UNIT II: Servlet

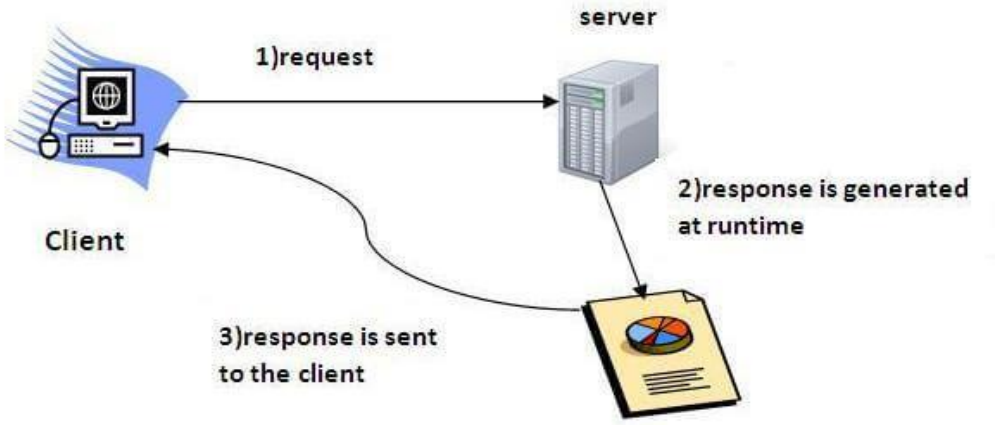
2. Introduction

- **Servlet** technology is used to create a web application (resides at server side and generates a dynamic web page).
- **Servlet** technology is robust and scalable because of java language.
- Before Servlet, CGI (Common Gateway Interface) scripting language was common as a server-side programming language.
- There are many interfaces and classes in the Servlet API such as Servlet, GenericServlet, HttpServlet, ServletRequest, ServletResponse, etc.

Servlet

Servlet can be described in many ways, depending on the context.

- Servlet is a technology which is used to create a web application.
- Servlet is an API that provides many interfaces and classes including documentation.
- Servlet is an interface that must be implemented for creating any Servlet.
- Servlet is a class that extends the capabilities of the servers and responds to the incoming requests. It can respond to any requests.
- Servlet is a web component that is deployed on the server to create a dynamic web page.

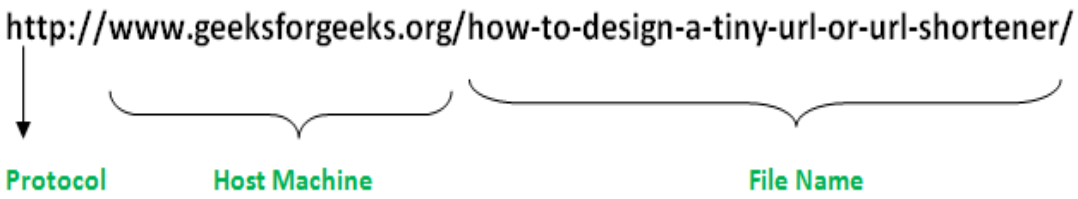


2.1 URL

- The URL class is the gateway to any of the resource available on internet.
- A Class URL represents a Uniform Resource Locator, which is a pointer to a "resource" on the World Wide Web.
- A resource can point to a simple file or directory, or it can refer to a more complicated object, such as a query to a database or to a search engine

- As many of you must be knowing that Uniform Resource Locator-URL is a string of text that identifies all the resources on Internet, telling us the address of the resource, how to communicate with it and retrieve something from it.

A Simple URL looks like:



Components of a URL

A URL can have many forms. The most general however follows three-components system-

1. **Protocol:** HTTP is the protocol here
2. **Hostname:** Name of the machine on which the resource lives.
3. **File Name:** The path name to the file on the machine.
4. **Port Number:** Port number to which to connect (typically optional).

2.2 URL Encoding

- URL encoding translates special characters from the URL to a representation that adheres to the spec and can be correctly understood and interpreted.

