Веб-разработка на Django

Мастер-класс

❓ Что такое Django?

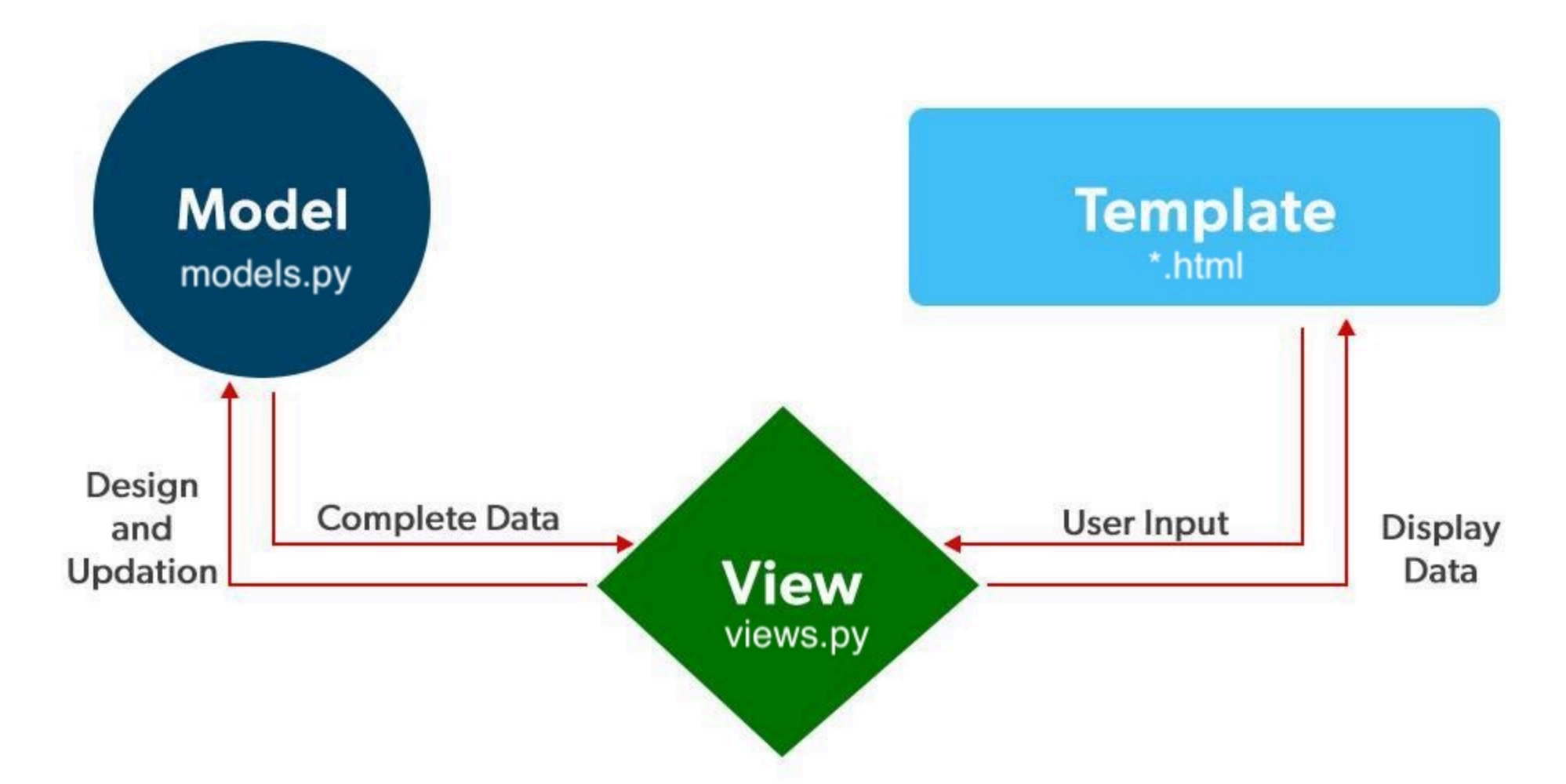
Официально: "Высокоуровневый Python веб-фреймворк, который поощряет быструю разработку и чистый, прагматичный дизайн"

Неофициально: "Веб-фреймворк для перфекционистов с дедлайнами"

❓ Почему именно Django?

