

AIコーディングのための ローカルLLM構築入門

みやはら とおる
株式会社びぎねっと

講師 自己紹介

宮原 徹

- 1972年1月 神奈川県生まれ
- 1994年3月 中央大学法学部法律学科卒業
- 1994年4月 日本オラクル株式会社入社
 - PCサーバー向けRDBMS製品マーケティングに従事
 - Linux版Oracle8の日本市場向け出荷に貢献
- 2001年1月 株式会社びぎねっと 設立
- 2004年9月 Open Source Conferenceを開始
- 2006年12月 日本仮想化技術株式会社 設立



本日のアジェンダ

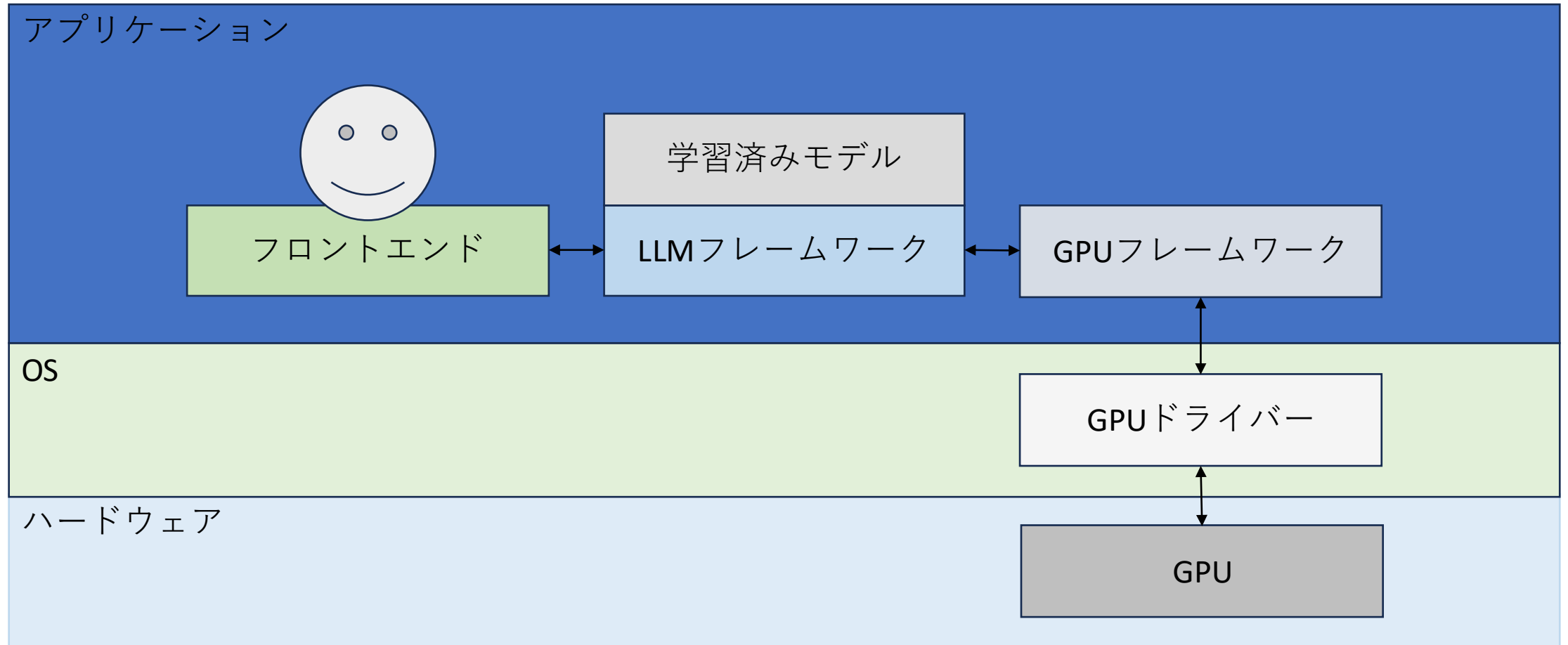
- AIを使うには
- AlmaLinuxでLLMを動かそう
- AIコーディングしてみよう

AIを使うには

AIについて

- Artificial Intelligence（人工知能）
- コンピューターの性能が向上したことで人間の知的生産活動を模して処理が行えるようになった
- 文章や画像、動画を生成する「生成AI」
 - 文章を生成するLLM（Large Language Models：大規模言語モデル）
 - 画像や動画を生成する拡散モデル（Diffusion Model）

