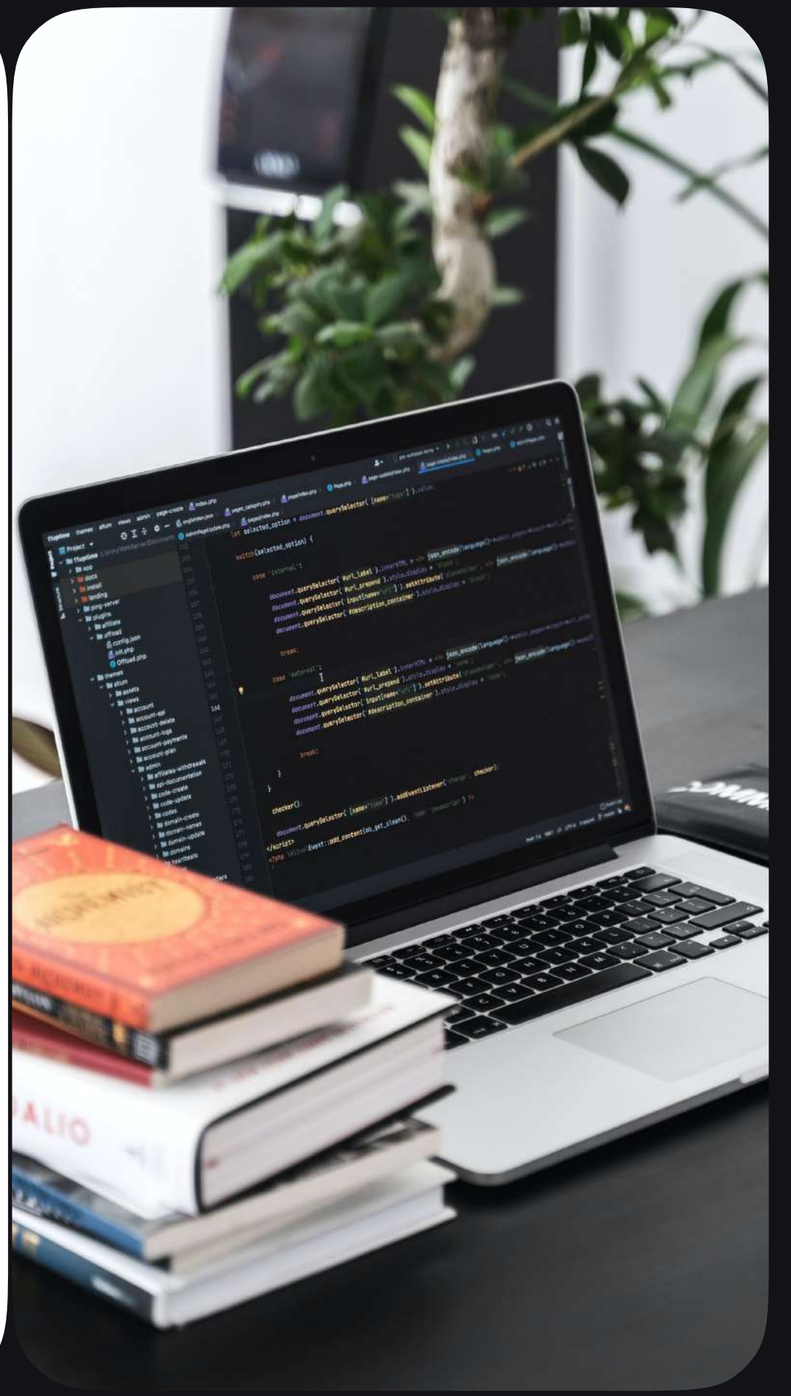
Модуль 4 Занятие 2

Маршрутизация Flask



Маршрутизация

Динамические адреса

Введение в HTML

+

Начнем создавать новостной портал

Вместо повторения

Создайте новый проект с названием flask_news, установите и активируйте виртуальное окружение, установите flask. Создайте WEB-приложение, которое:

- По адресу «/» ведет на страницу с текстом «Главная страница»
- По адресу «/news» ведет на страницу с текстом «Новости»
- По адресу «/news_detail/1» ведет на страницу с текстом «Новость 1»
- По адресу «/news_detail/2» ведет на страницу с текстом «Новость 2»

Маршрутизация

```
from flask import Flask

app = Flask(__name__)


...


@app.route('news_detail/1')
def news_detail1():
    return 'Новость 1'


@app.route('news_detail/2')
def news_detail2():
    return 'Новость 2'
```

Декоратор @app.route() (route в переводе — путь, маршрут) служит для связи конкретного URL-адреса с функцией, которая возвращает ответ на запрос. Такую функцию называют **view function** или **функция представление**.

В данном случае мы работаем со статическими URL-адресами, но также можно использовать и динамические URL, т.е. связать функцию сразу с несколькими адресами.

