**Session Management**

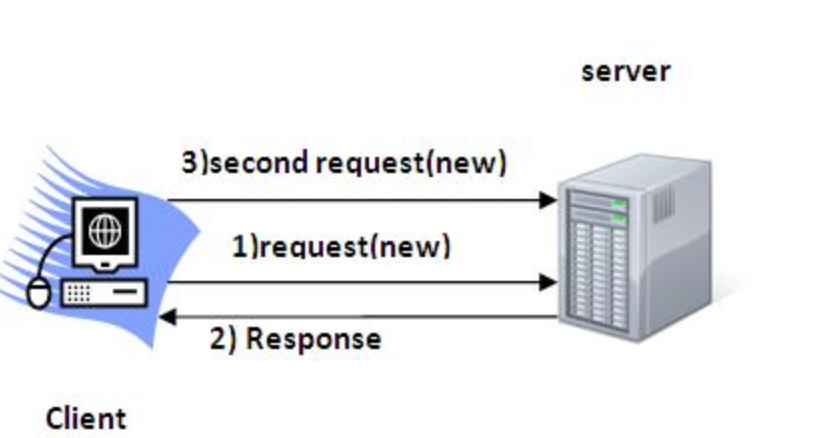
[**https://www.youtube.com/watch?v=32UGARg8AzU**](https://www.youtube.com/watch?v=32UGARg8AzU)

[**http://www.javatpoint.com/session-tracking-in-servlets**](http://www.javatpoint.com/session-tracking-in-servlets)

* **Session** means a particular interval of time.
* **Session Tracking** is a way to maintain state (data) of a user. It is also known as **session management** in servlet.

HTTP protocol is a stateless so we need to maintain state using session tracking techniques. Each time user requests to server, server treats the request as new request. So we need to maintain the state of a user to recognize to particular user.

HTTP is stateless that means each request is considered as the new request. It is shown in the figure below:



**Why use session tracking?**

**To recognize the user,** it is used to recognize the particular user.

**Session Tracking Techniques:**

There are four techniques used in Session Tracking:

1. Cookies
2. Hidden Form field
3. URL rewriting
4. HttpSession

**Cookies in Servlet:**

A cookie is a small piece of information that is persisted between the multiple client requests.

A cookie has a name, a single value, and optional attributes such as a component, path and domain qualifiers, a maximum age, and a version number.

**How Cookie works?**

By default, each request is considered as a new request. In cookies technique, we add cookie with response from the servlet. So cookie is stored in the cache of the browser. After that if request is sent by the user, cookie is added with the request by default. Thus we recognize the user as the old user.

**Disadvantage of Cookies:**

* It will not work if cookie is disabled from the browser.

Note: Gmail uses cookie technique for login. If you disable the cookie, Gmail wont work.

**Cookie class:**

Javax.servlet.http.Cookie class provides the functionality of using cookies. It provides a lot of useful methods for cookies.

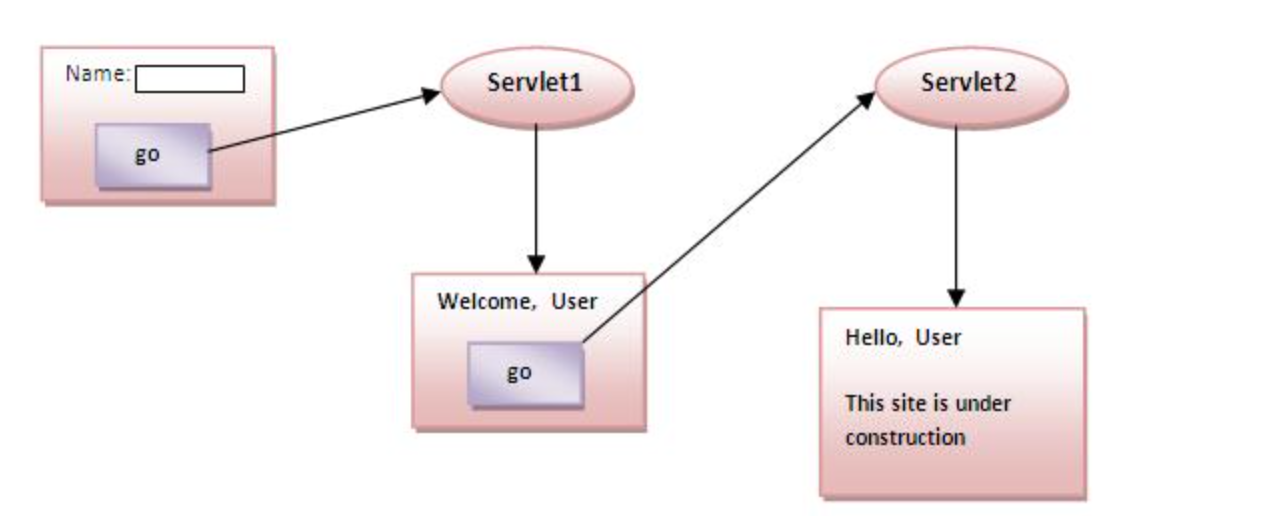
* setMaxAge(int expiry)
* getName()
* getValue()
* setName(String name)
* setValue(String value)

For adding a cookie or getting the value from the cookie, we need some methods provided by other interfaces.

