UNIVERSITY of INFORMATION TECHNOLOGY

# Java Servlet

Hay Mar Mo Mo Lwin
Ph.D (IT), Associate Professor
Faculty of Computer Science

# What is Servlet ?

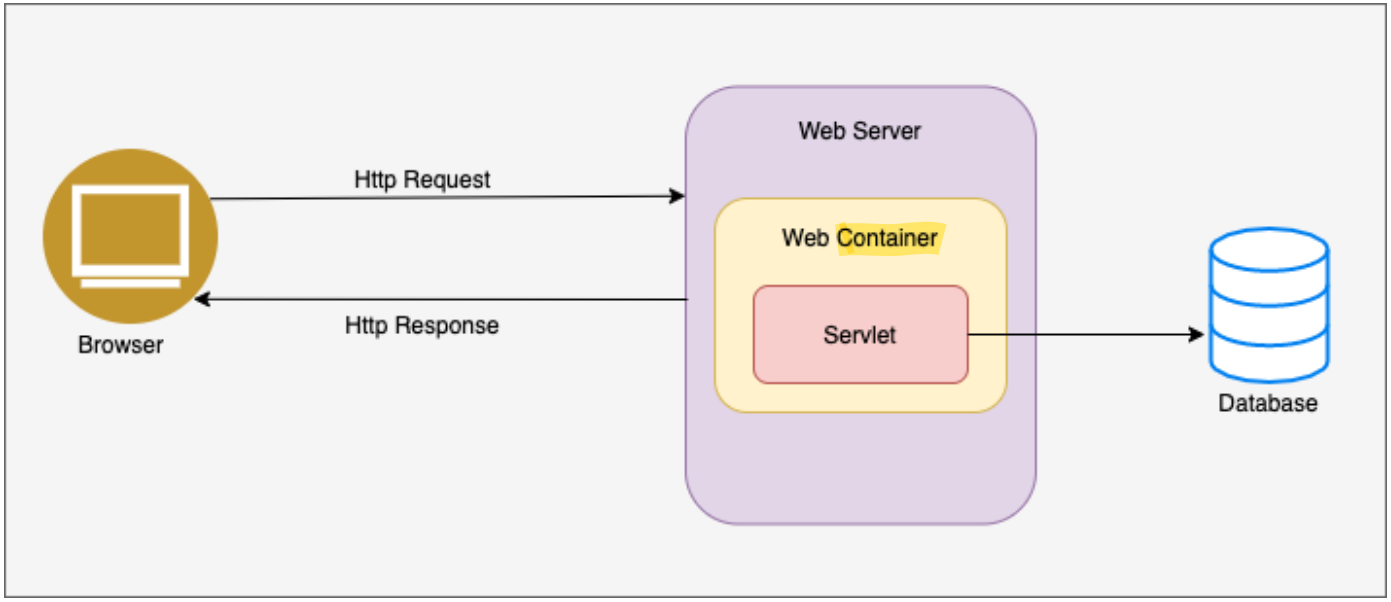
UNIVERSITY of INFORMATION TECHNOLOGY

- **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. However, there were many disadvantages to this technology.

- There are many interfaces and classes in the Servlet API such as Servlet, GenericServlet, HttpServlet, ServletRequest, ServletResponse, etc.

# Servlet lifecycle and architecture




- The user sends HTTP requests to the web server
- The server has the web container containing servlet, which gathers data from the database and creates a response.
- The response created by servlet is sent through HTTP Response to the client browser.

# Advantages of Servlet

- Efficient
  - Servlet spawns a light weight threat for each browser request.
- Convenient
  - Servlets include built-in functionality for reading HTML form data, handling cookies and tracking user sessions.
- Powerful
  - Servlet can communicate directly to the web servers.
  - Multiple servlets can share data

# Advantages of Servlet (Cont'd)

UNIVERSITY of INFORMATION TECHNOLOGY

- Portable
  - access across different operating systems.
- Secure
  - has a number of built-in security layers
- Inexpensive
  - there are number of free web servers available for personal use or commercial purpose

# Using Apache Tomcat Server

UNIVERSITY of INFORMATION TECHNOLOGY

- Apache Tomcat is a Java-capable HTTP server, which could execute special Java programs known as "Java Servlet" and "Java Server Pages (JSP)".
- It is an open-source project, under the "Apache Software Foundation".
- Tomcat 9.0 (Jan 2018): RI for Servlet 4.0. http://tomcat.apache.org
- You need to install Tomcat to try out Java servlets.

