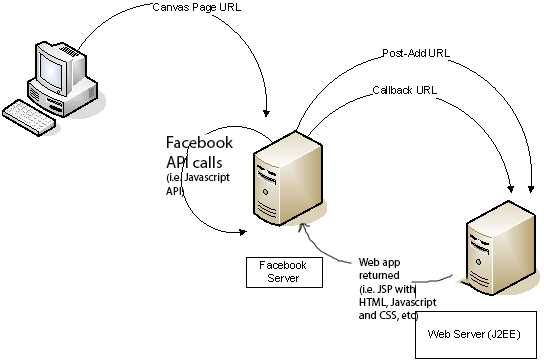
**6 Social Network Programming:**

-> Applications that users can "add" to their account (called as Canvas apps). Applications run inside of FB in a Canvas Page

-> FB runs your app in the "Canvas Page" inside an iframe using the Canvas URL you provided when registering your application

|  |
| --- |
| -> can turn these stories off by disabling the social discovery field in the [Developer app](http://developers.facebook.com/apps) in the ‘On Facebook’ tab under the ‘Canvas’ section. |

-> publish content to the stream using the [Feed Dialog](http://developers.facebook.com/docs/reference/dialogs/feed). -> On  [Application Settings Page](https://www.facebook.com/settings/?tab=privacy) users can control the maximum audience of each app ---for sharing. -> User needs to grant app **publish\_actions** permission before you can publish a user's scores and achievements. -> Must have 10 active monthly user to be discoverable on Facebook. -> Go to the Dev. App and visit the Contact Info section in the Advanced tab of the Dev App and click on the “Create Facebook Page” button to create a new Facebook Page. -> Facebook has a [Games and Apps Dashboard](https://www.facebook.com/games) where users can see requests, recommended Apps and Games, list of current apps

 << Callback URL over the years and Post-Add URL have taken on different names and today we only have one URL specified Canvas URL. Note that URL cannot forward to another URL via HTTP redirect responses, e.g. response code 301, but has to return the response directly.

-> Facebook apps are included by Facebook inside its page using **iFrames.**

iFrame: -> Standard part of HTML, embeds one webpage inside another -> You can view pages through a normal web-browser to test them -> Every Facebook page view goes via your webserver, so can debug-> Requires you to provide own webserver

-> There are two “modes” in which you can create a webpage to display as part of your app: 1) Default (not-authenticated)

2) Authenticated ::: FB.login(\*\*) = function call from Facebook Javascript SDK …. <fb:login-button scope="public\_profile,email" onlogin="checkLoginState();"> </fb:login-button> => This is special HTML Facebook markup that works in conjunction with having loaded Facebook Javascript SDK..

Steps for Using Facebook Login With the JavaScript SDK:::: **STEP 1**: Connect your app to Facebook, initialize

window.fbAsyncInit = function() { FB.init({ appId : '{your-app-id}', cookie : true, // enable cookies to allow the server to access the session xfbml : true, // parse social plugins on this page version : 'v2.2' // use version 2.2 });

**STEP 2:** Checking the login status to see if someone's already logged into your app.

FB.getLoginStatus(function(response) { statusChangeCallback(response); }); The response object that's provided to your callback contains a number of fields: { status: 'connected', authResponse: { accessToken: '...', expiresIn:'...', signedRequest:'...',userID:'...' } } …. Status : {connected, not\_authorized, unknown}.. document.getElementById('status').innerHTML = 'Please log ' + 'into this app.';

**STEP 3:** If they are not logged in, invoke the login dialog and ask for a set of data permissions.

**OPTION 1** prompt them with the Login dialog with FB.login() …FB.login(\*\*) = function call from Facebook Javascript SDK **OPTION 2** or show them the Login Button. <fb:login-button scope="public\_profile,email" onlogin="checkLoginState();"> </fb:login-button> 🡪 This is special HTML Facebook markup that works in conjunction with having loaded FB Javascript SDK.

