Java Interview Questions

edureka

Q3. What are the differences between processes and threads?

	Process	Thread
Definition	An executing instance of a program is called a process.	A thread is a subset of the process.
Communication	Processes must use inter-process communication to communicate with sibling processes.	Threads can directly communicate with other threads of its process.
Control	Processes can only exercise control over child processes.	Threads can exercise considerable control over threads of the same process.
Changes	Any change in the parent process does not affect child processes.	Any change in the main thread may affect the behavior of the other threads of the process.
Memory	Run in separate memory spaces.	Run in shared memory spaces.
Controlled by	Process is controlled by the operating system.	Threads are controlled by programmer in a program.
Dependence	Processes are independent.	Threads are dependent.

Q4. What are Wrapper classes?

Each of Java's eight primitive data types has a class dedicated to it. These are known as wrapper classes because they "wrap" the primitive data type into an object of that class

Primitive	Wrapper Class	Constructor Argument
boolean	Boolean	boolean or String
byte	Byte	byte or String
char	Character	char
int	Integer	int or String
float	Float	float, double or String
double	Double	double or String
long	Long	long or String
short	Short	short or String

Q5. What purpose does the keywords final, finally, and finalize fulfill?

final

- Final is used to apply restrictions on class, method and variable.
- Final class can't be inherited, final method can't be overridden and final variable value can't be changed.

```
class finalExample{
    public static void main(String args[]){
        final int a=1000;
        a=500;
    }
}
```

finally

- Finally is used to place important code, it will be executed whether exception is handled or not.

```
class FinallyExample{
    public static void main(String[] args){
        try{
                int x=300;
        }catch(Exception e){System.out.println(e);}
        finally
        {System.out.println("finally block is executed");}
    }
}
```

finalize

- Finalize is used to perform clean up processing just before object is garbage collected.

```
class finalizeExample{
    public void finalize(){
        System.out.println("Finalize is called);
    }
    public static void main(String args[])
    {
        finalizeExample f1 = new finalizeExample();
        finalizeExample f2 = new finalizeExample();
        f1 = NULL;
        f2 = NULL;
        System.gc();
    }
}
```

Q6. What is the difference between StringBuffer and StringBuilder?

- StringBuffer operations are thread-safe and synchronized where StringBuilder operations are not thread-safe.
- StringBuffer is to be used when multiple threads are working on same String and StringBuilder in the single threaded environment.
- StringBuilder performance is faster when compared to StringBuffer because of no overhead of synchronized

```
public class performanceTest {
    public static void main(String[] args) {
        int N = 999999999;
        long t;

        {
            StringBuffer sb = new StringBuffer();
            t = System.currentTimeMillis();
            for (int i = N; i --> 0 ;) {
                sb.append("");
            }
            System.out.println(System.currentTimeMillis() - t);
        }

        {
            StringBuilder sb = new StringBuilder();
            t = System.currentTimeMillis();
            for (int i = N; i --> 0 ;) {
                sb.append("");
            }
            System.out.println(System.currentTimeMillis() - t);
        }
    }
}
```

Q7. What are the differences between Heap and Stack Memory?

	Stack	Heap
Memory	Stack memory is used only by one thread of execution.	Heap memory is used by all the parts of the application.
Access	Objects stored in the heap are globally accessible	Stack memory can't be accessed by other threads.
Memory Management	Follows LIFO manner to free memory.	Memory management is based on generation associated to each object.
Lifetime	Exists until the end of execution of the tread.	Heap memory lives from the start till the end of application execution.
Usage	Stack memory only contains local primitive variables and reference variables to objects in heap space.	Whenever an object is created, it's always stored in the Heap space

Q8. What is the difference between Arrylist and Vector?

Array List	Vector
Array List is not synchronized.	Vector is synchronized.
Array List is fast as it's non-synchronized	Vector is slow as it is thread safe.
If an element is inserted into the Array List, it increases its Array size by 50%.	Vector defaults to doubling size of its array.
Array List does not define the increment size.	Vector defines the increment size.
Array List can only use Iterator for traversing an Array List.	Except Hashtable, Vector is the only other class which uses both Enumeration and Iterator.

Q9. What is the difference between HashMap and HashTable

HashMap	HashTable
HashMap is non synchronized and are not-thread safe.	Hashtable is synchronized. It is thread-safe and can be shared with many threads.
HashMap allows one null key and multiple null values.	Hashtable doesn't allow any null key or value.
HashMap is fast compared to HashTable	Hashtable is slow compared to HashMap
We can make the HashMap as synchronized by calling Collections.synchronizedMap(hashMap)	Hashtable is internally synchronized and can't be unsynchronized.
HashMap is traversed by Iterator.	Hashtable is traversed by Enumerator and Iterator.
HashMap inherits AbstractMap class.	Hashtable inherits Dictionary class.

