TTools Documentation

Release 2.1

Erika Heidi

Aug 06, 2018

Contents

TTools (Twitter Tools) Library aims to make life easier for twitter app developers, providing a simple workflow for authentication, while maintaining a high-level of flexibility for various types of applications.

CHAPTER 1

Getting Started

TTools is a clean and straightforward library for dealing with the Twitter API.

1.1 Requirements

TTools only requires the php5-curl extension.

1.2 Installing

Installation can be easily made through Composer:

```
$ composer require ttools/ttools
```

Or, add ttools/ttools directly to your composer.json:

```
{
    "require": {
        "ttools/ttools": "~2.1"
    }
}
```

After running *composer install/update* you will be able to use TTools in your application.

1.3 Creating your first application

The easiest way to play around with the Twitter API is creating a single-user application. This means you will not need to authenticate users, as you will be working with fixed user API keys.

The code below shows the authenticated user (the owner of the provided API keys) timeline:

```
<?php
require( __DIR__ . '/../vendor/autoload.php');

$config = array(
    'consumer_key'        => 'API_KEY',
    'consumer_secret'     => 'API_SECRET',
    'access_token'        => 'USER_TOKEN',
    'access_token_secret' => 'USER_TOKEN_SECRET',
);

$app = new \TTools\App($config);
$timeline = $app->getTimeline();
print_r($timeline);
```

For this example to work, you need to have the 4 keys necessary to authenticate requests.

To get your keys, you need to first register your application at http://dev.twitter.com . On the application details page, there's an option to generate your access tokens; use them along with the application tokens (*API Key* and *API Secret*) in the config array (as shown in the example above) and you will have a ready-to-request Twitter App.

If you never created a Twitter App before, check this step-by-step guide.

CHAPTER 2

Single User Application

The most basic usage of TTools is for a single user application, like a Twitter Bot. As no authorization is required, you don't need to worry about Request and Session Storage.

For this you will need to have all 4 keys needed to authenticate a request:

- **consumer_key**: The application API Key (also known as *consumer key*)
- **consumer_secret**: The application API Secret (also known as *consumer secret*)
- **access_token**: The User Access Token
- **access_token_secret**: The User Access Token Secret

You can obtain these keys after registering your application on Twitter: https://dev.twitter.com. On the application details page, there's an option to generate the access tokens for your user.

If you never created a Twitter App before, check this step-by-step guide.

2.1 Example

The code to create a single user application is very simple, you don't need to deal with Request or Session storage. Below is an example:

```
$config = array(
    'consumer_key'        => 'APP_CONSUMER_KEY',
    'consumer_secret'     => 'APP_CONSUMER_SECRET',
    'access_token'        => 'USER_TOKEN',
    'access_token_secret' => 'USER_TOKEN_SECRET',
);

$app = new \TTools\App($config);

echo "static user credentials:<br/><pre>";
print_r($app->getCredentials());
```

(continues on next page)

(continued from previous page)

```
echo "</pre><br>last req info:<br>";
print_r($app->getLastReqInfo());
```

For a more real-life example, check the Great Zoltar application (a Twitter auto-reply bot): https://github.com/erikaheidi/greatzoltar

CHAPTER 3

Multi-user Application

For authenticating users with Twitter, you just need the application keys (API key, a.k.a. *consumer_key*, and API secret, a.k.a *consumer_secret*), but you need to store tokens and deal with the request response. Luckily, TTools has a very simple workflow and comes with some default providers to deal with this.

You'll need:

- **consumer_key**: The application API Key (also known as *consumer key*)
- **consumer_secret**: The application API Secret (also known as *consumer secret*)

You can obtain this keys after registering your application on Twitter: https://dev.twitter.com If you never created a Twitter App before, check this step-by-step guide.

