**VK Standalone SDK for Unity**

SDK Documentation (v.0.4.2)

**Содержание:**

1. [Введение](#lxg8t8ek2bc7);

2. [Установка SDK и необходимых компонентов](#esp5hmmscdox);

3. [Начало работы](#vr0u5bm0zavx);

4. [Авторизация](#7t07qhkxkmq1);

5. [Работа с VK API](#mrhlk1bwwli);

6. [В заключении](#cw6s29o2vsi8);

**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>).

**Поддерживаемые ОС:**

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

- Windows 7 и выше (x86 и x64);

- MacOS X;

**Поддержка SDK:**

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

**1. Введение**

**VK Standalone SDK** позволяет авторизироваться 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](#esp5hmmscdox).

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 Demo-проект и ознакомиться с принципом работы с VK SDK для Unity.**

**Структура проекта:**

**/VKSDK/** - Основная директория SDK.

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

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

**/VKSDK/Documentation/** - Здесь расположена документация.

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

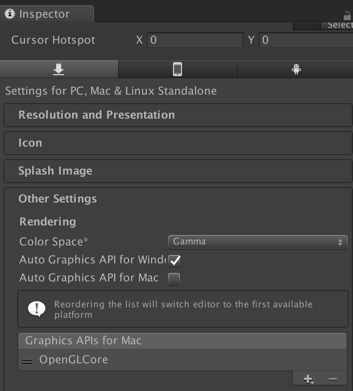
**/VKSDK/Scripts/** - Здесь расположены скрипты самого **VK SDK**.

**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**.

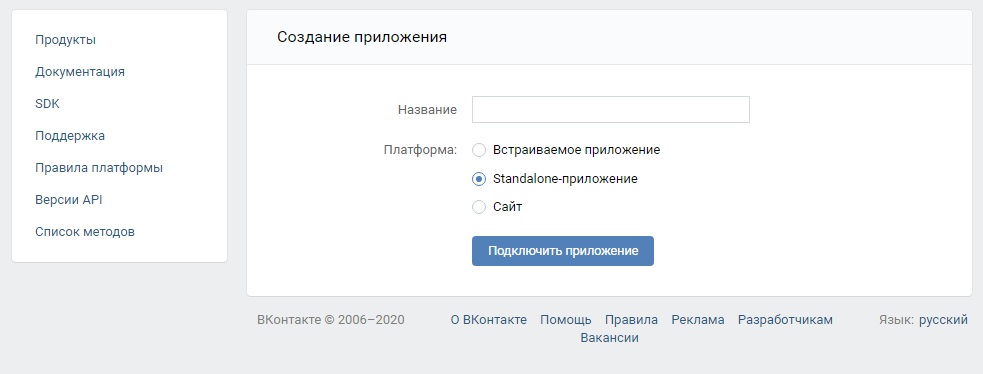
****

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**.

