01_Servlet监听器概述

- 什么是监听器
  - 监听器就是一个实现了特定接口的Java类.这个Java类用来监听另一个Java类的方法调用或者属性改变.当被监听的对象的发生上述的事件后.监听器某个方法就会立即执行.
- 什么是Servlet监听器
  - 在Servlet的规范中定义了多种类型的监听器.监听器用来分别监听ServletContext,HttpSession,ServletRequest三个域对象.
- Servlet监听器分类
  - Servlet的规范中提供了8个监听器
  - 按功能进行划分将其分成三类:
    - 一类 :监听三个域对象的创建和销毁的监听器.
    - 二类 :监听三个域对象的属性变更的监听器.(属性添加,属性移除,属性替换).
    - 三类 :监听HttpSession对象中的JavaBean的状态的改变.(绑定,解除绑定,钝化,活化)
- Servlet监听器的开发步骤
  - web.xml方式
    - 自定义监听器类
    - 将自定义监听器类配置到工程的web.xml中
  - @WebListener注解
    - 在自定义监听器上加上注解

02_一类监听器

- 监听三个域对象的创建和销毁的监听器
- ServletContextListener监听器
  - 监听ServletContext域对象的创建和销毁.
  - ServletContext何时创建和销毁?
    - 创建
      - 服务器启动的时候创建ServletContext对象.
    - 销毁
      - 服务器关闭的时候.
- HttpSessionListener
  - 监听HttpSession域对象创建和销毁.
  - Session何时创建和销毁?
    - 创建
      - 服务器端第一次调用getSession方法时
    - 销毁
      - session过期
      - 调用了invalidate方法
  - 问题
    - 访问一个Servlet会不会创建session对象。
      - 是否有调用getSession方法

        - 访问一个JSP页面会不会创建session对象。
            - 会!
        - 访问一个HTML页面会不会创建session对象
            - 不会!
- ServletRequestListener
    - 监听ServletRequest域对象创建和销毁.
    - ServletRequest对象何时创建和销毁?
    - 创建:
        - 客户的向服务器发送了一次请求,那么服务器就会为这次请求创建一个request对象.
    - 销毁:
        - 当服务器为这次请求作出了响应之后,将request对象销毁了
    - 问题:
        - 访问一个Servlet会不会创建request对象.
            - 会!
        - 访问一个JSP会不会创建request对象
            - 会!
        - 访问一个HTML会不会创建request对象
            - 要看页面中是否有request请求

03_二类监听器

- 监听三个域对象的属性变更的监听器.(属性添加,属性移除,属性替换).
- ServletContextAttributeListener
    - 听ServletContext中的属性变更
- HttpSessionAttributeListener
    - 监听HttpSession中的属性变更.
- ServletRequestAttributeListener
    - 监听ServletRequest中的属性变更.

04_三类监听器

- HttpSessionBindingListener
- 三类监听器特殊的特性
    - 监听HttpSession的对象.
        - 监听向session中存java对象.
    - 这类监听器不需要在web.xml中进行配置:
        - 需要让JavaBean自己感知在session中状态.
- HttpSessionBindingListener
    - 监听HttpSession中的JavaBean的状态(绑定和解除绑定状态)

05_记录登录人数

- 当一个用户登录成功后，将这个用户记录到在线人数中；同时，如果这个用户注销登录就从在线人数中移除。

```
@Override
public void valueBound(HttpSessionBindingEvent httpSessionBindingEvent) {
    System.out.println("绑定了");
    ServletContext servletContext =
httpSessionBindingEvent.getSession().getServletContext();
    Integer count = (Integer) servletContext.getAttribute("count");
    count++;
    servletContext.setAttribute("count",count);

}

@Override
public void valueUnbound(HttpSessionBindingEvent httpSessionBindingEvent) {
    System.out.println("解绑了");
    ServletContext servletContext =
httpSessionBindingEvent.getSession().getServletContext();
    Integer count = (Integer) servletContext.getAttribute("count");
    count--;
    servletContext.setAttribute("count",count);
}
```