Q10. What is the difference between equals() and == operator ?

- Equals() method is defined in Object class in Java and used for checking equality of two objects defined by business logic.
- "==" or equality operator in Java is a binary operator provided by Java programming language and used to compare primitives and objects.

```
1  public class equalTest {
2      public static void main(String[] args) {
3
4          String Str1 = new String("ABCD");
5          String Str2 = new String("ABCD");
6
7          if (Str1 == Str2)
8          {
9              System.out.println("String 1 == String 2 is True");
10         }
11         else
12             System.out.println("String 1 == String 2 is False");
13
14         String Str3 = Str2;
15
16         if (Str2 == Str3)
17         {
18             System.out.println("String 2 == String 3 is True");
19         }
20         else
21             System.out.println("String 2 == String 3 is False");
22
23         if (Str1.equals(Str2))
24         {
25             System.out.println("String 1 Equals String 2 is True");
26         }
27         else
28             System.out.println("String 1 Equals String 2 is False");
29
30     }
31 }
```

Q12. What is the difference between Abstract classes and Interfaces?

Abstract Class	Interfaces
An abstract class can provide complete, default code and/or just the details that have to be overridden.	An interface cannot provide any code at all,just the signature.
In case of abstract class, a class may extend only one abstract class.	A Class may implement several interfaces.
An abstract class can have non-abstract methods.	All methods of an Interface are abstract.
An abstract class can have instance variables.	An Interface cannot have instance variables.
An abstract class can have any visibility: public, private, protected.	An Interface visibility must be public (or) none.
An abstract class can contain constructors .	An Interface cannot contain constructors .
Abstract classes are fast.	Interfaces are slow as it requires extra indirection to find corresponding method in the actual class.

edureka!

Q14. What is runtime polymorphism or dynamic method dispatch?

- It is a process in which a call to an overridden method is resolved at runtime rather than at compile-time.
- In this process, an overridden method is called through the reference variable of a superclass.
- The determination of the method to be called is based on the object being referred to by the reference variable.

```
class Car{
    void run(){System.out.println("Car is running");}
}
class Audi extends Car{
    void run(){System.out.println("Audi is running safely with 100km");}

    public static void main(String args[]){
        Car b = new Audi();//upcasting
        b.run();
    }
}
```

edureka!

Q15. What is the difference between method overloading and method overriding?

	Overloaded Method	Overridden Method
Definition	In Method Overloading, Methods of the same class shares the same name but each method must have different number of parameters or parameters having different types and order.	In Method Overriding, sub class have the same method with same name and exactly the same number and type of parameters and same return type as a super class.
Behavior	Method Overloading is to "add" or "extend" more to method's behavior.	Method Overriding is to "Change" existing behavior of method.
Polymorphism	It is a compile time polymorphism.	It is a run time polymorphism.
Signature	The methods must have different signature.	The methods must have same signature.
Inheritance	It may or may not need inheritance in Method Overloading.	It always requires inheritance in Method Overriding.

edureka!

Q15. What is the difference between Method overloading and Method overriding?

Method overloading

```
class Adder{
    static int add(int a, int b)
    {return a+b;}
    static double add(double a, double b)
    {return a+b;}
}
class TestOverloading2 {
    public static void main(String[] args) {
        System.out.println(Adder.add( a: 11,  b: 11));
        System.out.println(Adder.add( a: 12.3,  b: 12.6));
    }
}
```

Method overriding

```
class Car{
    void run(){System.out.println("Car is running");}
}
class Audi extends Car{
    void run(){System.out.println("Audi is running safely with 100km");}

    public static void main(String args[]){
        Car b = new Audi();//upcasting
        b.run();
    }
}
```

edureka!

Q16. Can you override a private or static method in Java?

