

< Teach
Me
Skills />

Занятие 17.

Django. REST. Templates

1

REST

REST - Representation State Transfer

- Передача репрезентативного состояния
 - Это стиль архитектуры для веб-приложений
 - Системы, поддерживающие этот стиль, называются RESTful
 - Нужен для того чтобы системы могли легко обмениваться данными и приложение можно было масштабировать
 - Чаще всего реализован через HTTP протокол
-
- Описывает четкий стандартизированный интерфейс
 - Каждая единица информации однозначно определяется глобальным идентификатором URL определенного формата

REST - Representation State Transfer

- **GET /book/** - получить список всех книг
- **GET /book/1** - получить одну книгу по id
- **PUT /book/** - добавить книгу (данные в теле запроса)
- **POST /book/3** - изменить книгу (данные в теле запроса)
- **DELETE /book/3** - удалить книгу по id

- **GET /book/3/page/89** - получить определенную страницу книги

REST - Принципы

1- **Единый интерфейс** (передаем информацию в стандартном формате):

Единый интерфейс накладывает четыре архитектурных ограничения:

- Запросы должны идентифицировать ресурсы. Это происходит за счет единого идентификатора ресурсов.
- Клиенты имеют достаточно информации в представлении ресурса, чтобы при желании изменить или удалить ресурс. Сервер выполняет это условие, отправляя метаданные, которые дополнительно описывают ресурс.
- Клиенты получают информацию о дальнейшей обработке представлений. Сервер реализует это, отправляя описательные сообщения, где содержатся метаданные о том, как клиент может использовать их оптимальным образом.
- Клиенты получают информацию обо всех связанных ресурсах, необходимых для выполнения задачи. Сервер реализует это, отправляя гиперссылки в представлении, чтобы клиенты могли динамически обнаруживать больше ресурсов.

REST - Принципы

2- **Отсутствие состояния** - каждый запрос выполняется вне зависимости от всех предыдущих

3- **Многоуровневая система** - клиент может подключаться к любым авторизованным посредникам и получать ответы от сервера. Сервер также может отправлять запросы куда-то, эти уровни остаются невидимым для клиента

4- **Кэш** - сохранение результата ответа для сокращения времени ответа

REST - Преимущества

- Масштабирование
- Гибкость (можно развивать части независимо, в тч менять ЯП, делить на множество уровней)
- Независимость от технологий

HTTP виды ответов

- Информационные 100 - 199
- Успешные 200 - 299
- Перенаправления 300 - 399
- Клиентские ошибки 400 - 499
- Серверные ошибки 500 - 599

- 200 - OK
- 201 - Created
- 304 - Not Modified
- 400 - Bad Request
- 401 - Unauthorized
- 403 - Forbidden
- 404 - Not Found
- 500 - Internal Server Error

Реализация в Django

```
urlpatterns = [  
    path('/', views.books, name="books"),  
    path('/<int:book_id>/', views.get_book),  
]
```

```
def get_book(request, book_id: int):  
    book = next((book for book in BOOKS if book['id'] == book_id), None)  
    if book:  
        if request.method == 'GET':  
            return JsonResponse(book)  
        elif request.method == 'DELETE':  
            BOOKS.pop(book_id)  
            return HttpResponse(status=200)  
    else:  
        return HttpResponse(status=404)  
  
def books(request):  
    if request.method == 'GET':  
        return JsonResponse({'books': BOOKS})  
    elif request.method == 'POST':  
        book = dict(request.POST)  
        for key in book:  
            book[key] = book[key][0]  
        BOOKS.append(book)  
        return HttpResponse(status=201)
```

2

Django Templates

HTML разметка

HTML - язык гипертекстовой разметки, состоящий из тегов, позволяющий отображать различные конструкции, типа изображений, ссылок, форм и других элементов

<html></html> - корень

<head></head> - тут заголовок, описание, подключение стилей и тд

<body></body> - тело документа, все что написано тут отображается на странице в браузере

<h1></h1> - хэдеры

<p></p> - текст

<div></div> - блок

**** - ссылка

Templates - шаблонизатор

Инструмент для html-шаблонизации, позволяет выполнять python логику и подставлять python переменные прямо в html страницы

Шаблоны нужны не для логики, а для отображения!

```
{% block title %}{{ section.title }}{% endblock %}

{% block content %}
<h1>{{ section.title }}</h1>

{% for story in story_list %}
<h2>
  <a href="{{ story.get_absolute_url }}">
    {{ story.headline|upper }}
  </a>
</h2>
<p>{{ story.tease|truncatewords:"100" }}</p>
{% endfor %}
{% endblock %}
```

3

Домашнее задание

Домашнее задание

- Настроить отображение todos таким образом чтобы каждый todo представлял из себя отдельный html блок (div)
- Шаблоны должны быть разделены на base, todos и todo
- Шаблоны должны быть стилизованы при помощи css и классов
- При нажатии на кнопку complete на экране должно появляться сообщение о том что задача завершена

The background is a vibrant yellow color, covered with a dense, repeating pattern of various geometric shapes. These shapes include circles, squares, triangles, and lines, some of which are filled with a fine grid pattern. The shapes are scattered across the entire surface, creating a dynamic and modern aesthetic.

Спасибо за внимание!