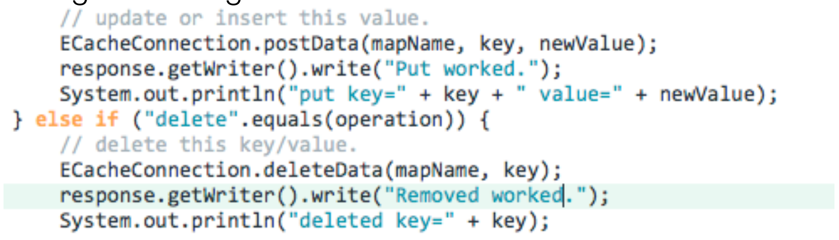


Exercise 4.b – working in DevOps Services

In DevOps Services switch to the 'EDIT CODE' section so we can edit the code.

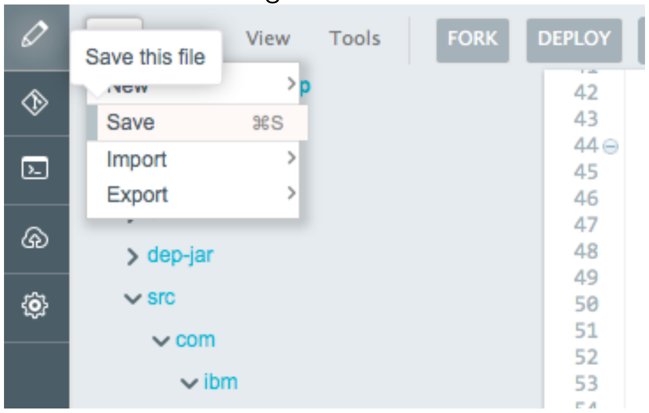


Open the ECaaSSample.java file and scroll to the bottom to find the 2 strings that are displayed on a successful put ("Put successfull.") or delete ("Remove successfull.") and change the strings.



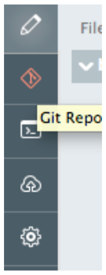
```
    // update or insert this value.
    ECacheConnection.postData(mapName, key, newValue);
    response.getWriter().write("Put worked.");
    System.out.println("put key=" + key + " value=" + newValue);
} else if ("delete".equals(operation)) {
    // delete this key/value.
    ECacheConnection.deleteData(mapName, key);
    response.getWriter().write("Removed worked.");
    System.out.println("deleted key=" + key);
```

now save the changes

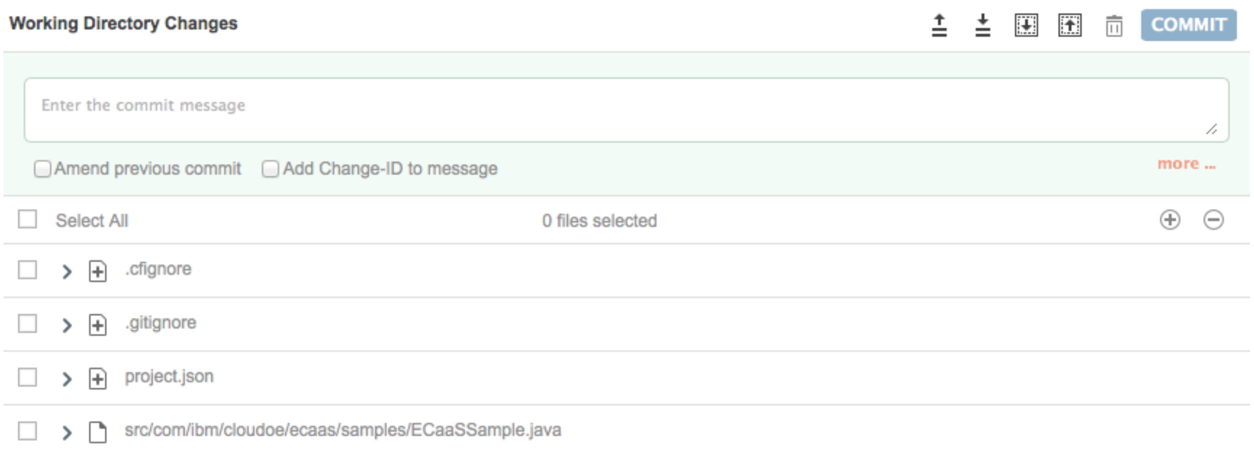


To get the changes pushed to Bluemix we need to commit the changes to the Git repository.