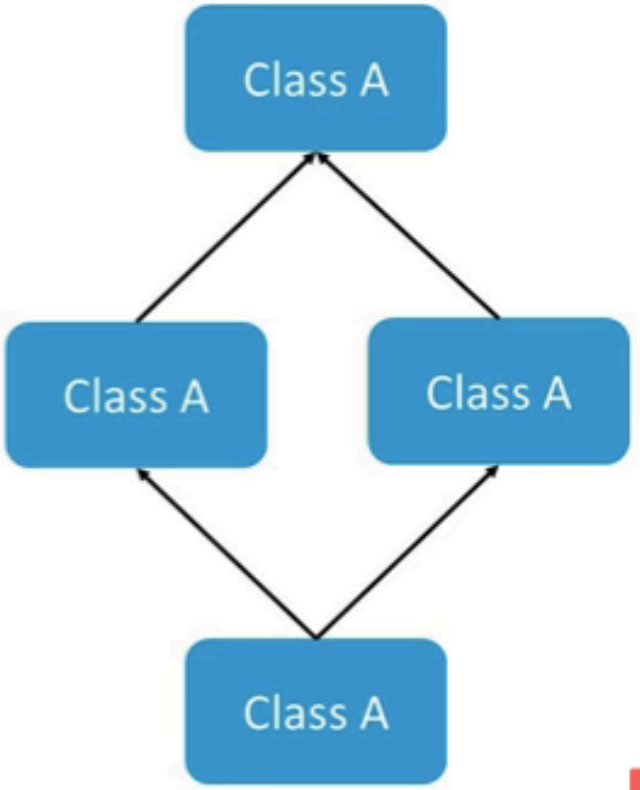
```
class Base {

    private static void display() { System.out.println("Static or class method from Base"); }

    public void print() { System.out.println("Non-static or Instance method from Base"); }
}

class Derived extends Base {

    private static void display() { System.out.println("Static or class method from Derived"); }

    public void print() { System.out.println("Non-static or Instance method from Derived"); }
}

public class Test {
    public static void main(String args[ ])  {
        Base obj1 = new Derived();

        obj1.display();

        obj1.print();
    }
}
```

- A private method cannot be overridden since it is not visible from any other class.
- If you create a similar method with same return type and same method arguments in child class then it will hide the super class method; this is known as method hiding.

edureka!

Q17. What is multiple inheritance and does java support?

- If a child class inherits the property from multiple classes is known as multiple inheritance.
- The problem with multiple inheritance is that if multiple parent classes have a same method name, then at runtime it becomes difficult for the compiler to decide which method to execute from the child class

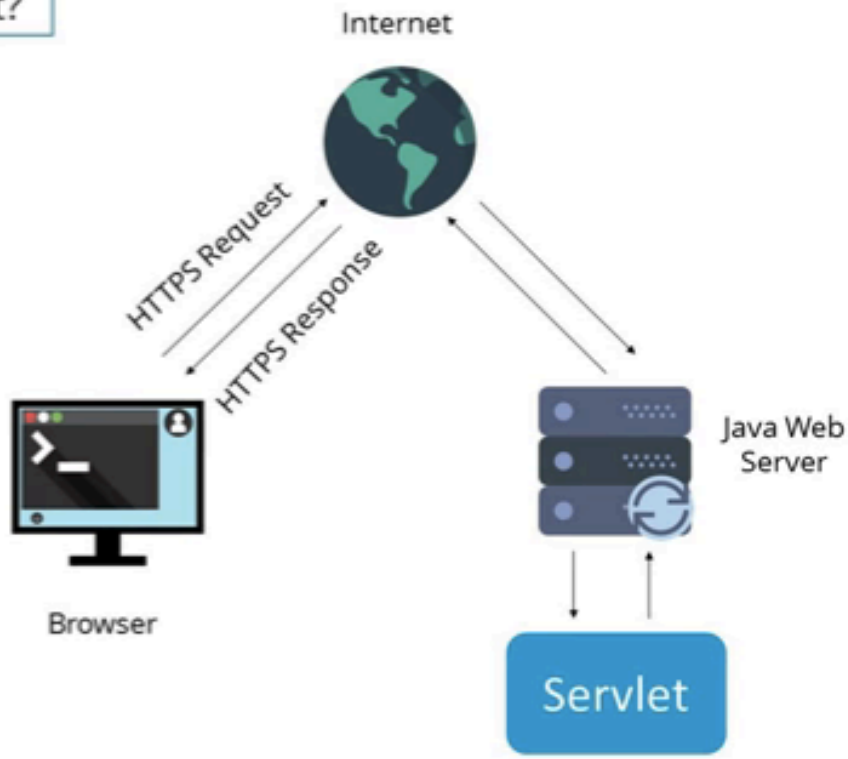


Java Interview Questions

edureka!

Q21. What is a Servlet?

- Java Servlet is server side technologies to extend the capability of web servers by providing support for dynamic response and data persistence.
- The javax.servlet and javax.servlet.http packages provide interfaces and classes for writing our own servlets.
- All servlets must implement the javax.servlet.Servlet interface, which defines servlet lifecycle methods.
- As most web applications are accessed using HTTP protocol ,we mostly extend HttpServlet class. Servlet API hierarchy.



edureka!

