**Тема занятия № 30: Обработка выгруженных файлов**

1. Низкоуровневые средства для сохранения выгруженных файлов

Низкоуровневые средства предоставляют опытному разработчику полный контроль над сохранением и выводом выгруженных файлов.

Класс uploadedfile: выгруженный файл.

**Сохранение выгруженных файлов**

Ранее говорилось, что выгруженные файлы хранятся в словаре, доступном через атрибут files объекта запроса. Каждый такой файл представляется экземпляром класса uploadedfile.

Атрибуты этого класса:

□ name — изначальное имя выгруженного файла;

□ size — размер выгруженного файла в байтах;

О content type — MIME-тип файла в виде строки;

□ content type extra — дополнительные параметры MIME-типа файла, представленные в виде словаря;

□ charset — кодировка, если файл текстовый. методы класса uploadedfile:

□ multiple\_chunks ( [chunk\_size=None] ) — возвращает True, если файл настолько велик, что для обработки его придется разбивать на отдельные части, и False, если он может быть обработан как единое целое.

Необязательный параметр chunk size указывает размер отдельной части (собственно, файл считается слишком большим, если его размер превышает размер части). Если этот параметр не указан, размер принимается равным 64 Кбайт;

□ read () — считывает и возвращает в качестве результата все содержимое файла.

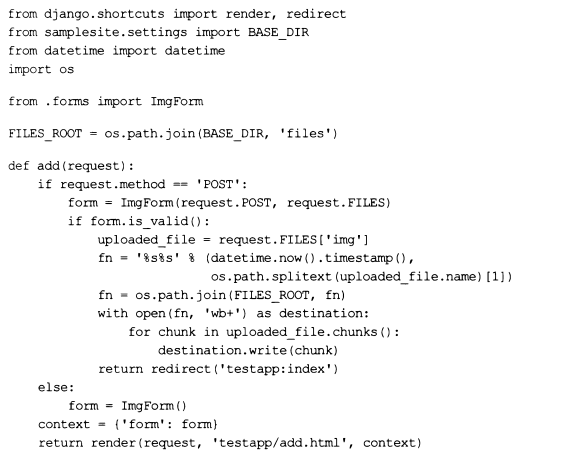
Этот метод можно использовать, если файл не слишком велик (метод Multiple\_chunks () возвращает False);

□ chunks ([chunk\_size=None]) — возвращает итератор, который на каждой итерации выдает очередную часть файла.

Необязательный параметр chunk size указывает размер отдельной части. Если он не указан, размер принимается равным 64 Кбайт.

Разработчики Django рекомендуют использовать этот метод, если файл слишком велик, чтобы быть обработанным за один раз (метод multiple chunks о возвращает True). На практике же его можно применять в любом случае — это позволит упростить код.

Приведен код контроллера, сохраняющего выгруженный файл низкоуровневыми средствами.

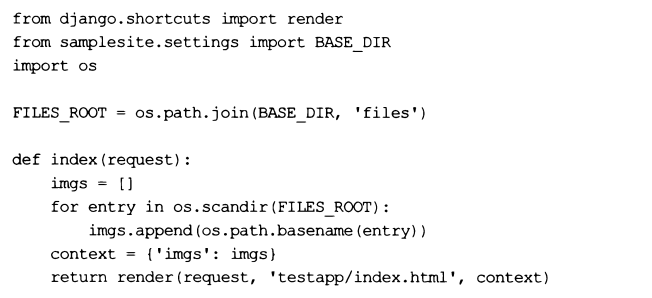


Выгруженный файл сохраняется под именем, сформированным на основе текущей временной отметки, под изначальным расширением, в папке files, находящейся в папке проекта. Как видим, применяя низкоуровневые средства, файл можно сохранить в произвольной папке.

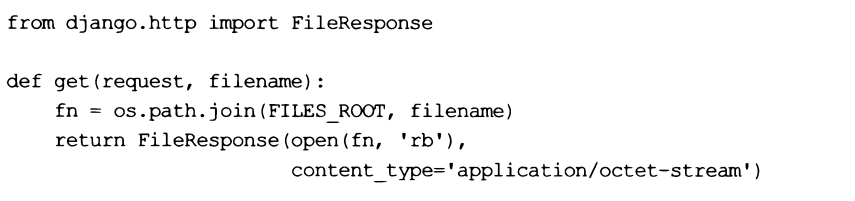
**Вывод выгруженных файлов низкоуровневыми средствами**

Вывести список выгруженных файлов можно, выполнив поиск всех файлов в нужной папке и сформировав их список. Для этого удобно применять функцию Scandir () ИЗ модуля os.

