HEADS UP This site is currently under heavy development.

THE AI TOOLCHAIN

NO. 005

Head

THE WEEK'S TOP PICKS

PUBLISHED SEPTEMBER 18, 2026 · FRIDAYS

// SHARE THIS WEEK ▽ Top 5 ▽ Top Pick

The best tooling updates that shipped this week.

▸ // HOW THESE PICKS ARE MADE SHOW

▾ // THE BRIEF HIDE

What stands out in this week's picks, grouped by what it lets you do. Every tool named links to its pick below.

BUILD

Catch a bad fine-tune before you pay for the whole run

Together AI streams per-step metrics during training and validates the dataset pre-flight, with early stopping and gradient accumulation, so a malformed JSONL or a diverging loss curve surfaces in minutes instead of at the end of a billed job. KTransformers adds RAWINT4 LoRA fine-tuning for Kimi K2.5/K2.6 and BF16 image-text LoRA for Qwen3-VL, keeping large-model adaptation on hardware that could not hold the full-precision weights.

Together AI · KTransformers

DEPLOY

Fit more vectors per node without trading away recall

Weaviate 1.39's 4-bit rotational quantization with centering drops heap usage 45% and improves recall, so the index that needed a bigger instance now fits — and new node-level query admission control sheds load instead of letting one heavy query take the cluster down. LMCache

v0.5.5 makes isolated CUDA IPC the default for multiprocess serving and extends support to seven new accelerators, so KV reuse across requests stops being a single-process trick.

Weaviate · LMCache

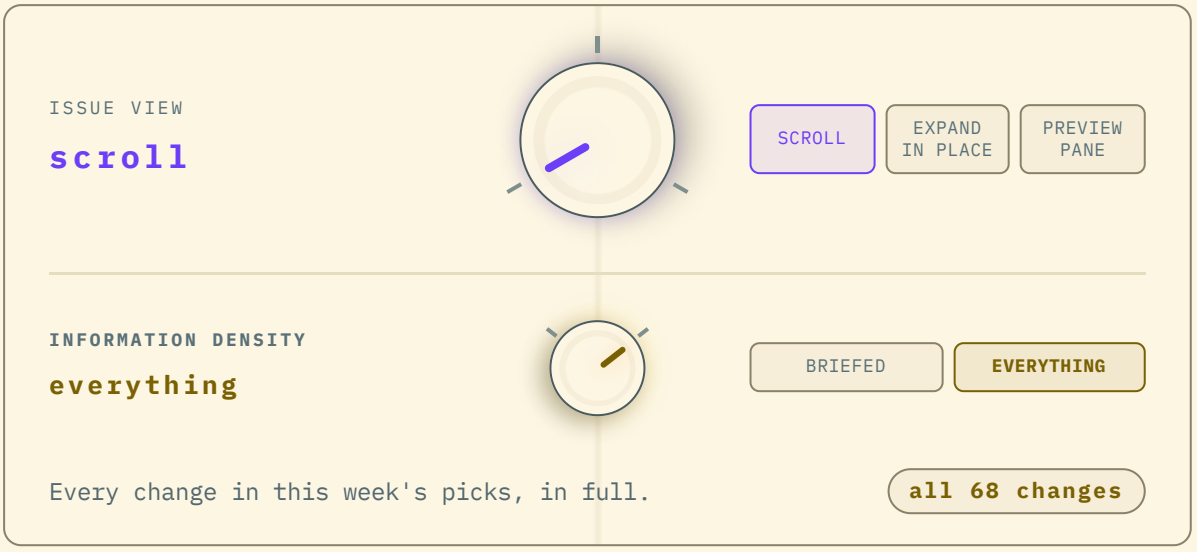
BUILD

Stop shipping prompt and model choices inside your application code

OpenRouter Presets put the model, parameters and prompt config behind a versioned `@preset/your-slug` reference, so changing a production model is a preset bump rather than a deploy, and you can see which version a call used. Cline now runs whole sessions over SSH on a remote host from desktop or SDK, so the agent works where the repo and toolchain actually live instead of on a laptop.

OpenRouter · Cline

VIEW



Do you prefer scroll? 👍 👎

// THE PICKS

01	**Together AI**	AI Model & Data Infrastructure
02	**OpenRouter**	AI Model & Data Infrastructure
03	**Cloudflare Agents**	AI Agent Frameworks
04	**Weaviate**	Vector DBs & RAG
05	**Cline**	AI Coding Agents

01 Together AI

6 RELEASES · 2026-09-10 → 2026-09-17

AI Model & Data Infrastructure

Together AI provides an API platform for running and fine-tuning open-source large language models at scale.

// WHY IT MADE THE LIST

DEPTH 5/5 IMPACT 4/5

Fine-tuning now includes expert-layer LoRA on Mixture-of-Experts models — up to 89% recall on newly learned facts versus 15% with attention-only adapters — alongside live per-step metrics, early stopping with step refunds, and pre-flight dataset validation; preemptible GPU clusters at 50% of on-demand and custom LoRA-adapter serving ship too.

> Together AI's biggest push this window was a major fine-tuning expansion — live per-step metrics, expert-layer LoRA , early stopping, gradient accumulation, dataset inspection, sample weighting, and pre-flight validation — alongside a full canary rollout API for endpoints, preemptible GPU compute across clusters, custom LoRA adapter serving, a new Python SDK v2, and Prometheus /Grafana monitoring for dedicated endpoints.

└──▷ WHAT SHIPPED · 16 FEATURES most completely described first ? what's the number?

01 Preemptible GPU compute for GPU Clusters NEW

100

Introduces a second compute type for Together GPU Clusters billed at a flat 50% of the on-demand rate, sub-hourly, metered every one to two minutes, with no spot-market price fluctuation, for running interruptible workloads at discounted rates within a cluster. Configure it with the `--num-preemptible-gpus` flag on `tg beta clusters create` and `tg beta clusters update`; the Together API exposes `desired_preemptible_gpus` and `allocated_preemptible_gpus` fields to report requested vs. live capacity. Preemption notices arrive as Kubernetes events with `reason=TogetherPreempted` and via `node_lifecycle_events` in the API, giving workloads up to `terminationGracePeriodSeconds` (max 300s) to checkpoint before removal, and preemptible nodes carry the `together.ai/compute-class=preemptible` label so workloads can be targeted with `nodeSelector` while coordinators and serving replicas stay on standard nodes.

