Morning Activity Servlet , JSP and JDBC

Q.1 What is a servlet?

Ans:- Servlets are the Java programs that run on the Java-enabled web server or application server. They are used to handle the request obtained from the webserver, process the request, produce the response, then send a response back to the webserver.

Properties of Servlets are as follows:

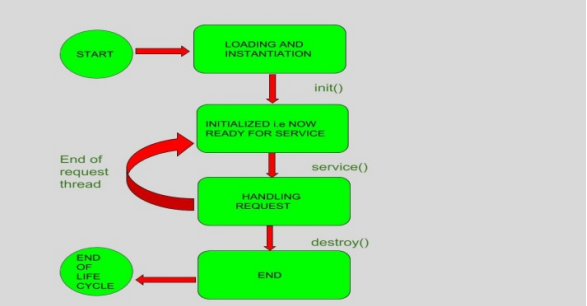
1. Servlets work on the server-side.
2. Servlets are capable of handling complex requests obtained from the webserver.

Q.2 Explain Servlets Lifecycle? OR

Q.2 How does the servlet container manage the servlet life cycle, when and what methods are called?

Ans:- The entire life cycle of a Servlet is managed by the **Servlet container** which uses the **javax.servlet.Servlet** interface

Stages of the Servlet Life Cycle: The Servlet life cycle mainly goes through four stages,



1. Loading a Servlet.
2. Initializing the Servlet.
3. Request handling.
4. Destroying the Servlet.

Loading a Servlet:

- The Servlet container performs two operations in this stage :

1. **Loading :** Loads the Servlet class.
2. **Instantiation :** Creates an instance of the Servlet. To create a new instance of the Servlet, the container uses the no-argument constructor.

Initializing a Servlet:

- The container initializes the Servlet object by invoking the **Servlet.init(ServletConfig)** method
- The Servlet container invokes the **Servlet.init(ServletConfig)** method only once.
- if the Servlet fails to initialize, then it informs the Servlet container by throwing the **ServletException** or **UnavailableException**.

Handling request:

- It creates the **ServletRequest** and **ServletResponse** objects. In this case, if this is a HTTP request, then the Web container creates **HttpServletRequest** and **HttpServletResponse** objects which are subtypes of the **ServletRequest** and **ServletResponse** objects respectively.
- After creating the request and response objects it invokes the Servlet.service(ServletRequest, ServletResponse) method by passing the request and response objects.
- The **service()** method while processing the request may throw the **ServletException** or **UnavailableException** or **IOException**.

Destroying a Servlet:

- When a Servlet container decides to destroy the Servlet, it performs the following operations,
- It allows all the threads currently running in the service method of the Servlet instance to complete their jobs and get released.
- After currently running threads have completed their jobs, the Servlet container calls the **destroy()** method on the Servlet instance.
- After the **destroy()** method is executed, the Servlet container releases all the references of this Servlet instance so that it becomes eligible for garbage collection.

Q.3 What is a servlet container?

Ans :-

Container is part of web server which interacts with the servlet for handling the dynamic web pages from client. **Servlet container**, also known as **Servlet engine** is an integrated set of objects that provide a run time environment for Java Servlet components.

Q.4 What is ServletContext Interface?

Ans:-

- Defines a set of methods that a servlet uses to communicate with its servlet container, for example, dispatch requests, or write to a log file.
- There is one context per "web application" per Java Virtual Machine.
- ServletContext is the object created by Servlet Container to share initial parameters or configuration information to the whole application.

Q. How does the servlet container manage the servlet life cycle, when and what methods are called?

Ans : see question 2

Q.5 What actions do you need to do when creating servlets?

Ans :-

Step 1: Create a Directory Structure under Tomcat-- Create a directory called myApp under the webapps directory. The directory name is important because this also appears in the URL to your servlet

Step 2: Write the Servlet Source Code –

Step 3: Compile Your Source Code-- For your servlet source code to compile, you need to include in your CLASSPATH environment variable the path to the servlet.jar file.

Step 4: Run Tomcat--If it is not already running, you need to start Tomcat.

