Using Maven for HJlib projects

Maven is a build automation tool used primarily for Java projects. Projects using Maven or other build systems are easiest to use as they simplify compiling, building, and testing the project. In addition, major IDEs like IntelliJ and Eclipse have excellent support via maven plugins which simplifies the development process. Dependency management is one of the areas where Maven excels, for our purposes this means that we will save a lot of effort in configuring the projects to set up the dependency on the HJlib jars. Hence, for the labs and assignments in COMP322 we will be distributing maven project templates to be used by students to complete their work.

Maven incorporates the concept of convention over configuration by providing sensible default behavior for projects. Without customization, source code is assumed to be in *\${basedir}/src/main/java* and resources are assumed to be in *\${basedir}/src/main/resources*. Tests are assumed to be in *\${basedir}/src/rest*, and a project is assumed to produce a JAR file. The pom.xml file is the core of a project's configuration in Maven. It is a single configuration file that contains the majority of information required to build a project in just the way you want. The interested reader can learn more about maven by reading various articles and tutorials.

Maven Installation:

Maven is a Java tool, so you must have Java installed in order to proceed. Next, follow the instructions on the maven site to install maven. Once installed, open a new command prompt and run mvn --version to verify that it is correctly installed.

Your first HJlib Maven project:

Once you have maven installed, it is now time to run your first HJlib project using maven. Download the <project-name>.zip file provided to you for use as the lab or assignment. Unzip the contents of the zip file, for example lab1.zip unzips the contents into a directory named lab1. After unzipping this is how the directory structure should look like:

```
README
pom.xml
src
    main
       java
           edu
               rice
                    comp322
                        HelloWorldError.java
                        ReciprocalArraySum.java
    test
        java
            edu
                 rice
                     comp322
                         LablTest.java
```

In all the projects, the README file will provide additional instructions on how to build and run the projects. Note that there are separate directories for the source code and unit tests. We will rely on writing JUnit tests for our labs and assignments.

Building your project

To build your project, simply run mvn clean compile from the command line. Running the command should produce an output like below:

```
[12:30:06 ~/projects/comp322-s2015-projects/lab1]
shamsimam@MacbookPro $ mvn clean compile
[INFO] Scanning for projects...
[INFO] ------
[INFO] Building lab1 1.0-SNAPSHOT
[INFO] ------
. . .
Downloading: http://www.cs.rice.edu/~vs3/hjlib/code/maven-repo/edu/rice/hjlib-cooperative/0.1.4-SNAPSHOT
/hjlib-cooperative-0.1.4-SNAPSHOT.jar
. . .
Downloaded: http://www.cs.rice.edu/~vs3/hjlib/code/maven-repo/edu/rice/hjlib-cooperative/0.1.4-SNAPSHOT
/hjlib-cooperative-0.1.4-SNAPSHOT.jar (447 KB at 208.5 KB/sec)
[INFO]
[INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ lab1 ---
[INFO]
[INFO] --- maven-dependency-plugin:2.9:properties (default) @ lab1 ---
[INFO]
[INFO] --- mayen-resources-plugin:2.6:resources (default-resources) @ lab1 ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory /Users/shamsimam/projects/comp322-s2015-projects/lab1/src/main
/resources
[INFO]
[INFO] --- maven-compiler-plugin:2.3.2:compile (default-compile) @ lab1 ---
[INFO] Compiling 2 source files to /Users/shamsimam/projects/comp322-s2015-projects/lab1/target/classes
[INFO] ------
[INFO] BUILD SUCCESS
[INFO] ------
[INFO] Total time: 8.352s
[INFO] Finished at: Sun Jan 04 12:33:31 CST 2015
[INFO] Final Memory: 17M/102M
[INFO] ------
```

Note that maven automatically downloaded the hjlib-cooperative-0.1.4-SNAPSHOT.jar file, this is maven's automatic dependency management at work. Jar files are only downloaded if they have not been previously downloaded by maven. One of the benefits of using maven will be that you will receive automatic updates to the HJlib jar files as we progress through the course.

