java.net

Java程序设计

第02-01讲 会话管理（Session）

Java课程组

本讲教学目标

- 了解会话管理的基本原理
- 掌握会话管理的四种技术实现

知识回顾/本讲先行知识

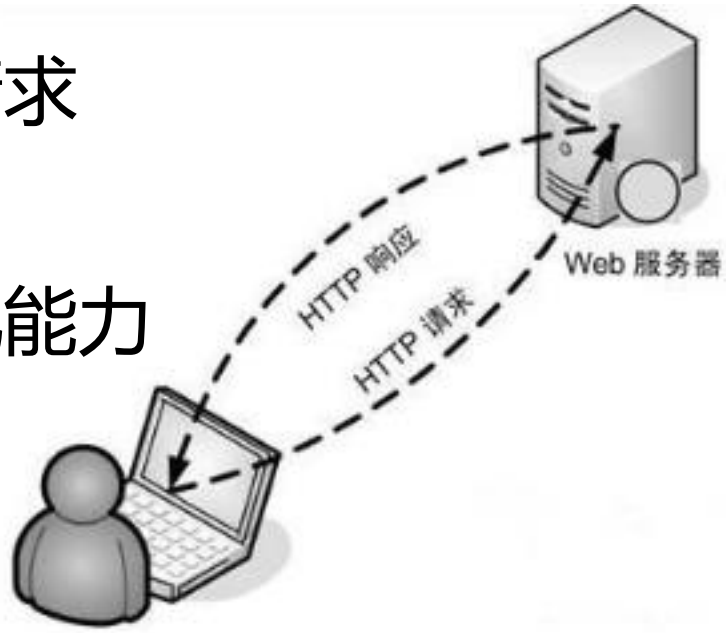
- Servlet模型（二）
  - 理解Servlet的生命周期
  - 熟悉Servlet相关对象

本讲内容

- 会话管理的概念和基本原理
- 使用隐藏域、Cookie、URL重写实现会话管理
- Session会话管理的原理和技术实现

为什么进行会话管理

- Web应用程序基于HTTP协议
- HTTP基于请求/响应模式
  - 所有请求都是相互独立的，无连续性的
- HTTP是无连接的协议
  - 限制每次连接只处理一个请求
- HTTP是无状态的协议
  - 协议对于事务处理没有记忆能力



会话管理的产生

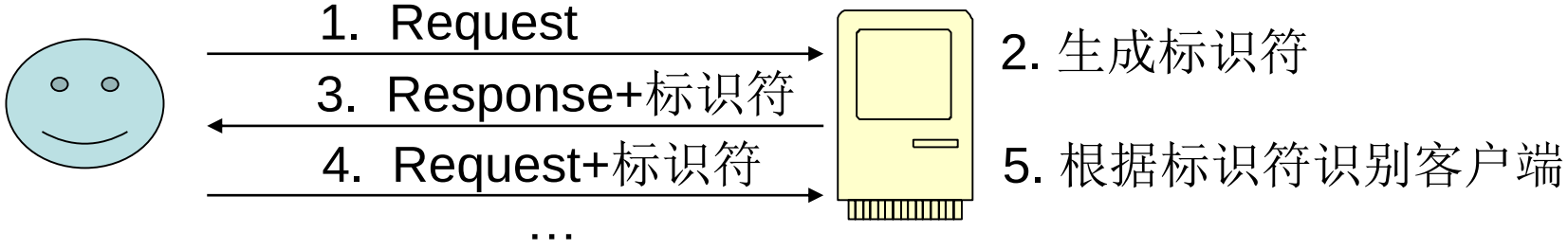
- 对于简单的页面浏览或信息获取，HTTP协议即可胜任
  - 浏览资讯
  - 查看在线图书目录
- 对于需要客户端和服务器端多次交互的网络应用，则必须记住客户端状态
  - 网上的购物车
  - 用户登录

会话管理概念

- 会话就是一个客户端连续不断地和服务器端进行请求/响应的一系列交互
- 多次请求间建立关联的方式称为会话管理，或会话跟踪
  - 会话状态，指服务器与浏览器在会话过程中产生的状态信息

会话的实现过程

- HTTP没有提供任何记住客户端的途径，服务器如何建立、维护与客户端的会话



  - 当服务器接收到客户端的首次请求时，服务器初始化一个会话并分配给该会话一个唯一标识符
  - 在以后的请求中，客户端必须将唯一标识符包含在请求中，服务器根据此标识符将请求与对应的会话联系起来

