**监听器用于监听web应用中某些对象、信息的创建、销毁、增加，修改，删除等动作的发生时，然后作出相应的响应处理。当范围对象的状态发生变化的时候，服务器自动调用监听器对象中的方法。常用于统计在线人数和在线用户，系统加载时进行信息初始化，统计网站的访问量等等。**

**使用Listener步骤**

**①通过实现具体接口创建实现类（可实现多个监听器接口）**

**②配置实现类成为监听器，或者将注册监听器,有两种配置方式：**

**通过web.xml方式配置，代码如下：**

**<listener>**

**<listener-class>com.zrgk.listener.MyListener</lisener-class>**

**</listener>**

**直接用@WebListener注解修饰实现类**

**常用的Web事件的监听接口如下：**

**ServletContextListener：用于监听Web的启动及关闭**

**ServletContextAttributeListener：用于监听ServletContext范围内属性的改变**

**ServletRequestListener：用于监听用户请求**

**ServletRequestAttributeListener：用于监听ServletRequest范围属性的改变**

**HttpSessionListener：用于监听用户session的开始及结束**

**HttpSessionAttributeListener：用于监听HttpSession范围内的属性改变**

**1. ServletContextListener**

**①该接口用于监听Web应用的启动与关闭**

**②该接口的两个方法：**

**contextInitialized(ServletContextEvent event); // 启动web应用时调用**

**contextDestroyed(ServletContextEvent event); // 关闭web应用时调用**

**③如何获得application对象：**

**ServletContext application = event.getServletContext();**

**代码片段:**

**@WebListener**

**public class MyServetContextListener implements ServletContextListener{**

**//web应用关闭时调用该方法**

**@Override**

**public void contextDestroyed(ServletContextEvent event) {**

**ServletContext application = event.getServletContext();**

**String userName = application.getInitParameter("userName");**

**System.out.println("关闭web应用的用户名字为："+userName);**

**}**

**//web应用启动时调用该方法**

**@Override**

**public void contextInitialized(ServletContextEvent event) {**

**ServletContext application = event.getServletContext();**

**String userName = application.getInitParameter("userName");**

**System.out.println("启动web应用的用户名字为："+userName);**

**}**

**}**

**2. ServletContextAttributeListener**

**①该接口 用于监听ServletContext范围（application）内属性的改变。**

**②该接口的三个方法：**

**attributeAdded(ServletContextAttributeEvent event); // 当把一个属性存进application时触发**

**attributeRemoved(ServletContextAttributeEvent event); // 当把一个属性从application删除时触发**

**attributeReplaced(ServletContextAttributeEvent event); // 当替换application内的某个属性值时触发**

**③如何获得application对象：**

**ServletContext application = event.getServletContext();**

**示例**

**public class MyServletContenxtAttributeListener implements ServletContextAttributeListener{**

**//当程序向application范围新增属性时触发此方法**

**public void attributeAdded(ServletContextAttributeEvent scab) {**

**// TODO Auto-generated method stub**

**ServletContext application=scab.getServletContext();**

**//获取新增的属性名与值**

**String name=scab.getName();**

**Object value=scab.getValue();**

**System.out.println(application+"范围内新增了name值为="+name+"value值="+value);**

**}**

**//当程序向application范围移除属性时触发此方法**

**public void attributeRemoved(ServletContextAttributeEvent scab) {**

**// TODO Auto-generated method stub**

**ServletContext application=scab.getServletContext();**

**//获取移除的属性名与值**

**String name=scab.getName();**

**Object value=scab.getValue();**

**System.out.println(application+"范围内移除了name值为="+name+"value值="+value);**

**}**

**//当程序向application范围替换属性时触发此方法**

**public void attributeReplaced(ServletContextAttributeEvent scab) {**

**// TODO Auto-generated method stub**

**ServletContext application=scab.getServletContext();**

**//获取替换的属性名与值**

**String name=scab.getName();**

**Object value=scab.getValue();**

**System.out.println(application+"范围内替换了name值为="+name+"value值="+value);**

**}**

**}**

**3. ServletRequestListener与ServletRequestAttributeListener**

**ServletRequestListener用于监听用户请求。