**监听器学习**

**问题：**

* Servlet 技术中我们学习了 request 、 session 、

application (Sevlet中称为ServletContext)作用域对象，其主要作用是实现数据的在不同场景中的灵活流转。但是数据的具体流转过程我们是看不到的，比如作用域对象是什么时候创建和销毁的，数据是什么时候存取，改变和删除的。因为具体的流转过程看不到，所以也就无法再指定的时机对数据和对象进行操作。

**解决：**

使用监听器

**概念：**

Servlet 监听器是 Servlet 规范中定义的一种特殊类，用

于监听 ServletContext、HttpSession 和 ServletRequest 等域

对象的创建与销毁事件，以及监听这些域对象中属性发生

修改的事件。

**监听对象：**

Request

Session

Application(ServletContext)

**监听内容：**

创建、销毁、属性改变事件**监听作用：**

在事件发生之前，之后进行一些处理，比如统计在线人数

**使用：**

监听 request

### ServletRequestListener接口

requestInitialized(ServletRequestEvent sre)

形参ServletRequestEvent sre：可以获取当前被监听的request对象，在request对象创建的同时可以给request对象中的资源进行操作

|  |
| --- |
| **package** com.manager.listener;  **import** javax.servlet.ServletRequest; **import** javax.servlet.ServletRequestEvent; **import** javax.servlet.ServletRequestListener; **import** javax.servlet.annotation.WebListener; **import** javax.servlet.annotation.WebServlet; **import** javax.servlet.http.HttpServletRequest;  @WebListener(value = **"MyRequestLisenter"**) **public class** MyRequestLisenter **implements** ServletRequestListener {  @Override  **public void** requestDestroyed(ServletRequestEvent sre) {  System.***out***.println(**"request被销毁了"**);  ServletRequest servletRequest = sre.getServletRequest();  HttpServletRequest req = (HttpServletRequest)servletRequest;  String uname = req.getParameter(**"uname"**);  String pwd = req.getParameter(**"pwd"**);  System.***out***.println(**"截取到:"**+uname);  System.***out***.println(**"截取到:"**+pwd);  req.setAttribute(**"error"**,**"病毒"**); *// req.getRequestDispatcher("请求转到的Servlet").forward(req,resp);//理论上可以做到  //监听器,可以偷摸着截取用户提交的信息,或往系统中偷摸着加入一些数据等等* }   @Override  **public void** requestInitialized(ServletRequestEvent sre) {  System.***out***.println(**"request被初始化了"**);  } } |

### ServletRequestAttributeListener

|  |
| --- |
| **package** com.manager.servlet;  **import** com.google.gson.Gson; **import** com.manager.pojo.User; **import** com.manager.util.DBUtil;  **import** javax.servlet.ServletException; **import** javax.servlet.annotation.WebServlet; **import** javax.servlet.http.HttpServlet; **import** javax.servlet.http.HttpServletRequest; **import** javax.servlet.http.HttpServletResponse; **import** java.io.IOException; **import** java.sql.PreparedStatement; **import** java.sql.ResultSet; **import** java.sql.SQLException; **import** java.util.ArrayList;  @WebServlet(name = **"ServletTest"**,urlPatterns = **"/ServletTest"**) **public class** ServletTest **extends** HttpServlet {  @Override  **protected void** service(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {  resp.setContentType(**"text/html;charset=utf-8"**);  PreparedStatement pstmt = DBUtil.*getPstmt*(**"select \* from t\_user2 where sid = 21"**);  req.setAttribute(**"abc"**,**"abc"**);  req.setAttribute(**"abc"**,**"阿部察察ccc"**);  req.removeAttribute(**"abc"**);  } } |
| **package** com.manager.listener;  **import** javax.servlet.\*; **import** javax.servlet.annotation.WebListener; **import** javax.servlet.annotation.WebServlet; **import** javax.servlet.http.HttpServletRequest;  @WebListener(value = **"MyRequestLisenter"**) **public class** MyRequestLisenter **implements** ServletRequestAttributeListener {    **public void** attributeAdded(ServletRequestAttributeEvent srae) {  System.***out***.println(**"request对象中放入了一个属性"**);  ServletRequest servletRequest = srae.getServletRequest();  HttpServletRequest req = (HttpServletRequest)servletRequest;  String msg = (String)req.getAttribute(**"msg"**);  req.setAttribute(**"msg"**,**"系统出大问题了"**);  System.***out***.println(msg);  }   @Override  **public void** attributeRemoved(ServletRequestAttributeEvent srae) {  System.***out***.println(**"被移除"**);  }   @Override  **public void** attributeReplaced(ServletRequestAttributeEvent srae) {  System.***out***.println(**"被修改了"**);  } } |
|  |