Spin up a new cluster with both standard and preemptible GPUs to run batch fine-tuning jobs at half the cost.

```
$ tg beta clusters create \
  --name my-cluster --region us-central-2 --gpu-type RTX_6000_PCI
\
  --cluster-type KUBERNETES --num-gpus 8 --billing-type ON_DEMAND
\
  --num-preemptible-gpus 8 \
  --project-id <project-id>
```

Scale preemptible capacity up on a running cluster to absorb a batch inference burst, then scale it back down when done.

```
$ tg beta clusters update --num-preemptible-gpus 16
```

Pin interruptible worker pods to preemptible nodes while keeping coordinators on standard nodes, and give pods the full drain window to checkpoint on SIGTERM.

```yaml
nodeSelector:
  together.ai/compute-class: preemptible
terminationGracePeriodSeconds: 300
```

— Exact flags, API fields, event names and runnable commands given `snapshot-20260916` `snapshot-20260914` `snapshot-20260911` `launch-20260910-ad82b592`

02 **Canary rollout management API for endpoints** NEW 94

Adds a full rollout lifecycle API: a List Rollouts endpoint with `filter=ROLLOUT_FILTER_ACTIVE` to return only the active rollout, a Create Rollout API that creates a rollout in the pending state without shifting traffic so strategy and metric gates can be reviewed before a separate Start request, a Start Rollout API that begins the configured traffic-shifting workflow, a Get Rollout API returning strategy, lifecycle state, current traffic percentage, step history and metric-gate results, a Pause Rollout API that accepts an optional reason and returns the PAUSING snapshot, a Resume Rollout API that resumes from the current step and traffic split, a Promote Rollout API that sends all live traffic to the target deployment immediately, a Cancel Rollout API that freezes the current split into standing weights and moves the rollout to CANCELED, and a Delete Rollout API for aborted or completed rollouts. Metric-gate support automatically pauses a rollout when the new deployment regresses.

List only the currently active rollout on an endpoint to quickly confirm live traffic state without wading through historical records.

```
$ curl -X GET
'https://api.together.ai/v1/endpoints/<endpoint_id>/rollouts?
filter=ROLLOUT_FILTER_ACTIVE' \
  -H 'Authorization: Bearer <TOGETHER_API_KEY>'
```

— Names every rollout endpoint, states and a runnable curl example `snapshot-20260916`

03 **Expert-layer LoRA for MoE fine-tuning** NEW 85

Adds a `lora_trainable_modules` config to include expert layers in LoRA fine-tuning jobs on Mixture-of-Experts models, enabling up to 89% recall on newly learned facts vs. 15% with attention-only adapters.

Fine-tune a Mixture-of-Experts model with expert-layer LoRA to maximize retention of newly learned knowledge rather than just adapting attention.

```json
{ "lora_trainable_modules": ["attention", "experts"] }
```

— Named config key, concrete recall numbers, and a runnable example `launch-20260911-5550c113`

04 **Configurable effective batch size via gradient accumulation** NEW 80

Adds `gradient_accumulation_steps` config to support arbitrary effective batch sizes (effective batch = `batch_size` x `gradient_accumulation_steps`) regardless of single-pass GPU memory limits.

Train with a large effective batch size on a long-sequence model that cannot fit the full batch in a single GPU pass.

```json
{ "batch_size": 4, "gradient_accumulation_steps": 16 }
```

— Named config key with formula and a runnable example `launch-20260911-5550c113`

05 **Per-example sample weighting in training data** NEW 75

Adds a `weight` field to JSONL training examples for per-example sample weighting, controlling each data point's contribution to the loss across all supported formats and training methods.

Add per-example sample weights in a JSONL training file to up-weight high-quality human-annotated examples over synthetic ones.

```json
{"messages": [{"role": "user", "content": "Explain zero-day exploits."}, {"role": "assistant", "content": "A zero-day exploit..."}], "weight": 3.0}
{"messages": [{"role": "user", "content": "What is phishing?"}, {"role": "assistant", "content": "Phishing is..."}], "weight": 1.0}
```

— Named field with scope and a runnable JSONL example `launch-20260911-5550c113`

06 **Dataset inspection and download before training** NEW

75

Adds dataset inspection: preview tokenized rows (token IDs, token strings, per-token labels, loss spans, truncation status) before a job starts, and retrieve the fully prepared training dataset via the Python SDK or the Download link in the console.

— Names preview fields and two concrete retrieval paths `launch-20260911-5550c113`

07 **Live training metrics across API, CLI, SDK and dashboard** NEW

71

Exposes live per-step training metrics (loss, gradient norm, learning rate) and evaluation metrics through the Together API, CLI, Python SDK, and UI dashboard Metrics tab, with filtering by range and downsampling.

— Names metrics and surfaces; dashboard tab is a clear starting point

`launch-20260911-5550c113`

08 **Pre-flight dataset validation on upload** NEW

70

Adds pre-flight server-side dataset validation that runs immediately on upload and writes results to the file object, covering message structure, tool definitions, tool-call/response pairing, reasoning fields, multimodal content, preference-pair structure, and sample weight types.

— Lists checks covered but no command to trigger or inspect `launch-20260911-5550c113`

09 **Early stopping for fine-tuning jobs** NEW

70

Adds early stopping for fine-tuning jobs: enable it with a validation set and the run halts when validation loss plateaus, retaining the best checkpoint and refunding unused training steps; patience, sensitivity, and warmup are tunable.

— Clear mechanism and benefits but tunable params unnamed as keys `launch-20260911-5550c113`

THINNER COVERAGE BELOW

10 **Multi-run comparison view in dashboard Metrics tab** NEW

50

Adds a multi-run comparison view in the dashboard Metrics tab, allowing several jobs to be charted together on one set of axes while runs are still in flight.

– Clear UI location but no config or API detail given `launch-20260911-5550c113`

---

11 **New fine-tunable model support** NEW 50

Adds fine-tuning support for GLM 5.3, GLM-5.2, GLM-5.1, DeepSeek-V4-Flash-0731, DeepSeek-V4-Flash, Kimi K2.7-Code, Kimi K2.6, Qwen 3.8-27B, Qwen 3.6-35B-A3B, Qwen 3.6-27B, Qwen 3.5-27B, Qwen 3.5-9B, Qwen 3.5-4B, Qwen 3.5-2B, Qwen 3.5-0.8B, Gemma 4-31B, and Gemma 4-26B-A4B.

– Just a model list with no mechanism or usage detail `launch-20260911-5550c113`

---

12 **Prometheus and Grafana monitoring for dedicated endpoints** NEW 50

Supports scraping a metrics endpoint with Prometheus and importing an example Grafana dashboard for dedicated endpoints.

– Names the tools but no endpoint path or config given `snapshot-20260917`

---

13 **Sequence packing control via position_ids** NEW 48

Adds `position_ids` support in the pre-tokenized data format to control the degree of sequence packing, and an unnamed flag to disable packing entirely.

– Names the field but the disabling flag itself is unnamed `launch-20260911-5550c113`

---

14 **Together Python SDK v2** NEW 45

Introduces Together Python SDK v2 with improved type safety and modern architecture, with a migration guide from v1.

– Names the release but no specific API changes listed `snapshot-20260917`

---

15 **Custom LoRA adapter serving on dedicated clusters** NEW 40

Supports uploading and serving custom LoRA adapters sourced from Hugging Face or S3 on dedicated clusters.

– States sources and target but no config keys or commands `snapshot-20260914` `snapshot-20260911`

---

16 **Reserved and on-demand cluster creation** NEW 25

Supports reserved and on-demand cluster creation with connectivity to workloads.

– Bare mention with no config, flag or command named `snapshot-20260916`

GOOD PICK? 👍 👎 GOOD REASON? 👍 👎

02 OpenRouter

5 RELEASES · 2026-08-17 → 2026-09-10

AI Model & Data Infrastructure

OpenRouter is a routing gateway that exposes hundreds of models from many providers behind one OpenAI-compatible API with failover and unified billing.