Приведен код контроллера, который выводит список выгруженных файлов, хранящихся в папке files папки проекта.



В шаблоне testapp\index.html нам нужно вывести изображения, хранящиеся в выгруженных файлах. Обратиться к атрибуту url мы не можем по вполне понятной причине. Однако мы можем написать еще один контроллер, который получит через URL-параметр имя выгруженного файла и сформирует на его основе ответ — экземпляр класса fiieresponse. Код этого контроллера приведен.



Маршрут, ведущий к этому контроллеру, может быть таким:



Обратим внимание, что здесь используется обозначение формата path, т. е. любая непустая строка, включающая в себя любые символы.

И наконец, для вывода списка файлов мы напишем в шаблоне testapp\index.html следующий код:

****

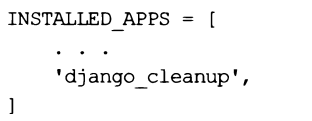
Низкоуровневые средства выгрузки файлов, поддерживаемые Django, основаны на инструментах Python, предназначенных для работы с файлами и папками (как мы только что убедились). Для хранения файлов они не требуют создания модели и имеют более высокое быстродействие. Однако им присущ ряд недостатков: невозможность сохранения дополнительной информации о выгруженном файле (например, описания или сведений о пользователе, выгрузившем файл) и трудности в реализации фильтрации и сортировки файлов по произвольным критериям.

1. Библиотека django-cleanup: автоматическое удаление ненужных файлов

Ранее говорилось, что при удалении записи модели, которая содержит поле типа filefieid или imagefieid, файл, сохраненный в этом поле, не удаляется. Аналогично, при записи в такое поле другого файла старый файл также не удаляется, а остается на диске.

Дополнительная библиотека django-cleanup отслеживает появление ненужных файлов, и сама их удаляет. Установить ее можно подачей команды:

Pip install django-cleanup ядро этой библиотеки— приложение django\_cieanup, которое следует добавить в список зарегистрированных в проекте:

****

На этом какие-либо действия с нашей стороны закончены. Далее библиотека Django-cleanup начнет работать самостоятельно.

1. Библиотека easy-thumbnails: вывод миниатюр

Очень часто при выводе списка графических изображений, хранящихся на сайте, показывают их миниатюры — уменьшенные копии. А при переходе на страницу выбранного изображения уже демонстрируют его полную редакцию.

Для автоматического формирования миниатюр согласно заданным параметрам удобно применять дополнительную библиотеку easy-thumbnails.

На заметку!

Полную документацию по библиотеке easy-thumbnails можно найти здесь:

Http://easy-thumbnails.readthedocs.io/en/latest/.

**Установка easy-thumbnails**

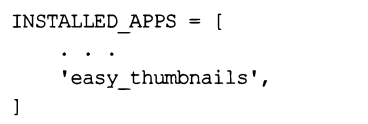
Для установки библиотеки следует набрать в командной строке команду:



Помимо easy-thumbnails, будет установлена библиотека Pillow, необходимая для работы.

Программное ядро библиотеки реализовано в виде приложения easy-thumbnails.

Это приложение необходимо добавить в список зарегистрированных в проекте:

****

Наконец, чтобы все заработало, нужно выполнить миграции.

На заметку!

Для своих нужд easy-thumbnails создает в базе данных таблицы easy\_thumbnaiis\_Source, easy\_thumbnails\_thumbnail И easy\_thumbnails\_thunibnaildimensions.

**Пресеты миниатюр**

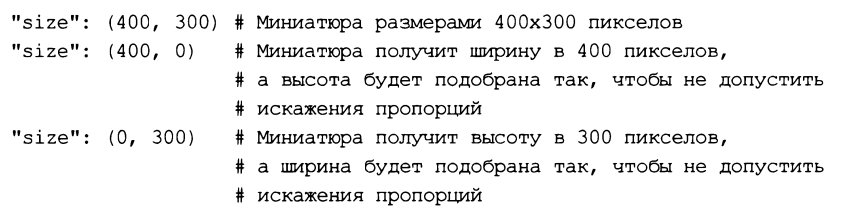
Прежде всего, необходимо указать набор пресетов (предопределенных комбинаций настроек), на основе которых будут создаваться миниатюры. Все параметры, которые можно указать в таких пресетах и которые затрагивают создаваемые библиотекой миниатюры, приведены далее:

□ size —размеры миниатюры. Значением должен быть кортеж из двух элементов:

Ширины и высоты, заданных в пикселах.

