11.测试方法安全

这节我们家过年回讲述如何使用Spring Security的Test支持去测试方法安全。我们首先介绍一个`MessageService`，用户为了访问它需要进行授权。

```
public class HelloMessageService implements MessageService {

@PreAuthorize("authenticated")
public String getMessage() {
        Authentication authentication = SecurityContextHolder.getContext()

.getAuthentication();
        return "Hello " + authentication;
}
}
```

`getMessage`的结果是返回一个对当前Spring Security `Authentication` say hello的字符串。输出的例子如下：

```
Hello
org.springframework.security.authentication.UsernamePasswordAuthenticationToken@ca
25360: Principal: org.springframework.security.core.userdetails.User@36ebcb:
Username: user; Password: [PROTECTED]; Enabled: true; AccountNonExpired: true;
credentialsNonExpired: true; AccountNonLocked: true; Granted Authorities:
ROLE_USER; Credentials: [PROTECTED]; Authenticated: true; Details: null; Granted
Authorities: ROLE_USER
```

11.1 Security测试配置

在我们使用Spring Security Test支持时，我们要进行一些配置，下面是一个例子：

```
@RunWith(SpringJUnit4ClassRunner.class) 1
@ContextConfiguration 2
public class WithMockUserTests {
```

这是启动Spring Security Test的基本例子：

- `@RunWith` 指示spring-test模块它应该创建一个ApplicationContext。这与使用现有的Spring Test支持没有区别。对于另外的信息，看Spring 文档去吧。
- `@ContextConfiguration`指示spring-test用来创建ApplicationContext的配置。如果没有配置被指定，将会使用默认配置位置。这与使用现有的Spring Test支持没有区别。对于另外的信息，看Spring 文档去吧。

Spring Security使用`WithSecurityContextTestExecutionListener`挂钩到Spring Test支持中，这将确保我们的测试以正确的用户运行。它通过在运行我们的测试之前填充`SecurityContextHolder`来完成此操作。在测试

完成后，将会清除SecurityContextHolder。如果你仅仅需要有关Spring的支持，你可以使用@ContextConfiguration代替@SecurityTestExecutionListeners。

记得子啊我们的HelloMessageService添加上@PreAuthorize注解，以便于需要一个被认证的用户去执行它。如果你运行下面的测试，我们期望下面的测试通过：

```
@Test(expected = AuthenticationCredentialsNotFoundException.class)
public void getMessageUnauthenticated() {
    messageService.getMessage();
}
```

11.2 @WithMockUser

我们如何能使用一个特定用户最简单的进行测试呢？答案是使用@WithMockUser。下面的测试将会被运行靠使用一个用户名为"user"，密码为"password"和角色为"ROLE_USER"的用户。

```
@Test
@WithMockUser
public void getMessageWithMockUser() {
String message = messageService.getMessage();
...
}
```

具体如下：

- 用户名为"user"的用户没必要真是存在，因为我们正在模拟这个用户。
- 在SecurityContext中填充的Authentication类型为UsernamePasswordAuthenticationToken。
- 在Authentication中的实体是一个User对象。
- 这个User的用户名为"user"，密码是"password"，有一个单一的GrantedAuthority名字是"ROLE_USER"。

我们的例子很好，因为我们可以利用很多默认值。如果我们想要去使用一个不同的用户名去进行测试要怎么做呢？