

Vk Api For Mobile 2.0

Changes:

Added support for authorization using official Vk mobile application

Added support for WP8

Changed the way of making requests: now instead of one global handler for all results you can specify custom handler for each request

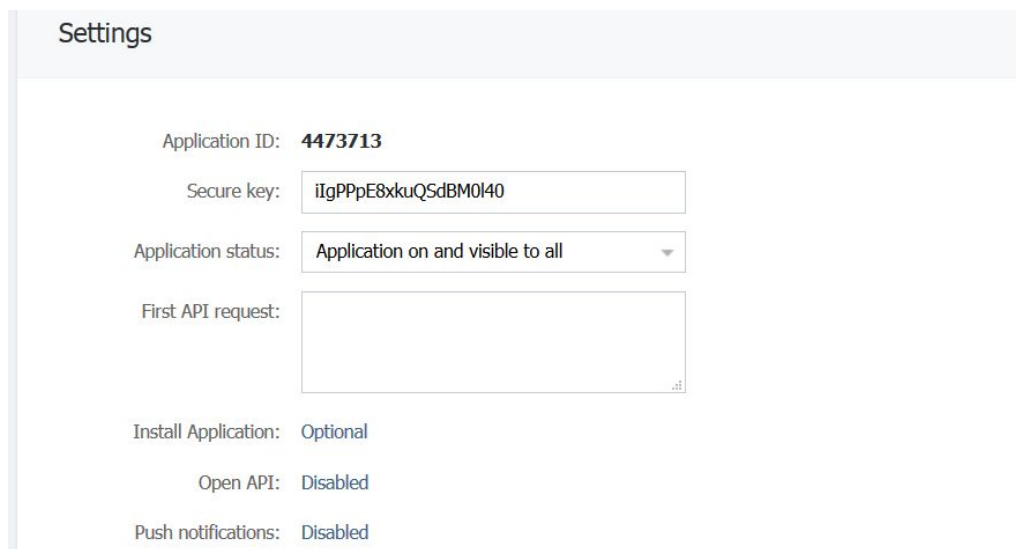
Added predefined models for vk data parsing

Changed json parser to one more stable with iOS MiniJSON

Getting Started

First, need to set up you vk application on vk.com/dev site

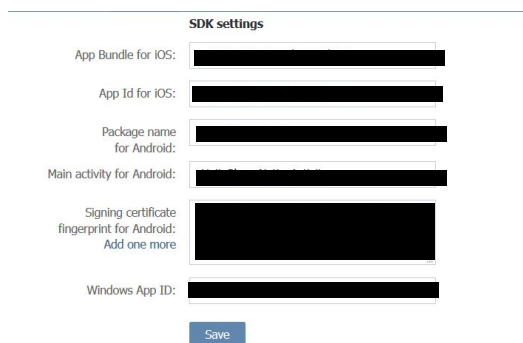
Then go to your setting page and copy somewhere Application id, we will need it later



The screenshot shows the 'Settings' page for a VK application. It includes the following fields and options:

- Application ID: **4473713**
- Secure key:
- Application status:
- First API request:
- Install Application:
- Open API:
- Push notifications:

The same page in SDK setting section you will also need to feel all the entries relative to platform you want to use. Can be left blank if you won't use authorization trough vk mobile app



The screenshot shows the 'SDK settings' page with the following fields:

- App Bundle for iOS:
- App Id for iOS:
- Package name for Android:
- Main activity for Android:
- Signing certificate fingerprint for Android: [Add one more](#)
- Windows App ID:

A 'Save' button is located at the bottom.

