OAuth on Windows Phone

This article explains how to implement oAuth 2.0 on Windows Phone

Introduction

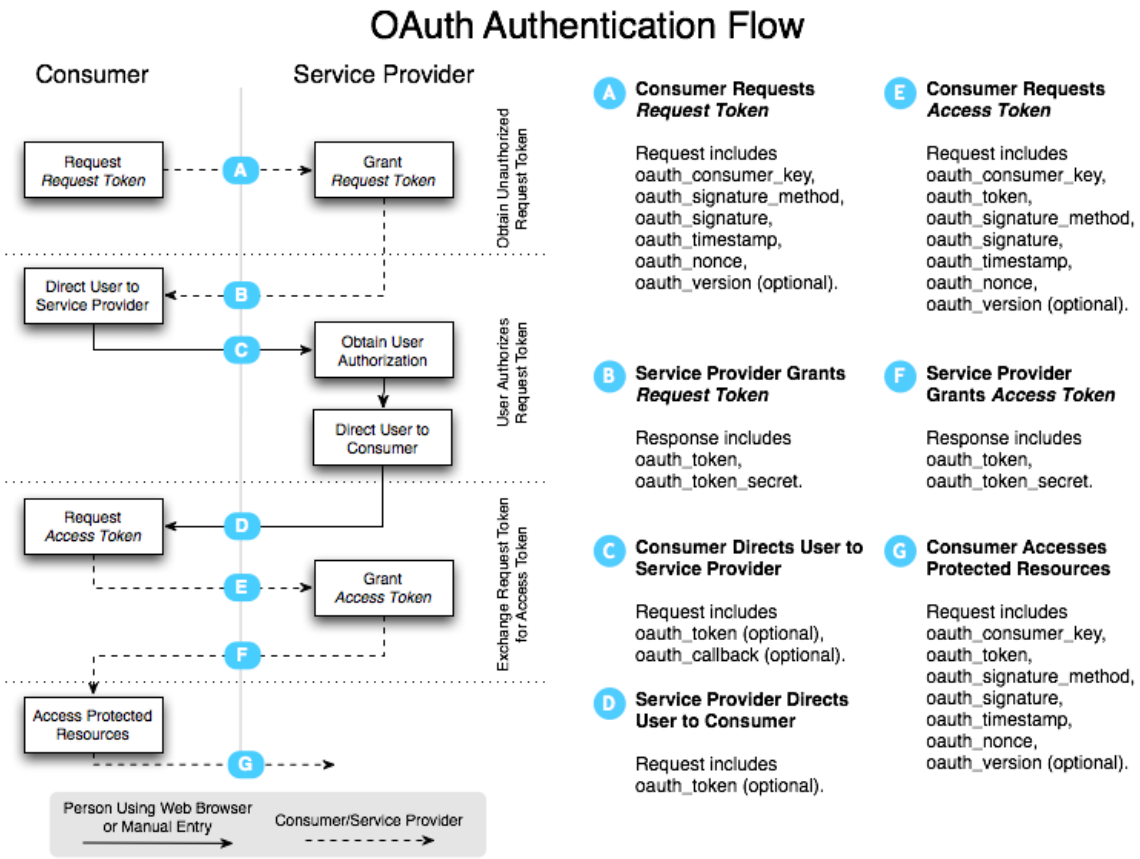
This article explains how oAuth 2.0 workflow which can be implemented on Windows Phone 7/8. For an example in this article Pocket (read it later) API has been used.

Pocket API can be considered as a representative of oAuth API as it has been seen that its same with most of the services. Same example can be used with Twitter. There are few oAuth articles on net showing how to use it on Windows Phone with twitter some reference was taken from them, none of them although explains it with required clarity. There is a trick on Windows Phone that you have to employ while working with oAuth API particularly when service redirects the user back to App implementing oAuth. On other platforms such as Android and iOS there exists a URI scheme with the help of which an app can be run using a URL from browser, unfortunately this scheme doesn't exists on Windows Phone which makes it hard to go back to an app after user authentication from service.

Note: Please obtain your own API KEY in order to get this example working.

oAuth authentication flow

A figure (infographic) below will explain the required workflow in case of oAuth.



A generalized oAuth flow consists of few major steps which are as follows

1. Get API key
2. Obtain request token
3. Perform user authentication on website and redirection to client
4. Convert request token to access token upon successful authentication.

Once client application has an access token further API calls to the service can be made using it.

