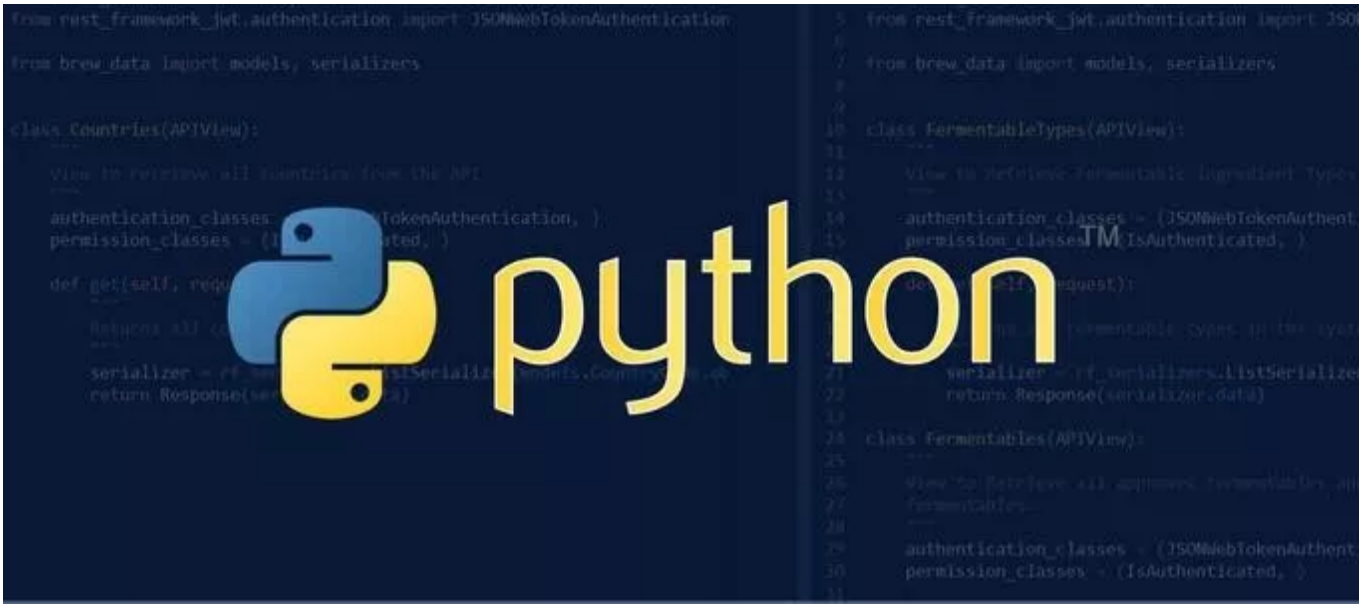
第44天：Flask 框架集成Bootstrap

原创 太阳雪 Python技术 2019-10-31



前面学习了 Flask 框架的基本用法，以及模板引擎 Jinja2，按理说可以开始自己的 Web 之旅了，不过在启程之前，还有个重要的武器需要了解一下，就是著名的 Bootstrap 框架和 Flask 的结合，这将大大提高开发 Web 应用的效率。

简介

Bootstrap 是 Twitter 公司的设计师 Mark Otto 和 Jacob Thornton 开发的 Web 项目开发框架，简洁、直观、强悍，使得 Web 开发更加快捷，一经推出后颇受欢迎，一直是 GitHub 上的热门开源项目，这么好的开发框架 Flask 一定少不了。

期初在 Flask 的扩展包中有 flask-bootstrap, 不过后来更新很少，后来 greyli 基于 flask-bootstrap 开发了 bootstrap-flask 扩展模块，不经支持最新的 Bootstrap4， 还基于 Jinjia2 模板引擎的宏，做了更多的扩展。值得一提的是 greyli 是个中国帅小伙，能在文章后面的参考链接中找到他。

安装

使用 pip 安装：

```
pip install bootstrap-flask
```

> 如果之前安装过 `flask-bootstrap`，需要将其卸载掉，不然两者会有冲突，如果您遇到错误提示：`jinja2.exceptions.UndefinedError: 'bootstrap' is undefined`，很大可能就是这个问题

小试牛刀

导入模块

创建一个 `flaskbootstrap.py` 程序文件，引入模块 `flask_bootstrap`，您没看错，引入的并不是 `bootstrap-flask`：

```
from flask_bootstrap import Bootstrap
```

> 注意：我在做示例时，将代码文件名定义为 `flask_bootstrap.py`，运行时提示 无法导入 Bootstrap 模块，这是因为文件名与模块 `flask_bootstrap` 冲突了