The *encode* method accepts two parameters:

1. *data* – string to be translated
2. *encodingScheme* – name of the character encoding

- This *encode* method converts the string into ***application/x-www-form-urlencoded*** format.
- The encoding scheme will convert special characters into two digits hexadecimal representation of 8 bits that will be represented in the form of "*%xy*". When we are dealing with path parameters or adding parameters which are dynamic, then we will encode the data and then send to the server.

2.3 Web Server

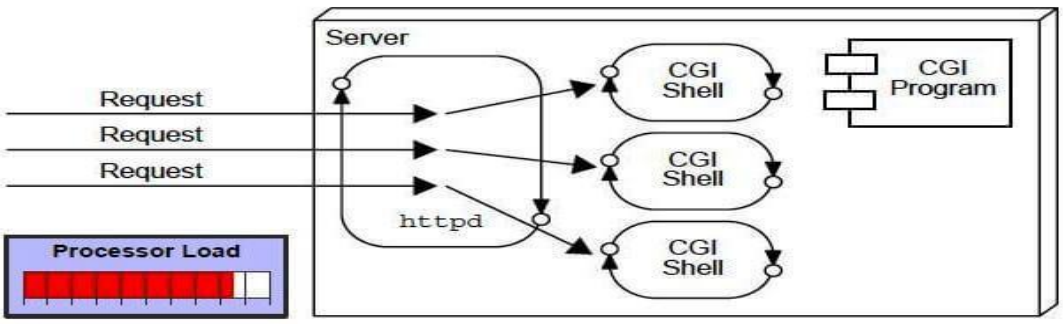
A web application is an application accessible from the web. A web application is composed of web components like Servlet, JSP, Filter, etc. and other elements such as HTML, CSS, and JavaScript. The web components typically execute in Web Server and respond to the HTTP request.

2.4 CGI

- The common gateway interface (CGI) is a standard way for a Web server to pass a Web user's request to an application program and to receive data back to forward to the user.
- When the user requests a Web page, the server sends back the requested page. However, when a user fills out a form on a Web page and sends it in, it usually needs to be processed by an application program.
- The Web server typically passes the form information to a small application program that processes the data and may send back a confirmation message. This method or convention for passing data back and forth between the server and the application is called the common gateway interface (CGI).
- It is part of the Web's Hypertext Transfer Protocol (HTTP).

In CGI application, when a client makes a request to access dynamic Web pages, the Web server performs the following operations:

- It first locates the requested web page *i.e* the required CGI application using URL.
- It then creates a new process to service the client's request.
- Invokes the CGI application within the process and passes the request information to the server.
- Collects the response from CGI application.
- Destroys the process, prepares the HTTP response and sends it to the client.



Disadvantages of CGI

There are many problems in CGI technology:

1. If the number of clients increases, it takes more time for sending the response.
2. For each request, it starts a process, and the web server is limited to start processes.
3. It uses platform dependent language e.g. C, C++, perl.

Servlet

- Servlets are the Java programs that runs on the Java-enabled web server or application server. They are used to handle the request obtained from the web server, process the request, produce the response, then send response back to the web server.

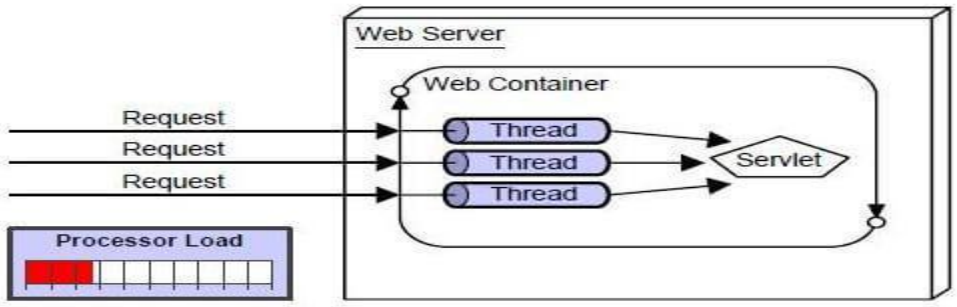
  Properties of Servlets :

  - Servlets work on the server-side.
  - Servlets capable of handling complex request obtained from web server.

