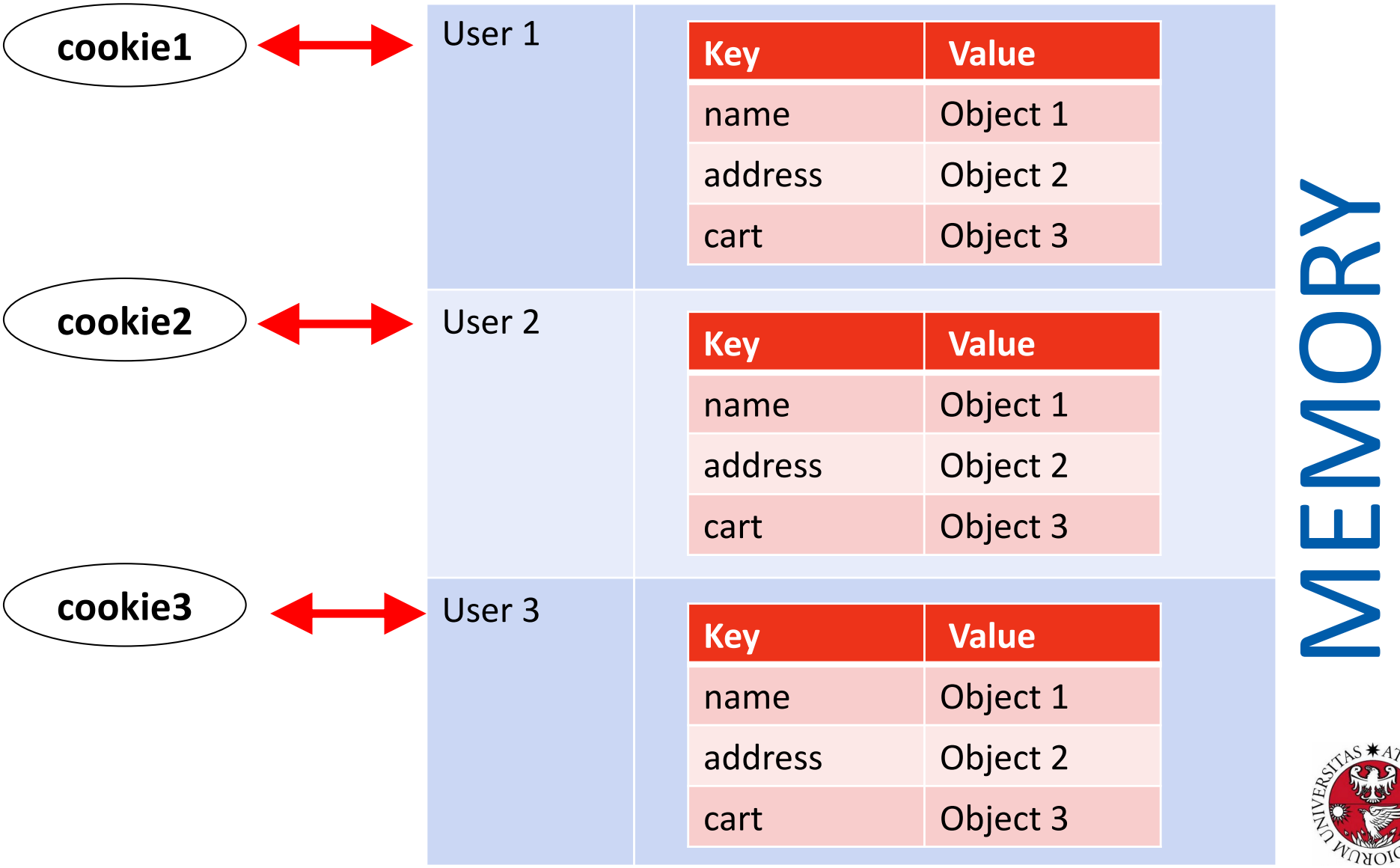
Session

Introduction

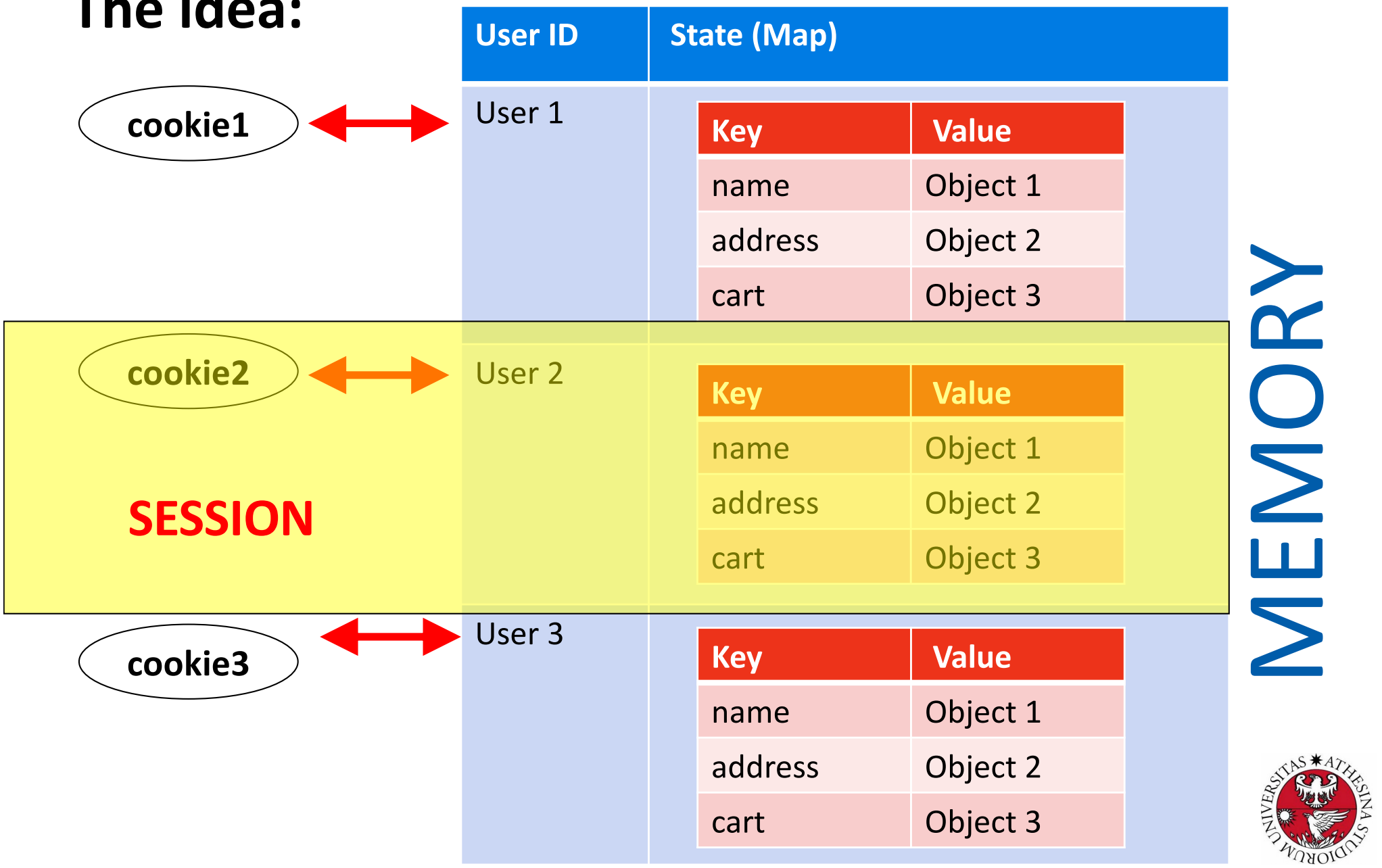
Session tracking using cookies

The idea:



Session tracking using cookies

The idea:



Session concept

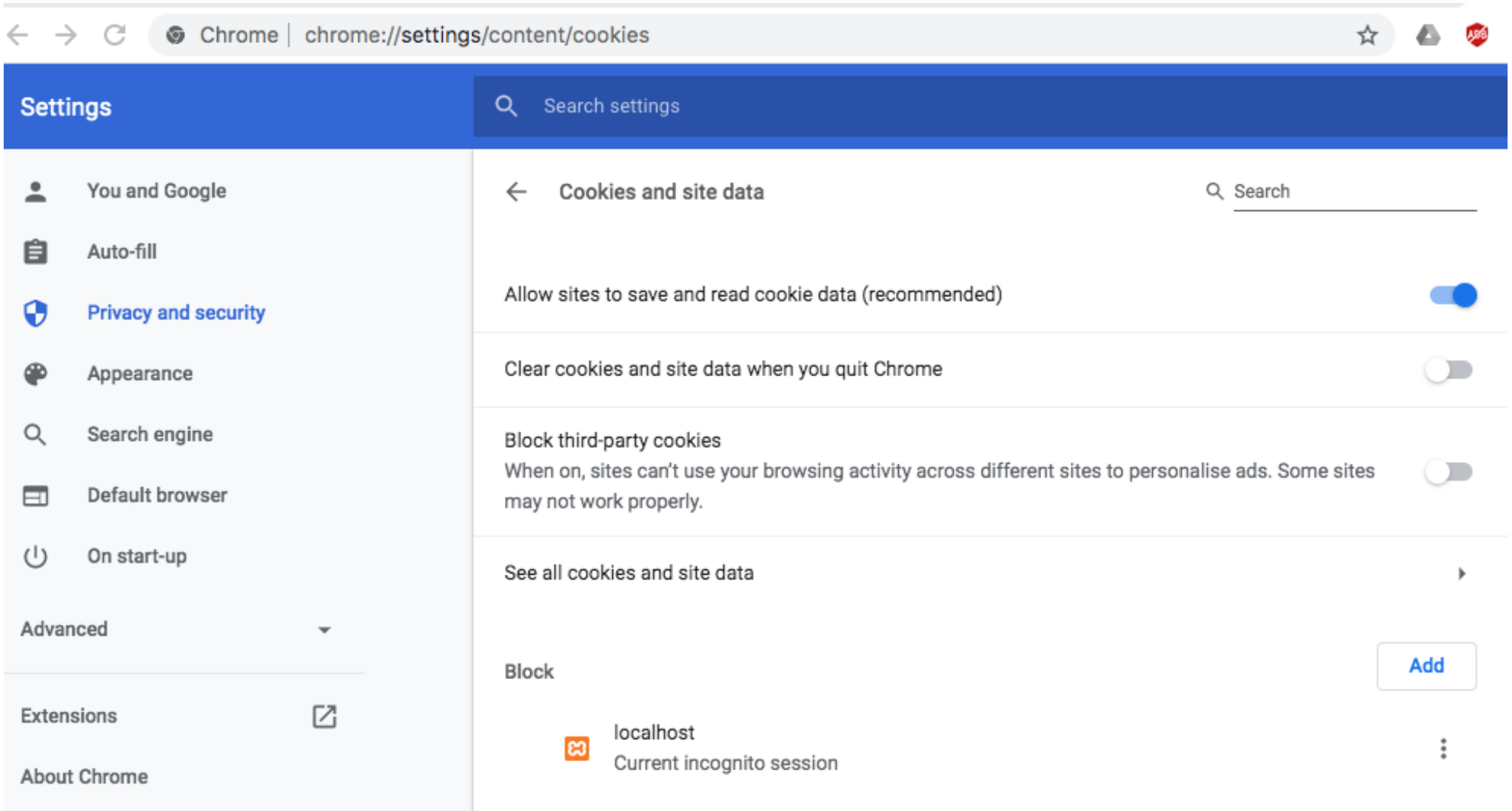
- To support applications that need to maintain state, Java Servlet technology provides an API for managing sessions and allows several mechanisms for implementing sessions.

- Sessions are represented by an HttpSession object.

