Dt : 21/12/2023

faq:

define Annotattion?

=>The tag-based-information which is added to the programming components like

Variable or method or Class or Interface is known as Annotation.

=>we use '@' symbol to represent annotations.

=>These annotations will give information to Compiler at compilation stage or will give information to execution-control while execution process.

CoreJava Annotations:

@Override - will give information to compiler to check the method is Overriding method or not

@SuppressWarnings - will give information to compiler to close the raised warnings

@FunctionalInterface - will give information to check the interface is Functional Interface or not

AdvJava Annotations:

@WebServlet - will hold Servlet-url-pattern to identify Servlet-program for execution.

@WebFilter - will hold Servlet-url-pattern to identify Filter-program for execution.

@WebListener - is used to identify Listener-program for execution.

==========

**imp*

Servlet Life-Cycle:

=>Servlet Life-Cycle demonstrates different states of Servlet-program from Starting of program to ending of program.

=>The following are the stages of Servlet Life-Cycle:

1. *Loading process*
2. *Instantiation process*
3. *Initialization process*
4. *Request Handling Process*
5. *Destroying Process*

1.Loading process:

=>The process of identifying the servlet-program using url-pattern and loading for execution is known as Loading Process.

2.Instantiation process:

=>When Servlet-program loaded,it is instantiated automatically known as Instantiation process.(Object creation process)

=>After Instantiation process,the execution-controls will identify the following

life-cycle methods:

GenericServlet

init()

service()

destroy()

HttpServlet

init()

service()/doPost()/doGet()

destroy()

Filter

init()

doFilter()

destroy()

3.Initialization process:

=>The process of making the programming components ready for service()-method

is known as Initialization process.

=>we use init()-method for initialization process.

Note:

=>init()-method is executed only once for Servlet-program,which means

Initialization process is performed only once.

4.Request Handling Process:

=>The process of accepting request from user and providing the response is known as Request Handling process.

=>we use service()/doPost()/doGet() method to perform Request handling process.

Note:

=>This service()-method is executed for all multiple requests generated from multiple users.

5.Destroying Process:

=>The process of closing the resources which are opened part of Request Handling process is known as destroying process.

=>we use destroy()-method to perform destroying process.

Note:

=>The destroy()-method is executed after service()-method execution completed.

==========

Ex-application:

input.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
 <form action="dis" method="post">
 UserName:<input type="text" name="uname"><br>
 <input type="submit" value="Display">
 </form>
</body>
</html>
```

DisplayServlet.java

```
package test;

import java.io.*;

import javax.servlet.*;

import javax.servlet.http.*;

import javax.servlet.annotation.*;

@SuppressWarnings("serial")

@WebServlet("/dis")

public class DisplayServlet extends HttpServlet

{

	public DisplayServlet()

	{

		System.out.println("Instantiation process...");

	}
```

```
    @Override
 public void init()throws ServletException
 {
     System.out.println("Initialization process...");
 }
    @Override
    protected void doPost(HttpServletRequest req,HttpServletResponse
res)throws
    ServletException,IOException{
        PrintWriter pw = res.getWriter();
        res.setContentType("text/html");
        pw.println("Welcome : "+req.getParameter("uname"));
        System.out.println("Request handling process...");
    }
    @Override
    public void destroy()
    {
        System.out.println("Destroying process....");
    }
}
```

web.xml

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

```
  <welcome-file-list>
    <welcome-file>input.html</welcome-file>
  </welcome-file-list>
</web-app>

=============================================================================
```