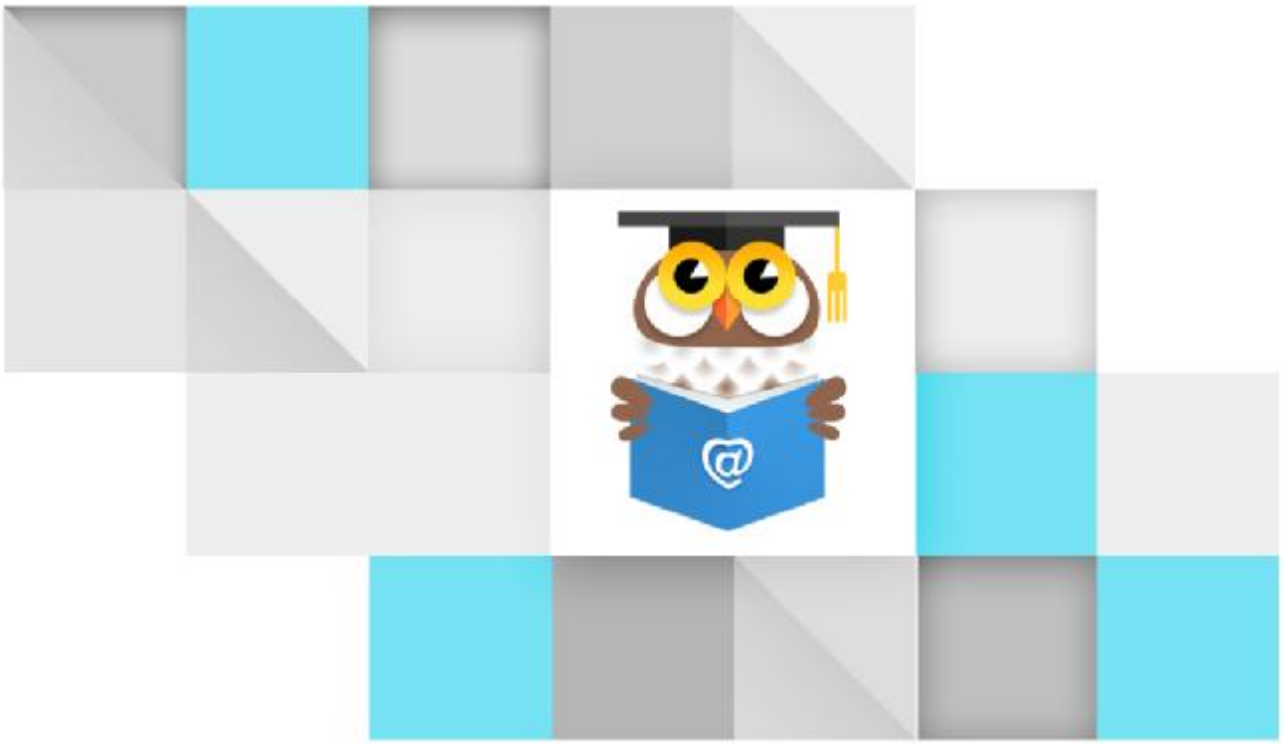
去哪玩旅游网（动态 WEB）

教学指导手册

PRJ-WTP-JEE-018 – 单态登录（EX）



Campus Solution Group

目 录

一、场景说明

1、完成效果

图 **1-1-1**

2、业务描述

本场景主要用于实现，《去哪玩》旅游网页面的单用户访问（单态登录）功能。

2-1. 什么是单态登录？

2-1.1. 单态登录意指：一个账号只允许被一人使用。

2-1.2. 例如：**A**客户端使用**X**账号登录站点，**B**客户端也使用**X**账号，则**A**客户端下线。

3、实现思路

3-1. 如何实现单态登录？

3-1.1. 整体思路见【核心组件】图。

3-1.2. 在站点的**Application**作用域中维护一个用户列表，该用户列表的作用是：

1）记录，登录账号与客户端唯一会话**ID**（**Session**）的对应关系。

说明**1**：**Tomcat**会为每个客户端定义一个唯一标识：会话**ID**（**SessionID**）。

说明**2**：记录账号与会话**ID**的对应关系，可知晓账号正被哪个客户端使用。

说明**3**：客户端发送请求时，请求中包含了与其对应的会话**ID**，若与用户列表中记录的会话**ID**不一致时，即可认定为多个用户试图使用同一账户登录系统。

