ADVANCE JAVA INTERVIE QUESTIONS

1) What is difference between jdbc and hibernate?

Ans: -

- JDBC is a technology and hibernate is a framework.
- JDBC works with rows and columns (RDBMS approach) whereas hibernate works with class and variables (object-oriented approach).
- JDBC does not built-in functions for performing CRUD operations on Database.
- Hibernate has built in methods for performing CRUD operations.
- JDBC code is database dependant code and Hibernate code is Database independent code.

2) What are in built methods in hibernate? Can u explain each of them in brief.

Hibernate has below built-in methods: -

1) save(Object o): - It saves given object's state (object's data) into database table.

```
Student obj = new Student (1,90);
session. save(obj);
```

2) delete(Object o): - It delete given object's data from the database table.

```
Student std =session.load(Student.class, 1);
session.delete(std);
```

3) update(Object o): - It update object's data.

```
Student std = session.load(Student.class,1);
std.setstudAddress("Mumbai");
session.update(std);
```

4) Object load(Class c, Serializable id): - it loads data of an object from the database.

e.g. load(Student.class,1) will load rollnumber 1's data from the database and gives us object of Student class [rno=1 marks=90] Student class object

```
Student std = session.load(Student.class,1);
```

3) Can you explain Criteria in hibernate?

Criteria API is used to load object's data from database based on some conditions.It has Restrictions class which contains many methods for writing condition on what basis object's data will be loaded.

e.g. Criteria criteria=session.createCriteria(Student.class);

criteria.add(Restrictions.gt("rno", 1)).add(Restrictions.like("studName","%k%"))

It will load those students whose rno is greater than 1 and whose name has k character Criteria API contains Projections class using which we can read partial object means only some property's values not all property's values. e.g. If Student has 4 properties like rno, name, age, x marks and we want only 2 property's values rno, age then using Projections class, it is possible to do it.

Projections class also contains some aggregate functions like **rowCount(), max(),min(),sum().**

4) Can you explain Query in Hibernate?

Query interface is used to execute HQL queries. for executing select HQL query we use **list()** and for non-select query we use **executeUpdtae().**HQL queries are Database independent queries whereas SQL queries are database dependant queries.

5) What are files required to do simple insert operation in hibernate?

1) Hibernate configuration file: - it contains Database connection details; some hibernate properties like hbm2ddl.autodialect class , show_sql etc.

2) Entity class: - This is the class whose object's contents will be saved in a database using **save()**. @Entity annotation is used to specify that class is Entity class. It should contain primary key field which is marked with @Id annotations.

3) Client class: - This class contains **main()** method where we call **save()** of session interface for saving object's contents

Note: - All hibernate jar files should be kept in project's build path

6) What are advantages of hibernate?

- Hibernate works with class and variables (object oriented approach). So, we don't need to have SQL knowledge (RDBMS).
- JDBC code is database dependant code and Hibernate code is Database independent code. So even if we change the database, we don't need to rewrite HQL queries Hibernate speed up development process as it contains many built-in methods for doing some common database operations In hibernate we don't have methods which are throwing checked exception, otherwise in JDBC, we had to compulsory handle exception as JDBC methods throw checked exception.

7) What is difference between servlet and jsp?

- Servlet contains java code whereas jsp contains tags.
- Servlet is faster than JSP whereas JSP is slower as it converts into Servlet first and then compiles.
- In Traditional MVC based web application, servlet plays role of controller whereas JSP is used to show response on web page.
- using HTML tags in servlet is not easy, whereas in jsp , we use html tags along with jsp tags easily.
- Modification in servlet is time consuming as it needs reloading, recompiling and restarting the server whereas in jsp , we just need to refresh JSP page to see effect of modification.

8) Which is faster servlet or jsp?

Servlet is faster than JSP whereas JSP is slower as it converts into Servlet first and then compiles.

9) Can you explain lifecycle of servlet explain in detail?

Servlet Container is part of Server program.

We know that Servlet Container manages the life cycle of Servlet, there are four phases of servlet life cycle.

A: Servlet Class Loading – When container receives request for a servlet, it first loads the class into memory and calls its default no-args constructor.

B: invoking init() – Servlet container invoke servlet's init method for performing some initialization.

C: Request Handling – Servlet container invokes the **service()** method.

service() method calls **doGet()** or **doPost()** based on which request method was used by client while issuing request.

D: Removal from Service – When container stops or we stop the application, servlet container destroys the servlet class by invoking it's **destroy()** method.

