

# 课程回顾

## 1 Cookie的使用步骤

```
1 1-1 存入数据
2 //存入中文，中文进行编码
3 value=URLEncoder.encode(value,"码表");
4 Cookie myCookie=new Cookie(name,value);
5 //设置cookie的有效期
6 myCookie.setMaxAge(n);//秒
7 //发送给浏览器
8 response.addCookie(myCookie);
9
10 1-2 获取Cookie
11 Cookie[] cookies=request.getCookies();
12 for(Cookie c:cookies){
13     if(c.getName.equals("要找的cookie的名称")){
14         value=c.getValue();
15         //取出中文，对中文进行解码
16         value=URLDecoder.decode(value,"码表");
17     }
18 }
```

## 2 Cookie的有效期

- 1 默认情况：Cookie存储的数据，随着浏览器关闭，数据丢失
- 2 一般，如果想延长cookie的有效期，可以setMaxAge()设置有效期。Cookie丢失与否与浏览器关闭没有关系

## 3 会话跟踪技术

- 1 会话：服务器与浏览器之间一次会晤
- 2 随着浏览器打开，会话就产生，浏览器关闭，会话就结束
- 3 一个会话：包含多次的请求和响应

# 课程目标

## 1 Session对象

## 2 验证码实现

## 3 JSP实现原理

## 4 文件下载

# 课程实施

## 1 Session

## 1-1 概念

- 1 特点：存储服务器端

## 1-2 原理

- 1 思考：Session的有效期的问题？
- 2 Session的数据确实随着浏览器关闭，数据就读取不到！！session的数据在服务器依然存在，默认保存30分钟，为什么浏览器重启之后，session数据获取不到？
- 3 原因：
- 4 保存JSESSIONID的cookie的生命周期默认的：cookie随着浏览器关闭，数据都会丢失
- 5
- 6 如果希望session在浏览器重启后还能拿到session保存的数据，解决方案：
- 7 延长保存JSESSIONID的cookie的生命周期

## 修改session有效期的方式

- web.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <web-app xmlns="http://java.sun.com/xml/ns/javaee"
3           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4           xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
5                               http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
6           version="2.5">
7 <!--
8 服务器上，一个session的有效期是设置的，设置的顺序是什么，如果自己的项目没有配置session的
9 有效期，
10 服务器就读取服务器的web.xml里面的配置
11 -->
12 <session-config>
13     <session-timeout>30</session-timeout>
14 </session-config>
15 </web-app>
```

## 1-3 课堂案例

- 1 需求：
- 2 1.登录页面 jsp
- 3 2.实现登录功能 servlet
- 4 登录成功，将登录的用户信息保存session
- 5 重定向到index.jsp
- 6 index.jsp上面显示当前登录用户名

## 参考代码

- login.jsp

```
1 <%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>
2
3 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
4 <html>
5 <head>
```

```

6      <title>login.jsp</title>
7  </head>
8  <script type="text/javascript">
9      function change(obj){
10          //obj:当前显示验证码的img标签
11          //?num=new Date().getTime()作用：实现服务器识别到这是一个新的请求，所以生成
            一个新的验证码
12          obj.src="${pageContext.servletContext.contextPath}/vc?num="+new
            Date().getTime();
13      }
14  </script>
15  <body>
16  <h1>login.jsp</h1>
17  <hr/>
18  ${msg}
19  <form action="${pageContext.servletContext.contextPath}/LoginServlet"
            method="post">
20      用户名: <input type="text" name="username" /><br/>
21      <input type="submit" value="登录"/>
22  </form>
23  </body>
24  </html>

```

- LoginServlet.java

```

1  package cn.kgc.controller; /**
2      * @Author: lc
3      * @Date: 2022/5/24
4      * @Description: ${PACKAGE_NAME}
5      * @Version: 1.0
6      */
7
8  import javax.servlet.*;
9  import javax.servlet.http.*;
10 import javax.servlet.annotation.*;
11 import java.io.IOException;
12
13 @WebServlet("/LoginServlet")
14 public class LoginServlet extends HttpServlet {
15     @Override
16     protected void doGet(HttpServletRequest request, HttpServletResponse
            response) throws ServletException, IOException {
17         request.setCharacterEncoding("utf-8");
18         //1.取
19         String username = request.getParameter("username");
20         //2.存 Session对象
21         //2-1 获取Session对象,getSession(true)和getSession() 尝试获取，如果获取不
            到，创建一个Session对象
22         //getSession(false):只取，不会创建
23         //Servlet获取session，建议getSession()或getSession(true)
24         HttpSession session = request.getSession();
25         //2-2 存数据
26         session.setAttribute("user",username);
27         //2-3 延长Cookie的生命周期
28         //同名的情况下，新的cookie覆盖以前cookie
29         Cookie c=new Cookie("JSESSIONID",session.getId());
30         //设置cookie的有效期

```

