

# Getting familiar with Tomcat



# Apache Tomcat

<https://tomcat.apache.org/whichversion.html>

Apache Tomcat Versions

Servlet Spec	JSP Spec	EL Spec	WebSocket Spec	Authentication (JASIC) Spec	Apache Tomcat Version	Latest Released Version	Supported Java Versions
6.0	3.1	5.0	TBD	TBD	10.1.x	10.1.0-M5 (alpha)	11 and later
5.0	3.0	4.0	2.0	2.0	10.0.x	10.0.11	8 and later
4.0	2.3	3.0	1.1	1.1	9.0.x	9.0.53	8 and later
3.1	2.3	3.0	1.1	1.1	8.5.x	8.5.71	7 and later
3.1	2.3	3.0	1.1	N/A	8.0.x (superseded)	8.0.53 (superseded)	7 and later
3.0	2.2	2.2	1.1	N/A	7.0.x (archived)	7.0.109 (archived)	6 and later (7 and later for WebSocket)
2.5	2.1	2.1	N/A	N/A	6.0.x (archived)	6.0.53 (archived)	5 and later
2.4	2.0	N/A	N/A	N/A	5.5.x (archived)	5.5.36 (archived)	1.4 and later
2.3	1.2	N/A	N/A	N/A	4.1.x (archived)	4.1.40 (archived)	1.3 and later
2.2	1.1	N/A	N/A	N/A	3.3.x (archived)	3.3.2 (archived)	1.1 and later

**SELECT VERSION 9 (Why?)**



# if necessary...

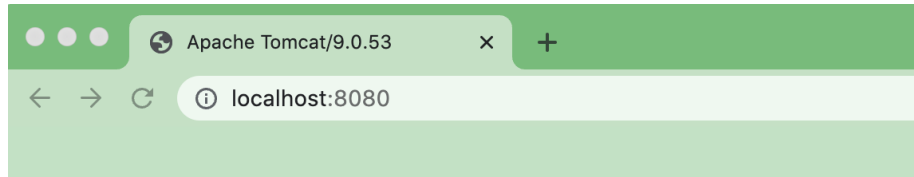
- 1) cd into your tomcat bin directory
- 2) make all .sh files executable

```
MR-MacBookPro:~ ronchet$ cd /Users/ronchet/Download/apache-tomcat-9.0.53/bin
MR-MacBookPro:bin ronchet$ ls -la
total 1760
drwxr-xr-x@ 29 ronchet  staff    928 Sep  6 23:09 .
drwxr-xr-x@ 17 ronchet  staff    544 Sep 16 10:36 ..
-rw-r--r--@  1 ronchet  staff  34705 Sep  6 23:09 bootstrap.jar
-rw-r--r--@  1 ronchet  staff   1703 Sep  6 23:09 catalina-tasks.xml
-rw-r--r--@  1 ronchet  staff  16840 Sep  6 23:09 catalina.bat
-rw-r--r--@  1 ronchet  staff 25294 Sep  6 23:09 catalina.sh
-...
-rw-r--r--@  1 ronchet  staff   2026 Sep  6 23:09 version.bat
-rw-r--r--@  1 ronchet  staff   1908 Sep  6 23:09 version.sh
MR-MacBookPro:bin ronchet$ chmod a+x *.sh
MR-MacBookPro:bin ronchet$
```




# start Tomcat from command line:

- 1) make sure you are NOT using Tomcat from inside IntelliJ
- 2) cd into your tomcat bin directory
- 3) execute startup.sh or startup.bat
- 4) go to the browser



## Apache Tomcat/9.0.53

If you're seeing this, you've successfully installed Tomcat. Co



**Recommended Reading:**

- [Security Considerations How-To](#)
- [Manager Application How-To](#)
- [Clustering/Session Replication How-To](#)

**Developer Quick Start**

- [Tomcat Setup](#)
- [Realms & AAA](#)
- [Examples](#)
- [First Web Application](#)
- [JDBC DataSources](#)

**Managing Tomcat**

For security, access to the [manager webapp](#) is restricted. Users are defined in:

```
$CATALINA_HOME/conf/tomcat-users.xml
```

In Tomcat 9.0 access to the manager

**Documentation**

- [Tomcat 9.0 Documentation](#)
- [Tomcat 9.0 Configuration](#)
- [Tomcat Wiki](#)

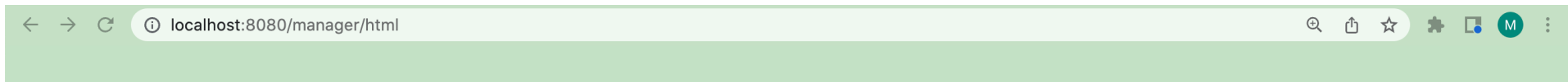
Find additional important configuration

# Web App Manager

## Managing Tomcat

For security, access to the manager webapp is restricted. Users are defined in:

```
$CATALINA_HOME/conf/tomcat-users.xml
```



### Tomcat Web Application Manager

Message: OK

<b>Manager</b>			
<a href="#">List Applications</a>	<a href="#">HTML Manager Help</a>	<a href="#">Manager Help</a>	<a href="#">Server Status</a>

<b>Applications</b>					
Path	Version	Display Name	Running	Sessions	Commands
/	None specified	Welcome to Tomcat	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/MyFirstServletProject	None specified		true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/demo1_war_explored	None specified		true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/docs	None specified	Tomcat Documentation	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/examples	None specified	Servlet and JSP Examples	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/host-manager	None specified	Tomcat Host Manager Application	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes



# Create a user

- Make sure IntelliJ is quit.
- edit file [TOMCAT-HOME]/conf/tomcat-users.xml
- towards the end, add a line like

```
49 <!--
50 <role rolename="tomcat"/>
51 <role rolename="role1"/>
52 <user username="tomcat" password="<must-be-changed>" roles="tomcat"/>
53 <user username="both" password="<must-be-changed>" roles="tomcat,role1"/>
54 <user username="role1" password="<must-be-changed>" roles="role1"/>
55 -->
56 <user username="ronchet" password="secret" roles="standard,manager-script,manager-gui" />
57 </tomcat-users>
58
```

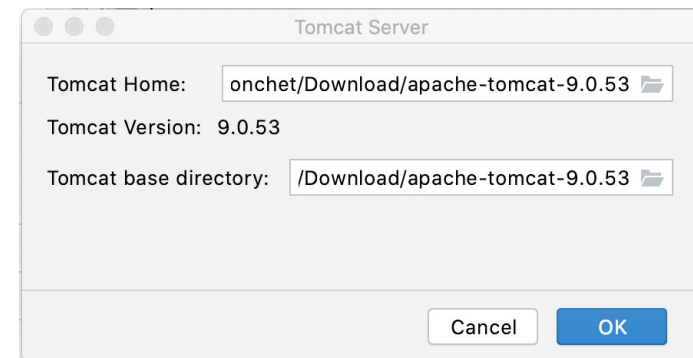
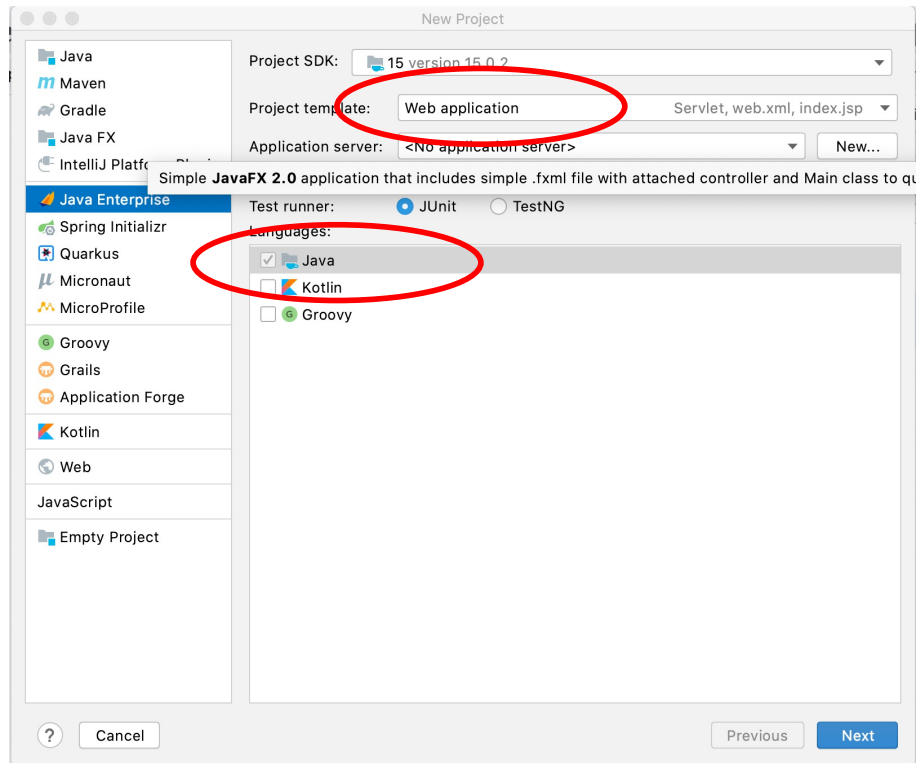
- start tomcat:
- cd file [TOMCAT-HOME]/bin
- ./startup.sh

```
MR-MacBookPro:bin ronchet$ ./startup.sh
Using CATALINA_BASE:   /Users/ronchet/Download/apache-tomcat-9.0.53
Using CATALINA_HOME:   /Users/ronchet/Download/apache-tomcat-9.0.53
Using CATALINA_TMPDIR: /Users/ronchet/Download/apache-tomcat-9.0.53/temp
Using JRE_HOME:        /Users/ronchet/Library/Java/JavaVirtualMachines/adopt-ope
njdk-15.0.2/Contents/Home
Using CLASSPATH:       /Users/ronchet/Download/apache-tomcat-9.0.53/bin/bootstra
p.jar:/Users/ronchet/Download/apache-tomcat-9.0.53/bin/tomcat-juli.jar
Using CATALINA_OPTS:
Tomcat started.
MR-MacBookPro:bin ronchet$
```

