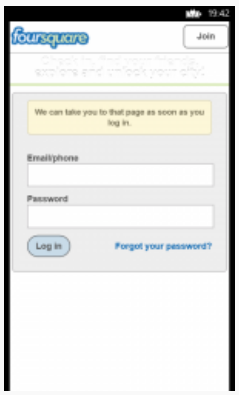
Foursquare authentication in Windows Phone application

This article demonstrates how to implement Foursquare OAuth2.0 authentication and fetch Foursquare data without using any third party library.

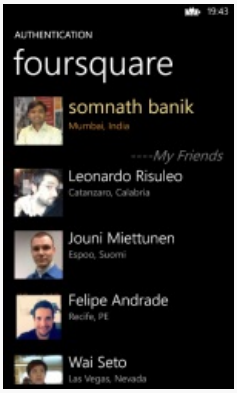
Introduction

Foursquare OAuth2.0 authentication enables third party application to access user data using Foursquare API. In this article we will create an application which will pass Foursquare OAuth2.0 authentication to access user profile information and friends list.



Login Page

Application Authentication



Foursquare User Profile

Implementation

Create a standard Windows Phone Project

Let's create a standard Windows Phone Application Project

- Launch **Visual Studio**
- Click on **File**
- Click on **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 Foursquare

To access Foursquare API, your application must be registered through Foursquare Developer account. If you are registering your app for the first time and have never created any Foursquare app before, then please perform the following steps:

- Go to Foursquare Developer account.
- Click on **My Apps** and then click on the button **CREATE A NEW APP**
- Follow the instruction on the page and complete the form
- Select **Web Connect Support** from *Install options*.
- Make sure to add a **Redirect URI** before submitting the form.
- This will create a Client Id, Client Secret and a Redirected Uri. Copy the set of values to be used in our application.

At the end of the authentication process, the web browser will be redirected back to the provided Redirected URL along with the Access Token; we parse the Access Token from the response URL for future use.

Adding UI controls on application page

In our application we have two pages **MainPage.xaml** and **FourSquareLoginPage.xaml**. **MainPage.xaml** consist of Button, TextBlock, Image and a LongListSelector control.

```
<!--ContentPanel - place additional content here-->
        <Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">
            <StackPanel Orientation="Vertical" Margin="0,0,0,17">
```


(C) Copyright Nokia 2014. All rights reserved.

```
            <StackPanel  Orientation="Horizontal" Margin="0,0,0,0" >
                <Image Name="UserImage" HorizontalAlignment="Left" Height="100"
Margin="12,0,0,0" VerticalAlignment="Top" Width="100"/>
                <StackPanel  Margin="12,0,0,0" >
                    <TextBlock Name="UserName" Margin="0,-9,0,0" Foreground="#efd584"
HorizontalAlignment="Left"  TextWrapping="Wrap" Text="" FontSize="40"
VerticalAlignment="Top"/>
                    <TextBlock  Foreground="Peru" Margin="0,0,0,0" Name="UserHomeCity"
HorizontalAlignment="Left"  TextWrapping="Wrap" Text="" VerticalAlignment="Top" />
                </StackPanel>
            </StackPanel>
                <TextBlock Foreground="Gray" FontStyle="Italic"  Margin="0,0,12,0"
Name="FriendTag" Visibility="Collapsed" HorizontalAlignment="Right"  TextWrapping="Wrap"
Text="----My Friends" FontSize="30" VerticalAlignment="Top" />
            </StackPanel>
            <Button Name="LoginButton" Content="Login" HorizontalAlignment="Left"
Margin="37,483,0,0" VerticalAlignment="Top" Width="393" Height="84"
Click="Button_Click_1"/>
            <phone:LongListSelector Visibility="Collapsed" Name="ListBoxFriends"
HorizontalAlignment="Left" Height="450" Margin="12,147,0,0" VerticalAlignment="Top"
Width="427">
                <phone:LongListSelector.ItemTemplate>
                    <DataTemplate>
                        <StackPanel  Orientation="Horizontal" Margin="0,12,0,17" >
                            <Image Source="{Binding FriendPhoto}"
Stretch="UniformToFill" Height="100" Width="100"  VerticalAlignment="Top"
Margin="0,0,0,0"/>
                            <StackPanel  Margin="0,0,0,0" >
                                <TextBlock Text="{Binding FriendName}"
TextWrapping="NoWrap" Margin="12,-9,0,0"    Style="{StaticResource PhoneTextLargeStyle}"
/>
                                <TextBlock   Text="{Binding FriendHomeCity}"
TextWrapping="NoWrap" Margin="12,0,0,0" Style="{StaticResource PhoneTextSubtleStyle}" />
                            </StackPanel>
                        </StackPanel>
                    </DataTemplate>
                </phone:LongListSelector.ItemTemplate>
            </phone:LongListSelector>
        </Grid>
```

The Button control in **MainPage.xaml** is used to navigate to **FourSquareLoginPage.xaml** which has a WebBrowser control to load the FourSquare login page. To enable the browser scripting make sure the `IsScriptEnabled` property of the WebBrowser control is `True` or else you might experience a blank page.

```
<!--ContentPanel - place additional content here-->
        <Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">
            <phone:WebBrowser Name="WebBrowserLogin" IsScriptEnabled="True" Height="768"
Width="480" VerticalAlignment="Top" HorizontalAlignment="Center" />
        </Grid>
```

When the Foursquare login is successful, the page **FourSquareLoginPage.xaml** gets closed and the control goes back to **MainPage.xaml** which then displays user's profile along with user's friend list.

