

Intel® HTML5 Development Environment

**Article - Native Application Facebook*
Integration**

V3.06 : 07.16.2013

Legal Information

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

A "Mission Critical Application" is any application in which failure of the Intel Product could result, directly or indirectly, in personal injury or death. SHOULD YOU PURCHASE OR USE INTEL'S PRODUCTS FOR ANY SUCH MISSION CRITICAL APPLICATION, YOU SHALL INDEMNIFY AND HOLD INTEL AND ITS SUBSIDIARIES, SUBCONTRACTORS AND AFFILIATES, AND THE DIRECTORS, OFFICERS, AND EMPLOYEES OF EACH, HARMLESS AGAINST ALL CLAIMS COSTS, DAMAGES, AND EXPENSES AND REASONABLE ATTORNEYS' FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PRODUCT LIABILITY, PERSONAL INJURY, OR DEATH ARISING IN ANY WAY OUT OF SUCH MISSION CRITICAL APPLICATION, WHETHER OR NOT INTEL OR ITS SUBCONTRACTOR WAS NEGLIGENT IN THE DESIGN, MANUFACTURE, OR WARNING OF THE INTEL PRODUCT OR ANY OF ITS PARTS.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined". Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request. Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order. Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or go to: <http://www.intel.com/design/literature.htm>

Intel and the Intel logo are trademarks of Intel Corporation in the U.S. and/or other countries.

*Other names and brands may be claimed as the property of others.

Microsoft, Windows, and the Windows logo are trademarks, or registered trademarks of Microsoft Corporation in the United States and/or other countries.

Copyright © 2012-2013, Intel Corporation. All rights reserved.

1.0 Purpose

The purpose of this documentation is to describe how to use the Facebook* JavaScript* API commands in a native application. This document assumes that the audience has a working knowledge of the Intel® XDK as well as the Facebook developer page (<http://developer.facebook.com>).

2.0 The JavaScript Bridge APIs

The Intel® HTML5 Development Environment JavaScript API includes a Facebook object that will trigger the native Facebook API within a native application built using the App Dev Center build system. For more information on the Facebook technology that is used in the build system, see Facebook's site here:

<http://developers.facebook.com/docs/guides/mobile/>

2.1 Login

This command is used to log the user into Facebook. When it is run, a dialog generated by Facebook should appear allowing the user to enter their credentials. Depending on the result of that interaction with Facebook, the application will receive a JavaScript event (*appMobi.facebook.login*) indicating whether the login was successful or not.

```
AppMobi.facebook.login(permissions)
```

Log in to Facebook with the requested permissions. If none are specified, defaults will be used. For more information about Facebook login permissions, see Facebook's site here:

<http://developers.facebook.com/docs/authentication/permissions/>

Here is a block of sample code to illustrate how the login process works:

```
document.addEventListener("appMobi.facebook.login", function(e) {
    if (e.success == true)
    { console.log("Facebook Log in Successful"); }
    else
    { console.log("Unsuccessful Login"); }
}, false);
AppMobi.facebook.login("publish stream,publish actions,offline access");
```

2.2 Logout

This command is used to log the user out of Facebook.

```
AppMobi.facebook.logout()
```

Here is a block of sample code to illustrate how the logout process works:

```
document.addEventListener("appMobi.facebook.logout",function(e) {
    if (e.success == true)
    { console.log("Logged out of Facebook"); }
    else
    { console.log("Unsuccessful Logout"); }
},false);
AppMobi.facebook.logout();
```

2.3 Request with REST API

This API command is used to make requests against the older Facebook REST API. This API is being deprecated by Facebook, and may not be available in the future. For more information on Facebook's REST API see the Facebook documentation here:

<http://developers.facebook.com/docs/reference/rest/>

```
AppMobi.facebook.requestWithRestAPI(command, method, parameters)
```

2.4 Request with Graph API

This API command is used to make requests from the newer Facebook Graph API. For more information on this particular API, see Facebook's documentation here:

<http://developers.facebook.com/docs/reference/api/>

```
AppMobi.facebook.requestWithGraphAPI(path, method, parameters)
```

Here is a block of sample code that illustrates how one might use the Facebook Graph API to get the pictures of all the friends of a particular Facebook user. Alternatively, use Facebook's "me" placeholder to refer to the Facebook ID of the user currently logged in.

```
var facebookUserID = "me"; //me = the user currently logged into Facebook
document.addEventListener("appMobi.facebook.request.response",function(e) {
    console.log("Facebook User Friends Data Returned");
    if (e.success == true) {
        var data = e.data.data;
        var outHTML = "";

        for (var r=0; r< data.length; r++) {
            outHTML += "<img src='http://graph.facebook.com/" + data[r]["id"]
+ "/picture' info='" + data[r]["name"] + "' />";
        }
        document.getElementsByTagName("body")[0].innerHTML = outHTML;

        document.removeEventListener("appMobi.facebook.request.response");
    }
},false);
AppMobi.facebook.requestWithGraphAPI(facebookUserID + "/friends","GET","");
```

