# Build setup

## Workflow

Our work flow for build is as follows

1. Run maven locally with goal as clean, compile, test
2. Commit code to github
3. Jenkins picks up change
4. Jenkins runs maven with goals clean, compile, test
5. If build fails, check Jenkins logs
6. Debug till Jenkins passes

## Maven

We use Maven as our build system.

We choose maven due to its dependency management and flexibility. Maven runs all of our unit tests and compiles and packages our project.

Plugins

1. Maven shade – to build executable jar
2. Maven resources – to copy requires resources into the target folder
3. Maven inbuilt junit

POM

<project xmlns=**"http://maven.apache.org/POM/4.0.0"** xmlns:xsi=**"http://www.w3.org/2001/XMLSchema-instance"** xsi:schemaLocation=**"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"**>

<modelVersion>**4.0.0**</modelVersion>

<groupId>**sept-assignment-1**</groupId>

<artifactId>**sept-assignment-1**</artifactId>

<version>**0.0.1-SNAPSHOT**</version>

<build>

<sourceDirectory>**src**</sourceDirectory>

<testSourceDirectory>**src/com/sept01/tests/**</testSourceDirectory>

<plugins>

<plugin>

<artifactId>**maven-compiler-plugin**</artifactId>

<version>**3.3**</version>

<configuration>

<source>**1.8**</source>

<target>**1.8**</target>

</configuration>

</plugin>

<plugin>

<groupId>**org.apache.maven.plugins**</groupId>

<artifactId>**maven-shade-plugin**</artifactId>

<version>**2.4.3**</version>

<executions>

<execution>

<phase>**package**</phase>

<goals>

<goal>**shade**</goal>

</goals>

<configuration>

<transformers>

<transformer implementation=**"org.apache.maven.plugins.shade.resource.ManifestResourceTransformer"**>

<mainClass>**com.sept01.main.WISApplication**</mainClass>

</transformer>

</transformers>

</configuration>

</execution>

</executions>

</plugin>

<plugin>

<artifactId>**maven-resources-plugin**</artifactId>

<version>**3.0.0**</version>

<executions>

<execution>

<id>**copy-resources**</id>

<!-- here the phase you need -->

<phase>**validate**</phase>

<goals>

<goal>**copy-resources**</goal>

</goals>

<configuration>

<outputDirectory>**${basedir}/target/images**</outputDirectory>

<resources>

<resource>

<directory>**images**</directory>

<filtering>**true**</filtering>

</resource>

</resources>

</configuration>

</execution>

</executions>

</plugin>

</plugins>

</build>

<dependencies>

<dependency>

<groupId>**org.jsoup**</groupId>

<artifactId>**jsoup**</artifactId>

<version>**1.9.2**</version>

</dependency>

<dependency>

<groupId>**org.jfree**</groupId>

<artifactId>**jfreechart**</artifactId>

<version>**1.0.19**</version>

</dependency>

<dependency>

<groupId>**org.jfree**</groupId>

<artifactId>**jcommon**</artifactId>

<version>**1.0.23**</version>

</dependency>

<dependency>

<groupId>**junit**</groupId>

<artifactId>**junit**</artifactId>

<version>**4.12**</version>

</dependency>

<!-- http://mvnrepository.com/artifact/org.json/json -->

<dependency>

<groupId>**org.json**</groupId>

<artifactId>**json**</artifactId>

<version>**20160212**</version>

</dependency>

<!-- http://mvnrepository.com/artifact/com.eclipsesource.minimal-json/minimal-json -->

<dependency>

<groupId>**com.eclipsesource.minimal-json**</groupId>

<artifactId>**minimal-json**</artifactId>

<version>**0.9.4**</version>

</dependency>

</dependencies>

</project>

## Jenkins

We chose Jenkins for our build server. We have this running a small netbook running Ubuntu.   
Jenkins build and executes the maven build with the goals clean compile test. Jenkins will change the build to failed if any part of the maven builds fails including any tests. It builds every couple hours as well as when a commit is pushed and no current build is running.

Accessible at wolfzimmermann.ddns.net:8080