Q22. What is difference between Get and Post method?

Get	Post
Limited amount of data can be sent because data is sent in header.	Large amount of data can be sent because data is sent in body.
Not Secured because data is exposed in URL bar.	Secured because data is not exposed in URL bar.
Can be bookmarked	Cannot be bookmarked
Idempotent	Non-Idempotent
It is more efficient and used than Post	It is less efficient and used

Q23. What are different methods of session management in servlets?

- Session is a conversational state between client and server and it can consists of multiple request and response between client and server.
- Since HTTP and Web Server both are stateless, the only way to maintain a session is when some unique information about the session (session id) is passed between server and client in every request and response.

edureka!

Q24. What is the .Difference between ServletContext vs ServletConfig?

ServletConfig	ServletContext
Servlet config object represent single servlet	It represent whole web application running on particular JVM and common for all the servlet
Its like local parameter associated with particular servlet	Its like global parameter associated with whole application
It's a name value pair defined inside the servlet section of web.xml file so it has servlet wide scope	ServletContext has application wide scope so define outside of servlet tag in web.xml file.
getServletConfig() method is used to get the config object	getServletContext() method is used to get the context object.
E.g. Shopping cart of a user is a specific to particular user so here we can use servlet config	To get the MIME type of a file or application session related information is stored using servlet context object.

edureka!

Q25. What is the life-cycle of a servlet?

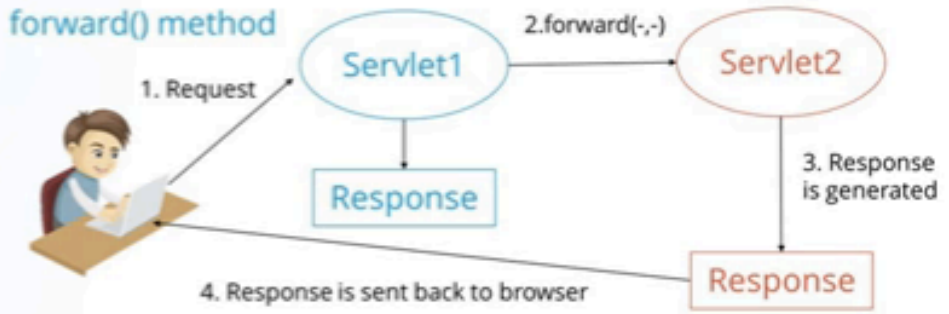
There are 5 stages in the lifecycle of a servlet:

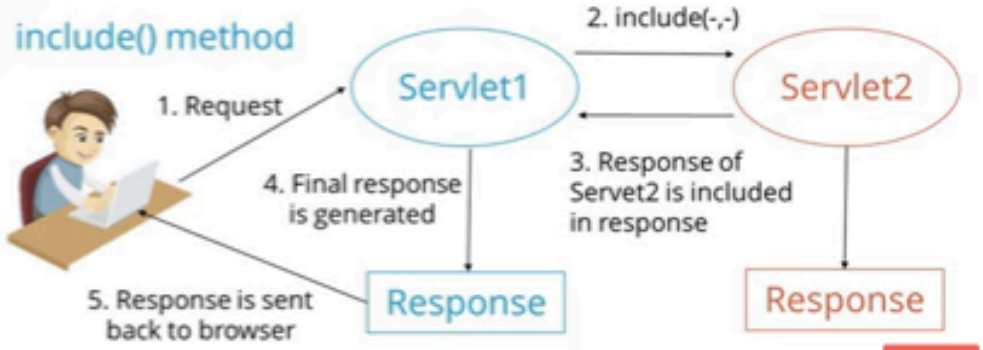
- Servlet is loaded
- servlet is instantiated
- servlet is initialized
- service the request
- servlet is destroyed

edureka!

Q26. What is Request Dispatcher?

- RequestDispatcher interface is used to forward the request to another resource that can be HTML, JSP or another servlet in same application.
- We can also use this to include the content of another resource to the response.
- There are two methods defined in this interface:
  1. void forward()
  2. void include()





edureka!

Q27. How does Cookies work in Servlets?