Execution of Servlets :

Execution of Servlets involves the six basic steps:

1. The clients send the request to the web server.
2. The web server receives the request.
3. The web server passes the request to the corresponding servlet.
4. The servlet processes the request and generate the response in the form of output.
5. The servlet send the response back to the web server.
6. The web server sends the response back to the client and the client browser displays it on the screen.



Advantage of Servlet

- ✓ There are many advantages of Servlet over CGI. The web container creates threads for handling the multiple requests to the Servlet. Threads have many benefits over the Processes such as they share a common memory area, lightweight, cost of communication between the threads are low. The advantages of Servlet are as follows:
- ✓ Servlet is **faster** than CGI as it not involves the creation of a new process for every new request received.
- ✓ Servlets as written in Java are **platform independent**.
- ✓ Removes the overhead of creating a **new process** for each request as Servlet doesn't run in a separate process. There is only a single instance which handles all requests concurrently. This also saves the memory and allows a Servlet to easily manage client state.
- ✓ It is a server-side component, so Servlet inherits the **security** provided by the Web server.
- ✓ The **API** designed for Java Servlet automatically acquires the advantages of Java platform such as platform independent and portability. In addition, it obviously can use the wide range of APIs created on Java platform such as **JDBC** to access the database.

The Servlet Container

- ❖ **Servlet container**, also known as **Servlet engine** is an integrated set of objects that provide run time environment for Java Servlet components.
- ❖ In simple words, it is a system that manages Java Servlet components on top of the Web server to handle the Web client requests.

Services provided by the Servlet container :

- **Network Services :** Loads a Servlet class. The loading may be from a local file system, a remote file system or other network services. The Servlet container provides the network services over which the request and response are sent.
- **Decode and Encode MIME based messages :** Provides the service of decoding and encoding MIME-based messages.
- **Manage Servlet container :** Manages the lifecycle of a Servlet.
- **Resource management :** Manages the static and dynamic resource, such as HTML files, Servlets and JSP pages.

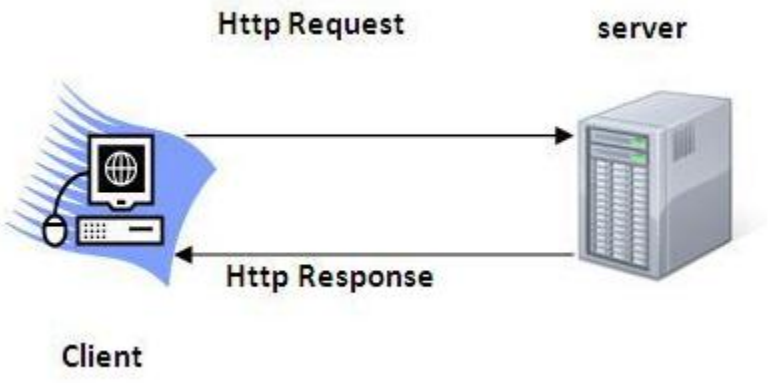
- **Security Service :** Handles authorization and authentication of resource access.
- **Session Management :** Maintains a session by appending a **session ID** to the URL path.

2.5 Difference between Servlet Vs CGI

Servlet	CGI(Common Gateway Interface)
Servlets are portable and efficient.	CGI is not portable
In Servlets, sharing of data is possible.	In CGI, sharing of data is not possible.
Servlets can directly communicate with the web server.	CGI cannot directly communicate with the web server.
Servlets are less expensive than CGI.	CGI are more expensive than Servlets.
Servlets can handle the cookies.	CGI cannot handle the cookies.

2.6 HTTP

- ✓ The Hypertext Transfer Protocol (HTTP) is application-level protocol for collaborative, distributed, hypermedia information systems.
- ✓ It is the data communication protocol used to establish communication between client and server.
- ✓ HTTP is TCP/IP based communication protocol, which is used to deliver the data like image files, query results, HTML files etc on the World Wide Web (WWW) with the default port is TCP 80.



The Basic Characteristics of HTTP (Hyper Text Transfer Protocol):

- It is the protocol that allows web servers and browsers to exchange data over the web.
- It is a request response protocol.
- It uses the reliable TCP connections by default on TCP port 80.
- It is stateless means each request is considered as the new request. In other words, server doesn't recognize the user by default.