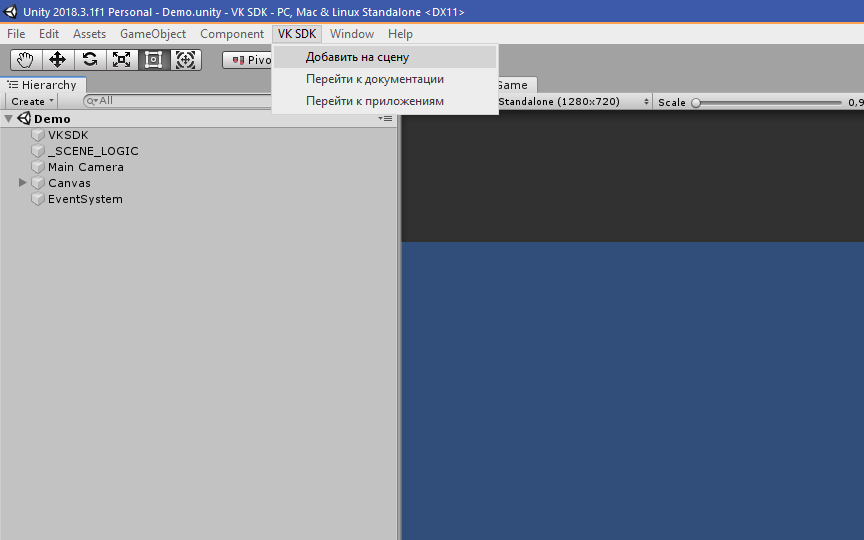
**3. Начало работы**

**3.1. Создание приложения**

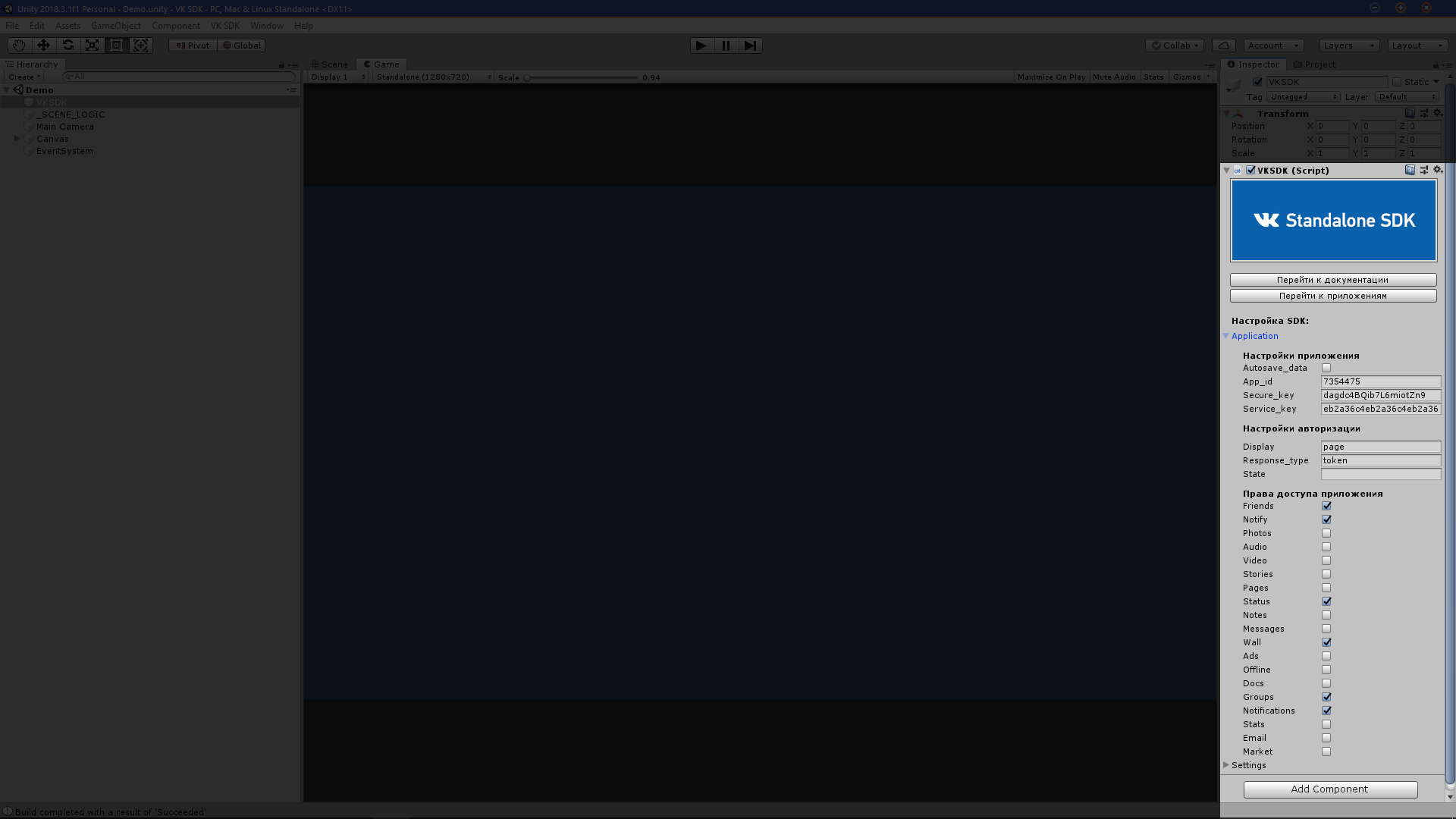
Прежде чем начать работу с **VK Unity SDK для Standalone** приложений, вам необходимо [создать приложение внутри социальной сети вконтакте](https://vk.com/editapp?act=create) и указать тип **Standalone-приложение** (как показано на скриноште ниже):



После чего вы можете перейти в Unity для настройки вашего проекта. После того, как вы [импортировали все нужные файлы](#esp5hmmscdox), в верхнем меню редактора Unity появится новый пункт **“VK SDK”**. Выберите его и нажмите **“Добавить на сцену”**.

