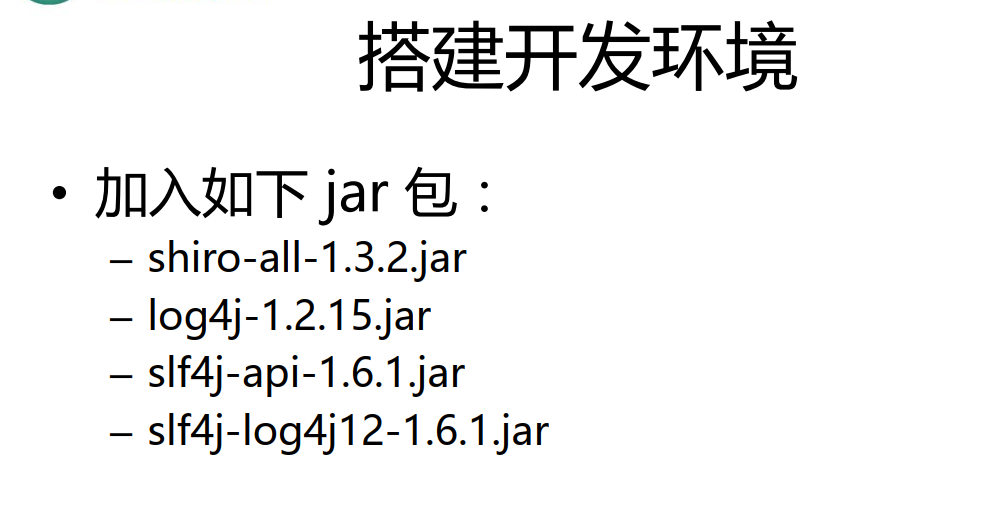


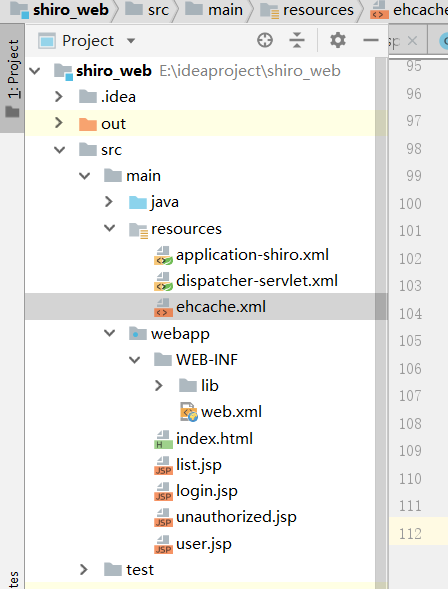


## HelloWorld





## 集成Spring



### Web.xml

<?xml version="1.0" encoding="UTF-8"?>

<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app\_4\_0.xsd"

version="4.0">

<context-param>

<param-name>contextConfigLocation</param-name>

<param-value>classpath:application-shiro.xml</param-value>

</context-param>

<listener>

<listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>

</listener>

<!-- Shiro Filter is defined in the spring application context: -->

<filter>

<filter-name>shiroFilter</filter-name>

<filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>

<init-param>

<param-name>targetFilterLifecycle</param-name>

<param-value>true</param-value>

</init-param>

</filter>

<filter-mapping>

<filter-name>shiroFilter</filter-name>

<url-pattern>/\*</url-pattern>

</filter-mapping>

<servlet>

<servlet-name>dispatcher</servlet-name>

<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

<init-param>

<param-name>contextConfigLocation</param-name>

<param-value>classpath:dispatcher-servlet.xml</param-value>

</init-param>

<load-on-startup>1</load-on-startup>

</servlet>

<servlet-mapping>

<servlet-name>dispatcher</servlet-name>

<url-pattern>/</url-pattern>

</servlet-mapping>

</web-app>

### Spring mvc

<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns:context="http://www.springframework.org/schema/context"

xmlns:mvc="http://www.springframework.org/schema/mvc"

xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd

http://www.springframework.org/schema/mvc http://www.springframework.org/schema/mvc/spring-mvc-4.0.xsd

http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-4.0.xsd">

<context:component-scan base-package="com.liu.shiro"></context:component-scan>

<bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">

<property name="prefix" value="/"></property>

<property name="suffix" value=".jsp"></property>

</bean>

<mvc:annotation-driven></mvc:annotation-driven>

<mvc:default-servlet-handler/>

</beans>

Ehcache.xml

<ehcache>

<!-- Sets the path to the directory where cache .data files are created.