# Servlet API

- Servlet are based on two main packages:
  - javax.Servlet
  - javax.Servlet.http

- The **javax.servlet** package contains many interfaces and classes that are used by the servlet or web container.

- The **javax.servlet.http** package contains interfaces and classes that are responsible for http requests only.

# The javax.Servlet.http Package

- Some interfaces and class in *javax.Servlet.http* package
  - HttpServletRequest
  - HttpServletResponse
  - HttpSession
  - Cookie
  - HttpServlet

# The javax.Servlet Exception Class

- Two exceptions are present in the *javax.Servlet* package.
  - ServletException
  - UnavailableException

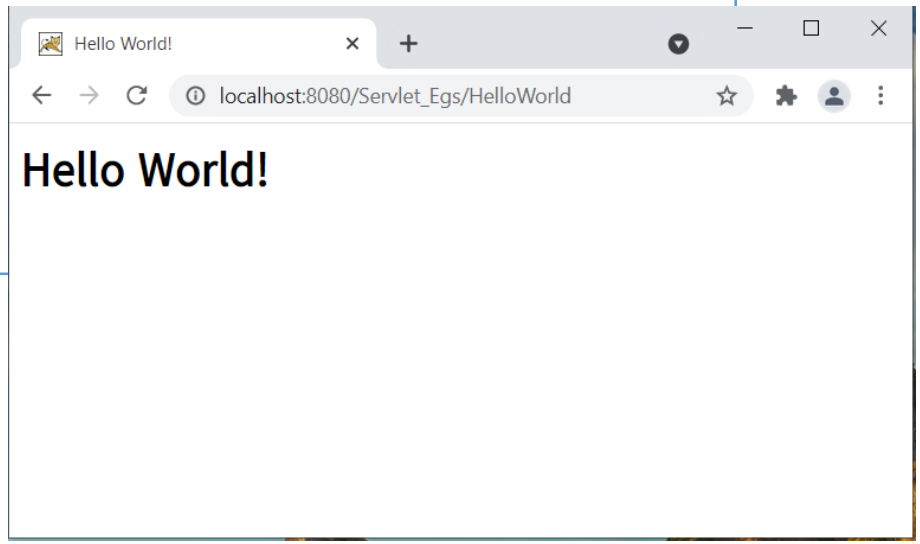
# HelloWorld Servlet

UNIVERSITY of INFORMATION TECHNOLOGY

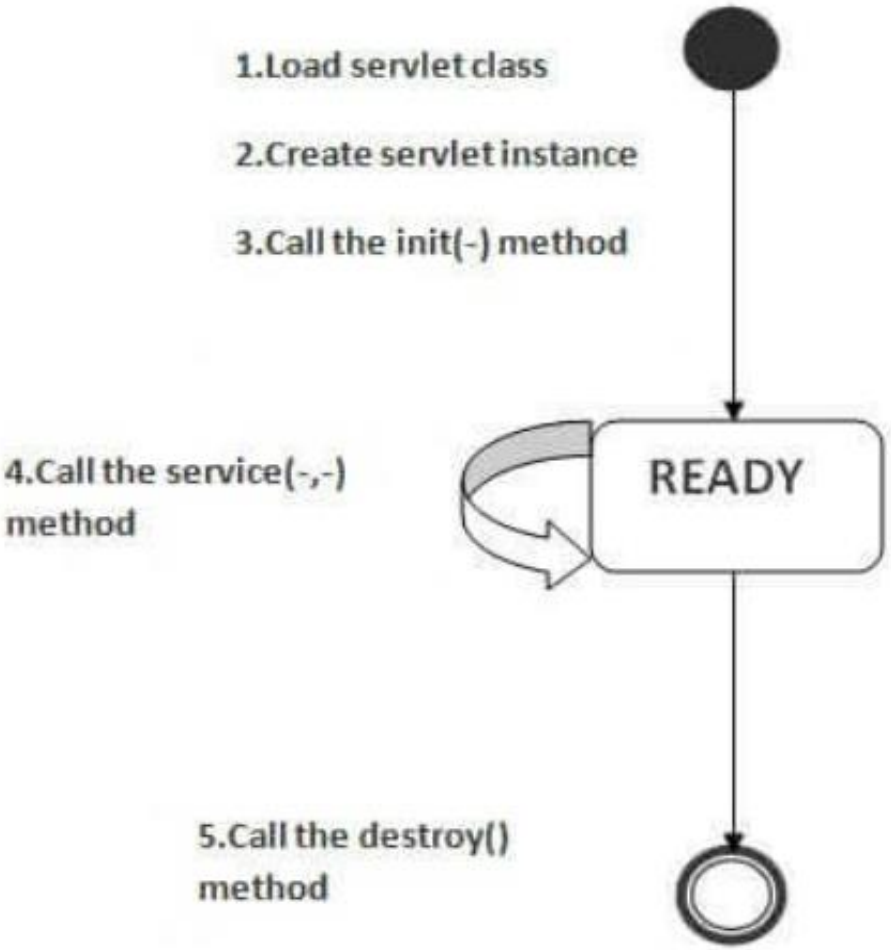
```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class HelloWorld extends HttpServlet {
      public void doGet(HttpServletRequest request, HttpServletResponse
   response)
                                  throws IOException, ServletException
     {
           response.setContentType("text/html");
           PrintWriter out = response.getWriter();
```

```
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Hello World!</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1>Hello World!</h1>");
        out.println("</body>");
        out.println("</html>");
    }
}
```



