OAuth

A Simple Open Standard for Secure API Authentication.

Preamble

- ✓ Web 2.0 means sharing data, through API
- ✓ Users want to access their data using many services
- ✓ Developers want to satisfy their users (and make it easy for them)
- ✓ Service providers need to keep their users data secure

Problematic

Passwords are precious

How to delegate access?

Summary

1 - OAuth Presentation

2 - TD : Implementing OAuth on Web Site

OAuth Presentation

What is OAuth ?

It's a protocol that allows to share private data hosted on X website (or API) with Y website (Or API).

History :

- OAuth project started in November 2006, when Blaine Cook was working on the Twitter OpenID implementation.
- In 2007 OAuth V1
- In 2012 OAuth V2

Application Programming Interface

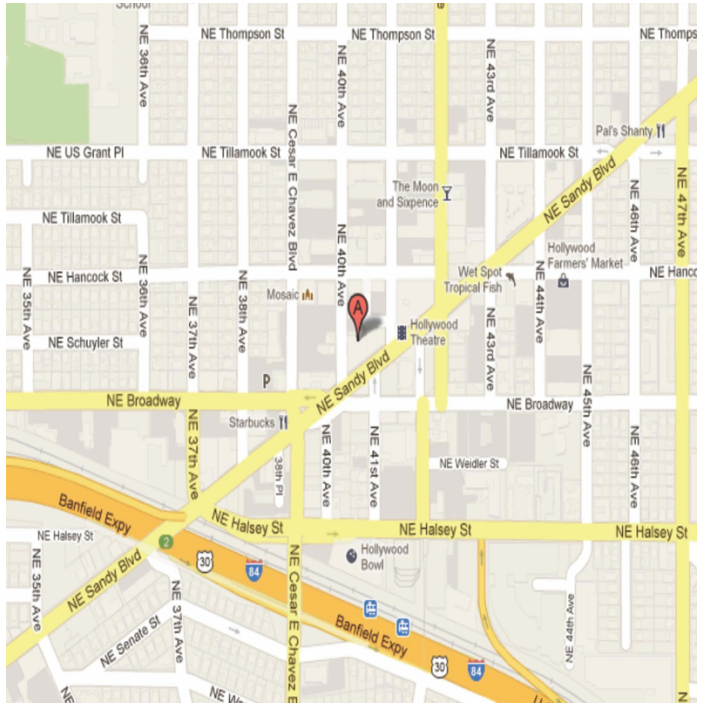
What is an API ?

An API is a interface by which software provides services to other programs.

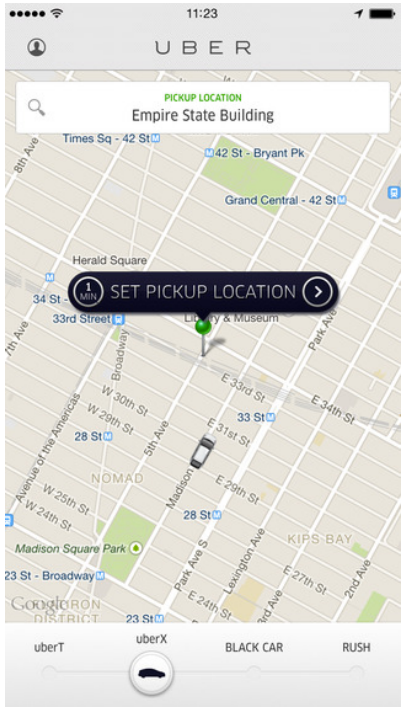
Example of API :

