**Тема занятия № 32: Модуль 17. Формы и наборы форм: расширенные инструменты**

1. Формы, не связанные с моделями

Помимо форм и наборов форм, связанных с моделями, Django предлагает аналогичные инструменты, которые не связаны с моделями. Они могут применяться для указания как данных, не предназначенных для занесения в базу данных (например, ключевого слова для поиска), так и сведений, которые должны быть сохранены в базе после дополнительной обработки.

Кроме того, фреймворк предлагает расширенные инструменты для вывода форм и наборов форм на экран. А дополнительная библиотека Django Simple Captcha позволяет обрабатывать CAPTCHA.

Формы, не связанные с моделями, создаются и обрабатываются так же, как их “Коллеги”, которые связаны с моделями, за несколькими исключениями:

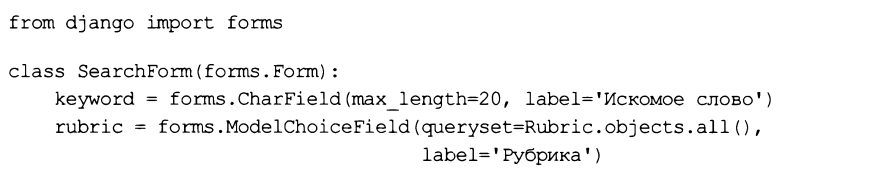
□ форма, не связанная с моделью, объявляется как подкласс класса Form из модуля Django.forms;

□ все поля, которые должны присутствовать в форме, необходимо объявлять в виде атрибутов класса формы;

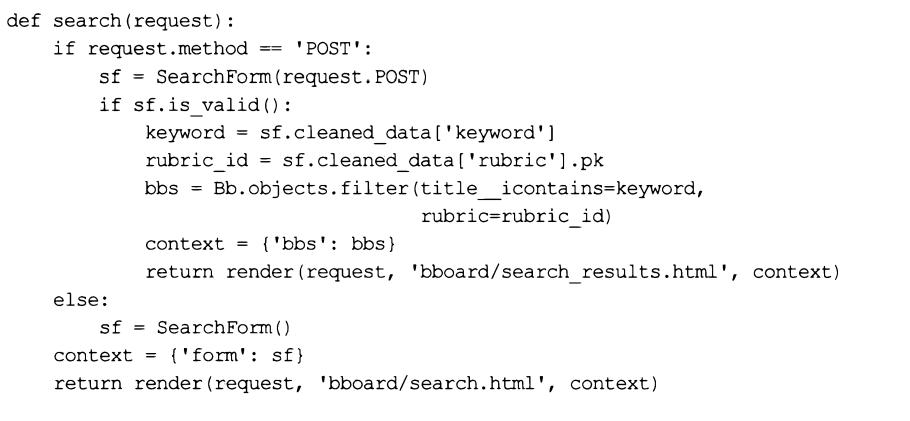
П вложенный класс Meta в такой форме не объявляется (что вполне понятно — ведь в этом классе задаются сведения о модели, с которой связана форма);

□ средства для сохранения введенных в форму данных в базе, включая метод Save () и параметр instance конструктора, не поддерживаются.

Приведен код не связанной с моделью формы, предназначенной для указания искомого ключевого слова и рубрики, в которой будет осуществляться поиск объявлений.

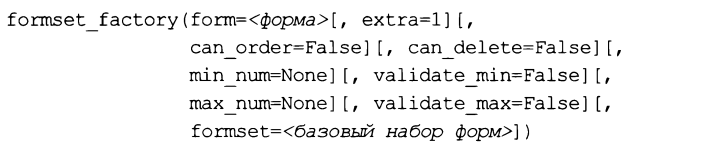


Приведен код контроллера, который извлекает данные из этой формы и использует их для указания параметров фильтрации объявлений (предполагается, что форма пересылает данные методом POST).

****

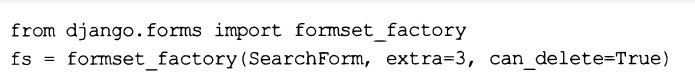
1. Наборы форм, не связанные с моделями

Наборы форм, не связанные с моделями, создаются с применением функции Formset\_factory () ИЗ модуля django. Forms:

****

Здесь форма, на основе которой создается набор форм, является обязательным параметром. Базовый набор форм должен быть производным ОТ класса baseformset ИЗ модуля django.forms.formsets.

