## day19\_Filter&Listener

### 学习目标

* 理解并掌握如何编写一个Filter(过滤器)
* 理解并掌握Filter的各项配置
* 理解并完成登陆验证案例和过滤敏感词汇案例
* 理解并掌握动态代理增强某个方法
* 理解并掌握ServletContextListener监听器

### 课程内容

* Filter快速入门
* Filter在web.xml中的各项配置
* Filter的执行流程和生命周期
* Filter拦截路径配置
* 登陆验证案例
* 敏感词汇案例
* 动态代理
* ServletContextListener

### 第一堂课

本节知识点

1.1 今日内容

1.2 Filter\_概述

1.3 Filter快速入门

1.4 Filter\_细节\_web.xml配置方式

视频时间

26分41秒

本节目标

理解Filter的概念和作用

能够快速写出一个Filter的demo

知道在web.xml中如何配置Filter

#### 今日内容

##### 1.1.1知识概述

了解今日学习内容

##### 1.1.2视频详情



##### 1.1.3总结与补充

无

##### 1.1.4课堂提问与练习

无

##### 1.1.5习题答案

无

##### 1.1.6练习答案

无

##### 1.1.7视频缺陷

无

##### 1.1.8视频扩展

无

#### 1.2 Filter\_概述

##### 1.2.1知识概述

1. 概念：

\* 生活中的过滤器：净水器,空气净化器，土匪、

\* web中的过滤器：当访问服务器的资源时，过滤器可以将请求拦截下来，完成一些特殊的功能。

\* 过滤器的作用：

\* 一般用于完成通用的操作。如：登录验证、统一编码处理、敏感字符过滤...

##### 1.2.2视频详情



##### 1.2.3总结与补充

无

##### 1.2.4课堂提问与练习

无

##### 1.2.5习题答案

无

##### 1.2.6练习答案

无

##### 1.2.7视频缺陷

##### 1.2.8视频扩展

#### 1.3 Filter快速入门

##### 1.3.1知识概述

2. 快速入门：

1. 步骤：

1. 定义一个类，实现接口Filter

2. 复写方法

3. 配置拦截路径

1. web.xml

2. 注解

##### 1.3.2视频详情



##### 1.3.3总结与补充

无

##### 1.3.4课堂提问与练习

无

##### 1.3.5习题答案

无

##### 1.3.6 练习答案

@WebFilter("/\*")//访问所有资源之前，都会执行该过滤器

public class FilterDemo1 implements Filter {

@Override

public void init(FilterConfig filterConfig) throws ServletException {

}

@Override

public void doFilter(ServletRequest servletRequest, ServletResponse servletResponse, FilterChain filterChain) throws IOException, ServletException {

System.out.println("filterDemo1被执行了....");

//放行

filterChain.doFilter(servletRequest,servletResponse);