Running unit tests

To run the unit tests, run the mvn clean compile test command:

```
[12:33:31 ~/projects/comp322-s2015-projects/lab1]
shamsimam@MacbookPro $ mvn clean compile test
[INFO] Scanning for projects...
[INFO]
[INFO] ------
[INFO] Building lab1 1.0-SNAPSHOT
[INFO] ------
[INFO]
[INFO] ...
[INFO] --- maven-surefire-plugin:2.17:test (default-test) @ lab1 ---
[INFO] Surefire report directory: /Users/shamsimam/projects/comp322-s2015-projects/lab1/target/surefire-
reports
_____
TESTS
          -----
objc[23579]: Class JavaLaunchHelper is implemented in both /Library/JavaVirtualMachines/jdk1.8.0_25.jdk
/Contents/Home/jre/bin/java and /Library/Java/JavaVirtualMachines/jdk1.8.0_25.jdk/Contents/Home/jre/lib
/libinstrument.dylib. One of the two will be used. Which one is undefined.
Running edu.rice.comp322.Lab1Test
LablTest.testSimple() starts...
LablTest.testSimple() ends.
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.163 sec - in edu.rice.comp322.LablTest
Results :
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0
[INFO] ------
[INFO] BUILD SUCCESS
[INFO] ------
[INFO] Total time: 3.917s
[INFO] Finished at: Sun Jan 04 12:36:59 CST 2015
[INFO] Final Memory: 24M/211M
[INFO] ------
```

If all the tests run successfully, a successful build is reported. As you work through your labs and assignments running the unit tests will provide an initial sanitary check on whether your implementations work as exepceted in terms of providing the correct result.

Running the main program

If you instead want to run your individual programs, we will have configured the maven pom.xml file to allow your to run the individual classes. Please refer to the README files for instructions on how to run the individual programs. For example, running the HelloWorld example from lab1 requires the following command: mvn clean compile exec:exec -Phelloworld

```
[12:37:02 ~/projects/comp322-s2015-projects/lab1]
shamsimam@MacbookPro $ mvn clean compile exec:exec -Phelloworld
[INFO]
[INFO] ------
[INFO] Building lab1 1.0-SNAPSHOT
[INFO] -----
[INFO]
[INFO] --- exec-maven-plugin:1.3.2:exec (default-cli) @ lab1 ---
First: HelloWorld
Second: HelloWorld
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 2.599s
[INFO] Finished at: Sun Jan 04 12:40:54 CST 2015
[INFO] Final Memory: 17M/214M
[INFO] ------
```

Using IntelliJ with your Maven project

IntelliJ IDEA fully integrates with Maven version 2.2 and later versions, allowing you to create or import Maven modules, download artifacts and perform the goals of the build lifecycle and plugins. If you want to use an existing Maven project, you can import it directly by opening its pom.xml file. When a Maven project is imported, IntelliJ IDEA analyzes the pom.xml file and automatically downloads the necessary dependencies.

IntelliJ IDEA provides two ways of running the Maven goals:

· Create run/debug configuration and launch it.

• Use the Run Maven Build command in the Maven Projects tool window. This way doesn't require any run/debug configuration.

To import lab1 as a maven project simply follow the following steps as shown in the images:

IntelliJ IDEA	File	Edit	View	Navigate	Co	de		
	New Project							
ect -lib-git [habaner ternal Libraries	lm; Im; Nev	^ N						
	Op Clo	Open Open Recent Close Project						
	Otł	ж; ►						
	lmı Exp Exp							
		Save A	.11		жs			
	∽ Synchronize て第Y Invalidate Caches / Restart							
	🖶 Print							
	Pov	ver Sav	ve Mode					