# Now let's create a new project with IntelliJ

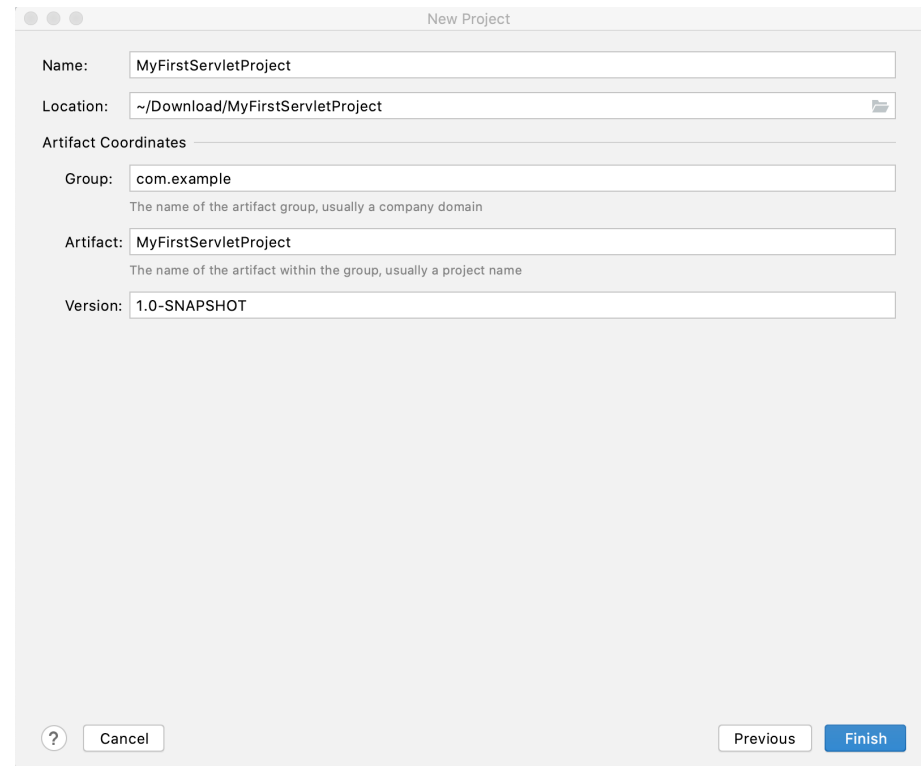
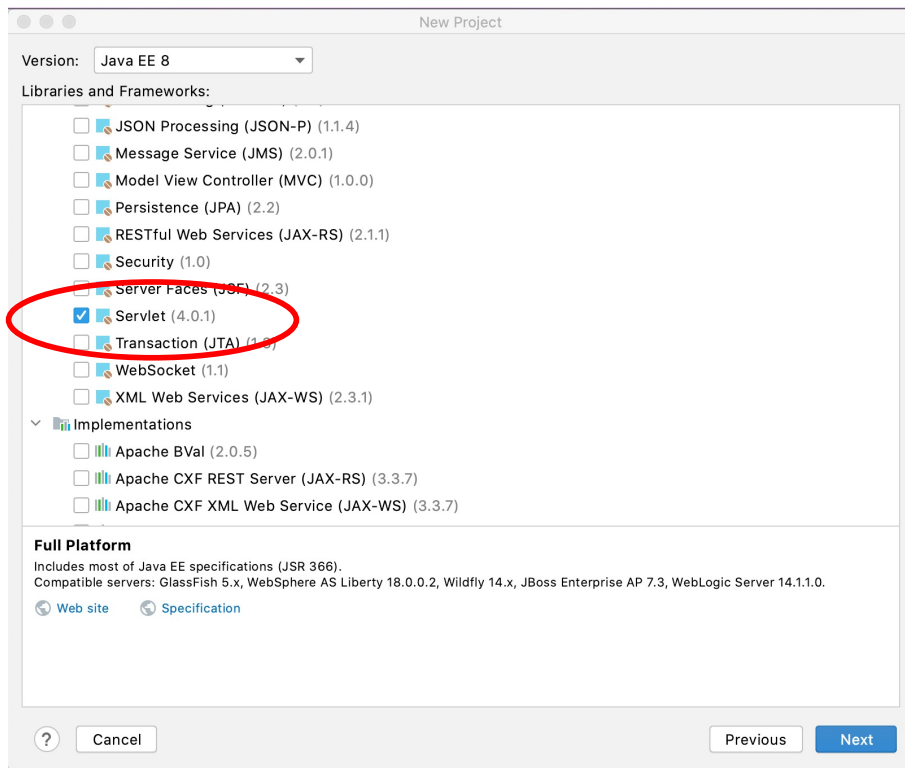


# Creating new Project - 1





# Creating new project -2



# Creating new project - 3

The screenshot displays an IDE interface for a web application project named 'demo2'. The project structure is visible in the left sidebar, showing a hierarchy of folders: 'demo2' (root), '.idea', 'src', 'main', 'java', 'com.example.demo2', 'resources', 'webapp', 'WEB-INF', and 'index.jsp'. The 'HelloServlet' class is highlighted in the 'com.example.demo2' package. The 'index.jsp' file is also highlighted in the 'WEB-INF' directory. The main editor area shows the content of 'index.jsp', which is a simple HTML page with a title 'JSP - Hello World' and a body containing 'Hello World!' and a link to 'Hello Servlet'. The code is as follows:

```
1 <%@ page contentType="text/html; charset=UTF-8" %>
2 <!DOCTYPE html>
3 <html>
4 <head>
5     <title>JSP - Hello World</title>
6 </head>
7 <body>
8     <h1><%= "Hello World!" %>
9 </h1>
10    <br/>
11    <a href="hello-servlet">Hello Servlet</a>
12 </body>
13 </html>
```

The Services panel at the bottom shows a Tomcat Server configuration. The server is currently 'Not Started'. Under the 'Tomcat 9.0.53 [local]' instance, the 'demo2:war exploded' deployment is visible.



# Index.jsp

For now, let's consider it as a simple HTML page

```
m pom.xml (MyFirstServletProject) x HelloServlet.java x JSP index.jsp x
1 <%@ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" %>
2 <!DOCTYPE html>
3 <html>
4 <head>
5     <title>JSP - Hello World</title>
6 </head>
7 <body>
8 <h1><%= "Hello World!" %>
9 </h1>
10 <br/>
11 <a href="hello-servlet">Hello Servlet</a>
12 </body>
13 </html>
```

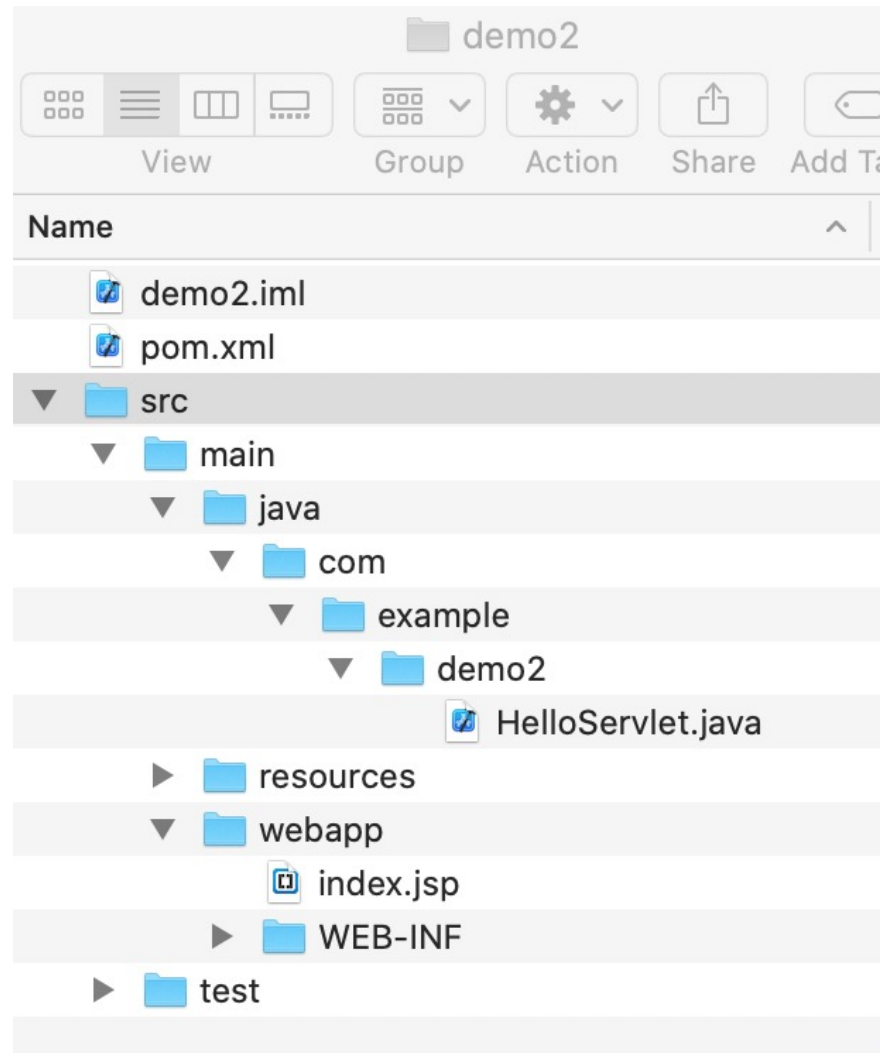


# HelloServlet.java

```
1 package com.example.MyFirstServletProject;
2
3 import ...
4
5
6
7 @WebServlet(name = "helloServlet", value = "/hello-servlet")
8 public class HelloServlet extends HttpServlet {
9     private String message;
10
11     public void init() { message = "Hello World!"; }
12
13
14
15     public void doGet(HttpServletRequest request, HttpServletResponse response) throws IOException {
16         response.setContentType("text/html");
17
18         // Hello
19         PrintWriter out = response.getWriter();
20         out.println("<html><body>");
21         out.println("<h1>" + message + "</h1>");
22         out.println("</body></html>");
23     }
24
25     public void destroy() {
26     }
27 }
```



# Development directory



# Servlet Deployment

- By default, a servlet application APP is located at the path

`<Tomcat-installationdirectory>/webapps/APP`

and the class file would reside in

`<Tomcat-installationdirectory>/webapps/APP/WEB-INF/classes.`

If you have a fully qualified class name of **com.myorg.MyServlet**, then this servlet class must be located in `WEB-INF/classes/com/myorg/MyServlet.class`.



