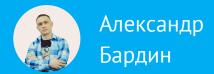


Обработка запросов





Александр Бардин

Python-разработчик в Open Solutions



План занятия

- 1. <u>Работа с конфигом Django</u>
- 2. Простой и динамические запросы
- 3. Динамические урлы
- 4. <u>Пагинация</u>

Вспоминаем прошлые занятия

- Что такое клиент и сервер?
- Что такое проект и приложение в Django?
- Опишите структуру проекта в Django. Почему нужно придерживаться определенной структуры?
- Как можно дебажить Django-проект?

Работа с конфигом Django

КАК РАБОТАТЬ С НАСТРОЙКАМИ В DJANGO

Для получения значений из конфигурации, необходимо обращаться к полям в объекте settings .

```
from django.conf import settings
from django.http import HttpResponse

def hello_view(request):
    msg = f'Свяжитесь с админом {settings.CONTANCT_EMAIL}'
    return HttpResponse('Всем привет! Я Django! ' + msg)
```

КАК РАБОТАТЬ С НАСТРОЙКАМИ В DJANGO

Импортировать значения настроек напрямую из модуля — **антипаттерн в Django.**

Как думаете, почему?

Простой и динамические запросы

Демонстрация: вспомним, как выглядит статический view

ДОБАВЛЯЕМ ДИНАМИКИ

Статичных запросов зачастую не хватает.

Варианты добавления динамики в запросы:

- 1. Параметрами запроса;
- 2. Через часть урла;
- 3. НТТР-заголовками.

ПАРАМЕТРЫ ЗАПРОСА

Передаются после символа? в конце запроса:

https://example.org?name=ivan

В Django их можно получить из объекта request.

Hапример: request.GET.get('name').

<u>Документация.</u>

ОБРАБОТКА ПАРАМЕТРОВ ЗАПРОСА

```
def home_view(request):
name = request.GET.get('name')
if name:
    response = f'Здравствуйте, {name}!'
else:
    response = f'Пожалуйста, представьтесь'
return HttpResponse(response)
```

Динамические урлы

ДИНАМИЧЕСКИЙ URL

Часть URL также может быть динамической. Django умеет парсить URL в соответствии с заданным шаблоном и передавать полученные параметры во view.

https://docs.djangoproject.com/en/3.1/topics/http/urls/#example

ДИНАМИЧЕСКИЙ URL. ПРИМЕР

Необходимо передать дату события через URL в формате since/201809-01/ и сообщить, сколько дней прошло с этой даты до сегодняшнего дня.

Если передана дата в будущем, сообщить об этом.

ВОПРОС

Какое поведение ожидается, если фактический параметр не соответствует шаблону?

Hапример, /since/2018-09-hello

ОДНОТИПНЫЕ СТРАНИЦЫ

Если на сайте представлено много однотипных страниц (например, карточка товара или профили пользователей), то эти однотипные страницы могут обрабатываться единственным view.

В таком случае, view будет извлекать динамические параметры из запроса и возвращаться пользователю ответ.

Пагинация

ПАГИНАЦИЯ

Пагинация — это способ выводить контент постранично.

Например, записи в блоге: на каждой странице содержится определенное число записей, навигация может осуществляться как на конкретную страницу, так и просто на следующую/предыдущую.

ЧТО НУЖНО УЧИТЫВАТЬ ПРИ ПАГИНАЦИИ

- количество страниц (если есть навигация на произвольную страницу);
- есть ли следующая/предыдущая страницы;
- переход на несуществующую страницу.

ДЕМОНСТРАЦИЯ: НАПИШЕМ СОБСТВЕННУЮ ПАГИНАЦИЮ

Примечание: мы будем использовать шаблон для рендеринга HTML, подробнее о шаблонах будет на следующей лекции, сейчас мы будем использовать самые базовые возможности шаблонов.

ДЕМОНСТРАЦИЯ: шаблон для рендеринга

Можете скопировать шаблон и использовать его, если захотите повторить результат с лекций. Пока эта тема не рассматривалась.

```
<!DOCTYPE html>
<head>
   <title>Cтатьи</title>
</head>
<body>
   {% for article in articles %}
        <div>{{ article }}</div>
   {% endfor %}
   <g>>
        Текущая страница: {{ current page }}
   {% if next page %}
        <br>
        \langle a \text{ href="?page={{ next page }}"} Cледующая страница\langle a \rangle
   {% endif %}
   {% if prev page %}
        <br>
        \langle a \text{ href="?page={{ prev page }}}" \rangle \Pi peдыдущая страница <math>\langle a \rangle
   {% endif %}
</body>
</html>
```

ДЕМОНСТРАЦИЯ: напишем собственную пагинацию

Получится что-то похожее:

```
def articles view(request):
     current page = request.GET.get('page', 1)
     current page = int(current page)
     items per page = 2
     total pages = math.ceil(len(all articles) / items per page)
     if current page < 1 or current page > total pages:
         current page = 1
     articles = all articles \( \)(current page - 1) \( \) items per page: \( \)
         current page * items per page]
     prev page, next page = None, None
     if current page > 1:
         prev page = urlencode({'page': current page - 1})
     if current page * items per page < len(all articles):</pre>
         next page = urlencode({'page': current page + 1})
     context = {
         'articles': articles,
         'prev page': prev page,
         'next page': next page,
         'current page': current page
     return render(request, 'demo/articles.html',
         context=context)
```

ДЕМОНСТРАЦИЯ: напишем собственную пагинацию

Получилось довольно громоздко. Поэтому рекомендуем использовать встроенный в Django класс для пагинации.

СТАНДАРТНЫЙ ПАГИНАТОР DJANGO

Для пагинации в Django используйте класс Paginator:

https://docs.djangoproject.com/en/3.1/topics/pagination/

Демонстрация:

предыдущий пример с пагинацией с использованием Paginator

РЕЗУЛЬТАТЫ

```
def articles_view(request):
   paginator = Paginator(all_articles, 2)
   current_page = request.GET.get('page', 1)
   articles = paginator.get_page(current_page)
   prev_page, next_page = None, None
    if articles.has previous():
       prev page = articles.previous page number
   if articles.has next():
       next page = articles.next page number
   context = {
        'articles': articles,
        'prev page': prev page,
        'next_page': next_page,
        'current page': articles.number
   }
   return render(request, 'demo/articles.html',
       context=context)
```

СРАВНЕНИЕ

Какой вариант лучше и почему?

СРАВНЕНИЕ

В случае с использованием стандартных инструментов, можно сосредоточиться на написании логики приложения, в то время как обработка различных ошибок и краевых ситуаций будет уже сделана за вас.

Код становится чище и понятнее. В этом преимущество библиотек.

ЧЕМУ МЫ НАУЧИЛИСЬ?

- 1. Обращаться к объекту с конфигурацией проекта.
- 2. Создавать динамические страницы с использованием параметров запроса и урлов.
- 3. Использовать пагинацию.

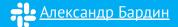
Домашнее задание

Давайте посмотрим ваше домашнее задание.

- Вопросы по домашней работе задавайте **в чате** мессенджера Slack.
- Задачи можно сдавать по частям.
- Зачёт по домашней работе проставляется после того, как приняты все задачи.



Задавайте вопросы и пишите отзыв о лекции!



Александр Бардин