



28 сентября 2012 в 20:55

Django-регистрация с соц. сетями

Введение

Здравствуйтесь, давайте немного пофантазируем. Лариса Петровна, активный участник собраний правления своего дома увидела объявление, что недавно был открыт некий интернет-ресурс, который позволяет вести дебаты с правлением не выходя из дома. За долгих 5 минут она все же нашла все английские буквы в домене zhilcomservice239.ru. Еще «немного» времени она потратила, чтобы найти интересующую её тему собрания. Наконец то, Лариса захотела высказать своё мнение, но не всё так просто. Перед Ларисой Петровной появилось окно регистрации на портале, — следующий этап, который она уже не преодолит. Так и осталось мнение Ларисы невысказанным, а сайт правления лишился активного участника.

Большинство пользователей негативно относится к регистрациям на сайтах. Причинами негативного отношения могут быть: опять пароль придумывать, я же его забуду через день; не хочу ящик светить, а то спамить могут начать; лень заморачиваться, чтобы просто оставить комментарий «Спасибо», и т.д.

Намного «легче» люди относятся к регистрации через сторонние сервисы, социальные сети. Чаще всего, пользователи уже имеют зарегистрированный аккаунт в одной или нескольких популярных социальных сетях. Именно с помощью этого аккаунта многие люди и предпочитают «войти» на сайт правления или любого другого интернет-ресурса.

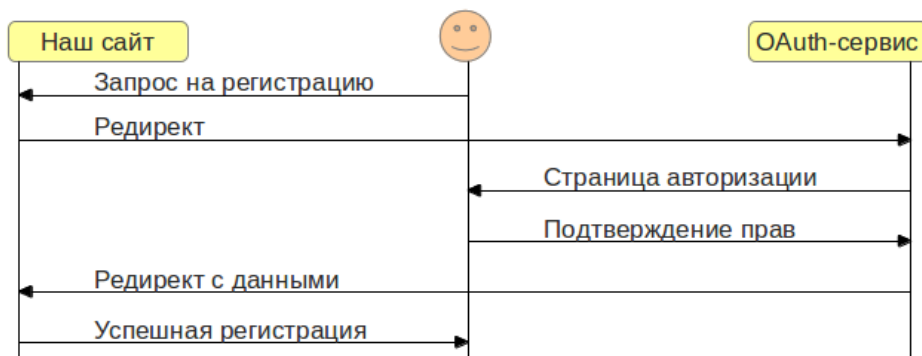
В этой статье мы пройдем с Вами весь путь от регистрации новых приложений в соц. сетях до готового сайта с кнопками входа через социальные сети. В своей работе мы задействуем: python в общем, Django в частности; OAuth в общем, библиотеку django-social-auth в частности. Тема OAuth и взаимодействия с соц. сетями также рассматривается в [статье на Хабре](#). Обе статьи описывают похожие задачи, но в данной упор делается на работу с приложением django-social-auth.

Постановка задачи

Ну что же, давайте теперь представим, что мы, группа разработчиков, получили заказ от правления дома Ларисы Петровны на реализацию такого вида регистрации у них на сайте. Сразу после получения заказа мы узнали, что сайт представляет из себя простейший проект, написанный на Python с использованием известного фреймворка Django, а регистрация на сайте сделана с помощью обычного django.contrib.auth.

Техническое задание мы получили в следующем виде:

сделать на сайте ссылки, при нажатии на которые юзер регистрировался бы на нашем сайте с использованием данных, полученных от соц. сети, соответствующей нажатой ссылке. Для пользователя это должно выглядеть максимально просто: обычный переход по ссылке и он уже зарегистрирован на сайте правления. На рисунке отображено будущее поведение регистрации с точки зрения пользователя сайта, например Ларисы Петровны.



Для взаимодействия со сторонними сервисами мы выбираем OAuth протокол. Сам протокол описывает процедуру авторизации и получения access_token для дальнейшего взаимодействия с API сервиса. Этот протокол прекрасно рассмотрен в [статье](#). К тому же, есть замечательное [приложение от Google](#), которое предоставляет удобный веб-интерфейс, чтобы вручную пройти OAuth авторизацию и «потрогать» большинство открытых Google API.

Итак, пришло время решить, какие сторонние сервисы будет поддерживать сайт правления дома. Давайте реализуем поддержку самых распространенных сервисов в рунете на мой взгляд: ВКонтакте, Facebook, Google, Twitter, Мой Мир, Мой Круг, Одноклассники.