If the path is a Java System Property it is replaced by

its value in the running VM.

The following properties are translated:

user.home - User's home directory

user.dir - User's current working directory

java.io.tmpdir - Default temp file path -->

<diskStore path="java.io.tmpdir"/>

<cache name="authorizationCache"

eternal="false"

timeToIdleSeconds="3600"

timeToLiveSeconds="0"

overflowToDisk="false"

statistics="true">

</cache>

<cache name="authenticationCache"

eternal="false"

timeToIdleSeconds="3600"

timeToLiveSeconds="0"

overflowToDisk="false"

statistics="true">

</cache>

<cache name="shiro-activeSessionCache"

eternal="false"

timeToIdleSeconds="3600"

timeToLiveSeconds="0"

overflowToDisk="false"

statistics="true">

</cache>

<!--Default Cache configuration. These will applied to caches programmatically created through

the CacheManager.

The following attributes are required for defaultCache:

maxInMemory - Sets the maximum number of objects that will be created in memory

eternal - Sets whether elements are eternal. If eternal, timeouts are ignored and the element

is never expired.

timeToIdleSeconds - Sets the time to idle for an element before it expires. Is only used

if the element is not eternal. Idle time is now - last accessed time

timeToLiveSeconds - Sets the time to live for an element before it expires. Is only used

if the element is not eternal. TTL is now - creation time

overflowToDisk - Sets whether elements can overflow to disk when the in-memory cache

has reached the maxInMemory limit.

-->

<defaultCache

maxElementsInMemory="10000"

eternal="false"

timeToIdleSeconds="120"

timeToLiveSeconds="120"

overflowToDisk="true"

/>

<!--Predefined caches. Add your cache configuration settings here.

If you do not have a configuration for your cache a WARNING will be issued when the

CacheManager starts

The following attributes are required for defaultCache:

name - Sets the name of the cache. This is used to identify the cache. It must be unique.

maxInMemory - Sets the maximum number of objects that will be created in memory

eternal - Sets whether elements are eternal. If eternal, timeouts are ignored and the element

is never expired.

timeToIdleSeconds - Sets the time to idle for an element before it expires. Is only used

if the element is not eternal. Idle time is now - last accessed time

timeToLiveSeconds - Sets the time to live for an element before it expires. Is only used

if the element is not eternal. TTL is now - creation time

overflowToDisk - Sets whether elements can overflow to disk when the in-memory cache

has reached the maxInMemory limit.

-->

<!-- Sample cache named sampleCache1

This cache contains a maximum in memory of 10000 elements, and will expire

an element if it is idle for more than 5 minutes and lives for more than

10 minutes.

If there are more than 10000 elements it will overflow to the

disk cache, which in this configuration will go to wherever java.io.tmp is

defined on your system. On a standard Linux system this will be /tmp"

-->

<cache name="sampleCache1"

maxElementsInMemory="10000"

eternal="false"

timeToIdleSeconds="300"

timeToLiveSeconds="600"

overflowToDisk="true"

/>

<!-- Sample cache named sampleCache2

This cache contains 1000 elements. Elements will always be held in memory.

They are not expired. -->

<cache name="sampleCache2"

maxElementsInMemory="1000"

eternal="true"

timeToIdleSeconds="0"