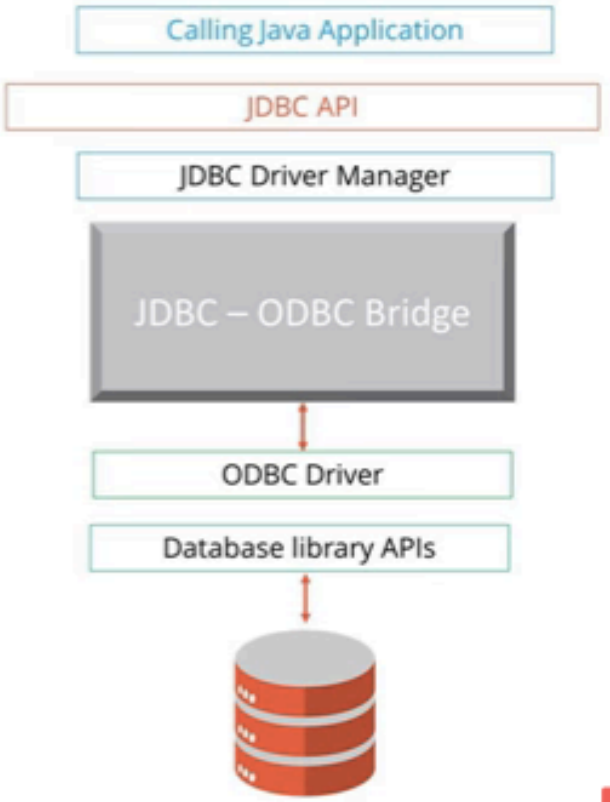
- Cookies are text data sent by server to the client and it gets saved at the client local machine.
- Servlet API provides cookies support through javax.servlet.http.Cookie class that implements Serializable and Cloneable interfaces.
- HttpServletRequest getCookies() method is provided to get the array of Cookies from request, since there is no point of adding Cookie to request, there are no methods to set or add cookie to request.
- Similarly HttpServletResponse addCookie(Cookie c) method is provided to attach cookie in response header, there are no getter methods for cookie.

edureka!

Q28. What is JDBC Driver?

JDBC Driver is a software component that enables java application to interact with the database. There are 4 types of JDBC drivers:

- JDBC-ODBC bridge driver
- Native-API driver (partially java driver)
- Network Protocol driver (fully java driver)
- Thin driver (fully java driver)



edureka!

Q29. What are the steps to connect to the database in java?

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;

