浙江理工大学
ZHEJIANG SCI-TECH UNIVERSITY

2. Flask与HTTP

《Web应用开发》 / 任课教师：罗志一

计算机科学与技术学院

概览 Overview

- 注册路由
- HTTP
- 数据爬取Tips

注册路由

- 为视图绑定多个URL

```
@app.route("/hi")
@app.route("/hello")
def say_hello():
    return "<h1>Hello, World!</h1>"
```

注册路由

- 为视图绑定多个URL

Hello, World!

http://127.0.0.1:5000/hello

Hello, World!

URL

http://127.0.0.1:5000/hi

注册路由：动态URL

- 在URL规则中添加变量部分，使用<变量名>的形式表示
  - Flask处理请求时会把变量传入视图函数，所以我们可以添加参数获取这个变量值

URL规则

```
@app.route("/greet/<name>")
def greet(name):
    return f"<h1>Hello, {name}!</h1>"
```

- 因为URL中可以包含变量，所以我们将传入app.route()的字符串称为URL规则，而不是URL。Flask会解析请求并把请求的URL与视图函数的URL规则进行匹配。

注册路由：动态URL

- 在URL规则中添加变量部分，使用<变量名>的形式表示
  - Flask处理请求时会把变量传入视图函数，所以我们可以添加参数获取这个变量值

变量

```
@app.route("/greet/<name>")
def greet(name):
    return f"<h1>Hello, {name}!</h1>"
```

```
def greet(other):
```

```
TypeError
TypeError: greet() got an unexpected
keyword argument 'name'
```

注册路由：动态URL

- 在URL规则中添加变量部分，使用<变量名>的形式表示
  - Flask处理请求时会把变量传入视图函数，所以我们可以添加参数获取这个变量值

```
@app.route("/greet/<name>")
def greet(name):
    return f"<h1>Hello, {name}!</h1>"
```

Not Found

The requested URL was not found on the server. If you entered the URL manually please check your spelling and try again.

如果用户访问的URL中没有添加变量，那么Flask匹配失败后会返回一个404错误相应

注册路由：动态URL

- 在URL规则中添加变量部分，使用<变量名>的形式表示
  - Flask处理请求时会把变量传入视图函数，所以我们可以添加参数获取这个变量值

```
@app.route(“/greet”, defaults={'name': 'Programmer'})
@app.route("/greet/<name>")
def greet(name):
    return f"<h1>Hello, {name}!</h1>"
```

可以在app.route()装饰器里使用defaults参数设置URL变量的默认值，这个参数接受字典作为输入，存储URL变量和默认值的映射。

注册路由：动态URL

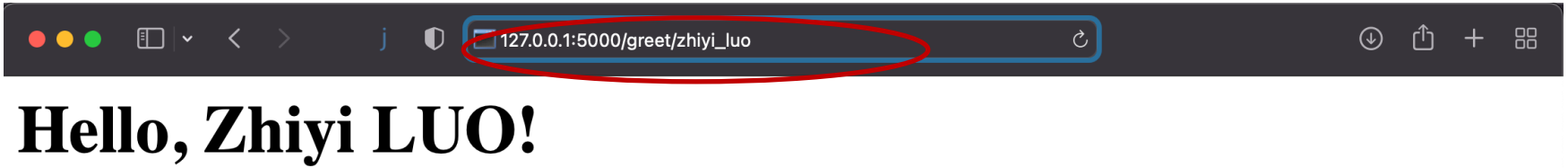
- 在URL规则中添加变量部分，使用<变量名>的形式表示
  - Flask处理请求时会把变量传入视图函数，所以我们可以添加参数获取这个变量值

```
@app.route("/greeting/<first_name>_<last_name>")
def greeting(first_name, last_name):
    first_name = first_name[0].upper() + first_name[1:]
    last_name = last_name.upper()
    return f"<h1>Hello, {first_name} {last_name}!</h1>"
```

注册路由：动态URL

- 在URL规则中添加变量部分，使用<变量名>的形式表示
  - Flask处理请求时会把变量传入视图函数，所以我们可以添加参数获取这个变量值

```
@app.route("/greet/<first_name>_<last_name>")
```



项目配置

- Flask项目的配置，都是通过app.config对象来进行配置的。
- 内置配置项可以在这里查看
  - https://flask.palletsprojects.com/en/2.2.x/config/
- 重载器
  - 当我们对代码做了修改后，期望的行为是这些改动立刻作用到程序上。重载器的作用就是检测文件变动，然后重新启动开发服务器。

