**服务器：Flask 路由。**

课前作业，第5A节（约2小时50分）

视频可以在学习中心中找到此模块下的> Learn Plus 录制内容 与此内容关联的示例文件可在本节的 .zip 文件夹中找到。

内容

[初始简要阅读 （5m） 1](#_bookmark0)

[会议介绍 1](#_bookmark1)

[简介（10分钟） 2](#_bookmark2)

[The Flask framework （10分钟） 2](#_bookmark3)

[运动（15分钟） 3](#_bookmark4)

[路由导致代码的执行 （ 20分钟 ） 4](#_bookmark5)

[运动 （30m） 5](#_bookmark6)

[处理表单 GET（25m） 6](#_bookmark7)

[处理表单 POST（20 分钟） 7](#_bookmark8)

[Doing something with the data （5分钟） 8](#_bookmark9)

[运动 （20m） 8](#_bookmark10)

[可能破坏事物的问题 8](#_bookmark11)

[额外阅读 9](#_bookmark12)

[附录 1：如何启动 python flask 服务器 9](#_bookmark13)

## 初始简要阅读（5m）：

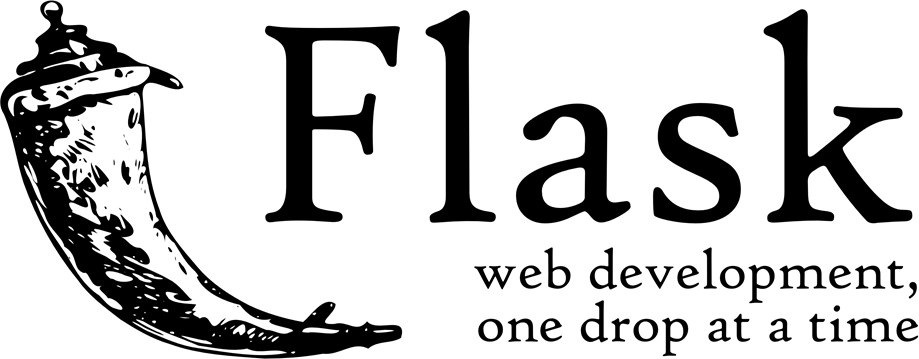
简要熟悉一下 Flask 文档。 <http://flask.pocoo.org/> （这将重定向到新站点 <https://palletsprojects.com/p/flask/> ），即查看索引，并查看“快速入门”5分钟

## 会议介绍

### 关联的视频5B\_ServerFlask.mp4

此会话将允许您创建一个服务器，该服务器将响应不同的 URL 请求、不同的消息类型并提取通过表单发送的数据。它是对 Flask 服务器框架的介绍。

## 简介（10分钟）

什么是烧瓶？

* Flask 是一个基于 python 的微框架
* 一个简单易用的服务器，涵盖了我们需要涵盖的基本原则。
* 这些原则与其他框架非常相似，例如

nodeJS 或 Django 甚至 Laravel。

* <https://palletsprojects.com/p/flask/>

在之前的会话中，我们看到了几个服务器（0\_1\_pyserver 和 1\_StaticFlaskServer），一个是用 python 编写的，它只提供单个资源，并且需要大量代码，第二个使用框架并启动服务器以提供对静态文件夹中任何资源的访问。

关键是服务器框架可以通过一些标准的高效编码做很多事情。当服务器收到请求时，它将处理 URL 并提取请求的路由。这是资源的方向。在 flask 中，默认情况下，任何以 /static/ 开头的路由都将从文件夹 static 中提供资源。例如 http://localhost:5000/static/hello.html。

回顾：启动烧瓶服务器

创建一个虚拟环境 （pipenv shell –python 3.7） 或 8 或 ？如果有 pip lock 文件，则加载依赖项 （pipenv install）。

运行服务器 （py <Server>.py

## The Flask framework （10分钟）

与视频一起阅读 **5B\_ServerFlask.mp4** 要设置 Flask Web 应用程序框架，我们需要编写几行代码。

#server.py

from flask import Flask app = Flask( name )

…

if name == " main ": app.run(debug=True)

第一行是注释，说明文件已 server.py，不需要它，但有助于会话。然后，我们导入 Flask 库代码。

我们需要使用 flask 模块中的库代码创建我们的 flask 应用程序，因此我们调用构造函数来创建一个 Flask 对象。 **app = Flask（ 名称 )**

在此之后，我们通常会放置创建定制服务器的代码。

接下来，我们需要运行服务器 - 要运行应用程序，我们可以： 从代码运行应用：

**如果 名字 == " 主要 ":** 是关于从其他模块导入。如果此文件（server.py）是从另一个模块导入的，则未正确设置

并且可能有另一台服务器在运行，所以不要运行。这可以在计算思维模块中介绍 - 现在只需使用文档中的这些行。

**app.run（debug=True）** 从启动服务器的 app 对象调用 run 方法。现在我们只需要运行代码 – **py server.py**

或者使用 python 的 flask 命令：

为此，我们需要在 Windows cmd 提示符下告诉系统哪个文件包含应用程序，这是通过**设置 FLASK\_APP=server**  来完成的.py然后我们可以使用命令 **flask run 运行应用程序**