3-1.3. 操作**Application**作用域中的用户列表：

1）用户登录成功时，将请求中包含的会话**ID**更新入用户列表。

说明**1**：该逻辑定义，用户列表中始终记录着实现登录操作的客户端会话**ID**。

2）当用户注销时，移除用户列表中保存的账号和客户端会话**ID**。

3-1.4. 利用过滤器监控：登录账户与客户端会话**ID**的一致性：

1）当用户试图访问隐私资源时，过滤器判断请求中包含的会话**ID**与用户列表中存储的会话**ID**是否一致。

2）不一致，说明其他用户已经使用同一账号登录系统了。

3）一致，表示没有其他用户使用当前账号。

3-2. 本场景建议将“单态登录”业务分为四个任务依次实现：

3-2.1. 任务**1.** 创建容器监听器，用于构建由站点**Application**作用域维护的用户列表。

3-2.2. 任务**2.** 创建会话监听器，用于获取发送登录、注销请求的客户端会话**ID**。

3-2.3. 任务**3.** 操作用户列表，确保登录账号始终匹配发送登录请求的客户端会话**ID**。

3-2.4. 任务**4.** 利用过滤器监控：登录账户与客户端会话**ID**的一致性。

4、核心组件

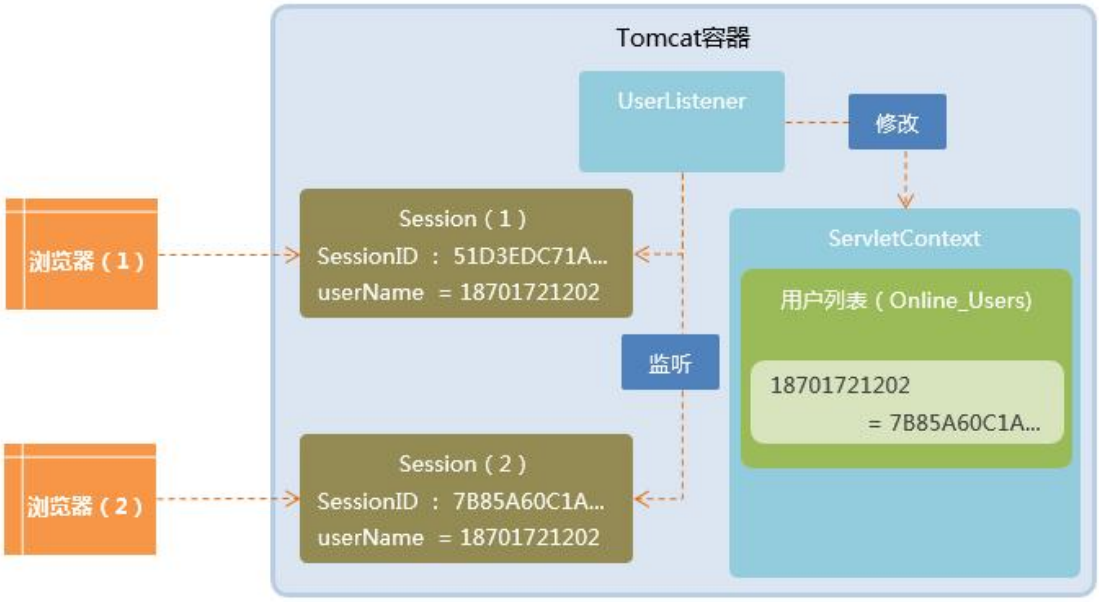


图 1-4-1

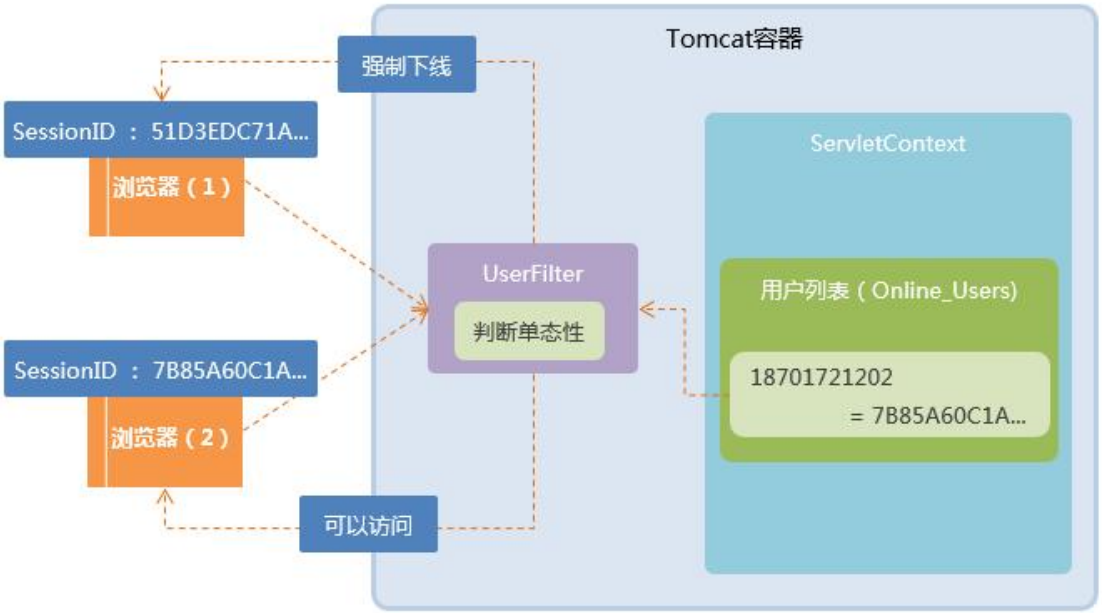


图 1-4-2

4-1. Session（系统组件）

4-1.1. Web应用四大作用域之一：会话作用域。

4-1.2. 每个客户端都会在**Web**服务端对应一个独立的**Session**容器。

4-1.3. 每个客户端通过独一无二的**SessionID**对应服务器的**Session**容器。

4-2. ServletContext（系统组件）

4-2.1. **Web**应用四大作用域之一：**Application**作用域。

4-2.2. 一个**Web**应用仅有一个**ServletContext**，所有请求共享该作用域。

4-3. 用户列表（需实现）

4-3.1. 用户列表，用于保存登录账户与客户端**SessionID**对应关系的集合。

4-3.2. 通过该集合可知晓，登录账户是由哪个客户端发送并执行登录操作的。

4-4. UserFitler（需实现）

4-4.1. 单态登录过滤器，负责拦截所有客户端请求。

4-4.2. 当前场景，该过滤器负责判断登录账户与客户端会话**ID**的一致性。

