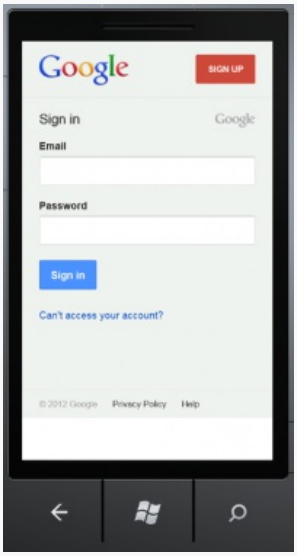
Google+ authentication in Windows Phone application

This article demonstrates how to implement Google+ OAuth2.0 authentication and fetch user data on Windows Phone, without using a third party library.

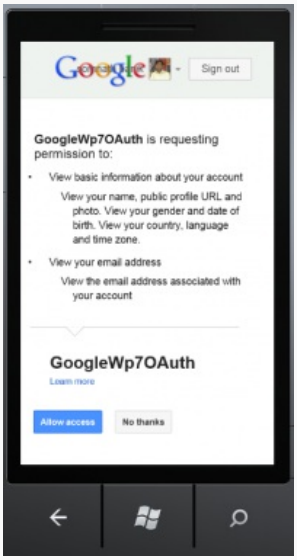
Note: Google provides a OAuth2.0 .Net Client Library that can be used to access Google+ services. The author has not tried this library.

Introduction

Google+ OAuth2.0 authentication authorizes third party applications to access user data and other Google+ features using the Google+ API. In this article we will create an application which will authenticate and authorize an application to access user profile information.



Login Page



Application Authentication

Google+ User Profile

Implementation

Create an empty Windows Phone Project

Launch **Visual Studio** -> click on **File**->**New Project**-> select **Windows Phone Application** (Visual C# Template) -> add **Name** and **Location** of the project -> click **OK** to create the project.

Register your application with Google

To access Google+ API your application much be registered through Google APIs Console. If you are using the console for the first time and have never created any Google App before, then here are the steps for you:

- Go to Google APIs Console
- Click on the blue button **Create Project**
- This will open the list of all *Google API Services*.
- Switch **ON** the *Google+ API* service from the list and accept the license agreement.
- Now go to *API Access* tab in the left panel.
- Click on the blue button **Create an OAuth2.0 client ID**
- Enter the *Product Name* (this is the name of your application, *GoogleWp7OAuth* is the app name used in this article), *Product Logo* (optional) and click **Next**
- Select **Installed Application** from the application type and click on **Create Client Id** button.
- This will create *Client ID*, *Client secret* and *Redirected URIs* of the application. Copy the set values for future use.

See Google APIs Console Help for more information about about creating and managing projects the console.

Adding controls on application page

(C) Copyright Nokia 2013. All rights reserved.

In our application we have two pages **MainPage.xaml** and **GooglePlusLoginPage.xaml**. **MainPage.xaml** consist of one Button, three TextBlock and one Image control.

```
<Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">
  <Button Content="Login " Height="72" HorizontalAlignment="Left" Margin="31,517,0,0" Name="buttonLogin" VerticalAlignment="Top" Wid
  <Image Height="238" HorizontalAlignment="Left" Margin="31,48,0,0" Name="imageGooglePlusUserPicture" Stretch="Fill" VerticalAlignme
  <TextBlock Height="74" HorizontalAlignment="Left" Margin="31,309,0,0" Name="textBlockGooglePlusUserName" Text="" VerticalAlignment
  <TextBlock Height="30" HorizontalAlignment="Right" Margin="0,402,32,0" Name="textBlockGooglePlusUserEmail" Text="" VerticalAlignme
  <TextBlock Height="30" HorizontalAlignment="Left" Margin="31,452,0,0" Name="textBlockGooglePlusUserGender" Text="" VerticalAlignme
</Grid>
```

The three `TextBlock` displays user *name*, *email* and *gender*, and the `Image` control displays the user *profile picture*. The `Button` control in **MainPage.xaml** is used to navigate to **GooglePlusLoginPage.xaml** which has a WebBrowser control to load the Google+ login page.

```
<Grid x:Name="ContentPanel" Grid.Row="1" Margin="0,0,0,0">
          <phone:WebBrowser HorizontalAlignment="Left" Margin="0,0,0,0" Name="webBrowserGooglePlusLogin" VerticalAlignment="Top" H
</Grid>
```

When the Google+ login is successful the page **GooglePlusLoginPage.xaml** gets closed and the control goes back to **MainPage.xaml** which then displays user's *name*, *email*, *gender* and *profile picture*.