**STEP 4**: Verify their identity(AUTOMATICALLY DONE by Javascrip SDK)…**STEP 5**: Store the resulting access token.

However, a common pattern is to take the access token and pass it back to a server and the server makes calls on behalf of a person. In order to get the token from the browser you can use theresponse object that's returned via FB.getLoginStatus(): -- code below is simply printing out to console ---but, you can access response.authResponse.accessToken and save in a variable that is passed back to your server web app. -🡪 FB.getLoginStatus(function(response) { if (response.status === 'connected') { console.log(response.authResponse.accessToken); } }); **STEP 6:** Make API calls-> Once the user is "logged in/ authenticated" you can make Facebook API call…This code INSIDE JSP/Client side returned code: FB.api('/me', function(response) { console.log(JSON.stringify(response)); }); **STEP 7:** Log out (OPTIONAL) :: FB.logout(function(response) { // user logged out });

MANUALLY USING JavaScript to ask for Authentication: User information is passed to your **Canvas URL using HTTP POST** within a single signed\_request parameter which contains a **base64url encoded JSON object**…**signed\_request** parameter contains ->**user**=>A JSON array containing the locale string, country string and the age object (min & max age range) for the current user…**algorithm**=> A JSON string containing the mechanism used to sign the request…**issued\_at** =>A JSON number containing the Unix timestamp when the request was signed…**user\_id**=>A JSON string containing the Facebook user identifier (UID) of the current user…**oauth\_token**=>A JSON string that you can pass to the Graph API..**expires=>**A JSON number containing the Unix timestamp when the oauth\_token expires….OAuth Dialog Box to authorize and give permissions to fields: <https://www.facebook.com/dialog/oauth?client_id=YOUR_APP_ID&redirect_uri=YOUR_CANVAS_PAGE>..For email+newsfeed: <https://www.facebook.com/dialog/oauth?client_id=YOUR_APP_ID&redirect_uri=YOUR_CANVAS_PAGE&scope=email,read_stream>…NOTE: The signed\_request parameter is utilized to share info between FB and app in a number of different scenarios:

-> A signed\_request is passed to Apps on Facebook.com when they are loaded into the Facebook environment-> A signed\_request is passed to any app that has registered an Deauthorized Callback in the Developer App whenever a given user removes the app using the App Dashboard-> A signed\_request is passed to apps that use the Registration Plugin whenever a user successfully registers with their app…..The signed\_request parameter is the concatenation of a HMAC SHA-256 signature string, a period (.), and a base64url encoded JSON object:::vlXgu64BQGFSQrY0ZcJBZASMvYvTHu9GQ0YM9rjPSso . eyJhbGdvcml0aG0iOiJITUFDLVNIQTI1NiIsIjAiOiJwYXlsb2FkIn0

it is important that you navigate the top window of the user's browser to the OAuth Dialog. Many apps do this by sending a script fragment to the user's browser setting the top.location.href property to the dialog URL.

$data = json\_decode(base64\_decode(strtr($payload, '-\_', '+/')), true); -> for Decode…Redirect to OAuth Dialog upon page load <script> var oauth\_url = 'https://www.facebook.com/dialog/oauth/'; oauth\_url += '?client\_id=YOUR\_APP\_ID'; oauth\_url += '&redirect\_uri=' + encodeURIComponent('https://apps.facebook.com/YOUR\_APP\_NAMESPACE/'); oauth\_url += '&scope=COMMA\_SEPARATED\_LIST\_OF\_PERMISSION\_NAMES'; window.top.location = oauth\_url; </script>

JavaScript SDK:: enables you to access all of the features of the Graph API and Dialogs via JavaScript. It provides a rich set of client-side functionality for authentication and rendering the XFBML versions of our Social Plugins….**Core Methods::** .init() .api() .ui()….**Facebook Login Methods::** .getLoginStatus() .login().logout() .getAuthResponse()… **Event Handling Methods::** .Event.subscribe() .Event.unsubscribe()…**App Events::** .AppEvents.LogEvent() .AppEvents.logPurchase().AppEvents.activateApp()

**XFBML Methods:** XFBML is a markup language like HTML, with special tags that are used to insert social plugins into HTML pages…. .XFBML.parse()

Tools::We provide a variety of development tools that you can use to develop, test and monitor your app.

Graph API Explorer...Access Token Tool...Sharing Debugger etc

-> Each permission has its own set of requirements and suggested use cases. All these permissions, except the default, public\_profile, require that you have Client OAuth Login enabled for your app on the Facebook Login tab of your app dashboard. Basic permissions, (public\_profile, user\_friends, and email) do not require Review, but all other permissions do.