// WHY IT MADE THE LIST

DEPTH 4/5 IMPACT 5/5

Any model on OpenRouter now gets a hosted, isolated Linux container to run shell commands with stdout/stderr/exit codes returned, plus a Files API to move files in and out — closing the gap of having to build your own sandbox to let a model execute code. Versioned, named Presets referenceable as @preset/slug across apps round it out.

> OpenRouter's biggest addition this window is Presets, versioned LLM call configs callable via `@preset/your-slug`, alongside a hosted shell tool with a companion Files API for sandboxed command execution, and expanded team-level analytics including a queryable Analytics API.

└──▷ WHAT SHIPPED · 3 FEATURES most completely described first ? what's the number?

01 Presets for versioned LLM call configs NEW

87

Presets bundle models, system prompts, provider routing, and sampling parameters into a named, versioned unit referenceable as `@preset/your-slug` in any API call, e.g. `curl https://openrouter.ai/api/v1/chat/completions` with `"model": "@preset/your-slug"`. Updating a Preset from the dashboard propagates the change to every app that references it without requiring a redeploy.

Reference a saved Preset in a chat completions call to apply your versioned model, routing, and sampling config without hardcoding parameters.

```
$ curl https://openrouter.ai/api/v1/chat/completions \
  -H 'Authorization: Bearer <your-api-key>' \
  -H 'Content-Type: application/json' \
  -d '{"model": "@preset/your-slug", "messages": [{"role":
"user", "content": "Hello"}]}'
```

— Names exact slug syntax and includes a runnable call `snapshot-20260915`

02 Hosted shell tool and Files API NEW

83

A shell server tool gives any model on OpenRouter a hosted, isolated Linux container; the model receives `stdout`, `stderr`, and exit codes from commands it runs, invoked

with `{"tools": [{"type": "shell"}]}` on the Responses and Messages APIs (beta). A companion Files API moves files into and out of the shell container, enabling download of files written during a run.

Run an arbitrary shell command through a model on the Messages API and retrieve stdout — useful for automating code execution or environment probing via LLM.

```
$ curl https://openrouter.ai/api/v1/messages \
  -H 'Authorization: Bearer <your_api_key>' \
  -H 'Content-Type: application/json' \
  -d '{
    "model": "<model_id>",
    "tools": [{"type": "shell"}],
    "messages": [{"role": "user", "content": "Run `uname -a` and
show me the output."}]
  }'
```

— Names APIs, tool schema and includes runnable curl example `snapshot-20260916` `snapshot-20260913` `snapshot-20260912`

THINNER COVERAGE BELOW

03 **Team analytics API and dashboard enhancements** NEW 50

Adds an Analytics API so teams can query spend and usage data from the terminal, per-model spend breakdowns scoped to team-level usage in the analytics dashboard, saveable analytics charts so practitioners can preserve views they rebuild repeatedly, and click-through from any chart bar directly into the underlying request logs.

— Named surfaces but no endpoint or flag detail given `snapshot-20260911`

GOOD PICK? 👍 👎 GOOD REASON? 👍 👎

03 Cloudflare Agents

3 RELEASES · 2026-09-11 → 2026-09-18

AI Agent Frameworks

The Cloudflare Agents SDK builds stateful agents on Workers and Durable Objects, with persistent state, scheduling, WebSockets, and MCP server support.

// WHY IT MADE THE LIST

DEPTH 5/5 IMPACT 4/5

Adds `agents/sessions`, durable conversation storage with message branches, compaction overlays, byte-budgeted streamed reads and full-text search — a real lift on what an agent can hold — plus a durable alarm-driven job queue and RoutedAgents for one Durable Object per chat.

> Cloudflare Agents' headline change this window is a durable job queue built into Lifecycle, paired with a new `agents/sessions` conversation-storage module that both AIChatAgent and Think migrated onto with atomic, crash-safe stream settlement. The window also adds RoutedAgents, a dynamicAgents facade, an `agents/context` prompt module, and a separate `agents/queue` capability, alongside numerous breaking migrations of legacy schedule, queue, and message tables.

└──▷ **WHAT SHIPPED · 12 FEATURES** most completely described first ? what's the number?

01 Sessions-backed message storage and rollover stream blocks

100

BREAKING

AIChatAgent (`@cloudflare/ai-chat`) and Think (`@cloudflare/think`) both migrate message storage onto `agents/sessions` : `cf_ai_chat_agent_messages` and Think's `assistant_messages` / `assistant_compactions` / `assistant_config` rows migrate into the Sessions chain on first wake, dropping the legacy tables (rollback loses the migrated conversation). `hydrationByteBudget` becomes a hard ceiling raised to 32 MiB (from 24 MiB for Think), charging each row its full stored size; `maxPersistedMessages` now counts stored rows. The flat `cf_agents_stream_chunks` append log is replaced by mutable rollover blocks that grow to 256 KB before opening a new row, and writer.close({ commit, discard }) / error(reason, {...}) settle a stream and delete its rows atomically in one SQLite transaction so a crash never leaves an orphaned state. The `get-messages` response is streamed rather than fully serialized, and oversized messages split across continuation rows and reassemble transparently on read.

Read conversation history only after the agent wakes, now that `this.messages` is populated in `onStart` rather than the constructor.

```typescript
import { AIChatAgent } from '@cloudflare/ai-chat';

export class MyChatAgent extends AIChatAgent<Env> {
  async onStart() {
    await super.onStart();
    // this.messages is now populated — safe to inspect history
here
    console.log(`Resuming with ${this.messages.length}
messages`);
  }
}
```

Observe in-flight chat streams from another consumer on the same Durable Object using the newly exposed `readonly streams` capability.

```typescript
import { AIChatAgent } from '@cloudflare/ai-chat';

export class MyChatAgent extends AIChatAgent<Env> {
  async onMessage(message: string) {
    // Tap the live stream for the current turn from within the
same DO
    const reader = await this.streams.read('<stream-id>');
    for await (const chunk of reader) {
      console.log('chunk', chunk);
    }
  }
}
```

— Full mechanism, exact tables/limits, and two runnable examples

`@cloudflare/ai-chat@0.12.0` `@cloudflare/think@0.18.0`

02 **Durable job queue in Lifecycle** NEW 95

Adds a durable job queue to Lifecycle, driven as an alarm event loop: this.lifecycle.jobs.push({ id, fn, time }) schedules serializable callbacks, and a capability's onJob({ job }: LifecycleJobContext) handler returns nothing (complete), `{ rescheduleAt }` (suspend), or `'yield'` (wake immediately). Adds onJobError() for terminal application job failures and onAlarmMemoryLimit() as a host hook for alarm-memory-limit policy. Existing `cf_agents_schedules` rows migrate into the new `cf_agents_jobs` queue on startup and the legacy table is dropped; Scheduler's public API is unchanged.