You can check Pocket (Read it Later) documentation in order to understand how above steps are suggested to be done and more information on related API calls here

(C) Copyright Nokia 2014. All rights reserved.

Solution for Windows Phone oAuth 2.0

The actual solution example is realized in four major steps. Very first thing one needs to do is get an API key from Pocket's developer website. While calling Pocket oAuth API following things must be kept in mind:

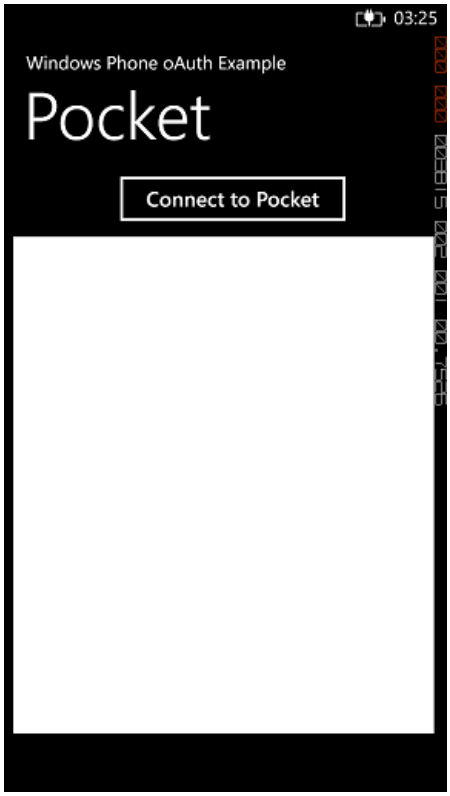
1. All the calls should be made on HTTPS
2. Call can be made in plain text ovet HTTPS or JSON over HTTPS
3. If call was using JSON reply will be in JSON format ony
4. All the API requests are using POST

On most of the services like Twitter, Facebook above points make sense.

- **First step is to make call to an API url which actually returns us a request token**. In order to connect to the service example uses `HttpWebRequest` class API.

To obtain request token call "https://getpocket.com/v3/oauth/request" with API key and redirect url parameters through POST request. `HttpWebRequest` class has `BeginRequestStream` method to get connected with an Internet resource. The method takes a callback method as one of the parameter. This callback method gets called once connection is done with the resource. Inside the callback method then we pass the rest of the parameters using POST method as follows.

User triggers the process by hitting a button / menu




```
System.Uri myUri = new System.Uri("https://getpocket.com/v3/oauth/request");
HttpWebRequest myRequest = (HttpWebRequest)HttpWebRequest.Create(myUri);
myRequest.Method = "POST";
myRequest.ContentType = "application/x-www-form-urlencoded";
myRequest.BeginGetRequestStream(new AsyncCallback(GetRequestStreamCallback), myRequest);
```

The callback method `GetRequestStreamCallback()` passes the parameters as follows

```
HttpWebRequest myRequest = (HttpWebRequest)callbackResult.AsyncState;
// End the stream request operation
Stream postStream = myRequest.EndGetRequestStream(callbackResult);

// Create the post data
string postData = "consumer_key=consumerkey&redirect_uri=http://www.google.com";
```


(C) Copyright Nokia 2014. All rights reserved.

```
byte[] byteArray = Encoding.UTF8.GetBytes(postData);

// Add the post data to the web request
postStream.Write(byteArray, 0, byteArray.Length);
postStream.Close();

// Start the web request
myRequest.BeginGetResponse(new AsyncCallback(GetResponsetStreamCallback), myRequest);
```

Once the parameters are passed to the stream , we call `BeginGetResponse()` method and pass a callback method reference which will be called once response is available. In response to our request service will return as an access token which we collect as follows

```
HttpWebRequest request = (HttpWebRequest)callbackResult.AsyncState;
HttpWebResponse response = (HttpWebResponse)request.EndGetResponse(callbackResult);
StreamReader httpWebStreamReader = new StreamReader(response.GetResponseStream());
string result = httpWebStreamReader.ReadToEnd();
string[] data = result.Split('=');
```

From the response extract the request token as shown above. The response coming from getpocket.com is as follows (example)

```
code=dcba4321-dcba-4321-dcba-4321dc
```

- **Next step is to redirect user to the service providers website for authentication** (getpocket.com in this case), API provides following API request to do it.

```
https://getpocket.com/auth/authorize?
request_token=YOUR_REQUEST_TOKEN&redirect_uri=YOUR_REDIRECT_URI
```

Pass the request token obtained in previous step and a redirect URL which you have provided in earlier call i.e. www.google.com in this case.

