infogain

CDP : Spring boot Developer

Objective

- Define and demonstrate Spring boot
- Developing and deploying application using spring boot
- Design and develop web applications using spring boot
- Develop and deploy applications using JPA and Spring Boot.
- Develop and deploy rest application
- Define and Demonstrate Actuator
- Define and demonstrate HAL
- Define and Demonstrate spring boot messaging (RabbitMQ)
- Develop and deploy application for Spring boot documentation
- Developing end to end application using spring boot
- Developing and deploying cloud native application.

<table>
<tr><th colspan="3">CDP : Infogain Certified Spring Boot Developer Schedule</th></tr>
<tr><th>Date</th><th>Day</th><th>Topic</th></tr>
<tr><td>18-Jul-18</td><td>Wednesday</td><td>Spring Boot -1</td></tr>
<tr><td>20-Jul-18</td><td>Friday</td><td>Spring Boot -2</td></tr>
<tr><td>26-Jul-18</td><td>Thursday</td><td>Spring Boot-3</td></tr>
<tr><td colspan="3"><b>TBD</b></td></tr>
</table>

Session 1 - Objective

- Define and demonstrate Spring boot
- Developing and deploying application using spring boot
- Design and develop web applications using spring boot

Introduction to Spring Boot

- Introduction to Spring Boot
- What are the Primary Goals of Spring Boot?
- What is Spring Boot Starter?
- Ways of Creating Spring Boot Application?
- Spring Boot different annotation
- Creating spring boot application and deploying
- Working with different webjars.

What is Spring Boot?

- It is not a framework. it is a tool that allows you to create spring based application within no time.
- it helps you to build ,package and deploy the spring application with minimal or absolutely no configurations.
- It provides a set of Starter Pom’s which one can use to add required dependencies and also facilitate auto configuration.

Spring Boot Primary Goals

- Allows you to focus more on business features and less on infrastructure.
- Spring Boot dynamically wires up beans and settings and applies them to your application context on startup of your application.
- To avoid XML Configuration completely
- To avoid defining more Annotation Configuration(It combined some existing Spring Framework Annotations to a simple and single Annotation)
- To avoid writing lots of import statements
- To provide some defaults to quick start new projects within no time.

Spring Boot Starter

- **Spring Boot Starter**
  - Spring Boot starters are jar dependencies which provide a quick way to configure your application, without manually including a lot of related dependencies.
  - It contain a lot of the dependencies that you need to get a project up and running quickly.
  - The starter POMs are convenient dependency descriptors that can be added to your application’s Maven.
  - You get a one-stop-shop for all the Spring and related technology that you need, without having to hunt through sample code and copy paste the jar’s.

List of some Important Spring Starter POM

Name	Description
spring-boot-starter	Core starter, including auto-configuration support, logging
spring-boot-starter-test	Starter for testing Spring Boot applications with libraries including JUnit, Hamcrest and Mockito
spring-boot-starter-data-jpa	Starter for using Spring Data JPA with Hibernate
spring-boot-starter-web	Starter for building web, including RESTful, applications using Spring MVC. Uses Tomcat as the default embedded container

Spring-boot-starter-Parent

- **spring-boot-starter-parent**
  - It is a special starter that provides useful Maven defaults.
  - Maven users can inherit from the **spring-boot-starter-parent project** to obtain sensible defaults.
  - It declare your project as child project of **spring starter-parent project**. The idea is that configuration defined in the parent project is inherited to the child project
  - By using a specific version of this starter, we are going to implicitly select compatible versions of targeted spring modules and third party tools
- **The parent project provides the following features:**
  - Java 1.8 as the default compiler level.
  - UTF-8 source encoding.
  - Sensible resource filtering for application.properties and application.yml.

Dependency Management

- Each release of Spring Boot provides a curated list of dependencies it supports.
- In practice, you do not need to provide a version for any of these dependencies in your build configuration as Spring Boot is managing that for you.
- When you upgrade Spring Boot itself, these dependencies will be upgraded as well in a consistent way.
- **Note**
    - Each release of Spring Boot is associated with a base version of the Spring Framework so it is highly recommend you to not specify its version on your own.

```
<parent>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-parent</artifactId>
<version>2.0.3.RELEASE</version>
</parent>
```

- **Spring Boot 2.0.3 work well with Spring 5.0.5 Release**

Different ways of Creating Spring Boot Application

- The traditional approach to developing RESTful services as a web application.
- From the microservices point of view, one needs a mechanism to develop services as executables, self-contained JAR files with an embedded HTTP listener.
- Spring Boot is a tool that allows easy development of such kinds of services.
- Dropwizard and WildFly Swarm are alternate server-less RESTful stacks.
- Spring Boot not only increases the speed of development but also provides a set of production-ready ops features such as **health checks and metrics collection**.
- **Creating a Spring Boot App**
  - Using Spring Boot CLI Tool

  https://docs.spring.io/spring-boot/docs/current/reference/html/getting-started-installing-spring-boot.html
  - Using Spring STS IDE

  https://spring.io/tools/sts/all
  - Using Spring Initializer

  http://start.spring.io/

  Running Application using Maven :