4-5. UserListener（需实现）

4-5.1. 会话监听器。

4-5.2. 当前场景，该监听器负责操作用户列表。

4-6. QunawanListener（需实现）

4-6.1. 容器监听器。

4-6.2. 当前场景，该监听器负责创建并存储用户列表。

二、实训技能

1、重点演练

1-1. 创建、配置、使用**ServletContextListener**监听器。

1-2. 创建、配置、使用**SessionAttributeListener**监听器。

1-3. 理解监听器的生命周期。

1-4. 理解单态登录的业务处理流程。

2、相关技能

- **Listener** 技术应用

3、相关知识点

- 使用 **Listener**
- 监听容器对象
- 监听属性变化
- 监听 **Session**
- 单态登录

4、前置条件

4-1. 前置场景：**PRJ-WTP-JEE-017** 登录状态过滤器

4-2. 已学技能：

4-2.1. Java开发工具（**Eclipse**）。

4-2.2. Web容器（**Tomcat**）。

4-2.3. Servlet应用技术。

4-2.4. JSP应用技术。

4-2.5. EL表达式应用技术。

4-2.6. Filter应用技术。

4-2.7. Listener应用技术。

5、搭建环境

三、场景任务

任务 1、创建并配置容器监听器

1. 任务说明：

1-1. 完成效果：

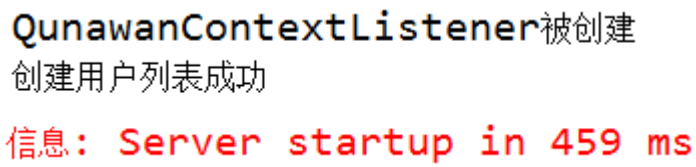


图 **3-1-1**

1-2. 任务目标：

1-2.1. 创建容器监听器，用于构建由站点**Application**作用域维护的用户列表。

1-3. 任务要求：

1-3.1. 创建并配置《去哪玩》旅游网容器监听器（**Listener**）：

1）限定**1. Listener**取名：**QunawanListener**。

2）限制**2.** 监听器创建于**campsg.qunawan.listener**包中。

说明：监控容器监听器，需要实现：**ServletContextListener**接口。

1-3.2. 创建用户列表，用于存储登录账户与客户端**SessionID**的对应关系。

1） 当**Tomcat**启动时，容器监听器创建用户列表集合。

1-3.3. 将用户列表保存于**Application**作用域中：

1）限定**1.** 存于**Application**作用域中的**key = online_users**。

