Attributes in Servlet Programming

An attribute is a variable that can be added to the ServletContext, ServletRequest, or HttpSession objects in Servlet Programming. Attributes are used to share data between servlets, JSPs, and other components in a web application.

1) ***setAttribute() :*** Adds or updates an attribute to one of the objects (ServletContext, ServletRequest, or HttpSession).

Syntax : public abstract void **setAttribute**(java.lang.String name, java.lang.Object value);

2) ***getAttribute() :*** Retrieves an attribute from the objects.

Syntax : public abstract java.lang.Object **getAttribute**(java.lang.String name);

3) ***removeAttribute()*** : Removes an attribute from the objects.

Syntax : public abstract void **removeAttribute**(java.lang.String name);

4) ***getAttributeNames()*** : Returns an enumeration of all the attribute names that have been set.

Syntax : public abstract java.util.Enumeration<java.lang.String> **getAttributeNames**();

Servlet Programming Requests (GET Vs POST)

Requests in Servlet Programming

A request is generated when a web browser (client) communicates with the server. In HTTP, requests are categorized into different types, the most common being POST and GET.

1) **POST Request:**

- This request used to send data from the client to the server.

Features:

1. Can transmit various types of data: Text, Audio, Video, Images, etc.
2. No size limit for data transmission.
3. Data is secure since it is sent in the body of the HTTP request.

Example:

```
<form action="url" method="POST">
    <!-- form fields -->
</form>
```

We use doPost() method from the HttpServlet class to handle POST requests.

Syntax: protected void **doPost**(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException;

2) GET Request:

- This request used to send data from the client to the server.

Features:

1. Can only transmit text data.
2. Limited to sending up to 4KB or 8KB of data.
3. Data is not secure since it is appended to the URL and is visible in the browser's address bar

We use the following '4' ways to raise GET request:

1st way :Using the GET method in a form

```
<form action="url" method="GET">
    <!-- form fields -->
</form>
```

2nd way : Without specifying a method in a form

```
<form action="url">
    <!-- form fields -->
</form>
```

3) Using Hyperlinks: Clicking on a link generates a GET request.

4) Entering the URL directly in the address bar: This also raises a GET request.

Syntax: protected void **doGet**(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException;

Session Management (Session Tracking)

Session:

The period between a user logging in and logging out is called a session. During this time, the server needs to remember the user's actions and data.

Session Management (Session Tracking):

The process of tracking and managing a user's interaction state from login to logout is called Session Management or Session Tracking. Common Session Tracking Techniques are listed below,

1) Cookies

2) HttpSession

3) URL Rewriting

4) Hidden Form Fields

1) Cookies:

It is a small piece of information that travels between client (browser) and server across multiple requests. Cookies help track the user during a session. The server creates cookies and sends them to the client's browser, where they are stored. They are then sent back with each subsequent request to the server to identify the user.

Types of Cookies There are '2' types of cookies which are listed below

Persistent Cookies:

Stored in the browser until the user logs out or until a specified expiration time. They continue across browser sessions.

Non-Persistent Cookies:

Temporary and are deleted once the browser is closed.

The **javax.servlet.http.Cookie** class is used for managing cookies in servlets.

Important Methods Of Cookie Class

Method	Description
public Cookie(String name, String value)	Constructor to create a new cookie with a specified name and value.
public void setMaxAge(int expiry)	Sets the maximum age of the cookie in seconds. Use a positive value for persistent cookies & -1 for session cookies that disappear when the browser closes.
public int getMaxAge()	Returns the maximum age of the cookie in seconds.
public String getName()	Retrieves the name of the cookie.
public void setValue(String value)	Updates the cookie's value.
public String getValue()	Retrieves the current value of the cookie.