Facebook Developer Toolkit v2.1

Developed for Microsoft By Clarity Consulting Inc. - www.claritycon.com

Table Of Contents

1 Introduction

This document outlines the components and controls that are available to help simplify development with the Facebook API. The main entry point is the API class. This class wraps the Facebook API and provides an easy to use interface for calling the different methods currently available in the Facebook API. In addition, some Windows Forms and web controls are included which will provide a quick way to start leveraging Facebook data in your application. We've also provided you with some cool, fun samples. Additionally, we've provided all the source code for the API, components, controls and samples for you to explore.

This toolkit is meant to help leverage the facebook platform from .net, that said the main place to start is not here with the toolkit, but with facebook's platform documentation. There is a wiki with most of what you need to know before writing your first application. Here are some important links to use as a starting point.

http://wiki.developers.facebook.com/index.php/Anatomy_of_a_Facebook_App

http://wiki.developers.facebook.com/index.php/Platform_Core_Components

http://wiki.developers.facebook.com/index.php/How-to_Guides

http://wiki.developers.facebook.com/index.php/Creating_a_Platform_Application

2 Pre-requisites

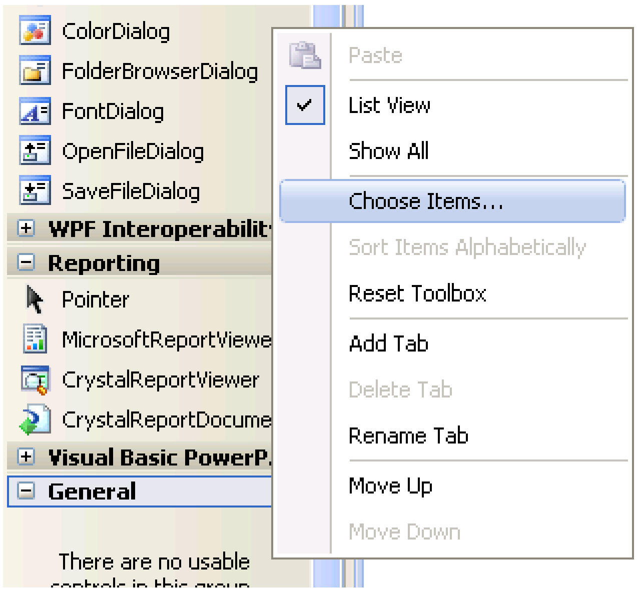
First, you download one of the Visual Studio Express products . To develop a desktop application, you can use Visual C# 2008 Express Edition (or later) or Visual Basic 2008 Express Edition (or later). If you want to build a web application, you can use Visual Web Developer 2008 Express Edition (or later). You may also use any full version of the express products.

Next, in order to use the Facebook API, you must first add the Developer application to your Facebook account at http://www.facebook.com/developers/. This will allow you to manage all your applications from one place.

To create a new application, go into the Developer application, click the "Set Up New Application" button, and fill in the required fields. After registering your application, you will receive an API key and secret. These are critical for using the Facebook API and you should keep these readily available.

3 Toolbox Configuration

If you are developing a **Winform** application you will might want to add the FacebookService component to your toolbox in Visual Studio (and the rest of this documentation assumes you have done so). To do this, right-click in the area of the toolbox where you want the controls to go, and select "Choose Items" from the pop-up menu.



In the window that appears, click the "Browse…" button, locate facebook.dll, and open it. Click "OK", and the controls should appear in your toolbox. If you want to add the Facebook controls for your application as well, repeat the above instructions for the appropriate assembly (facebook.desktop.dll for desktop applications, or facebook.web.dll for web applications).

4 Facebook Toolkit Overview

All of the sections below assume that the developer has added the Developer application to their account and received an API key and secret for a new application from Facebook. Please see http://wiki.developers.facebook.com/index.php/Creating_your_first_application for more instructions on how to do this.

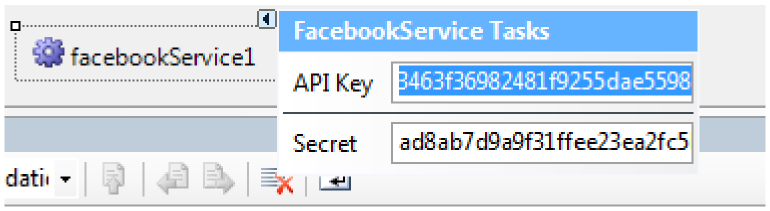
4.1 Facebook API Overview

The Facebook API is a REST-like interface allowing applications to post a specific HTTP address and receive structured XML representing the requested Facebook data. The current version of the API contains many different ways to interact with Facebook, including getting a user's information, sending messages to users in various ways, and setting application-specific information on certain parts of the user's profile.

For more information on the available functions, see the API documentation at http://wiki.developers.facebook.com/index.php/API

4.2 Facebook Winform Development

To start utilizing the Facebook API, you will need to add the FacebookService component to your project. To do this just drag an instance of FacebookService from the toolbox onto the component tray and enter your API Key and Secret. If you don't see FacebookService in your toolbox, see the Configuration section.



API Key: 3463f36982481f9255dae5598

Secret: ad8ab7d9a9f31ffee23ea2fc5

The FacebookService component contains methods that wrap all of the methods of the Facebook API. The following provides an example for getting all the friends of the logged in user. This method returns a generic collection of user objects. The user object is a strongly typed representation of a Facebook User profile.

Visual C#

```
using System.Collections.Generic;
using System.Collections.ObjectModel;
using facebook.Schema;
…
IList<user> friends = facebookService1.friends.getUserObjects();
```

Visual Basic

```
Imports System.Collections.Generic
Imports System.Collections.ObjectModel
Imports facebook.Schema
…
 Dim friends As IList(Of user) = facebookService1.friends.getUserObjects()
```

4.3 Facebook Web Development

There are several different approaches that you can take for developing a facebook web application. We will cover several of those approaches in the upcoming sections. But, the first key is deciding what type of application you want.

1. Canvas Application – These are web applications that run completely integrated with the facebook platform. You application will feel like it is actually part of facebook. Users will use your application without ever leaving facebook. There are two approaches to writing facebook canvas applications, you can read this to help you decide. http://wiki.developers.facebook.com/index.php/Choosing_between_an_FBML_or_IFrame_Application

   a. FBML- Facebook Markup Language – This is a model where you provide a url for facebook to call and return formatted FBML that facebook will then render on your behalf. This is the easiest way to get a very integrated experience and to take advantage of a lot of the facebook user interface items that your users are used to using.

    b. IFrame – This is a model where facebook will host an iframe pointing at pages that you host on your own servers. Facebook will provide you some context on who the facebook user is. This is a good model if you really want to write your app similar to other web apps you might have written. You do have the ability to still leverage some FBML using XFBML http://wiki.developers.facebook.com/index.php/XFBML.

2. Facebook Connect – If you don't want to have a canvas application, but you want to leverage facebook data or provide other integration with a user's facebook account from a stand-alone web application you should look at facebook connect. http://wiki.developers.facebook.com/index.php/Facebook_Connect. Facebook connect provides a mechanism that you can leverage facebook authentication within your site and also leverage new forms of distributing content from your site to your user's friends and facebook profile.