监听 session

**HttpSessionListener**

**HttpSessionAttributeListener**

|  |
| --- |
| **package** com.manager.listener;  **import** javax.servlet.annotation.WebListener; **import** javax.servlet.http.\*;  @WebListener(value = **"MySessionListener"**) **public class** MySessionListener **implements** HttpSessionListener, HttpSessionAttributeListener {  @Override  **public void** sessionCreated(HttpSessionEvent se) {  HttpSession session = se.getSession();  String aaa = (String)session.getAttribute(**"aaa"**);  System.***out***.println(**"session被创建了"**);  }   @Override  **public void** sessionDestroyed(HttpSessionEvent se) {  System.***out***.println(**"session被销毁了"**);  }   @Override  **public void** attributeAdded(HttpSessionBindingEvent se) {  System.***out***.println(**"session中添加了属性"**);  }   @Override  **public void** attributeRemoved(HttpSessionBindingEvent se) {  System.***out***.println(**"session中属性被移除"**);  }   @Override  **public void** attributeReplaced(HttpSessionBindingEvent se) {  System.***out***.println(**"session中属性被修改"**);  } } |
| **package** com.manager.servlet;  **import** com.google.gson.Gson; **import** com.manager.pojo.User; **import** com.manager.util.DBUtil;  **import** javax.servlet.ServletException; **import** javax.servlet.annotation.WebServlet; **import** javax.servlet.http.HttpServlet; **import** javax.servlet.http.HttpServletRequest; **import** javax.servlet.http.HttpServletResponse; **import** javax.servlet.http.HttpSession; **import** java.io.IOException; **import** java.sql.PreparedStatement; **import** java.sql.ResultSet; **import** java.sql.SQLException; **import** java.util.ArrayList;  @WebServlet(name = **"ServletTest"**,urlPatterns = **"/ServletTest"**) **public class** ServletTest **extends** HttpServlet {  @Override  **protected void** service(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {  resp.setContentType(**"text/html;charset=utf-8"**);  HttpSession session = req.getSession();  session.setAttribute(**"aaa"**,**"abc"**);  session.setAttribute(**"bbb"**,**"abc"**);  session.setAttribute(**"aaa"**,**"bcd"**);  session.removeAttribute(**"bbb"**);  session.invalidate();  } } |
|  |

监听ServletContext (application)

**ServletContextListener**

**ServletContextAttributeListener**

**案例：**用监听实现统计网站在线登录人数。

ServerletContext中的属性num来记录人数

监听器的实现方式：

图解:

|  |
| --- |
| 1.session记录登录的状态的,所以与监听session有关  2.记录登录人数的属性num,对所有的用户都可以访问到(ServletContext)  3.在什么时候让num++ ?,session中加入user属性时  4.下线,num-- ? ,退出登录或session(默认30分钟)失效了 |

|  |
| --- |
| **package** com.manager.listener;  **import** javax.servlet.ServletContext; **import** javax.servlet.ServletContextEvent; **import** javax.servlet.ServletContextListener; **import** javax.servlet.annotation.WebListener; **import** javax.servlet.http.\*;  @WebListener(value = **"ManagerListener"**) **public class** ManagerListener **implements** HttpSessionListener, HttpSessionAttributeListener, ServletContextListener {  */\*\*  \* 启动是,申明并初始化num变量  \** ***@param sce*** *\*/* @Override  **public void** contextInitialized(ServletContextEvent sce) {  ServletContext sc = sce.getServletContext();  sc.setAttribute(**"num"**,0);  System.***out***.println(**"num:"**+sc.getAttribute(**"num"**));  }   */\*\*  \* 退出登录或session(默认30分钟)失效了时num--  \** ***@param se*** *\*/* @Override  **public void** sessionDestroyed(HttpSessionEvent se) {  HttpSession session = se.getSession();  ServletContext sc = session.getServletContext();  **int** num = (Integer) sc.getAttribute(**"num"**);  num--;  sc.setAttribute(**"num"**,num);  }   @Override  **public void** sessionCreated(HttpSessionEvent se) {  System.***out***.println(**"session被创建了"**);  }   */\*\*  \* sesssion中加入属性user时num++  \** ***@param se*** *\*/* @Override  **public void** attributeAdded(HttpSessionBindingEvent se) {  System.***out***.println(**"被调用了"**);  HttpSession session = se.getSession();  Object user = session.getAttribute(**"user"**);  **if**(user != **null**){  *//num++* ServletContext sc = session.getServletContext();  **int** num =(Integer) sc.getAttribute(**"num"**);  num++;  *//放回ServletContext域中* sc.setAttribute(**"num"**,num);  }  } } |
| **测试的时候,用两个不同的浏览器去测试** |

**过滤器学习**

**问题：**

目前我们访问Servlet是可以直接访问的，没有进行任何防护，可能会造成服务器资源的浪费和非法请求，安全性不高，我们希望在请求被servlet处理之前，进行一次请求的校验，符合要求再调用响应的servlet.