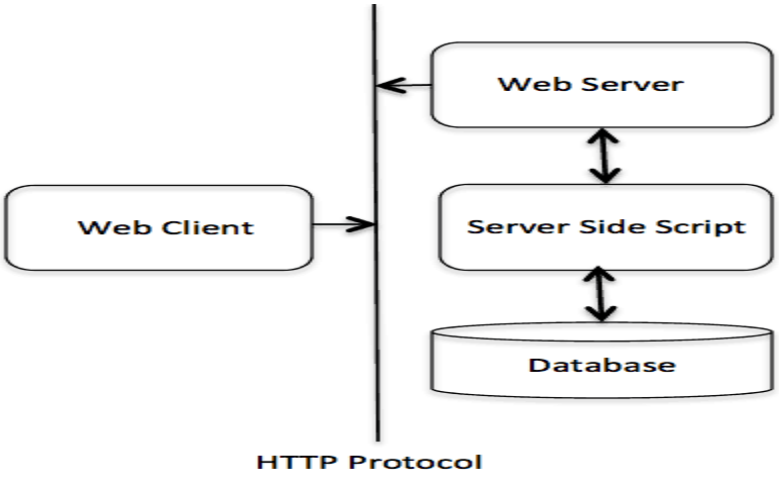
The Basic Features of HTTP (Hyper Text Transfer Protocol):

There are three fundamental features that make the HTTP a simple and powerful protocol used for communication:

- **HTTP is media independent:** It specifies that any type of media content can be sent by HTTP as long as both the server and the client can handle the data content.
- **HTTP is connectionless:** It is a connectionless approach in which HTTP client i.e., a browser initiates the HTTP request and after the request is sent the client disconnects from server and waits for the response.
- **HTTP is stateless:** The client and server are aware of each other during a current request only. Afterwards, both of them forget each other. Due to the stateless nature of protocol, neither the client nor the server can retain the information about different request across the web pages.

The Basic Architecture of HTTP (Hyper Text Transfer Protocol):

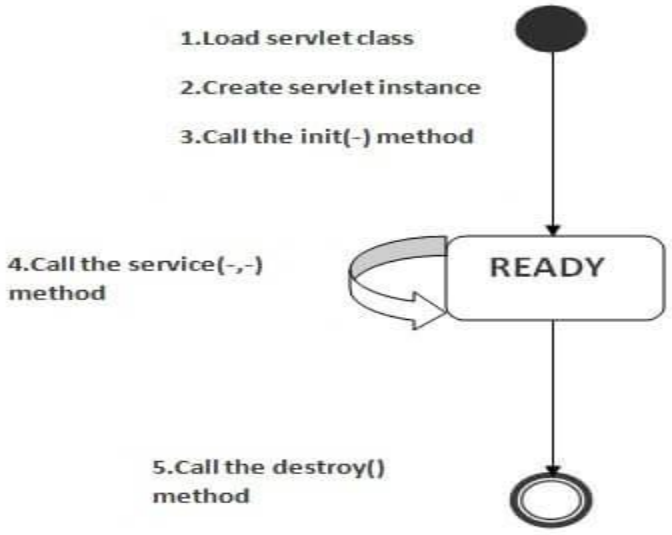
- ✓ The below diagram represents the basic architecture of web application and depicts where HTTP stands:
- ✓ HTTP is request/response protocol which is based on client/server based architecture. In this protocol, web browser, search engines, etc. behave as HTTP clients and the Web server like Servlet behaves as a server



2.7 Life Cycle of a Servlet (Servlet Life Cycle)

The web container maintains the life cycle of a servlet instance. Let's see the life cycle of the servlet:

1. Servlet class is loaded.
2. Servlet instance is created.
3. init method is invoked.
4. service method is invoked.
5. destroy method is invoked.



- ✓ Above diagram, there are three states of a servlet: new, ready and end.
- ✓ The servlet is in new state if servlet instance is created.
- ✓ After invoking the init() method, Servlet comes in the ready state. I
- ✓ In the ready state, servlet performs all the tasks.
- ✓ When the web container invokes the destroy() method, it shifts to the end state.