3. Standalone Web Application – If you have a web application and you just want to use the facebook apis to provide some integration there is support. It is recommended that you strongly consider facebook connect instead, but a stand alone web application not using facebook is supported.

4.4 Facebook Canvas Development

In addition to standalone web and desktop applications, Facebook also supports "Canvas" applications. Canvas applications are integrated directly into the user experience on the facebook.com website. Canvas applications have the ability to display information on a user's profile, publish to a user's mini-feeds, and spread themselves organically through Facebook's social network. For more information about building a Facebook Canvas application, read Anatomy of a Facebook Application at http://developers.facebook.com/anatomy.php.

Canvas applications can be one of two types. FBML Canvas applications leverage Facebook Markup Language (FBML) to provide a consistent integrated user experience for Facebook users. FBML canvas applications essentially are constructed by the developer building an FBML "web service" that can be called by the Facebook Server and return some FBML. The Facebook server will then translate that FBML into HTML to display on a Facebook-hosted page. The second type of Canvas application is the IFrame Canvas application. In this type of canvas application, the Facebook web page will simply show an IFrame linked to application content hosted by the developer's server. This type of canvas application can not leverage FBML functionality, but does have the advantage of being a much more typical web development experience.

The Facebook Developer Toolkit provides infrastructure for building both types of canvas applications. The facebook.web assembly contains CanvasFBMLBasePage, CanvasIFrameBasePage, CanvasFBMLMasterPage, and CanvasIFrameMasterPage. You can use the BasePages if you want each of your pages to inherit from that class, or you can use the MasterPages if you would rather create one master page which hosts all of your other pages. No matter which method you choose, these pages take care of all of the plumbing needed to successfully host a canvas application on the Facebook site. See

Sections 5.3 and 5.4 below for a detailed walkthrough. In addition, you can see samples of both types of Canvas applications at the below urls. The FBML sample is a reproduction of Facebook's Smiley application. This shows how to leverage all the various facebook integration points. We also put together a document that walks through and describes all of the code.
http://apps.facebook.com/fbmlcanvassamplevb/

http://apps.facebook.com/sampleiframeapp/

4.5 Facebook Connect

We are in the process of building out some more specific samples for supporting facebook connect. In the meantime, Bill Conrad has written a couple of blogs that provide everything you need to get started.

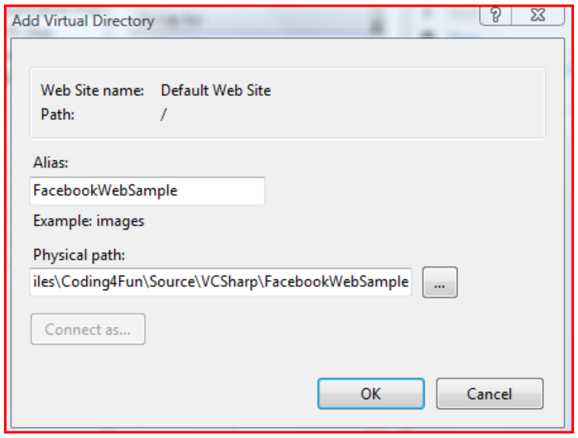
http://devtacular.com/articles/bkonrad/how-to-integrate-with-facebook-connect

http://devtacular.com/articles/bkonrad/how-to-retrieve-user-data-from-facebook-connect-in-aspnet

4.6 Standalone Web Application

The authentication process for a Web Application is more involved than for a Windows application. When the developer account is setup, the developer must choose whether their application is a Web application or Desktop application. In the case of a web application, the developer must specify a callback URL. This is the URL that users of the web application will be redirected to after logging in on the Facebook-hosted login page. For example, say you have a web application set up to return to http://localhost/FacebookWebSample/default.aspx. To run and debug the FacebookWebSample project, you must configure your Visual Studio Web Project to start your web application using this address. Because of this requirement, the simplest way to configure to run the WebSample is using IIS. (Since it runs on port 80 and handles addresses like the above)

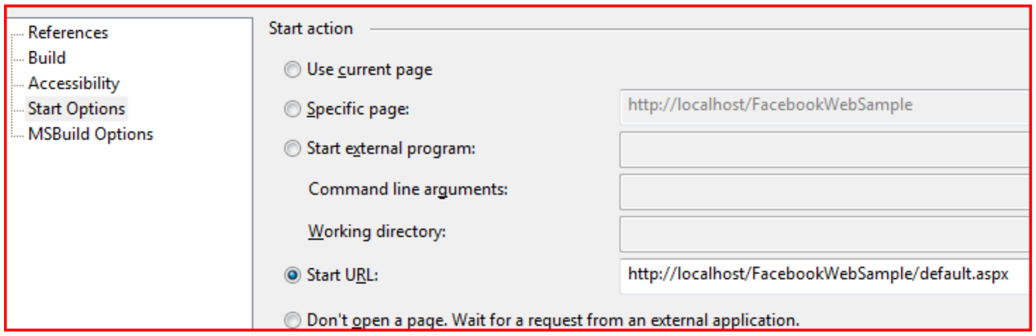
1. Create an IIS Virtual Directory called FacebookWebSample pointed at the location of your FacebookWebSample.



Web Site name: Default Web Site
Path: /
Alias: FacebookWebSample
Example: images
Physical path: iles\Coding4Fun\Source\VCSharp\FacebookWebSample

2. Set the Web Project to start using this website.
   a. Right Click FacebookWebSample in Visual Studio – Select Property Pages

b. Select Start Options
c. Click Start URL radio button – Enter http://localhost/FacebookWebSample/default.aspx



Use current page

Specific page: http://localhost/FacebookWebSample

Start external program:

Command line arguments:

Working directory:

[✓ Start URL:] http://localhost/FacebookWebSample/default.aspx

Don't open a page. Wait for a request from an external application.

3. Write code to handle 3 states.
   a. If we already have a valid session, use it.
   b. If we don't have a valid session but have an auth_token, start a session on behalf of the user.
   c. If we don't have an auth_token, redirect to the Facebook hosted login page.

