

小白学Django第六天| 一文快速搞懂模板的使用

原创 JAP君 Python进击者

2020-02-04原文



小白学Django系列：

- 小白学Django第一天| MVC、MVT以及Django的那些事
- 小白学Django第二天| Django原来是这么玩的！
- 小白学Django第三天| 一文带你快速理解模型Model
- 小白学Django第四天| Django后台管理及配置MySQL数据库
- 小白学Django第五天| 视图View的初步使用
- 持续更新中...

用最短的时间学最多的知识，本文大约花费6分钟

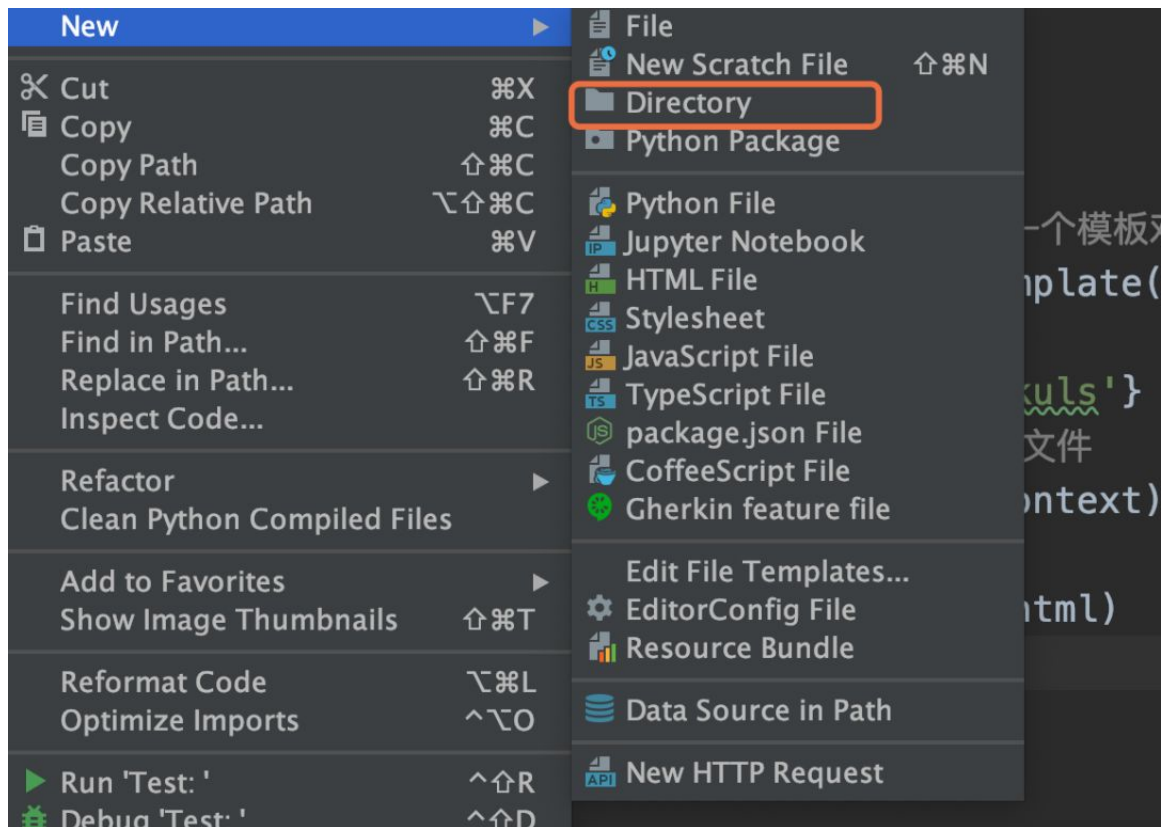
模板的初步使用

今天来教大家如何使用MVT中的T---模板，很多人认为模板仅仅就是一个HTML，其实这种观点是错误的，在模板templates里有很多的知识点需要我们学习，当然我们这篇文章只需要知道如何使用它。日后会有专门的文章来讲解模板的其他知识。