# Servlet Life Cycle



# The *init()* Method

UNIVERSITY of INFORMATION TECHNOLOGY

- The ***init()*** method is called at the time of creation of the servlet for the first time.
- It is called only once.
- It is used for one-time initializations.

```
public void init() throws ServletException {
      // Initialization code...
   }
 public void init(ServletConfig config) throws ServletException{
}
```

# The *service()* Method

UNIVERSITY of INFORMATION TECHNOLOGY

- The servlet container (i.e. web server) calls the ***service()*** method to handle requests coming from the client( browsers) and to write the formatted response back to the client.

- The service() method checks the HTTP request type (GET, POST) and calls ***doGet()***, ***doPost()***, etc. as appropriate.

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

# The *doGet()* Method

- A GET request results from a normal request for a URL or from an HTML form that has no METHOD specified.

```
public void doGet(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
    // Servlet code
}
```

# The *doPost()* Method

- A POST request results from an HTML form that specifically lists POST as the METHOD.

```
Public void doPost(HttpServletRequest request, HttpServletResponse response)
   throws ServletException, IOException {
   // Servlet code
}
```

# *destroy()* Method

UNIVERSITY of INFORMATION TECHNOLOGY

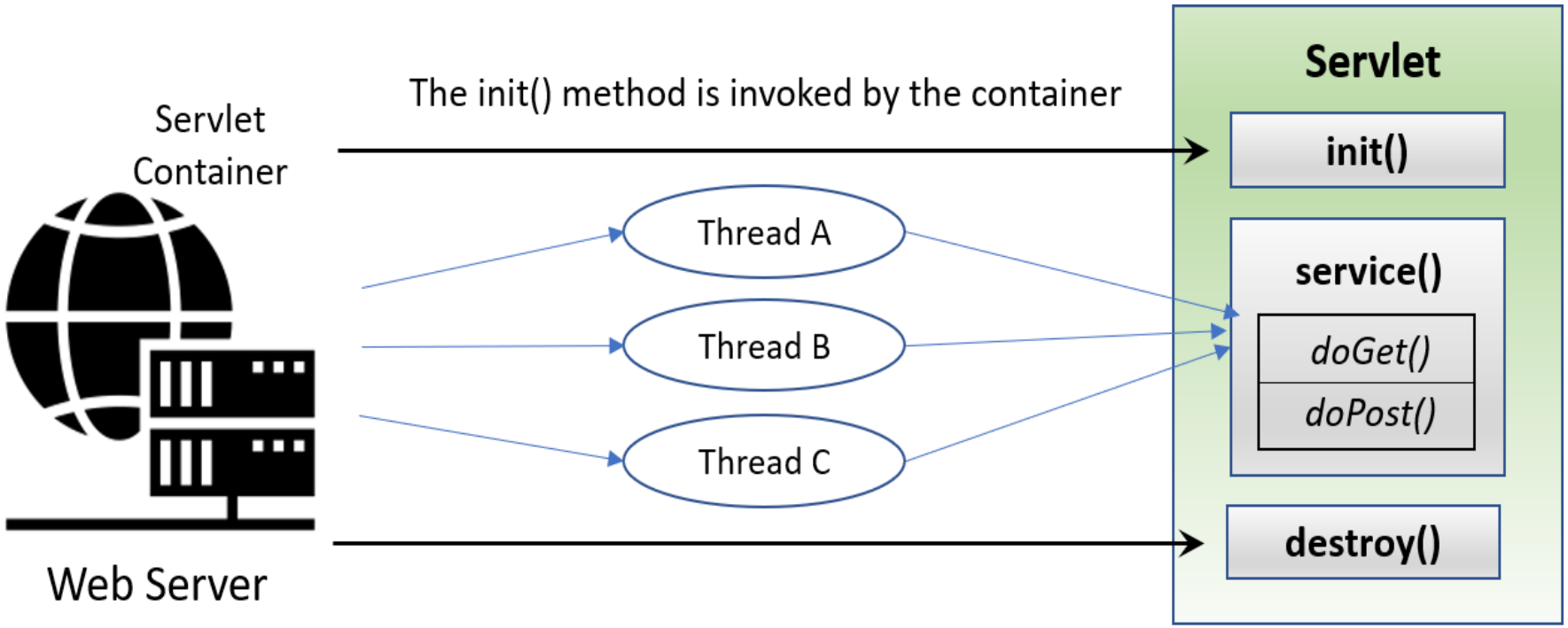
- In the destroy() method, the server may decide to remove a previously loaded Servlet instance.
- It is called only once at the end of the life cycle of a servlet.
- It is used to close database connections, halt background threads, write cookie lists, perform other cleanup activities.

```
public void destroy() {

  // Finalization code...

}
```

