International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 3 Issue 3, March - 2014

The Power of Facebook API

(Using a Powerful Data Mining Tool on The Most Popular Social Network)

Zubair Ahmed[1], Prof. Mausumi Goswami[2], Prof. K. Balachandran[3]
Department of Computer Science & Engineering
Christ University Faculty of Engineering
Bangalore

***Abstract*— This paper deals with learning the use of the Facebook Graph API for web development through the use of Facebook Applications built using two software development kits mainly the PHP SDK and the JavaScript SDK. Alongside the Graph API, a tailor made SQL-like language for querying data know as Facebook Query Language will be elaborated on. The power and usability of these tools and development kits will be demonstrated through an example which using the Graph API to retrieve the user's personal information and tokenize and display the various "likes" of the person which can be used to provide a personalized experience by a third-party web portal for that veryauthorized user.**

Index Terms—open graph, graph API, open graph protocol, javascript sdk, php sdk, fql, paths.

I. INTRODUCTION

If you want to develop applications that connect to the social web, then Facebook is the place from where you should begin your journey. Over the past seven years, Facebook has completely revolutionized the concept of social networking and transformed the way people interact over the web with each other. With an approximate of 800 million user base, this is definitely the place you won't want to miss.

Today, Facebook has become the synonym for social networking and it has surpassed its competitors, such as Myspace and Friendster, by huge proportions in the world of the social web. The structured form, the interaction portal, and the perfectly dedicated platform provide users with a seamless and easy sharing experience. "Facebooking" has become the 'key phrase' among people of most of the social domains and areas. From college students to the entrepreneurs, everyone happens to be a part of a web-based social graph and Facebook connects them all.

Since, social networking as a whole has a lot of competition Facebook excels as a social networking site by displaying the following attributes uniquely over other competitors:

1. It's simply "cool!": Keep in touch with friends, make groups for entertainment, search for people, view pictures, view videos, send messages, share wall posts, create forums, advertise, play games, and create applications; name any social activity and Facebook is already equipped with it. Facebook caters to the needs of virtually everyone and has everything for its users.

2. It's different: Facebook stands out from its competitors, such as Myspace and Friendster, and has done something which no other social networking site has done before. It has introduced the concept of page and millions of third-party applications, hosted both inside and outside Facebook, which gives users a complete social experience. It has moved beyond the social aspect of strictly being just a site by allowing being in touch with an always-changing network of friends and colleagues.

3. It inspires: Facebook is not just about sharing a similar interest, it's about networking. It allows users to express themselves, communicate with others, and createprofiles that highlight their talents and experience, and also advertise and market inexpensively.

4. It's not stopping: Facebook has continued to grow from the start. With the huge increase in its user base, it has been continuously updating and upgrading itself with all the technological changes happening. Facebook, with the advent of Graph API, Open Graph protocol, Facebook Ads API, and Social Plugins,Facebook Credits has taken care of the changing technology.

5. It's stated everywhere: The popularity of Facebook has been acknowledged worldwide.

With more than 800 million active users and integration of more than 7 million websites, Facebook has a lot of potential for the third-party application developers. It is an attractive platform to promote a particular idea or business and spread it among the millions of users. Facebook offers its users as potential customers, making it fascinating and potentially profitable for the developers. The establishment of the Facebook Developers Platform has provided developers with many new and exciting ways to engage Facebook users. So, the benefits for the developers are as follows:

Make money: Apart from advertising, the platform allows developers to earn money by running their apps on other websites.

It's free: Developing an application is exciting and free.

Allows collection of data: As part of the application, Facebook allows developers to collect selected information from their users. For example, you can collect users' views on a particular interest by using polling.

Spreading your word through millions of users: Updates by the Facebook application on a user's wall is another attractive way to publicize your idea or product. These updates, when viewed by friends and friends of friends, expose your application virally to the millions out there.

Integrate with Facebook Portal:Social plugins, such as the Likebox and Recommendation plugin, make it possible to draw more traffic.

Building business: An application provides a personalized interface to interact with people and caters to their specific needs. This, in turn, can help to promote and facilitate marketing of certain products and services.

Advertisement: You can promote Facebook applications simply and expeditiously. Facebook aims at creating its advertisers more satisfied. The introduction of the CPC advertising, which permits its advertisers to take control on the number ofadvertisements generate per click, is quite its boldest and bravest move.

II. The Open Graph

There are three parts to Open Graph. The most commonly used are Facebook's social plugins.

1. The Open Graph Protocol: The Open Graph protocol allows Web developers to tag their Web pages for the social graph. This simple protocol is presented to anybody, and can also be added to any Web-page. It enables Web site owners to identify their pages as "objects," such as people, activities, groups, organizations, products, placesand services. All this enables Facebook to index and map an ecosystem of all objects, their inter-relations and their connections with Facebook users.

2. Graph API: Facebook's Graph API is designed to provide access to every object in Facebook's database: users, photos, videos, statuses, conversations and their relations with each other. Developers use the Graph API to integrate custom applications such as apps, pages, groups, sites, widgets, mobile apps, and use the Facebook database as a social data provider. There are two levels of access. At the first, developers need to log into Facebook via the Graph API to urge the user's basic information. At the second level, they need extended permissions from the user to grant additional (read: non-public) information. With extended permission, the developers can query basic information and can also get a list of "Likes" relatedtoevery profile — providing the marketer more opportunities to form a custom experience.

3. Social plug-ins: The goal of social plug-ins is to extend the power of Facebook and socialize the Web with a simple pluggable interfaces and modules. There are 8 social plug-ins, which are the easiest way for anyone to integrate Facebook's social features on their website portal. By the addition of a few lines of code to a site, visitors have the ability to engage with both that site and its Facebook presence and perform social actions without ever leaving the site.

