# **基于thinkphp5.1框架搭建OAuth2.0服务端**

**<https://blog.csdn.net/a1264718192/article/details/84710183>**

IoT Boot Camp系列课程是由TorchIoTBootCamp团队发起，广大IoT领域的爱好者共同参与的项目。旨在高效率、高质量地传播IoT领域的相关知识，以促进物与物，人与人，人与信息的连接。  
[点击这里，观看博主更多有趣视频教程](https://space.bilibili.com/605447248?spm_id_from=333.33.b_73656375726974794f75744c696e6b.1)  
[点击这里，学习更多知识](https://github.com/MarkDing/IoT-Developer-Boot-Camp/wiki)

OAuth是用于服务端与客户端授权登录的协议，OAuth2.0是OAuth的第二个版本，关于OAuth2.0的基础知识，可以阅读阮一峰的一篇博文，对OAuth2.0的介绍非常详细，只要理解了OAuth2.0的授权过程，在自己网站实现OAuth2.0并不复杂。

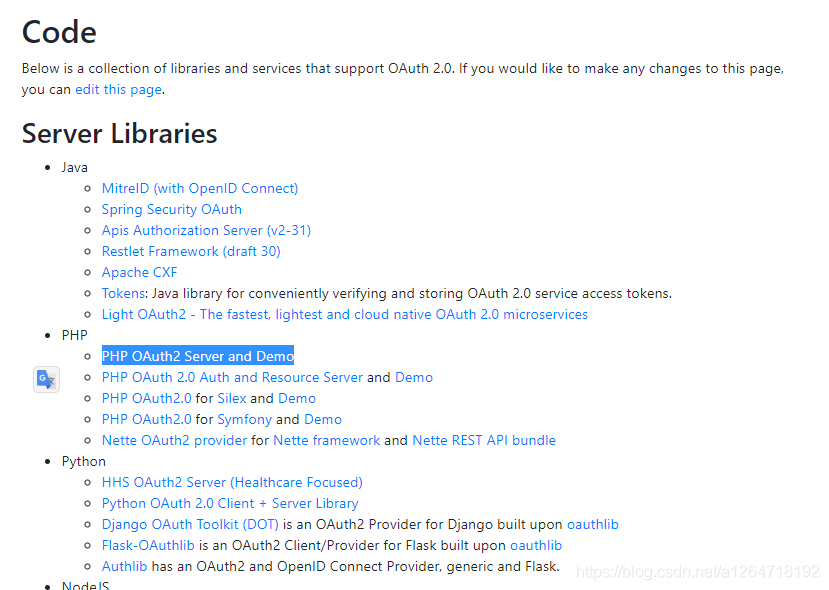
本文将讲解如何基于thinkphp5.1的框架实现OAuth2.0的服务端。

# **1 环境搭建**

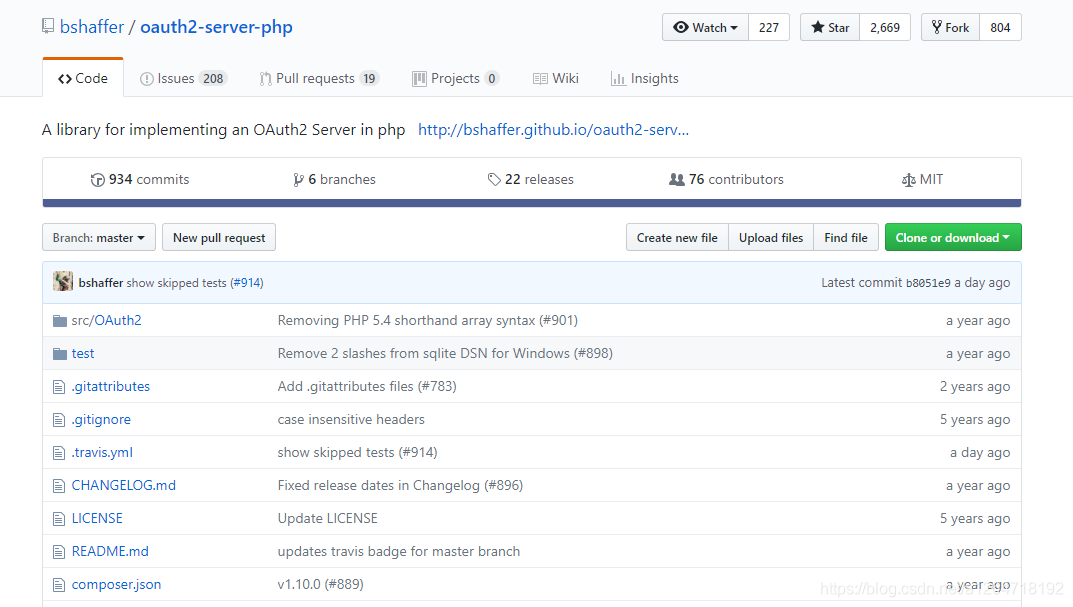
首先确保你已经搭建好了服务器，并且已经能够正常访问你的服务器。我的环境Xampp+thinkphp5.1.