Динамические адреса

```
from flask import Flask

app = Flask(__name__)



...
```

```
@app.route('/news_detail/<int:id>')
def news_detail(id):
    return f'Новость {id}'
```

При описании адреса можно использовать динамическую, изменяемую часть, как переменную, взяв ее в треугольные скобки, например, <variable_name>. Тогда функция представление получает значение переменной variable_name в качестве аргумента.

По умолчанию динамические части URL передаются в функцию как строки.
Но это поведение можно изменить с помощью **конвертера**, используя следующий синтаксис <converter:variable_name>.

Конвертеры

Стандартные конвертеры, доступные в Flask:

string	(по умолчанию) принимает любой текст без «/»
int	принимает положительные целые числа
float	принимает положительные значения с плавающей точкой
path	принимает тоже, что и string, но также «/»
uuid	принимает UUID строки

Пример

```
from flask import Flask

app = Flask(__name__)



@app.route('/news_detail/<int:id>')
def news_detail(id):
    return f'Новость {id}'



@app.route('/category/<string:name>')
def category_detail(name):
    return f'Категория {name}'
```

Добавим, например, также динамические URL-адреса для отображения категорий. В отличии от новостей, имена категорий будем указывать как строковые значения.

Поведение при перенаправлении

```python
from flask import Flask

app = Flask(__name__)



@app.route('/category/')
def category():
    return 'Категории новостей'



@app.route('/news')
def news():
    return 'Новости'
```

Обратите внимание на два адреса, в чем их отличие?

В первом случае адрес заканчивается на «/». Доступ к адресу без «/» будет перенаправлен на адрес с «/» в конце.

Во втором случае адрес записывается без «/» в конце. Доступ к адресу со слэшем в конце приведет к ошибке 404 «Не найдено».

add_url_rule()

Этот пример:

```
@app.route('/')
def index():
    pass
```

Эквивалентен этому:

```
def index():
    pass
app.add_url_rule('/', 'index', index)
# или app.add_url_rule('/', view_func=index)
```

Вместо декоратора **@app.route()** для маршрутизации можно использовать метод **add_url_rule()**.

add_url_rule() принимает, кроме адреса, аргумент endpoint и имя функции представления.

endpoint — это имя маршрута, имя, по которому вы можете ссылаться на маршрут из других частей вашего приложения. Если не указывать параметр endpoint, то имя endpoint будет совпадать с именем функции представления.

url_for()

```python
from flask import Flask, url_for

app = Flask(__name__)


@app.route('/')
def index():
    return 'Главная страница'


@app.route('/news')
def news():
    return 'Новости'


@app.route('/news_detail/<int:id>')
def news_detail(id):
    return f'Новость {id}'


with app.test_request_context():
    print(url_for('index'))
    print(url_for('news'))
    print(url_for('news_detail', id=1))
```

В процессе работы с WEB-приложением, вы столкнетесь с обратной задачей, задачей получения URL-адреса для определенной функции — для этого используется функция url_for() из модуля Flask.

В примере мы используем метод test_request_context() вместе с контекстным менеджером with, чтобы Flask вел себя так, как будто он обрабатывает запрос, несмотря на то, что мы используем оболочку Python.

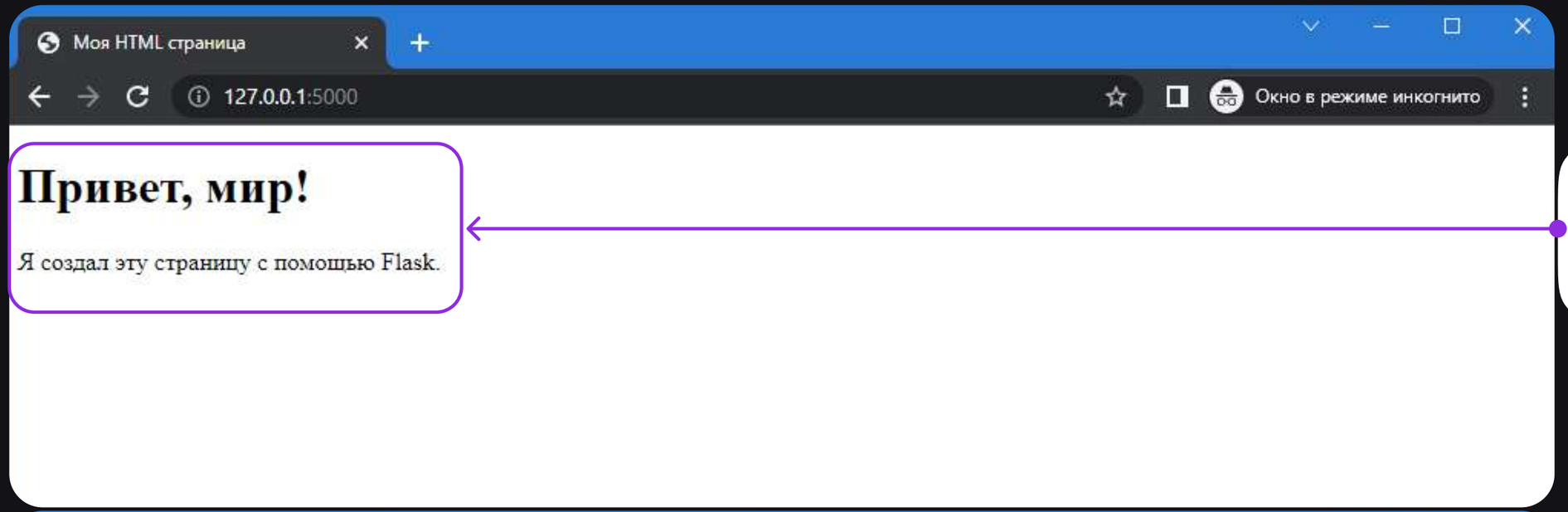
Возврат HTML

```python
from flask import Flask

app = Flask(__name__)


@app.route('/')
def index():
    return '''
    <!DOCTYPE html>
    <html>
      <head>
        <meta charset="utf-8">
        <title>Моя HTML страница</title>
      </head>
      <body>
        <h1>Привет, мир!</h1>
        <p>Я создал эту страницу с помощью Flask.</p>
      </body>
    </html>
    '''
```