Test out a path/method/parameters with the Facebook Graph Explorer:

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

2.5 Show Application Request Dialog

This API command is used to bring up Facebook's application request dialog. The Request Dialog sends a request from one user (the sender) to one or more users (the recipients). The Request Dialog can be used to send a request directly from one user to another or display a Multi Friend Selector Dialog, allowing the sending user to select multiple recipient users. Find more information on Facebook's request dialog here:

<http://developers.facebook.com/docs/reference/dialogs/requests/>

```
AppMobi.facebook.showAppRequestDialog(parameters);
```

Here's an example code block:

```
document.addEventListener("appMobi.facebook.dialog.complete",function(e) {
    console.log("Permissions Request Returned");
    if (e.success == true) {
        console.log("News feed updated successfully");
    } else { console.log("permissions request failed"); }
},false);
var objParameters = {"to":"USER_ID_HERE","message":"My Awesome
Message","title":"A title for this dialog would go here"}
AppMobi.facebook.showAppRequestDialog(objParameters);
```

2.6 Show News Feed Dialog

This API command is used to prompt the user to publish an individual story to a profile's feed. Find more information about the feed dialog here:

<http://developers.facebook.com/docs/reference/dialogs/feed/>

```
AppMobi.facebook.showNewsFeedDialog(parameters);
```

Here's a sample code block:

```
//This allows you to post to your Facebook Wall
document.addEventListener("appMobi.facebook.dialog.complete",function(e) {
    console.log("News Feed Event Returned");
    if (e.success == true) {
        console.log("News feed updated successfully");
    }
},false);
var objParameters = {
    "picture":"http://fbrell.com/f8.jpg",
    "name":"Facebook Dialog",
    "caption":"This is my caption",
    "description":"Using Dialogs to interact with users.",
```

```
"link":"http://www.appmobi.com/documentation"  
}  
AppMobi.facebook.showNewsFeedDialog(objParameters);
```

2.7 Enable Frictionless Requests

This command ensures that news feed requests are “frictionless” in that they enable users to send requests to specific friends without having to click on a pop-up confirmation dialog. This command option is for the Apple iOS* platform only since Google Android* requests are automatically set to be “frictionless”.

```
AppMobi.facebook.enableFrictionlessRequests();
```

3.0 Facebook Developer Page Setup

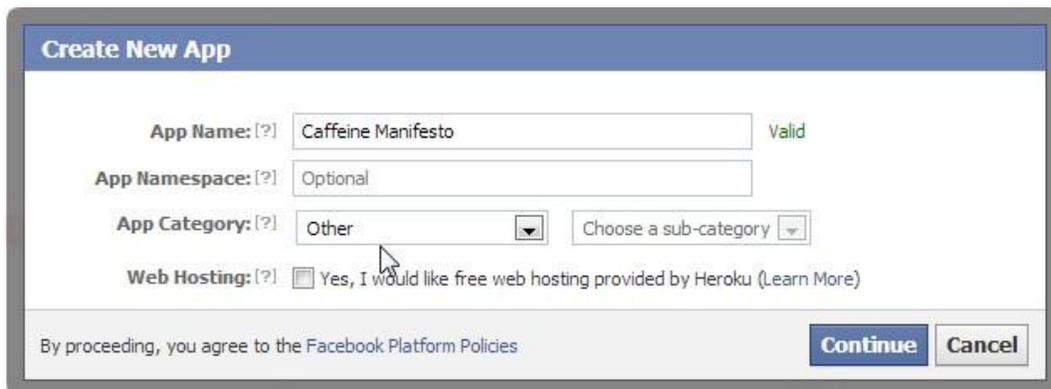
Adding JavaScript API commands to an Intel® XDK application is one part of setting up Facebook integration. Some integration between the App Dev Center and the Facebook Developer site must be done as well.

3.1 Testing in the Intel® XDK

Your Facebook-enabled application may be tested through the Intel XDK. The JavaScript API commands will work to simulate the same experiences that will happen on a mobile device. To test your application within the Intel XDK, a few settings must be arranged within the Facebook developer site as well as within App Dev Center.

3.1.1 Create a Facebook Application

Navigate to the Facebook developer site (<http://developer.facebook.com>) and set up a new Facebook application. Set up the basic information including the display name and namespace.



