

CS 5142

Scripting Languages

10/04/2013

PHP

Announcements

- Homework: Try to write a heap
- Make-up Prelim: Tell me by 10/18

Outline

- No HTML5 discussion.
- If interested in HTML and CSS:
<http://cs.nyu.edu/rgrimm/teaching/sp13-web/notes0206.html>
- Facebook SDK

Facebook SDK

- Install the Facebook PHP SDK from github:

```
git clone https://github.com/facebook/facebook-php-sdk.git
```

Git Basic Operations

```
# Download from the master branch
$ git clone <repository name>

# Add a file to the commit
$ git add <fileName>

# Commit files to local repository
$ git commit -a -m "<message>"

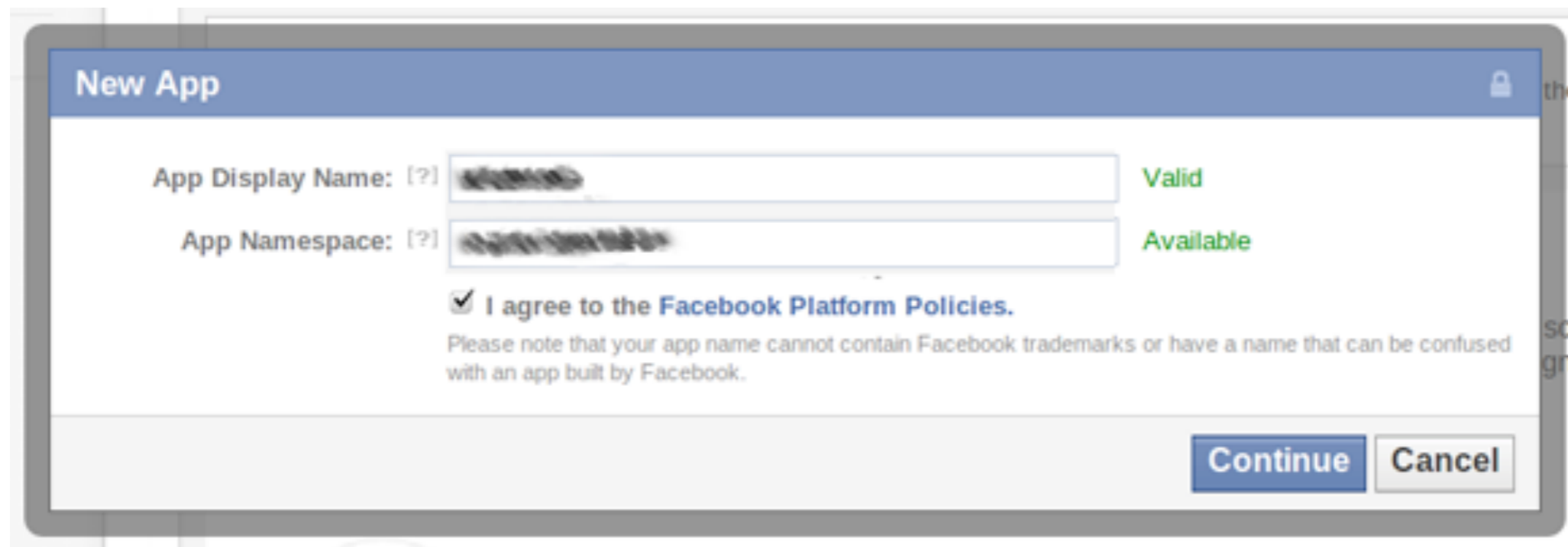
# Upload changes to the server
$ git push
```

- Distributed reversion control
- Can work “offline”
- Developed by Linus Torvalds for the Linux kernel

Create New App

Go to: <https://developers.facebook.com/apps/>

Create new app:



The screenshot shows the 'New App' form on the Facebook Developer portal. It includes two input fields: 'App Display Name' and 'App Namespace'. The 'App Display Name' field is marked as 'Valid' and the 'App Namespace' field is marked as 'Available'. Below these fields is a checkbox labeled 'I agree to the Facebook Platform Policies.' which is checked. A note below the checkbox states: 'Please note that your app name cannot contain Facebook trademarks or have a name that can be confused with an app built by Facebook.' At the bottom right of the form are two buttons: 'Continue' and 'Cancel'.