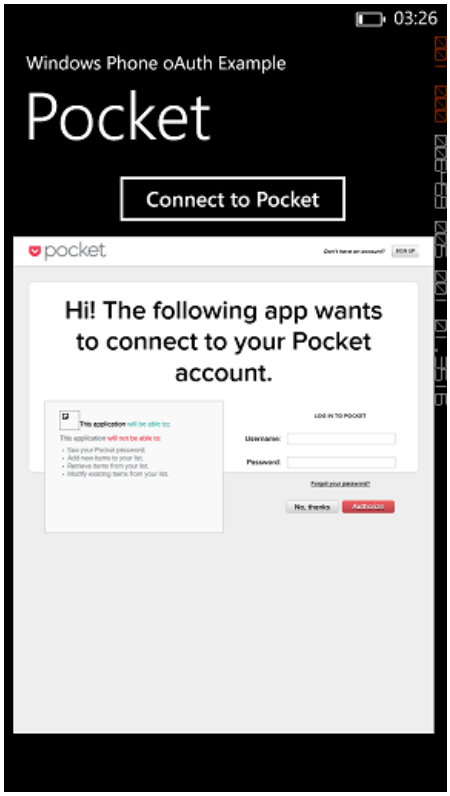
Do the API request from embedded web browser control , as this is necessary , so that when authentication is complete we can catch the event when the browser is getting redirected to www.google.com, this is where we come to know that authentication is over. Remember its only WP where probably we will be excused for using embedded web control, on other platforms all oAuth services like Twitter, Facebook dictates not to use embedded browser and app may get blocked forever if they find that out.

We monitor web browser control's *navigated event* and if the URL detected is Google then we are done with authentication.

```
url = "https://getpocket.com/auth/authorize?request_token=" + data[1] +
"&redirect_uri=http://www.google.com";
Deployment.Current.Dispatcher.BeginInvoke(() =>
{
    // change UI here
    web1.Visibility = System.Windows.Visibility.Visible;
    web1.Navigate(new Uri(url));
});
```

This is how browser control loads the authentication page

(C) Copyright Nokia 2014. All rights reserved.



Here is the code where we monitor where is the browser control being redirected by service.

```
private void web1_Navigated(object sender, NavigationEventArgs e)
{
    if (strCmp(e.Uri.AbsoluteUri,"http://www.google.") == true)
    {
        web1.Visibility = System.Windows.Visibility.Collapsed;
        MessageBox.Show("complete");
        HttpWebRequest myRequest =
(HttpWebRequest)HttpWebRequest.Create("https://getpocket.com/v3/oauth/authorize");
        myRequest.Method = "POST";
        myRequest.ContentType = "application/x-www-form-urlencoded";
        myRequest.BeginGetRequestStream(new AsyncCallback(GetRequestStreamCallback),
myRequest);
    }
}
```

Once the redirection is done we show appropriate message to user about authentication

(C) Copyright Nokia 2014. All rights reserved.

- **Next step is to obtain access token, with the help of request token.** Which can be easily done with the help of following API call

```
https://getpocket.com/v3/oauth/authorize
```

parameters are request token and consumer key, response is access token and user name of the user who authenticated himself. The parameters to this request are passed from `GetRequestStreamCallBack()`

```
HttpWebRequest myRequest = (HttpWebRequest)callbackResult.AsyncState;
// End the stream request operation
Stream postStream = myRequest.EndGetRequestStream(callbackResult);

// Create the post data
string postData = "consumer_key=consumerKey&code="+code;
byte[] byteArray = Encoding.UTF8.GetBytes(postData);

// Add the post data to the web request
postStream.Write(byteArray, 0, byteArray.Length);
postStream.Close();

// Start the web request
myRequest.BeginGetResponse(new AsyncCallback(GetResponsetStreamCallback), myRequest);
```

- Next step is to receive the response and extract the access token , username.