Using Grph API to Retrieve User data:

GET /v2.11/{user-id} HTTP/1.1 Host: graph.facebook.com….GET /v2.11/{user-id}/friends HTTP/1.1 Host: graph.facebook.com

public\_profile (Default) consists of below details

-id -cover-name-first\_name-last\_name-age\_range-link-gender-locale-picture-timezone-updated\_time-verified....

**gender & locale can only be accessed if::**

-> The person queried is the person using the app.

-> The person queried is using the app, and is a friend of the person using the app.

-> The person queried is using the app, is not a friend of the person using the app, but the app includes either an app access token or an appsecret\_proof argument with the call.

**timezone & verified can only be accessed if::** The person queried is equal to the person making the request.

**user\_friends::**In order for a person to show up in one person's friend list, both people must have decided to share their list of friends with your app and not disabled that permission during login. Also both friends must have been asked for user\_friends during the login process…**publish\_actions::**Provides access to publish Posts, Open Graph actions, and other activity on behalf of a person using your app.Your app does not need to request the publish\_actions permission in order to use the Feed Dialog, the Requests Dialog or the Send Dialog…**user\_about\_me::**Provides access to a person's personal description (the 'About Me' section on their Profile) through the User object...**user\_birthday::**Access the date and month of a person's birthday. This may or may not include the person's year of birth, dependent upon their privacy settings and the access token being used to query this field.Please note most integrations will only need age\_range which comes as part of the public\_profile permission.

**user\_events::**Provides read-only access to the Events a person is hosting or has RSVP'd to…**user\_groups::**This permission was removed after Graph API v2.3…**user\_managed\_groups::**Enables your app to read the Groups a person is an admin of through the groups edge on the User object.This permission does not allow you to create groups on behalf of a person. It is not possible to create groups via the Graph API. This does not let you read the groups a user is just a member of..**read\_custom\_friendlists::**

Provides access to the names of custom lists a person has created to organize their friends. This is useful for rendering an audience selector when someone is publishing stories to Facebook from your app.This permission does not give access to a list of person's friends. If you want to access a person's friends who also use your app, you should use the user\_friends permission.

**Facebook: User does not allow authentication::** REQUEST sent:: <https://www.facebook.com/dialog/oauth?client_id=YOUR_APP_ID&redirect_uri=YOUR_CANVAS_PAGE&scope=email,read_stream>…The OAuth Dialog will redirect (via HTTP 302) the user's browser to the URL you passed in the redirect\_uri parameter with OUTPUT sent to your program from Facebook as..[**http://your\_canvas\_page/?error\_reason=user\_denied&error=access\_denied&error\_description=The+user+denied+your+request.**](http://your_canvas_page/?error_reason=user_denied&error=access_denied&error_description=The+user+denied+your+request.)**..**.How to access a Node in the graph…**OPTION 1** - with ID:: You can access the properties of an object by requesting https://graph.facebook.com/ID ..**OPTION 2:** Use name (for people and pages)

How to Access "Connections" (social data) about YOU (user = me)…Friends: <https://graph.facebook.com/me/friends?access_token=...>News feed: me/home..Profile feed (Wall): me/feed ..lly for all others

How to access "Connections" (social data) about a user of your program (user\_id)…At a high level, you need to get an access token (OAuth) for the Facebook user. After you obtain the access token for the user, you can perform authorized requests on behalf of that user by including the access token in your Graph API requests:.. IMPORTANT: 220439 must be substitued with user\_id you get from your signed\_request data that is automatically sent to your program (after authentication) and on each call to your program. See php code to get the user\_id.that supports authentication if not completed. Also, the access token you got from user authentication must be given. <https://graph.facebook.com/220439?access_token=...>

You can also publish to Facebook's Social Graph...Example of **Publishing:**:You can publish to the Facebook graph by issuing **HTTP POST** requests to the appropriate connection URLs, using an access token.

For example, you can post a new wall post on Arjun's wall by issuing a POST request to<https://graph.facebook.com/arjun/feed>..