# Servlet Deployment

MyFirstServletProject

View Group Action Share Add Tags

Name

- MyFirstServletProject.iml
- pom.xml
- src
- target
  - classes
  - generated-sources
  - MyFirstServletProject-1.0-SNAPSHOT**
    - index.jsp
    - META-INF
      - MANIFEST.MF
    - WEB-INF
      - classes
        - com
          - example
            - MyFirstServletProject
              - HelloServlet.class
    - web.xml

you can change this name

# Servlet Deployment

Run/Debug Configurations

Name: Tomcat 9.0.53  Store as project file ⚙

Server Deployment Logs Code Coverage Startup/Connection

Deploy at the server startup

demo2:war exploded

**you can change this name**

Application context: /demo2\_war\_exploded

Before launch

- Build
- Build 'demo2:war exploded' artifact

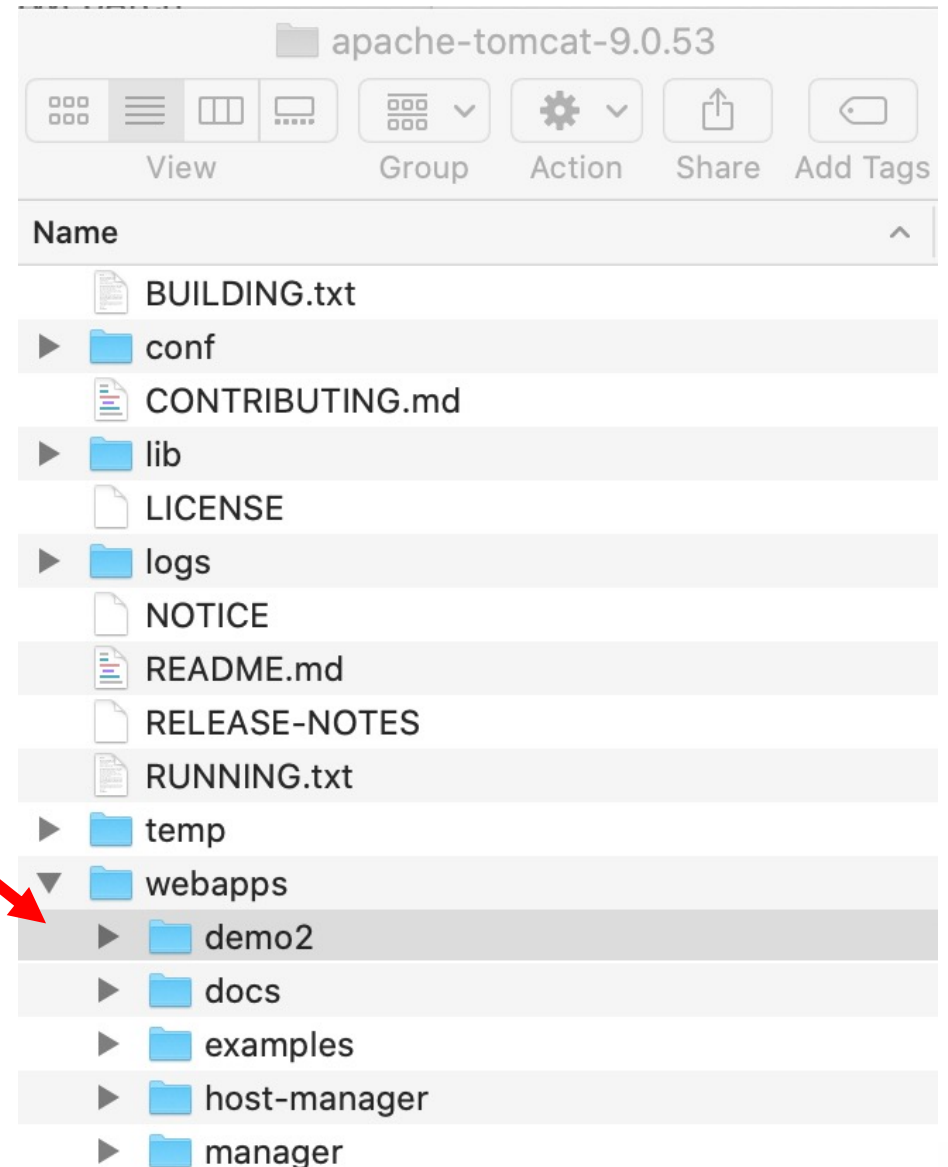
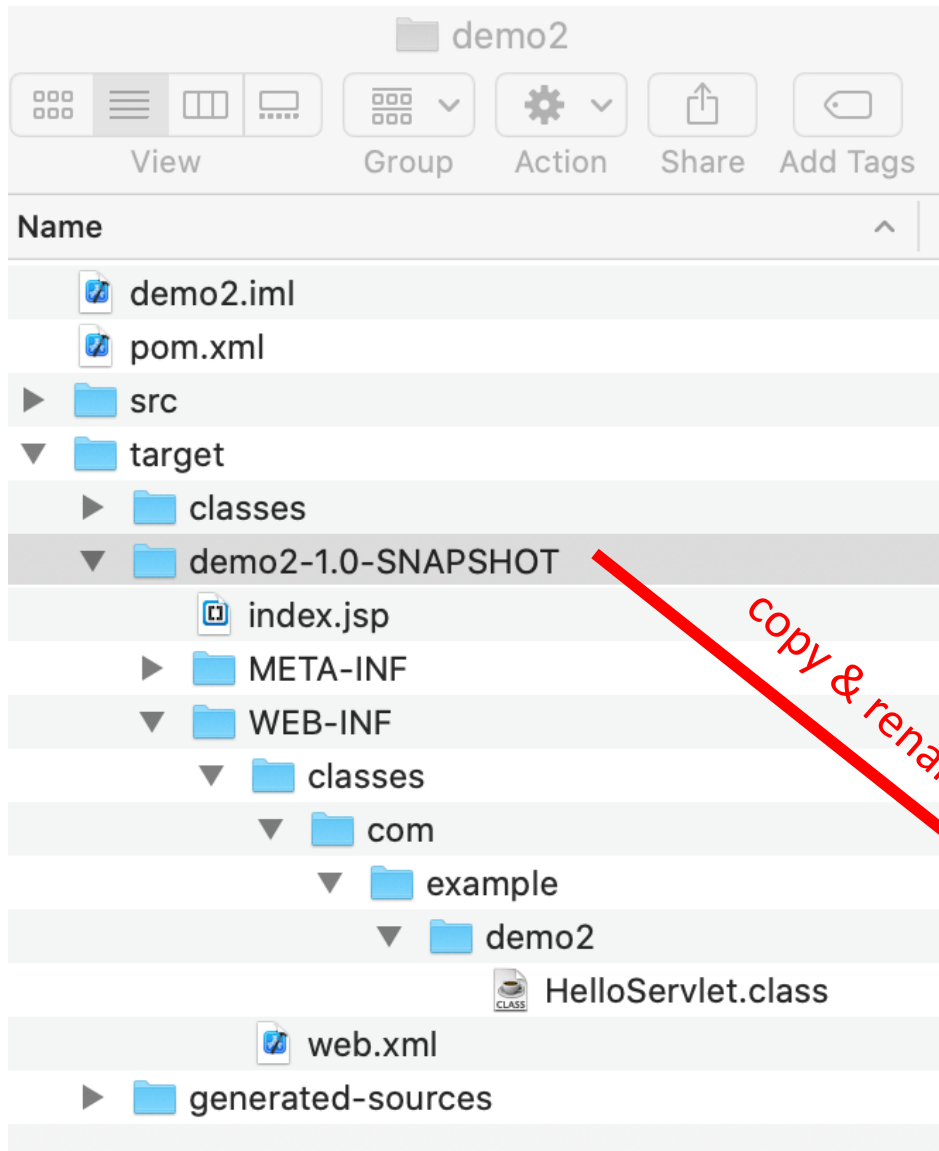
44

Cancel Apply OK





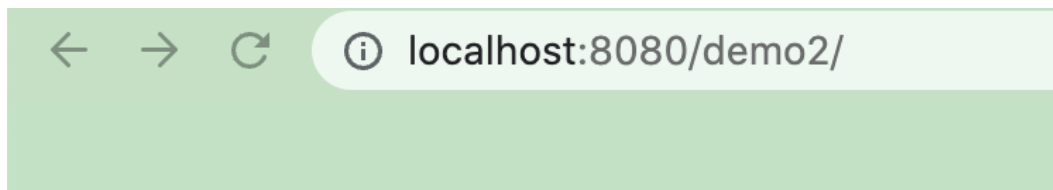
# Manual deployment



copy & rename

# Restart Tomcat...

```
MR-MacBookPro:bin ronchet$ ./startup.sh ]
Using CATALINA_BASE:   /Users/ronchet/Download/apache-tomcat-9.0.53
Using CATALINA_HOME:   /Users/ronchet/Download/apache-tomcat-9.0.53
Using CATALINA_TMPDIR: /Users/ronchet/Download/apache-tomcat-9.0.53/temp
Using JRE_HOME:        /Users/ronchet/.jenv/versions/11.0.10
Using CLASSPATH:       /Users/ronchet/Download/apache-tomcat-9.0.53/bin/bootstrap.jar:/Users/ronchet/Download/apache-tomcat-9.0.53/bin/tomcat-juli.jar
Using CATALINA_OPTS:
Tomcat started.
MR-MacBookPro:bin ronchet$ □
```





## Hello World!

[Hello Servlet](#)

# Our app is deployed!

localhost:8080/manager/html



## Tomcat Web Application Manager

Message: OK

**Manager**

<a href="#">List Applications</a>	<a href="#">HTML Manager Help</a>	<a href="#">Manager Help</a>	<a href="#">Sessions</a>
-----------------------------------	-----------------------------------	------------------------------	--------------------------

**Applications**

Path	Version	Display Name	Running	Sessions	Commands
/	None specified	Welcome to Tomcat	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/MyFirstServletProject	None specified		true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/demo2	None specified		true	1	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/docs	None specified	Tomcat Documentation	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/examples	None specified	Servlet and JSP Examples	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes



# Q

**Back to our servlet.**

**How do we factor out some code ?**

# Factoring out some HTML

- How can we avoid this horrible stuff?

```
out.println("<!DOCTYPE html>");
out.println("<html>");
out.println("<head>");
out.println("<title>Servlet ReadPost</title>");
out.println("</head>");
out.println("<body>");
out.println("<h1> fname="+name+"</h1>");
out.println("</body>");
out.println("</html>");
```



# Factoring out some HTML

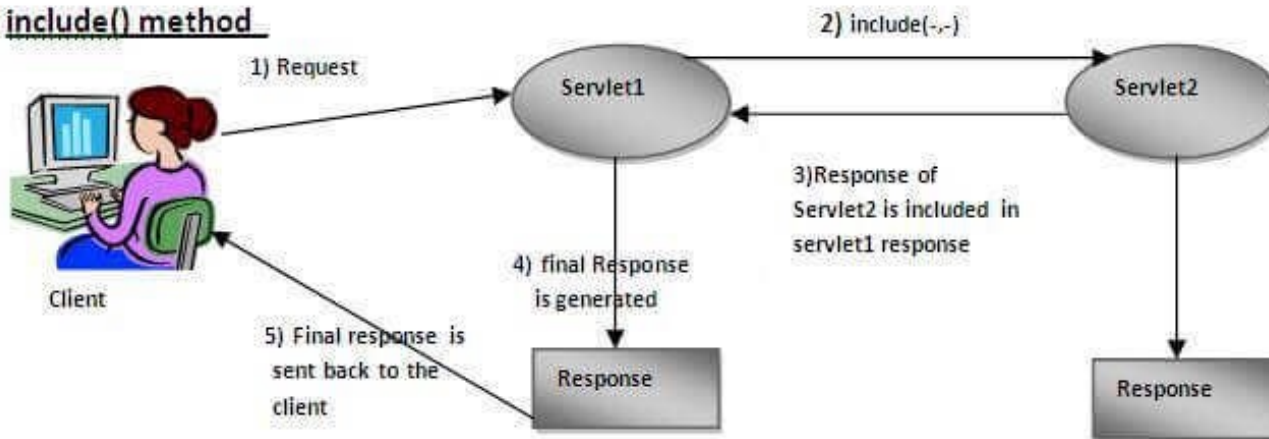
```
....xm  ReadPost.java x Demo1.java x fragment2.html x fragment1.html x Counter.java x readPost.html x
1      <!DOCTYPE html>
2      <html>
3      <head>
4          <title>TODO supply a title</title>
5          <meta charset="UTF-8">
6          <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      </head>
8      <body>
9          <h2>fragment 1</h2>
```

```
....xm  ReadPost.java x Demo1.java x fragment2.html x
1      <h2>fragment 2</h2>
2      </body>
3      </html>
4
```