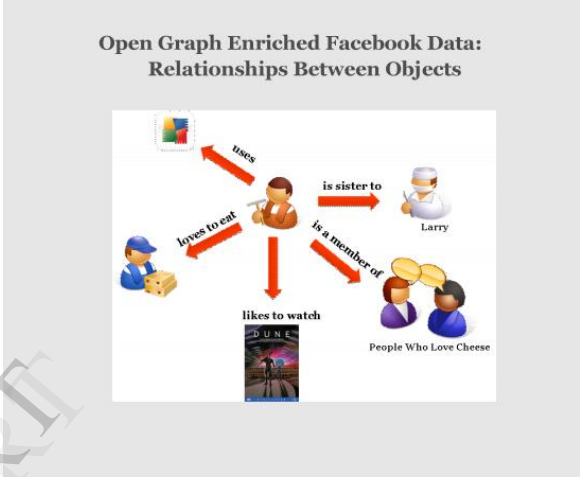


Fig 1. Relationship between objects

III. THE GRAPH API

The Graph API is the core of Facebook Platform, providing developers a mechanism to read from and write data into Facebook. The Facebook's Graph API presents a simple and a more consistent view of the Facebook social graph, also uniformly representing objects in the graph (e.g., people, photos, events, and pages) and the connections between them (e.g., friend relationships, shared content, and photo tags). The Graph takes all the existing data available on the web and makes it easily accessible through the API. We can use this API to do varies things such as make the web more social and find new friends who have certain tastes or are from some new geographical area. The web representing Facebook is basically considered as a bunch of documents showing special characteristics and categories, this data is consumable and there is demand to access this data and social graph and that's where the Graph API comes into play. There are entire set of companies in the market whose sole purpose is to make engagement more effective and efficient in social media and the power behind that is the Graph API. The Graph API by web developers of third party websites to authenticate users and view specific information of these users through the API for a more special experience which becomes a true game changer in today's competitive environment.There are over 550 million people using

Facebook around the world, making it the de facto, broadest and the bestreach social network in the world. Facebook collects personal profile information such as name, school, and a list of friends. Additionally, it is also collecting personal tastes — both through tracking which brands a user follows, but also by remembering which products, news stories, or other objects the user "Like", anywhere on the Internet. Combined with the information from the user's friend's "Likes", and follows, it can paint a detailed picture of the user'spersonality and life. Social is now becoming a feature of developers rather than a destination with new players in the content platforms with social auth and sharing such as Pinterest for graphics, Instagram for photos, Spotify for music, BranchOut for connections etc. forming a symbiosis between the content providers and the social network.

The Graph API is becoming the primary way of getting content and content into and from Facebook's social graph. Graph API is actually a low-level HTTP-based API to get or post data through different techniques and to perform tasks that normally an application would. Most Graph API's which an application can request my getting an access tokens by implementing Facebook Logins. This can be implemented with either or both of the two main SDKs that are the JavaScript SDK and the PHP SDK.PHP SDK has been released under Open Source License and is presently hosted at GitHub.

The Facebook SDK for PHP enables developers to implement a rich set of server-side functionality for accessing Facebook's API. This includes method to access all of the features of the Graph API and FQL. It additionally works alongside the Facebook SDK for JavaScript to help you implement Facebook Login. The Facebook SDK is usually and typicallyaccustomed to perform operations as an app administrator, however can even be used to perform operations on behalf of the current session user. By removing the requirement to manage access tokens manually, the SDK greatly simplifies the method of authentication and authorizing users for your app.

In order to use the Graph API and implement it Facebook application used be developed and in turn to develop, create, and launch a Facebook application, you need to have a domain name and a web hosting space. From here on, we will refer to this web hosting space as the server which should have two key PHP extensions installed:

1. PHP cURL extension
2. PHP JSON extension

PHP cURL extension provides us with a powerful library for making HTTP requests, known as cURL, and has been specifically designed to fetch data from remote sites. This library is used to post requests to Facebook servers using Facebook Graph API. Similarly, we need PHP JSON(JavaScript Object Notation) extensionto convert JSON encoded data to PHP arrays for our logic processing and data mining.

IV Registering application And Setup

Facebook application can be considered as a code snippet written by a web developer which is capable of extracting data of the users and able to perform some meaningful tasks on the data. Facebook application is always assigned a unique ID and a private key which when combined help distinguish each application from the other and manage security. Creating an application is a standard procedure by becoming a verified Facebook developer through a simple verification process.

Here is a sample application I built for my a project by using a web hosting on Heroku server which is an easy and free way to hosting application online; Heroku provides the developer with all the tools they need to iterate quickly, and adopt the right technologies for their project. Build modern, maintainable apps and instantly extend them with functionality from hundreds of cloud services providers without worrying about infrastructure.

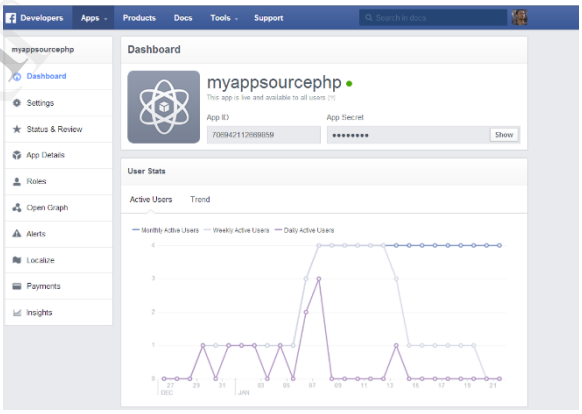


Fig 2. Facebook's "Manage App" Portal

The basic set up which can be followed through easily through the developer's portal. As a part of the basic application configuration the following should be looked into by the developer:

Secure connection: It is quite necessary if the application is going to have secure content. It can be achieved by setting the secure canvas URL, this secure version of the Canvas URL is used by Facebook when your application is accessed by a user over a secure connection (https).

