PLACEMENT QUESTIONS ANSWERS

JAVA BY KIRAN

ADVANCE JAVA

WRITTEN BY – VAIBHAV ANAND SHARMA

- Spring Boot
- Maven Project
- Aspect oriented programming (AOP)
- Inversion of Control (IOC)

Placement Questions and Answers

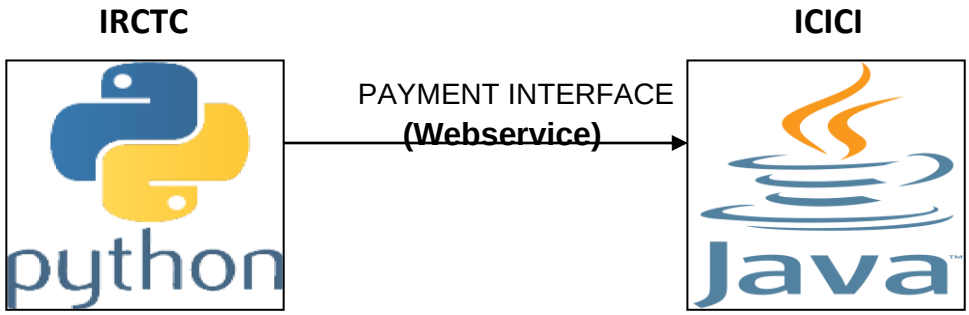
"SPRING BOOT"

Q.1 what are web services ?

Ans Web service is a technology to communicate between two same programming languages or two different programming languages of two different projects or applications.

Or

The method of communication between two devices over the network. It's a collection of protocols or standards for exchanging information between two devices or applications.



Types of Webservices :

- SOAP (SIMPLE OBJECT ACCESS PROTOCOL)
- REST (REPRESENTATONAL STATE TRANSFER)

Rest web services can be develop by many programming languages like java, python , .Net , PHP.

Q.2 what are the different technologies in java to build REST Webservices ?

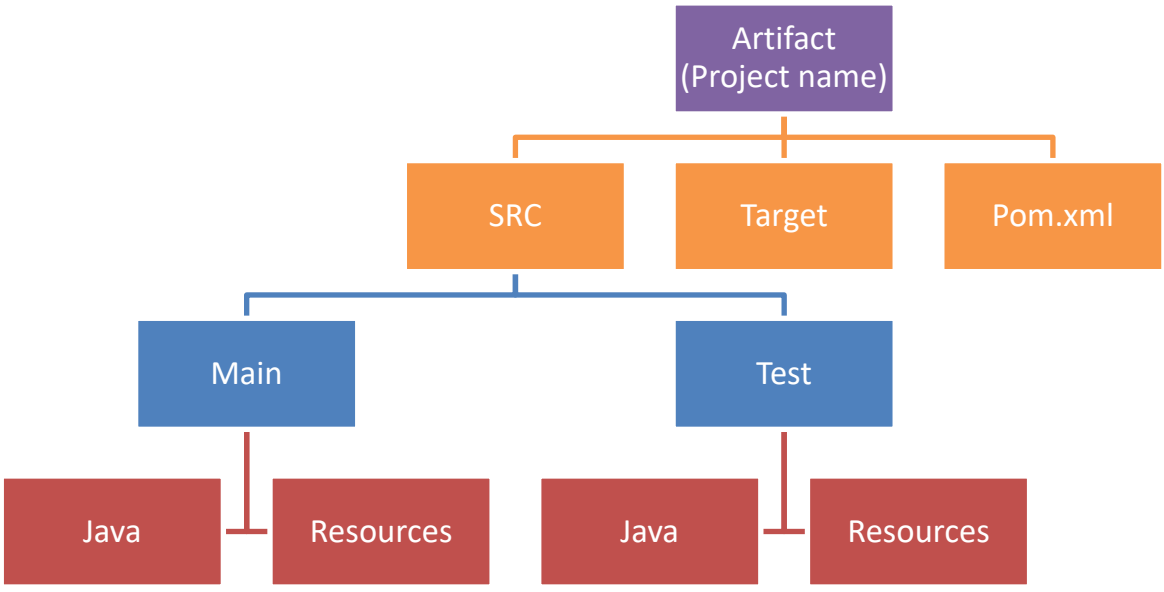
Ans There are following technologies used to build REST Webservices :

- Servelets
- Core java
- Jersy
- Spring
- Spring Boot

Q. 3 What is Maven ?

Ans Maven is a dependency management tool. It uses local and remote repository where .m2 is the local repository. It has pom.xml file in which we can mention dependencies and it download automatically the related jars. Maven download main jar and its dependent jars as well which can be very hectic to manage. So maven maintain dependencies.