Step 5: Call Your Servlet from a Web Browser--You are ready to call your servlet from a web browser. By default, Tomcat runs on port 8080 in myApp virtual directory under the servlet subdirectory.

Q.6 How to call another servlet from one servlet?

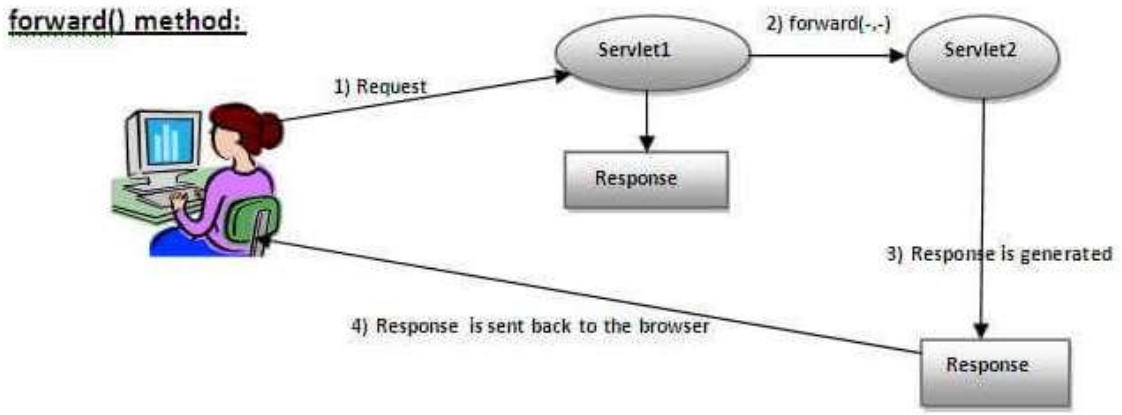
Q.7 What is Request Dispatcher?

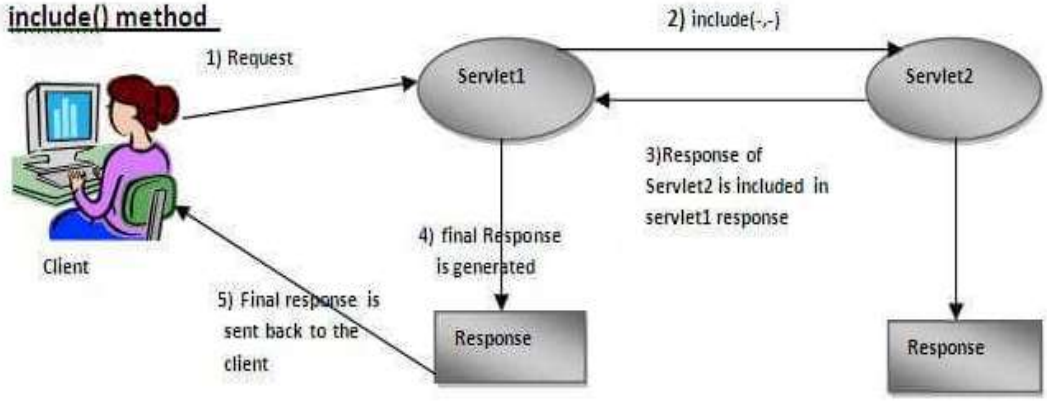
Ans :-

The RequestDispatcher interface provides the facility of dispatching the request to another resource it may be html, servlet or jsp. This interface can also be used to include the content of another resource also. It is one of the way of servlet collaboration.

There are two methods defined in the RequestDispatcher interface.

public void forward(ServletRequest request,ServletResponse response)throws ServletException,java.io.IOException:
Forwards a request from a servlet to another resource (servlet, JSP file, or HTML file) on the server.
public void include(ServletRequest request,ServletResponse response)throws ServletException,java.io.IOException:
Includes the content of a resource (servlet, JSP page, or HTML file) in the response.





