Eclipse Setup

To run junits for the Sample project from eclipse we need to change the eclipse setup currently there on our machines.

Installer

Download the latest installer for Spring Tool Suit (STS).

After installing please open **Help -> Eclipse Marketplace** and install following plugins:

1. ECLEMMA
2. PMD //Please configure the PMD file after installing plugin
3. CheckStyle //Please configure the Checkstyle file after installing plugin
4. FindBugs//Please configure the Findbugs file after installing plugin
5. Subversion client
6. Jupiter

Enabling ECLEMMA View

Configuring Workspace

We need to create a workspace with only those projects which are defined in the localextensions.xml file. Additionally we should also import platform and config project into the workspace.
After importing the projects please check for build path errors, Builder errors etc. Secondly also check for “Builders” for the given project. There should not be any problems with the builders.
Also Disable Spring Project Validators by browsing to Window-> Preferences

Configuring jUnits

jUnits run under its own VM and need few configuration to grasp the environment level changes required for the project to run.

New Configuration

Under “run configurations” create a new configuration for jUnits.

Configuring Arguments

After selecting project and providing a name for the jUnits run configuration select “Arguments” Tab

```
Xmx2G -ea -Dcom.sun.management.jmxremote -
Dcom.sun.management.jmxremote.authenticate=false -
Dcom.sun.management.jmxremote.ssl=false -
Dorg.tanukisoftware.wrapper.WrapperManager.mbean=true -
Djava.endorsed.dirs="%CATALINA_HOME%/lib/endorsed" -Dcatalina.base=
%CATALINA_BASE% -Dcatalina.home=%CATALINA_HOME% -
Dfile.encoding=UTF-8 -Dlog4j.configuration=log4j_init_tomcat.properties -
```

```
Djava.util.logging.config.file=jdk_logging.properties -Djava.io.tmpdir="$
{HYBRIS_TEMP_DIR}"
```

```
HYBRIS_BIN_DIR=${platformhome}/../../bin
HYBRIS_CONFIG_DIR==${platformhome}/../../config
HYBRIS_DATA_DIR=${platformhome}/../../data
HYBRIS_LOG_DIR=${platformhome}/../../log
HYBRIS_TEMP_DIR=${platformhome}/../../temp/hybris
platformhome=D:/Sample/Sandbox/Code/hybris/bin/platform [Please update this path to your platform directory.]
```

Configuring Environment

After selecting project and providing a name for the jUnits run configuration select “Environment ” Tab

```
HYBRIS_BIN_DIR=${platformhome}/../../bin
HYBRIS_CONFIG_DIR==${platformhome}/../../config
HYBRIS_DATA_DIR=${platformhome}/../../data
HYBRIS_LOG_DIR=${platformhome}/../../log
HYBRIS_TEMP_DIR=${platformhome}/../../temp/hybris
platformhome=D:/Sample/Sprint6/Sandbox/Code/hybris/bin/platform [Please update this path to your platform directory.]
```

Also configure the classpath and add all the other projects which have not been added. Steps:
Click on the “Classpath” tab
Select “Sampleweb” (in case you are configuring for Sampleweb, in case for some other project please select that project)project and from the right hand side buttons select “add project”.
Select and add all the other available projects shown in the list. Click on apply .

Environment Setup

Steps to enable jUnits

1. Take the checkout of the branch on which the jUints are to be executed
2. Build the application
3. Run the hybris server
4. Open admin console of the hybris server.
5. Browse to Console-> Tests -> Click on “Click here to open the hybris JUnitEE TestRunner”
6. Click on Initialize at the top of the new window.
7. Browse to admin console
8. Browse to Platform -> Tenants
9. Select the jUnit Tenant and click on activate.
10. Login with admin/nimda credentials.
11. Browse to Console-> Impex Import
12. or from command prompt fire the command **“ant initialize -Dtenant=junit”**

Create Test Data

Writing jUnit

Prerequisites

1. PMD,CheckStyle,Findbugs,jUnit,ECLEmma plugins are installed and configured as per project standards
2. The environment is stable with enough memory to support jUnit execution.
3. Developer is aware of java coding practices.

