|  |
| --- |
| www.missionarogya.org |
| Getting started with ArogyaCart |
| Environmental Setup Document |
|  |
| **Sumon Sadhukhan** |
| **5/22/2015** |

|  |
| --- |
| The document contains the outline of environment setup, required for ArogyaCart developer |

Table of Contents

[Overview 2](#_Toc420172418)

[Assumption 2](#_Toc420172419)

[1. Eclipse Setup 3](#_Toc420172420)

[*Configuration of Git* 3](#_Toc420172421)

[*Configuration of Maven* 5](#_Toc420172422)

[*Tomcat Configuration* 7](#_Toc420172423)

[*JDK configuration* 8](#_Toc420172424)

[2. Build and Deploy steps 8](#_Toc420172425)

[3. Pushing changes to Git. 9](#_Toc420172426)

Introduction

The document explains the setup procedure to start a web application. Required software and tools should be kept handy before starting up the procedure. The softwares, used in the application, are open source and easily available on internet.

# Overview

Below tools have been chosen to develop the application. The document gives an overview on each of the tools to be setup in Window Environment.

|  |  |
| --- | --- |
| Framework | Spring MVC |
| Database | mysql |
| Application Server | Tomcat 7.0.62 |
| Repository | Maven |
| Version Control | Git |
|  |  |
| IDE | Eclipse |
| Java | JDK 1.6.0 |

# Assumption

The developer should know how to setup the classpath in the windows environment. After installation of JDK and maven, the environment variable must be added in Classpath.

Developer should have high speed internet connection.

Getting start

For setting up the local environment you should download softwares mentioned below.

Some plugins need to be added in Eclipse for integrated development environment.

|  |  |
| --- | --- |
| Softwares | Link |
| Eclipse Juno | http://www.eclipse.org/downloads/download.php?file=/technology/epp/downloads/release/juno/SR2/eclipse-jee-juno-SR2-win32-x86\_64.zip |
| Maven | http://apache.mirrors.ionfish.org/maven/maven-3/3.3.3/binaries/apache-maven-3.3.3-bin.zip |
| Tomcat | http://supergsego.com/apache/tomcat/tomcat-7/v7.0.62/bin/apache-tomcat-7.0.62.zip |
| Java | http://www.oracle.com/technetwork/java/javase/downloads/java-archive-downloads-javase6-419409.html |

Once the download is complete extract all the zip files in proper location. For this document, following locations have been used.

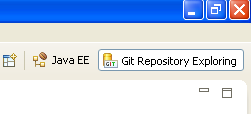
|  |  |
| --- | --- |
| Eclipse | E:\Eclipse\eclipse |
| JDK 1.6.0 | F:\jdk1.6.0 |
| Tomcat | E:\tomcat7\apache-tomcat-7.0.62 |

# Eclipse Setup

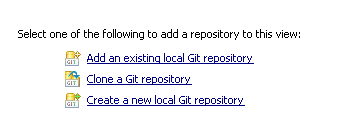
The following steps should be followed to configure Eclipse.

## *Configuration of Git*

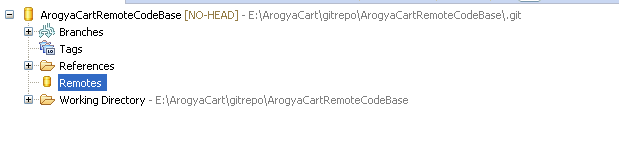
* After extracting eclipse, open the IDE.
* Enter workspace location.
* Open eclipse 🡪 Help 🡪 Eclipse Market space.
* Search with Git in Eclipse market place.
* Once the git installation is successful, switch your eclipse to “Git Repository Exploring”



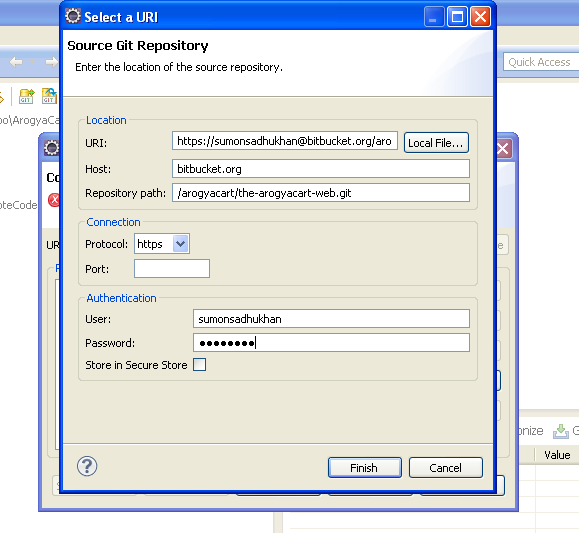
* Create a new local repository selecting the option **“Create a new local repository”**

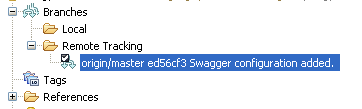


* Enter parent directory and Name on the form and click on finish.