1-3.4. 在控制台输出以下测试结果：

1） **Tomcat**启动时，输出 “**QunawanContextListener**被创建” 。

2） 用户列表保存于**Application**作用域后，输出：“创建用户列表成功”。

2. 实现思路：

2-1. 创建并配置容器监听器：**QunawanListener**：

2-1.1. 在**campsg.qunawan.listener**包中创建**Class**类，命名：**QunawanListener**。

2-1.2. 使**QunawanListener**实现**javax.servlet.http.ServletContextListener**接口。

2-1.3. 重写该接口中的**contextDestroyed**、**contextInitialized**方法。

2-1.4. 在**web.xml**中使用**<listener>**节点，配置**QunawanListener**。

2-2. 在**contextInitialized**方法中，创建用户列表集合。

2-2.1. 由于用户列表需保存登录账户和会话**ID**的对应关系，因此可考虑使用**HashMap**。

2-2.2. 由于两者均为字符，**HashMap**集合的泛型可考虑定义为：**<String,String>**。

2-2.3. 将用户列表集合保存于**ServletContext**中，**key = online_users**。

2-2.4. 向控制台输出：“**QunawanContextListener**被创建”。

2-2.5. 向控制台输出：“创建用户列表成功”。

2-3. 任务开发流程见下图：

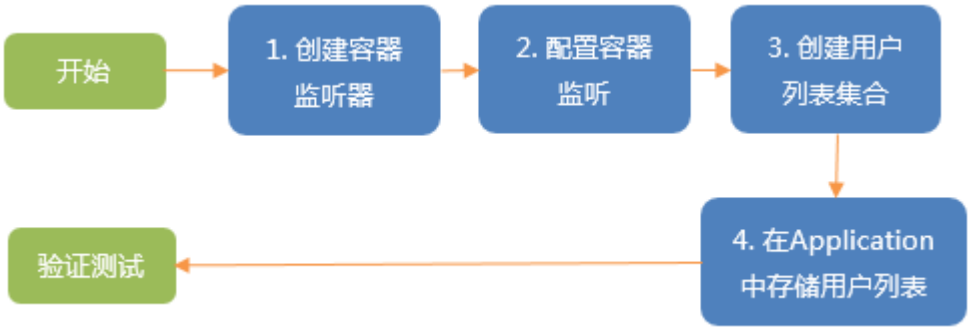


图 **3-1-2**

3. 推荐步骤：

3-1. 按实现思路，创建并配置名为**QunawanListener**的容器监听器。

3-2. 按实现思路，在**QunawanListener**启动时，创建用户列表：

3-2.1. 在**contextInitialized**方法中，创建用户列表集合。

3-2.2. 将用户列表保存于**Application**作用域：

+ 提示：

1）在监听器中获取**Application**作用域：

```
ServletContext context = event.getServletContext();
```