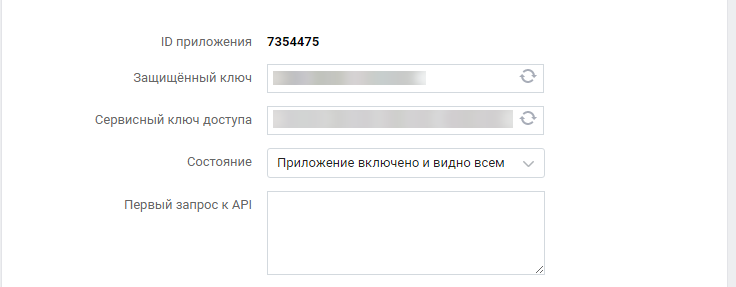


Теперь в менеджере сцены появится новый объект с именем **“VKSDK”**. Если вы работаете с Demo-проектом, **добавлять новый объект не нужно**. Выберите уже существующий и перейдите к его настройкам **в инспекторе Unity**:



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.



Также выделите флажками те права доступа, которые хотите использовать в приложении. Подробнее об этом [вы можете прочитать здесь](https://vk.com/dev/permissions).

**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 |

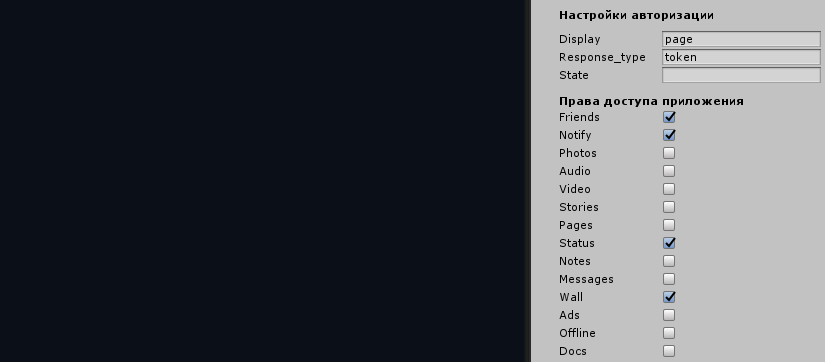
При инициализации сцены, объект **VKSDK** пытается загрузить настройки SDK, приложения и данные авторизации с устройства пользователя в зашифрованном формате и десериализовать их в параметры объекта. Если их нет - используются настройки по-умолчанию (указанные вами в инспекторе).

