小白学Flask第十天| 宏、继承、包含、特殊变量

JAP君 Python进击者
2019-09-21原文

主要内容：

1. 模板宏的使用
2. 宏定义在外部的使用
3. 模板的继承
4. 模板的包含
5. Flask中的特殊变量和方法

模板宏的使用

大家可能是第一次接触这个字，在这里是什么意思呢？**宏其实就类似于python中的函数，宏的作用就是在模板中重复利用代码，避免代码冗余。**

我们来看一个宏的写法（不带参数的宏）：

```
{% macro input() %}

  <input type="text"
```

```
        name="username"
        value=""
        size="30"/>
{% endmacro %}
```

上面代码中，其实就类似于在python中定义了一个名为input的函数。

定义了这个宏，那该如何去使用它？

```
{{ input()}}
```

直接通过两个大括号就ok了

既然有不带参数的宏，肯定有带参数的宏，如何写？看下面：

```
{% macro input(name,value='',type='text',size=20) %}
    <input type="{{ type }}"
           name="{{ name }}"
           value="{{ value }}"
           size="{{ size }}"/>
{% endmacro %}
```

使用：

```
{{ input(value='name',type='password',size=40)}}
```

相信大家仔细看一下代码就能够明白这是什么意思，大家完完全全可以按照函数的思想去对待宏。

宏定义在外部的使用

宏除了在模板当中去编写，还能单独一个模板专门来写宏吗？答案当然是可以的，其实这一点也很类似于python。

Jinja2支持宏，还可以导入宏，需要在多处重复使用的模板代码片段可以写入单独的文件，再包含在所有模板中，以避免重复。

我们创建一个macro.html文件

```
{% macro input() %}
    <input type="text" name="username" placeholde="Username">
    <input type="password" name="password"
placeholde="Password">
    <input type="submit">
{% endmacro %}
```

在其他模板当中使用，需要先导入：

```
{% import 'macro.html' as func %}
{% func.input() %}
```

模板的继承

模板继承是为了重用模板中的公共内容。一般Web开发中，继承主要使用在网站的顶部菜单、底部。这些内容可以定义在父模板中，子模板直接继承，而不需要重复书写。

`{% block top %}``{% endblock %}`标签定义的内容，相当于在父模板中挖个坑，当子模板继承父模板时，可以进行填充。

子模板使用extends指令声明这个模板继承自哪？父模板中定义的块在子模板中被重新定义，在子模板中调用父模板的内容可以使用super()。

例如我们创建一个父模板base.html

```
{% block top %}
    顶部菜单
{% endblock top %}


{% block content %}
{% endblock content %}


{% block bottom %}
  底部
{% endblock bottom %}
```

子模板：

```
{% extends 'base.html' %}
{% block content %}
   需要填充的内容
{% endblock content %}
```

模板继承使用时注意点：

- 不支持多继承。
- 为了便于阅读，在子模板中使用extends时，尽量写在模板的第一行。
- 不能在一个模板文件中定义多个相同名字的block标签。

- 当在页面中使用多个block标签时，建议给结束标签起个名字，当多个block嵌套时，阅读性更好。

模板的包含

Jinja2模板中，除了宏和继承，还支持一种代码重用的功能，叫包含(Include)。它的功能是将另一个模板整个加载到当前模板中，并直接渲染。

include怎么使用？

```
{\% include 'hello.html' %}
```

包含在使用时，如果包含的模板文件不存在时，程序会抛出TemplateNotFound异常，可以加上ignore missing关键字。如果包含的模板文件不存在，会忽略这条include语句。

示例：

include的使用加上关键字ignore missing

```
{\% include 'hello.html' ignore missing %}
```

总结一下上面讲的宏、继承、包含：

- 宏(Macro)、继承(Block)、包含(include)均能实现代码的复用。
- 继承(Block)的本质是代码替换，一般用来实现多个页面中重复不变的区域。
- 宏(Macro)的功能类似函数，可以传入参数，需要定义、调用。
- 包含(include)是直接将目标模板文件整个渲染出来。

Flask中的特殊变量和方法

这里给大家补充一下Flask当中的一些特殊的变量和方法，大家之前肯定也看到过。

config 对象:

```
config 对象就是Flask的config对象， 也就是 app.config 对象。

{{ config.SQLALCHEMY_DATABASE_URI }}
```

request 对象:

就是 Flask 中表示当前请求的 request
对象，request对象中保存了一次HTTP请求的一切信息。

request常用的属性如下：

属性	说明	类型
data	记录请求的数据，并转换为字符串	*
form	记录请求中的表单数据	MultiDict
args	记录请求中的查询参数	MultiDict
cookies	记录请求中的cookie信息	Dict
headers	记录请求中的报文头	EnvironHeaders
method	记录请求使用的HTTP方法	GET/POST
url	记录请求的URL地址	string
files	记录请求上传的文件	*

url_for 方法:

url_for() 会返回传入的路由函数对应的URL，所谓路由函数就是被 app.route()
路由装饰器装饰的函数。如果我们定义的路由函数是带有参数的，则可以将这些参数作为命名参数传入。

```
{{ url_for('index') }}


{{ url_for('post', post_id=1024) }}
```

get_flashed_messages方法：

返回之前在Flask中通过 flash()
传入的信息列表。把字符串对象表示的消息加入到一个消息队列中，然后通过调用 get_flashed_messages() 方法取出。

```
{% for message in get_flashed_messages() %}
    {{ message }}
{% endfor %}
```

这里多说两句，get_flashed_messages()
所处理的信息只能看一次，也就是说你访问过里面的信息一次了，第二次你就看不到了，我们也称之为闪现。

这个是如何实现的呢？它的数据其实是存储在session当中，当你访问了一次之后就会进行删除。用到了session所以我们需要设置SERECT_KEY(之前我们有写过)
不然就会报错。

Flask系列文章：

小白学Flask第一天 | 我的第一个Flask程序

小白学Flask第二天| app对象的初始化和配置

小白学Flask第三天| 今天把视图函数的路由给讲清楚！

小白学Flask第四天| 把路由转换器玩的更牛逼

小白学Flask第五天 | 详解很重要的request对象

小白学Flask第六天| abort函数、自定义错误方法、视图函数的返回值

小白学Flask第七天| 讲讲cookie和session的操作

小白学Flask第八天| Flask上下文和请求钩子

小白学Flask第九天| 看看模板的那些事（一）

持续更新中...

精选留言

暂无...