Declare a durable recurring sweep job from a LifecycleCapability so it survives isolate restarts and retries automatically.

```typescript
import { LifecycleCapability, LifecycleJobContext } from
'agents';

class Cleanup extends LifecycleCapability {
  async scheduleSweep(time: number) {
    await this.lifecycle.jobs.push({ id: 'sweep', fn: 'sweep',
time });
  }

  onJob({ job }: LifecycleJobContext) {
    // Return nothing to mark complete, { rescheduleAt } to
suspend,
    // or 'yield' to wake again immediately.
    console.log('Running sweep job', job);
  }
}
```

— Exact API signature, hooks, migration path, and runnable example `agents@0.23.0` `@cloudflare/think@0.18.0`

---

03 **Queue Lifecycle capability (agents/queue)** BREAKING 90

Adds a Queue Lifecycle capability (`agents/queue`) for durable, alarm-driven background work with typed callbacks, upsert IDs, and per-item retry. `dequeue` , `dequeueAll` , `dequeueAllByCallback` , `getQueue` , and `getQueues` are now asynchronous, and queue() gains `options.id` for stable item identity; `QueueItem.created_at` is renamed `createdAt` . Queued callbacks running from the alarm loop no longer see the enqueuing request's `connection` or `request` via getCurrentAgent(). The legacy `cf_agents_queues` table is dropped and rows migrate into the job queue on first start, and `agents >=0.24.0` is now required.

— Names exact APIs and migration path, no runnable example given `@cloudflare/think@0.19.0`

---

04 **Alarm memory-limit hooks and stream recovery tuning** IMPROVED 90

Adds `onMemoryLimit` hook on capabilities and makes Lifecycle.install(host, { maxAlarmMemoryLimitStrikes }) the canonical way to set the alarm memory-limit strike budget, replacing the former `setLifecycleAlarmMemoryLimitStrikes` side channel. Moves recovery-loop membership to the job row via `LifecycleJobPushOptions.recoveryLoop` , so the alarm memory-limit breaker backs off and purges only flagged recovery jobs. Adds ResumableStream.progressMarker() to derive the recovery forward-progress marker from durable stream-log segments rather than a KV counter per chunk, and

creditProgress() for parent agents forwarding a sub-agent's output; raises `DEFAULT_CHAT_RECOVERY_MAX_WORK` from 1000 to 10000 segments after switching the work-budget unit from chunks to segments.

React to alarm memory-limit strikes in a capability without coupling to Agent internals, using the new `onMemoryLimit` hook via `Lifecycle.install`.

typescript

```typescript
import { Lifecycle } from 'agents';
import { AIChatAgent } from '@cloudflare/ai-chat';

export class MyChatAgent extends AIChatAgent<Env> {
  constructor(ctx: DurableObjectState, env: Env) {
    super(ctx, env);
    Lifecycle.install(this, { maxAlarmMemoryLimitStrikes: 3 });
  }

  onAlarmMemoryLimit(ctx: { job?: unknown }) {
    console.warn('Memory limit strike recorded', ctx);
  }
}
```

— Exact APIs, numeric change, and a runnable hook example `@cloudflare/ai-chat@0.12.0`

05 **RoutedAgents user-hub-per-chat topology** NEW 85

Adds `RoutedAgents` (importable from `agents/routing`), a Lifecycle capability for a 'user hub with one Durable Object per chat' topology: create(), list(), setMetadata(), get(), and delete() manage a durable catalog of independent top-level Agents routed by public ID.

Model a per-user hub that owns many independent chat Durable Objects, routing browser requests to each chat without waking the hub for WebSocket frames.

typescript

```typescript
import { Agent } from 'agents';
import { RoutedAgents } from 'agents/routing';
import { ChatAgent } from './chat-agent';

class UserAgent extends Agent<Env> {
  readonly chats = new RoutedAgents<ChatAgent, { title: string }>
({
    namespace: this.env.ChatAgent,
    route: 'chats',
  });

  constructor(ctx: DurableObjectState, env: Env) {
    super(ctx, env);
    this.lifecycle.use(this.chats);
  }

  async createChat(title: string) {
    return this.chats.create({ title });
  }

  async listChats() {
    return this.chats.list();
  }
}
```

— Named methods and full runnable example of the topology `agents@0.23.0`

06 **agents/sessions durable conversation storage module** NEW

85

Adds `agents/sessions`, a Lifecycle capability providing durable conversation storage: a tree of messages with branches and compaction overlays, streamed and byte-budgeted reads via getRecentHistory(), and full-text search built on first call to search(). appendMessage(msg, { source: 'client' }) sanitizes reserved metadata; messages exceeding the 1.5 MiB row budget are split across `cf_agents_session_message_chunks` continuation rows and reassembled transparently on read.

— Names methods, table, and byte limit but lacks a usage snippet `agents@0.23.0`

07 **Think media eviction moved to context-window technique** BREAKING

85

Moves media eviction from a storage-layer concern to a context-window technique: evicted media is preserved at `/attachments/evicted/<messageId>-<n>.<ext>` in

the Workspace with its real MIME type, and a marker like `[evicted image/png, 812004 bytes; preserved at /attachments/evicted/<messageId>-<n>.png]` is left in context. `WorkspaceLike.writeFileBytes` is now optional — a custom workspace without it keeps working and eviction/skills projection log once and stand down; `MediaEvictionConfig.externalizeToWorkspace` is deprecated and ignored since evicted bytes are always preserved to the Workspace, and `sessionAttachments` configuration is removed.

— Exact path format, marker example, and deprecated config named `@cloudflare/think@0.18.0`

08 agents/context module for prompt-context management NEW

83

Adds `agents/context`, exposing `ContextBlocks`, `AgentContextProvider`, `AgentSearchProvider`, and skill providers for prompt-context management separate from conversation storage. Think adopts it via a configureContext() hook for declaring prompt blocks, exposing `this.context` (a `ContextBlocks`) as the primary way to read context.

Use configureContext() and `this.context` in a Think subclass instead of the deprecated withContext() / `this.session` pattern.

typescript

```typescript
import { Think } from '@cloudflare/think';
import { ContextBlocks } from 'agents/context';

class MyAgent extends Think<Env> {
  configureContext() {
    return [
      { id: 'system', content: 'You are a helpful assistant.' }
    ];
  }

  async onStart() {
    await super.onStart();
    const block = this.context.getBlock('system');
    console.log('System prompt loaded:', block?.content);
  }
}
```

— Named exports plus a runnable configureContext() example `agents@0.23.0` `@cloudflare/think@0.18.0`

09 dynamicAgents capability facade NEW

78

Adds `this.dynamicAgents.{get,abort,delete,has,list}` methods and the `DynamicAgentClass` / `DynamicAgentStub` types, extracting facet ('sub-agent')

machinery into `packages/agents/src/dynamic-agents/` as a Lifecycle capability (`capabilityId: 'dynamic-agents'`). Ships with `examples/next/dynamic-agents`, a supervisor running user-submitted Durable Object code as facets via Worker Loader, with a React + Vite UI and workers-pool tests.

— Named methods and types with example project, no inline snippet `agents@0.23.0`

THINNER COVERAGE BELOW

10 **Lifecycle.use() fallback dispatch option** NEW 55

Adds `{ fallback: true }` option to Lifecycle.use(), allowing a capability to be dispatched after all non-fallback capabilities regardless of installation order.

— Names the exact flag but no example of use `agents@0.23.0`

11 **Workflow-notification retry and backoff policy** IMPROVED 50

Workflow-notification delivery now retries with backoff capped at ten minutes (previously five) and gives up after twelve hours of continuous failure (previously never), reporting the failure through `onError`.

— Concrete before/after numbers but no configurable knob shown `@cloudflare/think@0.19.0`

12 **LifecycleServices status() replaces starting()** BREAKING 40

LifecycleServices.starting() is removed and replaced by status(), which returns `'zero' | 'starting' | 'started'`.