Допускается вместо одного из размеров указывать число о. Тогда библиотека сама подберет значение этого размера таким образом, чтобы пропорции изображения не искажались.

Примеры:

****

□ crop— управляет обрезкой или масштабированием изображения до размеров, указанных в параметре size. Значением может быть одна из строк:

• "scale" — изображение будет масштабироваться до указанных размеров. Обрезка проводиться не будет;

• "smart" — будут обрезаны малозначащие, с точки зрения библиотеки, края изображения ("умная" обрезка);

• ”<смещение слева>, <смещение сверху>"— ЯВНО указывает местоположение фрагмента изображения, который будет вырезан и превращен в миниатюру.

Величины смещения слева и сверху задаются в процентах от ширины и высоты изображения соответственно. Положительные значения указывают собственно смещение слева и сверху левой или верхней границы миниатюры, а отрицательные — смещения справа и снизу ее правой или нижней границы.

Если задать значение о, соответствующая граница миниатюры будет находиться на границе исходного изображения.

Значение параметра по умолчанию: "50,50";

□ autocrop —если True, то белые поля на границах изображения будут обрезаны;

□ bw —если True, то миниатюра станет черно-белой;

□ replace alpha— цвет, которым будет замещен прозрачный цвет в исходном изображении. Цвет указывается в формате krrggbb, где rr — доля красной составляющей, gg— зеленой, вв— синей. По умолчанию преобразование прозрачного цвета не выполняется;

□ quality— качество миниатюры в виде числа от 1 (наихудшее качество) до юо (наилучшее качество). Значение по умолчанию: 85;

□ subsampling — обозначение уровня подвыборки цвета в виде числа 2 (значение по умолчанию), 1 (более четкие границы, небольшое увеличение размера файла) или о (очень четкие границы, значительное увеличение размера файла).

Пресеты записываются в параметре thumbnail aliases в виде словаря. Ключи элементов этого словаря указывают области действия пресетов, записанные в одном из следующих форматов:

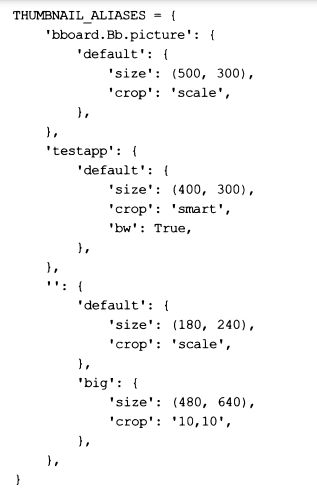
□ ’’пустая” строка — пресет действует во всех приложениях проекта;

□ ”<псевдоним приложения^' — пресет действует только в приложении с указанным псевдонимом', о "<псевдоним приложения>. <имя модели>"— пресет действует только в модели с заданным именем в приложении с указанным псевдонимом;

□ "<псевдоним приложения>. <имя модели>.<имя поля>" — пресет действует только для поля с указанным именем, в модели с заданным именем, в приложении С указанным псевдонимом.

Значениями элементов этого словаря должны быть словари, указывающие сами пресеты. Ключи элементов зададут имена пресетов, а элементы, также словари, укажут настройки, относящиеся к соответствующему пресету.

Приведен пример указания пресетов для библиотеки easy-thumbnails.

****

Для поля picture модели вь приложения bboard мы создали пресет default, в котором указали размеры миниатюры 500x300 пикселов и масштабирование без обрезки. Для приложения testapp мы также создали пресет default, где задали размеры 400x300 пикселов, ’’умную” обрезку и преобразование в черно-белый вид. А для всего проекта мы расстарались на два пресета: default (размеры 180x240 пикселов и масштабирование) и big (размеры 480x640 пикселов и обрезка, причем миниатюра будет находиться на расстоянии 10% от левой и верхней границ исходного изображения).

Параметр thumbnail default options указывает параметры по умолчанию, применяемые ко всем пресетам, в которых они не были переопределены. Значением этого параметра должен быть словарь, аналогичный тому, который задает параметры отдельного пресета. Пример:



**Остальные параметры библиотеки**

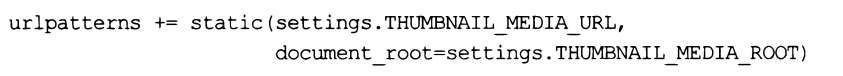
Далее перечислены остальные параметры библиотеки easy-thumbnails, которые могут пригодиться:

□ thumbnail media url — префикс, добавляемый к интернет-адресу файла со сгенерированной миниатюрой. Если указать ’’пустую” строку, то будет использоваться префикс из параметра media url. По умолчанию — ’’пустая” строка;

□ thumbnail media root — полный путь к папке, хранящей файлы с миниатюрами.