Настройка соц. сервисов

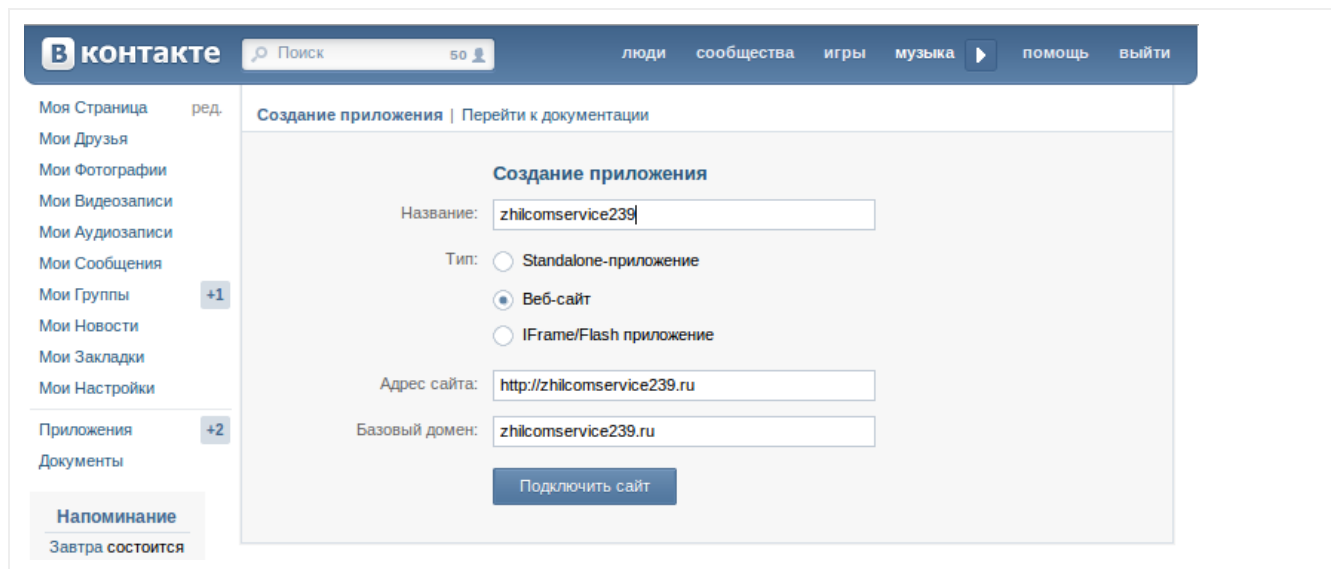
На первом этапе необходимо подготовить социальные сервисы для прохождения OAuth авторизации. Для этого регистрируем на

каждом сервисе новое приложение. К слову сказать, на данный момент у нас нет никакого приложения, но не переживайте, загружать ничего не придется. Мы просто регистрируем сайт (приложение, игру) и возьмем оттуда необходимые для авторизации данные. Для регистрации Вам понадобится аккаунт на каждом из этих сервисов. Полученные данные для авторизации мы сохраним в файл, для удобства дальнейшего использования.

ВКонтакте

Для создания приложения необходимо перейти на [страницу создания приложения](#). Заполнить текстовые поля и выбрать тип приложения — Веб-сайт.

[Создание приложения](#)



ВКонтакте

Поиск 50

люди сообщества игры музыка ▶ помощь выйти

Моя Страница ред.
Мои Друзья
Мои Фотографии
Мои Видеозаписи
Мои Аудиозаписи
Мои Сообщения
Мои Группы +1
Мои Новости
Мои Закладки
Мои Настройки

Приложения +2
Документы

Напоминание
Завтра состоится

Создание приложения | Перейти к документации

Создание приложения

Название:

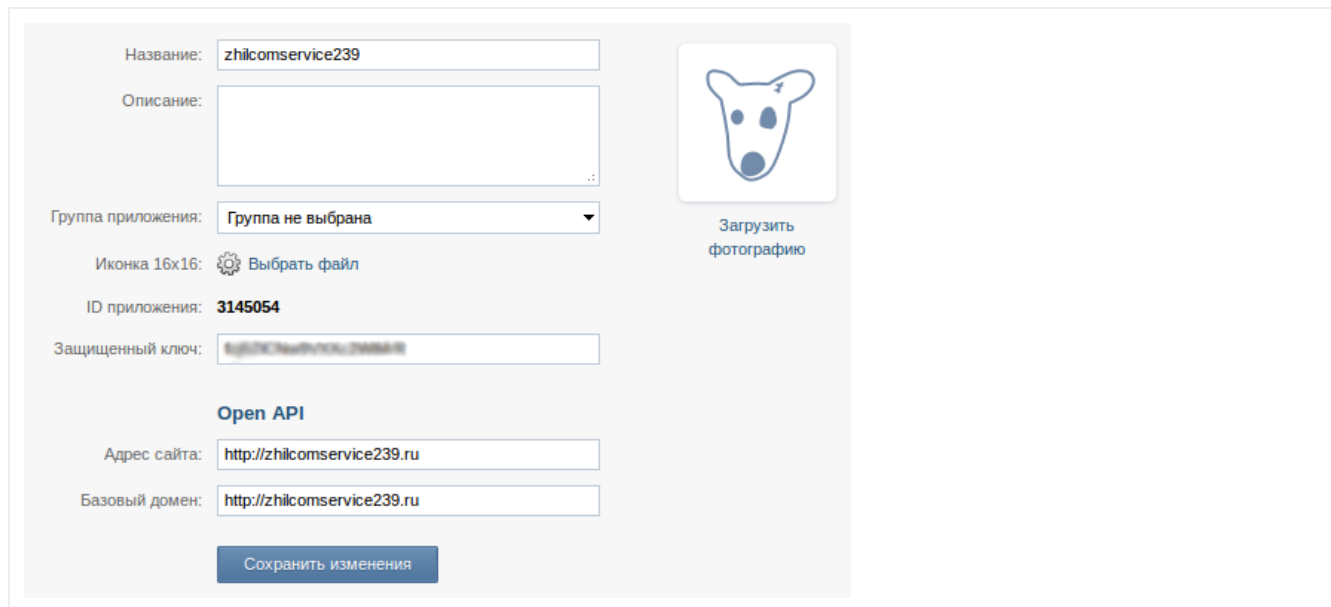
Тип: ☐ Standalone-приложение
☒ Веб-сайт
☐ IFrame/Flash приложение

Адрес сайта:

Базовый домен:

После подтверждения у Вас должна появиться страница редактирования приложения. На этой странице находятся необходимые нам параметры: **ID приложения** и **Защищенный ключ**.