# Architecture of Servlet Life Cycle



# Example: Servlet Life Cycle

UNIVERSITY of INFORMATION TECHNOLOGY

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class ServletLifeCycle extends HttpServlet {

private String message = "";
    public ServletLifeCycle() {
        super();  // TODO Auto-generated constructor stub
    }
    public void init() throws ServletException {
        message = “World";  // Do required initialization
    }
```

# Example: Servlet Life Cycle (Cont'd)

UNIVERSITY of INFORMATION TECHNOLOGY

```
protected void doGet(HttpServletRequest  request, HttpServletResponse
response)
        throws ServletException, IOException  {

            response.setContentType("text/html"); // Set response content type
            PrintWriter out = response.getWriter();
            out.println("<h1>" + "Hello " + message + "</h1>");
}
```

doGet() or doPost() method which navigate from service() methodT

```
public void destroy() {
            System.out.println("This is Destroy methods");
}
}
```

destroy() method

UNIVERSITY of INFORMATION TECHNOLOGY

# web.xml

```
</web-app>
<servlet>
    <servlet-name>ServletLifeCycle</servlet-name>
    <servlet-class>ServletLifeCycle</servlet-class>
</servlet>

<servlet-mapping>
    <servlet-name>ServletLifeCycle</servlet-name>
    <url-pattern>/ServletLifeCycle</url-pattern>
</servlet-mapping>
</web-app>
```

# Servlet Example

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
@WebServlet("/ServletLifeCycle")
public class ServletLifeCycle extends HttpServlet {

private String message = "";
    public ServletLifeCycle() {
        super();  // TODO Auto-generated constructor stub
    }
    public void init() throws ServletException {
        message = "Testing";
    }
```

import javax.servlet.annotation.WebServlet;

Do not need to configure at web.xml mapping

```
</web-app>
    <servlet>…
    </servlet>
    <servlet-mapping>…
    </servlet-mapping>
</web-app>
```