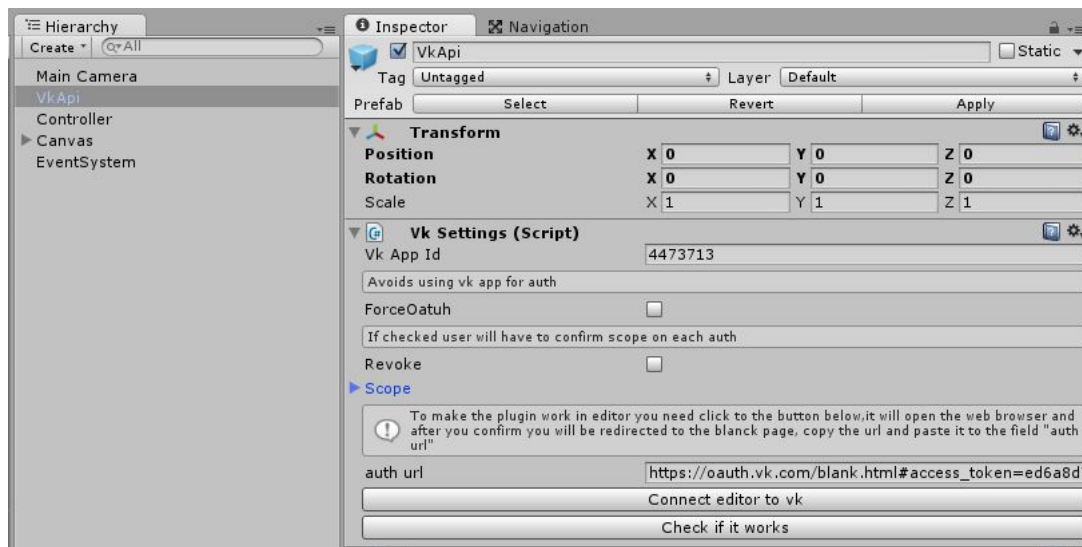
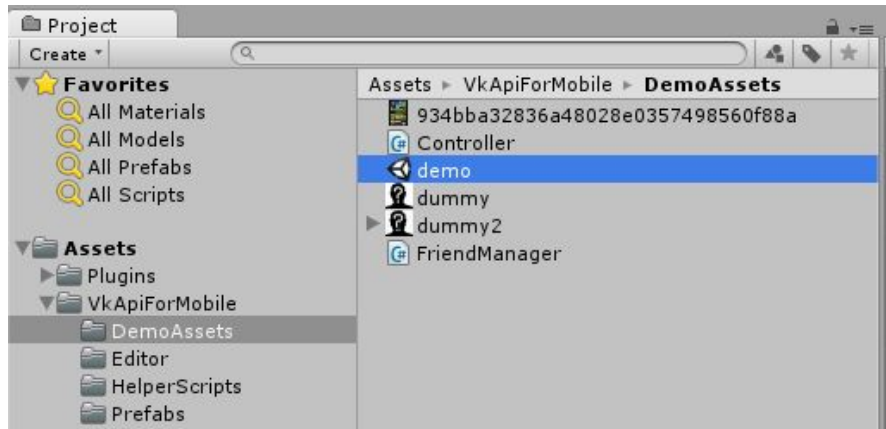
more info here <http://vk.com/dev/SDK>

!!Only part relative to setup on vk site. Don't modify any project files.All handled automatically.

Now we are ready to open Unity.

I will use Unity 5 but it is also compatible with 4.6.5+ version

Open demo scene



First you need to enter app id(from vk web site)

Check if needed **ForceOauth**(this will force you mobile application to use web view for authorization even if vk app is present on device)

Revoke if checked will ask to confirm access permissions each time you want to log in.

VKSdk does not provide a method for logging out on iOS. So a simple trick can be done. Get the reference of VKSetting in you controller script, and simply set up this two variable to true (**ForceOauth** and **Revoke**) So the next time you call login method user will be able to logout and login with another account.

Next toggle **scope** section and select permissions you need

If you want to be able to test you app in Editor you need to click Connect editor to vk. This will redirect you to vk web page, confirm permissions and copy the url to auth url field.

Hit “Check if it works” - this will open web browser, and if you see json with you name, everything is set up and you are ready to work with vk.

Let's look to the api

You can open controller script.

In every script using the api for vk you will need to import these namespaces.

```
5
6 using Facebook.MiniJSON;
7 using com.playGenesis.VkUnityPlugin;
8
```

Now let's look at variable the **VkApi vkapi** object. It is the one responsible for calling api methods.

Downloader d is the script I wrote to help download stuff using coroutines.

And **List<VKUser> friends** is the list that will be holding friends objects

Vkapi api is singleton. so you need to have only one in you scene.

And we can get it by calling static method **VkApi.VkApiInstance**.

Vkapi has an event **LoggedIn**. We can subscribe to this event and start calling vk methods

```
9 public class Controller : MonoBehaviour {
10     public VkApi vkapi;
11     public Downloader d;
12     public List<VKUser> friends=new List<VKUser>();
13     // Use this for initialization
14
15     void Start () {
16         vkapi=VkApi.VkApiInstance;
17         vkapi.LoggedIn+=()=>
18         {
19             StartWorkingWithVk();
20         };
21     }
```

After we've logged in, method **StartWorkingWithVk** is called which calls method

Get3FriendsDataFromVk -this method actually calls vk api

First we need to form request string, this string is formatted as if it would be a url without beginning part.