Folder structure :



Q.4 What is Spring ?

Ans The Spring Framework (Spring) is an open-source application framework that provides infrastructure support for developing Java applications. Spring helps developers create high performing applications using plain old java objects (POJO).

But in spring we need to implement from scratch.

Q.5 What is Spring Boot ?

Ans Spring Boot is an open source, microservice based java web framework. The Spring Boot framework creates a fully production-ready environment that is completely configurable using its prebuilt code within its codebase.

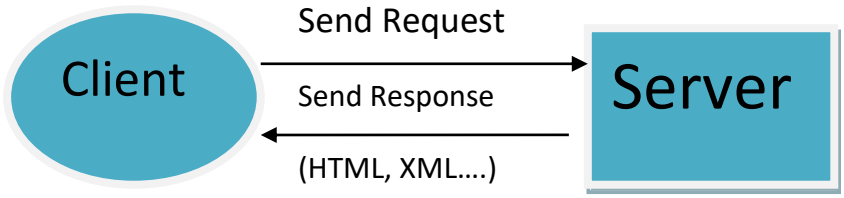
Spring boot is introduced to reduce the effort make by programmer and to reduce time taken to write piece of codes which are important but repetitive (Boiler plate code).

Q.6 What is REST API ?

Ans Representational State Transfer (REST) is an architectural style that defines a set of **rules or protocols** to be used for creating web services. It is used to give or fetch some information from web service.

Working :

A request is sent from client to server in the form of web URL as HTTP GET or POST or PUT or DELETE request. After that , a response comes back from server in the form of resource which can be anything like HTML, XML, Image or JSON. But now JSON is the most popular format being used in Web Services.



Q.7 What are the different types of methods in HTTP ?

Ans In HTTP there are some methods which are commonly used in a REST based architecture i.e. POST, GET, PUT, DELETE these are used to create, read, update and delete (OR CRUD) operations.

@GetMapping - This HTTP method is used to read or retrieve a representation of resource. GET returns a representation in XML or JSON.

@PostMapping - This is used to create new resources. Data posted in JSON or XML by hitting URL get converted into object by @RequestBody.

@PutMapping - This HTTP method is used to update in data.

@DeleteMapping - This HTTP method is used to Delete the data.

Q.8 What is JSON ?

Ans JavaScript object notation (JSON) is the lightweight data interchange format. JSON is used to store data in the particular format and to perform data sending or receiving operation between client and service provider JSON is used.

Data traverse in the format of JSON which is send by producer of webservices after call by client.

Q.9 What is endpoint ?

Ans Endpoint is the URL produced by producer company to use by the client at the time of HTTP Request Calling. For example ***http://localhost:8080/showemployee*** here "showemployee" is the endpoint which is given by developer at the HTTP request like @GetMapping("showemployee").

Q.10 What is @RestController ?

Ans

- It identifies that we are creating a REST API.
- It is a class level annotation.
- It is a starting point for any URL or endpoint we hit

Q.11 What is @RequestMapping ?

Ans it's can be defined by following points :

- It allows all type of requests (Get, Post, Put, Delete)
- It is a method level annotation.
- We specify URL here.

Q.12 What is @Pathvariable ?

Ans It is used to receive any parameter from end point. Parameter provided in method is written in curly braces after endpoint and "/".

example `@GetMapping("shwoemployee/{ename}")`

Q.13 What is @RequestBody ?

Ans This is used to receive JSON data after POST by client and populate that data into java object. Means JSON send by client is converted into object by using @RequestBody annotation which is written in method parameter section with Class and object.

```
Example :  public void getData(@RequestBody Student student)
```

Q.14 What are the different type of HTTP statuses ?