```

31         c.setMaxAge(30*60);
32         response.addCookie(c);
33
34         //3.重定向: 会产生两个请求
35
36         response.sendRedirect(getServletContext().getContextPath()+"/index.jsp");
37     }
38
39     @Override
40     protected void doPost(HttpServletRequest request, HttpServletResponse
41     response) throws ServletException, IOException {
42         doGet(request, response);
43     }
44 }

```

- index.jsp

```

1  <%@ page contentType="text/html;charset=UTF-8" language="java" %>
2  <html>
3  <head>
4      <title>Title</title>
5  </head>
6  <body>
7      用户名: ${user} <a
8      href="${pageContext.servletContext.contextPath}/LogoutServlet">安全退出</a>
9  </body>
10 </html>

```

- LogoutServlet.java

```

1  package cn.kgc.controller; /**
2      * @Author: lc
3      * @Date: 2022/5/24
4      * @Description: ${PACKAGE_NAME}
5      * @Version: 1.0
6      */
7
8  import javax.servlet.*;
9  import javax.servlet.http.*;
10 import javax.servlet.annotation.*;
11 import java.io.IOException;
12
13 @WebServlet("/LogoutServlet")
14 public class LogoutServlet extends HttpServlet {
15     @Override
16     protected void doGet(HttpServletRequest request, HttpServletResponse
17     response) throws ServletException, IOException {
18         //1.获取session对象 session如果不存在, 返回值NULL
19         HttpSession session = request.getSession(false);
20         //2.调用方法, 使session失效
21         if(session!=null){
22             //session失效的方案
23             session.invalidate();
24             //session存入用户信息, 手动删除
25         }
26     }
27 }

```

```

24         //session.removeAttribute("user");
25     }
26 }
27
28 @Override
29 protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
30     doGet(request, response);
31 }
32 }

```

## 2 验证码使用步骤

- 1 1. 项目中引入ValidateCode.java文件
- 2 2. jsp使用
- 3 3. loginServlet处理用户输入的验证码和正确的验证码一样的
- 4 如果不一样，就不要再做登录验证

### 2-1 验证码作用

- 1 防范暴力破解、暴力登录...
- 2 在我们注册时，如果没有验证码的话，可以使用while(true)来注册！那么服务器就废了！
- 3 验证码可以去识别发出请求的是人还是程序！当然，如果聪明的程序可以去分析验证码图片！但分析图片也不是一件容易的事，因为一般验证码图片都会带有干扰线，人都看不清，那么程序一定分析不出来。

### 2-2 验证码的使用案例

- login.jsp

```

1  <%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>
2
3  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
4  <html>
5  <head>
6      <title>login.jsp</title>
7  </head>
8  <script type="text/javascript">
9      function change(obj){
10         //obj:当前显示验证码的img标签
11         //?num=new Date().getTime()作用：实现服务器识别到这是一个新的请求，所以生成
            一个新的验证码
12         obj.src="${pageContext.servletContext.contextPath}/vc?num="+new
            Date().getTime();
13     }
14 </script>
15 <body>
16 <h1>login.jsp</h1>
17 <hr/>
18 ${msg}
19 <form action="${pageContext.servletContext.contextPath}/LoginServlet"
            method="post">
20     用户名: <input type="text" name="username" /><br/>
21     验证码: <input type="text" name="vc"/>
22     <br/>

```

```

23     <input type="submit" value="登录"/>
24 </form>
25 </body>
26 </html>

```

- LoginServlet.jsp