然后对 Flask 应用初始化：

```
app = Flask(__name__) # 创建一个 Flask 应用
bootstrap = Bootstrap(app) # 为应用初始化 bootstrap
```

给应用加载 bootstrap 主要是给应用加上 Jinja2 的扩展，下面的工作就是写模板文件。

创建基础模板

`bootstrap-flask` 虽然基于 `flask-bootstrap`，但是却没有提供默认的模板文件，期望在后续版本中能有吧，不过自己写也不麻烦，我们在文件夹 templates 中创建一个 base.html，内容是：

{% raw %}

```
<!-- 引入导航模块的宏 render_nav_item -->
{% from 'bootstrap/nav.html' import render_nav_item %}
<!-- 下面是正常的 Bootstrap 页面代码，看起来很复杂，不过可以从官网上拷贝 -->
<!DOCTYPE html>
<html lang="cn">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=
        <title>{% block title %} Flask Bootstrap {% endblock %}</title> <
        <link rel="icon" href="{{ url_for('static', filename='favicon.ico
        {{ bootstrap.load_css() }}  <!-- 动态加载 Bootstrap 样式 -->
    </head>
    <body>
    <main class="container">
        <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
            <div class="navbar-header">
                <a class="navbar-brand" href="#">Python100</a>
            </div>
            <button class="navbar-toggler" type="button" data-toggle="col
                    aria-controls="navbarSupportedContent" aria-expanded=
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarSupportedCont
                <ul class="navbar-nav mr-auto">
                    <!-- 利用宏 render_nav_item 创建菜单 -->
                    {{ render_nav_item('index', '首页', use_li=True) }}
                </ul>
            </div>
        </nav>
        {% block content %}  <!-- 定义内容块 -->
        <h1> Hello Flask Bootstrap! </h1>
        {% endblock %}
        <footer class="text-center">
            {% block footer %} <!-- 定义页脚块 -->
                <small> © 2019 <a href="http://justdopython.com" title="p
                </small>
            {% endblock %}
        </footer>
    </main>
```

```
    <!-- 动态加载 Bootstrap js 脚本 -->
    {{ bootstrap.load_js() }}
    </body>
</html>
```

{% endraw %}

是不是感觉很头大，不过是代码多了些，结构很其实很简单，大部分代码是 Bootstrap 提供的基本框架代码，然后加入了一些 `bootstrap-flask` 的扩展。我们来分析一下：

- 首先引入导航元素宏 `render_nav_item` ，因为在后面制作导航菜单时要用
- 然后是大段的 Bootstrap 框架代码
- 定义块，用来在继承 base.html 的子模板中做替换，其中有 标题、内容和页尾
- 动态加载 Bootstrap 样式和脚本，通过 `bootstrap.load_css()` 和 `bootstrap.load_js()`
- 在导航菜单的位置，使用宏 `render_nav_item` 创建一个首页菜单

这样就完成基础模板的定义，稍后会对 `render_nav_item` 、 `bootstrap.load_css()` 和 `bootstrap.load_js()` 做解释。

创建页面模板

有了基础模板，就可以做具体的页面模板了，先做首页 index.html，代码如下:

{% raw %}

```
{% extends "base.html" %} <!-- 继承基础模板 -->

{% block content %} <!-- 替换页面内容 -->
<h1> Hello Flask Bootstrap </h1>
{% endblock %}
```

{% endraw %}

是不是简洁多了？下面在 `flaskbootstrap.py` 中加上首页的视图函数:

```
@app.route('/')
def index():
    return render_template('index.html')
```

启动

如果一切顺利，访问 `localhost:5000` 就能看到如下效果:

首页效果

调整浏览器页面大小，可以看到页面自适应效果，是不是很神奇。接下来详细说明一下 `bootstrap-flask` 模块

资源助手

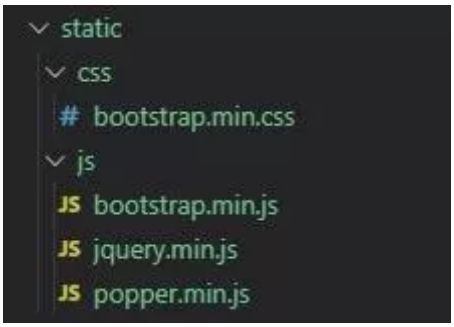
在小试牛刀中，能看到对 Bootstrap 样式和脚本引用的方法 `bootstrap.load_css()` 和 `bootstrap.load_js()` ，默认情况下会自动从 Bootstrap 的 CDN 上引用最新版的 Bootstrap 资源，例如: `https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/css/bootstrap.min.css` ，这样就方便了开发者对 Bootstrap 资源的引用和升级。