Launching Google+ login page

When user clicks on Login `Button` of the **MainPage.xaml** it navigates to **GooglePlusLoginPage.xaml** which then launch the `WebBrowser` and loads the Google+ login dialog.

```
#1
//load google plus login page
   void GooglePlus_LoginPage_Loaded(object sender, RoutedEventArgs e)
        {
           var url = "https://accounts.google.com/o/oauth2/auth?response_type=code&redirect_uri=" +RedirctedUri+ "&scope=https://www
           webBrowserGooglePlusLogin.Navigate(new Uri(url, UriKind.RelativeOrAbsolute));
        }
```

After user enters Google+ credentials it asks for application authentication. In this page you will see the name (*GoogleWp7OAuth* in this example) of your application requesting for permission to access user's information.

Get the Authorization Code and Access Token

When user clicks on the *Allow access* button on the application permission page, the Google server returns an authorization code to the application either in the title bar of the browser or as a query string parameter to `http://localhost` which depends on the parameter sent in the request. For more information, see Choosing a Redirect URI.

We have used the parameter `http://localhost` for `redirect_uri` in **#1** so that we get authorization code as a query string parameter. The following code receives an authorization code from Google server as a query string parameter when the user grants permission to your application.

```
    private void webBrowserGooglePlusLogin_Navigating(object sender, NavigatingEventArgs e)
         {
             if (e.Uri.Host.Equals("localhost"))
             {
                 webBrowserGooglePlusLogin.Visibility = Visibility.Collapsed;
                 e.Cancel = true;
                 int pos = e.Uri.Query.IndexOf("=");
                 //get the access code
                  string messageCode = pos > -1 ? e.Uri.Query.Substring(pos + 1) : null;
                 //when code is not equeals to null get the access token
                 if (messageCode != null)
                 {
                     //get the access token
                     Parameters = "code=" + messageCode + "&client_id=" + ClientId + "&client_secret=" + ClientSecret + "&redirect_ur
                     HttpWebRequest webRequest = (HttpWebRequest)WebRequest.Create("https://accounts.google.com/o/oauth2/token");
                     webRequest.Method = "POST";
                     webRequest.ContentType = "application/x-www-form-urlencoded";
                     // Start web request
                     webRequest.BeginGetRequestStream(new AsyncCallback(GetRequestStreamCallback), webRequest);
                 }
             }
         }
```

We get the authorization code from the query string and assign it to a string variable called `messageCode`. Once the application receives the authorization code it then sends a HTTP Post request to get the *access token* and *refresh token*. To learn more on HTTP post request parameters used to get *access token* and *refresh token* as a response, see Handling the Response.

The *Refresh Token* is used to generate a new *access token* when it expires (this is an optional feature and is not needed to implement in this example for application authentication). See Using a Refresh Token for more information. When we get the response we then parse the *access token* for future use.

```
    void GetResponseCallback(IAsyncResult asynchronousResult)
         {
             try
             {
                 var request = (HttpWebRequest)asynchronousResult.AsyncState;
                 using (var resp = (HttpWebResponse)request.EndGetResponse(asynchronousResult))
                 {
                     using (var streamResponse = resp.GetResponseStream())
                     {
                         var GooglePlusSerializerData = new DataContractJsonSerializer(typeof(GooglePlusAccessToken));
                         var GooglePlusProfileData = GooglePlusSerializerData.ReadObject(streamResponse) as GooglePlusAccessToken;
```