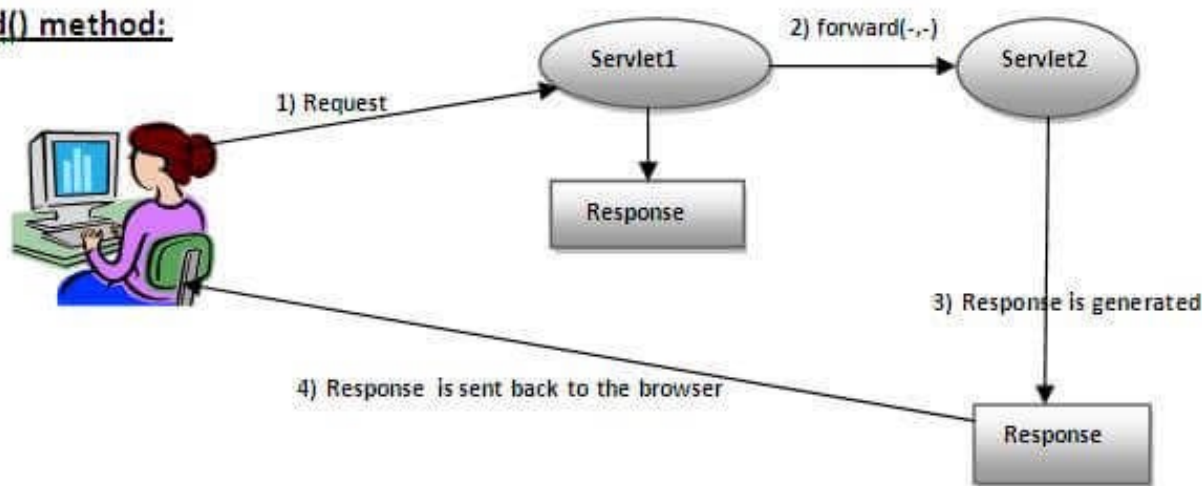


# Include vs forward

## include() method



## forward() method:



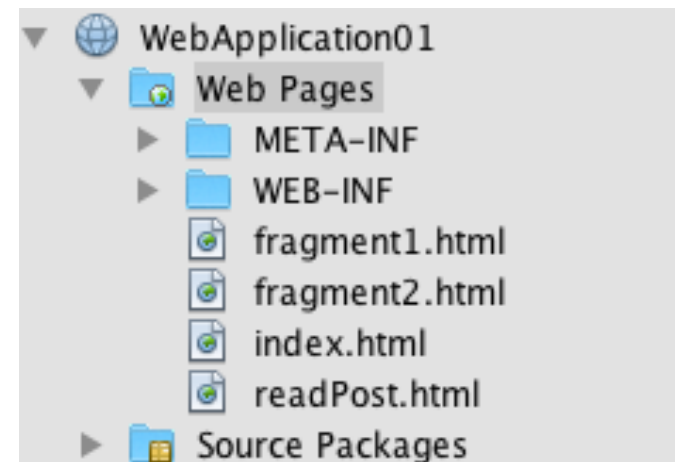
images from  
[/www.javatpoint.com](http://www.javatpoint.com)

example: see <https://www.javatpoint.com/requestdispatcher-in-servlet>



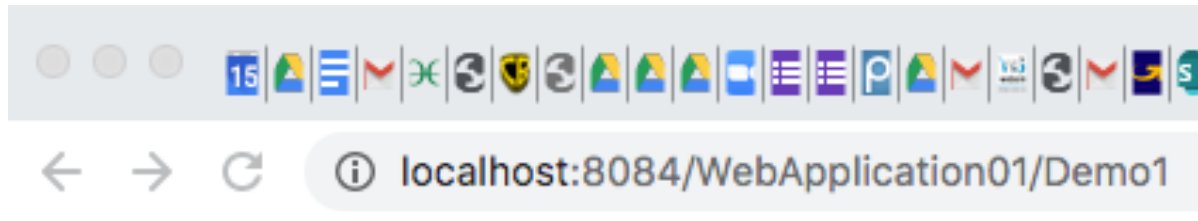
## @Override

```
protected void doGet(HttpServletRequest request,
HttpServletRequest response) throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    try (PrintWriter out = response.getWriter()) {
        request.getRequestDispatcher("/fragment1.html")
            .include(request, response);
        out.println("Servlet generated content");
        request.getRequestDispatcher("/fragment2.html")
            .include(request, response);
    }
}
```





# output



## fragment 1

Servlet generated content

## fragment 2

Q

**How do we monitor the execution?**

```
public void log(String msg) {  
    getServletContext().log(msg);  
}
```

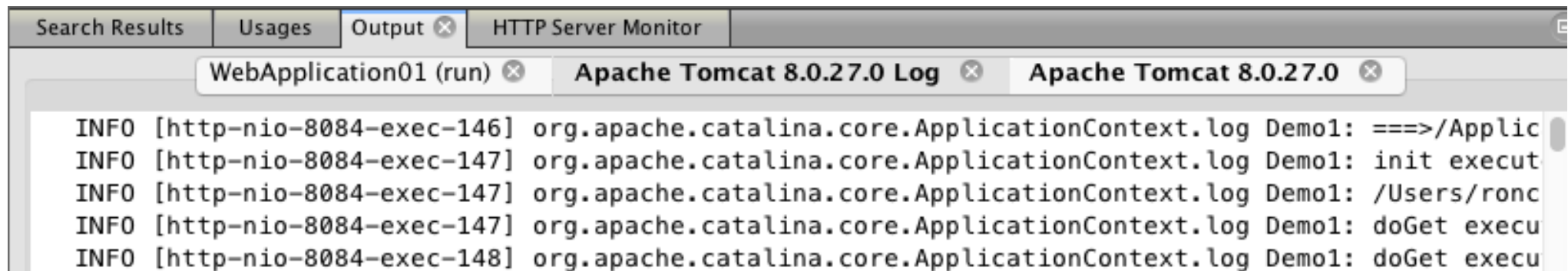
@Override

```
public void destroy() {  
    log("destroy executed");  
}
```

@Override

```
public void init() {  
    log("init executed");  
}
```

# Logging



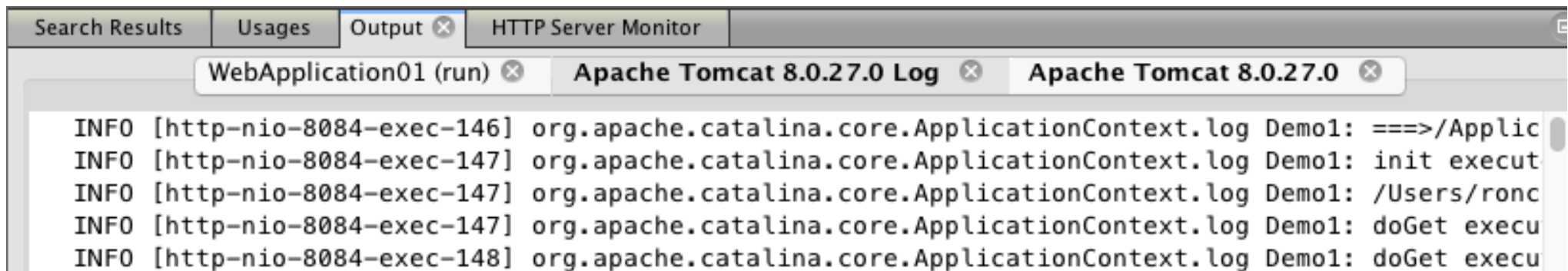
The screenshot shows an IDE window with several tabs: 'Search Results', 'Usages', 'Output', and 'HTTP Server Monitor'. The 'Output' tab is active, displaying the 'Apache Tomcat 8.0.27.0 Log'. The log contains five entries, all at the INFO level, showing the execution of the log method in the Demo1 application context.

```
INFO [http-nio-8084-exec-146] org.apache.catalina.core.ApplicationContext.log Demo1: ==>/Applic  
INFO [http-nio-8084-exec-147] org.apache.catalina.core.ApplicationContext.log Demo1: init execut  
INFO [http-nio-8084-exec-147] org.apache.catalina.core.ApplicationContext.log Demo1: /Users/ronc  
INFO [http-nio-8084-exec-147] org.apache.catalina.core.ApplicationContext.log Demo1: doGet execu  
INFO [http-nio-8084-exec-148] org.apache.catalina.core.ApplicationContext.log Demo1: doGet execu
```



```
public void log(String msg) {  
    filterConfig.getServletContext().log(msg);  
}
```

## Logging in filters



The screenshot shows an IDE window with several tabs: 'Search Results', 'Usages', 'Output', and 'HTTP Server Monitor'. The 'Output' tab is active, displaying the following log messages:

```
WebApplication01 (run) x Apache Tomcat 8.0.27.0 Log x Apache Tomcat 8.0.27.0 x  
INFO [http-nio-8084-exec-146] org.apache.catalina.core.ApplicationContext.log Demo1: ==>/Applic  
INFO [http-nio-8084-exec-147] org.apache.catalina.core.ApplicationContext.log Demo1: init execut  
INFO [http-nio-8084-exec-147] org.apache.catalina.core.ApplicationContext.log Demo1: /Users/ronc  
INFO [http-nio-8084-exec-147] org.apache.catalina.core.ApplicationContext.log Demo1: doGet execu  
INFO [http-nio-8084-exec-148] org.apache.catalina.core.ApplicationContext.log Demo1: doGet execu
```



# Getting deeper with Servlets

Let's build a hit counter



# Let us build a hit counter - 1

```
public class Counter {
    int count = 0;
    Calendar timeStamp = Calendar.getInstance();
    public void increase(){
        count++;
        timeStamp = Calendar.getInstance();
    }
    @Override
    public String toString() {
        StringBuffer s = null;
        if (count == 0)
            s = new StringBuffer("<p>no hits yet</p>");
        else {
            s = new StringBuffer("<p>hits = ");
            s.append(count)
                .append("<br>last hit on ")
                .append(timeStamp.getTime().toString());
        }
        return s.toString();
    }
}
```



# Let us build a hit counter - 2

```
@WebServlet(name = "Demo1", urlPatterns = {"/Demo1"})
public class Demo1 extends HttpServlet {
    Counter counter=new Counter();
    @Override
    protected void doGet(HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {
            request.getRequestDispatcher("/fragment1.html")
                .include(request, response);
            counter.increase();
            out.println(counter);
            request.getRequestDispatcher("/fragment2.html")
                .include(request, response);
        }
    }
}
```



# Output

The image displays three sequential browser screenshots of a web application running on localhost:8084/WebApplication01/Demo1. Each screenshot shows the browser's address bar and the page content. The page content consists of two bolded text elements: 'fragment 1' and 'fragment 2', followed by a hit counter and a timestamp. The hit counter starts at 1 in the first screenshot, increases to 2 in the second, and reaches 3 in the third. The timestamps are 'last hit on Sun Mar 15 14:41:01 CET 2020', 'last hit on Sun Mar 15 14:41:24 CET 2020', and 'last hit on Sun Mar 15 14:41:43 CET 2020' respectively.