LLM実行環境の構成図



LLMの構成要素

- 学習済みモデル
 - 様々な文章データを学習させたデータの集合体
 - モデルリポジトリとしてHugging Faceが有名
- LLMフレームワーク
 - モデルを読み込んで推論を実行するプログラム
- フロントエンド
 - チャットや外部APIなどを提供するプログラム

よく使われているLLMフレームワーク

- LM Studio
 - LM Studioはフロントエンド（非OSS・業務可）
 - バックエンドでllama.cppというLLMフレームワークを実行（OSS）
- Ollama（OSS）
 - 性能と使いやすさのバランスがいい（らしい）
 - 様々なAI連携ツールと組み合わせられる
- vLLM（OSS）
 - 高速実行を目指したフレームワーク
 - 並列実行に強み

LLMを動かすには

- GPUを使用できるハードウェア
 - OSからGPUを認識できる（仮想マシンではPCIパススルーが必要）
 - CPUでも動かさなくはないが遅い
- GPUを動作させるドライバー
- GPUを活用するためのGPUフレームワーク
 - NVIDIAのCUDA
 - AMDのROCm（ロックスエム）
 - 汎用的なVulkan（旧OpenGL）
- LLMフレームワーク
 - 使用するGPUフレームワークへの対応が必要
- フロントエンドツール

GPU選び

- とことん性能を追求しないのであれば、とりあえず有り物でOK
- メモリ**8GB**ぐらいでもそこそこのモデルは動く
- モデルのパラメータ数である**○B > ○G**ぐらいのイメージ
- 大きめのモデルを動かしたいなら**20GB～32GB**ぐらいのGPUを
 - お高くなります（25万～）
- さらに大きいモデルを動かしたいなら**UMA**
 - UMA（Unified Memory Architecture）はメインメモリとGPUメモリが共用のアーキテクチャ
 - Apple SiliconやRyzen AI Max+ 395、NVIDIA GB10など
 - 128GB搭載モデルで50万～100万程度

AlmaLinuxでLLMを動かそう

AlmaLinuxでNVIDIA製GPUを動かす

1. NVIDIA製GPUを搭載したPCにAlmaLinuxをインストール
2. GPUドライバーをインストール
3. CUDAをインストール
4. LLMをインストール
5. モデルをダウンロード

LLM動かすまでの作業手順

1. リポジトリの登録
2. NVIDIAドライバーのインストール
3. NVIDIA GPU関連ツールのインストール
4. GPUを認識しているかの確認
5. CUDAのインストール
6. LM Studioのインストール
7. LLMモデルのダウンロードと実行

リポジトリの登録

- `$ sudo dnf install almalinux-release-nvidia-driver`
 - NVIDIA Driver & CUDA リポジトリ
- `epel-release` パッケージも入る
 - EPEL10 リポジトリが有効化
 - 実際には CodeReady Builder (CRB) も有効化される

NVIDIAドライバのインストール

- `$ sudo dnf install nvidia-open-kmod nvidia-driver`
- インストール後に再起動
- `lsmod`コマンドでnvidia関連モジュールがロードされていることを確認

```
tmiyahar@AL10CUDA:~$ lsmod | grep nvidia
nvidia_drm                163840  22
nvidia_modeset            1929216  7 nvidia_drm
nvidia                    17973248 109 nvidia_modeset
drm_ttm_helper            16384   1 nvidia_drm
video                     81920   1 nvidia_modeset
tmiyahar@AL10CUDA:~$
```

関連ツールのインストールとGPU確認

- \$ sudo dnf install nvidia-driver-cuda
- \$ nvidia-smi

```
tmiyahar@AL10CUDA:~$ nvidia-smi
Sun Aug 23 18:40:36 2026

+-----+
| NVIDIA-SMI 610.57.04                  KMD Version: 610.57.04    CUDA UMD Version: 13.3   |
+-----+
+-----+
| GPU  Name                Persistence-M | Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |      Memory-Usage | GPU-Util  Compute M. |
|                                           | MIG M.         |
+-----+
|  0  NVIDIA RTX A1000      Off          | 00000000:00:10.0  On  |          N/A         |
| 30%   51C    P8          N/A / 50W     |  98MiB / 8188MiB |    0%      Default   |
|                                           |                  | N/A         |
+-----+

+-----+
| Processes: |
| GPU   GI    CI          PID    Type    Process name                  GPU Memory |
|      ID    ID                          |              | Usage      |
+-----+
|  0   N/A   N/A         1982     G   /usr/bin/gnome-shell          67MiB |
|  0   N/A   N/A         2283     G   /usr/bin/Xwayland             2MiB  |
+-----+

tmiyahar@AL10CUDA:~$
```


