SPRING BOOT

AND

MICROSERVICE

BY

Kumar4Java

SPRING BOOT

1. Spring Boot :--

=>Spring boot is a spring-based framework which is open source and developed by Pivotal Team.

=>Available versions of Spring Boot are

a>Spring Boot 1.x.

b>Spring Boot 2.x.

c>Spring Boot 3.x

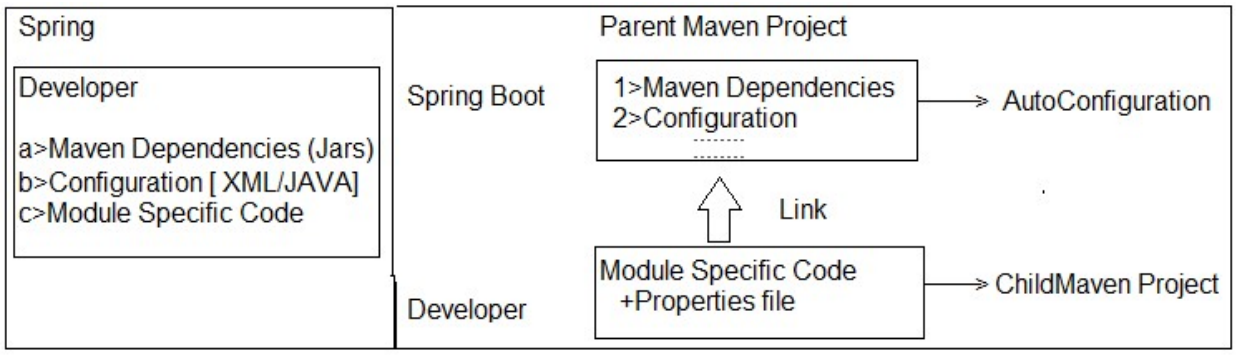
=>Spring Boot provides **AutoConfiguration** which means reduce Common lines of code in Application which is written by Programmers and handles Jars with version management. (i.e. Providing Configuration code XML/Java and maintaining all jars required for Project **Parent Jars + Child Jars**)

=>Spring Boot is an Abstract Maven project also called as Parent Maven Project (A Project with partial code and jars)

=>Here Programmer will not write configuration code but need to give input data using

a>Properties File (application.properties).

b>YAMAL File (application.yml).



2>Spring Boot:-- a>application.yml:--

```
server.port: 8082
spring:
 datasource:
  url: jdbc:mysql://${MYSQL_HOST:localhost}:3306/db
  username: user
  password: password
 jpa:
  hibernate.ddl-auto: update
```

b>Starter Dependency (which gives config code and Jars):--

```
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-web</artifactId>
</dependency>
```

=>Spring Boot supports end to end process that is called.

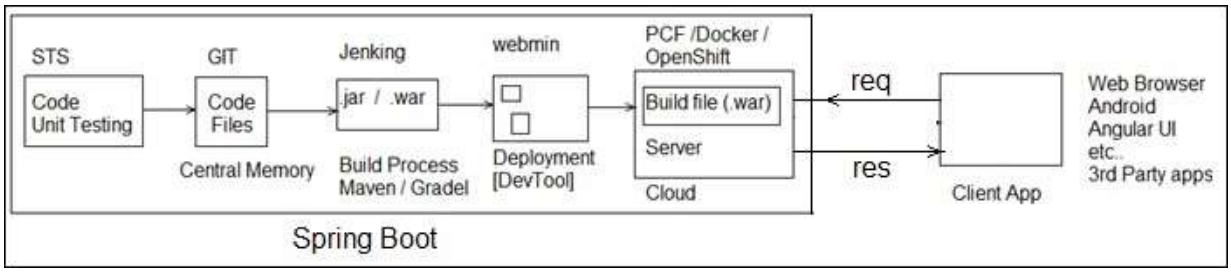
=>Coding => Unit testing => Version control => Build => Deployment => Client Integration.

a.>GIT (github.com) is used to store our code files. It is called as **Central Repository or version Control Tool.**

b.>.Java is converted to .class (Compile) .class + (other files .xml, .html…) converted to .jar/.war finally (build process).

c.>Place .jar/.war in server and start server is called as Deployment.

d.>Spring Boot Application is a service provider app which can be integrated with any UI client like Android, Angular UI, RFID (Swiping Machine), Any 3rd party Apps, Web Apps using Rest and JMS.



NOTE:--

a>Spring Boot supports two build tools **Maven** and **Gradle**.

b>Spring Boot supports 3 embedded servers and 3 embedded databases. These are not required to download and install.

i>Embedded Servers:--

1>Apache Tomcat (default)

2>JBoos Jetty

3>Undertow

ii>Embedded DataBase:--

1>H2

2>HSQL DB

3>Apache Derby

c>Spring Boot supports cloud apps with micro services pattern. ["Both coding and Deployment"].

=>Coding is done using Java and Netflix Tools

=>Deployment can be done on various clouds.

d>Spring Boot supports basic Operations:--

1>WebMVC and WebServices (Rest).

MVC→Model, View , Controller

Important Annotations:

@Controller +@ResponseBody = RestController

@RequestMapping + GET = @GetMapping

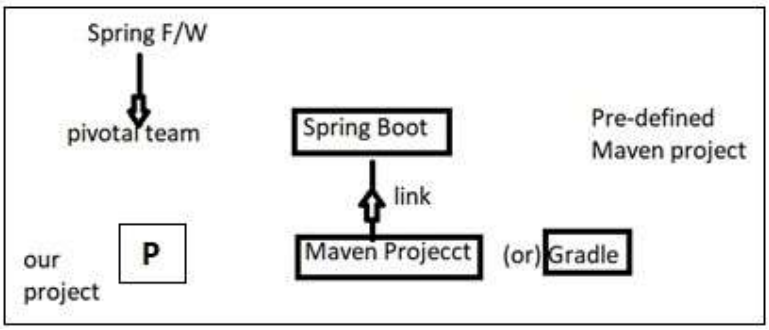
2>JDBC and ORM (Hibernate with JPA).
3>Email, Scheduling, JMS, Security.
4>Cache and Connection Pooling.
5>DevTools, Swagger UI, Actuator and Profiles.
6>UI Design using HTML, JSP, Thymeleaf …etc.

e> Supports Input Data (Key = val) Using (for AutoConfiguration code):--
=>Properties file or YAML files.

1.1 Spring Boot Application Folder System:--

=>We can write spring Boot application either using Maven or using Gradle (one of build tool).

=>Our project contains one parent project of spring boot which is internally maven project (hold version of parent).



=>Application should contain 3 major and required files.

Those are

1. SpringBootStarter class
2. application.properties /application.yml
3. pom.xml/build.gradle

1. SpringBootStarter class:-- It is a main method class used to start our app. It is entry point in execution. Even for both stand alone and web this file used.

2. application.properties/application.yml:-- This is input file for Spring boot (Spring container). It holds data in key=value format.

** File name must be “application” or its extended type.

** Even .yml (YAML) file is finally converted to .properties only using SnakeYaml API

** yml is better approach to write length properties code.

3. pom.xml (or) build.gradle:-- This file holds all information about

a. Parent boot project version

b. App properties (JDK version/maven/cloud versions….)

c. Dependencies (JARS Details)

d. Plugins (Compiler/WAR…etc)

Application Folder System