**fragment 1**  
hits = 1  
last hit on Sun Mar 15 14:41:01 CET 2020

**fragment 2**

**fragment 1**  
hits = 2  
last hit on Sun Mar 15 14:41:24 CET 2020

**fragment 2**

**fragment 1**  
hits = 3  
last hit on Sun Mar 15 14:41:43 CET 2020

**fragment 2**

But if we restart the server, counter restarts from 1!





# How can we persist the counter?

In a file (named `counterData`)!

- 1) In `init`, let us check if file exists. If yes, let us resume the counter, else, let us create a new one.
- 2) In `destroy`, let us save counter in `conterData`.



# Java serialization

```
class A implements Serializable {...}
```

```
A a1=new A();
```

```
A a2;
```

```
...
```

```
File myFile = new File(filePath);
```

```
...
```

```
ObjectOutputStream oi = new ObjectOutputStream(new  
    FileOutputStream(myFile));
```

```
oi.writeObject(a1);
```

(throws various exceptions...

```
...
```

```
ObjectInputStream oi = new ObjectInputStream(new  
    FileInputStream(myFile));
```

```
a2 = (A) oi.readObject();
```

(throws various exceptions...



# Let us build a hit counter - 1

```
public class Counter implements Serializable {
    int count = 0;
    Calendar timeStamp = Calendar.getInstance();
    public void increase(){
        count++;
        timeStamp = Calendar.getInstance();
    }
    @Override
    public String toString() {
        StringBuffer s = null;
        if (count == 0)
            s = new StringBuffer("<p>no hits yet</p>");
        else {
            s = new StringBuffer("<p>hits = ");
            s.append(count)
              .append("<br>last hit on ")
              .append(timeStamp.getTime().toString());
        }
        return s.toString();
    }
}
```

# Cookies



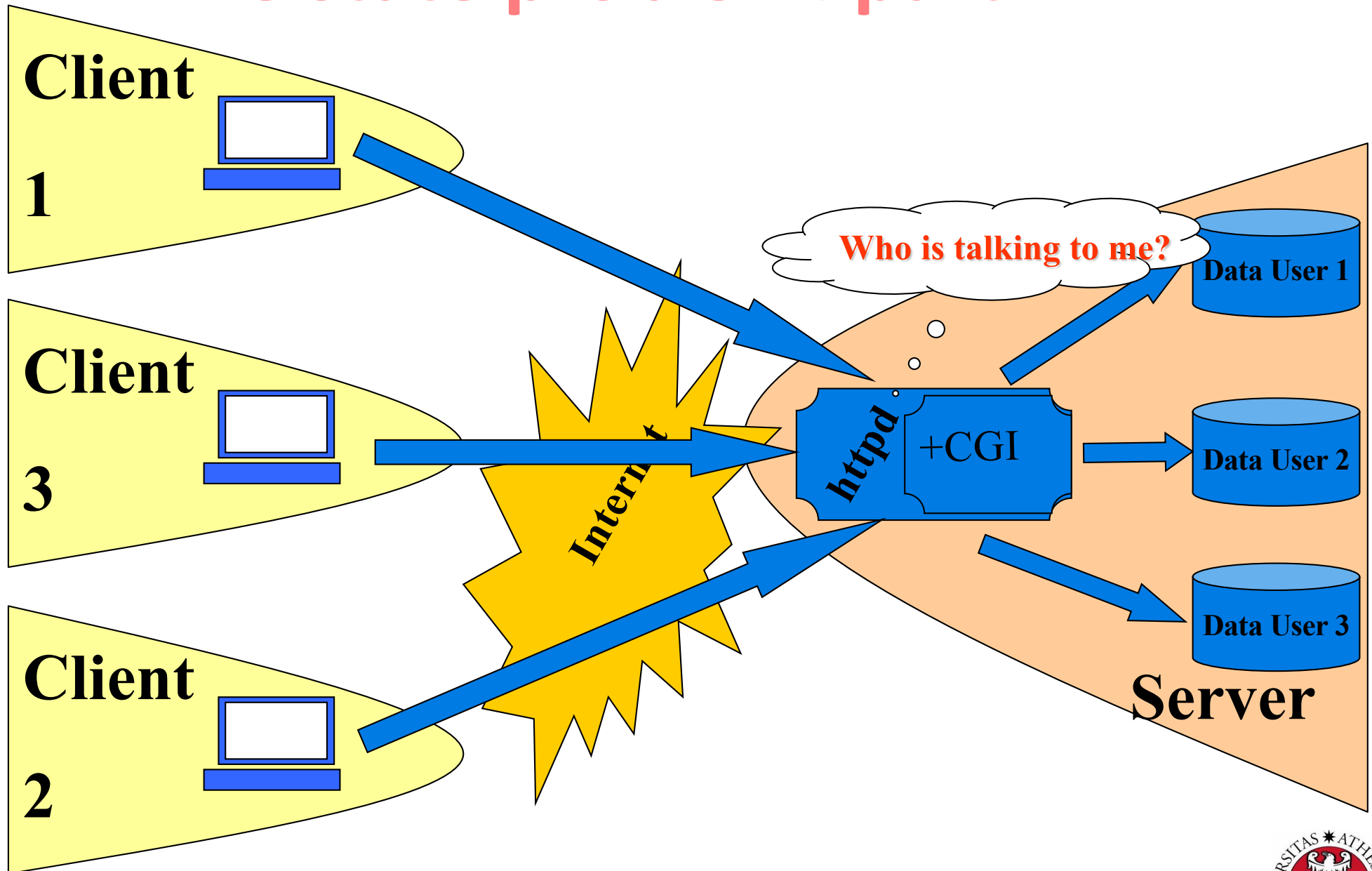
# HTTP is stateless

How can we keep track of who is who, and which state of the process s/he is at?

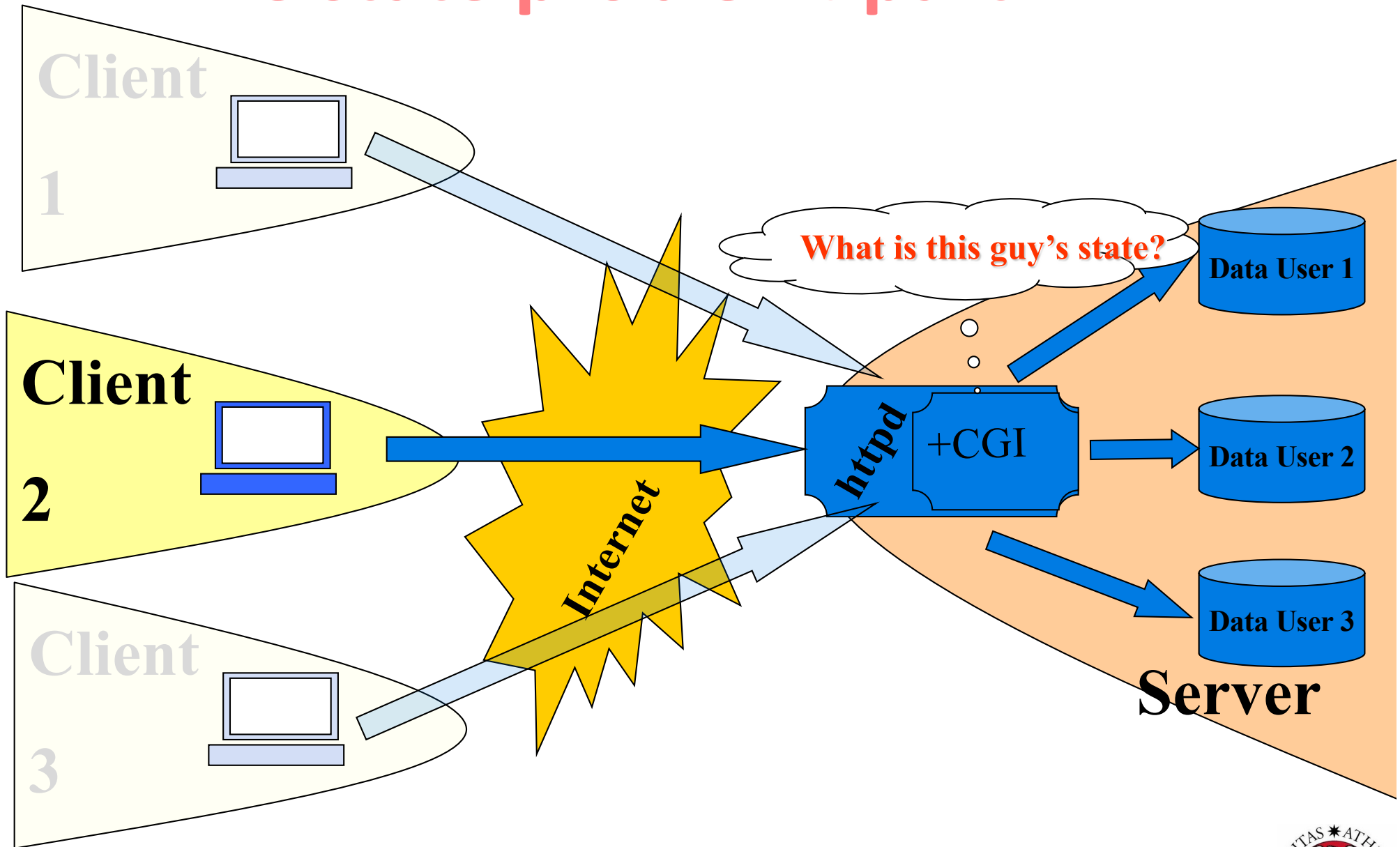
There is no way of doing it server-side...