CUDAのインストール

- `$ sudo dnf install cuda`
- CUDA関連の各種パッケージがインストールされる
- ダウンロードされるパッケージの合計サイズが大きいのので注意

LM Studioのインストール

- LM Studioのダウンロードページへ
 - <https://lmstudio.ai/download>
 - \$ wget
<https://lmstudio.ai/download/latest/linux/x64?format=ApplImage>
- ApplImage形式で提供
 - 実行にはFUSEが必要なので事前にインストール
 - \$ sudo dnf install fuse
 - ダウンロードしたファイルに実行権限を付与
 - \$ chmod +x LM*
 - ダブルクリック、またはコマンドラインから実行

実行環境の確認

- LM Studioの環境設定の呼び出し
- 「Hardware」の確認
 - Memory Capacity、GPUを確認
- 「Runtime」の確認
 - Engines & Frameworksが最新かを確認
 - Updateがある場合にはアップデートしておく
 - Runtime Selectionsで適切なものを選んでるかを確認

Hardware

The screenshot shows the 'Hardware' settings page. On the left is a sidebar with navigation options: Settings (General, Appearance, Developer, Chat, Model Defaults, Integrations, LM Link), System (Runtime, Hardware), and Account (Signed out). The 'Hardware' section is selected. The main content area is titled 'Hardware' and includes a 'Copy Info' button and a 'Community' link. It is divided into several sections: 'CPU' (marked as compatible) showing architecture (x86_64, AVX2, AVX); 'Memory Capacity' showing RAM (15.36 GB) and VRAM (7.65 GB); 'GPUs' (1 GPU detected with CUDA) showing an NVIDIA RTX A1000 with a toggle switch set to 'ON'; 'Limit Model Offload to Dedicated GPU Memory' (toggle set to 'OFF'); 'Offload KV Cache to GPU Memory' (toggle set to 'ON'); 'Resource Monitor' showing RAM + VRAM at 0 GB and CPU at 0.00%; and 'Guardrails' (Model loading guardrails).

Settings

- General
- Appearance
- Developer
- Chat
- Model Defaults
- Integrations
- LM Link

System

- Runtime (Ctrl + ⬆ + R)
- Hardware (Ctrl + ⬆ + H)**

Hardware [Copy Info] [Community] [X]

CPU ? ✓ Compatible

Architecture: x86_64 AVX2 AVX

Memory Capacity ?

RAM: 15.36 GB

VRAM: 7.65 GB

GPUs ? [Reset to default] [Open in new window]

1 GPU detected with CUDA

NVIDIA RTX A1000
VRAM Capacity: 7.65 GB • CUDA • deviceId: 0 [OFF] **[ON]**

Limit Model Offload to Dedicated GPU Memory [OFF] [ON]

OFF: Allow model weights to offload to shared memory if dedicated GPU memory is full

Offload KV Cache to GPU Memory ? [OFF] **[ON]**

Resource Monitor ?