public class JdbcConnection {

    public static void main(String a[]){

        try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
            Connection con = DriverManager.getConnection( url: "jdbc:oracle:thin:@<hostname>:<port num>:<DB name>"
                        , user: "user", password: "password");
            Statement stmt = con.createStatement();
            System.out.println("Created DB Connection....");
        } catch (ClassNotFoundException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } catch (SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
}
```

There are 5 steps we need to follow to connect to a database in Java

- Registering the driver class
- Creating connection
- Creating statement
- Executing queries
- Closing connection

Java Interview Questions

edureka!

Q30. What are the different JDBC API components?

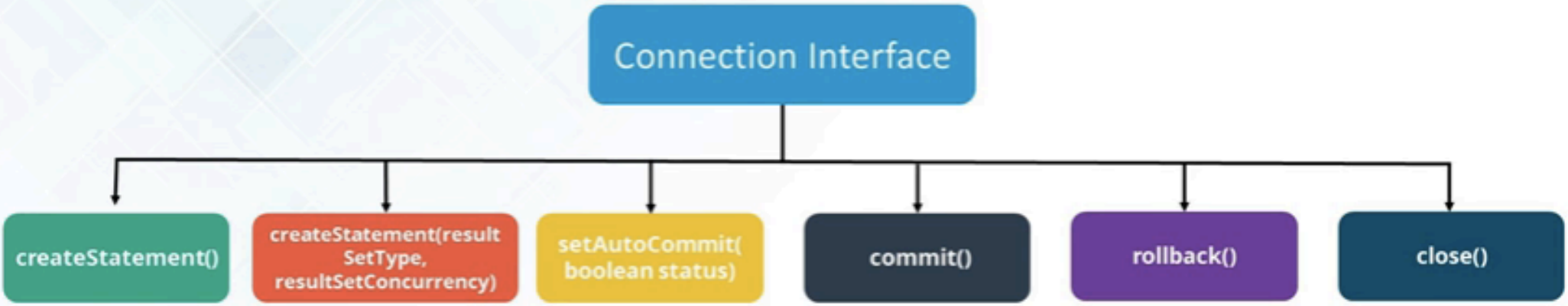
Interfaces

Classes

edureka!

Q31. What is a JDBC Connection interface?

The Connection interface maintains a session with the database. It can be used for transaction management.



edureka!

Q32. What is the difference between execute, executeQuery, executeUpdate?

executeQuery()	executeUpdate()	execute()
This method is used to execute the SQL statements which retrieve some data from the database.	This method is used to execute the SQL statements which update or modify the database.	This method can be used for any kind of SQL statements.
This method returns a ResultSet object which contains the results returned by the query.	This method returns an int value which represents the number of rows affected by the query. This value will be the 0 for the statements which return nothing.	This method returns a boolean value. TRUE indicates that query returned a ResultSet object and FALSE indicates that query returned an int value or returned nothing.
This method is used to execute only select queries.	This method is used to execute only non-select queries.	This method can be used for both select and non-select queries.
Ex : SELECT	Ex : DML → INSERT, UPDATE and DELETE DDL → CREATE, ALTER	This method can be used for any type of SQL statements.

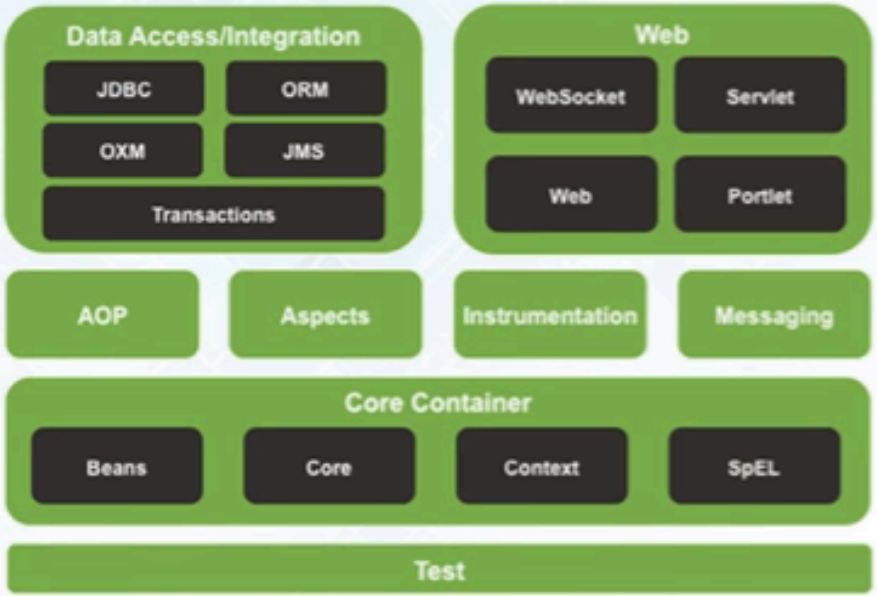
edureka!

Q33. What is Spring?

- It is an application framework and inversion of control container for the Java platform.
- The framework's core features can be used by any Java application, but there are extensions for building web applications on top of the Java EE platform."
- Spring is essentially a lightweight, integrated framework that can be used for developing enterprise applications in java.

edureka!

Q34. What are the different modules of the Spring framework.



Some of the important Spring Framework modules are:

- Spring Context – for dependency injection.
- Spring AOP – for aspect oriented programming.
- Spring DAO – for database operations using DAO pattern
- Spring JDBC – for JDBC and DataSource support.
- Spring ORM – for ORM tools support such as Hibernate
- Spring Web Module – for creating web applications.
- Spring MVC – Model-View-Controller implementation for creating web applications, web services etc.

Java Interview Questions

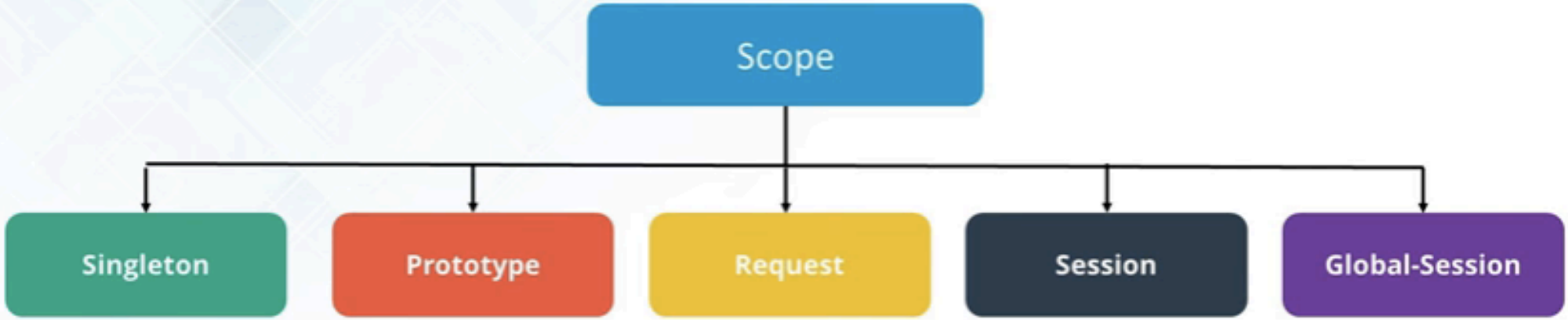
edureka!

Q35. List some of the important annotations in annotation-based Spring configuration

edureka!

Q36. What is a Bean in Spring and Explain the different scopes of bean in Spring?

- .A bean is an object that is instantiated, assembled, and managed by a Spring IoC container.
- They are managed by the Spring IoC container.
- There are 5 scopes of beans in Spring :



Q37. How is a bean added to a Spring application?

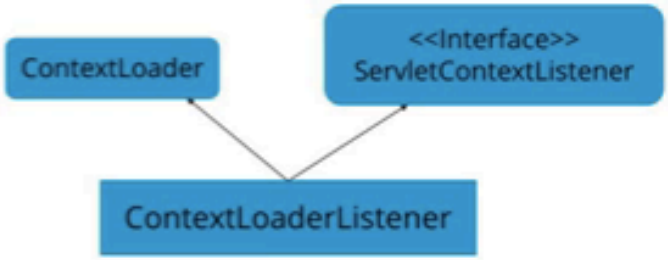
```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN">