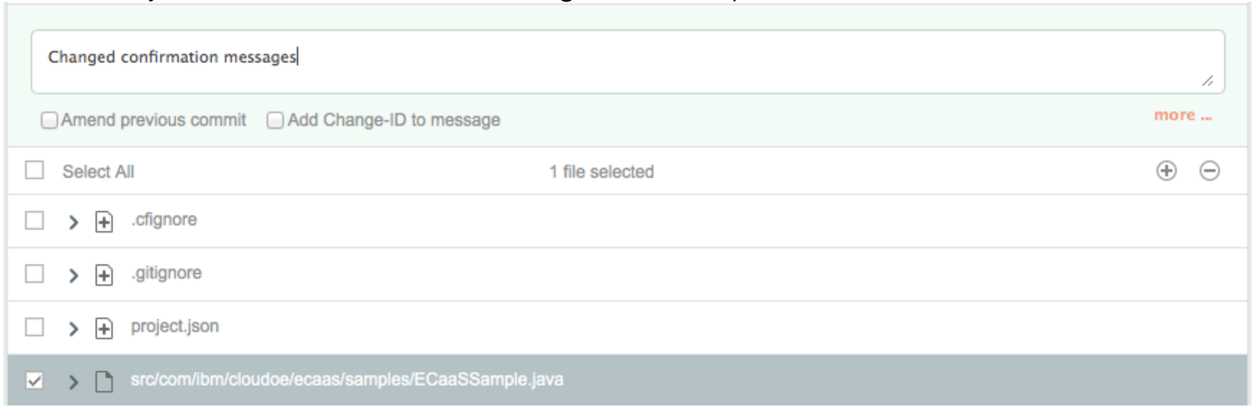
To work with the Git repository switch into the Git section



The files that have local changes are shown in the Changed Files section

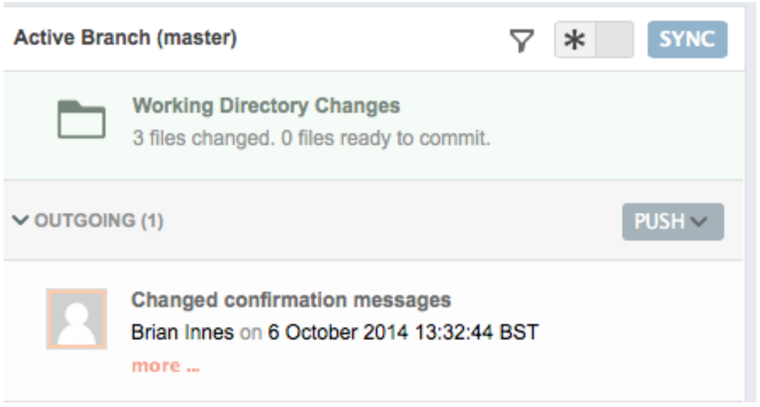


Select the java file, add a commit message and then press 'COMMIT'



Changed confirmation messages

The commit was made to the local branch – the builder works from the remote branch, so we need to push out changes back to the remote. Expand the 'PUSH' dropdown and select 'Push All'


Active Branch (master)
SYNC
Working Directory Changes
3 files changed. 0 files ready to commit.
OUTGOING (1)
PUSH
Changed confirmation messages
Brian Innes on 6 October 2014 13:32:44 BST
more ...

If you quickly jump to the BUILD & DEPLOY screen you will see a build has automatically been started, which will then be automatically deployed to Bluemix if successful.

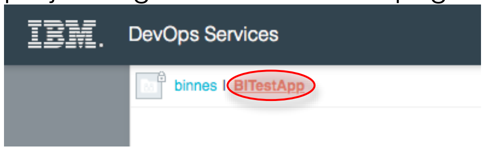
Once the deploy has completed test the application to verify the code changes are now running.

Note: if a build fails you can click on the build in the builder and it will take you to a detailed screen of build history, where you can get access to the logs, files and details of changes included in the builds. This can help determine why the build failed.