说明**1**：**event**对象是**contextInitialized**方法的参数，类型为：**ServletContextEvent**

2）将数据存放入**Application**作用域的方式，与使用**Request**作用域完全一致。

4. 验证与测试：

4-1. 运行项目工程。

4-2. 观察控制台，应见【任务完成效果截图】。

任务 2、创建并配置会话监听器

1. 任务说明：

1-1. 完成效果：

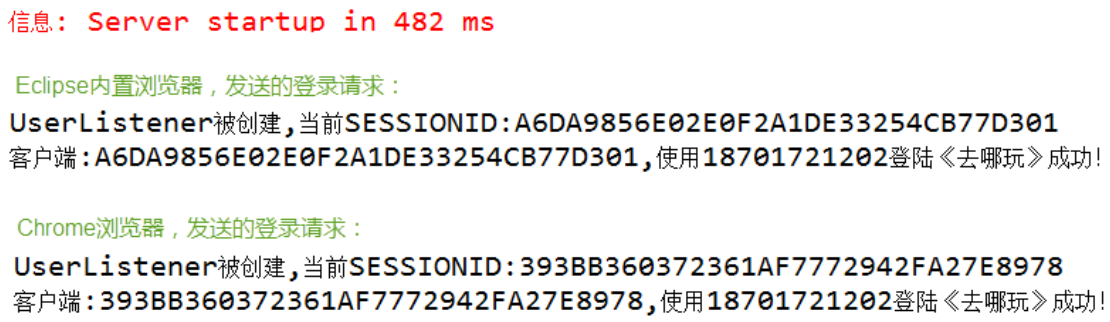


图 3-2-1

1-2. 任务目标：

1-2.1. 创建会话监听器，用于获取发送登录、注销请求的客户端**SessionID**。

1-3. 任务要求：

1-3.1. 创建会话监听器（**Listener**），监控**Session**的创建与销毁：

1）限定**1. Listener**取名：**UserListener**。

2）限制**2.** 监听器创建于**campsg.qunawan.listener**包中。

说明：监控**Session**的创建与销毁，需要实现：**HttpSessionListener**接口。

1-3.2. 获取创建**Session**的客户端，并打印该客户端的会话**ID**（**SessionID**）。

1） 在**Session**被创建时，获取客户端**SessionID**。

2） 输出： “**UserListener**被创建，当前**SESSIONID**：**"** 拼接客户端**SessionID**。

1-3.3. 获取发送登录、注销请求的客户端，并打印会话**ID**（**SessionID**）。

1）登录成功时输出： **"**客户端：**X**，使用**Y**登录《去哪玩》成功！**"**

说明：**X**为获取的客户端**SessionID**，**Y**为获取的用户登录账号。

2）注销时输出："客户端：X，执行注销操作"。

说明1：监控Session数据变化，需要实现：HttpSessionAttributeListener接口。

说明2：登录成功时，Servlet会将用户对象存入Session（key = user）。

说明3：注销时，Servlet会将存储在Session中的User移除。

2. 实现思路：

2-1. 创建并配置会话监听器：UserListener：

2-1.1. 在campsg.qunawan.listener包中创建Class类，命名：UserListener。

2-1.2. 在web.xml中使用<listener>节点，配置UserListener。

2-2. 获取创建Session的客户端，并打印该客户端的会话ID（SessionID）：

2-2.1. 使UserListener实现javax.servlet.http.HttpSessionListener接口。

2-2.2. 重写该接口中的sessionCreated、sessionDestroyed方法。

2-2.3. 在sessionCreated方法中获取HttpSession对象，并获取该对象的会话ID。

2-2.4. 输出："UserListener被创建，当前SESSIONID：" + 获取的SessionID。

2-3. 获取发送登录、注销请求的客户端，并打印会话ID（SessionID）：

2-3.1. 使UserListener实现javax.servlet.http.HttpSessionAttributeListener接口。

2-3.2. 重写该接口中的attributeAdded、attributeRemoved方法。

2-3.3. 在attributeAdded方法中获取发送登录请求的客户端SessionID：

说明：由于登录成功时Servlet会将用户对象存入Session中，因此UserListener的attributeAdded方法会监听到数据存入的操作。

1） 获取保存入Session的用户对象：User，并获取登录账号。

2） 获取Session中的SessionID。

3） 输出： "客户端："+会话ID+",使用"+登录账号+"登录《去哪玩》成功！"。

2-3.4. 在**attributeRemoved**方法中获取发送注销请求的客户端**SessionID**：

说明：由于注销时**Servlet**会将用户对象从**Session**中移除，因此**UserListener**的**attributeRemoved**方法会监听到数据移除的操作。