Response is received in following format (example)

```
access_token=5678defg-5678-defg-5678-defg56& username=pocketuser
```

Here is the complete source of **mainpage.xaml.cs** in order to understand the complete flow of authentication process.

```
using System;
using System.Collections.Generic;
```


(C) Copyright Nokia 2014. All rights reserved.

```
using System.Linq;
using System.Net;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Navigation;
using Microsoft.Phone.Controls;
using Microsoft.Phone.Shell;
using test.Resources;

using System.IO;
using System.Text;
using System.Diagnostics;
using Microsoft.Phone.Tasks;


namespace test
{
    public partial class MainPage : PhoneApplicationPage
    {
        private string url;

        private int AUTH = 0;
        private int TOKEN = 1;
        private int TOKEN2 = 2;
        private int count = 0;
        private int state;
        private string code;

        // Constructor
        public MainPage()
        {
            InitializeComponent();

            // Sample code to localize the ApplicationBar
            //BuildLocalizedApplicationBar();
            state = AUTH;
        }
        void GetRequestStreamCallback(IAsyncResult callbackResult)
        {
            if (state == AUTH)
            {
                HttpWebRequest myRequest = (HttpWebRequest)callbackResult.AsyncState;
                // End the stream request operation
                Stream postStream = myRequest.EndGetRequestStream(callbackResult);

                // Create the post data
                string postData =
"consumer_key=cosumerKey&redirect_uri=http://www.google.com";
                byte[] byteArray = Encoding.UTF8.GetBytes(postData);

                // Add the post data to the web request
                postStream.Write(byteArray, 0, byteArray.Length);
                postStream.Close();

                // Start the web request
                myRequest.BeginGetResponse(new
AsyncCallback(GetResponsetStreamCallback), myRequest);
```


(C) Copyright Nokia 2014. All rights reserved.

```
            }
           else if (state == TOKEN)
            {
                HttpWebRequest myRequest = (HttpWebRequest)callbackResult.AsyncState;
                // End the stream request operation
                Stream postStream = myRequest.EndGetRequestStream(callbackResult);

                // Create the post data
                string postData = "consumer_key=consumerKey="+code;
                byte[] byteArray = Encoding.UTF8.GetBytes(postData);

                // Add the post data to the web request
                postStream.Write(byteArray, 0, byteArray.Length);
                postStream.Close();

                // Start the web request
                myRequest.BeginGetResponse(new
AsyncCallback(GetResponsetStreamCallback), myRequest);
            }
        }
        void GetResponsetStreamCallback(IAsyncResult callbackResult)
        {
            if (state == AUTH)
            {
                HttpWebRequest request = (HttpWebRequest)callbackResult.AsyncState;
                HttpWebResponse response =
(HttpWebResponse)request.EndGetResponse(callbackResult);
                if (response.StatusCode == System.Net.HttpStatusCode.OK)
                {
                    Debug.WriteLine("Ok");
                }
                else
                {
                    Debug.WriteLine(response.StatusCode);
                }
                using (StreamReader httpWebStreamReader = new
StreamReader(response.GetResponseStream()))
                {
                    string result = httpWebStreamReader.ReadToEnd();
                    //For debug: show results
                    Debug.WriteLine(result);
                    string[] data = result.Split('=');
                    url = "https://getpocket.com/auth/authorize?request_token=" +
data[1] + "&redirect_uri=http://www.google.com";
                    code = data[1];

                    Deployment.Current.Dispatcher.BeginInvoke(() =>
                    {
                        // change UI here
                        web1.Visibility = System.Windows.Visibility.Visible;
                        web1.Navigate(new Uri(url));
                    });
                }
            }
            else if (state == TOKEN)
            {
```


(C) Copyright Nokia 2014. All rights reserved.

```
                HttpWebRequest request = (HttpWebRequest)callbackResult.AsyncState;
                HttpWebResponse response =
(HttpWebResponse)request.EndGetResponse(callbackResult);
                if (response.StatusCode == System.Net.HttpStatusCode.OK)
                {
                    Debug.WriteLine("Ok1");
                }
                else
                {
                    Debug.WriteLine(response.StatusCode);
                }
                using (StreamReader httpWebStreamReader = new
StreamReader(response.GetResponseStream()))
                {
                    string result = httpWebStreamReader.ReadToEnd();
                    //For debug: show results
                    Debug.WriteLine(result);

                }
            }
        }

        private void Button_Click_1(object sender, RoutedEventArgs e)
        {
            System.Uri myUri = new System.Uri("https://getpocket.com/v3/oauth/request");
            HttpWebRequest myRequest = (HttpWebRequest)HttpWebRequest.Create(myUri);
            myRequest.Method = "POST";
            myRequest.ContentType = "application/x-www-form-urlencoded";
            myRequest.BeginGetRequestStream(new AsyncCallback(GetRequestStreamCallback),
myRequest);

        }

        private void Button_Click_2(object sender, RoutedEventArgs e)
        {
            web1.Navigate(new Uri(url));

        }

        private void web1_Navigated(object sender, NavigationEventArgs e)
        {


            if (strCmp(e.Uri.AbsoluteUri,"http://www.google.") == true)
            {
                web1.Visibility = System.Windows.Visibility.Collapsed;
                MessageBox.Show("complete");
                HttpWebRequest myRequest =
(HttpWebRequest)HttpWebRequest.Create("https://getpocket.com/v3/oauth/authorize");
                myRequest.Method = "POST";
                myRequest.ContentType = "application/x-www-form-urlencoded";
                myRequest.BeginGetRequestStream(new
AsyncCallback(GetRequestStreamCallback), myRequest);
                state = TOKEN;
            }

        }
```