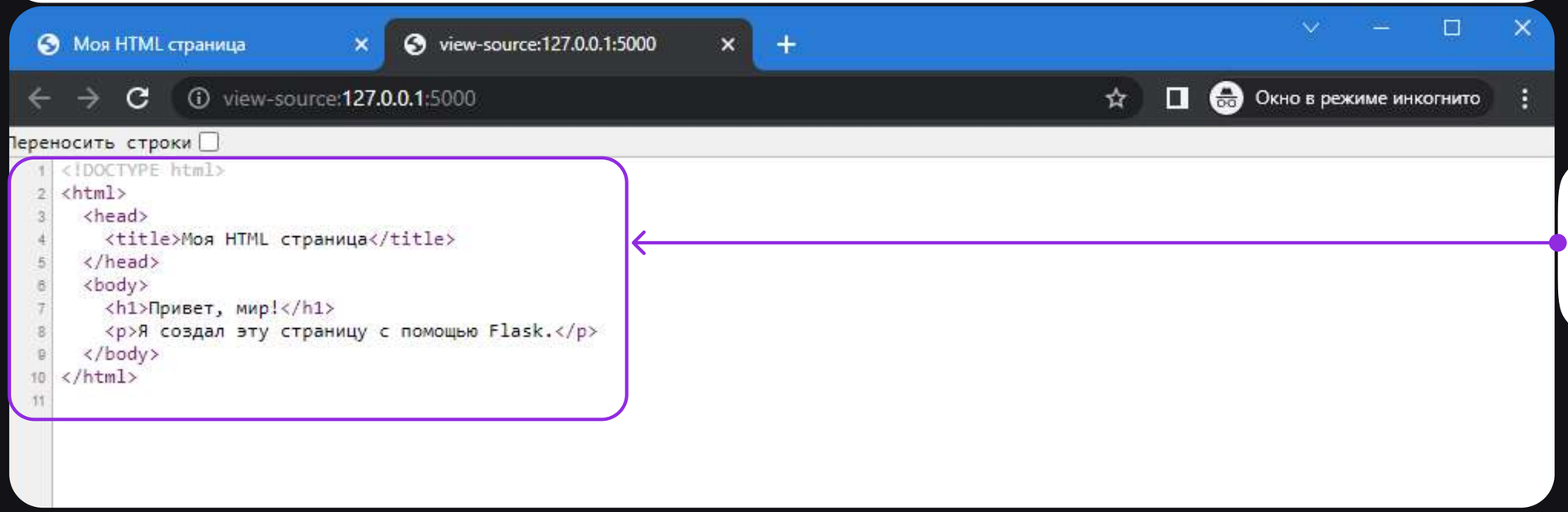
Для оформления содержимого веб-страниц используется язык разметки HTML (HyperText Markup Language).

Функция возвращает текст в формате HTML, а браузер отображает содержимое с оформлением.

HTML



Так отображается этот код в браузере



HTML код, который возвращает функция

Итоги

- Маршрутизация — это связь URL-адреса с функцией представлением, которая возвращает ответ на запрос.
- Связать URL-адрес с функцией можно с помощью декоратора @app.route() или метода add_url_rule().
- Flask позволяет при описании URL-адреса использовать изменяемую часть, как переменную, взяв ее в треугольные скобки, используя синтаксис <converter:variable_name>.
- Функция url_for() позволяет получить URL-адрес определенной функции.
- Для оформления содержимого веб-страниц используется язык разметки HTML (HyperText Markup Language).
- Функция представление возвращает текст в формате HTML, а браузер отображает содержимое с оформлением.