SERVLETS PART-5

The object creation order by web container:

1) ServletContext

2) User defined Servlet

3) ServletConfig

4) ServletRequest & ServletResponse

ServletContext is created by web container whenever web application is deployed on server.

User defined servlet object is created by web container whenever first request comes to a servlet.

ServletConfig is created by web container whenever init() method called by web container.

ServletRequest & ServletResponse are created by web container whenever service() method is called.

Request Parameters:

Request parameters are used to process the request & construct the response. Request parameters are retrieved in a Servlet by using getParameter() method of javax.servlet.ServletRequest interface.

The following methods of javax.servlet.ServletRequest are used to get the request parameters:

public abstract String getParameter(String);

=>It is used to get the request parameter value

public abstract String[] getParameterValues(String);

=>It is used to get the request parameter values

public abstract Enumeration<String> getParameterNames();

=>It is used to get the request parameter names

public abstract Map<String, String[]> getParameterMap();

=>It is used to get the request parameter names & values

ServletRequest & ServletResponse Example:

login.html

```
<html>
<body bgcolor=green text=yellow>
<center>
<h1><u>Login Form</u></h1>
<form method=POST action=login>
Username <input type=text name=uname><br>
Password <input type=password name=pword><br><br>
<input type=Submit><input type=Reset>
</form>
</center>
</body>
</html>
```

registration.html

```
<html>
```

```
<body bgcolor=green text=yellow>
<center>
<h1><u>Registration Form</u></h1>
<form method=POST action=reg>
First Name <input type=text name=fname><br>
Last Name <input type=text name=lname><br>
Username <input type=text name=uname><br>
Password <input type=password name=pword><br><br>
<input type=Submit><input type=Reset>
</form>
</center>
</body>
</html>
```

There are seven http methods:

1) GET
2) POST
3) PUT
4) DELETE
5) HEAD
6) TRACE
7) OPTIONS

GET Vs. POST:

GET	POST
1) It includes the request parameters in a request header in a packet.	1) It includes the request parameters in a request body in a packet.
2) In this method request parameters are displayed in address bar.	2) In this method request parameters are not displayed in address bar.
3) Get requests can be cached.	3) Post requests are never cached.
4) Get requests can be bookmarked.	4) Post request cannot be bookmarked.
5) Get requests are stored in browser history.	5) Post requests are not stored in browser history.
6) Here size of the data is limited.(maximum 2048 characters)	6) Here size of the data is not limited.
7) It is not secure.	7) It is secure.
8) It supports text only.	8) It supports all types of data.

There are three ways to write servlet program:

1) By implementing jakarta.servlet.Servlet interface.
2) By extending jakarta.servlet.GenericServlet class.
3) By extending jakarta.servlet.http.HttpServlet class.

GenericServlet Vs. HttpServlet:

GenericServlet	HttpServlet
1) It is a protocol independent.	1) It is a http protocol dependent.
2) It supports common services. Examples: Forwarding and Including.	2) It supports http specific services. Examples: Redirecting and Session Tracking.
3) By using GenericServlet we can access common services only.	3) By using HttpServlet we can access both common services and http specific services because it is a sub class of GenericServlet class.

javax.servlet.http.HttpServlet

Methods:

1) protected void doGet(javax.servlet.http.HttpServletRequest,
   javax.servlet.http.HttpServletResponse)
   throws javax.servlet.ServletException, java.io.IOException;
2) protected void doPost(javax.servlet.http.HttpServletRequest,
   javax.servlet.http.HttpServletResponse)
   throws javax.servlet.ServletException, java.io.IOException;
3) protected void service(javax.servlet.http.HttpServletRequest,
   javax.servlet.http.HttpServletResponse)
   throws javax.servlet.ServletException, java.io.IOException;

4) public void service(javax.servlet.ServletRequest,

javax.servlet.ServletResponse) throws javax.servlet.ServletException, java.io.IOException;

doGet() method handles GET type requests only.

doPost() method handles POST type requests only.

service() method handles both GET & POST type requests.

To handle only GET type requests, override doGet() method only.

To handle only POST type requests, override doPost() method only.

To handle both GET & POST type requests and if the task is same then override non life cycle service() method only.

To handle both GET & POST type requests and if the task is different then override both doGet() & doPost() methods.

By

Mr. Venkatesh Mansani

Naresh i Technologies