

# Python03

## 作业

```
# 创建商品表
from mysql.connector import connect

link = connect(user='root', passwd="", database="web2010")

print(link)

sql = '''
CREATE TABLE IF NOT EXISTS product(
    id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL COMMENT '名称',
    price DECIMAL(12,2) NOT NULL COMMENT '单价',
    count INT NOT NULL COMMENT '数量',
    phone CHAR(11) NOT NULL UNIQUE COMMENT '手机号',
    email VARCHAR(255) NOT NULL UNIQUE COMMENT '邮箱',
    address VARCHAR(255) NOT NULL COMMENT '地址'
) CHARSET=utf8
'''

link.cursor().execute(sql)
```

```
# 增
import random
from mysql.connector import connect
from random import randint

link = connect(user='root', passwd="", database='web2010')

sql = 'INSERT INTO product(name,price,count,phone,email,address)
VALUES(%s,%s,%s,%s,%s,%s)'

name = 'iPhone%s' % randint(4, 30)
price = '%s.%s' % (randint(4e3, 2e4), randint(10, 99))
count = randint(0, 100)
phone = '188%s' % randint(1e7, 9e7)
email = '%s@qq.com' % randint(1e5, 9e10)
address = '北京市%s区%s路%s街%s号' % (randint(1, 100), randint(
    1, 100), randint(1, 100), randint(1, 100))

args = (name, price, count, phone, email, address)

link.cursor().execute(sql, args)
# 增删改:都会变动数据, 必须二次确认
link.commit()
```

```
# 改
from mysql.connector import connect

link = connect(database='web2010', user='root', passwd="")

name = '然然的裹脚布'

sql = 'UPDATE product SET name=%s WHERE price > 6000 and price < 8000'

# 参数就算只有一个, 也必须是元组类型. 且元组类型至少一个逗号
link.cursor().execute(sql, (name,))

link.commit()
```

```
# 查

from mysql.connector import connect

link = connect(database='web2010', user='root', passwd="")

sql = 'SELECT * FROM product'

cur = link.cursor(dictionary=True)
cur.execute(sql)

res = cur.fetchall()

print(res)
```

## 服务器

类似于 node.js 有 express 模块, 用来实现 web服务器

python 有 django 模块, 用于实现 web服务器

```
pip install django
```

安装完毕后, 使用 `pip list` 可以查看已安装模块列表

```
E:\WEB2010\Python\Day03\src>pip list
Package            Version
-----
asgiref            3.3.1
autopep8           1.5.6
Django             3.1.7
mysql-connector    2.2.9
pip               20.1.1
python-dotenv      0.14.0
```