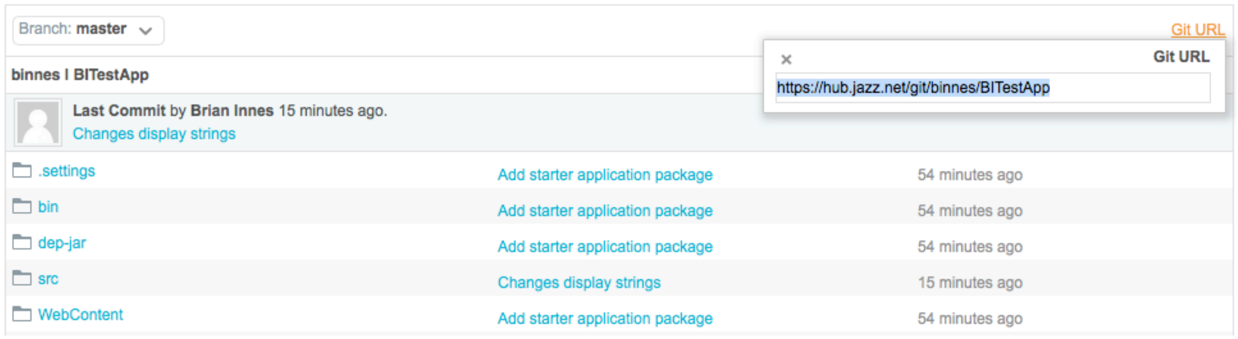
Exercise 4.c – DevOps Services integration with Eclipse

Eclipse can work with Git projects.

To add a DevOps Services project to Eclipse get the git URL from your DevOps Services project – go to the overview page by selecting the project name from the top left of screen



then select the 'Git URL' link



https://hub.jazz.net/git/binnes/BITestApp

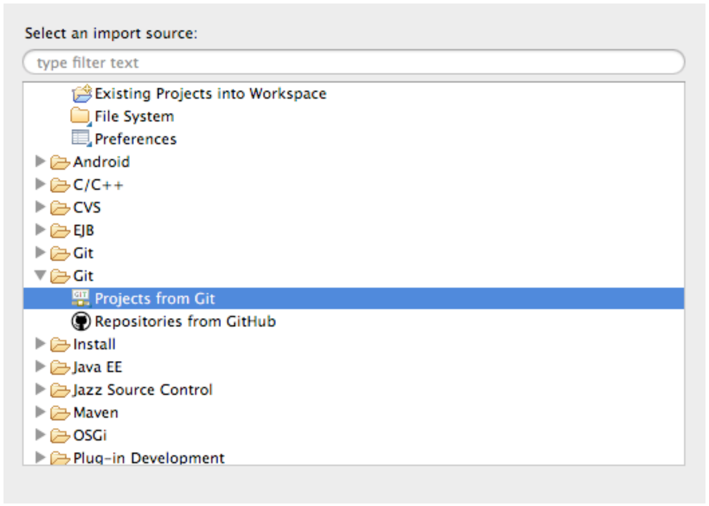
binnes | BITestApp

Last Commit by **Brian Innes** 15 minutes ago.
Changes display strings

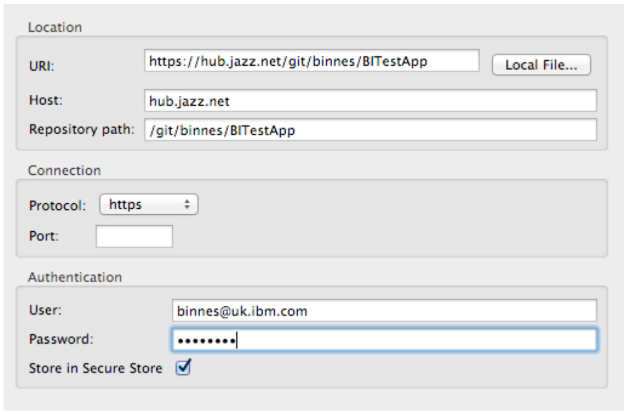
.settings	Add starter application package	54 minutes ago
bin	Add starter application package	54 minutes ago
dep-jar	Add starter application package	54 minutes ago
src	Changes display strings	15 minutes ago
WebContent	Add starter application package	54 minutes ago

Copy the link Ctrl-C.

In Eclipse select File -> Import then when the import dialog is displayed select 'Projects from Git'



on the next page select 'Clone URI' then on the next page past the Git URL in the to Location box. Enter your email and password (used to log into DevOps Services)



Location

URI: https://hub.jazz.net/git/binnes/BITestApp

Host: hub.jazz.net

Repository path: /git/binnes/BITestApp

Connection

Protocol: https

Port:

Authentication

User: binnes@uk.ibm.com

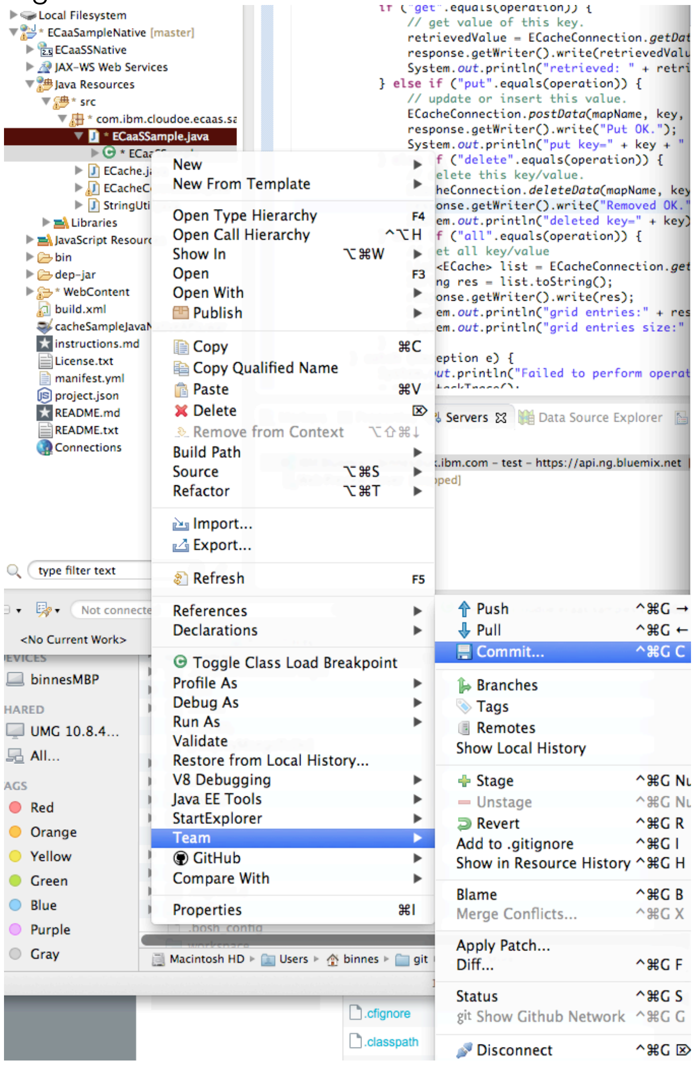
Password: ••••••••

Store in Secure Store ✓

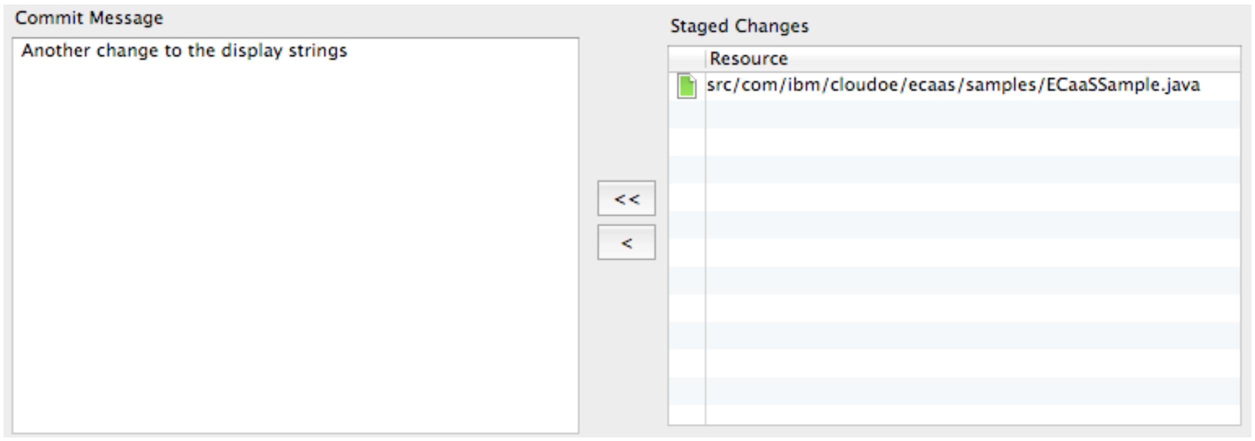
leave the Branch selection as master then on the next confirm the directory you want to use for the local storage location. On the select wizard page select 'Import exiting projects'. Confirm the ECaaSampleNative project is shown on the last page then select Finish.

Open up the ECaaSSample.java file and modify the display strings again, as we did using the DevOps Services editor in the previous exercise, then save the file. We now want to push this change back to DevOpsServices.

