Eclipse/Spring STS Integration

Bryan Hansen
twitter: bh5k
http://www.linkedin.com/in/hansenbryan

pluralsight
hardcore developer training

Outline

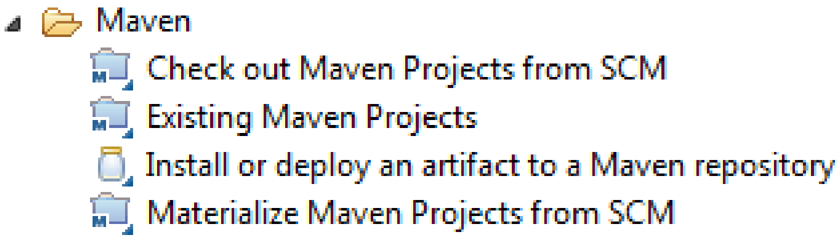
- Installing Eclipse/Spring STS
- Importing Maven Projects
- Pom viewer
- Dependency Overview
- Adding a dependency
- Dependency Heirarchy
- Effective Pom

Installation

- Eclipse/Spring STS doesn’t use the registry
- Java and Maven installed the same regardless of using an IDE
- Some IDEs do include a bundled version of Maven
- http://www.springsource.org/downloads/sts-ggts

Importing Maven Projects

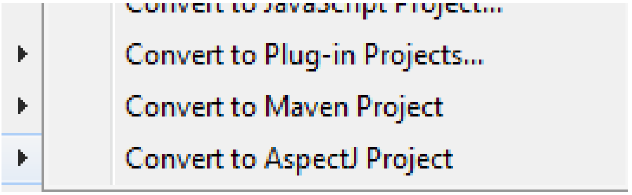
- Modern IDEs have Maven integration built into them
- Maven integration will allow us to execute default maven goals within our IDE
- IDE configuration and Classpath will be set from Maven
- Right Click in the Package Explorer > Import > Maven > Existing Maven Projects



Maven
- Check out Maven Projects from SCM
- Existing Maven Projects
- Install or deploy an artifact to a Maven repository
- Materialize Maven Projects from SCM

Converting Existing Projects

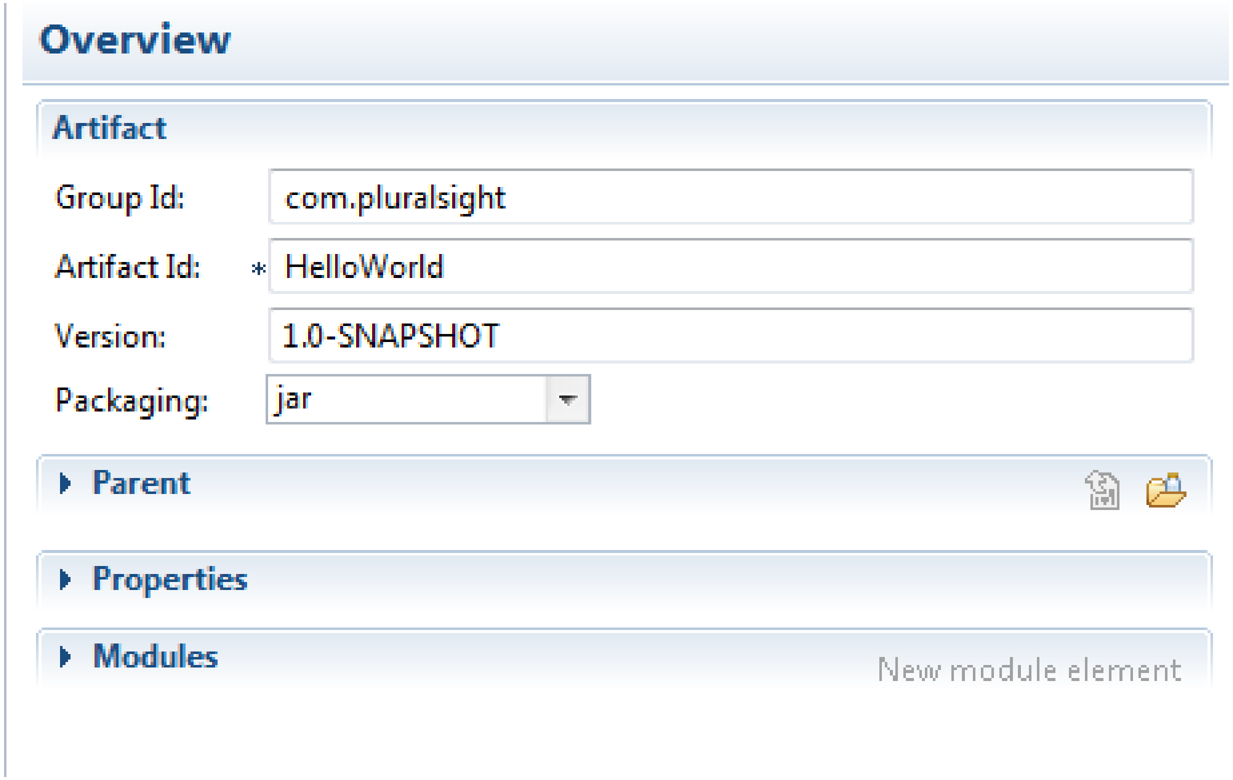
- If you have a pom.xml file, you can convert the project to a Maven project
- Right click on the project containing the pom.xml file > Configure > Convert to Maven Project