3.1 Example

This example uses the basic providers that comes with TTools. It will use the default PHP session for storing the request keys and authenticate the user. After the user has successfully authenticated, the app will print the user's timeline:

```
$config = array(
    'consumer_key'    => 'APP_CONSUMER_KEY',
    'consumer_secret' => 'APP_CONSUMER_SECRET'
);

$app = new \TTools\App($config);

if ($app->isLogged()) {
    $user = $app->getCurrentUser();

    echo "<h3>Logged in as @". $user['screen_name'] . "</h3>";
    echo '<p>[ <a href=".?logout=1">Logout</a> ]';

    $tl = $app->getTimeline();
```

(continues on next page)

(continued from previous page)

```
    echo "<h3>Your Timeline</h3><pre>";
    print_r($tl);
    echo "</pre>";

    echo "Last Req Info:<br/>";
    print_r($app->getLastReqInfo());
} else {
    $login_url = $app->getLoginUrl();
    echo 'Please log in: <a href="'. $login_url . '">' . $login_url . '</a>';
}
```

CHAPTER 4

Making Requests to the Twitter API

TTools has a couple helper methods to get you started, but you can make any request you want to the Twitter API.

For the examples below, we are going to consider that you already have an APP configured with authorization tokens. Have a look at the basic single-user and basic multi-user examples to get started.

Request responses come as **arrays** representing the Twitter API objects (json decoded).

for all the snippets, consider this initialization code (using your keys):

```
$config = array(
'consumer_key'        => 'APP_CONSUMER_KEY',
'consumer_secret'     => 'APP_CONSUMER_SECRET',
'access_token'        => 'USER_TOKEN',
'access_token_secret' => 'USER_TOKEN_SECRET',
);

$app = new \TTools\App($config);
```

4.1 Getting User Timeline

using the helper method getTimeline():

```
$timeline = $app->getTimeline();
print_r($timeline);
```

making the request manually:

```
$timeline = $app->get('/statuses/home_timeline.json',array("count"=> 10));
print_r($timeline);
```

4.2 Getting User Mentions

using the helper method getMentions():

```
$mentions = $app->getMentions();
print_r($mentions);
```

making the request manually:

```
$mentions = $app->get('/statuses/mentions_timeline.json',array("count"=> 10));
print_r($mentions);
```

4.3 Posting a Tweet

using the helper method update():

```
$app->update('This is my awesome tweet update');
```

manually posting an update:

```
$app->post('/statuses/update.json', array('status' => 'This is my awesome tweet update
↪');
```

4.4 Posting an Image

using the helper method updateWithMedia():

```
$app->updateWithMedia('path_to_my_awesome_image', 'This is my awesome image');
```

4.5 Other

Consult the Twitter API documentation to see what else you can do with the API. You can use the manual requests with any endpoint using **$app->get** and **$app->post**.

CHAPTER 5

TTools with Silex

This is a very basic example usage for authorizing users from a Silex Application. TTools has Silex ServiceProvider to take care of the object initialization using the correct Session and Request instances.

Note: If you want to create a new Twitter application using Silex, have a look at the Twilex Project.

5.1 Example App

The example below is a one-page app that will authenticate the user and show its timeline:

```
<?php
require( __DIR__ . '/../vendor/autoload.php');

$app = new Silex\Application();
$app['debug'] = true;

$app->register(new Silex\Provider\SessionServiceProvider());
$app->register(new TTools\Provider\Silex\TToolsServiceProvider(), array(
    'ttools.consumer_key'        => 'API_KEY',
    'ttools.consumer_secret'     => 'API_SECRET'
));

$app->get('/', function() use($app) {

    if ($app['ttools']->isLogged()) {
        $user = $app['ttools']->getCurrentUser();

        echo "<h3>Logged in as @". $user['screen_name'] . "</h3>";
        echo '<p>[ <a href=".?logout=1">Logout</a> ]';

        $tl = $app['ttools']->getTimeline();

        echo "<h3>Your Timeline</h3><pre>";
```

(continues on next page)

(continued from previous page)

```
        print_r($tl);
        echo "</pre>";

        echo "Last Req Info:<br/>";
        print_r($app['ttools']->getLastReqInfo());
    } else {
        $login_url = $app['ttools']->getLoginUrl();
        echo 'Please log in: <a href="'. $login_url . '">' . $login_url . '</a>';
    }

    return 'Hello !!!';

});

$app->run();
```

CHAPTER 6

TTools with Symfony

This is a very basic example usage for authorizing users from a Symfony Application. TTools comes with Symfony **Session** and **Request** support.

