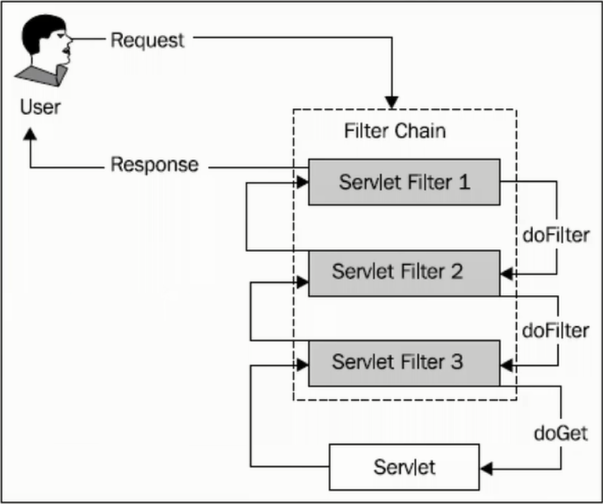
**Spring Security-权限拦截**



* **SecurityContextPersistenceFilter**
* 以前是HttpSesstionContextIntegrationFilter,位于过滤器的顶端，是第一个起作用的过滤器
* 第一个用途：在执行其他过滤器之前，率先判断用户的session是否已经存在了一个spring security上下文的securityContext,如果存在，就把securityContext拿出来，放在securityContextHolder中，供security的其他部分使用。如果不存在，就创建一个securityContext出来，放在securityContextHolder中，供security的其他部分使用。
* 第二个用途：在所有过滤器执行完毕后，清空securityContextHolder中的内容，因为securityContextHolder是基于ThreadLocal的，如果不清空，会受到服务器线程池机制的影响。
* ThreadLocal存放的值是线程内共享的，线程间互斥的，主要用于线程内共享一些数据，避免通过参数来传递。这样处理后，能够解决实际中的一些并发问题。ThreadLocalMap是ThreadLocal的一个内部类，是不对外使用的。当使用ThreadLocal存值时，首先获取到当前线程对象，然后获取到当前线程本地对象，本地变量map,最后将当前使用的所有local和传入的值放在map中。也就是说ThreadLocalMap中的key是ThreadLocal对象。这样，每个线程都对应一个本地的map,所以，一个线程可以存在多个线程本地变量。
* ThreadLocal是解决线程并发问题的一个很好的思路，通过对每个线程提供一个独立的变量副本，解决线程并发访问变量的一个冲突问题
* 当一个线程结束的时候，记得把ThreadLocal里的变量移除掉remove();
* **LogoutFilter**
* 只处理注销请求。在用户发送注销请求时，销毁用户的session,清空securityContextHolder,重定向到注销成功页面。
* **AbstractAuthenticationProcessingFilter**
* 处理form登录的过滤器，与form登录有关的操作都在此进行。
* **~~DefaultLoginPageGeneratingFilter~~**
* ~~用来生成一个默认的登录页面，默认的访问地址为spring\_security\_login,这个登录页面虽然支持用户输入用户名密码，也支持remember me等功能，但是因为太难看了，只能在演示时做个样子，不能直接在实际项目中使用~~
* **BasicAutenticationFilter**
* 用来进行basic验证，与AbstractAuthenticationProcessingFilter类似。
* **SecurityContextHolderAwareRequestFilter**
* 用来包装客户的请求，目的是在原来请求的基础上，为后续程序提供一些额外的数据，比如getRomoteUser时，直接返回当前登录的用户名
* **RememberMeAuthenticationFilter**
* 实现Remember me功能，当用户cookie中存在remember me标记时，它会根据标记自动实现用户登录，并创建securityContext,授予对应的权限。spring security中的remember me依赖cookie实现，用户在登录时选择remember me,系统就会在登录成功后为用户生成一个唯一的标识，并将这个标识保存进cookie中，我们可以通过浏览器查看用户电脑中的cookie
* **AnonymousAutenticationFilter**
* 当用户没有登录时，默认为用户分配匿名用户的权限
* **ExceptionTranslationFilter**
* 处理filterSecurityInterceptor中抛出的异常，然后将请求重定向到对应页面，或返回应用的错误代码
* **SessionManagementFilter**
* 在用户登录成功之后，销毁用户的当前session，并重新生成一个session
* **filterSecurityInterceptor**
* 用户的权限控制都包含在这个过滤器中
* 第一个功能，如果用户尚未登录，抛出尚未认证的异常
* 第二个功能，如果用户已登录，但是没有访问当前资源的权限，会抛出拒绝访问的异常
* 第三个功能，如果用户已登录，也具有访问当前资源的权限，那么放行
* FilterChainProxy
* 按照顺序调用一组filter,使他们既能完成验证授权的本职工作，又能相应spring Ioc的功能来很方便地得到其他依赖的资源