晨测

- 1. 创建一个python3.5.2的虚拟环境
- 2. 启动django服务,并且能够使用127.0.0.1:9000访问
- 3. url输入 127.0.0.1:8000/index/2019/hello发出请求,能够返回 2019年说了一句hello,写出路由 视图
- 4. 画出MVT模型,并说明mvt都代表什么

回顾

创建虚拟环境

conda create -n MINGZI python=3.5.2

激活虚拟环境

activate DjangoPath

安装django

pip install django==2.2.1

创建工程

启动项目

pycarm打开并编写代码

路由

模板

1. 调用一个html页面

```
from django.template import Template,Context

def gethtml(request):
    html="""
    <html>
        <head>
        <head>
        <body>
        <h1>我是一个h1标签</h1>
        <h2>我是{{ name }}</h2>
        <a href =
"https://baike.baidu.com/item/%E5%BE%B7%E6%80%80%E6%81%A9%C2%B7%E9%9F%A6%E5%BE%B7/5457042?fromtitle=%E9%9F%A6%E5%BE%B7&fromid=64575&fr=aladdin">
```

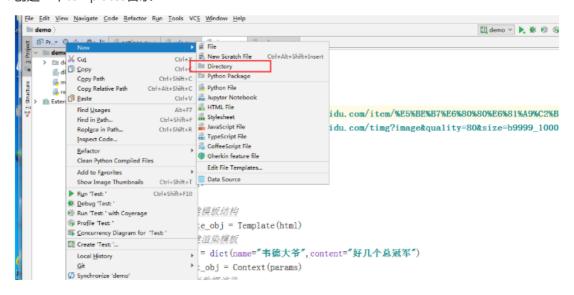
```
<img src='https://timgsa.baidu.com/timg?</pre>
image&quality=80&size=b9999_10000&sec=1568003957396&di=a9ef056162060cc8d0bb4e830
d80f1a4&imgtype=0&src=http%3A%2F%2F03imgmini.eastday.com%2Fmobile%2F20180917%2F6
2855583b6b5e607bdd7c8d54c34c8dc_wmk.jpeg' title = '这是韦德' alt='这个是韦德'>
       {{ content }}
       </body>
   </html>
   #1.构建模板结构
   tempalte_obj = Template(html)
   #2. 创建渲染模板
   params = dict(name="韦德大爷",content="好几个总冠军")
   content_obj = Context(params)
   #3.进行数据渲染
   result = tempalte_obj.render(content_obj)
   #4.返回结果
   # return HttpResponse(html)
   return HttpResponse(result)
```

这种方式虽然能够实现调用一个html页面,并且能够将数据渲染上去,但是页面依然嵌套在python代码中,编写不方便。因此不常用。

模板标签系统介绍

在做web开发,要给用户提供一个页面,页面包括静态页面+数据,两者结合起来就是完整的可视化的页面,django的模板系统支持这种功能,首先需要写一个静态页面(结构html,样式css,行为js)然后通过python的模板语法将数据渲染上去。

1. 创建一个templates目录



2. 配置模板

```
'demo.urls
 m demo C:\Users\Ad
                        ROOT URLCONF =

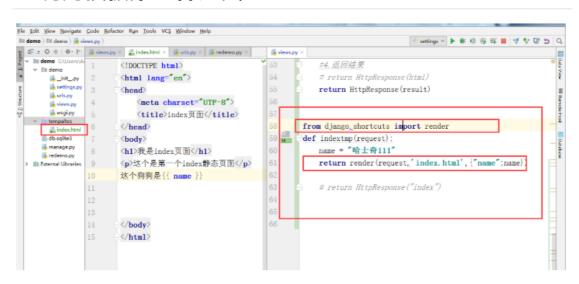
√ I demo

                        # 模版设置
     settings.py
                        TEMPLATES = [
                64
      wsgi.py
  tempaltes
                                '\,BACKE\underline{ND'}:\ '\,d\,jango.\ template.\ backends.\ d\,jango.\ D\,jangoTemplates'\,,
                                'DIRS': [os.path.join(BASE_DIR, 'tempaltes')],
                66
   illi-manage.py
    a reder
                                'APP_DIRS': True,
                67
                                 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',
                       \triangle 1
                                                                                                          ●英 5 0 2 四 4 単 4
```

配置项介绍

```
TEMPLATES = [
       'BACKEND': 'django. template. backends. django. DjangoTemplates',
       # 指向django的模板引擎 现在用的是django模板引擎 jinja2
       # 定义一个目录列表, 寻找模板的位置
       'DIRS': [os.path.join(BASE_DIR,'tempaltes')],
       # 默认 True 默认会在app中寻找模板(templates)当APP_DIRS 为True的时候DIRS 后面可以为空
       'APP_DIRS': True,
       # 针对后端的一些设置
       'OPTIONS': {
           'context processors': [
              'django.template.context_processors.debug',
              'django.template.context_processors.request',
              ^{\prime} django. contrib. auth. context_processors. auth ^{\prime} ,
              \tt 'django.\ contrib.\ messages.\ context\_processors.\ messages',
           ],
       },
                                                                             🔼 ф о. 0 🛂 📟 💄 🕶 🕏
```

