ZingMe SDK : OAuth2-0 & GraphAPI

Version 1.0

- draft version 1.01-

- Sep 2011-

Table of Contents

1. How to use ZingMe SDK:

1.1. Getting started:

Before be able to integrate with ZingMe PHP SDK, 3rd-party needs to prepare some information below:

a. Application Name: must be unique; only contains **a-zA-Z0-9, '_', '-'**. No other special character is allowed.

b. Application Title: this title will appear on the title of browser.

c. 3rd-party development server **callbackURL**: the domain of this callbackURL must be the same as the domain of redirect_uri when 3rd-party requests for authorization code using OAuth SDK.

d. Development languague of 3rd-party backend: the language that 3rd-party uses for developing app/game so that ZingMe will be able to provide corresponding client library.

e. Description and Instruction of 3rd-party app/game.

After ZingMe has recieved enough information, we will create neccesary configuration for you on ZingMe and revert to you: <**appname**>, <**client_id**> & <**client_secret**>.

1.2. Intergration with SDK:

1.2.1. Installing and Initializing:

- To install ZingMe PHP SDK, copy the folder <zingme-sdk> (in SDK library) on the server where you will host your app.
- Include below files wherever you want you use the SDK.

```
include_once 'zingme-sdk/BaseZingMe.php';
include_once 'zingme-sdk/ZME_Me.php';
include_once 'zingme-sdk/ZME_User.php';
```

BaseZingMe.php: the "**must have**" file to be include, use for retrieving authorization code (authentication) and access token (calling graphAPI).

ZME_Me.php: use for retrieving information of logged user.

ZME_User.php: user for retrieving information of friends of logged user.

- Initialize configuration of your app/game:

```
$config= array (
    'appname' => 'test',
    'apikey' => 'db177303aa0b1498e512f112803f846d',
    'secretkey' => '8d3738883de7a75b382bb1253f59c943'
```

```
);
```

- Instantiate a ZME_Me object with configuration above:

```
$zm_Me = new ZME_Me($config);
```

1.2.2. Integrate with Authentication SDK:

1.2.2.1. For app/game running inside ZingMe:

- When ZingMe user accesses to your app/game, he/she will need to log in in order to use your app/game.
- When your app/game is loaded, ZingMe will provide you a <**signed_request**> that grants the authorization for your app/game. The <**signed_request**> can be retrieved by calling:

```
$signed_request = $_REQUEST['signed_request']
```

- After you've got the <**signed_request**>, next step is to retrieve the <**access_token**> in order to access ZingMe GraphAPI for later usage.

```
$access_token = $zm_Me->getAccessTokenFromSignedRequest($signed_request);
```

Note:

- The <**access_token**> is only valid in duration of time. If you face **ZingMeApiException** with code **-13** and message "**Access token had been expired**", you have to reload your app/game so that ZingMe will provide you a new valid <**signed_request**>. Then you can get new <**access_token**> by using ZME_Me::getAccessTokenFromSignedRequest API (section 2.1.c).

1.2.2.2. For app/game running outside ZingMe

- After instantiated ZME_Me object, you need to call the API ZME_Me:: getUrlAuthorized (section 2.1.a).

```
$zm_Me->getUrlAuthorized($redirect_uri,$state);
```

  - If ZingMe user has already logged in, ZingMe will redirect him/her to your provided <**redirect_uri**> and also pass you an authorization code for further action.
  - If ZingMe user hasn't logged in yet, he/she will be redirected to ZingMe login page. After he/she logged in, ZingMe will redirect him/her to your provided <redirect_uri> and pass you an authorization code for further action.
- You can get the authorization code by calling:

```
$authorization_code = $_REQUEST['code'];
```

- After you've got the <code> (authorization code), next step is to retrieve the <access_token> in order to access ZingMe GraphAPI for later usage.

```
$access_token = $zm_Me->getAccessTokenFromCode($authorization_code);
```

Note:

- Please note that: the domain of <redirect_uri> must be the same as the domain of callbackurl which you provided us to register your app (ref to section 1.1.c). If not, you will give an exception
- The <**code**> (authorization code) will be expired if the user logged out ZingMe. You have to repeat the authentication from scratch in order to gain accessibility.
- The <**access_token**> will be expired after duration of time. You have to use the <**code**> (authorization code) to get new <**access_token**> by using ZME_Me::getAccessTokenFromCode (section 2.1.b).

