

Name: Pavankumari

Roll no: 320227720007

1st semester

Sub: OOPS with Java programming

1.)

Define Servlet and explain the servlet life cycle.

a) SERVLET: A servlet is a Java programming language class that is used to

extend the capabilities of servers that host applications accessed by means of a

request-response programming model. Although servlets can respond to any

type of request, they are commonly used to extend the applications hosted by

web servers. For such applications, Java Servlet technology defines HTTP specific servlet classes.

SERVLET LIFE CYCLE:

The life cycle of a servlet is controlled by the container in which the servlet has

been deployed. When a request is mapped to a servlet,

the container performs
the following steps.

Loading a Servlet.

Initializing the Servlet.

Request handling.

Destorying the Servlet.

a.) Loading a Servlet: The frst stage of the Servlet lifecycle involves loading and initializing the servlet byc the servlet container. The Web container or Servlet container can load the servlet at either of the following two stages :

Initializing the context, on configuring the servlet with a zero or positive integer value.

If the servlet is not preceding stage, it mayc delaye the loading process until the web container determines that this servlet is needed to service a request.

b.) Initialzing the Servlet: After the servlet is instantated

successfully, the servlet container initializes the instantiated servlet object. The container initializes the servlet object by invoking the `Servlet.init(ServletConfig)` method which accepts `ServletConfig` object reference as parameter.

The servlet container invokes the `Servlet.init(ServletConfig)` method only once, immediately after the `Servlet.init(ServletConfig)` object is instantiated

successfully. This method is used to initialize the resources, such as JDBC datasource.

c.) Handling Request: After initialization, the servlet instance is ready to serve the client requests. The servlet container performs the following operations when the servlet instance is located to service a 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(HttpServletRequest, HttpServletResponse)` method by passing the request and response objects.

d.) 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. There are three life cycle methods of a Servlet :

init()

service()

destroy()

init(): The `Servlet.init()` method is called by the Servlet container to

indicate that this Servlet instance is instantiated successfully and is about to put into service.

service(): The `service()` method of the Servlet is invoked to inform the Servlet about the client requests.

This method uses `ServletRequest` object to collect the data requested by the client.

This method uses `ServletResponse` object to generate the output content.

destroy(): The `destroy()` method runs only once during the lifetime of a Servlet and signals the end of the Servlet

instance

2.) Explain about JDBC DRIVERS?

a) JDBC Driver is a software component that enables java application to interact

with the database. There are 4 types of JDBC drivers:

1. JDBC-ODBC bridge driver
2. Native-API driver (partially java driver)
3. Network Protocol driver (fully java driver)
4. Thin driver (fully java driver)

1.) JDBC-ODBC bridge driver:

The JDBC-ODBC bridge driver uses ODBC driver to connect to the database. The JDBC-driver converts JDBC method calls into the ODBC function calls. This is now discouraged driver.

Advantages:

- o easy to use.
- o can be easily connected to any database.

Disadvantages:

- o Performance degraded because JDBC method call is

converted into the ODBC function calls.

o The ODBC driver needs to be installed on the client machine.

2.) Native-API driver (partially java driver): The Native API driver uses the client-side libraries of the database. The driver converts JDBC method calls into native calls of the database API. It is not written entirely in java.

Advantage:

o performance upgraded than JDBC-ODBC bridge driver.

Disadvantage:

o The Native driver needs to be installed on the each client machine.

o The Vendor client library needs to be installed on client machine.

3.) Network Protocol driver (fully java driver): The Network Protocol driver

uses middleware (application server) that converts JDBC calls directly or

indirectly into the vendor-specific database protocol. It is fully written in java.

Advantage:

- o No client side library is required because of application server that can perform many tasks like auditing, load balancing, logging etc.

Disadvantages:

- o Network support is required on client machine.
- o Requires database-specific coding to be done in the middle tier.
- o Maintenance of Network Protocol driver becomes costly because it requires database-specific coding to be done in the middle tier.

4.) Thin driver (fully written java driver): The thin driver converts JDBC calls directly into the vendor-specific database protocol. That is why it is known as thin driver. It is fully written in Java language

Advantage:

- o Better performance than all other drivers.
- o No software is required at client side or server side.

Disadvantage: Drivers depend on the database.

3.) Explain Java FX-UI controls?

a) The graphical user interface of every desktop application mainly considers UI elements, layouts and behaviour.

The UI elements are the one which are actually shown to the user for

interaction or information exchange. Layout defines the organization of the UI

elements on the screen. Behaviour is the reaction of the UI element when

some event is occurred on it.

However, the package javafx.scene.control provides all the necessary classes

for the UI components like Button, Label, etc. Every class

represents a specific UI control and defines some methods for their styling.

*Label- Label is a component that is used to define a simple text on the screen.
Typically, a label is placed with the node, it describes.

*Button- Button is a component that controls the function of the application.
Button class is used to create a labelled button.

*RadioButton- The Radio Button is used to provide various options to the user.
The user can only choose one option among all. A radio button is either selected or deselected.

*checkbox- Check Box is used to get the kind of information from the user which contains various choices. User marked the checkbox either on (true) or off(false).

*TextField- Text Field is basically used to get the input from the user in the form of text. `javafx.scene.control.TextField` represents

TextField.

- * PasswordField- PasswordField is used to get the user's password. Whatever is typed in the passwordfeld is not shown on the screen to anyone.
- *Hyperlink- HyperLink are used to refer any of the webpage through your appication. It is represented byc the class javafx.scene.control.HyperLink
- *Slider- Slider is used to provide a pane of options to the user in a graphical form where the user needs to move a slider over the range of values to select one of them.