**

**ServletRequestAttributeListener用于监听request范围内属性的变化。**

**ServletRequestListener两个需要实现的方法**

**requestInitialized(ServletRequestEvent event); //用户请求到达、被初始化时触发**

**requestDestroyed(ServletRequestEvent event); // 用户请求结束、被销毁时触发**

**ServletRequestAttributeListener三个需要实现的方法**

**attributeAdded(ServletRequestAttributeEvent event); // 向request范围内添加属性时触发**

**attributeRemoved(ServletRequestAttributeEvent event); // 从request范围内删除某个属性时触发**

**attributeReplaced(ServletRequestAttributeEvent event); // 替换request范围内某个属性值时触发**

**获取reqeust对象**

**HttpServletRequest req = (HttpServletRequest)event.getServletRequest();**

**代码片**

**@WebListener**

**public class MyRequestListener implements ServletRequestListener,ServletRequestAttributeListener{**

**//用户请求结束、被销毁时触发**

**@Override**

**public void requestDestroyed(ServletRequestEvent event) {**

**HttpServletRequest req = (HttpServletRequest) event.getServletRequest();**

**String ip = req.getRemoteAddr();**

**System.out.println("IP为:"+ip+"的用户发送到"+req.getRequestURI()+"的请求结束");**

**}**

**//用户请求到达、被初始化时触发**

**@Override**

**public void requestInitialized(ServletRequestEvent event) {**

**HttpServletRequest req = (HttpServletRequest) event.getServletRequest();**

**String ip = req.getRemoteAddr();**

**System.out.println("IP为:"+ip+"的用户发送到"+req.getRequestURI()+"的请求被初始化");**

**}**

**//向request范围内添加属性时触发**

**@Override**

**public void attributeAdded(ServletRequestAttributeEvent event) {**

**String name = event.getName();**

**Object val = event.getValue();**

**System.out.println("向request范围内添加了名为："+name+"，值为："+val+"的属性");**

**}**

**//删除request范围内某个属性时触发**

**@Override**

**public void attributeRemoved(ServletRequestAttributeEvent event) {**

**//...**

**}**

**//替换request范围内某个属性值时触发**

**@Override**

**public void attributeReplaced(ServletRequestAttributeEvent event) {**

**// ...**

**}**

**}**

**4. HttpSessionListener与HttpSessionAttributeListener**

**HttpSessionListener监听用户session的开始与结束。**

**HttpSessionAttributeListener监听HttpSession范围（session）内的属性的改变。**

**HttpSessionListener要实现的方法：**

**sessionCreated(HttpSessionEvent event); // 用户与服务器的会话开始、创建时触发**

**sessionDestroyed(HttpSessionEvent event); // 用户与服务器的会话结束时触发**

**HttpSessionAttributeListener要实现的方法：**

**attributeAdded(HttpSessionBindingEvent event) ; // 向session范围内添加属性时触发**

**attributeRemoved(HttpSessionBindingEvent event); // 删除session范围内某个属性时触发**

**attributeReplaced(HttpSessionBindingEvent event); // 替换session范围内某个属性值时触发**

**如何得到session对象**

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

**代码片**

**@WebListener**

**public class MySessionListener implements HttpSessionListener,HttpSessionAttributeListener {**

**//建立session会话时触发**

**@Override**

**public void sessionCreated(HttpSessionEvent event) {**

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

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

**System.out.println("建立了会话，会话ID为："+sessionId);**

**}**

**@Override**

**public void sessionDestroyed(HttpSessionEvent event) {**

**// ...**

**}**

**//向session范围内添加属性时触发**

**@Override**

**public void attributeAdded(HttpSessionBindingEvent event) {**

**String name = event.getName();**

**Object val = event.getValue();**

**System.out.println("向session范围内添加了名为："+name+",值为："+val+"的属性");**

**}**

**@Override**

**public void attributeRemoved(HttpSessionBindingEvent event) {**

**// ...**

**}**

**@Override**

**public void attributeReplaced(HttpSessionBindingEvent event) {**

**// ...**

**}**

**}**