1.2.3. Integrate with GraphAPI SDK:

(This section applies for both app/game running inside & outside ZingMe).

- After you've got the <**access_token**> from section 1.2.2, you can now using ZingMe GraphAPI to access to ZingMe logged-in-user's resource. Details for the APIs are listed in section 2.2 below.

Example:

```
$me = $zm_Me->getInfo($access_token);
$friends = $zm_Me->getFriends($access_token);
```

- For retrieving ZingMe logged-in-user's friends' information, you need to instantiate new object called **ZME_User**

```
$zm_User = new ZME_User($config);
```

* The $config here is the same $config in code snippet used in section 1.2.1 above

- Then you can use ZME_User:: getInfo API to retrieve ZingMe logged-in-user's friends' information (reference to section 2.2.2).

```
$user = $zm_User->getInfo($access_token, $uids,$fields="id,username,displayname");
```

1.2.4. Integrate with GraphAPI SDK – Feed:

- To generate signature key to use with Feed client library, you have to initialize instance of ZME_Feed:

Example:

```
$feed = new ZME_Feed($userIdFrom, $userIdTo, $actId, $tplId, $objectId,
        $attachName, $attachHref, $attachCaption, $attachDescription, $mediaType,
        $mediaImage, $mediaSource, $actionLinkText, $actionLinkHref);
$key = $feed->generateSignatureKey($secretKey);
```

2. OAuth2-0 API:

2.1. Authentication: Authentication on ZingMe for 3rd party is followed by OAuth-2.0 flow

1. *ZME_Me :: getUrlAuthorized*

Description: get the URL of authorization. (Use for app/game running outside ZingMe)

- If the user hasn't logged in ZingMe yet, user will be redirect to Zing Me login page. After user logged in, ZingME will redirect user to <**redirect_URI**>, which is pre-defined by 3rd-party, with <**code**> (authorization code) & <**state**>
- If the user logged in ZingMe already, ZingMe will directly redirect user to <**redirect_URI**>, which is pre-defined by 3rd-party, with <**code**>(authorization code) & <**state**>

Parameters:

Required	Name	Type	Description
required	redirect_uri	string	The url to go to after a successful login
optional	state	string	A CSRF state variable to assist in the defense against CSRF attacks, providing by the 3rd-party itself.

Return:

Example:

http://localhost/dev?
code=m4ATgWgpDMoWT86sDlDQ4w9KXCSstd4R_G2ca1BGT6YjFRVi1hXdHyjkikWHtm4jpW_2rW664N-HND36OlbLRFqXYiqt_0iq_dJQc62dNM7sUjoUCwOCFSeflCnEkquZoIUft0ItAIshA9MpBwaKVwDCzxP-fHGXyXltgBUFN35d7kMJ-k2JNW5PS-BuySLOGszsjRdUnm4M6cwneDdUU4Tb7_74sDrhNbacd8DG032aTJa%3D
&state=15233

Return	Description
code	authorization code
state	A CSRF state variable to assist in the defense against CSRF attacks

2. *ZME_Me :: getAccessTokenFromCode*

Description: retrieve an access token for the given authorization code (use for app/game running outside ZingMe)

Parameters:

Required	Name	Type	Description
required	code	string	an authorization code

Return:
Example:
{“**expires**”: 7200,
“**access_token**”:
“Dtra6X3U6oytwXvQUTm1AbAdL6aEic8OHCmoMKtL02uAfmLTGzWsEZ3DVH9ucb5ILlS056NAMrDmI7a2LK4cD42hqE0ugVwZ0hFTirtkowS0zFMu3x”}

Return	Description
expires	the expire time of the access token in seconds
access_token	the access token for app/game to access ZingMe GraphAPI