1) Servlet class is loaded

The classloader is responsible to load the servlet class. The servlet class is loaded when the first request for the servlet is received by the web container.

2) Servlet instance is created

The web container creates the instance of a servlet after loading the servlet class. The servlet instance is created only once in the servlet life cycle.

3) init method is invoked

The web container calls the init method only once after creating the servlet instance. The init method is used to initialize the servlet. It is the life cycle method of the javax.servlet.Servlet interface.

Syntax

```
public void init(ServletConfig config) throws ServletException
```

4) service method is invoked

The web container calls the service method each time when request for the servlet is received. If servlet is not initialized, it follows the first three steps as described above then calls the service method. If servlet is initialized, it calls the service method. Notice that servlet is initialized only once.

Syntax:

```
public void service(ServletRequest request, ServletResponse response) throws
ServletException, IOException
```

5) destroy method is invoked

The web container calls the destroy method before removing the servlet instance from the service. It gives the servlet an opportunity to clean up any resource for example memory, thread etc.

Syntax:

```
public void destroy()
```

2.8 Servlet API

- ✓ The javax.servlet and javax.servlet.http packages represent interfaces and classes for servlet api.
- ✓ The **javax.servlet** package contains many interfaces and classes that are used by the servlet or web container. These are not specific to any protocol.
- ✓ The **javax.servlet.http** package contains interfaces and classes that are responsible for http requests only.

Interfaces in javax.servlet package

There are many interfaces in javax.servlet package. They are as follows:

1. Servlet
2. ServletRequest
3. ServletResponse
4. RequestDispatcher
5. ServletConfig
6. ServletContext
7. SingleThreadModel
8. Filter
9. FilterConfig
10. FilterChain
11. ServletRequestListener
12. ServletRequestAttributeListener
13. ServletContextListener
14. ServletContextAttributeListener

ServletRequest Interface

- ✓ An object of ServletRequest is used to provide the client request information to a servlet such as content type, content length, parameter names and values, header informations, attributes etc.

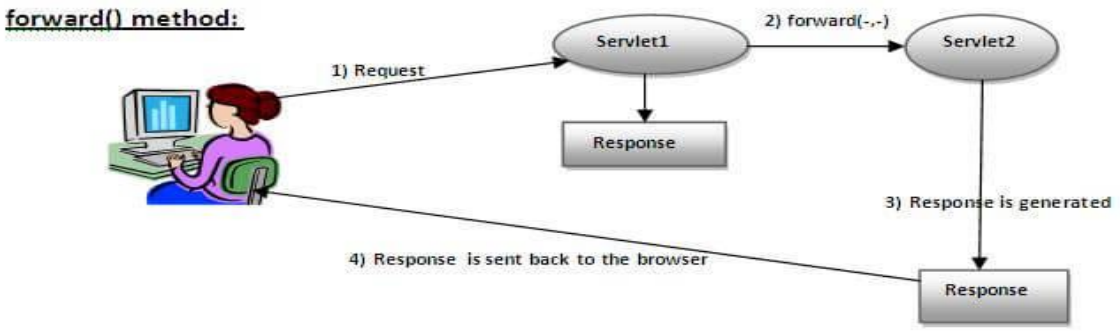
Request Dispatcher in Servlet

- ✓ The RequestDispatcher interface provides the facility of dispatching the request to another resource it may be html, servlet or jsp. This interface can also be used to include the content of another resource also. It is one of the way of servlet collaboration.
- ✓ There are two methods defined in the RequestDispatcher interface.