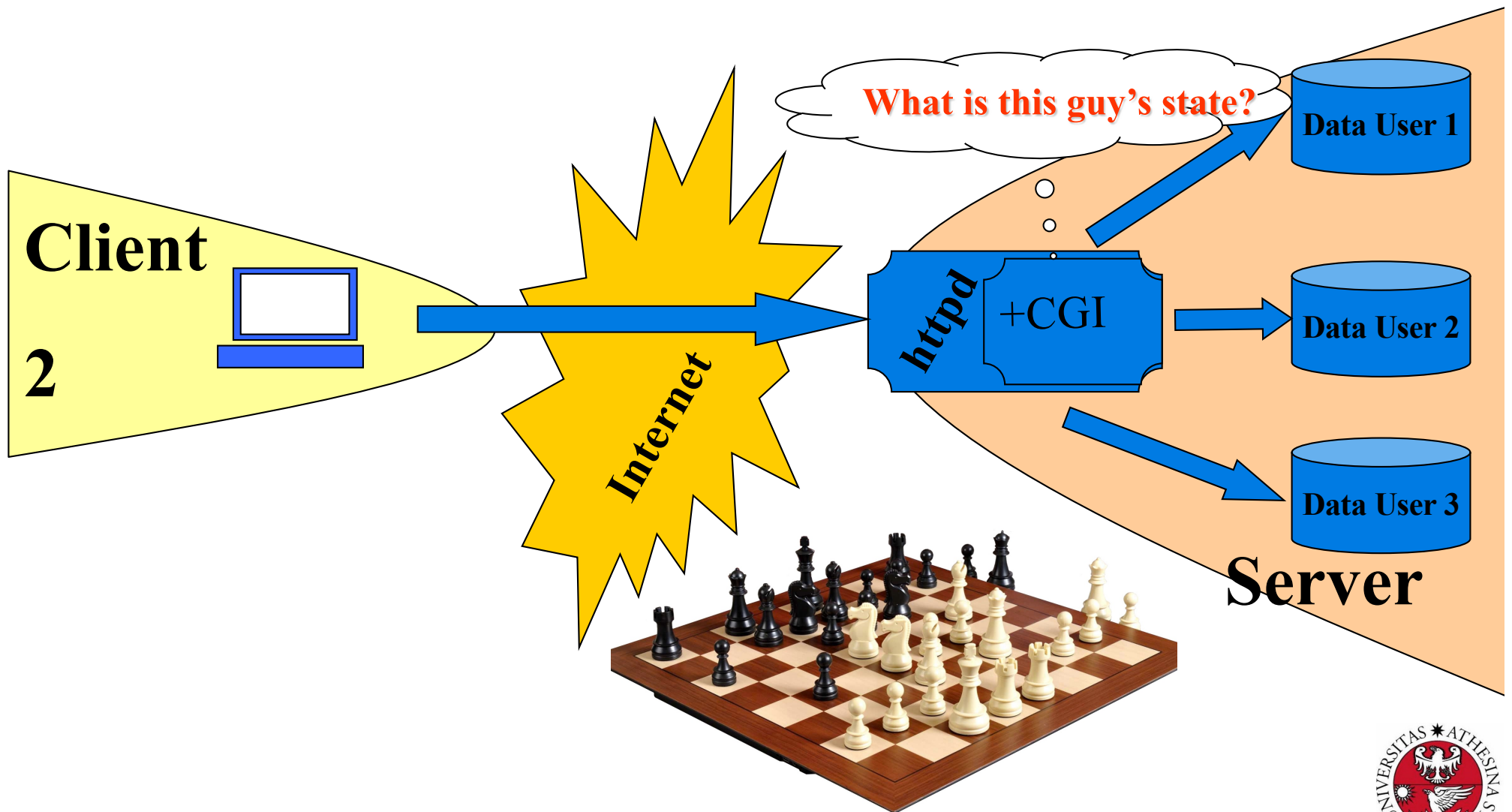
# The state problem: part 1



# The state problem: part 2

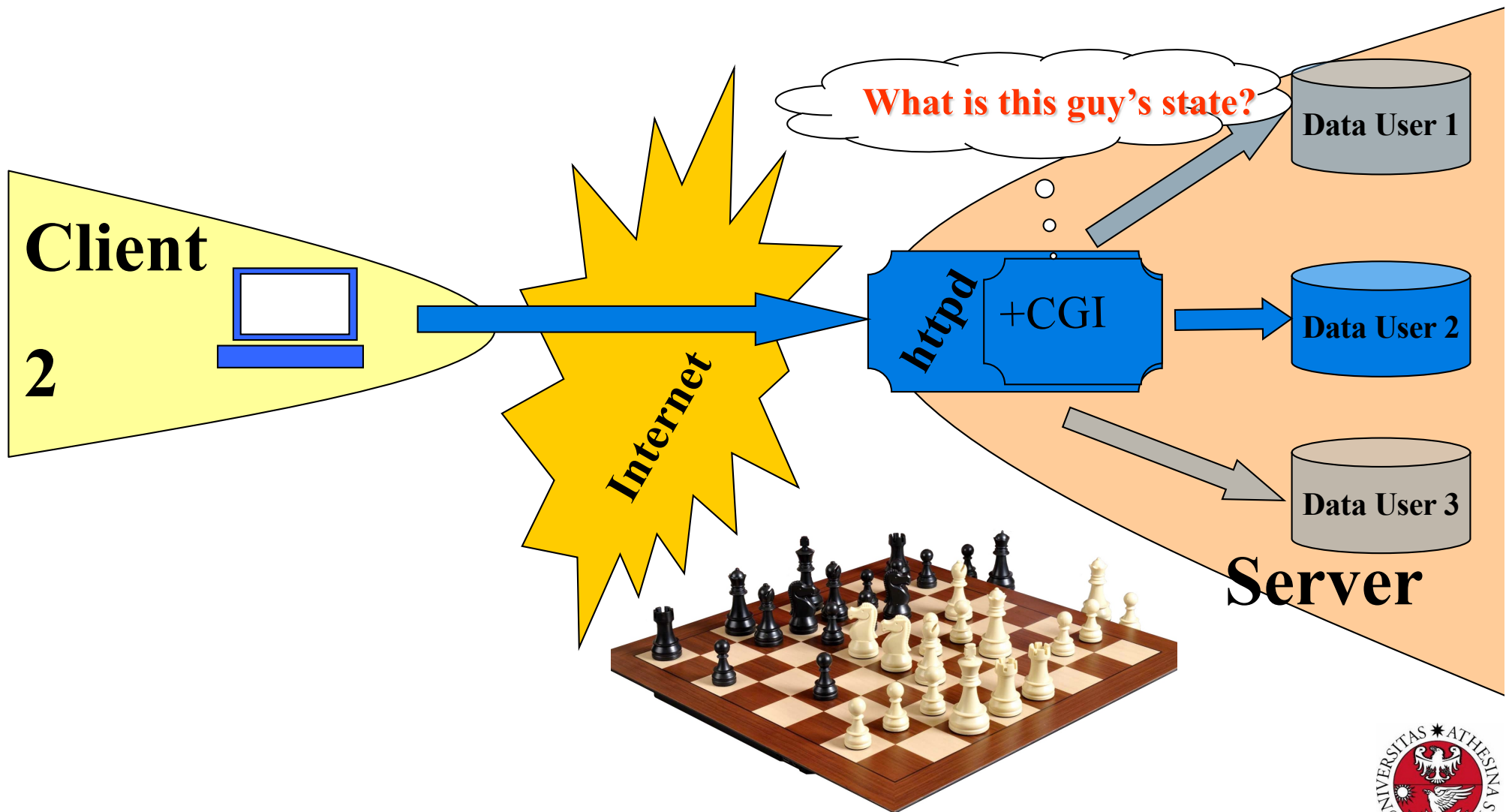


# The state problem: part 2 - example

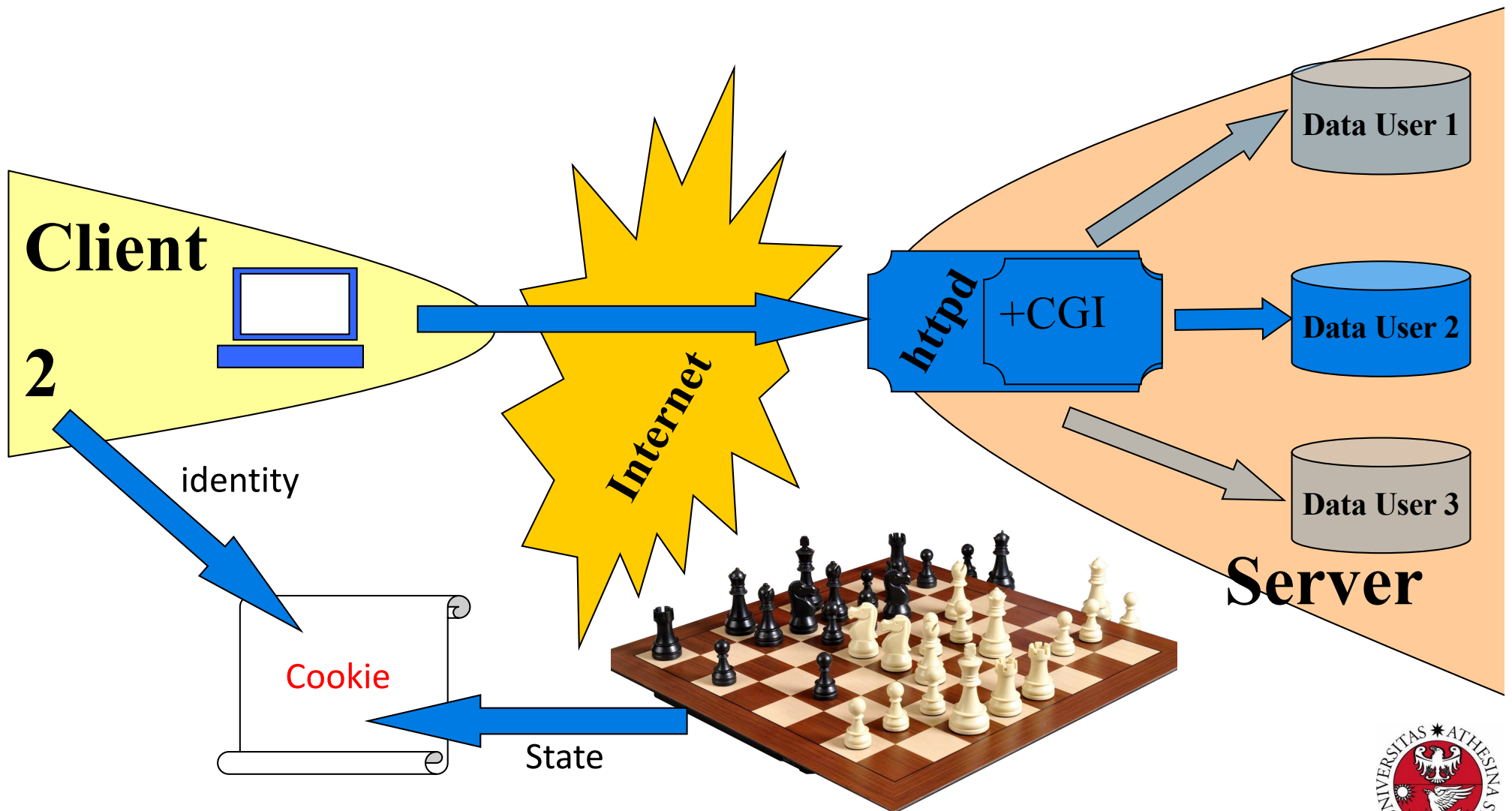




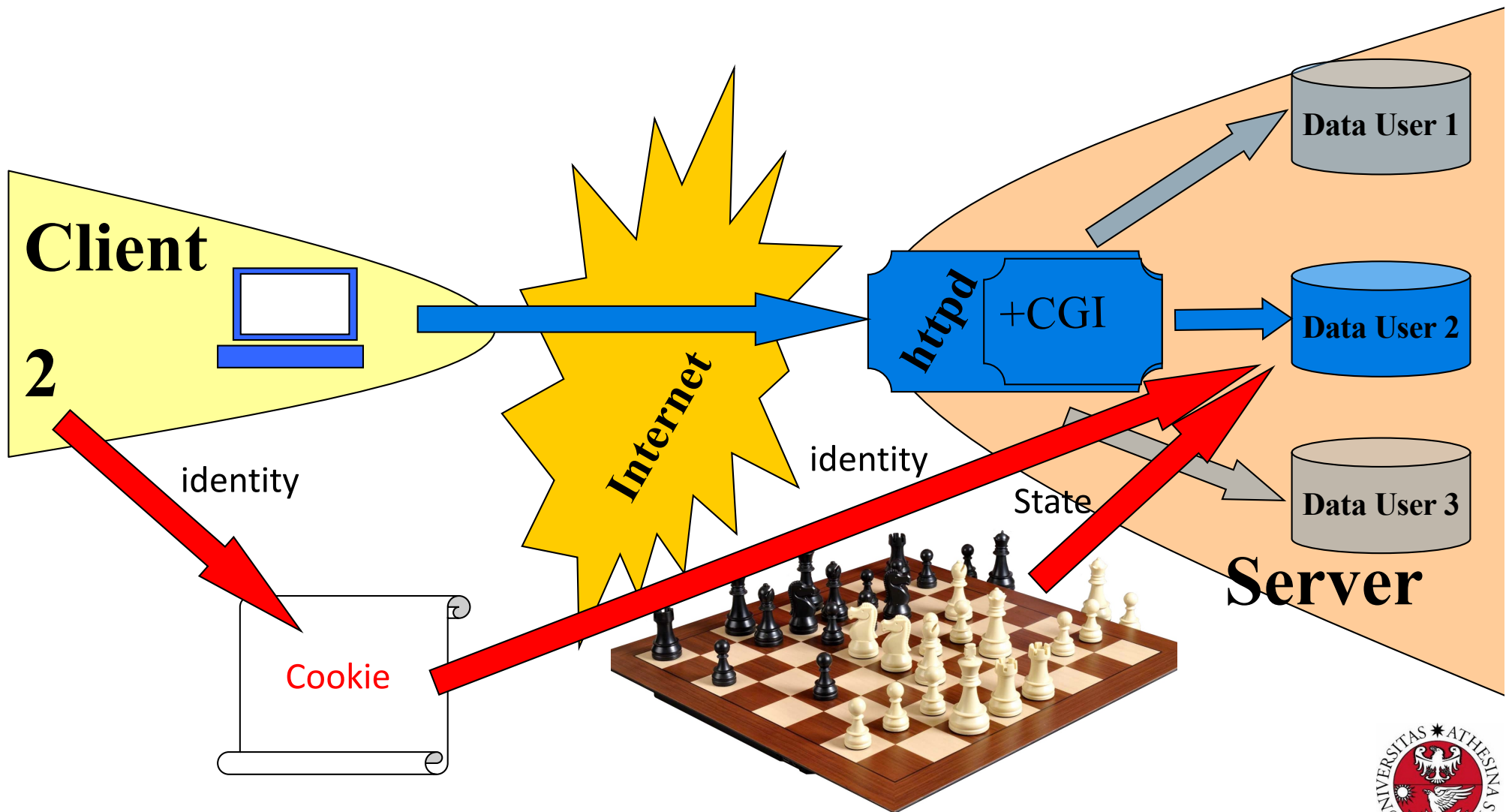
# The state problem: part 2 - example



# The state problem: solution 1



# The state problem: solution 2



# Cookies: what are they

A Cookie is a small amount of information sent by a servlet to a Web browser, saved by the browser, and later sent back to the server.

A cookie's value can uniquely identify a client, so cookies are commonly used for session management.

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.



# Where is the name coming from...



# Cookies: why?

- To maintain status across a “user session”
- To maintain infos across multiple interactions:
  - Customer identification
  - Targeted advertisement
  - Elimination of username e password



# Cookies

- The servlet sends cookies to the browser by using the `HttpServletResponse.addCookie(javax.servlet.http.Cookie)` method, which adds fields to **HTTP response headers** to send cookies to the browser, one at a time. The browser is expected to support 20 cookies for each Web server, 300 cookies total, and may limit cookie size to 4 KB each.
- The browser returns cookies to the servlet by adding fields to **HTTP request headers**. Cookies can be retrieved from a request by using the `HttpServletRequest.getCookies()` method.



# Cookies and caching

- Cookies affect the caching of the Web pages that use them. HTTP 1.0 does not cache pages that use cookies created with this class.
- The Java class “**Cookie**” does not support the cache control defined with HTTP 1.1. This class supports both the Version 0 (by Netscape) and Version 1 (by RFC 2109) cookie specifications. By default, cookies are created using Version 0 to ensure the best interoperability





# Attribute summary

- **String getComment() / void setComment(String s)**
  - Gets/sets a comment associated with this cookie.
  
- **String getDomain() / setDomain(String s)**
  - Gets/sets the domain to which cookie applies. Normally, cookies are returned only to the exact hostname that sent them. You can use this method to instruct the browser to return them to other hosts within the same domain. Note that the domain should start with a dot (e.g. .prehall.com), and must contain two dots for non-country domains like .com, .edu, and .gov, and three dots for country domains like .co.uk and .edu.es.



# Attribute summary

- **int getMaxAge() / void setMaxAge(int i)**
  - Gets/sets how much time (in seconds) should elapse before the cookie expires. If you don't set this, the cookie will last only for the current session (i.e. until the user quits the browser), and will not be stored on disk. See the LongLivedCookie class below, which defines a subclass of Cookie with a maximum age automatically set one year in the future.
- **String getName() / void setName(String s)**
  - Gets/sets the name of the cookie. The name and the value are the two pieces you virtually always care about. Since the getCookies method of HttpServletRequest returns an array of Cookie objects, it is common to loop down this array until you have a particular name, then check the value with getValue. See the getCookieValue method shown below.