构造URL

- Web程序中，URL无处不在
  - 某个路由的URL规则可能会经常发生变化
  - 将URL硬编码在代码中会降低代码的易用性和可扩展性
- 调用url_for()函数获取URL
  - 第一个参数为端点（endpoint）值，端点的默认值为视图函数的名称
  - 如果URL含有动态部分，则需要在url_for()函数中传入相应的参数
- 端点（endpoint）
  - 在Flask中，用端点来标记一个视图函数以及对应的URL规则
  - 端点的默认值为视图函数的名称
  - 为什么不直接使用视图函数名，而引入端点这个概念？

构造URL

- 调用url_for()函数获取URL
  - 第一个参数为端点（endpoint）值，端点的默认值为视图函数的名称
  - 如果URL含有动态部分，则需要在url_for()函数中传入相应的参数

```
@app.route("/greet/<name>")
def greet(name):
    return f"<h1>Hello, {name}!</h1>"
```

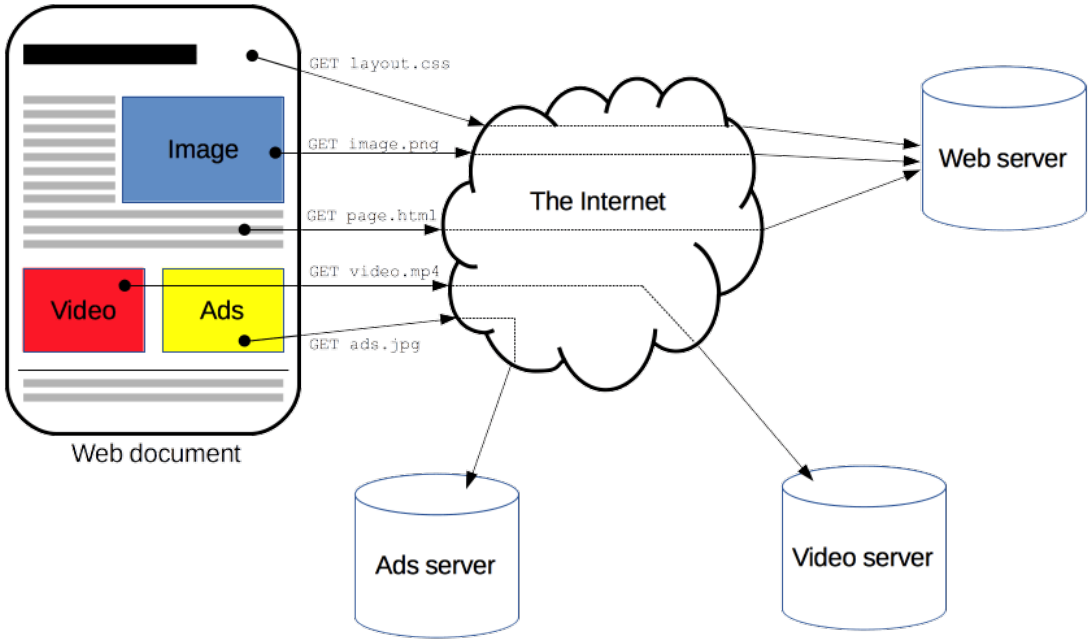
```
from flask import url_for
url_for('greet', name='Jack')
```

HTTP概述

- HTTP(Hypertext Transfer Protocal)定义了服务器和客户端之间信息交流的格式和传递方式，它是万维网中数据交换的基础。
- 通过HTTP协议获取如HTML这样的网络资源。
- 我们已经了解了Flask的基本知识，如果想要进一步开发更复杂的Flask应用，我们就得了解Flask与HTTP协议的交互方式。