使用此方法，我们不需要服务器文件中的最后 2 行。 **如果名称** **== “主要”：** 和 **app.run（debug=True）**

这些方法将允许服务器在机器上进行内部侦听（localhost 或 IP 127.0.0.1） 要允许外部连接，您需要指定 host = 0.0.0.0

在代码中，这意味着：

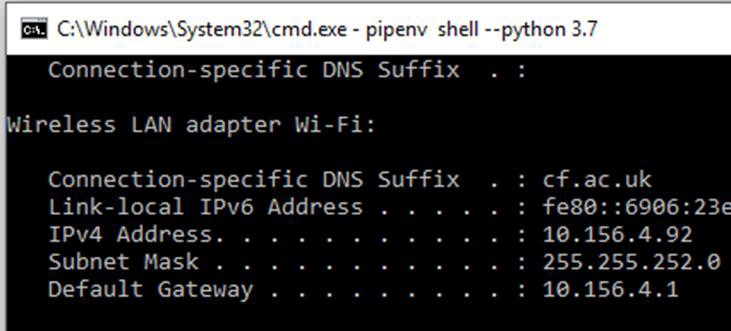
app.run(host='0.0.0.0', port=5000)

并使用 flask 命令，这意味着添加一个开关来让 flask **运行 --host=0.0.0.0**

Addition:

Default folder for static files is static, but this can be changed: app = Flask( name , static\_folder = 'web')

## 运动（15分钟）

* 设置1\_StaticFlaskServer.py示例
* 添加一个“静态”目录（如果还没有）
* 将一些页面添加到 /static
* 从计算机上的客户端请求页面 127.0.0.1：5000/static/?????
* 使用局域网（另一台电脑或手机）上的客户端请求页面

打开一个新的 cmd 窗口并键入

**ipconfig**

您的 IP 地址 

注意：默认情况下，Flask 在端口 5000 上提供服务

在此示例中，使用 10.156.4.92：5000/static/??????

**完成后关闭服务器。 （在命令窗口中按 Ctrl +C）**

## 路由导致代码的执行 （ 20分钟 ）

到目前为止，我们已经设置了一个服务器，该服务器仅提供来自指定目录的静态页面。帧工作分析了路由，例如 **/static/hello.html**，这导致服务器执行检索 hello.html 文件的代码并发送包含文件内容的响应。

但这是对服务器的非常有限的使用。服务器框架可以侦听指定 /static 目录中的路由以外的其他路由的请求。这些路由可用于触发自定义代码。

您可以在 flask 框架中创建自己的路由，方法是指定路由并创建一个函数，该函数将在请求路由时运行。

@app.route("/hello") def myHello():

return "Hello World!"

*@ Annotation：调用框架代码，将函数 myHello 设置为服务器收到路由 /hello 请求时要调用的函数*

查看服务器 2\_ **的服务器代码**  FlaskServer.py 确保 /hello 路由未注释并**运行服务器**。浏览到 **localhost：5000/hello**，您应该会看到一条响应消息。这不是 HTML，但浏览器仍会显示它。此外，该代码还会将消息打印到服务器控制台。

所以。。。路由导致代码的执行 – 这可能是返回文件内容的代码、返回数据的代码或触发操作（启动电机以关闭窗帘）的代码。

我们开始了解应用程序编程接口（API），它是一组具有指定操作的路由。

重定向

from flask import Flask, **redirect**

**…**

@app.route("/redirect") def redirectToStatic():

print(“do some thing here”)

return **redirect**("/static/hello.html")

我们可能希望执行代码，然后返回文件的内容。在这种情况下，按照我们希望执行的代码，我们可以使用一个函数重定向到返回文件内容的框架代码。

将其添加到服务器文件（或取消注释），然后重新启动服务器以尝试新路由。

URL 变量

有时，我们希望将 URL 的一部分用作路由，将部分 URL 用作路由中的变量。即在同一路由代码中处理许多不同的 URL。

例如，如果我想访问资源（工作人员的数据）“Ian”，也许我可以写一个路由 /staff/Ian，但是 /staff/Alexia 和 /staff/Wendy 和 /staff/Louise 呢？我不想重写所有这些路线。是的

@app.route("/Staff/**<person>**", methods=['GET']) def returnPerson(**person=None**):

if request.method == 'GET':

print("getting person "+ staff[**person**])

