1. 监听器的接口主要在javax.servlet和javax.servlet.http的包中，有以下接口：

•javax.servlet.ServletContextListener:它能够响应ServletContext生命周期事件，它提供了ServletContext创建之后和ServletContext关闭之前的会被调用的方法。

•Javax.servlet.ServletContextAttributeListener:它能够响应ServletContext范围的属性添加、删除、替换事件。

•Javax.servlet.http.HttpSessionListener: 它能够响应HttpSession的创建、超时和失效事件。

•Javax.servlet.http.HttpSessionAttributeListener：它能响应HttpSession范围的属性添加、删除、替换事件。

•Javax.servlet.http.HttpSessionActivationListener: 它在一个HttpSession激活或者失效时被调用。

•Javax.servlet.http.HttpSessionBindingListener: 可以实现这个接口来保存HttpSession范围的属性。当有属性从HttpSession添加或删除时，HttpSessionBindingListener接口能够做出响应。

•Javax.servlet.ServletRequestListener:它能够响应一个ServletRequest的创建或删除。

•Javax.servlet.ServletRequestAttributeListener: 它能响应ServletRequest范围的属性值添加、删除、修改事件。

•Javax.servlet.AsyncListener: 一个用于异步操作的监听器。

1. 注册监听器的方法：

Servlet3.0

@WebListener  
public class ListenerClass implements ServletContextListener{  
}

Servlet3.1

<listener>  
 <listener-class>fully-qualified listener class</listener-class>  
</listener>

1. ServletContextListener

作用：

•当ServletContext初始化时，容器会调用所有注册ServletContextLIstener的contextInitialize方法：void contextInitialized(ServletContextEvent event)

•当ServletContext 要销毁时，容器会调用所有注册ServletContextListener的contextDestroyed方法：void contextDestroyed(ServletContextEvent event)

•cntextInitialized和contextDestroyed方法都会从容器获取到一个ServletContextEvent（是java.util.EventObject的子类）,它定义了一个访问ServletContext的getServletContext方法：ServletContext getServletContext()

1. ServletContextAttributeListener

作用：

•当ServletContext范围的属性被添加、删除或者替换时，ServletContextAttributeListener接口的实现类会接收到消息。

void attributeAdded(ServletContextAttributeEvent event)

void attributeRemoved(ServletContextAttributeEvent event)

void attributeReplaced(ServletContextAttributeEvent event)

•ServletContextAttributeEvent类继承自ServletContextEvent，并增加了下面两个用于获取该属性名称和值的方法：

String getName()

String getValue()

1. HttpSessionListener

•当一个HttpSession创建或者销毁时，容器会通知所有的HttpSessionListener监听器。HttpSessionListener接口有两个方法：

void sessionCreated(HttpSessionEvent event)

void sessionDestroyed(HttpSessionEvent event)

•HttpSessionEvent继承自java.util.EventObject,有如下方法：

HttpSession getSession()

注：可以用来监听连接的用户数量

1. HttpSessionAttributeListener

•当HttpSession范围属性的添加、删除和替换。

void attributeAdded(HttpSessionBindingEvent event)

void attributeRemoved(HttpSessionBindingEvent event)

void attributeReplaced(HttpSessionBindingEvent event)

•HttpSessionBingdingEvent是HttpSessionEvent的子类，可以使用：

String getName()

Object getValue()方法获取这个属性的名称和值。

1. HttpSessionActivationListener

在分布式环境下，会用多个容器来进行负载均衡，有可能需要将session保存起来，在容器之间传递。例如当一个容器内存不足时，会把很少用的对象转存到其他容器上。这时候，容器就会通知所有HttpSessionActivationListener接口的实现类。

•HttpSessionActivationLIstenere接口有两个方法：

void sessionDidActivate(HttpSessionEvent event): 当HttpSession被转移到其他容器后，sessionDidActivate方法会被调用。容器将一个HttpSessionEvent传递到方法里，可以从这个对象获得HttpSession。

void sessionWillPassivate(HttpSessionEvent event)：当一个HttpSession将要失效时，容器将会调用sessionWillPassivate方法。容器将一个HttpSessionEvent传递到方法里，可以从这个对象获取HttpSession。

1. HttpSessionBindingListener

当有属性绑定或者解绑到HttpSession上时，HttpSessionBindingListener监听器会被调用。

所谓对session进行数据绑定，就是调用session.setAttribute()把HttpSessionBindingListener保存进session中。我们在LoginServlet.java中进行这一步。

valueBound()和valueUnbound()，分别对应数据绑定，和取消绑定两个事件。

在触发session.invalidate()时ValueUnbound()方法被触发。

这就是HttpSessionBindingListener和HttpSessionListener之间的最大区别： HttpSessionListener只需要设置到web.xml中就可以监听整个应用中的所有session。 HttpSessionBindingListener必须实例化后放入某一个session中，才可以进行监听。

从监听范围上比较，HttpSessionListener设置一次就可以监听所有session，HttpSessionBindingListener通常都是一对一的。

例如：

@WebListener  
public class SessionBindingListener implements HttpSessionBindingListener{  
 String username;  
 public String getUsername() {  
 return username;  
 }  
 public void setUsername(String username) {  
 this.username = username;  
 }  
 @Override  
 public void valueBound(HttpSessionBindingEvent httpSessionBindingEvent) {  
 System.*out*.println(username);  
 }  
 @Override  
 public void valueUnbound(HttpSessionBindingEvent httpSessionBindingEvent) {  
 System.*out*.println("valueUnbound");  
 }  
}

@Override  
 protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {  
 HttpSession session = req.getSession();  
 SessionBindingListener sessionBindingListener = new SessionBindingListener();  
 sessionBindingListener.setUsername("wwwwwwwwwxxxxxx");  
  
 session.setAttribute("sessionBindingListener", sessionBindingListener);  
 }

1. ServletRequestListener

•ServletRequestListener监听器会对ServletRequest的创建和销毁事件进行响应。容器会通过一个池来存放并重复利用多个ServletRequest，ServletRequest的创建是从容器池里被分配出来的时刻开始，而它的销毁时刻是放回容器池里的时间。

•ServletRequestListener接口有两个方法：

void requestInitialized(ServletRequestEvent event)

void requestDestroyed(ServletRequestEvent event)

•可以通过ServletRequestEvent的 ServletRequest getServletRequest()方法获取ServletRequest对象。

可以通过ServletContext getServletContext()方法获取ServletContext对象。

1. ServletRequestAttributeListener

•当一个Servlet范围的属性被添加、删除或替换时，ServletRequestAttributeListener接口会被调用。

void attributeAdded(ServletRequestAttributeEvent event)

void attributeRemoved(ServletRequestAttributeEvent event)

void attributeReplaced(ServletRequestAttributeEvent event)

•ServletRequestAttributeEvent继承自ServletRequestEvent

ServletRequestAttributeEvent提供：

String getName()和Object getValue()两个方法访问属性的名称和值。