Написание вашего первого приложения Django, часть 1

Давайте научимся на примере.

На протяжении всего этого учебника мы расскажем вам о создании базового приложения для опроса.

Он будет состоять из двух частей:

- Публичный сайт, который позволяет людям просматривать опросы и голосовать в них.
- Административная площадка, которая позволяет добавлять, изменять и удалять опросы.

Мы предполагаем, что у вас уже установлен Django. Вы можете сказать, что Django установлен и какая версия, выполнив следующую команду в приглашении оболочки (указана префиксом $):

```
$ python -m django --version
```

Если Django установлен, вы должны увидеть версию вашей установки. Если это не так, вы получите ошибку, сообщающую «Нет модуля с именем django».

Это учебное пособие написано для Django 3.2, который поддерживает Python 3.6 и более поздние версии. Если версия Django не совпадает, вы можете обратиться к учебнику для вашей версии Django, используя переключатель версий в правом нижнем углу этой страницы, или обновить Django до последней версии. Если вы используете более старую версию Python, проверьте, какую версию Python я могу использовать с Django? чтобы найти совместимую версию Django.

См. раздел Как установить Django для получения советов по удалению старых версий Django и более новой.

Где получить помощь:

Если у вас возникли проблемы с прохождением этого учебника, перейдите в раздел «Получение справки» в разделе часто задаваемых вопросов.

Создание проекта

Если вы используете Django в первый раз, вам придется позаботиться о первоначальной настройке. А именно, вам нужно будет автоматически сгенерировать некоторый код, который устанавливает проект Django - набор настроек для экземпляра Django, включая конфигурацию базы данных, параметры, специфичные для Django, и настройки для конкретного приложения.

From the command line, **`cd`** into a directory where you'd like to store your code, then run the following command:

```
$ django-admin startproject mysite
```

This will create a **`mysite`** directory in your current directory. If it didn't work, see Problems running django-admin.

Примечание

You'll need to avoid naming projects after built-in Python or Django components. In particular, this means you should avoid using names like **`django`** (which will conflict with Django itself) or **`test`** (which conflicts with a built-in Python package).

Где должен жить этот код?

Если ваш фон находится в простом старом PHP (без использования современных фреймворков), вы, вероятно, привыкли помещать код под корень документа веб-сервера (в таком месте, как **`/var/www`**). С Django вы этого не делаете. Не стоит поместить какой-либо из этого кода Python в корень документа вашего веб-сервера, потому что это рискует, что люди смогут просматривать ваш код через Интернет. Это не хорошо для безопасности.

Поместите свой код в какой-то каталог **за пределами** корня документа, например,**`/home/mycode`**.

Let's look at what **`startproject`** created:

```
mysite/
    manage.py
    mysite/
        __init__.py
        settings.py
        urls.py
        asgi.py
        wsgi.py
```

Эти файлы:

- The outer **`mysite/`** root directory is a container for your project. Its name doesn't matter to Django; you can rename it to anything you like.
- **`manage.py`**: A command-line utility that lets you interact with this Django project in various ways. You can read all the details about **`manage.py`** in django-admin and manage.py.
- The inner **`mysite/`** directory is the actual Python package for your project. Its name is the Python package name you'll need to use to import anything inside it (e.g. **`mysite.urls`**).
- **`mysite/__init__.py`**: Пустой файл, который сообщает Python, что этот каталог следует считать пакетом Python. Если вы новичок в Python, читайте больше о пакетах в официальных документах Python.
- **`mysite/settings.py`**: Настройки/конфигурация для этого проекта Django. Настройки Django 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 о URL-адресах в диспетчере URL-адресов.
- **`mysite/asgi.py`**: Точка входа для веб-серверов, совместимых с ASGI, для обслуживания вашего проекта. Более подробную информацию см. в разделе Как развертывать с ASGI.
- **`mysite/wsgi.py`**: Точка входа для WSGI-совместимых веб-серверов для обслуживания вашего проекта. Дополнительную информацию см. в разделе Как развертывать с WSGI.

Сервер разработки

Давайте проверим, что ваш проект Django работает. Перемените внешний каталог **`mysite`**, если вы еще этого не сделали, и выполните следующие команды:

```
$ python manage.py runserver
```

Вы увидите следующие выходные данные в командной строке:

```
Выполнение системных проверок...  Проверка системы не выявила
никаких проблем (0 заглушено).  У вас есть непримененные
миграции; ваше приложение может работать неправильно, пока они
не будут применены. Запустите 'python manage.py migration',
чтобы применить их.  30 апреля 2021 года - 15:50:53 Django
версии 3.2, используя настройки 'mysite.settings' Запуск сервера
разработки по адресу http://127.0.0.1:8000/Выйдите из сервера с
помощью CONTROL-C.
```

Примечание

Проигнорируйте предупреждение о миграции непримененных баз данных на данный момент; мы рассмотрим базу данных в ближайшее время.

Вы запустили сервер разработки Django, легкий веб-сервер, написанный исключительно на Python. Мы включили это в Django, чтобы вы могли быстро разрабатывать вещи, без необходимости иметь дело с настройкой производственного сервера, такого как Apache, пока не будете готовы к производству.

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

Теперь, когда сервер запущен, посетите http://127.0.0.1:8000/ с помощью веб-браузера. Вы увидите «Поздравления!» страница, с взлетавшейся ракетой. Это сработало!

Изменение порта

By default, the **`runserver`** command starts the development server on the internal IP at port 8000.

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

```
$ python manage.py runserver 8080
```

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

```
$ python manage.py runserver 0:8000
```

0 - это ярлык для **0.0.0.0**. Полные документы для сервера разработки можно найти в справочнике **`runserver`**.

Автоматическая перезагрузка`runserver`

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

Создание приложения Polls

Теперь, когда ваша среда - "проект" - настроена, вы готовы начать работу.

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, так что 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 веб-блогов, базу данных публичных записей или небольшое приложение для опроса. Проект - это набор конфигурации и приложений для конкретного веб-сайта. Проект может содержать несколько приложений. Приложение может быть в нескольких проектах.

Ваши приложения могут жить в любом месте по пути Python. В этом учебном пособии мы создадим наше приложение для опроса в том же каталоге, что и ваш файл **`manage.py`**, чтобы его можно было импортировать как собственный модуль верхнего уровня, а не как подмодуль **`mysite`**.

To create your app, make sure you're in the same directory as **`manage.py`** and type this command:

```
$ python manage.py startapp polls
```

That'll create a directory **`polls`**, which is laid out like this:

```
polls/
    __init__.py
    admin.py
    apps.py
    migrations/
        __init__.py
    models.py
    tests.py
    views.py
```

В этой структуре каталогов будет размещено приложение опроса.

Напишите свой первый вид

Let's write the first view. Open the file **`polls/views.py`** and put the following Python code in it:

`polls/views.py`

```
from django.http import HttpResponse


def index(request):
    return HttpResponse("Hello, world. You're at the polls
index.")
```

Это самый простой вид в Джанго. 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 ваш каталог приложений должен выглядеть следующим образом:

```
polls/
    __init__.py
    admin.py
    apps.py
    migrations/
        __init__.py
    models.py
    tests.py
    urls.py
    views.py
```

In the **`polls/urls.py`** file include the following code:

`polls/urls.py`

```
from django.urls import path

from . import views

urlpatterns = [
    path('', views.index, name='index'),
]
```

The next step is to point the root URLconf at the **`polls.urls`** module. In **`mysite/urls.py`**, add an import for **`django.urls.include`** and insert an **`include()`** in the **`urlpatterns`** list, so you have:

`mysite/urls.py`

```
from django.contrib import admin
from django.urls import include, path

urlpatterns = [
    path('polls/', include('polls.urls')),
    path('admin/', admin.site.urls),
]
```

The **`include()`** function allows referencing other URLconfs. Whenever Django encounters **`include()`**, it chops off whatever part of the URL matched up to that point and sends the remaining string to the included URLconf for further processing.

The idea behind **`include()`** is to make it easy to plug-and-play URLs. Since polls are in their own URLconf (**`polls/urls.py`**), they can be placed under "/polls/", or under "/fun_polls/", or under "/content/polls/", or any other path root, and the app will still work.