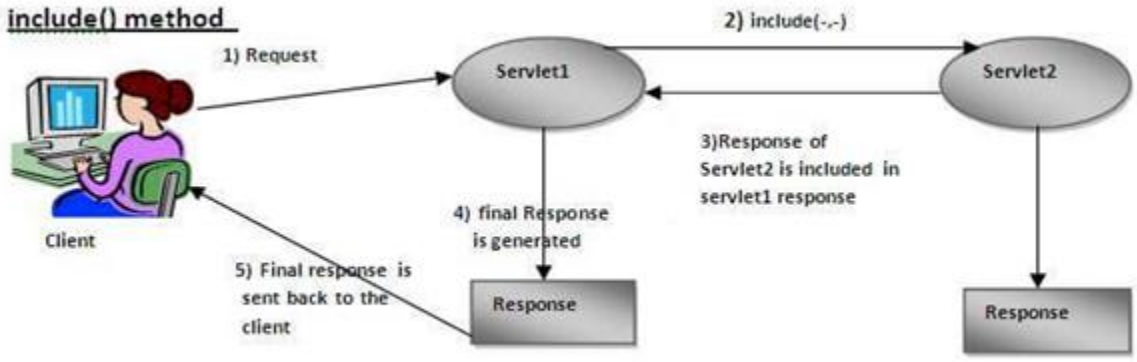
Methods of RequestDispatcher interface

The RequestDispatcher interface provides two methods. They are:

1. **public void forward(ServletRequest request,ServletResponse response)throws ServletException,java.io.IOException:**Forwards a request from a servlet to another resource (servlet, JSP file, or HTML file) on the server.
2. **public void include(ServletRequest request,ServletResponse response)throws ServletException,java.io.IOException:**Includes the content of a resource (servlet, JSP page, or HTML file) in the response.



- ✓ Response of second servlet is sent to the client. Response of the first servlet is not displayed to the user.



2.9 Running a Servlet:

- ❖ It is important to learn how servlet works for understanding the servlet well. Here, we are going to get the internal detail about the first servlet program.
- ❖ The server checks if the servlet is requested **for the first time**.

If yes, web container does the following tasks:

- ✓ loads the servlet class.
- ✓ instantiates the servlet class.
- ✓ calls the init method passing the ServletConfig object

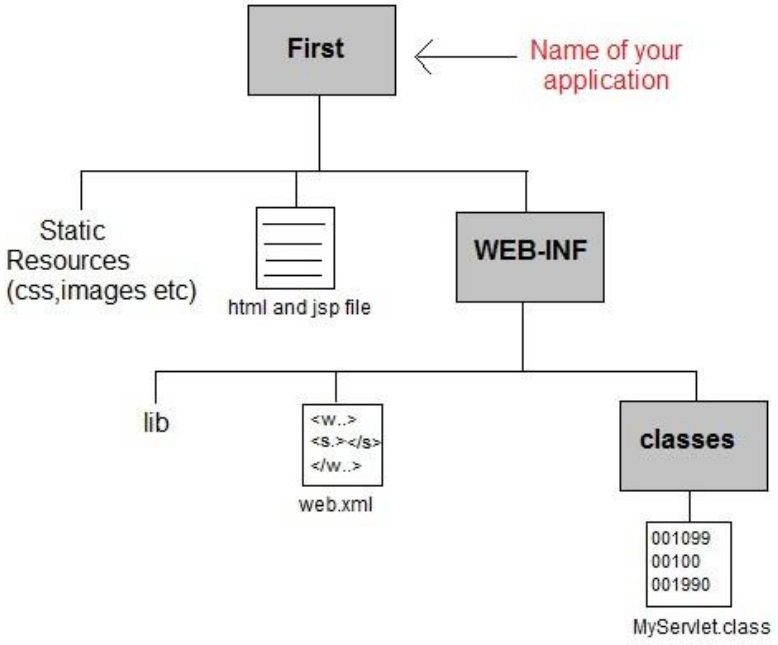
else

- ❖ calls the service method passing request and response objects
- ❖ The web container calls the destroy method when it needs to remove the servlet such as at time of stopping server or undeploying the project.

The web container is responsible to handle the request. Let's see how it handles the request.

- maps the request with the servlet in the web.xml file.
- creates request and response objects for this request
- calls the service method on the thread
- The public service method internally calls the protected service method
- The protected service method calls the doGet method depending on the type of request.
- The doGet method generates the response and it is passed to the client.

- After sending the response, the web container deletes the request and response objects. The thread is contained in the thread pool or deleted depends on the server implementation.