6.1 Authorizing users

In order to use TTools on Symfony, you need to specify the Symfony providers when creating the TTools Application object. A good practice is defining credentials as parameters in your application:

```
#parameters.yml
ttools.credentials:
    consumer_key: 'APP_CONSUMER_KEY'
    consumer_secret: 'APP_CONSUMER_SECRET'
    access_token: 'USER_ACCESS_TOKEN'
    access_token_secret: 'USER_ACCESS_TOKEN_SECRET'
```

then, instantiate the TTools object using the Symfony providers (the method below should go in a controller):

```
use TTools\Provider\Symfony\SymfonyRequestProvider;
use TTools\Provider\Symfony\SymfonyStorageSession;

(...)

public function twitterAction(Request $request)
{
    $sp = new SymfonyStorageSession($this->get('session'));
    $rp = new SymfonyRequestProvider($this->get('request'));
    $config = $this->container->getParameter('ttools.credentials');
    $twitter = new \TTools\App($config, $sp, $rp);

    if ($twitter->isLogged()) {
        $user = $twitter->getCurrentUser();
```

(continues on next page)

(continued from previous page)

```
        return new Response("Logged in as: " . $user['screen_name']);

    } else {
        $login_url = $twitter->getLoginUrl();

        return new RedirectResponse($login_url);
    }
}
```

CHAPTER 7

TTools with Laravel

This is a very basic example usage for setting TTools up with a Laravel Application.

In order to use TTools on Laravel, you need to add the TTools Laravel Service Provider to your Laravel config this'll instruct Laravel on runtime to bind the TTools Application object into the Laravel IoC.

7.1 Configuration

You'll need to add your Twitter credentials into your applications config files.

/app/config/ttools.php

```
<?php

return array(
    // Required Options
    'consumer_key' => '',
    'consumer_secret' => '',
    // Optional
    'access_token' => '',
    'access_token_secret' => '',
    'auth_method' => ''
);
```

/app/config/app.php

Add *'TToolsProviderLaravelTToolsServiceProvider'* to the providers array

Optional

Add *'TTools'* => *'TToolsProviderLaravelTToolsFacade'* to the aliases array

CHAPTER 8

Twitter APP Creation

In order to use TTools and have access to the Twitter API, you need to create an application on the Twitter Developers page. If this is the first time you do this, here you can find a step-by-step guide.

