monkehTweets v1.2.6

Thank you for downloading monkehTweets, a ColdFusion Package developed to interact with the Twitter APIs.

License and Credits

For full License details, please read the LICENSE file available with this download.

Authors

Developed by Matt Gifford AKA coldfumonkeh
http://www.mattgifford.co.uk

Got a lot out of this package? Saved you time and money?
Share the love and visit Matt’s wishlist:
http://www.amazon.co.uk/wishlist/B9PFNDZNH4PY

Requirements

monkehTweets requires ColdFusion 8+

Installation

Unzip the package archive to the desired location (typically in the web root). After installation, you will see the following directories:

- Installation (contains this installation guide, no more, no less)
- com (contains all of the ColdFusion Components required for the application to interact with the Twitter API)

The application does not interact with any database, and as such needs no datasources.

Getting Started

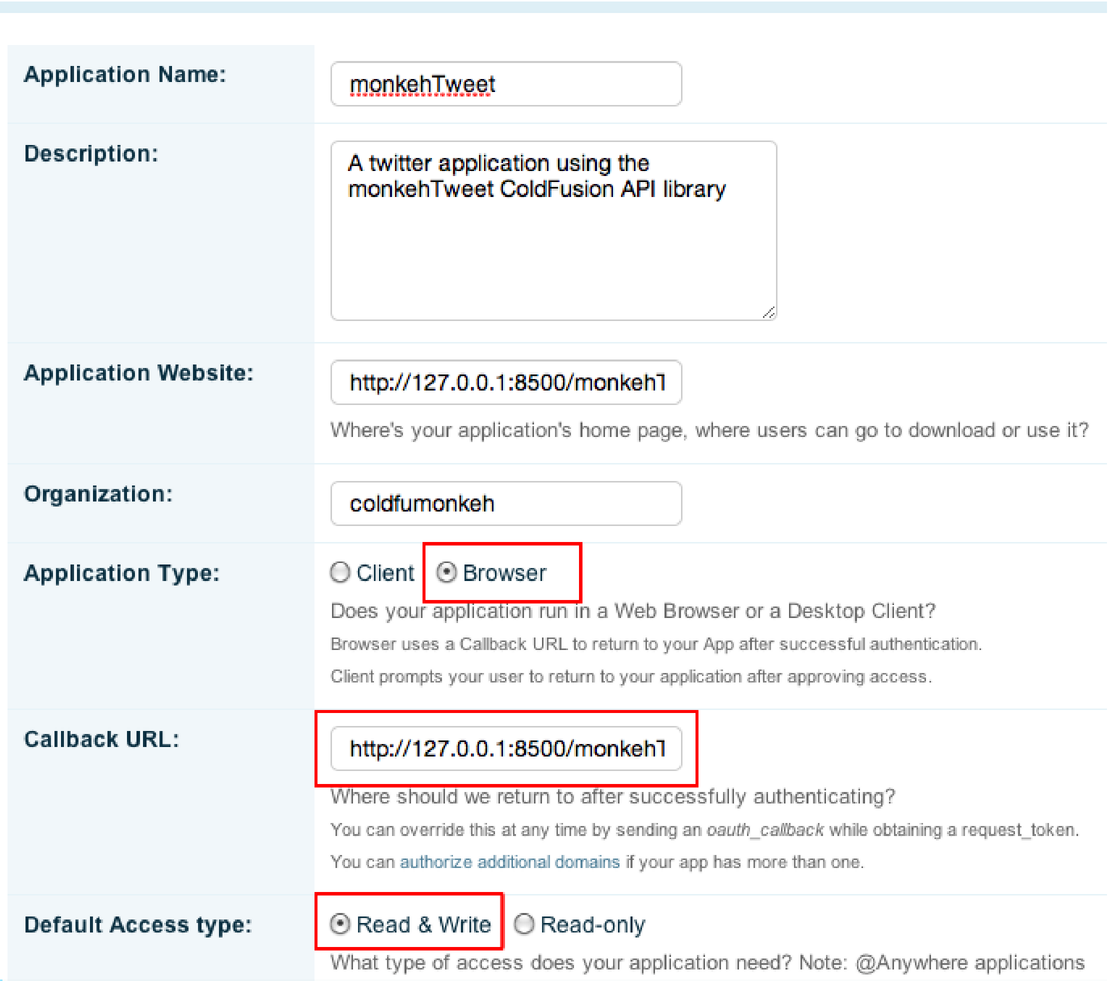
Version 1.2.4+ of monkehTweets was released to include the OAuth authentication protocol, which now completely replaces the basic authentication method previously offered by Twitter.