Q.8 What are the differences between forward() and sendRedirect() methods?
Ans :-

forward()	sendRedirect()
forward() is method of RequestDispatcher interface	sendRedirect() is the method of HttpServletResponse In forward().
In forward() redirect happens at server end and not visible to client.	In sendRedirect() ,redirection happens at client end and it's visible to client.
It is faster than the redirect.	It is slower than a forward, since it requires two browser requests(one for actual request and another for redirected request).
In case of forwarding() original URL remains unaffected.	In the case of sendRedirect() browser knows that it's making a new request, so the original URL changes.
Transfer the request to the same server.	Transfer the request different server.
When forward is called on RequestDispather object we pass request and response object so our old request object is present on a new resource which is going to process our request	In the case of SendRedirect call old request and response, the object is lost because it's treated as a new request by the browser
Syntax: forward(ServletRequest request, ServletResponse response)	Syntax: void sendRedirect(String url)

Q. 9 What is the JDBC?

- JDBC is a Java API to connect and execute the query with the database.
- It is a part of JavaSE (Java Standard Edition).
- JDBC API uses JDBC drivers to connect with the database.
- By the help of JDBC API, we can save, update, delete and fetch data from the database.
- The **java.sql** package contains classes and interfaces for JDBC API.
- A list of popular *interfaces* of JDBC API are given below:
    1) Driver interface
    2) Connection interface
    3) Statement interface
    4) PreparedStatement interface
    5) CallableStatement interface
    6) ResultSet interface
    7) ResultSetMetaData interface
    8) DatabaseMetaData interface
    9) RowSet interface

- A list of popular *classes* of JDBC API are given below:
  1) DriverManager class
  2) Blob class
  3) Clob class
  4) Types class

Note :- API (Application programming interface) is a document that contains a description of all the features of a product or software. It represents classes and interfaces that software programs can follow to communicate with each other. An API can be created for applications, libraries, operating systems, etc.

Q.10 Explain the JDBC Architecture?

Ans ;-

If you want to develop a Java application that communicates with a database, you should use JDBC API. A driver is the implementation of the said API; various vendors provide various drivers, you need to use a suitable driver with respect to the database you need to communicate with. The driver manager loads the driver and manages the driver.

Following are the components of JDBC:

- **JDBC DriverManager:** The DriverManager class of the java.sql package manages different types of JDBC drivers. This class loads the driver classes. In addition to this whenever a new connection establishes it chooses and loads the suitable driver from the previously loaded ones.

Note: From JDBC 4.0 the drivers in the CLASSPATH will be loaded automatically.

- **JDBC API:** It is a Java abstraction which enables applications to communicate with relational databases. It provides two main packages namely, java.sql and javax.sql. It provides classes and methods to connect with a database, create statements (quires), execute statements and handle the results.

Q. 11 What are the new features available in JDBC 4.0?

Ans:-

The important features of JDBC API 4.0 are given below:

1) **Automatic Loading of Driver class** You don't need to write Class.forName() now because it is loaded bydefault since jdbc4.
2) **Subclasses of SQLException** Jdbc 4 provides new subclasses of SQLException class for better readability and handling.
3) **New methods** There are many new methods introduced in Connection, PreparedStatement, CallableStatement, ResultSet etc.
4) **Improved DataSource** Now data source implementation is improved.
5) **Event Handling support in Statement for Connection Pooling** Now Connection Pooling can listen statement error and statement closing events.

Q.12 What are the advantages of using PreparedStatement in Java?

Ans ;-

- By avoiding multiple compilation and execution of statements, prepared statements perform faster.
- Using prepared statements, we can insert values to advanced datatypes such as BLOB, CLOB, OBJECT easily with the help of the setter methods provided by the PreparedStatement interface.
- By providing setter method to set values prepared statement avoids the use of quotes and other special characters with in the query, and thereby it escapes the SQL injection attacks

Q.13 What is ResultSet?

- The object of ResultSet maintains a cursor pointing to a row of a table. Initially, cursor points to before the first row.
- By default ResultSet object can be moved forward only and it is not updatable.

