

VK Standalone SDK for Unity

SDK Documentation (v.0.4.2)

Содержание:

1. [Введение](#);
2. [Установка SDK и необходимых компонентов](#);
3. [Начало работы](#);
4. [Авторизация](#);
5. [Работа с VK API](#);
6. [В заключении](#);

Требования к работе с SDK:

Для работы с VK Standalone Unity SDK вам потребуется:

- 1) **Unity 5.6** или выше;
- 2) **.Net Framework 4.5+**;
- 3) Созданное **Standalone** приложение в социальной сети **Вконтакте** (<https://vk.com/editapp?act=create>).

VK SDK не поддерживает сборку через IL2CPP!

Поддерживаемые ОС:

В настоящий момент VK Standalone SDK for Unity поддерживает следующие платформы:

- Windows 7 и выше (x86 и x64);
- MacOS X;

Зависимости:

Для организации работы **WebView** в авторизации используется сторонний модуль (<https://github.com/gree/unity-webview>) для **Mac** и **WWebView** (<https://github.com/icodes-studio/WWebView>) для **Windows**.

Поддержка SDK:

По всем вопросам в работе SDK или найденным ошибкам, рекомендуем вам писать на почту: help@cdbits.net

1. Введение

VK Standalone SDK позволяет авторизоваться и использовать **API Вконтакте** для приложений и игр, разработанных на Unity под **Windows** и **macOS**.

Для начала работы с **VK Standalone SDK под Unity**, необходимо произвести первоначальную [настройку вашего проекта](#).

На данный момент, SDK поддерживает работу с **Windows 7 и выше**, а также **macOS X**.

2. Установка SDK и настройка необходимых компонентов

2.1. Установка SDK

Для работы с **SDK** необходимо добавить его в свой проект. Вы можете сделать это, распаковав все необходимые файлы в архиве **VKSDK.zip** в папку **/Assets/** вашего проекта.

Вы также можете открыть Демо-проект и ознакомиться с принципом работы с VK SDK для Unity.

Структура проекта:

/VKSDK/ - Основная директория SDK.

/VKSDK/Demo/ - Здесь расположены файлы, связанные с Демо.

/VKSDK/Demo/Scenes/ - Здесь расположены сцены Демо-игры.

/VKSDK/Documentation/ - Здесь расположена документация.

/VKSDK/Resources/ - Здесь расположены доп. ресурсы.

/VKSDK/Scripts/ - Здесь расположены скрипты самого **VK SDK**.

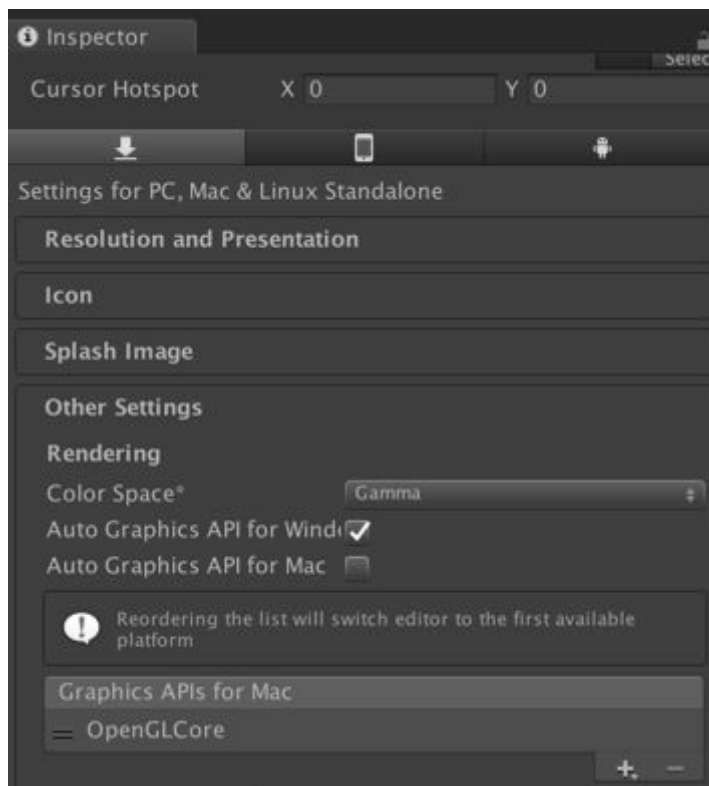
/Plugins/ - Здесь расположены Third-Party плагины, необходимые для работы WebView на различных платформах. **Не удаляйте и не изменяйте данные в этой папке, поскольку это может привести к потере работоспособности SDK.**

2.2. Настройка проекта

Перед началом работы следует произвести дополнительную настройку проекта, чтобы **SDK корректно работал**.

