链滴

关于 spring 的一两点

作者：Ouyuone

原文链接：https://ld246.com/article/1571643074807

来源网站：链滴

众所周知使用spring会大大的增加我们的开发效率和节约开发时间,那如何来使用Spring提供的自建类帮助我们更好的使用它:

1.提供统一的controller拦截

@ControllerAdvice 加上方法上的@ExceptionHandler(RuntimeException.class) 参数RuntimeException.class是抛出的异常

2.使用过滤器来实现一些拦截事情

我们的类上面实现Filter重新它的doFilter方法

```
public class HttpServletFilter implements Filter {
    @Override
    public void init(FilterConfig filterConfig) throws ServletException {

    }

    @Override
    public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain) throws IOException, ServletException {
        ServletRequest requestWrapper = null;
        if(request instanceof HttpServletRequest) {
            requestWrapper = new RequestWrapper((HttpServletRequest) request);
        }
        if(requestWrapper == null) {
            chain.doFilter(request, response);
        } else {
            chain.doFilter(requestWrapper, response);
        }
    }

    @Override
    public void destroy() {

    }
}
```

这样写好后我们需要注册到springBean中让它帮我们管理在需要拦截的资源前帮我们激活此方法

```
@Configuration
public class FilterRegistrationConfige {
    @Bean
    public FilterRegistrationBean httpServletFilter() {
        FilterRegistrationBean myFilter = new FilterRegistrationBean();
        myFilter.addUrlPatterns("/*");
        myFilter.setFilter(new HttpServletFilter());
        return myFilter;
    }
}
```

这样就完美的拦截所以资源了.

3.使用拦截器

在我们的类继承HandlerInterceptorAdapter实现它的方法preHandle``如下:

```
@Component
public class CommonInterceptor extends HandlerInterceptorAdapter {
    private final static String PREFIX = "Bearer";
    private final static String AUTH_HEADER = "Authorization";

    @Override
    public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object handler) throws Exception {
        boolean flag = true;
        String authHeader = request.getHeader(AUTH_HEADER);
        if (StringUtils.isBlank(authHeader)) {
            flag = false;
        }else {
            String token = authHeader.substring(7);
            if (StringUtils.isBlank(token))
                flag = false;
            ApiUtils.currentUid(token);
        }
        if (!flag) {
            throw new ApiException(ErrorCodeEnum.UNAUTHORIZED);
        }
        return flag;
    }
}
```

原文链接：关于 spring 的一两点

跟filter一样我们需要找个注册点让拦截器生效

在springboot2.+版本上我们实现WebMvcConfigurer这个类在springboot1.+我们实现WebMvcConfigurerAdapter

```
@Configuration
public class WebConfigurer implements WebMvcConfigurer {


    @Override
    public void addInterceptors(InterceptorRegistry registry) {
        registry.addInterceptor(new CommonInterceptor()).addPathPatterns("api/video/upload");
        registry.addInterceptor(new CommonInterceptor()).addPathPatterns("api/video/download");
        registry.addInterceptor(new CommonInterceptor()).addPathPatterns("api/video/rank");
        registry.addInterceptor(new CommonInterceptor()).addPathPatterns("/api/user/info");
    }


}
```

这样是不是很完美呀~~~~