

Фреймворк Django



Знакомство и содержание урока



Давайте знакомиться!



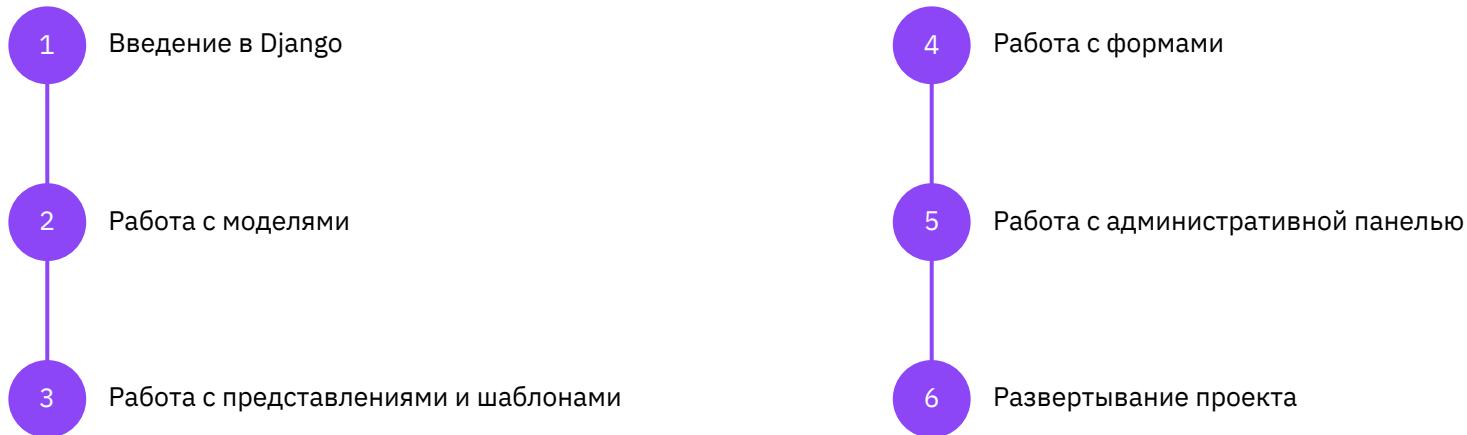
Алексей Петренко

Freelance developer

- 💥 Python-developer, Team Leader
- 💥 Опыт программирования более 20 лет
- 💥 Разрабатывал IT-решения для Министерства обороны РФ
- 💥 Преподаватель GeekBrains с 2018 года



План курса

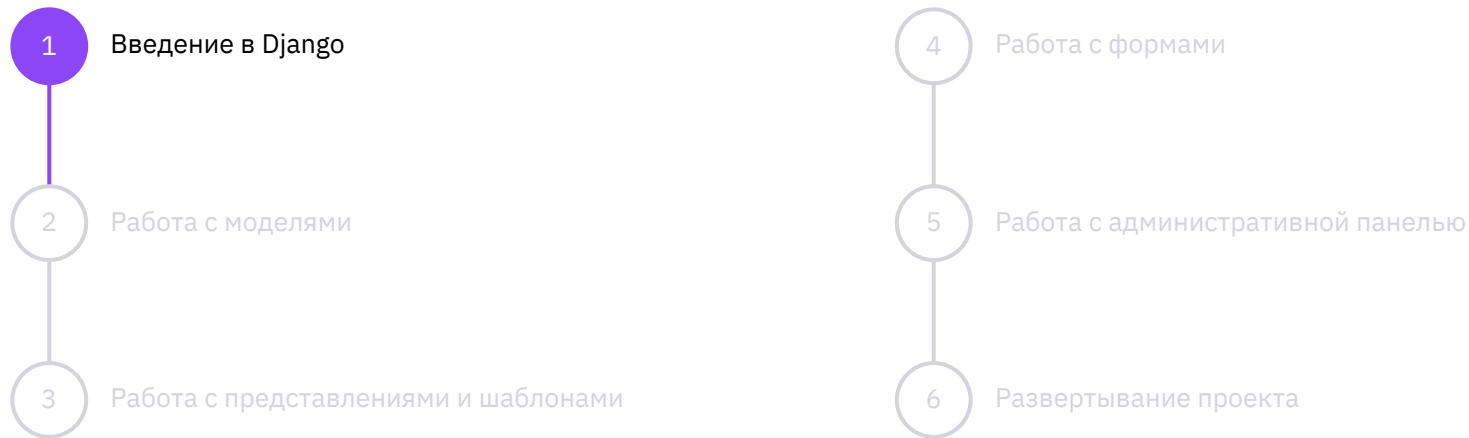


Введение в Django

Урок 1



План курса





Содержание урока





Что будет на уроке сегодня

- 📌 Узнаем о фреймворке Django
- 📌 Разберёмся в его установке и настройке для первого запуска
- 📌 Изучим структуру проекта и работу с ним
- 📌 Узнаем о приложениях как частях проекта
- 📌 Изучим настройки логирования в Django



Что такое Django?