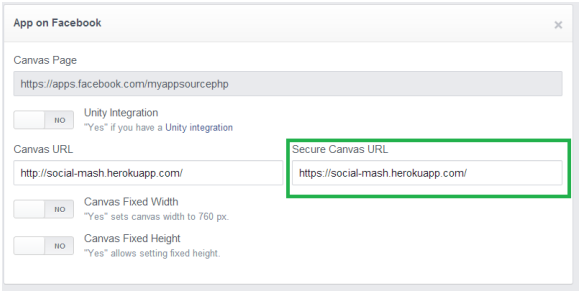


Fig 3. Specifying the Canvas URL

Sandbox Mode:Sandbox mode restricts the access of an application to only its developers. This can be used by developers in the development and testing phase.

Resetting an application's secret key: If the security of the application seemed to be compromised then the application's secret key can be easily reset.

Apart from these there are varies tweaks and features that can be changed easily to suit the needs of the developer.

To configure the application to work we need to first import or upload the PHP SDK file named facebook.php on to the server and some php configuration code along with it.

```
<?php
require_once 'facebook.php';
$facebook = new Facebook(array(
'appId' => 'your_application_id',
'secret' => 'your_application_secret',
'cookie' => true,
));
?>
```

Snippet of the basic configuration setup

We have created an instance of our Facebook application by declaring a new object, $facebook, of the Facebookclass. We pass an array with various settings as an argument to the constructor. These parameters are appId, secret, and cookie. Here, appIdrefers to the application ID and secretrefers to the secret key. The cookie value truesimply implies that the cookie will be used to store the session information after authentication.Usually, a Facebook application consists of various pages and hence there are multiple PHP files. Instead of specifying the Facebook configuration parameters again and again it is advisable to create a common configuration file for all pages.

Using Facebook API is all about getting the personalized experience so hence we need authentication and session verification which becomes an important aspect.

V. FACEBOOK LOGIN

Facebook uses the OAuth 2.0 protocol for authorization and authentication, it involves three steps they are user authentication, app authorization, app authentication.

Below is a snippet for logging in a user and displaying the user ID.

```
<?php
$user = $facebook->getUser();
if ($user):
echo '<p>User ID: ', $user, '</p>';
echo '<p><a href="logout.php">logout</a></p>';
else:
$loginUrl = $facebook->getLoginUrl(array(
'diplay'=>'popup','scope'=>'read_mailbox,read_str
eam',
'redirect_uri' =>
'http://apps.facebook.com/myappsourcephp' ));
echo '<p><a href="', $loginUrl, '"
target="_top">login</a></p>';
endif;
?>
```

Snippet for the login procedure

Here we first get the user from the Facebook object then we try and output the user ID,$user, we receive back from the Facebook object, which might return a null or 0 ID by running it. It is because the Facebook user has not been authenticated. In order to authenticate them we need to get a login URL, the login URL is not just a regular link but a link that passes the proper security transactions to Facebook. So we check if the user is with a normal user ID we print the normal user ID, otherwise, we use the get login URL method from the Facebook object and feed it into a variable,$loginUrl. It has options specifying parameters and we see that we can specify the 'scope'which specifies the permissions to request from the user, if not specified the basic permission will my default be requested from the user. 'redirect_uri' which is quite similar to URL and which specifies where the user will go after the user has logged in, if not specified the user will be redirected to the same URL. Which is optional just specifieshow to render the dialog box for the login. It depends on whether the user is logging in through an iFrame or through a third party website. The procedure is something a developer quite necessarily has to play around with.

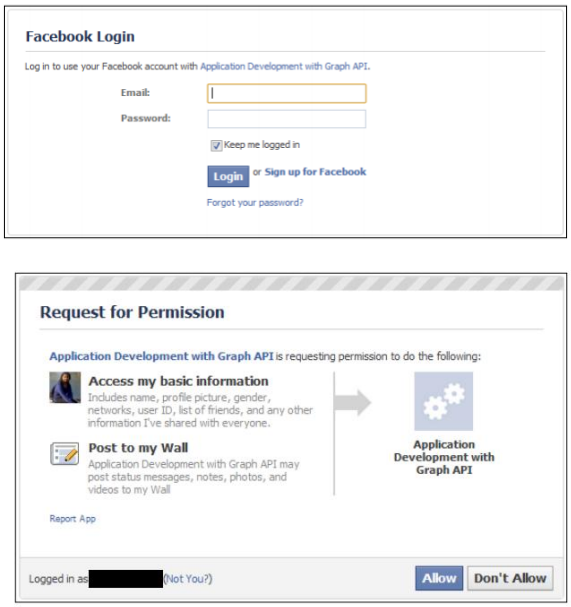


Fig 4. Authorization to getting the required permissions

VI. Working with The Open Graph Data

The whole purpose of working with the API is to work with the data that Facebook collects and stores about its users. The Graph API can be considered as the built in language to use the social graph. Here we will see can example of how to view and work with data of users who have been authenticated by the Facebook application giving it the necessary permissions. Through PHP we can make a connection to the API by using the API object of the Facebook Class. To call the Graph API method we pass in the endpoint path we wish to retrieve as the first parameter and other optional parameters if necessary.

```
$facebook->api($path,$mathod,$params);
```

Graph API method format

The parameters are as follows:

1. `path` : It specifies the Graph API path for the request. For example "`/me`" to refer to the user's profile.
2. `Method` : It is an optional parameter which specifies the HTTP method for the specific request. To could be "`GET`", "`POST`" or "`DELETE`".
3. Params : It is an optional parameter which in turn shows parameters specific to the Graph API method being called. It is passed in as an associative array of "`'name' => 'value'`" pair. It comes of use especially when we need to send data back to Facebook.