Пример создания набора форм на основе формы searchform, код которой приведен:



Опять же, нужно иметь в виду, что набор форм, не связанный с моделью, не поддерживает средств для сохранения в базе занесенных в него данных, включая атрибуты new\_objects, changed\_objects И deteled\_objects. Код, сохраняющий введенные данные, придется писать самостоятельно.

В остальном же работа с такими наборами форм протекает аналогично работе с наборами форм, связанными с моделями. Мы можем перебирать формы, содержащиеся в наборе, и извлекать из них данные для обработки.

Если набор форм поддерживает переупорядочение форм (т. е. при его создании в вызове функции formset factory о был указан параметр can order со значением True), то в составе каждой формы появится поле order типа integerfield, хранящее порядковый номер текущей формы. Мы можем использовать его, чтобы выстроить в нужном порядке какие-либо сущности, или иным образом.

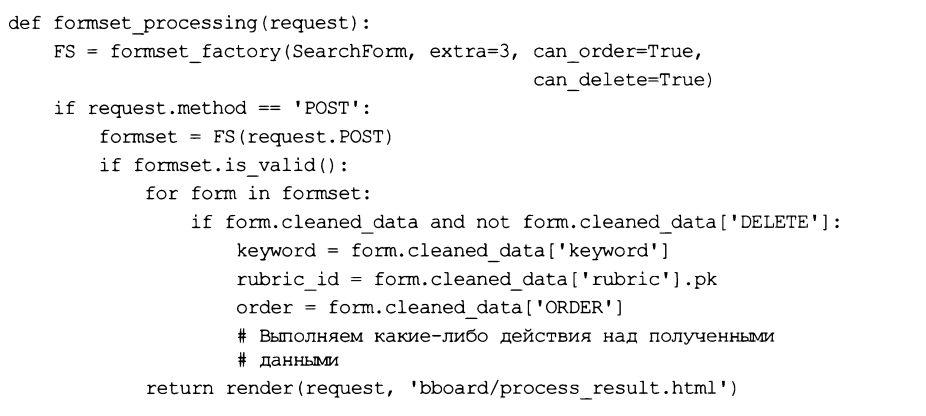
Если набор форм поддерживает удаление отдельных форм (добавить эту поддержку МОЖНО, записав в вызове функции formset\_factory () параметр can delete СО значением True), то в составе каждой формы появится поле delete типа Booleanfield. Это поле будет хранить значение True, если форма была помечена на удаление, и False — в противном случае.

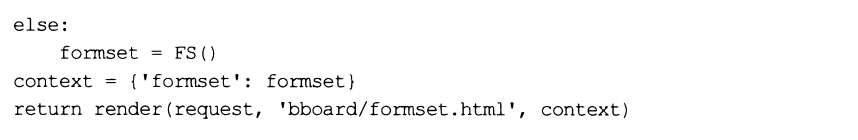
Класс набора форм, не связанного с моделью, поддерживает два полезных атрибута:

□ ordered forms — последовательность форм, которые были переупорядочены;

□ deleted—forms — последовательность удаленных форм.

Приведен код контроллера, который обрабатывает набор форм, не связанный с моделью.





1. Расширенные средства для вывода форм и наборов форм

Эти средства поддерживаются обеими разновидностями форм: и связанными с моделями, и не связанными с ними.

**Указание CSS-стилей у форм**

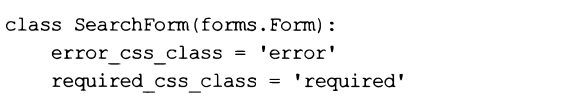
Для указания CSS-стилей, которые будут применены к отдельным элементам выводимой формы, классы форм поддерживают два атрибута класса:

□ required css ciass — имя стилевого класса, которым будут помечаться элементы управления, обязательные для заполнения;

□ error css class — имя стилевого класса, которым будут помечаться элементы управления с некорректными данными.

Эти стилевые классы будут привязываться к тегам <р>, <и> или <tr>, в зависимости от того, посредством каких HTML-тегов форма была выведена на экран.

Пример:

 **Настройка выводимых форм**

Некоторые настройки, затрагивающие выводимые на экран формы, указываются в виде именованных параметров конструктора класса формы. Вот эти параметры:

□ fieid\_order — задает порядок следования полей формы при ее выводе на экран.