8.1 Sign-up / Sig-in

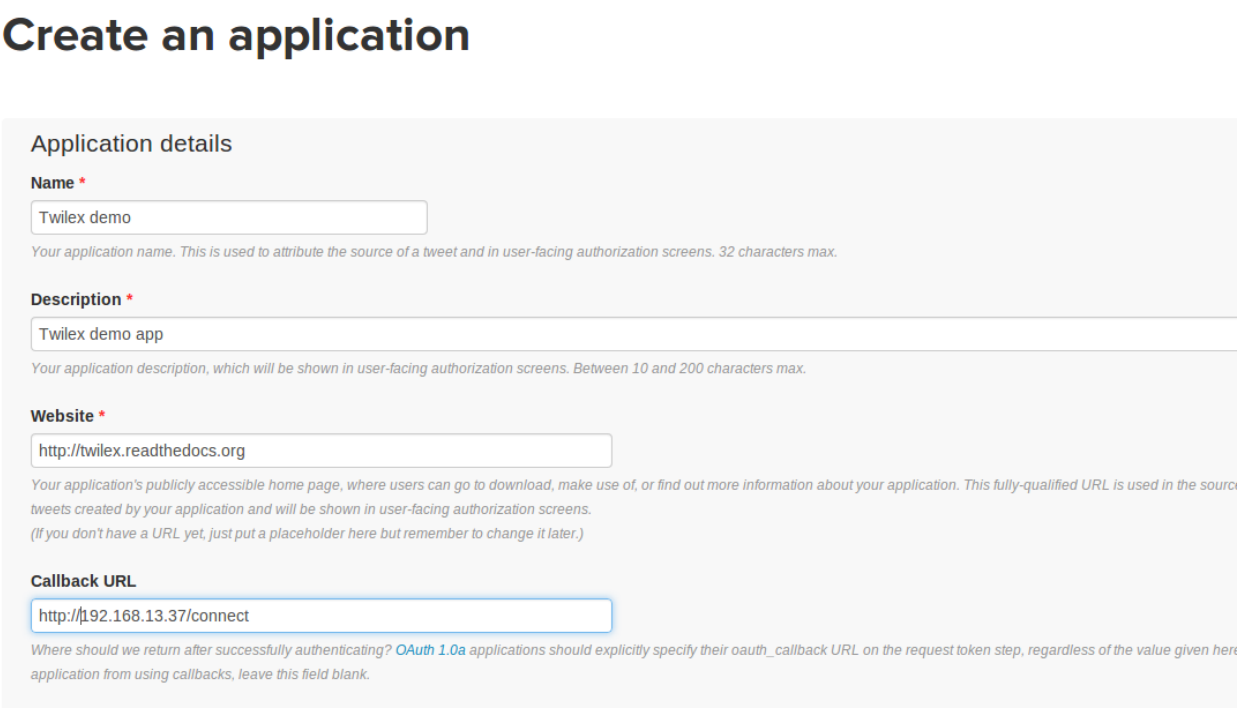
Go to http://dev.twitter.com and sign in.

8.2 Access your applications

Access the item "Manage Your Apps" from the footer.

8.3 Create the application

Now click on "Create New App". You'll fill a form like this:



8.3.1 Name

The application name that will be used in the authorization screen.

8.3.2 Description

A comprehensive description that will be used in the authorization screen.

8.3.3 Website

A website were people can find more information about the application.

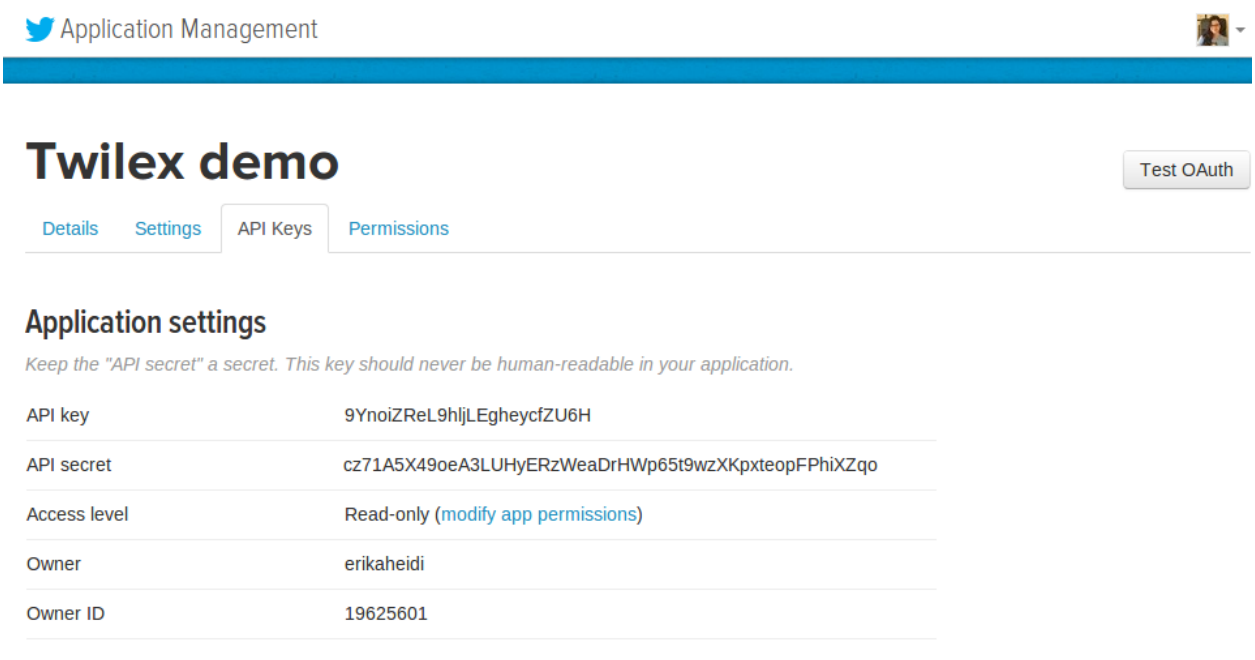
8.3.4 Callback URL

This is **important** when you want to create an application that authenticates users.

