Ollama 0.30.0: What's New, What's Faster, What Breaks on Upgrade

June 2, 2026 · by Mark Bartlett

Download this guide as PDF

> **Quick Answer:** Been putting off updating Ollama? The current build (v0.30.0) may be a lot further ahead than you think — I just jumped 13 versions in one go on my RTX 3090 box. The good news: the upgrade was clean and the API never skipped a beat. The thing to know: your first model run kicks off a one-time storage migration that looks like a freeze but isn't. This guide covers what actually changed (flash attention now turns on by default for modern Qwen and Gemma, plus a new engine and broader Hugging Face model support), the one upgrade gotcha that'll make you think it's broken, which models still aren't supported, and whether you should upgrade now.

📚 **Related:** Ollama Troubleshooting Guide · Open WebUI Connection Fixes · Ollama API Connection Refused · Qwen 3.6 Complete Guide · Qwen 3.5 Cheat Sheet

I just jumped 13 versions on my Ubuntu + RTX 3090 box. Ollama 0.17.5 → 0.30.0, in one go.

If you've been holding off on updating Ollama for "a while," the gap between where you are and where the current build is may be larger than you think. The good news: the upgrade was clean and the API still answers on port 11434 like nothing happened. The interesting news: a few things have shifted under the hood that are worth knowing before you run your first model on the new build.

Here's what's actually different in 0.30.0, what to expect on first launch, and whether you should upgrade right now.

What's actually new in 0.30.0

llama.cpp integrated alongside MLX. The headline architectural change. On Apple Silicon, MLX has been Ollama's engine for a while; v0.30.0 layers llama.cpp into the picture too, "for improved compatibility and performance" per the release notes, plus broader hardware support. On Linux and Windows boxes (where MLX was never available) this is the path forward for faster NVIDIA inference and broader model coverage. On Mac you're still MLX-first; the llama.cpp integration matters more once you step off Apple Silicon.

Flash attention auto-enabled at runtime for supported architectures. This is the change most people will actually notice. On supported model families — `qwen3`, `qwen3moe`, `qwen3vl`, `qwen3vlmoe`, `gemma3`, `gpt-oss`, `mistral3` — running on Ampere-or-newer NVIDIA GPUs (RTX 30-series and up) or RDNA3-or-newer AMD GPUs (RX 7000-series and up), flash attention engages automatically without you setting any flag. I checked the config source: `OLLAMA_FLASH_ATTENTION` still defaults to `false`; it's now an explicit override, not the master switch. If you're on a supported model + GPU combination, the engine just turns it on. Should improve throughput on those combinations per the release notes; I haven't benchmarked it myself yet, and real tok/s lands in a follow-up.

Broader model support: GGUF from Hugging Face plus fine-tunes. The 0.30.0 notes call out wider compatibility with GGUF-based models pulled directly from Hugging Face, including your own fine-tuned ones. If you've been juggling Modelfile boilerplate for community GGUFs, this is a real ergonomics win.

Cloud models have matured. Cloud-routed models (the `:cloud`-tagged ones, run on Ollama's hosted infrastructure rather than your local GPU) have been quietly building out across the 0.2x line. By 0.30.0 they're production-shaped: there's an `OLLAMA_API_KEY` env var for auth (per the Ollama Cloud docs), and the local server acts as an authenticated proxy. If you wrote cloud models off the first time you tried them, they're worth a second look.

systemd service auto-created on Linux. Confirmed on my Ubuntu box today: the v0.30.0 install script sets up `/etc/systemd/system/ollama.service` automatically. You don't need to manually `ollama serve` and `tmux` it; it runs as a managed service. `sudo systemctl status ollama` is your new first check, and the systemd-edit pattern (`sudo systemctl edit ollama.service`) is how you persist env vars. Worth knowing because plenty of older guides still walk people through manual `ollama serve` workflows that no longer match the current install. The Ollama troubleshooting guide and the API connection-refused guide both reflect the systemd-first pattern now.

The upgrade experience, firsthand

I went from 0.17.5 to 0.30.0 directly. No intermediate hops. Here's what actually happened on my Ubuntu + RTX 3090 setup.

The upgrade itself: clean. Standard `curl -fsSL https://ollama.com/install.sh | sh`. The install script handled the systemd service transition without me intervening.

First model run: storage migration. This is the thing nobody mentions, and it's worth setting expectations. The first time I ran a model on 0.30.0, Ollama did a one-time storage migration to a

content-addressable format — sat there churning for a few minutes, and I almost reached for `ollama ps` to check what was happening. Models survived the upgrade fine; nothing got re-downloaded. But it isn't instant. Don't panic if your first run feels frozen for a couple of minutes after the version jump.