1） 获取**Session**中的**SessionID**。

2） 输出：**"客户端："** + 会话**ID** + **,执行注销操作"**。

2-4. 任务开发流程见下图：

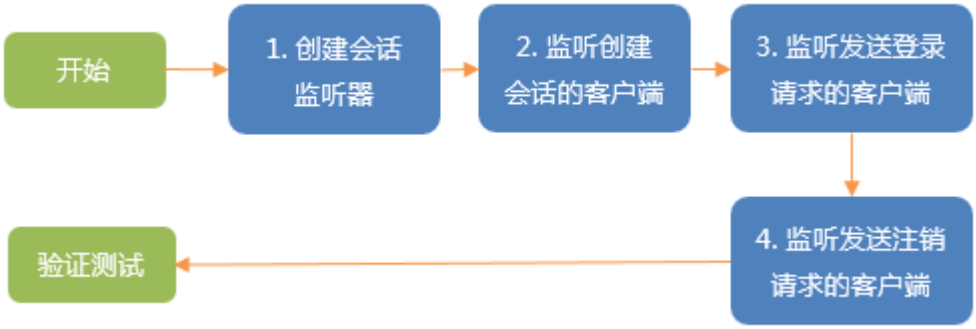


图 **3-2-2**

3. 推荐步骤：

3-1. 按实现思路，创建并配置名为**UserListener**的会话监听器。

3-1.1. **UserListener**实现**HttpSessionListener**接口。

3-1.2. **UserListener**实现**HttpSessionAttributeListener**接口。

3-1.3. 在**web.xml**文件中配置**UserListener**。

3-2. 获取创建**Session**的客户端会话**ID**（**SessionID**）：

3-2.1. 在**sessionCreated**方法中获取会话**ID**。

+ 提示：

1）在**sessionCreated**方法中获取**HttpSession**对象：

```
HttpSession session = event.getSession();
```

说明1：**event**对象是**sessionCreated**方法的参数，类型为：**HttpSessionBindingEvent**

2）通过**HttpSession**对象获取**SessionID**：

```
String sessionId = session.getId();
```

3-3. 获取发送登录请求的客户端**SessionID**：

3-3.1. 在**attributeAdded**方法中获取保存入**Session**的用户对象：**User**。

+ 提示：

1）在**sessionCreated**方法中获取**User**对象：

```
if (!event.getName().equals("user"))
    return;
// 进行属性添加操作中的值
User user = (User) event.getValue();
```