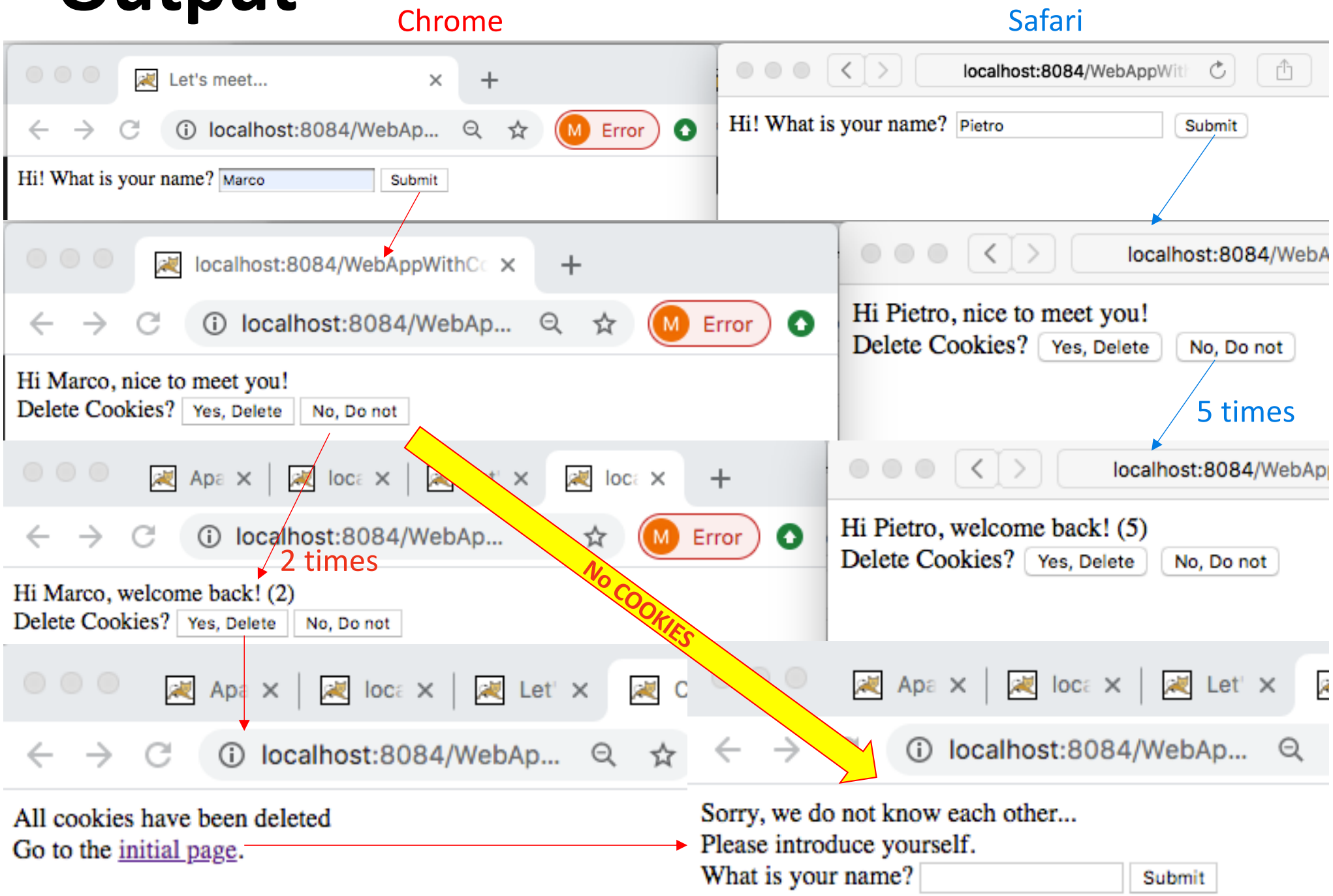
Session tracking

- To associate a session with a user, a web container can use several methods, all of which involve passing an identifier between the client and the server. The identifier can be
- maintained on the client as a cookie, or
- the web component can include the identifier in every URL that is returned to the client. See URL rewriting later)

Disallowing cookies



Output


Chrome
Safari
Let's meet...
localhost:8084/WebAp...
Error
Hi! What is your name? Marco
Submit
localhost:8084/WebAppWit
Hi! What is your name? Pietro
Submit
localhost:8084/WebAppWithC
Hi Marco, nice to meet you!
Delete Cookies? Yes, Delete
No, Do not
localhost:8084/WebA
Hi Pietro, nice to meet you!
Delete Cookies? Yes, Delete
No, Do not
5 times
2 times
No COOKIES
Hi Marco, welcome back! (2)
Delete Cookies? Yes, Delete
No, Do not
localhost:8084/WebAp
Hi Pietro, welcome back! (5)
Delete Cookies? Yes, Delete
No, Do not
All cookies have been deleted
Go to the initial page.
Sorry, we do not know each other...
Please introduce yourself.
What is your name?
Submit

How can I use sessions with servlets?

Accessing Session

- You access an HttpSession object by calling the getSession method of a request object.
- This method returns the current session associated with this request; or, if the request does not have a session, this method creates one.

Associating objects with Session

You can associate object-valued attributes with an HttpSession by name.

Such attributes are accessible by any web component that belongs to the same web context *and* is handling a request that is part of the same session.

HttpSession methods: attributes

- public Enumeration **getAttributeNames**()
  - Returns an Enumeration of String objects containing the names of all the objects bound to this session.
- public Object **getAttribute**(String name)
  - Returns the object bound with the specified name in this session, or null if no object is bound under the name.
- public void **setAttribute**(String name, Object value)
  - Binds an object to this session, using the name specified. If an object of the same name is already bound to the session, the object is replaced.
- public void **removeAttribute**(String name)
  - Removes the object bound with the specified name from this session. If the session does not have an object bound with the specified name, this method does nothing.

Session lifecycle

- A session consumes resources (memory), hence it has to be managed. Since http is stateless, there is no notion of “log out”.
- The way of solving the problem, is to decide an expiry time for sessions (timeout)

HttpSession methods: timing

- public long **getCreationTime**()
  - Returns the time when this session was created, measured in milliseconds since midnight January 1, 1970 GMT.
- 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, and marked by the time the container received the request.
- public void **setMaxInactiveInterval**(int interval)
  - Specifies the time, in seconds, between client requests before the servlet container will invalidate this session. A negative time indicates the session should never timeout.
- public int **getMaxInactiveInterval**()
  - Returns the maximum time interval, in seconds, that the servlet container will keep this session open between client accesses. After this interval, the servlet container will invalidate the session.