[Страница редактирования приложения](#)



Название:

Описание:

Группа приложения:

Иконка 16x16:

ID приложения: **3145054**

Защищенный ключ:

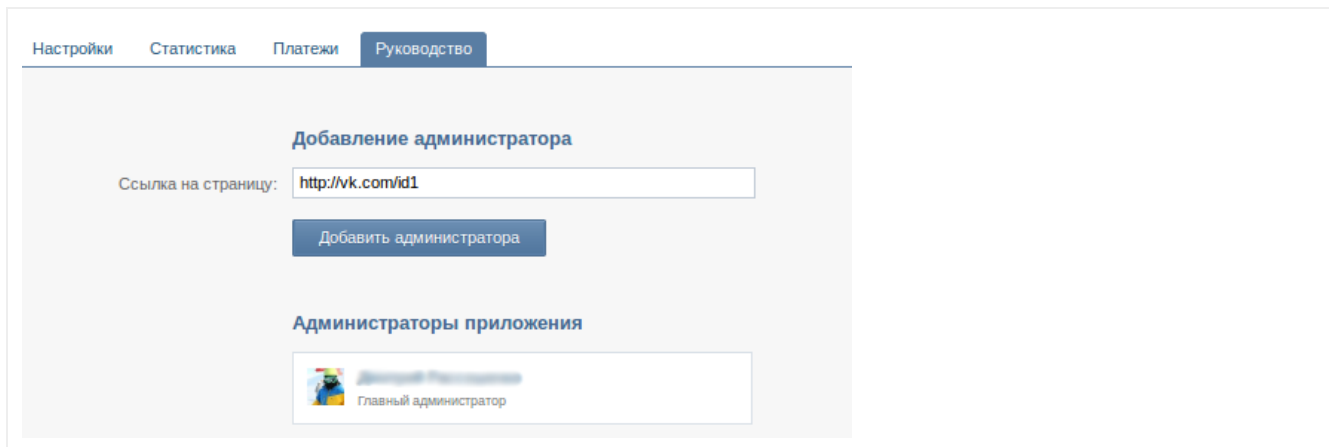
Open API

Адрес сайта:

Базовый домен:

Если Вы работаете в команде, то сможете дать доступ к настройкам приложения или полностью передать приложение другому человеку на вкладке «Руководство».

[Добавление администратора](#)

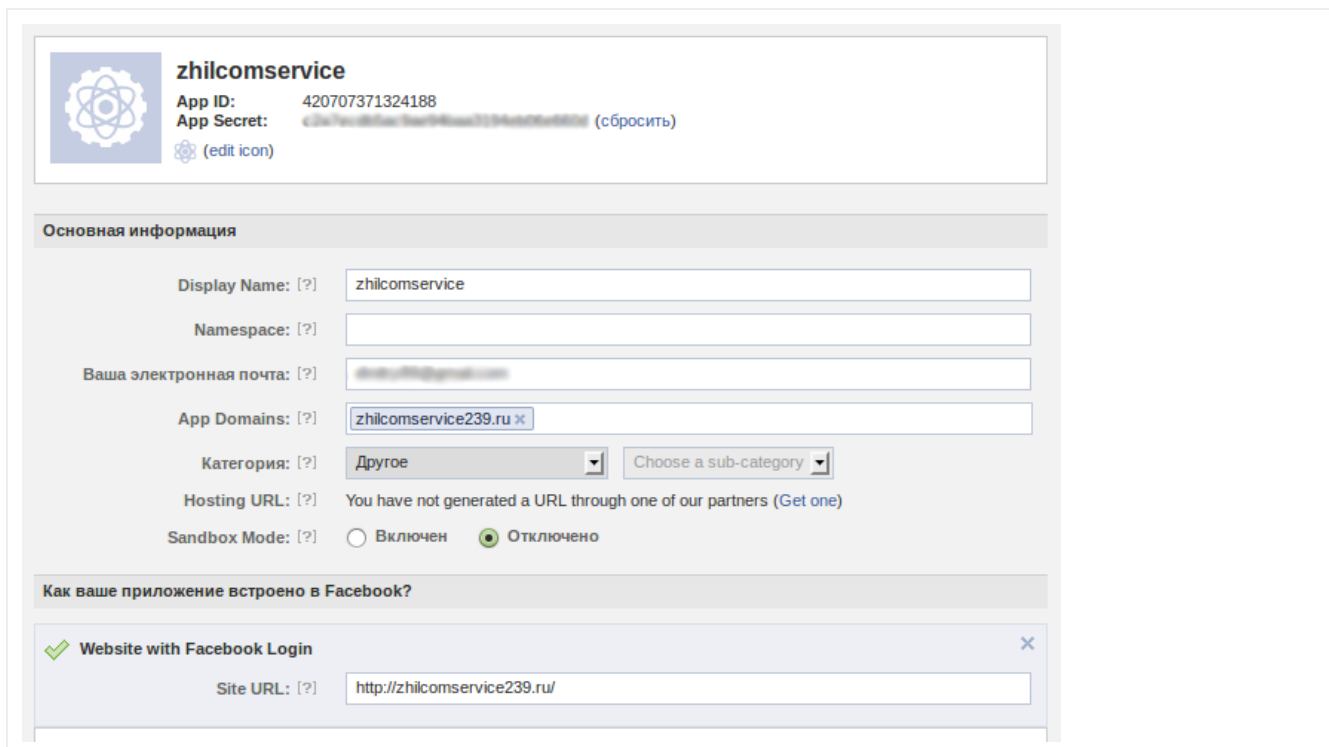


Facebook

Для создания приложения необходимо перейти на [страницу приложений](#) и нажать кнопку «Создать новое приложение». Далее нужно указать App Name: *zhilcomservice*, остальные поля не трогать.

