**Введение:**

Фильтр авторизации нужен чтобы проверить права доступа пользователя к тем или иным ресурсам, срабатывает этот фильтр самым последним. Этот фильтр проверяет Authorities у нашего объекта аутентификации.

FilterSecurityInterceptor – старая версия фильтра авторизации. Уже есть Authorization Filter, который является новой версией. В настоящее время, именно Authorization Filter отвечают за авторизацию в Spring Security.

Работа Authorization Filter осуществляется с помощью Authorization Manager, а он в свою очередь является функциональным интерфейсом с одним единственным методом check, этот метод возвращает AuthorizationDecision.

AuthorizationDecision – это класс с одним полем, granted типа Boolean (Разрешен доступ или нет). Если доступ не разрешен, то мы получим AccessDeniedException и спринг вернет нам 403 статус.

**RequestMatcherDelegatingAuthorizationManager** – Главный класс, который используется спрингом по умолчанию. Суть его в том, что он по пути, определяет, есть ли доступ к ресурсу у пользователя и эти проверки он передает другим менеджерам. Мы так же можем использовать совместно с другими Authorization Manage, например, мы можем использовать **RequestMatcherDelegatingAuthorizationManager** совместно с **PreAuthorizeAuthorizationManager и PostAuthorizeAuthorizationManager**

**RequestMatcherDelegatingAuthorizationManager –**

**Наследники Authorization Manager:**

**AuthenticatedAuthorizationManager** - Проверяет, прошел ли пользователь аутентификацию.

**AuthorityAuthorizationManager** -Проверяет, прошел ли пользователь аутентификацию, а также проверяет роли объекта аутентификации.

**Jsr250AuthorizationManager** -Спецификация Java EE по Security, Spring в том числе следует этой спецификации.

**PreAuthorizeAuthorizationManager и PostAuthorizeAuthorizationManager** позволяют добавлять проверку прав доступа (авторизация) с помощью аннотаций **@PreAuthorize и @PostAuthorize.**

**SecuredAuthorizationManager** - Позволяет использовать новую аннотацию @Security.