需要使用模板，只需要按照下面几个步骤做就行了：

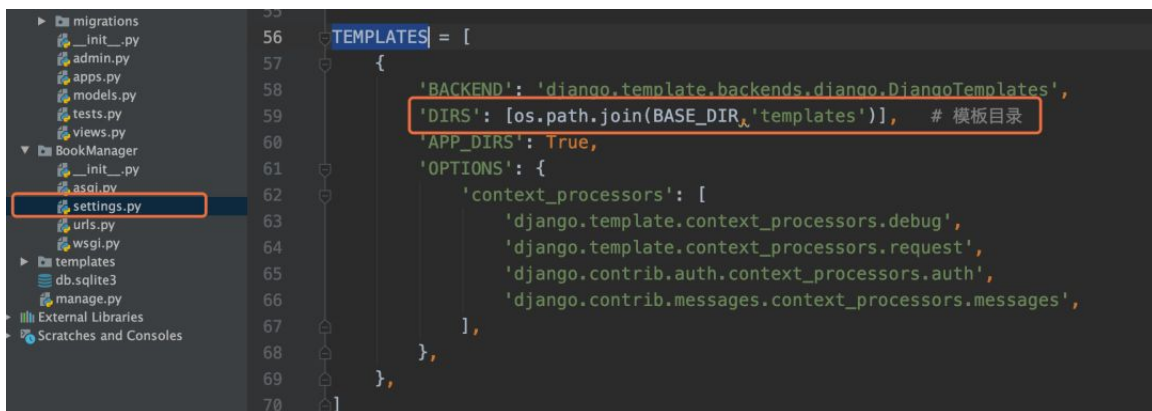
1. 创建模板目录

New -> Directory ->命名为templates



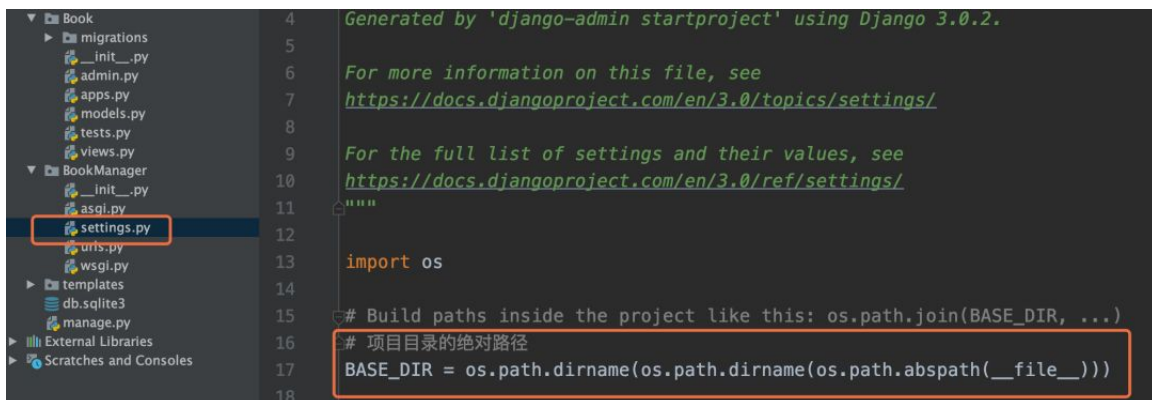
2.配置模板目录

这一步我们需要打开setting.py文件，在里面的TEMPLATES中配置目录。



```
56 TEMPLATES = [
57     {
58         'BACKEND': 'django.template.backends.django.DjangoTemplates',
59         'DIRS': [os.path.join(BASE_DIR, 'templates')], # 模板目录
60         'APP_DIRS': True,
61         'OPTIONS': {
62             'context_processors': [
63                 'django.template.context_processors.debug',
64                 'django.template.context_processors.request',
65                 'django.contrib.auth.context_processors.auth',
66                 'django.contrib.messages.context_processors.messages',
67             ],
68         },
69     ],
70 ]
```