Что такое Django?

Django - это высокоуровневый фреймворк для веб-приложений на языке Python. Он был создан в 2005 году и с тех пор активно развивается и обновляется сообществом разработчиков по всему миру.

Использование Django имеет множество преимуществ, таких как:

- 💡 Быстрая разработка веб-приложений
- 💡 Простота и удобство использования
- 💡 Высокая производительность
- 💡 Безопасность
- 💡 Масштабируемость



История версий

Сентябрь 2008 - версия 1.0.

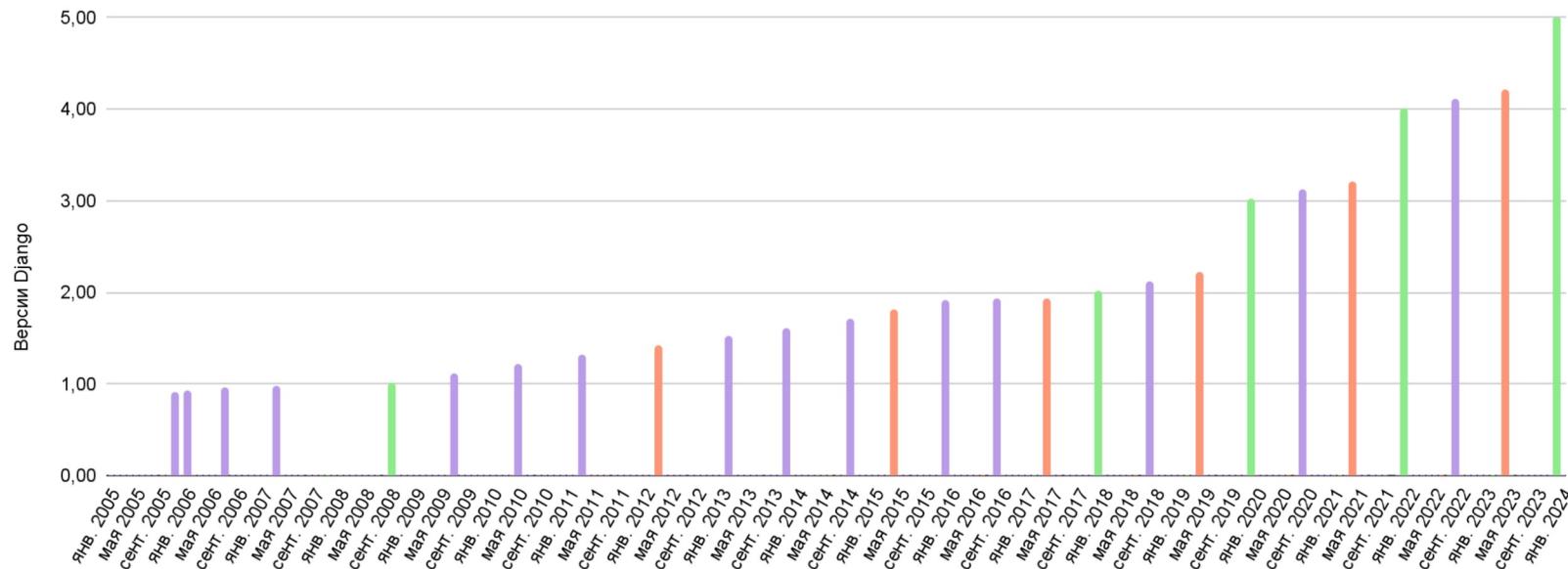
Декабрь 2017 - версия 2.0.

Март 2012 - первая LTS версия 1.4.