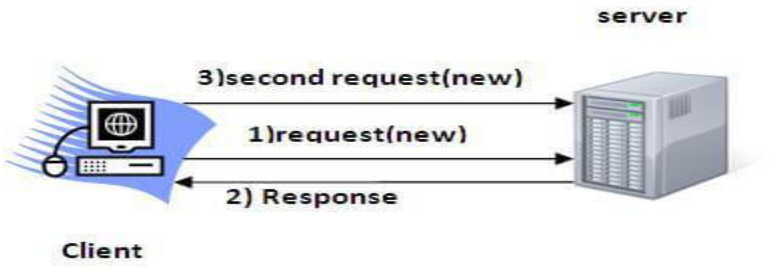
2.10 Session Tracking in Servlets

1. Session Tracking
2. Session Tracking Techniques

Session simply means a particular interval of time.

- **Session Tracking** is a way to maintain state (data) of an 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 the server, server treats the request as the new request. So we need to maintain the state of an 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 given 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:

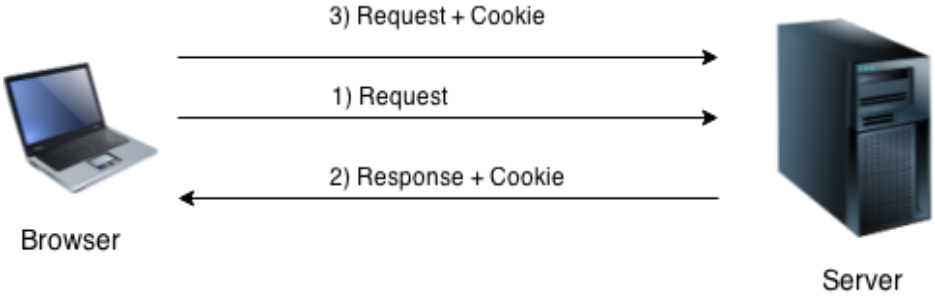
1. **Cookies**
2. **Hidden Form Field**
3. **URL Rewriting**
4. **HttpSession**

1. 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 comment, 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 request by default. Thus, we recognize the user as the old user.



Types of Cookie

There are 2 types of cookies in Servlet.

1. Non-persistent cookie
2. Persistent cookie

- ✓ **Non-persistent cookie**

  It is **valid for single session** only. It is removed each time when user closes the browser.

- ✓ **Persistent cookie**

  It is **valid for multiple session** . It is not removed each time when user closes the browser. It is removed only if user logout or signout.

Advantage of Cookies

1. Simplest technique of maintaining the state.
2. Cookies are maintained at client side.

Disadvantage of Cookies

1. It will not work if cookie is disabled from the browser.
2. Only textual information can be set in Cookie object.

Cookie class

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

Constructor of Cookie class

Constructor	**Description**
Cookie()	constructs a cookie.
Cookie(String name, String value)	constructs a cookie with a specified name and value.

Useful Methods of Cookie class

There are given some commonly used methods of the Cookie class.

Method	**Description**
public void setMaxAge(int expiry)	Sets the maximum age of the cookie in seconds.

public String getName()	Returns the name of the cookie. The name cannot be changed after creation.
public String getValue()	Returns the value of the cookie.
public void setName(String name)	changes the name of the cookie.
public void setValue(String value)	changes the value of the cookie.

Other methods required for using Cookies

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

1. **public void addCookie(Cookie ck):**method of HttpServletResponse interface is used to add cookie in response object.
2. **public Cookie[] getCookies():**method of HttpServletRequest interface is used to return all the cookies from the browser.

How to create Cookie?

Let's see the simple code to create cookie.

```
1. Cookie ck=new Cookie("user","sonoo jaiswal");//creating cookie object
2. response.addCookie(ck);//adding cookie in the response
```

How to delete Cookie?

Let's see the simple code to delete cookie. It is mainly used to logout or signout the user.

```
1. Cookie ck=new Cookie("user","");//deleting value of cookie
2. ck.setMaxAge(0);//changing the maximum age to 0 seconds
3. response.addCookie(ck);//adding cookie in the response
```

How to get Cookies?

