Spring Security是一款基于Spring的安全框架，主要包含认证和授权两大安全模块，和另外一款流行的安全框架Apache Shiro相比，它拥有更为强大的功能。Spring Security也可以轻松的自定义扩展以满足各种需求，并且对常见的Web安全攻击提供了防护支持。

#### 1、开启Spring Security

引入依赖：

<dependency>

<groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-security</artifactId>  
</dependency>

引入Spring Security依赖时，项目会默认开启如下配置：

security:

basic:  
 enabled: true

这个配置开启了一个HTTP Basic类型的认证，所有服务的访问都必须先过这个认证，默认用户名为user，密码由Spring Security自动生成（控制台日志查看）。

当访问接口时，可看到页面弹出HTTP Basic认证框：



#### 2、基于表单认证

可以通过一些配置将HTTP Basic认证修改为表单的认证方式。

创建一个配置类BrowserSecurityConfig继承org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapter这个抽象类并重写configure(HttpSecurity http)方法。WebSecurityConfigurerAdapter是由Spring Security提供的Web应用安全配置的适配器：

@Configuration

public class BrowserSecurityConfig extends WebSecurityConfigurerAdapter {  
   
 @Override  
 protected void configure(HttpSecurity http) throws Exception {  
 // 提供链式方法调用  
  
 http.formLogin() // 表单登录  
 // http.httpBasic() // 默认开启的 HTTP Basic 类型的认证  
 .and()  
 .authorizeRequests() // 授权配置  
 .anyRequest() // 所有请求  
 .authenticated(); // 都需要认证  
// .permitAll() // 都不需要认证  
 }  
}

#### 3、基本原理

上面开启了一个简单的Spring Security安全配置，通过上面的配置，代码执行过程可以简化为下图表示：



Spring Security包含了众多的过滤器，这些过滤器形成了一条链，所有请求都必须通过这些过滤器后才能成功访问到资源。

* UsernamePasswordAuthenticationFilter过滤器：用于处理基于表单方式的登录认证。
* BasicAuthenticationFilter过滤器：用于处理基于HTTP Basic方式的登录验证。
* 后面还有一系列的过滤器（可以通过相应配置开启）。。。
* 在过滤器链的末尾是一个名为FilterSecurityInterceptor的拦截器，用于判断当前请求身份认证是否成功，是否有相应的权限，当身份认证失败或者权限不足的时候便会抛出相应的异常。ExceptionTranslationFilter捕获并处理，比如需要身份认证时将请求重定向到相应的认证页面，失败或者权限不足时返回相应的提示信息。