使用本地资源

如果想要从其他位置或者本地引用资源，只要在给应用初始化完 Bootstrap 之后，将应用的 `BOOTSTRAP_SERVE_LOCAL` 属性设置为 `True` 就好了:

```
app = Flask(__name__) # 创建一个 Flask 应用
bootstrap = Bootstrap(app) # 为应用初始化 bootstrap
app.config['BOOTSTRAP_SERVE_LOCAL'] = True  # 设置为使用本地资源
```

本地资源放在 `static` 目录下(这个目录是 Flask 框架默认的静态资源目录)，Bootstrap 本地资源放置如下:



Bootstrap 资源

bootstrap.load_css()

加载 Bootstrap 样式资源， 参数:

- `version` ，指定 Bootstrap 样式版本号，默认值为 4.3.1，本地资源无效

bootstrap.load_js()

加载 Bootstrap 脚本资源

参数：

- `version` ，指定 Bootstrap 脚本版本号，默认为 4.3.1，本地资源无效
- `jquery_version` ，指定 JQuery 版本号，默认为 3.3.1，本地资源无效
- `popper_version` ，指定 Popper 版本号，默认为 1.14.0，本地资源无效
- `with_jquery` ，是否加载 JQuery， 默认为 `True`
- `with_popper` ，是否加载 Popper, 默认为 `True`

宏

`bootstrap-flask` 主要的改进是加入了很多方法是用的宏，让对页面效果的编辑像写逻辑代码一样，并在宏中还对所处环境信息进行了判断，比如菜单是否要激活，分页控件动态效果等等，下面介绍几个典型的宏

render_nav_item()

生成一个导航菜单

参数:

- endpoint，路径点，可以直接写视图函数名
- text，标题

代码示例:

{% raw %}

```
{% from 'bootstrap/nav.html' import render_nav_item %}

<nav class="navbar navbar-expand-lg navbar-light bg-light">
    <div class="navbar-nav mr-auto">
        {{ render_nav_item('index', 'Home') }}
        {{ render_nav_item('explore', 'Explore') }}
        {{ render_nav_item('about', 'About') }}
    </div>
</nav>
```

{% endraw %}

render_breadcrumb_item()

面包屑导航条 参数:

- endpoint，路径点，可以直接写视图函数名
- text，标题

代码示例:

{% raw %}

```
{% from 'bootstrap/nav.html' import render_breadcrumb_item %}

<nav aria-label="breadcrumb">
    <ol class="breadcrumb">
        {{ render_breadcrumb_item('home', 'Home') }}
        {{ render_breadcrumb_item('users', 'Users') }}
        {{ render_breadcrumb_item('posts', 'Posts') }}
        {{ render_breadcrumb_item('comments', 'Comments') }}
    </ol>
</nav>
```

{% endraw %}

render_static()

静态资源引用，例如引用 css 、js 或者 图标

参数:

- type 资源类型，可以是 css 或 js 或 icon
- filename_or_url 资源路径，文件名 或者 参数 `local` 为 `False` 时的远程 url
- local 是否本地资源，默认为 `True`

代码示例:

{% raw %}

```
{% from 'bootstrap/utils.html' import render_static %}

```

```
{{ render_static('css', 'style.css') }}
```

{% endraw %}

其他宏

还有有些宏很有用，例如 表单(form)相关的，还有分页相关的，不过这些会涉及到其他 Flask 扩展模块，我们会在介绍 表单 和 数据库 的章节中做进一步学习，如果您有兴趣，可以浏览参考链接中的内容。

总结

这篇文章简单介绍了 Flask 框架中如何使用 Bootstrap 扩展，从一个简单的示例开始，讲解了基于 Jinja2 模板引擎的 `bootstrap-flask` 模块的使用，其中包括资源助手 `bootstrap.load_css()` 和 `bootstrap.load_css()` ，以及一些基本的宏的用法，在参考代码中有较为完整的例子，您可以作为参考。在后面的文章中将会陆续介绍 表单 和 数据库 的使用，敬请期待。

> 示例代码：Python-100-days-day044

参考

- bootstrap-flask 文档: https://bootstrap-flask.readthedocs.io/en/latest/basic.html
- bootstrap-flask 源码: https://github.com/greyli/bootstrap-flask
- Bootstrap: https://getbootstrap.com/
- bootstrap-flask 作者: http://greyli.com/

系列文章

第43天：Python filecmp&difflib模块

第42天：Python paramiko 模块

第41天：Python operator 模块

第0-40天：从0学习Python 0-40合集

PS：公号内回复 ：Python，即可进入Python 新手学习交流群，一起**100**天计划！

-END-

Python 技术

关于 Python 都在这里