

Лайвкодинг Декораторы. FBV vs CBV. Работа с формами.

Михаил Землянухин

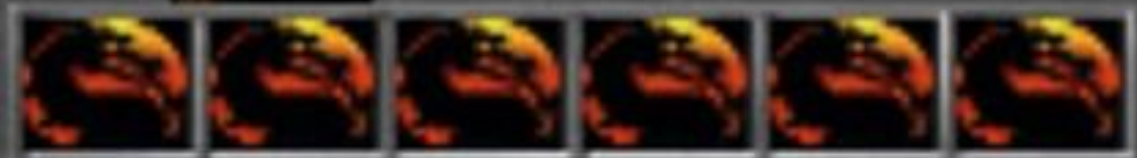
Яндекс Практикум

Лайвкодинг Прототитирование

FBV

CBV

vs



Представления на основе функций (FBV)

Плюсы и минусы

Плюсы

- Явный поток кода (у вас есть полный контроль над тем, что происходит)

- Проста в реализации

- Легко понять

- Отлично подходит для уникальной логики представления

- Легко интегрировать с декораторами

Против

- Много повторяющегося кода

- Обработка HTTP методов через условное ветвление

- Не используют преимущества ООП

- Труднее поддерживать

Представления на основе классов (CBV)

Плюсы и минусы

Плюсы

- Являются расширяемыми

- Они используют преимущества концепций ООП (самое главное - наследование)

- Отлично подходят для написания CRUD представлений

- Более чистый и многократно используемый код

- Встроенные в Django общие CBVs

- Они похожи на представления REST фреймворка Django

Против

- Неявный поток кода (многое происходит в фоновом режиме)

- Используется много миксинов, что может запутать

- Более сложный и трудный для освоения

- Декораторы требуют дополнительного импорта или переопределения кода

Работа с формами

Формы

- Отправка пользовательских данных на сервер.
- Использует http-методы
- Данные нужно валидировать.

Shipping Address

Name:

Address:

City:

State:

Zip:

```
<form action="formprocessor.html" method="get">
  <fieldset>
    <legend>Shipping Address</legend>
    <p>Name: <input type="text" name="Name" /></p>
    <p>Address: <input type="text" name="Address" /></p>
    <p>City: <input type="text" name="City" /></p>
    <p>State: <input type="text" name="State" /></p>
    <p>Zip: <input type="text" name="Zip" /></p>
  </fieldset>
</form>
```

Формы

```
1 <input type="button" value="Кнопка" />
2 <br /><br />
3 <label><input type="checkbox" /> Выбор №1</label>
4 <label><input type="checkbox" /> Выбор №2</label>
5 <label><input type="checkbox" /> Выбор №3</label>
6 <br /><br />
7
8 <input type="file" />
9 <br />
10
11 <input type="hidden" />
12 <br />
13
14 <input type="password" />
15 <br /><br />
16
17 <label><input type="radio" /> Выбор №1</label>
18 <label><input type="radio" /> Выбор №2</label>
19 <label><input type="radio" /> Выбор №3</label>
20 <br /><br />
21
22 <input type="reset" />
23 <br /><br />
24
25 <input type="submit" />
26 <br /><br />
27
28 <input type="text" />
29 <br />
```

HTML

Кнопка

☐ Выбор №1 ☐ Выбор №2 ☐ Выбор №3

Выберите файл Файл не выбран

.....

☐ Выбор №1 ☐ Выбор №2 ☐ Выбор №3

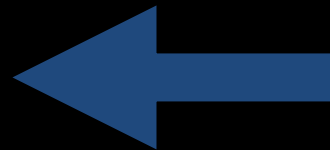
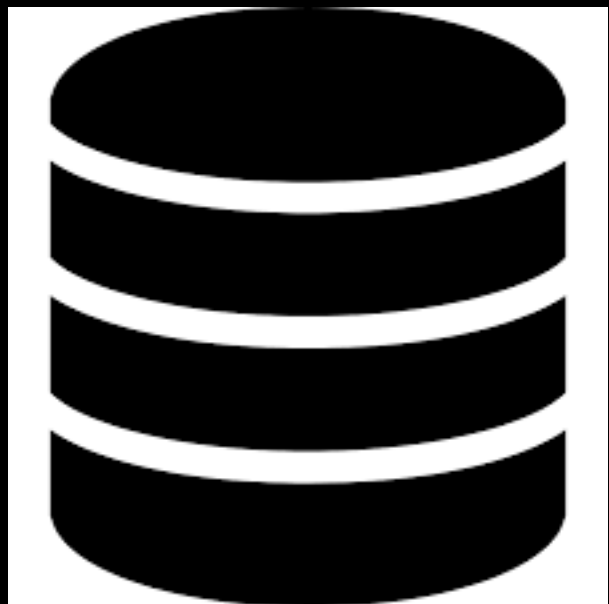
Сбросить

Отправить

Текстовое поле

Зачем нам формы?