Setting Session global Timeout

To set the timeout period in the deployment descriptor using NetBeans IDE, follow these steps.

- Expand the node of your project in the **Projects** tab.
- Expand the **Web Pages** and **WEB-INF** nodes that are under the project node.
- If WebInf is empty, select it and right-click new->other->Standard Deployment Descriptor
- Double-click web.xml
- If not present, add

```
<session-config>
    <session-timeout>
        30
    </session-timeout>
</session-config>
```

(30 is the number of minuts after which the session will expire)

HttpSession methods: other

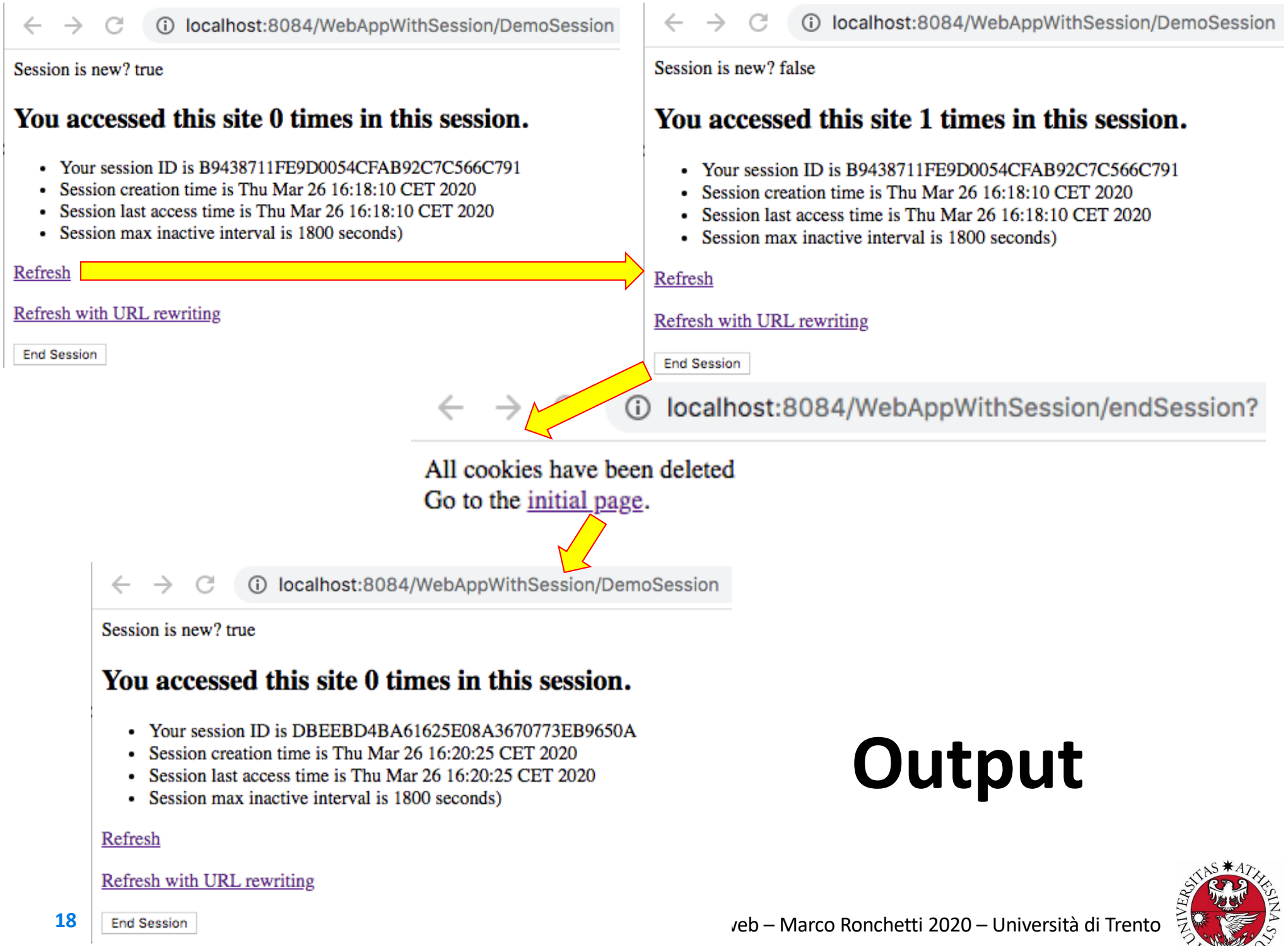
- public java.lang.String **getId**()
  - Returns a string containing the unique identifier assigned to this session. The identifier is assigned by the servlet container and is implementation dependent.
- public boolean **isNew**()
  - Returns true if the client does not yet know about the session or if the client chooses not to join the session (e.g., if client had disabled the use of cookies).
- public void **invalidate**()
  - Invalidates this session then unbinds any objects bound to it.

