8个监听者

1. 上下文属性监听者javax.servlet.ServletContextAttributeListener
   1. 场景：想知道一个Web应用上下文中是否增加、删除或者替换了一个属性
   2. 事件类型：ServletContextAttributeEvent
   3. 接口方法：attributeAdded、attributeRemoved、attributeReplaced
2. 会话监听者 javax.servlet.http.HttpSessionListener
   1. 场景：想知道有多少个并发用户，跟踪活动的会话
   2. 事件类型：HttpSessionEvent
   3. 接口方法：sessionCreated、sessionDestroyed
3. 请求监听者 javax.servlet.ServletRequestListener
   1. 场景：每次请求到来的时候你都想知道，以便建立日志记录
   2. 事件类型：ServletRequestEvent
   3. 接口方法：requestInitialized、requestDestroyed
4. 请求属性监听者 javax.servlet.ServletAttributeListener
   1. 场景：想知道什么时候增加、删除或替换一个请求属性
   2. 事件类型：ServletRequestAttributeEvent
   3. 接口方法：attributeAdded、attributeRemoved、attributeReplaced
5. 会话绑定监听者 javax.servlet.http.HttpSessionBindingListener
   1. 场景：有一个**属性类**，且希望这个类型的对象都绑定在一个会话或从会话删除时得到通知，这个接口是针对想要绑定到session的属性的！！！
   2. 事件类型：HttpSessionBindingEvent
   3. 接口方法：valueBound、valueUnbound
6. 会话属性监听者 javax.servlet.HttpSessionAttributeListener
   1. 场景：你想知道什么时候增加、删除或替换一个**会话属性**
   2. 事件类型：HttpSessionBindingEvent（和5相同）
   3. 接口方法：attributeAdded、attributeRemoved、attributeReplaced
7. 上下文监听者 javax.servlet.ServletContextListener
   1. 场景：你想知道是否创建或撤销了一个上下文
   2. 事件类型：ServletContextEvent
   3. 接口方法:contextInitialized、contextDestroyed
8. 会话激活监听者 javax.servlet.http.HttpSessionActicationListener
   1. 场景：你有一个**属性类**，而且希望此类对象绑定的会话迁移到另一个JVM时得到通知
   2. 事件类型：HttpSessionEvent（和2相同）
   3. 接口方法：sessionDidActive、sessionWillPassivate

补充：

对应**属性对象**的监听器：HttpSessionBindingListener和HttpSessionActivationListener

**这两者都不用在DD配置**

1. HttpSessionAttributeListener与HttpSessionBindingListener的区别
   1. 前者只想知道会话中何时增加、删除或替换了某种类型的**属性**，但是后者让属性本身知道它何时增加到一个会话中，或者何时从会话删除
   2. HttpSessionBindingListener示例：

Import javax.servlet.http.\*;

public class Dog implements HttpSessionBindingListener {

private String breed;

public Dog(String breed){

this.breed = breed;

}

public String getBreed(){

return this.breed;

}

public void valueBound(HttpSessionBindingEvent event){

//如果Dog对象知道自己在一个会话中，就运行这些代码

}

public void valueUnbound(HttpSessionBindingEvent event){

//如果Dog对象知道自己脱离了一个会话，就运行这些代码

}

}

解析：这一次Dog属性也是一个监听者，绑定监听者并不在DD中注册，而是会自动进行

用途：保持数据库记录和Customer信息同步

Q：什么时候使用？

A：只要有一个Customer对象增加到会话session.setAttribute(cus)，就应该用数据库中相应记录的客户数据来刷新Customer对象的字段，所以valueBound就是“用数据库的最新数据来加载我”，因为上一次用完之后可能有变化，valueUnbound就是“用customer对象字段的值来更新数据库”