Ans The Status-Code element in a server response, is a 3-digit integer where the first digit of the Status-Code defines the class of response and the last two digits do not have any categorization role. There are 5 values for the first digit:

1	**1xx: Informational** It means the request has been received and the process is continuing.
2	**2xx: Success** It means the action was successfully received, understood, and accepted.

3	**3xx: Redirection** It means further action must be taken in order to complete the request.
4	**4xx: Client Error** It means the request contains incorrect syntax or cannot be fulfilled.
5	**5xx: Server Error** It means the server failed to fulfill an apparently valid request.

Given below is a list of all the HTTP status codes:

- 200 OK : The request is OK
- 400 Bad Reques : The server did not understand the request.
- 403 Forbidden : Access is forbidden to the requested page.
- 404 Not Found : The server can not find the requested page.
- 405 Method Not Allowed : The method specified in the request is not allowed.
- 500 Internal Server Error : The request was not completed. The server met an unexpected condition.
- 505 HTTP Version Not Supported : The server does not support the "http protocol" version.

Q.15 Explain integration of SpringBoot with hibernate.

Ans) To integrate hibernate with spring boot we require to do following steps :

- First of all we should have JDK, Eclipse, MySQL installed.
- Create Maven project and add dependencies in pom.xml file like JPA, Spring Web, Devtools and MySQL connector of same version as JPA and MySQL.
- OR we can use spring io initializer and create existing maven project by adding files in directory and finally adding JPA and MySQL connector only.
- Now we need Application.Properties file in which we mention the configuration which is fixed like username, ID, password etc.

- Now we need to create controller class in which we need to write SessionFactory object with @Autowire and for this we need a POJO class with @Component and @Entity and @Table annotations at class level and all variables we need to mention in table as column with @ID annotation at the variable used for Primary key.
- Now we require starter class to run our project and connect database it will create table automatically because Application.properties file has function to create table automatically.

Q.16 Explain the build life cycle of maven.

Ans A build life cycle of a maven project consists of a sequence of build processes , and each build phase consists of a sequence of goals , which define the order in which the goals are to be executed. Here phase represents a stage in life cycle. As an example, a typical **Maven Build Lifecycle** consists of the following **maven goals**.

- Clean
- Default (Build, Build…)
- Compile
- Test
- Install

Explaination:

a) **Clean** : Deletes the files from target folder if any recent files are present.
b) **Build/ Build…** : Build is used to set the goal like compile or clean or install and once the goal is set it works as default but if we have to set custom goal then we use Build…
c) **Compile** : This compile the all the source files and .Class file (byte code) is saved in target folder.
d) **Test** : Test cases will run if that are in the test folder it is for testing our project.
e) **Install** : In this goal the jar file of our .Class file is created and moved to the local repository (.m2 folder).

Q.17 How to integrate two Maven projects ?

Ans To integrate one maven project with other the initial things we need is the jar file of the project which is need to be integrated with the other project, in this question we will cover the steps require to achieve this goal with example, suppose we have two projects (Artifacts) named ***com.goi.healthdata*** and ***com.goi.minofhealth*** both are belongs to goi group, and we need to fetch ***com.goi.healthdata's*** data in ***com.goi.minofhealth*** for that integration of both project is require.

a) First of all we will create a maven project and by providing GroupID and Artifact (project name).
b) Then we will create a package and a class in which we write the data of ***com.goi.healthdata*** .
c) Then we will compile the project and .class file will be generated in target folder.
d) Then we install the maven project after it jar file of our project with version name created and moved to local repository (m2 folder).
e) Now we have two choice first is copy the dependency of ***com.goi.healthdata*** or to copy the jar file from .m2 folder.
f) If we copy the data from pom.xml file of ***com.goi.healthdata*** project then we will have to add this dependency in ***com.goi.minofhealth*** project's pom.xml file or we will add the jar in external jar lib of ***com.goi.minofhealth*** .
g) After adding jar we can easily access the class and members of the ***com.goi.healthdata*** and the projects will be integrated successfullty.

Q.18 Explain Aspect Oriented Programming (AOP) in detail .

Ans) AspectJ oriented programming as the name suggests that it uses "Aspect" in programming. It can be defined as the breaking of code into different module. Where "Aspect" is the key unit to achieve modularity. And breaking parts have called concerns.