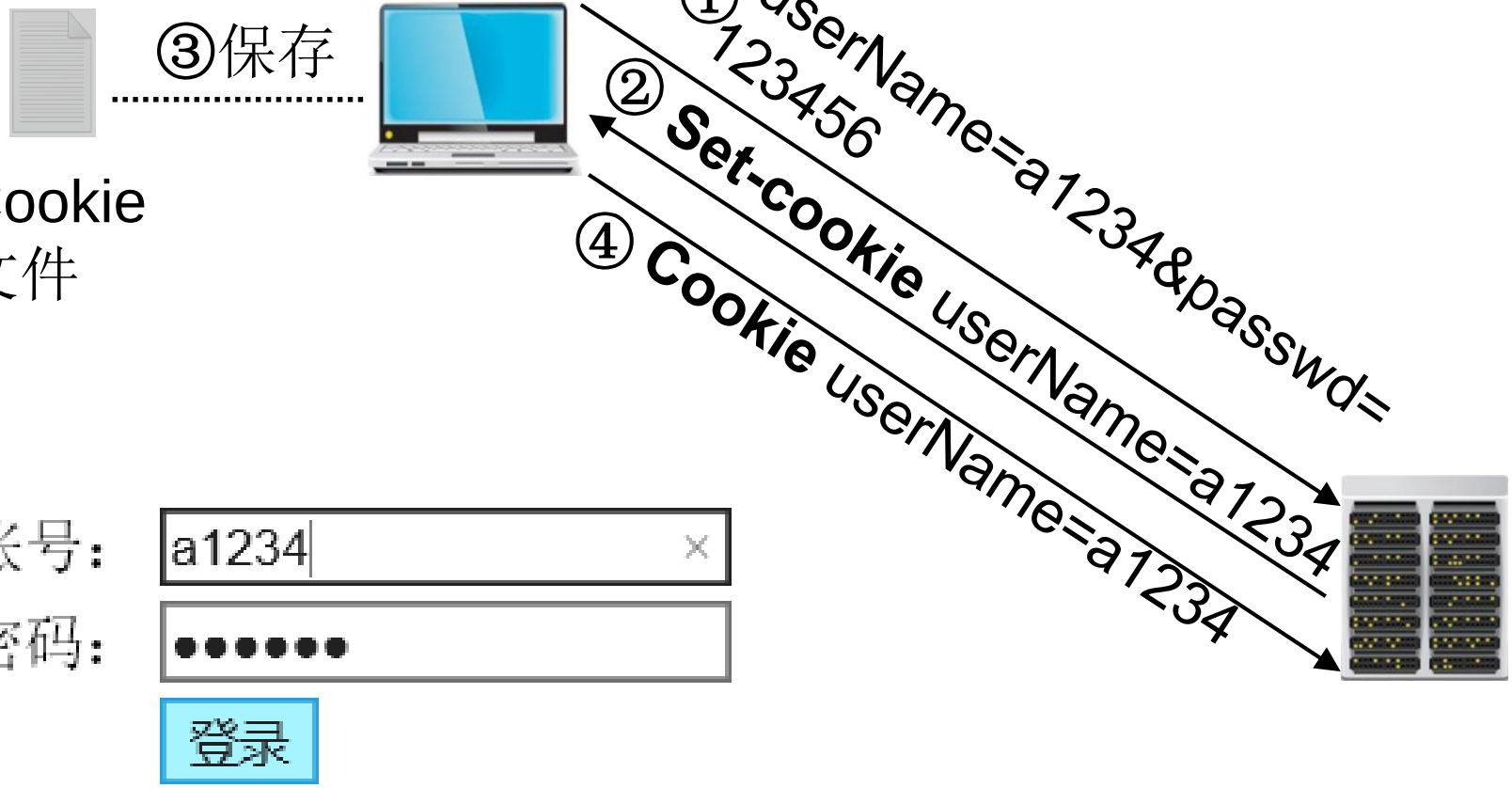
本讲内容

- 会话管理的概念和基本原理
- 使用Cookie、隐藏域、URL重写实现会话管理
- Session会话管理的原理和技术实现

会话管理：使用Cookie

- 所有的HTTP消息，不管是请求还是响应均包含头信息
  - 当服务器返回响应给客户端时，Servlet容器把会话的信息添加到响应头信息中
  - 客户端浏览器接收到响应后提取头信息，并将其存储在本地机中，以后发送请求时会自动将该信息带回服务器端
- 浏览器存储在客户端机器上的头信息称作Cookie，它以“属性名=属性值; …”方式组成文本信息

