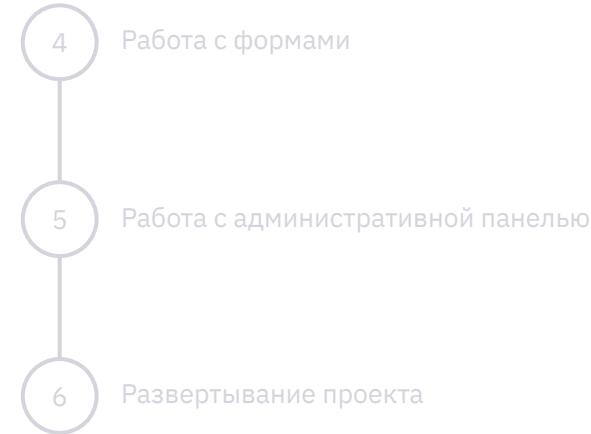
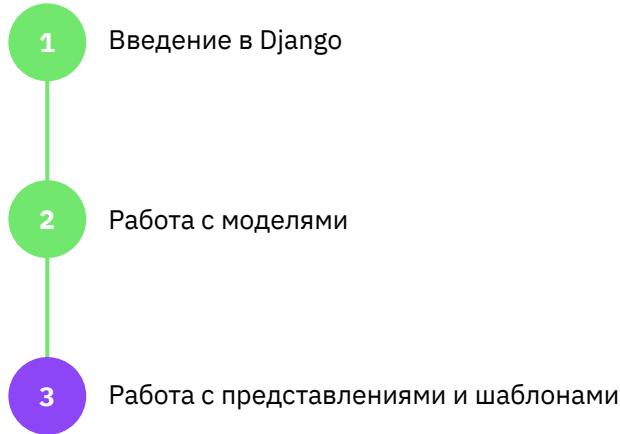


Работа с представлениями и шаблонами

Урок 3



План курса





Содержание урока





Что будет на уроке сегодня

- 📌 Узнаем о представлениях Django
- 📌 Разберёмся в работе диспетчера URL
- 📌 Изучим шаблоны и передачу контекста в них
- 📌 Узнаем о условиях, циклах и наследовании шаблонов
- 📌 Объединяем модели, представления, шаблоны и маршруты



Введение в MVT(U)





Введение в MVT(U)

Представление в Django – это функция или класс, которая обрабатывает запрос и возвращает ответ в виде HTTP-ответа.

Диспетчер URL Django отвечает за обработку входящих запросов и направление их на соответствующие обработчики.

Шаблоны в Django представляют собой файлы, содержащие HTML-код с дополнительными тегами и переменными, которые могут быть заменены на значения из контекста.





Представления





Представления

Представления располагаются в файле views.py вашего приложения. Если проект состоит из нескольких приложений, каждое будет иметь свои “вьюшки” в собственном каталоге



Представления на основе функций

```
from django.http import  
HttpResponse  
  
def hello(request):  
    return HttpResponse("Hello  
World from function!")
```



Представления на основе классов

```
from django.views import View  
from django.http import HttpResponse  
  
class HelloView(View):  
    def get(self, request):  
        return HttpResponse("Hello World from  
class!")
```



Диспетчер URL





Сопоставление представления с маршрутом

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('les3/', include('myapp3.urls')),
]
```



urls.py приложения

```
from django.urls import path
from .views import hello, HelloView

urlpatterns = [
    path('hello/', hello, name='hello'),
    path('hello2/', HelloView.as_view(),
         name='hello2'),
]
```



Передача параметров

Преобразования пути – это процесс преобразования URL-адреса запроса в формат, понятный для Django

```
from django.urls import path
...
from .views import year_post, MonthPost, post_detail

urlpatterns = [
    ...
    path('posts/<int:year>', year_post, name='year_post'),
    path('posts/<int:year>/<int:month>/<slug:slug>', post_detail,
         name='post_detail'),
    ...
]
```



Преобразование пути в типы Python

Преобразование путей осуществляется с помощью приставок, которые определяют тип передаваемых в качестве параметра в представление данных

- 💡 str — приставка для передачи строки любых символов, кроме слэша.
- 💡 int — приставка для передачи целого числа.
- 💡 slug — приставка для передачи строки, содержащей только буквы, цифры, дефисы и знаки подчеркивания.
- 💡 uuid — приставка для передачи уникального идентификатора.
- 💡 path — приставка для передачи строки любых символов, включая слэши.



Из URL во view

Представление в качестве параметров получаем значения в python объектов.

Например можно передать год и месяц, если перейти по адресу

<http://127.0.0.1:8000/les3/posts/2022/6/>

```
from django.views import View
from django.http import HttpResponse

class MonthPost(View):
    def get(self, request, year, month):
        text = ""
        ... # формируем статьи за год и месяц
        return HttpResponse(f"Posts from {month}/{year}<br>{text}")
```



Шаблоны





Каталог шаблона

Внутри каталога приложения необходимо создать каталог `templates`. Далее в нём создаётся каталог с именем приложения.