10) Can you explain lifecycle of jsp explain in detail?

JSP lifecycle

1) Translating jsp to servlet (test.jsp➔test_jsp.java).

2) Compiles the translated servlet (test_jsp.java➔test_jsp.class)

3) Loads the translated servlet (test_jsp.class file).

4) Creates the instance of translated servlet.

5) Calls the lifecycle method _jspInit ()

6) When user sends the request then container calls the lifecycle **method _JspService ( )**

7) At container shutdown time, container calls lifecycle **method _JspDestroy ( )**

Translation phase is extra, once translation to servlet done then servlet lifecycle starts.

11) Can you explain complete flow of on scenario where student will get added into database from jsp?

In traditional web application development, from JSP page, we can get details of Student object from user. e.g., we will ask user to enter username, age, mobileno of Student through a web page. This data will be given to server. server will create request object using this data.

we will retrieve this data using **getParameter()** of request object and then we will add it in database using JDBC In Spring MVC, Spring will create object of Student class using data submitted by user through web page. Then we can use Hibernate to add this object into database.

12)What are implicit objects of jsp?

there are Certain objects that are available for the use in JSP documents without being declared first, these objects are called implicit objects.

The frequently used implicit objects are listed below:

request: - it points to HttpServletRequest object. It contains data submitted by user through browser.

response: - it points to HttpServletResponse object.

session: - it points to HttpSession object. It should be used to store user specific data like username, items selected by user in online shopping. session object is global object as it is available everywhere in web application. Hence, we use it to store that data which we require in all JSP

out: - it points to JSPWriter object. it is used to print something on a browser.

13) How to write servlet?

we define a class which extends HttpServlet class. Then we override **doGet()** or **doPost().**

@WebServlet defines URL, which is used to invoke servlet.

```
@WebServlet("/hello")

public class Hello extends HttpServlet
{
    public void doGet(HttpServletRequest request, HttpServletResponse response) throws Exception
    {
        // write some logic which will either generate web page or get a web page which will be given to user who invoked servlet
    }
}
```

To invoke this servlet in browser we write: -

http://localhost:8080/mywebapp/hello

here localhost:8080 is location where tomcat server is running.

mywebapp is name of web application hello is URL of servlet.

14) What is difference between the init(), service() and destroy() method?

-Once Servlet object is created, **init()** is called only once while initializing servlet.

-service() is called for every request, so that appropriate response will be generated for user's request.

-destroy() is called by Servlet container When container stops or we stop the application. it removes Servlet object from memory.

15) What is difference between servlet jsp programming and spring MVC?

In traditional web application, we have to create input object manually, whereas in Spring we get it readymade.

In traditional web application, we had to create object of user defined class User using client supplied data.

whereas in spring-based web application, Spring creates object of user defined class User using client supplied data.

Also, Spring provides a lot of annotation, which makes web development fast and easy.

we do not need to use XML based configuration in Spring MVC.

Server is a not managing life cycle of Spring MVC application, DispatcherServlet is used for that. It means less dependency on server. But in traditional web application development, Server is responsible for entire flow of web application.

In traditional web application development, we had to configure Tomcat Server manually inside eclipse.

In Spring MVC, we do not need to configure Tomcat Server manually. It is embedded into Spring project.

16) Tell me some frameworks available in java. Explain them in detail?

Spring framework and Hibernate Framework are used mostly out of many frameworks available.

Spring framework is used to develop web services using Spring REST API inside Spring Boot Project

Spring framework is used to develop web application using Spring MVC inside Spring Boot Project.

Hibernate is used to write database logic which is database independent and object oriented. It gives HQL, Criteria API, many built-in methods which makes writing database logic very fast as compare to JDBC.

17) Why to use spring?

Spring is used for Rapid Web Application development as well as for developing web services.

Spring provides a lot of annotation, which makes web development fast and easy.

we do not need to use XML based configuration in Spring MVC

Server is a not managing life cycle of Spring MVC application, DispatcherServlet is used for that. It means less dependency on server. But in traditional web application development, Server is responsible for entire flow of web application.

In traditional web application development, we had to configure Tomcat Server manually inside eclipse. In Spring MVC, we do not need to configure Tomcat Server manually. It is embedded into Spring project.

18) What is IOC in spring explain in detail?

Through Spring IOC, life cycle of java object (bean) is managed by Spring container.

Dependent object is injected into container bean object automatically which is called dependency injection.

e.g.

```
class LoginService
{
	@Autowired
	LoginDAO dao ;
	boolean validate(User user)
	{
		dao.getPasswordFromDatabase(user.getUserName());
	}
}
```

@Autowired annotation indicates that Spring will create object of LoginDAO class, we do not need to create it.