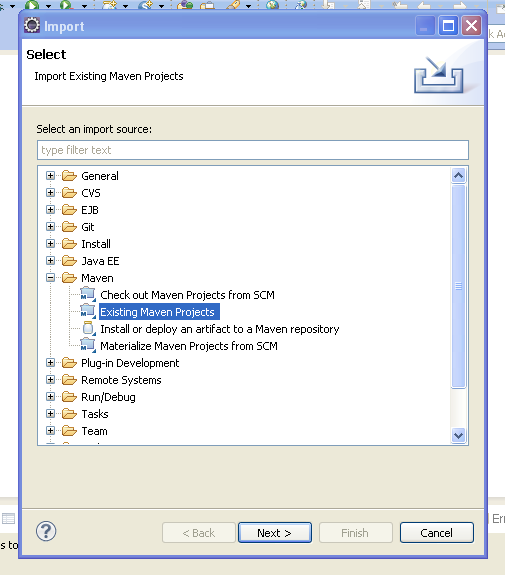
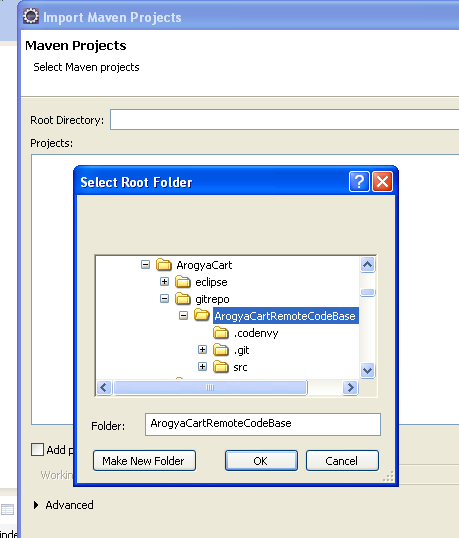


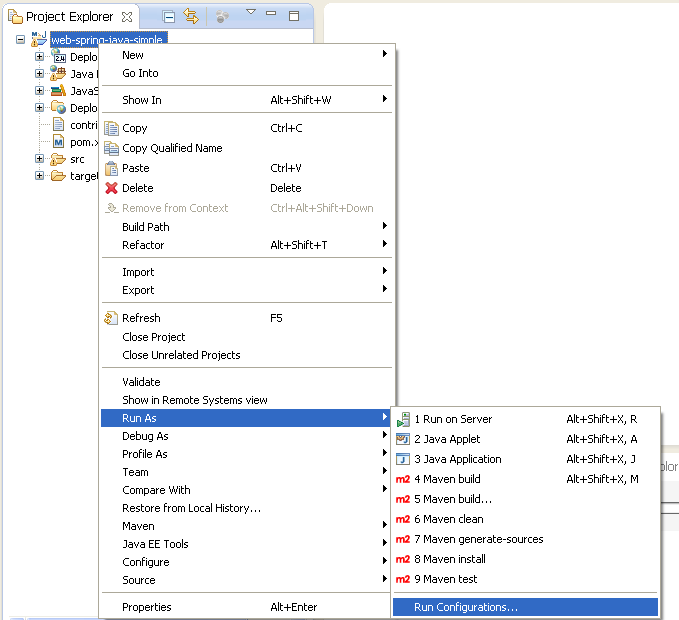
* Once the local repo is created, try to add one new Remote branch. Right Click on Remote 🡪 Create Remote then select configure fetch
* Enter the origin name. Ideally it is origin and click on OK.
* Copy the git location from bitbucket and paste in URI field. The other fields will be filled automatically. Click on Finish.