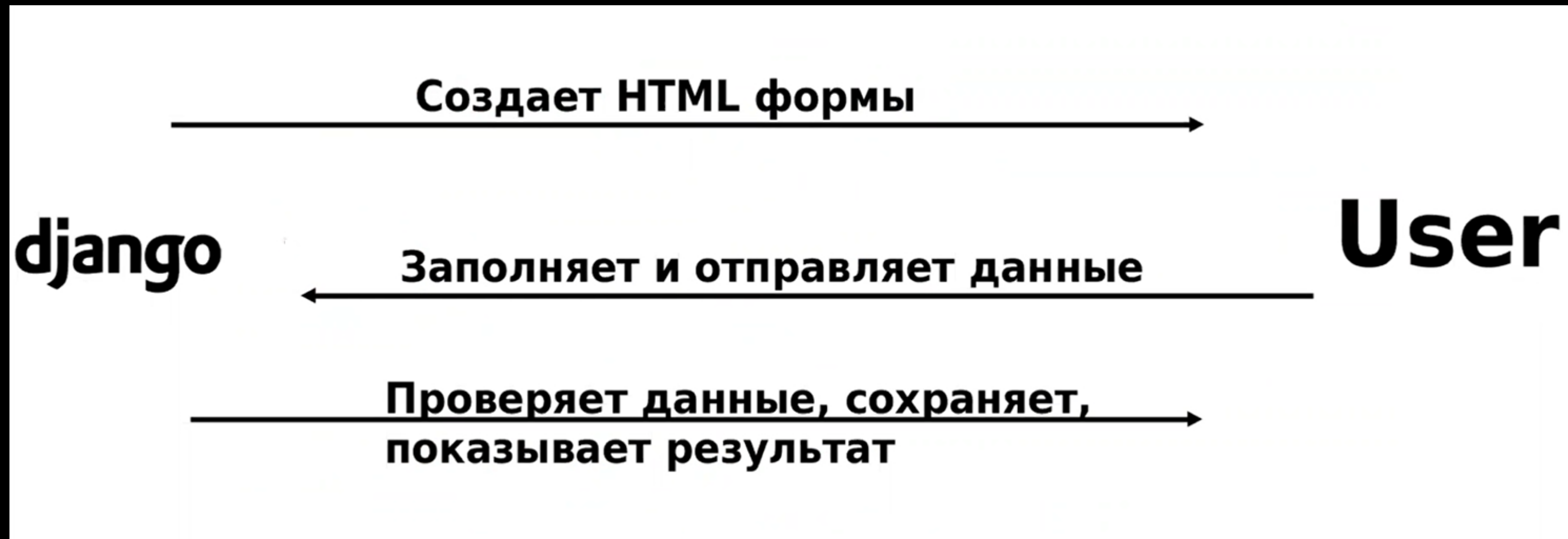
Валидация
вХОДНЫХ данных



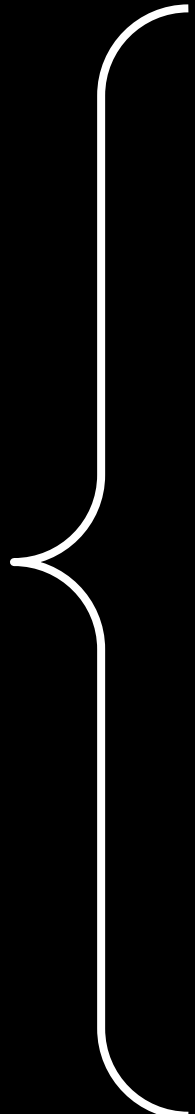
Отрисовка
html-формы

```
<form>
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname"><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname">
</form>
```

Формы “под капотом”



Формы “под капотом”

<code>class Form</code>		Поля		Виджеты
		<code>CharField</code>	—	<code>TextInput</code>
		<code>IntegerField</code>	—	<code>NumberInput</code>
		<code>BooleanField</code>	—	<code>CheckboxInput</code>
		<code>DateField</code>	—	<code>DateInput</code>
		<code>EmailField</code>	—	<code>EmailInput</code>
		<code>ChoiceField</code>	—	<code>Select</code>

Вывод формы в шаблоне

По умолчанию

```
<form method="post">  
    {% csrf_token %}  
    {{ form }}  
    <input type="submit" value="Submit">  
</form>
```


Вывод формы в шаблоне

Ближе к реальности

```
<form method="post">
    {% csrf_token %}
    {% for field in form %}
        <div class="field_wrapper">
            {{ field.errors }}
            {{ field.label_tag }} {{ field }}
            {% if field.help_text %}
                <p class="help">{{ field.help_text|safe }}</p>
            {% endif %}
        </div>
    {% endfor %}
    <input type="submit" value="Submit">
</form>
```