- Graph API de Facebook
- Twitter API
- Google Maps API

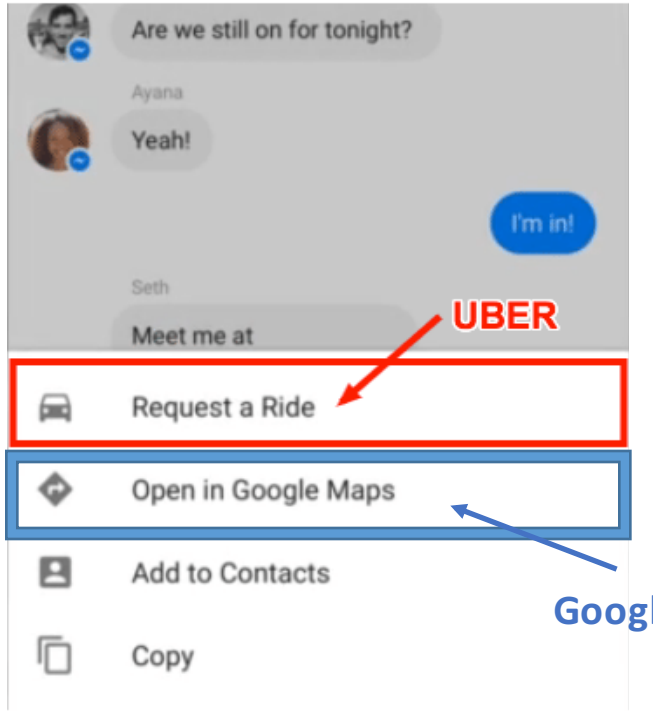
Example of API interaction



Google Maps API



Uber API use
Google Maps API

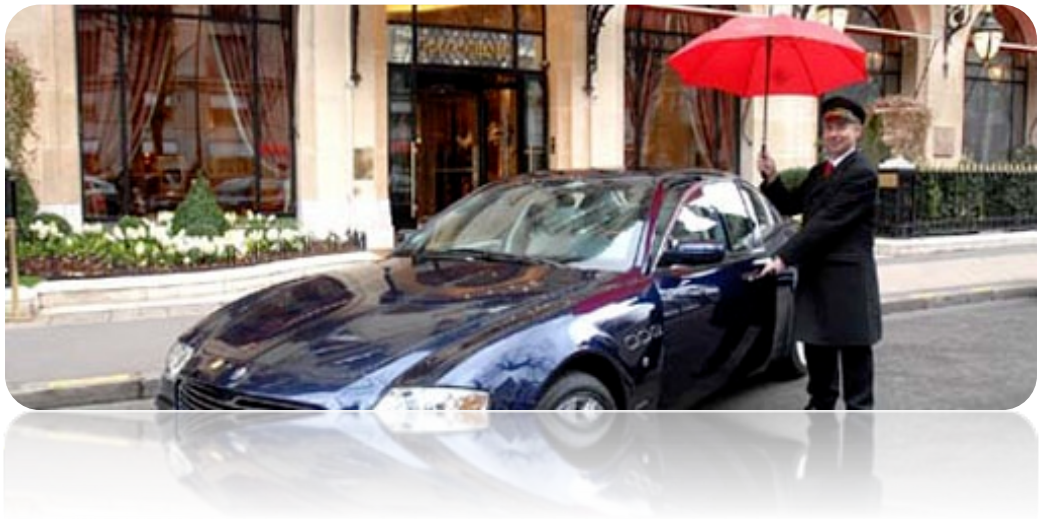


Facebook Messenger
use Uber API and Google Maps

Concept : The Valet Key Car

OAuth is the Service key of API.

« Une clé de service donne la possibilité de garer votre voiture, mais pas la possibilité d'en ouvrir le coffre, de conduire plus de 2km ou d'amener dans le rouge le régime moteur de votre toute nouvelle voiture de sport allemande. De la même façon, un jeton Oauth laisse à un tiers la possibilité de lire votre mail, mais pas la possibilité d'envoyer à votre nom des mails aux personnes présentes dans votre carnet d'adresses. »



John Panzer, Google

Concept : The Valet Key Car

Main Key :

- Full Access
- Can remove or manage access to Service Key

Service Key :

- Limited Access
- Can’t change access or settings

OAuth : Terminologies

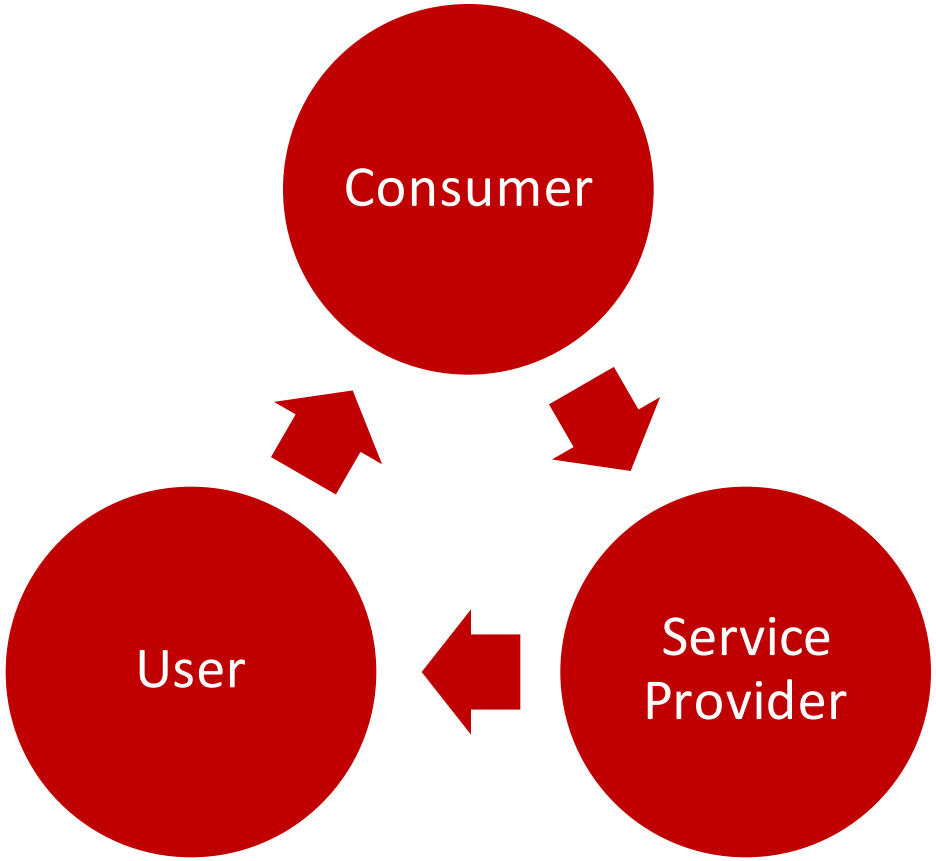
- ✓ **Consumer :** Application trying to access protected data
- ✓ **Service Provider :** Website or Web-service hosting protected resource
- ✓ **User :** Owner of the protected data
- ✓ **Protected Resource :** Images, Videos or documents hosted on web site or web-service which are protected by the user
- ✓ **Tokens :** Random string of letters and numbers which is unique. Request token, Access Token
- ✓ **Scope :** Set of data hosted on service provider that users wants to share with consumer