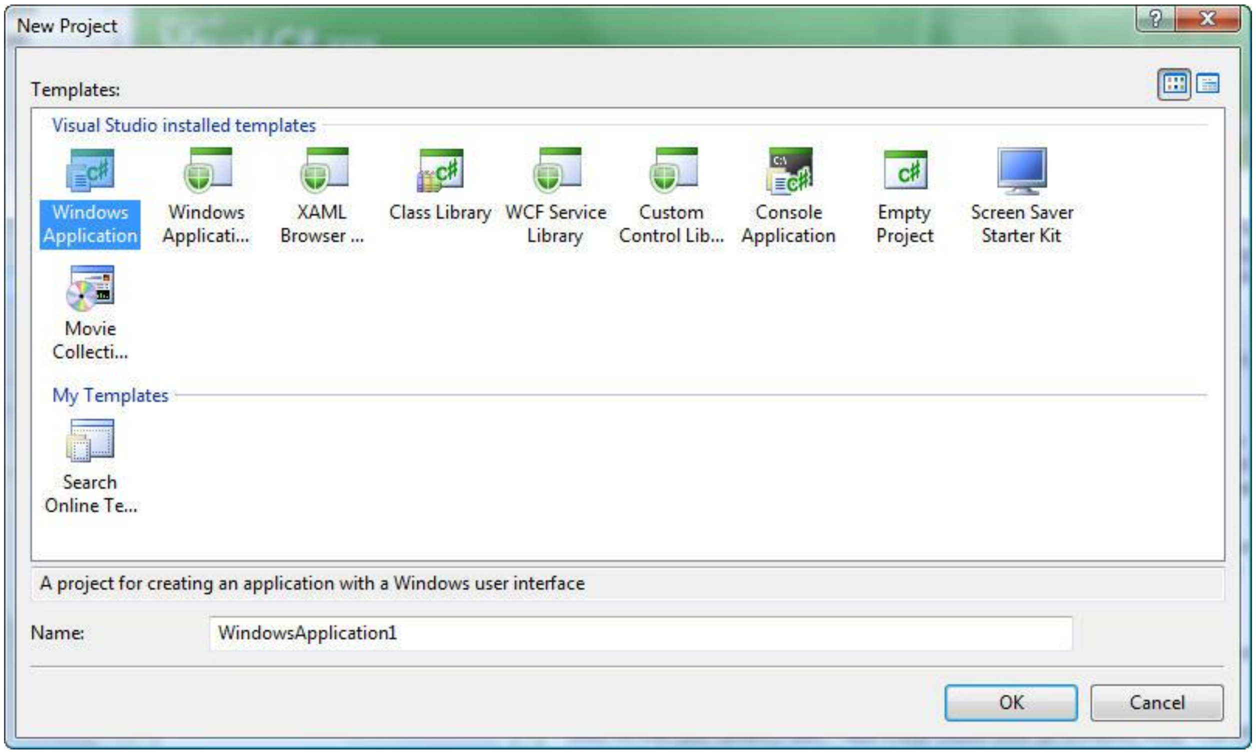
See the Code Snippets section for more details.

5 Leveraging the Facebook API in 5 minutes

5.1 Winform Development

- Satisfy Pre-requisites described above
- Start a new Windows application project
  - File – New – Project
  - Windows Application

Visual C#


New Project

Templates:

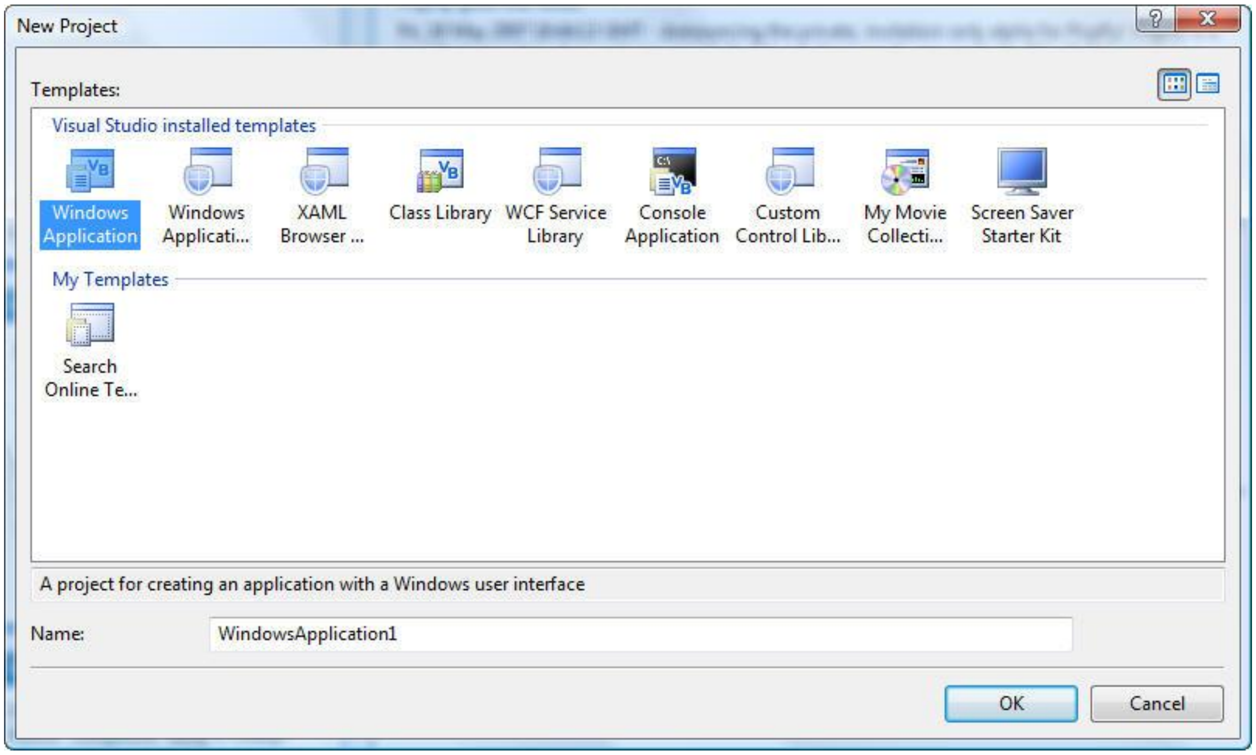
Visual Studio installed templates

My Templates

A project for creating an application with a Windows user interface

Name: WindowsApplication1

Visual Basic


New Project

Templates:

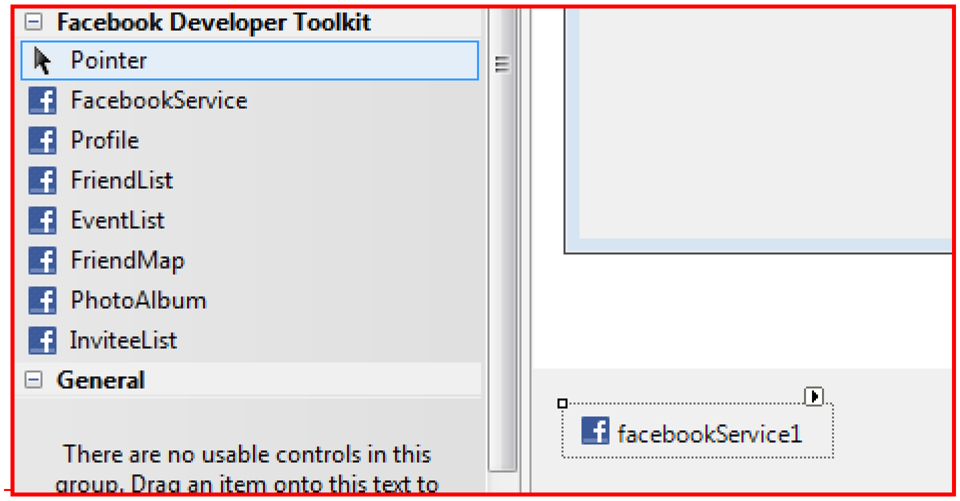
Visual Studio installed templates

My Templates

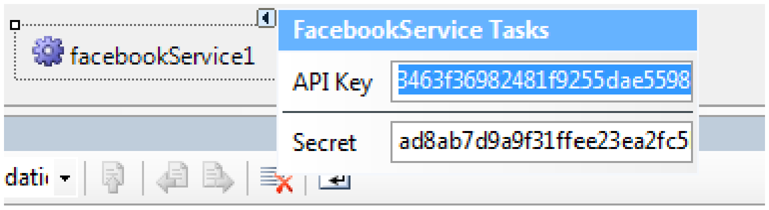
A project for creating an application with a Windows user interface

Name: WindowsApplication1

- Add the FacebookService component from the toolbox to the Form's component tray. If you do not see the component in your toolbox, see the Configuration section to find out how to add it and the other Toolkit controls.

