

## Blog Export: Rasmus' Toys Page, <http://toys.lerdorf.com/>

Monday, April 27, 2009

### Using pecl/oauth to post to Twitter

I have seen a lot of questions about OAuth and specifically how to do OAuth from PHP. We have a new pecl oauth extension written by John Jawed which does a really good job simplifying OAuth.

I added Twitter support to Slowgeek.com the other day and it was extremely painless. The goal was to let users have a way to have Slowgeek send a tweet on their behalf when they have completed a Nike+ run. Here is a simplified description of what I did.

First, I needed to get the user to authorize Slowgeek to tweet on their behalf. This is done by asking Twitter for an access token and secret which will be stored on Slowgeek. This access token and secret will allow us to act on behalf of the user. This is made a bit easier by the fact that Twitter does not expire access tokens at this point, so I didn't need to worry about an access token refresh workflow.

First, a bit of DB groundwork that has nothing to do with OAuth itself. I needed a place to store the access tokens and secrets and relate them to the existing user ids. My Slowgeek user table has a numeric u\_id field for each user, so that is what I am using as my primary key here:

```
CREATE TABLE twitter (  
  u_id int(10) not null,  
  name char(32) default NULL,  
  state smallint default 0,  
  token varchar(64) default NULL,  
  secret varchar(64) default NULL,  
  description varchar(255) default NULL,  
  status varchar(140) default NULL,  
  location varchar(80) default NULL,  
  followers smallint default 0,  
  mtime timestamp default CURRENT_TIMESTAMP on update CURRENT_TIMESTAMP,  
  PRIMARY KEY (u_id)  
) TYPE=MyISAM;
```

And the related DB code using PDO.

So, I now have a mechanism for storing Twitter-related data. Now for the real oauth work. First I registered my application with Twitter to get a consumer key and consumer secret. You do that at [http://twitter.com/oauth\\_clients/new](http://twitter.com/oauth_clients/new).

Because I am updating the status I needed read/write access for this app. Now the PHP code. It will probably be easier to read if you copy and paste it out of the iframe into your favourite editor.

The last thing this script did after it had gotten the access token and secret was to call `verify_credentials.json` to get the user's credentials.

That gives us json back which we can decode with `json_decode()`. There is some interesting stuff in your Twitter record. Here is mine from a few minutes ago:

## Blog Export: Rasmus' Toys Page, <http://toys.lerdorf.com/>

stdClass Object

```
(
  [favourites_count] => 0
  [profile_text_color] => 666666
  [description] => Breaking the Web
  [screen_name] => rasmus
  [utc_offset] => -28800
  [profile_background_image_url] => http://static.twitter.com/images/themes/theme9/bg.gif
  [profile_link_color] => 2FC2EF
  [following] =>
  [profile_sidebar_fill_color] => 252429
  [url] =>
  [name] => Rasmus Lerdorf
  [time_zone] => Pacific Time (US & Canada)
  [protected] =>
  [status] => stdClass Object
  (
    [truncated] =>
    [in_reply_to_status_id] => 1642930101
    [text] => @DonMacAskill Floating point values are approximations in all computer languages #php #broken
    [in_reply_to_user_id] => 813491
    [favorited] =>
    [in_reply_to_screen_name] => DonMacAskill
    [id] => 1643107564
    [source] => Nambu
    [created_at] => Tue Apr 28 21:57:56 +0000 2009
  )

  [profile_sidebar_border_color] => 181A1E
  [notifications] =>
  [profile_background_tile] =>
  [followers_count] => 2112
  [friends_count] => 71
  [profile_background_color] => 1A1B1F
  [profile_image_url] => http://s3.amazonaws.com/twitter_production/profile_images/52489510/rl_normal.jpg
  [location] => Sunnyvale, California
  [id] => 928961
  [statuses_count] => 577
  [created_at] => Sun Mar 11 15:39:19 +0000 2007
)
```

The `$debug = $oauth->getLastResponseInfo();` after an oauth call will always give you info about the last call. For this verify\_credentials call the LastResponseInfo is (with the actual token and secret deleted):

```
[u_id] => 1823955881
[name] => rasmus
[state] => 2
[token] => XXX
[secret] => XXX
[description] => Breaking the Web
[status] => @cdamian I don't like things strapped to my chest. My brain is a perfectly good built-in heart rate monitor.
[location] => Sunnyvale, California
[followers] => 2080
[utype] =>
[mtime] => 2009-04-26 09:07:08
```

And now you can actually send a tweet. This assumes the filename is 'twitter.php'. This shows how to POST a tweet to

## Blog Export: Rasmus' Toys Page, <http://toys.lerdorf.com/>

twitter. Two things to note here. Because Twitter requires a status update to be sent as a POST request, we have to use OAUTH\_AUTH\_TYPE\_FORM when we instantiate the oauth object here. And second, I have a CSRF-preventing crumb in the POST data. The idea here is to tie the POST body to the current user's login cookie so the bad guys can't spoof a form post to our twitter.php script.

I am hoping the code is for the most part self-explaining. You can also have a look at the great pecl/oauth examples.

Posted by Rasmus in PHP at 15:20