Creating Test Data

Good Coding Practices

Developer is able to highlight the problems in Code/jUnits that he is working on.

1. YTODO : for code improvements in jUnits. Example : //YTODO add more scenarios around multiple betslip
2. TODO : code changes , addition for Functional code. Example: //TODO implement this method properly to check channel
3. FIXME : Code changes for fixing an obvious issue. Example: //FIXME check for NPE for the given list before using it.
4. DOCTODO : For updating comments in the code base. Where ever it seems not understandable. Example: //DOCTODO Please elaborate why this if statement is needed.

Running jUnits

Run jUnits from admin console

Open Admin Console and Browse to Console-> Tests -> Click on “Click here to open the hybris JUnitEE TestRunner”.
Under the available Unit tests , select the extension’s or junit that you wish to execute and click on run.

Run jUnits from Eclipse

The main purpose is to run the jUnits from eclipse and also debug and check their code coverage. This enables the developer to finetune his jUnits to a level where jUnits are precise and cover all the necessary paths.

Running for Extensions/Projects

Browse to Run-> Run Configurations
Select the jUnit -> [Extension /Project Name] that you wish to modify
Make sure that “Run all tests in the selected project , package or source folder” is selected and it has your valid project.
Click on Apply and then click on Run.

Running for Files

Browse to Run-> Run Configurations
Select the jUnit -> [Extension /Project Name] that you wish to modify
Make sure that “Run a single test” is selected and it has your valid project and test class.
Click on Apply and then click on Run.

Running for Methods

Browse to Run-> Run Configurations
Select the jUnit -> [Extension /Project Name] that you wish to modify

Make sure that “Run a single test” is selected and it has your valid project ,test class and test method.
Click on Apply and then click on Run.

Code Coverage

Running for Extensions/Projects

Browse to Run-> Coverage…
Select the jUnit -> [Extension /Project Name] that you wish to modify
Make sure that “Run a single test” is selected and it has your valid project and test class.
Click on Apply and then click on Coverage.

Running for Files

Browse to Run-> Coverage…
Select the jUnit -> [Extension /Project Name] that you wish to modify
Make sure that “Run a single test” is selected and it has your valid project and test class.
Click on Apply and then click on Coverage.

Running for Methods

Browse to Run-> Coverage…
Select the jUnit -> [Extension /Project Name] that you wish to modify
Make sure that “Run a single test” is selected and it has your valid project and test class.
Click on Apply and then click on Coverage.

Debugging jUnits

Debug jUnits from admin console

Start hybris server in debug mode.
Open Admin Console and Browse to Console-> Tests -> Click on “Click here to open the hybris JUnitEE TestRunner”.
Under the available Unit tests , select the extension’s or junit that you wish to execute and click on run.

Debug jUnits from Eclipse

The main purpose is to run the jUnits from eclipse and also debug and check their code coverage. This enables the developer to finetune his jUnits to a level where jUnits are precise and cover all the necessary paths.

Running for Extensions/Projects

Browse to Run-> Debug Configurations
Select the jUnit -> [Extension /Project Name] that you wish to modify
Make sure that “Run all tests in the selected project , package or source folder” is selected and it has your valid project.
Make sure that “Keep JUnit running after a test run when debugging” is checked if you want to modified code is auto picked by the server.
Click on Apply and then click on Debug.

Running for Files

Browse to Run-> Debug Configurations
Select the jUnit -> [Extension /Project Name] that you wish to modify

Make sure that “Run a single test” is selected and it has your valid project and test class.
Click on Apply and then click on Debug.

Running for Methods

Browse to Run-> Debug Configurations
Select the jUnit -> [Extension /Project Name] that you wish to modify
Make sure that “Run a single test” is selected and it has your valid project ,test class and test method.
Click on Apply and then click on Debug.