(C) Copyright Nokia 2014. All rights reserved.

```
        private bool strCmp(string a, string b)
        {
            if(a.Length < b.Length)
                return false;
            bool equal = false;
            for (int i = 0; i < b.Length; i++)
            {

                if (a[i] == b[i])
                    equal = true;
                else
                {
                    equal = false;
                    break;
                }

            }

            return equal;
        }


    }
}
```

and complete mainpage.xaml listing

```
<phone:PhoneApplicationPage
    x:Class="test.MainPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:phone="clr-namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone"
    xmlns:shell="clr-namespace:Microsoft.Phone.Shell;assembly=Microsoft.Phone"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d"
    FontFamily="{StaticResource PhoneFontFamilyNormal}"
    FontSize="{StaticResource PhoneFontSizeNormal}"
    Foreground="{StaticResource PhoneForegroundBrush}"
    SupportedOrientations="Portrait" Orientation="Portrait"
    shell:SystemTray.IsVisible="True">

    <!--LayoutRoot is the root grid where all page content is placed-->
    <Grid x:Name="LayoutRoot" Background="Transparent">
        <Grid.RowDefinitions>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="*"/>
        </Grid.RowDefinitions>

        <!-- LOCALIZATION NOTE:
            To localize the displayed strings copy their values to appropriately named
            keys in the app's neutral language resource file (AppResources.resx) then
            replace the hard-coded text value between the attributes' quotation marks
            with the binding clause whose path points to that string name.

            For example:
```


(C) Copyright Nokia 2014. All rights reserved.

```
                Text="{Binding Path=LocalizedResources.ApplicationTitle,
Source={StaticResource LocalizedStrings}}"

            This binding points to the template's string resource named
"ApplicationTitle".

            Adding supported languages in the Project Properties tab will create a
            new resx file per language that can carry the translated values of your
            UI strings. The binding in these examples will cause the value of the
            attributes to be drawn from the .resx file that matches the
            CurrentUICulture of the app at run time.
         -->

        <!--TitlePanel contains the name of the application and page title-->
        <StackPanel x:Name="TitlePanel" Grid.Row="0" Margin="12,17,0,28">
            <TextBlock Text="MY APPLICATION" Style="{StaticResource
PhoneTextNormalStyle}" Margin="12,0"/>
            <TextBlock Text="page name" Margin="9,-7,0,0" Style="{StaticResource
PhoneTextTitle1Style}"/>
        </StackPanel>

        <!--ContentPanel - place additional content here-->
        <Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">
            <Button Content="Login" HorizontalAlignment="Left" Margin="130,0,0,0"
VerticalAlignment="Top" Click="Button_Click_1"/>
            <phone:WebBrowser Name="web1" HorizontalAlignment="Left" Margin="10,77,0,0"
VerticalAlignment="Top" Height="505" Width="436" Navigated="web1_Navigated"
Navigating="web1_Navigating"/>

        </Grid>

        <!--Uncomment to see an alignment grid to help ensure your controls are
            aligned on common boundaries.  The image has a top margin of -32px to
            account for the System Tray. Set this to 0 (or remove the margin altogether)
            if the System Tray is hidden.

            Before shipping remove this XAML and the image itself.-->
        <!--<Image Source="/Assets/AlignmentGrid.png" VerticalAlignment="Top"
Height="800" Width="480" Margin="0,-32,0,0" Grid.Row="0" Grid.RowSpan="2"
IsHitTestVisible="False" />-->
    </Grid>

</phone:PhoneApplicationPage>
```

Summary

This article explains how to implement oAuth 2.0 using Pocket API , even though the code is specific to Pocket but most of the code is reusable with Twitter / Facebook and any other oAuth 2.0 API.

(C) Copyright Nokia 2014. All rights reserved.

(C) Copyright Nokia 2014. All rights reserved.