Here is a snippet to demonstrate a simple call to the graph API method of the facebook object to get the logged in user's information:

```
$user_profile=$facebook->api('/me','GET');
```

API call for getting the user's information

The object that we get from this call is structure in the following manner:

```
{
  "id": "100006524327666",
  "name": "Maximus Decimus Meridius",
  "first_name": "Maximus",
  "middle_name": "Decimus",
  "last_name": "Meridius",
  "link":
"https://www.facebook.com/profile.php?id=10000
6524327666",
  "birthday": "04/21/1991",
  "hometown": {
    "id": "115353315143936",
    "name": "Rome, Italy"
  },
  "location": {
    "id": "106377336067638",
    "name": "Bangalore, India"
  },
  "favorite_teams": [
    {
      "id": "107844962567241",
      "name": "Chennai Super Kings"
    },
    {
      "id": "86037497258",
      "name": "Chelsea Football Club"
    },
    {
      "id": "197394889304",
      "name": "FC Barcelona"
    }
  ],
  "gender": "male",
  "political": "Indian National Congress ()",
  "timezone": 5.5,
  "locale": "en_US",
  "verified": true,
  "updated_time": "2014-01-07T05:23:08+0000"
}
```

The result in the form of a JSON object

Now we graph the specific name and location of the user in the following manner:

```
echo "<br><br>Name: " . $user_profile['name'];

echo "<br><br>From: " .
$user_graph1['location']['name'];
```

Snippet for displaying the user's name and location

Which in turn gives the result:

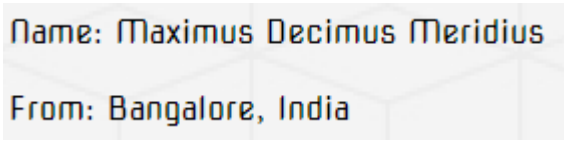


Fig 5. Output

The data as it can be seen comes back as an associative array which looks quite similar to a JSON object, we can access any part of this data. So if we wanted to just graph the name and location of the user. Depending on how much

of the profile the user has filled, privacy settings specified by the users for the profile and the permissions granted to the application a lot more useful data can be retrieved and give a more personalized service to the user can be provided with ease. Though the path specified looks suspiciously like a URL that's because it actually is. Facebook takes our path and send it to an application at graph.facebook.com so we can use it to generate different data through different more specialized paths the online API reference gives a bunch of examples that can be reviewed and tested. The results are seen in the form of JSON objects which are similar to PHP associative arrays.The is viewing the basic information about a user. The same can be done on the Facebook pages too.

```
$user_profile = $facebook-
>api('/cocacola','GET');
```

Snippet to get the information about a Facebook Page

The path above could be used to get all the generic information of official Coca-Cola Facebook page in the same JSON object format.

Quite similarly developers can retrieve information about other pages, events, groups, status messages, other application information, photos, photo albums, profile pictures, videos, notes check-ins etc. which can be quite useful in providing the much needed and important personalized experience for the users of the application or the web portal. Sometimes its need to view personal information not available for public view this can be achieved by permission or access tokens. An access token is an opaque string that identifies a user, app, or page and can be used by the app to make graph API calls. Access tokens are obtained via a number of methods, each of which are covered later in this document. The token includes information about when the token will expire and which app generated the token. Because of privacy checks, the majority of API calls on Facebook need to include an access token. There are different types of access tokens to support different use cases:

User Access Token – The user token is the most commonly used type of token. This kind of access token is needed any time the app calls an API to read, modify or write a specific person's Facebook data on their behalf. User access tokens are generally obtained via a login dialog and require a person to permit your app to obtain one.

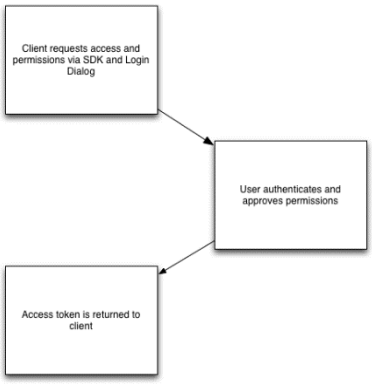


Fig 6. Procedure to obtain the access tokens

App Access Token – This kind of access token is needed to modify and read the app settings. It can also be used to publish Open Graph actions. It is generated using a pre-agreed secret between the app and Facebook and is then used during calls that change app-wide settings. You obtain an app access token via a server-to-server call.

Page Access Token – These access tokens are similar to user access tokens, except that they provide permission to APIs that read, write or modify the data belonging to a Facebook Page. To obtain a page access token you need to start by obtaining a user access token and asking for the manage_pages permission. Once you have the user access token you then get the page access token via the Graph API.

Client Token - The client token is an identifier that you can embed into native mobile binaries or desktop apps to identify your app. The client token isn't meant to be a secret identifier because it's embedded in applications. The client token is used to access app-level APIs, but only a very limited subset. The client token is found in your app's dashboard. Since the client token is used rarely, we won't talk about it in this document. Instead it's covered in any API documentation that uses the client token.