To interact with the Twiter API, you will need to create an application with Twitter, which generates the consumer key and secret values that are required to instantiate and work with monkehTweets (and indeed the API itself).

Visit http://dev.twitter.com/apps to register a new application.

Items to note in here:

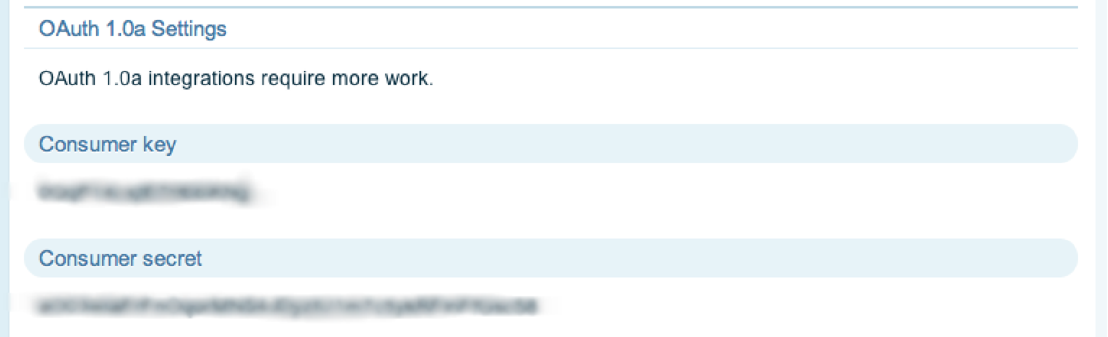
- Application type – set to browser
- Callback URL – you can set your initial callback URL here (e.g. http://localhost:8500/authorize.cfm) but you can overwrite this when making the actual authorization call from monkehTweets
- Access type – set to Read & Write to allow posts



Application Name:	monkehTweet
Description:	A twitter application using the monkehTweet ColdFusion API library
Application Website:	http://127.0.0.1:8500/monkehT Where's your application's home page, where users can go to download or use it?
Organization:	coldfumonkeh
Application Type:	○ Client ● Browser Does your application run in a Web Browser or a Desktop Client? Browser uses a Callback URL to return to your App after successful authentication. Client prompts your user to return to your application after approving access.
Callback URL:	http://127.0.0.1:8500/monkehT Where should we return to after successfully authenticating? You can override this at any time by sending an *oauth_callback* while obtaining a request_token. You can authorize additional domains if your app has more than one.
Default Access type:	● Read & Write ○ Read-only What type of access does your application need? Note: @Anywhere applications

Figure 1 – The initial application page

Having created your application, you need to make a note of the consumer details (the consumer key and consumer secret). These are required when instantiating the monkehTweets component.



OAuth 1.0a Settings

OAuth 1.0a integrations require more work.

Consumer key

[redacted]

Consumer secret

[redacted]

Figure 2 – The required consumer details from the application page

If you are using the application for numerous users (a number of accounts using your application to interact with Twitter) then you're job here is done.

