Django 创建第一个项目 Django 模型

Django 模板

在上一章节中我们使用 django.http.HttpResponse() 来输出 "Hello World！"。该方式将数据与视图混合在一起，不符合 Django 的 MVC 思想。

本章节我们将为大家详细介绍 Django 模板的应用，模板是一个文本，用于分离文档的表现形式和内容。

模板应用实例

我们接着上一章节的项目将在 HelloWorld 目录底下创建 templates 目录并建立 hello.html文件，整个目录结构如下：

```
HelloWorld/
|-- HelloWorld
|   |-- __init__.py
|   |-- __init__.pyc
|   |-- settings.py
|   |-- settings.pyc
|   |-- urls.py
|   |-- urls.pyc
|   |-- view.py
|   |-- view.pyc
|   |-- wsgi.py
|   `-- wsgi.pyc
|-- manage.py
`-- templates
    `-- hello.html
```

hello.html 文件代码如下：

HelloWorld/templates/hello.html 文件代码：

```
<h1>{{ hello }}</h1>
```

从模板中我们知道变量使用了双括号。

接下来我们需要向Django说明模板文件的路径，修改HelloWorld/settings.py，修改 TEMPLATES 中的 DIRS 为 **[BASE_DIR+"/templates",]**，如下所示:

HelloWorld/HelloWorld/settings.py 文件代码：

```
...TEMPLATES = [
{
'BACKEND': 'django.template.backends.django.DjangoTemplates',
'DIRS': [BASE_DIR+"/templates",], # 修改位置
'APP_DIRS': True,
'OPTIONS': {
'context_processors': [
'django.template.context_processors.debug',
```

```
'django.template.context_processors.request',
'django.contrib.auth.context_processors.auth',
'django.contrib.messages.context_processors.messages',
],
},
},
]
...
```

我们现在修改 view.py，增加一个新的对象，用于向模板提交数据：

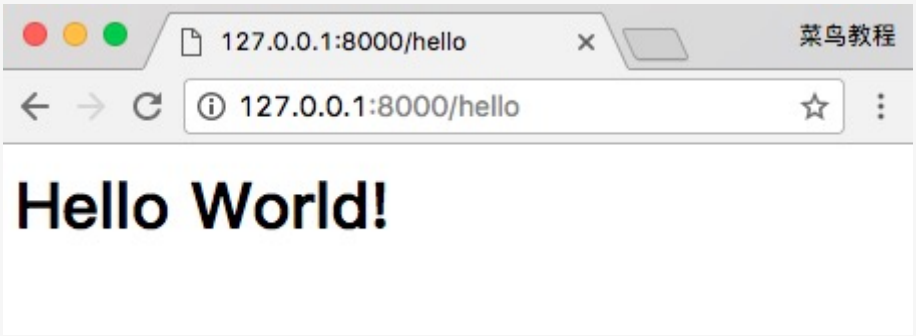
HelloWorld/HelloWorld/view.py 文件代码：

```
from django.shortcuts import render
def hello(request):
context = {}
context['hello'] = 'Hello World!'
return render(request, 'hello.html', context)
```

可以看到，我们这里使用 render 来替代之前使用的 HttpResponse。render 还使用了一个字典 context 作为参数。

context 字典中元素的键值 "hello" 对应了模板中的变量 "{{ hello }}"。

再访问访问 http://127.0.0.1:8000/hello，可以看到页面：



这样我们就完成了使用模板来输出数据，从而实现数据与视图分离。

接下来我们将具体介绍模板中常用的语法规则。

Django 模板标签

if/else 标签

基本语法格式如下：

```
{% if condition %}
     ... display
{% endif %}
```

或者：

```
{% if condition1 %}
   ... display 1
{% elif condition2 %}
```

```
    ... display 2
{% else %}
    ... display 3
{% endif %}
```

根据条件判断是否输出。if/else 支持嵌套。

{% if %} 标签接受 and ， or 或者 not 关键字来对多个变量做判断 ，或者对变量取反（ not)，例如：

```
{% if athlete_list and coach_list %}
     athletes 和 coaches 变量都是可用的。
{% endif %}
```

for 标签

{% for %} 允许我们在一个序列上迭代。

与Python的 for 语句的情形类似，循环语法是 for X in Y ，Y是要迭代的序列而X是在每一个特定的循环中使用的变量名称。

每一次循环中，模板系统会渲染在 {% for %} 和 {% endfor %} 之间的所有内容。

例如，给定一个运动员列表 athlete_list 变量，我们可以使用下面的代码来显示这个列表：

```
<ul>
{% for athlete in athlete_list %}
    <li>{{ athlete.name }}</li>
{% endfor %}
</ul>
```

给标签增加一个 reversed 使得该列表被反向迭代：

```
{% for athlete in athlete_list reversed %}
...
{% endfor %}
```

