SpringAI+Ollama三部曲之一：极速体验

欢迎访问我的GitHub

> 这里分类和汇总了欣宸的全部原创(含配套源码
): [github.com/zq2599/blog...](http://cxyroad.com/
"https://github.com/zq2599/blog_demos")

关于ollama

* ollama和LLM（大型语言模型）的关系，类似于docker和镜像，可以在ollama服务中管理和运行各种LLM，下面是ollama命令的参数，与docker管理镜像很类似，可以下载、删除、运行各种LLM

```
Available Commands:
  serve       Start ollama
  create      Create a model from a Modelfile
  show        Show information for a model
  run         Run a model
  pull        Pull a model from a registry
  push        Push a model to a registry
  list        List models
  cp          Copy a model
  rm          Remove a model
  help        Help about any command
```

* 官网：[ollama.com/](http://cxyroad.com/ "https://ollama.com/")
![在这里插入图片描述](https://p3-juejin.byteimg.com/tos-cn-i-k3u1fbpfcp/165f48d6ecf443c78b38cb3202dc140a~tplv-k3u1fbpfcp-jj-mark:3024:0:0:0:q75.awebp#?w=890&h=902&s=103421&e=png&b=fefefe)
* 简单的说，有了ollama，咱们就可以在本地使用各种大模型了，ollama支持的全量模型在这里：[ollama.com/library](http://cxyroad.com/ "https://ollama.com/library")
* 官方给出的部分模型

Model	Parameters	Size	下载命令
Llama 3	8B	4.7GB	`ollama run llama3`
Llama 3	70B	40GB	`ollama run llama3:70b`
Phi-3	3.8B	2.3GB	`ollama run phi3`
Mistral	7B	4.1GB	`ollama run mistral`
Neural Chat	7B	4.1GB	`ollama run neural-chat`
Starling	7B	4.1GB	`ollama run starling-lm`
Code Llama	7B	3.8GB	`ollama run codellama`
Llama 2 Uncensored	7B	3.8GB	`ollama run llama2-uncensored`
LLaVA	7B	4.5GB	`ollama run llava`
Gemma	2B	1.4GB	`ollama run gemma:2b`
Gemma	7B	4.8GB	`ollama run gemma:7b`
Solar	10.7B	6.1GB	`ollama run solar`

* 另外需要注意的是本地内存是否充足，7B参数的模型需要8G内存，13B需要16G内存，33B需要32G内存

关于《SpringAI+Ollama三部曲》系列

* 《SpringAI+Ollama三部曲》是《Spring AI实战》的子系列，特点是专注于使用SpringAI来发挥Ollama的功能，由以下三篇文章构成

1. 极速体验：用最简单的操作，在最短时间内体验Java调用Ollama的效果
2. 细说开发：说明《极速体验》的功能对应的整个开发过程，把代码的每一步都说得清清楚楚（含前端）
3. 延伸扩展：SpringAI为Ollama定制了丰富的功能，以进一步释放Ollama的能力，文章会聚焦这些扩展能力

本篇概览

* 本篇聚焦操作和体验，不涉及开发（后面的文章会有详细的开发过程），力求用最短时间完成本地部署和体验，感受Java版本大模型应用的效果
* 今天要体验的服务，整体部署架构如下

mark:3024:0:0:0:q75.awebp#?w=368&h=830&s=21145&e=png&b=ffffff)

* 最终效果如下

![image.png](https://p3-juejin.byteimg.com/tos-cn-i-k3u1fbpfcp/b7668ba24f3e42ffbfe3d59971b651a9~tplv-k3u1fbpfcp-jj-mark:3024:0:0:0:q75.awebp#?w=982&h=323&s=108433&e=png&b=fcfcfc)

* 今天要做的所有事情汇总如下，嗯，好像挺简单的

![image.png](https://p1-juejin.byteimg.com/tos-cn-i-k3u1fbpfcp/d6c7cfc58b504d60aa06d6db4117ce49~tplv-k3u1fbpfcp-jj-mark:3024:0:0:0:q75.awebp#?w=510&h=1592&s=53301&e=png&b=f0f4fc)

* 接下来咱们开始行动呗，正宗的Java程序员开始参与大模型(应用)相关开发工作了，此刻的我内心是激动的

环境要求

* 完成本篇的实战需要一台Linux操作系统的电脑(虚拟机、WSL2也行)，电脑上部署了docker+docker-compose，电脑需要8G内存

确定本篇要用的模型

* Ollama支持的模型有很多，这里打算使用[通义千问](http://cxyroad.com/ "https://ollama.com/library/qwen")，官方网页
![在这里插入图片描述](https://p3-juejin.byteimg.com/tos-cn-i-k3u1fbpfcp/4da688ab4fa94f938f1731fd26c26d18~tplv-k3u1fbpfcp-jj-mark:3024:0:0:0:q75.awebp#?w=609&h=275&s=41800&e=png&b=fcfcfc)
* 本次实战选择了较小的1.8b版本，您可以根据自己的实际情况在[官网](http://cxyroad.com/ "https://ollama.com/library")选择其他版本
* 稍后会将选好的模型写在配置文件中