Launching Foursquare login page

When user clicks on Login Button on the **MainPage.xaml**, it navigates to **FourSquareLoginPage.xaml** which calls the

(C) Copyright Nokia 2014. All rights reserved.

Foursquare login URL and launch the Foursquare login dialog.

```
void FourSquareLoginPage_Loaded(object sender, RoutedEventArgs e)
{
            var url = "https://foursquare.com/oauth2/authorize?display=touch&client_id="
+ ClientId + "&client_secret=" + ClientSecret + "&response_type=token&redirect_uri=" +
RedirectedUri;
            WebBrowserLogin.Navigate(new Uri(url, UriKind.RelativeOrAbsolute));
            WebBrowserLogin.NavigationFailed += WebBrowserLogin_NavigationFailed;
            WebBrowserLogin.Navigating += WebBrowserLogin_Navigating;
}
```

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

Get the Access Token

When user clicks on the *Allow* button on the application permission page, the Foursquare server returns an *Access Token* as a query string parameter to *http://www.foursquare.com*(in this case). We have used the parameter *http://www.foursquare.com* for `RedirectedUri` so that we get *Access Token* as a query string parameter.

```
void WebBrowserLogin_Navigating(object sender, NavigatingEventArgs e)
{
            if (e.Uri.IsAbsoluteUri)
            {
                if (e.Uri.Host.Equals("www.foursquare.com"))
                {
                    string responseString = e.Uri.ToString();
                    string[] splitAccessToken = responseString.Split(new Char[] { '#',
'=' });
                    string accessTokenString = splitAccessToken.GetValue(1).ToString();
                    string accessTokenValue = splitAccessToken.GetValue(2).ToString();
                    if (accessTokenString == "access_token")
                    {
                        AccessToken = accessTokenValue;
                        if(AccessToken != null)
                          LoadUserProfile();
                    }
                    else
                        MessageBox.Show("Authentication Failed!");
                    WebBrowserLogin.Visibility = System.Windows.Visibility.Collapsed;
                }
            }
}
```

Load Foursquare User Profile and Friend List

`LoadUserProfile()` method will put a web request to get the user profile information and friend list using Foursquare API.

```
private void LoadUserProfile()
{
            var urlProfile ="https://api.foursquare.com/v2/users/self?
oauth_token="+AccessToken;
            WebRequest request = WebRequest.Create(urlProfile);
            request.BeginGetResponse(new AsyncCallback(this.ResponseCallbackProfile),
request);
```


(C) Copyright Nokia 2014. All rights reserved.

```
}
```

We catch the asynchronous call at `ResponseCallbackProfile()` and parse the details such as user name, profile picture, home location and friend list and display them on the **MainPage.xaml**.

```
private void ResponseCallbackProfile(IAsyncResult asynchronousResult)
{
           try
            {
                var request = (HttpWebRequest)asynchronousResult.AsyncState;
                using (var resp =
(HttpWebResponse)request.EndGetResponse(asynchronousResult))
                {
                    using (var streamResponse = resp.GetResponseStream())
                    {
                        var fourSquareSerializerData = new
DataContractJsonSerializer(typeof(FourSquareUserProfile));
                        var fourSquareProfileData =
fourSquareSerializerData.ReadObject(streamResponse) as FourSquareUserProfile;
                        this.Dispatcher.BeginInvoke(
                            (Action<FourSquareUserProfile>)((FourSquareUserData) =>
                            {
                                App.UserName =
FourSquareUserData.UserResponse.UserData.UserFirstName + " " +
FourSquareUserData.UserResponse.UserData.UserLastName;
                                App.UserImage =
FourSquareUserData.UserResponse.UserData.UserPhoto;
                                App.UserHomeCity =
FourSquareUserData.UserResponse.UserData.UserHomeCity;
                                NavigationService.GoBack();
                            }), fourSquareProfileData);
                        Dispatcher.BeginInvoke(()=>
                        {
                            foreach (var item in
fourSquareProfileData.UserResponse.UserData.UserFriends.UserGroups[1].UserItems)
                            {
                                FriendClass friendClass = new FriendClass();
                                friendClass.FirstName = item.FriendFirstName;
                                friendClass.LastName = item.FriendLastName;
                                friendClass.Photo = item.FriendPhoto;
                                friendClass.HomeCity = item.FriendHomeCity;
                                App.iFriendList.ListClassFriend.Add(friendClass);
                            }
                        });
                    }
                }
            }
            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*

(C) Copyright Nokia 2014. All rights reserved.

Secret that you got from Foursquare Developer account in the project.

Summary

This application shows how to use the Foursquare API to authenticate our application without using any third party library and thus removes dependency. Though I have not tried but there are also .Net libraries for Foursquare, to explore more view `See Also` section.

Related Articles

- Facebook authentication in Windows Phone application
- Google+ authentication in Windows Phone application
- LinkedIn: Authentication and viewing own basic profile

Source Code

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

(C) Copyright Nokia 2014. All rights reserved.