График выхода версий - каждый 8 месяцев, LTS и новая версия раз в

2 года

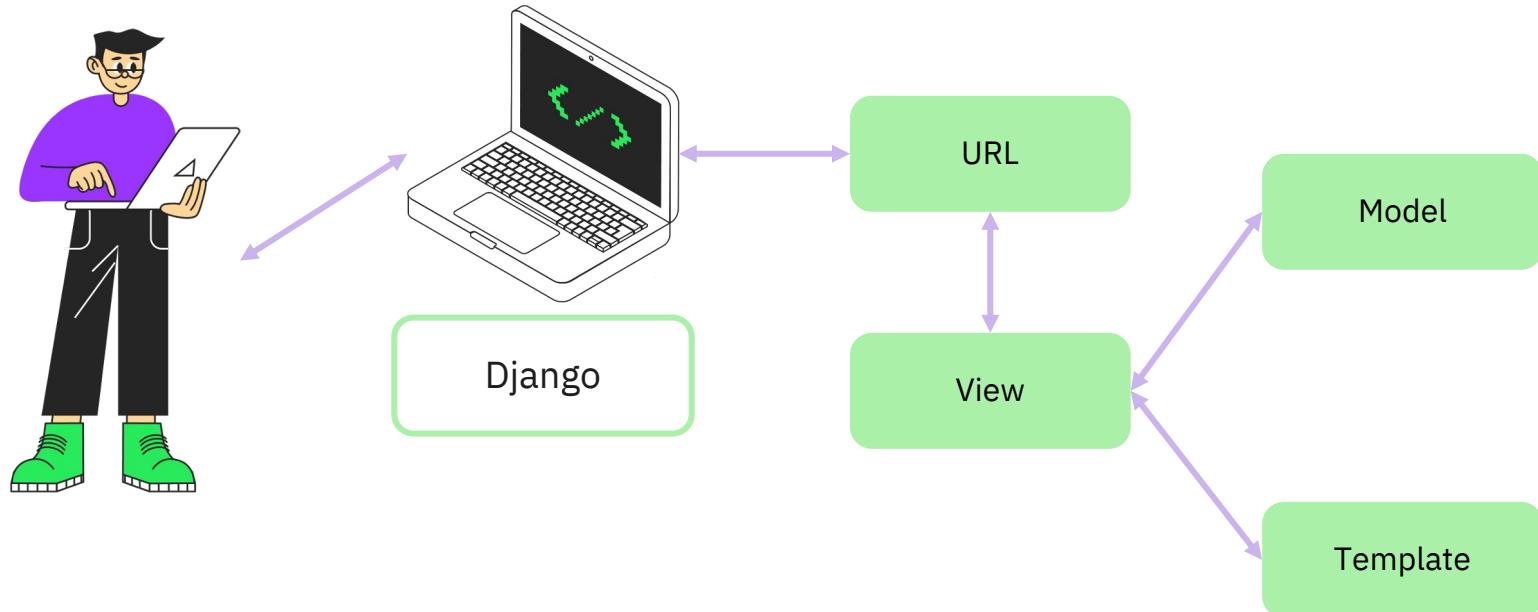
Месяц и год релиза версий Django





Концепция MVT

Концепция Модель-Представление-Шаблон (MVT) является основой фреймворка Django. Она разделяет приложение на три основных компонента: модель (Model), представление (View) и шаблон (Template). Маршруты (URL) играют роль посредника между пользователем и представлением.





Установка и настройка Django





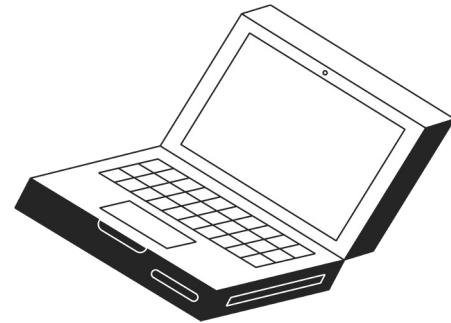
Установка и настройка Django

Как и с любым проектом, вначале создаём и активируем виртуальное окружение Python. Далее устанавливаем Django через менеджер пакетов PIP

```
mkdir project  
cd project  
python3 -m venv venv
```

```
venv/bin/activate # Linux/MacOS  
venv\Scripts\activate # Windows  
venv\Scripts\activate.ps1 # Windows PowerShell
```

```
pip install django
```





Создание первого
проекта

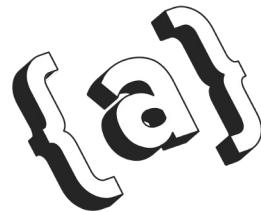




Создание проекта и обзор его структуры

Для создания нового проекта в Django необходимо выполнить команду `django-admin startproject <project_name>`

- Проект (код) создаётся один раз в начале проекта (реализации идеи)
- Проект является базовым каркасом Django. Хранит настройки и т.п.
- Проект наполняется приложениями (пару слайдов терпения) 😊
- Приложения не связаны между собой. Могут быть перенесены в другой проект





Запуск сервера и проверка работоспособности

Для запуска сервера необходимо выполнить команду

`python manage.py runserver`

Эта команда запустит сервер на локальном хосте и порту 8000.

