# 什么是Spring Security

Spring 是一个非常流行和成功的 Java 应用开发框架。Spring Security 基于 Spring 框架，提供了一套 Web 应用安全性的完整解决方案。

Spring Security是一个高度自定义的安全框架，利用Spring的IOC/DI和AOP功能，为系统提供了声明式安全访问功能。减少了为了系统安全而编写大量重复代码的工作。

关于安全方面的两个主要区域是认证和授权（访问控制）。这也是spring Security重要核心功能。认证即系统认为用户是否登录。授权即系统判断用户是否有权限去做某事。

# Springboot中使用Spring Security登录

Spring Security默认环境中，拦截所有的请求，要求必须认证（登录）后才能访问。默认用户是user,每次启动的时候，动态生成一个长度为128字节的密码,在控制台输出。

默认环境中，提供登录的页面请求地址是：/login,请求方式是：get。

默认环境中，处理登录请求的地址是：/login，请求方式是post。

## 自定义用户名和密码

#静态用户，一般只在内部网络认证中使用,如内部服务器1访问服务器2。

spring:

security:

user:

name: root #设置静态用户名

password: root #设置静态登录密码

## UserDetailsService

当什么也没有配置的时候，账号和密码时由sprjing security定义生成的，实际项目中账号和用户名是从数据库中查询出来的，然后通过自定义逻辑控制认证逻辑。

如果需要自定义逻辑时，只需要实现**UserDetailsService**接口即可。

## 自定义登录逻辑

当进行自定义登录逻辑是需要用到UserDetailService和PasswordEncoder。

Spring Security要求：当自定义登录逻辑时容器必须有PasswordEncoder实例。所以不能直接new对象。

1. 编写配置类(SccurityConfig)

@Configration

public class ScurityConfig{

@Bean

public PasswordEncoder getPwdEncoder(){

return new PasswordEncoder();

}

1. 编写方法根据用户名查询用户，密码的验证在内存中处理。

Spring Scurity需要使用一个服务对象，Spring Scurity提供了一个接口，命名为UserDetailsService，其中定义了唯一认证的方法，命名为loadUserByUsername。方法代表用户认证，抛出异常：UsernameNotFoundException，当用户查询失败的时候抛出。

用户查询只能用用户名作为条件，查询用户。

密码匹配是有SpringSecurity内部逻辑自动完成的，只需要包查询的用户名正确返回即可。返回结果时通过UserDetails类型的实现类，可以自定义或使用Security框架提供的实现类。Security框架提供了UserDetails接口实现类型User,构造时，需要提供3个参数或7个参数。3个参数：用户身份(用户名)、用户正确密码（数据库中的密码）,权限集合(Collection集合)

AuthorityUtils工具类，可以通过字符串，创建权限集合。

SpringScurity内部自动匹配密码是否正确的时候，一定要进行加密和解密处理，要求spring容器中必须存在一个PassWordEncoder实现类对象。对象中提供加密逻辑和解密逻辑。解密逻辑不是要求反向计算明文，而是要求实现明文和密文的验证。解密验证和加密验证都可以。

自定义凭证匹配器实现PassWordEncoder接口，专门用于做流程的凭证校验使用的类型；其中要两个核心的方法：

encode(CharSequence charsequence)：把明文密码加密成密文密码，其中charsequence是页面收集的源密码

matches(CharSequence charsequence，String s)：校验明文和密文是否匹配，其中charsequence是页面收集的源密码，s是存储在数据库中的密码；

## SpringScurity默认流程

## 自定义登录页面

SpringSecurity提供了默认的登录页面，实际开发中我们希望使用自定义的登录页面。只需要修改配置类即可。