Throws: ZingMeApiException

3. *ZME_Me :: getAccessTokenFromSignedRequest*

Description: retrieve an access token from signed request (use for app/game running inside ZingMe

Parameters:

Required	Name	Type	Description
required	signed_request	string	a signed token given by Zingme to app/game running inside ZingMe

Return: a access token for app/game to access ZingMe GraphAPI
Example:
4bc575ea590a0b297e526815a38e9565.YzY3MGM2YzE=RFyluCP5CsDko5MWhqSeR0ZG6ClPV4jPJkyWhijE9sG2rn6bn458T3-qO-7xMoqN2uyVvxSB9nRbGMRnJ_5QSJO_d6t1XD6OT7UHRAQJuficPESofPVDYN1vuopNcUx-Oqhj8vQxu_FFkknQt00OIG==

Throws: ZingMeApiException (-10002): if the signed_request has been expired

4. *ZME_Me :: getUserLoggedIn*

Description: get <**uid**> of logged in user from a signed token/signed_request (use for app/game running inside ZingMe)

Parameters:

Required	Name	Type	Description

required	signed_request	string	a signed token given by Zingme to app/game running inside ZingMe

Return: an uid of logged in user
Example: 8481406

Throws: ZingMeApiException (-10004): if the signed_request has been expired

5. *ZME_Me :: getAccessToken*
Description: get the access token that should be used for GraphAPI calls
Return: an access token string

2.2. GraphAPI:

2.2.1. Me (logged in user):

a. *ZME_Me :: getInfo*
Description: retrieve profile info of user logged in with access_token. Default user information which will be returned includes <**id**> & <**username**>. However, 3rd-party can request for more user information by providing extra fields.

The user information can be retrieve include:

- id
- username
- displayname
- tinyurl
- profile_url
- gender
- dob

Parameters:

Required	Name	Type	Description
required	access_token	string	the access token provided by ZingMe to access GraphAPI
	fields	string	a string defined the extra user information to be retrieved, seperated by ",". e.g: ***fields="id,username,displayname"***

Direct URL to test:
Develope environment:

https://dev-graphapi-me.zing.vn/me/@<appname>?access_token=<access_token>

Production environment:

https://graphapi-me.zing.vn/me/@<appname>?access_token=<access_token>

Return: an array contains logged in user information (default contents include: id & username)

Example:
["**id**"]=> 8481406
["**username**"]=> "username234"

Throws: ZingMeApiException

6. *ZME_Me :: getFriends*

Description: Get friend list of logged user with access_token. The result will be a list of ids of user's friends.

Parameters:

Required	Name	Type	Description
required	access_token	string	the access token provided by ZingMe to access GraphAPI

Direct URL to test:

Develope environment:

https://dev-graphapi-me.zing.vn/me/friends/@<appname>?access_token=<access_token>

Production environment:

https://graphapi-me.zing.vn/me/friends/@<appname>?access_token=<access_token>

Return: an array contains all uids of logged user's friends.
Example:
[0]=> 273770
[1]=> 590794
[2]=> 984839
[3]=> 1039342
[4]=> 1669266
[5]=> 1985907
[6]=> 2157498
[7]=> 2234646
[8]=> 2989154
[9]=> 3036134

2.2.2. ZingMe User:

a. ***ZME_User :: getInfo***

Description: - Get profile info of user(s) by <**uid(s)**>.

- The <**uid(s)**> must be the friends with logged user. The <**uid(s)>** is/are mostly got by calling ***ZME_Me :: getFriends API (section 2.2.1.g)***

- Default user information which will be returned includes <**id**> & <**username**>. However, 3rd-party can request for more user information by providing extra fields.

The user information can be retrieve include:

- id
- username
- displayname
- tinyurl
- profile_url
- gender
- dob

- The return result only provides you 50 sets of ZingMe User information to ensure the performance of both your system and ZingMe system.

