# 1概念

监听器:主要是用来监听特定对象的创建或销毁,属性的变化,是一个普通实现特定接口的普通java类

对象:

1. 自己创建自己用(不用监听)
2. 别人创建自己用(需要监听)

Servlet中需要监听的对象:

request/session/servletContext

分别对应的是request监听器,session相关监听器,servletContext监听器

# 2.监听器接口

## 2.1监听对象创建/销毁的监听接口

### 2.1.1ServletRequestListener接口

1. void requestDestoryed(ServletRequestEvent sre)
2. void requestInitialized(ServletRequestEvent sre)

### 2.1.2HttpSessionListener接口

1. void sessionCreated(HttpSessionEvent se)
2. void sessionDestroyed(HttpSessionEvent se)

### 2.1.3ServletContextListener接口

1. void contextDestroyed(ServletContextEvent sce)
2. void contextInitialized(ServletContextEvent sce)

## 2.2监听对象属性的变化

### 2.2.1ServletRequestAtrributeListener接口

1) void attributeAdded(ServletRequestAttributeEvent srae) //添加

2) void attributeRemoved(ServletRequestAttributeEvent srae) //移除

3) void attributeReplaced(ServletRequestAttributeEvent srae) //修改

### 2.2.2HttpSessionAttributeListener接口

1) void attributeAdded(HttpSessionBindingEvent se)

2) void attributeRemoved(HttpSessionBindingEvent se)

3) void attributeReplaced(HttpSessionBindingEvent se)

### 2.2.3ServletContextAttributeListener接口

1) void attributeAdded(ServletContextAttributeEvent scab)

2) void attributeRemoved(ServletContextAttributeEvent scab)

3) void attributeReplaced(ServletContextAttributeEvent scab)

## 2.3其他监听器

### 2.3.1HttpSessionBindingListener接口

监听对象绑定到session上的事件

1. void valueBound(HttpSessionBindingEvent event)
2. void valueUnbound(HttpSessionBindingEvent event)

### 2.3.2HttpSessionActivationListener接口

监听session序列化及反序列化的事件

1) void sessionDidActivate(HttpSessionEvent se);

2) void SessionWillPassivate(HttpSessionEvent se);

# 3.监听器开发步骤

1)写一个普通java类,实现相关接口

2)配置web.xml

## 3.1ServletRequestListener范例:

MyRequestListener.java代码:

|  |
| --- |
| import javax.servlet.ServletRequestEvent;  import javax.servlet.ServletRequestListener;  public class MyRequestListener implements ServletRequestListener{  //对象销毁  @Override  public void requestDestroyed(ServletRequestEvent servletRequestEvent) {  //获取request中存放的数据  String str = (String)servletRequestEvent.getServletRequest().getAttribute("cn");  System.out.println(str);  System.out.println("requestDestroyed");  }  //对象创建  @Override  public void requestInitialized(ServletRequestEvent servletRequestEvent) {  System.out.println("requestInitialized");  }  } |

web.xml中配置:

|  |
| --- |
| <listener>  <listener-class>listener.MyRequestListener</listener-class>  </listener> |

## 3.2session属性的监听

代码:

|  |
| --- |
| /\*\*  \* 监听session对象属性的变化  \*/  public class MySessionAttrListener implements HttpSessionAttributeListener{  //属性添加  @Override  public void attributeAdded(HttpSessionBindingEvent httpSessionBindingEvent) {  //获取添加的属性  HttpSession session = httpSessionBindingEvent.getSession();  String name = (String)session.getAttribute("userName");  System.out.println("添加的属性值为:" + name);  }  //属性移除  @Override  public void attributeRemoved(HttpSessionBindingEvent httpSessionBindingEvent) {  System.out.println("属性被移除:");  }  //属性替换  @Override  public void attributeReplaced(HttpSessionBindingEvent httpSessionBindingEvent) {  System.out.println("属性被替换");  HttpSession session = httpSessionBindingEvent.getSession();  Object obj1 = httpSessionBindingEvent.getValue();  Object obj2 = session.getAttribute("userName");  System.out.println("原来的值:" + obj1);  System.out.println("新值:" + obj2);  }  } |

web.xml

|  |
| --- |
| <listener>  <listener-class>listener.MySessionAttrListener</listener-class>  </listener> |

01.jsp

|  |
| --- |
| <%@ page contentType="text/html;charset=UTF-8" language="java" %>  <html>  <head>  <title>Title</title>  </head>  <body>  test!  <%  session.setAttribute("userName","linwei");  // session.removeAttribute("userName");  session.setAttribute("userName","jack");  %>  </body>  </html> |

## 3.3HttpSessionBindingListener

监听对象绑定/解除绑定到session上的事件

步骤:对象实现接口,再把对象绑定/接触绑定到session上就会触发监听代码

首先建一个普通java类Admin.java:

|  |
| --- |
| /\*\*  \* 监听此对象绑定到session上的过程,需要实现session特定接口  \*/  public class Admin implements HttpSessionBindingListener{  private int id;  private String name;  public Admin(int id,String name){  this.id = id;  this.name = name;  }  public int getId() {  return id;  }  public void setId(int id) {  this.id = id;  }  public void setName(String name) {  this.name = name;  }  //对象放入session  @Override  public void valueBound(HttpSessionBindingEvent httpSessionBindingEvent) {  System.out.println("Admin对象已经放入session");  }  //对象从session中移除  @Override  public void valueUnbound(HttpSessionBindingEvent httpSessionBindingEvent) {  System.out.println("Admin对象从session中移除");  }  } |

**这时不需要在web.xml中进行配置,不然报错**

**因为监听的对象是自己创建的对象,不是服务器对象,所以不需要在web.xml中配置**

01.jsp中:

|  |
| --- |
| <%  session.setAttribute("userInfo",new Admin(1,"linwei"));  session.removeAttribute("userInfo");  %> |