# 18.Listener监听器

Servlet技术中三大组件之一 - Servlet Filter Listener

Servlet技术规范中sun预先定义了八大监听器用来监听web开发中对应的事件。

监听器可以用来监听web应用执行过程中相关的事件，来进行对应的处理。

## 18.1.开发监听器步骤

（1）写一个监听器类: 要求必须实现对应的监听器接口，如：

public class MyXxxListener implements XxxListener{...}

（2）注册监听: 在web.xml中配置完成注册

<listener>

<listener-class>cn.tedu.listener.MyXxxListener</listener-class>

</listener>

## 18.2.javaweb中的监听器（八个）

### 18.2.1.监听三大作用域创建和销毁的监听器（三个）

**ServletContextListener**

生命周期：在web应用启动时创建出来，之后一直驻留在内存中唯一的代表当前web应用，直到web应用移除出容易或服务器关闭时，随着web应用的销毁ServletContext对象跟着被销毁。

作用范围：整个web应用范围

主要功能：在整个web应用范围内，整个web应用存活期间，共享数据

**HttpSessionListener**

生命周期：在第一次调用request.getSession()时创建，自杀： session.invalidate，超时：30分钟，意外身亡：服务器非正常关闭

作用范围：整个会话

主要功能：在整个会话范围内，整个会话的生命周期期间，共享数据

钝化：在服务器正常关闭时，仍然在存活期间的session，会被序列化后 保存在tomcat的work目录下，这个过程称之为session的钝化

活化：在服务器正常启动时，会将钝化的session再恢复到内存中，继续 使用，这个过程称之为session的活化

**ServletRequestListener，**

生命周期：请求开始时创建，请求结束时销毁

作用范围：整个请求

主要功能：在请求过程中，在请求范围内，共享数据

这三个接口中都定义了两个方法来处理被监听对象中的创建和销毁事件。

void contextInitialized(ServletContextEvent sce) 创建ServletContext时调用

void contextDestroyed(ServletContextEvent sce) 销毁ServletContext是调用

### 18.2.2.监听三大作用域中属性变化的监听器（三个）

ServletContextAttributeListener

HttpSessionAttributeListener

ServletRequestAttributeListener

这三个接口中都定义了三个方法来处理被监听对象中的属性的增加，删除和替换的事件。

void attributeAdded(ServletContextAttributeEvent scae) 添加属性时调用

void attributeReplaced(ServletContextAttributeEvent scae) 替换属性时调用

void attributeRemoved(ServletContextAttributeEvent scae) 删除属性时调用

### 18.2.3.监听JavaBean在Session域中状态变化的监听器

HttpSessionBindingListener：

使JavaBean 可以感知自己被绑定到 Session 中和从 Session 中删除的监听器。

//当对象被绑定到 HttpSession 对象中时触发

void valueBound(HttpSessionBindingEvent event)

//当对象从 HttpSession 对象中解除绑定时触发

void valueUnbound(HttpSessionBindingEvent event)

HttpSessionActivationListener：

使javabean可以感知自己在session域中随着session被钝化、活化的状态变化的监听器

//当 HttpSession 对象中的对象将要被钝化时触发

sessionWillPassivate(HttpSessionBindingEvent event)

//当 HttpSession 对象中的对象将要被活化时触发（注意：javabean对象如果需要被活化，需要实现序列化接口：Serializable）

void sessionDidActive(HttpSessionBindingEvent event)方法。

这两个监听器比较特殊，不需要单独写类来实现，也不需要在web.xml中进行配置，只需要让javabean自己来实现即可。

代码示例：

|  |
| --- |
| **public** **class** Person **implements** HttpSessionBindingListener,HttpSessionActivationListener,Serializable{  **private** String name;  **private** **int** age;  **public** Person(String name, **int** age) {  **this**.name = name;  **this**.age = age;  }  **public** Person() {  }  **public** String getName() {  **return** name;  }  **public** **void** setName(String name) {  **this**.name = name;  }  **public** **int** getAge() {  **return** age;  }  **public** **void** setAge(**int** age) {  **this**.age = age;  }  **public** **void** valueBound(HttpSessionBindingEvent event) {  System.*out*.println("Person感知到自己被存入session中了。。。。");  }  **public** **void** valueUnbound(HttpSessionBindingEvent event) {  System.*out*.println("Person感知到自己移出session中了。。。。");  }  **public** **void** sessionWillPassivate(HttpSessionEvent se) {  System.*out*.println("Person感知到自己被钝化了。。。。");  }  **public** **void** sessionDidActivate(HttpSessionEvent se) {  System.*out*.println("Person感知到自己被活化了。。。。");  }  } |

## 18.3.EasyMall中监听器的应用

代码如下：

|  |
| --- |
| 在src下创建包：cn.tedu.listener，创建类：MyServletContextListener，实现ServletContextListener接口，并在contextInitialized方法中添加如下代码：  **public** **void** contextInitialized(ServletContextEvent sce) {  ServletContext sc = sce.getServletContext();  sc.setAttribute("app", sc.getContextPath());  } |
| 修改所有jsp文件中的获取web应用虚拟路径的el表达式，将：  ${ pageContext.request.contextPath }改为：${ app } |