**4. Авторизация**

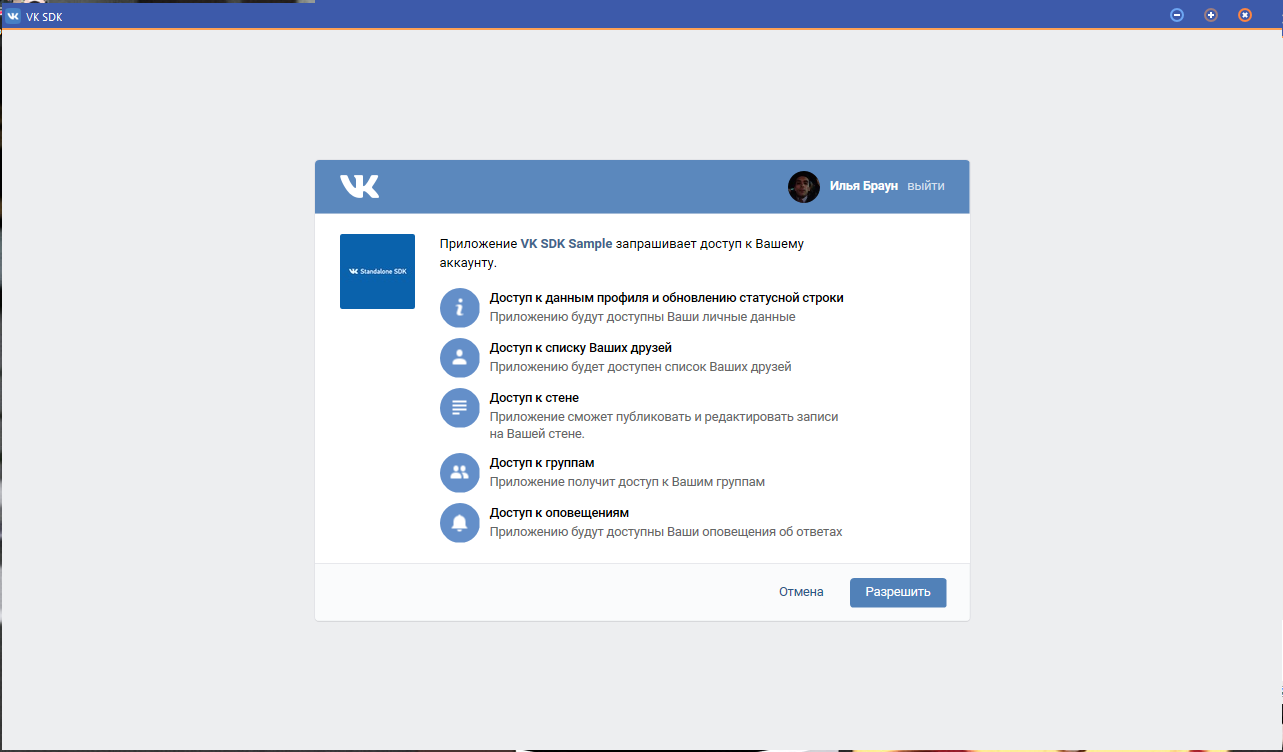
**VK Standalone Unity SDK** использует метод авторизации **Implicit flow** (подробнее об авторизации можно прочитать здесь: <https://vk.com/dev/access_token>). Для открытия окна авторизации используется браузер пользователя по-умолчанию. Для отслеживания авторизации - клиент отправляет запрос к серверу вконтакте с заданным в настройках SDK интервалом.

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

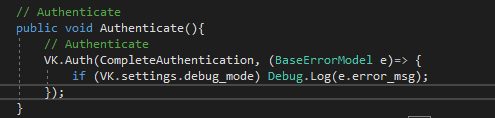
Для вызова окна авторизации, используйте метод **VKSDK.Auth()**. Данный метод не имеет параметров, однако обладает возможностью захвата обратного вызова (callback), а настройки авторизации могут быть найдены в инспекторе объекта **VK SDK**.



При инициализации метода **VKSDK.Auth()** открывается стандартный браузер с URL авторизации, где пользователь проходит процедуру, [согласно схеме Implicit Flow](https://vk.com/dev/implicit_flow_user), 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.



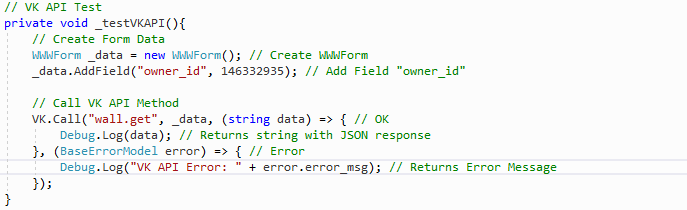
После того, как вы произвели авторизацию - вы можете вызывать методы VK API, используя **VKSDK.Call()**.

*Подробнее о том, как взаимодействовать с API вконтакте - вы можете просмотреть в примере кода в Demo-проекте.*

**5. Работа с VK API**

Для вызова методов **VK API** требуется получить **Access Token** для пользователя. [Здесь вы можете прочитать, как сделать это](#7t07qhkxkmq1).

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



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 ошибки 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 Demo-проект](https://github.com/OcugineGames/Unity-VK-SDK).