过滤器：

Servlet程序分类：

·标准的Servlet —— MVC

·过滤Servlet（过滤器）

·监听Servlet（监听器）

过滤？什么是过滤器？

过滤：安全、干净

过滤应该是后加入的功能

在WEB中过滤主要起什么作用？

·过滤是在程序运行之后加入的

·功能：

·判断用户是否登录（之前用session）

·屏蔽非法文字（反动、骂人之类的话）

·对请求内容进行统一编码

如果实现一个过滤器呢？

1都是在实现Filter接口

void init(FilterConfig filterConfig) throws ServletException：过滤器初始化，过滤器初始化都是在容器（服务器）启动的时候初始化

void doFilter(ServletRequest request, ServletResponse response, FilterChain chain) throws IOException, ServletException：在标准的Servlet中，doget和dopost方法，传递的都是Http协议的子接口，而在doFilter中，参数都是父接口

void destroy()：过滤器销毁

第一个程序：

|  |
| --- |
| package com.jjm.filterdemo;  import java.io.IOException;  import javax.servlet.Filter;  import javax.servlet.FilterChain;  import javax.servlet.FilterConfig;  import javax.servlet.ServletException;  import javax.servlet.ServletRequest;  import javax.servlet.ServletResponse;  public class FilterDemo02 implements Filter {  //覆写接口的所有方法  public void init(FilterConfig fconfig) throws ServletException {  System.out.println("\*\*\*\*Filter初始化\*\*\*\*");  }    public void doFilter(ServletRequest request, ServletResponse response,  FilterChain chain) throws IOException, ServletException {  System.out.println("\*\*\*\*Filter\*\*doFilter\*\*\*\*");    }  public void destroy() {  System.out.println("\*\*\*\*Filter销毁\*\*\*\*");  }  }/\*<filter>  <filter-name>filter01</filter-name>  <filter-class>com.jjm.filterdemo.FilterDemo02</filter-class>  </filter>  <filter-mapping>  <filter-name>filter01</filter-name>  **<url-pattern>/\*</url-pattern>**  </filter-mapping>  \*/ |

以上程序将所有的页面都屏蔽掉了，浏览器中什么也没有出现

|  |
| --- |
| **<url-pattern>/\*</url-pattern>** |

之前在标准的Servlet ，这句话表示的是Servlet的访问地址，而在过滤器里面，表示对哪个页面进行过滤，如果说写上“/\*”，表示对所有的页面进行过滤

此时如果过滤器想要将内容传递到目的地，则需要FilterChain,表示将请求向下转发

void doFilter(ServletRequest request, ServletResponse response)throws IOException, ServletException：此方法就表示将请求向下转发

|  |
| --- |
| package com.jjm.filterdemo;  import java.io.IOException;  import javax.servlet.Filter;  import javax.servlet.FilterChain;  import javax.servlet.FilterConfig;  import javax.servlet.ServletException;  import javax.servlet.ServletRequest;  import javax.servlet.ServletResponse;  public class FilterDemo02 implements Filter {  //覆写接口的所有方法  public void init(FilterConfig fconfig) throws ServletException {  System.out.println("\*\*\*\*Filter初始化\*\*\*\*");  }    public void doFilter(ServletRequest request, ServletResponse response,  FilterChain chain) throws IOException, ServletException {  System.out.println("\*\*\*\*Filter\*\*doFilter\*\*\*\*");  chain.doFilter(request, response);//表示向下执行  }  public void destroy() {  System.out.println("\*\*\*\*Filter销毁\*\*\*\*");  }  }/\*<filter>  <filter-name>filter01</filter-name>  <filter-class>com.jjm.filterdemo.FilterDemo02</filter-class>  </filter>  <filter-mapping>  <filter-name>filter01</filter-name>  <url-pattern>/\*</url-pattern>  </filter-mapping>  \*/ |

现在列表已经显示了

过滤器什么时候销毁呢？

容器关闭的时候销毁

1过滤非法文字

|  |
| --- |
| package com.jjm.filterdemo;  import java.io.IOException;  import javax.servlet.Filter;  import javax.servlet.FilterChain;  import javax.servlet.FilterConfig;  import javax.servlet.ServletException;  import javax.servlet.ServletRequest;  import javax.servlet.ServletResponse;  public class FilterDemo01 implements Filter {  //覆写接口的所有方法  public void init(FilterConfig fconfig) throws ServletException {  }    public void doFilter(ServletRequest request, ServletResponse response,  FilterChain chain) throws IOException, ServletException {  **request.setCharacterEncoding("GBK");**  String str = request.getParameter("invalue");  if(null!=str&&!str.equals("")){  if(str.indexOf("去死")==-1){  chain.doFilter(request, response);//表示向下执行  }else{  System.out.println("过滤文字");  }  }else{  chain.doFilter(request, response);//表示向下执行  }  }  public void destroy() {  try {  Thread.sleep(3000);  } catch (InterruptedException e) {  e.printStackTrace();  }  }  }/\*<filter>  <filter-name>filter02</filter-name>  <filter-class>com.jjm.filterdemo.FilterDemo01</filter-class>  </filter>  <filter-mapping>  <filter-name>filter02</filter-name>  <url-pattern>/\*</url-pattern>  </filter-mapping>  \*/ |

以上程序已经实现了文字的过滤，并且进行统一的编码

登陆验证：

|  |
| --- |
| package com.jjm.filterdemo;  import java.io.IOException;  import javax.servlet.Filter;  import javax.servlet.FilterChain;  import javax.servlet.FilterConfig;  import javax.servlet.ServletException;  import javax.servlet.ServletRequest;  import javax.servlet.ServletResponse;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpSession;  public class FilterDemo03 implements Filter {  //覆写接口的所有方法  public void init(FilterConfig fconfig) throws ServletException {  }    public void doFilter(ServletRequest request, ServletResponse response,  FilterChain chain) throws IOException, ServletException {  request.setCharacterEncoding("GBK");  HttpServletRequest res = (HttpServletRequest)request;  HttpSession session = res.getSession();  if(session.getAttribute("uname")!=null){  System.out.println(session.getAttribute("uname"));  chain.doFilter(request, response);  }else{    }  }  public void destroy() {  try {  Thread.sleep(3000);  } catch (InterruptedException e) {  e.printStackTrace();  }  }  }/\*<filter>  <filter-name>filter02</filter-name>  <filter-class>com.jjm.filterdemo.FilterDemo01</filter-class>  </filter>  <filter-mapping>  <filter-name>filter02</filter-name>  <url-pattern>/\*</url-pattern>  </filter-mapping>  \*/ |

以上就是 登陆验证