— Single API rename with return values, no example `@cloudflare/think@0.19.0`

└──▷ **BREAKING ON UPGRADE**

! `dequeue`, `dequeueAll`, `dequeueAllByCallback`, `getQueue`, and `getQueues` are now asynchronous — callers that did not await them will silently drop results.

! `QueueItem.created_at` is renamed `createdAt` — any code reading `created_at` will see `undefined` after upgrade.

! LifecycleServices.starting() is removed and replaced by status() — any call to starting() will throw after upgrade.

! Queued callbacks running from the alarm loop no longer see the enqueuing request's `connection` or `request` through getCurrentAgent() — code that relied on those values inside a queued callback will find them absent.

! The `cf_agents_queues` table is gone; legacy rows migrate into the job queue on next start. Deployments skipping this release must upgrade through it before the next minor release removes the migration path.

! The `cf_think_workflow_notifications` table migrates and is dropped on start; deployments skipping this release must upgrade through it before the migration is removed

in the next minor release.

- ! `agents >=0.24.0` is now required for Think's background work.
- ! The experimental memory stack is removed: `agents/experimental/memory/session` and `agents/experimental/memory/utils` subpaths no longer exist, taking Session.create(), `SessionManager`, `PostgresSessionProvider`, `PostgresContextProvider`, `PostgresSearchProvider`, `R2SkillProvider`, and the `SessionProvider` interface with them.
- ! The pull-based alarm-contribution model is removed: capability getNextAlarm() / onAlarm(), host getNextAlarm(), `LifecycleServices.alarms` (`rearm` / `disabled`), and `AlarmContribution` are gone.
- ! `Scheduler`'s `__DO_NOT_USE_WILL_BREAK__handleAlarmMemoryLimit` escape hatch is removed; alarm memory-limit policy now flows through the new onAlarmMemoryLimit() host hook.
- ! The `cf_agents_schedules` table is dropped on startup after its rows are migrated into `cf_agents_jobs`; any tooling or queries that read `cf_agents_schedules` directly will fail.
- ! `this.messages` is empty in the constructor and remains so until `onStart`; subclasses that read `this.messages` in their constructor now see `[]` — move such reads to `onStart` or later.
- ! Once an `AIChatAgent` object migrates its `cf_ai_chat_agent_messages` rows to the Sessions chain, rolling back to the previous release leaves it with an empty conversation (rolling forward again is safe).
- ! `cleanupStreamBuffers` and `STREAM_CLEANUP_DELAY_SECONDS` are removed from `agents/chat`.
- ! `Agent.onAlarmMemoryLimit`'s policy relay, the `_cf_recoveryAlarmCallbacks` template hook, `Scheduler.applyMemoryLimitPolicy`, and `setLifecycleAlarmMemoryLimitStrikes` are removed.
- ! `AIChatAgent` and `Think` now require `agents >= 0.23.0`.
- ! Storage migrates on first wake and cannot be rolled back: `assistant_messages`, `assistant_compactions`, and `assistant_config` rows are lifted into `cf_agents_session_*` tables and the old tables are dropped; an object that has woken on this version has an empty conversation if you roll back to the previous release.
- ! Existing `cf_agents_schedules` rows are migrated into `cf_agents_jobs` on startup and the legacy table is dropped.
- ! Existing `cf_agents_stream_chunks` rows are folded into rollover blocks lazily and the table is dropped once empty.
- ! `hydrationByteBudget` default changed from 24 MiB to 32 MiB and is now a hard ceiling with no message-count floor, so an unusually large recent window can hydrate fewer than four messages.
- ! The pull-based alarm-contribution model is removed: LifecycleCapability.getNextAlarm(), LifecycleCapability.onAlarm(), host getNextAlarm(), `LifecycleServices.alarms` (`rearm` / `disabled`), and `AlarmContribution` are gone.
- ! `Scheduler.__DO_NOT_USE_WILL_BREAK__handleAlarmMemoryLimit` is removed; alarm memory-limit handling now goes through the new onAlarmMemoryLimit() host hook.

! `sessionAttachments` configuration is removed; media layout in Sessions is automatic and lossless.

! getRecentHistory(budget, minRecentMessages): the `minRecentMessages` second argument is accepted and ignored.

! compactAfter(threshold, { tokenCounter }): the `tokenCounter` option is accepted and ignored.

! `MediaEvictionConfig.externalizeToWorkspace` is deprecated and ignored; evicted bytes are always preserved to the Workspace.

! Direct SQL queries to `assistant_messages` are no longer supported; use this.session.history() or getHistory() instead.

! Think requires `agents >=0.23.0`.

`@cloudflare/think@0.19.0`

Adds Queue Lifecycle capability (`agents/queue`) for durable, alarm-driven background work with typed callbacks, upsert IDs, and per-item retry.

└──▷ **GET THIS VERSION**

```
$ git clone --branch @cloudflare/think@0.19.0
https://github.com/cloudflare/agents.git
# already have the repo? check out this version:
$ git checkout @cloudflare/think@0.19.0
```

› Makes `dequeue`, `dequeueAll`, `dequeueAllByCallback`, `getQueue`, and `getQueues` asynchronous, and adds `options.id` support to queue() for stable item identity.

› Replaces LifecycleServices.starting() with status(), which returns `'zero' | 'starting' | 'started'`.

› Workflow-notification delivery now retries with backoff capped at ten minutes (previously five) and gives up after twelve hours of continuous failure (previously never), reporting the failure through `onError`.

└──▷ **BREAKING ON UPGRADE**

! `dequeue`, `dequeueAll`, `dequeueAllByCallback`, `getQueue`, and `getQueues` are now asynchronous — callers that did not await them will silently drop results.

! `QueueItem.created_at` is renamed `createdAt` — any code reading `created_at` will see `undefined` after upgrade.

! LifecycleServices.starting() is removed and replaced by status() — any call to starting() will throw after upgrade.

- ! Queued callbacks running from the alarm loop no longer see the enqueuing request's `connection` or `request` through getCurrentAgent() — code that relied on those values inside a queued callback will find them absent.
- ! The `cf_agents_queues` table is gone; legacy rows migrate into the job queue on next start. Deployments skipping this release must upgrade through it before the next minor release removes the migration path.
- ! The `cf_think_workflow_notifications` table migrates and is dropped on start; deployments skipping this release must upgrade through it before the migration is removed in the next minor release.
- ! `agents >=0.24.0` is now required for Think's background work.

GOOD PICK? 👍 👎 GOOD REASON? 👍 👎

04 Weaviate

2 RELEASES · 2026-09-11 → 2026-09-17

Vector DBs & RAG

Weaviate is an open-source vector database that stores both objects and vectors, allowing for the combination of vector search with structured filtering with the fault tolerance and scalability of a cloud-native database.

// WHY IT MADE THE LIST

DEPTH 5/5 IMPACT 4/5

4-bit Rotational Quantization (RQ4) cuts vector-index heap by 45% versus 8-bit RQ, with an optional centering flag that lifts recall@10 on anisotropic embeddings and SIMD-accelerated Fast Walsh-Hadamard encoding — letting larger indexes fit the same memory.