В качестве значения указывается последовательность имен полей, представленных в виде строк. Если задать None, поля будут следовать друг за другом в том же порядке, в котором они были объявлены в классе формы. Значение по умолчанию — None. Пример:



□ label suffix— строка с суффиксом, который будет добавлен к тексту надписи при выводе. Значение по умолчанию — символ двоеточия;

□ auto id— управляет формированием якорей элементов управления, которые указываются в атрибутах id формирующих их тегов и тегов <iabei>, создающих надписи. В качестве значения параметра можно указать:

• строку формата — идентификаторы будут формироваться согласно ей. Символ-заменитель %s указывает местоположение в строке формата имени поля, соответствующего элементу управления. Пример:



Для поля keyword будет сгенерирован якорь id for keyword, а для поля

Rubric — якорь id\_for\_rubric;

• True — в качестве якорей будут использоваться имена полей формы, соответствующих элементам управления;

• False— якоря вообще не будут формироваться. Также не будут формироваться теги <iabei>, а надписи будут представлять собой простой текст.

Значение параметра по умолчанию: "id %s";

□ use required attribute — если True, ТО В теги, формирующие обязательные ДЛЯ заполнения элементы управления, будут помещены атрибуты required, если False, ТО ЭТОГО Не Произойдет (по умолчанию---True);

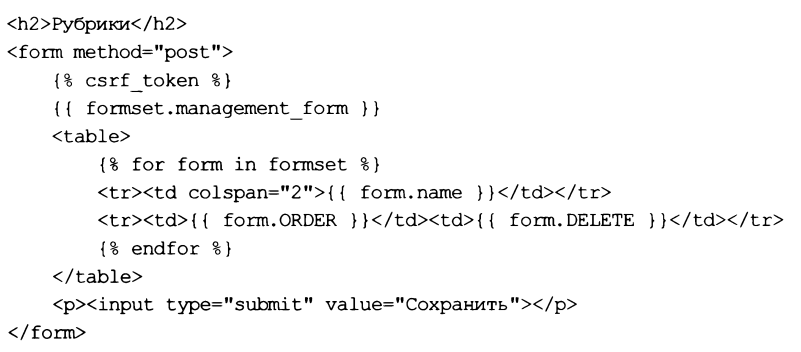
О prefix — строковый префикс для имен полей в выводимой форме. Применяется, если в один тег <form> нужно поместить несколько форм. По умолчанию — None (префикс отсутствует).

**Настройка наборов форм**

Конструкторы классов наборов форм поддерживают именованные параметры Autoid и prefix:

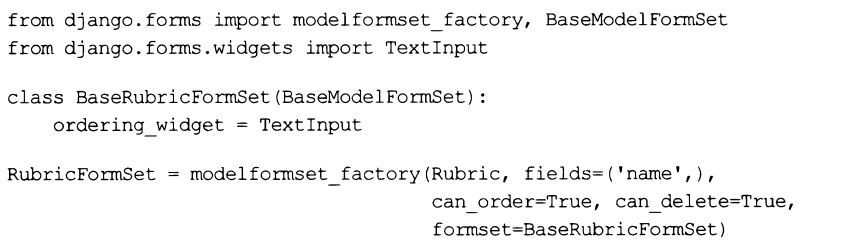


Поле порядкового номера order, посредством которого выполняется переупорядочивание форм, и поле удаления формы delete доступны через одноименные элементы. Это можно использовать, чтобы вывести служебные поля отдельно от остальных. Пример:

****

Начиная c Django 3.0, классы наборов форм поддерживают атрибут класса Ordering widget. В нем указывается класс элемента управления, посредством которого выводится поле порядкового номера формы order. По умолчанию используется класс Numberinput (поле для ввода целого числа).

Пример переупорядочивания форм в наборе с помощью обычного поля ввода:

****

Эта возможность пригодится при необходимости задействовать какую-либо дополнительную библиотеку, содержащую подходящий элемент управления.

1. Библиотека Django Simple Captcha: поддержка CAPTCHA

Если планируется дать пользователям-гостям возможность добавлять какие-либо данные в базу (например, оставлять комментарии), не помешает как-то обезопасить форму, в которую вводятся эти данные, от программ-роботов. Одно из решений — применение CAPTCHA (Completely Automated Public Turing test to tell Computers And Humans Apart, полностью автоматизированный публичный тест Тьюринга для различения компьютеров и людей).

CAPTCHA выводится на веб-страницу в виде графического изображения, содержащего сильно искаженный или зашумленный текст, который нужно прочитать и занести в расположенное рядом поле ввода. Если результат оказался верным, то, скорее всего, данные занесены человеком, поскольку программам такие сложные задачи пока еще не по плечу.