# Cookies

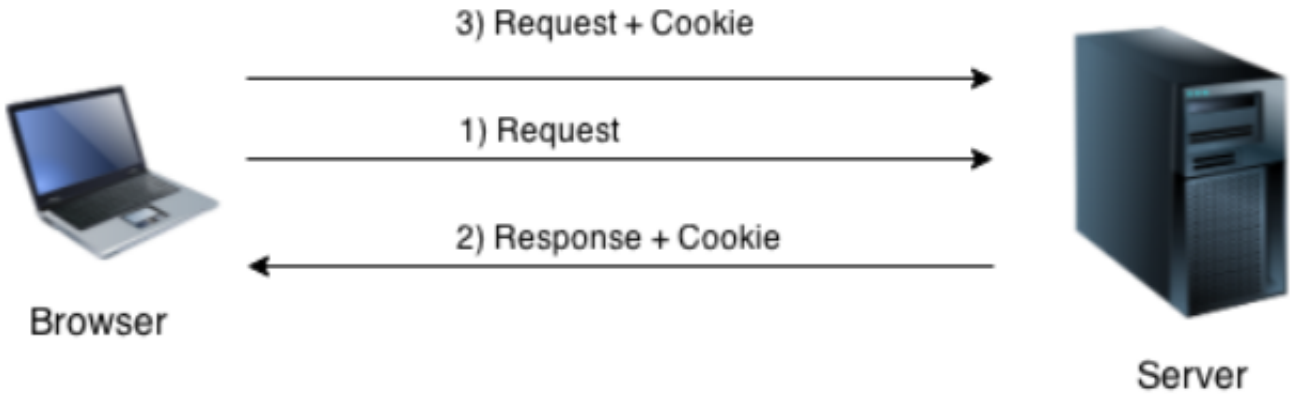
UNIVERSITY of INFORMATION TECHNOLOGY

- A cookie is a small amount of data that is saved on the client's machine and can be created by and referenced by a Java servlet using the Java servlet cookie API.
- A cookie is composed of two pieces. These are the cookie name and the cookie value, both of which are created by the Java servlet.
- The cookie name is used to identify a particular cookie from among other cookies stored at the client.
- Any character can be used except the following: [] () =," / ? @ : ;. However, version 1.0 can use these characters.

# How Cookie works

By default, each request is considered as a new request.
In cookies technique,

- 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.



# Methods of Cookie class

UNIVERSITY of INFORMATION TECHNOLOGY

| 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. | |

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

**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?

UNIVERSITY of INFORMATION TECHNOLOGY

```
Cookie myCookie = new Cookie ( "userID", "123");
response .addCookie(myCookie) ;
```