# Attribute summary

- **String getPath() / void setPath(String s)**
  - Gets/sets the path to which this cookie applies. If you don't specify a path, the cookie is returned for all URLs in the same directory as the current page as well as all subdirectories. This method can be used to specify something more general. For example, `someCookie.setPath("/")` specifies that all pages on the server should receive the cookie. Note that the path specified must include the current directory.
- **boolean getSecure / setSecure(boolean b)**
  - Gets/sets the boolean value indicating whether the cookie should only be sent over encrypted (i.e. SSL) connections.



# Attribute summary

- **String getValue() / void setValue(String s)**
- Gets/sets the value associated with the cookie. Again, the name and the value are the two parts of a cookie that you almost always care about, although in a few cases a name is used as a boolean flag, and its value is ignored (i.e the existence of the name means true).
  
- **int getVersion() / void setVersion(int i)**
- Gets/sets the cookie protocol version this cookie complies with. Version 0, the default, adheres to the original Netscape specification. Version 1, not yet widely supported, adheres to RFC 2109.



## Placing Cookies in the Response Headers

The cookie is added to the Set-Cookie response header by means of the addCookie method of HttpServletResponse. Here's an example:

```
Cookie userCookie = new Cookie("user",  
"uid1234");  
response.addCookie(userCookie);  
// before opening the body of response!  
// i.e. before any out.print
```



# Reading Cookies from the Client

- **To read the cookies that come back from the client, you call `getCookies` on the `HttpServletRequest`. This returns an array of `Cookie` objects corresponding to the values that came in on the `Cookie` HTTP request header.**
- **Once you have this array, you typically loop down it, calling `getName` on each `Cookie` until you find one matching the name you have in mind. You then call `getValue` on the matching `Cookie`, doing some processing specific to the resultant value.**



# Cookies: examples

- `Cookie userCookie = new Cookie("user", "uid1234");`
- `userCookie.setMaxAge(60*60*24*365);`
- `response.addCookie(userCookie);`



# Cookies

Demo: `setCookies` - `showCookies`





# SetCookies - 1

Sets six cookies:

- three that apply only to the current session (regardless of how long that session lasts)
- three that persist for five minutes (regardless of whether the browser is restarted).

```
import ...
```

```
public class SetCookies extends HttpServlet {
```

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

```
        for(int i=0; i<3; i++) {
```

```
            Cookie cookie = new Cookie("Session-Cookie-" + i,  
                "Cookie-Value-S" + i);
```

```
            response.addCookie(cookie);
```

```
            cookie = new Cookie("Persistent-Cookie-" + i,  
                "Cookie-Value-P" + i);
```

```
            cookie.setMaxAge(3600);
```

```
            response.addCookie(cookie);
```

```
        }
```

Default maxAge is -1, indicating cookie applies only to current browsing session

Cookie is valid for an hour, regardless of whether user quits browser, reboots computer, or whatever.