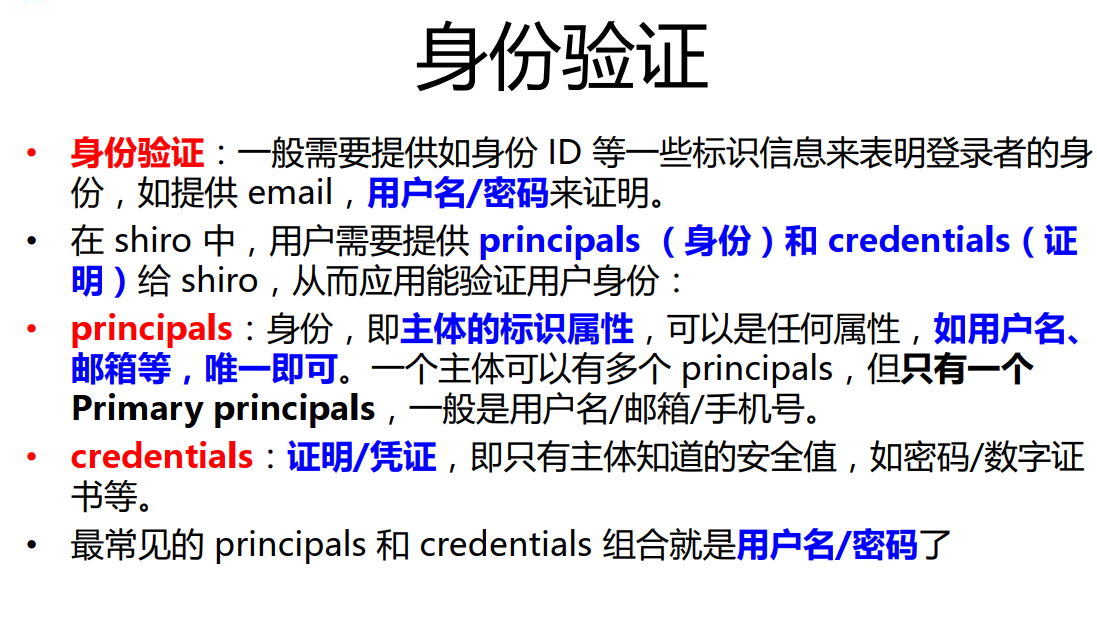
timeToLiveSeconds="0"

overflowToDisk="false"