说明1：**event**对象是**sessionCreated**方法的参数，类型为：**HttpSessionBindingEvent**

说明2：**event.getName()**的含义是：被监听到的**Session**数据存入操作中，存入会话的**key**值。

说明3：**event.getValue()**的含义是：被监听到的**Session**数据存入操作中，存入会话的数据。

3-3.2. 获取**User**对象中的登录账号。

3-3.3. 获取**Session**中的**SessionID**。

3-3.4. 输出： **"客户端："+会话ID+",使用"+登录账号+"登录《去哪玩》成功！"**。

3-4. 获取发送注销请求的客户端**SessionID**：

3-4.1. 在**attributeRemoved**方法中获取**Session**中的**SessionID**。

3-4.2. 输出：**"客户端：" + 会话ID + ,执行注销操作"**。

4. 验证与测试：

4-1. 运行项目工程。

4-2. 测试**1.** 使用客户端**A**登录《去哪玩》旅游网。

4-2.1. 使用**Eclipse**内置浏览器（必须）。

4-2.2. 输入： **http://localhost/PRJ_WTP_JEE_018/** 访问登录页。

4-2.3. 在登录页输入正确的验证码。

4-2.4. 输入正确的用户名、密码【用户名：**18701721202**，密码：**123456**】。

4-2.5. 登录后，应见【任务完成效果截图】第一部分。

4-3. 测试**2.** 使用客户端**B**登录《去哪玩》旅游网。

4-3.1. 使用**Chrome**浏览器（必须）。

4-3.2. 输入： **http://localhost/PRJ_WTP_JEE_018/** 访问登录页。

4-3.3. 在登录页输入正确的验证码。

4-3.4. 输入正确的用户名、密码【用户名：**18701721202**，密码：**123456**】。

4-3.5. 登录后，应见【任务完成效果截图】第二部分。

任务 3、操作用户列表

1. 任务说明：

1-1. 完成效果：

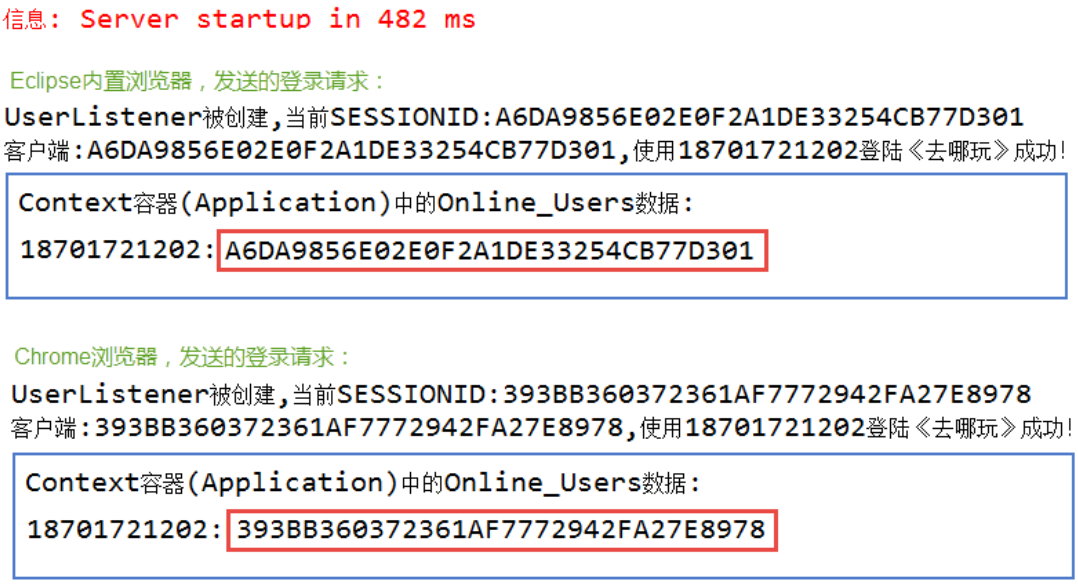


图 3-3-1

1-2. 任务目标：

1-2.1. 操作用户列表，确保登录账号始终匹配发送登录请求的客户端会话ID。

1-3. 任务要求：

1-3.1. 确保登录账号与发送登录请求的客户端会话ID始终保持一致：

1） 客户端A登录成功后，设置用户列表的登录账号X = 客户端A客户端会话ID。

2） 客户端B登录成功后，设置用户列表的登录账号X = 客户端B客户端会话ID。

说明1：登录账号X是用户登录时使用的登录用户名（手机号）。

说明2：通过以上操作可知，用户列表中始终保存着最后一次使用账号X的用户会话ID。

1-3.2. 注销时，移除用户列表中保存的账号X和对应的会话ID。

2. 实现思路：

2-1. 确保登录账号与发送登录请求的客户端会话 **ID** 始终保持一致：

2-1.1. 任务 **2** 已在 **attributeAdded** 方法中获取了：登录账号和会话 **ID**。

2-1.2. 从 **Application** 作用域中获取用户列表。

2-1.3. 从用户列表中获取当前登录账号对应的集合记录（账号 **=** 会话 **ID**）。

2-1.4. 将会话 **ID** 保存入该集合记录中。

2-2. 注销时，移除用户列表中保存的账号 **X**。

2-2.1. 任务 **2** 已在 **attributeRemoved** 方法中获取了：会话 **ID**。

2-2.2. 从 **Session** 中获取登录账号。

2-2.3. 从 **Application** 作用域中获取用户列表。

2-2.4. 移除用户列表中登录账号对应的记录。

3. 推荐步骤：

3-1. 按实现思路，完成本任务。

4. 验证与测试：

4-1. 在**UserListener**的**attributeAdded**方法末尾，调用测试函数：**debugUsers**。

4-2. 在**UserListener**的**attributeRemoved**方法末尾，调用测试函数：**debugUsers**。

4-3. 运行项目工程。

4-4. 测试**1.** 使用客户端**A**登录《去哪玩》旅游网。