So you start with method name “method_name?parameter1=value1¶meter2=value2” and so on. At the end of each request you need to put

“&v= VK_API_VERSION&access_token="+VkApi.currentToken.access_token;

Replace **VK_API_VERSION** with version number you want to use

You can read more here http://vk.com/dev/api_requests

Actual call to vk server is triggered by this method

vkapi.Call(string request, Action< VkResponseRaw , object[] > handler , Object[] data)

String **request** is the string we formed previously

handler is function responsible for handling result from vk server

And **data** is an array of custom objects (relative to this particular request) you want to pass to a handler

In this example I pass integer “attempt”, so in my handler function I get this value and if I got an error from vk server I can recall the method, incrementing this variable by 1.

It is important to limit attempts

```
31
32     public void StartWorkingWithVk()
33     {
34         Get3FriendsDataFromVk(0);
35     }
36
37     public void Get3FriendsDataFromVk(int attempt)
38     {
39         var r="friends.get?user_id="+VkApi.currentToken.user_id+"&count=3&fields=photo_200&v=5.29&access_token="+VkApi
40         vkapi.Call(r, OnGet3FriendsCompleted, new object[] { attempt });
41         //vkapi.call принимает 3 параметра строку запроса, функцию обработчика запроса,
42         //и массив объектов (можно передать любые объекты, их можно использовать в обработчике
43         //например, можно реализовать повторные запросы при неудаче как здесь.
44     }
45
46
```

Let's look closer at handler function **arg1** contains all info about result of execution vk api method and **arg2** is that custom array of data we've passed previously.

First we handle errors

```
void OnGet5FriendsCompleted (VkResponseRaw arg1, object[] arg2)
{
    //проверяем на ошибки
    if (arg1.ei != null)
    {
        //если ошибка, помним что мы передали номер запроса, можно повто
        var attempt=(int)arg2[0];
        if(attempt<5)
        {
            Get3FriendsDataFromVk(attempt+1);
        }
        return;
    }
}
```

Next code in this function is executed only if no errors happened and we can proceed with deserialisation

MiniJSON example

```
{
  "response": {
    "count": 717,
    "items": [
      {
        "id": 46,
        "first_name": "Andrey",
        "last_name": "Lesokhin",
        "domain": "lesokhin",
        "city": {
          "id": 2,
          "title": "Saint Petersburg"
        },
        "online": 1
      },
      {
        "id": 11191,
        "first_name": "Andrey",
        "last_name": "Lopatin",
        "domain": "kotehok",
        "city": {
          "id": 2,
          "title": "Saint Petersburg"
        },
        "online": 0
      },
      {
        "id": 172823,
        "first_name": "Andrey",
        "last_name": "Melnik",
        "domain": "a.melnik",
        "city": {
          "id": 2,
          "title": "Saint Petersburg"
        },
        "online": 1
      }
    ]
  }
}
```

A little note MiniJSON parses integers as longs and floats as doubles

So, if you need to get an integer, for example, you do it like this:

Example json {"myint":3}
long myint=(long)dict["myint"]

You can think about json as a dictionary with keys and values. Keys are always strings and values can be ints, floats, strings or objects or lists of objects.

"response" is the key followed by ":" and then the value, if value is enclosed in "{}" this means that it is an object with its own keys and values inside.

So how do we deserialise it with miniJSON?

First we create

```
var dict=Json.Deserialize(json) as Dictionary<string,object>;
```

Now let's get value of "response"

I know the "response" is an object (remember enclosed in {})

```
var response=(object)dict["response"];
```

Inside response object I see field "items", it is an array (enclosed in [])

Each element of this array enclosed in {}, so items is an array of objects

Let's get all these objects

```
var items=(List<object>)response["items"];
```

It is pretty time consuming, but once you've reached any **vk type** it gets easy. In this example each item in items is a **VKUser** object. I've already defined these C# classes for you. They have static Deserialise function, that does all the work for you. Here I'll use linq