/> -->

<!-- Place configuration for your caches following -->

</ehcache>



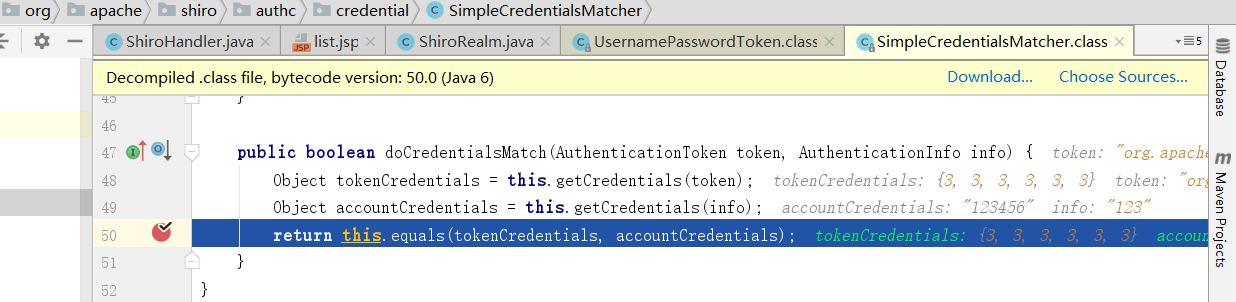
身份认证写在realm里面

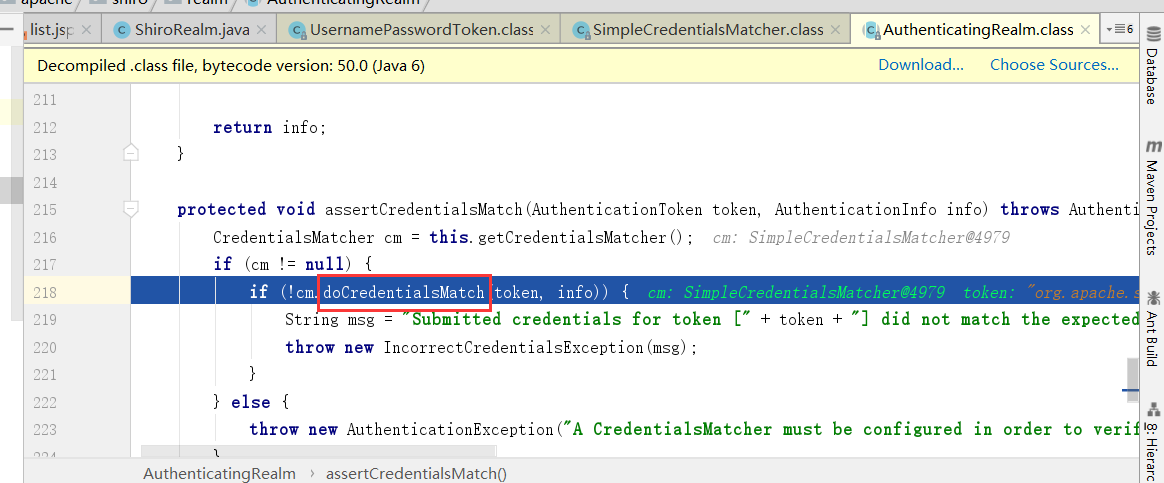




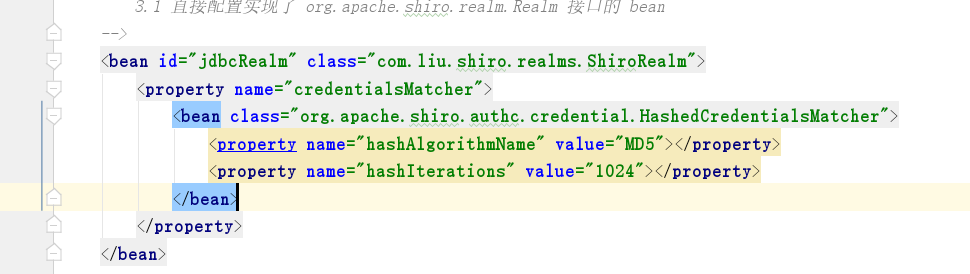
## 密码对比

对比时，一定会调用UsernamePasswordToken. getCredentials 在此打断点可以找到，比较密码的代码块。