# **2 安装OAuth2.0 php包**

你页根据OAuth2.0的协议自己去实现代码，但是最快捷最安全最可靠的方法当然是移植第三方OAuth2.0包。OAuth官网提供了很多第三方包，详见网站<https://oauth.net/code/>， 如下图，因为thinkphp是基于php语言，因此我选择了PHP下第一个。



点击[PHP OAuth2 Server](https://github.com/bshaffer/oauth2-server-php)会跳入源码下载库，将其下载到电脑即可。



下载后解压，我们只需要将里面/src/OAuth文件夹整个拷贝到tp5/extend/目录下，就可以自动注册对应的命名空间。之后我们就可以使用\OAuth2\...的方式去使用OAuth里面的任何方法。

# **3 实现OAuth服务端**

## **3.1 创建数据库**

由于我们之前下载的OAuth包有用到很多数据表，所以需要按照其要求创建好数据表，创建代码如下：

CREATE TABLE oauth\_clients (

client\_id VARCHAR(80) NOT NULL,

client\_secret VARCHAR(80),

redirect\_uri VARCHAR(2000),

grant\_types VARCHAR(80),

scope VARCHAR(4000),

user\_id VARCHAR(80),

PRIMARY KEY (client\_id)

);

CREATE TABLE oauth\_access\_tokens (

access\_token VARCHAR(40) NOT NULL,

client\_id VARCHAR(80) NOT NULL,

user\_id VARCHAR(80),

expires TIMESTAMP NOT NULL,

scope VARCHAR(4000),

PRIMARY KEY (access\_token)

);

CREATE TABLE oauth\_authorization\_codes (

authorization\_code VARCHAR(40) NOT NULL,

client\_id VARCHAR(80) NOT NULL,

user\_id VARCHAR(80),

redirect\_uri VARCHAR(2000),

expires TIMESTAMP NOT NULL,

scope VARCHAR(4000),

id\_token VARCHAR(1000),

PRIMARY KEY (authorization\_code)

);

CREATE TABLE oauth\_refresh\_tokens (

refresh\_token VARCHAR(40) NOT NULL,

client\_id VARCHAR(80) NOT NULL,

user\_id VARCHAR(80),

expires TIMESTAMP NOT NULL,

scope VARCHAR(4000),

PRIMARY KEY (refresh\_token)

);

CREATE TABLE oauth\_users (

username VARCHAR(80),

password VARCHAR(80),

first\_name VARCHAR(80),

last\_name VARCHAR(80),

email VARCHAR(80),

email\_verified BOOLEAN,

scope VARCHAR(4000)

);

CREATE TABLE oauth\_scopes (

scope VARCHAR(80) NOT NULL,

is\_default BOOLEAN,

PRIMARY KEY (scope)

);

CREATE TABLE oauth\_jwt (

client\_id VARCHAR(80) NOT NULL,

subject VARCHAR(80),

public\_key VARCHAR(2000) NOT NULL

);