Session tracking without cookies

- If your application uses session objects, you must ensure that session tracking is enabled by having the application rewrite URLs whenever the client turns off cookies.

- You do this by calling the response's encodeURL(URL) method on all URLs returned by a servlet.
- This method includes the session ID in the URL only if cookies are disabled; otherwise, the method returns the URL unchanged.

Example: managing session, also without cookies


localhost:8084/WebAppWithSession/DemoSession
Session is new? true
You accessed this site 0 times in this session.
Your session ID is B9438711FE9D0054CFAB92C7C566C791
Session creation time is Thu Mar 26 16:18:10 CET 2020
Session last access time is Thu Mar 26 16:18:10 CET 2020
Session max inactive interval is 1800 seconds)
Refresh
Refresh with URL rewriting
End Session
localhost:8084/WebAppWithSession/DemoSession
Session is new? false
You accessed this site 1 times in this session.
Your session ID is B9438711FE9D0054CFAB92C7C566C791
Session creation time is Thu Mar 26 16:18:10 CET 2020
Session last access time is Thu Mar 26 16:18:10 CET 2020
Session max inactive interval is 1800 seconds)
Refresh
Refresh with URL rewriting
End Session
localhost:8084/WebAppWithSession/endSession?
All cookies have been deleted
Go to the initial page.
localhost:8084/WebAppWithSession/DemoSession
Session is new? true
You accessed this site 0 times in this session.
Your session ID is DBEEBD4BA61625E08A3670773EB9650A
Session creation time is Thu Mar 26 16:20:25 CET 2020
Session last access time is Thu Mar 26 16:20:25 CET 2020
Session max inactive interval is 1800 seconds)
Refresh
Refresh with URL rewriting
End Session

Output

Session in action - 1

```
package it.unitn.disi.ronchet.myservlets;

import …

 @WebServlet(urlPatterns = {"/DemoSession"})
public class DemoSession extends HttpServlet {

    PrintWriter out=null;
    private void p(String s) {
       out.println(s);
   }
```

Session in action - 2

```
@Override
   public void doGet(HttpServletRequest request,
                     HttpServletResponse response)
                     throws IOException, ServletException {
      out = response.getWriter();

      // Return the existing session if there is one.
      // Create a new session otherwise.
      HttpSession session = request.getSession();
      Integer accessCount;
      synchronized(session) {
         accessCount =
              (Integer)session.getAttribute("accessCount");
         if (accessCount == null) {
            accessCount = 0;   // autobox int to Integer
         } else {
            accessCount = new Integer(accessCount + 1);
         }
         session.setAttribute("accessCount", accessCount);
      }
```

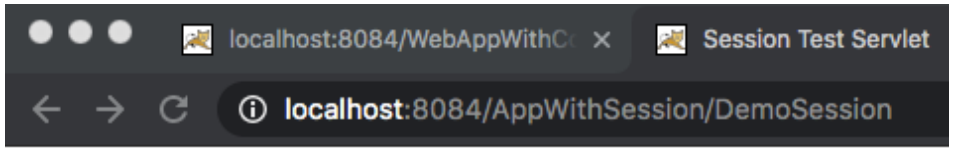

Session in action - 3

```
try {
  response.setContentType("text/html;charset=UTF-8");
  p("<!DOCTYPE html>"
      +"<html>"
      +"<head><title>Session Test Servlet</title></head><body>")
  p("Session is new? "+session.isNew());
  p("<h2>You accessed this site " + accessCount
      + " times in this session.</h2>");
  p("<ul><li>Your session ID is " + session.getId() + "</li>");
  p("<li>Session creation time is " +
      new Date(session.getCreationTime()) + "</li>");
  p("<li>Session last access time is " +
      new Date(session.getLastAccessedTime()) + "</li>");
  p("<li>Session max inactive interval  is " +
      session.getMaxInactiveInterval() + " seconds)</li></ul>");
```

Session in action - 4

```
   p("<p><a  href='" + request.getRequestURI()
      +  "'>Refresh</a>");
   p("<p><a  href='”
      + response.encodeURL(request.getRequestURI())
      + "'>Refresh with  URL rewriting</a>\n");
         p("<form method=\"GET\" action=\"endSession\">\n"
            +"<input type=\"submit\" value=\"End Session\">\n"
            +"</form>");
         p("</body></html>");
 } finally {
     out.close(); // Always close the output writer
 }
} // end DoGet
```

Without cookies…



You have access this site 0 times in this session.