…

最好写路线

/staff/<person>并使用 person 作为路由代码中的变量。

URL 变量在尖括号（或胡萝卜）中，我们可以

为函数参数中的变量设置默认值（例如 **person=None**）。

如果 URL 变量是可选的，那么我们可以使用多个路由，并为可能不存在的值设置默认值。

@app.route("/pathVars/<name")

@app.route("/pathVars/<name>/<surname>") def pathVars(name, surname="Doe"):

**将** pathVars 路由添加到服务器文件（或取消注释），然后重新启动服务器。尝试一些适用的 URL。

## 运动 （30m）

打开2\_FlaskServer.py并完成文件底部的练习。

1. 添加新路由：/goodbye
   * 返回文本“goodbye”
2. 添加路由：/goodbye/<name>
   * 返回“再见伊恩” 其中 Ian 是 url 中的名称。
3. 添加新路由：/time
   * 返回当前时间 - 提示：导入日期时间

current\_time = datetime.datetime.now（） current\_time.isoformat（）

### 额外：

1. 添加新路由：/hitCounter
   * 返回“Hello World！- 从 Flask 服务器 1 开始，您是访客编号。[5]"
   * 提示：

使用全局变量。

1. 添加新路由以将用户的 IP 地址打印到服务器控制台， **并** 重定向到您选择的静态页面。
   * 提示：

从 Flask 导入请求打印（request.environ['REMOTE\_ADDR']）

# 处理表单 GET（25m）：

你还记得你的表格吗......我们忽略了一个 action 属性。看

### static/form.html

<form action=" /**HomeForm**" method= "get" id="myForm" name="myForm">

<label>first name:

<input type = "text" **name = "firstname"**><br>

</label>

…

</form>

如果您 **浏览** 到表单页面，填写表单并单击提交，您将获得如下内容： http://localhost:5000/HomeForm?firstname=Ian&text=werwerwe

在 URL 栏中。？是表单数据。（并且可能是由于表单服务器未运行而出错）

我们需要添加代码来处理服务器中的表单。该操作（在 HTML 文件中）指向

/HomeForm，因此浏览器会创建对 /HomeForm 的请求，因此我们需要为

/HomeForm。在这个路由中，我们需要指定我们将接受哪些请求方法（例如 GET、POST 或 PUT 等），然后从请求中提取表单数据，以便我们可以在路由代码中使用它。

from flask import Flask, redirect, **request**

**…**

@app.route("/**HomeForm**", methods=[‘GET']) def form():

if **request.method** == 'GET':

print(**request.args.get**(**'firstname**')) return "Hello World!"

我们可以导入更多的库功能（请求），这是 flask 服务器填充并使其可访问的对象

到路由的代码。如果请求方法是 GET 消息（我们通过检查请求对象的方法变量 - if **request.method** == 'GET'： 来发现这一点），我们可以使用请求对象的 args 对象中的方法来提取特定的请求参数

**request.args.get**（**'名字'**）。

查看3\_FormServer.py运行此服务器，然后重复请求/填写/提交表单的过程。查看控制台以查看打印出来的名字。

### 快速练习：

1. 向 form.html **添加姓氏输入字段**
2. 添加到 **3\_FormServer.py** 以在通过 GET 请求提交表单时将姓氏打印到服务器控制台。

### 另一个快速练习：

记住表格.....

* + 安装“Screenreader for Google Chrome”
  + <https://chrome.google.com/webstore/category/extensions>
  + 搜索适用于 Chrome 的屏幕阅读器
  + 添加到 chrome
* 通过单击文本的不同部分来试用屏幕阅读器
  + 这只能从提供的页面工作，而不是文件！
* 将标签标签换成 div 标签。屏幕阅读器会发生什么情况？
* 删除 div 标签。屏幕阅读器会发生什么情况？

# 处理表单 POST（20m）：

发出请求的另一种方式是发送 POST 消息。发布消息在消息内容中发送表单数据，而不是在URL中发送表单数据。因此，发布到 /HomeForm 的 URL 将如下所示 http://127.0.0.1:5000/HomeForm。HTML 表单中的唯一区别是 method 属性需要是 post。

<form action=" /**HomeForm**" method= "**post**" id="myForm" name="myForm">

<label>first name:

<input type = "text" **name = "firstname"**><br>

</label>

…

</form>

在服务器文件路由代码中，我们可以将 GET 更改为 POST，其中我们指定要接受的方法或添加处理 ['GET'， 'POST'] 的能力。为了从表单中检索输入参数，我们使用了不同的技术。该请求具有带有键值对的表单数据结构（字典），输入名称用作键。我们使用此 **request.form['firstname'] 访问表单数据**。

@app.route("/HomeForm", methods=['POST'])

def form():

if request.method == **'POST**':

print(**request.form['firstname']**) return "Hello World!"