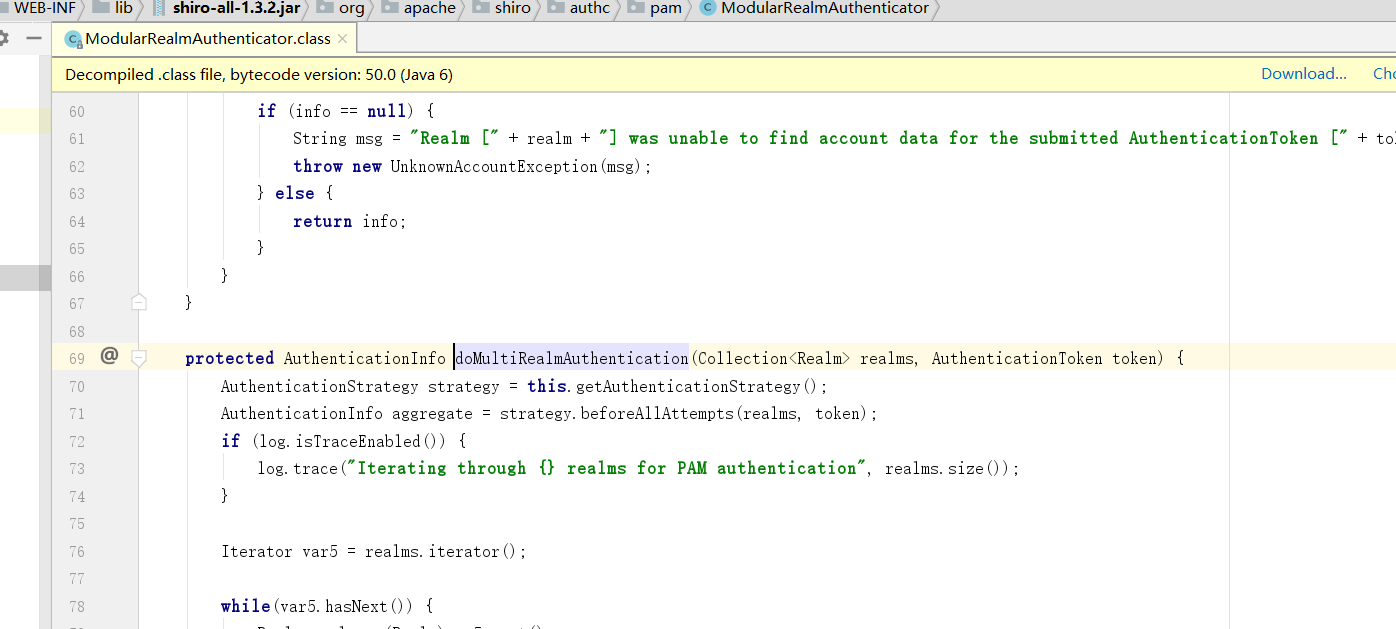


所以需要替换掉Matcher就可以进行MD5对比认证了。

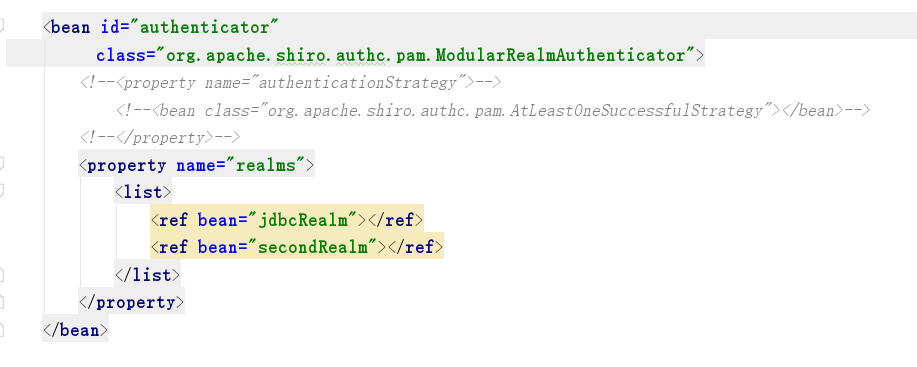


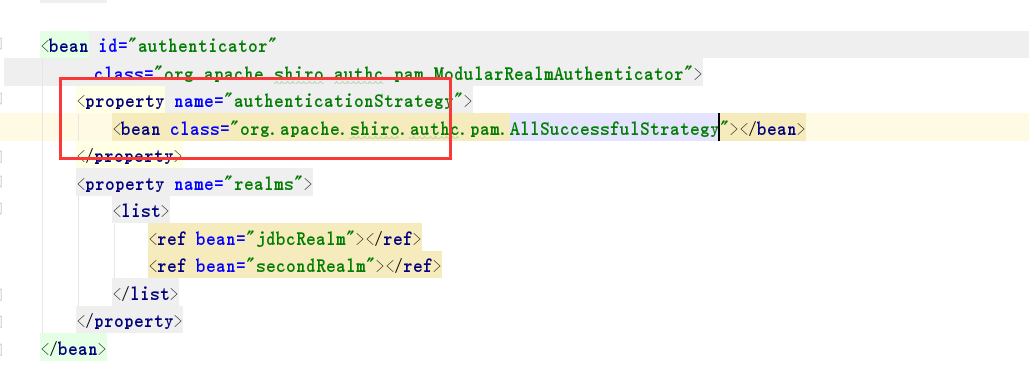
## 多realm





需要配置单独的bean ModularRealmAuthenticator 给 securityManager

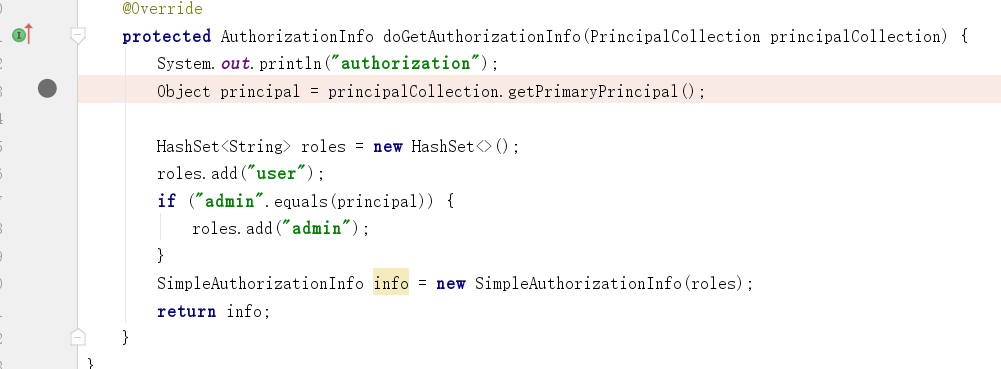