<beans>

<bean id="foo"/>

<bean id="bar"/>

</beans>
```

Q38. Explain the role of DispatcherServlet and ContextLoaderListener.

DispatcherServlet

DispatcherServlet is the front controller in the Spring MVC application as it loads the spring bean configuration file and initializes all the beans that have been configured.

ContextLoaderListener



ContextLoaderListener, the listener to start up and shut down the WebApplicationContext in Spring root.

edureka!

Q39. What is the difference between constructor injection and setter injection?

Constructor Injection	Setter Injection
No Partial Injection	Partial Injection
Desn't override the setter property	Overrides the constructor property if both are defined.
Creates new instance if any modification occurs	Doesn't create new instance if you change the property value
Better for too many properties	Better for few properties.

edureka!

Q40. What is autowiring in spring? What are the autowiring modes?

Autowiring enables the programmer to inject the bean automatically. We don't need to write explicit injection logic. Let's see the code to inject bean using dependency injection.

```
1. <bean id="emp" class="com.javatpoint.Employee" autowire="byName" />
```

Mode	Description
no	This is the default mode, it means autowiring is not enabled.
byName	injects the bean based on the property name. It uses setter method.
byType	injects the bean based on the property type. It uses setter method.
constructor	It injects the bean using constructor

edureka!

Q41. How to handle exceptions in Spring MVC Framework?

```
@Controller
public class ExceptionHandlingController {

  // Convert a predefined exception to an HTTP Status code
  @ResponseStatus(value=HttpStatus.CONFLICT,
                  reason="Data integrity violation")  // 409
  @ExceptionHandler(DataIntegrityViolationException.class)
  public void conflict()

  // Specify name of a specific view that will be used to display the error:
  @ExceptionHandler({SQLException.class,DataAccessException.class})
  public String databaseError() {

