CLI Reference

- Run a model
- Launch integrations
  - Supported integrations
  - Examples
  - Multiline input
  - Multimodal models
- Generate embeddings
- Ask a decision model
- Speak with a speech model
- Download a model
- Remove a model
- List models
- Sign in to Ollama
- Sign out of Ollama
- Create a customized model
- List running models
- Stop a running model
- Start Ollama

Run a model

```
ollama run gemma4
```

Launch integrations

```
ollama launch
```

Configure and launch external applications to use Ollama models. This provides an interactive way to set up and start integrations with supported apps.

Supported integrations

- **OpenCode** - Open-source coding assistant
- **Claude Code** - Anthropic's agentic coding tool
- **Codex** - OpenAI's coding assistant
- **VS Code** - Microsoft's IDE with built-in AI chat
- **Droid** - Factory's AI coding agent

Examples

Launch an integration interactively:

```
ollama launch
```

Launch a specific integration:

```
ollama launch claude
```

Launch with a specific model:

```
ollama launch claude --model qwen3.5
```

Configure without launching:

```
ollama launch droid --config
```

Multiline input

For multiline input, you can wrap text with `"""`:

```
>>> """Hello,
... world!
... """
I'm a basic program that prints the famous "Hello, world!" message to the console.
```

Multimodal models

```
ollama run gemma4 "What's in this image? /Users/jmorgan/Desktop/smile.png"
```

Generate embeddings

```
ollama run embeddinggemma "Hello world"
```

Output is a JSON array:

```
echo "Hello world" | ollama run nomic-embed-text
```

Ask a decision model

```
ollama run nimble --ask "Is a refund being requested?" < ticket.txt

ollama run nimble --ask "Which team should handle this?" \
  --choice billing="Payments and refunds" \
  --choice technical="Software problems" < ticket.txt

ollama run nimble --ask "How urgent is this?" \
  --level "Routine" \
  --level "Soon" \
  --level "Immediate" < ticket.txt
```

The state is piped in or given after the model name, as text or JSON. Each `--ask` question is answered yes or no, unless it's followed by labels to choose from (`--choice`) or levels to score it on (`--level`). See Decision.

Speak with a speech model

```
ollama run phonon "Hello from Ollama."

ollama run phonon --voice Zoey "Hi, this is Zoey."

ollama run phonon "Save this to a file." > speech.wav
```

The audio plays through your speakers, or is written as a WAV file when the output is redirected. Run it without text to speak each line you type. See Speech.

Download a model

```
ollama pull gemma4
```

Remove a model

```
ollama rm gemma4
```

List models

```
ollama ls
```

Sign in to Ollama

```
ollama signin
```

Sign out of Ollama

```
ollama signout
```

Create a customized model

First, create a `Modelfile`

```
FROM gemma4
SYSTEM """You are a happy cat."""
```

Then run `ollama create`:

```
ollama create -f Modelfile
```

List running models

```
ollama ps
```

Stop a running model

```
ollama stop gemma4
```

Start Ollama

```
ollama serve
```

To view a list of environment variables that can be set run `ollama serve --help`

□□□ #2
□ □□□ □□□ 2026-10-31 13:17:01 UTC
□ □□□ □□□ 2026-10-31 13:17:02 UTC