1. 调用模板第一种方法



2. 调用模板第二种方法

```
ற்ச இரு ் இரு பிரும் ws.py × விர்வி index.html × விரும்.html × விரும்.html ×
                     <!DOCTYPE html>
                                                                                           # return HttpResponse(html)
 ∨ 🖿 demo
ili__init__py
                                                             O O O
                        <html lang="en">
                                                                                           return HttpResponse(result)
                        <head>
                                                                              56
                            <meta charset="UTF-8">
     // views.pv
     🐌 wsgi.py
                            <title>abc</title>
                                                                                       from django. shortcuts import render
    tempaltes
abc.html
                                                                              59
                                                                                       def indextmp(request):
                                                                                           name = "哈士奇111"
                        <body>
   🔓 db.sqlite3
                        <h1>abc</h1>
                                                                                           return render (request, 'index. html', {"nam
   amanage.py
 i redemo.py
I External Librari
                        {{ name }}
                                                                                           # return HttpResponse("index")
                        </body>
                        </html>
                                                                                       from django. shortcuts import render_to_response
                                                                                       def abc(request):
                                                                                           name = "hello"
                                                                                           return render_to_response("abc. html", {"na
                                                                                                                         ○英 5 ② 2 四 4 1 1
```

3. 调用模板的第三种方法

```
from django. template. loader import get_template

def abc(request):

template = get_template("abc. html")
name = "hello"
result = template. render({'name':name})

return HttpResponse(result)
```

模板语法

1. 变量的使用

在摸板中,如果想要使用后端传入的数据,就要用到模板语法去实现。

django提供的模板语法:变量的使用,控制语句,循环

django模板html文件中使用变量语法: {{ }}

```
params - urcu(name- znansan ,age-ro,noovy-l eat , sing , pingpang ,
                                                                                \title/tprtest\/trtle/
      return render (request, "tpltest. html", params)
                                                                        6
                                                                                               ◎ ◎ ⊘ ○
                                                                               <body>
                                                                        8
                                                                               <h1>这个是模板测试页面</h1>
                                                                     9
def tpltest(request):
   name = "zhangsan"
                                                                               姓名: {{ name }}
    age = 19
    hobby = ["eat", "sing", "pingpang", 'football']
                                                                              年龄: {{ age }}
                                                                               <h2>爱好{{ hobby }}</h2>
    score = {"shuxue":89, "yingyu":90, "yuwen":100}
                                                                      14
                                                                               <h2>第一个爱好{{ hobby. 1 }}</h2>
    return render (request, "tpltest. html", {"name":name, "age":age, "hobby" 15
                                                                               <h2>分数{{ score }}</h2>
                                                                      16
                                                                               <h2>数学成绩: {{ score. shuxue }}</
 tpltest()
```

使用locals()方法,作用是将当前视图函数的所有局部变量返回

```
def tpltest(request):
    name = "zhangsan"
    age = 19
    hobby = ["eat", "sing", "pingpang", 'football']
    score = {"shuxue":89, "yingyu":90, "yuwen":100}

# return render(request, "tpltest. html" {"name":name, "age":age, "hobby":hobby,
    return render(request, "tpltest. html", locals())
```

2. 控制语句

```
if else elif
模板中使用控制语句,语法结构
{% if 条件 %}
{% elif 条件 %}
{% else %}
{% endif %}
```

```
urlpatterns = [
                                                 <h2>爱好{{ hobb o d d hobo
86
                                                                                              path('admin/', admin.site.urls),
                                                 <h2>第一个爱好{{ hobby. 1 }}</h
87
                                                                                             path('index/', views.index),
path('about/', views.about),
         def tpltest(request, age
                                                 <h2>分数{{ score }}</h2>
             print(age)
                                                 ◆b2>数学成绩: {{ score. shuxue
89
             print_(type(age))
                                                                                              re_path(r'^*, views.index),
                                                 <h2>{{ say. say }}</h2>
90
             class Say(object):
                                                                                              re_path(r'urltest/(\d)', views.urltest)
91
                def say(self):
                                                                                              re_path(r'urltestnew/(?P<year>\d{4})/(
                                                 \{\% \text{ if age} > 18 \text{ and age} < 40 \}
                    return "hello"
92
                                                                                              path("gethtml/", views. gethtml),
                                                     青壮年
93
                                                                                              path("indextmp/", views. indextmp),
                                                 {% elif age >= 40 %}
             name = "zhangsan"
94
                                                                                              path("abc/", views.abc),
                                                     \p>中年
95
                                                                                              re_path("tpltest/(\d+)", views. tpltest),
                                                 {% else %}
96
            age= int(age)
                                                    >少年
             hobby = ["eat", "sing", "p
97
             score = {"shuxue":89, "yir 25
                                                 {% endif %}
98
99
             say = Say()
             # return render (request.
                                                 </html>
             return render (request, "tr
                                                 html \rightarrow body \rightarrow h2
```

3. 循环语句

1. 遍历列表

2. 遍历字典

3. forloop

forloop是一个django自带的一个变量,作用于循环,记录循环的次数,可以对循环的内容进行控制, 判断,通常和if连用

```
41
             {% endtor %}
42
      {% for one in hobby reversed %}
43
44
            {% if forloop.first %}
                \begin{tabular}{ll} $\langle p$ style="color: red" $\rangle $ ({ forloop. counter }) $. $ ({ one }) $ 
45
             {% elif forloop.last %}
46
               fp style="color: blue">{{ forloop.counter }}.{{ one }}
47
             {% else %}
48
                {{ forloop. counter }}. {{ one }}
49
50
            {% endif %}
        {% endfor %}
```