}

@Override

public void destroy() {

}

##### 1.3.7 视频缺陷

##### 1.3.8 视频扩展

#### 1.4 Filter\_细节\_web.xml配置方式

##### 1.4.1知识概述

1. web.xml配置

<filter>

<filter-name>demo1</filter-name>

<filter-class>cn.itcast.web.filter.FilterDemo1</filter-class>

</filter>

<filter-mapping>

<filter-name>demo1</filter-name>

<!-- 拦截路径 -->

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

</filter-mapping>

##### 1.4.2视频详情



##### 1.4.3总结与补充

无

##### 1.4.4课堂提问与练习

无

##### 1.4.5习题答案

无

##### 1.4.6 练习答案

无

##### 1.4.7 视频缺陷

##### 1.4.8 视频扩展

### 第二堂课

本节知识点：

2.1 Filter\_细节\_执行流程&生命周期

2.2 Filter\_细节\_过滤器拦截路径配置

2.3 Filter\_细节\_过滤器拦截方式配置

视频时间：

35分33秒

本节目标 ：

1. 能够说出Filter的执行流程和生命周期
2. 能够配置过滤器的拦截路径
3. 能够配置过滤器的拦截方式

#### Filter\_细节\_执行流程&生命周期

##### 2.1.1知识概述

2. 过滤器执行流程

1. 执行过滤器

2. 执行放行后的资源

3. 回来执行过滤器放行代码下边的代码

3. 过滤器生命周期方法

1. init:在服务器启动后，会创建Filter对象，然后调用init方法。只执行一次。用于加载资源

2. doFilter:每一次请求被拦截资源时，会执行。执行多次

3. destroy:在服务器关闭后，Filter对象被销毁。如果服务器是正常关闭，则会执行destroy方法。只执行一次。用于释放资源

##### 2.1.2视频详情



##### 2.1.3总结与补充

1. 无

##### 2.1.4课堂提问与练习

* 请描述Filter的执行流程？

2. 过滤器执行流程

1. 执行过滤器

2. 执行放行后的资源

3. 回来执行过滤器放行代码下边的代码

* 请描述Filter的生命周期？

3. 过滤器生命周期方法

1. init:在服务器启动后，会创建Filter对象，然后调用init方法。只执行一次。用于加载资源

2. doFilter:每一次请求被拦截资源时，会执行。执行多次

3. destroy:在服务器关闭后，Filter对象被销毁。如果服务器是正常关闭，则会执行destroy方法。只执行一次。用于释放资源

##### 2.1.5习题答案

无

##### 2.1.6 练习答案

无

##### 2.1.7 视频缺陷

无

##### 2.1.8 视频扩展

无

#### 2.2 Filter\_细节\_过滤器拦截路径配置

##### 2.2.1知识概述

4. 过滤器配置详解

\* 拦截路径配置：

1. 具体资源路径： /index.jsp 只有访问index.jsp资源时，过滤器才会被执行

2. 拦截目录： /user/\* 访问/user下的所有资源时，过滤器都会被执行

3. 后缀名拦截： \*.jsp 访问所有后缀名为jsp资源时，过滤器都会被执行

4. 拦截所有资源：/\* 访问所有资源时，过滤器都会被执行

##### 2.2.2视频详情



##### 2.2.3总结与补充

无

##### 2.2.4课堂提问与练习

* 过滤器的拦截路径有几种配置方式？

4. 过滤器配置详解

\* 拦截路径配置：

1. 具体资源路径： /index.jsp 只有访问index.jsp资源时，过滤器才会被执行

2. 拦截目录： /user/\* 访问/user下的所有资源时，过滤器都会被执行

3. 后缀名拦截： \*.jsp 访问所有后缀名为jsp资源时，过滤器都会被执行

4. 拦截所有资源：/\* 访问所有资源时，过滤器都会被执行

##### 2.2.5习题答案

无

##### 2.2.6 练习答案

无

##### 2.2.7 视频缺陷

无

##### 2.2.8 视频扩展

无

#### 2.3 Filter\_细节\_过滤器拦截方式配置

##### 2.3.1知识概述

\* 拦截方式配置：资源被访问的方式

\* 注解配置：

\* 设置dispatcherTypes属性

1. REQUEST：默认值。浏览器直接请求资源

2. FORWARD：转发访问资源

3. INCLUDE：包含访问资源

4. ERROR：错误跳转资源

5. ASYNC：异步访问资源

\* web.xml配置

\* 设置<dispatcher></dispatcher>标签即可

##### 2.3.2视频详情



##### 2.3.3总结与补充

无

##### 2.3.4课堂提问与练习

* 过滤器的拦截方式有哪些？

\* 拦截方式配置：资源被访问的方式

\* 注解配置：

\* 设置dispatcherTypes属性

1. REQUEST：默认值。浏览器直接请求资源

2. FORWARD：转发访问资源

3. INCLUDE：包含访问资源

4. ERROR：错误跳转资源

5. ASYNC：异步访问资源

\* web.xml配置

\* 设置<dispatcher></dispatcher>标签即可

##### 2.3.5习题答案

无

##### 2.3.6 练习答案

无

##### 2.3.7 视频缺陷

无

##### 2.3.8 视频扩展

无

### 第三堂课

本节知识点：

3.1 Filter\_细节\_过滤器链(多个过滤器)

3.2 Filter\_案例1\_登录验证\_分析

3.3 Filter\_案例1\_登录验证\_代码实现

视频时间：

29分21秒

本节目标 ：

理解过滤器链的概念和执行顺序

完成登陆验证案例

#### Filter\_细节\_过滤器链(多个过滤器)

##### 3.1.1知识概述

5. 过滤器链(配置多个过滤器)