Right click the file -> Team 0 -> Commit



```
        // get value of this key.
        retrievedValue = ECacheConnection.getDat
        response.getWriter().write(retrievedValu
        System.out.println("retrieved: " + retri
    } else if ("put".equals(operation)) {
        // update or insert this value.
        ECacheConnection.postData(mapName, key,
        response.getWriter().write("Put OK.");
        System.out.println("put key=" + key + "
```



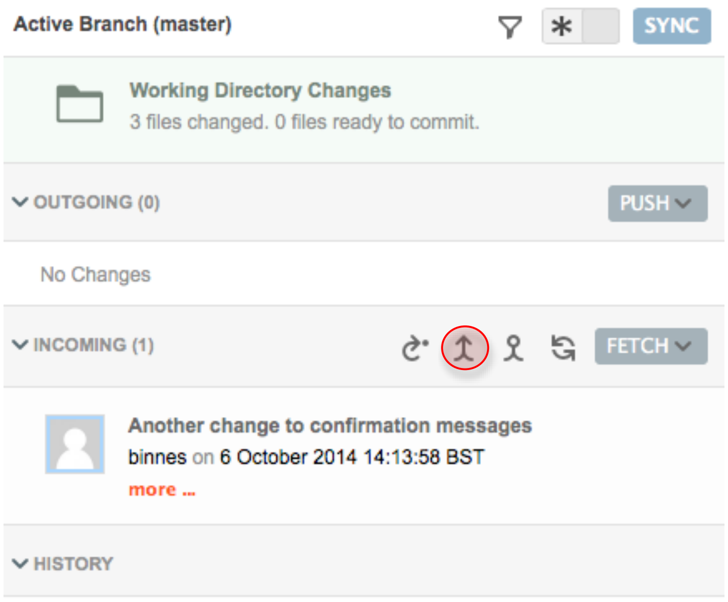
Another change to the display strings

src/com/ibm/cloudoe/ecaas/samples/ECaaSSample.java

Add a commit message and move the file to the Staged Changes window. Then press Commit.

This has added the change to the local branch, so we now need to push the change to the remote branch on DevOps Services. Right-click the project name (ECaaSampleNative) and select Team -> Push to Upstream. This will push the changes back to the master on DevOps, but will not initiate an autodeploy. In DevOps Services in the BUILD & DEPLOY section request a build – once it has built and deployed the code test it to ensure the changes from Eclipse are running.

If you open up the Editor in DevOps services you will not see the changes to the code made in Eclipse. However, if you go into to the Git section you will now see an incoming change



3 files changed. 0 files ready to commit.

No Changes

Another change to confirmation messages
binnes on 6 October 2014 14:13:58 BST

Select Merge (⤒) to bring the changes into the local branch in DevOps – after the merge you will see the changes from Eclipse in your DevOps editor.

Note, there was no automatic build and deploy triggered – this only happens when changes are pushed from within DevOps Services.

Exercise 4.d – Modify the readme.md for your project

The readme.md is shown in the overview window of your DevOps project. It is useful to make the content of this file describe the project.

Make changes to the readme.md testing out some of the formatting options - this is in md format - :

```
#MD Syntax

#Header 1

Header 1
========


##Header 2

Header2
-------



###Header3
####Header4
#####Header5
######Header6


***

using \*

* List item1
* List item2
* List item3

using \+

+ List item4
+ List item5
+ List item6

using numbers:

1. List item7
2. List item8
3. List item9
```

```
***



> Indented text
can also pans multiple lines

>Nested block text also works
>> Double indented text
can be used, *in addition* to other markup

>Like this

***

10. This is a list item

10\. This is not a list item

***



The following create a horizontal line (3 or more
astericks, dashes or underscores with optional spaces
between them)

Option 1 : \* \* \*
* * *
Option 2 : \*\*\*
***
Option 3 : \-\-\-

---
Option 4 : \_\_\_\_\_

_______


Emphasis *strong* or _strong_

Strong **strong** or __strong__

Code `code`

***
```

```
##Inine images and links:

![Alt Text](/path/to/image.jpg)
![Alt Text](/path/to/image.jpg "Optional title")


[BBC](http://bbc.co.uk)
[](http://bbc.co.uk "Optional Title")




***


Markdown provides backslash escapes for the following
characters:

\\   backslash
\`   backtick
\*   asterisk
\_   underscore
\{\}  curly braces
\[\]  square brackets
\(\)  parentheses
\#   hash mark
\+   plus sign
\-   minus sign (hyphen)
\.   dot
\!   exclamation mark



Note : 2 space characters at the end of a line will cause a
line break.  Otherwise no line break will appear in the
output


this line has 2 spaces at the end  
this line does not
and should join with this line
```

Check the result of your changes on the project overview page.