"Батарейки в комплекте" – Django поставляется "из коробки" с решениями для 90% типовых задач

Архитектура Model-View-Template



Как это работает вместе

1. Пользователь заходит на URL.
2. Django находит нужный View через urls.py.
3. View обращается к Model за данными.
4. Model (через ORM) делает запрос к Базе Данных.
5. View получает данные и передает их в Template.
6. Template генерирует HTML.
7. Пользователь видит готовую страницу.

❓ Что мы изучим сегодня

- ✅ Работу с базой данных
- ✅ Модели Django
- ✅ Админ-панель Django
- ✅ Представления (views)
- ✅ Систему шаблонов
- ✅ URL-маршрутизацию

Установка Python

python.org/downloads

docs.astral.sh/uv/getting-started/installation

Настройка виртуального окружения

```
python -m venv venv
source venv/bin/activate # Для Linux/Mac
# или
venv\Scripts\activate # Для Windows
pip install django
django-admin startproject todo_list .
python manage.py startapp tasks
```

Либо же (с 📎 uv)

```
uv venv
uv pip install django
uv run django-admin startproject todo_list .
uv run manage.py startapp tasks
```

🚀 Запуск проекта

```
python manage.py runserver
# Либо же
uv run manage.py runserver
```

📎 Полный список команд manage.py

Структура проекта

```
.
├── manage.py               # Скрипт для выполнения административных задач
├── tasks
│   ├── __init__.py
│   ├── admin.py            # Управление админкой
│   ├── apps.py             # Конфиг приложения
│   ├── migrations          # Миграции базы данных
│   │   └── __init__.py
│   ├── models.py           # Модели Django
│   ├── tests.py            # Тесты
│   ├── templates           # Шаблоны HTML
│   ├── urls.py             # URL-маршруты
│   └── views.py            # Посредник между моделями и html-страницами
└── todo_list
    ├── __init__.py
    ├── asgi.py             # Точка входа для запуска асинхронного интерфейса приложения
    ├── settings.py         # Настройки проекта
    ├── urls.py             # Центральная точка URL-маршрутизации
    └── wsgi.py             # Точка входа для запуска синхронного интерфейса приложения
```

⚙️ Настройки проекта (settings.py)

📎 Полный список настроек

BASE_DIR	Корневая директория
INSTALLED_APPS	Компоненты проекта
TEMPLATES	Конфиг системы шаблонов
DATABASES	Настройки БД

Добавим наше приложение в `INSTALLED_APPS`

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'tasks',
]
```

Модели Django

📎 Документация по моделям

```
class Task(models.Model):
    title = models.CharField("Название", max_length=255, unique=True)
    description = models.TextField("Описание", blank=True)
    completed = models.BooleanField("Выполнено", default=False)
    created_at = models.DateTimeField("Дата создания", auto_now_add=True)

    class Meta:
        verbose_name = "Задача"
        verbose_name_plural = "Задачи"
        ordering = ['-created_at']

    def __str__(self):
        return self.title

    def get_absolute_url(self):
        return reverse("task_detail", kwargs={"pk": self.pk})
```

Миграции

Создаём файл миграции

```
python manage.py makemigrations
# Либо же
uv run manage.py makemigrations
```

Применяем миграцию к БД

```
python manage.py migrate
# Либо же
uv run manage.py migrate
```

Админ-панель

Создадим суперпользователя

```
python manage.py createsuperuser
# Либо же
uv run manage.py createsuperuser
```

Зарегистрируем модель

```
from django.contrib import admin

from .models import Task

admin.site.register(Task)
```

http://127.0.0.1:8000/admin

Представления (views)