\* 执行顺序：如果有两个过滤器：过滤器1和过滤器2

1. 过滤器1

2. 过滤器2

3. 资源执行

4. 过滤器2

5. 过滤器1

\* 过滤器先后顺序问题：

1. 注解配置：按照类名的字符串比较规则比较，值小的先执行

\* 如： AFilter 和 BFilter，AFilter就先执行了。

2. web.xml配置： <filter-mapping>谁定义在上边，谁先执行

##### 3.1.2视频详情



##### 3.1.3总结与补充

无

##### 3.1.4课堂提问与练习

* 过滤器链中的过滤器执行顺序如何决定？

5. 过滤器链(配置多个过滤器)

\* 执行顺序：如果有两个过滤器：过滤器1和过滤器2

1. 过滤器1

2. 过滤器2

3. 资源执行

4. 过滤器2

5. 过滤器1

\* 过滤器先后顺序问题：

1. 注解配置：按照类名的字符串比较规则比较，值小的先执行

\* 如： AFilter 和 BFilter，AFilter就先执行了。

2. web.xml配置： <filter-mapping>谁定义在上边，谁先执行

##### 3.1.5习题答案

无

##### 3.1.6练习答案

无

##### 3.1.7视频缺陷

##### 3.1.8视频扩展

#### Filter\_案例1\_登录验证\_分析

##### 3.2.1知识概述

1. 案例1\_登录验证

\* 需求：

1. 访问day17\_case案例的资源。验证其是否登录

2. 如果登录了，则直接放行。

3. 如果没有登录，则跳转到登录页面，提示"您尚未登录，请先登录"。

##### 3.2.2视频详情



##### 3.2.3总结与补充

无

##### 3.2.4课堂提问与练习

无

##### 3.2.5习题答案

无

##### 3.2.6练习答案

无

##### 3.2.6视频缺陷

无

##### 3.2.6视频扩展

无

#### Filter\_案例1\_登录验证\_代码实现

##### 3.3.1知识概述

实现登陆验证案例

##### 3.3.2视频详情



##### 3.3.3总结与补充

无

##### 3.3.4课堂提问与练习

无

##### 3.3.5习题答案

无

##### 3.3.6 练习答案

无

##### 3.3.6 视频缺陷

无

##### 3.3.6 视频扩展

无

### 第四堂课

本节知识点：

4.1 Filter\_案例2\_过滤敏感词汇\_分析

4.2动态代理1\_基本实现步骤

视频时间

31分02秒

本节目标 ：

1. 理解过滤敏感词汇案例的需求
2. 理解动态代理实现的基本步骤

#### 4.1 Filter\_案例2\_过滤敏感词汇\_分析

##### 4.1.1知识概述

2. 案例2\_敏感词汇过滤

\* 需求：

1. 对day17\_case案例录入的数据进行敏感词汇过滤

2. 敏感词汇参考《敏感词汇.txt》

3. 如果是敏感词汇，替换为 \*\*\*

\* 分析：

1. 对request对象进行增强。增强获取参数相关方法

2. 放行。传递代理对象

##### 4.1.2视频详情



##### 4.1.3总结与补充

无

##### 4.1.4课堂提问与练习

无

##### 4.1.5习题答案

无

##### 4.1.6练习答案

无

##### 4.1.7视频缺陷

无

##### 4.1.6视频扩展

无

#### 4.2动态代理1\_基本实现步骤

##### 4.2.1知识概述

\* 增强对象的功能：

\* 设计模式：一些通用的解决固定问题的方式

1. 装饰模式

2. 代理模式

\* 概念：

1. 真实对象：被代理的对象

2. 代理对象：

3. 代理模式：代理对象代理真实对象，达到增强真实对象功能的目的

\* 实现方式：

1. 静态代理：有一个类文件描述代理模式

2. 动态代理：在内存中形成代理类

\* 实现步骤：

1. 代理对象和真实对象实现相同的接口

2. 代理对象 = Proxy.newProxyInstance();

