# Написание вашего первого приложения Django, часть 1

Давайте разберемся на примере.

В этом руководстве мы расскажем вам о создании базового приложения для опроса.

Оно будет состоять из двух частей:

* Общедоступный сайт, который позволяет людям просматривать опросы и голосовать в них.
* Сайт администратора, который позволяет добавлять, изменять и удалять опросы.

Предположим, что у вас уже [установлен Django](https://docs.djangoproject.com/en/5.0/intro/install/). Вы можете определить, установлен ли Django и какая версия, выполнив следующую команду в командной строке (обозначается префиксом $).:

/

**... \>** py -m django --версия

Если Django установлен, вы должны увидеть версию вашей установки. Если это не так, вы получите сообщение об ошибке “Нет модуля с именем django”.

Этот учебник написан для Django 5.0, который поддерживает Python 3.10 и более поздних версий. Если версия Django не соответствует, вы можете обратиться к руководству по вашей версии Django, используя переключатель версий в правом нижнем углу этой страницы, или обновить Django до новейшей версии. Если вы используете более старую версию Python, проверьте, [какую версию Python я могу использовать с Django?](https://docs.djangoproject.com/en/5.0/faq/install/#faq-python-version-support) чтобы найти совместимую версию Django.

Смотрите [Как установить Django](https://docs.djangoproject.com/en/5.0/topics/install/), чтобы получить рекомендации по удалению старых версий Django и установке более новой.

**Где получить помощь:**

Если у вас возникли проблемы с прохождением этого руководства, пожалуйста, перейдите к разделу "[Получение справки" в разделе часто задаваемых вопросов.](https://docs.djangoproject.com/en/5.0/faq/help/)

## Создание проекта

Если вы впервые используете Django, вам придется позаботиться о некоторой начальной настройке. А именно, вам потребуется автоматически генерировать некоторый код, который устанавливает [проект](https://docs.djangoproject.com/en/5.0/glossary/#term-project) Django – набор настроек для экземпляра Django, включая конфигурацию базы данных, параметры, специфичные для Django, и настройки, специфичные для приложения.

Из командной строки, **cd** в каталог, где вы хотели бы сохранить свой код, затем выполните следующую команду:

/

**... \>** django-admin startproject mysite

Это создаст **mysite** каталог в вашем текущем каталоге. Если это не сработало, см. раздел [Проблемы с запуском django-admin](https://docs.djangoproject.com/en/5.0/faq/troubleshooting/#troubleshooting-django-admin).

**Примечание**

Вам нужно избегать именования проектов в честь встроенных компонентов Python и Django. В частности, это означает, что вам следует избегать использования таких имен, как **django** (которые будут конфликтовать с самим Django) или **test** (которые конфликтуют со встроенным пакетом Python).

**Где должен жить этот код?**

Если вы работаете на простом старом PHP (без использования современных фреймворков), вы, вероятно, привыкли размещать код в корневом каталоге документов веб-сервера (в таком месте, как **/var/www**). С Django вы этого не делаете. Не стоит помещать какой-либо из этих кодов Python в корневой каталог документов вашего веб-сервера, потому что это создает риск того, что люди смогут просматривать ваш код через Интернет. Это плохо сказывается на безопасности.

Поместите свой код в какой-нибудь каталог **вне** корневого каталога документа, например **/home/mycode**.

Давайте посмотрим, что [**startproject**](https://docs.djangoproject.com/en/5.0/ref/django-admin/#django-admin-startproject) создало:

mysite/

manage.py

mysite/

\_\_init\_\_.py

настройки.py

urls.py

asgi.py

wsgi.py

Эти файлы:

* Внешний **mysite/** корневой каталог - это контейнер для вашего проекта. Его название не имеет значения для Django; вы можете переименовать его как угодно.
* **manage.py**: Утилита командной строки, которая позволяет вам взаимодействовать с этим проектом Django различными способами. Вы можете прочитать все подробности о **manage.py** в [django-admin и manage.py](https://docs.djangoproject.com/en/5.0/ref/django-admin/) .
* Внутренний **mysite/** каталог - это фактический пакет Python для вашего проекта. Его имя - это имя пакета Python, которое вам нужно будет использовать для импорта чего-либо внутри него (например, **mysite.urls**).
* **mysite/\_\_init\_\_.py**: Пустой файл, который сообщает Python, что этот каталог следует рассматривать как пакет Python. Если вы новичок в Python, прочитайте [больше о пакетах](https://docs.python.org/3/tutorial/modules.html#tut-packages) в официальных документах Python.
* **mysite/settings.py**: Настройки / конфигурация для этого проекта Django. [Настройки Django](https://docs.djangoproject.com/en/5.0/topics/settings/) расскажут вам все о том, как работают настройки.
* **mysite/urls.py**: Объявления URL-адресов для этого проекта Django; "оглавление” вашего сайта на базе Django. Вы можете прочитать больше об URL-адресах в [диспетчере URL](https://docs.djangoproject.com/en/5.0/topics/http/urls/).
* **mysite/asgi.py**: Точка входа для веб-серверов, совместимых с ASGI, для обслуживания вашего проекта. Смотрите [Как выполнить развертывание с помощью ASGI](https://docs.djangoproject.com/en/5.0/howto/deployment/asgi/) для получения более подробной информации.
* **mysite/wsgi.py**: Точка входа для веб-серверов, совместимых с WSGI, для обслуживания вашего проекта. Смотрите [Как выполнить развертывание с помощью WSGI](https://docs.djangoproject.com/en/5.0/howto/deployment/wsgi/) для получения более подробной информации.

## Сервер разработки

Давайте убедимся, что ваш проект Django работает. Перейдите во внешний **mysite** каталог, если вы еще этого не сделали, и выполните следующие команды:

/

**... \>** py manage.py runserver

В командной строке вы увидите следующий вывод:

Выполнение системных проверок...

Проверка системы не выявила проблем (отключено 0).

У вас есть неприменимые миграции; ваше приложение может не работать должным образом, пока они не будут применены.

Запустите "python manage.py migrate", чтобы применить их.

01 апреля 2024 г. - 15:50:53

Django версии 5.0, с использованием настроек 'mysite.settings'

Запуск сервера разработки по адресу [http://127.0.0.1:8000 /](http://127.0.0.1:8000/)

Завершите работу сервера с помощью CONTROL-C.

**Примечание**

Пока игнорируйте предупреждение о непримиримых миграциях базы данных; вскоре мы разберемся с базой данных.

Вы запустили Django development server, легкий веб-сервер, написанный исключительно на Python. Мы включили это в Django, чтобы вы могли быстро разрабатывать, не занимаясь настройкой производственного сервера, такого как Apache, до тех пор, пока вы не будете готовы к производству.

Сейчас самое время отметить: **не** используйте этот сервер ни в чем, напоминающем производственную среду. Он предназначен только для использования во время разработки. (Мы занимаемся созданием веб-фреймворков, а не веб-серверов.)

Теперь, когда сервер запущен, перейдите на страницу [http://127.0.0.1:8000 /](http://127.0.0.1:8000/) с помощью вашего веб-браузера. Вы увидите страницу ”Поздравляем!" со взлетающей ракетой. Это сработало!

**Изменение порта**

По умолчанию команда [**runserver**](https://docs.djangoproject.com/en/5.0/ref/django-admin/#django-admin-runserver) запускает сервер разработки по внутреннему IP-адресу на порту 8000.

Если вы хотите изменить порт сервера, передайте его в качестве аргумента командной строки. Например, эта команда запускает сервер с порта 8080:

/

**... \>** py manage.py runserver 8080

Если вы хотите изменить IP сервера, передайте его вместе с портом. Например, чтобы прослушивать все доступные общедоступные IP-адреса (что полезно, если вы используете Vagrant или хотите продемонстрировать свою работу на других компьютерах в сети), используйте:

/

**...\>** py manage.py runserver 0.0.0.0:8000

Полные документы для сервера разработки можно найти в [**runserver**](https://docs.djangoproject.com/en/5.0/ref/django-admin/#django-admin-runserver) справочнике.

