Spring Boot

Spring :

Spring framework is a java platform that provides comprehensive infrastructure support for developing java applications. Spring handles the infrastructure so you can focus on your application.

- Spring is a advanced java framework (J2EE)
- It also have java features
- Spring is a light weight and open source framework (jar file)
- It can be used for all layer implementations for a real time application
- Simplicity (simple because as it is non-invasive)
- Testability
- Loose coupling (objects are loosely coupled)
- Non-invasive (no need to import or export the headers/files)
- *Spring is not depend on one class to another class*

Spring Boot :

Spring Boot makes it easy to create stand-alone, production-grade Spring based Applications that you can "just run".

- We take an opinionated view of the spring platform
    - Configuration: (to change the some user interface)

        JAR files - java archive files ->zip files containing java class files

        WAR files - web application files

- Spring Boot is opinionated
- Spring Boot is stand alone
- Spring Boot is Production Grade
- Spring integrates all other frameworks like Hibernate and Struts.

Spring Boot (adv):

- Separate web server is not needed for Spring Boot.
- No WAR files configuration and management
- WAR files / JAR files
- Bill of materials

Spring Features:

1. POJO (Plain old Java Object) - class - private variables → getter and setters methods
2. Dependency Injection
3. REST API/SOP
4. MVC - model view controller
5. Security

MVC (Model View Controller):

- It is design pattern (arch pattern)
- It used loosely coupled
- **View :** responsible for rendering of model -UI
- **Model :** responsible for storing and retrieving data
- **Controller :** responsible for responding to user input

REST API:

API -> Application Programming Interface

- When user request the (*httprequest)* but server response the (*XML/JSON)* formate
- REST -> Representational State Transfer
- ***Example :*** *ticket booking app (*payment - user pay the amount go to gpay / any successful means automatically close then return to application)
- It perform (sql) CREATE READ UPDATE DELETE -CRUD
- Web application POST GET PUT DELETE

1. Dependency Injection:

It mainly used in to avoid Tight coupling to create loosely coupling

- Interface Injection: In this type of injection, the injector uses Interface to provide dependency to the client class.
- It is simplify the upcasting concept
- ***@Component*** : it declare means that class will create a object in spring container
- ***@Autowired*** : it declare in upcast of class then automatically connect to object in spring container
- ***@Qualifier(" ") :*** incase more than two class available means use this annotation
- ***@Scope(value="prototype")*** : it means object call many times
- ***@WebServlet("/aboutus.html")*** : it use to show the servlet web page

