SL4 (In Browser) Sample Notes

In order to login to Facebook and use the Facebook graph data you need to acquire an Access Token. This sample app does this with minimal effort on your part. Customizing the first 2 lines of the MainPage class (lines 21 and 22) in the MainPage.xaml.cs is all that you need to do to see the application in action (you do need to create a Facebook application and have your Application ID handy). The Application ID goes into the apiKey variable (line 21). The Permissions that you want to get from the user goes into the “permission” string variable and the values are separated by commas (line 22).

This example is hosted on a web site. There are a couple pages that have to be served up from a webserver. This gets around a few issues. These pages aren’t ASPX pages, so you could host them anywhere you can host a Silverlight application (which is virtually any web server that knows about the XAP mimetype); you can move these files elsewhere (and customize lines 24 and 25), but they need to be hosted on the same domain.

The failedLogin method (lines 32-39) is where you should handle a login failure (probably caused by the user canceling).

The loginSucceeded method (lines 41-59) is the routine that is called when the login process succeeds. The original demo asks for permissions to the user’s basic information and the results bound to a listbox control.

## Flow

This app is a scriptable (JavaScript accessible) application. When the user clicks the Login button, the button is disabled. Next, the application constucts a Facebook OAuth Login URI using a “popup” display. In this URI , Facebook is told which permissions are needed and which pages to land on, and when the process is complete or has failed (user aborts). Finally Javascript is called to popup a new window with the login. At this point the user will interact with the browser control (you have no control at this point). Once the user completes or cancels the login process one of the 2 landing pages will be called and these pages will pass the QueryString (containing the access token) or a message telling the application that the user canceled login via the ScriptableMembers we have exposed to JavaScript.

When the demo app gets a QueryString it looks for the Access Token and when it has one it will update a few variables, set up the fbApp variable (A FacebookApp instance from the SDK), and then call the loginSucceeded method. If it gets a failure message (or a “user canceled” message) then we it will call the loginFailed method. Additionally if the user gets to the success page but there is no Access Token then the loginFailed method will be called.

For this sample when the loginSucceeded method is called the button is hidden and the listbox is shown. The FacebookApp’s ApiAsync method is then called to get the user’s “me” data (and when this completes we set this IDictionary<string, object> as the item source for the listbox using the UI Thread –aka Dispatcher).

If loginFailed is called then the Login button is re-enabled and shown, and the listbox is hidden.