示例：使用Cookie实现用户登录



创建并向客户端发送Cookie

- 创建Cookie对象
  - 调用Cookie的构造方法，给出Cookie的名称和Cookie的值，二者都是字符串
    - Cookie c = new Cookie(“userName”,”a1234”)
- 设置最大时效
  - 如果要告诉浏览器将Cookie存储到磁盘上，而非仅保存在内存中，使用setMaxAge方法(参数为秒数)
  - c.setMaxAge(60*60*24*7)//一周(正数)
- 将Cookie放入到HTTP响应中
  - 使用response.addCookie(c)
  - 如没有这一步，将不会有任何Cookie被发送到浏览器

从客户端读取Cookie

- 调用request.getCookies
  - 得到Cookie对象组成的数组
  - 循环数组，调用每个对象的getName找到想要的cookie
  - 根据应用程序调用getValue方法使用这个Cookie

```
Cookie[] cookies = request.getCookies();
if(cookies != null){
    for(Cookie cookie : cookies){
        if("userId".equals(cookie.getName())){
            //doSomethingWith(cookie.getValue());
        }
    }
}
```


Cookie的方法

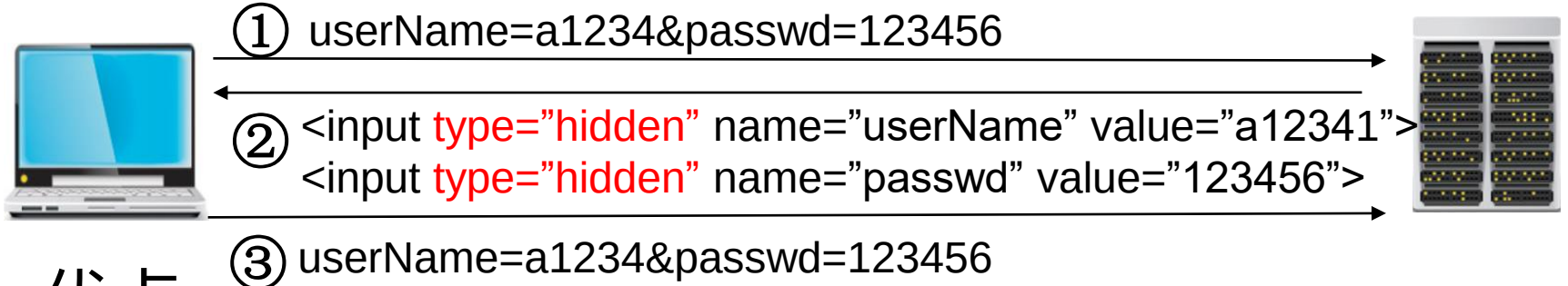
- getMaxAge()/setMaxAge()
  - 读取/设置Cookie到期时间(秒)
    - 如果值为0，表示删除对应的Cookie
    - 如果值为负数，表示Cookie只适用于当前的浏览会话
    - 默认值为-1
- getName()
  - 读取Cookie的名称。不存在setName()方法
- getValue()/setValue()
  - 读取/设置与Cookie关联的值
    - 如果重新设置了某Cookie的值，需要用addCookie()方法将其发送回去

Cookie的优缺点

- 优点
  - 可配置到期规则，数据可持久保存
  - 不需要服务器资源，数据保存在客户端
  - 简单性，基于文本的Key-Value对
- 缺点
  - 大小受到限制(总数300;20/站点;4KB/Cookie)
  - 用户可禁用客户端接收Cookie的功能
  - 潜在的安全风险

