**java基础篇---Servlet监听器**

2015-02-06      [**0**](javascript:void(0);)[个评论](http://www.2cto.com/kf/201502/375451.html#SOHUCS)

[收藏](javascript:;)    [[http://www.2cto.com/statics/images/dl.jpg](http://www.2cto.com/index.php?m=member&c=content&a=publish&modid=1&siteid=1)我要投稿](http://www.2cto.com/index.php?m=member&c=content&a=publish&modid=1&siteid=1)

我就写一下关于监听器的具体应用：

功能是负责监听WEB的各种操作，当相关的事件触发之后将产生事件，并对此事件进行处理，在WEB中可以对application、session、request三种操作进行监听。

对application监听:

对application监听，实际上就是对ServletContext（Servlet上下文）监听，主要使用以下两个接口：ServletContextListener ,ServletContextAttributeListener

package com.oumyye.监听器;

import javax.servlet.ServletContextEvent;

import javax.servlet.ServletContextListener;

public class ServletContextListenerDemo implements ServletContextListener {

    public void contextInitialized(ServletContextEvent event) {    // 上下文初始化时触发

        System.out.println("\*\* 容器初始化 --> "

                + event.getServletContext().getContextPath());

    }

    public void contextDestroyed(ServletContextEvent event) {        // 上下文销毁时触发

        System.out.println("\*\* 容器销毁 --> "

                + event.getServletContext().getContextPath());

    }

}

web.xml配置

<listener>

    <listener-class>

        com.oumyye.监听器.ServletContextListenerDemo

    </listener-class>

</listener>

对session监听

在监听器中，针对于session的监听操作提供了三个接口：HttpSessionListener，HttpSessionAttributeListener，HttpSessionBindingListener

session状态监听：HttpSessionListener接口

当需要对创建或销毁session的操作进行监听的时候，可以实现javax.servlet.http.HttpSessionListener接口，此接口定义的方法如下：public void sessionCreated(HttpSessionEvent se)，public void sessionDestroyed(HttpSessionEvent se)

当session创建或销毁后，将产生HttpSessionEvent事件，此事件定义的操作如下：public HttpSession getSession()

对session监听

package com.oumyye.监听器;

import javax.servlet.http.HttpSessionEvent;

import javax.servlet.http.HttpSessionListener;

public class HttpSessionListenerDemo implements HttpSessionListener {

    public void sessionCreated(HttpSessionEvent event) {        // 创建session触发

        System.out.println("\*\* SESSION创建，SESSION ID = " + event.getSession().getId());

    }

    public void sessionDestroyed(HttpSessionEvent event) {        // 销毁session触发

        System.out.println("\*\* SESSION销毁，SESSION ID = " + event.getSession().getId());

    }

}

web.xml配置

    <listener>

        <listener-class>

                <listener>

        <listener-class>

            com.oumyye.监听器.HttpSessionListenerDemo

        </listener-class>

    </listener>

session销毁的操作

当一个新用户打开一个动态页时，服务器是会为新用户分配session，并且触发HttpSessionListener接口中的sessionCreated()事件，但是在用户销毁时却有两种不同的方式来触发sessionDestroyed()事件：

方式一：调用HttpSession接口的invalidate()方法，让一个session失效。

方式二：超过了配置的session超时时间，session超时时间，可以直接在项目中的web.xml配置。

session属性监听：HttpSessionAttributeListener接口

在session监听中也可以对session的属性操作进行监听，这一点与监听上下文属性的道理是一样的，要对session的属性操作监听，则可以使用javax.servlet.http.HttpSessionAttributeListener接口完成，此接口的方法如下：

public void attributeAdded(HttpSessionBindingEvent se),

public void attributeRemoved(HttpSessionBindingEvent se),

public void attributeReplaced(HttpSessionBindingEvent se)

当进行属性操作时，将根据属性的操作触发HttpSessionAttributeListener接口中的方法，每个操作方法都将产生HttpSessionBindingEvent事件，此事件定义操作如下：

public HttpSession getSession(),

public String getName(),

public Object getValue()

对session的属性操作监听 :

package com.oumyye.监听器;

import javax.servlet.http.HttpSessionAttributeListener;

import javax.servlet.http.HttpSessionBindingEvent;

public class HttpSessionAttributeListenerDemo implements

        HttpSessionAttributeListener {

    public void attributeAdded(HttpSessionBindingEvent event) {// 属性增加时调用

        System.out.println(event.getSession().getId() + "，增加属性 --> 属性名称："

                + event.getName() + "，属性内容：" + event.getValue());

    }

    public void attributeRemoved(HttpSessionBindingEvent event) {// 属性删除时调用

        System.out.println(event.getSession().getId() + "，删除属性 --> 属性名称："

                + event.getName() + "，属性内容：" + event.getValue());

    }

    public void attributeReplaced(HttpSessionBindingEvent event) {// 属性替换时调用

        System.out.println(event.getSession().getId() + "，替换属性 --> 属性名称："

                + event.getName() + "，属性内容：" + event.getValue());

    }

}

web.xml配置

<listener>

    <listener-class>

        com.oumyye.监听器.HttpSessionAttributeListenerDemo

    </listener-class>

</listener>

session属性监听：HttpSessionBindingListener接口

在session监听中也可以对session的属性操作进行监听，这一点与监听上下文属性的道理是一样的，要对session的属性操作监听，则可以使用javax.servlet.http.HttpSessionAttributeListener接口完成：

public void attributeAdded(HttpSessionBindingEvent se)

public void attributeRemoved(HttpSessionBindingEvent se)

public void attributeReplaced(HttpSessionBindingEvent se)