## How to delete Cookie?

```
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?

```
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 value of cookie
  }
```

# Simple example of Servlet Cookies

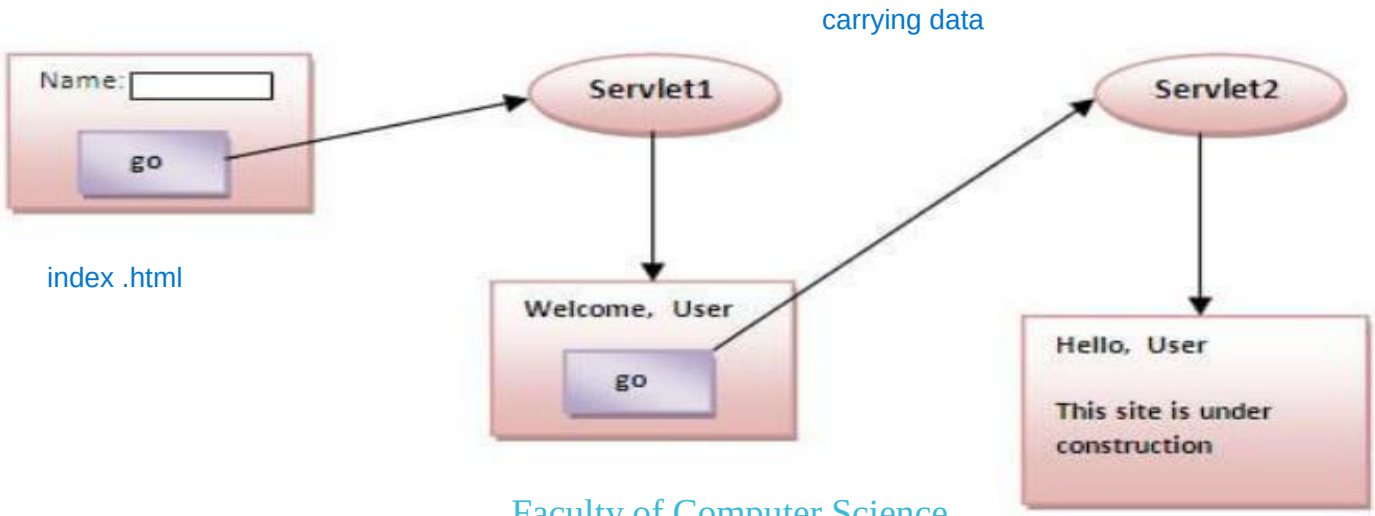
UNIVERSITY of INFORMATION TECHNOLOGY

In this example,

- Storing the name of the user in the cookie object and accessing it in another servlet.

- Session corresponds to the particular user.

- Access it from too many browsers with different values, you will get the different value.

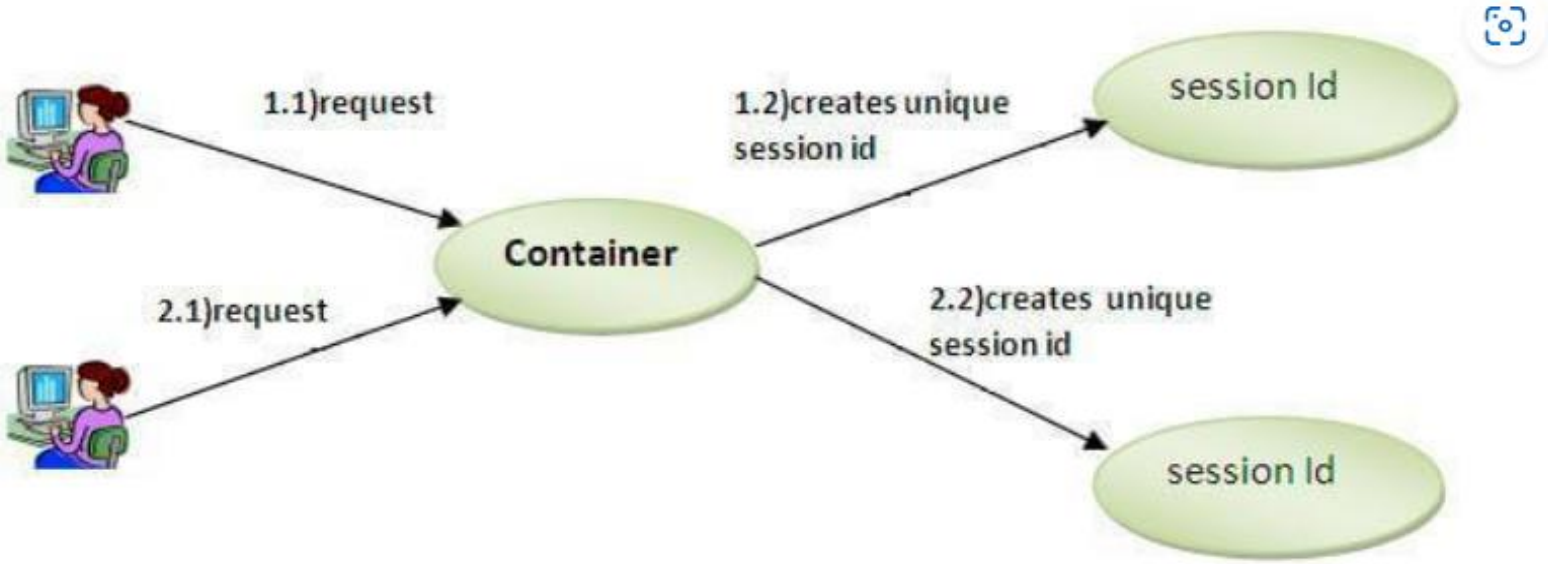


# Http Session

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:

1.bind objects

2.view and manipulate information about a session, such as the session identifier, creation time, and last accessed time.



# How to get the HttpSession 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 does not have a session, creates one.

**2.public HttpSession getSession(boolean create):**Returns the current HttpSession associated with this 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():**Returns a string containing the unique identifier value.

**2.public long getCreationTime():**Returns the time when this session was created, measured in milliseconds since midnight January 1, 1970 GMT.

**3.public long getLastAccessedTime():**Returns the last time the client sent a request associated with this session, as the number of milliseconds since midnight January 1, 1970 GMT.

**4.public void invalidate():**Invalidates this session then unbinds any objects bound to it.

```java
@WebServlet("/example")
public class ExampleServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws 
        // Setting a cookie
        Cookie cookie = new Cookie("user", "john_doe");
        cookie.setMaxAge(60 * 60 * 24); // 1 day
        response.addCookie(cookie);

        // Creating a session
        HttpSession session = request.getSession();
        session.setAttribute("user", "john_doe");
        session.setMaxInactiveInterval(30 * 60); // 30 minutes

        response.setContentType("text/html");
        response.getWriter().println("Cookie and session have been set.");
    }
}
```

# Assignment

Develop a LogIn and LogOut web application by using servlet HTTPSession.