Для Django существует довольно много библиотек, реализующих в формах поддержку CAPTCHA. Одна из них — Django Simple Captcha.

На заметку!

Полная документация по библиотеке Django Simple Captcha находится здесь:

Https://django-simple-captcha.readthedocs.io/.

**Установка Django Simple Captcha**

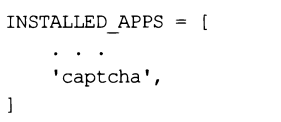
Установка этой библиотеки выполняется отдачей в командной строке следующей команды:



Вместе с Django Simple Captcha будут установлены библиотеки Pillow, djangoranged-response и six, необходимые ей для работы.

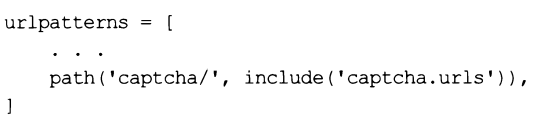
Чтобы задействовать библиотеку после установки, необходимо:

□ добавить входящее в ее состав приложение captcha в список приложений проекта (параметр installed\_apps настроек проекта, подробне):



Я выполнить миграции, входящие в состав библиотеки: Manage.py migrate

Я в списке маршрутов уровня проекта (в модуле urls.py пакета конфигурации) создать маршрут, связывающий префикс captcha и вложенный список маршрутов ИЗ МОДУЛЯ captcha.urls:

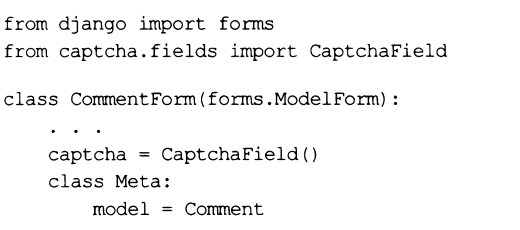


На заметку!

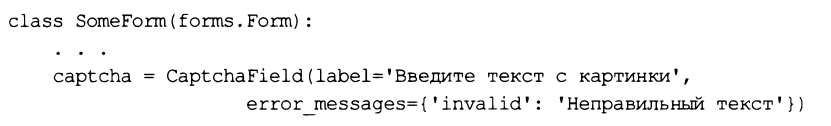
Для СВОИХ нужд Приложение captcha создает в базе данных таблицу captcha\_Captchastore. Она хранит сведения о сгенерированных на данный момент CAPTCHA, включая временную отметку их устаревания.

**Использование Django Simple Captcha**

В форме, в которой должна присутствовать CAPTCHA, следует объявить поле типа Captchafieid из модуля captcha. Fields. Это может быть как форма, связанная с моделью:

****

Так и несвязанная форма:

****

На веб-странице элемент управления, представляющий CAPTCHA, выглядит так, как показано на рис. 17.1. Если в параметре label конструктора не была указана надпись для него, то он получит надпись по умолчанию — Captcha.

Проверка правильности ввода CAPTCHA будет выполняться при валидации формы. Если был введен неправильный текст, форма не пройдет валидацию и будет повторно выведена на экран с указанием сообщения об ошибке.

Как было продемонстрировано в примере ранее, поле captchafieid поддерживает параметры, единые для всех типов полей, и, кроме того, еще два:

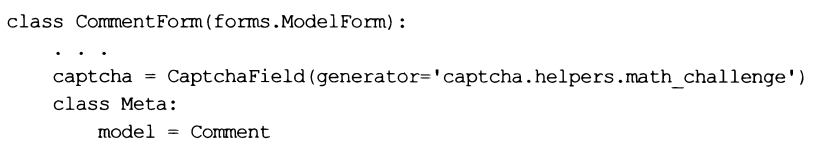
□ generator — полное имя функции, генерирующей текст для CAPTCHA, в виде строки. В библиотеке доступны следующие функции:

• captcha.helpers.random\_char\_challenge — классическая CAPTCHA в виде случайного набора из четырех букв. Нечувствительна к регистру;

• captcha.helpers.math chaiienge — математическая CAPTCHA, в которой посетителю нужно вычислить результат арифметического выражения и ввести получившийся результат;

• captcha.helpers .word challenge — словарная CAPTCHA, представляющая собой случайно выбранное слово из заданного словаря.

Пример:

****

Значение параметра по умолчанию берется из параметра captcha challenge Funct;

□ id prefix— строковый префикс, добавляемый к якорю, который указывается в атрибутах id тегов, формирующих элементы управления для ввода CAPTCHA.