**解决：**

使用过滤器。

**作用：**

对服务器资源进行管理

保护 servlet

校验请求是否合法（比如：权限是否够）

**使用：**

Filter 接口

Filter 过滤器的配置

详解过滤器：

服务器在接收到浏览器发过来的请求后，先解析请求信息，创建对象request和response，然后根据请求URL地址判断如果符合过滤器的过滤范围，则会调用过滤器中的dofilter来进行请求拦截，并将request和response对象作为实参传递给doFilter方法，我们可以在doFilter方法中申明过滤器的拦截代码

Init方法：服务器开启时调用

Destory方法：服务器关闭时调用

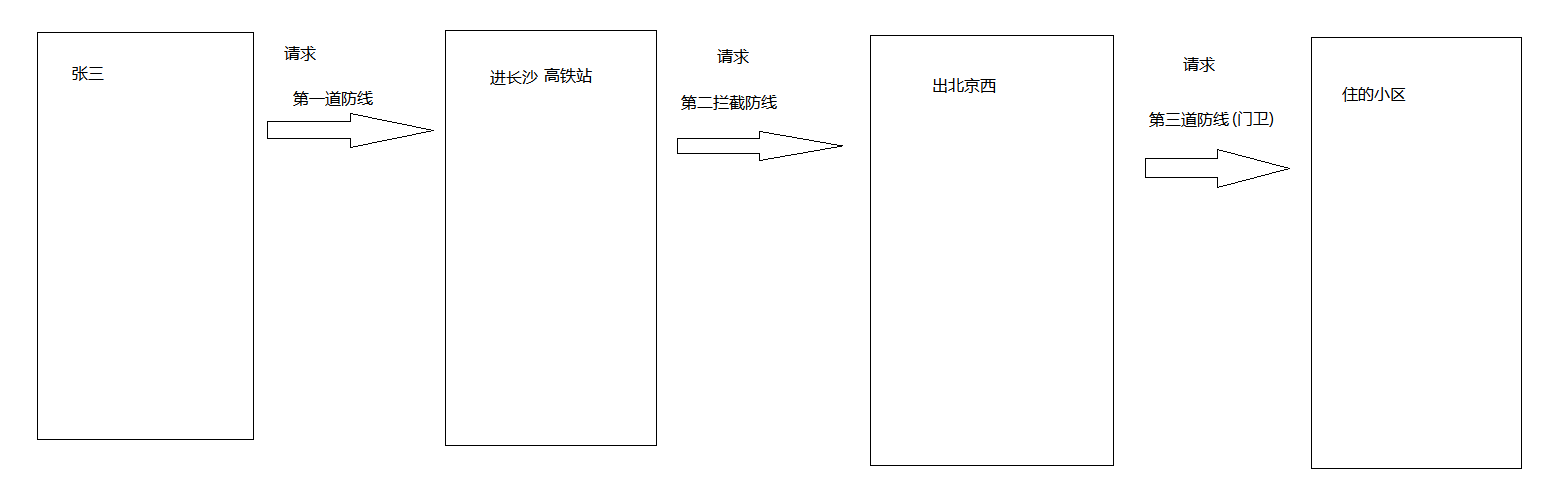
证明过滤器的生命周期：从服务器开启，到服务器关闭

### 拦截范围设置

拦截所有：/\*

拦截部分servlet请求：\*.do

拦截指定servlet请求：和要拦截的Servlet配置相同的url-pattern即可==>/home.do



|  |
| --- |
| **package** com.manager.filter;  **import** javax.servlet.\*; **import** javax.servlet.annotation.WebFilter; **import** java.io.IOException;  @WebFilter(urlPatterns = **"/\*"**) **public class** MyFilter **implements** Filter {  @Override  **public void** doFilter(ServletRequest servletRequest, ServletResponse servletResponse, FilterChain filterChain) **throws** IOException, ServletException {  System.***out***.println(**"被长沙高铁站拦截"**);  *//放行请求* filterChain.doFilter(servletRequest,servletResponse);  } } |
| **package** com.manager.filter;  **import** javax.servlet.\*; **import** javax.servlet.annotation.WebFilter; **import** java.io.IOException;  @WebFilter(urlPatterns = **"\*.do"**) **public class** MyFilter2 **implements** Filter {  @Override  **public void** doFilter(ServletRequest servletRequest, ServletResponse servletResponse, FilterChain filterChain) **throws** IOException, ServletException {  System.***out***.println(**"北京西高铁站"**);  filterChain.doFilter(servletRequest,servletResponse);  } } |
| **package** com.manager.servlet;  **import** javax.servlet.ServletException; **import** javax.servlet.annotation.WebServlet; **import** javax.servlet.http.HttpServlet; **import** javax.servlet.http.HttpServletRequest; **import** javax.servlet.http.HttpServletResponse; **import** java.io.IOException;  @WebServlet(name = **"SanServlet"**,urlPatterns = **"/SanServlet.do"**) **public class** SanServlet **extends** HttpServlet {  @Override  **protected void** service(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {  System.***out***.println(**"张三在北京回龙观小区的家"**);  } } |
| 注意:  第二道过滤器\*.do这个是过滤所有以.do结尾的请求uri,这个do可以随便自定义  如果配了这个过滤器,被访问的Servlet的urlPatterns记住也加上.do结尾,不然不匹配 |
|  |

**拦截规则：先大范围，后小范围**

### 过滤器的使用

案例：

1. 统一管理字符编码

|  |
| --- |