The screenshot shows the 'Create New App' form on the Facebook Developer site. The form has a blue header with the text 'Create New App'. Below the header, there are four main sections:

- App Name:** A text input field containing 'Caffeine Manifesto' with a green 'Valid' status indicator to its right.
- App Namespace:** A text input field containing 'Optional'.
- App Category:** A dropdown menu with 'Other' selected, and a secondary dropdown menu labeled 'Choose a sub-category'.
- Web Hosting:** A checkbox labeled 'Yes, I would like free web hosting provided by Heroku (Learn More)'. The checkbox is currently unchecked.

At the bottom of the form, there is a grey bar containing the text 'By proceeding, you agree to the Facebook Platform Policies' and two buttons: 'Continue' (in blue) and 'Cancel' (in grey).

3.1.2 Set the Mobile Web URL

After setting up the Facebook application, edit the application and set the *Mobile Web URL* to:

<http://html5tools-software.intel.com>

Select how your app integrates with Facebook

Mobile Web ✕

Mobile Site URL: [?]

Accept Mobile Web Payments: [?] Enabled Disabled

If your app accepts payments through any non-iOS approved service it will be restricted on iOS.
[Learn more.](#)

3.1.3 Copy the Facebook App ID

Once the *Mobile Web* URL is set, copy down the Facebook Application's *App ID* for the next step. The *App ID* is a unique 15-digit number generated by Facebook to identify an application running on their system.

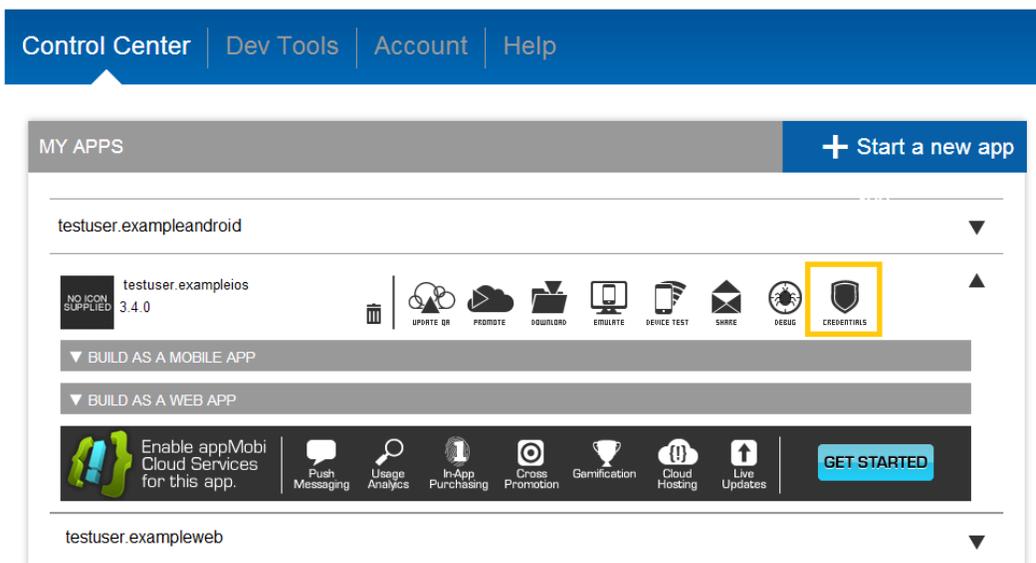


3.1.4 Give the App Dev Center the Facebook App ID

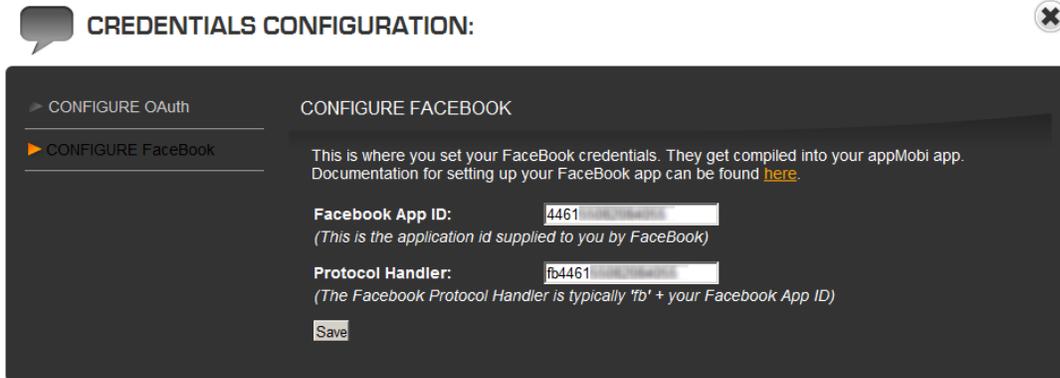
Log into your App Dev Center account at:

<http://appcenter.html5tools-software.intel.com>

Find the application to test Facebook integration with in the Control Center, and click on the *Credentials* icon.