It means we DO NOT NEED to write below line: -

```
LoginDAO  dao = new LoginDAO();
```

It provides loose coupling (aggregation) as we just declare dependent object in a container object and don't create that object manually (Composition/strong coupling).

19) What is AOP in spring explain in detail?

Suppose we have to write method to transfer money. then we can define it like below: -

```
void  transferMoney(int SourceAccountNo,int destinationAccountNo , double amount)
{
	// some user validation logic
	// then actual business logic
	SourceAccountNo.balance = SourceAccountNo.balance - amount;
	destinationAccountNo .balance = destinationAccountNo .balance + amount;
	// then Transactional logic to decide whether to commit or rollback transaction
}
```

However, this is NOT ideal way to define such method.

Spring AOP suggest that in business logic method only business logic should be written and should not write any other logic in it.

Other logic can be injected into business logic method later.

```
class Transfer {
        void  transferMoney(int SourceAccountNo,int destinationAccountNo , double amount)
        {
                // ONLY business logic
                SourceAccountNo.balance = SourceAccountNo.balance - amount;
                destinationAccountNo .balance = destinationAccountNo .balance + amount;
        }
}
@Aspect
public class AspectCode
{
   @Before("execution(* com.javabykiran.Transfer.transferMoney())")
   boolean validate(){   }
   @After("execution(* com.javabykiran.Transfer.transferMoney())")
   boolean transactionStatus() {  }
}
```

Now, **validate()** will execute before business logic written in **transferMoney()** and **transactionStatus()** will execute after business logic written in **transferMoney()**

It means @Autowire is used to inject object and @Aspect is used to inject methods.

20) What is spring MVC can you explain flow from the jsp to Database?

Consider Registration process. Whenever user register himself, he provides some data.

In Spring MVC, Spring will create object of User class using this data, Then we can use Hibernate to add this object into database.

1) define Entity class User. we will have table with name user in database having columns userName and userPassword .

```
@Entity
public class User {
	@Id
	String userName;
	String userPassword;
}
```

register.jsp page: -

```
<form action="saveUserData">
	<input type="text" name="userName" placeholder="enter user name"><br><br>
	<input type="password" name="userPassword" placeholder="enter password"><br><br>
	<input type="submit" value="register">
</form>
```

Make sure that entity class variable names and input element's names are same, so that Spring will create object of User class using data submitted by user inside input element (tag).

```
@Controller
public class RegistrationController {
	@Autowired
	SessionFactory factory;
	@RequestMapping("saveUserData")
	String saveUserData(User user)
	{
		Session session=factory.openSession();
		Transaction transaction =session.beginTransaction();
					session.save(user);
		transaction.commit();
		return "login";
	}
}
```

21)What is difference between servlet jsp MVC and Spring MVC?

In servlet jsp MVC, Servlet class is used as Controller class whereas in Spring MVC, simple java class classes are used as Controller classes In servlet jsp MVC, to decide which data will be displayed on view page (jsp page), we had to use request. setAttribte("data",arrayList) explicitly in Servlet.

In Spring MVC, to decide which data will be displayed on view page (jsp page), we DO NOT NEED to use request.setAttribte("data",arrayList) explicitely in Controller classes.

```
ModelAndView  mv = new ModelAndView();

mv.setViewName("welcome");

mv.addObject("data",arrayList);
```

In traditional Servlet JSP MVC web application, we had to create object of user defined class User using client supplied data.

whereas in spring-based web application, Spring creates object of user defined class User using client supplied data.

In Spring, due to dependency injection (@Autowired), we don't need create object manually. In servlet jsp MVC, we had to do it manually.

22)What all annotations you know in spring explain each of them any 5 annotations?

In spring Object is called as bean.

Spring Boot Annotations

1. @Autowired

Spring container is going to create the object

```
class LoginService

{

        @Autowired

        LoginDAO dao;

}
```

LoginDAO object will be created by Spring automatically as we have used @Autowired.

2. @Component

it allows spring to auto detect classes for object creation.

It is class level annotation Spring boot will create object of these classes at the time of starting application.

@Controller, @Service, @Repository are similar to @Component. They are also autodetected for object creation during start-up of application.

3. @Value

if we want to read any key from application.properties file then we can read it using @Value

```
application.properties
country = india
@Value("${country}")
private String country;
```

4. @Bean

if u want to create object of class explicitly, we use @Bean at method level. Here we can assign values of our choice to

attributes of bean object. whereas @Component create object using default constructor.