Если указать ’’пустую” строку, то будет использована папка из параметра Media root. По умолчанию — ’’пустая” строка.

В случае указания параметров thumbnail\_media\_url и thumbnail\_media\_root необходимо записать соответствующий маршрут, чтобы Django смог загрузить созданные библиотекой миниатюры. Код, создающий этот маршрут может выглядеть так:



□ thumbnail basedir— имя папки, хранящей файлы миниатюр и находящейся в папке из параметра thumbnail media root. Так, если задать значение ’’thumbs", то миниатюра изображения images\others\img1.jpg будет сохранена в файле Thumbs\images\others\img1.jpg. По умолчанию — ’’пустая” строка (т. е. файлы миниатюр будут сохраняться непосредственно в папке из параметра thumbnail\_MEDIA\_ROOT);

□ thumbnail subdir— имя вложенной папки, хранящей файлы миниатюр и создаваемой в каждой из вложенных папок, которые есть в папке из параметра Thumbnail media root. Так, если задать значение "thumbs", то миниатюра изображения images\others\img1.jpg будет сохранена в файле images\others\thumbs\irng1.jpg.

По умолчанию — ’’пустая” строка (т. Е. Вложенные папки для миниатюр создаваться не будут);

□ thumbnail prefix — префикс, добавляемый в начало имен файлов с миниатюрами (по умолчанию — ’’пустая” строка);

□ thumbnail extension — формат файлов для сохранения миниатюр без поддержки прозрачности (по умолчанию: "jpg");

□ thumbnail transparency extension — формат файлов для сохранения миниатюр с поддержкой прозрачности (по умолчанию: "png");

□ thumbnail preserve extensions — последовательность расширений файлов, для которых следует создать миниатюры в тех же форматах, в которых были сохранены оригинальные файлы. Расширения должны быть указаны без начальных точек в нижнем регистре. Пример:



Теперь для файлов с расширением png будут созданы миниатюры также в формате PNG, а не формате из параметра thumbnail extension.

Если указать значение True, то для файлов всех форматов будут создаваться миниатюры в тех же форматах, что и исходные файлы.

Значение по умолчанию — None;

□ thumbnail progressive— величина размера изображения в пикселах, при превышении которой изображение будет сохранено в прогрессивном формате JPEG. Учитывается любой размер— как ширина, так и высота. Если указать значение False, то прогрессивный JPEG вообще не будет использоваться. По умолчанию: 100;

□ thumbnail quality— качество изображения JPEG в диапазоне от 1 до 100 (по умолчанию: 85);

□ thumbnail widget options — параметры миниатюры, генерируемой для элемента управления imageciearabiefiieinput (будет описан позже). Записываются в виде словаря в том же формате, что и параметры отдельного пресета.

По умолчанию: {'size': (80, 80)} (размеры 80x80 пикселов).

**Вывод миниатюр в шаблонах**

Прежде чем выводить в шаблонах сгенерированные библиотекой easy-thumbnails миниатюры, нужно загрузить библиотеку тегов с псевдонимом thumbnail:



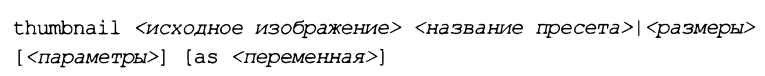
Для вывода миниатюры можно использовать:

□ thumbnail\_url: <название пресета> — фильтр, ВЫВОДИТ интернет-адрес файла с миниатюрой, созданной на основе пресета с указанным названием и исходного изображения, взятого из поля типа filefieid или imagefieid. Если пресета с указанным названием нет, будет выведена ’’пустая” строка. Пример:



□ thumbnail — тег, выводит интернет-адрес файла с миниатюрой, созданной на основе исходного изображения, ВЗЯТОГО ИЗ ПОЛЯ типа filefieid ИЛИ imagefieid.

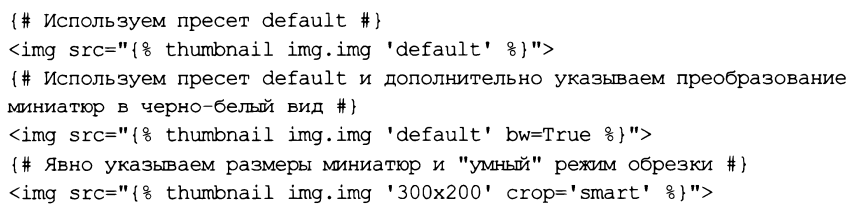
Формат тега:



Размеры могут быть указаны либо в виде строки формата "<ширина>х<высота>", либо в виде переменной, которая может содержать строку описанного ранее формата или кортеж из двух элементов, из которых первый укажет ширину, а второй — высоту.

