

# LLM教学助手说明

## 文件说明

### 1. Bash 脚本 ( `use_agent.sh` )

- **功能**：用于决定使用哪一个agent（文心一言&openai）。
- **逻辑**：
  - 如果没有提供参数，或提供的参数为 1，则运行第一个 agent（文心一言）。
  - 如果提供的参数为 2，则运行第二个 agent（openai）。
  - 如果提供了其他参数，则显示错误信息并退出。

### 2. Python 脚本 - 第一个 ( `QianFan-agent.py` )

- **功能**：与文心一言API交互，以模拟一个操作系统课程的教学助手。
- **主要组件**：
  - 获取百度 API 的访问令牌。
  - 使用多线程来处理用户输入和计时。
  - 在指定超时时间内等待用户输入问题。
  - 将用户输入发送到 AI 服务接口并显示返回的答案。
  - 会话结束时保存聊天历史。

### 3. Python 脚本 - 第二个 ( `ChatGPT-agent.py` )

- **功能**：利用 OpenAI 的 GPT 模型，模拟操作系统课程的教学助手。
- **主要组件**：
  - 使用 OpenAI 的 API。
  - 使用多线程处理用户输入和计时。
  - 收集用户问题，使用 GPT 模型生成答案。
  - 会话结束时，保存聊天历史。

## 使用说明

1. 在操作系统easy-lab的docker容器中使用，将上述三个文件一起放到一个名为 `Amadeus`（懂得都懂）的目录下，为了方便这里指定路径为 `/workspaces/project/Amadeus`
2. 使用 `cd -` 命令，在这个目录下有一个 `.bashrc` 文件，在这个文件的末尾增加一行 `alias amadeus='/workspaces/project/Amadeus/use_agent.sh'`，这行的意思是添加脚本文件进命令里，方便调用。
3. 如果第二步发现找不到方法打开这个文件，可以使用 `apt-get update` 然后使用 `apt-get install vim` 来安装vim编辑器（原装的Linux容器里居然连sudo都用不了，还好你是高贵的root用户），然后使用命令 `vim .bashrc` 即可编辑。（具体怎么用vim请自己查攻略捏）
4. 回到那个 `/workspaces/project/Amadeus` 中使用 `chmod +x use_agent.sh` 让你的脚本可以执行。
5. 按照python文件中使用的库进行库的安装，这时候你就会发现连pip都没有。先后运行命令：`wget https://bootstrap.pypa.io/get-pip.py` 和 `python3 get-pip.py` 安装pip，然后正常 `pip install` 就行了(应该只有 `openai` 和 `requests` 两个库需要安装)
6. 现在在命令行中输入 `amadeus` 就可以使用了，要使用openai的接口就用 `amadeus 2` 命令即可（我试了一下下，这个openai的接口需要绑定美国银行卡，我没有所以这里没有办法用，你要用的话就用你的密钥替换 `openai.api_key` 变量即可）
7. 过一定的时间（默认5分钟）不进行问答，助手就会关闭，再次按下回车的话，当次对话会以json文件的形式保存在history文件夹里，命名逻辑是：年-月-日-时-分-秒（可以忽略第一次对话，那个对话是我设置的一些prompt来告知LLM此次对话是有关操作系统课程的），如果要再次使用就再开一个吧，考虑到操作系统相关问答不需要太强的连续性，没有考虑历史加载的问题。

## 使用实例（仅使用文心一言）

- 1.在终端输入amadeus进入问答系统，然后输入问题即可

```
root@d3fc415bdbd6:/workspaces/project/Amadeus# amadeus
请输入您的问题：
我想知道进程和线程之间的区别
```

## 2.等几秒，便能得到答案

回答：

进程和线程是操作系统中两种不同的多任务处理方式，它们之间存在一些重要的区别。

进程（Process）：

- \* 进程是操作系统进行资源分配和执行的基本单位。每个进程都有自己的内存空间和系统资源，如内存、文件描述符等。
- \* 进程之间是相互独立的，它们拥有自己的地址空间，运行时拥有独立的执行序列。
- \* 进程是操作系统资源分配的基本单位，它与其他进程是分离的，只有在进程之外才有它的外部视角（如文件名）。

线程（Thread）：

- \* 线程是进程的一个执行实体，它是操作系统调度任务的基本单位。
- \* 线程共享其父进程的资源，如内存空间和系统资源。这意味着线程可以访问其父进程的数据和代码段。
- \* 线程之间的通信主要通过共享内存进行，因此它们之间的通信更加高效。
- \* 线程的创建、切换和销毁的成本通常比进程要高一些，因为操作系统需要为每个线程分配和管理其自己的资源。

主要区别：

1. 资源占用：进程拥有独立的内存空间和系统资源，而线程共享其父进程的资源。
2. 执行方式：进程是独立的执行实体，有自己的执行序列；而线程是进程的一部分，依赖于进程的存在。
3. 通信方式：进程间的通信需要通过操作系统提供的消息传递机制进行；线程间的通信主要通过共享内存进行。
4. 创建和销毁成本：创建和销毁线程的成本通常比进程要高一些。

总的来说，进程和线程各有优缺点，选择使用哪种方式取决于具体的应用场景和需求。在某些情况下，使用多个进程可以提供更好的资源隔离和安全性；而在其他情况下，使用多个线程可以提供更高的性能和效率。

## 3.接下来如果还有问题可以继续提问

请输入您的问题：  
操作系统通过什么方法切换内核态和用户态

## 4.自然也能得到回答（其实感觉答案有点长，这里可以在api那里做一个回答长度的限制）

回答：

操作系统通过硬件异常和系统调用两种方法来切换内核态和用户态。

在硬件异常情况下，当程序尝试执行操作系统分配给内核的内存区域时，会触发一个硬件异常，操作系统接收到异常后，会切换到内核态，并执行相应的异常处理程序。在系统调用情况下，用户程序可以通过系统调用号和参数向操作系统请求执行特定的系统调用，操作系统接收到请求后，会切换到内核态，并执行相应的系统调用程序。在执行完系统调用后，操作系统会将程序切换回用户态，以使用户程序可以继续执行。

总之，切换内核态和用户态是操作系统安全性和可靠性的重要保障，操作系统通过硬件异常和系统调用两种方法来实现这种切换。

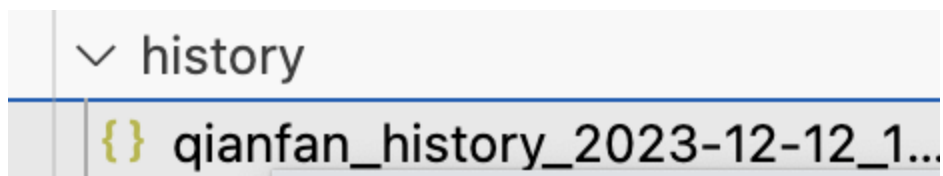
答案还比较正确。

5.等5min没有输入，我们的Kurusu Makise就要休息了

请输入您的问题：

小助手要休息咯，有什么问题再来问我吧

可以看到生成了结果记录



```
{
  "role": "user",
  "content": "我想知道进程和线程之间的区别"
},
{
  "role": "assistant",
  "content": "进程和线程是操作系统中两种不同的多任务处理方式，它们之间存在一些重要的区别。进程是操作系统中的基本单位，它拥有独立的内存空间，可以执行独立的程序。而线程是进程的一部分，它共享进程的内存空间，可以并行执行。进程和线程的主要区别在于内存空间、资源消耗和执行效率等方面。"
},
```