Select the *Configure Facebook* option, and then paste the Facebook *App ID* copied from the previous step into the appropriate edit field. Enter the same *App ID* with the letters 'fb' in front of it for the *Protocol Handler* field, unless the particular application uses a different protocol handler for some reason.



3.2 Google Android* Applications

In order to get a native Google Android application to access Facebook, follow these instructions.

3.2.1 Create a Facebook Application

Follow the instructions in section 3.1.1 to create a Facebook application if one has not already been created.

3.2.2 Copy the Facebook App ID to App Dev Center

Follow the instructions in sections 3.1.3 and 3.1.4 to get the unique *App ID* for the Facebook Application and pass it to App Dev Center.

3.2.3 Build for Android

Build the Android application. For more information on building an Android application in App Dev Center, check out this document:

http://www.html5dev-software.intel.com/documentation/index.php?DOC=TUTORIAL_BUILD_BINARY_ANDROID

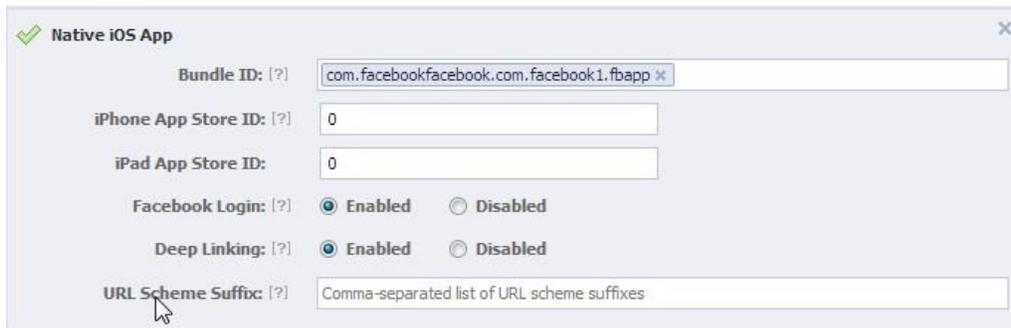
3.2.4 Copy the Android Key Hash from App Dev Center

During step 4 of the Android build process, copy down the Facebook Application Signature from App Dev Center.



3.2.5 Give Facebook the Application Signature

Log back into the Facebook Developer page, and open up the settings for a *Native Android App*. Enable the setting for *Configured for Android SSO*: and paste the application signature taken from the App Dev Center build process into the field marked *Android Key Hash*.



3.1 Apple iOS* Applications

There are two types of builds for an Apple iOS application. An adhoc build that includes commands to the Facebook application only requires the steps required to test in the Intel XDK outlined in section 3.1. However, once an iOS application is available in the application store, a few settings must be tweaked in the Facebook Developer site to complete the integration.

For more information on building an iOS application in App Dev Center, check out this document: http://www.html5dev-software.intel.com/documentation/index.php?DOC=TUTORIAL_BUILD_BINARY_IOS

3.1.1 Get the iOS Application ID

During an iOS build, a *bundle ID* is generated and passed to the Apple Developer Site. Copy that unique identifier down during the production build.

5c **OBTAIN APP I.D.**
Paste the bundle I.D. (below) in the "Bundle identifier" field on the Apple Developer Site.

com.facebookfacebook.com.facebook1.fbapp

Ready to obtain your App I.D. from the [Apple Developer's Site](#)? Click to their site tab, complete the steps outlined in our [screenshot walkthrough](#) and return to this page to continue.

Have you completed this step? Click the button on right to alert our system.
This step must be completed correctly or your build will fail.

Go To Next Step

3.1.2 Give Facebook the Application ID

In the Facebook Developer site, open up the settings for *Native iOS Application*. Paste the *bundle ID* into the field for *iOS Bundle ID*.



Native iOS App

Bundle ID: [?] com.facebookfacebook.com.facebook1.fbapp x

iPhone App Store ID: [?] 0

iPad App Store ID: [?] 0

Facebook Login: [?] Enabled Disabled

Deep Linking: [?] Enabled Disabled

URL Scheme Suffix: [?] Comma-separated list of URL scheme suffixes

Next, copy down the store ID Apple assigns to the application once it has been submitted to the app store, and drop them into the appropriate *App Store ID* fields. Finally enable both *Configured for iOS SSO* and *iOS Native Deep Linking*.

4.0 Conclusion

This tutorial should help developers get started using both the Facebook Graph API as well as the legacy REST API. For more information on App Dev Center find a document on how it works here:

http://www.html5dev-software.intel.com/documentation/index.php?DOC=ARTICLE_APPDEVCENTER