Graphics API для MacOS:

Для корректной работы на **MacOS** необходимо **отключить поддержку Mental Renderer** в настройках проекта. Также, если вы работаете в Unity на Mac, возможно потребуется отключить поддержку **Mental Editor Support**.

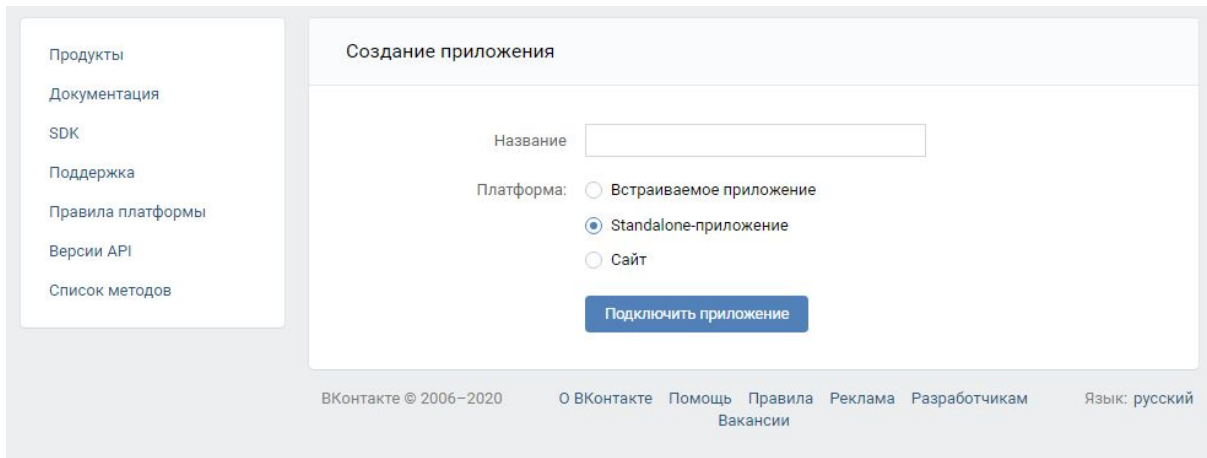


Для этого перейдите в настройки проекта ("**Edit => Project Settings => Player**"), выберите настройки для **Standalone** приложения и в графе **Auto Graphics API for Mac** уберите флажок. Также удалите **Mental** из списка **Graphics APIs for Mac**.