After authorization, Facebook will redirect you to a new URL

Provide a URL for your application (REDIRECT_URI)

Note the *App ID* and *App Secret*

Minimal Application

```
// Get the Facebook SDK
require_once("facebook.php");

$config = array();
$config['appId'] = 'YOUR_APP_ID';
$config['secret'] = 'YOUR_APP_SECRET';
$config['fileUpload'] = false; // optional

// Facebook object
$facebook = new Facebook($config);

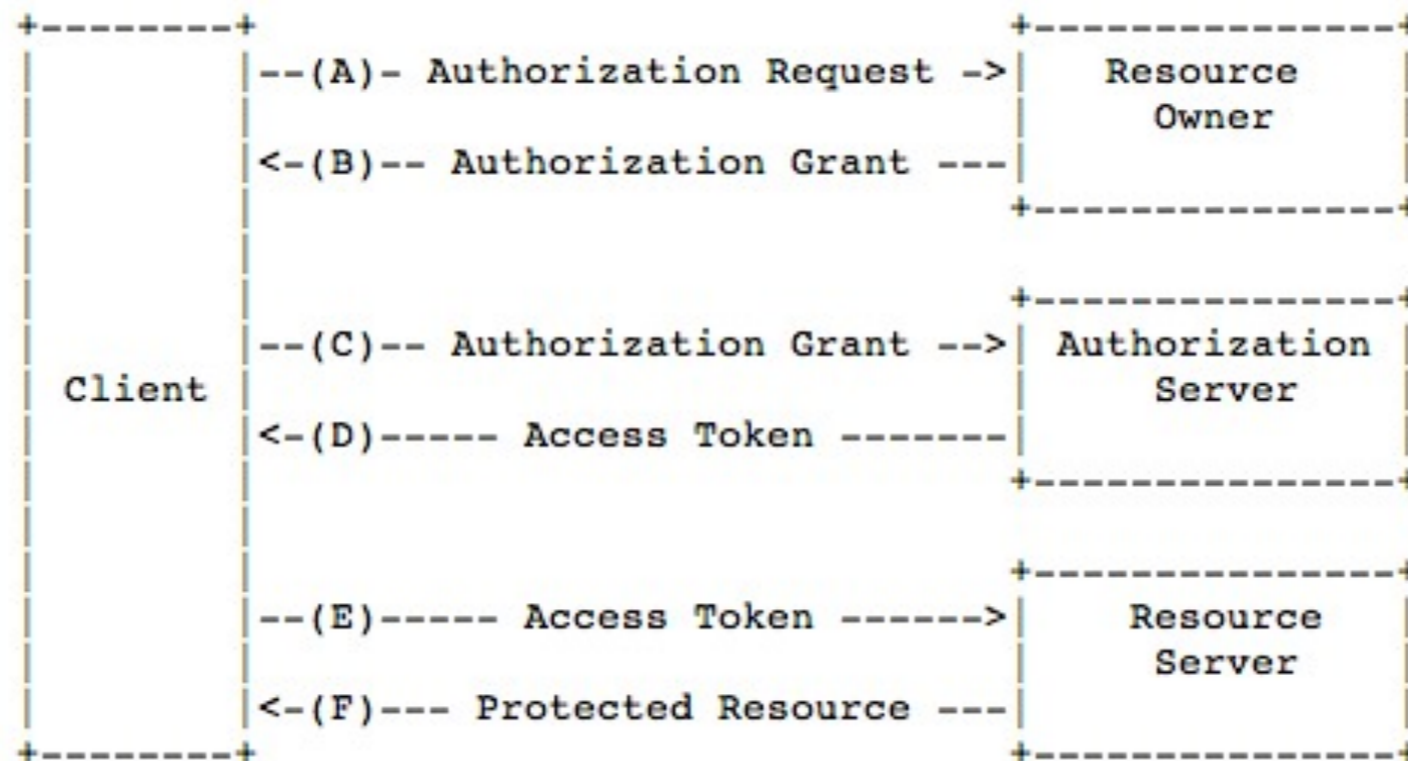
// Get User ID (if available)
$user = $facebook->getUser();
```

