Hello Backend

本篇将带你完成最新手友好的后端语言 Python 的开发环境搭建速通，并带你实现你的第一个 API

环境搭建

现场教学如何在 Windows 安装 Python 环境，和 VS Code。

用 VS Code 打开本项目压缩包文件，下载工作区推荐拓展，即本项目所需的 Python 拓展。

安装好后，按下 F5, 即可运行项目

第一个 API

Hello World

访问根目录时，返回一个 Hello World, 或是任何你想的欢迎用语

```
from fastapi import FastAPI

# 创建一个 FastAPI 应用实例
app = FastAPI()

# 定义一个“路由” (route)
# 当有人访问我们网站的根目录 ("/") 时，就执行下面的函数
@app.get("/")
def read_root():
    return {"message": "Hello, World!"}
```

按下 F5，访问 http://localhost:8000/ ，即可看到我们的输出结果

现代后端框架可以给我们提供很多有意思的功能。访问 http://localhost:8000/docs/, FastAPI 框架已经为我们准备好了交互式 API 文档。试试在这里测试我们刚刚写好的路由！

路径参数 - 让 API 动起来

API 当然是可以接受外部输入的。

如果 `/items` 是所有商品的货架，那 `/items/1` 就是货架上编号为 1 的那个特定商品。这个 1 就是路径参数。

```
fake_items_db = {
    1: {"name": "苹果"},
    2: {"name": "香蕉"},
    3: {"name": "橙子"}
}
```

```python
@app.get("/items/{item_id}")
def read_item(item_id: int):
    if item_id in fake_items_db:
        return fake_items_db[item_id]
    return {"error": "Item not found"}
```

路径中的 `{item_id}` 是一个占位符。函数参数 `item_id: int` 中的 `: int` 是“类型提示”，它告诉 FastAPI “我希望这里的 `item_id` 是一个整数”。FastAPI 会自动做检查，如果不是整数，会自动报错。

回到 /docs，会发现多了一个 /items/{item_id} 接口。

- 输入 1，成功获取苹果。
- 输入 99，看到我们自己定义的 {"error": "Item not found"}。
- 输入 abc (一个字符串)，看到 FastAPI 自动生成的清晰的错误信息（Validation Error）。

现代后端框架的强大之处就在于此，它帮我们省去了不少的麻烦。

查询参数 - 更灵活的筛选

你去图书馆，告诉管理员‘我要找书’（这是路径 `/books`），然后递上一张纸条写着‘作者是鲁迅，1920年之后出版’（这就是查询参数 `?author=鲁迅&year_after=1920`）。它是用来筛选和过滤的。

不过，查询参数也不一定真的是为了“查询”某一个东西。这里我们用查询参数做一个简单的加法。

```python
# /add?x=5&y=10
@app.get("/add")
def add_numbers(x: int = 0, y: int = 0):
    result = x + y
    return {"x": x, "y": y, "result": result}
```

查询参数不需要在路径 @app.get("/add") 中定义。函数参数 x: int = 0 表示：这是一个叫 x 的查询参数，我希望它是整数，如果用户不提供，它的默认值就是 0。框架已经帮我们神奇的从 HTTP 请求中提取出了这些查询参数。

再次回到 /docs，测试 /add 接口：

- 不输入任何参数，直接执行，看到 x=0, y=0, result=0 的默认结果。
- 输入 x=5, y=10，看到正确的计算结果 15。
- 同时，可以直接在浏览器地址栏访问 http://127.0.0.1:8000/add?x=5&y=10

在不到一个小时里，我们从一个空文件开始，构建了一个可以响应不同请求、能接收和验证参数、并且自带专业文档的API服务！你们已经掌握了后端开发最核心的几个概念！