API and Open WebUI: working as expected. I checked the two things I care about right after the upgrade:

```
# API health check
curl http://localhost:11434/api/tags
# returns the model list cleanly

# Open WebUI health
curl http://localhost:8080/health
# returns ok
```

Both came back clean on first try. The connection contracts have not changed — port 11434 for the Ollama API, port 8080 for Open WebUI inside its container — and the Open WebUI connection fixes guide still applies as written.

Existing systemd config: preserved. I had `Environment="OLLAMA_HOST=0.0.0.0:11434"` set via `systemctl edit ollama.service` from way back. The upgrade left it in place. No re-config needed.

Flash attention: the detail worth getting right

This is where I see the most confusion floating around, so let me lock the specifics:

Aspect	Status in v0.30.0
Default behavior	Auto-enables at runtime for supported architectures
Supported architectures	`qwen3`, `qwen3moe`, `qwen3vl`, `qwen3vlmoe`, `gemma3`, `gpt-oss`, `mistral3`
NVIDIA GPU floor	Ampere (RTX 30-series) or newer
AMD GPU floor	RDNA3 (RX 7000-series) or newer
`OLLAMA_FLASH_ATTENTION` env var	Still defaults `false` in config — explicit override, not the master switch

If you're on a 3090, 4090, or 5090 running Qwen 3.5, Qwen 3.6, or Gemma 4 — basically the bulk of what r/LocalLLaMA is actually running this month — you're getting flash attention for free on 0.30.0. For everything else (older GPUs, Llama, other architectures), the env var is your manual lever and nothing's changed there.

I haven't benchmarked the speedup on my box yet. The release notes say "faster NVIDIA performance" and flash attention on supported architectures should improve throughput, but I'm not going to put a multiplier on it here until I've measured one. I'll publish real numbers from a controlled bench in a follow-up; this article is the engine-change overview, not the speed test. If you want the model-side picture, the Qwen 3.6 guide and Qwen 3.5 cheat sheet both now run flash-attention-by-default on a current GPU + 0.30.0 combo.

Known issues to watch

The 0.30.0 release notes call out three specific things. If you depend on any of them, hold off:

- `laguna-xs.2` **is not yet supported on Windows or Linux.** If you're running that specific model, you're stuck on an older Ollama until support lands.
- `llama3.2-vision` **is not yet supported.** If your stack uses Llama 3.2 vision specifically, this is a hard block. Most other vision-capable models (the Qwen 3.5 small-model vision path, for example) are unaffected.
- `nomic-embed-text` **now converts inputs to lowercase**, per the model card, where prior Ollama versions incorrectly preserved mixed case. This is a behavior change, not a bug fix everyone will like silently. If you have a RAG pipeline that depended on case-sensitive embeddings (relatively rare, but it exists), your embeddings will land slightly differently on 0.30.0. Re-index if you care about exact reproducibility.

Beyond those three, normal upgrade caveats apply: have a downgrade plan (pin via `OLLAMA_VERSION=0.X.Y` in the install script) if anything in your production pipeline breaks, and re-test agent harnesses against the new build before treating it as your new baseline.

Should you upgrade?

Upgrade now if:

- You're on a pre-0.20 build and catching up. The gap is bigger than it looks, and 0.30.0 is the canonical current target.
- You run Qwen 3.x, Gemma 3 or 4, gpt-oss, or mistral3 on Ampere-or-newer NVIDIA or RDNA3-or-newer AMD; you'll benefit from flash-attention-by-default without lifting a finger.

- You want to pull GGUFs straight from Hugging Face without Modelfile gymnastics.
- You're on Linux and tired of manual `ollama serve` patterns; the auto-systemd service is genuinely nicer to live with.

Hold off if:

- Your stack hits `laguna-xs.2` or `llama3.2-vision`. You're explicitly blocked until support lands.
- You have a RAG pipeline pinned to case-sensitive `nomic-embed-text` behavior. Re-index before you migrate, or wait until you're ready to re-index.
- You're running a production agent harness you haven't re-tested against 0.30.0 yet. Not a reason to never upgrade, just a reason to put it through your validation pipeline first.

Bottom line

Thirteen versions in one jump went cleanly, the API contract held, and the parts of my setup I depend on (systemd, ports, Open WebUI, the connection mechanics covered in the Ollama troubleshooting guide and the API connection-refused guide) all still work the same way. The first-launch storage migration is the only "wait, what's it doing?" moment, and it's a one-time event.

The flash-attention-by-default change is the biggest behavioral shift, and it's the right kind of change: opt-out via an explicit env var, supported on the specific architecture + GPU combinations where it's a clean win, no quiet behavior shift on models that aren't on the supported list. If you're running modern Qwen or Gemma on a 30-series-or-newer card, 0.30.0 is a free upgrade.

Benchmarks land in a follow-up. For now: update if your stack isn't in the known-issues list, set expectations for the one-time storage migration, and don't be surprised when `ollama serve` isn't where you left it.

— Mark Bartlett