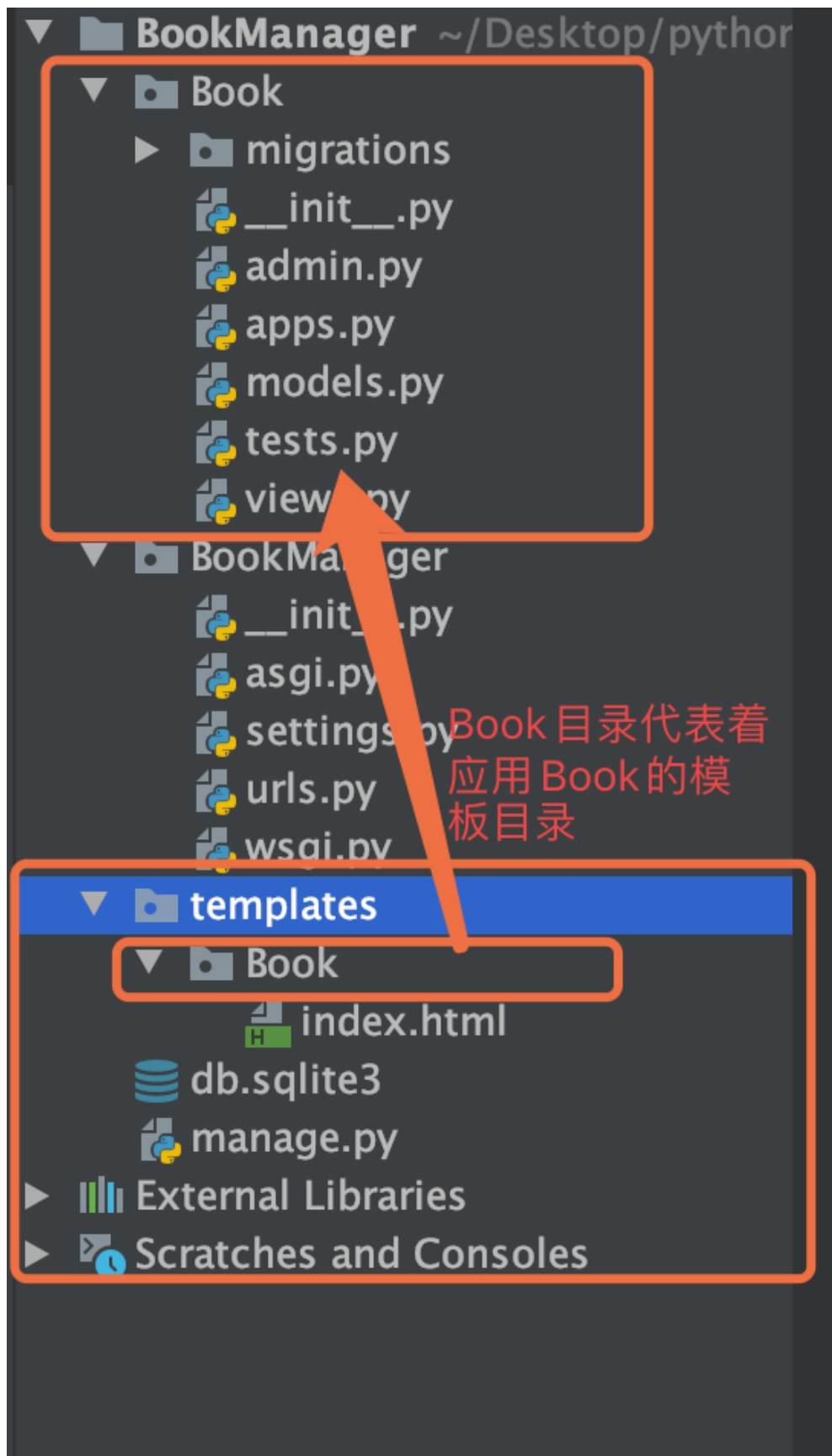
可能有些人好奇**BASE_DIR**是什么，其实在setting.py文件的上面是做了说明的，它就是**项目目录的绝对路径**



```
4 Generated by 'django-admin startproject' using Django 3.0.2.
5
6 For more information on this file, see
7 https://docs.djangoproject.com/en/3.0/topics/settings/
8
9 For the full list of settings and their values, see
10 https://docs.djangoproject.com/en/3.0/ref/settings/
11 """
12
13 import os
14
15 # Build paths inside the project like this: os.path.join(BASE_DIR, ...)
16 # 项目目录的绝对路径
17 BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
18
```

