How to Set-up and Use Apache Maven for your project

1. What is Apache Maven?

Apache Maven is a build tool, that also handles dependency management and it provides lots of plugins for managing your software project. It is typically used for creating and managing Java projects. It is an open-source product, from the Apache Software Foundation.

2. Obtaining and Setting-up Apache Maven:

To obtain Maven, simply download it from the Apache Maven project website at http://maven.apache.org/download.cgi. More specifically, you download the Maven binaries (i.e. the zip or tar.gz package containing the precompiled, read-to-use Maven tool) from an Apache mirror site, such as http://www.gtlib.gatech.edu/pub/apache/maven/maven-3/3.5.3/binaries/ or http://apache.osuosl.org/maven/maven-3/3.5.3/binaries/. To set it up and begin working with it, do the following steps (NOTE: Maven works with Java, as a build tool. So, having the Java JDK installed, is a prerequisite):

2.1 Download one of the Maven binaries package into a folder on your computer e.g. download http://apache.osuosl.org/maven/maven-3/3.5.3/binaries/apache-maven-3.5.3-bin.zip to your Windows PC's Downloads folder.

2.2 Unzip the package, to produce a folder named, *apache-maven-3.5.3*. You may then copy this folder to a suitable, preferred location/folder on your computer. e.g. C:\apache\apache-maven-3.5.3.

2.3 To complete the basic Maven setup, the following couple of environment variables-related settings are required:

2.3.1 Add 2 new environment variables, M2_HOME and MAVEN_HOME, both pointing to the maven folder, e.g. C:\apache\apache-maven-3.5.3

2.3.2 Add the path to the maven executable binaries folder, i.e. C:\apache\apache-maven-3.5.3\bin, to the Path environment variable. e.g. simply appending, %MAVEN_HOME%\bin to the existing Path variable values, will do.

2.4 To test/verify the Maven setup, simply launch open a Command or terminal window, and execute the command such as,

c:\>*mvn --version*

This should display the Apache Maven product version information, along with Java version and OS version information, as well; indicating that you now have Maven setup and ready-to-use.

3. Using/Working with Maven:

3.1 Using Maven on the command or terminal window to create and manage a Java project:

3.1.1 Creating a simple Java command-line application project:

3.1.1.1 Launch open a command/terminal window.

3.1.1.2 Create a new folder/directory for the project and cd into it.
e.g. c:\> *mkdir MyFirstMavenJavaConsoleApp && cd MyFirstMavenJavaConsoleApp*

3.1.1.3 Execute the following command:

```
mvn archetype:generate -DgroupId=edu.mum.cs.myapp -DartifactId=myapp -DarchetypeArtifactId=maven-archetype-quickstart
-DinteractiveMode=true
```

This will generate a new Java command-line app project inside a folder named, myapp, and using the archetype named, maven-archetype-quickstart.
(As you can imagine, there are several other archetypes, which can be used for creating various other kinds of Java projects, including a springmvc web application project, a struts2 webapp project etc).

3.1.1.4 Having used maven to generate/create the project, you can now load the project into an IDE tool, such as Eclipse, Netbeans or IntelliJ IDEA, and continue the rest of your coding. And then use maven tool to build, package and deploy the finished application.

Note: This has been just a brief intro to the Apache Maven tool. As you can imagine, there is much more that you can do with it. Including using a Maven plugin for your favorite IDE, without using the command/terminal window.

//-- The End --//