Commonly used methods of ResultSet interface

1) public boolean next():	is used to move the cursor to the one row next from the current position.
2) public boolean previous():	is used to move the cursor to the one row previous from the current position.
3) public boolean first():	is used to move the cursor to the first row in result set object.
4) public boolean last():	is used to move the cursor to the last row in result set object.
5) public boolean absolute(int row):	is used to move the cursor to the specified row number in the ResultSet object.
6) public boolean relative(int row):	is used to move the cursor to the relative row number in the ResultSet object, it may be positive or negative.
7) public int getInt(int columnIndex):	is used to return the data of specified column index of the current row as int.
8) public int getInt(String columnName):	is used to return the data of specified column name of the current row as int.
9) public String getString(int columnIndex):	is used to return the data of specified column index of the current row as String.
10) public String getString(String columnName):	is used to return the data of specified column name of the current row as String.

Q.14 What are types of ResultSet?

Ans :-

There are two types of result sets namely, forward only and, bidirectional.

Forward only ResultSet: The ResultSet object whose cursor moves only in one direction is known as forward only ResultSet. By default, JDBC result sets are forward-only result sets.

You can move the cursor of the forward only **ResultSets** using the **next()** method of the ResultSet interface. It moves the pointer to the next row from the current position. This method returns a boolean value. If there are no rows next to its current position it returns false, else it returns true.

Therefore, using this method in the while loop you can iterate the contents of the ResultSet object.

Bidirectional ResultSet: A bi-directional ResultSet object is the one whose cursor moves in both forward and backward directions.

The createStatement() method of the Connection interface has a variant which accepts two integer values representing the result set type and the concurrency type.

Q.15 What is the difference between executing, executeQuery, executeUpdate in JDBC?

Ans:-

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.

Q.16 What is JSP? What are the different Life-Cycle methods?

Ans:-

JSP is an abbreviation for **Java Servlet Page**. It is a Server-Side Programming Language used to create dynamic web-pages in the form of HTML. The JSP page is implicitly converted into a servlet and it enables some additional features such as Expression Language, Custom Tags, and many more.

The different Life-Cycle Methods are as follows:

- **jspInit() —** Container calls **jspInit()** method to initialize servlet instance. It is called **once** for the servlet instance and **preceded** every other method.

- **_jspService():** Container calls **_jspService()** method for each **request** and passes it on to the **objects.**
- **jspDestroy():** Container calls the **jspDestroy()** just before destruction of the instance.

Q.17 Mention some important JSP Action Tags.

Ans;-

There are many JSP action tags or elements. Each JSP action tag is used to perform some specific tasks.
The action tags are used to control the flow between pages and to use Java Bean. The Jsp action tags are given below

JSP Action Tags	Description
jsp:forward	forwards the request and response to another resource.
jsp:include	includes another resource.
jsp:useBean	creates or locates bean object.
jsp:setProperty	sets the value of property in bean object.
jsp:getProperty	prints the value of property of the bean.
jsp:plugin	embeds another components such as applet.
jsp:param	sets the parameter value. It is used in forward and include mostly.
jsp:fallback	can be used to print the message if plugin is working. It is used in jsp:plugin.

Q.18 Why are the request.getRequestDispatcher() and context.getRequestDispatcher() used?

Ans:-

The **RequestDispatcher()** and the **context.getRequestDispatcher()** are used for the following purposes.

request.getRequestDispatcher() is used to create request. We need to give the relative path of the resource.

context.getRequestDispatcher() is used to create context. We need to give the absolute path of the resource.

Q.19 What are the various Implicit Objects used in JSP .

Ans:-The **Web Container** creates certain objects that include the information related to a particular *Request, Application* or a *Page*. These Objects are called as **Implicit**

Objects. The Implicit Objects in the JSP are as follows:

Request

1. Response
2. Application
3. Exception
4. Config
5. Page
6. Session
7. PageContext
8. Out

Q.How can we stop errors on Display in a JSP Page?

Ans:-

You first set "errorPage" attribute of PAGE directive to the name of the error page (ie errorPage="error.jsp")in your jsp page .

Then in the error.jsp page set "isErrorpage=TRUE".

When an error occur in your jsp page, then the control will be automatically forward to error page.