* Public void addCookies(Cookie ck): method of HTTPServletResponse interface is used to add cookie in response object.
* Public Cookie[] getCookies(): method od HTTPServletRequest interface is used to return all the cookies from the browser.

**Simple Ex:**

We are storing the name of the user in the cookie object and accessing it in another servlet. As we know well about that session corresponds to the particular user. So if you access it from too many browsers with different values, you will get the different value.



**Hidden Form Field:**

In case of Hidden form field, a **hidden (invisible) textfield** is used for maintaining the state of a user.

In such case, we store the information in the hidden field and get it from another servlet. This approach is better if we have to submit form in all the pages and we don’t want to depend on the browser.

Let’s see the code to store value in hidden field.

|  |
| --- |
| <input type=”hidden” name=” JSESSIONID” value=” B5sWkPOuZ50l-GZjTJrucU9r”> |

Here, uname is the hidden field name and SSS is the hidden field value.

**Real application of hidden form field**

It is widely used in comment form a website. In such case, we store page id or page name in the hidden field so that each page can be uniquely identified.

**Advantages of hidden Form field:**

* It will always work whether cookie is disabled or not.

**Disadvantages:**

* It is maintained at server side
* Extra form submission is required on each pages
* Only textual information can be used.

**URL Rewriting:**

In URL rewriting, we append a token or identifier to the URL of the next servlet or the next resource. We can send parameter name/value pairs using the following format:

|  |
| --- |
| url?name1=value1&name2=value2&?? |

A name and value is separated using an equal = sign, a parameter name/value pair is separated from another parameter using the ampersand (&). When the user clicks the hyperlink, the parameter name/value pairs will be passed to the server. From a Servlet, we can use getParameter() method to obtain a parameter value.

**Advantages:**

* It will always work whether cookie is disabled or not
* Extra form submission is not required on each pages

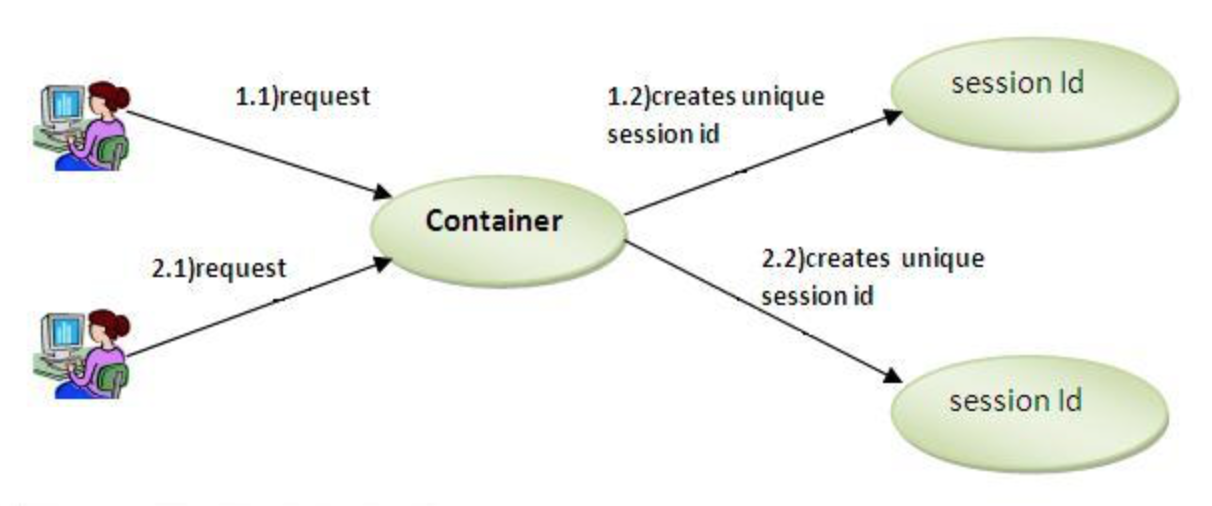
**Disadvantages:**

* It will work only with links
* It can send only textual information.

**HttpSession Interface:**

In such case, container creates a session id for each user. The container uses this id to identify the particular user. An object of HttpSession can be used to perform two tasks:

* Bind objects
* View and manipulate information about a session, such as the session identifier, creation time, and last accessed time.



**How to get the session object?**

The HttpServletRequest interface provides two methods to get the object of HttpSession:

1. **Public HttpSession getSession()**: Returns the current session associated with this request, or if the request doesn’t have a session, creates one.
2. **Public HttpSession getSession(Boolean create)**: Returns the current session associated with the request or, if there is no current session and create is true, returns a new session.

**Commonly used methods of HttpSession interface:**

1. Public String getId():
2. Public long getCreationTime():
3. Public long getLastAccessedTime():
4. Public void invalidate(): Invalidates the session then unbinds any objects bound to it.

**Example using HttpSession:**

In this example, we are setting the attribute in the session scope in one servlet and getting that value from the session scope in another servlet. To set the attribute in the session scope, we have to use setAttribute() method of HttpSession interface and to get the attribute we have to use the getAttribute() method.