OAuth : Third-Party System

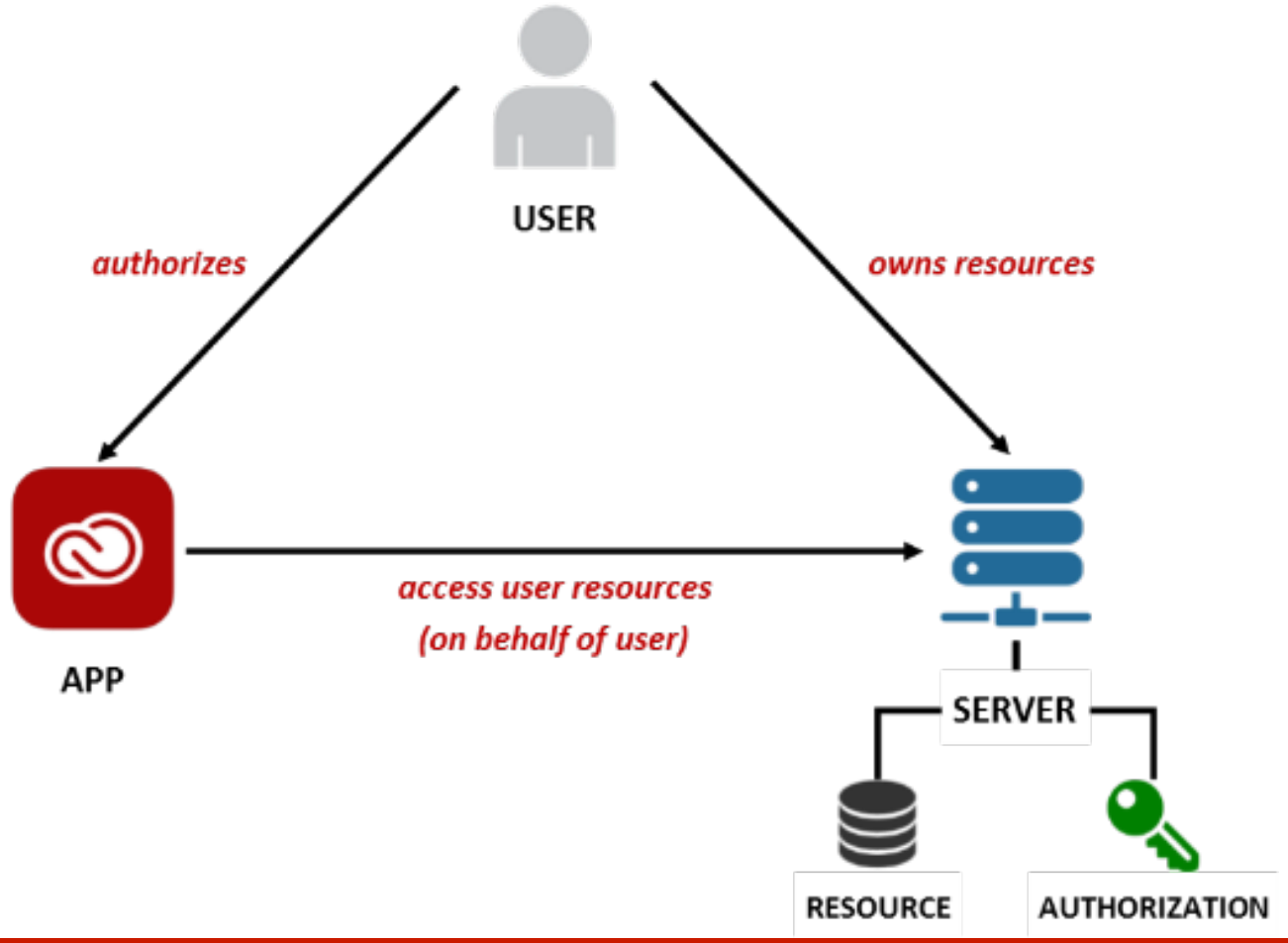
Consumer => Application trying to access protected data (e.g. Allociné)

Service Provider => hosting protected resource (e.g. : Facebook)

User => Accept or Not to share data



OAuth : Global Process



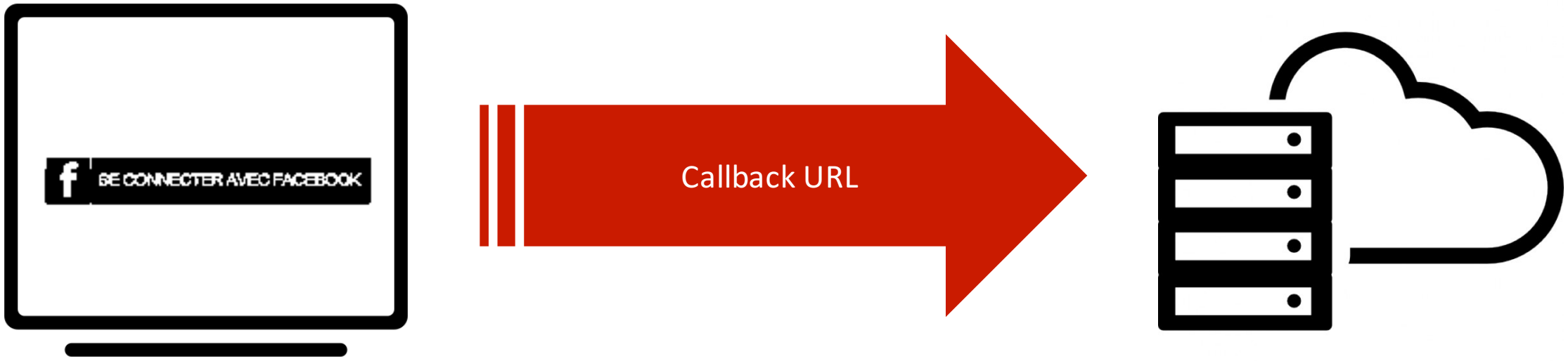
Focusing OAuth Procedure : Step 1

User

Users start the process by informing the customer that wants to connect to the server