3. Начало работы

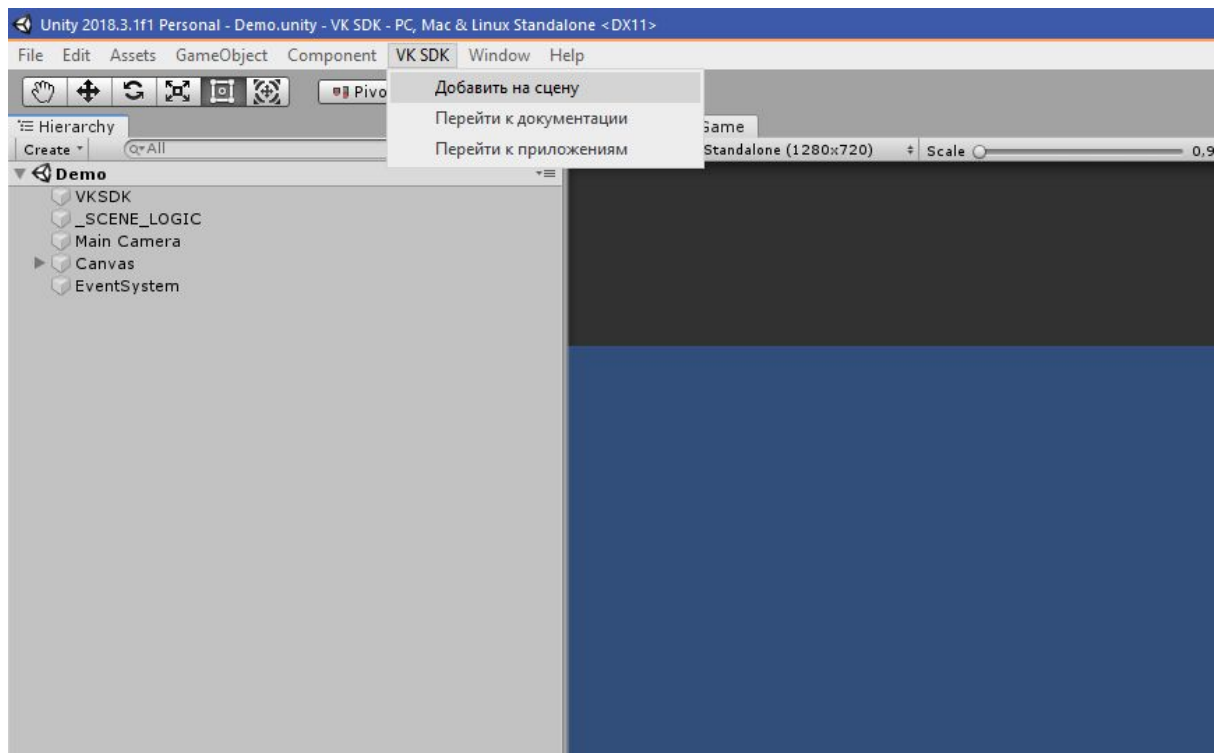
3.1. Создание приложения

Прежде чем начать работу с **VK Unity SDK для Standalone** приложений, вам необходимо [создать приложение внутри социальной сети вконтакте](#) и указать тип **Standalone-приложение** (как показано на скриноште ниже):