5. @RestController

class will be identified as the Rest Controller Class where we define some methods which are called using URL assigned to them.

```
@RestController = @Controller + @ResponseBody
```

6. @RequestMapping

used for mapping the URL

```
http://localhost:8080/hi
```

In browser, hi.jsp page will be displayed.

```
class HelloWorldController
{
	@RequestMapping("hi")
	public String hi()
	{
		return "hi";
	}
}
```

can be used at class as well as method level

7. @ComponentScan

used to specify directory where Component classes are present.

23)What is difference between the spring and spring boot?

Spring is a framework using which we can develop web application and web services both. Before Spring Boot, Spring developers would develop this application using XML based configuration?

To make these development easy, Pivotal software company gives us a Maven based project which is called as Spring Boot project.

In a spring boot project, dependencies are managed using pom.xml file, no need to configure tomcat server separately, no need to write configuration using XML.

24) what is Spring Boot? Advantages of Spring Boot?

Spring Boot is open-source framework maintained by company called Pivotal.

Spring Boot is completely new project from Spring community.

When Spring Boot project is used, we don't need to write configuration using XML.

We use properties file for writing configuration.e.g., to specify on which port number tomcat server should run we use properties file to specify it. e.g., in application.properties file we write like below: -

application.properties

server.port=8087

Spring boot shortens development required time as it provides features like Live Development Tools, lot of annotations to make development of Rest webservices easier.

Spring Boot provides embedded tomcat server. No need to configure tomcat server separately like we do in traditional web applications.

25) what is REST WebService or Rest API?

Answer: - It is special program whose methods can be called using URL. Any program of any language can call these methods using URL.

It means it is language independent. REST means Representational State Transfer. It means REST webservice sends state of an object in the form of JSON String. State of an object means object's data. e.g., new Student(1,90), here rno=1 and marks=90 is state of an object.

request method: - get and request URL: - localhost:8082/student/1. It should return JSON String having data of Student with rno 1

Output JSON :- {"rno":1 , "marks":90}

```
// @PathVariable annotation assign value of path variable to local variable

//@GetMapping("student/{rno}"). rno is path variable

@GetMapping("student/{rno}")

Student getStudent(@PathVariable int rno)
```

```
{
                    for(Student student : al)
                    {
                        if(student.rno==rno)
                        {
                            System.out.println(student);
                            return student;
                        }
                    }
                    return null;
        }
```

26) How to develop REST API?

```
public class Student
{
        int rno ,marks;
}
/* @RequestBody annotation will create object of Student class using json string data sent by client (postman) */
        @PostMapping("student")
        List<Student> addStudent(@RequestBody Student student)
        {
            al.add(student); // we are adding object in ArrayList
            System.out.println(al);
            return al;
        }
```

27)What is webservices?

Answer: - It is special program whose methods can be called using URL. Any program of any language can call these methods using URL.

It means it is language independent. REST means Representational State Transfer. It means REST webservice sends state of

an object in the form of JSON String. State of an object means object's data. e.g., new Student(1,90), here rno=1 and marks=90 is state of an object.

request method: - get and request URL: - localhost:8082/student/1. It should return JSON String having data of Student with rno 1

Output JSON :- {"rno":1 , "marks":90}

28)Do you remember annotations in spring boot that makes your job easy?

1) @RestController: - If this annotation is used at the top of class, then that class becomes Spring REST Webservice class means this class will have methods which will return json string as an output.

2) @RequestMapping: - Any request whether it is get/post/put/delete is handled by a method which is annotated with @RequestMapping.

3) @GetMapping: - If method is annotated with @GetMapping then this method will handle get request. From browser we can issue only get request. To issue other type of request like post, put we have to use postman or any other client application developed using any client-side technology like JavaScript. methods annotated with @GetMapping, will return JSON String, representing state of an object (object's data).

4) @PostMapping: - If method is annotated with @PostMapping then this method will handle post request. It is used to create new entity/resource/object. For this it needs JSON String from client.

5) @PutMapping: - If method is annotated with @PutMapping then this method will handle put request. It is used to update existing entity/resource/object. For this it needs JSON String from client.

6) @DeleteMapping: - If method is annotated with @DeleteMapping then this method will handle delete request means it will delete existing entity/resource/object.