Whereas, when it comes to permissions during the basic login flow, your app receives access to a person's public profile and friend list. In order to access additional elements of their Facebook profile, or to publish content to Facebook on their behalf, you need to request permissions using the Login dialog as we had done in the snippet we had seen earlier. These permissions can be requested at the time of initial login or at any other point during the app experience.

There are various categories of permissions:

Email permissions—A user's email is a protected property and access to that information must be specifically requested by the app and granted by the user. It can be noted that there is no way for apps to obtain email addresses for a user's friends.

Extended Profile Properties—These Permissions cannot be revoked in the Login Dialog during the login flow, meaning they are non-optional for Users when logging into your app. If you want them to be optional, you should structure your app to only request them when absolutely necessary and not upon initial login.

Page Permissions—These are used for seamless ease of getting a given Facebook page's information completely. This permission is optional when presented to users in the Login Dialog. It can also be removed by a user after having granted it in their Privacy Settings.

Open Graph Permissions—Open Graph permissions allow your app to publish actions to the Open Graph and also to retrieve actions published by other apps.

Public Profile and Friend List—When a user logs into your app and you request no additional permissions, the app will have access to only the user's public profile and also their friend list.

Extended permissions—Extended Permissions give access to more sensitive info and the ability to publish and delete data, they are optional when presented to users in the Login Dialog. They can also be removed by a user after having granted them in their Privacy Settings. Apps should be built to handle revoked Permissions without reducing the user experience.

The Graph API Explorer provided online by Facebook helps to view and decide the various permission need for the application.

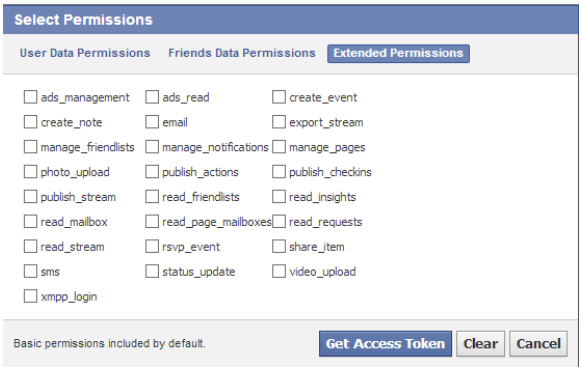


Fig 7. Different permissions that can be added to the scope as displayed in the Graph API Explorer

VII. SNIPPET OF AN SMALL EXAMPLE

Let's just demonstrate an example of querying the "likes" of a user. Through the Graph API we write a path and assign the result to a variable.

```
$user_likes = $facebook->api('/me?fields=likes');
```

Snippet to get the current user's "likes"

The data that will be returned will be in the form of a JSON object and pictured as follows:


```
"data": [
    {
      "category": "Book",
      "name": "Nelson Mandela: A Biography",
      "created_time": "2014-01-
07T05:28:54+0000",
      "id": "314754965302617"
    },
    {
      "category": "Book",
      "name": "Harry Potter and the Half-Blood
Prince",
      "created_time": "2014-01-
07T05:28:53+0000",
      "id": "103116576395865"
    },
    {
      "category": "Book",
      "name": "Two States",
      "created_time": "2014-01-
07T05:28:47+0000",
      "id": "107970509236401"
    },
    {
      "category": "Tv show",
      "name": "Batman: The Animated Series",
      "created_time": "2014-01-
07T05:28:14+0000",
      "id": "188119097891828"
    },
    {
      "category": "Tv show",
      "name": "MTV Roadies",
      "created_time": "2014-01-
07T05:27:59+0000",
      "id": "117394281646"
    },
    {
      "category": "Tv show",
      "name": "WWE Monday Night Raw",
      "created_time": "2014-01-
07T05:27:40+0000",
      "id": "114016181979317"
    },
    {
      "category": "Tv show",
      "name": "Popeye the Sailor",
      "created_time": "2014-01-
07T05:27:35+0000",
      "id": "108611845829948"
    },
    {
      "category": "Tv show",
      "name": "Man vs. Wild",
      "created_time": "2014-01-
07T05:27:30+0000",
      "id": "41036834883"
    },
    {
      "category": "Movie",
      "name": "Up",
      "created_time": "2014-01-
07T05:26:56+0000",
      "id": "118194755634"
    },
    {
      "category": "Movie",
      "name": "The Lion King",
      "created_time": "2014-01-
07T05:26:50+0000",
      "id": "12393266550"
    },
```

```
    {
      "category": "Movie",
      "name": "Iron Man",
      "created_time": "2014-01-
07T05:26:41+0000",
      "id": "7057882289"
    },
    {
      "category": "Movie",
      "name": "Sherlock Holmes: A Game of
Shadows",
      "created_time": "2014-01-
07T05:26:38+0000",
      "id": "138409202896344"
    },
    {
      "category": "Movie",
      "name": "X-Men Movies",
      "created_time": "2014-01-
07T05:26:35+0000",
      "id": "112783808793521"
    },
    {
      "category": "Movie",
      "name": "How to Train Your Dragon",
      "created_time": "2014-01-
07T05:26:26+0000",
      "id": "96698020019"
    },
    {
      "category": "Movie",
      "name": "Avatar",
      "created_time": "2014-01-
07T05:26:23+0000",
      "id": "202953045644"
    },
    {
      "category": "Movie",
      "name": "Avengers",
      "created_time": "2014-01-
07T05:26:14+0000",
      "id": "126757470715601"
    },
    {
      "category": "Movie",
      "name": "Iron Man 2",
      "created_time": "2014-01-
07T05:26:09+0000",
      "id": "128825002158"
    },
    {
      "category": "Movie",
      "name": "Batman: The Dark Knight",
      "created_time": "2014-01-
07T05:26:07+0000",
      "id": "12887942787"
    },
    {
      "category": "Movie",
      "name": "Fast & Furious",
      "created_time": "2014-01-
07T05:25:59+0000",
      "id": "533822029986779"
    },
    {
      "category": "Professional sports team",
      "name": "Chennai Super Kings",
      "created_time": "2013-12-
30T07:21:26+0000",
      "id": "107844962567241"
    },
    {
      "category": "Professional sports team",
      "category_list": [
        {
          "id": "109976259083543",
          "name": "Sports Venue & Stadium"
        },
        {
          "id": "186982054657561",
          "name": "Sports & Recreation"
        }
      ],
      "name": "Chelsea Football Club",
      "created_time": "2013-12-
30T07:21:10+0000",
      "id": "86037497258"
    },
    {
      "category": "Professional sports team",
      "name": "FC Barcelona",
      "created_time": "2013-12-
30T07:20:57+0000",
      "id": "197394889304"
    },
    {
      "category": "Interest",
      "name": "Boxing",
      "created_time": "2013-12-
30T07:20:12+0000",
      "id": "111693808849802"
    },
    {
      "category": "Sport",
      "name": "Rugby football",
      "created_time": "2013-12-
30T07:20:09+0000",
      "id": "160156990676711"
    },
    {
      "category": "Sport",
      "name": "Sport sailing",
      "created_time": "2013-12-
30T07:20:01+0000",
      "id": "108163715884176"
    }
  ]
```