## **3.2 创建控制器**

需要在tp5/application/index/controller下创建一个控制器，命名为OAuth.php,写入以下代码，控制器就创建完成了。

<?php

namespace app\index\controller;

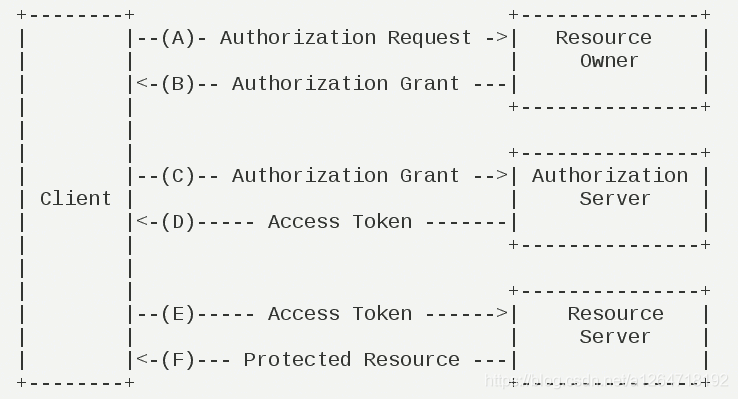
class OAuth extends \think\Controller

{

}

## **3.3 实现authorize**

OAuth 2.0的运行流程如下图。



所以第一步是实现authorization。

我们在之前创建好的控制器中添加一个函数authorize()

代码如下（注意，dbname需要换成你自己的数据库的名字，下同）：

<?php

namespace app\index\controller;

class OAuth extends \think\Controller

{

public function authorize()

{

global $server;

$dsn = 'mysql:dbname=XXX;host=127.0.0.1';

$username = 'root';

$password = '';

\OAuth2\Autoloader::register();

// $dsn is the Data Source Name for your database, for exmaple "mysql:dbname=my\_oauth2\_db;host=localhost"

$storage = new \OAuth2\Storage\Pdo(array('dsn' => $dsn, 'username' => $username, 'password' => $password));

// Pass a storage object or array of storage objects to the OAuth2 server class

$server = new \OAuth2\Server($storage);

// Add the "Client Credentials" grant type (it is the simplest of the grant types)

$server->addGrantType(new \OAuth2\GrantType\ClientCredentials($storage));

// Add the "Authorization Code" grant type (this is where the oauth magic happens)

$server->addGrantType(new \OAuth2\GrantType\AuthorizationCode($storage));

$request = \OAuth2\Request::createFromGlobals();

$response = new \OAuth2\Response();

// validate the authorize request

if (!$server->validateAuthorizeRequest($request, $response)) {

die;

}

// display an authorization form

if (empty($\_POST)) {

exit('

<form method="post">

<label>Do You Authorize TestClient?</label><br />

<input type="submit" name="authorized" value="yes">

<input type="submit" name="authorized" value="no">

</form>');

}

// print the authorization code if the user has authorized your client

$is\_authorized = ($\_POST['authorized'] === 'yes');

$server->handleAuthorizeRequest($request, $response, $is\_authorized);

if ($is\_authorized) {

// this is only here so that you get to see your code in the cURL request. Otherwise, we'd redirect back to the client

$code = substr($response->getHttpHeader('Location'), strpos($response->getHttpHeader('Location'), 'code=')+5, 40);

exit("SUCCESS! Authorization Code: $code");

}

$response->send();

}

}

在tp5/route/route.php中创建相应路由，post方法和get方法都创建

Route::get('authorize', 'OAuth/authorize');

Route::post('authorize', 'OAuth/authorize');

接下来验证创建的authorize是否成功，通过以下链接去访问，在浏览器中输入以下链接，回车后就会显示一个验证表单，当你点击yes按钮后，如果窗口显示一串字符，那么就表示authorize创建成功了，这串字符就是code，接下来需要通过这个code去获取token。

<http://localhost/authorize.php?response_type=code&client_id=testclient&state=xyz>