会话管理：使用隐藏的表单域

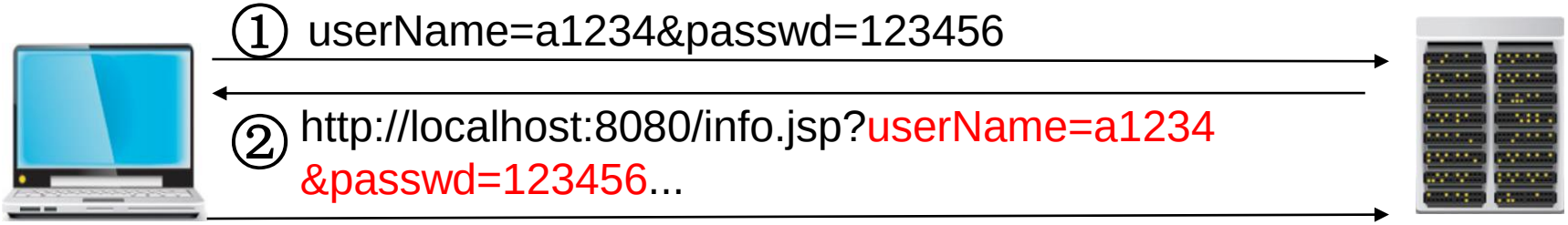
- 思想
  - 通过使用隐藏域，由浏览器主动告知服务器多次请求间必要的信息，如在线问卷作答



- 优点
  - Cookie被禁用或者根本不支持的情况下依旧能够工作
- 缺点
  - 关掉网页后会遗失先前的请求信息
  - 所有的页面必须是表单提交之后的结果