JSON output object displaying the user's "likes"

Usually the data for a typical Facebook page will be provided in a number of pages and can be traversed by the use of pagination links.This data is quite unreadable to the common user and has to be made use of. So, we build a tokenizer which breaks down the result into meaningful tokens.Below is a snippet for a tokenizer which retrieves the relevant data and categories them into "Movies", "TV shows", "Sports" and the "Sports Team".

```
echo 'Books<ul>';
foreach ($user_likes['likes']['data'] as $likekey
=> $likevalue)
{
if ($likevalue['category']=="Book")
        {
echo '<li>',$likevalue['name'],'</li>';
        }
}
echo "</ul>";
echo 'Movies<ul>';
```

```
foreach ($user_likes['likes']['data'] as $likekey
=> $likevalue)
{
if ($likevalue['category']=="Movie")
        {
echo '<li>',$likevalue['name'],'</li>';
        }
}
echo "</ul>";
echo 'TV Shows<ul>';
foreach ($user_likes['likes']['data'] as $likekey
=> $likevalue)
{
if ($likevalue['category']=="Tv show")
        {
echo '<li>',$likevalue['name'],'</li>';
        }
}
echo "</ul>";
echo ' Sports Teams<ul>';
foreach ($user_likes['likes']['data'] as $likekey
=> $likevalue)
{
if ($likevalue['category']=="Professional sports
team")
        {
echo '<li>',$likevalue['name'],'</li>';
        }
}
echo "</ul>";
echo ' Sports<ul>';
foreach ($user_likes['likes']['data'] as $likekey
=> $likevalue)
{
if ($likevalue['category']=="Sport")
        {
echo '<li>',$likevalue['name'],'</li>';
        }
}
echo "</ul>";
```

Snippet of the tokenizer for tokenizing the user's "likes"

Algorithm

Step 1: START

Step 2: Assign the object returned by $facebook->api('/me?fields=likes') to $user_likes variable

Step 3: Check if $user_likes?

If true goto step 4 else goto step 10

Step 4: Iterate through $user_likes['likes']['data'] using an index $likevalue as $likekey

Step 5: Check if $likevalue['category']=="Book"

Output $likevalue['name']

Goto step 3

Step 6: Check if $likevalue['category']=="Movies"

Output $likevalue['name']

Goto step 3

Step 7: Check if $likevalue['category']==" TV Shows"

Output $likevalue['name']

Goto step 3

Step 8: Check if $likevalue['category']=="Sports"

Output $likevalue['name']

Goto step 3

Step 9: Check if $likevalue['category']=="Sports Team"

Output $likevalue['name']

Goto step 3

Step 10: STOP

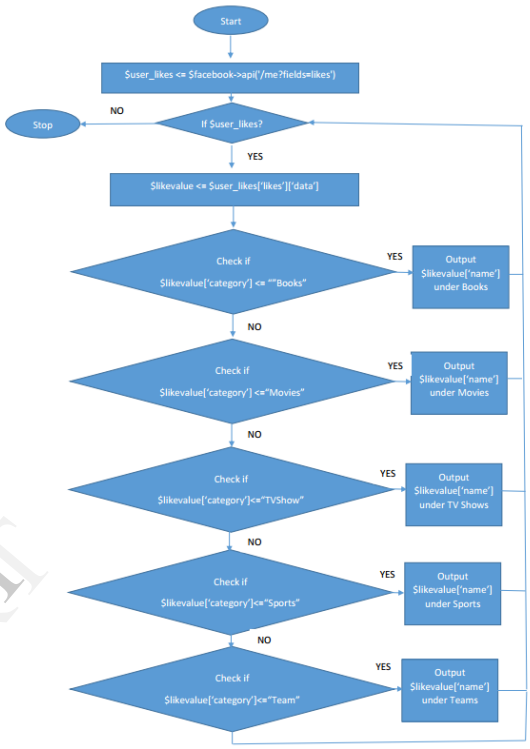


Fig 8. Flow Chart Diagram

The tokenizer uses the multidimensional array concept and traverses through a given page to categories the likes by the 'category', this can be done through all the pages present in the JSON object through the paginationconcept. This provides a given web developer the very interests and likes of a person so that only the relevant information can be provided to the user through which they can provide a personalized experience. This can be seen by an RSS reader example which analyses a person's interest, geographical location, other relevant information if need and provides a more personalized experience to the user by modifying the news feed through relevant modifications.