更改表单 .html，使其使用 post 方法提交和重新提交表单。检查消息以查看使用网络选项卡提交的参数（来自客户端服务器会话）。

### 快速练习：

添加到 **3\_FormServer.py** 将姓氏打印到

服务器控制台，当表单通过 **POST** 请求提交时。

提示：如果您尚未更改**表单.html**，您还需要这样做。

# 对数据（5m）做一些事情。

names = [] surnames = []

…

…

if request.method == 'POST':

name = request.form['firstname'] surname = request.form['lastname'] names.append(name)

surnames.append(surname)

如前所述，我们现在可以对数据做任何事情，所以这里给出了一个简单的例子。（存储在列表中）——参见 （4\_FormServer.py）

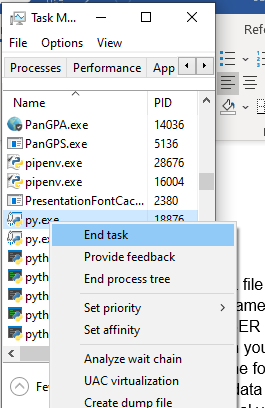
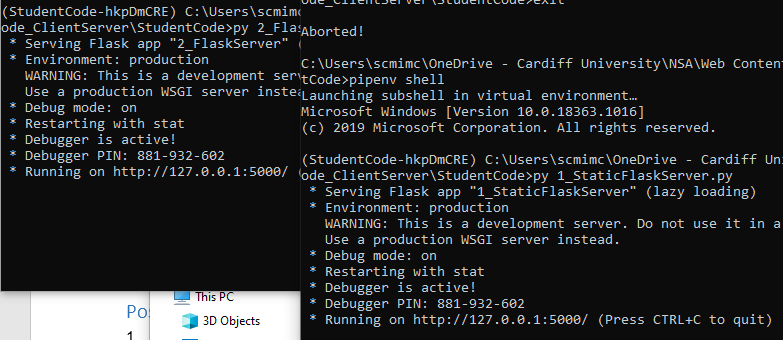
# 运动 （20m）

### 个人资料收集

1. 创建一个带有表单元素的 HTML 文件。
   * 收集：姓名、出生日期、电子邮件地址和邮政地址。
   * 记住 - 将其放在静态文件夹中
2. 在服务器中创建路由以处理表单：
   * 在服务器控制台上打印所有表单数据。
3. 使用 python 数据结构，存储每次提交的所有表单数据。
   * 提示：全局变量
   * 创建另一个路由以将所有存储的数据打印到服务器控制台

# 可能破坏事物的问题：

1. **错误的**服务器 – 您没有运行您认为的服务器 – 检查所有命令窗口并关闭所有服务器（ctrl +c 如果一开始失败，请继续按下）



2. **2 servers running** – if you do not terminate the server properly a command window may shut and leave the server running.

第一台服务器将被分配端口 5000，因此第二台服务器无用。您可能需要查看任务管理器以结束任何其他 python 进程。

1. **错误的路线** – 检查您请求的路线是否正确。查看检查工具网络分流器进行验证。
2. **虚拟环境中的虚拟环境** – 如果您在包含 pip 文件的目录中设置了 pip 文件，则虚拟环境可能会变得非常混乱。要小心。您可以删除 pipfile 和 pipfile.lock，但需要重新启动虚拟环境并手动重新安装依赖项。
3. **当它仍然无法正常工作时** ，请关闭电脑并重新启动 - 可能还有其他服务器正在运行，您找不到。

## 额外阅读：

阅读 <https://termly.io/resources/articles/gdpr-for-dummies/> GDPR 将在该计划中详细介绍，但这只是一个开端。

# 附录 1：如何启动 python flask 服务器。

* + 在会话文件目录（包含服务器的目录）中打开命令提示符，
  + 启动虚拟环境（**pipenv shell**），
  + 安装依赖项 （**pipenv** install） 如果这不起作用，请尝试 **pipenv install flask**
  + 运行 Python 服务器文件，例如 server.py （**py server.py**）
  + 在 Flask 中，静态文件是来自静态目录的服务器。因此，要加载静态 hello.html 页面，请转到浏览器并键入 **http://localhost:5000/static/hello.html**
  + 一切顺利，您应该会看到您的页面。

### 使用 ctrl +c 终止此程序并关闭 cmd 窗口。

附录 2：常见问题

**问题）** 服务器似乎没有更新 - 似乎正在运行与您更改、保存并重新启动服务器之前相同的代码。

**解决方案）** 最可能的原因是您没有正确关闭以前的服务器，并且端口 （5000） 仍与以前运行的服务器实例相关联。

要解决此问题，您需要完全关闭所有服务器实例。最简单的方法是打开任务管理器，单击详细信息选项卡，按名称对进程进行排序，然后结束所有 python 进程。（右键单击“处理”>结束任务）。