Когда использовать`include()`

You should always use **`include()`** when you include other URL patterns. **`admin.site.urls`** is the only exception to this.

You have now wired an **`index`** view into the URLconf. Verify it's working with the following command:

```
$ python manage.py runserver
```

Go to http://localhost:8000/polls/ in your browser, and you should see the text "*Hello, world. You're at the polls index.*", which you defined in the **`index`** view.

Страница не найдена?

Если вы получаете здесь страницу ошибки, убедитесь, что вы идете на http://localhost:8000/polls/, а не на http://localhost:8000/.

The **`path()`** function is passed four arguments, two required: **`route`** and **`view`**, and two optional: **`kwargs`**, and **`name`**. At this point, it's worth reviewing what these arguments are for.

`path()`аргумент: `route`

`route` is a string that contains a URL pattern. When processing a request, Django starts at the first pattern in **`urlpatterns`** and makes its way down the list, comparing the requested URL against each pattern until it finds one that matches.

Patterns don't search GET and POST parameters, or the domain name. For example, in a request to **`https://www.example.com/myapp/`**, the URLconf will look for **`myapp/`**. In a request to **`https://www.example.com/myapp/?page=3`**, the URLconf will also look for **`myapp/`**.

`path()`аргумент: `view`

When Django finds a matching pattern, it calls the specified view function with an **`HttpRequest`** object as the first argument and any "captured" values from the route as keyword arguments. We'll give an example of this in a bit.

`path()`аргумент: `kwargs`

Произвольные аргументы ключевого слова могут быть переданы в словарь в целевое представление. Мы не собираемся использовать эту функцию Django в учебнике.

`path()`аргумент: `name`

Именование URL-адреса позволяет однозначно ссылаться на него из других мест Джанго, особенно из шаблонов. Эта мощная функция позволяет вносить глобальные изменения в шаблоны URL-адресов вашего проекта, касаясь только одного файла.

Когда вы будете удовлетворены базовым потоком запросов и ответов, прочитайте часть 2 этого учебника, чтобы начать работу с базой данных.

‹ Краткое руководство по установке

Написание вашего первого приложения Django, часть 2 ›

Поддержите Django!

DjangoCongress JP 2019 (django-ja) пожертвовал Фонду программного обеспечения Django для поддержки разработки Django. Пожертвуйте сегодня!

Содержание

Просмотр

- Предыдущий: Краткое руководство по установке
- Далее: Написание вашего первого приложения Django, часть 2
- Содержание
- Общий индекс
- Индекс модуля Python

Вы находитесь здесь:

- Документация Django 3.2
  - Введение
    - Написание вашего первого приложения Django, часть 1

Получение помощи

Часто задаваемые вопросы
Попробуйте FAQ — в нем есть ответы на многие распространенные вопросы.

Индекс, индекс модуля или оглавление
Удобно при поиске конкретной информации.

список рассылки django-users
Найдите информацию в архивах списка рассылки django-users или оставьте вопрос.

#django IRC-канал
Задайте вопрос в IRC-канале #django или выполните поиск в журналах IRC, чтобы узнать, задавался ли он раньше.

Трекер билетов
Сообщите об ошибках с документацией Django или Django в нашем трекере билетов.

Скачать:

Автономный (Django 3.2): HTML | PDF | ePub
Предоставлено Read the Docs.

Дополнительная информация
О Django
Начало работы с Django
Организация команды
Django Software Foundation
Правила поведения
Заявление о разнообразии

Принять участие
Присоединиться к группе
Внести свой вклад в Django
Отправить ошибку
Сообщить о проблеме безопасности

Подписывайтесь на нас
GitHub
Twitter
Новости RSS
Список рассылки пользователей Django

Поддержать нас
Спонсор Джанго
Официальный магазин товаров
Улыбка Amazon
Программа благотворительности на рабочем месте

django
Хостинг по rackspace
Дизайн по threespot & andrevv
Получение помощи
Язык: **ru**
Версия документации: **3.2**