Convert to JavaScript Project...
Convert to Plug-in Projects...
Convert to Maven Project
Convert to AspectJ Project

- Once converted the project will set the classpath and automatically build the project

Pom Viewer

- Default view when you open the pom file
- Pom overview shows the high level elements of your project
- Changes made here are directly changing the source



Overview

Artifact

Group Id: com.pluralsight

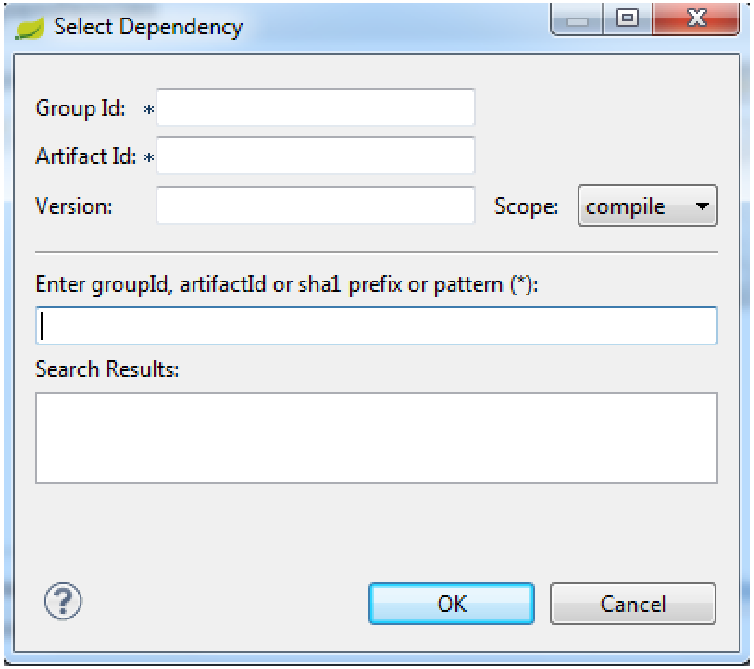
Artifact Id: * HelloWorld

Version: 1.0-SNAPSHOT

Packaging: jar

Dependencies

- Shows which dependencies we have installed and allows us to manipulate dependencies too
- Dependency Management (advanced topic) is also displayed
- The add screen has searching capability



Select Dependency

Group Id: *

Artifact Id: *

Version:

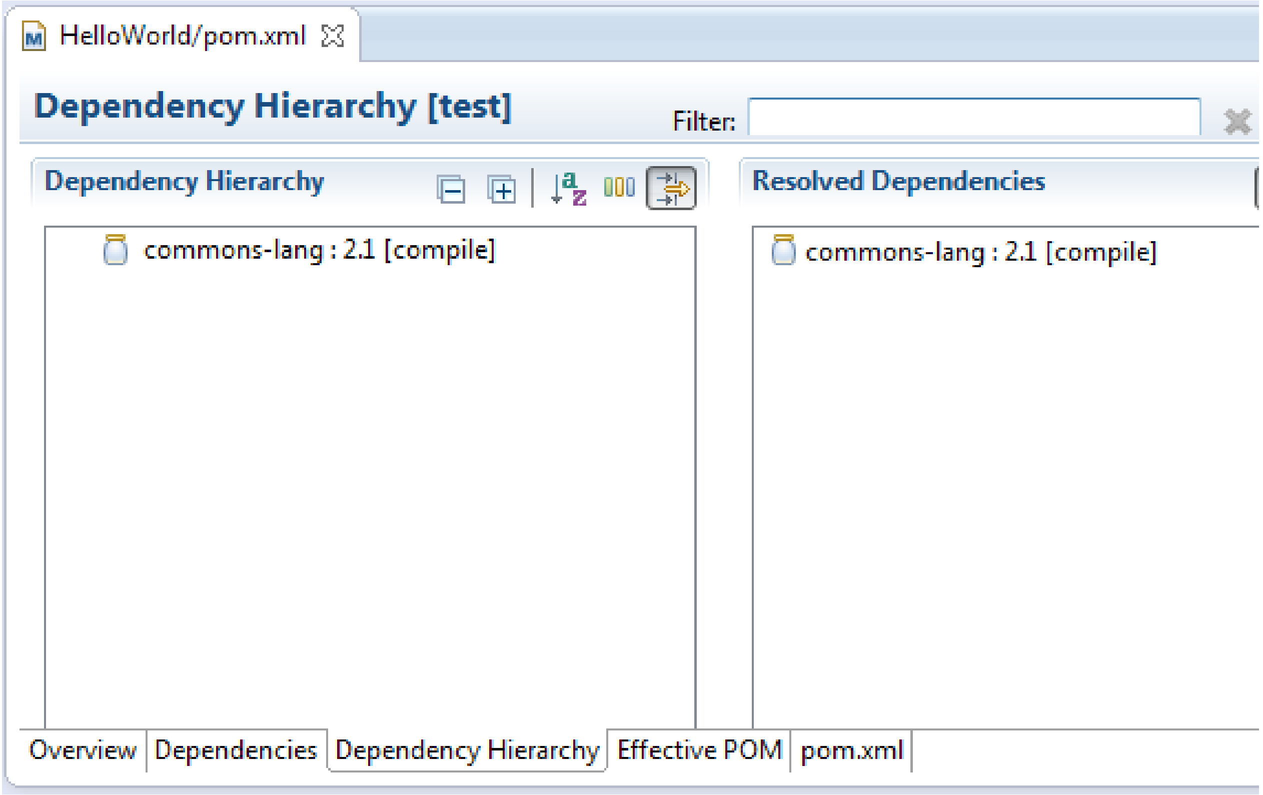
Scope: compile

Enter groupId, artifactId or sha1 prefix or pattern (*):

Search Results:

Dependency Heirarchy

- Displays the complete dependency tree, including transitive dependencies as well overridden dependencies
- Scope of the resource is also displayed



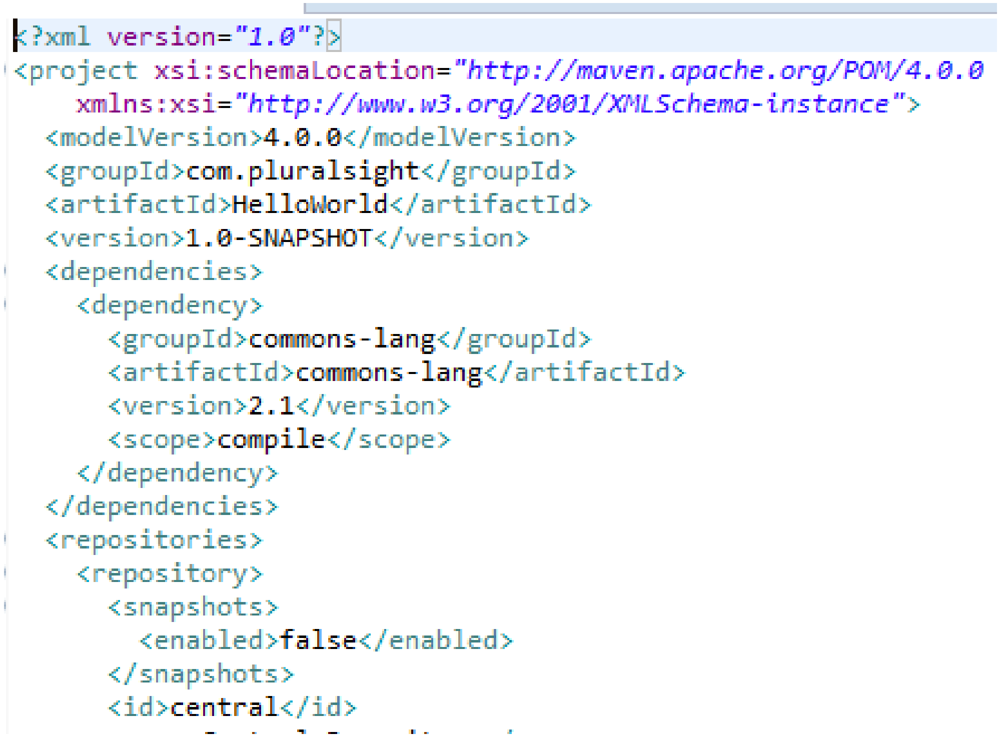
Dependency Hierarchy [test]

commons-lang : 2.1 [compile]

commons-lang : 2.1 [compile]

Effective Pom

- The complete pom with everything inherited from the project pom, if we have a parent pom, and the default super pom
- More of a debugging tool to see what the pom is doing



```
<?xml version="1.0"?>
<project xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.pluralsight</groupId>
  <artifactId>HelloWorld</artifactId>
  <version>1.0-SNAPSHOT</version>
  <dependencies>
    <dependency>
      <groupId>commons-lang</groupId>
      <artifactId>commons-lang</artifactId>
      <version>2.1</version>
      <scope>compile</scope>
    </dependency>
  </dependencies>
  <repositories>
    <repository>
      <snapshots>
        <enabled>false</enabled>
      </snapshots>
      <id>central</id>
```

Summary

- Eclipse/STS installation is really unzipping
- Existing projects can be imported easily and converting projects can be easily as well
- Adding dependencies inside the IDE can be easier using the searching tools
- Solving dependency resolution errors is more convenient in the IDE
- Configuring your IDE is also more convenient with Maven