Facebook Object

api	Call Graph API or an FQL query
getAccessToken	Get the access token, give permission to app to use email, profile properties, friend list, etc.
getLoginStatusUrl	Return a URL based on user's login status
getLoginUrl	Return a URL that user click to login, authorize app, redirect back to app
getLogoutUrl	Return a URL that user click to logout of Facebook session, redirect back to app
getUser	Returns userid of the current user

OAuth

- OAuth 2.0 open standard authorization protocol
- Gives 3rd party apps access to an HTTP service for a limited time



Authorization

```
$params = array(  
    "redirect_uri" => REDIRECT_URI,  
    "scope" => "email,read_stream,publish_stream,  
                user_photos,user_videos");  
echo '<a href="' . $fb->getLoginUrl($params) . '">Login</a>';
```

Authorization

```
// Permission denied  
http://example.com/facebook/myapp.php?  
error=access\_denied&error\_reason=user\_denied&error\_description=The+user  
+denied+your+request.
```

```
// Permission granted, with code parameter appended  
http://example.com/facebook/myapp.php?code=TOKEN\_VALUE
```

```
// Code value used to get access token  
https://graph.facebook.com/oauth/access\_token?  
client\_id=FACEBOOK\_APP\_ID&redirect\_uri=FACEBOOK\_REDIRECT\_URI&client\_secre  
t=FACEBOOK\_APP\_SECRET&code=TOKEN\_VALUE
```

SOAP vs. REST

- **A web API is a server-side interface for web applications**
- **SOAP = Simple Object Access Protocol**
 - XML on top of HTTP or TCP, describes functions and types of data
 - Successor of XML-RPC
 - Usually associated with a service-oriented architecture
- **REST = Representational State Transfer**
 - Very lightweight
 - Typically uses HTTP PUT/GET/POST/DELETE instead of an XML
 - Usually associated with a resource-oriented architecture
 - Facebook uses REST

Graph API

Read the user's feed of status messages with a GET request:

```
https://graph.facebook.com/me/feed?access\_token=ACCESS\_TOKEN
```

In your script, it looks like this:

```
$data = $facebook->api('/me/feed');
```

Public Profile

GET request:

```
https://graph.facebook.com/soule/
```

In your PHP script:

```
if ($user) {  
    try {  
        // Proceed knowing you have a logged  
        // in user who's authenticated.  
        $user_profile = $facebook->api('/me/');  
    } catch (FacebookApiException $e) {  
        error_log($e);  
        $user = null;  
    }  
}
```

User Information

```
{  
  "id": "123456789",  
  "name": "Robert Soule",  
  "first_name": "Robert",  
  "last_name": "Soule",  
  "link": "http://www.facebook.com/soule",  
  "username": "soule",  
  "gender": "male",  
  "locale": "en_US"  
}
```

JSON

- Javascript Object Notation
- Derived from JavaScript
- Human-readable data format
- Alternative to XML
- Basic data types: Number, String, Boolean, Array, Object, null
- Structure characters { } [] : ,