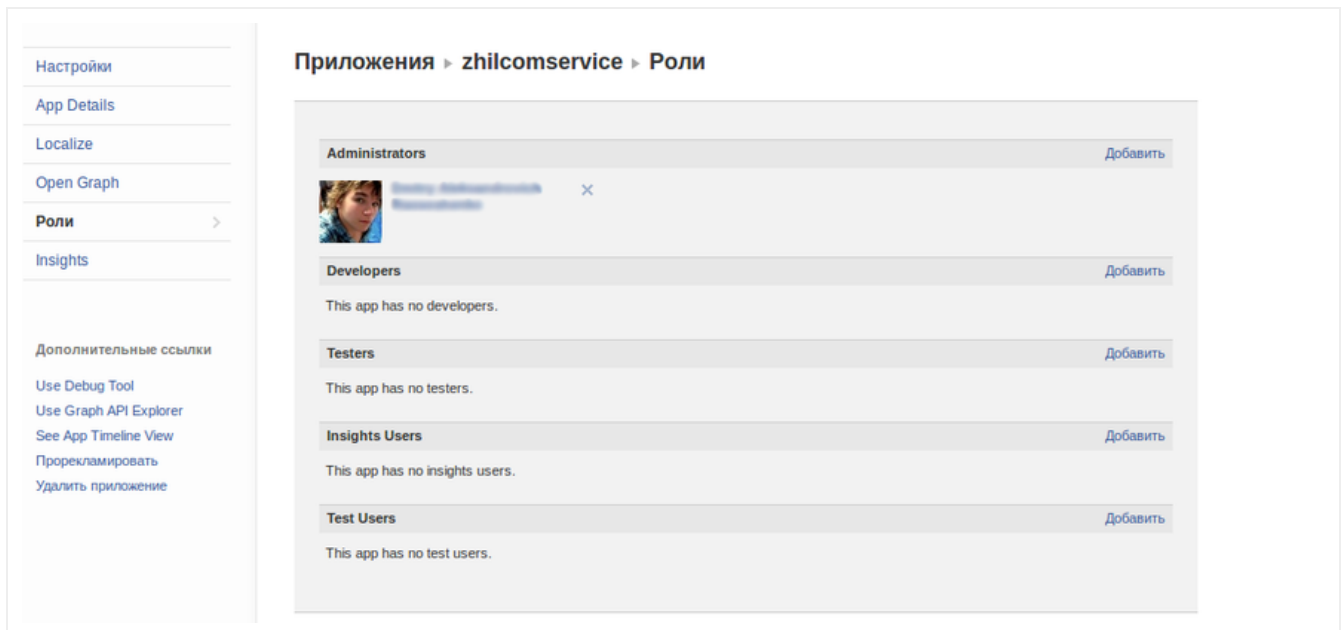
Откроется страница редактирования только что созданного приложения. Нам необходимо указать App Domains: *zhilcomservice239.ru*. В дальнейшем, нам также понадобится указать адрес «Website with Facebook Login: Site URL». Это адрес, на который будет перенаправлять facebook пользователей после авторизации. Пока что мы его не знаем, так что можно написать туда просто адрес сайта. Необходимые нам данные для авторизации Facebook: **App ID, App Secret**.

[Редактирование приложения](#)



Как и ВКонтакте, Facebook позволяет «поделиться» своим приложением с другими участниками разработки. Эта возможность доступна по ссылке «Роли» в левом меню. Стоит заметить, что для добавления разработчика в команду, Вам необходимо быть с ним друзьями.

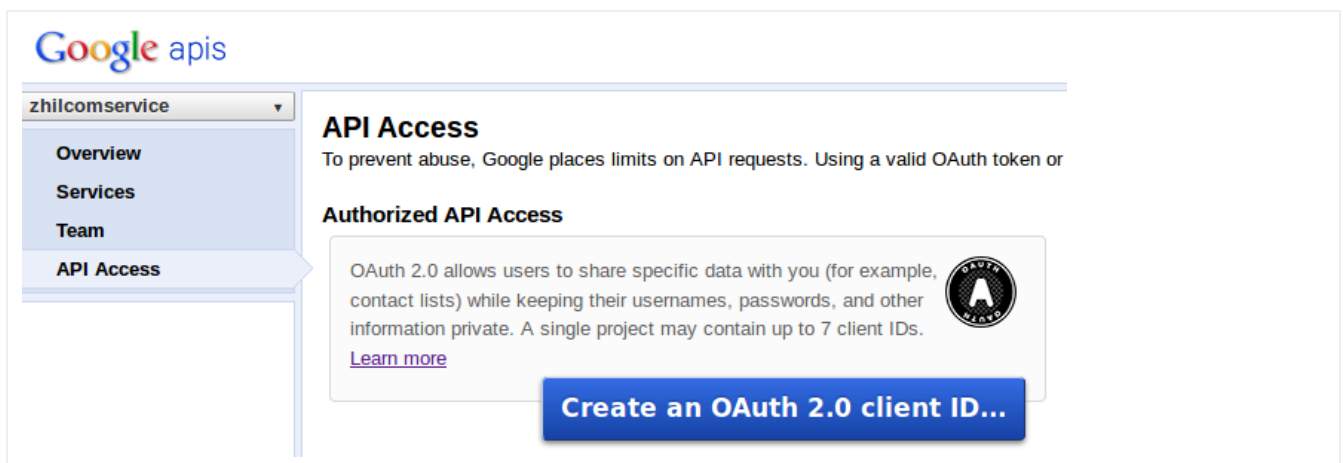
[Роли](#)



Google

Для создания приложения необходимо перейти в [консоль Google приложений](#) и в раскрывающемся списке выбрать Other Projects → Create, ввести название проекта *zhilcomservice*. Далее, слева выбрать пункт API Access и на открывшейся странице нажать «Create an OAuth 2.0 client ID...»