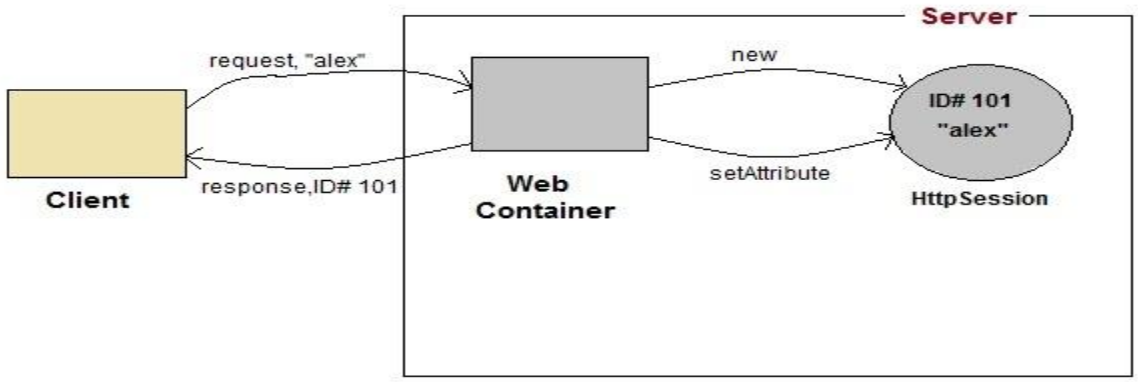
Let's see the simple code to get all the cookies.

```
1. Cookie ck[]=request.getCookies();
2. for(int i=0;i<ck.length;i++){
3.  out.print("<br>"+ck[i].getName()+" "+ck[i].getValue());//printing name and valu
   e of cookie
4. }
```

2. HttpSession

HttpSession object is used to store entire session with a specific client. We can store, retrieve and remove attribute from **HttpSession** object. Any servlet can have access to **HttpSession** object throughout the getSession() method of the **HttpServletRequest** object.

Working Concept of HttpSession



1. On client's first request, the **Web Container** generates a unique session ID and gives it back to the client with response. This is a temporary session created by web container.
2. The client sends back the session ID with each request. Making it easier for the web container to identify where the request is coming from.
3. The **Web Container** uses this ID, finds the matching session with the ID and associates the session with the request.

HttpSession Interface

Creating a new session

```
HttpSession session = request.getSession();
```

getSession() method returns a session. If the session already exist, it return the esisting session else create a new sesion

```
HttpSession session = request.getSession(true);
```

getSession(true) always return a new session

Getting a pre-existing session

```
HttpSession session = request.getSession(false);
```

return a pre-existing session

Destroying a session

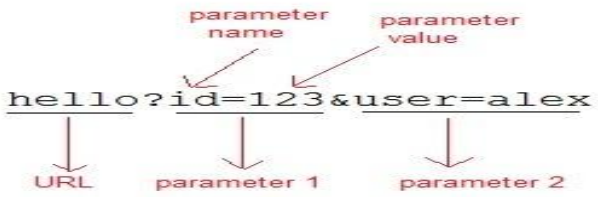
```
session.invalidate();
```

destroy a session

3. Using URL Rewriting for Session Management

- If the client has disabled cookies in the browser then session management using cookie wont work. In that case **URL Rewriting** can be used as a backup. **URL rewriting** will always work.
- In URL rewriting, a token(parameter) is added at the end of the URL. The token consist of name/value pair seperated by an equal(=) sign.

For Example:



- When the User clicks on the URL having parameters, the request goes to the **Web Container** with extra bit of information at the end of URL. The **Web Container** will fetch the extra part of the requested URL and use it for session management.
- The getParameter() method is used to get the parameter value at the server side.

4. Using Hidden Form Field for Session Management

Hidden form field can also be used to store session information for a particular client. In case of hidden form field a hidden field is used to store client state. In this case user information is stored in hidden field value and retrieved from another servlet.

Advantages :

- Does not have to depend on browser whether the cookie is disabled or not.
- Inserting a simple HTML Input field of type hidden is required. Hence, its easier to implement.

Disadvantage :

- Extra form submission is required on every page. This is a big overhead.