Parameters:

Required	Name	Type	Description
required	access_token	string	the access token provided by ZingMe to access GraphAPI
required	uids	array	a list of uid(s) of ZingMe user needed to be retrieved
	fields	string	a string defined the extra user information to be retrieved, seperated by ",". e.g: ***fields="id,username,displayname"***

Direct URL to test:

Develope environment:

https://dev-graphapi-me.zing.vn/user/info/@<appname>?access_token=<access_token>

Production environment:

https://graphapi-me.zing.vn/user/info /@<appname>?access_token=<access_token>

Return: a list of logged in user's friends information

Example:

[0] => Array ([id] => 273770 [username] => username1 [displayname] => displayname1)
[1] => Array ([id] => 3160610 [username] => username2 [displayname] => displayname2)
[2] => Array ([id] => 4024738 [username] => username3 [displayname] => displayname3)
[3] => Array ([id] => 1669266 [username] => username4 [displayname] => displayname4)
[4] => Array ([id] => 1985907 [username] => username5 [displayname] => username5)

2.2.3. Feed:

a. ZME_Feed :: generateSignatureKey

Description: generate a signature key for using with feed client library

Parameters:

Required	Name	Type	Description
required	secretKey	string	application secret key provided by ZingMe

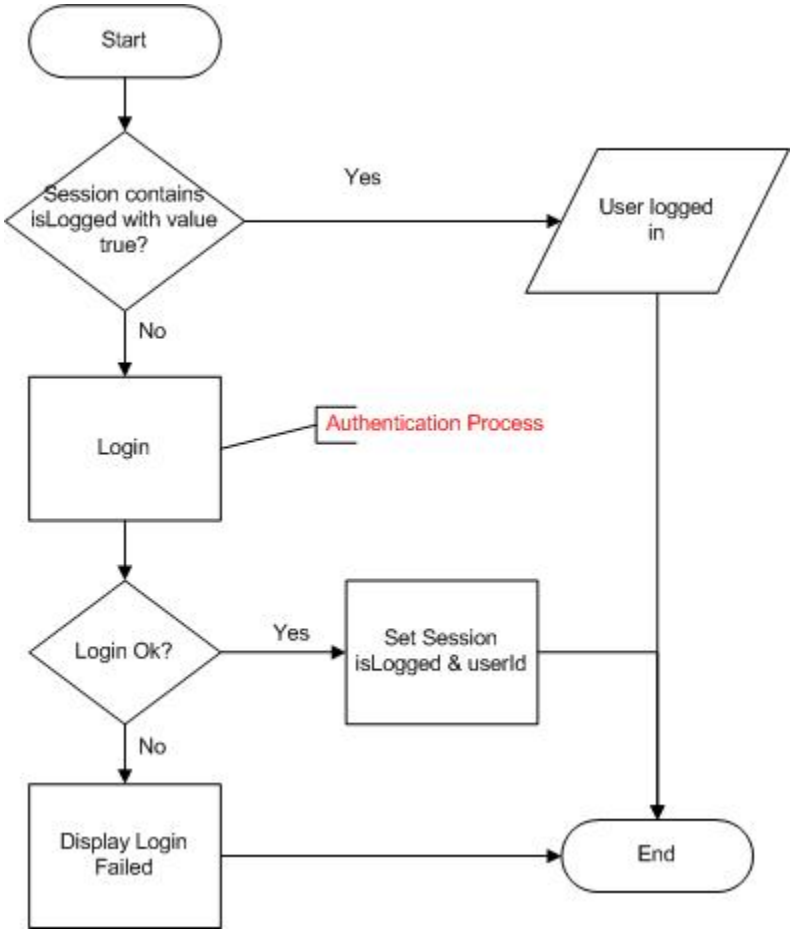
Return: a MD5 String

Example: 0913e821ac753ff83378d419573ddb3a

3. Case study:

3.1. Common Flow:

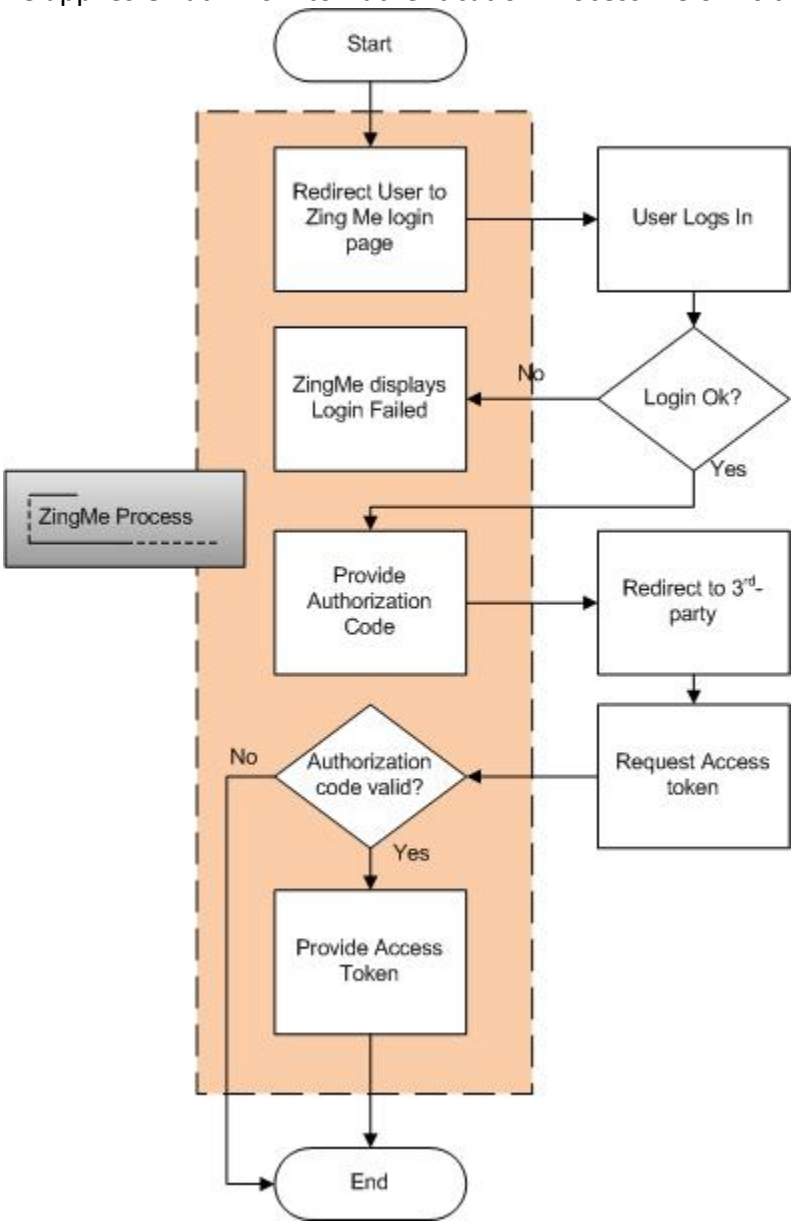
3.1.1. Authentication:



3.2. ZingMe in Action:

3.2.1. Authentication:

- ZingMe applies OAuth 2.0 into Authentication Process. Below is the flow :



4. Appendix:

4.1. ZingMeAPI Exception:

4.1.1. OAuthException (Range -1 => -99):

Exception Code	Message
-1	Authorized code invalid
-2	Authorized code expired
-11	Access token missing
-12	Invalid access token
-13	Access token had been expired

4.1.2. GraphAPIException (Range -1000 => -1100):

Exception Code	Message
-1000	Unexpected Exception
-1001	Request Invalid
-1002	Appname invalid

4.1.3. FunctionException (Range -100 => -1000):

Exception Code	Message
-100	User.getInfo: list of users larger then 50
-101	User.getInfo: uids param is not a list
-102	User.getInfo: uids param missing

4.1.4. ClientGraphAPIException:

Exception Code	Message
-10000	Invalid or no certificate authority found
-10001	Json decode error
-10002	Cannot get access token from signed request
-10003	Cannot get access token from authorized code
-10004	Cannot get userId from signed request
-10005	Bad signed JSON signature