7) @PathVariable: -

@PathVariable annotation assign value of path variable to local variable

e.g. request method: - get and request URL: - localhost:8082/student/1

then path variable rno's value 1 will be applied to **getStudent()** methods's local variable rno.

@GetMapping("student/{rno}") // here rno is path variable (path means URL. variable which is part of URL is called path variable)

Student getStudent(@PathVariable int rno);

29) what is use of properties file in Spring Boot project?

Answer: - Properties file is used to define configuration like Database Connection details, Hibernate Properties, Server's port number etc.

e.g., SpringBootProject\src\main\resources folder contains application.properties file where we can wite below details: -

```
server.port=8085
spring.datasource.url=jdbc:mysql://localhost:3306/test
spring.datasource.username=root
spring.datasource.password=root
spring.datasource.driver-class-name=com.mysql.jdbc.Driver
spring.jpa.hibernate.ddl-auto=update
spring.jpa.database-platform=org.hibernate.dialect.MySQL5Dialect
spring.jpa.show-sql=true
```

30) what is runner class in Spring Boot project?

Answer: - In Spring boot project, we have one class with main method, which we execute to start Spring Boot application.

```
@SpringBootApplication
public class BasicMicroServiceProjectApplication
{
	public static void main(String[] args)
	{
		SpringApplication.run(BasicMicroServiceProjectApplication.class, args);
	}
}
```

@SpringBootApplication indicates that it is runner class.

31) what is Spring initializer?

Spring Initializer is a web-based tool provided by the Pivotal Web Service. With the help of Spring Initializer, we can easily generate the structure of the Spring Boot Project. we need to visit https://start.spring.io/. Here we need to specify some details and we will get ready made spring Boot maven project. We specify dependencies here. Dependencies means Jar files which are required to run our project. e.g., If our project having Hibernate Code then we need Hibernate Jar files, MySQL jar files. We specify all these dependencies here.

Spring Boot Maven Project will download all required jar files from Maven Repository Website and add those jar files in project's build path.

To import this project, we need to follow below steps: -

1) select file menu 2) select import 3) select maven 4) select existing maven projects 5) select root directory of project

32) what is JSON?

JSON is a string written in a JavaScript Object Format. JSON (JavaScript Object Format)

It is used for exchanging data between applications.

It consists of name and value pairs, where names are always written in double quotes

e.g. { "rollNumber" : 1 , "marks" : 90 }

here rollNumber is name and 1 is value. marks are name and 90 is a value.

Any program of any language can understand JSON string as it is just a strings and string can be understood by any language

33) Where do we use REST API?

Answer: - REST API is used to exchange data between applications. This data is mostly in the form of JSON String.

e.g., If JBKTest.com wants to display cricket score on its web page, then it can contact cricketbuzz.com and ask them to provide current score. Then cricketbuzz.com will give score in the form of JSON String to JBKTest.com. Then JBKTest.com will process data from this JSON String on its web page.

cricketbuzz.com must have one method which gives this JSON String having current score. JBKTest.com will use URL of this method to request it.

e.g.

```
@RestController("api")
class ManageScore
```

```
{
	@GetMapping("getScore")
	Score getScore()
	{
		// return Score object which will be converted into JSON String and passed to client
	}
}
```

JBKTest.com will use below URL to invoke this method: -

https://cricketbuzz.com/api/getScore

34) How to run REST webservice?

Steps to run Spring REST API program

1) run BasicMicroServiceProjectApplication class.

2) open postman.

3) request method: - get and request URL: - localhost:8082/all Students. It should display 2 students objects from ArrayList.

4) request method: - get and request URL: - localhost:8082/student/1. It should display students object from ArrayList with rno 1.

5) request method: - post and request URL: - localhost:8082/student. We need to send JSON string to server. hence select body==>raw==>JSON and then type {"rno":3,"marks":80} and then press send This will add new Student object into arraylist .

6) request method: - get and request URL: - localhost:8082/allStudents. It should display 3 students objects from ArrayList . 3rd object which we just added should be displayed along with previous 2 objects.

7)request method: - put and request URL: - localhost:8082/student. We need to send JSON string to server. hence select body==>raw==>JSON and then type {"rno":3,"marks":100} and then press send This will update Student object 3rd's marks to 100

8) request method: - get and request URL: - localhost:8082/allStudents. It should display 3 students objects. observe marks of rno 3 is 100 now.

9) request method: - delete and request URL: - localhost:8082/student/1. It should remove student object from ArrayList with rno 1.

10) request method: - get and request URL: - localhost:8082/allStudents. It should display 2 students objects. observe rno 1 student is not present.