Flask 使用手册

前　言

教程内容

本教程涵盖了如何使用 Flask 配置和管理，通俗易懂，图文并茂，所涉及到的知识点都通过示例讲解

目 录

第1章 介绍

1.1 Flask 是什么

Flask 是一个用 Python 编写的 Web 应用程序架构（一个库和模块的集合，使得 Web 应用程序开发人员能够编写应用程序，而不必担心协议、线程管理等低级细节），也被称为“microframework”，因为它使用简单的核心，用 extension 增加其他功能。flask 没有默认使用的数据库、窗体验证工具。

WSGI 应用程序：Web Server Gateway Interface（Web 服务器网关接口，WSGI）已被用作 Python Web 应用程序开发的标准。WSGI 是 Web 服务器和 Web 应用程序之间通用接口的规范。

1.2 安装

建议在 python2.7 以上安装 flask，为实现不同版本库之间的兼容性问题，可在当前目录下创建虚拟环境：

```
virtualenv venv
venv\scripts\activate
pip install Flask
```

第2章 Flask 应用

2.1 创建 Hello.py

```
from flask import Flask
app = Flask(__name__)//构造函数使用当前模块(__name__)的名称作为参数。

@app.route('/')//装饰器，指出 url 与调用函数的关联性,options：要转发到底层的 Werkzeug 服务器
def hello_world():
   return 'Hello World'

if __name__ == '__main__':
 app.run()//本地开发服务器上运行应用程序, app.run(host, port, debug, options)
```

host: 要监听的主机名。 默认为 127.0.0.1（localhost）。设置为“0.0.0.0”以使服务器在外部可用

port：默认 5000

debug：默认为 false，如果设置为 true 则提供调试信息，并能实时更新接口。

options：要转发到底层的 Werkzeug 服务器

2.2 动态增加路由

app.add_url_rule(‘/’ ,’ hello’ ,hello_word)

2.3 路由传参

```
@app.route('/hello/<name>')
def hello_name(name):
   return 'Hello %s!' % name
```

注：参数可指定类型，如@app.route('/blog/<int:postID>')，表明是传入参数必须是 int，也可以是 float path，后者主要是传递字符串。

2.4 Flask url 构建（后台 url 跳转）

```
app.route('/user/<name>')
def hello_user(name):
 if name =='admin':
 return redirect(url_for('hello_admin'))
else:
 return redirect(url_for('hello_guest',guest = name))
```

第3章 HTTP 方法

3.1 GET 方法

以未加密的形式将数据发送到服务器。获取参数方式是 request.args.get 方式得到。

3.2 POST 方法

用于将 HTML 表单数据发送到服务器。POST 方法接收的数据不由服务器缓存。

3.3 HEAD

和 GET 方法相同，但没有响应体。

3.4 PUT

用上传的内容替换目标资源的所有当前表示。

3.5 DELETE

删除由 URL 给出的目标资源的所有当前表示。

```
@app.route('/login',methods = ['POST', 'GET'])
def login():
   if request.method == 'POST':
      user = request.form['nm']
      return redirect(url_for('success',name = user))
else:
      user = request.args.get('nm')
      return redirect(url_for('success',name = user))
```

第4章 Flask 模板

第5章 vue 风格指南

5.1 A 必要建议

1 组件名称尽量包含-，不能是单个单词。

2 组件的 data 必须是一个函数。

3 Prop 定义应该尽量详细。

```
props: {
  status: {
    type: String,
    required: true,
    validator: function (value) {
      return [
        'syncing',
        'synced',
        'version-conflict',
        'error'
      ].indexOf(value) !== -1
    }
  }
}
```

4　为 v-for 设置键值 key

5　避免 v-if 和 v-for 用在同一个元素上

为了过滤一个列表中的项目（比如 v-for="user in users" v-if="user.isActive"）。在这种情形下，请将 users 替换为一个计算属性（比如 activeUsers），让其返回过滤后的列表。

为了避免渲染本应该被隐藏的列表（比如 v-for="user in users" v-if="shouldShowUsers"）。这种情形下，请将 v-if 移动至容器元素上（比如 ul、ol）。

6　为组件样式设置作用域

倾向于使用 class 的 Modeles 策略而不是 scoped attribute。

7　私有 property 名

使用模块作用域保持不允许外部访问的函数的私有性。如果无法做到这一点，就始终为插件、混入等不考虑作为对外公共 API 的自定义私有 property 使用 $_ 前缀。并附带一个命名空间以回避和其它作者的冲突（比如 $_yourPluginName_）。