

## 使用 FastAPI 构建后端的 API 服务

在现代 web 开发中，API（应用程序编程接口）成为了系统之间交互的核心。API 允许不同的软件应用之间交换数据，它是 前后端，Web 服务、移动应用间沟通的桥梁。FastAPI 是一个非常不错的框架，它简单高效，适合初学者快速上手。

---

### 1. 什么是 API?

**API（应用程序编程接口）** 是一组定义不同软件组件之间如何交互的规则和协议。API 是系统间“对话”的桥梁。比如，当你使用手机应用发送一条消息时，应用程序会向服务器发起一个请求，服务器根据请求返回响应数据。这个过程就通过 API 完成。

API 通常采用 HTTP 协议工作。常见的 HTTP 请求方法包括：

- **GET**：从服务器获取数据。
- **POST**：向服务器发送数据。
- **PUT**：更新服务器上的数据。
- **DELETE**：删除服务器上的数据。

### 2. 为什么使用 FastAPI?

**FastAPI** 是一个用于构建 API 服务的 Python 框架。它的优势包括：

- **高效**：FastAPI 基于 Python 的异步特性，能够高效处理大量请求。
- **自动化**：FastAPI 自动生成 API 文档，自动验证请求数据，减少开发者的负担。
- **易用性**：FastAPI 使用 Python 类型注解，代码简洁直观，适合初学者快速上手。

### 3. 基础概念

在开始编码之前，先了解几个关键概念：

- **路由 (Route)**：是 API 中用来指定请求路径和处理方法的规则。比如，`GET /users` 可能表示获取用户列表。
- **请求 (Request) 和响应 (Response)**：用户通过 HTTP 请求发送数据，服务器通过响应返回数据。
- **ASGI 服务器**：  
**ASGI** 是一种新的 Web 服务器协议，支持异步处理多个请求，适合高并发、实时应用等场景。

- **FastAPI** 是基于 ASGI 构建的框架，充分利用异步能力，能够在高并发环境下高效工作。
- **ASGI 服务器**（如 Uvicorn）是用于运行 ASGI 应用程序的服务器，能够接收、处理并响应多个并发请求。

## 4. 准备工作

安装 FastAPI 和 Uvicorn

在使用 FastAPI 构建 API 服务之前，首先需要安装相关的依赖库。你可以使用 `pip` 来安装它们：

```
pip install fastapi uvicorn
```

- **FastAPI** 是框架本身。
- **Uvicorn** 是 ASGI 服务器，用于运行 FastAPI 应用。

## 5. 创建你的第一个 API 服务

现在，我们开始构建一个简单的 API 服务。

代码示例

```
from fastapi import FastAPI
from pydantic import BaseModel

# 创建 FastAPI 应用实例，这行代码创建了一个 `FastAPI` 应用实例。你可以将 `app` 看作是
# 整个 API 服务的核心。
app = FastAPI()

# 定义请求体的数据结构，这里我们定义了一个 Pydantic 数据模型 `ChatRequest`，它
# 包含了一个 `message` 字段，类型是 `str`。Pydantic 是一个用于数据验证的库，
# FastAPI 使用它来确保请求数据符合预期格式。
class ChatRequest(BaseModel):
    message: str

# 创建路由，处理 POST 请求。`@app.post("/chat")` 是 FastAPI 的装饰器，表示当
# 有用户发送 POST 请求到 `/chat` 路径时，调用 `chat` 函数来处理。
@app.post("/chat")
def chat(chat_request: ChatRequest):
    # 返回响应
    return {"response": f"Received message: {chat_request.message}"}
```

```
# 启动 Uvicorn 服务器，这行代码启动了 Uvicorn 服务器，并监听所有网络接口的
8000 端口。通过 `http://localhost:8000`，你可以访问这个 API 服务。
if __name__ == "__main__":
    import uvicorn
    uvicorn.run(app, host="0.0.0.0", port=8000)
```

## 6. 测试你的 API

1. **运行应用**：运行这个文件脚本（其app.py更换为你的文件名）：

```
python app.py
```

这将启动你的 API 服务。

2. **发送请求**：你可以使用 Postman、cURL 或者浏览器访问以下地址：

```
POST http://localhost:8000/chat
```

或者在[`http://localhost:8001/docs#/`](http://localhost:8001/docs#/)

它的自动化接口文档平台上面，注意保持格式，只能更改值里面的内容：

The screenshot shows the Postman interface for a POST request to `/chat`. The request body is a JSON object: `{ "message": "你好呀" }`. A red arrow points to the text `"你好呀"` with the label **更改这里的文字** (Change the text here). Another red arrow points to the **Execute** button at the bottom with the label **点击执行** (Click to execute).

## Curl

```
curl -X 'POST' \
  'http://localhost:8001/chat' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
    "message": "你好呀"
  }'
```

请求体可以是：

```
{
  "message": "你好呀"
}
```

3. **查看响应：**服务器会返回类似的响应：

```
{ "response": "Received message: 你好呀" }
```

### Code

### Details

200

#### Response body

```
{
  "response": "Received message: 你好呀"
}
```

#### Response headers

```
content-length: 42
content-type: application/json
date: Thu, 05 Dec 2024 11:54:29 GMT
server: uvicorn
```

## 7. 自动化文档

FastAPI 提供了自动化生成 API 文档的功能。只需要访问以下链接，你就可以查看和测试你的 API，FastAPI 会自动生成这些文档，让你无需额外编写 API 文档：

- [Swagger UI 文档](#)
- [ReDoc 文档](#)