**Тема занятия № 8: Маршрутизация**

1. Как работает маршрутизатор

**Маршрутизация** — это определение контроллера, который следует выполнить при получении в составе клиентского запроса интернет-адреса заданного формата. Подсистема фреймворка, выполняющая маршрутизацию, носит название маршрутизатора.

**Как работает маршрутизатор?**

Маршрутизатор Django работает, основываясь на написанном разработчиком списке маршрутов. Каждый элемент такого списка — маршрут — устанавливает связь между путем определенного формата (шаблонным путем) и контроллером. Вместо контроллера в маршруте может быть указан вложенный список маршрутов.

Алгоритм работы маршрутизатора, следующий:

1. Из полученного запроса извлекается запрашиваемый клиентом интернет-адрес.

2. Из интернет-адреса удаляются обозначение протокола, доменное имя (или IP-адрес) хоста, номер TCP-порта, набор GET-параметров и имя якоря, если они там присутствуют. В результате остается один только путь.

3. Этот путь последовательно сравнивается с шаблонными путями, записанными во всех маршрутах, имеющихся в списке.

4. Как только шаблонный путь из очередного проверяемого маршрута совпадёт с началом полученного пути:

• если в маршруте указан контроллер, он выполняется;

• если в маршруте указан вложенный список маршрутов, то из полученного пути удаляется префикс, совпадающий с шаблонным путем, и начинается просмотр маршрутов из вложенного списка.

5. Если ни один из шаблонных путей не совпал с полученным в запросе, то клиенту отправляется стандартная страница сообщения об ошибке 404 (запрошенная страница не найдена).

Обратим особое внимание на то, что маршрутизатор считает полученный путь совпавшим с шаблонным, если последний присутствует в начале первого. Это может привести к неприятной коллизии. Например, если мы имеем список маршрутов с шаблонными путями create/ и create/comment/ (расположенными именно в таком порядке), то при получении запроса с путем /create/comment/ маршрутизатор посчитает, что произошло совпадение с шаблонным путем create/, т. К. Он присутствует в начале запрашиваемого пути. Поэтому в рассматриваемом случае нужно расположить маршруты в порядке create/comment/ и create/, тогда описанная здесь коллизия не возникнет.

В составе запрашиваемого пути может быть передано какое-либо значение. Чтобы получить его, достаточно поместить в нужное место шаблонного пути в соответствующем маршруте обозначение URL-параметра. Маршрутизатор извлечет значение и передаст его либо в контроллер, либо вложенному списку маршрутов (и тогда полученное значение станет доступно всем контроллерам, объявленным во вложенном списке).

По возможности маршруты должны содержать уникальные шаблонные пути. Создавать маршруты с одинаковыми шаблонными путями допускается, но не имеет смысла, т.к. В любом случае будет срабатывать самый первый маршрут из совпадающих, а последующие окажутся бесполезны.

**Списки маршрутов уровня проекта и уровня приложения**

Поскольку маршрутизатор Django поддерживает объявление вложенных списков маршрутов, все маршруты проекта описываются в виде своего рода иерархии:

□ в списке маршрутов, входящем в состав пакета конфигурации, записываются маршруты, ведущие на отдельные приложения проекта.

Каждый из этих маршрутов указывает на вложенный список, принадлежащий соответствующему приложению.

По умолчанию список маршрутов уровня проекта записывается в модуле uris.py пакета конфигурации. Можно сохранить его и в другом модуле, указав его путь, отсчитанный от папки проекта, в параметре root urlconf модуля settings.ру пакета конфигурации;

□ а в списках маршрутов, принадлежащих отдельным приложениям, записываются маршруты, указывающие непосредственно на контроллеры, которые входят в состав этих приложений.

Под хранение маршрута уровня приложения, как правило, в пакете приложения создается отдельный модуль, обычно с именем uris.py. Его придется создать вручную, поскольку команда startapp утилиты manage.py этого не делает.

1. Объявление маршрутов

**Объявление маршрутов**

Любые списки маршрутов — неважно, уровня проекта или приложения — объявляются согласно схожим правилам.

Список маршрутов оформляется как обычный список Python и присваивается переменной с именем uripatterns — именно там маршрутизатор Django будет искать его.

Каждый элемент списка маршрутов должен представлять собой результат, возвращаемый функцией path о из модуля django.urls. Формат вызова этой функции таков:

Path (<шаблонный путь>, <контроллер> | <вложенный список маршрутов> [, дополнительные параметры>] [, пагае=<имя маршрута>])

Шаблонный путь записывается в виде строки. В конце он должен содержать прямой слеш, а в начале таковой, напротив, не ставится. О дополнительных параметрах и имени маршрута МЫ поговорим позже.

В качестве шаблонного пути можно указать пустую строку, создав корневой маршрут. Он будет связан с ’’корнем” приложения (если задан в списке маршрутов уровня этого приложения) или всего сайта (будучи заданным в списке уровня проекта).

