Sessions and memory

Session Pruning

Session pruning trims **old tool results** from the in-memory context right before each LLM call. It does **not** rewrite the on-disk session history (`*.jsonl`).

When it runs

When `mode: "cache-ttl"` is enabled and the last Anthropic call for the session is older than `ttl` .

Only affects the messages sent to the model for that request.

Only active for Anthropic API calls (and OpenRouter Anthropic models).

For best results, match `ttl` to your model `cacheControlTtl` .

After a prune, the TTL window resets so subsequent requests keep cache until `ttl` expires again.

Smart defaults (Anthropic)

OAuth or setup-token profiles: enable `cache-ttl` pruning and set heartbeat to `1h` .

API key profiles: enable `cache-ttl` pruning, set heartbeat to `30m` , and default `cacheControlTtl` to `1h` on Anthropic models.

If you set any of these values explicitly, OpenClaw does **not** override them.

What this improves (cost + cache behavior)

Why prune: Anthropic prompt caching only applies within the TTL. If a session goes idle past the TTL, the next request re-caches the full prompt unless you trim it first.

What gets cheaper: pruning reduces the **cacheWrite** size for that first request after the TTL expires.

Why the TTL reset matters: once pruning runs, the cache window resets, so follow-up requests can reuse the freshly cached prompt instead of re-caching the full history again.

What it does not do: pruning doesn't add tokens or "double" costs; it only changes what gets cached on that first post-TTL request.

What can be pruned

Only `toolResult` messages.

User + assistant messages are **never** modified.

The last `keepLastAssistants` assistant messages are protected; tool results after that cutoff are not pruned.

If there aren't enough assistant messages to establish the cutoff, pruning is skipped.

Tool results containing **image blocks** are skipped (never trimmed/cleared).

Context window estimation

Pruning uses an estimated context window (chars ≈ tokens × 4). The base window is resolved in this order:

1. `models.providers.*.models[].contextWindow` override.
2. Model definition `contextWindow` (from the model registry).
3. Default `200000` tokens.

If `agents.defaults.contextTokens` is set, it is treated as a cap (min) on the resolved window.

›

Mode

cache-ttl

Pruning only runs if the last Anthropic call is older than `ttl` (default `5m`).

When it runs: same soft-trim + hard-clear behavior as before.

Soft vs hard pruning

Soft-trim: only for oversized tool results.

Keeps head + tail, inserts `...`, and appends a note with the original size.

Skips results with image blocks.

Hard-clear: replaces the entire tool result with `hardClear.placeholder`.

Tool selection

`tools.allow` / `tools.deny` support `*` wildcards.

Deny wins.

Matching is case-insensitive.

Empty allow list ⇒ all tools allowed.

Interaction with other limits

Built-in tools already truncate their own output; session pruning is an extra layer that prevents long-running chats from

accumulating too much tool output in the model context.

Compaction is separate: compaction summarizes and persists, pruning is transient per request. See /concepts/compaction.

Defaults (when enabled)

```
ttl : "5m"
keepLastAssistants : 3
softTrimRatio : 0.3
hardClearRatio : 0.5
minPrunableToolChars : 50000
softTrim : { maxChars: 4000, headChars: 1500, tailChars: 1500 }
hardClear : { enabled: true, placeholder: "[Old tool result content cleared]" }
```

Examples

Default (off):

```
{
  agent: {
    contextPruning: { mode: "off" },
  },
}
```

Enable TTL-aware pruning:

```
{
  agent: {
    contextPruning: { mode: "cache-ttl", ttl: "5m" },
  },
}
```

Restrict pruning to specific tools:

```
{
  agent: {
    contextPruning: {
      mode: "cache-ttl",
      tools: { allow: ["exec", "read"], deny: ["*image*"] },
    },
  },
}
```

See config reference:

‹ Sessions Session Tools ›