Ollama 0.16-0.17: Everything That Changed in Two Weeks

February 23, 2026 · by Mark Bartlett

Download this guide as PDF

> **Quick Answer:** Ollama shipped five releases in two weeks (0.16.0 through 0.17.0, Feb 12-22 2026). The highlights: up to 40% faster prompt processing on NVIDIA via a rewritten inference engine. KV cache quantization to 8-bit (halves cache memory). `ollama launch openclaw` for one-command OpenClaw setup. Experimental image generation on macOS. MLX runner expands to Gemma 3, Llama, and Qwen 3. AMD RX 9070 (RDNA 4) support. Web search API with 100 free searches/day. The performance numbers are real: gemma3:12b at 128k context went from 52 to 85 tok/s on an RTX 4090.

📚 **Related:** Run Your First Local LLM · Ollama vs LM Studio · llama.cpp vs Ollama vs vLLM · Planning Tool

Ollama pushed five releases between February 12 and February 22, 2026. That's a pace that makes sense only if you look at what they're building toward: a single tool that handles local inference, cloud fallback, coding tool integration, image generation, and web search.

Here's every meaningful change, organized by what actually matters for running local AI.

Release Timeline

Version	Date	Headline
0.16.0	Feb 12	`ollama launch` expansion, GLM-5, MiniMax-M2.5
0.16.1	Feb 12	Installer fixes, image gen timeout fix
0.16.2	Feb 14	Cloud model toggle (`OLLAMA_NO_CLOUD=1`), web search for cloud models
0.16.3	Feb 19	`ollama launch cline` , MLX runner adds Gemma 3/Llama/Qwen 3
0.17.0	Feb 21	New inference engine, up to 40% faster, KV cache q8, OpenClaw onboarding, RDNA 4

The New Inference Engine (0.17.0)

This is the big one. Ollama 0.17 rewrites the internal scheduling and memory management layer.

Performance

Test	Hardware	Before	After	Improvement
Long context (gemma3:12b, 128k)	RTX 4090	52.02 tok/s	85.54 tok/s	+64%
Image input (mistral-small3.2, 32k)	2x RTX 4090	127.84 tok/s prompt eval	1,380.24 tok/s	+979%
Image input token gen	2x RTX 4090	43.15 tok/s	55.61 tok/s	+29%

The general headlines: up to 40% faster prompt processing and 18% faster token generation on NVIDIA. Apple Silicon sees 10-15% improvement on prompt processing.

KV Cache Quantization

Ollama 0.17 now supports 8-bit KV cache quantization. The cache that stores key-value pairs for previous tokens gets compressed to half its usual memory footprint with minimal quality impact.

Why this matters: the KV cache is the thing that eats your VRAM as conversations get longer. At 128K context with a 12B model, the cache alone can take several gigabytes. Halving that means either longer conversations in the same VRAM budget or room for a larger model.

Better Memory Management

Instead of estimating memory requirements, the new scheduler measures exact memory needs. Models schedule more effectively across multiple GPUs, and NVIDIA's `nvidia-smi` output now aligns with `ollama ps` (transparent reporting).

The scheduler also prevents loading more layers than VRAM can handle, which significantly reduces OOM crashes.

Auto-Context on macOS/Windows

The desktop apps now default context length based on available memory. If you have 16GB of unified memory on a Mac, Ollama picks a context length that won't OOM rather than using the model's maximum and hoping for the best.

`ollama launch` (0.15-0.16)

A single command that sets up coding tools with local or cloud models. No environment variables, no config files.

Supported Tools

```
ollama launch claude       # Claude Code
ollama launch opencode     # OpenCode
ollama launch codex        # Codex
ollama launch droid        # Droid
ollama launch cline        # Cline CLI (added 0.16.3)
ollama launch pi           # Pi (added 0.16.0)
ollama launch openclaw     # OpenClaw (improved 0.17.0)
```

How it works: Ollama exposes a local API that implements an Anthropic-compatible messages endpoint. The launch command configures the chosen tool to point at that endpoint, presents a model picker, and launches the tool. The `--config` flag sets up the config without launching (useful for custom workflows).

Recommended Models for Coding

Model	VRAM Needed	Notes
glm-4.7-flash	~23 GB	Local, strong for coding
qwen3-coder	Varies by size	Local
gpt-oss:20b	~12 GB	Local
glm-4.7:cloud	None (cloud)	Free tier
qwen3-coder:480b-cloud	None (cloud)	Paid

Ollama recommends 64,000-token context for optimal coding performance. That means roughly 23GB VRAM for local models at full context.

OpenClaw One-Command Setup (0.17.0)

`ollama launch openclaw` now handles the full OpenClaw onboarding:

1. Checks if onboarding has been completed
2. Runs the OpenClaw onboarding wizard if needed
3. Configures Ollama as the inference provider
4. Installs the gateway daemon
5. Sets your selected model as the primary
6. Starts the gateway in the background
7. Opens the OpenClaw TUI (text user interface)

Previously, connecting OpenClaw to Ollama required manual provider configuration, daemon setup, and model selection. Now it's one command.

For more on OpenClaw itself, see the setup guide and how OpenClaw works.

Image Generation (macOS Only, Experimental)

Ollama added image generation in January 2026. Currently macOS only, with Windows and Linux planned.