```

1  package cn.kgc.controller; /**
2   * @Author: lc
3   * @Date: 2022/5/24
4   * @Description: ${PACKAGE_NAME}
5   * @Version: 1.0
6   */
7
8  import javax.servlet.*;
9  import javax.servlet.http.*;
10 import javax.servlet.annotation.*;
11 import java.io.IOException;
12
13 @WebServlet("/LoginServlet")
14 public class LoginServlet extends HttpServlet {
15     @Override
16     protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
17         request.setCharacterEncoding("utf-8");
18         //1.取
19         String username = request.getParameter("username");
20         String vc = request.getParameter("vc");
21         //获取正确的验证码
22         String rightVC =
request.getSession(false).getAttribute("vc").toString();
23         if(!vc.equalsIgnoreCase(rightVC)){
24             request.setAttribute("msg","验证码输入有误! ");
25             //重新发回login.jsp
26             //带着错误的消息
27
28         request.getRequestDispatcher("/login.jsp").forward(request,response);
29         return;
30     }
31     //2.存 Session对象
32     HttpSession session = request.getSession();
33     //2-2 存数据
34     session.setAttribute("user",username);
35     //2-3 延长Cookie的生命周期
36     //同名的情况下,新的cookie覆盖以前cookie
37     Cookie c=new Cookie("JSESSIONID",session.getId());
38     //设置cookie的有效期
39     c.setMaxAge(30*60);
40     response.addCookie(c);
41
42     //3.重定向: 会产生两个请求
43
44     response.sendRedirect(getServletContext().getContextPath()+"/index.jsp");
45 }
46
47 @Override

```

```

46     protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
47         doGet(request, response);
48     }
49 }
50

```

## 坑点

### HTTP Status 500 - Cannot call sendRedirect() after the response has been committed

**type** Exception report

**message** Cannot call sendRedirect() after the response has been committed

**description** The server encountered an internal error that prevented it from fulfilling this request.

**exception**

```

java.lang.IllegalStateException: Cannot call sendRedirect() after the response has been committed
org.apache.catalina.connector.ResponseFacade.sendRedirect(ResponseFacade.java:483)
cn.kgc.controller.LoginServlet.doGet(LoginServlet.java:44)
cn.kgc.controller.LoginServlet.doPost(LoginServlet.java:49)
javax.servlet.http.HttpServlet.service(HttpServlet.java:647)
javax.servlet.http.HttpServlet.service(HttpServlet.java:728)

```

转发后，不能使用重定向

**note** The full stack trace of the root cause is available in the Apache Tomcat/7.0.42 logs.

Apache Tomcat/7.0.42

## 3 URL实现方式

### 3-1 url使用场景

- 1 当浏览器禁用cookie时，为了保证session数据可以正常获取，就必须在各个资源之间带着JSESSIONID来实现session数据的获取。

### 3-2 URL重写的实现方式

#### jsp页面的实现url重写

```

1  <%
2  String webName=request.getServletContext().getContextPath();
3  %>
4
5  <form action='<%=response.encodeURL(webName+"/LoginServlet")%>'
method="post">
6      用户名: <input type="text" name="username" /><br/>
7      验证码: <input type="text" name="vc"/>
8      <br/>
9      <input type="submit" value="登录"/>
10 </form>

```

#### servlet实现url重写

```

1 package cn.kgc.controller;
2 import javax.servlet.*;
3 import javax.servlet.http.*;
4 import javax.servlet.annotation.*;
5 import java.io.IOException;
6
7 @WebServlet("/LoginServlet")

```

```

8 public class LoginServlet extends HttpServlet {
9     @Override
10    protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
11        request.setCharacterEncoding("utf-8");
12        //1.取
13        String username = request.getParameter("username");
14
15        //2.存 Session对象
16        //2-1 获取Session对象,getSession(true)和getSession() 尝试获取, 如果获取不
到, 创建一个Session对象
17        //getSession(false):只取, 不会创建
18        //Servlet获取session, 建议getSession()或getSession(true)
19        HttpSession session = request.getSession();
20        //2-2 存数据
21        session.setAttribute("user",username);
22        //2-3 延长Cookie的生命周期
23        //同名的情况下, 新的cookie覆盖以前cookie
24        Cookie c=new Cookie("JSESSIONID",session.getId());
25        //设置cookie的有效期
26        c.setMaxAge(30*60);
27        response.addCookie(c);
28
29        //3.重定向: 会产生两个请求
30        //cookie禁用之后, 希望能够正常获取Session的数据, 原理: 程序员需要手动在各个资源
之间传递JSESSIONID
31
32        //response.sendRedirect(getServletContext().getContextPath()+"/index.jsp?
JSESSIONID="+session.getId());
33        //url重写
34        String url = response.encodeRedirectURL("/index.jsp");
35        response.sendRedirect(url);
36    }
37    @Override
38    protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
39        doGet(request, response);
40    }
41 }
42

```

## 4 文件下载

### 4-1 实现步骤

- 1 1. 提供下载资源:
  - 2 下载图片、exe文件
  - 3 下载资源存放位置: WEB-INF下面
  - 4 WEB-INF下面原因: WEB-INF下面的资源在浏览器上面不能直接访问。都是安全的
  - 5
  - 6 \*\*\*\*\*设置一个响应头: content-disposition\*\*\*\*\*
  - 7
- 8 2. 服务器文件实现复制到客户端
- 9 U1使用IO流: 图片复制



## 4-2 参考代码

### download.jsp