可以嵌套使用 {% for %} 标签：

```
{% for athlete in athlete_list %}
    <h1>{{ athlete.name }}</h1>
    <ul>
    {% for sport in athlete.sports_played %}
        <li>{{ sport }}</li>
    {% endfor %}
    </ul>
{% endfor %}
```

ifequal/ifnotequal 标签

{% ifequal %} 标签比较两个值，当他们相等时，显示在 {% ifequal %} 和 {% endifequal %} 之中所有的值。

下面的例子比较两个模板变量 user 和 currentuser :

```
{% ifequal user currentuser %}
    <h1>Welcome!</h1>
{% endifequal %}
```

和 {% if %} 类似， {% ifequal %} 支持可选的 {% else%} 标签：8

```
{% ifequal section 'sitenews' %}
    <h1>Site News</h1>
{% else %}
    <h1>No News Here</h1>
{% endifequal %}
```

注释标签

Django 注释使用 {# #}。

```
{# 这是一个注释 #}
```

过滤器

模板过滤器可以在变量被显示前修改它，过滤器使用管道字符，如下所示：

```
{{ name|lower }}
```

{{ name }} 变量被过滤器 lower 处理后，文档大写转换文本为小写。

过滤管道可以被* 套接* ，既是说，一个过滤器管道的输出又可以作为下一个管道的输入：

```
{{ my_list|first|upper }}
```

以上实例将第一个元素并将其转化为大写。

有些过滤器有参数。 过滤器的参数跟随冒号之后并且总是以双引号包含。 例如：

```
{{ bio|truncatewords:"30" }}
```

这个将显示变量 bio 的前30个词。

其他过滤器：

- addslashes : 添加反斜杠到任何反斜杠、单引号或者双引号前面。
- date : 按指定的格式字符串参数格式化 date 或者 datetime 对象，实例：

```
{{ pub_date|date:"F j, Y" }}
```

- length : 返回变量的长度。

include 标签

{% include %} 标签允许在模板中包含其它的模板的内容。

下面这个例子都包含了 nav.html 模板：

```
{% include "nav.html" %}
```

模板继承

模板可以用继承的方式来实现复用。

接下来我们先创建之前项目的 templates 目录中添加 base.html 文件，代码如下：

HelloWorld/templates/base.html 文件代码：

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>菜鸟教程(runoob.com)</title>
</head>
<body>
<h1>Hello World!</h1>
<p>菜鸟教程 Django 测试。</p>
{% block mainbody %}
<p>original</p>
{% endblock %}
</body>
</html>
```

以上代码中，名为 mainbody 的 block 标签是可以被继承者们替换掉的部分。

所有的 {% block %} 标签告诉模板引擎，子模板可以重载这些部分。

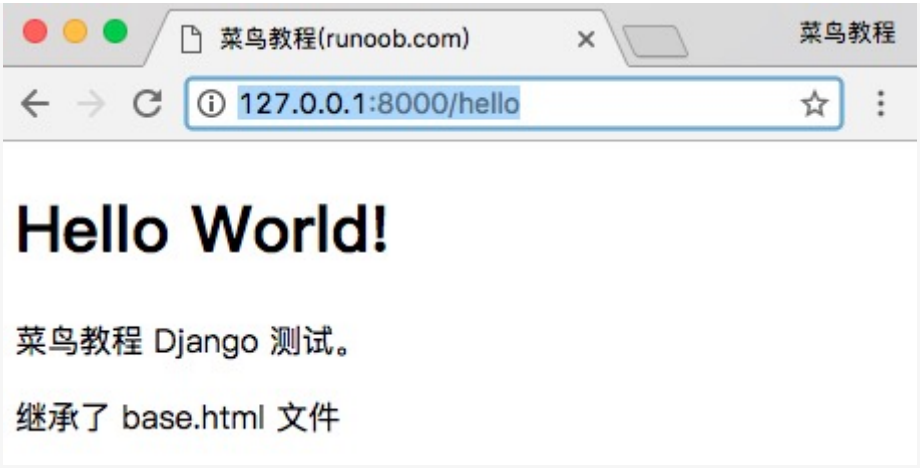
hello.html 中继承 base.html，并替换特定 block，hello.html 修改后的代码如下：

HelloWorld/templates/hello.html 文件代码：

```
{%extends "base.html" %}
{% block mainbody %}
<p>继承了 base.html 文件</p>
{% endblock %}
```

第一行代码说明 hello.html 继承了 base.html 文件。可以看到，这里相同名字的 block 标签用以替换 base.html 的相应 block。

重新访问地址 http://127.0.0.1:8000/hello，输出结果如下：



Django 创建第一个项目 Django 模型

1 篇笔记

写笔记

关于报错：**TemplateDoesNotExist （ Django 1.11.6 Python 3.6)**

没找到模板的问题一般都较为简单，在print(BASE_DIR)之后发现目录还有一级，再填上就好了。

附上配置：

```
'DIRS': [os.path.join(BASE_DIR, 'HelloWorld/templates')],
```

PS: 注意斜杠是 `/`。

Linux 严格区分大小写，所以该大写大写该小写小写。

xxdd 1年前 (2017-10-16)