2. Servlet :

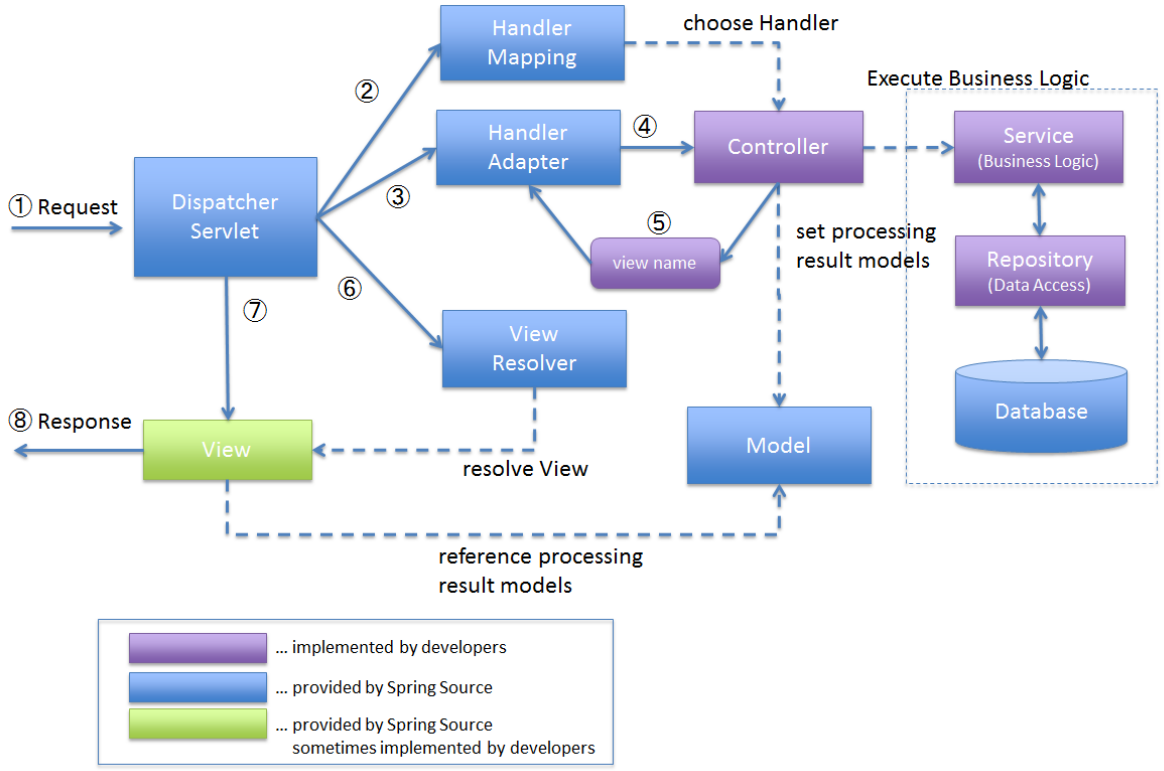
- It's used to create web applications and REST services in Spring MVC.
- In a traditional Spring web application, this servlet is defined in the web.xml file.
- To request from client to response from server
  Server component
  Two type of request there
  1. Static req : already available to response
  2. Dynamic req : it create new response

JSP (java server page):

- It is a server side technology. It is used for creating web application. It is used to create dynamic web content.
- In this JSP tags are used to insert JAVA code into HTML pages. Example : **<% java code %>**

3. Spring MVC:

- Model View Controller
- Easy, Flexible, Separation of Concerns
- Add dependency for ***jasper : tomcat jasper*** version : ***tomcat version(9.0.48)***



@Controller : Typically used in combination with annotated handler methods based on the *@RequestMapping* annotation

@RequestMapping : It's used to mark a class as a web request handler. It's mostly used with Spring MVC applications

@ResponseBody : It's used to mark a class as a web request handler time pass / return the values

4. Application.properties:

in a spring boot application, application.properties file is **used to write the application-related property into that file**. This file contains the different configuration which is required to run the application in a different environment, and each environment will have a different property defined by it.

HttpSession:

- Security
- Traffic
- **Provides a way to identify a user across more than one page request or visit to a Web site and to store information about that user**. The servlet container uses this interface to create a session between an HTTP client and an HTTP server.

@RequestParam : annotation enables spring to extract input data that may be passed as a query, form data, or any arbitrary custom data.

Spring MVC DB sample web appliation:

Application.properties:

- Spring.h2.console.enabled = true
- Spring.datasource.platform = h2
- Spring.datasource.url = jdbc:h2:mem:user (jdbc connection->h2 databse->members->some user class

Then type chrome in ***localhost:port:h2-console*** to show database then change ***data.url***

@Entity : The @Entity annotation specifies that the class is an entity and is mapped to a database table. The @Table annotation specifies the name of the database table to be used for mapping

@ID : The @Id annotation specifies the primary key of an entity and the @GeneratedValue provides for the specification of generation strategies for the values of primary keys.

- Create **userDAO** interface and extends of **crudRepository<user, integer>** use to perform (*create read update delete)* operations.

- Then ***autowired*** that interface *userdao and create adduser*
- That method call ***userdao.save(user)*** use to perform crud operator of database

@SpringBootApplication : it is used to mark a configuration class that declares one or more @Bean methods and also triggers auto-configuration and component scanning.

It's is equal to @configuration @EnableautoConfiguration @componentScan

@Configuration (used for Java-based configuration), @ComponentScan (used for component scanning), and @EnableAutoConfiguration (used to enable auto-configuration in Spring Boot).

@ComponentScan is searching packages for Components

Demo sample web application source code

Demo sample web application source code updated