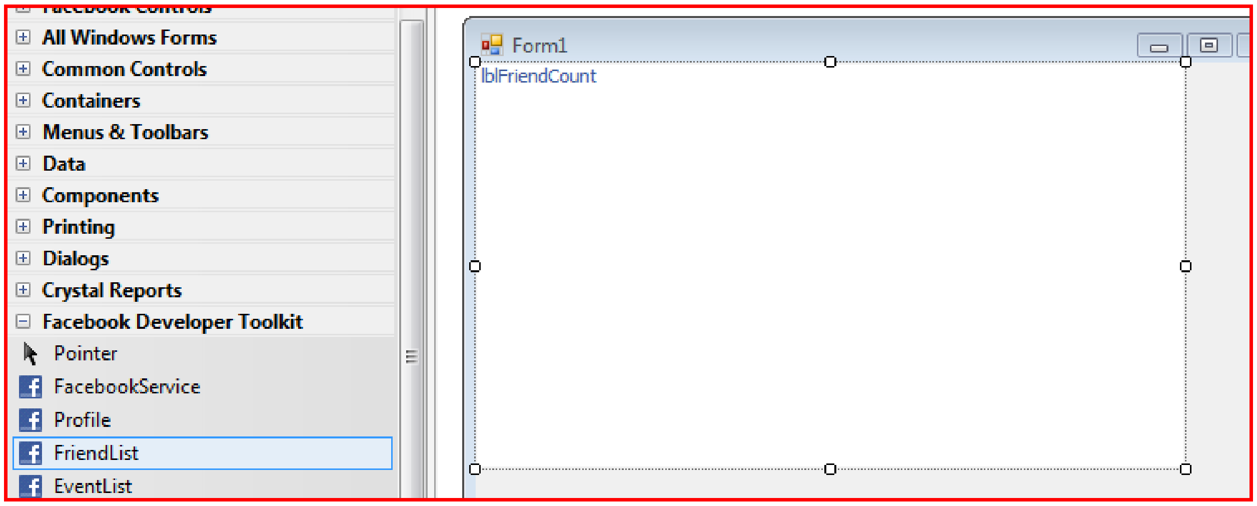


- Provide the API Key and Secret values

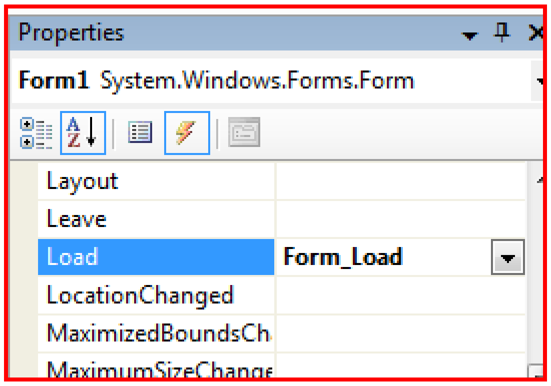


API Key 3463f36982481f9255dae5598
Secret ad8ab7d9a9f31ffee23ea2fc5

- Drag a FriendList from the toolbox onto the design surface for the form



- Hook up the form Load Event
    - Find Load in the Event list in the property window, type Form_Load. Press Enter.


Properties

Form1 System.Windows.Forms.Form

Layout	
Leave	
Load	**Form_Load**
LocationChanged	
MaximizedBoundsCh	
MaximumSizeChange	

- In the generated Form_Load method, set the Friends property of the FriendList control to the Collection returned by calling the GetFriends method of the FacebookService. As shown here:

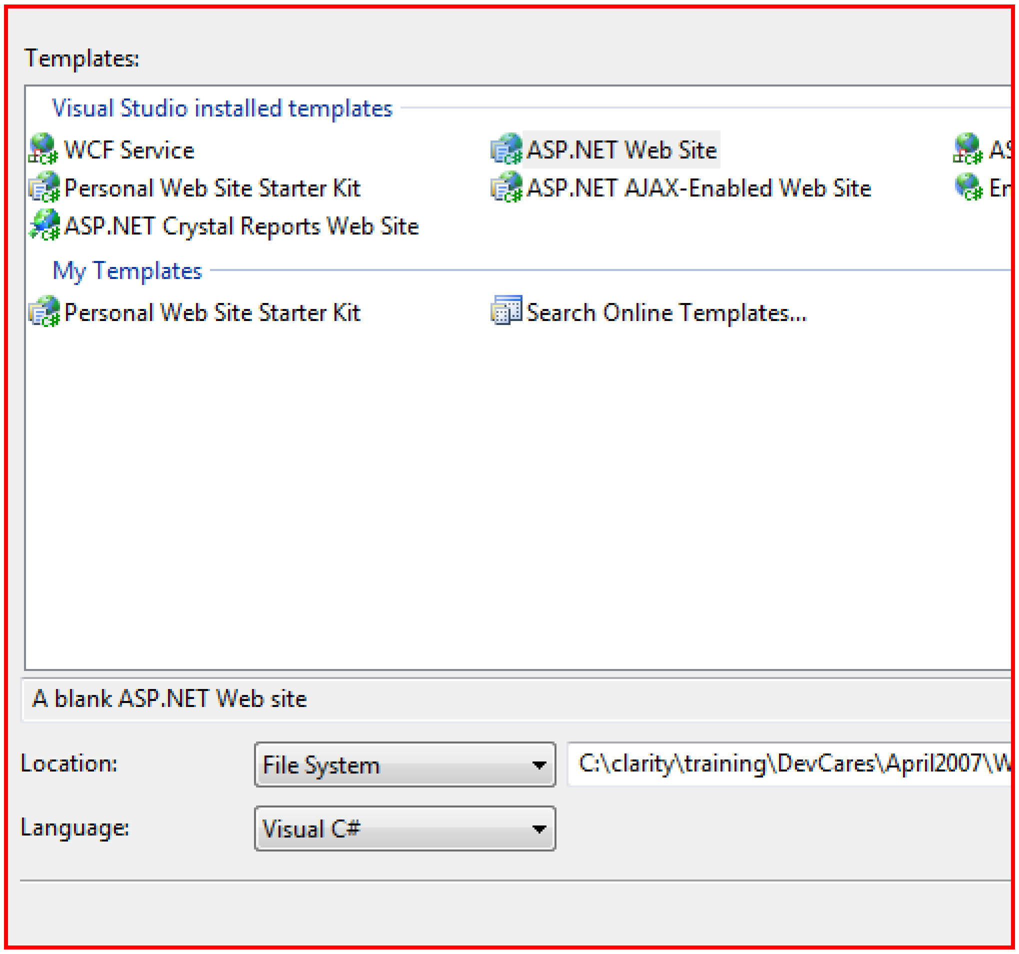
  Visual C#
  ```
  private void Form_Load(object sender, EventArgs e)
  {
      friendList1.Friends = facebookService1.friends.getUserObjects();
  }
  ```
  Visual Basic

  ```
  Private Overloads Sub OnLoad()
        friendList1.Friends = FacebookService1.friends.getUserObjects()
  End Sub
  ```
- Press F5 to run the application