4-4.1. 使用**Eclipse**内置浏览器（必须）。

4-4.2. 输入： **http://localhost/PRJ_WTP_JEE_018/** 访问登录页。

4-4.3. 在登录页输入正确的验证码。

4-4.4. 输入正确的用户名、密码【用户名：**18701721202**，密码：**123456**】。

4-4.5. 登录后，应见【任务完成效果截图】第一部分。

4-5. 测试**2.** 使用客户端**B**登录《去哪玩》旅游网。

4-5.1. 使用**Chrome**浏览器（必须）。

4-5.2. 输入： **http://localhost/PRJ_WTP_JEE_018/** 访问登录页。

4-5.3. 在登录页输入正确的验证码。

4-5.4. 输入正确的用户名、密码【用户名：**18701721202**，密码：**123456**】。

4-5.5. 登录后，应见【任务完成效果截图】第二部分。

任务 4、监控账户与会话 ID 的一致性

1. 任务说明：

1-1. 完成效果：

图 3-4-1

1-2. 任务目标：

1-2.1. 利用过滤器监控：登录账户与客户端会话**ID**的一致性。

1-3. 任务要求：

1-3.1. 利用过滤器监控：登录账户与客户端会话**ID**的一致性：

1） 过滤器仅在用户访问隐私资源有效时才判定单态登录（用户已经登录）。

2） 过滤器判断：发送请求的客户端会话**ID**与用户列表中登录账号对应的会话**ID**是否一致。

3） 相同，表示同一用户使用登录账号**X**访问站点，可放行。

4） 不同，表示多个用户使用登录账号**X**访问站点，迫使当前用户下线。

说明**1**：登录账号**X**，为用户登录时使用的登录用户名。

1-3.2. 迫使当前用户下线（其他用户把当前用户挤下线）：

1）跳转到登录页面。

2）在登录页面上显示：“您的账号在其他地方登录，被迫下线”。

2. 实现思路：

2-1. 判断 **SessionID** 是否一致：

2-1.1. 在过滤器 **UserFilter** 内，实现 **SessionID** 的判断。

说明：单态登录的判断逻辑需编写在有效性访问判断逻辑之后。

2-1.2. 获取当前登录账号和 **SessionID**。

2-1.3. 获取 **Application** 作用域，并按登录账号获取用户列表中的 **SessionID**。

2-1.4. 判断两者是否一致，不一致则迫使当前用户下线。

2-2. 迫使当前用户下线（其他用户把当前用户挤下线）。

2-2.1. 设置错误信息："您的账号在其他地方登录，被迫下线！"。

2-2.2. 将错误信息保存入 **Session**，**key = message**。

2-2.3. 移除 **Session** 中的 **User** 对象（挤下线后，清除 **User** 表示注销）。

2-2.4. 客户端重定向的登录页：**login.jsp**。

3. 推荐步骤：

3-1. 按实现思路，判断**SessionID**是否一致。

3-2. 按实现思路，迫使当前用户下线。

4. 验证与测试：

4-1. 运行项目工程。

4-2. 测试**1.** 使用客户端**A**登录《去哪玩》旅游网。

4-2.1. 使用**Eclipse**内置浏览器（必须）。

4-2.2. 输入： **http://localhost/PRJ_WTP_JEE_018/** 访问登录页。

4-2.3. 在登录页输入正确的验证码。

4-2.4. 输入正确的用户名、密码【用户名：**18701721202**，密码：**123456**】。

4-2.5. 进入首页（此时随意点击任何连接都不会被迫下线）。

4-3. 测试**2.** 使用客户端**B**登录《去哪玩》旅游网。

4-3.1. 使用**Chrome**浏览器（必须）。

4-3.2. 输入： **http://localhost/PRJ_WTP_JEE_018/** 访问登录页。

4-3.3. 在登录页输入正确的验证码。

4-3.4. 输入正确的用户名、密码【用户名：**18701721202**，密码：**123456**】。

4-3.5. 进入首页。

4-4. 测试**3.** 迫使用户**A**下线。

4-4.1. 用户**A**在**Eclipse**浏览器中，单击【个人中心】下拉菜单中的：个人资料。

4-4.2. 应见【任务完成效果截图】。

作者：**Roger.Huang**