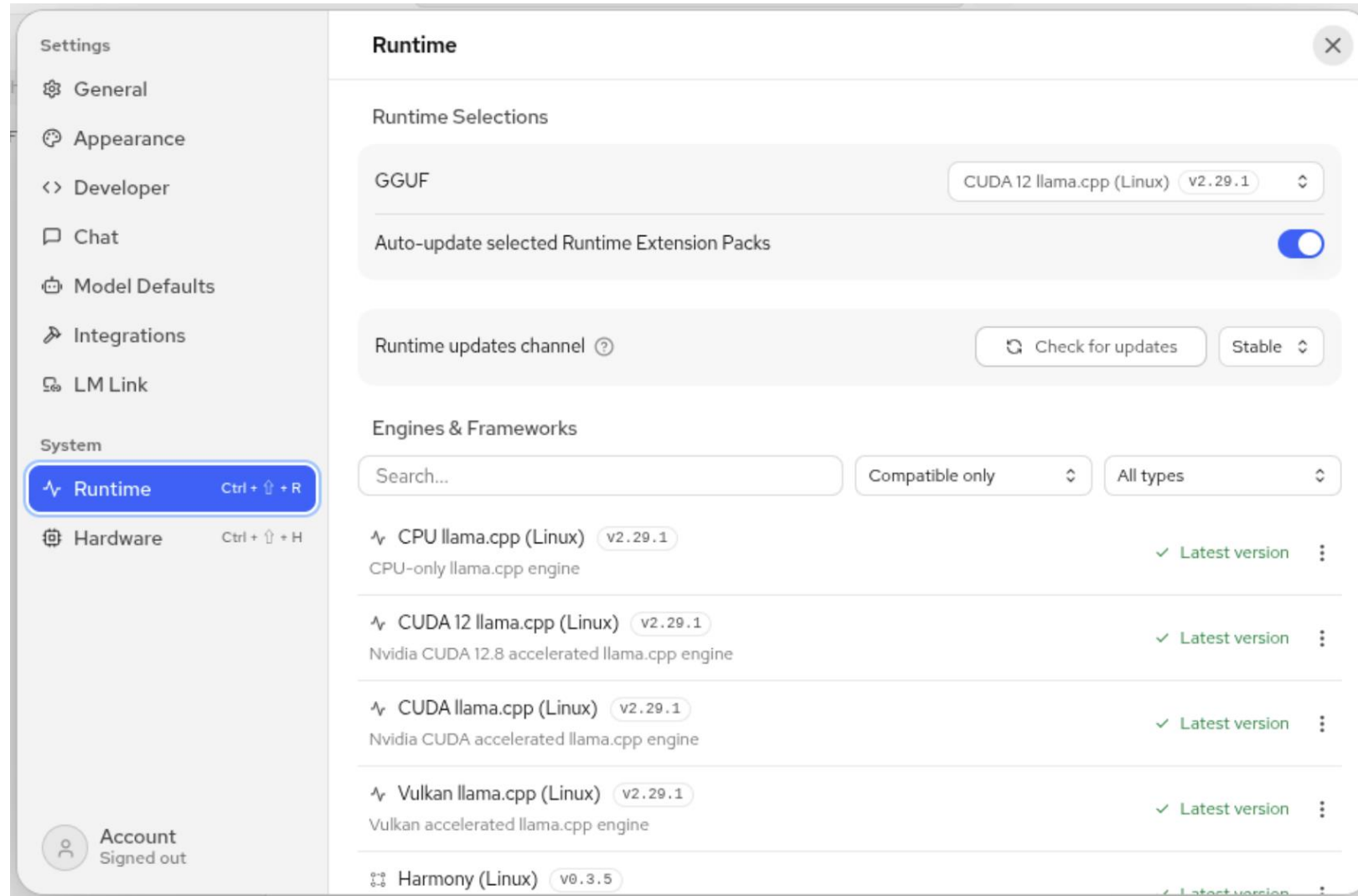
RAM + VRAM: 0 GB CPU: 0.00%

Guardrails ?

Model loading guardrails ?

Account
Signed out

Runtime



Runtimeについて

- 実行環境に合わせて色々な選択肢
 - CUDA：NVIDIA GPU環境
 - ROCm：AMD GPU環境
 - CPU：CPU環境
 - Vulkan：OpenGLの後継API
 - Metal、MLX：macOS用

Model Search

The screenshot displays the Hugging Face Model Search interface. On the left, a sidebar shows a list of models under the 'Staff picks' tab, sorted by 'Best Match'. The models listed are:

- Qwen3.8 27B** (8 days ago): State-of-the-art laptop size model capable of reasoning, tools calling, and image understanding.
- Bonsai 27B** (22 days ago): Bonsai is a 27B model, but takes only about 4GB.
- Gemma 4 12B QAT** (78 days ago): Gemma 4 12B optimized with Quantized Aware Training.
- Gemma 4 26B A4B QAT** (78 days ago): Gemma 4 26B A4B optimized with Quantization-Aware Training.
- Gemma 4 31B QAT** (78 days ago): Gemma 4 31B optimized with Quantization-Aware Training.
- Gemma 4 12B** (80 days ago): The new Gemma 4 12B Unified reasoning model.
- Qwen3.6 27B** (122 days ago): Dense 27B Qwen 3.6 prioritizes stability and reasoning.