000	Select File or Directory to Import	
Select directory with existin Eclipse project (.project) or Maven project file (pom.xr Gradle build script (*.gradl	ng sources, r classpath (.classpath) file, ml), le).	
A 🗔 🌒 🖬 🛤 🗙 🤉	5	Hide path
/Users/shamsimam/projec	cts/comp322-s2015-projects/lab1	Ľ
▼ 🗖 comp322 ► 🗖 home	2–s2015–projects work1	
🔻 🗖 lab1		
Stress	c raet	
	om.xml	
RE	EADME	
READ	ME.txt	
Drag and dro	op a file into the space above to quickly locate it in the tree.	
(?)	Cancel	ОК

e O O Import Project
Create project from existing sources
Import project from external model
Eclipse
© Gradle
m Maven
Import Project
Root directory ~/projects/comp322-s2015-projects/lab1
Search for projects recursively
Project format: .idea (directory based) 🛟
Keep project files in:
Import Maven projects automatically
Create IntelliJ IDEA modules for aggregator projects (with 'pom' packaging)
Create module groups for multi-module Maven projects
Keep source and test folders on reimport
Exclude build directory (%PROJECT_ROOT%/target)
✓ Use Maven output directories
Generated sources folders: Detect automatically \$
Phase to be used for folders update: process-resources 💠
IDEA needs to execute one of the listed phases in order to discover all source folders that are configured via Maven plugins. Note that all test-* phases firstly generate and compile production sources.
Automatically download: Sources Documentation
Dependency types: jar, test-jar, maven-plugin, ejb, ejb-client, jboss-har, jboss-sar, war, ear, bundle
Comma separated list of dependency types that should be imported
Environment settings
? Cancel Previous Next

000	Import Project
Select Maven projects to import	
du.rice.comp322:lab1:1.0-SNAPSHOT	
	Select all Unselect all
Open Project Structure after import	
? Cancel	Previous Next

Ensure that you are using the JDK8 distribution in your machine.

$\Theta \cap \Theta$				Import	Project				
Please select project SDM This SDK will be used by	(. default by	y all pro	ject m	odules.					
+ -	Name	1.8							
1.7	Name.	1.0							
1.8	IDV how	o noth.	Liber	n dava da	va Vietua IV	lachinac /id		idle/Content	c/Homo
👾 Android API 19 Platfo	JUK nom	e path:	LIDra	iry/Java/Ja	vavirtualiv	achines/ju	K1.8.0_05	.juk/Content	s/Home
DEA IC-135.909	Classpat	h Sourc	epath	Annotations	Documen	tation Paths			
	 /Librar	y/Java/J	avaVirt	ualMachine	s/jdk1.8.0	_05.jdk/Con	ntents/Hom	e/lib/ant-java	fx.jar
	/Librar	y/Java/J	avaVirt	ualMachine	s/jdk1.8.0	_05.jdk/Con	ntents/Hom	e/lib/dt.jar	
	/Librar	y/Java/J	avaVirt	ualMachine	s/jdk1.8.0	_05.jdk/Con	ntents/Hom	e/lib/javafx-n	nx.jar
	/Librar	y/Java/J	avaVirt	ualMachine	s/jdk1.8.0	_05.jdk/Con	ntents/Hom	e/lib/jconsole.	.jar
	/Librar	y/Java/J	avaVirt	ualMachine	s/jdk1.8.0	_05.jdk/Con	ntents/Hom	e/lib/sa-jdi.ja	r
	/Librar	y/Java/J	avaVirt	ualMachine	s/jdk1.8.0	_05.jdk/Con	ntents/Hom	e/lib/tools.jar	
	/Librar	y/Java/J	avaVirt	ualMachine	s/jdk1.8.0	_05.jdk/Con	ntents/Hom	e/jre/lib/char	sets.jar
	/Librar	y/Java/J	avaVirt	ualMachine	s/jdk1.8.0	_05.jdk/Con	ntents/Hom	e/jre/lib/depl	oy.jar
	/Librar	y/Java/J	avaVirt	ualMachine	s/jdk1.8.0	_05.jdk/Con	ntents/Hom	e/jre/lib/html	converter.jar
	/Librar	y/Java/J	avaVirt	ualMachine	s/jdk1.8.0	_05.jdk/Con	ntents/Hom	e/jre/lib/javav	ws.jar
	/Librar	y/Java/J	avaVirt	ualMachine	s/jdk1.8.0	_05.jdk/Con	ntents/Hom	e/jre/lib/jce.ja	ar
	/Librar	y/Java/J	avaVirt	ualMachine	s/jdk1.8.0	_05.jdk/Con	ntents/Hom	e/jre/lib/jfr.ja	ir I
	/Librar	y/Java/J	avavirt	uaiMachine	s/jak1.8.0	_05.jdk/Con	itents/Hom	e/jre/lib/jfxsv	vt.jar
	/Librar	y/Java/J	avavirt	ualMachine	s/jak1.8.0	_US.Jak/Con	itents/Hom	e/Jre/lib/Jsse.	jar Sachart Sacht i
	/Librar	y/Java/J	ava Virt	ualMachine:	s/juk1.0.0	_05.juk/Con	tents/Hom	e/jre/lib/man	agement-agent.j
	/Librar	y/Java/J v/lava/l	ava Virt	ualMachine	s/juk1.0.0 s/idk1 8 0	_05.juk/Con	tents/Hom	e/jre/lib/plugi	urces iar
	/Librar	v/lava/l	avaVirt	ualMachine	s/jdk1.8.0	_05.jdk/Con	tents/Hom	e/jre/lib/reso	arces.jai
	/Librar	v/lava/l	avaVirt	ualMachine	s/jdk1.8.0	05.jdk/Con	tents/Hom	e/jre/lib/ext/	drdata iar
	/Librar	v/lava/l	avaVirt	ualMachine	s/jdk1.8.0	05.idk/Con	tents/Hom	e/ire/lib/ext/	dnsns.iar
	/Librar	v/lava/l	avaVirt	ualMachine	s/idk1.8.0	05.idk/Con	tents/Hom	e/ire/lib/ext/i	fxrt.iar
	/Librar	v/Java/J	avaVirt	ualMachine	s/idk1.8.0	05.jdk/Con	ntents/Hom	e/ire/lib/ext/l	ocaledata.jar
	/Librar	y/Java/J	avaVirt	ualMachine	s/jdk1.8.0	05.jdk/Con	ntents/Hom	e/jre/lib/ext/r	nashorn.jar
	/Librar	y/Java/J	avaVirt	ualMachine	s/jdk1.8.0	_05.jdk/Con	ntents/Hom	e/jre/lib/ext/s	sunec.jar
	+ -								
? Cancel								Previous	Next