```
myproject/
    myapp1/
        templates/
            myapp1/
                index.html
            ...
    myapp2/
        templates/
            myapp2/
                index.html
            ...
myproject/
    ...
manage.py
```



Передача контекста в шаблон

Чтобы передать данные в шаблон, мы можем использовать функцию `render` из модуля `django.shortcuts`

```
from django.shortcuts import render

def my_view(request):
    context = {"name": "John"}
    return render(request, "myapp3/my_template.html", context)
```



Проверка условия в шаблонах

В Django условные операторы в шаблонах имеют синтаксис похожий на Python и заключаются в фигурные скобки вида `{% %}`.

Обязательным является закрывающий оператор кода вида `{% endif %}`.

```
{% if message %}
    <p>Вам доступно сообщение: <br> {{ message }}</p>
{% endif %}
```



Вывод в цикле

В Django для вывода данных в цикле используется тег `for`.

Запись аналогична проверке условий

```
{% for item in my_list %}
    <li>{{ item }}</li>
{% endfor %}
```



Наследование шаблонов Django

Дочерние шаблоны расширяют содержимое базового шаблона



Базовый шаблон

```
...
{% block content %}
    Страница не заполнена
{% endblock %}
...
...
```



Дочерний шаблон

```
{% extends 'myapp3/base.html' %}

{% block content %}
    <h1 class="display-2">Страница заполнена</h1>
{% endblock %}
```



Объединяем
модели, представления,
шаблоны и маршруты





Модели

В файле models.py приложения сохраняем код моделей.

Обязательно создаём и применяем миграции

```
from django.db import models

class Author(models.Model):
    name = models.CharField(max_length=100)
    ...
    ...
```



Представления

В файле views.py приложения сохраняем код представлений, которые получают данные из моделей и передают их в шаблоны

```
from django.shortcuts import render, get_object_or_404
from .models import Author, Post

def post_full(request, post_id):
    post = get_object_or_404(Post, pk=post_id)
    return render(request, 'myapp3/post_full.html', {'post': post})
```



Маршруты

В файле urls.py приложения сохраняем код маршрутов.

Они соединяют URL-адреса с представлениями

```
from django.urls import path
...
from .views import author_posts, post_full

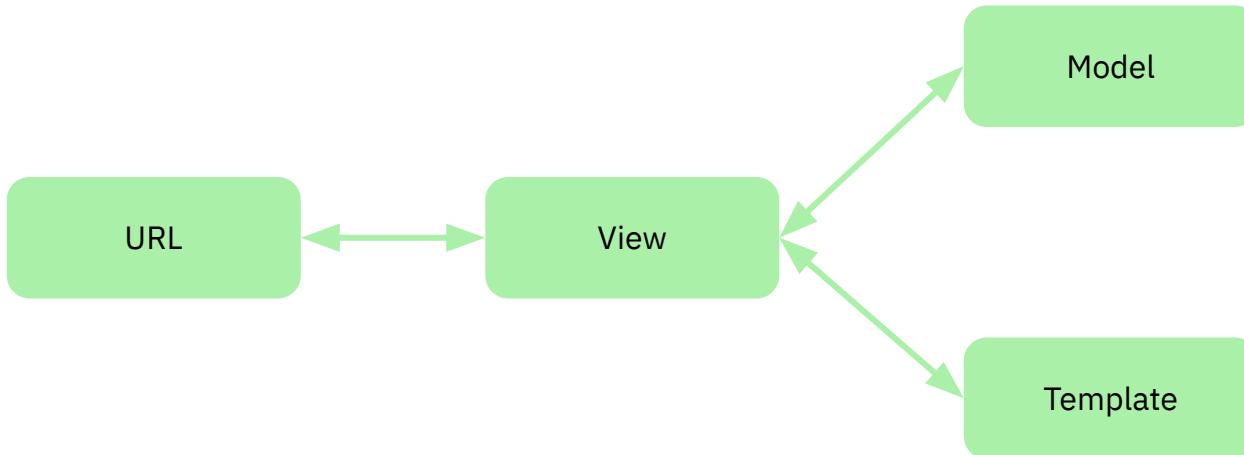
urlpatterns = [
    ...
    path('post/<int:post_id>', post_full, name='post_full'),
]
```



Шаблоны

Расположите базовый шаблон в каталоге templates проекта.

Дочерние шаблоны сохраняйте в каталоге templates/myapp3/ приложения.





Итоги занятия





На этой лекции мы

- 📌 Узнали о представлениях Django
- 📌 Разобрались в работе диспетчера URL
- 📌 Изучили шаблоны и передачу контекста в них
- 📌 Узнали о условиях, циклах и наследовании шаблонов
- 📌 Объединили модели, представления, шаблоны и маршруты



Задание



1. Для закрепления материалов лекции попробуйте самостоятельно набрать и запустить демонстрируемые примеры.
2. *Загляните в официальную документацию Django и изучите дополнительные возможности работы со статическими файлами.



Спасибо за внимание