会话管理：使用URL重写

- 思想
  - 当服务器响应浏览器上一次请求时,将某些相关信息以超链接方式响应给浏览器,超链接中包括请求参数信息

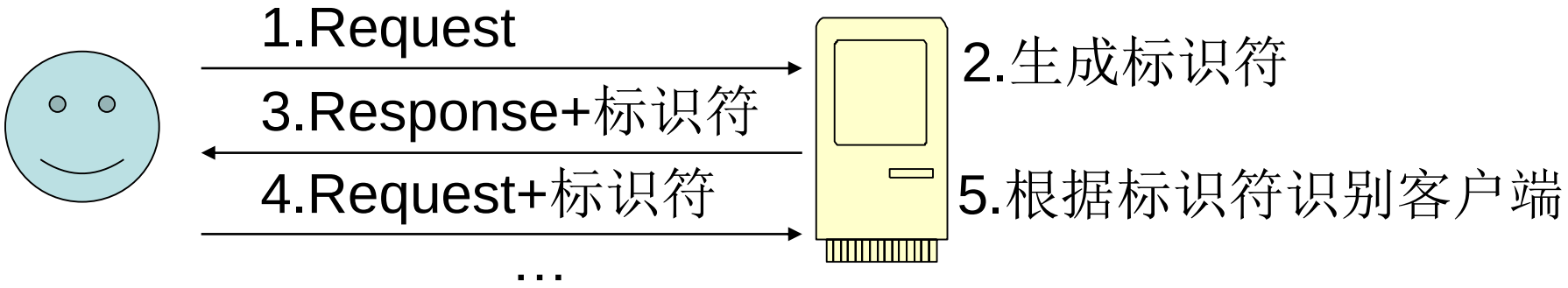


  - 由于GET方式发送请求,通常URL重写用于简单客户端信息保留,或辅助Session会话管理

本讲内容

- 会话管理的概念和基本原理
- 使用Cookie、隐藏域、URL重写实现会话管理
- Session会话管理的原理和技术实现

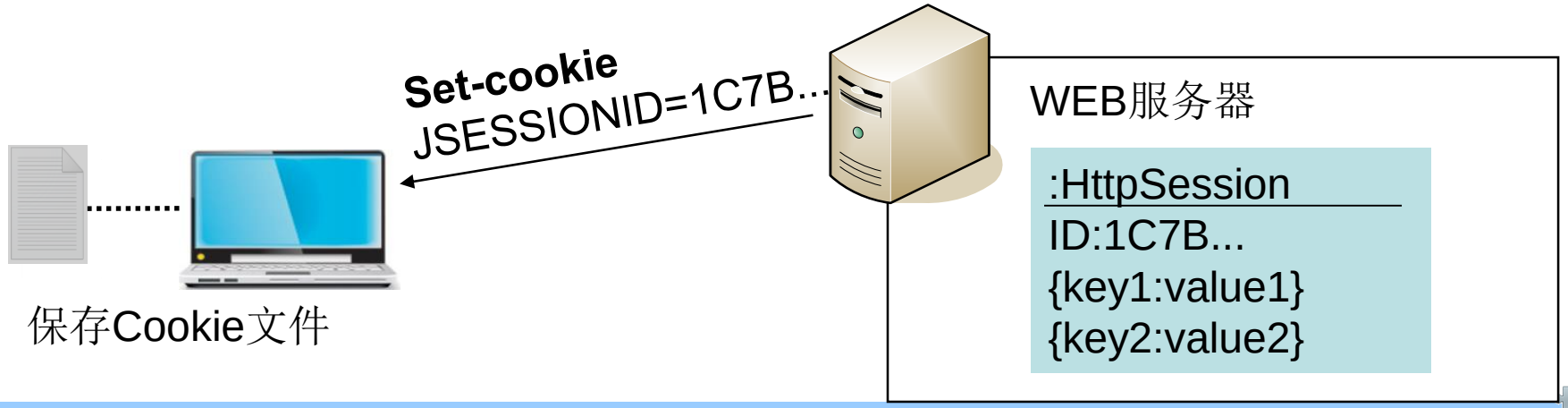
Session



- 在Servlet中进行会话管理，可以使用HttpServletRequest的getSession()方法取得HttpSession对象(简称为Session),通过设置/获取服务器端Session对象的属性,来保留请求之间的相关信息

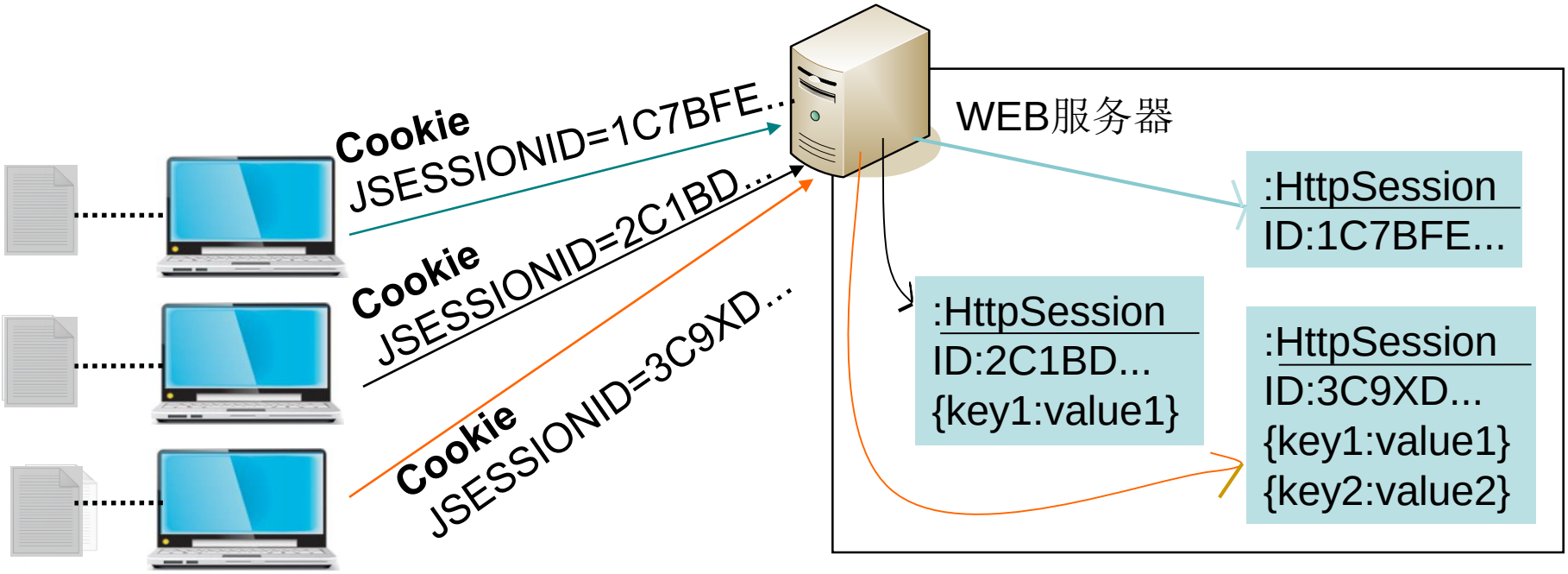
Session

- Servlet容器提供Session接口来代表服务器端和客户端的会话
  - 当一个WEB服务器为客户端开始一个会话时,创建一个新的Session对象(含有特殊ID,称为Session ID,默认用Cookie存放在浏览器中。在Tomcat中,Cookie的名称为JSESSIONID)



Session

– Session将数据存储在服务器的内存中,供以后来自同一个客户端的请求使用



示例：使用Session实现用户登录

– 登录时,创建Session对象并设置相关属性

```
HttpSession session = req.getSession();
session.setAttribute("userName", userName);
```