Для объявления в шаблонном пути URL-параметров (параметризованный маршрут) применяется следующий формат:

< [Обозначение формата>:] <имя URL-параметра> > поддерживаются следующие обозначения форматов для значений URL-параметров:

□ str —любая непустая строка, не включающая слеши (формат по умолчанию);

□ int —положительное целое число, включая 0;

□ slug— строковый слаг, содержащий латинские буквы, цифры, знаки дефиса и подчеркивания;

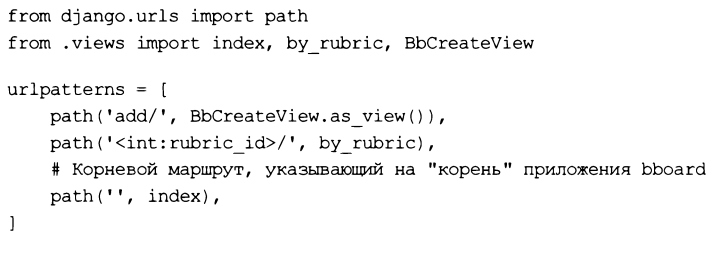
□ uuid—уникальный универсальный идентификатор. Он должен быть правильно отформатирован: должны присутствовать дефисы, а буквы следует указать в нижнем регистре;

□ path —фрагмент пути — любая непустая строка, включающая слеши.

Имя URL-параметра задает имя для параметра контроллера, через который последний сможет получить значение, переданное в составе интернет-адреса. Это имя должно удовлетворять правилам именования переменных Python.

Вернемся к функции path (). Контроллер указывается в виде ссылки на функцию (если это контроллер-функция, речь о которых пойдет далее) или результата, возвращенного методом as viewo контроллера-класса (контроллерам-классам посвящена тема позже).

Пример написания маршрутов, указывающих на контроллеры, можно ранее. Первый маршрут указывает на контроллер-класс bbcreateview, а второй и третий--- на контроллеры-функции by rubric () И index ().

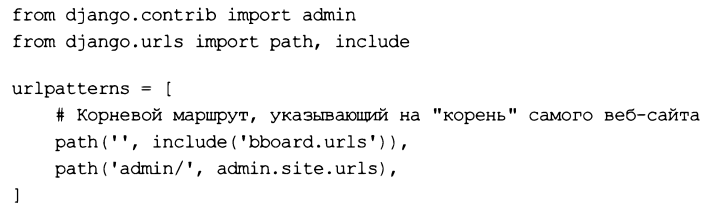
****

Вложенный список маршрутов указывается в виде результата, возвращенного функцией include () из того же модуля django.urls. Вот ее формат вызова:

Include (<путь к модулю>\ <вложенный список маршрутов>[,Namespaсе=<имя пространства имен>])

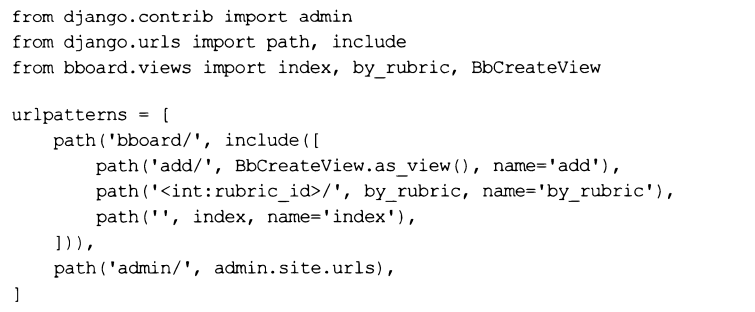
В качестве первого параметра обычно указывается строка с путем к модулю, содержащему вложенный список маршрутов. О пространствах имен и их именах мы поговорим позже.

Пример указания маршрутов, ведущих на вложенные списки маршрутов, показан далее. В первом маршруте вложенный список задан в виде пути к модулю, а во втором— в виде готового списка маршрутов (он хранится в свойстве uris экземпляра класса Adminsite, содержащегося в переменной site модуля django.Contrib. Admin).

****

В первом параметре функции include () можно указать непосредственно вложенный список маршрутов. Это может пригодиться при программировании простого сайта — отпадет нужда создавать отдельные модули под списки маршрутов уровня приложений.

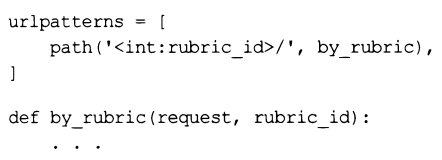
Пример списка маршрутов уровня проекта, непосредственно включающего вложенный список уровня приложения, приведен. В таком случае нужда в отдельном модуле uris.py, хранящем список маршрутов уровня приложения, отпадает.

****

1. Передача данных в контроллеры

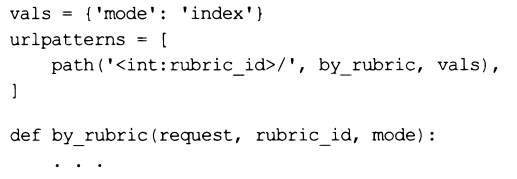
Значения URL-параметров контроллер-функция получает через параметры, имена которых совпадают с именами URL-параметров.

Так, в приведенном далее примере контроллер-функция by rubrico получает значение URL-параметра rubric id через параметр rubric id:

****

Есть еще один способ передать какие-либо данные в контроллер. Для этого нужно объявить словарь Python, создать в нем столько элементов, сколько нужно передать значений, присвоить передаваемые значения этим элементам и передать полученный словарь функции path () в третьем параметре. Эти значения контроллер сможет получить также через одноименные параметры.

Вот пример передачи контроллеру-функции значения mode:

****