5.2 Creating a stand alone web application

- Satisfy Pre-requisites described above
- In Visual Web Developer 2005 Express edition
  - File – New – Web Site

Visual C#



Templates:

Visual Studio installed templates

WCF Service

ASP.NET Web Site

AS

Personal Web Site Starter Kit

ASP.NET AJAX-Enabled Web Site

En

ASP.NET Crystal Reports Web Site

My Templates

Personal Web Site Starter Kit

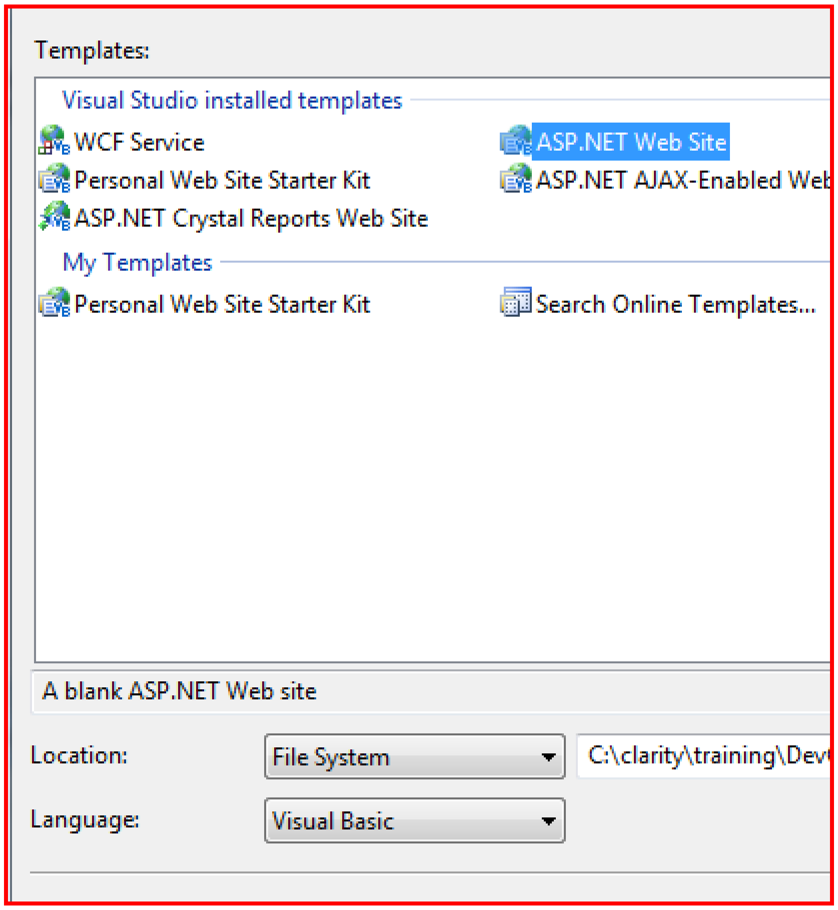
Search Online Templates...

A blank ASP.NET Web site

Location: File System C:\clarity\training\DevCares\April2007\W

Language: Visual C#

Visual Basic


Templates:

Visual Studio installed templates

WCF Service | ASP.NET Web Site
Personal Web Site Starter Kit | ASP.NET AJAX-Enabled Wel
ASP.NET Crystal Reports Web Site

My Templates

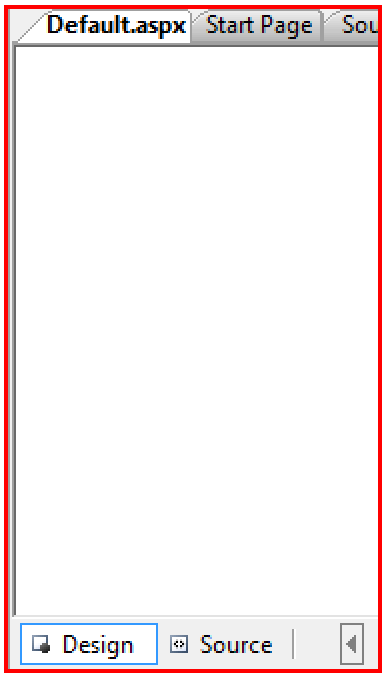
Personal Web Site Starter Kit | Search Online Templates...

A blank ASP.NET Web site

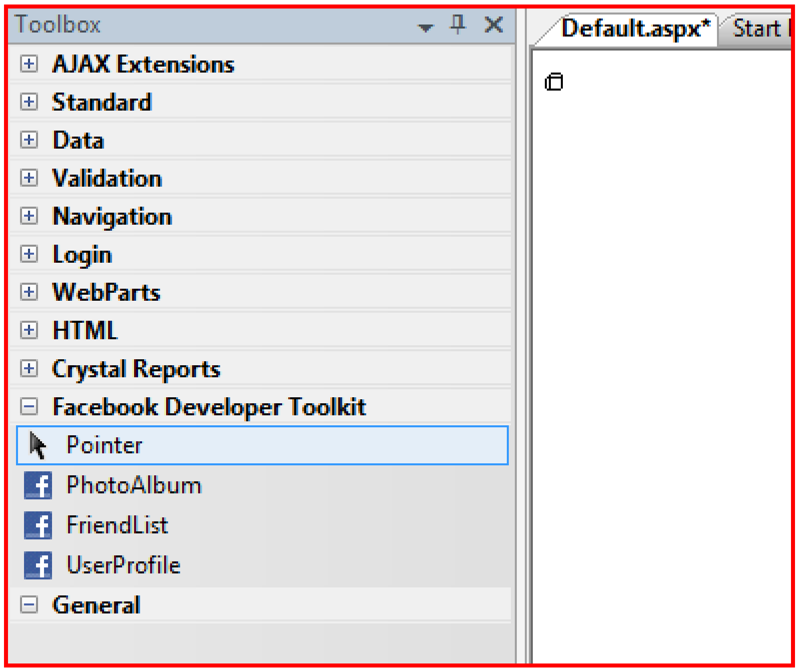
Location: File System C:\clarity\training\Dev

Language: Visual Basic

- Make sure that the AutoEventWireup attribute in the Page directive at the top of your markup is set to “true”.
- Switch to Design View of Default.aspx



- On Default.aspx, drag a FriendList from the toolbox. If you do not see the control in your toolbox, see the [Configuration]() section to find out how to add it and the other Toolkit controls.



