Блог Толика Вострякова

Живой опыт программирования, Python, Django, современные языки и немного фотографий

Переход от Django темплэйтов к Jinja2 - 08.03.2010

Весь день переводил свой блог со стандартных Django темплэйтов на Jinja2, а на второй день тестировал скорость работы. Очень было интересно попробовать :) Опять же из-за в несколько раз более высокой скорости рендеринга темплэйтов. Сейчас блог работает на Jinja2. Делюсь опытом перехода.

Установка

Ставим Jinja2 с помощью pip:

```
sudo pip-2.6 install jinja2
```

Можно поставить и из исходников, как описано в документации, в чем вроде есть преимущество, так как можно использовать опцию "--with-speedups", для еще большей скорости работы.

Отличия от темплэйтов Django 1.2

1. Нужно явно указывать, что вы хотите вызвать метод с помощью круглых скобок.

   Django:

   ```
   {% for page in user.get_created_pages %}
       ...
   {% endfor %}
   ```

   Jinja2:

   ```
   {% for page in user.get_created_pages() %}
       ...
   {% endfor %}
   ```

2. Аргументы фильтров, если они конечно есть, нужно указывать в круглых скобках, а не после ":". Здесь проявляется преимущество Jinja2: можно указывать несколько аргументов.

   Django:

   ```
   {{ items|join:", " }}
   ```

   Jinja2:

   ```
   {{ items|join(', ') }}
   ```

3. Переменная "forloop", используемая в циклах {% for ... %}...{% endfor %}, называется "loop". Аналогом тэга {% cycle ... %}, является loop.cycle(...).

4. До Django 1.2 были отличия в тэге {% if ... %}, но сейчас их почти не осталось, в обоих системах вы можете писать, например так:

   ```
   {% if foo == 'bar' %}
       ...
   {% endif %}
   ```

   Соответственно нужно будет переписать все {% ifequal foo %}...{% endif %}, по аналогии с вышеприведенном примером, но изменения продолжат работать в Django 1.2.

   Теперь более серьезные отличия.

5. Нет многих стандартных джанговских тэгов и фильтров, о чем я напишу ниже.

6. И самое занимающее много времени при переходе: нет тэга {% load ... %}, а это значит, что все custom tags and filters нужно немного видоизменить под jinja2, что возможно, но займет основное время, если вы их активно пользуете.

   В остальном отличий нет, то есть, если посмотреть в целом, то языки темплэйтов действительно близки. Jinja2 конечно поддерживает много, чего не поддерживает Django, но это выходит за рамки рассматриваемой темы.

Процесс перехода

Сначала я хотел все сделать сам и написал для начала свой аналог render_to_response, который уже использовал для

рендерига Jinja2. Привел тэмплэйт центральной страницы блога к Jinja2-виду и попробовал загрузить ее в браузере. Наткнулся на то, что нет тэга url :)

В этот момент мне пришла в голову мысль, что я не первый перехожу с джанго-темплэйтов к Jinja2. Погуглил в интернете и нашел Django приложение chouwa, сильно облегчающее переход. А именно:

- реализованы недостающий тэг url и распростаненный фильтр date,
- реализован продуманный аналог метода render_to_response,
- продумана достаточно удобная система перевода сторонних тэгов и фильтров на Jinja2 фильтры и глобальные функции,
- есть замена функций показа страниц 404 и 500,
- к темплэйтам подключается локализация

Самому пришлось дописать несколько стандартых джанго-фильтров и тэги для поддержки джанго-комментариев. В основном просто копировал код из исходников джанги, с небольшими изменениями. Добавил новый код в файл defaultglobals.py в конец:

```
from jinja2 import contextfunction

@jinjafilter
def linebreaksbr(value, autoescape=None):
    """
    Converts all newlines in a piece of plain text to HTML line breaks
    (``
``).
    """
    if autoescape and not isinstance(value, SafeData):
        from django.utils.html import escape
        value = escape(value)
    return value.replace('\n', '
')

@jinjaglobal
@contextfunction
def get_csrf_token(context):
    csrf_token = context.get('csrf_token', None)
    if csrf_token:
        if csrf_token == 'NOTPROVIDED':
            return u""
        else:
            return u"" % (csrf_token)

# Django comments tags
@jinjaglobal
def comment_count(obj):
    return _get_query_set(obj).count()

@jinjaglobal
def comment_list(obj):
    return list(_get_query_set(obj))

@jinjaglobal
def get_comment_form(obj):
    content_type = ContentType.objects.get(app_label=obj._meta.app_label, model=obj._meta.object_name)
    return comments.get_form()(content_type.get_object_for_this_type(pk=obj.pk))

@jinjaglobal
def comment_form_target():
    return comments.get_form_target()

def _get_query_set(obj):
    content_type = ContentType.objects.get(app_label=obj._meta.app_label, model=obj._meta.object_name)

    qs = Comment.objects.filter(
        content_type = content_type,
        object_pk    = obj.pk,
        site__pk     = settings.SITE_ID,
    )

    # The is_public and is_removed fields are implementation details of the
    # built-in comment model's spam filtering system, so they might not
    # be present on a custom comment model subclass. If they exist, we
    # should filter on them.
    field_names = [f.name for f in Comment._meta.fields]
    if 'is_public' in field_names:
```

```
        qs = qs.filter(is_public=True)
    if getattr(settings, 'COMMENTS_HIDE_REMOVED', True) and 'is_removed' in field_names:
        qs = qs.filter(is_removed=False)

    return qs
```

Подправил немного код функции показа страницы 404 в файле views.py:

```
from chouwa.loader import get_template, render_to_string

def page_not_found(request, template_name='404.html'):
    return HttpResponseNotFound(render_to_string(template_name, None, request))
```

В остальном изменения по мелочи.