После запуска сервера можно открыть браузер и перейти по адресу

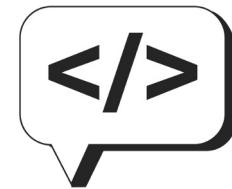
<http://localhost:8000/>

Особенности встроенного сервера

SECURITY WARNING: don't run with debug turned on in production!

`DEBUG = True`

You have **18** unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin, auth, contenttypes, sessions.





Пара дополнительных параметров runserver

При запуске сервера можно явно заменить порт 8000 на другой:

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

Кроме того можно изменить IP-адрес сервера.

Для этого он передаётся вместе с портом:

```
python manage.py runserver 0.0.0.0:8080
```

В константу ALLOWED_HOSTS файла settings.py необходимо добавить допустимые адреса в виде списка строк. Например так:

```
ALLOWED_HOSTS = [  
    '127.0.0.1',  
    '10.13.88.78',  
]
```



Создание первого
приложения





Создание приложения и обзор его структуры

Создание приложения в Django представляет собой создание отдельного модуля, который будет содержать логику и шаблоны для определенной функциональности.

Для создания приложения нужно выполнить команду "python manage.py startapp <app_name>", где <app_name> - название приложения.

```
python manage.py startapp myapp
```



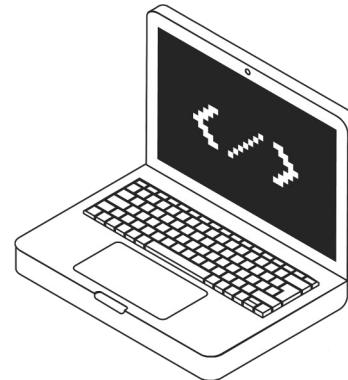


Добавление приложения в проект

Чтобы добавить созданное приложение в проект, необходимо указать его в настройках проекта (файл `settings.py`). Для этого нужно добавить название приложения в список `INSTALLED_APPS`

💡 Хорошой привычкой будет делать два действия сразу друг за другом. А именно:

- создавать приложение через `startapp`
- сразу добавлять его в список `INSTALLED_APPS`





Создание представления в приложении

Для создания представления нужно определить функцию в файле views.py, которая будет обрабатывать запрос пользователя

```
from django.http import HttpResponse

def index(request):
    return HttpResponse("Hello, world!")
```





Настройка путей

Для настройки URL в Django необходимо определить маршруты (routes), которые будут связывать определенные URL с соответствующими представлениями (views) в приложении.

urls.py проекта

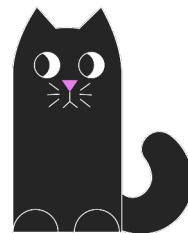
```
from django.contrib import admin
from django.urls import path, include

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

urls.py приложения

```
from django.urls import path
from . import views

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





Проверка работоспособности и «секреты» путей

Убедитесь, что сервер работает. Если нет, нужно выполнить команду:

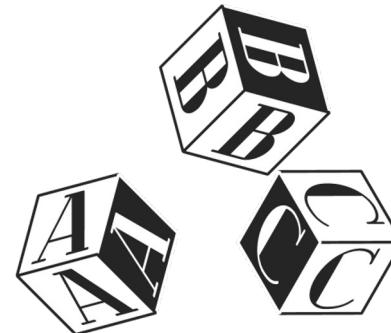
```
python manage.py runserver
```

После этого можно открыть браузер и перейти на страницу <http://127.0.0.1:8000/>

Исправим первый аргумент path в urls.py проекта и посмотрим на результат

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

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





Логирование в Django

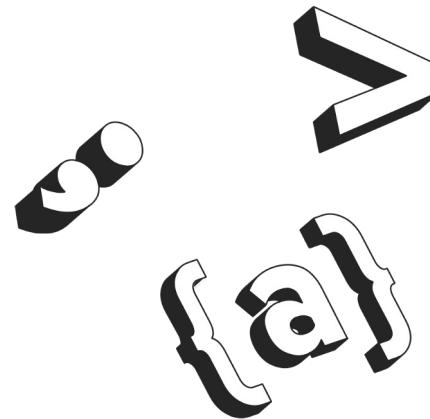




Что такое логирование и зачем оно нужно

Логирование - это процесс записи информации о работе приложения.