可以修改认证策略



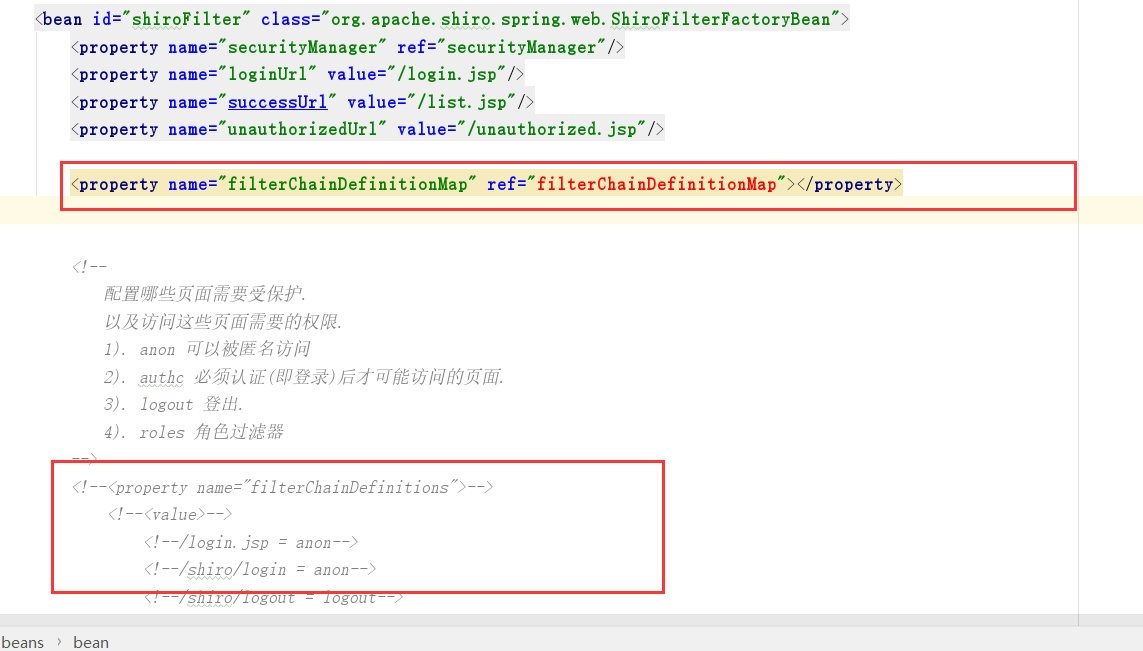
## 授权

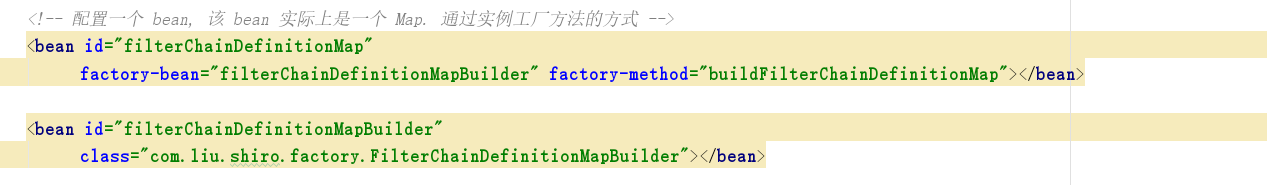
Realm继承AuthorizingRealm

实现方法。

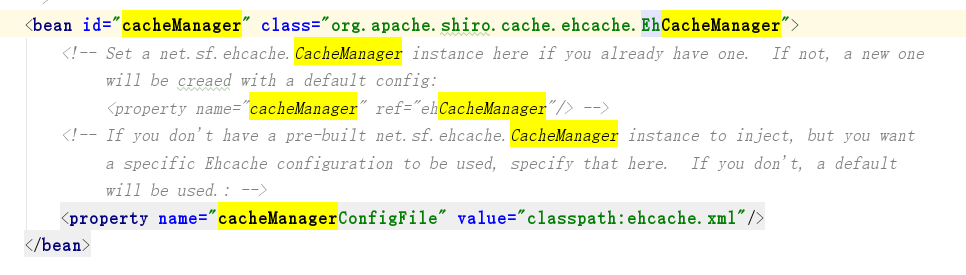


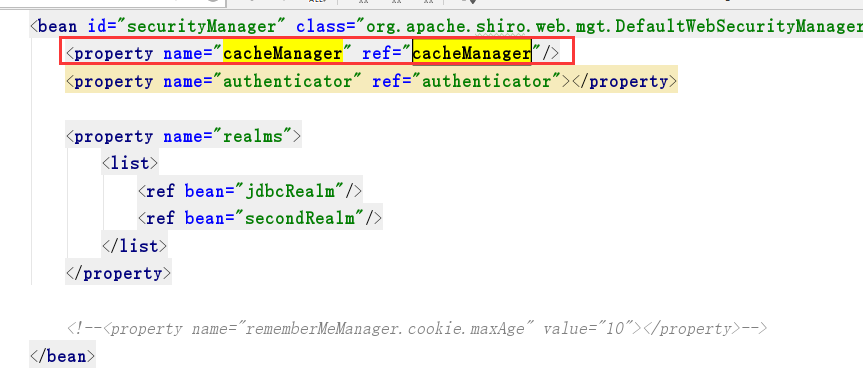
### 拦截器修改成数据库取出方式





## 缓存配置





可以指定realm缓存策略。