- Add the following code to the Default.aspx
- Configure the API key and secret

Visual C#

```
public partial class _Default : Page
{
    facebook.Components.FacebookService _fbService = new
    facebook.Components.FacebookService();
```

```
        protected void Page_Load(object sender, EventArgs e)
        {

            // ApplicationKey and Secret are acquired when you sign up for
            _fbService.ApplicationKey = "YOURKEYHERE";
            _fbService.Secret = "YOURSECRETHERE";
```

Visual Basic

```
Public Partial Class _Default
      Inherits Page
      Private _fbService As facebook.Components.FacebookService = New
      facebook.Components.FacebookService()

      Protected Sub Page_Load(ByVal sender As Object, ByVal e As EventArgs)

      ' ApplicationKey and Secret are acquired when you sign up for
        _fbService.ApplicationKey = "YOURKEYHERE"
        _fbService.Secret = "YOURSECRETHERE"
```

- Set the IsDesktopApplication property of the FacebookService component

  Visual C#

  ```
  _fbService.IsDesktopApplication = false;
  ```

  Visual Basic

  ```
  _fbService.IsDesktopApplication = False
  ```

- Check if we have already stored the Facebook session information or if the auth_token is in the query params. We will store what we know about the current user's Facebook Session in a server side variable. We will then check that variable to see if we already have established a Facebook session on behalf of the current user.

  Visual C#

  ```
  string sessionKey = (string)Session["Facebook_session_key"];
  long userId = (long)Session["Facebook_userId"];

  // When the user uses the Facebook login page, the redirect back here
  will will have the auth_token in the query params
  string authToken = Request.QueryString["auth_token"];
  ```

  Visual Basic

```
            Dim sessionKey As String =
TryCast(Session("Facebook_session_key"), String)
            Dim userId As String = TryCast(Session("Facebook_userId"),
String)

            ' When the user uses the Facebook login page, the redirect back
here will will have the auth_token in the query params
            Dim authToken As String = Request.QueryString("auth_token")
```

- If we have an established session, set it into our instance of the service

Visual C#
```
if (!String.IsNullOrEmpty(sessionKey))
{
    _fbService.SessionKey = sessionKey;
    _fbService.UserId = userId;
}
```

Visual Basic

```
    ' We have already established a session on behalf of this user
    If (Not String.IsNullOrEmpty(sessionKey)) Then
        _fbService.SessionKey = sessionKey
        _fbService.UserId = userId
```

- If not, check if we have the auth_token in the query params. If we do, it means we just got called from the Facebook login page.

Visual C#
```
else if (!String.IsNullOrEmpty(authToken))
{
    _fbService.CreateSession(authToken);
    Session["Facebook_session_key"] = _fbService.SessionKey;
    Session["Facebook_userId"] = _fbService.UserId;
    Session["Facebook_session_expires"] = _fbService.SessionExpires;
}
```

Visual Basic
```
    ' This will be executed when Facebook login redirects to our page
    ElseIf (Not String.IsNullOrEmpty(authToken)) Then
        _fbService.CreateSession(authToken)
        Session("Facebook_session_key") = _fbService.SessionKey
        Session("Facebook_userId") = _fbService.UserId
        Session("Facebook_session_expires") =
_fbService.SessionExpires
```

- If neither, we need to redirect the user to the Facebook hosted login page

Visual C#
```
else
{
    Response.Redirect(@"http://www.Facebook.com/login.php?api_key=" +
_fbService.ApplicationKey + @"&v=1.0");
}
```

Visual Basic
```
    Else

Response.Redirect("http://www.Facebook.com/login.php?api_key=" &
_fbService.ApplicationKey & "&v=1.0")
    End If
```

- Set the friends property of the FriendList

```
  Visual C#
  if (!IsPostBack)
  {
      // Use the FacebookService Component to populate Friends
      FriendList1.Friends = _fbService.friends.getUserObjects();
  }

  Visual Basic
If (Not IsPostBack) Then
      ' Use the FacebookService Component to populate Friends
      FriendList1.Friends = _fbService.GetFriends()
End If
```

5.3 IFrame Canvas Development

One type of canvas page supported by the Facebook platform is the IFrame canvas. With this type of canvas page, Facebook will host an iframe containing the content for you application. This type of canvas can not take advantage of the features of FBML, but allows for more traditional web development practices for Facebook applications.

How to start your IFrame canvas application using the Facebook Developer Toolkit.

- Request a new developer API key from http://www.facebook.com/developers/
- Configure the following fields. The values here are the values used by the sample, you can replace with your own values.
    - Callback URL: http://facebook.claritycon.com/IFrame/
    - Canvas Page URL : aspnetcanvasiframe
        - Use IFRAME: true; Use FBML: false
    - Application Type: Website
    - Can your application be added on facebook: yes
    - Post-Add URL: http://apps.facebook.com/aspnetcanvasiframe (same as canvas page)
    - Default Profile Box: Wide
    - Side Nav URL: http://apps.facebook.com/aspnetcanvasiframe (same as canvas page)
- Create new asp.net website in Visual Studio
- Add reference to Facebook.dll
- Add reference to Facebook.WebControls.dll
- Update codebehind (Default.aspx.cs) on Default.aspx to subclass Facebook.WebControls.CanvasIFrameBasePage

```
Visual C#

using facebook.Schema;
using facebook.web;

public partial class _Default : CanvasIFrameBasePage
{

Visual Basic

Imports facebook.Schema
Imports facebook.web

Public Partial Class _Default
```

```
    Inherits CanvasIFrameBasePage
```

- Add code to set API key, secret (Make sure to use your own api key and secret acquired during above)
- Add code to call base.Page_Loaded

Visual C#

```
 private const string FACEBOOK_API_KEY = "YOURAPIKEYHERE";
 private const string FACEBOOK_SECRET = "YOURSECRETHERE";

 new protected void Page_Load(object sender, EventArgs e)
 {
     base.ApiKey = FACEBOOK_API_KEY;
     base.Secret = FACEBOOK_SECRET;
     base.Page_Load(sender, e);
```

Visual Basic

```
   Private Const FACEBOOK_API_KEY As String = "YOURAPIKEYHERE"
   Private Const FACEBOOK_SECRET As String = "YOURSECRETHERE"

   Shadows Protected Sub Page_Load(ByVal sender As Object, ByVal e As
EventArgs)
      MyBase.ApiKey = FACEBOOK_API_KEY
      MyBase.Secret = FACEBOOK_SECRET
      MyBase.Page_Load(sender, e)
```

- Add code to use the base page's API client to invoke API calls on behalf of the logged in user.