Логи позволяют отслеживать работу приложения, выявлять ошибки и проблемы, а также анализировать производительность приложения.

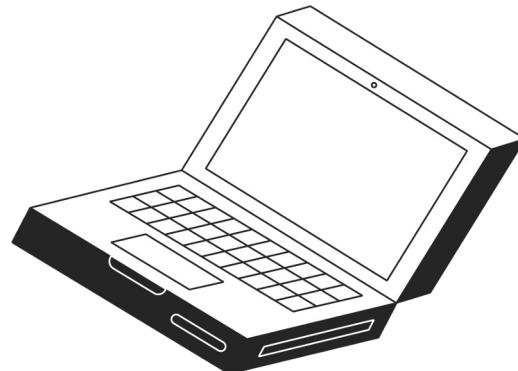




Конфигурация логирования в Django

Конфигурация логирования в файле settings.py в Django происходит через словарь LOGGING

```
...
LOGGING = {
    'version': 1,
    'disable_existing_loggers': False,
    'handlers': {
        ...
    }
}
```





Запись логов

Для использования логирования в Django необходимо импортировать модуль `logging` и создать объект логгера

```
import logging

from django.http import HttpResponse

logger = logging.getLogger(__name__)

def index(request):
    logger.info('Index page accessed')
    return HttpResponse("Hello, world!")
```

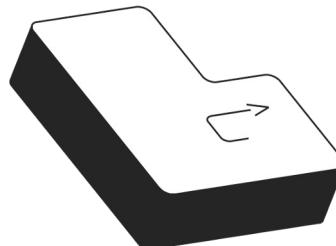


Добавление форматирования в логи

Как и в случае со стандартным `logging` из Python мы можем прописать формат вывода сообщений. Например такая строка добавляет вывод уровня предупреждения и само сообщение:

```
{'format': '%(levelname)s %(message)s'}
```

<https://docs.python.org/3/library/logging.html>





Итоги занятия





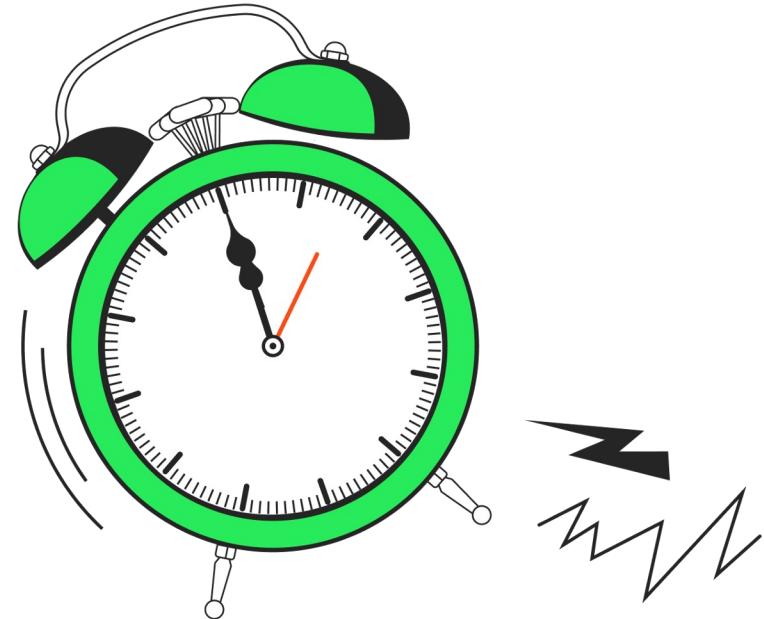
На этой лекции мы

- 📌 Узнали о фреймворке Django
- 📌 Разобрались в его установке и настройке для первого запуска
- 📌 Изучили структуру проекта и работу с ним
- 📌 Узнали о приложениях как частях проекта
- 📌 Изучили настройки логирования в Django



Задание

1. Для закрепления материалов лекции попробуйте самостоятельно набрать и запустить демонстрируемые примеры.





Спасибо за внимание

