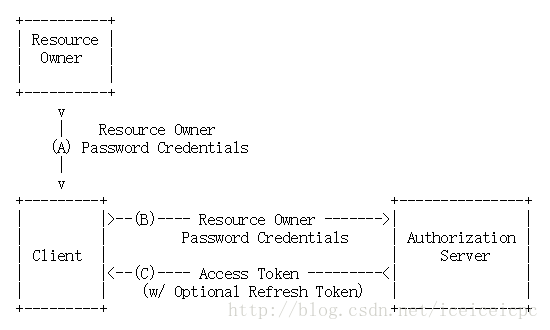
OAuth简单说就是一种授权的**协议**，只要授权方和被授权方遵守这个协议去写代码提供服务，那双方就是实现了OAuth模式，像我们经常使用的qq、微信第三方登录都是基于oauth2.0实现的。

Oauth2.0主要有四种实现模式：

* 授权码（Authorization）模式(获取code、code换取access\_token)
* 简化（Impilict）模式(直接换取access\_token，基本不用)
* 密码（Resource Owner Password）模式(客户端向用户索取账号密码，然后客户端向服务端索取授权)
* 客户端（Client）模式（客户端以自己的名义要求"服务提供商"提供服务；场景：提供接口服务)

比较常用的两种模式就是授权码模式和密码模式，由于业务需要先只对密码模式进行了学习与研究。

Oauth2.0密码模式实现流程图：



它的步骤如下：

（A）用户向客户端提供用户名和密码。   
（B）客户端将用户名和密码发给认证服务器，向后者请求令牌。   
（C）认证服务器确认无误后，向客户端提供访问令牌。

所用技术：springboot2.0+ spring-security-oauth2实现

搭建一个springboot工程，pom依赖如下：

|  |
| --- |
| <dependency>  <groupId>org.springframework.security.oauth</groupId>  <artifactId>spring-security-oauth2</artifactId>  </dependency>  <dependency>  <groupId>org.mybatis.spring.boot</groupId>  <artifactId>mybatis-spring-boot-starter</artifactId>  </dependency> |

Oauth2.0分为认证服务、资源控制服务。

认证服务：

|  |
| --- |
| @Configuration @EnableAuthorizationServer 该注解为开启认证服务 public class AuthorizationServerConfiguration{} |

资源控制服务：

|  |
| --- |
| @Configuration @EnableResourceServer 该注解为资源访问控制服务，默认为都可以访问 public class ResourceServerConfig { } |

只需要配置这两个注解，我们的spring-security-oauth2就生效了，在项目启动的控制台会打印出三条信息

|  |
| --- |
|  |

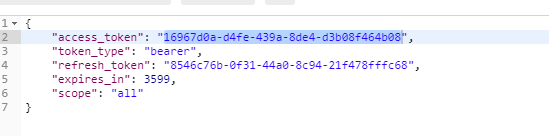
我们访问一下我们的测试接口：

|  |
| --- |
|  |

可以看到我们需要认证通过才能访问。下面我们通过上面的信息去获取token

|  |
| --- |
|  |

请求之后的结果：



然后我们通过携带token去访问接口：

|  |
| --- |
|  |

到此一个简单的oauth服务就算完成了。

目前实现了通过jdbc的方式进行认证及保存token，代码在https://github.com/wcl157321/OAuthDemo不一一贴图了自行下载吧。

有什么不对的地方，请指正。

Ps：用jdbc的方式必须需要这三张表：

|  |
| --- |
| CREATE TABLE `oauth\_access\_token` (  `token\_id` varchar(256) DEFAULT NULL,  `token` blob,  `authentication\_id` varchar(250) NOT NULL,  `user\_name` varchar(256) DEFAULT NULL,  `client\_id` varchar(256) DEFAULT NULL,  `authentication` blob,  `refresh\_token` varchar(256) DEFAULT NULL,  PRIMARY KEY (`authentication\_id`) USING BTREE  ) ENGINE=InnoDB DEFAULT CHARSET=utf8 ROW\_FORMAT=DYNAMIC;  CREATE TABLE `oauth\_client\_details` (  `client\_id` varchar(250) NOT NULL,  `resource\_ids` varchar(256) DEFAULT NULL,  `client\_secret` varchar(256) DEFAULT NULL,  `scope` varchar(256) DEFAULT NULL,  `authorized\_grant\_types` varchar(256) DEFAULT NULL,  `web\_server\_redirect\_uri` varchar(256) DEFAULT NULL,  `authorities` varchar(256) DEFAULT NULL,  `access\_token\_validity` int(11) DEFAULT NULL,  `refresh\_token\_validity` int(11) DEFAULT NULL,  `additional\_information` varchar(4096) DEFAULT NULL,  `autoapprove` varchar(256) DEFAULT NULL,  PRIMARY KEY (`client\_id`) USING BTREE  ) ENGINE=InnoDB DEFAULT CHARSET=utf8 ROW\_FORMAT=DYNAMIC;  CREATE TABLE `oauth\_refresh\_token` (  `token\_id` varchar(256) DEFAULT NULL,  `token` blob,  `authentication` blob  ) ENGINE=InnoDB DEFAULT CHARSET=utf8 ROW\_FORMAT=DYNAMIC; |