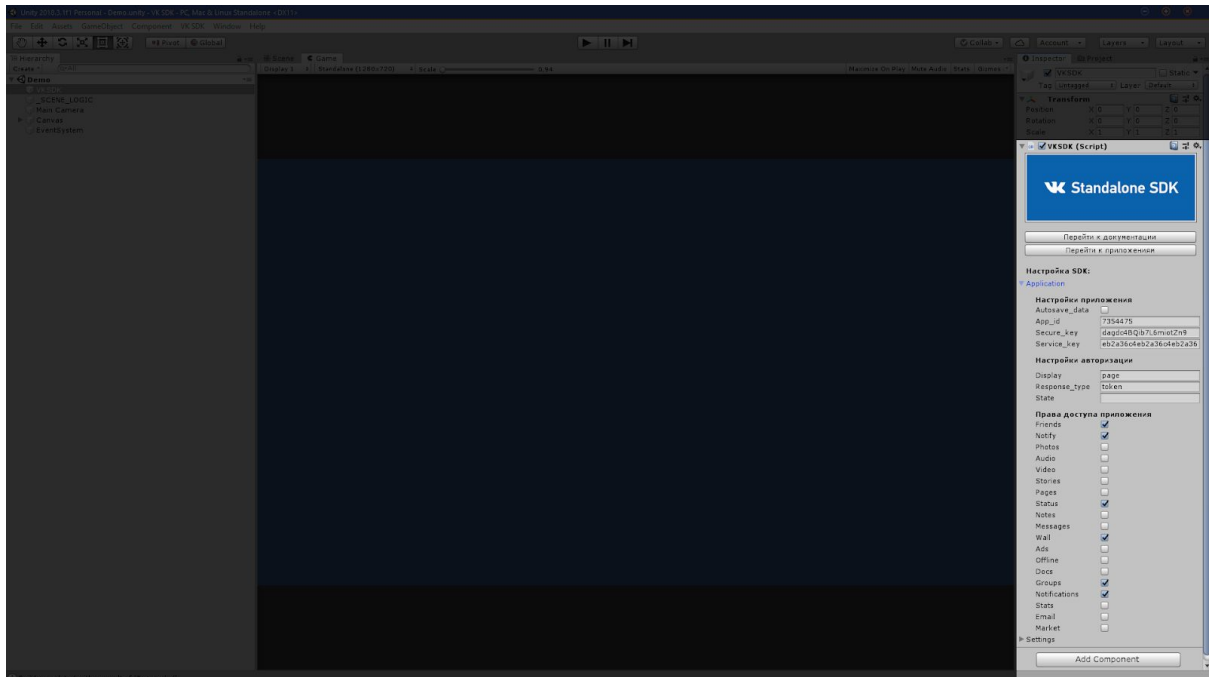


The screenshot shows the 'Создание приложения' (Create application) page on the VK website. On the left is a sidebar menu with links: 'Продукты', 'Документация', 'SDK', 'Поддержка', 'Правила платформы', 'Версии API', and 'Список методов'. The main area is titled 'Создание приложения' and contains a form. It has a 'Название' (Name) text input field. Below it, the 'Платформа' (Platform) section has three radio button options: 'Встраиваемое приложение' (Embedded application), 'Standalone-приложение' (Standalone application), and 'Сайт' (Site). The 'Standalone-приложение' option is selected. At the bottom of the form is a blue button labeled 'Подключить приложение' (Connect application). The footer of the page includes copyright information 'ВКонтакте © 2006–2020', links to 'О ВКонтакте', 'Помощь', 'Правила', 'Реклама', 'Разработчикам', and 'Вакансии', and a language selector 'Язык: русский'.

После чего вы можете перейти в Unity для настройки вашего проекта. После того, как вы [импортировали все нужные файлы](#), в верхнем меню редактора Unity появится новый пункт **“VK SDK”**. Выберите его и нажмите **“Добавить на сцену”**.



Теперь в менеджере сцены появится новый объект с именем **"VKSDK"**. Если вы работаете с Демо-проектом, **добавлять новый объект не нужно**. Выберите уже существующий и перейдите к его настройкам в инспекторе Unity:



Там вы сможете найти все самые важные настройки для вашего приложения.

Самые важные параметры, которые следует указать - **App_id (ID приложения)**, а также при необходимости **Secure_Key (Защищённый ключ)** и **Service_Key (Сервисный ключ доступа)**. Вы сможете найти их в настройках вашего приложения вконтакте.

The image shows a form for VK application settings. It includes fields for 'ID приложения' (7354475), 'Защищённый ключ' (Secure key), 'Сервисный ключ доступа' (Service key), 'Состояние' (Status) set to 'Приложение включено и видно всем' (Application is on and visible to everyone), and 'Первый запрос к API' (First API request).

Также выделите флажками те права доступа, которые хотите использовать в приложении. Подробнее об этом [вы можете прочитать здесь](#).

Работа с объектом VKSDK.

Объект **VKSDK** представляет собой обычный **GameObject**, с компонентом **VKSDK**, наследуемым от **MonoBehaviour**. Поэтому он поддерживает все его возможности.

К примеру, для вызова методов API можно вызвать метод **Call()** у объекта **VKSDK**. Данный метод поддерживает обратные вызовы и работу с ним мы рассмотрим ниже.

У данного объекта имеются следующие параметры:

Параметр	Тип	Описание
application	Object / ApplicationSettingsModel	Используется для хранения настроек приложения
settings	Object / SDKSettingsModel	Используется для хранения настроек SDK
authenticaion	Object / AuthenticationModel	Используется для хранения данных авторизации

При инициализации сцены, объект **VKSDK** пытается загрузить настройки SDK, приложения и данные авторизации с устройства пользователя в зашифрованном формате и десериализовать их в параметры объекта. Если их нет - используются настройки по-умолчанию (указанные вами в инспекторе).

После чего, объект инициализирует **WebView** для дальнейшей работы с авторизацией.