On the right, the detailed view for the **qwen/qwen3.8-27b** model is shown. It includes the following information:

- Stats:** 872,724 downloads, 55 stars, last updated 5 days ago. It is a Staff Pick.
- Description:** State-of-the-art laptop size model capable of reasoning, tools calling, and image understanding.
- Metadata:** PARAMS: 27B, ARCH: qwen35, DOMAIN: llm, FORMAT: GGUF, MLX.
- Capabilities:** Vision, Tool Use, Reasoning.
- Download Options:** A dropdown menu shows 'GGUF' selected for 'Qwen3.8 27B' in 'Q4_K_M' format, resulting in a 17.74 GB file. A red warning message states 'Likely too large'. A 'Download' button is available.
- README:**
 - Qwen3.8 27B**
 - Qwen3.8 27B is a compact, deployment-friendly dense vision-language model built on the Qwen3.5 architecture. It delivers stronger performance across coding, professional work, research, and long-horizon agentic tasks.
 - Highlights**

モデルのダウンロード

- Model Search画面を呼び出し
 - ウィンドウ左側の虫眼鏡アイコン
- GPUメモリ、メインメモリの容量などを考慮してモデルを選択
 - モデルサイズ < GPUメモリかどうか (Full GPU Offload)
 - モデルサイズ < GPU + メインメモリかどうか (Partial GPU Offload)

LLMモデルについて

- GGUFフォーマット
 - Georgi Gerganov氏が作成
 - 元々GGMLだったが拡張性の高いフォーマットに
- MLXフォーマット
 - Apple Silicon用に最適化
- Q○
 - 量子ビットを減らすQuantizeしたもの
 - 4ビット（Q4）ぐらいが良いが、8ビット（Q8）も
- E○B
 - 効率的な、効果的なモデル
- A○B
 - MoEアーキテクチャのアクティブパラメータ数
- QAT
 - 量子化認識トレーニングモデル

LLMモデルの機能

- Reasoning
 - 論理的推論を行ってくれる
- Tool Use
 - 開発ツールなどと連携して動作できる
- Vision
 - 画像入力から推論できる

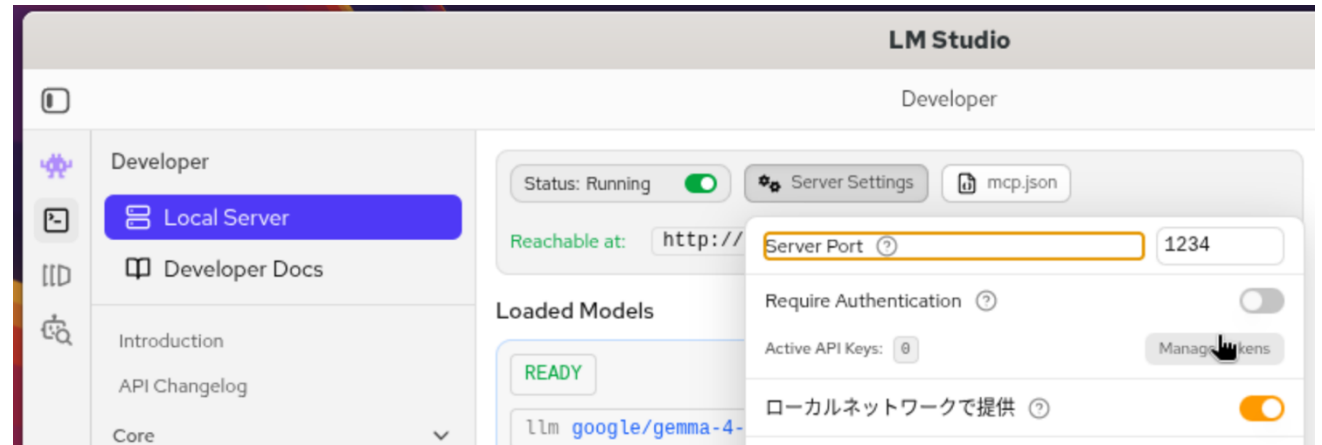
モデルのロード

- Chatを作成し、画面上部中央からロード
 - 切り替える毎にモデルをロードし直し
- Developer画面からモデルをロード
 - 同時に複数のモデルをロード可能
 - ロードされているモデルの分だけメモリを消費
 - Chatで簡単にモデル切り替え可能
 - Serverを起動し外部に対してAPIを提供

