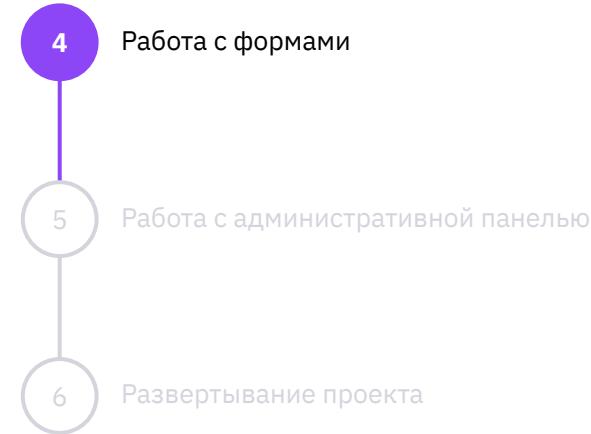
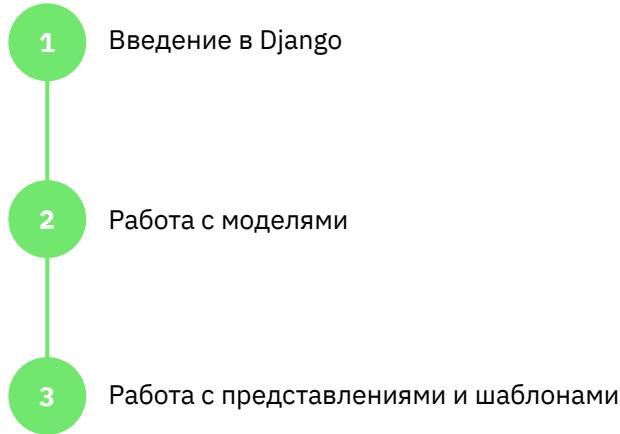


Работа с формами

Урок 4



План курса





Содержание урока





Что будет на уроке сегодня

- 📌 Узнаем о формах Django
- 📌 Разберёмся в создании классов форм
- 📌 Изучим поля и виджеты форм
- 📌 Узнаем о обработке данных из формы
- 📌 Разберёмся в сохранении файлов



Что такое формы
в Django?





Формы Django

Формы в Django — это инструмент, который позволяет создавать и обрабатывать HTML-формы. Формы используются для сбора данных от пользователей и отправки их на сервер для дальнейшей обработки.





Создание форм





Класс формы

В Django для создания форм используются классы, которые наследуются от класса forms.Form.

```
from django import forms

class UserForm(forms.Form):
    name = forms.CharField(max_length=50)
    email = forms.EmailField()
    age = forms.IntegerField(min_value=0, max_value=120)
```



Из класса формы в HTML форму



Представление для формы

- Проверка на POST и GET
- Создание объекта формы
- Проверка валидации
- Обработка данных
- Возврат формы



Прописываем url

- Маршрут для связи представления с URL-адресом



Отрисовка шаблона

- Вывод тегов формы с указанием метода и адреса
- Вывод csrf-токена
- Вывод элементов формы
- Вывод кнопки “Отправить”



Поля и виджеты форм





Поля форм

Перечислим некоторые из наиболее популярных классов Field в Django

- 💡 CharField – используется для создания текстовых полей
- 💡 EmailField – используется для создания поля электронной почты
- 💡 IntegerField – используется для создания поля для ввода целых чисел.
- 💡 FloatField – используется для ввода чисел с плавающей точкой.
- 💡 BooleanField – используется для создания поля флагка.
- 💡 DateTimeField – используется для создания поля даты и времени.
- 💡 FileField – используется для создания поля для загрузки файла.
- 💡 ImageField – используется для создания поля для загрузки изображения.
- 💡 ChoiceField – используется для создания выпадающего списка с выбором одного из нескольких вариантов.



Виджеты форм

Для работы с формами используются виджеты, которые определяют внешний вид и поведение полей формы на странице

-  TextInput — используется для создания текстового поля ввода.
-  Textarea — используется для создания многострочного текстового поля ввода.
-  PasswordInput — используется для создания поля ввода пароля.
-  NumberInput — используется для создания поля ввода чисел.
-  CheckboxInput — используется для создания флажка.
-  DateTimeInput — используется для создания поля ввода даты и времени.
-  FileInput — используется для создания поля загрузки файла.
-  Select — используется для создания выпадающего списка с выбором одного из нескольких вариантов.
-  RadioSelect — используется для создания списка радиокнопок.



