**Maven COMMAND LINE Quick Reference:**

Lets say your class is located in:   
**C:/PRJ\_ROOT/classes/hello/HelloWorld.class**   
And that your HelloWorld.java package declaration was:   
**package hello;**

|  |
| --- |
| Compile, package, and run, if you have NO library dependencies: |
| SOURCES: COMPILE: https://spring.io/guides/gs/maven/ RUN: https://www.youtube.com/watch?v=20U2aLtWYu8&list=PLN4rUakF78aB0dpR0\_ngbldjzcC6qsXRt |
| cd C:/PRJ\_ROOT mvn compile mvn package cd C:/PRJ\_ROOT/classes/  java -cp . hello.HelloWorld  //NOTE: Directory is NOT C:/PRJ\_ROOT/classes/hello because //We need to have cd point to root of where package starts.  //IF YOU HAVE DEPENDENCIES in your project, running this way will NOT work //Well, it could if you hand-paste the .jar files you need into the class folder. But that defeats the point of maven being a build tool if you are going to do something manual like that.  The example at: https://spring.io/guides/gs/maven/ That uses: import org.joda.time.LocalTime; Will not run using:  java -cp . hello.HelloWorld |

|  |
| --- |
| Run maven build when you have library dependencies: |
| SOURCE: http://www.vineetmanohar.com/2009/11/3-ways-to-run-java-main-from-maven/ |
| Step1: Move project root on command line: I.g: **cd C:/PRJ\_ROOT**  Step2: Choose how you would like to run: Without arguments:  **mvn exec:java -Dexec.mainClass="hello.HelloWorld"**  With arguments:  **mvn exec:java -Dexec.mainClass="hello.HelloWorld" -Dexec.args="arg0 arg1 arg2"**  With runtime dependencies in the CLASSPATH:  **mvn exec:java -Dexec.mainClass="hello.HelloWorld" -Dexec.classpathScope=runtime** |

**UPDATE**: Want **.WAR** files?

|  |
| --- |
| 1. Install the apche **maven-war-plugin** http://maven.apache.org/plugins/maven-war-plugin/ Then: **mvn install** //NOT mvn package, as intuitive as that may seem. (mvn package makes .jar files) |