Aspect class enables the implementation of cross cutting concerns those are beyond the business logic it can be regular annotated class annotated @Aspect.

Some frameworks used for AOP are JBOSS and AspectJ.

Q.19 Why we use AOP ?

Ans) AOP is used to introduce new interfaces to any advised object. With this we can perform some activities like run some particular message before or after advised method. OR We can add dynamically additional concerns before or after advised method.

Q.20 What is Advice ?

Ans) It is an action taken by the Aspect at a particular point like before or after method. Advice divided into following parts :

A. **Before** : Runs before the advised method is invoked.
B. **After** : Runs after the advised method completes, regardless of the outcome, whether successful or not.
C. **Around** : This is strong advice it runs before and after the advised method.
D. **Throwable** : If any exception occurs throwable advice is used.

Q.21 What is target ?

Ans) In AOP we can say Target to the methods which are advised to perform the particular activity before or after.

Q.22 How to implement AOP ? Explain in detail.

Ans) There are following steps to implement AOP :

- After successful installation of Spring we can use AOP.
- We need to add dependency of AOP to perform this first.
- Secondly we need to provide @EnableAspectJAutoProxy annotation at the class level of starter class of Spring boot application.
- Then we have to create a class to define advised method which require @Aspect at class level with @Component at class level and creation of methods.
- Now put advice annotation at method level and message to print in method annotation with the location of class which has targets as follow : @Before("execution(*com.jbk.package.ClassName.*()")

Q.23 What is IOC ?

Ans. IOC (Inversion Of Control) is a concept not the part of Spring framework it means to create the object of a class and call its method in another class to invert the related coding in another class just to make code simple and understandable.

Or

Spring IOC container creates the object configure and assemble their dependencies manage their entire life cycle.

The container use dependencies injection (DI) to manage the component that makes up the application. It gates the information about objects from Annotations or from conf. file (XML, JSON) or from Java code. These objects called **"Beans"** .

In the IOC concepts we divide our work in different classes like

- Controller : to receive request and send response.
- Service : to write the business logic.
- DAO (Data access object) : data base related code.

Note : Every class should has "Bean" , bean is nothing but Obejct , so it can be used for injection.

Q.24 Explain injection ?

Ans. In spring framework injection means inject the members of a class into another class , just similarly we do in core java by creating object with new keyword and constructor call in spring we define bean and inject using @Autowired annotation.

For injection controller class require a bean of Service class which is connected by @Autowired annotation in controller class and @Service in Service class at class level.

Q.25 Explain @Component ?

Ans. This annotation is preferably used in POJO class at class level but it can also be used in Service and Dao class at class level. @Component means now this class is the part of Spring we can inject this class with the help of @Autowired.

Q.26 Explain @Autowired ?

Ans. This annotation is used to inject beans (objects) of the classes which have @Component or @Service or @Repository at the class level. This annotation is used on the Object of class which has to be injected.

Q.27 Explain @Service ?

Ans. @Service means now this class is the part of Spring we can inject this class with the help of @Autowired. While working with the service layer we need to use this annotation but internally @Component annotation is used.

Q.28 Explain @Repository ?

Ans. While working with JDBC or Hibernate it is recommended that we separate that code from service or controller. At class level we should mention @Repository while writing database code.

Q.29 Explain @Value ?

Ans. This annotation is used to inject Strings and primitives or values from Appilcation.property file. Which we created manually or defined by default by spring in the form of Key and value, we use key and get value.

@Value(" ${key}")

Q.30 Explain @Bean ?

Ans. By using this annotation object will be created manually by us and will be injected instead of using @Autowire.

Q.31 Explain @Qualifier ?

Ans. When we create more than one bean of same type and want to wire only one of them with a property, in this case we should use @Qualifier along with @Autowired to remove the confusion by specifying which exact bean will be wired.

Q.32 Explain @ComponentScan ?

Ans. It scans all the packages mentioned in the project. This is used at the starter class which is start of Spring boot application at class level.

When our component class is in another package that cannot be injected using @Autowired to do this we use @ComponentScan is used at class level with @SpringBootApllication.