生成项目包： 在哪里执行cmd 就在哪里生成

```
django-admin startproject 包名
```

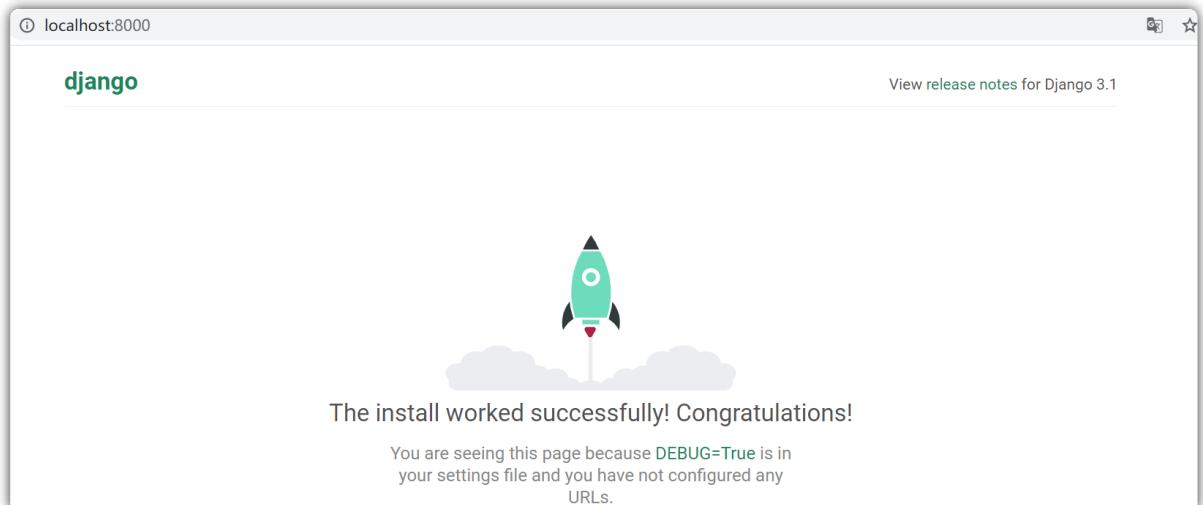
例如： `django-admin startproject djpro`

到项目下执行，启动命令： `py manage.py runserver` ，启动之后，访问： `localhost:8000`

```
E:\WEB2010\Python\Day03\djpro>py manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly until you apply the
migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
March 24, 2021 - 10:09:58
Django version 3.1.7, using settings 'djpro.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```



## 路由的执行

当在浏览器中，输入 `localhost:8000` 发生了什么？

- 到 `urls.py` 文件中，查找 路由的对应函数

```
urlpatterns = [
    path('admin/', admin.site.urls),
    # localhost:8000 对应 index 函数
    path('', index),
]
```

- 然后调用 对应的函数，把函数返回值 显示到页面上

```
def index(req):
    # req: 必带参数, 其中存放了 请求相关的信息, 例如参数.
    # HttpResponse: 用于反馈文本信息的类
    return HttpResponse('<h1>Hello World!!!!</h1>')
```

## JSON数据反馈

JSON: **JavaScript Object Notation** JS对象简谱.

JSON是一款轻量级的 在**不同编程语言之间交互**数据的 数据格式

例子:

如果一个屋子里, 有 韩国人 中国人 日本人 欧洲人 美国人... 要聊天!

找一个最简单的语言: 英语. 每个人都学会英语, 就可以互相交流

生成项目包: **django-admin startproject djjson**

服务器不能重复启动, 启动新的之前, 要关闭之前的服务器

```
py manage.py runserver
```

```
# JsonResponse: 负责把 数组/字典 类型, 转化成 JSON格式的字符串 进行反馈
from django.http import HttpResponse, JsonResponse

def index(req):
    # 字典 or 数组 类型, 都需要转为 字符串 才能返回
    data = {
        'title': 'Hello World!',
        'name': '东东',
        'skills': ['vue', 'vuex', 'json', 'elementUI', 'mintUI', 'echarts']
    }

    # return HttpResponse('<h1>Hello World</h1>')
    # 推荐使用 火狐 浏览器, 自带json格式化工具
    return JsonResponse(data)
```

## vue

利用 vue 和 接口服务器 进行联合开发

- vue: 制作前端
- django: 制作后端

实现 **前后端分离** 开发过程

检查是否安装了脚手架: **vue -V**, 最新版本 **@vue/cli 4.5.12**

如果 版本不对 或者 没有安装: `npm i -g @vue/cli --force`