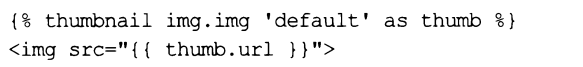
Параметры записываются в том же формате, что и в объявлении пресетов. Если вместо размеров указано название пресета, то заданные параметры переопределят значения, записанные в пресете.

Примеры:

****

Мы можем сохранить созданную тегом миниатюру в переменной, чтобы использовать ее впоследствии. Эта миниатюра представляется экземпляром класса thumbnailfile, являющегося ПРОИЗВОДНЫМ ОТ класса imagefieldfile и поддерживающего те же атрибуты.

Пример:



**Хранение миниатюр в моделях**

Библиотека easy-thumbnails поддерживает два класса полей модели, объявленных в модуле easy\_thumbnails. Fields:

□ thumbnaiierfieid— подкласс класса filefieid. Выполняет часть работ по генерированию миниатюры непосредственно при сохранении записи, а при ее удалении также удаляет все миниатюры, сгенерированные на основе сохраненного в поле изображения. В остальном ведет себя так же, как знакомое нам поле Filefieid;

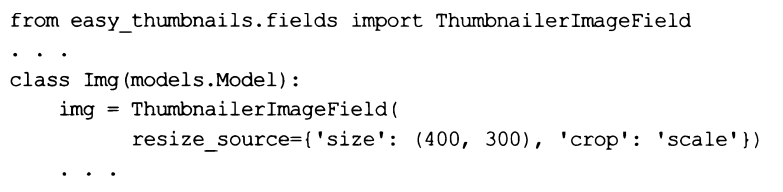
□ thumbnailerlmagefield —подкласс классов imagefield И thumbnaiierfieid.

Конструктор класса поддерживает дополнительный параметр resize source, задающий параметры генерируемой миниатюры. Эти параметры записываются в виде словаря в том же формате, что и при объявлении пресета.

Внимание!

Если В конструкторе класса thumbnailerlmagefield указать Параметр resize source, в поле будет сохранено не исходное изображение, а сгенерированная на его основе миниатюра.

Пример:

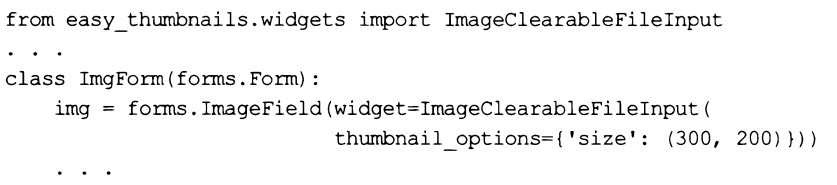


Теперь для вывода миниатюры, сохраненной в поле, можно использовать:



Если же параметр resize source в конструкторе поля не указан, то в поле будет сохранено оригинальное изображение.

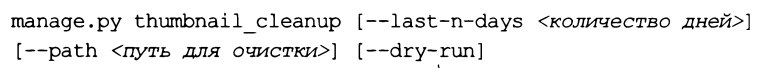
Эти поля могут представляться в форме элементом управления imageciearabiefilelnput, класс которого объявлен в модуле easy\_thumbnails. Widgets. Он является подклассом класса ciearabiefiieinput, но дополнительно выводит миниатюру выбранного в нем изображения. Параметры этой миниатюры можно указать в необязательном параметре thumbnail options в виде словаря. Пример:



Если параметры миниатюры для этого элемента не указаны, то они будут взяты из параметра thumbnail widget options настроек проекта.

**Дополнительная команда thumbnail cleanup**

Библиотека easy-thumbnails добавляет утилите manage.py поддержку команды Thumbnail cleanup, удаляющей все сгенерированные миниатюры. Формат ее вызова следующий:



Поддерживаются следующие дополнительные ключи:

□ -last-n-days — оставляет миниатюры, сгенерированные в течение указанного количества дней. Если не задан, будут удалены все миниатюры;

□ —path —удаляет миниатюры, хранящиеся по указанному пути. Если не указан, будут удалены все миниатюры;

□ -dry-run — выводит на экран сведения об удаляемых миниатюрах, но не удаляет их.