

## Filter

实现接口Filter，implements Filter

有三个方法：构造方法（），init()，destroy（）在tomcat启动中只执行一次

doFilter() : 拦截之后执行的方法

chian.Filter(request, response) 将请求和响应都转发给目标资源或者下一个过滤器。写在doFilter中。FilterChain接口，只有一个方法doFIlter。

配置拦截路径：使用注解@WebFilter 或者使用xml文件@WebFilter(filterName=”ABCFilter”, urlPatterns= {“/a”, “/b”}),可以配置多条过滤路径，同时一个类可以被多个过滤器过滤，过滤

的先后顺序为：在web.xml文件中设置。

<filter>

<filter-name>

<filter-class>

</filter>

<filter-mapping>

<filter-name>BCDFilter</filter-name>

<urlPattern>/以项目的虚拟路径为起点</ urlPattern >

</filter-mapping> 相当于是注册

在spring boot中使用Filter

自动注解的方法：

使用注解@WebFilter（name = “”， urlPatterns= { }），配置顺序@Order

因为@WebFilter不是spring的原生注解。所以再启动类上要开启扫描@ServletComponentScan 可以扫描过滤器，监听器，servlet，即扫描组件@WebFilter，@WebListener，@WebServlet

或者使用注解类的方法

类似于web.xml文件中注册Filter

@Configuration 这个注解也是一个次配置类，会自动的被扫描到ioc容器中。

public class WebComponent2Config {

@Bean

public FilterRegistrationBean someFilterRegistration1() {

//新建过滤器注册类

FilterRegistrationBean registration = new FilterRegistrationBean();

// 添加我们写好的过滤器

registration.setFilter( new SessionFilter());

// 设置过滤器的URL模式

registration.addUrlPatterns("/\*");

return registration;

}

}

## 监听器Listener

有连个接口，通过实现着6个接口，就可以自动监听他们的状态和属性

ServletContextListener，ServletRequestListener, HttpSessionListener(监听创建和销毁)

ServletContextAttributeListener, ServletRequestAttributeListener, HttpSessionAttributeListener,(监听属性的添加，修改，删除)

为了表明这是一个监听器，使用注解@WebListener

在springboot中使用的话，添加扫描注解@WebComponentScance

使用注解类@Configuration 中@Bean 返回一个ServletListenRegistrationBean