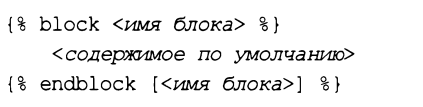
**Тема занятия № 24: Модуль 11. Шаблоны и статические файлы**

1. Наследование шаблонов

Аналогично наследованию классов в Python, Django предлагает механизм наследования шаблонов. Базовый шаблон содержит элементы, присутствующие на всех страницах сайта: шапку, поддон, главную панель навигации, элементы разметки и др. А производный шаблон выводит лишь уникальное содержимое генерируемой им страницы: список объявлений, объявление, выбранное посетителем, и др.

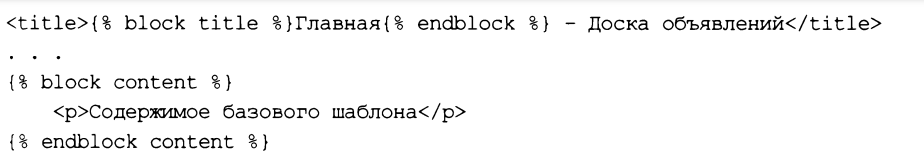
Базовый шаблон содержит блоки, помечающие места, куда будут выведены фрагменты уникального содержимого, сгенерированного производным шаблоном. Блоки в базовом шаблоне объявляются с применением парного тега block . . .Endblock:

****

Имя блока должно быть уникальным в пределах базового шаблона. Его также можно указать в теге endblock, чтобы в дальнейшем не гадать, какому тегу block он соответствует.

Содержимое по умолчанию будет выведено, если производный шаблон его не сгенерирует.

Пример:

****

В коде производного шаблона необходимо явно указать, что он является производным от определенного базового шаблона, вставив в начало его кода тег {% extends<путь к базовому шаблону> %}.

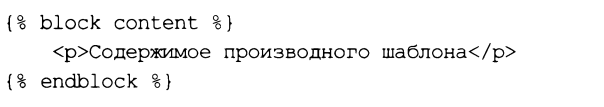
Внимание!

Тег extends должен находиться в самом начале кода шаблона, на отдельной строке.

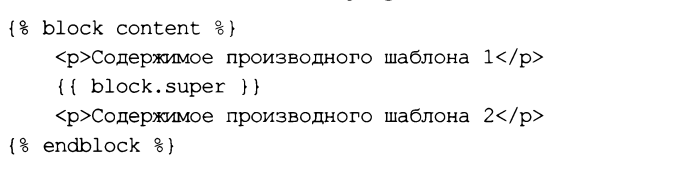
Пример (подразумевается, что базовый шаблон хранится в файле layout\basic.html):



После этого в производном шаблоне точно так же объявляются блоки, но теперь уже в них записывается создаваемое шаблоном содержимое:

****

Содержимое по умолчанию, заданное в базовом шаблоне, в соответствующем блоке производного шаблона доступно через переменную block.super, создаваемую в контексте шаблона самим Django:

****

В результате на экран будут выведены три абзаца:

* Содержимое производного шаблона 1
* Содержимое базового шаблона
* Содержимое производного шаблона 2

В производном шаблоне можно создать другие блоки и таким образом сделать его базовым для других производных шаблонов.

1. Обработка статических файлов

В терминологии Django статическими называются файлы, отправляемые клиенту как есть: таблицы стилей, графические изображения, аудио- и видеоролики, файлы статических веб-страниц и т. п.

Обработку статических файлов выполняет подсистема, реализованная во встроенном приложении django.contrib.staticfiles. Оно включается в список зарегистрированных приложений уже при создании нового проекта, и, если сайт содержит статические файлы, удалять его оттуда нельзя.

Настройка подсистемы статических файлов подсистемой статических файлов управляет ряд настроек, записываемых в модуле Settings.py пакета конфигурации:

□ static url— префикс, добавляемый к интернет-пути статического файла.

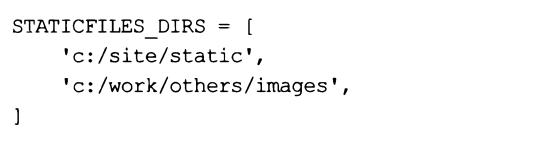
Встретив в начале полученного в запросе пути этот префикс, Django “ поймет”, что запрашивается статический файл. Значение по умолчанию — None, но при создании нового проекта оно устанавливается в ’’/static/";

□ static root — файловый путь к основной папке, в которой хранятся все статические файлы (значение по умолчанию — None).