```
+ @vue/cli@4.5.12
added 11 packages from 4 contributors, removed 499 packages, updated 91 packages and moved
8 packages in 231.315s
```

生成项目包: `vue create 包名`

例如: `vue create vuepro`

`空格` 选择, `回车` 确认, `方向键` 移动选项

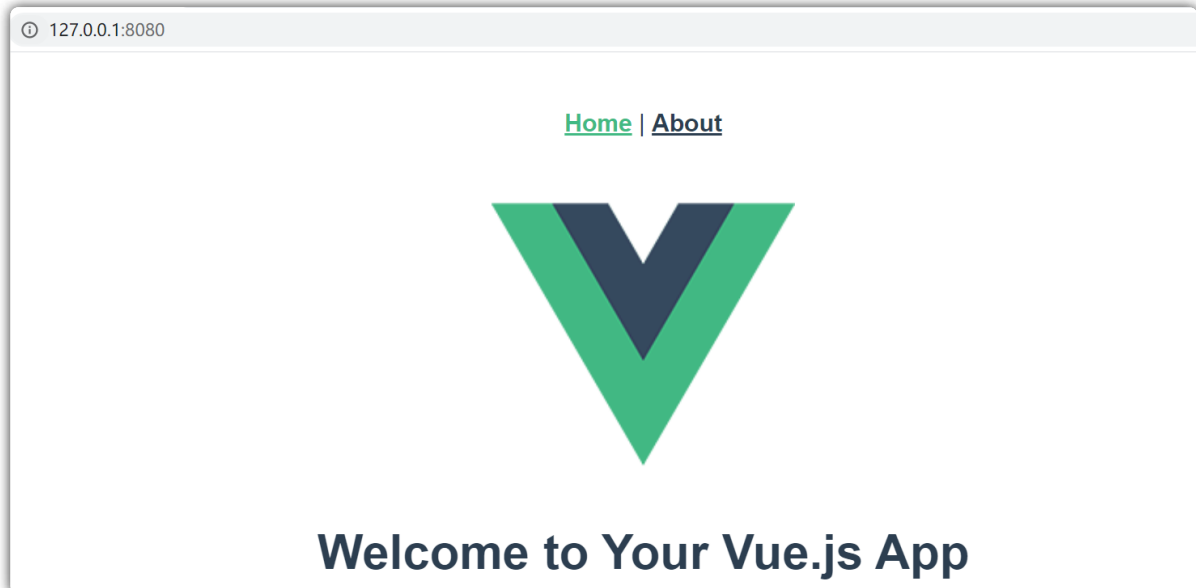
```
Vue CLI v4.5.12
? Please pick a preset: Manually select features
? Check the features needed for your project: Babel, Router, Vuex, CSS Pre-processors
? Use history mode for router? (Requires proper server setup for index fallback in production) Yes
? Pick a CSS pre-processor (PostCSS, Autoprefixer and CSS Modules are supported by default): Sass/SCSS (with dart-sass)
? Where do you prefer placing config for Babel, ESLint, etc.? In dedicated config files
? Save this as a preset for future projects? No
```

加装 axios 模块: `项目目录下执行`



```
npm i axios vue-axios
```

```
+ axios@0.21.1
+ vue-axios@3.2.4
added 2 packages from 2 contributors in 7.227s
```

启动命令: `npm run serve` 然后访问: `localhost:8080`



插件

|                                                                                     |                                                                                              |            |    |
|-------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------|------------|----|
|  | <b>Vetur</b> 0.33.1<br>Vue tooling for VS Code<br>Pine Wu                                    | 6.9M ★ 4.5 | ⚙️ |
|  | <b>Vue 3 Snippets</b> 1.0.4<br>A Vue.js 3 And Vue.js 2 Code Snippets Extension<br>hollowtree | 1.6M ★ 5   | ⚙️ |

## 前后端分离项目

早期的网站开发没有专门的 **前端开发** 分支。

都是服务器开发：流行 前后端在一起的项目：以 php 和 java 为主。

服务器人员 又做服务器开发 同时 兼顾 前端开发。

随着时代发展，前端质量要求越来越高，独立出来 专门的 **前端开发**

## 跨域