Fig 9. Output returned after tokenization in an HTML format

VIII.FACEBOOK QUERY LANGUAGE

Using the PHP SDK gives you the benefit of being able to access the Open Graph by a special language called the Facebook Query Language. It is very similar to SQL (structured query language) and is easy for PHP developers to pick up. The language has some feature which are not available in the Graph API such as the ability to batch multiple queries in a single call. Unlike SQL the FROM clause can only contain a single table that means we cannot do joints with this language. However, the IN keyword can do sub-queries but only within its own scope. Queries must be "indexable" which means we can only perform queries that include WHERE on fields that are "indexable", this can be done by simple qualification. The reference documentation can be looked up on the online to know which fields are "indexable" and can be used with the WHERE clause. It is not as powerful as SQL but is in fact another great tool to dig into the Open Graph.

Here is a simple example to demonstrate how power it is and how easy it is to use:

```
$moviefriends_graph = $facebook->api(array(
'method' => 'fql.query','query' => "SELECT name,
uid, movies, pic_square FROM user WHERE uid IN
(SELECT uid2 FROM friend WHERE uid1 = me()) AND
movies !='' LIMIT 10"));
```

Snippet of an FQL query to get the current users friend's movie likes or preferences

Facebook Query Language, or FQL, enables you to use a SQL-style interface to query the data exposed by the Graph API. It provides advanced features not available in the Graph API. In the above simple example we create a call to the Graph API this is going to be fed an array with a couple of variables they are `'method'` and a `'query'`.This query returns the name, user ID, movies and a picture from the user's friend(s) and it returns the following data:

```
{
  "data": [
    {
      "name": "Julius Ceaser",
      "uid": 608583697,
      "movies": "The Terminator, Bourne,
Transformers, School of Rock, Harry Potter,
500 Days of Summer, Avatar, Pirates of the
Caribbean, Inception, Batman: The Dark Knight,
The DaVinci Code, Devil, Saw, Terminator,
1408, Bourne, School of Rock, The Sixth Sense,
Alien vs. Predator, The Da Vinci Code (film),
The Dark Knight, Gridiron Gang, Transformers,
The Happening, Blood Diamond, The Lord of the
Rings film trilogy, Saw (franchise),300, The
Hangover, Angels & Demons, WALL•E, Fast &
Furious, Watchmen",
      "pic_square": "https://fbcdn-profile-
a.akamaihd.net/hprofile-ak-
ash2/1116351_608583697_1092558106_q.jpg"
    }
  ]
}
```

JSON output object format after execution of the query

We don't always get the same data and more than likely we don't get the richness of the information as we did by the Graph API. The Query then can be structured to precisely present the data of the user in a format that seem fit by the Web developer. This feature allows the developer to have his/her own liberties.

It's definitely always better to combine queries to get the data as per the developer's needs.

We can check the documentation at the following link for further information on FQL:

https://developers.facebook.com/docs/reference/fql

IX. JAVASCRIPT SDK AND JSON

The data in the Open Graph is stored in the JSON (JavaScript Object Notation) format. As we have seen earlier it structures the data in a particular useful format. It can be considered as a way to describe data so that JavaScript can easily interpret it. It is quite similar to working with RSS and XML feedsas we have already seen, though it is much easier to parse but can be a little bit harder to read. JavaScript SDK heavily relies on JSON.

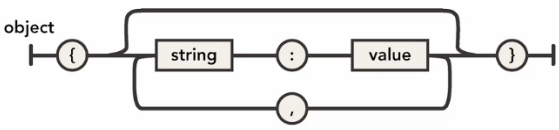


Fig 10. Basic JSON object format

A JSON object can be illustrated in the above diagram where it is wrapped in curly brackets, the strings (often variable names) and values are be nested with commas.

Here is an example demonstrating the use of JavaScript SDK by implementing the following JavaScript in a separate JavaScript File:

```
FB.getLoginStatus(function(response) {
if (response.status === 'connected') {
var uid = response.authResponse.userID;
accessToken = response.authResponse.accessToken;
```

```
//var fdata=" ";
FB.api('/me/friends', function(response) {
if(response.data) {
$.each(response.data,function(index,friend) {
$('#friends').html("* Friend Name :
"+friend.name+" ( ID : "+friend.id+" )");
        $("friend").html(fdata);
         } else {
             alert("Error!");
         }
    });
```

Javascript code for logging in and display the current user's friendlist

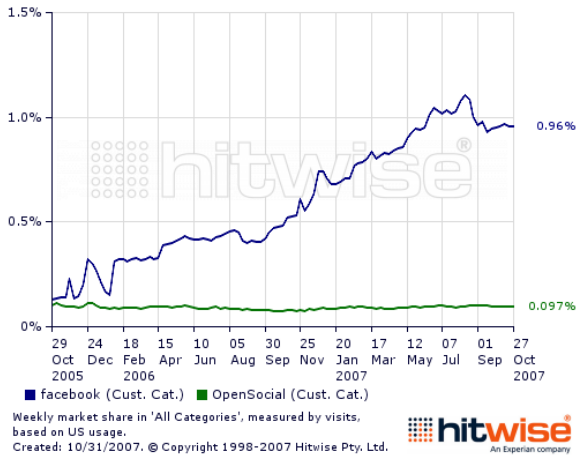
In the about JavaScript the access token is requested through which a list of the user's friends are printed along with their respective user ID in the div tag id which we specified as '#friends'in the HTML page.