Обработка данных форм





Пользовательская валидация данных с помощью метода `clean()`

Мы можем прописать свои методы, которые начинаются со слова `clean_` и далее указать имя поля.

```
class UserForm(forms.Form):
    email = forms.EmailField()
    ...

    def clean_email(self):
        email = self.cleaned_data['email']
        if not (email.endswith('vk.team') or email.endswith('corp.mail.ru')):
            raise forms.ValidationError('Используйте корпоративную почту')
        return email

    ...
```



Сохранение формы в базу данных

После того как форма заполнена пользователем, отправлена Django, проверена и прошла валидацию данные можно использовать.

Обычно использование — это сохранение в базу данных

```
...
def add_user(request):
    if request.method == 'POST':
        form = UserForm(request.POST)
        if form.is_valid():
            name = form.cleaned_data['name']
            email = form.cleaned_data['email']
            user = User(name=name, email=email)
            user.save()
        ...
    ...
```



Сохранение изображений /файлов





Особенности сохранения изображений / файлов



Настройка settings.py

```
...
MEDIA_URL = '/media/'
MEDIA_ROOT = BASE_DIR / 'media'
...
```



Представление views.py

```
...
form = ImageForm(request.POST,
request.FILES)
if form.is_valid():
    image = form.cleaned_data['img']
    fs = FileSystemStorage()
    fs.save(image.name, image)
...
```



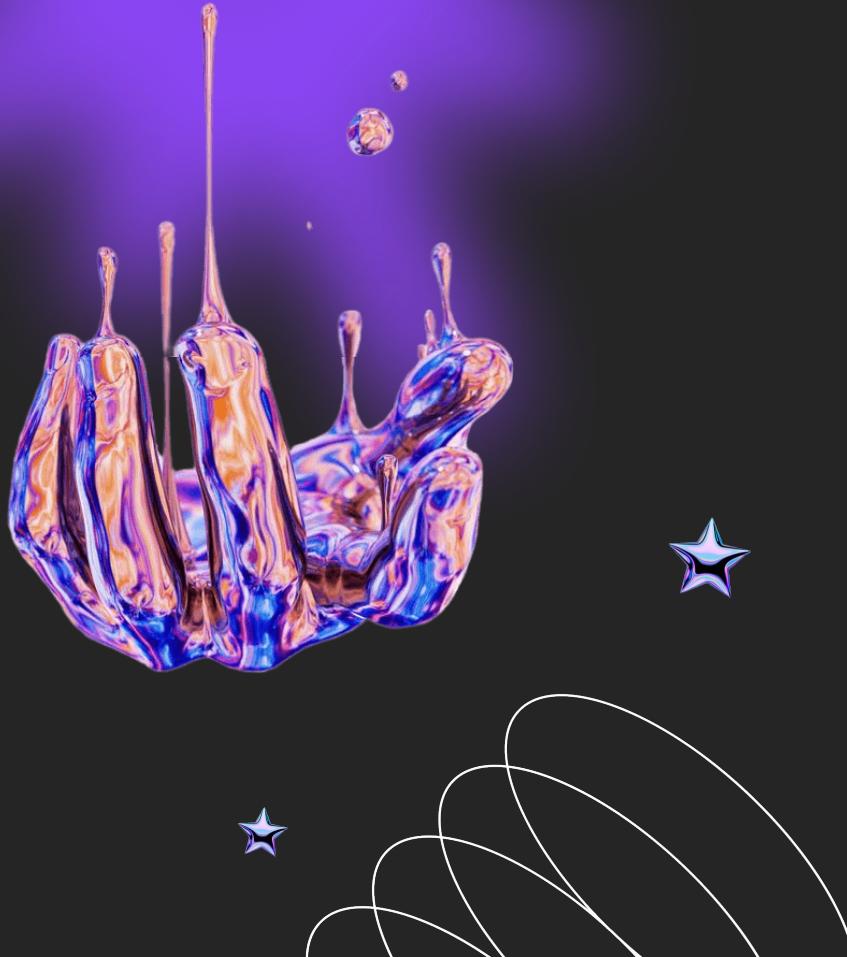
Шаблон upload.html

```
...
<form method="post"
enctype="multipart/form-data">
...

```



Итоги занятия





На этой лекции мы

- 📌 Узнали о формах Django
- 📌 Разобрались в создании классов форм
- 📌 Изучили поля и виджеты форм
- 📌 Узнали о обработке данных из формы
- 📌 Разобрались в сохранении файлов



Задание

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



Спасибо за внимание