GET /v2.11/{post-id} HTTP/1.1 Host: graph.facebook.com….in PHP using curl library::curl -F 'access\_token=...' \ -F 'message=Hello, Arjun. I like this new API.' \https://graph.facebook.com/arjun/feed

Most write operations require extended permissions for the active user. See the authentication guide for details on how you can request extended permissions from the user during the authentication step.

We support writing the following types of objects:

Method Description Arguments

/PROFILE\_ID/feed->Publish a new post on the given profile's feed/wall->message, pictur, link, name, caption,description, source

/OBJECT\_ID/comments->Comment on the given object (if it has a /comments connection)->message

/OBJECT\_ID/likes->Like the given object (if it has a /likes connection)->none

/PROFILE\_ID/notes->Publish a note on the given profile->message, subject

/PROFILE\_ID/links->Publish a link on the given profile->link, message, picture, name, caption, description

/PROFILE\_ID/events->Create an event->name, start\_time, end\_time

/EVENT\_ID/attending->RSVP "attending" to the given event->none

/EVENT\_ID/maybe->RSVP "maybe" to the given event->none

/EVENT\_ID/declined->RSVP "declined" to the given event->none

/PROFILE\_ID/albums->Create an album->name, message

/ALBUM\_ID/photos->Upload a photo to an album->message, source (multipart/form-data)

/PROFILE\_ID/checkins->Create a checkin at a location represented by a Page->coordinates, p

Graph API Overview::The Graph API is the primary way to get data out of, and put data into, Facebook's platform. It's a low-level HTTP-based API that you can use to programmatically query data, post new stories, manage ads, upload photos, and perform a variety of other tasks that an app might implement. **The Basics**::The Graph API is named after the idea of a 'social graph' - a representation of the information on Facebook composed of:-> nodes - basically "things" such as a User, a Photo, a Page, a Comment-> edges - the connections between those "things", such as a Page's Photos, or a Photo's Comments -> fields - info about those "things", such as a person's birthday, or the name of a Page…..The Graph API is HTTP-based, so it works with any language that has an HTTP library, such as cURL and urllib. We'll explain a bit more about what you can do with this in the section below, but it means you can also use the Graph API directly in your browser, for example a Graph API request is equivalent to::: GET graph.facebook.com \n(new line) /facebook/picture? \n redirect=false

Most Graph API requests require the use of access tokens, which your app can generate by implementing Facebook Login.

**How it's Structured::**We cover this fully in our Using Graph API guide, but in general you can read APIs by making HTTP GET requests to nodes or edges on those nodes…Almost all requests are passed to the API at graph.facebook.com - the single exception is video uploads which use graph-video.facebook.com…GET graph.facebook.com /{node-id}/{edge-name}.....POST graph.facebook.com /{node-id}/{edge-name}…Deleting via APIs is accomplished using HTTP DELETE requests (and updating via POST requests) to the same endpoints….This works for all versions, in this general form:: GET graph.facebook.com /n /vX.Y/{request-path}…Generate a Basic User Access Token -> through the Graph API Explorer:: 1) Click on the Get Token button in the top right of the Explorer. 2) Choose the option Get User Access Token.3) In the following dialog don't check any boxes, just click the blue Get Access Token button. 4) You'll see a Facebook Login Dialog, click OK to proceed. …this Graph API Explorer is the equivalent of the following Graph API 'read' request:: GET graph.facebook.com /me..../me is a special endpoint that translates to the user ID of the person whose access token is being used to make the request.

**Reading::** Returns a single user node..>Validation Rules:: Error-> Description..100-> Invalid parameter..200->Permissions error..210->User not visible..190->Invalid OAuth 2.0 Access Token..110->Invalid user id..452->Session key invalid. This could be because the session key has an incorrect format, or because the user has revoked this session..278->Reading advertisements requires an access token with the extended permission ads\_read..3001->Invalid query..294->Managing advertisements requires an access token with the extended permission for ads\_management…**Creating->**Validation Rules:: Error- >Description..200->Permissions error..100->Invalid parameter..240->Desktop applications cannot call this function for other users..368->The action attempted has been deemed abusive or is otherwise disallowed..195->The name you are trying to use is invalid..275->Cannot determine the target object for this request. Currently supported objects include ad account, business account and associated objects…210->User not visible…**Updating::**Example-> You can update a User by making a POST request to /{user\_id}..Validation Rules::Error->Description..200->Permissions error..100->Invalid parameter..240->Desktop applications cannot call this function for other users..368->The action attempted has been deemed abusive or is otherwise disallowed..195