准备工作（操作系统和docker）

* 本次实战的两个重要前提条件：

1. 操作系统是Linux，我这里用的是Ubuntu 24.04 LTS服务器版
2. docker和docker-compose已经部署好，我这里docker版本是26.1.2

* 由于docker镜像较大，所以请提前准备好docker镜像加速，方法很多，我这里用的是阿里云的，如下图

* 前面介绍Ollama时提到过官方对内存的要求，所以这里请确保本次实战的电脑配置不要过低，我这边运行的模型是通义千问的1.8b，总消耗如下

```
 free -g
               total        used        free      shared  buff/cache   available
Mem:              31           3          21           0           6          27
Swap:              7           0           7
```

准备工作（保存文件的目录）

* 在电脑上准备两个干净目录，用来保存docker容器中的数据，这样即便是容器被销毁了数据也会被保留（例如模型文件），等到再次启动容器时这些文件可以继续使用
* 第一个是用来保存ollama的文件，我这里是/home/will/data/ollama
* 第二个是用来保存ollama webui的文件，我这里是/home/will/data/webui
* 这两个目录会配置到稍后的docker-compose.yml文件中，您要注意同步修改

准备工作(SpringBoot应用的配置文件application.properties)

* 准备好SpringBoot应用的配置文件application.properties，这样便于各种个性化设置
* 我这边在/home/will/temp/202405/15目录准备好配置文件，内容如下

```
```

```
spring.ai.ollama.base-url=http://ollama:11434
spring.ai.ollama.chat.options.model=qwen:1.8b
spring.ai.ollama.chat.options.temperature=0.7
spring.main.web-application-type=reactive
```

* 注意：本篇使用的模型是**qwen:1.8b**，如果您要用其他模型，请在这里修改好
* 至此，准备完毕，进入部署阶段

部署工作(编写docker-compose文件)

* 新增名为docker-compose.yml的文件，内容如下

```
version: '3.8'
services:
  ollama:
    image: ollama/ollama:latest
    ports:
      - 11434:11434
    volumes:
      - /home/will/data/ollama:/root/.ollama
    container_name: ollama
    pull_policy: if_not_present
    tty: true
    restart: always
    networks:
      - ollama-docker

  open-webui:
    image: ghcr.io/open-webui/open-webui:main
    container_name: open-webui
    pull_policy: if_not_present
    volumes:
      - /home/will/data/webui:/app/backend/data
    depends_on:
      - ollama
    ports:
      - 13000:8080
    environment:
      - 'OLLAMA_BASE_URL=http://ollama:11434'
      - 'WEBUI_SECRET_KEY=123456'
```

```
      - 'HF_ENDPOINT=https://hf-mirror.com'
    extra_hosts:
      - host.docker.internal:host-gateway
    restart: unless-stopped
    networks:
      - ollama-docker

  java-app:
    image: bolingcavalry/ollam-tutorial:0.0.1-SNAPSHOT
    volumes:
      -
/home/will/temp/202405/15/application.properties:/app/application.properties
    container_name: java-app
    pull_policy: if_not_present
    depends_on:
      - ollama
    ports:
      - 18080:8080
    restart: always
    networks:
      - ollama-docker

networks:
  ollama-docker:
    external: false
```

* 上面的内容中，前两个volumes的配置对应的是准备工作中新建的两个目录，第三个volumes对应的是刚才新建的application.properties，请按照您的实际情况进行修改

部署(运行docker-compose)

* 进入docker-compose.yml文件所在目录，执行以下命令就完成了部署和启动

```
docker-compose up -d
```

* 本次启动会用到电脑的这三个端口：11434、13000、18080，如果这些端口

有的已被使用就会导致启动失败，请在docker-compose.yml上就行修改，改为没有占用就行，然后执行以下命令（先停掉再启动）

```
docker-compose down
docker-compose up -d
```