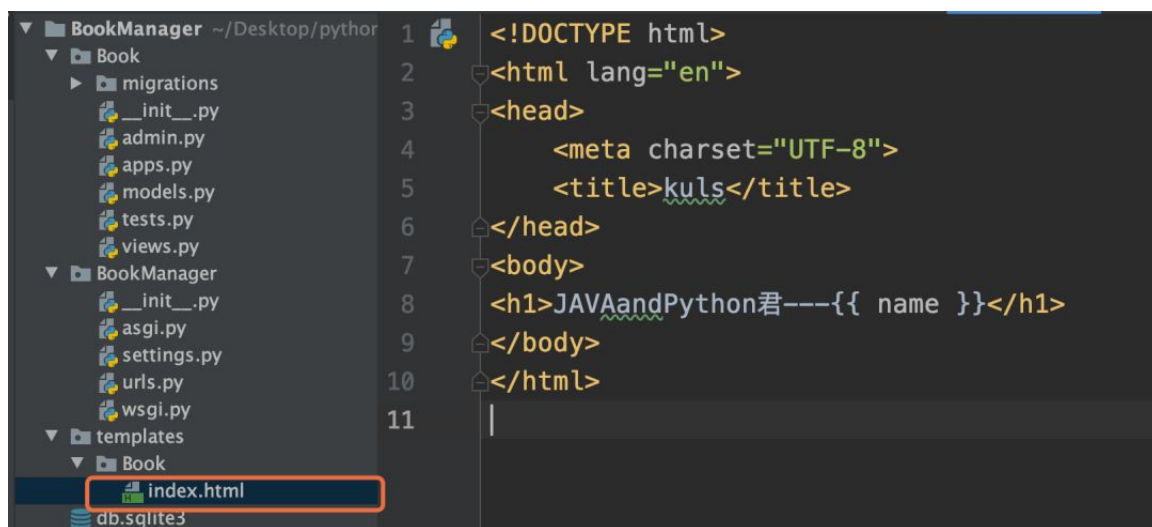
3.使用模板文件

开发时，我们对于每一个应用都有着相应的模板，所以我建议大家**在templates文件夹中新建一个名称和应用名称相同的模板目录**，没理解的可以看下图：



说完这个小技巧，接下来给大家正式说如何去使用一个模板文件。

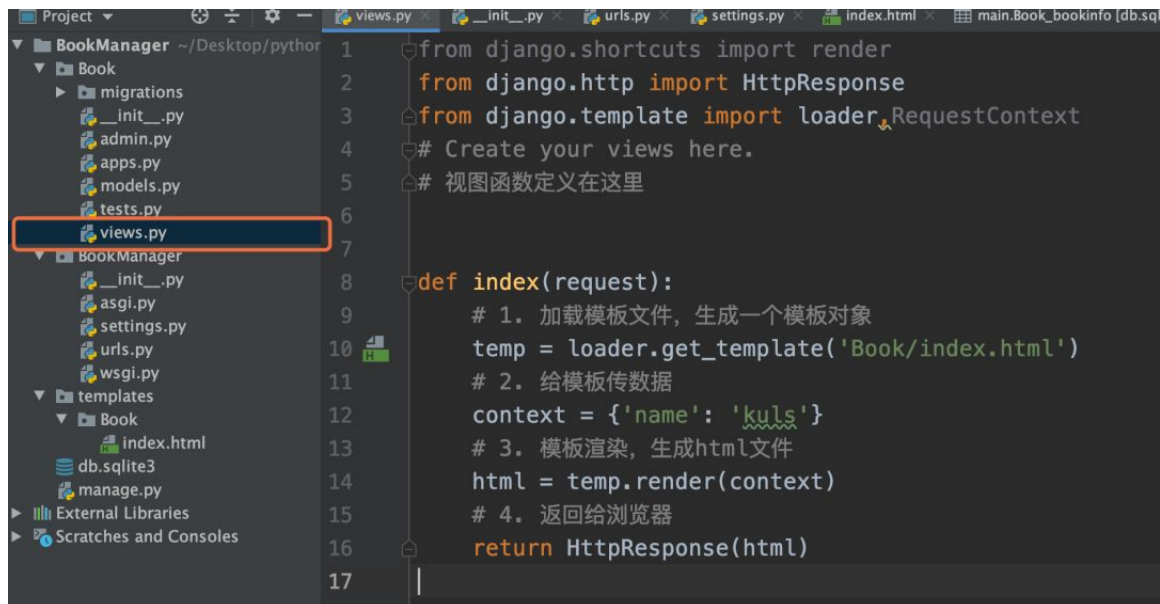
① 首先我们在模板目录中创建一个html文件，例如我创建的index.html



The screenshot shows a code editor with a file explorer on the left and a code editor on the right. In the file explorer, the 'templates' directory is expanded, and a new file 'index.html' is being created under the 'Book' subdirectory. The code editor shows the following HTML code:

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>kuls</title>
6 </head>
7 <body>
8     <h1>JAVAandPython君---{{ name }}</h1>
9 </body>
10 </html>
11
```

②进入我们相应的视图文件Book/view.py



The screenshot shows a code editor with a file explorer on the left and a code editor on the right. In the file explorer, the 'views.py' file is selected under the 'Book' subdirectory. The code editor shows the following Python code:

```
1 from django.shortcuts import render
2 from django.http import HttpResponse
3 from django.template import loader, RequestContext
4 # Create your views here.
5 # 视图函数定义在这里
6
7
8 def index(request):
9     # 1. 加载模板文件，生成一个模板对象
10    temp = loader.get_template('Book/index.html')
11    # 2. 给模板传数据
12    context = {'name': 'kuls'}
13    # 3. 模板渲染，生成html文件
14    html = temp.render(context)
15    # 4. 返回给浏览器
16    return HttpResponse(html)
17
```

代码我这里给出：

```
from django.shortcuts import render

from django.http import HttpResponse

from django.template import loader

# Create your views here.

# 视图函数定义在这里

def index(request):

    # 1. 加载模板文件，生成一个模板对象

        temp = loader.get_template('Book/index.html')

    # 2. 给模板传数据

        context = {'name': 'kuls'}

    # 3. 模板渲染，生成html文件

        html = temp.render(context)

    # 4. 返回给浏览器

        return HttpResponse(html)
```

每一句代码我都作了相应的注释，大家应该还是能够理解。**首先根据相应的html模板生成相应的模板对象，然后context当中传递的数据是给html中的数据，之后渲染模板，生成html文件，最后通过HttpResponse把文件返回给浏览器。**

最后的效果：

← → ↻ ⓘ 127.0.0.1:8000/index/

 应用  翻译  百度一下，你就知道  微信公众平台 

JAVAandPython君---kuls

4. 给模板文件传输数据

有些朋友可能看到我的index.html里面的代码有点好奇，那个双括号是干啥的？

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>kuls</title>

</head>

<body>

<h1>JAVAandPython君---{{ name }}</h1>

</body>

</html>
```

双括号的意思是括号中间那个是变量，也就是说此时它需要视图给它带来一个名字叫name的数据。

给模板传递数据的方法也有很多，除了上面的双括号变量外，**也有在html文件中使用for循环**

给大家一个简单的演示：

view.py

```
from django.shortcuts import render

from django.http import HttpResponse

from django.template import loader, RequestContext

# Create your views here.

# 视图函数定义在这里

def index(request):

    # 1. 加载模板文件，生成一个模板对象

    temp = loader.get_template('Book/index.html')

    # 2. 给模板传数据

    context = {'name': 'kuls', 'datas': list(range(1,20))}

    # 3. 模板渲染，生成html文件

    html = temp.render(context)

    # 4. 返回给浏览器

    return HttpResponse(html)
```

index.html

```
<!DOCTYPE html>
```



```
<html lang="en">

<head>

<meta charset="UTF-8">

<title>kuls</title>

</head>

<body>

<h1>JAVAandPython君---{{ name }}</h1>

{% for i in datas %}

<li>{{ i }}</li>

{% endfor %}

</body>

</html>
```

效果：



JAVAandPython君---kuls

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16
- 17
- 18
- 19

5. Django中的render函数

其实写到这里大家有没有发现我们的index视图函数当中的代码基本都是固定的，只有着 模板路径、传输数据是属于动态的。

```
def index(request):  
    # 1. 加载模板文件，生成一个模板对象  
    temp = loader.get_template('Book/index.html')  
    # 2. 给模板传数据  
    context = {'name': 'kuls', 'datas': list(range(1,20))}  
    # 3. 模板渲染，生成html文件  
    html = temp.render(context)  
    # 4. 返回给浏览器  
    return HttpResponse(html)
```

红色框柱的是变量，
其余的代码都没有变化。

那么既然这样我们可以封装它呀：

```
def render(request, templates_lj, context_dict={}):  
    # 1. 加载模板文件，生成一个模板对象  
    temp = loader.get_template(templates_lj)  
    # 2. 给模板传数据  
    context = context_dict  
    # 3. 模板渲染，生成html文件  
    html = temp.render(context)  
    # 4. 返回给浏览器  
    return HttpResponse(html)
```

其实Django这个框架是比较完善的框架，开发者自然也能想到这一点，所以每当我们创建一个应用时，view.py文件当中会自动帮我们导入Django框架中封装好的render。

```
from django.shortcuts import render
from django.http import HttpResponse
from django.template import loader, RequestContext
# Create your views here.
# 视图函数定义在这里
```

也就是说我们之前写的代码都可以通过这个函数来代替。既然这样为什么还要学呢？废话，要想真的掌握一个知识，只有把它的本质给理解了，你才算真真掌握了。

我是kuls

欢迎加我微信

备注姓名+学习方向+学校（公司）

拉你进Java、Python交流群



设为星标 ★



JAVAandPython君

📍 扫码关注不迷路

技术干货 · 经验分享 · 成长心得 · 系统教程

精选留言

暂无...