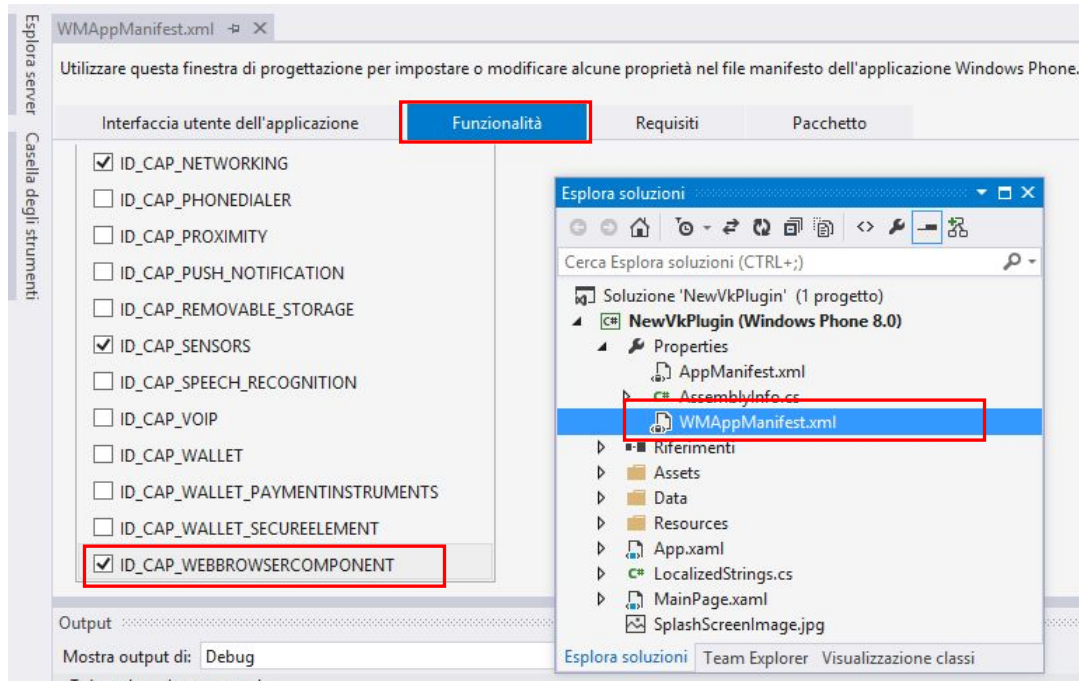
```
List<VKUser> friends=new List<VKUser>();
items.ForEach(i=>friends.add(VKUser.Deserialise(i));
```

Or you can do like this

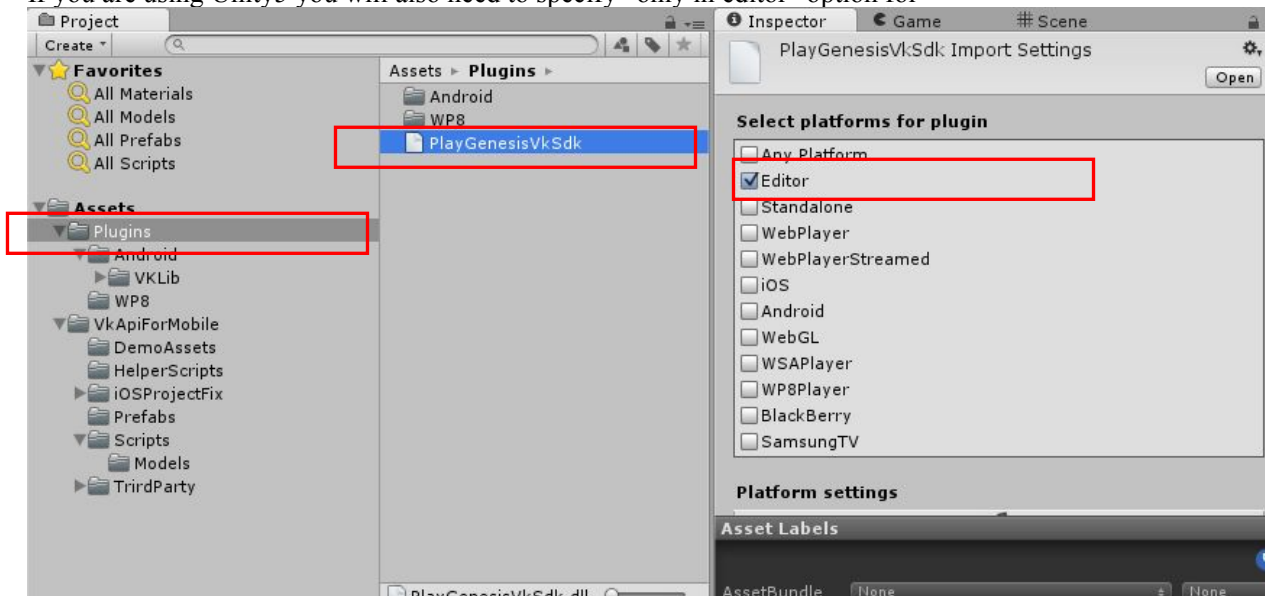
```
foreach(var item in items)
{
    friends.Add(VKUser.Deserialize(item));
}
```