[Create an OAuth 2.0 client ID](#)



Откроется окошко «Create Client ID». Необходимо будет заполнить *Product name*, логотип указывать не обязательно. На следующей странице нужно отметить «Web Application» и ввести адрес сайта: zhilcomservice239.ru. Redirect URI пока оставляем без изменения. Жмем «Create Client ID».

[Create Client ID](#)

Create Client ID

Client ID Settings

Application type

☒ Web application
Accessed by web browsers over a network.

☐ Service account
Calls Google APIs on behalf of your application instead of an end-user. [Learn more](#)

☐ Installed application
Runs on a desktop computer or handheld device (like Android or iPhone).

Your site or hostname ([more options](#))

For example: `www.example.com` or `localhost`

http://

zhilcomservice239.ru

Redirect URI

http://zhilcomservice239.ru/oauth2callback

Create client ID

Back

Cancel

[Learn more](#)

После создания приложения откроется страница, где будут указаны «**Client ID**» и «**Client secret**».

[Client ID и Client Secret](#)

Google apis

zhilcomservice

Overview

Services

Team

API Access

API Access

To prevent abuse, Google places limits on API requests. Using a valid OAuth token or A

Authorized API Access

OAuth 2.0 allows users to share specific data with you (for example, contact lists) while

[more](#)

Branding information

The following information is shown to users whenever you request access to their private

Product name: zhilcomservice

Google account: [zhilcomservice@gmail.com](#)

Edit branding information...

Client ID for web applications

Client ID: 1041478033656.apps.googleusercontent.com

Email address: 1041478033656@developer.gserviceaccount.com

Client secret: [1.yrmpg1jhw-0P5u481-u0u015k8B](#)

Redirect URIs: http://zhilcomservice239.ru/oauth2callback

JavaScript origins: http://zhilcomservice239.ru

Create another client ID...

Пока что Redirect URIs ведет на некорректный адрес, но впоследствии мы снова зайдем в редактирование приложения и исправим адрес на верный.

Twitter

На главной странице [портала разработчиков twitter](#) Вы найдете ссылку «[Create an app](#)».

[Create an application](#)

Create an application

Application Details

Name: *

Your application name. This is used to attribute the source of a tweet and in user-facing authorization screens. 32 characters max.

Description: *

Your application description, which will be shown in user-facing authorization screens. Between 10 and 200 characters max.

Website: *

Your application's publicly accessible home page, where users can go to download, make use of, or find out more information about your application. This fully-qualified URL is used in the source attribution for tweets created by your application and will be shown in user-facing authorization screens.
(If you don't have a URL yet, just put a placeholder here but remember to change it later.)

Callback URL:

Where should we return after successfully authenticating? For [@Anywhere applications](#), only the domain specified in the callback will be used. [OAuth 1.0a](#) applications should explicitly specify their `oauth_callback` URL on the request token step, regardless of the value given here. To restrict your application from using callbacks, leave this field blank.

По сложившейся закономерности, указание Callback URL оставляем на будущее. После создания отобразятся необходимые нам данные: **Consumer Key**, **Consumer Secret**.

В отличие от предыдущих сервисов, Твиттер не позволяет назначать дополнительных администраторов приложения. Так что если Вы работаете в команде, придется разослать необходимые настройки всем участникам любым доступным методом.

Мой круг (Яндекс)

У Яндекса есть [страничка, посвященная OAuth](#). Получение данных для авторизации происходит на [странице создания приложения](#).

[Регистрация нового приложения](#)

Регистрация нового приложения

Название*:

Описание:

Права*: [Мой Круг](#)
[Я.ру](#)

☐ Изменение профиля, создание постов и комментариев в Я.ру

☒ Чтение профилей, постов и комментариев в Я.ру

[Яндекс.Вебмастер](#)
[Яндекс.Директ](#)
[Яндекс.Диск](#)
[Яндекс.Логин](#)
[Яндекс.Маркет](#)
[Яндекс.Метрика](#)
[Яндекс.Подписки](#)
[Яндекс.Фотки](#)

Время жизни токена: неограниченно

Ссылка на иконку:

Ссылка на приложение:

Callback URI:

Клиент для разработки: ☐

После заполнения минимума обязательных полей, права установим на *чтение профилей, постов и комментариев* из API Я.ру, Вам отобразятся необходимые данные: **Id приложения, пароль приложения.**

Результат регистрации

zhilcomservice

Страница: <http://zhilcomservice239.ru/>

Приложение может:

- Чтение профилей, постов и комментариев в Я.ру

Id приложения: beef1e539d73432fa

Пароль приложения: 7e5e7f3f5869452f81

Количество авторизаций: 0

[Редактировать приложение](#)

Мой Мир (Mail.ru)

У мэйла есть [портал подключения сайтов](#) для работы с их API. Нажав на кнопку «[Подключить сайт](#)», Вы сможете зарегистрировать сайт и получить необходимые для авторизации параметры. Нужно будет ввести *название приложения и адрес главной страницы*.