```
✖ Access to XMLHttpRequest at 'http://127.0.0.1:8000/students' from origin 'http://127.0.0.1:8080' has been blocked by CORS policy: No 'Access-Control-Allow-Origin' header is present on the requested resource.
✖ Failed to load resource: net::ERR_FAILED
✖ Uncaught (in promise) Error: Network Error
    at createError (createError.js?2d83:16)
    at XMLHttpRequest.handleError (xhr.js?b50d:84)
```

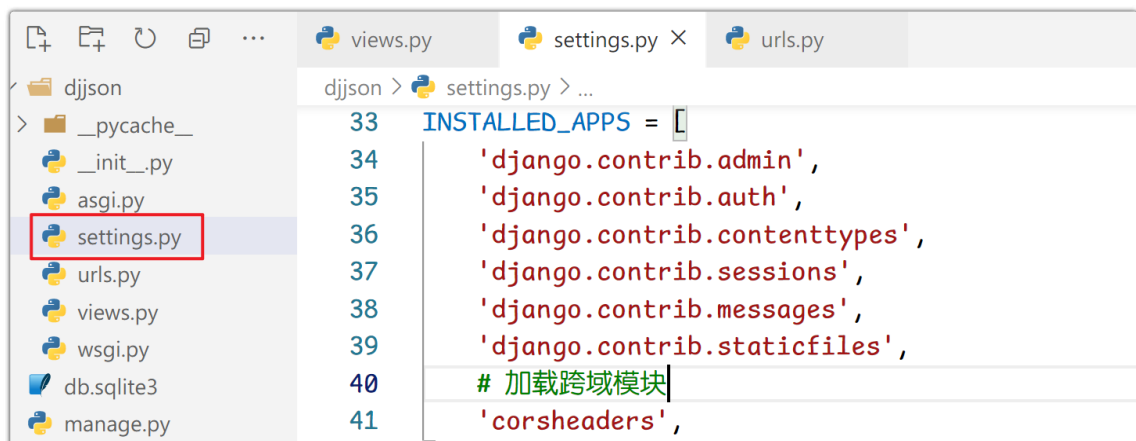
- 跨域的原因：浏览器具有 **同源策略**
  - **同源策略**：发送请求时，请求的 **协议**，**域名**，**端口号** 必须和当前网站完全一致

```
协议://域名:端口号
例如：
http://localhost:8080
```

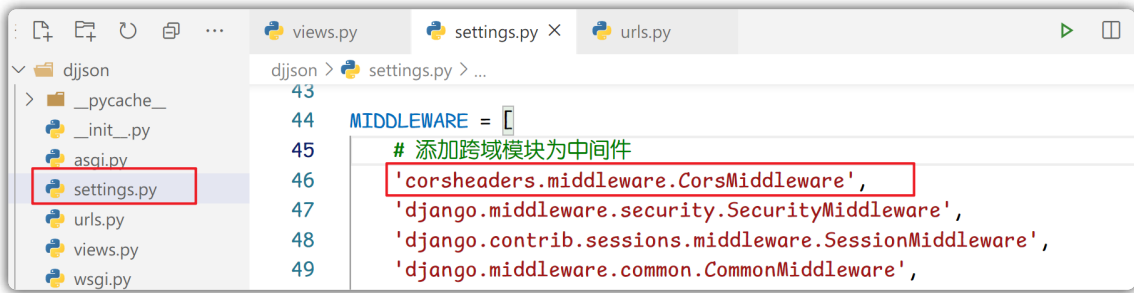
- 解决方案：常用3种
  - cors：由服务器**独立完成**，无需前端做任何操作；**最常见**的 上线策略
  - proxy：由前端**独立完成**，不需要服务器做操作；适合 **开发阶段的临时解决**。一定要上线，则必须使用 NGINX 服务器进行反向代理 --- **非正途**
  - jsonp：需要 服务器 和 前端 共同**合作**解决；属于**不常用**的方案；

## Django解决跨域 cors

- 安装跨域模块：`pip install django-cors-headers`
- 测试：`pip list`
- 配置文件中，加载跨域模块：`corsheaders`

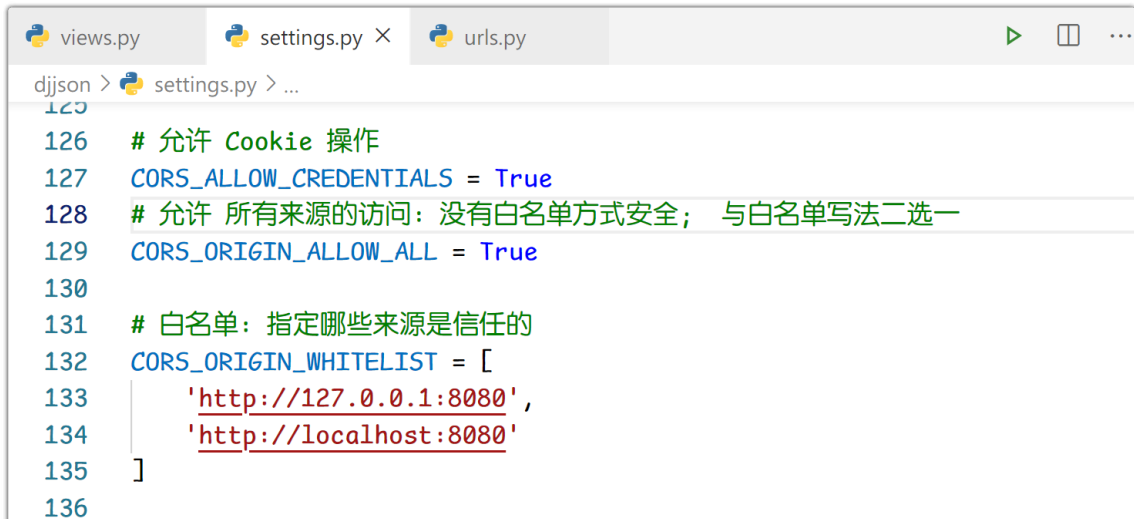


- 配置文件中，添加跨域模块为 中间件： `'corsheaders.middleware.CorsMiddleware',`



