Friday, July 1, 2022 9:33 AM

Managed Components
Managed Classes

Container manages the beans . How ?
-- Creating instances in different ways(Singleton /Prototype , eager/lazy ) --- BeansFactory
-- DI = Dependency Injection ---- properties are WIRED ( through setters , constructors , autowired properties ) = D ---dependency means the properties

---

MVC = Model View Controller ---- Architecture

Model = Data ( DO , DAO , Data processing Logic )
View = Presentation of the Data ( HTML , JSP ) ---- BROWSER

To **Loosely Couple Model and View** , A CONTROLLER is used

Model , Model , Model , Model

Controller

View , View , View, View

---

**MVC in Servlet JSP**

Model = DAO ,DO
SERVLET ----Controller
View = JSP

---

**Spring MVC**

Model = DAOBeans , BLBeans

Controller = Web MVC Controller Bean

View = JSP / Thymeleaf

---

Spring Annotation written for **class Level** -------

@Configuration } What will spring think ? This is java config , I will get @Bean tags inside it………

**@Component** } What will spring think ? This is a java bean that I have to manage ,Instantiate ,DI !!!!
**@Service** } - " -
**@Repository** } - " -

Three names mean the same for container !!! They are only used for programmer readability.
For programmer @Service means some feature of business logic is in the class
@Repository means some DAO or DB related tasks are done in the class
@Component --- general bean , not specifically DB related or service

@Controller }What will spring think ? This is a java class that I have to manage , Instantiate, DI
PLUS

Spring has to call the methods of this class when different REQUEST come

---

We write a Controller : MAP the request to Model and View

Request = GET or POST

Output ---- a JSP

JSP goes to the browser

---

Development Phase

1. Write JSP
2. Write Configuration
3. Write Controller

Deployment Phase -----

Run the tomcat server --- the embedded tomcat runs and deploys

Production Phase / Request Phase ---

Client starts sending requests from browser

---

1. webapp /WEB-INF/views/ hello.jsp
2. java class controller
3. Application.properties ---- add prefix and suffix

---

Spring Boot = Many boiler plate codes are WRAPPED inside
--- many configurations can be given through key value pair setting in application. properties

1. Application context is created by spring boot
2. If the WEbController class is in the same package as main class then scan is not needed
   Spring boot will auto scan it
3. When spring finds the controller ---- it will instantiate the controller

---

FLOW --- hello ----------shows a button ----on click ------------goes to welcome url -----shows welcome

Flow2 ---- we passed uname to welcome

---

| | |
|---|---|
| URL = input | See a screen that accepts a number and a button |
| On button click = URL showTable | Generate a table and show |