3. 使用代理对象调用方法。

4. 增强方法

##### 4.2.2视频详情



##### 4.2.3总结与补充

无

##### 4.2.4课堂提问与练习

无

##### 4.2.5习题答案

无

##### 4.2.6练习答案

无

##### 4.2.6视频缺陷

无

##### 4.2.6视频扩展

无

### 第五堂课

本节知识点：

5.1 动态代理2\_增强方法

5.2 Filter\_案例2\_过滤敏感词汇\_实现

视频时间

27分50秒

本节目标 ：

能够使用动态代理增强方法

完成过滤敏感词汇案例

#### 动态代理2\_增强方法

##### 5.1.1知识概述

\* 增强方法：

1. 增强参数列表

2. 增强返回值类型

3. 增强方法体执行逻辑

##### 5.1.2视频详情



##### 5.1.3总结与补充

无

##### 5.1.4课堂提问与练习

无

##### 5.1.5习题答案

无

##### 5.1.6练习答案

无

##### 5.1.7视频缺陷

##### 5.1.8 视频扩展

#### Filter\_案例2\_过滤敏感词汇\_实现

##### 5.2.1知识概述

实现过滤敏感词汇案例

##### 5.2.2视频详情



##### 5.2.3总结与补充

无

##### 5.2.4课堂提问与练习

无

##### 5.2.5习题答案

无

##### 5.2.6练习答案

无

##### 5.2.7练习答案

##### 5.2.8练习答案

### 第六堂课

本节知识点：

6.1 练习过滤敏感词汇案例

6.2 Listener\_概述

6.3 Listener\_ServletContextListener使用

视频时间

17分43秒

本节目标 ：

理解ServletContextListener

#### 6.1 练习过滤敏感词汇案例(没有视频)

#### 6.2 Listener\_概述

##### 6.2.1知识概述

\* 概念：web的三大组件之一。

\* 事件监听机制

\* 事件 ：一件事情

\* 事件源 ：事件发生的地方

\* 监听器 ：一个对象

\* 注册监听：将事件、事件源、监听器绑定在一起。 当事件源上发生某个事件后，执行监听器代码

##### 6.2.2视频详情



##### 6.2.3总结与补充

无

##### 6.2.4课堂提问与练习

无

##### 6.2..5习题答案

无

##### 6.2.6练习答案

无

##### 6.2.7视频缺陷

无

##### 6.2.8视频扩展

无

#### 6.3 Listener\_ServletContextListener使用

##### 6.3.1知识概述

\* ServletContextListener:监听ServletContext对象的创建和销毁

\* 方法：

\* void contextDestroyed(ServletContextEvent sce) ：ServletContext对象被销毁之前会调用该方法

\* void contextInitialized(ServletContextEvent sce) ：ServletContext对象创建后会调用该方法

\* 步骤：

1. 定义一个类，实现ServletContextListener接口

2. 复写方法

3. 配置

1. web.xml

<listener>

<listener-class>cn.itcast.web.listener.ContextLoaderListener</listener-class>

\* 指定初始化参数<context-param>

2. 注解：

\* @WebListener

##### 6.3.2视频详情



##### 6.3.3总结与补充

无

##### 6.3.4课堂提问与练习

无

##### 6.3..5习题答案

无

##### 6.3.6练习答案

无

##### 6.3.6视频缺陷

无

##### 6.3.6视频扩展

无