```
from django.shortcuts import render

from .models import Task


def task_list(request):
    tasks = Task.objects.all()
    return render(request, 'task_list.html', {'tasks': tasks})

def task_detail(request, pk):
    task = Task.objects.get(pk=pk)
    return render(request, 'task_detail.html', {'task': task})
```

Шаблоны

```
<!DOCTYPE html>
<head><title>Задачи</title></head>
<body>
    <h1>Задачи</h1>
    <p class="meta">Всего: {{ tasks|length }}</p>
    {% if tasks %}
        <ul>
            {% for task in tasks %}
                <li>
                    <div>
                        <a href="{% url 'task_detail' task.pk %}">{{ task.title }}</a>
                        <div class="meta">Создано: {{ task.created_at|date:"Y-m-d H:i" }}</div>
                    </div>
                    <span class="badge {% if task.completed %}done{% else %}todo{% endif %}">
                        {% if task.completed %}Выполнено{% else %}В работе{% endif %}
                    </span>
                </li>
            {% endfor %}
        </ul>
    {% else %}
        <p class="empty">Список задач пуст.</p>
    {% endif %}
</body>
</html>
```

Шаблоны

```
<!DOCTYPE html>
<head><title><title>{{ task.title }}</title></title></head>
<body>
    <div class="crumbs"><a href="{% url 'task_list' %}">← Назад к списку</a></div>
    <div class="card">
        <h1>{{ task.title }}</h1>
        <div class="meta">
            Статус:
            <span class="badge {% if task.completed %}done{% else %}todo{% endif %}">
                {% if task.completed %}Выполнено{% else %}В работе{% endif %}
            </span>
            · Создано: {{ task.created_at|date:"Y-m-d H:i" }}
        </div>
        {% if task.description %}
            <div class="desc">{{ task.description }}</div>
        {% else %}
            <div class="desc" style="color:#6b7280;font-style:italic;">Описание отсутствует.</div>
        {% endif %}
    </div>
</body>
</html>
```

URL-маршрутизация

Последний шаг: "прикручиваем" наши View к URL

Основной urls.py

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

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

📎 Документация

URL-маршрутизация

Последний шаг: "прикручиваем" наши View к URL

tasks/urls.py

```
from django.urls import path

from . import views

urlpatterns = [
    path('', views.task_list, name='task_list'),
    path('<int:pk>/', views.task_detail, name='task_detail'),
]
```

📎 Документация

Код проекта

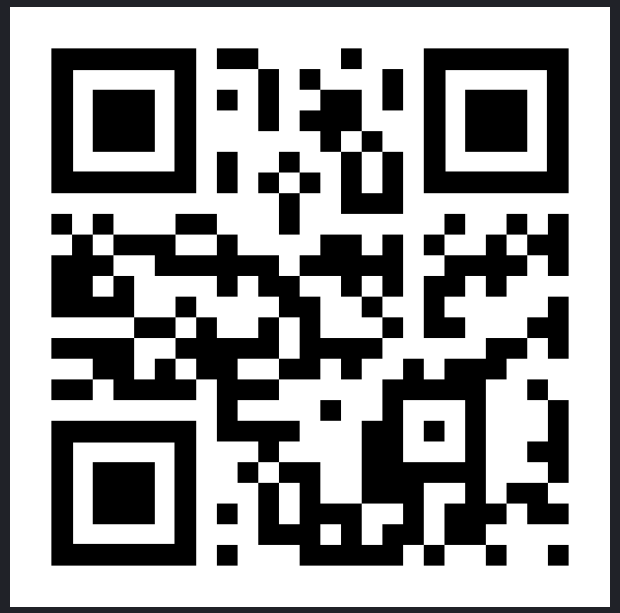
github.com/vyhuholl/todo_list

📚 Полезные ресурсы

- 📎 Официальная документация
- 📎 Туториал
- 📎 Django Community
- 📎 MDN Web Docs (Django)

📬 Остаёмся на связи!

@DebugSkills



@IT_Chuyana

@olgap981