Ввод названия и адреса сайта

Шаг 2 из 3**Информация о сайте**

Название:	<input type="text" value="zhilcomservice"/>
Адрес главной страницы	<input type="text" value="http://zhilcomservice239.ru"/>
<input type="button" value="Продолжить"/>	

Далее Вам предложат скачать файл receiver.html. На этом этапе можно смело не обращать внимания на предупреждение Mail.Ru, что ничего не будет работать без этого файла, и нажать продолжить. Они могут быть настойчивы, но мы все равно можем беззаботно нажать ссылку «Пропустить».

[Скачивание receiver.html](#)

Шаг 3 из 3**Настройка сайта**

Для работы Платформы@Mail.Ru необходимо разместить файл receiver.html на вашем сайте. Это необходимо сделать один раз, больше его менять не придется. [Информация о receiver.html](#).

Скачайте файл receiver.html
Разместите его по адресу **zhilcomservice239.ru/receiver.html**

На следующей странице отобразятся основные данные для прохождения процедуры авторизации. Сохраним **ID** и **Секретный ключ**. Приватный ключ нам не понадобится.

[Сообщение об успешной регистрации](#)

Готово

Ваш сайт зарегистрирован в Платформе@Mail.Ru. Теперь вы можете начинать интеграцию.

Данные вашего сайта:

ID: 68
Приватный ключ: 2ea33159011c76e6c6a
Секретный ключ: 4b0950fdb65b

Для работы в команде, Mail.Ru не сделали прямой возможности поделиться настройками привязанного сайта с другими разработчиками. Но есть достаточно странная возможность: в ссылку «<http://api.mail.ru/apps/my/11111/access/>» нужно вместо 11111 вставить ID привязанного сайта. Вы перейдете на страницу редактирования настроек доступа к приложению, но приложением и будет Ваш привязанный сайт. Я написал в поддержку вопрос, связанный с этим редактированием, но ответа я так и не получил. К сожалению, добавление пользователя через интерфейс приложений не дает ему прав на редактирование привязанного сайта, так что это остается просто наглядным примером глючности Мэйл.Ру.

[Редактирование прав доступа](#)

Мои разработки → **Управление приложением **zhilcomservice****

[Базовые настройки](#) [Настройки](#) [Публикация](#) [Биллинг](#) [Доступ](#) [Хостинг](#) [Удалить](#)

Вы можете предоставить привилегированный доступ к вашему приложению другим пользователям Моего Мира. Подробнее о привилегиях читайте в [документации](#).

<input type="text" value="e-mail пользователя Моего Мира"/>	<input type="button" value="Добавить"/>	Тестирование	Статистика	Редактирование
Ваше приложение		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>


Доступ есть только у вас

Одноклассники

Я удивлен, что люди все еще пользуются этим сервисом. Насколько же сильно реклама может вьестись в головы людей. Это самый глючный и неудобный сервис из моего списка, но что поделаешь, — он действительно популярен, и им нельзя пренебрегать.

На одноклассниках на портале разработчика есть [небольшая документация по OAuth](#). [Форма регистрации](#) приложения выглядит совсем примитивно.

[Форма регистрации OAuth доступа](#)



Форма регистрации OAuth доступа

Added by [wiki-api-admin](#), last edited by [Edgars Strods](#) on Oct 13, 2011 ([view change](#))

Имя, фамилия	<input type="text"/>
URL	<input type="text" value="http://zhilcomservice239.ru"/>
Логин на Одноклассники.ru	<input type="text"/>
Ваш адрес эл.почты	<input type="text"/>



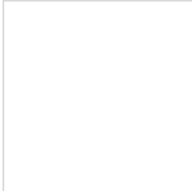
После отправки регистрационной формы придется подождать некоторое время, пока заявка будет [рассмотрена вручную](#). После рассмотрения и одобрения Вашей заявки, Вам на почту придет уведомление о предоставлении прав разработчика на сайте Одноклассники.ру и ссылка на инструкцию по регистрации нового приложения.

В общих словах: у вас появляется ссылка на добавление нового приложения (игры) в пункте меню Игры → Мои загруженные. При добавлении Вы указываете, что приложение будет размещено «Вне Одноклассников», а тип авторизации будет «OAuth».

Далее необходимо будет ввести название Вашего приложения, короткое имя для ссылки на него в Одноклассниках, ссылку на приложение (<http://...>), ссылки на картинку и аватарку приложения. Ссылка на иконку не обязательна. После этого Вам на почту придет письмо с данными для авторизации: **Application ID**, **Публичный ключ приложения** и **Секретный ключ приложения**. Все три параметра нам понадобятся.

[Скрытый текст](#)

Название	<input type="text" value="zhilcomservice"/>
Shortname	<input type="text" value="www.odnoklassniki.ru/game/zhilcomservice"/> <small>Например, game-name</small>
Ссылка на приложение	<input type="text" value="http://zhilcomservice239.ru"/>
Ссылка на картинку	<input type="text" value="1/1320507835_1294124996_img_117012_google-logo.jpg"/> <small>Введите ссылку на картинку</small>
Ссылка на аватарку	<input type="text" value="1/1320507835_1294124996_img_117012_google-logo.jpg"/> <small>Введите ссылку на аватарку</small>
Ссылка на иконку	<input type="text"/>



Мы зарегистрировались во всех желаемых сервисах и теперь готовы к началу настройки самой авторизации. Далее будет рассказано, как с помощью библиотеки [django-social-auth](#) сделать поддержку регистрации на сайте через социальную сеть.