    return "databaseError";
  }

  @ExceptionHandler(Exception.class)
  public ModelAndView handleError(HttpServletRequest req, Exception ex) {
    logger.error("Request: " + req.getRequestURL() + " raised " + ex);
    ModelAndView mav = new ModelAndView();
    mav.addObject("exception", ex);
    mav.addObject("url", req.getRequestURL());
    mav.setViewName("error");
    return mav;
  }
```

Spring MVC Framework provides following ways to help us achieving robust exception handling.:

- Controller Based – We can define exception handler methods in our controller classes.
- Global Exception Handler – Exception Handling is a cross-cutting concern and Spring provides .
- HandlerExceptionResolver –Any Spring bean declared in the DispatcherServlet's application context that implements HandlerExceptionResolver will be used to intercept and process any exception raised in the MVC system and not handled by a Controller.

edureka!

Q42. What are some of the important Spring annotations used?

edureka!

Q43. How to integrate Spring and Hibernate Frameworks?

- We can use Spring ORM module to integrate Spring and Hibernate frameworks
- Spring ORM also provides support for using Spring declarative transaction management,

Q44. What is Hibernate Framework?

HIBERNATE

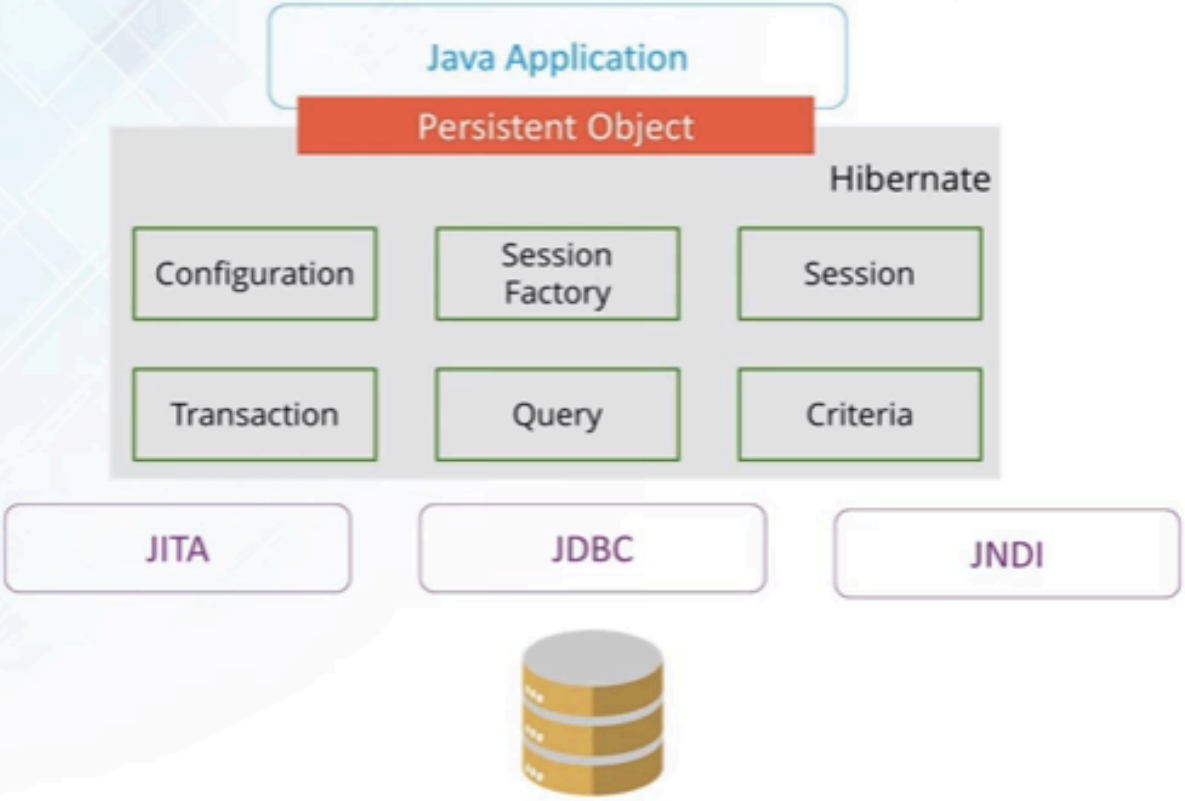
- Hibernate is java based ORM tool that provides framework for mapping application domain objects to the relational database tables and vice versa.
- Hibernate provides reference implementation of Java Persistence API, that makes it a great choice as ORM tool with benefits of loose coupling.
- Hibernate configurations are flexible and can be done from XML configuration file as well as programmatically.

Q45. What are the important benefits of using Hibernate Framework?

1. Hibernate eliminates all the boiler-plate code.
2. Hibernate framework provides support for XML as well as JPA annotations.
3. Hibernate provides a powerful query language (HQL) that is similar to SQL.
4. Hibernate is an open source project from Red Hat Community and used worldwide.
5. For database vendor specific feature, hibernate is suitable because we can also execute native sql queries.
6. 7. Hibernate cache helps us in getting better performance.

edureka!

Q46. Explain hibernate architecture?



edureka!

Q47. What is the difference between get and load method?

get()	load()
Returns **null** if object is not found.	Throws **ObjectNotFoundException** if object is not found.
get() method always **hit the database**.	load() method **doesn't hit** the database.
It returns real object **not proxy**.	It returns **proxy object.**
It should be used if **you are not sure** about the existence of instance.	It should be used if **you are sure** that instance exists.

Q48. What are the advantages of Hibernate over JDBC?

1. Hibernate removes boiler-plate code that comes with JDBC API
2. Hibernate supports inheritance, associations and collections which are not present with JDBC API.
3. Hibernate implicitly provides transaction management.
4. JDBC API throws SQLException that is a checked exception, so we need to write a lot of try-catch block code.
5. Hibernate Query Language (HQL) is more object oriented and close to java programming language. For JDBC, we need to write native sql queries.
6. Hibernate supports caching that is better for performance, JDBC queries are not cached hence performance is low.