For WP8 builds some extra work needed

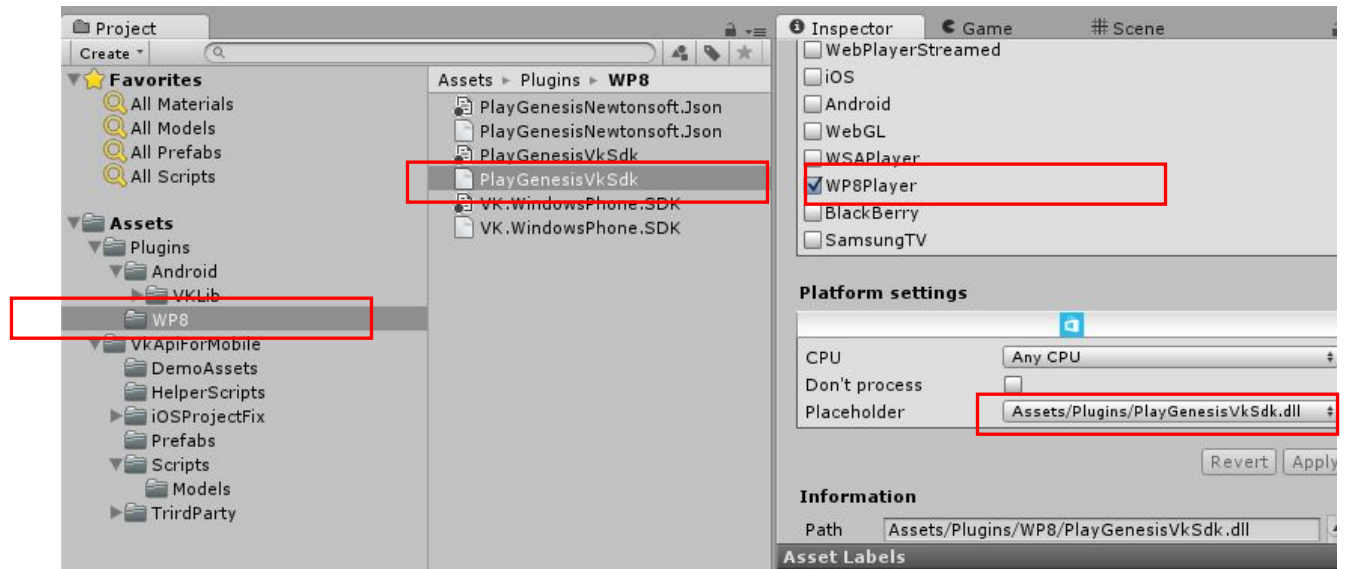
When you finished building your project open it inside visual studio and add
id_cap_webbrowsercomponent permission to WMapManifest.xml file



If you are using Unity5 you will also need to specify “only in editor” option for



And also



Now run demo scene if it works in editor it will work on iOS, Android and WP8

Note, in this plugin I'm using facebook implementation of miniJSON contained in IFacebook.dll. If you will import facebook sdk, there is a possibility that another IFacebook.dll will be added to you project - delete one.

The other thing to keep in mind when you are building for iOS is that there some custom .h and .m files added to you project in postbuild script, but they are referenced, so if you build on your windows machine and then move xcode project to mac, it will be missing these files. Another warning about you ios build that window sometimes breaks .framework files, so if xcode will complain about not being able to resolve #import "VKSDK/VKSdk.h", just go to the github download and replace VKSdk.framework from official vk sdk for ios.

If you need some help, write me an email at vitaly.korobchuk@gmail.com

Thank you.