Consumer

Focusing OAuth Procedure : Step 2

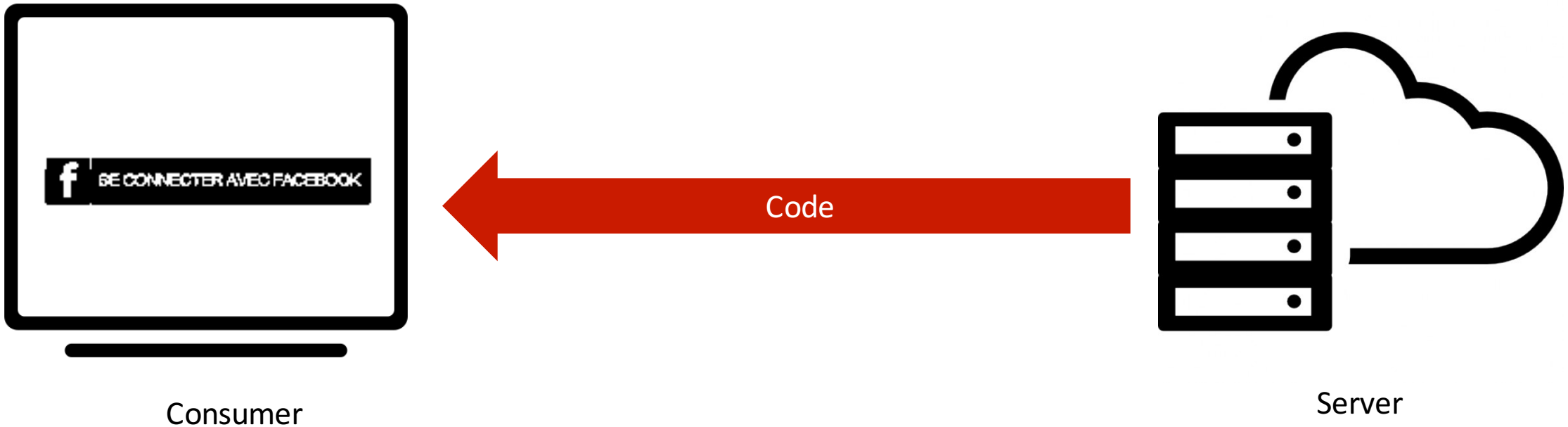


Focusing OAuth Procedure : Step 3

user

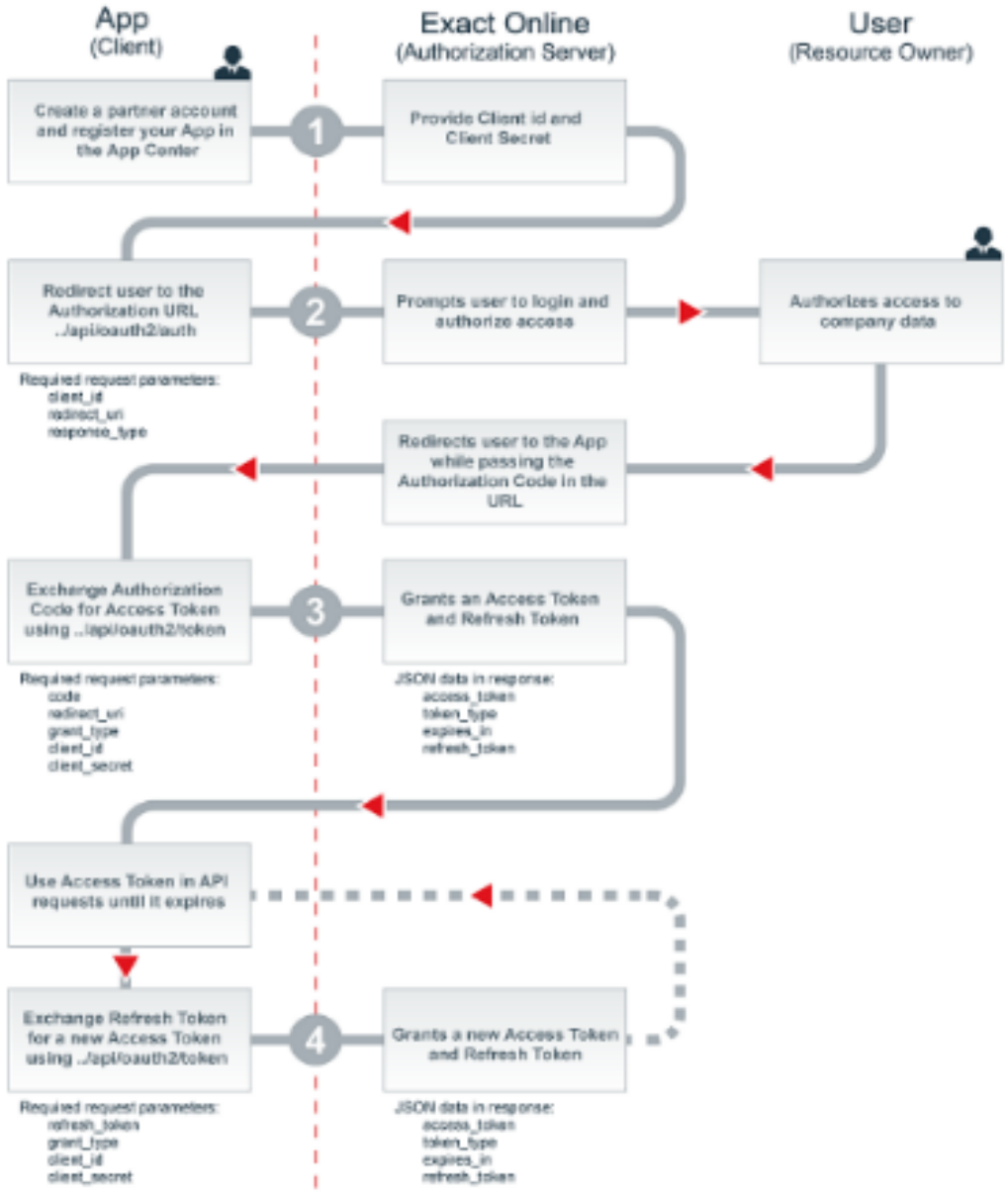
Server

Focusing OAuth Procedure : Step 4



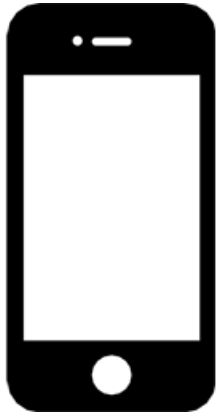
OAuth Flow diagram

1. User wants to use app to do e.g. time registration
2. App wants to access protected resources of user like e.g. date of birth
3. App redirects user to authorization server
4. Authorization server ask for authentication and access authorization
5. Authorization server redirects user to app with authorization code
6. App exchanges authorization code for access token / refresh token
7. App requests protected resource data from resource server
8. Resource server responds with protected resource data
9. App exchanges refresh token for new access token



OAuth Integration

Cloud

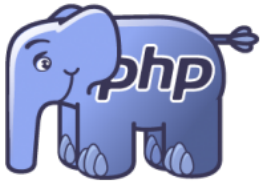


Mobile

Desktop

Server

OAuth Library



And some more …

Example of Application who use OAuth :

- Facebook
- Foursquare
- Github
- Google
- Microsoft (Hotmail, Messenger, Xbox)
- LinkedIn
- MySpace
- Netflix
- StatusNet
- Twitter
- Vimeo
- Yahoo!
- Paypal
- Dailymotion
- Numericable

Advantage of OAuth

- ✓ Can be integrated into a website, mobile and other devices.
- ✓ No password or user identification information with other applications.
- ✓ Simple to set up.
- ✓ Users can revoked tokens at any time.
- ✓ Nefarious customers can have their credentials revoked and all associated access tokens immediately destroyed.
- ✓ Standardization of access to the API

Security on OAuth

- Tokens aren’t passing username/password
- Timestamp and nonce verify unique requests
- Signature encrypted parameters help service provider recognize consumer
- Signature methods HMAC-SHA1, RSA-SHAI, Plaintext over a secure channel (such as SSL)

Demo

http://opauth.org/#demo

TD

Goal : Create a Website who use social authentication

TD

Prerequisites :

1. Create Twitter account
2. Create and Configure Apps on Twitter DEV
3. Have an Access on UP2
4. Create data table on PhpMyAdmin
5. Implementing OAuth