3.4 实现token申请方法

在OAuth.php控制器中添加函数token()，代码如下

public function token(){

global $server;

$dsn = 'mysql:dbname=XXX;host=127.0.0.1';

$username = 'root';

$password = '';

\OAuth2\Autoloader::register();

// $dsn is the Data Source Name for your database, for exmaple "mysql:dbname=my\_oauth2\_db;host=localhost"

$storage = new \OAuth2\Storage\Pdo(array('dsn' => $dsn, 'username' => $username, 'password' => $password));

// Pass a storage object or array of storage objects to the OAuth2 server class

$server = new \OAuth2\Server($storage);

// Add the "Client Credentials" grant type (it is the simplest of the grant types)

$server->addGrantType(new \OAuth2\GrantType\ClientCredentials($storage));

// Add the "Authorization Code" grant type (this is where the oauth magic happens)

$server->addGrantType(new \OAuth2\GrantType\AuthorizationCode($storage));

// Handle a request for an OAuth2.0 Access Token and send the response to the client

$server->handleTokenRequest(\OAuth2\Request::createFromGlobals())->send();

}

在tp5/route/route.php中创建相应路由，post方法和get方法都创建

Route::get('token', 'OAuth/token');

Route::post('token', 'OAuth/token');

在测试是否获取token之前，我们需要在oauth\_clients表中加一条数据，可执行如下SQL：

INSERT INTO oauth\_clients (client\_id, client\_secret, redirect\_uri) VALUES ("testclient", "testpass", "http://fake/");

接下来从CMD运行以下内容，注意：code的值需要换成你上一步生成的code

curl -u testclient:testpass http://localhost/token.php -d 'grant\_type=authorization\_code&code=YOUR\_CODE'

如果成功的话，你应该会得到access token，如下内容

{"access\_token":"6f05ad622a3d32a5a81aee5d73a5826adb8cbf63","expires\_in":3600,"token\_type":"bearer","scope":null}

3.5 实现Resource获取

 在OAuth.php控制器中添加函数resource()，代码如下

public function resource()

{

// include our OAuth2 Server object

global $server;

$dsn = 'mysql:dbname=XXX;host=127.0.0.1';

$username = 'root';

$password = '';

\OAuth2\Autoloader::register();

// $dsn is the Data Source Name for your database, for exmaple "mysql:dbname=my\_oauth2\_db;host=localhost"

$storage = new \OAuth2\Storage\Pdo(array('dsn' => $dsn, 'username' => $username, 'password' => $password));

// Pass a storage object or array of storage objects to the OAuth2 server class

$server = new \OAuth2\Server($storage);

// Add the "Client Credentials" grant type (it is the simplest of the grant types)

$server->addGrantType(new \OAuth2\GrantType\ClientCredentials($storage));

// Add the "Authorization Code" grant type (this is where the oauth magic happens)

$server->addGrantType(new \OAuth2\GrantType\AuthorizationCode($storage));

// Handle a request to a resource and authenticate the access token

if (!$server->verifyResourceRequest(\OAuth2\Request::createFromGlobals())) {

$server->getResponse()->send();

die;

}

echo json\_encode(array('success' => true, 'message' => 'You accessed my APIs!'));

}

在tp5/route/route.php中创建相应路由，post方法和get方法都创建

Route::get('resource', 'OAuth/resource');

Route::post('resource', 'OAuth/resource');

验证：通过CMD运行以下内容（将access token的值换成上一次获取的access token）：

curl http://localhost/resource.php -d 'access\_token=YOUR\_TOKEN'

如果成功，将会获得以下响应：

{"success":true,"message":"You accessed my APIs!"}

# **4 总结**

至此，OAuth所有相关的都实现了，整个过程就是客户端去想服务端申请token，然后拿着这个token向服务端获取资源的过程。后续有什么不明白的地方，大家可以在下面评论，我有时间会回答大家。关于oauth2-server-php库的更多详情，大家可以访问http://bshaffer.github.io/oauth2-server-php-docs/。