```
43
44 MIDDLEWARE = [
45     # 添加跨域模块为中间件
46     'corsheaders.middleware.CorsMiddleware',
47     'django.middleware.security.SecurityMiddleware',
48     'django.contrib.sessions.middleware.SessionMiddleware',
49     'django.middleware.common.CommonMiddleware',
```

- 添加白名单



```
126 # 允许 Cookie 操作
127 CORS_ALLOW_CREDENTIALS = True
128 # 允许 所有来源的访问：没有白名单方式安全； 与白名单写法二选一
129 CORS_ORIGIN_ALLOW_ALL = True
130
131 # 白名单：指定哪些来源是信任的
132 CORS_ORIGIN_WHITELIST = [
133     'http://127.0.0.1:8080',
134     'http://localhost:8080'
135 ]
136
```

## POST请求

post请求的测试，必须通过特殊的软件来实现：

<https://www.apipost.cn/download.html>

下载之后，可以自行安装； 打开时，需要注册账号之后使用。



关于错误: **Forbidden** 默认带有 CSRF 跨域攻击防御功能; 前后端分离项目不需要开启这个设置, 关闭即可!

```
settings.py ×
djson > settings.py > ...

44 MIDDLEWARE = [
45     # 添加跨域模块为中间件
46     'corsheaders.middleware.CorsMiddleware',
47     'django.middleware.security.SecurityMiddleware',
48     'django.contrib.sessions.middleware.SessionMiddleware',
49     'django.middleware.common.CommonMiddleware',
50     # 关闭 csrf 功能, 才能发送POST请求
51     # 'django.middleware.csrf.CsrfViewMiddleware',
52     'django.contrib.auth.middleware.AuthenticationMiddleware',
53     'django.contrib.messages.middleware.MessageMiddleware',
54     'django.middleware.clickjacking.XFrameOptionsMiddleware',
```



## 作业

把 vue 项目中的 `About.vue` 页面制作成添加学生页面; 使用新增学生接口实现录入操作

姓名:

年龄:

性别:

手机号:

邮箱:

确定

## 明日预告

使用产品表 `product`, 练习 前后端分离项目的制作过程