The callback URL is a URL to where the user will be redirected right after authorizing the application. You can use the code provided in the basic multi-user application example for this endpoint.

8.4 Now get your Keys

You'll need the **API key** (consumer_key) and **API secret** (consumer_secret) in order to configure TTools. These keys are available at the tab "API Keys" in your application settings:



8.5 Note about app permissions

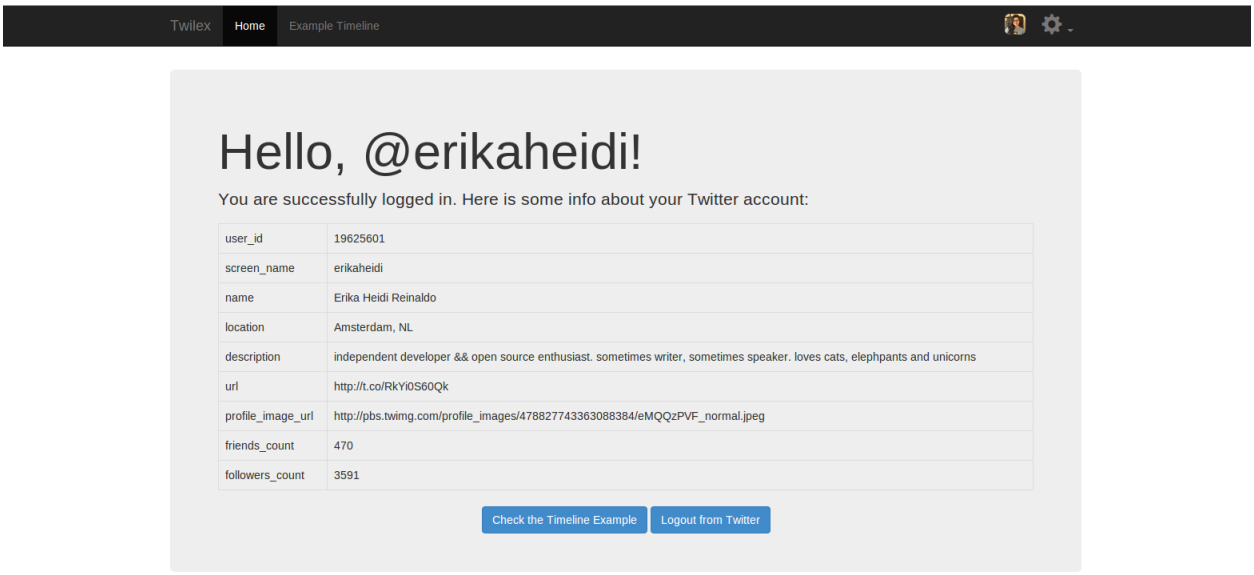
The default permission for new apps is **read only**. With this permission you won't be able to create tweets or change any data. If you think you will need write permission, you can edit your application settings to make the change.

CHAPTER 9

Twilex - a simple framework for Twitter apps

Twilex provides a simple way for bootstrapping Twitter applications, using Silex, Flint and TTools. If you are used to Symfony conventions, it will be really easy for you to get started.

Check the official Twilex docs to get started.



CHAPTER 10

Requirements

TTools only requires the php5-curl extension.

CHAPTER 11

Installing

Installation can be easily made through Composer. Add ttools/ttools to your composer.json:

```
{
    "require": {
        "ttools/ttools": "2.1.*"
    }
}
```

After running *composer install/update* you will be able to use TTools in your application.