TD : Create Twitter account

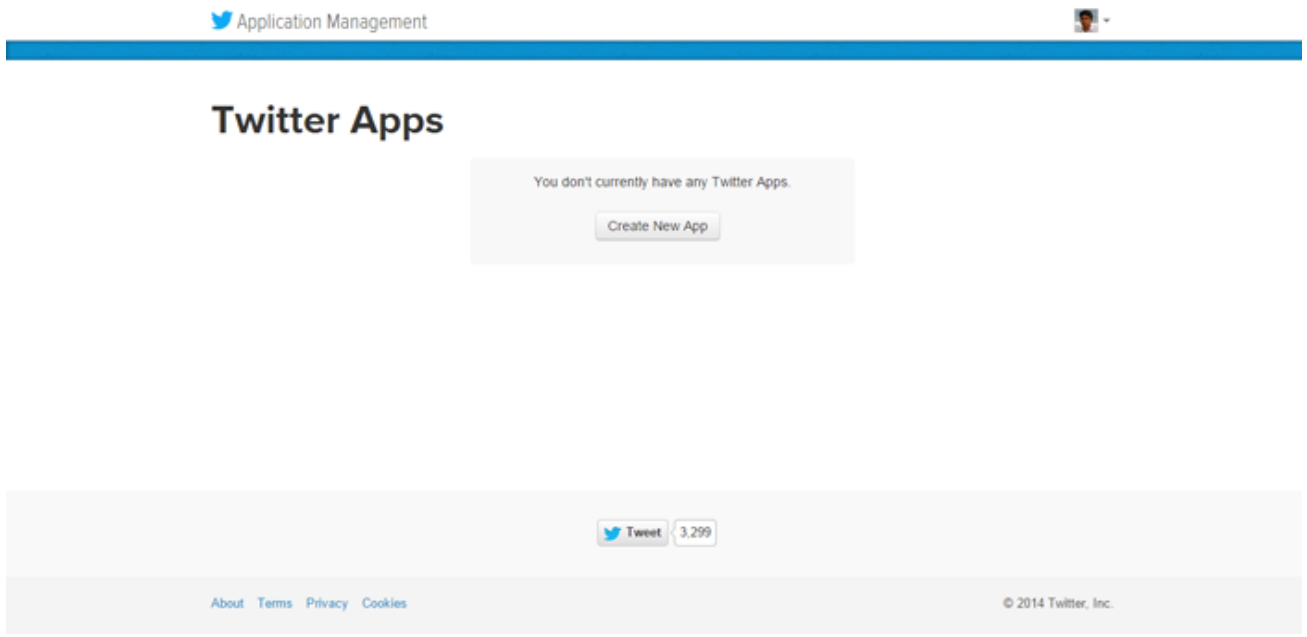
https://twitter.com/

TD : Create Apps on Twitter DEV

We need to create Twitter Application using twitter app dashboard. To create Twitter App please going to following URL .

https://apps.twitter.com/

Once you logged in, click on **Create New App** button to continue.

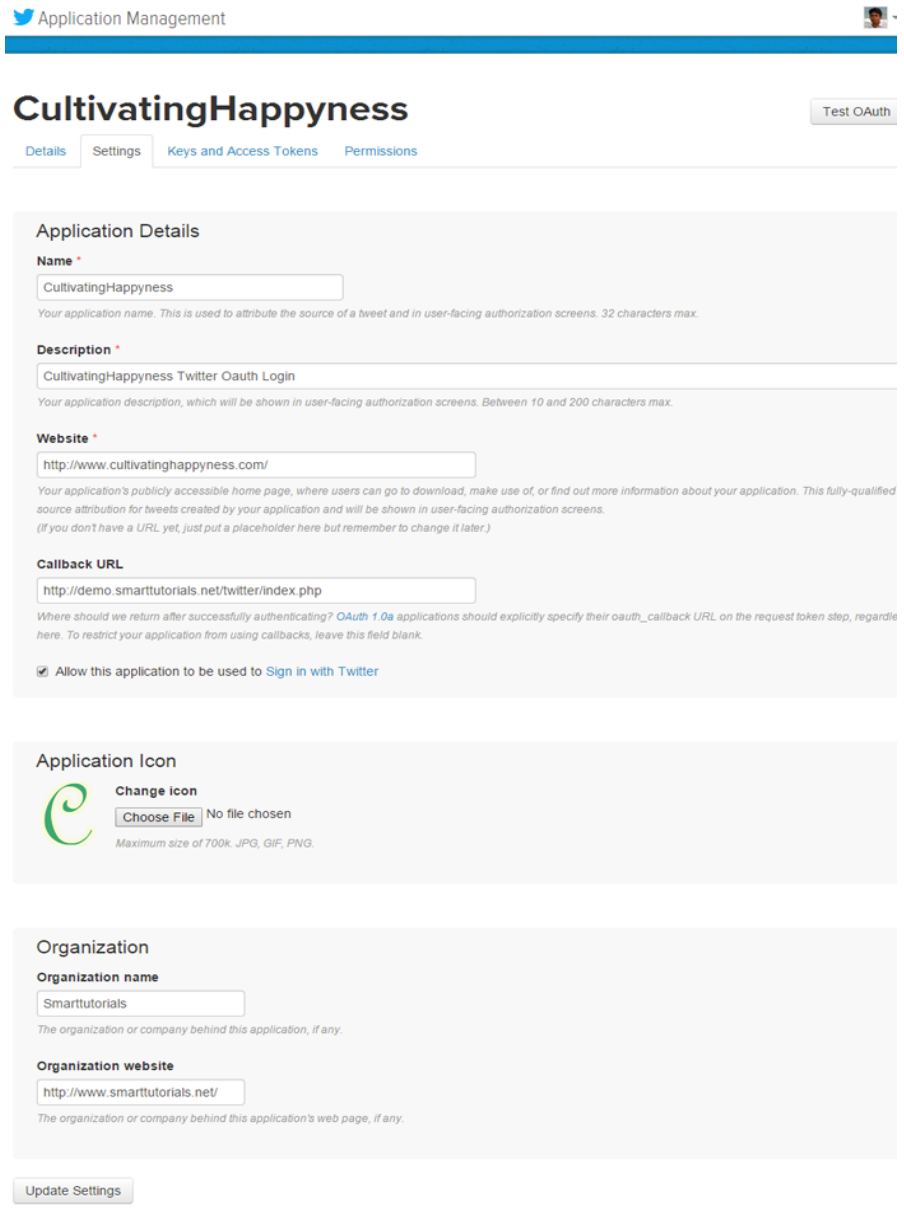


TD : Configure Apps on Twitter DEV

Fill the following filelds to create Twitter Applications like in the images below.