> Weaviate 1.39 headlines this window with 4-bit Rotational Quantization and centering that cut heap usage by 45% while boosting recall, alongside SIMD-accelerated encoding, faster HNSW traversal, and new node-level query admission control for cluster stability.

└──▷ WHAT SHIPPED · 4 FEATURES most completely described first ? what's the number?

01 4-bit Rotational Quantization with centering and SIMD encoding

NEW 95

Adds a `bits: 4` setting to the `rq` vector index quantizer in `vectorIndexConfig`, enabling 4-bit Rotational Quantization (RQ4) with a 45% heap reduction versus 8-bit RQ. Adds a `centering: true` option to the `rq` quantizer config (and `centering=True` via Configure.VectorIndex.Quantizer.rq() in the Python client), improving recall@10 by +0.1 to +6.1 percentage points on anisotropic embedding datasets. Also adds SIMD-accelerated Fast Walsh-Hadamard Transform (FWHT) encoding on amd64/AVX and arm64/NEON, speeding up RQ8 encoding 3.8× and running RQ4 encoding at 6.36–8.08 µs on Intel Xeon.

Create a collection using 4-bit centered RQ to cut heap usage by ~45% while maintaining high recall on anisotropic embeddings.

```python
from weaviate.classes.config import Configure, Property, DataType

client.collections.create(
    name="Recipes",
    vector_config=Configure.Vectors.text2vec_openai(
        quantizer=Configure.VectorIndex.Quantizer.rq(
            bits=4,
            centering=True,
        )
    ),
    properties=[
        Property(name="title", data_type=DataType.TEXT),
    ],
)
```

Configure 4-bit centered RQ directly in the vector index config JSON when using the REST API or schema definition files.

```json
{
  "vectorIndexConfig": {
    "rq": {
      "enabled": true,
      "centering": true,
      "bits": 4
    }
  }
}
```

– Names config keys, client call, exact heap/recall/latency numbers.

launch-20260917-923249be

02 **Optional shard for HFresh debug reassignment** IMPROVED

65

Makes shard optional for HFresh reassignment in the `debug` API, allowing cluster-wide orphaned-postings recovery without specifying a shard.

– Names the debug API and behavior change, but no example given. v1.39.4

THINNER COVERAGE BELOW

03 **CPU prefetching for HNSW graph traversal** IMPROVED 50

Adds efficient CPU prefetching for AMD64 and ARM64 in HNSW graph traversal, contributing a 7–11% query throughput improvement and 12% faster imports.

— States mechanism and figures but no config or API surface. launch-20260917-923249be

04 **Node-level query admission control** NEW 35

Adds node-level query admission control (P1b) that prevents overloaded nodes from accepting new queries, shedding load on individual nodes to improve cluster stability under high query concurrency.

— Describes purpose only; no config key or endpoint named. v1.38.15 v1.39.4

announced · Sep 17, 2026

Weaviate 1.39 adds 4-bit Rotational Quantization (RQ4) with optional centering, delivering 45% heap reduction and faster vector search.

└──▷ **USE IT**

Create a collection using 4-bit centered RQ to cut heap usage by ~45% while maintaining high recall on anisotropic embeddings.

python

```python
from weaviate.classes.config import Configure, Property, DataType

client.collections.create(
    name="Recipes",
    vector_config=Configure.Vectors.text2vec_openai(
        quantizer=Configure.VectorIndex.Quantizer.rq(
            bits=4,
            centering=True,
        )
    ),
    properties=[
        Property(name="title", data_type=DataType.TEXT),
    ],
)
```

Configure 4-bit centered RQ directly in the vector index config JSON when using the REST API or schema definition files.

```json
{
  "vectorIndexConfig": {
    "rq": {
      "enabled": true,
      "centering": true,
      "bits": 4
    }
  }
}
```

- Adds `bits: 4` setting to the `rq` vector index quantizer in `vectorIndexConfig`, enabling 4-bit Rotational Quantization (RQ4) with a 45% heap reduction versus 8-bit RQ.
- Adds `centering: true` option to the `rq` quantizer config (and `centering=True` in the Python client via Configure.VectorIndex.Quantizer.rq()), improving recall@10 by +0.1 to +6.1 percentage points on anisotropic embedding datasets.
- Adds SIMD-accelerated Fast Walsh-Hadamard Transform (FWHT) encoding on amd64/AVX and arm64/NEON, speeding up RQ8 encoding 3.8× and RQ4 encoding is new in 1.39 at 6.36–8.08 µs on Intel Xeon.
- Adds efficient CPU prefetching for AMD64 and ARM64 in HNSW graph traversal, contributing 7–11% query throughput improvement and 12% faster imports.

GOOD PICK? 👍 👎 GOOD REASON? 👍 👎

05 Cline

7 RELEASES · 2026-09-11 → 2026-09-17

AI Coding Agents

Autonomous coding agent as an SDK, IDE extension, or CLI assistant.

// WHY IT MADE THE LIST

DEPTH 4/5 IMPACT 4/5

Cline desktop can now run agent sessions on a remote SSH host while the client stays local, uploading a self-contained helper that needs no apt, npm, root, or open port — plus sub-agents spawned in one step now run concurrently instead of serially. It closes the gap between working locally and driving an agent on a remote machine.

> Cline's biggest addition this window is SSH remote execution — sessions can now run entirely on a remote host from both the desktop app and SDK — alongside concurrent sub-agent parallelism, automatic retries for failed model turns, web search enabled by default, Hub-managed Agent Plugins, and a wave of provider, scheduling, and UI polish across desktop, CLI, and SDK.

└──▷ WHAT SHIPPED · 33 FEATURES most completely described first ? what's the number?

01 SSH remote execution in desktop and SDK NEW

100

Cline can now run a full session on a remote SSH host while the client stays local. In the desktop app, add and test a host under Settings → Remote, then pick it from the environment selector beside the workspace picker — Add project… browses the remote machine and Local disconnects; SSH config aliases are honored, with a blank Port field using the alias's own configuration. Profiles are stored at `~/.cline/data/settings/remote-environments.json` (mode `0600`), recording only the identity-file path, never private-key contents, and unknown or changed host keys are rejected. On first connect a self-contained remote helper is uploaded to `~/.cline/remote/` with no `apt`, `npm`, root access, global CLI install, or public port required; `CLINE_REMOTE_HELPER_BINARY` supplies a locally built helper for macOS hosts, and only Linux x64/arm64 hosts are supported natively (32-bit Raspberry Pi OS is not). At the SDK level, RemoteEnvironmentService.connect() performs the same upload, starts a loopback-bound Hub, and opens an SSH tunnel, handing the resulting endpoint and token to ClineCore.create({ backendMode: 'remote' }) so tools, sessions, and persistence all execute on the remote host; Hub daemons can also be started with connector management disabled so an SSH-spawned Hub doesn't adopt or restart the account's Slack and Telegram connectors.

— Full mechanism, config paths, env var, and API named. desktop-v0.0.31 sdk/sdk/v0.0.83

02 Hub-managed Agent Plugins NEW

90

Agent Plugins are now managed through the Hub: packages under `~/.agents/plugins/*` are auto-discovered, their skills are exposed as `plugin-name:skill-name` through the skills tool, and their MCP servers start automatically without touching `cline_mcp_settings.json`.