Необходим, если в одной форме нужно вывести несколько CAPTCHA. По умолчанию —None (префикс отсутствует).

**Настройка Django Simple Captcha**

Параметры библиотеки указываются в настройках проекта— в модуле settings.py пакета конфигурации. Далее перечислены наиболее полезные из них (полный список параметров приведен в документации по библиотеке):

□ captcha challenge funct — полное имя функции, генерирующей текст для CAPTCHA, в виде строки.

По умолчанию:

"captcha.helpers.random\_char\_challenge";

□ captcha length — длина CAPTCHA в символах текста. Принимается во внимание только при использовании классической CAPTCHA. По умолчанию: 4;

□ captcha math challenge operator — строка с символом, обозначающим оператор умножения. Принимается во внимание только при использовании математической CAPTCHA. По умолчанию: Пример указания крестика в качестве оператора умножения:

CAPTCHA\_MATH\_CHALLENGE\_OPERATOR = 'х'

□ captcha words dictionary— полный путь к файлу со словарем, используемым в случае выбора словарной CAPTCHA. Словарь должен представлять собой текстовый файл, в котором каждое слово находится на отдельной строке;

□ CAPTCHA dictionary min LENGTH— минимальная длина слова, взятого из словаря, в символах. Применяется в случае выбора словарной CAPTCHA. По умолчанию: о;

□ captcha\_dictionary\_max\_length — максимальная длина слова, взятого из словаря, в символах. Применяется в случае выбора словарной CAPTCHA. По умолчанию: 99;

□ captcha timeout — промежуток времени в минутах, в течение которого сгенерированная CAPTCHA останется действительной. По умолчанию: 5;

□ captcha font path — полный путь к файлу шрифта, используемого для вывода текста. По умолчанию — путь "<папка, в которой установлен Python>\Lib\sitepackages\captcha\fonts\vera.ttf" (шрифт Vera, хранящийся в файле по этому пути, является свободным для распространения).

Также можно указать последовательность путей к шрифтам— в этом случае шрифты будут выбираться случайным образом;

□ captcha font size — кегль шрифта текста в пикселах. По умолчанию: 22;

□ captcha letter rotation— диапазон углов поворота букв в тексте CAPTCHA в виде кортежа, элементы которого укажут предельные углы поворота в градусах. По умолчанию: (-35, 35);

□ captcha foreground color— цвет текста на изображении CAPTCHA в любом формате, поддерживаемом CSS. По умолчанию: ”#001100” (очень темный, практически черный цвет);

□ captcha background color — цвет фона изображения CAPTCHA в любом формате, поддерживаемом CSS. По умолчанию: "#ffffff” (белый цвет);

□ captcha image size— геометрические размеры изображения в виде кортежа, первым элементом которого должна быть ширина, вторым — высота. Размеры исчисляются в пикселах. Если указать None, то размер изображения будет устанавливаться самой библиотекой. По умолчанию — None.

Элемент управления для ввода CAPTCHA выводится согласно шаблону, хранящемуся по пути <папка, в которой установлен Pyt/)on>\Lib\site-packages\captcha\templates\Captcha\widgets\captcha.html. Мы можем сделать копию этого шаблона, поместив ее в папке templates\captcha\widgets пакета приложения, и исправить соответственно своим нуждам. После этого CAPTCHA на странице будет выводиться с применением исправленного шаблона.

1. Дополнительные настройки проекта, имеющие отношение к формам

Дополнительные настройки проекта, имеющие отношение к формам осталось рассмотреть пару параметров, указываемых в настройках проекта и влияющих на обработку форм:

□ data upload max memory size — максимально допустимый объем полученных от посетителя данных в виде числа в байтах. Если этот объем был превышен, то генерируется исключение suspiciousoperation ИЗ модуля django.core.exceptions. если указать значение None, то проверка на превышение допустимого объема выполняться не будет. По умолчанию: 2621440 (2,5 Мбайт);

□ data\_upload\_max\_number\_fields — максимально допустимое количество POST-naраметров в полученном запросе (т. Е. Полей в выведенной форме). Если это количество было превышено, ТО генерируется исключение suspiciousoperation.

Если указать значение None, то проверка на превышение допустимого количества значений выполняться не будет. По умолчанию: юоо.

Ограничения, налагаемые этими параметрами, предусмотрены для предотвращения сетевых атак DOS (Denial Of Service, отказ от обслуживания). Увеличивать значения параметров следует, только если сайт должен обрабатывать данные большого объема.