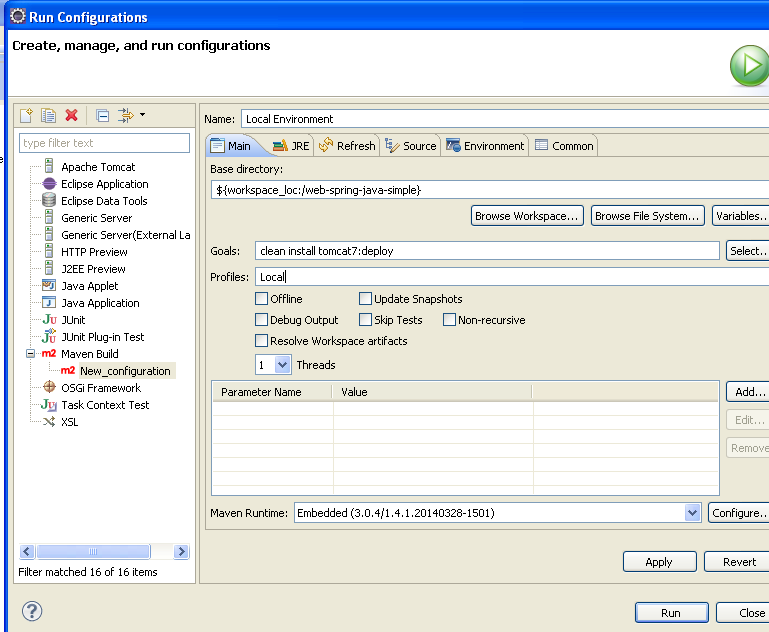


* Remove the reference mapping add a new reference mapping after click on Add.
* Enter master in the branch field and choose master.
* Click on Save and Fetch.
* Click on OK. it will fetch the updated code from bitbucket
* Click on remote Tracking and click on checkout. The code will be downloaded in local IDE.
* 
* Once the checkout is completed, the **Working Directory** folder will be updated.
* Switch the workspace J2EE perspective.

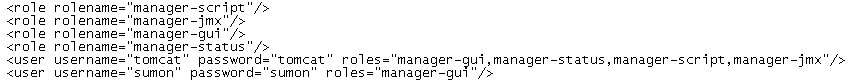
## *Configuration of Maven*

* Open eclipse market place. Install Maven from eclipse market place search result.
* Install maven integration plugins for juno.
* Once installation is done, Right click on **Project Explorer 🡪 Import 🡪 Maven 🡪 Existing Maven Project**
* 
* Click on Next button then select the Git project checkout location, after click on Browse button.
* 
* It will start analyzing the project pom.xml file.
* Click on next button and Click on finish. It will fetch all the dependency of the project.



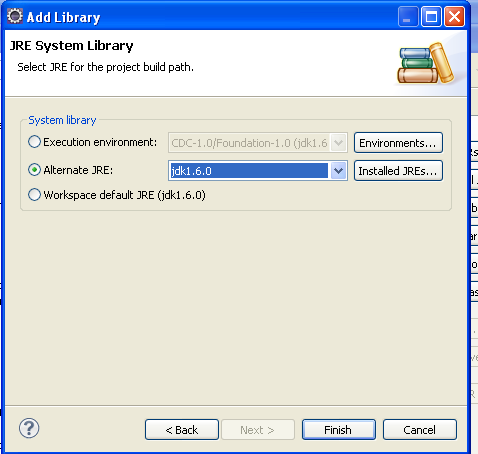
* Go to Run Configuration after right clicking on the project 🡪 Maven 🡪 Run configuration.
* Click on Browse Workspace and select the current project.
* In Goals enter **clean install tomcat7:deploy.**
* Enter **Local** in Profile.
* 
* Click on Apply to save the configuration and then close.

## *Tomcat Configuration*

* Extract tomcat7 to your local machine.
* Go to E:\tomcat7\apache-tomcat-7.0.62\conf location.
* Open tomcat-users.xml file.
* Add a new user with below configuration.
* Open command line. Enter the below text and press enter. It will start the server.
* 

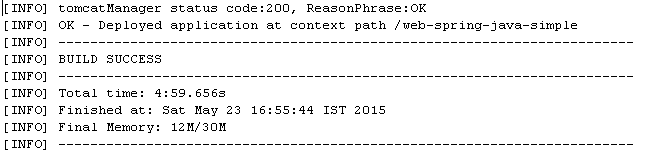
Note : Don’t start the server from Eclipse.

## *JDK configuration*

* Extract JDK 1.6 in some location.
* Select the project and right click on it and goto **Properties**
* Select Java Build path 🡪 Library.
* Add JDK 1.6.0 under JRE System Library.
* 
* Click on finish.

# Build and Deploy steps

Before deploying the code check if tomcat home page is rendering or not by opening the browser.

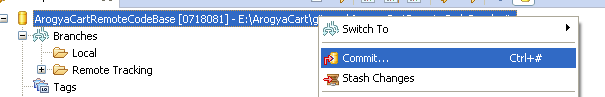
* Open pom.xml in eclipse.
* Check the username/password for **Local** profile is matching with server configuration or not. If it is not matching change it.
* Again go to **run configuration** (Right Click on the project 🡪 Run As 🡪 Run configuration.
* Select **Local Environment**.
* Check the JRE is selected as JDK 1.6.0 or not. If it is not selected change it to JDK 1.6
* Click on Apply and Run.
* You will receive a successful message once the build is completed.