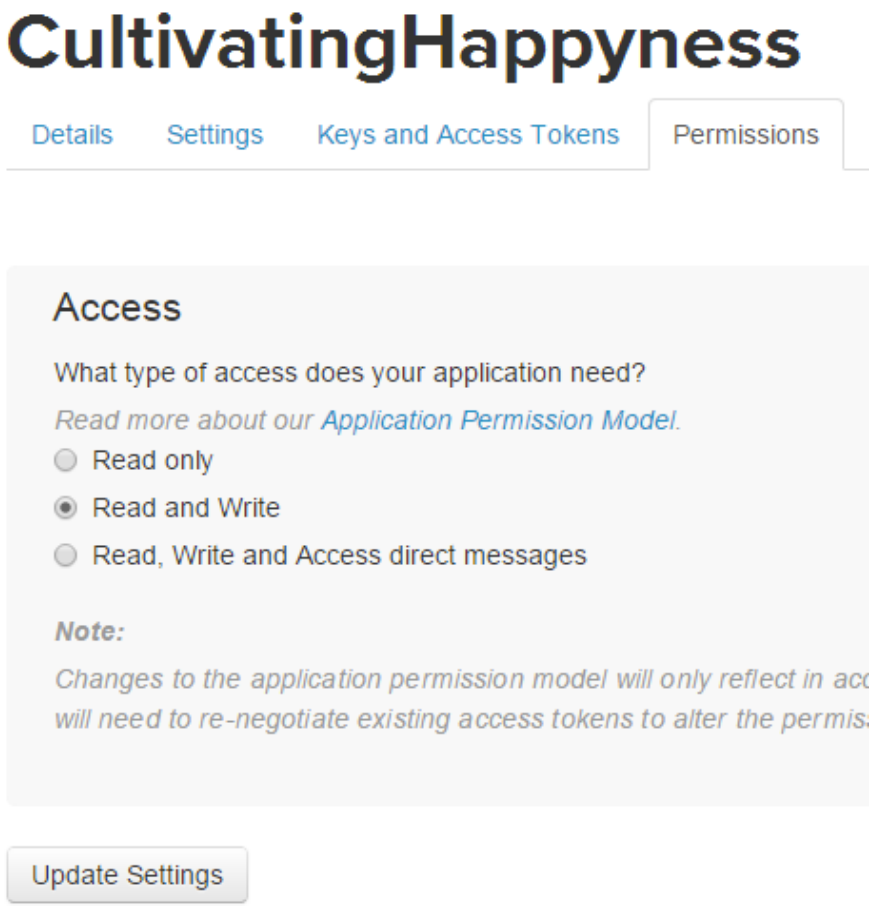
Important :

- ➢ Callback URL = http://up2.fr/M1/etudiants/(Your_ID)/index.php
- ✓ Check Allow this application to be used « Sign in with Twitter »



TD : Configure Apps on Twitter DEV

Please select Read and Write Permission in the Permission tab and click on update settings button to update.



Create table using following SQL queries

1. Go to http://www.up2.fr/M1/outils/phpMyAdmin/
2. Log in
3. Click to your Table ID
4. Click SQL Tab
5. Write this SQL code

```sql
CREATE TABLE IF NOT EXISTS `users` (
  `user_id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `name` varchar(50) NOT NULL,
  `email` varchar(60) NOT NULL,
  `password` varchar(60) NOT NULL,
  `social_id` varchar(100) NOT NULL,
  `picture` varchar(250) NOT NULL,
  `created` datetime NOT NULL,
  PRIMARY KEY (`user_id`),
  KEY `email` (`email`),
  KEY `login` (`password`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=1;
```

Modify config.php file

Go to http://www.up2.fr/M1/outils/edit/
Modify Config.php with your ID

```
//site specific configuration declartion
define( 'BASE_PATH', 'http://www.up2.fr/M1/etudiants/ID_Student/');
define( 'DB_HOST', 'localhost' );
define( 'DB_USERNAME', 'ID_Student');
define( 'DB_PASSWORD', 'Password_Student');
define( 'DB_NAME', 'ID_Student');


//Twitter login
define('TWITTER_CONSUMER_KEY', '"YOUR_TWITTER_CONSUMER_KEY"');
define('TWITTER_CONSUMER_SECRET', 'YOUR_TWITTER_CONSUMER_SECRET');
define('TWITTER_OAUTH_CALLBACK', 'http://www.up2.fr/M1/etudiants/ID_Student/index.php');
```

Sample Index.php

```php
/*** Twitter****/
require_once('twitteroauth/twitteroauth.php');

if (TWITTER_CONSUMER_KEY === '' || TWITTER_CONSUMER_SECRET === '' || TWITTER_CONSUMER_KEY === 'TWITTER_CONSUMER_KEY_HERE' || TWITTER_CONSUMER_SECRET === 'CONSUMER_SECRET_HERE')
    echo 'You need a consumer key and secret to test the sample code. Get one from <a href="https://dev.twitter.com/apps">dev.twitter.com/apps</a>';
    exit;
}

if(!isset( $_SESSION['oauth_token'] )){
    $connection = new TwitterOAuth(TWITTER_CONSUMER_KEY, TWITTER_CONSUMER_SECRET);
    $request_token = $connection->getRequestToken(TWITTER_OAUTH_CALLBACK);
    $_SESSION['oauth_token'] = $token = $request_token['oauth_token'];
    $_SESSION['oauth_token_secret'] = $request_token['oauth_token_secret'];
    switch ($connection->http_code) {
        case 200:
            $url = $connection->getAuthorizeURL($token);
            break;
        default:
            $error = 'Could not connect to Twitter. Refresh the page or try again later.';
    }
}else{
    $connection = new TwitterOAuth(TWITTER_CONSUMER_KEY, TWITTER_CONSUMER_SECRET, $_SESSION['oauth_token'], $_SESSION['oauth_token_secret']);
    $access_token = $connection->getAccessToken($_REQUEST['oauth_verifier']);
    $_SESSION['access_token'] = $access_token;
    $content = $connection->get('account/verify_credentials');
    $data = array();
    if( !empty( $content->id )){
        $data['id'] = $content->id;
        $data['name'] = $content->name;
        $data['screen_name'] = $content->screen_name;
        $data['picture'] = $content->profile_image_url;
        try {
            $user_obj->twitter_login($data);
            header('Location: home.php');exit;
        }catch (Exception $e) {
            $error = $e->getMessage();
        }

    }else{
        session_unset();
        session_destroy();
        header('Location: index.php');
    }
}
/*** Twitter****/
```

