

**常见的jwt生成算法**: HS256 RS256 ES256

**JWT（JSON Web Tokens）的生成和验证过程涉及几个关键步骤，确保了在不同系统间安全地传输和验证用户身份信息。以下是JWT的整个工作流程的详细说明：**



\*\*过滤器（Filter）和拦截器（Interceptor）\*\*在 Web 开发中都是常见的工具，主要用于拦截请求和响应，执行前置或后置处理。两者的区别和应用如下：

过滤器（Filter）

概念：基于 Servlet 规范，用于在请求到达 Servlet 之前或响应返回客户端之前处理。

处理范围：应用于整个 Web 层，适用于所有请求和响应。

典型用途：日志记录、字符编码处理、跨域资源共享（CORS）、安全过滤。

实现方式：通过 Filter 接口，并在 doFilter 方法中进行前后处理。

@WebFilter(urlPatterns = "/\*")

public class MyFilter implements Filter {

public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain) {

// 处理请求

chain.doFilter(request, response);

// 处理响应

}

}

拦截器（Interceptor）

概念：Spring MVC 机制，拦截控制器的请求处理前后。

处理范围：专注于控制器层，处理与业务逻辑相关的请求。

典型用途：权限校验、用户认证、控制器执行时间统计。

实现方式：通过 HandlerInterceptor 接口，提供 preHandle、postHandle、afterCompletion 方法。

public class MyInterceptor implements HandlerInterceptor {

public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object handler) {

return true; // 继续处理请求

}

}

区别

范围：过滤器作用于整个应用请求，拦截器仅处理控制器相关请求。

管理：过滤器由 Servlet 容器管理，拦截器由 Spring 容器管理。

应用场景：过滤器用于全局请求处理，拦截器则用于业务逻辑层面的验证与处理。

两者可以搭配使用：过滤器处理跨域、编码等全局问题，拦截器处理具体业务逻辑的校验与处理。

1. 什么是Servlet？其生命周期有哪些阶段？

答案：

Servlet是运行在服务器端的Java程序，用于处理客户端请求并生成动态响应。

生命周期阶段：

加载和实例化：Servlet容器加载Servlet类并实例化对象。

初始化：调用init()方法，进行一次初始化。

服务请求：每次请求调用service()方法，依次执行doGet()或doPost()等方法。

销毁：调用destroy()方法，在Servlet容器卸载Servlet时执行。

8. 如何避免SQL注入？

答案：

使用预编译语句（PreparedStatement）代替直接拼接SQL语句。

对用户输入进行验证和过滤，确保输入符合预期格式。

9. Session失效的原因有哪些？如何避免？

答案：

原因：

服务器重启或宕机。

Session超时（到达最大空闲时间）。

用户主动注销。

避免方法：

使用Session持久化机制（如Redis、数据库）。

设置合理的Session超时时间，避免频繁超时。