Тестирование скорости работы

Сравнивал сразу три варианта: темплэйты Dajngo, темплэйты Django с кэшированием, о котором я писал в предыдущем посте и темплэйты Jinja2 вместе с интеграцией через chouwa.

Первый по скорости, конечно оказалась Jinja2. В среднем быстрее на 1,5 секунды, при 100 запросах к центральной странице блога по сравнению с кэшированием темплэйтов Django. Немного, но и тэмплэйт в общем легкий. Что еще интересно - кэширование темплэйтов Django дает ощутимый прирост скорости, в среднем 5 секунд на теже 100 запросов.

Выводы

Переходить на Jinja2 в целом не сложно, если вы не используете массово сторонние тэги и фильтры в темплэйтах, иначе придется много менять. Из недостатков отмечу, что не везде вы можете заменить стандартную функцию render_to_response, на поддерживающую Jinja2 тэмплэйты. Такие тэмплэйты придется оставить с джанго-синтаксисом. Я наткнулся на такую ситуация при работе с джанго-комментариями.

Главный вывод, что прирост скорости в целом небольшой на проектах с простыми темплэйтами, как у меня. Поэтому переходить на Jinja2 стоит только, когда рендеринг темплэйтов действительно станет узким местом. С другой стороны, если вы начинаете проект с нуля, то почему бы не начать сразу с Jinja2 :)

Комментарии: 13

w31rd0 - 09.03.2010

А чем обусловлен выбор именно Jinja2, а не, скажем, Mako? По скорости они вроде бы примерно одинаковы (http://jinja.pocoo.org/2/documentation/faq#how-fast-is-it).

vostryakov - 09.03.2010

С Mako просто совсем не знаком, а про Jinja давно слышал, читал статью его автора про сравнение темплэйтов джанги и Jinja. Еще тогда меня это заинтересовало. Автор грамотный, он еще и отладчик делает для питона. Не знаю, как у Mako, а у Jinja2 есть, по мимо скорости, еще преимущества в более продвинутой системе темплэйтов.

Roman Imankulov - 12.03.2010

Кстати, еще у Jinja2, начиная с версии 2.2, есть такая приятная возможность, как отслеживание зависимостей одних шаблонов от других: (find*referenced*templates) [http://jinja.pocoo.org/2/documentation/api#the-meta-api].

На основе этого можно приделать кеширование страниц с умной инвалидацией кеша. Уже поиспользовал эту возможность, скрестив с pygraph для построения деревьев, оказалось весьма полезно (вот пример).

Roman Imankulov - 12.03.2010

Как жестоко не предоставлять возможность предпросмотра и редактирования постов. Первая ссылка выглядит как-то так: find_referenced_templates

vostryakov - 12.03.2010

Это из-за markdown. Подумаю, что можно сделать, чтобы подчеркивания не заменялись или действительно предпросмотр добавлю

slav0nic - 07.04.2010

неожиданно http://docs.djangoproject.com/en/dev/ref/templates/api/#using-an-alternative-template-language ... ;)

Толик Востряков - 08.04.2010

Я пришел к выводу, что лучше так не делать, а применять jinja2 темплэйты только там, где хочешь, то есть не на уровне всего проекта в целом. Просто не все темплёйты можно заменить на jinja2. Например темплэйты джанго-админки. Поэтому я написал свой вариант render*to*response и использую его во view, где я использую jinja2 темплэйты.

Sizeineds - 07.05.2010

интересно было бы с вами лично пообщаться)) вы в аське бываете? или в скайпе?...

Толик Востряков - 07.05.2010

"нтересно было бы с вами лично пообщаться)) вы в аське бываете? или в скайпе?... "

Да, можно. Написал в личку.

lega - 21.06.2010

> Поэтому переходить на Jinja2 стоит только, когда рендеринг темплэйтов действительно станет узким местом.

если рендеринг темплэйтов действительно станет узким местом, то нужно его кешировать, - не делать рендер на каждого клиента, а выдавать заранее построенные страницы (или частично построенные).

если нужно быстрый рендер то лучше воспользоваться Spitfire, (тесты http://stackoverflow.com/questions/1324238/what-is-the-fastest-template-system-for-python), но не нужно забывать что шаблонизатор создан в первую очередь для удобства разработки.

Алекс - 02.01.2011

С Новым Годом Вас !

Переход от Django темплэйтов к Jinja2 - Блог Толика Вострякова

Вашему сайту ставлю +

biznesmlm.ru

Побег

Фильмы

Sergey - 23.11.2011

А не подскажите ли вы, как использовать модуль sorl thumbnail в связке с django+jinja2 ?

Толик Востряков - 23.11.2011

Написать свои тэгэ для jinja2. За основу взять тэг из sorl thumbnail для стандартной джанго. Это достаточно просто на самом деле.

Добавить комментарий:

Имя или nickname:

E-mail:

Буду использовать, только для связи с тобой

Твой сайт или блог:

Чтобы и другие могли узнать о тебе

Текст комментария:

Вы можете использовать markdown разметку.

Например:
жирный
курсив
Заголовок

Не используйте html-тэги. Все ссылки станут активными, все переводы строк будут заменены на

А вы случаем не робот?

Отправить комментарий