  $ maven install

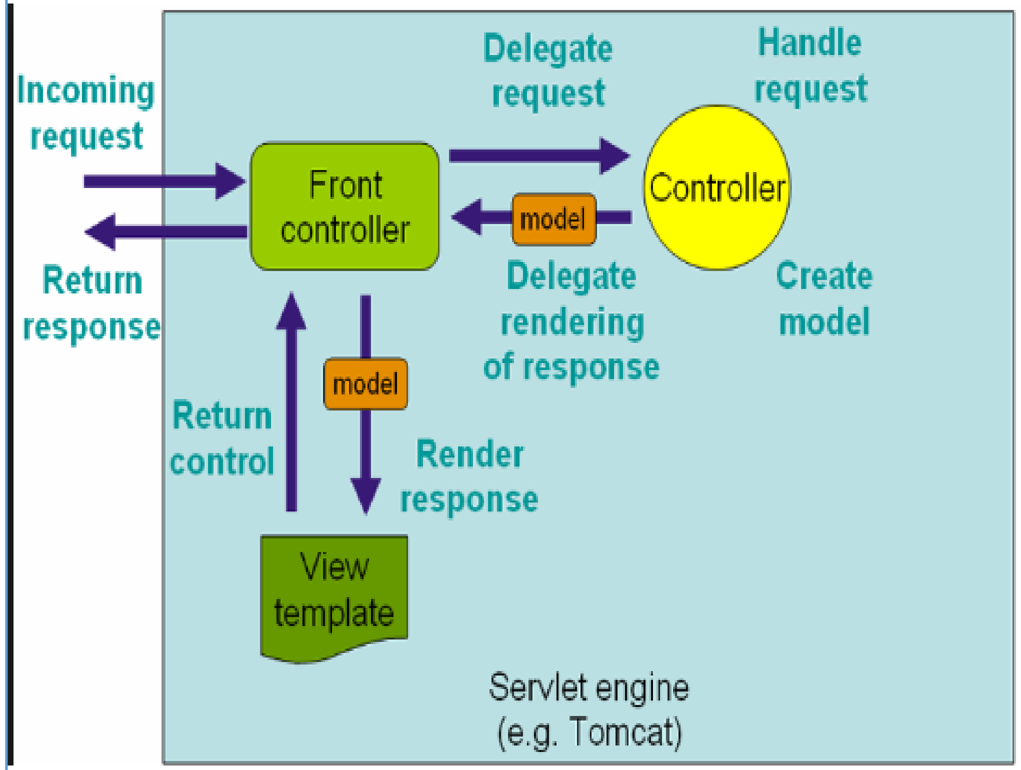
Spring Boot and other important Annotations

- **@SpringBootApplication: Adds all of the following:**
  - **@Configuration :** Tags the class as a source of bean definitions for the application context.
  - **@EnableAutoConfiguration :** Tells Spring Boot to start adding beans based on class path settings, other beans, and various property settings.
  - **@ComponentScan :** Tells Spring to look for other components, configurations, and services in the specified package allowing it to find the controllers.
  - @Controller
  - @RequestMapping
  - @RequestParam
  - @ModelAttribute
  - @RequestBody and @ResponseBody
  - @RequestHeader and @ResponseHeader
  - @SessionAttribute

Spring MVC :

Spring MVC Request Flow

- DispatcherServlet receives HTTP Request.
- DispatcherServlet identifies the right Controller based on the URL.
- Controller executes Business Logic.
- Controller returns a) Model b) View Name Back to DispatcherServlet.
- DispatcherServlet identifies the correct view (ViewResolver).
- DispatcherServlet makes the model available to view and executes it.
- DispatcherServlet returns HTTP Response Back.
- Flow : http://docs.spring.io/spring-framework/docs/2.0.8/reference/images/mvc.png



Working with WebJars

- Among the other features that Spring Boot simplifies, one of them is the webjars.
- Webjars are the static client-side dependencies (for eg. javascript libraries) packaged into JARs.
- Manually adding the client side libraries could be difficult to maintain, so with webjars (feature enhanced in Spring Boot), we can add these libraries just by making required pom configuration entries, and we are all-set to start using them instantly.

```
<dependency>
        <groupId>org.webjars</groupId>
        <artifactId>bootstrap</artifactId>
        <version>3.3.6</version>
     </dependency>
     <dependency>
        <groupId>org.webjars</groupId>
        <artifactId>jquery</artifactId>
        <version>1.9.1</version>
     </dependency>
```

- **Using Wejars in jsp page :**

```
<link href="webjars/bootstrap/3.3.6/css/bootstrap.min.css" rel="stylesheet">
<script src="webjars/jquery/1.9.1/jquery.min.js"></script>
<script src="webjars/bootstrap/3.3.6/js/bootstrap.min.js"></script>
```

infogain

www.infogain.com

Infogain Corporation, HQ

485 Alberto Way Los Gatos,
CA 95032 USA
Phone: 408-355-6000
Fax: 408-355-7000

Pune

7th Floor, Bhalerao Towers, CTS No.1669 -
1670, Behind Hotel Pride,
Shivaji Nagar, Pune - 411005
Phone : +91-20-66236700

Infogain Irvine

41 Corporate Park,
Suite 390 Irvine, CA 2606 USA
Phone: 949-223-5100
Fax: 949-223-5110

Infogain Austin

Stratum Executive Center Building D
11044 Research Boulevard Suite 200
Austin, Texas 78759

Noida

A-16, Sector 60, Noida Gautam Budh agar,
201301 (U.P.) India
Phone: +91-120-2445144
Fax: +91-120-2580406

Dubai

P O Box 500588 Office No.105,
Building No. 4, Dubai Outsource Zone,
Dubai, United Arab Emirates
Tel: +971-4-458-7336