RaspberryPi5 で Ollama を使って､LLM を動かしてみよう！

初心者向けに、ChatGPT のような対話 AI の機能を安価に、ローカル内で使えるように **Ollama サーバーの構築** と **簡易呼び出しページ** の構築を行う教材です。

やること

Raspberry Pi 5 で Ollama を使用可能な状態にし、動作するモデルのインストール、サーバー化（**詳細: Ollama_install.md**）
同一ネットワーク上と Raspberry Pi 5 本体（ローカル）からの呼び出しを簡単にする、サイトのホスティング（**詳細: Hosting.md**）

※本教材ではワークショップ都合上、**SSH は使いません**（キーボード・マウス・ディスプレイ直結で操作）。

※ **インストールとサーバー化** の手順は Ollama_install.md、**ホスティング** の手順は Hosting.md に集約しています。本 README は **概念・用語・コマンド早見表** を中心にしています。

目次

1. はじめに：Raspberry Pi / LLM / Ollama とは？
2. 関連する名称・用語早見表（コーディングで出てくる名前も）
3. 準備するものと環境の前提
4. ディレクトリと「カレント」の考え方
5. ターミナルでよく使うコマンド（超基本）
6. Ollama の基本操作（よく使うコマンド / 環境変数 / ポート）
7. API 呼び出し例（curl / JavaScript / Python）
8. ホスティング（詳細は Hosting.md）
9. 注意点（電源・熱・SD・セキュリティ・権限）
10. よくあるつまずき Q&A
11. 付録：プロジェクト構成の例

1. はじめに：Raspberry Pi / LLM / Ollama とは？

- **Raspberry Pi（ラズベリーパイ）**：小型・低価格のシングルボードコンピュータ。Raspberry Pi 5 は従来より大幅に高性能。一般的に **Raspberry Pi OS（Debian 系）** を使います。CPU は **ARM64（aarch64）** アーキテクチャです。
- **LLM（Large Language Model）**：大量のテキストから学習した、文章の生成や要約、翻訳、会話ができる AI のこと。ChatGPT の背後にも LLM があり、代表的なモデルに Llama 3 / Mistral / Phi-3 / Gemma などがあります。
- **Ollama**：ローカル PC やサーバーで LLM を簡単に動かせるツール。**モデルの取得（pull）→ 実行（run）→ サーバー化（serve）** まで一気通貫で扱え、HTTP API（既定ポート **11434**）も提供します。

2. 関連する名称・用語早見表（コーディングで出てくる名前も）

Raspberry Pi / OS / ネットワーク系

- **Raspberry Pi OS**：標準 OS。パッケージ管理は apt。
- **ARM64 / aarch64**：Pi 5 の CPU アーキテクチャ。ソフトの対応表記で見かけます。
- **hostname / IP**：機器名とネットワーク上の住所。LAN 内アクセスで使用。
- **/home/pi**：既定ユーザー pi のホームディレクトリ。
- **systemd / systemctl**：Linux のサービス管理。
- **LAN（同一ネットワーク）**：ご家庭や会場内のネットワーク。**WAN（インターネット公開）は非推奨**。

LLM / 推論系

- **モデル（model）**：学習済みデータ本体（例：gemma3:1b、yuiseki/sarashina2.2:1b など）。
- **トークン（token）**：文章を小さな単位に分けたもの。**コンテキスト長（context window）** は同時に扱えるトークン数。
- **プロンプト（prompt）**：モデルに与える指示文。
- **推論（inference）**：学習済みモデルで回答を生成する処理。
- **量子化（quantization）**：精度を少し落としてメモリを節約する変換（Pi では有利）。

Ollama / API / コーディング系

- **Ollama サーバー**：ollama serve で起動。HTTP API は **http://<IP>:11434/**。
- **エンドポイント**：/api/generate（テキスト生成）、/api/embeddings（埋め込み生成）など。
- **環境変数 OLLAMA_HOST**：0.0.0.0:11434 で全インターフェース待受、127.0.0.1:11434 でローカルのみ。
- **よくあるモデル名**：gemma3:1b、yuiseki/sarashina2.2:1b 等（表記は例）。

