Get vs Post

In HTML, one can specify two different submission methods for a form. The method is specified inside a *FORM* element, using the *METHOD* attribute. The difference between *METHOD="GET"* (the default) and *METHOD="POST"* is primarily defined in terms of form data encoding. According to the technical HTML specifications *GET* means that form data is to be encoded (by a browser) into a URL while *POST* means that the form data is to appear within the message body of the HTTP request.

Comparison chart

	GET	**POST**
History	Parameters remain in browser history because they are part of the URL	Parameters are not saved in browser history.
Bookmarked	Can be bookmarked.	Can not be bookmarked.
BACK button/re-submit behaviour	GET requests are re-executed but may not be re-submitted to server if the HTML is stored in the browser cache.	The browser usually alerts the user that data will need to be re-submitted.
Encoding type (enctype attribute)	application/x-www-form-urlencoded	multipart/form-data or application/x-www-form-urlencoded Use multipart encoding for binary data.
Parameters	can send but the parameter data is limited to what we can stuff into the request line (URL). Safest to use less than 2K of parameters, some servers handle up to 64K	Can send parameters, including uploading files, to the server.
Hacked	Easier to hack for script kiddies	More difficult to hack
Restrictions on form data type	Yes, only ASCII characters allowed.	No restrictions. Binary data is also allowed.
Security	GET is less secure compared to POST because data sent is part of the URL. So it's saved in browser history and server logs in plaintext.	POST is a little safer than GET because the parameters are not stored in browser history or in web server logs.
Restrictions on form data length	Yes, since form data is in the URL and URL length is restricted. A safe URL length limit is often 2048 characters but varies by browser and web server.	No restrictions
Usability	GET method should not be used when sending passwords or other sensitive information.	POST method used when sending passwords or other sensitive information.
Visibility	GET method is visible to everyone (it will be displayed in the browser's address bar)	POST method variables are not displayed in the URL.

	GET	POST
	and has limits on the amount of information to send.	
Cached	Can be cached	Not cached
Large variable values	7607 character maximum size.	8 Mb max size for the POST method.

<u>ServletConfig and ServletContext</u>

ServletConfig

- ServletConfig available in javax.servlet.*; package
- ServletConfig object is one per servlet class
- Object of ServletConfig will be created during initialization process of the servlet
- This Config object is public to a particular servlet only
- *Scope*: As long as a servlet is executing, ServletConfig object will be available, it will be destroyed once the servlet execution is completed.
- We should give request explicitly, in order to create ServletConfig object for the first time
- In web.xml – <*init-param*> tag will be appear under <*servlet-class*> tag

ServletContext

- ServletContext available in javax.servlet.*; package
- ServletContext object is global to entire web application
- Object of ServletContext will be created at the time of web application deployment
- *Scope*: As long as web application is executing, ServletContext object will be available, and it will be destroyed once the application is removed from the server.
- ServletContext object will be available even before giving the first request
- In web.xml – <*context-param*> tag will be appear under <*web-app*> tag

So finally…….

No. of web applications	= That many number of ServletContext objects [1 per web application]
No. of servlet classes	= That many number of ServletConfig objects

Example of Login Form in Servlet Tutorial

Here, we are going to create the simple example to create the login form using servlet. We have used oracle10g as the database. There are 5 files required for this application.

- index.html
- FirstServlet.java
- LoginDao.java
- SecondServlet.java
- web.xml

You must need to create a table userreg with name and pass fields. Moreover, it must have contained some data. The table should be as:

1. create table userreg(name varchar2(40),pass varchar2(40));

index.html

```
<form action="FirstServlet" method="post">
        Name:<input type="text" name="username"/><br/><br/>
        Password:<input type="password" name="userpass"/><br/><br/>
        <input type="submit" value="login"/>
</form>
```

FirstServlet.java

```
import java.io.*
import java.io.PrintWriter;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;


public class FirstServlet extends HttpServlet
{
        public void doPost(HttpServletRequest request, HttpServletResponse response)
                        throws ServletException, IOException
        {

           response.setContentType("text/html");
           PrintWriter out = response.getWriter();

           String n=request.getParameter("username");
           String p=request.getParameter("userpass");

           if(LoginDao.validate(n, p))
           {
              RequestDispatcher rd=request.getRequestDispatcher("SecondServlet");
              rd.forward(request,response);
```

```
                }
                else
                {
                    out.print("Sorry username or password error");
                    RequestDispatcher rd=request.getRequestDispatcher("index.html");
                    rd.include(request,response);
                }

                out.close();
            }
}
```

LoginDao.java

```
import java.sql.*;

public class LoginDao
{
        public static boolean validate(String name,String pass)
        {
                boolean status=false;
                try
                {
                        Class.forName("com.mysql.jdbc.Driver");
                        Connection con=DriverManager.getConnection(
                                                "jdbc:mysql://localhost/MyDB","root","");

                        PreparedStatement ps=con.prepareStatement(
                                                "select * from userreg where name=? and pass=?");
                        ps.setString(1,name);
                        ps.setString(2,pass);

                        ResultSet rs=ps.executeQuery();
                        status=rs.next();

                }
                catch(Exception e)
                {
                        System.out.println(e);
                }
                return status;
        }
}
```

WelcomeServlet.java

```
import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class SecondServlet extends HttpServlet
{
	public void doPost(HttpServletRequest request, HttpServletResponse response)
						throws ServletException, IOException
	{

		response.setContentType("text/html");
		PrintWriter out = response.getWriter();

		String n=request.getParameter("username");
		out.print("Welcome "+n);

		out.close();
	}

}
```