Drop a plugin directory on the hub host to expose custom skills and MCP servers to all sessions without editing cline_mcp_settings.json.

```
$ mkdir -p ~/.agents/plugins/my-plugin/skills
cat > ~/.agents/plugins/my-plugin/plugin.json <<'EOF'
{ "name": "my-plugin", "version": "1.0.0" }
EOF
# Skills appear as my-plugin:<skill-name>; MCP servers listed in mcp.json start automatically
```

— Exact path, naming scheme, and config file named with a runnable example. `cli-v3.0.62` `sdk/sdk/v0.0.83`

---

03 **Sampled OTLP tracing for AI SDK calls** NEW — 85

Adds `OTEL_TRACES_EXPORTER`, `CLINE_TRACE_SAMPLE_PERCENT`, and `CLINE_TRACE_RECORD_CONTENT` environment variables to enable sampled AI SDK tracing exported via the host OTLP exporter.

Enable sampled OTLP tracing for Cline AI SDK calls to debug or audit model interactions via your existing observability stack.

```
$ OTEL_TRACES_EXPORTER=otlp
OTEL_EXPORTER_OTLP_ENDPOINT=http://localhost:4318
CLINE_TRACE_SAMPLE_PERCENT=10 CLINE_TRACE_RECORD_CONTENT=true
cline 'Refactor auth module'
```

— Three exact env vars with a runnable example. `desktop-v0.0.27`

---

04 **Web search enabled by default** IMPROVED — 80

Web search is now enabled by default in non-YOLO sessions on models and provider combinations that support it, across desktop, CLI, and SDK. A settings toggle disables it, `tools.web_search.enabled = false` opts out at the config level, and an unreadable or unreachable settings file leaves it off rather than silently on.

Opt a specific environment out of provider-native web search when you want deterministic, search-free responses in CI.

```yaml
tools:
  web_search:
    enabled: false
```

— Config key and fail-closed behavior are named and demoed. `v4.1.19` `cli-v3.0.62` `sdk/sdk/v0.0.83`

05 **Default model resolution changes across providers** BREAKING 75

The resolved default model changed for 44 providers in one release and 33 in the next when no model is pinned: most now land on DeepSeek V4.1 Flash, Gemini and Vertex resolve to Gemini 3.8 Flash, GitHub Copilot and Vivgrid to GPT-6 Astra, NVIDIA to GLM 5.3 Flash, and NanoGPT to GPT Astra. Any provider used without a pinned model switches to a different default after each update.

— Names exact providers and replacement models but no pinning instructions. `v4.1.19` `desktop-v0.0.28`

06 **Live GitHub PR and CI status in composer** NEW 75

The composer now shows live GitHub PR status — PR number, merge status, changed-line totals, and CI checks with log inspection — refreshing every 30 seconds, on window focus, and on demand; it requires the `gh` CLI installed and signed in with a `GitHub.com` `origin` remote.

— Exact refresh cadence and CLI prerequisite named. `desktop-v0.0.26`

07 **Automatic retry for failed model turns** IMPROVED 70

Turns that die mid-stream from a transient provider error are retried automatically up to 3 times with exponential backoff, across desktop, CLI, and SDK; a turn that has already streamed any text, reasoning, media, or tool call is never retried, replacing CLI behavior that previously exited immediately with code 1. SDK-level retries for request-start 429/5xx/network failures also increased from 2 to 5.

— Exact retry counts and conditions given; behavior is automatic. `desktop-v0.0.28` `cli-v3.0.62` `sdk/sdk/v0.0.83`

08 **Startup notice suppression and Desktop launch notice** NEW 70

The `CLINE_DISABLE_CLINE_PASS_NOTICE=1` environment variable now gates both the ClinePass intro notice and a new one-time CLI startup notice pointing at cline.bot/desktop; at most one notice appears per launch.

Silence the new Cline Desktop startup notice (and all other startup notices) in CI or scripting environments where interactive prompts are unwanted.

```
$ CLINE_DISABLE_CLINE_PASS_NOTICE=1 cline "Run tests and fix any failures"
```

— Exact env var and runnable suppression example given. `cli-v3.0.62`

09 **Scheduling and automation reliability** IMPROVED 70

Scheduled and recurring runs are now more reliable: new schedules default to the host's local IANA timezone instead of UTC when none is specified; parallelism limits are enforced at claim time inside the SQLite claim transaction with per-schedule and global limits, dispatch is decoupled from execution so schedules no longer stall each other, locally active claims are renewed each tick, runs correctly resume after system sleep, and automation event acceptance runs atomically in one write transaction that rolls back and propagates failures for redelivery.

— Names the transaction mechanism but offers no user-facing control. `cli-v3.0.62` `sdk/sdk/v0.0.83` `desktop-v0.0.26`

10 **Steer queued message into active turn with Enter** NEW 65

Pressing Enter with an empty composer while the agent is running now steers the first queued message into the active turn; the steer is claimed atomically so a fast second Enter cannot steer a message that has already left the queue, and a failed steer surfaces as a toast instead of silently doing nothing.

— Exact key and failure behavior named, easy to try. `desktop-v0.0.29`

11 **PowerShell edition detection for commands** IMPROVED 65

The command-running tool now reports which PowerShell edition is active — Windows PowerShell (powershell.exe) versus PowerShell (pwsh.exe) — and instructs the model to write commands directly rather than wrapping them in a second shell invocation, preventing pipeline corruption with `$_`.

— Names both PowerShell binaries and the exact corruption avoided. `desktop-v0.0.28`

12 **Unified Customize view for Tools, Skills, Rules** IMPROVED 65

The Customize view's Tools, Skills, and Rules tabs are unified into a consistent list style, with a search bar on the Tools tab that filters both sections, per-section Enable all/Disable all controls, and an in-place enable/disable toggle for Skills with Copy path /

Uninstall menu options; tabs are reordered to Tools, Plugins, Skills, Rules, MCP, Hooks, and the sidebar's 'New Task' button becomes 'New Session'.

— Names every control added and the new tab order. `desktop-v0.0.26`

13 **Imported session history summarization** IMPROVED 60

Sessions imported from Claude Code, Codex, or opencode now summarize the foreign history on first resume rather than replaying tool calls the current agent cannot make; import origin is stamped on telemetry, and resuming no longer overwrites stored import or automation provenance with a default 'user' origin.

— Names the three source tools and provenance behavior. `sdk/sdk/v0.0.83`

THINNER COVERAGE BELOW

14 **New model providers: Infer by Flow7, Melious, Wallaby, Crusoe** NEW 55

Adds Infer by Flow7, Melious, and Wallaby to the model catalog (5,923 → 6,079 models), plus Crusoe as an OpenAI-compatible provider listed under the renamed API Providers settings section.

— Names each new provider and catalog size, but no config detail. `desktop-v0.0.28` `desktop-v0.0.27`

15 **Faster, self-healing checkpoint snapshots** IMPROVED 55

Checkpoints now use a persistent per-session snapshot index so git's stat cache skips unchanged untracked files, reducing each snapshot after the second turn to about one git process; a corrupt index heals with one rebuild-and-retry that also clears a stale `index.lock`.