```
1  <%@ page contentType="text/html; charset=UTF-8" language="java" %>
2  <html>
3  <head>
4      <title>下载页面</title>
5  </head>
6  <body>
7  <a href="${pageContext.servletContext.contextPath}/DownloadServlet?
  fileName=aa.jpg">宠物猫</a>
8  <a href="${pageContext.servletContext.contextPath}/DownloadServlet?fileName=
  花.jpg">向日葵</a>
9  </body>
10 </html>
```

### DownloadServlet.java

```
1  package cn.kgc.controller; /**
2      * @Author: lc
3      * @Date: 2022/5/24
4      * @Description: ${PACKAGE_NAME}
5      * @Version: 1.0
6      */
7
8  import javax.servlet.ServletException;
9  import javax.servlet.ServletOutputStream;
10 import javax.servlet.annotation.WebServlet;
11 import javax.servlet.http.HttpServlet;
12 import javax.servlet.http.HttpServletRequest;
13 import javax.servlet.http.HttpServletResponse;
14 import java.io.IOException;
15 import java.io.InputStream;
16 import java.net.URLEncoder;
17
18 @WebServlet("/DownloadServlet")
19 public class DownloadServlet extends HttpServlet {
20     @Override
21     protected void doGet(HttpServletRequest request, HttpServletResponse
  response) throws ServletException, IOException {
22         //取：下载的图片的名称
23         String fileName = request.getParameter("fileName");
24         //文件名称：中文乱码问题
25         fileName=new String(fileName.getBytes("iso8859-1"),"utf-8");
26         //实现复制：服务器的图片使用IO写出到浏览器
27         //InputStream is=new FileInputStream("/WEB-INF/download/"+fileName);
28         //getServletContext():获取tomcat服务器上实际发布的web项目对象
29         InputStream is=getServletContext().getResourceAsStream("/WEB-
  INF/download/"+fileName);
30         byte[] bs=new byte[1024];
31         int len;
32         String agent=request.getHeader("User-Agent");
33
34         //解决文件另存时设置中文名称问题，火狐不识别
35         if(agent.toLowerCase().contains("firefox")){
```

```

36         //解决火狐的识别问题 IE识别
37         fileName=new String(fileName.getBytes("utf-8"),"iso8859-1");
38     }else{//IE Chrome
39         fileName= URLEncoder.encode(fileName,"utf-8");
40     }
41     //下载关键弹出另存为的窗口：设置响应头 content-disposition头名
    称, "attachment;filename=" + 另存默认保存文件名称
42     response.setHeader("content-disposition", "attachment;filename=" +
    fileName);
43     ServletOutputStream os = response.getOutputStream();
44     while((len=is.read(bs))!=0){
45         //一边读取图片的字节，一边输出到新的文件中
46         os.write(bs,0,len);
47         os.flush();
48     }
49     //先开后关
50     os.close();
51     is.close();
52 }
53
54 @Override
55 protected void doPost(HttpServletRequest request, HttpServletResponse
    response) throws ServletException, IOException {
56     doGet(request, response);
57 }
58 }
59

```

## 总结

---

1 session实现数据的存取

2 session作为服务器端存储数据的形式，当浏览器禁用cookie之后，可以使用URL重写的技术解决Session取值问题！！

3.文件下载

## 预习安排

---

文件上传和下载

AJAX处理