打开security.yaml配置文件，在配置文件中，我们通过配置access_control，将管理端的页面进行了保护。我们注释这行配置，打开管理端的controller，打开DashboardController，我们还可以通过使用注解的方式将管理端保护起来。

在DashboardController index()方法前，我们添加IsGranted注解。注解的参数，我们可以使用ROLE_SUPER_ADMIN进行保护。回到浏览器，我们访问管理端同样的跳转到了登录页面。我们的管理端已经被保护了起来。

回到项目，DashboardController它继承了AbstractDashboardController，我们查看父类，按着command键点击父类。父类继承了AbstractController，我们查看AbstractController，我们查看一下类中的方法。在类中有一个isGranted()方法。我们可以在controller方法中可以通过使用isGranted()方法来进行权限的验证。

回到DashboardController，我们删除IsGranted注解，在index()方法中我们添加一个条件判断，if($this->isGranted())。参数这里我们仍然传入ROLE_SUPER_ADMIN，这里我们要添加一个反向判断。如果用户没有这个角色，我们需要抛出一个异常，throw new AccessDeniedException()。这里我们选择安全系统下的exception。

回到浏览器再次访问管理端，现在我们的管理端仍然被保护起来。我们修改Fixtures类添加一个用户，用户对象叫做$tom，用户名叫做tom，我们不设置他的角色，只是一个普通的用户，密码我们设置为tom。对象$tom。最后保存一下$tom。这里要修改一下，重置一下数据库，现在我们使用tom用户来登录一下系统。

点击登录，抛出了403错误，没有访问的权限。回到项目，我们继续查看AbstractController，在类中有个denyAccessUnlessGranted()，我们查看一下这个方法，这个方法是对isGranted()方法的一个封装。如果没有这个权限的话，就抛出一个AccessEeniedException异常。

我们可以在Controller中使用这个方法，回到DashboardController。注释之前的代码，直接使用$this->denyAccessUnlessGranted()，参数这里我们输入ROLE_SUPER_ADMIN，回到浏览器。

```
#src/Controller/Admin/DashboardController.php

class DashboardController extends AbstractDashboardController
{
    /**
     * @Route("/admin", name="admin")
     */
    public function index(): Response
    {
//        if (!$this->isGranted('ROLE_SUPER_ADMIN')){
//            throw new AccessDeniedException();
//        }
        $this->denyAccessUnlessGranted('ROLE_SUPER_ADMIN ');
        return parent::index();
    }

    // ...
}
```

我们再次进行访问，现在我们的tom用户已经登录了，我们再次访问管理端，仍然提示没有权限访问。在controller方法中，我们可以使用注解或者使用这两个方法来对用户的权限进行验证。

现在我们通过对用户的角色进行了验证，Symfony还内置了其他几种属性可以进行验证。在下节课我们来了解一下这些属性。