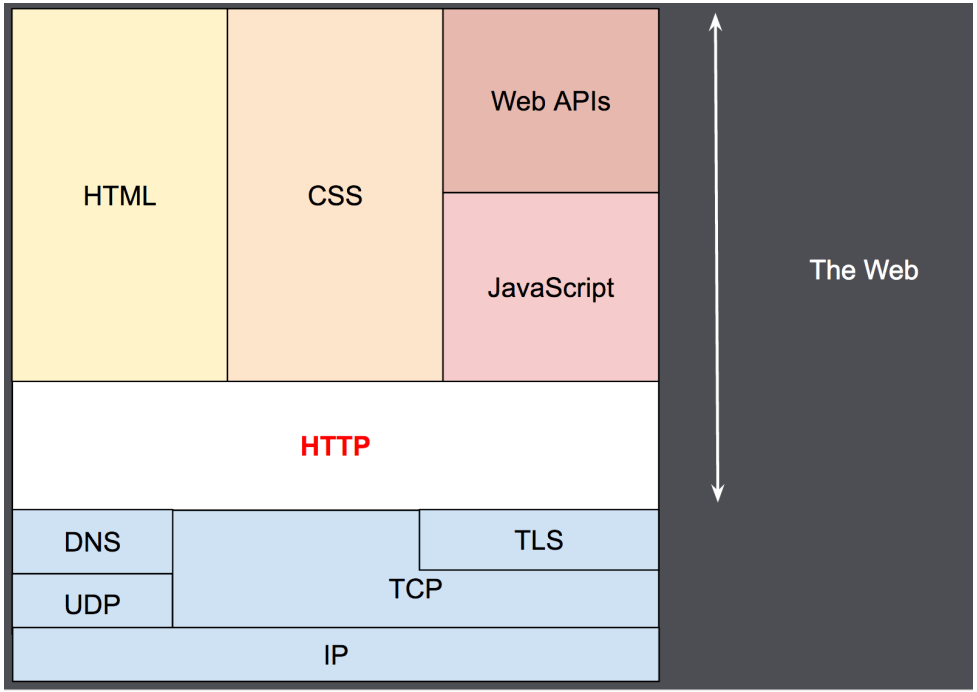
HTTP概述

- 一个完整的Web文档通常是由不同的子文档拼接而成的，像是文本、布局描述、图片、视频、脚本等。



HTTP概述

- 客户端和服务端通过交换各自的消息进行交互。像浏览器这样的客户端发出的消息叫做request，被服务端响应的消息叫做response。



HTTP请求

- URL是一个请求的起源

  https://baike.baidu.com/item/李白/1043

- 什么是查询字符串？

  https://www.gaokao.cn/school/243?fromcoop=bdkp

- 请求报文

Flask的Request对象

- Flask的Request对象封装了从客户端发来的请求报文，我们能从它获取请求报文中的所有数据。

```
from flask import Flask, request
app = Flask(__name_)

@app.route("/hello")
def hello():
    name = request.args.get(“name”, “Flask”)
    last_name = last_name.upper()
    return f"<h1>Hello, {name}!</h1>"
```

HTTP响应

- 在Flask程序中，客户端发出的请求触发相应的视图函数，获取返回值会作为响应的主体，最后生成完整的响应，即响应报文。
- 响应报文主要由协议版本、状态码、原因短语、**响应首部**和响应主体组成。响应报文的首部包含一些关于响应和服务器的信息，这些内容由Flask生成，而我们在视图函数中返回的内容即为响应报文中的主体内容。浏览器接收到响应后，会把返回的**响应主体**解析并显式在浏览器窗口上。
- 常见状态码：200，302，404

在Flask中生成响应

- 响应在Flask中使用Response对象表示，响应报文中的大部分内容由服务器处理，大多数情况下我们只负责返回主体内容。
- 完整地说，视图函数可以返回最多由三个元素组成的元组：响应主体、状态码、首部字段，其中首部字段可以为字典，或是量元素组成的列表。

```
@app.route("/hello")
def hello():
    return “”, 302, {“Location”: “https://www.baidu.com”}
```

- Flask会调用make_response方法将视图函数返回值转换为响应对象。

Flask与HTTP

- 当用户访问一个URL，浏览器便生成对应的HTTP请求，经由互联网发送到对应的Web服务器。Web服务器接收请求，通过WSGI将HTTP格式的请求数据转换成我们的Flask程序能够使用的Python数据。
- 在程序中，Flask根据请求的URL执行对应的视图函数，获取返回值生成响应。
- 响应依次经过WSGI转换成HTTP响应，再经由Web服务器传递，最终被发出请求的客户端接收。
- 浏览器渲染响应中包含的HTML和CSS代码，并执行JavaScript代码，最终把解析后的页面呈现在用户浏览器的窗口中。

数据爬取Tips

爬取百度知道会遇到的问题

- IP被封

爬取百度知道会遇到的问题

- IP被封

爬取百度知道会遇到的问题

- 百度安全验证

爬取百度知道会遇到的问题

- 百度安全验证

https://wappass.baidu.com/static/captcha/tuxing.html?ak=1094e769f295bafeea0d4b33108991d1&backurl=https%3A%2F%2Fzhidao.baidu.com%2Fsearch%3Fct%3D17%26pn%3D0%26tn%3Dikaslist%26rn%3D10%26fr%3Dwwwt%26ie%3Dutf-8%26dyTabStr%3DMCwzLDYsMSw1LDQsNywyLDgsOQ%253D%253D%26word%3D%25E6%259D%258E%25E7%2599%25BD&timestamp=1658721973&signature=7a27bc66d83a72c65ef3307aa0d89dcb

解决方案提示

- Selenium模拟浏览器 + sleep 模拟用户浏览网页
  - chromedriver配置
- 提速
  - 换IP（代理池）+ 单线程 + 划分数据，开多个实例
  - 换IP（代理池）+ 多线程、异步