**What is Maven?**

* Maven is a Project Management tool
* Most popular use of Maven is for build management and dependencies

**What Problems Does Maven Solve?**

* When building your Java project, you may need additional JAR files
  + For example: Spring, Hibernate, Commons Logging, Json and etc…

1. One way is to download the JAR files from each project web site and then manually add the JAR files to your build path

for example, how my project would work without maven. I need Spring JAR file, Hibernate JAR file, Apache Commons JAR files.  
I need to go to each project’s website and download them manually and then add those to my build path.

So **MAVEN** can help us in this process and do that instead of us.

**Maven Solution**

* Tell Maven the projects you are working with (dependencies)(POM project object management)
  + Spring, Hibernates
* Maven will go out and download the JAR files for those projects for you
* And Maven will make those JAR files available during compile/run
* Maven is like personal shopper, who you give your shopping list and then it goes and buy those things

For example, my project with maven.  
I give my shopping list to Maven, Maven will go off and download Spring JAR files, the Hibernate Jar files, Commons and etc, pull it to my computer and make it available to use

**How Maven works behind the scene**

Project config file is read by config, then Maven checks your maven local repository (maven cache) for those dependencies, then if it doesn’t find those dependencies, Maven go in Maven Central Repository (remote), and it will download those Jar files and save them in Maven local Repository. So, finally Maven will use that files to build and run your application

**Handling JAR Dependencies**

* When Maven retrieves a project dependency
  + It will also download supporting dependencies
  + For example, Spring depends on commons-logging, so maven also will download commons-logging

**Building and Running**

* When you build and run your app
* Maven will handle class and build path for you
* Based on config file, Maven will add Jar files accordingly

**Maven Advantages**

* Dependency management
  + Maven will find Jar files for you
  + No more missing JARs
* Building and Running your pojects
  +  **Build Path**: This refers to the directories and files where your source code and dependencies are located. In traditional Java projects, you often need to manually configure your build path in your IDE (like Eclipse or IntelliJ), specifying which directories and libraries should be included during compilation and execution.
  +  **Classpath**: This is an environment variable used by Java to specify the locations where Java classes can be found. When you run a Java application, the **classpath** tells the JVM where to look for classes and resources (like JAR files or compiled classes).
* Standard directory structure
* Minimal local configuration

**Maven Key Concepts**

* POM File – pom.xml
  + POM-Project Object Model file
  + Configuration file for project (“Shopping list” for maven)
  + Always, Located in root

**Pom.xml structure:**project meta data

Dependencies

Plug ins

**Project meta data** – project name, version, output file type (JAR, WAR)

**Dependencies** – what dependencies should we need to build and run

**Plug ins**- additional repositories

* **Project Coordinates**
* <groupId>com.LukaCode.springboot.demo</groupId>  
  <artifactId>myCoolApp</artifactId>  
  <version>0.0.1-SNAPSHOT</version>  
  <name>myCoolApp</name>  
  <description>Demo project for Spring Boot</description>

**GroupID** – name of company, group or organization

**ArtifcatId** – name for this project

**Version** – release version. If project is under active development then 1.0 SNAPSHOT

**Dependency Coordinates**

* **GAV -G**roup ID, **A**rtifact ID, **V**ersion

**<build>  
 <plugins>  
 <plugin>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-maven-plugin</artifactId>  
 </plugin>  
 </plugins>  
</build>**

ეს ფლაგინი ეხმარება მეივენს დაბილდვაში. JAR ან WAR ფაილად დაპაკეტებაში და ასევე mvn spring-boot:run ბრძანება ამ ფლაგინის დახმარებით ბილდავს და უშვებს პროექტს.