Visual C#

```
if (!IsPostBack)
{
   // Use the API client to populate Friends
   user u = this.API.users.getInfo();
   IList<user> friends = this.API.friends.getUserObjects();
```

Visual Basic

```
If (Not IsPostBack) Then
   ' Use the API client to populate friends
   Dim u As user = Me.API.users.getInfo()
   Dim f As IList(Of user) = Me.API.friends.getUserObjects()
```

- Optionally, if you prefer to use Cookies instead of Sessions to pass Facebook Session information between pages, set the UseSession property of the CanvasIFrameBasePage to false
- Optionally, if you prefer to not automatically redirect users who have not already added your application to the add page (if you prefer to have an information page with a link to the add functionality), set the AutoAdd property of CanvasIFrameBasePage to false
- Publish Website to publicly available location
- Access application using http://apps.facebook.com/YOURCANVASPAGE

5.4 FBML Canvas Development

FBML (Facebook Markup Lanaguage) is a subset of HTML with some Facebook-specific features. The second type of Facebook Canvas application that is supported is the FBML canvas. With an FBML canvas, the developer is essentially developing an FBML web service that the Facebook server can invoke and the service will emit FBML that can be rendered as HTML by the displayed page. This canvas type allows for using some powerful Facebook-specific constructs that help to developers provide a better Facebook integrated experience.

How to start your FBML canvas application using the Facebook Developer Toolkit.

- Request a new developer API key from http://www.facebook.com/developers/
- Configure the following fields. The values here are the values used by the sample, you can replace with your own values.
    - Callback URL: http://facebook.claritycon.com/FBML/
    - Canvas Page URL : aspnetcanvasfbml
        - Use IFRAME: false; Use FBML: true
    - Application Type: Website
    - Can your application be added on facebook: yes
    - Post-Add URL: http://apps.facebook.com/aspnetcanvasfbml (same as canvas page)
    - Default Profile Box: Wide
    - Side Nav URL: http://apps.facebook.com/aspnetcanvasfbml (same as canvas page)
- Create new asp.net website in Visual Studio
- Add reference to facebook.dll
- Add reference to facebook.web.dll
- Update Default.aspx to remove all of the default html in the page, to ensure no FBML incompatibilities.
- Update codebehind (Default.aspx.cs) on Default.aspx to subclass Facebook.WebControls.CanvasFBMLBasePage

Visual C#

```
using facebook.Schema;
using facebook.web;

public partial class _Default : CanvasFBMLBasePage
{
```

Visual Basic

```
Imports facebook.Schema
Imports facebook.web

Public Partial Class _Default
    Inherits CanvasFBMLBasePage
```

- Add code to set API key, secret (Make sure to use your own API key and secret acquired during above)
- Add code to call base.Page_Load

Visual C#

```
     private const string FACEBOOK_API_KEY = "YOURAPIKEYHERE";
     private const string FACEBOOK_SECRET = "YOURSECRETHERE";

     new protected void Page_Load(object sender, EventArgs e)
     {
         base.Api = FACEBOOK_API_KEY;
         base.Secret = FACEBOOK_SECRET;
         base.Page_Load(sender, e);
```

Visual Basic

```
    Private Const FACEBOOK_API_KEY As String = "YOURAPIKEYHERE"
    Private Const FACEBOOK_SECRET As String = "YOURSECRETHERE"

    Shadows Protected Sub Page_Load(ByVal sender As Object, ByVal e As
EventArgs)
        MyBase.Api = FACEBOOK_API_KEY
        MyBase.Secret = FACEBOOK_SECRET
        MyBase.Page_Load(sender, e)
```

- Add code to use FacebookService to invoke API calls on behalf of the logged in user.

Visual C#

```
if (!IsPostBack)
{
   // Use the API client to populate Friends
   user u = this.API.users.getInfo();
   IList<user> friends = this.API.friends.getUserObjects();
```

Visual Basic

```
If (Not IsPostBack) Then
   ' Use the API client to populate friends
   Dim u As user = Me.API.users.getInfo()
   Dim f As IList(Of user) = Me.API.friends.getUserObjects()
```

- Optionally, if you prefer to not automatically redirect users who have not already added your application to the add page (if you prefer to have an information page with a link to the add functionality), set the AutoAdd property of CanvasIFrameBasePage to false
- Publish Website to publicly available location
- Access application using http://apps.facebook.com/YOURCANVASPAGE

5.5 Canvas Development with Master Pages

New to version 1.3 of the Facebook Developer Toolkit are the CanvasIFrameMasterPage and CanvasFBMLMasterPage classes. These simplify development of a canvas page by allowing you to create just one master page which handles the authorization with Facebook, session information, etc. Each page in your application is then a content page for that master page, and each one relies on the master page to provide the API, which prevents you from writing more code.

To use a master page for your application, follow these steps:

- Register a new Facebook application as in the previous sections

- Create a new ASP.NET web site in Visual Studio
- Add references to Facebook.dll and Facebook.WebControls.dll
- In the Web Site menu, "click Add New Item…". Select Master Page and click Add.
- In the code behind for the master page, change the class to inherit from Facebook.WebControls.CanvasIFrameMasterPage or Facebook.WebControls.CanvasFBMLMasterPage, depending on which one you want to use.

```
public partial class MasterPage : facebook.web.CanvasIFrameMasterPage
```

- In each of these classes, to make things easier, you can get the authorization settings from your web.config file instead of in code. To do so, add the following entries to your web.config's appSettings section:

```
<add key="APIKey" value="a0d032ca16ca9047e155b1f3b8044067"/>
<add key="Secret" value="95b582ed74a9ddac45c234b10aa006a4"/>
```

- You will also need to add your callback and canvas URL suffix to appSettings (the following are sample values):

```
      <add key="Callback"
value="http://fbtest2.claritycon.com/FBMLCanvasSample/"/>
```

- `<add key="Suffix" value="fbmlcanvassample"/>`
- Finally, if your application is in a subdirectory or your web application's root directory (not necessarily the server root), you will need to add an entry indicating that subdirectory:
- `<add key="WebApplicationSubdirectory" value="FBMLCanvasSample/"/>`

- For each new page, right-click your master page in Solution Explorer and click Add Content Page
    - The content of your pages will go inside the <asp:Content> tags. Use the one "head" one for your header information, and the other one for anything that goes in the body of the page
    - You may also wish to change Default.aspx to use the master page. You can either delete the original one and create a new one as a content page, or modify the old one to have a MasterPageFile attribute in its Page directive and give it <asp:Content> tags like in the other pages.
- Build and publish your web site as normal, and use it as you would a normal Canvas page.

5.6 AJAX in IFrame Canvas

(Note: this section has not been updated for the 2.0 version of the toolkit. However, the code should be relatively similar. If you would still like for this section to be updated with the correct code, please post your request on the Codeplex site - http://www.codeplex.com/FacebookToolkit .)

Using the ASP.NET AJAX extensions within for IFrame Canvas application is pretty straightforward. However, one important note, AJAX from an IFrame is not supported in without Framework 3.5 installed on the WebServer. This is a big obstacle for anyone using a public hosting service until Framework 3.5 GoLive licensing is available. The following walkthrough assumes that Framework 3.5 is installed. This walkthrough describes the web.config updates required to enable the asp.net ajax extensions on your existing asp.net web site. And I simple page illustrating the usage.