Sample of OAuth.php

```php
/* Generic exception class
 */

class OAuthConsumer {
  public $key;
  public $secret;

  function __construct($key, $secret, $callback_url=NULL) {
    $this->key = $key;
    $this->secret = $secret;
    $this->callback_url = $callback_url;
  }

  function __toString() {
    return "OAuthConsumer[key=$this->key,secret=$this->secret]";
  }
}

class OAuthToken {
  // access tokens and request tokens
  public $key;
  public $secret;

  /**
   * key = the token
   * secret = the token secret
   */
  function __construct($key, $secret) {
    $this->key = $key;
    $this->secret = $secret;
  }

  /**
   * generates the basic string serialization of a token that a server
   * would respond to request_token and access_token calls with
   */
  function to_string() {
    return "oauth_token=" .
           OAuthUtil::urlencode_rfc3986($this->key) .
           "&oauth_token_secret=" .
           OAuthUtil::urlencode_rfc3986($this->secret);
  }

  function __toString() {
    return $this->to_string();
  }
}
```

Sample TwitterOAuth.php

```php
/**
 * Twitter OAuth class
 */
class TwitterOAuth {
  /* Contains the last HTTP status code returned. */
  public $http_code;
  /* Contains the last API call. */
  public $url;
  /* Set up the API root URL. */
  public $host = "https://api.twitter.com/1.1/";
  /* Set timeout default. */
  public $timeout = 30;
  /* Set connect timeout. */
  public $connecttimeout = 30;
  /* Verify SSL Cert. */
  public $ssl_verifypeer = FALSE;
  /* Respons format. */
  public $format = 'json';
  /* Decode returned json data. */
  public $decode_json = TRUE;
  /* Contains the last HTTP headers returned. */
  public $http_info;
  /* Set the useragnet. */
  public $useragent = 'TwitterOAuth v0.2.0-beta2';
  /* Immediately retry the API call if the response was not successful. */
  //public $retry = TRUE;



  /**
   * Set API URLS
   */
  function accessTokenURL()  { return 'https://api.twitter.com/oauth/access_token'; }
  function authenticateURL() { return 'https://api.twitter.com/oauth/authenticate'; }
  function authorizeURL()    { return 'https://api.twitter.com/oauth/authorize'; }
  function requestTokenURL() { return 'https://api.twitter.com/oauth/request_token'; }
```

Other Application with OAuth and Twitter

With OAuth and Twitter for example, you can stream Data issue to Twitter for Big Data analysis

```r
# @Authors : Guillaume VIMONT
# Collecting Twitter data with Stream API on R and analyse Hashtag

#Step 1 : Prerequite
library(ROAuth)
library(streamR)
require(twitteR)
require(wordcloud)
require(igraph)

# Step 2 : Declare Twitter API Credentials
api_key <- "XXXXXXXXXXXXXXXX" # From dev.twitter.com
api_secret <- "XXXXXXXXXXXXXXXXXXXXXXXXXXXXX" # From dev.twitter.com
token <- "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX" # From dev.twitter.com
token_secret <- "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX" # From dev.twitter.com

# Step 3 : Create Twitter Connection
setup_twitter_oauth(api_key, api_secret, token, token_secret)

#function to actually scrape Twitter
filterStream( file.name="tweets_test.json",
              track="", tweets=100, oauth=cred, timeout=10, lang='fr' )
#Parses the tweets
tweet_df <- parseTweets(tweets='tweets_test.json')
#using the Twitter data frame
tweet_df$created_at
tweet_df$text


plot(tweet_df$friends_count, tweet_df$followers_count) #plots scatterplot
cor(tweet_df$friends_count, tweet_df$followers_count) #returns the correlation coefficient

# Create an empty character vector of the same length as the number of
# tweets
l <- length(tweets)
tweeter <- vector(mode = "character", length = l)

# Extract the screen names from each tweet status
for (i in 1:l) tweeter[i] <- tweets[[i]]$getScreenName()


# Compile the frequencies of each screen name
tweeter.freq <- table(tweeter)
# Black backgrounds are 'this year's colour' in terms of dataviz!
par(bg = "black")

# Draw the word cloud
wordcloud(names(tweeter.freq), tweeter.freq, colors = c("tomato",
                                                        "wheat", "lightblue"), scale = c(6, 0.5), random.color = TRUE, rot.per = 0.5,
          min.freq = 1, font = 2, family = "serif")

# Create an empty vector to store the tweet texts
texts <- vector(mode = "character", length = l)

# Extract the tweet text from each tweet status
for (i in 1:l) texts[i] <- tweets[[i]]$getText()
```

Questions ?

Links

http://oauth.net

http://oauth.net/core/1.0/

http://code.google.com/p/oauth/

http://groups.google.com/group/oauth/

http://oauth.net/discovery/1.0

Thank you

Contact :

vimontguillaume@gmail.com