## SetCookies - 2

```
response.setContentType("text/html");
PrintWriter out = response.getWriter();
String title = "Setting Cookies";
out.println("<HTML><HEAD><TITLE>" +title+ "</TITLE></HEAD>"
    +"<BODY BGCOLOR=\"#FDF5E6\">" +"<H1 ALIGN=\"CENTER\">"
    + title + "</H1>"
    +"There are six cookies associated with this page.\n"
    + "</BODY></HTML>");
}
}
```



# ShowCookies - 1

Creates a table of the cookies associated with the current page

```
import ...;

public class ShowCookies extends HttpServlet {

    public void doGet(HttpServletRequest request, HttpServletResponse
        response) throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        String title = "Active Cookies";
        out.println("<HTML><HEAD><TITLE>" + title + "</TITLE></HEAD>" +
            "<BODY BGCOLOR=\"#FDF5E6\">\n" +
            "<H1 ALIGN=\"CENTER\">" + title + "</H1>\n" +
            "<TABLE BORDER=1 ALIGN=\"CENTER\">\n" +
            "<TR BGCOLOR=\"#FFAD00\">\n" +
            "    <TH>Cookie Name\n" + "    <TH>Cookie Value");
```



# ShowCookies - 2

```
Cookie[] cookies = request.getCookies();
Cookie cookie;
for(int i=0; i<cookies.length; i++) {
    cookie = cookies[i];
    out.println("<TR>\n" +
               "    <TD>" + cookie.getName() + "\n" +
               "    <TD>" + cookie.getValue());
}
out.println("</TABLE></BODY></HTML>");
}
}
```



# Cookies

Demo: Cookies in action



# Output

Chrome

Safari

Hi! What is your name?

Hi! What is your name?

Hi Marco, nice to meet you!  
Delete Cookies?

Hi Pietro, nice to meet you!  
Delete Cookies?

Hi Marco, welcome back! (2)  
Delete Cookies?

Hi Pietro, welcome back! (5)  
Delete Cookies?

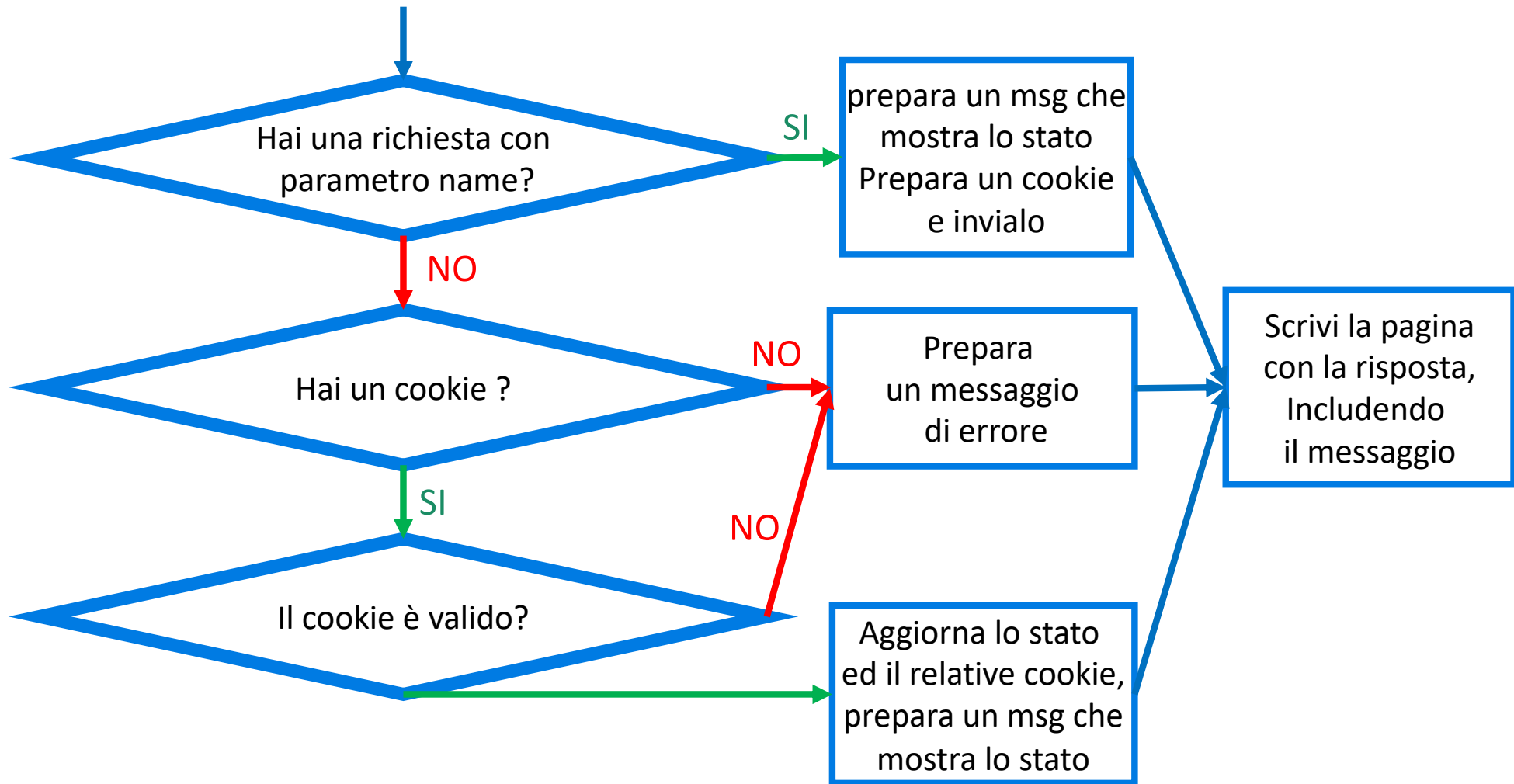
All cookies have been deleted  
Go to the [initial page](#).

Sorry, we do not know each other...  
Please introduce yourself.  
What is your name?

2 times

5 times

# Cookies in action - outline



# Cookies in action - 1

```
... Page index.html Welcome.java WhatsYourNameFragment.html DeleteCookiesFragment.html DeleteCookies.java...
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>Let's meet...</title>
5     <meta charset="UTF-8">
6     <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   </head>
8   <body>
9     <form method="GET" action="welcome">
10      <label for="name">Hi! What is your name?</label>
11      <input type="text" name="name">
12      <input type="submit" >
13    </form>
14  </body>
15 </html>
```

```
... Page index.html Welcome.java WhatsYourNameFragment.html DeleteCookiesFragment.html
3 <input type="text" name="name" value="">
4 <input type="submit" >
5 </form>
```





# Cookies in action - 2

```
... Page index.html x Welcome.java x WhatsYourNameFragment.html x DeleteCookiesFragment.html x DeleteCookies.java...  
2 <form method="GET" action="deleteCookies">  
3   Delete Cookies?  
4   <input type="submit" value="Yes, Delete">  
5   <input type="submit" value="No, Do not" formaction="welcome" >  
6 </form>
```

```
Files Servi... ...java x WhatsYourNameFragment.html x DeleteCookiesFragment.html x DeleteCookies.java x CookiesHaveBeenDeleted.html x  
WebAppWithCookies  
  Web Pages  
    META-INF  
    WEB-INF  
    CookiesHaveBe  
    DeleteCookiesF  
    WhatsYourNam  
    index.html  
  Source Packages  
    it.unitn.disi.ron  
      DeleteCooki  
      Welcome.jav  
  Test Packages  
Navigator x  
1 <!DOCTYPE html>  
2 <html>  
3 <head>  
4   <title>Cookies have been deleted</title>  
5   <meta charset="UTF-8">  
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">  
7 </head>  
8 <body>  
9   All cookies have been deleted <br>  
10  Go to the <a href="/WebAppWithCookies/welcome">initial page</a>.  
11 </body>  
12 </html>
```



# Cookies in action - 3

```
package it.unitn.disi.ronchet.myservlets;  
  
import ...  
  
@WebServlet(urlPatterns = {"/welcome"})  
public class Welcome extends HttpServlet {  
  
    String msg;  
    boolean isInitialIteration ;  
  
    private void dealWithInvalidCookie() {  
        msg = "Sorry, we do not know each other...<br>"  
            + "Please introduce yourself.<br>";  
        isInitialIteration = true;  
    }  
}
```



# Cookies in action - 4

```
protected void doGet(HttpServletRequest request,
    HttpServletResponse response)
    throws ServletException, IOException {
    isInitialIteration=false;
    // manage params and cookies
    String name = request.getParameter("name");
    if (name != null && ! name.equals("")) {
        // there is the right parameter,
        // no need to read cookie, but we set them
        log("name != null && ! name.equals(\"\")");
        Cookie cookie = new Cookie("name", name);
        msg = "Hi " + name + ", nice to meet you!";
        response.addCookie(cookie); // identity
        Cookie cookie1 = new Cookie("counter", "0");
        response.addCookie(cookie1); // state
        // finished! Go to end
    } else {
```



# Cookies in action - 5

```
} else {
    // no parameter, let's try with cookies
    Cookie cookies[] = request.getCookies();
    if (cookies==null || cookies.length == 0) {
        // no cookies
        log("no cookies found");
        dealWithInvalidCookie();
    } else {
        Cookie n_Cookie=null; // cookie con il nome
        Cookie c_Cookie=null; // cookie con il contatore
        for (Cookie c:cookies) {
            String cookieName = c.getName();
            if (cookieName.equals("name")) {
                n_Cookie=c;
            } else if (cookieName.equals("counter")) {
                c_Cookie=c;
            }
        }
        if (n_Cookie==null) {
            log ("valid cookies not found");
            // invalid cookie
            dealWithInvalidCookie();
        } else {
```



# Cookies in action - 6

```
    } else {  
        // ok, the cookie is good!  
        String userName=n_Cookie.getValue();  
        String counterAsString=c_Cookie.getValue();  
        log ("name == "+userName);  
        // let's update the counter, and the cookie  
        int counter=Integer.valueOf(counterAsString)+1;  
        c_Cookie.setValue(""+counter);  
        response.addCookie(c_Cookie);  
        msg = "Hi " + userName + ", welcome back! ("  
            +counter+");";  
    }  
}
```



# Cookies in action - 7

```
// prepare response and send it
    response.setContentType("text/html;charset=UTF-8");
    try (PrintWriter out = response.getWriter()) {
        out.println("<!DOCTYPE html>");
        out.println("<html><body>");
        out.println(msg);
        if (isInitialIteration) {
            request.getRequestDispatcher(
                "WhatsYourNameFragment.html")
                .include(request, response);
        } else {
            request.getRequestDispatcher(
                "DeleteCookiesFragment.html")
                .include(request, response);
        }
        out.println("</body></html>");
    }
}
```



# Cookies in action – 8 - deleteCookies

```
@WebServlet(name = "DeleteCookies",
            urlPatterns = {"/deleteCookies"})
public class DeleteCookies extends HttpServlet {

    protected void doGet(HttpServletRequest request,
                        HttpServletResponse response)
        throws ServletException, IOException {
        Cookie cookies[]=request.getCookies();
        if (cookies != null) {
            for (Cookie c : cookies) {
                c.setMaxAge(0);
                response.addCookie(c);
            }
        }
        response.setContentType("text/html;charset=UTF-8");
        request.getRequestDispatcher(
            "CookiesHaveBeenDeleted.html")
            .include(request, response);
    }
}
```



# Cookies in PHP

[https://www.w3schools.com/php/php\\_cookies.asp](https://www.w3schools.com/php/php_cookies.asp)

