Spring Framework

- Prof. Hariom A. Pandya
- Prof. Vivek S. Patel

Introduction

- It provides everything you need to embrace the Java language in an enterprise environment, with support for Groovy and Kotlin as alternative languages on the JVM.
- As of Spring Framework 5.0, Spring requires JDK 8+ (Java SE 8+) and provides out-of-the-box support for JDK 9 already.

Features

- **Core technologies:** dependency injection, events, resources, i18n, validation, data binding, type conversion, SpEL, AOP.
- **Testing:** mock objects, TestContext framework, Spring MVC Test, WebTestClient.
- **Data Access:** transactions, DAO support, JDBC, ORM, Marshalling XML.
- Spring MVC and Spring WebFlux web frameworks
- **Integration:** remoting, JMS, JCA, JMX, email, tasks, scheduling, cache.
- **Languages:** Kotlin, Groovy, dynamic languages.

Building Tools v/s IDE?

- Build tools are programs that automate the creation of executable applications from source code(eg. .apk for android app).
- Building incorporates compiling,linking and packaging the code into a usable or executable form.

Setup Project for Maven to Build

- Create the directory structure(src->main->java->hello)
- Within the src/main/java/hello directory, you can create any Java classes you want
- HelloWorld.java Greeter.java
- Download Maven, unzip and add the bin folder to your path. And Test it.

```
[Hariom@localhost Desktop]$ mvn -v
Apache Maven 3.5.3 (3383c37e1f9e9b3bc3df5050c29c8aff9f295297; 2018-02-25T01:19:0
5+05:30)
Maven home: /home/Hariom/Desktop/Spring/apache-maven-3.5.3
Java version: 1.8.0_131, vendor: Oracle Corporation
Java home: /usr/java/jdk1.8.0_131/jre
Default locale: en_US, platform encoding: UTF-8
OS name: "linux", version: "2.6.32-573.el6.x86_64", arch: "amd64", family: "unix
"
```

Define a simple Maven build

- Create Maven project definition.
- Maven projects are defined with an XML file named pom.xml.
- Among other things, this file gives the project's name, version, and dependencies that it has on external libraries.
- It includes the following details of the project configuration:
  - <**modelVersion**>. POM model version (always 4.0.0).
  - <**groupId**>. Group or organization that the project belongs to. Often expressed as an inverted domain name.
  - <**artifactId**>. Name to be given to the project's library artifact (for example, the name of its JAR or WAR file).
  - <**version**>. Version of the project that is being built.
  - <**packaging**> - How the project should be packaged. Defaults to "jar" for JAR file packaging. Use "war" for WAR file packaging

Build Java code

- **mvn compile**
- execute the compile goal. When it's finished, you should find the compiled .class files in the target/classes directory.
- **mvn package**
- package goal will compile your Java code, run any tests, and finish by packaging the code up in a JAR file within the target directory. The name of the JAR file will be based on the project's <artifactId> and <version>
- If you'd like to install your project's JAR file to that local repository, then you should invoke the install goal.
- **mvn install**
- For output **java -jar <jar-file>**

Spring Bean Factory