(Session ID is ECB0CF247DD8C156A0EFE73FF9A3AA4A)

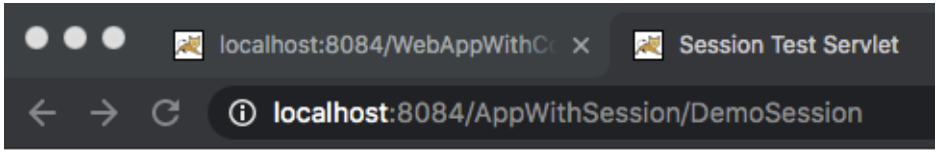
(Session creation time is Thu Mar 26 12:08:11 CET 2020)

(Session last access time is Thu Mar 26 12:08:11 CET 2020)

(Session max inactive interval is 1800 seconds)

Refresh

Refresh with URL rewriting



You have access this site 0 times in this session.

(Session ID is 0E13C8D5828616D80EA34BF2213CAB4C)

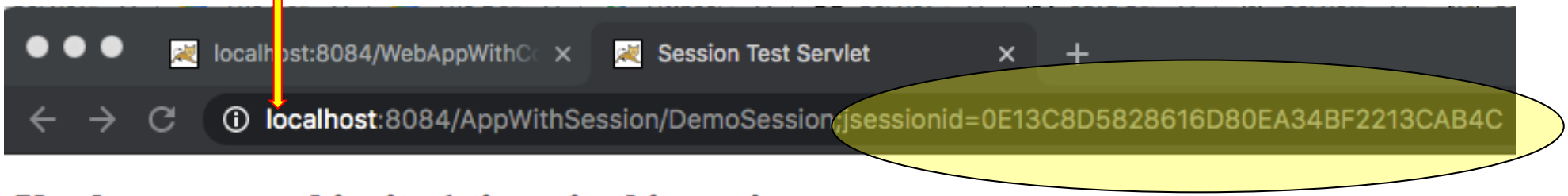
(Session creation time is Thu Mar 26 12:09:34 CET 2020)

(Session last access time is Thu Mar 26 12:09:34 CET 2020)

(Session max inactive interval is 1800 seconds)

Refresh

Refresh with URL rewriting



You have access this site 1 times in this session.

(Session ID is 0E13C8D5828616D80EA34BF2213CAB4C)

(Session creation time is Thu Mar 26 12:09:34 CET 2020)

(Session last access time is Thu Mar 26 12:09:34 CET 2020)

(Session max inactive interval is 1800 seconds)

Refresh

Refresh with URL rewriting

Session in action – 5 - endSession

```
package it.unitn.disi.ronchet.webProg;

import …

@WebServlet(name = "endSession", urlPatterns = {"/endSession"})
public class DeleteSession extends HttpServlet {

    protected void doGet(HttpServletRequest request,
            HttpServletResponse response)
            throws ServletException, IOException {
      HttpSession s= request.getSession();
      s.invalidate();
      response.setContentType("text/html;charset=UTF-8");
      request.getRequestDispatcher("SessionHasBeenDeleted.html")
             .include(request, response);

    }
}
```


Session in action – 6 - SessionHasBeenDeleted.html

```
<!DOCTYPE html>
<html>
    <head>
        <title>Session has been deleted</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width,
            initial-scale=1.0">
    </head>
    <body>
        All cookies have been deleted <br>
        Go to the <a href="/WebAppWithSession/DemoSession">
            initial page</a>.
    </body>
</html>
```

Session in action – 7 – web.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
             http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd"
         version="3.1">
    <welcome-file-list>
        <welcome-file>DemoSession</welcome-file>
    </welcome-file-list>
    <session-config>
        <session-timeout>
            30
        </session-timeout>
    </session-config>
</web-app>
```

How can I use sessions with JSPs?

Predefined Objects

out	Writer
request	HttpServletRequest
response	HttpServletResponse
session	HttpSession
page	this in the Servlet
application	servlet.getServletContext
	area shared among all servlets within the same webapp
config	ServletConfig
exception	only in a errorPage
pageContext	

JSP in action – Example part 1

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@page import="java.util.Date"%>
<%@page language="java"  session="true" %>

<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type"
              content="text/html; charset=UTF-8">
        <title>Session Test JSP</title>
    </head>
    <body>
      <%!  Integer accessCount; %>
      <%
        accessCount=(Integer)session.getAttribute("accessCount");
         if (accessCount == null) {
            accessCount = 0;   // autobox int to Integer
         } else {
            accessCount = new Integer(accessCount + 1);
         }
         session.setAttribute("accessCount", accessCount);
      %>
```