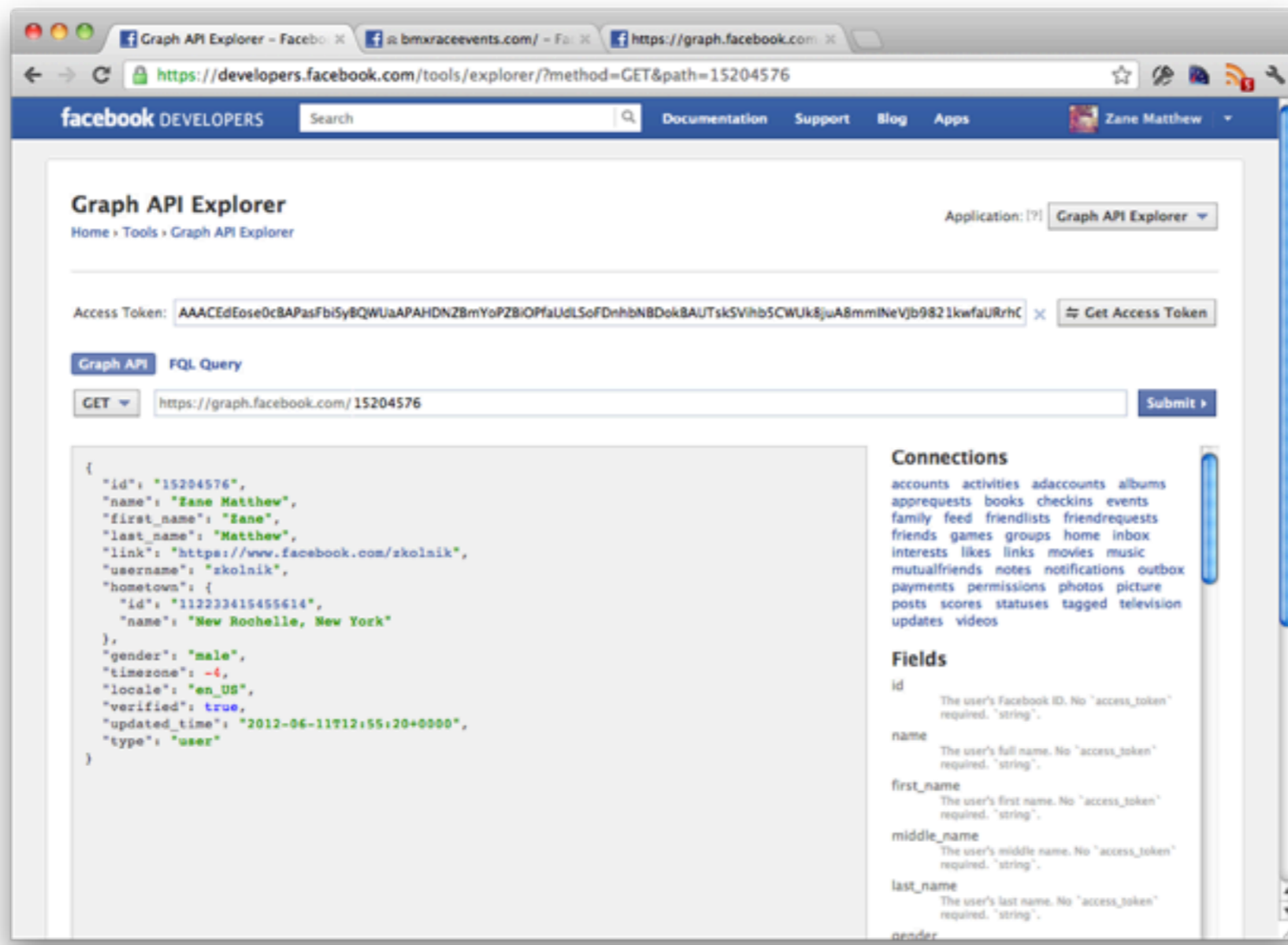
JSON Example

```
<!DOCTYPE html>
<html>
<body>
<h2>JSON Object Creation in JavaScript</h2>
<p>
Name: <span id="jname"></span><br />
Age: <span id="jage"></span><br />
Address: <span id="jstreet"></span><br />
Phone: <span id="jphone"></span><br />
</p>
<script>
var JSONObject= {
"name":"John Johnson",
"street":"Oslo West 555",
"age":33,
"phone":"555 1234567"};
document.getElementById("jname").innerHTML=JSONObject.name
document.getElementById("jage").innerHTML=JSONObject.age
document.getElementById("jstreet").innerHTML=JSONObject.street
document.getElementById("jphone").innerHTML=JSONObject.phone
</script>

</body>
</html>
```