* 启动期间，下载docker镜像时因为文件较大，需耐心等待（再次提醒，请配置好docker镜像加速）
`![在这里插入图片描述](https://p3-juejin.byteimg.com/tos-cn-i-k3u1fbpfcp/d6479061c67a4a269ff855e80f607971~tplv-k3u1fbpfcp-jj-mark:3024:0:0:0:q75.awebp#?w=846&h=341&s=100709&e=png&b=181818)`
* 启动成功后，控制台显示如下

```
[+] Building 0.0s (0/0)

[+] Running 4/4
  Network files_ollama-docker  Created                                                             0.1s
  Container ollama             Started                                                          0.2s
  Container java-app           Started                                                           0.4s
  Container open-webui         Started
```

* 现在服务都启动起来了，但是还不能用，咱们还要把大模型下载下来

部署（指定大模型）

* 登录webui服务，地址是[http://192.168.50.134:13000](http://cxyroad.com/ "http://192.168.50.134:13000")，192.168.50.134是运行docker-compose的电脑IP
* 打开地址，会提示注册或者登录，这里要注册一下
![在这里插入图片描述](https://p3-juejin.byteimg.com/tos-cn-i-k3u1fbpfcp/9a0db3c2feda458aa438d0169d74153b~tplv-k3u1fbpfcp-jj-

mark:3024:0:0:0:q75.awebp#?w=415&h=373&s=28038&e=png&b=fdfdfd
)
* 注册成功后显示登录成功的页面，如下图
![在这里插入图片描述](https://p3-juejin.byteimg.com/tos-cn-i-
k3u1fbpfcp/38d3a33a54d34788ab70363cc3e71f24~tplv-k3u1fbpfcp-jj-
mark:3024:0:0:0:q75.awebp#?w=2466&h=1452&s=253168&e=png&b=66
6666)
* 现在来下载模型，操作如下
![在这里插入图片描述](https://p3-juejin.byteimg.com/tos-cn-i-
k3u1fbpfcp/9d6271a4f28e4c5d856bdd59ccb12a8f~tplv-k3u1fbpfcp-jj-
mark:3024:0:0:0:q75.awebp#?w=1540&h=1214&s=151030&e=png&b=fcf
cfc)
* 输入模型名称然后开始下载
![在这里插入图片描述](https://p3-juejin.byteimg.com/tos-cn-i-
k3u1fbpfcp/608106945ba74b53b27e7d8e7792c013~tplv-k3u1fbpfcp-
jj-
mark:3024:0:0:0:q75.awebp#?w=1408&h=856&s=127260&e=png&b=f9f9
f9)
* 模型下载完成后会有如下提示
![在这里插入图片描述](https://p3-juejin.byteimg.com/tos-cn-i-
k3u1fbpfcp/7074f30a90ed41e1a522e5538f401b1e~tplv-k3u1fbpfcp-jj-
mark:3024:0:0:0:q75.awebp#?w=1406&h=1290&s=144697&e=png&b=f9f
9f9)
* 可以直接在webui上体验刚下载的模型，尝试了基本的问答，没有问题
![在这里插入图片描述](https://p3-juejin.byteimg.com/tos-cn-i-
k3u1fbpfcp/609c47e47360462f99cf2ead49f2ad08~tplv-k3u1fbpfcp-jj-
mark:3024:0:0:0:q75.awebp#?w=2972&h=826&s=321209&e=png&b=fefe
fe)
* 至此，部署和启动都完成了，可以体验Java应用了

体验

* 浏览器打开地址[http://192.168.50.134:18080](http://cxyroad.com/
”http://192.168.50.134:18080”)，如下
![在这里插入图片描述](https://p3-juejin.byteimg.com/tos-cn-i-
k3u1fbpfcp/cee341bab9f64fab841ae0f61dbd9b38~tplv-k3u1fbpfcp-jj-
mark:3024:0:0:0:q75.awebp#?w=1376&h=582&s=60243&e=png&b=ffffff)
* 效果如下

![image.png](https://p3-juejin.byteimg.com/tos-cn-i-
k3u1fbpfcp/87175b596714455ea596586eb4df85d6~tplv-k3u1fbpfcp-jj-
mark:3024:0:0:0:q75.awebp#?w=982&h=323&s=108433&e=png&b=fcfcf
c)

* 至此，本篇的任务已经完成，一个本地化部署的大模型应用已经就绪，也完成了最基本的体验，过程十分简单（好像也就输入了几行命令，打开浏览器点了几下）
* 简单的背后，其实现是否也简单呢？先剧透一下吧，得益于Spring团队的一致性风格，调用Ollama的过程和操作数据库消息队列这些中间件差不多，几行代码几行配置就够了
* 至于完整的开发过程，就留到下一篇吧，那里会给出所有源码和说明

欢迎掘金：程序员欣宸

> 学习路上，你不孤单，欣宸原创一路相伴...

原文链接: https://juejin.cn/post/7369867966228840460