4. Авторизация

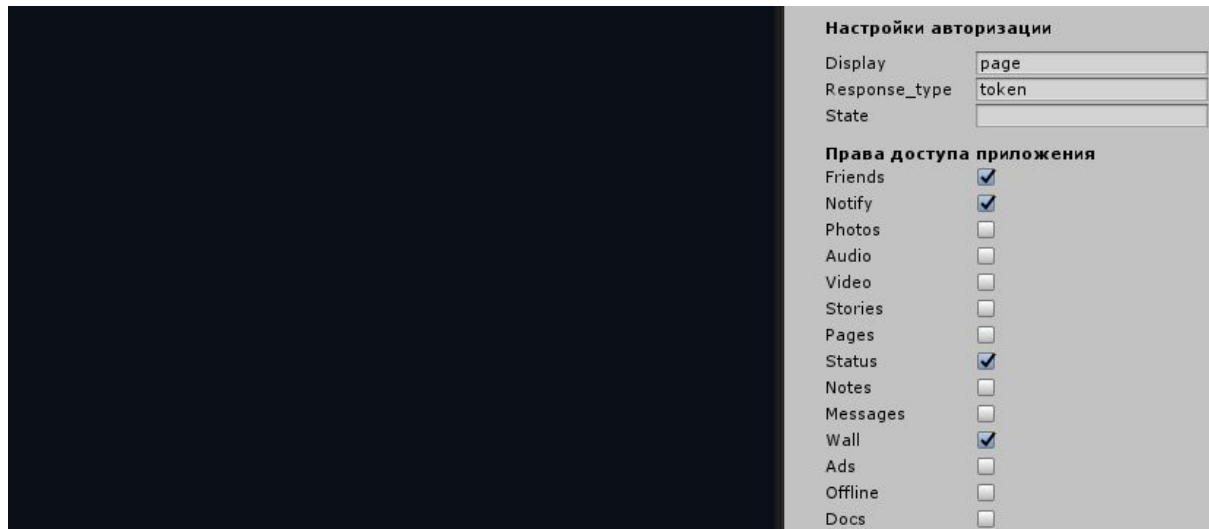
VK Standalone Unity SDK использует метод авторизации **Implicit flow**

(подробнее об авторизации можно прочитать здесь:

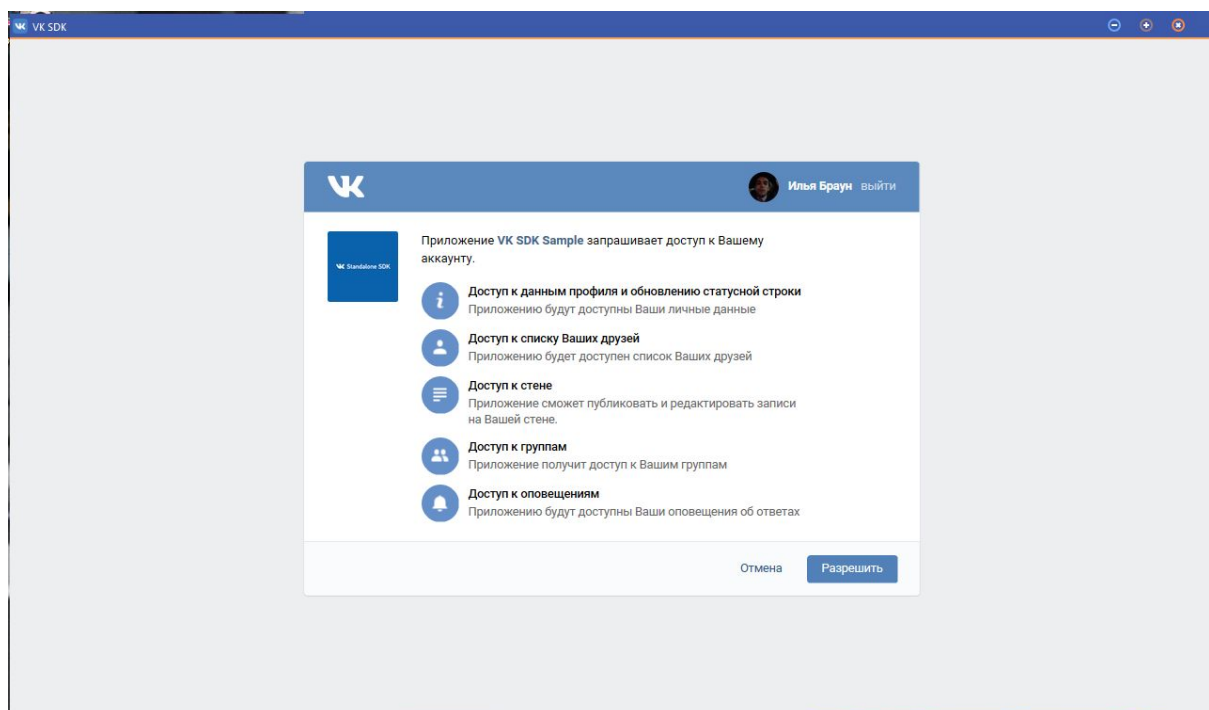
https://vk.com/dev/access_token). Для открытия окна авторизации на **Mac** используется модуль **unity-webview**, а на **Windows** - модуль **WWebView**.