AIコーディングしてみよう

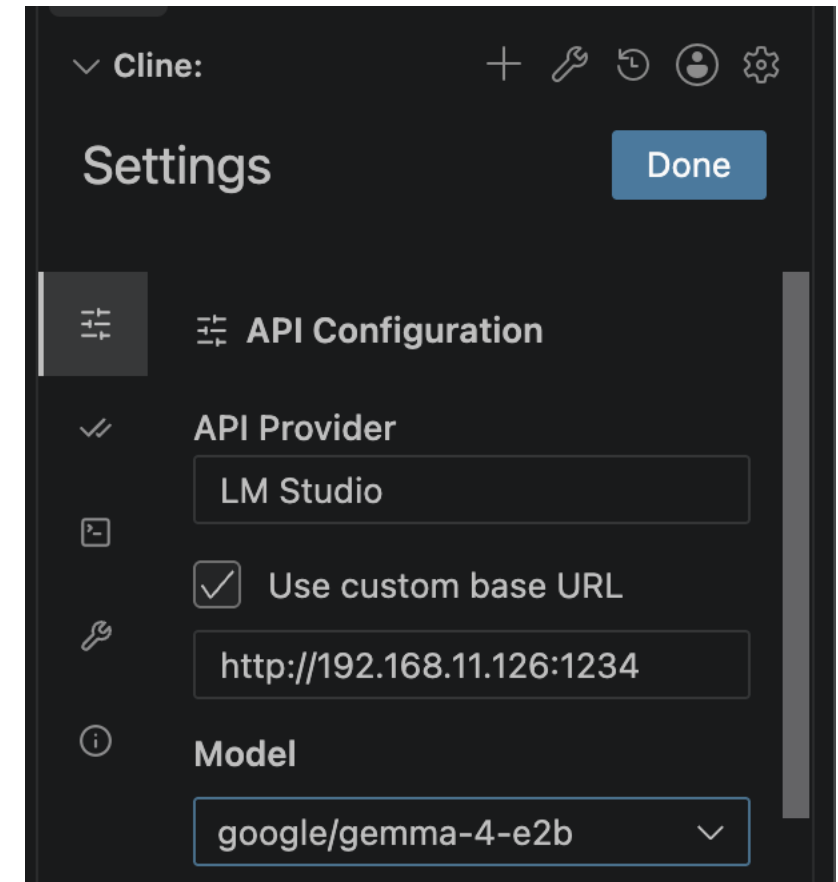
VS Code + Cline + LM StudioでAIコーディング

- LM StudioでLLM Serverを動作
 - Developerモードで起動
 - Server Settingsで「ローカルネットワークで提供」をオン
 - ファイアウォールの設定を変更
 - `$ sudo firewall-cmd --permanent --add-port=1234/tcp`
 - `$ sudo firewall-cmd --reload`



Clineプラグインの設定

- VS CodeにClineプラグインをインストール
- API設定でLM Studioに接続
 - 同じマシンで動作させていれば簡単
 - 別のマシンの場合カスタムURLで
 - LLMサーバーのファイアウォールを通せるようにするよう注意
 - 接続するとモデルが選択できます



Clineの使用

- Planモードで仕様を検討
 - 細かい仕様が聞いてくるので答えてあげる
 - ファイルの書き込みなどはできない
- Actモード
 - 実装を進めてくれる
 - ファイルの書き込みに失敗するなどストレスが多い

今のところの評価

- あまり複雑な開発は行わせられない？
- ファイルの保存が苦手
 - index.htmlを1度は作れるが、修正版を上書き保存できない
 - なんとか保存させようとするループする
- 小さいモデルの限界？
 - 小さいモデルでも性能が高いものが出てきている
 - 仕様の検討などはまだまだ大きいモデルが有利
 - モデルを大きくするとスループット性能は落ちる
 - 大きめのプログラムになるとコンテキスト長を長くする必要がある
 - GPUメモリ容量との兼ね合いが大きくなる