3. 準備するものと環境の前提

- **Raspberry Pi 5 本体**、公式または品質の良い **USB-C 電源（5V/5A 目安）**
- **冷却**（ヒートシンク + ファン推奨）
- **microSD カード**（A2 / U3 推奨。書き込み耐性の高いもの）
- **有線または安定した Wi-Fi**（同一 LAN 内で PC/スマホからアクセス）
- **Raspberry Pi OS（64-bit）** をセットアップ済みでログイン可能

> 省電力・省メモリのため、初回は小さめのモデル（gemma3:1b など）から試すのがコツ。

4. ディレクトリと「カレント」の考え方

- **カレントディレクトリ（現在位置）**：ターミナルが今いるフォルダ。pwd で確認。
- **ホーム**：~ は自分のホーム（例：/home/pi）。
- **相対パス / 絶対パス**：./file.txt（今いる場所から）/ /home/pi/file.txt（ルートから）。
- **. と ..**：カレント自身 / ひとつ上の階層。
- **プロジェクト用ディレクトリ例**：/home/pi/ollama-project に素材やスクリプトをまとめる。

5. ターミナルでよく使うコマンド（超基本）

移動・確認

```
pwd                 # 今いる場所を表示

ls -la              # 詳細表示（隠しファイル含む）cd /path/to/dir      # 指定ディレクトリへ移動 cd ..
# ひとつ上へ cd -                 # 直前の場所へ戻る
```

ファイル操作

```
mkdir -p mydir/sub  # ディレクトリ作成（-p で中間も一気に）

cp src.txt dst.txt  # コピー

mv a.txt b.txt      # 移動/改名

rm -i file.txt      # 削除（確認つき）

rm -r folder        # フォルダごと削除（慎重に！）
```

中身を見る・編集する

```
cat file.txt        # 中身を一気に表示

less file.txt       # スクロールして閲覧（q で終了）

head -n 20 file.txt # 先頭だけ

tail -n 50 file.txt # 末尾だけ（-f で追従）

nano file.txt       # 手軽なテキスト編集（Ctrl+O 保存 / Ctrl+X 終了）
```

システム情報・ネットワーク

```
uname -a            # OS/カーネル情報

hostname -I         # LAN 内の IP アドレス

ip addr             # 詳細なアドレス

free -h             # メモリ使用量

df -h               # ディスク使用量

top                 # プロセス監視（q で終了）
```

パッケージ管理（Raspberry Pi OS / Debian 系）

```
sudo apt update                     # パッケージ情報を更新

sudo apt upgrade -y                 # 更新を適用

sudo apt install -y htop curl wget  # よく使うツールの例
```

サービス管理（systemd）

```
sudo systemctl status <name>        # 状態を見る

sudo systemctl start <name>         # 起動

sudo systemctl stop <name>          # 停止

sudo systemctl enable <name>        # 起動時に自動起動

sudo journalctl -u <name> -f        # ログを追跡
```

6. Ollama の基本操作（よく使うコマンド / 環境変数 / ポート）

> インストール・常駐化の詳細は Ollama_install.md を参照。ここでは実行時の“型”をまとめます。

```
# モデルを取得（初回）

ollama pull gemma3:1b

# その場で対話実行（終了は Ctrl+C）

ollama run gemma3:1b

# サーバーとして起動（ローカルのみ）

ollama serve

# LAN からのアクセスを許可して起動（例）# ※ 教材の LAN 内のみで使う。公開ネットには出さない。

OLLAMA_HOST=0.0.0.0:11434 ollama serve

# 稼働中のモデル/ジョブ確認

ollama ps

# ローカルにあるモデル一覧

ollama list

# モデルの情報表示

ollama show gemma3:1b
```

> 既定ポート：**11434**（http://<Pi の IP>:11434/）
>
> よく使う環境変数：
>
> - OLLAMA_HOST=0.0.0.0:11434 … 全インターフェースで待受（LAN からも可）
> - OLLAMA_HOST=127.0.0.1:11434 … ローカル限定