**Автоматическая перезагрузка** [**runserver**](https://docs.djangoproject.com/en/5.0/ref/django-admin/#django-admin-runserver)

Сервер разработки автоматически перезагружает код Python для каждого запроса по мере необходимости. Вам не нужно перезапускать сервер, чтобы изменения кода вступили в силу. Однако некоторые действия, такие как добавление файлов, не запускают перезапуск, поэтому в этих случаях вам придется перезапустить сервер.

## Создание приложения для опросов

Теперь, когда ваша среда – “проект” – настроена, вы готовы приступить к работе.

Каждое приложение, которое вы пишете на Django, состоит из пакета Python, который следует определенному соглашению. Django поставляется с утилитой, которая автоматически генерирует базовую структуру каталогов приложения, поэтому вы можете сосредоточиться на написании кода, а не на создании каталогов.

**Проекты против приложений**

В чем разница между проектом и приложением? Приложение - это веб-приложение, которое что–то делает, например, систему блогов, базу общедоступных записей или небольшое приложение для опроса. Проект - это набор настроек и приложений для определенного веб-сайта. Проект может содержать несколько приложений. Приложение может находиться в нескольких проектах.

Ваши приложения могут располагаться в любом месте вашего [пути Python](https://docs.python.org/3/tutorial/modules.html#tut-searchpath). В этом руководстве мы создадим наше приложение для опроса в том же каталоге, что и ваш **manage.py** файл, чтобы его можно было импортировать как собственный модуль верхнего уровня, а не как подмодуль **mysite**.

Чтобы создать свое приложение, убедитесь, что вы находитесь в том же каталоге, что и **manage.py**, и введите эту команду:

/

**... \>** python manage.py startapp polls

Это создаст каталог **polls**, который выглядит следующим образом:

Опросы/

\_\_init\_\_.py

admin.py

приложения.py

миграции/

\_\_init\_\_.py

модели.py

тесты.py

просмотры.py

В этой структуре каталогов будет размещено приложение для опроса.

## Напишите свой первый вид

Давайте напишем первое представление. Откройте файл **polls/views.py** и поместите в него следующий код Python:

polls/views.py[**¶**](https://docs.djangoproject.com/en/5.0/intro/tutorial01/#id1)

**from** **django.http** **import** HttpResponse

**def** **index**(index):

**верните** HttpResponse("Привет, мир. Вы попали в индекс опросов.")

Это самое простое представление, возможное в Django. Чтобы вызвать представление, нам нужно сопоставить его с URL-адресом - и для этого нам нужен URLconf .

Чтобы создать URLconf в каталоге опросов, создайте файл с именем **urls.py**. Теперь каталог вашего приложения должен выглядеть следующим образом:

Опросы/

\_\_init\_\_.py

admin.py

приложения.py

миграции/

\_\_init\_\_.py

модели.py

тесты.py

urls.py

просмотры.py

В **polls/urls.py** файл включите следующий код:

polls/urls.py[**¶**](https://docs.djangoproject.com/en/5.0/intro/tutorial01/#id2)

**from django.urls import path**

**from . import views**

urlpatterns **=** [

path("", views**.**index, name**=**"index"),

]

Следующий шаг - указать корневой URLconf на **polls.urls** модуль. В **mysite/urls.py** добавьте импорт для **django.urls.include** и вставьте [**include()**](https://docs.djangoproject.com/en/5.0/ref/urls/#django.urls.include) в **urlpatterns** список, чтобы у вас было:

mysite/urls.py[**¶**](https://docs.djangoproject.com/en/5.0/intro/tutorial01/#id3)

**from** **django.contrib** **import** admin

**from** **django.urls** **import** include, path

urlpatterns **=** [

path("polls/", include("polls.urls")),

path("admin/", admin**.**сайт**.**urls),

]

Функция [**include()**](https://docs.djangoproject.com/en/5.0/ref/urls/#django.urls.include) позволяет ссылаться на другие URLconfs. Всякий раз, когда Django сталкивается с [**include()**](https://docs.djangoproject.com/en/5.0/ref/urls/#django.urls.include), он отсекает любую часть URL, соответствующую этому моменту, и отправляет оставшуюся строку во включенный URLconf для дальнейшей обработки.