– 再次访问时,获取Session对象和相关属性

```
HttpSession session = req.getSession();
String user = (String)session.getAttribute("userName");
```

- 默认在关闭浏览器前,所得Session都是相同实例

使用Session对象

- 通常分三个步骤
  - 获取一个与请求相关联的会话
    - HttpSession session = request.getSession();
  - 从Session中设置或获取一个属性
    - session.setAttribute(“userName”,userName);
    - session.getAttribute(“userName”,userName);
  - 根据需要关闭会话
    - session.invalidate();
    - 通常客户端不提供结束会话的通知，而是Servlet容器在用户处于一段非活动期后就会自动的使会话失效——这个时间段称为会话的超时期

会话失效

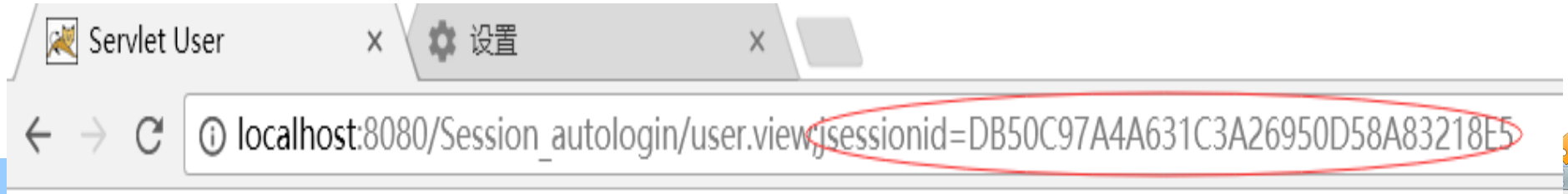
- Session对象失效：当用户超出指定会话期时间处于非活动状态时，会话自动结束
  - setMaxInactiveInterval()，设置会话的超时期
  - 通过web.xml的<session-timeout>标签设置
  - 使用方法invalidate()
- Cookie失效
  - 默认关闭浏览器Cookie消失
  - 在web.xml中设定存储Session ID的Cookie存活期限

Session与URL重写

- 当用户浏览器禁用Cookie时，仍打算运用Session来进行会话管理，可以对所有的URL使用URL重写
  - 可以使用HttpServletResponse的encodeURL()方法在URL后附加Session ID

```
//resp.sendRedirect("http://localhost:8080/Session_autologin/user.view");

resp.sendRedirect(resp.encodeURL("http://localhost:8080/Session_autologin/user.view"));
```



Session的方法

- getAttribute()
  - 从会话对象中提取出一个之前存储的属性值
    - 如果没有找到与名称相关联的值，则返回null
- setAttribute()
  - 设置会话对象的属性名称和属性值
- removeAttribute()
  - 移除与名称关联的值
- getAttributeNames()
  - 返回会话中所有属性的名称
- getId()
  - 返回唯一的标识符

Session的方法(续)

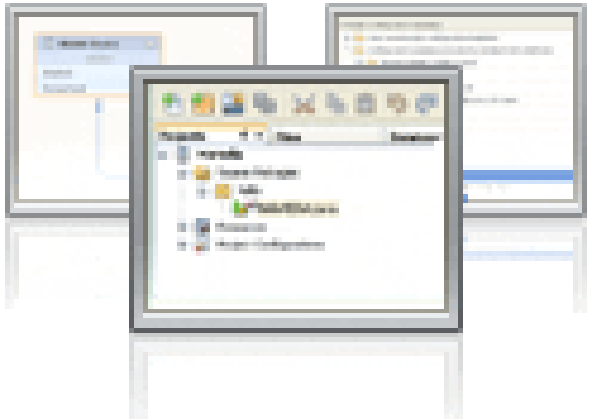
- isNew()
  - 确定会话对于客户来说是否为新创建
- getCreationTime()
  - 返回会话创建的时间
- getLastAccessedTime()
  - 返回客户端最近一次发送请求的时间
- getMaxInactiveInterval()、setMaxInactiveInterval()
  - 取得或设置会话的超时期
- invalidate()
  - 废弃当前的会话

小结

- 会话管理的概念和基本原理
- 使用隐藏域、Cookie、URL重写和Session进行会话管理的技术实现

实验

- 实验一
  - 分别使用Cookie和Session技术实现用户自动登录功能
- 实验二
  - 使用Session实现购物车功能



本讲结束

- 谢谢大家