7. API 呼び出し例（curl / JavaScript / Python）

curl（最小）

```
curl http://<PI の IP>:11434/api/generate ¥
  -H 'Content-Type: application/json' ¥
  -d '{"model": "gemma3:1b", "prompt": "Raspberry Pi とは？60 字で", "stream": false}'
```

JavaScript（fetch）

```
async function ask(prompt) {
  const res = await fetch("http://<PI の IP>:11434/api/generate", {
    method: "POST",
    headers: { "Content-Type": "application/json" },
    body: JSON.stringify({ model: "gemma3:1b", prompt, stream: false })
  });
  const data = await res.json();
  return data.response; // 生成されたテキスト}
```

Python（requests）

```
import requests
def ask(prompt: str) -> str:
    url = "http://<PI の IP>:11434/api/generate"
    payload = {"model": "gemma3:1b", "prompt": prompt, "stream": False}
    return requests.post(url, json=payload).json()["response"]
```

stream: false は一括返却の例。逐次ストリーミングも可能です。

8. ホスティング（詳細は Hosting.md）

ホスティングの設定・常駐化・TLS 等の具体手順は **Hosting.md** を参照してください。ここでは位置づけのみ：

- フロントは LAN 内配布が目的（**インターネット公開はしない**）
- 本番運用は nginx / lighttpd 等の常駐 Web サーバーを推奨

9. 注意点（電源・熱・SD・セキュリティ・権限）

- **電源**：Pi 5 は消費電力が高め。5V/5A クラス推奨。電圧降下に注意。
- **冷却**：ファン/ヒートシンク必須レベル。高負荷で温度上限に達すると性能低下（サーマルスロットリング）。
- **SD カード**：書き込みが多いと劣化。重要データはバックアップを。必要なら外付け SSD も検討。
- **セキュリティ**：**11434 をインターネットに公開しない**。LAN 内だけで使う。パスワードは強固に。
- **権限**：sudo rm -r 等の破壊的コマンドは慎重に。よく分からない sudo は実行しない。
- **メモリ**：大きすぎるモデルは動きません。小さいモデルから開始し、様子を見て段階的に。

10. よくあるつまずき Q&A

- **ollama: command not found**：インストールやパス設定を確認。シェルの再起動も試す。
- **ポートが使われている**：sudo lsof -i :11434 で占有プロセスを確認し停止。
- **メモリ不足で落ちる**：より小さいモデルにする / 同時実行を減らす / 不要プロセスを止める。
- **LAN から繋がらない**：OLLAMA_HOST が 0.0.0.0:11434 で起動しているか、Pi の IP を確認。
- **応答が遅い**：モデルを小さく / プロンプトを短く / 温度やバッテリー状態を確認（電源弱いと遅くなる）。

11. 付録：プロジェクト構成の例

```
/home/pi/ollama-project
├── Ollama_install.md        # インストール～サーバー化の詳細
├── Hosting.md               # ホスティング手順（nginx/lighttpd 等）
├── site/                    # 簡易ホスティング用のフロント
│   ├── index.html
│   ├── app.js                # fetch で Ollama API に問い合わせ
│   └── style.css
├── scripts/                 # 起動や補助スクリプト
│   ├── start_ollama.sh       # OLLAMA_HOST 設定 → serve 起動の例
│   └── healthcheck.sh
├── notes/                   # メモや手順書
│   └── README.md
└── models.txt               # 使うモデル名の控え
```

scripts/start_ollama.sh の例

```
#!/usr/bin/env bashset -euo pipefailexport OLLAMA_HOST=0.0.0.0:11434

nohup ollama serve > ~/ollama.log 2>&1 &echo "Ollama serving on $OLLAMA_HOST"
```

以上。ここまでの内容を手元の環境に合わせて調整しながら進めてください。困ったら「どこで・何をしたら・どうなった」をメモして質問すると解決が早いです。