1. Install ASP.NET Ajax Extensions from http://ajax.asp.net/
2. Install .NET Framework 3.5 from http://www.microsoft.com/downloads/details.aspx?FamilyId=E3715E6F-E123-428B-8A0F-028AFB9E0322&displaylang=en
3. Open your web.config, in the system.web section add the following line:

```
    <pages>
      <controls>
        <add tagPrefix="asp" namespace="System.Web.UI"
assembly="System.Web.Extensions, Version=1.0.61025.0, Culture=neutral,
PublicKeyToken=31bf3856ad364e35"/>
      </controls>
    </pages>
```

   This will save you having to have the following directive at the top of each page:

```
<%@ Register Assembly="System.Web.Extensions, Version=1.0.61025.0,
Culture=neutral, PublicKeyToken=31bf3856ad364e35"
    Namespace="System.Web.UI" TagPrefix="asp" %>
```

4. Verify that System.Web section has the following httpHandlers and httpModules configured, if not, add the missing settings:

```
    <httpHandlers>
      <remove verb="*" path="*.asmx"/>
      <add verb="*" path="*.asmx" validate="false"
type="System.Web.Script.Services.ScriptHandlerFactory,
System.Web.Extensions, Version=1.0.61025.0, Culture=neutral,
PublicKeyToken=31bf3856ad364e35"/>
      <add verb="*" path="*_AppService.axd" validate="false"
type="System.Web.Script.Services.ScriptHandlerFactory,
System.Web.Extensions, Version=1.0.61025.0, Culture=neutral,
PublicKeyToken=31bf3856ad364e35"/>
      <add verb="GET,HEAD" path="ScriptResource.axd"
type="System.Web.Handlers.ScriptResourceHandler, System.Web.Extensions,
Version=1.0.61025.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35"
validate="false"/>
    </httpHandlers>

    <httpModules>
      <add name="ScriptModule" type="System.Web.Handlers.ScriptModule,
System.Web.Extensions, Version=1.0.61025.0, Culture=neutral,
PublicKeyToken=31bf3856ad364e35"/>
    </httpModules>
```

5. Add a new .aspx file (in our example, AJAXSample.aspx
6. Add a ScriptManager to the .aspx page

```
    <asp:ScriptManager ID="ScriptManager1" runat="server" />
```

7. Add an UpdatePanel with ContentTemplate

```
    <asp:UpdatePanel ID="upPanel" runat="server">
    <ContentTemplate>
    <asp:Label id="lblUpdate" runat="server" Text="Before click" />
    <asp:Button ID="btnAjax" runat="server" Text="Fire Ajax"
OnClick="btnAjax_Click" />

    </ContentTemplate>
    </asp:UpdatePanel>
```

8. That is it, now controls within your UpdatePanel will be updated and fire events without posting back the main Browser thread.
9. In our example, we catch the button press and update the label and button in code behind. The page is updated without posting back.

Visual C#

```
 protected void btnAjax_Click(object sender, EventArgs e)
 {
     lblUpdate.Text = "After click";
     btnAjax.Text = "Fired Ajax";
     btnAjax.Enabled = false;
 }
```

Visual Basic

```
    Protected Sub btnAjax_Click(ByVal sender As Object, ByVal e As
EventArgs)
            lblUpdate.Text = "After click"
            btnAjax.Text = "Fired Ajax"
            btnAjax.Enabled = False
    End Sub
```

Release Notes:

Version 1.1:

- Added methods to FacebookService to wrap the Notifications.get Facebook API call. Added a new data object "Notifications" for wrapping the results.
- Added Status and ProfileUpdateDate to the User object. Status contains the user's Facebook Status Message and time when it was updated. ProfileUpdateDate contains the date when the user last updated their Facebook profile.
- Added SetFMBL method. Also correct GetRequestURL to correctly encode the format.
- Added DirectFQLQuery. As designed on f8 wiki.
- Fixed GetFriendsAppUsers. As designed on F8 wiki.
- Fixed to add error handling to Country parsing in LocationParser
- Added implementations of 2 notifications and 2 feeds apis.
- Fixed parameter sorting for feeds apis to work correctly.
- Change REST Call from GET to POST.
- Add GetLoggedInUser
- Fix handling for no returned secret
- Add overload to GetUserInfo that takes a Collection of user ids

Version 1.2:

- Refactored FacebookService
- Added AsyncFacebookService
- Refactored Parsers and Entities into their own directory and namespace
- Added improved photo api
- Added Canvas Base pages to Facebook.WebControls
- Added Canvas samples to Facebook.WebControls
- Added IFrame Ajax sample
- Added IFrame Silverlight Example

- Updated to new notifications API **INTERFACE CHANGE**
- Updated PublishStory and PublishAction interfaces **INTERFACE CHANGE**
- Updated CreateAlbum interface to return created Album **INTERFACE CHANGE**
- Added SetFBML override taking in userid
- Added small and big picture url and bitmap to photo and user
- Added GetFriendsNonAppUsers method

Version 1.3

- Made lots of bug fixes
- Added Support for .NET Compact Framework
- Added Size on photo album
- Added Master page versions of Canvas Base pages
- Added Support for new request/invitation interface with multi_friend_selector *** INTERFACE CHANGE ***
- Added Big/small pictures on group
- Added Upload Photo returns a photo id and album id
- Added Support for PublishTemplatizedAction
- Added GetFBML
- Added RefreshRefHandle
- Added RefreshImgSrc
- Added SetRefHandle
- Added Session Timeout handling to IFrame base page.

Version 1.4

- (skipped by request)

Version 1.5

- Created setup packages and .msi installer

Version 1.6

- Updated SetFBML() to support Facebook's API change applied on 1/17/2008

    - Profile.SetFBML no longer accepts a single ‘markup’ parameter to hold profile, profile action and mobile profile FBML markup. The new wrapper methods break the call into three parameters, profileFBML, profileActionFBML and mobileFBML.

Version 1.7

- Refactored code and fixed bugs.

Version 2.0

- Re-architected the API to create a lighter-weight wrapper around the API by using Linq2XSD to auto-generate the parsers and entity objects.
- Updated login logic to account for the new profile changes.
- Added new samples that work with the new API.
- Changed default value for RequireLogin on CanvasFBMLBasePage and CanvasFBMLMasterPage to be false instead of true
- Changed parameters for liveMessage.send
- Implemented feed.publishUserAction
- Made API.sendRequest public
- Renamed Api property on CanvasIFrameBasePage to ApiKey to follow conventions and avoid case-sensitivity issues
- Various bug fixes

Version 2.1

- Added support for missing apis
    - admin.getRestrictionInfo
    - admin.setRestrictionInfo
    - admin.banUsers
    - admin.unbanUsers
    - admin.getBannedUsers
    - auth.revokeExtendedPersmission
    - commnets.get
    - fbml.deleteCustomTags
    - fbml.getCustomTags
    - fbml.registerCustomTags
    - links.get
    - links.post
    - notes.create
    - notes.delete

    - notes.edit
    - notes.get
    - status.get
    - status.set
    - stream.addComment
    - stream.addLike
    - stream.get
    - stream.getComments
    - stream.getFilters
    - stream.publish
    - stream.remove
    - stream.removeComment
    - stream.removeLike
    - users.getStandardInfo
    - users.IsVerified
    - video.getUploadLimits
    - video.upload
- Fixed the following bugs
  10512,10705,11009,11215,11410,12026,12351,12386,12446,12590,12592,12593,12742,12818,13231