**Spring Framework : 7 days**

**Spring framework with Maven and Gradle**

**Spring Framework - 5 days**

**Spring boot -- 2 days**

**Day 1 - 16-10-2019**

**MVC**

**Model View Controller :**

**View ---> HTML/JSP**

**Controller ---> Servlet**

**Model ---->Java classes**

**JavaBean Service Dao Layer Resource class**

**JEE :**

**Servlet/JSP/EJB**

**C V M**

**Enterprise Java Bean : EJB is use to create distributed, high secure, remote invocation application.**

**EJB Container : Application Server**

**Glashfish, WebLogic etc**

**Model layer**

**EJB Client : Web Application (Servlet /JSP)**

**EJB**

**Client EJB Server**

**Servlet/JSP**

**interface interface**

**business methods business method**

**JNDI impl**

**Java Naming Directive Interface**

**types of ejb**

**EJB 2.x EJB 3.x**

**1. Session Bean 1. Session Bean**

**2. Entity Bean 2. JPA**

**3. Message Driven Bean (MDB) 3. MDB**

**MOM:**

**JMS**

**MQ**

**Framework : Framework contains lot of pre-defined classes and interfaces which internally connected to each other.**

**Framework is not a final product it is template or protocol.**

**70 to 80%task take care by framework.**

**Implementation of design pattern taken care by framework.**

**Single ton**

**factory method**

**MVC**

**Dao**

**etc**

**Struts : Struts is open source framework provided by apache. It is web framework which internally follow MVC pattern. Struts provided Front controller design pattern. ActionServlet pre-defined class provided by Struts which behave as FrontController design pattern.**

**Struts 1.x and 2.x, Struts is known as Controller centric framework.**

**JSF : Java Server Faces : It is open source framework provided by Oracle. JSF internally follow MVC provided front controller design pattern. FacesServlet class.**

**JSF is known as View Centric Framework.**

**JSF Vs HTML5/CSS6/JavaScript**

**Angular /React JS**

**Hibernate: Improve DAO layer.**

**JPA is a specification and hibernate is a implementation of JPA.**

**Spring : Spring is open source light weighted layered architecture framework. Spring also known as onion architecture framework.**

**Spring modules :**

**1. Spring core**

**2. Spring context**

**3. Spring JDBC**

**4. Spring ORM with Hibernate/ JPA**

**Spring data**

**5. Spring MVC: Model Centric framework.**

**6. Spring Rest**

**9. Spring boot**

**7. Spring security**

**8. Spring testing**

**10. Spring cloud**

**11. Spring web flux**

**12. Spring micro services**

**13. Spring integration**

**Spring Core and Context :**

**IOC : Inversion of Control :**

**IOC is a software design pattern. In place of creating object or resource and maintaining the life of the object or creating explicitly, allow to maintain life of resource or object to container or system or software etc.**

**DI : Dependency Injection : DI is a one type of implementation of IOC.**

**In place of creating object explicitly allow to create the object to container for POJO class.**

**Employee emp1= new Employee();**

**Employee emp2 = new Employee();**

**Employee emp3 = new Employee();**

**Types of DI :**

**1. Constructor base**

**2. Setter base**

**3. Property base**

**4.interface base : but spring doesn't support interface base DI.**

**We have to configure to achieve DI using XML or Annotation.**

**Spring version 2.x, 3.x, 4.x, 5.x**

**Spring boot :**

**Spring Constructor based DI Using XML Configuration :**

**commons-logging.jar :**

**spring.jar :**

**BeanFactory : BeanFactory is a core interface which provided set of methods which help to achieve DI.**

**auto-wired : Collaboration of more than one object, replacement of property ref or constructor-arg ref.**

**<bean class ="com.Employee" id="emp1" auto-wired="byName">**

**<property name="id" value="100"></property>**

**<property name="name" value="Raj"></property>**

**</bean>**

**<bean class ="com.Employee" id="emp2" auto-wired="byName">**

**<property name="id" value="100"></property>**

**<property name="name" value="Raj"></property>**

**</bean>**

**<bean id="laddress" class="com.Address">**

**<property name="city" value="Bangalore"></property>**

**<property name="state" value="Kar"></property>**

**</bean>**

**<bean id="paddress" class="com.Address">**

**<property name="city" value="Delhi"></property>**

**<property name="state" value="Delhi"></property>**

**</bean>**

**Spring framework**

**Day 2 - 17-10-2019**

**DI Using Annotation :**

**beans.xml**

**<beans>**

**<bean class="com.Employee" id="obj"></bean>**

**</beans>**

**@Component("emp")**

**class Employee {**

**@Value(name=100)**

**private int id;**

**@Value(name="Ravi")**

**private String name;**

**@Autowired**

**private Address add;**

**}**

**@Component("add")**

**class Address {**

**private String city;**

**private String state;**

**}**

**BeanFactory core interface**

**ApplicationContext is a interface extends BeanFactory interface.**

**Spring with DataSource with JDBC**

**Spring DI using XML and Annotation**

**@Component**

**class Employee {**

**@Autowired**

**private Address add;**

**}**

**@Component**

**class Address implements Abc{**

**}**

**interface Abc {**

**}**

**Spring JDBC :**

**Spring provided JdbcTemplate API, This API wrap core JDBC and provided more method to improve the DAO layer using JDBC.**

**interface Abc {**

**void dis();**

**}**

**new Abc(){**

**public void dis() {**

**}**

**}**

**Spring Framework**

**Day 3 - 18-10-2019**

**Spring MVC : Spring MVC internally follow MVC Architecture.**

**Provided lot of API on Service as well Controller layer. Provided extra Spring JSTL tag to improve View layer. Spring MVC internally follow Front Controller design pattern. It provided pre-defined class ie DispatcherServlet it behave as a FrontController.**

**Struts ---> ActionServlet 1.x**

**2.x Filter StrutsPreparedAndExecute**

**JSF --> FacesServlet**

**Spring -->DispatcherServlet**

**FrontController we can configure in web.xml file or using Java classes.**

**Struts web.xml ---> actions.xml or struts.xml**

**JSF --->web.xml --->faces-config.xml**

**Spring MVC -->web.xml -----> Servletname-servlet.xml**

**return** jt.queryForInt("select \* from login where user like ? and pass like ?", **new** Object[] {ll.getUser(),ll.getPass()});