Supported Models

Model	Size	License	Minimum VRAM
Z-Image Turbo (Alibaba)	6B	Apache 2.0	12-16 GB
FLUX.2 Klein 4B (Black Forest Labs)	4B	Apache 2.0	~13 GB
FLUX.2 Klein 9B	9B	Non-commercial	More

```
ollama run x/z-image-turbo "a shiba inu wearing a space helmet"
ollama run x/flux2-klein "a neon cyberpunk cityscape"
```

Inside the interactive session:

- `/set width 1024` and `/set height 768` for output dimensions
- Inline preview works in Ghostty and iTerm2
- Images save to the current working directory

Note: Image generation models had bugs in 0.16.0 and 0.16.1. Fixed in 0.16.2.

MLX Runner Expansion (0.16.0-0.16.3)

Ollama has been building a native MLX runner for Apple Silicon alongside its existing llama.cpp/ Metal backend. Model coverage expanded through the 0.16.x cycle:

Architecture	Added In
GLM-4.7-Flash	0.16.0
Gemma 3	0.16.3
Llama	0.16.3
Qwen 3	0.16.3

External benchmarks show standalone MLX can reach ~230 tok/s on M2 Ultra compared to Ollama's llama.cpp backend at ~20-40 tok/s for comparable models. The built-in MLX runner is still expanding model coverage, but it's the path toward closing that gap.

The MLX runner is not yet the default for all models. More architectures are being added incrementally.

Web Search API

Launched September 2025, relevant here because it integrates with the `ollama launch` tools.

Setup

1. Create a free Ollama account
2. Generate an API key at `https://ollama.com/settings/keys`
3. Set `OLLAMA_API_KEY` environment variable

Usage

```
# REST API
curl -X POST https://ollama.com/api/web_search \
  -H "Authorization: Bearer $OLLAMA_API_KEY" \
  -d '{"query": "RWKV-7 benchmarks", "max_results": 5}'
```

Free tier: 100 searches/day. Recommended minimum 32K context for web search agents.

Pricing

Tier	Price	For
Free	$0	Light usage, quick questions
Pro	$20/month	RAG, document analysis, coding
Max	$100/month	Coding agents, batch processing

All tiers include unlimited local deployment. The pricing is for cloud inference and web search quotas only.

AMD RDNA 4 Support (0.17.0)

The AMD Radeon RX 9070 series (RDNA 4) is now supported. If you bought one of these cards for local AI, Ollama 0.17 is the first version that works with it out of the box.

Anthropic API Compatibility

Ollama implements the Anthropic Messages API at `/v1/messages`. Any tool that expects the Anthropic API (notably Claude Code) can use local Ollama models instead.

```
# Manual setup for Claude Code
export ANTHROPIC_BASE_URL=http://localhost:11434
export ANTHROPIC_API_KEY=""
export ANTHROPIC_AUTH_TOKEN=ollama
claude --model qwen3-coder

# Or just:
ollama launch claude
```

Supported: message streaming, multi-turn, system prompts, tool calling, vision (base64 only), extended thinking parameters.

Not supported: token counting, prompt caching, batch API, URL-based images, citations.

How to Update

macOS/Windows: Ollama auto-downloads updates. Click the menubar/taskbar icon and select “Restart to update.”

Linux: Re-run the install script:

```
curl -fsSL https://ollama.com/install.sh | sh
```

Or update the binary directly:

```
sudo curl -L https://ollama.com/download/ollama-linux-amd64 -o /usr/bin/ollama
sudo chmod +x /usr/bin/ollama
```

Docker:

```
docker pull ollama/ollama:latest
```

Verify version:

```
ollama -v
```

There's still no dedicated `ollama update` or `ollama upgrade` CLI command on Linux. The feature request is open.

Cloud Model Toggle

If you're privacy-conscious, 0.16.2 added `OLLAMA_NO_CLOUD=1` to disable all cloud model routing. Set it as an environment variable or toggle it in the desktop app settings. When enabled, only local models appear in model selection.

Breaking Changes in 0.17

Two things to watch if you're running Ollama in production:

1. **API endpoint consolidation.** The `/api` and `/v1` endpoints moved to the main domain. If you used a separate subdomain or secondary host for API calls, update your URLs.
2. **Authentication change.** JWT tokens replaced by OpenAI-compatible API keys. If you were using JWT-based auth, migrate to API key auth.

Both are one-time fixes. The upgrade auto-migrates existing models and configurations.

Bottom Line

The 0.16-0.17 cycle moves Ollama from "local inference server" toward "local AI platform." The new inference engine is measurably faster — gemma3:12b at 128k went from 52 to 85 tok/s. KV cache quantization is a practical win for anyone running long-context workloads on limited VRAM. And `ollama launch` saves enough setup hassle to be worth learning.

The expanding scope (image gen, web search, cloud tiers, OpenClaw onboarding) is worth watching. Ollama is betting that one tool should handle everything from pulling a model to deploying a full AI agent. Whether that's the right architecture or an overstuffed Swiss army knife depends on how well they execute.

For now: update to 0.17, enjoy the speed boost, and try `ollama launch` with your preferred coding tool. The performance improvement alone justifies the upgrade.

📚 **Setup:** Run Your First Local LLM · Open WebUI Setup · Ollama Troubleshooting

📚 **Hardware:** VRAM Requirements · AMD vs NVIDIA for Local AI · Planning Tool

📚 **Agents:** How OpenClaw Works · OpenClaw Setup Guide · OpenClaw Token Optimization

---

Source: https://insiderllm.com/guides/ollama-0-17-new-features/

Free guides for running AI locally