Со стороны Django

Ради упрощения примера начальный сайт обладает минимальной функциональностью:

- пользователи смогут авторизоваться;
- увидеть приветствие со своим ником;
- выйти

Так как регистрации базовый сайт не предоставляет, то изначально войти сможет только администратор. Если Вы его не создавали, то команда

```
python manage.py createsuperuser
```

поможет Вам в этом.

Для начала добавим приложение *social_auth* в список используемых приложений INSTALLED_APPS:

```
# settings.py
INSTALLED_APPS = (
    ...
    'social_auth',
)
```

Далее, необходимо указать список AUTHENTICATION_BACKENDS в настройках Вашего проекта с добавлением в него бэкэндов для авторизации через соц. сети. Нужно не забыть указать джанговский бэкэнд ModelBackend, иначе войти на сайт по связке пароль&логин будет невозможно.

AUTHENTICATION_BACKENDS

```
# settings.py
AUTHENTICATION_BACKENDS = (
    'social_auth.backends.contrib.vkontakte.VKontakteOAuth2Backend',
    'social_auth.backends.facebook.FacebookBackend',
    'social_auth.backends.google.GoogleOAuth2Backend',
    'social_auth.backends.twitter.TwitterBackend',
    'social_auth.backends.contrib.yandex.YandexOAuth2Backend',
    'social_auth.backends.contrib.mailru.MailruBackend',
    'social_auth.backends.contrib.odnoklassniki.OdnoklassnikiBackend',
    'django.contrib.auth.backends.ModelBackend',
)
```

В файл настроек необходимо будет добавить параметры для OAuth авторизации для каждого социального сервиса. Для работы с Одноклассниками необходимо указать три полученных параметра.

OAuth settings

```
VK_APP_ID = '3049876'
VK_API_SECRET = 'KcviIeKlsapVIsd9Kvjasn'

FACEBOOK_APP_ID = '93345789454'
FACEBOOK_API_SECRET = 'vksdf78234jmiv90234ms0ds8g54miv9s'

TWITTER_CONSUMER_KEY = 'jJidwMvjasdKLzxcmaaPw'
TWITTER_CONSUMER_SECRET = 'SjvUJ80i8eoJKvskKe9GicIKD931'

GOOGLE_OAUTH2_CLIENT_ID = '19807546895.apps.googleusercontent.com'
GOOGLE_OAUTH2_CLIENT_SECRET = 'Ksjdaujvx-PoIeq81LxkV893D'

MAILRU_OAUTH2_CLIENT_KEY = '971648'
MAILRU_OAUTH2_CLIENT_SECRET = 'c93kfks046kcvb8et8f63jig0syw8jfq'

YANDEX_OAUTH2_CLIENT_KEY = 'xcvb7ewrtw7yuud58o3asvsihyeqx'
YANDEX_OAUTH2_CLIENT_SECRET = 'xybr8q91j235jhnviwytjsgkxp39kshwu'

ODNOKLASSNIKI_OAUTH2_CLIENT_KEY = '59824691'
ODNOKLASSNIKI_OAUTH2_APP_KEY = 'KLOAPPWIDUDUDUDU'
ODNOKLASSNIKI_OAUTH2_CLIENT_SECRET = 'D879S7VCYI80234KER8QXOTP2'

# эти три строчки необходимы для нормальной работы Яндекс авторизации
YANDEX_APP_ID = YANDEX_OAUTH2_CLIENT_KEY
```

```
YANDEX_API_SECRET = YANDEX_OAUTH2_CLIENT_SECRET
YANDEX_OAUTH2_API_URL = 'https://api-yaru.yandex.ru/me/'
```

На страницу входа мы добавим ссылки для входа через каждого «социального провайдера».

[login.html](#)!

```
<p style="margin: 0.4em;">
  <a href="{% url socialauth_begin 'vkontakte-oauth2' %}">Enter VK</a>
</p>
<p style="margin: 0.4em;">
  <a href="{% url socialauth_begin 'facebook' %}">Enter Facebook</a>
</p>
<p style="margin: 0.4em;">
  <a href="{% url socialauth_begin 'google-oauth2' %}">Enter Google</a>
</p>
<p style="margin: 0.4em;">
  <a href="{% url socialauth_begin 'twitter' %}">Enter Twitter</a>
</p>
<p style="margin: 0.4em;">
  <a href="{% url socialauth_begin 'yandex-oauth2' %}">Enter Yandex</a>
</p>
<p style="margin: 0.4em;">
  <a href="{% url socialauth_begin 'mailru-oauth2' %}">Enter Mail.Ru</a>
</p>
<p style="margin: 0.4em;">
  <a href="{% url socialauth_begin 'odnoklassniki' %}">Enter Odnoklassniki</a>
</p>
```

Теперь пришло время определить урлы для нашей авторизации. Давайте вынесем всю авторизацию, связанную с социальными сетями в отдельное пространство имен. Добавляем в наш [urls.py](#) следующее правило

```
urlpatterns = (
    ...
    url(r'^social/', include('social_auth.urls')),
)
```

Раз мы настроили пути в приложении, то теперь можно указать и пропущенные callback в настройках социальных сервисов. ВКонтакте не требует указания определенного Callback. Единственное правило, как и для многих OAuth провайдеров, нахождение адреса для обратного редиректа в пределах базового домена.