当进行属性操作时，将根据属性的操作触发HttpSessionAttributeListener接口中的方法，每个操作方法都将产生HttpSessionBindingEvent事件

public HttpSession getSession()

public String getName()

public Object getValue()

对session的属性操作监听

package com.oumyye.监听器;

import javax.servlet.http.HttpSessionAttributeListener;

import javax.servlet.http.HttpSessionBindingEvent;

public class HttpSessionAttributeListenerDemo implements

        HttpSessionAttributeListener {

    public void attributeAdded(HttpSessionBindingEvent event) {// 属性增加时调用

        System.out.println(event.getSession().getId() + "，增加属性 --> 属性名称："

                + event.getName() + "，属性内容：" + event.getValue());

    }

    public void attributeRemoved(HttpSessionBindingEvent event) {// 属性删除时调用

        System.out.println(event.getSession().getId() + "，删除属性 --> 属性名称："

                + event.getName() + "，属性内容：" + event.getValue());

    }

    public void attributeReplaced(HttpSessionBindingEvent event) {// 属性替换时调用

        System.out.println(event.getSession().getId() + "，替换属性 --> 属性名称："

                + event.getName() + "，属性内容：" + event.getValue());

    }

}

web.xml同上

session属性监听：HttpSessionBindingListener接口

在WEB里也提供了一个javax.servlet.http.HttpSessionBindingListener接口，通过此接口实现的监听程序可以不用配置而直接使用，此接口定义的方法如下：

public void valueBound(HttpSessionBindingEvent event)

public void valueUnbound(HttpSessionBindingEvent event)

对request监听

在Servlet 2.4之后增加了对request操作的监听，主要使用ServletRequestListener、ServletRequestAttributeListener两个接口。

请求状态监听：ServletRequestListener接口

当需要对用户的每次请求进行监听的时候，可以使用javax.servlet.ServletRequestListener接口，此接口定义方法如下：

public void requestInitialized(ServletRequestEvent sre)

public void requestDestroyed(ServletRequestEvent sre)

ServletRequestListener接口一旦监听到事件之后，将产生ServletRequestEvent的事件处理对象，此事件类定义的操作方法如下：

public ServletRequest getServletRequest()

public ServletContext getServletContext()

对用户请求request监听

package com.oumyye.监听器;

import javax.servlet.ServletRequestEvent;

import javax.servlet.ServletRequestListener;

public class ServletRequestListenerDemo implements ServletRequestListener {

    public void requestInitialized(ServletRequestEvent event) {

        System.out.println("\*\* request初始化。http://"

                + event.getServletRequest().getRemoteAddr()

                + event.getServletContext().getContextPath());

    }

    public void requestDestroyed(ServletRequestEvent event) {

        System.out.println("\*\* request销毁。http://"

                + event.getServletRequest().getRemoteAddr()

                + event.getServletContext().getContextPath());

    }

}

web.xml配置

<listener>

    <listener-class>

        com.oumyye.监听器.ServletRequestListenerDemo

    </listener-class>

</listener>

request属性监听：ServletRequestAttributeListener接口

对request范围属性的监听可以使用javax.servlet.ServletRequestAttributeListener接口，此接口定义的方法如下所示：

public void attributeAdded(ServletRequestAttributeEvent srae)

public void attributeReplaced(ServletRequestAttributeEvent srae)

public void attributeRemoved(ServletRequestAttributeEvent srae)

加入监听器之后request属性的操作都会产生ServletRequestAttributeEvent事件，此事件的定义的方法如下：

public String getName()

public Object getValue()

监听request属性操作

package com.oumyye.监听器;

import javax.servlet.ServletRequestAttributeEvent;

import javax.servlet.ServletRequestAttributeListener;

public class ServletRequestAttributeListenerDemo implements

        ServletRequestAttributeListener {

    public void attributeAdded(ServletRequestAttributeEvent event) {

        System.out.println("\*\* 增加request属性 --> 属性名称：" + event.getName()

                + "，属性内容：" + event.getValue());

    }

    public void attributeRemoved(ServletRequestAttributeEvent event) {

        System.out.println("\*\* 删除request属性 --> 属性名称：" + event.getName()

                + "，属性内容：" + event.getValue());

    }

    public void attributeReplaced(ServletRequestAttributeEvent event) {

        System.out.println("\*\* 替换request属性 --> 属性名称：" + event.getName()

                + "，属性内容：" + event.getValue());

    }

}

web.xml配置

<listener>

    <listener-class>

        com.oumyye.监听器.ServletRequestAttributeListenerDemo

    </listener-class>

</listener>

监听器实例 —— 在线人员统计

在线人员列表是一个较为常见的功能，每当用户登陆成功之后，就会在列表中增加此用户名称，这样就可以知道当前在线的用户有那些了，这个功能在WEB中只能靠监听器实现。

使用接口

要完成在线用户列表的监听器，需要使用如下几个接口：

ServletContextListener接口：在上下文初始化时设置一个空的集合到application之中；

HttpSessionAttributeListener接口：用户增加session属性时，表示新用户登陆，从sesion中取出此用户的登陆名，之后将此用户保存在列表之中；

HttpSessionListener接口：当用户注销（手工注销、会话超时）将此用户列表中删除此用户。