Теперь, рассмотрим сам процесс авторизации в Unity через **вконтакте**:

Для вызова окна авторизации, используйте метод **VKSDK.ShowLoginWindow()**. Данный метод не имеет параметров, а настройки авторизации могут быть найдены в инспекторе объекта **VK SDK**.



При инициализации метода **VKSDK.ShowLoginWindow()** открывается WebView с URL авторизации, где пользователь проходит процедуру, [согласно схеме Implicit Flow](#), после чего окно закрывается, а в параметры объекта **VKSDK.authentication** передается **access_token**, **expires_in** и **user_id**.



Если вам требуется постоянный токен доступа, сделайте активным флаг **"offline"** в правах доступа приложения и **"Autosave_data"** в настройках

приложения в инспекторе. Тогда **access_token** будет сохраняться в зашифрованный файл и его **не требуется повторно получать**, используя авторизацию для вызова методов вконтакте даже после перезапуска приложения.

```
// Authenticate
public void Authenticate(){
    VK.OnAuthenticationComplete += CompleteAuthentication;
    VK.ShowLoginWindow(); // Authenticate
}
```

После того, как вы произвели авторизацию - вы можете вызывать методы VK API, используя **VKSDK.Call()**.

Подробнее о том, как взаимодействовать с API вконтакте - вы можете посмотреть в примере кода в Демо-проекте.

5. Работа с VK API

Для вызова методов **VK API** требуется получить **Access Token** для пользователя. [Здесь вы можете прочитать, как сделать это.](#)

Чтобы вызвать метод API - следует обратиться к методу **VKSDK.Call()**, как показано на примере ниже:

```
// VK API Test
private void _testVKAPI(){
    // Create Form Data
    WWWForm _data = new WWWForm(); // Create WWWForm
    _data.AddField("owner_id", 146332935); // Add Field "owner_id"

    // Call VK API Method
    VK.Call("wall.get", _data, (string data) => { // OK
        Debug.Log(data); // Returns string with JSON response
    }, (BaseErrorModel error) => { // Error
        Debug.Log("VK API Error: " + error.error_msg); // Returns Error Message
    });
}
```

Когда вы вызываете методы **VK API**, вам не нужно подставлять такие параметры, как **access_token** и **v**. Они автоматически подставляются из параметров объекта **VKSDK**. Эти параметры можно настроить в инспекторе Unity.

Метод VKSDK.Call включает в себя следующие параметры:

Параметр	Тип	Описание
method	string	Имя метода. Например: wall.post
data	WWWForm	Параметры запроса
complete	Callback Function / Delegate	Функция, вызываемая при успешном вызове метода. Содержит параметр data (string), включающий в себя JSON-ответ сервера.
error	Callback Function / Delegate	Функция, вызываемая при ошибке. Содержит в себе данные об ошибке в виде объекта, созданного из класса BaseErrorModel

6. В заключении

Для работы с **VK API** в большинстве случаев достаточно двух методов **VKSDK**. Если вы хотите изучить пример работы с **VKSDK** - [загрузите Демо-проект](#).