(C) Copyright Nokia 2013. All rights reserved.

```
                        this.Dispatcher.BeginInvoke(
                            (Action<GooglePlusAccessToken>)((GooglePlusUserData) =>
                            {
                                //save the response
                                thisApp.AccessToken = googlePlusTokens.AccessToken=  GooglePlusUserData.AccessToken;
                                googlePlusTokens.RefreshToken = GooglePlusUserData.RefreshToken;
                                googlePlusTokens.ExpiresIn = GooglePlusUserData.ExpiresIn;
                                googlePlusTokens.TokenType = GooglePlusUserData.TokenType;
                                // request user profile
                                RequestForUserProfile();
                            }), GooglePlusProfileData);
                        }
                    }
                }
                catch (WebException ex)
                {

                }
            }
```

Load Google+ User Profile

`RequestForUserProfile()` method puts a web request with *access token* and gets the basic user's information like *name*, *email*, *gender*, *profile picture*, *birthday* etc. by Calling Google+ API

```
//request for user prifile
   void RequestForUserProfile()
        {
          //  var urlProfile = "https://www.googleapis.com/plus/v1/people/me?access_token=" + thisApp.AccessToken;
            var urlProfile = "https://www.googleapis.com/oauth2/v1/userinfo?access_token="+thisApp.AccessToken;
            // web request user profile
            WebRequest request = WebRequest.Create(urlProfile);
            request.BeginGetResponse(new AsyncCallback(this.ResponseCallbackProfile), request);
        }
```

We catch the asynchronous call at `ResponseCallbackProfile()` and parse the user's information and display it on the **MainPage.xaml** page.

```
// user profile response
   private void ResponseCallbackProfile(IAsyncResult asynchronousResult)
        {
            try
            {
                var request = (HttpWebRequest)asynchronousResult.AsyncState;
                using (var resp = (HttpWebResponse)request.EndGetResponse(asynchronousResult))
                {
                    using (var streamResponse = resp.GetResponseStream())
                    {
                        var GooglePlusSerializerData = new DataContractJsonSerializer(typeof(GooglePlusUserProfile));
                        var GooglePlusProfileData = GooglePlusSerializerData.ReadObject(streamResponse) as GooglePlusUserProfile;
                        this.Dispatcher.BeginInvoke(
                            (Action<GooglePlusUserProfile>)((GooglePlusUserData) =>
                            {
                             thisApp.UserName=  googlePlusUserInfo.UserName = GooglePlusUserData.name;
                             thisApp.UserImage = googlePlusUserInfo.UserPicture = GooglePlusUserData.picture;
                             if (thisApp.UserImage == null)
                             {
                                 thisApp.UserImage = googlePlusUserInfo.UserPicture=  "https://lh3.googleusercontent.com/-_kvINpT6jt
                             }
                               googlePlusUserInfo.UserBidthday = GooglePlusUserData.birthday;
                              thisApp.UserEmail= googlePlusUserInfo.UserEmail = GooglePlusUserData.email;
                               googlePlusUserInfo.UserFamilyName = GooglePlusUserData.family_name;
                               thisApp.UserGender= googlePlusUserInfo.UserGender= GooglePlusUserData.gender;
                               googlePlusUserInfo.UserGivenName=GooglePlusUserData.given_name;
                               googlePlusUserInfo.UserId=GooglePlusUserData.id;
                               googlePlusUserInfo.UserLink=GooglePlusUserData.link;
                               Deployment.Current.Dispatcher.BeginInvoke(() =>
                               {
                                   NavigationService.GoBack();
                               });
                            }), GooglePlusProfileData);
                    }
                }
            }
            catch (WebException ex)
            {

            }
        }
```

Note: To run the code make sure you have replaced **<Your Client ID>** with **Client ID** and **<Your Client Secret>** with **Client secret** that you got from Google APIs Console.

Summary

The goal of this example is to show Google+ OAuth2.0 authentication without using any third party library. Applications can also be authenticated using Google API key. To learn more, see Google Authorization Protocol .

Related Article

- Facebook authentication in Windows Phone application
- How to use Facebook Graph API in Windows Phone

Source Code

- The full source code of the example is available here: File:GooglePlusAuthentication.zip

(C) Copyright Nokia 2013. All rights reserved.

- If you are in a hurry to test the application, unzip the source code and install the binary file from the given location: *GooglePlusAuthentication->bin->Release->GooglePlusAuthentication.xap*

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.