Facebook, как бы они грозно не предупреждали, абсолютно спокойно работает и без точного указания Site URL. Можно не трогать, но лучше бы конечно изменить на zhihcomservice239.ru/social/complete/facebook/, вдруг в будущем сделают жесткую проверку.

С твиттером дела обстоят чуть серьезней. Если Вы получаете 401 ошибку при попытке зайти через твиттер, то первым делом нужно проверить callback на точное соответствие. Заходим на твиттер и на вкладке Settings нашего приложения меняем Callback URL на zhihcomservice239.ru/social/complete/twitter/

Теперь настал черед Яндекса. Заходим на страницу редактирования приложения (oauth.yandex.ru/client/{AppKey}/edit) и устанавливаем Callback URI равный zhihcomservice239.ru/social/complete/yandex-oauth2/.

Мэйл.Ру и Одноклассники не требуют настройки callback.

Проверка результата

Для проверки нашего приложения понадобится загрузить приложение на сервер и привязать указанный в настройках соц. сервисов домен к этому серверу. Далее развернуть приложение одним из множества способов.

Можно пойти более простым путем и устроить проверку не выходя за пределы Вашего компьютера, то есть локально. Для этого в `/etc/hosts` (WINDOWS\system32\drivers\etc\hosts) добавим строчку:

```
127.0.0.1 zhihcomservice239.ru
```

Следующую команду необходимо выполнять с root доступом, так как используем 80 порт:

```
# python manage.py runserver 127.0.0.1:80
```

После успешного запуска локального сервера по адресу zhlcomservice239.ru будет находиться наше приложение. Попробовав войти через каждую доступную соц. сеть? мы обнаружим неприятную ошибку: не работает вход через Yandex.ru (Мой Круг): **400 redirect_uri_mismatch**. Увы, но это ошибка в библиотеке django-social-auth. Если у Вас этой ошибки не возникает, значит [Матиа](#) молодец и уже исправил её, а информация из следующей главы Вам не понадобится.

Устранение ошибки логина через Yandex

Ошибка происходит из-за неправильной конкатенации передаваемого GET параметра `redirect_uri` с параметром `redirect_state`. Юзер с ником [Krvss](#) подсказал мне, что ошибку нужно начинать искать в `social_auth.utils.url_add_parameters`.

Здесь я эту ошибку обошел не совсем красиво, так как избавился от следствия, а в причине не стал разбираться. Надеюсь, Вас не устроит приводимый здесь вариант для работы с Яндексом и Вы избавите пользователей этого приложения (django-social-auth) от ошибки.

Мы просто отключаем добавление параметра `redirect_state` для YandexBackend. Для этого создадим файл [auth/yandex.py](#) и вставим туда следующее содержание:

[auth/yandex.ru](#)

Естественно, не забываем изменить путь в настройках `AUTHENTICATION_BACKENDS` с 'social_auth.backends.contrib.yandex.YandexOAuth2Backend' на 'auth.yandex.YandexOAuth2Backend'.

Pipeline

Важно отметить, где и как происходит вся магия с данными, такая как автоматическое создание пользователей по данным, переданным из социальной сети.

После авторизации и получения набора данных от социальной сети, подходящий бэкенд собирает словарь с определенными ключами для дальнейшей работы с данными пользователя: `email`, `username`, `first_name`, `last_name`. После чего все данные, в том числе и полученные необработанные данные от соц. сервиса передаются в финальную стадию обработки: pipeline.

Pipeline представляет из себя цепочку определенных функций, которые производят обработку полученных от социальной сети данных. В эту цепочку можно добавлять свои функции и удалять уже определенные. Примечательно, что каждая функция возвращает словарь, значения из которого будут переданы следующей функции именованными параметрами.

Цепочка функций задается параметром `SOCIAL_AUTH_PIPELINE` в настройках Вашего приложения. По-умолчанию эта цепочка выглядит так:

[SOCIAL_AUTH_PIPELINE](#)

```
(
    'social_auth.backends.pipeline.social.social_auth_user',
    #'social_auth.backends.pipeline.associate.associate_by_email',
    'social_auth.backends.pipeline.user.get_username',
    'social_auth.backends.pipeline.user.create_user',
    'social_auth.backends.pipeline.social.associate_user',
    'social_auth.backends.pipeline.social.load_extra_data',
    'social_auth.backends.pipeline.user.update_user_details'
)
```

Более подробно о работе с pipeline описано в [документации django-social-auth](#).

Заключение

Мы успешно реализовали необходимый функционал для портала правления. Особенную благодарность нам выразила Лариса Петровна, так как теперь она заходит на сайт одним кликом мышки и может сразу приступить к обсуждению интересующей её темы.

Руководство правления также осталось довольно, так как на их сайте кол-во активных участников выросло более чем в 2 раза, и похоже они планируют долговременное сотрудничество с нами, так как добавление поддержки дополнительных соц. сервисов

мы пообещали сделать быстро и недорого.

Заметный минус в нашем приложении: оно никак не обрабатывает ошибки. Пока администрация портала правления этого не заметила, и у нас есть шанс добавить обработку ошибок.

Приложение из данной статьи доступно на [Github](#).

```
git clone https://github.com/rda-dev/zhilcomservice239.git
```

django-social-auth, oauth, социальные сети, django

лейтенант Буэндиа

[Радиоастрон рвет шаблоны](#)

[Открытое письмо для РОИ и
ФИД по петиции об отмене
187-ФЗ
\(#ЗаконПротивИнтернета\)](#)

[DoS эксплоит для движка
WebKit](#)