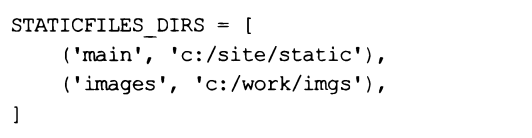
В этой же папке будут собираться все статические файлы, если отдать команду Collectstatic утилиты manage.py;

□ staticfiles dirs — список файловых путей к дополнительным папкам, в которых хранятся статические файлы. Каждый путь может быть задан в двух форматах:

• как строка с файловым путем. Пример:

****

• как кортеж из двух элементов: префикса интернет-пути и файлового пути к папке. Чтобы сослаться на файл, хранящийся в определенной папке, нужно предварить интернет-путь этого файла заданным для папки префиксом. Пример:

****

Теперь, чтобы вывести на страницу файл logo.png, хранящийся в папке с:\work\imgs\others\, следует записать в шаблоне тег:



□ staticfiles finders — список имен классов, реализующих подсистемы поиска статических файлов. По умолчанию включает два класса, объявленные в модуле Django.contrib.staticfiles.finders:

• filesystemfinder— ищет статические файлы в папках, заданных параметрами STATIC\_ROOT И STATICFILES\_DIRS;

• appdirectoriesfinder —ищет статические файлы в папках static, находящихся в пакетах приложений.

Если статические файлы хранятся в каком-то определенном местоположении (только в папках, заданных параметрами static root и staticfiles dirs, или только в папках static в пакетах приложений), можно указать в параметре Staticfiles finders только один класс— соответствующий случаю. Это несколько уменьшит потребление системных ресурсов;

□ staticfiles storage — имя класса, реализующего хранилище статических файлов. По умолчанию используется Хранилище staticfilesstorage из модуля Django.contrib.staticfiles.storage.

Обслуживание статических файлов встроенный в Django отладочный веб-сервер обслуживает статические файлы самостоятельно. Но если сайт находится в эксплуатационном режиме, придется позаботиться об обслуживании статических файлов веб-сервером самостоятельно.

Формирование интернет-адресов статических файлов формировать адреса статических файлов в коде шаблонов можно посредством трех разных программных механизмов:

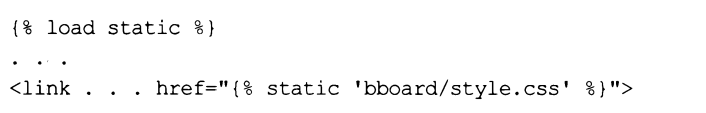
□ static — тег шаблонизатора, вставляющий в шаблон полностью сформированный интернет-адрес статического файла. Формат записи:



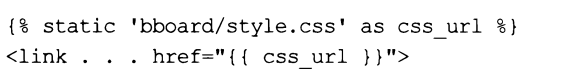
Относительный путь к статическому файлу записывается в виде строки и отсчитывается от папки, путь которой записан в параметрах static root и Staticfiles dirs, или папки static пакета приложения.

Тег реализован в библиотеке тегов с псевдонимом static, которую следует предварительно загрузить тегом load.

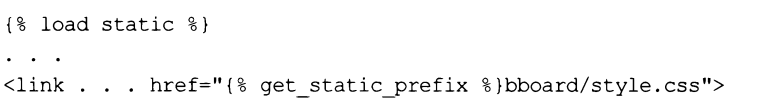
Пример:



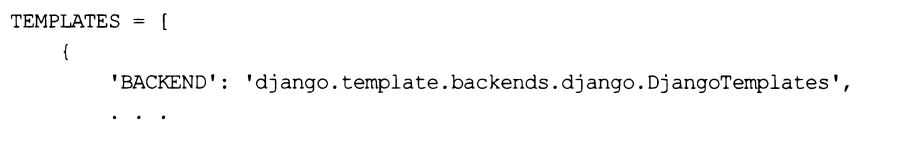
По умолчанию сформированный адрес непосредственно вставляется в код страницы. Также можно сохранить адрес в переменной, записав ее после ключевого слова as. Пример:

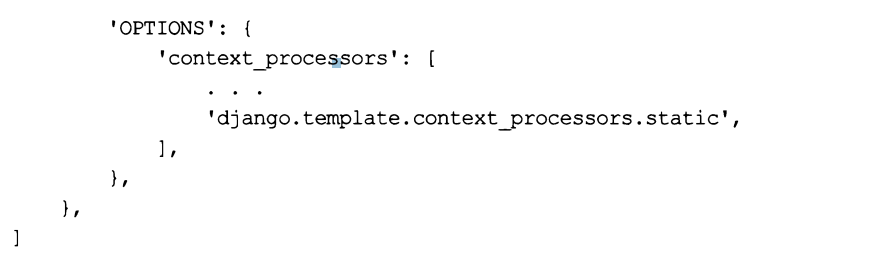
****

□ {% get static prefix %} — тег шаблонизатора, который вставляет в код страницы префикс из параметра static url. Также реализован в библиотеке тегов Static. Пример:



□ django.template.context\_processors.static — обработчик контекста, добавляющий в контекст шаблона переменную static url, которая хранит префикс из одноименного параметра проекта. Поскольку этот обработчик по умолчанию не включен в список активных (элемент context processors параметра options), его нужно добавить туда:





После этого можно обращаться к переменной static url, созданной данным обработчиком в контексте шаблона:

