Маршрутизация

№ урока:	2	**Курс:**	Django Starter

Средства обучения:	Персональный компьютер с установленными: Python 3.7 Django 3.0

Обзор, цель и назначение урока

Целью данного урока является ознакомиться с основами маршрутизации запросов в Django. Вся маршрутизация в джанго строится вокруг файла urls.py, поэтому мы подробно рассмотрим его структуру, функции, которые там можно использовать для создания маршрутизации, а также кратко ознакомимся с синтаксисом регулярных выражений.

Изучив материал данного занятия, учащийся сможет:

- Строить маршрут обработки запроса в **Django.**
- Использовать для построения маршрута функции **path**, **re_path** и **include**.
- Гибко задавать шаблон для обработки запроса с помощью атрибута **route**, в частности указывать, какие параметры из **URL** передавать в обрабатывающую функцию/класс.

Содержание урока

1. Как **Django** обрабатывает запросы
2. Сопоставление маршрутов и представлений
3. Файл **urls.py**
4. Модуль **path**
5. Модуль **include**
6. Возможные значения атрибута **route**
7. Использование регулярных выражений

Резюме

- **URL (Uniform Resource Locator)** – унифицированный указатель ресурса. Используется для записи ссылок на объекты в Интернете.
- **URL** состоит из схемы, хоста, порта, **URL-**пути, параметров и якоря.
- Обычно запрос в **Django** поступает в **urls.py**, откуда он направляется в соответствующий **view**, где используются модели и шаблоны для формирования ответа клиенту.
- Для обработки запроса Django по умолчанию использует файл **urls.py**.
- В **urls.py** находится **urlpatterns**, хранящий в себе список сопоставлений, который последовательно перебирается.
- Непосредственно сравнением шаблона и передачей объекта запроса, занимаются функции **path**, **re_path** и **include**.
- **path** имеет 2 обязательных атрибута **route** и **view**, и два необязательных **kwargs** и **name**. **route** задает шаблон, с которым сравнивается **URL**, **view** задает функцию или метод, обрабатывающую запрос с данного **url**. **kwargs** - дополнительные аргументы для передачи в функцию/метод, **name** - задать имя для данного **URL**.
- **Include** используется в **path**. Он используется для подключения в обработку других модулей, а точнее файлов **urls.py** в них.
- **include** имеет обязательные атрибут module, и необязательный **namespace**. **module** указывает модуль, в который нужно передать остаток **URL-**пути. Модуль может быть

CyberBionic Systematics ® 2020
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

указан как строкой, так и **python-**объектом. **namespace** – задает название пространства имен для модуля, в который передается остаток **URL-**пути.

- **route** атрибут в **path** задает строку с шаблоном, в **re_path** задает регулярное выражение, используемое для проверки соответствия **URL**.
- **route** в **path**, позволяет передавать в функцию именованные аргументы, а также задавать тип данных аргументов и осуществлять соответствующее преобразование из **str** типа.
- Основные правила синтаксиса регулярных выражений в python: простые символы совпадают сами с собой, метасимволы имеют особой значение. Рассмотрим основные метасимволы:
    - **[]** – используются для определения набора символов, с которым будет проводится сравнение, например **"c[abt]t"** совпадет с **"cat"**, **"cbt"**, **"ctt"**. Внутри [ ] большинство метасимволов не работают, т.е. воспринимаются как обычные символы: [a$] будет совпадать с a или $.
    - ^ внутри квадратных скобочек [^] используется для определения набора символов, с которым не должно быть совпадения. Работает только если стоит сразу после открывающей скобочки, иначе трактуется как простой символ.
    - \ - символ обратного слеша. Используется для экранирования любых метасимволов. Также может задавать наборы символов, например \d – любая цифра, \w – любая цифра или буква в английском языке, \s – любой пробельный символ.
    - Метасимволы для повторения:
        - ( ) – объединить символы в группу
        - {n,m} - повторить от n до m раз.
        - ? - повторить 0 или 1 раз.
        - + - повторить 1 или более раз.
        - * - повторить 0 или более раз.
        - **?** - Выключение «жадности»
    - Для проверки составленных регулярных выражений удобно использовать различные онлайн-сайты, например https://regex101.com/

Закрепление материала

- Что такое **URL**?
- Из чего состоит **URL**?
- Что происходит с запросом, поступившим в **Django**? Опишите его путь согласно модели MVT (аналог MVC).
- Какой файл Django использует по умолчанию для первоначальной обработки запроса?
- Где находится **urlpatterns**, хранящий в себе список сопоставлений, который последовательно перебирается.
- Какие функции занимаются непосредственно сравнением шаблона и передачей объекта запроса? За что каждая отвечает?
- Какие атрибуты есть у функции **path**? Что каждый из них задает?
- В чем отличие **re_path** от **path**?
- Какие атрибуты есть у функции **include**? Что каждый из них задает?
- Что задает атрибут **route** в **path** и в **re**_path?
- Как передать аргументы из URL в функцию, обрабатывающую запрос? Что для этого используется?
- Опишите основные правила синтаксиса регулярных выражений в python
- Какие метасимволы определяют набор символов, с которым будет проводится сравнение?
- Какой метасимвол используются для определения набора символов, с которым не должно быть совпадения?

CyberBionic Systematics ® 2020
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

- Как экранировать любые метасимволы?
- Какие вы знаете основные наборы символов в регулярных выражениях?
- Какие существуют метасимволы для повторения?

Дополнительное задание

- Составить urls.py файл, который:
    - направляет URL с 'index/' в метод views.index и задает имя для этого URL как ‘index-view'.
    - направляет URL с 'bio/{имя пользователя}/' в метод views.bio вместе с именем пользователя в качестве аргумента username, и задает имя для этого URL как ‘bio'.
    - передает запросы, начинающиеся с ‘lesson_1/' в модуль ‘lesson_1.urls', вместе с остатком URL.
- Написать пример регулярного выражения, которое совпадает сразу с тремя строками: "cat", "cbt", "ctt".
- Написать пример регулярного выражения, которое не совпадает сразу с тремя строками: "cat", "cbt", "ctt".
- Написать пример регулярного выражения, которое совпадает с "c", "cat", "catat" или "catatatatat".
- Написать пример регулярного выражения, которое будет совпадать “c", "cat", но не будет совпадать целиком с "catat" или "catatatatat", а будет совпадать только с “cat” в них.

Самостоятельная деятельность учащегося

Задание 1

Составить urls.py файл, который:

- направляет URL с home/' в метод views.home и задает имя для этого URL как ‘home-view'.
- направляет URL с 'book/{название главы}/' в метод views. Book вместе с названием главы в качестве аргумента title, и задает имя для этого URL как ‘book'.
- передает запросы, начинающиеся с ‘lesson_2/' в модуль ‘lesson_2.urls', вместе с остатком URL.

Задание 2

- Написать пример регулярного выражения, которое совпадает сразу с тремя строками: "dog", "box", "bog".
- Написать пример регулярного выражения, которое не совпадает сразу с тремя строками: "dog", "box", "bog", но совпадет с “cot”

Задание 3

Написать пример регулярного выражения, которое совпадает с "d", "dog", "dogog", но не совпадает с "dogogog".

Рекомендуемые ресурсы

Официальный сайт asp.net
http://www.asp.net/

https://docs.djangoproject.com/en/2.2/intro/tutorial01/ - официальное руководство по Django для новичков.

https://regex101.com/ - сайт для проверки регулярных выражений

CyberBionic Systematics ® 2020
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com