Graph Explorer

<https://developers.facebook.com/tools/explorer>



The screenshot shows the Facebook Graph API Explorer interface. The browser address bar displays the URL `https://developers.facebook.com/tools/explorer/?method=GET&path=15204576`. The page header includes the Facebook Developers logo, a search bar, and navigation links for Documentation, Support, Blog, and Apps. The user's name, Zane Matthew, is visible in the top right corner.

The main content area is titled "Graph API Explorer" and includes a breadcrumb trail: Home > Tools > Graph API Explorer. An "Application" dropdown menu is set to "Graph API Explorer".

The "Access Token" field contains the token `AAACEdEose0cBAPasFbi5yBQWUaAPAHDNZBmYoPZBIOPfaUdlSofDnhbNBDokBAUTsk5Vihb5CWUk8juA8mmiNeVjB9821kwfaURrhC`, with a "Get Access Token" button next to it.

Below the access token, there are two tabs: "Graph API" (selected) and "FQL Query". The "Graph API" tab shows a "GET" method selected and the path `https://graph.facebook.com/15204576` entered in the input field, with a "Submit" button.

The response is displayed in a code block on the left, showing a JSON object for a user profile:

```
{
  "id": "15204576",
  "name": "Zane Matthew",
  "first_name": "Zane",
  "last_name": "Matthew",
  "link": "https://www.facebook.com/zkolnik",
  "username": "zkolnik",
  "hometown": {
    "id": "112233415455614",
    "name": "New Rochelle, New York"
  },
  "gender": "male",
  "timezone": -4,
  "locale": "en_US",
  "verified": true,
  "updated_time": "2012-06-11T12:55:20+0000",
  "type": "user"
}
```

On the right side, there are two sections: "Connections" and "Fields".

The "Connections" section lists various graph endpoints: accounts, activities, adaccounts, albums, apprequests, books, checkins, events, family, feed, friendlists, friendrequests, friends, games, groups, home, inbox, interests, likes, links, movies, music, mutualfriends, notes, notifications, outbox, payments, permissions, photos, picture, posts, scores, statuses, tagged, television, updates, videos.

The "Fields" section lists the fields returned in the JSON response, with descriptions for each:

- `id`: The user's Facebook ID. No "access_token" required. "string".
- `name`: The user's full name. No "access_token" required. "string".
- `first_name`: The user's first name. No "access_token" required. "string".
- `middle_name`: The user's middle name. No "access_token" required. "string".
- `last_name`: The user's last name. No "access_token" required. "string".
- `gender`: (partially visible)

Update a User

Post to status feed:

```
<?php
$data = array("message" => "Hello World!");
$status = $facebook->api("/me/feed", "POST", $data);
```

Upload a photo:

```
<?php
$facebook->setFileUploadSupport(true);
$data = array(
    "name" => "a vacation photo",
    "image" => "@/home/soule/vacation/img42.jpg");
$status = $facebook->api("/me/photos", "POST", $data);
```

PHP: The Bad

Reference:

<http://me.veekun.com/blog/2012/04/09/php-a-fractal-of-bad-design/>

Examples:

```
“foo” == TRUE, “foo” == 0, but 0 != TRUE
```

```
NULL < -1 and NULL == 0
```

Evaluating PHP

Strengths

- Simplicity
- Portability
- Large libraries
- Many database bindings
- Popularity

Weaknesses

- Error handling
- Lack of scalability
 - Compared to Java
- Low-level
 - Compared to Ruby on Rails or Google Web Toolkit

Last Slide

- Next week: client-side JavaScript