Note: One of the reasons of Build failure is availability of old war packet in webapps folder. To remove old packet, you have to un-deploy the old packet from server.

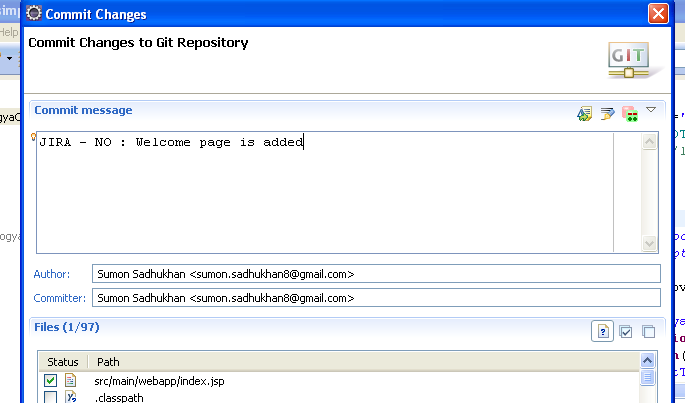
# Pushing changes to Git.

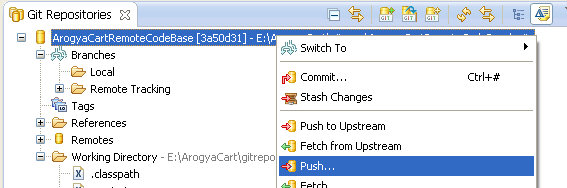
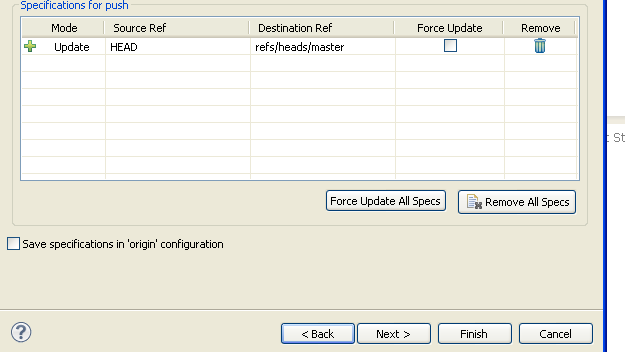
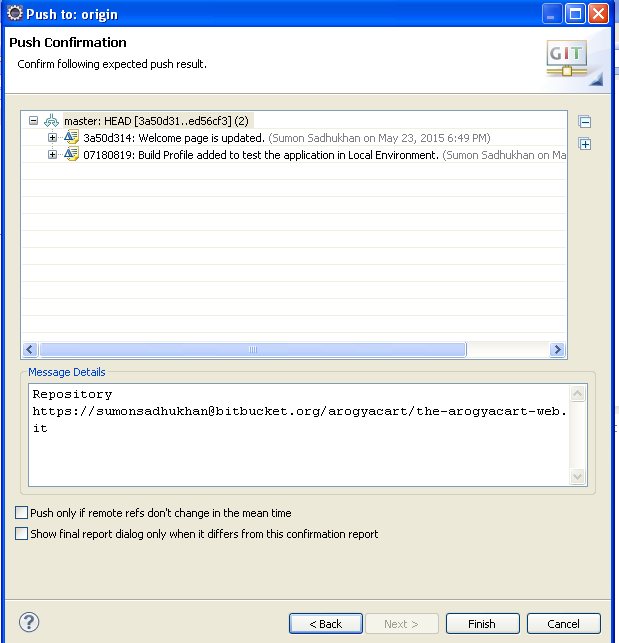
Before committing any change make sure, your change is not breaking any other functionality. Deploy the packet to the local tomcat server. Validate the change.

* Switch to Git Repository Exploring .
* Right click on the local repository, click on **Commit.**
* Select all the files related to particular JIRA. Try to commit all the files together for a particular jira along with a proper comment. The comment should contain the jira number and the remarks. Don’t commit the target folder into the repository.



* Now the local git is updated with all the changes. The change can be pushed to remote branch by click on Push.



* Right click on the branch. Click on push.
* 
* Check the remote and click on Next button.
* Check the **specification for push** if it is correct, navigate to next page clicking on next button
* 
* Click on next button and Click on Finish.
* 

Note: Before pushing any change always update your local repository, otherwise it will give error message.