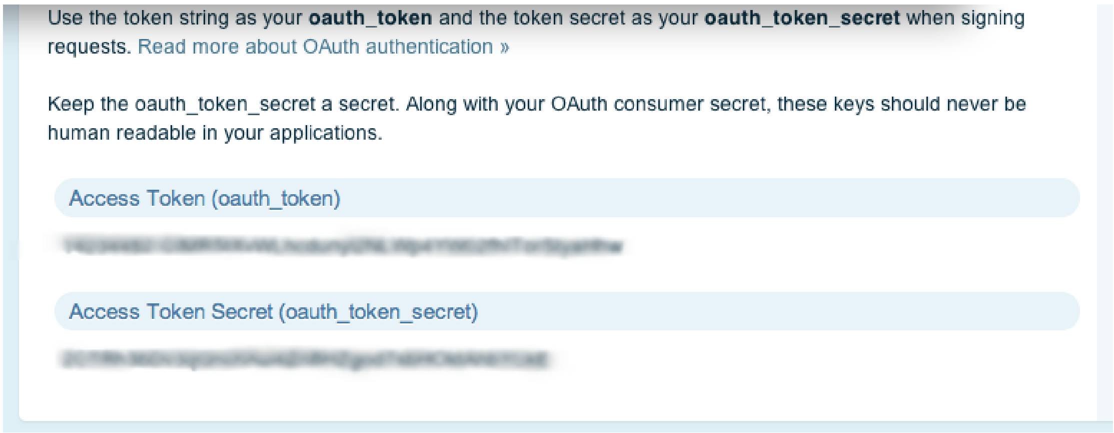
Simply instantiate the component with the consumer details and set up your code to request authentication and access for each user, as per the code samples within the monkehTweet download.

Instantiation is incredibly easy, as seen in the below example:

```
<cffunction name="OnApplicationStart"
             access="public"
             returntype="boolean"
             output="false">
       <cfscript>
       application.objMonkehTweet =
              createObject('component',
              'com.coldfumonkeh.monkehTweet')
              .init(
              consumerKey          = '< enter your consumer key >',
              consumerSecret       = '< enter your consumer secret >',
              parseResults         = true
              );
              return true;
       </cfscript>
</cffunction>
```

If you are intending to be the sole user (only your Twitter account will access the API) then you can bypass the requirement for authentication.

Within the application details page for your new application on http://dev.twitter.com, select the 'My Access Token' option from the right-hand menu.



Use the token string as your **oauth_token** and the token secret as your **oauth_token_secret** when signing requests. Read more about OAuth authentication »

Keep the oauth_token_secret a secret. Along with your OAuth consumer secret, these keys should never be human readable in your applications.

Access Token (oauth_token)

[redacted]

Access Token Secret (oauth_token_secret)

[redacted]

Figure 3 – Obtaining the access details from your application page

Copy the access token and access token secret (the oauth tokens) from this screen, and use these along with your Twitter account screen name in the init() method when instantiating the monkehTweets component, as seen below:

```
<cffunction name="OnApplicationStart"
              access="public"
              returntype="boolean"
              output="false">
       <cfscript>
       application.objMonkehTweet =
              createObject('component',
              'com.coldfumonkeh.monkehTweet')
              .init(
              consumerKey        = '< enter your consumer key >',
              consumerSecret     = '< enter your consumer secret >',
              oauthToken         = '< enter the access token >',
              oauthTokenSecret   = '< enter the access secret >',
              userAccountName    = '< enter your twitter screen name >',
              parseResults       = true
              );
              return true;
       </cfscript>
</cffunction>
```

This essentially bypasses the need for OAuth requests and redirects as we now have the access details for your account. You are ready to use monkehTweets instantly.

Lucky you. ☺

Limitations

Remaining functionality

There are a few methods currently missing from this release, namely:

addMultipleMembersToList
geoAddPlace
getListMembersByID
getListSubscriberByID

These will be revised and included in the next major release.

Testing

monkehTweets has been tested on the Adobe ColdFusion platform.
It has yet to be tested fully on Railo or OpenBlueDragon.