Идея, стоящая за [**include()**](https://docs.djangoproject.com/en/5.0/ref/urls/#django.urls.include), заключается в том, чтобы упростить использование URL-адресов plug-and-play. Поскольку опросы находятся в их собственном URLconf (**polls/urls.py**), их можно разместить в разделе “/polls /“, или в разделе ”/ fun\_polls /“, или в разделе ”/content / polls /", или в любом другом корневом каталоге, и приложение все равно будет работать.

**Когда использовать** [**include()**](https://docs.djangoproject.com/en/5.0/ref/urls/#django.urls.include)

Вы всегда должны использовать **include()** при добавлении других шаблонов URL. **admin.site.urls** это единственное исключение из этого правила.

Теперь вы подключили **index** представление к URLconf. Убедитесь, что оно работает, с помощью следующей команды:

/

**... \>** py manage.py runserver

Перейдите на страницу [http://localhost:8000/polls /](http://localhost:8000/polls/) в вашем браузере, и вы должны увидеть текст “*Привет, мир. Вы находитесь на странице ”Индекс опросов.*", которую вы определили в **index** представлении.

**Страница не найдена?**

Если вы получаете здесь страницу с ошибкой, убедитесь, что вы переходите к [http://localhost:8000/polls /](http://localhost:8000/polls/), а не [http://localhost:8000 /](http://localhost:8000/).

[**path()**](https://docs.djangoproject.com/en/5.0/ref/urls/#django.urls.path) Функции передаются четыре аргумента, два обязательных: **route** и **view** и два необязательных: **kwargs**, и **name**. На этом этапе стоит рассмотреть, для чего нужны эти аргументы.

### [**path()**](https://docs.djangoproject.com/en/5.0/ref/urls/#django.urls.path) **аргумент: route**

**route** это строка, содержащая шаблон URL. При обработке запроса Django начинает с первого шаблона в **urlpatterns** и продвигается вниз по списку, сравнивая запрошенный URL с каждым шаблоном, пока не найдет подходящий.

Шаблоны не выполняют поиск по параметрам GET и POST или по имени домена. Например, в запросе к **https://www.example.com/myapp/** URLconf будет искать **myapp/**. В запросе к **https://www.example.com/myapp/?page=3** URLconf также будет искать **myapp/**.

### [**path()**](https://docs.djangoproject.com/en/5.0/ref/urls/#django.urls.path) **аргумент: view**

Когда Django находит соответствующий шаблон, он вызывает указанную функцию просмотра с [**HttpRequest**](https://docs.djangoproject.com/en/5.0/ref/request-response/#django.http.HttpRequest) объектом в качестве первого аргумента и любыми “захваченными” значениями из маршрута в качестве аргументов ключевого слова. Чуть позже мы приведем пример этого.

### [**path()**](https://docs.djangoproject.com/en/5.0/ref/urls/#django.urls.path) **аргумент: kwargs**

Произвольные аргументы ключевых слов могут быть переданы в словаре в целевое представление. Мы не собираемся использовать эту функцию Django в руководстве.

### [**path()**](https://docs.djangoproject.com/en/5.0/ref/urls/#django.urls.path) **аргумент: name**

Присвоение имени вашему URL-адресу позволяет вам однозначно ссылаться на него из любого другого места в Django, особенно из шаблонов. Эта мощная функция позволяет вам вносить глобальные изменения в шаблоны URL-адресов вашего проекта, касаясь только одного файла.

Когда вы освоитесь с базовым потоком запросов и ответов, прочтите [часть 2 этого руководства](https://docs.djangoproject.com/en/5.0/intro/tutorial02/), чтобы начать работу с базой данных.