— Explains mechanism and named lock file but is automatic. `sdk/sdk/v0.0.83`

16 **Composer provider picker additions** NEW 55

The composer's provider picker gains a 'Set up another provider' row that opens Settings → Models, and a Cline Pass entry selectable alongside Cline, configured under the same Cline account credentials.

— Names both additions and the settings path they open. `desktop-v0.0.26`

17 **Warning for images sent to text-only models** NEW 50

Images attached to a text-only model are now flagged with a warning badge and a one-click switch to an image-capable model, instead of the attachment being silently discarded.

— Clear UI behavior but no named surface. `v4.1.19`

18 **Live-refreshing model picker** IMPROVED 50

The model picker now refreshes its provider and model list, including live Recommended and Free groupings, every time it is opened, rather than freezing until app restart.

— Behavior described; no config surface named. `desktop-v0.0.28`

19 **Plan details surfaced during sign-in** IMPROVED 50

The Cline account sign-in flow now surfaces plan details on the sign-in card and final onboarding step: free model promotions, ClinePass coverage across DeepSeek, Kimi, and GLM, and the absence of an API key requirement.

— Names covered models but is UI copy only. `desktop-v0.0.28`

20 **Streaming no longer throttled by hook forwarding** IMPROVED 50

Per-chunk text, reasoning, and tool-update deltas are no longer forwarded to remote `onEvent` hooks, removing a streaming throttle; the hub event log also moves to `synchronous = NORMAL`.

— Names the hook and setting changed but no usage guidance. `sdk/sdk/v0.0.83`

21 **Offline model picker fallback expanded** IMPROVED 45

When the models endpoint is unreachable, model pickers and SDK clients now fall back to full generated Recommended, Free, and Subscribed tier lists built at release time, replacing the previous hardcoded/hand-maintained six-model offline fallback that had no subscribed tier.

— Describes the fallback change; not directly actionable. `cli-v3.0.62` `sdk/sdk/v0.0.83`

22 **File index skip for home and root workspaces** IMPROVED 45

Cline now skips building a file index when the workspace root is the home directory or a filesystem root, preventing an OOM-kill when typing `@` mentions from `$HOME`.

— Names the failure prevented but is an automatic fix. `sdk/sdk/v0.0.83`

23 **Local-auth routing for Claude Code and OpenCode** IMPROVED 45

Claude Code and OpenCode are now declared local-auth providers with a named local CLI, so hosts route them to a local-CLI readiness screen instead of demanding an API key.

— Names both providers and the new UI path. `sdk/sdk/v0.0.83`

24 **Actionable auth failure prompts** IMPROVED 45

Auth failures on the Cline provider now show a single actionable 'Sign in to Cline' button; other providers show 'Open model settings'; local-CLI providers retain their CLI link.

— Names the exact buttons shown per provider type. `desktop-v0.0.27`

25 **Concurrent sub-agent parallelism** NEW 40

Sub-agents spawned in the same step now run concurrently instead of sequentially, so delegating N independent tasks takes as long as the slowest one rather than the sum of all of them.

— Describes behavior change but no controlling flag or limit. `desktop-v0.0.31`

26 **Langfuse tracing restricted to Cline providers** BREAKING 40

Langfuse tracing is now limited to the Cline and Cline Pass providers, preventing prompts and responses from third-party and BYOK providers from being exported in Langfuse-configured environments.

— Names affected providers; scope of impact is clear. `sdk/sdk/v0.0.83`

27 **Default system prompt for prompt-less sessions** IMPROVED 40

A session started without an explicit `systemPrompt` now gets a default Cline prompt built on the execution host, rather than running with none.

— Names the field but change is internal. `sdk/sdk/v0.0.83`

28 **Voice input failure toast** IMPROVED 40

Voice input failures now surface as a 'Speech input failed' toast in chat with the actual error, preserving the draft and microphone button for retry, instead of redirecting to Settings → Voice.

— Describes new failure UX with no further detail. `desktop-v0.0.27`

29 **Session history pagination fix** IMPROVED 35

Session history listing now widens its fetch window until the requested page fills and filters child rows in SQL, so root sessions are no longer hidden behind their children.

— Describes the bug fixed but no user action. `sdk/sdk/v0.0.83`

30 **Credential field sanitization** IMPROVED 35

Credential fields are now stripped of Unicode control and format characters and surrounding whitespace before being saved.

— States the sanitization but no scope of affected fields. `sdk/sdk/v0.0.83`

31 Session sidebar sort toggle shows next state IMPROVED

30

The session sidebar sort toggle now shows the icon for the mode a click will switch to — folder icon when sorted by time, clock icon when grouped by project — rather than the mode already active.

— Small UI fix with no further mechanism. `desktop-v0.0.29`

32 ClinePass defaults to subscription-tier model IMPROVED

30

ClinePass subscribers are now defaulted to a model from their subscription tier — in both the desktop app and CLI — rather than the most recently published free model.

— States the change but no configuration detail. `desktop-v0.0.28` `cli-v3.0.62`

33 API Providers settings renamed IMPROVED

30

The settings sidebar section formerly called 'Models' is renamed to 'API Providers', with a new icon.

— Simple rename with no further behavior described. `desktop-v0.0.27`

└──▷ **BREAKING ON UPGRADE**

! The resolved default model changes for 44 providers — most landing on DeepSeek V4.1 Flash; Gemini and Vertex now resolve to Gemini 3.8 Flash, GitHub Copilot and Vivgrid to GPT-6 Astra, and NVIDIA to GLM 5.3 Flash. Any provider used without a pinned model will switch to a different default after this update.

! The resolved default model changes for 33 providers; any provider used without a pinned model will now send requests to a different model (most land on DeepSeek V4.1 Flash, NVIDIA moves to GLM 5.3 Flash, NanoGPT moves to GPT Astra).

`desktop-v0.0.31`

Cline desktop gains SSH remote execution and concurrent sub-agent parallelism in v0.0.31

└──▷ **GET THIS VERSION**

```
$ git clone --branch desktop-v0.0.31
https://github.com/cline/cline.git
# already have the repo? check out this version:
$ git checkout desktop-v0.0.31
```

› Stores SSH remote-host profiles at `~/.cline/data/settings/remote-environments.json` (mode `0600`), recording identity-file path (never private-key contents); unknown or changed host keys are rejected.

- Adds SSH remote-environment support via `Settings → Remote`: add and test a host, then select it from the environment selector beside the workspace picker; `Add project…` browses the remote machine and `Local` disconnects.
- Uploads a self-contained remote helper to `~/.cline/remote/` on first connect — requires no `apt`, `npm`, root access, global CLI install, or public port — with a `CLINE_REMOTE_HELPER_BINARY` env var to supply a locally built helper for macOS hosts.
- Supports Linux x64 and arm64 remote hosts (32-bit Raspberry Pi OS is not supported; macOS hosts require a locally built helper via `CLINE_REMOTE_HELPER_BINARY`).
- SSH config aliases are honored: leave `Port` blank to use the alias's own configuration, or enter a value to override it; recent and last-used workspaces are tracked separately per host.

+1 more

GOOD PICK? 👍 👎 GOOD REASON? 👍 👎

// END OF TRANSMISSION

The Cyber Toolchain · This Week's Highlights · September 18, 2026
The full firehose — every feature release, every day — is in the newsletter.

▷ Subscribe