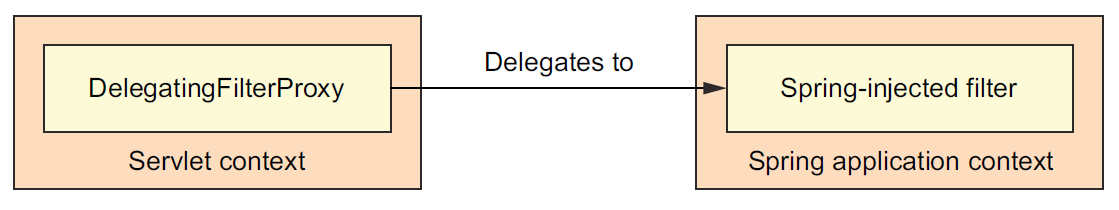
***Filtering web requests***

DelegatingFilterProxy is a special servlet filter that, by itself, doesn’t do much. Instead, it delegates to an implementation of javax.servlet.Filter that’s registered as a <bean> in the Spring application context.



**DelegatingFilterProxy proxies filter handling to a delegate filter bean in the Spring application context.**

If you like configuring servlets and filters in the traditional web.xml file, you can do that with the <filter> element, like this:

<filter>

<filter-name>springSecurityFilterChain</filter-name>

<filter-class>

org.springframework.web.filter.DelegatingFilterProxy

</filter-class>

</filter>

The most important thing here is that the <filter-name> be set to springSecurityFilterChain. That’s because you’ll soon be configuring Spring Security for web security, and there will be a filter bean named springSecurityFilterChain that DelegatingFilterProxy will need to delegate to.

* If you'd rather configure DelegatingFilterProxy in Java with a WebApplicationInitializer, then all you need to do is create a new class that extends AbstractSecurityWebApplicationInitializer:

package spitter.config;

import org.springframework.security.web.context.

AbstractSecurityWebApplicationInitializer;

public class SecurityWebInitializer extends AbstractSecurityWebApplicationInitializer {

}

* AbstractSecurityWebApplicationInitializer implements WebApplicationInitializer, so it will be discovered by Spring and be used to register DelegatingFilterProxy with the web container.
* Although you can override its appendFilters() or insertFilters() methods to register filters of your own choosing, you need not override anything to register DelegatingFilterProxy.
* Whether you configure DelegatingFilterProxy in web.xml or by subclassing AbstractSecurityWebApplicationInitializer, it will intercept requests coming into the application and delegate them to a bean whose ID is springSecurityFilterChain.
* As for the springSecurityFilterChain bean itself, it’s another special filter known as FilterChainProxy. It’s a single filter that chains together one or more additional filters. Spring Security relies on several servlet filters to provide different security features, but you should almost never need to know these details, as you likely won’t need to explicitly declare the springSecurityFilterChain bean or any of the filters it chains together. Those filters will be created when you enable web security.