We can check the documentation at the following link for further information on JavaScript SDK:

https://developers.facebook.com/docs/javascript/reference

X COMPARISON WITH OPENSOCIAL

The Open Graph Protocol and the Graph API has been quite the players in the social networking domain but it was not without tough competition from its competitors. OpenSocial that was developed by Google is a set of APIs rather than just one single API. OpenSocial was launched as a public specification that defines a component hosting environment and a set of APIs for web-based applications (in terms of Google Gadgets), which was a response to the Facebook Developer Platform in combination to work with the competitors such as Orkut, Bebo, hi5, LinkedIn, Friendster, Myspace etc. The inception of the idea was its large market potential at that time as compared to Facebook. Facebook's Platform and Google's OpenSocial share a lot of similarities and differ in quite a few grounds too.Though the development of Opensocket application made it possible to run gadgets on Facebook, it was not as successful.



FACEBOOK VS. OPEN SOCIAL (SIZE, GROWTH)
SOURCE: BILL TANCER, HITWISE

The difference between the two platforms can be elaborated as follows:
Development: OpenSocial client-side JavaScript oriented development. The presentation is mainly in the XHTML and XML formats. It utilized some common backend technologies such as PHP, Java, .NET which was all coded on a Canvas. Facebook Platform on the other hand is a server oriented API. The presentation is mainly in the HTML and a subset of HTML, i.e., FBML (Facebook markup language). Data is presented in the form of JSON objects which is called through Graph API paths or FQL as we have already seen. It also used PHP and JavaScript which was also coded/scripted on a Canvas.

Front-End: The front-end development with OpenSocial is open and uncontrolled. It provides access to thousands of networks spread across various social networking websites using Google Gadgets for UI programing model along with social plugins. The application is always hosted in an iframe.While, the Facebook Platform's front-end tethered and tightly controlled. It provides access to a single but extremely large and diverse network. The application may or may not be hosted in an iframe.

Application Structure: Opensocial can be either a client side application or a server-side application. While, Facebook applications are all hosted on a web site, that is, it will be hosted on a web server so it requires reasonable amount of server-side development using PHP libraries which are provided and supported by Facebook, while at the same time provides client libraries also.

Application Predominance: Opensocial applications can co-exist on a screen with different independent spaces assigned to them, while, in Facebook one application predominates since the canvas provided for it occupies most of the screen.

Control: Opensocial applications are provided a space the gadgets to be hosted on, after which the browser communicates directly with the application. In Facebook Platform, the web site acts as a reverse proxy between the browser and the application.

Application vs. applet: Opensocial concentrates on building small functionality focused applets as Google Gadgets infrastructure itself was not focused on heavy duty application development, while, Facebook concentrates on building fully functional applications.

Security: Since Opensocial was built from a thought of making Google Gadgets into a platform API and while Facebook spent a lot of time coming up with secure OAuth and tokens techniques, Facebook's platform seem to be much more secure.

Platform Data: In the case of server-side Opensocial applications the relationship graph and user preferences are

pushed into the application as a part of the URL itself, while, on the client side , this data is pulled from the Opensocial platform and combined with the application data on the client side.

Clearly it can be proclaimed the existence of quite a few dissimilarities. These dissimilarities can further be visualized in the following table:

Features	Opensocial	Facebook Platform	Note
Adaptability	Yes	No	Opensocial applications work anywhere on any website
Extensible	Yes	No	
Publish data	Yes	Yes	
Get data	Yes	Yes	
Persistence	Yes	No	OpenSocial providesan integrated solution for storing app data.
Application notifications	No	Yes	Facebook allows apps to communicate with users via email
Application requests	No	Yes	Facebook apps can send requests and invitations to non-app users
Spam Control	No	Yes	Facebook monitors and allows users to report spammy apps and takes appropriate actions
App permissions and privacy settings	No	Yes	Facebook provides fine-tuning of each app's permissions and privacy settings
Access to events, groups, photos, marketplace	No	Yes	
Application directory	No	Yes	
mage caching	No	Yes	

Though both the platforms have their own share of pros and cons the market opportunities and features of the Facebook Platform just seem to out shine Google's Opensocial platform.

XI Conclusion

The Open Graph Protocol and the Graph API leverages the power of hypermedia that proves that building simple representations of linked resources; in this case the Facebook object. We can see how useful and powerful these very tools are and how they can change the world of web semantics.The PHP SDK can be considered as better alternative to the JavaScript SDK and is more powerful. The utility of these tools and scenarios where they can be applied are vast and up to the developers to decide.This paper basically tries to show how easy and useful it is to use these tools irrespective of how confusing the documentation actually is. In a usual scenario it's better to implement both the PHP SDK and the JavaScript SDK hand-in-hand for better resource utility. Facebook's API is simple, consistent and inter-connected. It is true to the spirit of the web.

References

1. Shashwat Srivastava and Apeksha Singh, "Facebook Application Development with Graph API Cookbook" Packt Publishing, Birmingham – Mumbai, 2011.
2. Hilding Anderson and Rob Gonda, "Facebook Open Graph and the Future of Personalization", Marketing Strategy & Analysis, SapientNitro, Sapient Corporation, 2011.
3. Jesse Weaver and Paul Tarjan, "Facebook Linked Data via the Graph API", 2012
4. Facebook API reference documentation, https://developers.facebook.com/docs