->The name you are trying to use is invalid..275->Cannot determine the target object for this request. Currently supported objects include ad account, business account and associated objects…210->User not visible…**Deleting::**Delete a test user->Example..You can dissociate a User from a PageLabel by making a DELETE request to /{page\_label\_id}/users…Validation Rules::Error->Description..200->Permissions error..100->Invalid parameter..2903->Cannot delete this test account..240->Desktop applications cannot call this function for other users..2904->Cannot delete the OG Test User..

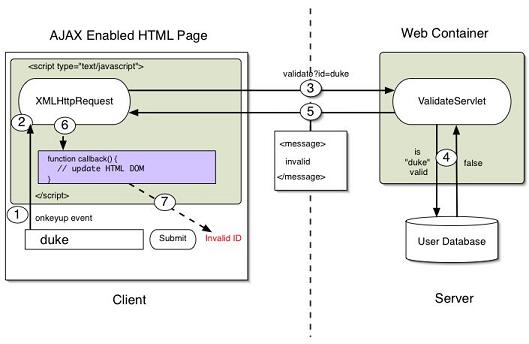
Graph API: GET -> /v2.1/me?fields=id,name,age dubmit button

**Facebook Example Code to Request Social Data = user's name**…General Steps ( index.jsp code )::STEP 1: Facebook Login process asking for permission to read/get info about user…STEP 2: Asking Facebook for information about user(me) via FP.api()..Later on reask for permission now to write/publish FP.api() --note to get permission go back through FB.login process..

AJAX- Asynchronous JavaScript And XML..uses asynchronous data transfer (HTTP requests) between browser and web server



What has changed is that the inclusion of support for the [XMLHttpRequest](http://algebra.sci.csueastbay.edu/~grewe/CS6320/Mat/AJAX/XMLHttpRequest.html) object

-> can get JSP tag libraries that let you implement AJAX by writing JSP tags only. AJAX uses JavaScript to send and receive data between a web browser and a web server…AJAX Applications Include:1)Real-Time Form Data Validation 2)Autocompletion 3)Master Details Operations 4)Sophisticated User Interface Controls 5)Refreshing Data on the Page 6)Server-side Notifications

SCENARIO: inside Javascript after you have gotten your userid make call to serverprogram via AJAX and pass as a parameter --

NOTE: in this scenario the USER DOES NOT HAVE OT CLICK ON ANYTHING…invoke via AJAX an XMLHttpRequest with URL = "http://myServerUrl//TweetServerProgram?userid=" + userid\_var…Facebook Analytics::Insights Dashboard OR by selecting your app in the Developer app and clicking the “insights" link…Insights -> Developer Site -> App Overview

JSON (JavaScript Object Notation) is a lightweight data-interchange format

**Sharing**::Enable people to post to Facebook from your app..**Scores & Achievements**::Both the Scores API and Achievements API are available only for apps that are categorized as Games and have already been granted the publish\_actions permission in the past. Please note that publish\_actions will not be granted for new apps with the sole purpose of accessing these APIs.

To retrieve Score and Achievement information about a player's friends, both the player and their friends need to grant the user\_friends permission..Creating, Reading and Deleting…You can create new acheivement types, read the existing list of types, and delete types by using the Graph API/{app-id}/achievements edge.

Note that instead of the non-secure URL of <http://puzzle.sci.csueastbay.edu:8080/whatever_your_deployed_webapp_called>

you use the Secure URL like <https://puzzle.sci.csueastbay.edu:8181/whatever_your_deployed_webapp_called>

FB.login(function() :: { FB.api('/me/feed', 'post', {message: 'Hello, world!'}); }, {scope: 'publish\_actions'});

SOME EXAMPLES ASKING FOR id, last\_name and email

FB.api('/me', {fields: 'id,last\_name,email'}, function(response) { \*\*\*whatever you want\*\*\*\* });

FB.api('/me?fields=id,last\_name,email', function(response) { \*\*\*whatever you want\*\*\*\* });