| **package** com.manager.filter;  **import** javax.servlet.\*; **import** javax.servlet.annotation.WebFilter; **import** javax.servlet.http.HttpServletRequest; **import** javax.servlet.http.HttpServletResponse; **import** java.io.IOException;  @WebFilter(urlPatterns = **"/\*"**) **public class** MyFilter **implements** Filter {  @Override  **public void** doFilter(ServletRequest servletRequest, ServletResponse servletResponse, FilterChain filterChain) **throws** IOException, ServletException {  HttpServletRequest req = (HttpServletRequest)servletRequest;  HttpServletResponse resp = (HttpServletResponse)servletResponse;  req.setCharacterEncoding(**"utf-8"**);  resp.setContentType(**"text/html;charset=utf-8"**);  *//放行请求* filterChain.doFilter(servletRequest,servletResponse);  } } |

1. session 管理

请求地址放行：

3.进行权限管理

|  |
| --- |
| **package** com.manager.filter;  **import** com.manager.pojo.User;  **import** javax.servlet.\*; **import** javax.servlet.annotation.WebFilter; **import** javax.servlet.http.HttpServletRequest; **import** javax.servlet.http.HttpServletResponse; **import** javax.servlet.http.HttpSession; **import** java.io.IOException;  @WebFilter(urlPatterns = **"/\*"**) **public class** MyFilter **implements** Filter {  @Override  **public void** doFilter(ServletRequest servletRequest, ServletResponse servletResponse, FilterChain filterChain) **throws** IOException, ServletException {  HttpServletRequest req = (HttpServletRequest)servletRequest;  HttpServletResponse resp = (HttpServletResponse)servletResponse;  req.setCharacterEncoding(**"utf-8"**);  resp.setContentType(**"text/html;charset=utf-8"**);  String uri = req.getRequestURI();  System.***out***.println(uri);  **boolean** url1 = uri.endsWith(**"/Manager\_war\_exploded/index.jsp"**);  **boolean** url2 = uri.endsWith(**"/Manager\_war\_exploded/LoginServlet"**);  *//放行静态资源* **boolean** url3 = uri.startsWith(**"/Manager\_war\_exploded/css/"**);  **boolean** url4 = uri.startsWith(**"/Manager\_war\_exploded/images/"**);  **boolean** url5 = uri.startsWith(**"/Manager\_war\_exploded/js/"**);  **if**(url1 || url2 || url3 ||url4 ||url5){  *//放行请求* filterChain.doFilter(servletRequest,servletResponse);  }**else**{  HttpSession session = req.getSession();  User user =(User)session.getAttribute(**"user"**);  **if**(user != **null**){ *//是登录状态  //全部放行* filterChain.doFilter(servletRequest,servletResponse);  }**else** {  req.setAttribute(**"msg"**,**"请先登录再访问"**);  req.getRequestDispatcher(**"index.jsp"**).forward(servletRequest,servletResponse);  }  }  } } |