JSP in action – Example part 2

```
Session is new? <% out.println(session.isNew()); %>
      <h2>You accessed this site " <%= accessCount %>
            times in this session.</h2>
      <ul><li>Your session ID is " <%= session.getId() %></li>
          <li>Session creation time is
               <%= new Date(session.getCreationTime()) %> </li>
          <li>Session last access time is  <%=
               new Date(session.getLastAccessedTime()) %> </li>
          <li>Session max inactive interval  is <%=
               session.getMaxInactiveInterval() %> seconds</li>
       </ul>

       <p><a  href='<%= request.getRequestURI() %>'>Refresh</a>
       <p><a  href='
            <%= response.encodeURL(request.getRequestURI())%>'>
               Refresh with  URL rewriting</a>
       <form method="GET" action="endSession.jsp">
           <input type="submit" value="End Session">
       </form>
    </body>
</html>
```

Are session objects thread safe?

Sessions and thread safety

- A *session* belongs to a user.
- Hence, when different users activates the same servlet, and this requests a session object, it gets a different object for every user – so no problems with multithreading.

BUT

- if a user opens two windows on the same browser, and accesses the same servlet, then we DO have a thread safety issue!
- that's very unlikely, but yet…

Q

How can we share information among the components of a WebApp?

VERY IMPORTANT!

Sharing information

- Within a request:

multiple servlets cooperating through *forward* and *include* mechanisms:

- pass request and response, add information to the **request** (request in jsp)

- Among different requests by the same user:

- use the **Session** object

getSession() in servlets

session in jsp

Sharing information

- Among different invocations of the same servlet:

- use instance variables (or static variables)

- Among different servlets/jsps of the same WebApp:

- use ServletContext
  - servlets: getServletConfig().getServletContext
  - jsps: application (a special, predefined object)

when sharing info, always think about thread-safety!

Can I monitor session lifecycle?

Advanced: associating events with session objects

- Your application can notify web context and session listener objects of servlet lifecycle events (Handling Servlet Lifecycle Events). You can also notify objects of certain events related to their association with a session, such as the following:
  - When the object is added to or removed from a session. To receive this notification, your object must implement the javax.servlet.http.HttpSessionBindingListener interface.
  - When the session to which the object is attached will be passivated or activated. A session will be passivated or activated when it is moved between virtual machines or saved to and restored from persistent storage. To receive this notification, your object must implement the javax.servlet.http.HttpSessionActivationListener interface.

Monitoring Servlets Lifecycle

Web context	Initialization and Destruction	ServletContextListener	ServletContextEvent
	Attribute added, removed, or replaced	ServletContextAttributeListener	ServletContextAttributeEvent
Session	Creation, invalidation, activation, passivation, and timeout	HttpSessionListenerHttpSessionActivationListener	HttpSessionEvent
	Attribute added, removed, or replaced	HttpSessionAttributeListener	HttpSessionBindingEvent
Request	A servlet request has started being processed by Web components	ServletRequestListener	ServletRequestEvent
	Attribute added, removed, or replaced	ServletRequestAttributeListener	ServletRequestAttributeEvent

Monitoring Servlets Lifecycle - Example

```
/* File : ApplicationWatch.java */
import javax.servlet.ServletContextListener;
import javax.servlet.ServletContextEvent;
public class ApplicationWatch implements ServletContextListener {
public static long applicationInitialized = 0L;
/* Application Startup Event */
public void contextInitialized(ServletContextEvent ce) { applicationInitialized =
System.currentTimeMillis(); }
/* Application Shutdown Event */
public void contextDestroyed(ServletContextEvent ce) {}
}
```

Monitoring Servlets Lifecycle - Example

```
/* File : SessionCounter.java */
import javax.servlet.http.HttpSessionListener;
import javax.servlet.http.HttpSessionEvent;
public class SessionCounter implements HttpSessionListener {
private static int activeSessions = 0;
/* Session Creation Event */
public void sessionCreated(HttpSessionEvent se) { activeSessions++; }
/* Session Invalidation Event */
public void sessionDestroyed(HttpSessionEvent se) { if(activeSessions > 0)
activeSessions--; }
public static int getActiveSessions() { return activeSessions; }
}
```

Monitoring Servlets Lifecycle - Example

- <!-- Web.xml -->
- <?xml version="1.0" encoding="ISO-8859-1"?>
- <!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN" "http://java.sun.com/j2ee/dtds/web-app_2.3.dtd">
- <web-app>
- <!-- Listeners -->
- <listener>
- <listener-class> com.stardeveloper.web.listener.SessionCounter </listener-class>
- </listener>
- <listener>
- <listener-class> com.stardeveloper.web.listener.ApplicationWatch </listener-class>
- </listener>
- </web-app>