000	Import Project		
Please enter a name t	o create a new IntelliJ IDEA project.		
Project name:	lab1		
Project file location:	~/projects/comp322-s2015-projects/lab1		
? Cancel		Previous	Finish

Open the maven plugin window to execute the maven build commands from IntelliJ. Of course, you can still right click on the individual files to run them by editing the run configuration to include the javaagent as described in the download and set up article.

IntelliJ IDEA	File	Edit	View	Navigate	Code	Analyze	Refactor	Build	Run	Tool
Э			Tool	Windows		•	J Projec	t	ж	1
ect •		2 - 201	Quic	k Documer pen Modul	ntation e Setting	^ J JS F4	★ Favori ▶ Run ₩ Debug	tes	#	2 4 5
b1 (~/projects/comp322-s201 (ternal Libraries		2-5201	Rece Rece Rece	ent Files ently Chang ent Changes	ed Files	B ዝድ ጋቴፓ	Properties and the second seco	ure Jes	ж ж	6 7 9
			Quic	k Switch Sc	heme	<u>^`</u>	Ant Bu Applic	uild ation Se	ervers	
			Tool Tool ✓ Stati ✓ Navi	bar Buttons us Bar gation Bar			Comm Design Event Maver	nander ner Log n Project	S	
			Ente Ente	r Presentati r Full Scree	on Mode n	° ^₩F	■ Palette ■ Palette ■ Termin I UI Des I Versio	e n al signer on Contre	r ol	F12

	01	Ŀ	1	776-	535
Maven Projects					
💋 📴 🛓 🕂 🕨 🖬 🌵 🏵 😤 🕸					
Profiles					
🔻 🖬 lab 1					
🔻 📴 Lifecycle					
🤹 clean					
🔅 validate					
🧔 compile					
🧔 test					
🔅 package					
verify					
🌻 install					
🔅 site					
deploy					
Plugins					
Dependencies					