TYGART MEDIA · CLAUDE & ANTHROPIC REFERENCE

Claude AI Context Window: Token Limits & Capacity

Edition of November 13, 2026 — facts verified November 13, 2026. Current edition always at https://tygartmedia.com/claude-ai-context-window/.

Direct Answer (November 13, 2026): Every current Claude model — **Fable 5.2, Opus 5.5, Sonnet 5.5, and Haiku 5.5** — supports a **1,000,000-token (1M) context window** and **128K max output** (models overview). Legacy Haiku 4.5 retains a 200,000-token context window and 64K max output.

Last refreshed: November 13, 2026 · Sources: Anthropic models overview & API pricing

Looking for quick answers? The FAQ section below covers every common question directly.

→ Context Window FAQ

Claude's context window is one of those specs that sounds simple until you actually need to use it. "1 million tokens" means almost nothing without a frame of reference. This is the guide we wish existed when we started building on Claude — written from our own experience running it in production, with numbers pulled directly from Anthropic's official documentation.

Quick Definition

The context window is Claude's working memory for a conversation. It holds everything Claude can see and reason about at once: your messages, Claude's responses, any documents you've shared, and system prompts. When the window fills up, earlier content drops out.

Current Context Window Sizes by Model (November 2026)

Current context window sizes by model.

These numbers come directly from Anthropic's official models page, (verified November 13, 2026). Model strings are exact API identifiers:

Model	API String	Context Window	Max Output
Claude Fable 5.2	claude-fable-5-2	1,000,000 tokens	128,000 tokens
Claude Opus 5.5	claude-opus-5-5	1,000,000 tokens	128,000 tokens
Claude Sonnet 5.5	claude-sonnet-5-5	1,000,000 tokens	128,000 tokens
Claude Haiku 5.5	claude-haiku-5-5	1,000,000 tokens	128,000 tokens
Claude Haiku 4.5 (legacy)	claude-haiku-4-5-20251001	200,000 tokens	64,000 tokens

All four current models — Fable 5.2, Opus 5.5, Sonnet 5.5, and Haiku 5.5 — share the full 1M token context window and 128K max output (note that Haiku 5.5 has a prompt-size price step above 100K tokens). Legacy Sonnet 5 capped output at 64K, and legacy Haiku 4.5 used a 200K context window with 64K output.

What Does 1 Million Tokens Actually Hold?

Token counts are an abstraction. Here's what 1 million tokens translates to in practical terms:

- **About 750,000 words of English text** — roughly 10 full-length novels, or 1,500 average blog posts
- **A full mid-size codebase** — a 50,000-line Python project with comments fits comfortably
- **Hours of meeting transcripts** — a full workday of recorded calls, transcribed, fits in one context window
- **Multiple large documents simultaneously** — 10 research PDFs at 30 pages each, all in the same conversation
- **Long conversation histories** — hundreds of back-and-forth exchanges before anything starts dropping off

We've loaded entire Notion exports, full project histories, and multi-document research packs into a single Claude session. At 1M tokens, you're unlikely to hit the ceiling in a normal working session. You hit it when you're doing things like: loading your entire codebase plus documentation plus conversation history and then asking Claude to do a full architectural review.

Context Window vs. Memory: What's the Difference?

Context window vs memory — what's the difference?

This is where a lot of people get confused. The context window and memory are not the same thing:

- **Context window:** What Claude can see right now, in this session. Once a session ends, it's gone.
- **Memory (in claude.ai):** A separate system that extracts and stores key information from past sessions. It surfaces relevant facts into future conversations as a snippet in the context.
- **Managed Agents memory stores:** A developer-layer construct where agents maintain and update knowledge bases across sessions — distinct from both the context window and the consumer memory feature.

The 1M token context window is your working memory for one session. It doesn't persist. Memory systems are what carry information across sessions — but they work by injecting a summary into the context window of the new session, not by giving Claude access to the full history.

Does a Bigger Context Window Mean Better Performance?

Mostly yes, with one important nuance. More context means Claude has more information to reason about, which generally produces better outputs for tasks that benefit from full context — code reviews, document synthesis, long-form writing, multi-document comparison.

The nuance: performance can degrade on tasks involving specific information buried deep in a very long context. This is sometimes called the "lost in the middle" problem — models tend to pay more attention to the beginning and end of a long context than the middle. Anthropic has worked on this with Claude's architecture, and it performs well on long-context tasks, but it's worth structuring important information at natural reference points rather than burying it in the middle of a 500-page document.

How We Actually Use the 1M Token Window

How we actually use the 1M token window.

We run Claude in production for content operations, site management, and agentic coding workflows. Here's where the 1M context window makes a concrete difference in our work:

- **Full site audits:** Loading every post from a WordPress site (200+ posts worth of content) into one session for comprehensive SEO analysis — without having to chunk and re-prompt
- **Cross-session context:** Pasting in long Notion briefings, prior session transcripts, and the current task in one go. The window is large enough that we don't have to decide what to leave out.
- **Codebase-wide reasoning:** In Claude Code, having the full project context means Claude can make changes that account for how files interact rather than reasoning only about the current file
- **Multi-document synthesis:** Research projects where we load 10-15 source documents and ask Claude to synthesize across them — something that was impossible at 100K context windows

The practical shift from 200K to 1M tokens wasn't just "more room." It changed what we could ask Claude to do in a single session.

Context Window on the API: Batch Output Extension

For API users: on the Message Batches API, current models support extended batch outputs; Fable 5.2, Opus 5.5, and Sonnet 5.5 supersede Fable 5.1/Sonnet 5 — check Anthropic's Message Batches docs for model-specific output headers using the `output-300k-2026-03-24` beta header. This is relevant for batch generation tasks where you need very long outputs — documentation generation, large codebases, book-length content.

Budgeting Context, Long-Context Pricing & Programmatic Checks

- **Shared input + output budget:** Token estimates run roughly **0.75 words per token** (~4 characters). The 1M context window spans both your accumulated prompt/history and Claude's replies, alongside each model's separate single-turn max output ceiling (128K on current 5.x models; 64K on legacy Haiku 4.5). At 1M tokens, you can hold roughly 5–6 full 300-page books (150K–200K tokens each), subject to per-file upload limits on claude.ai PDF uploads.
- **No long-context surcharge on flagship/mid-tier models:** Anthropic eliminated long-context surcharges in March 2026 — on Fable 5.2, Opus 5.5, and Sonnet 5.5, a 900,000-token API call bills at the same per-token rate as a 9,000-token call (only Haiku 5.5 applies a higher rate above 100K prompt tokens).
- **API hard limit vs. UI roll-off:** On claude.ai, exceeding the window pushes earlier turns out of active context with a notice; on the API, exceeding the context limit returns a hard request error.
- **Programmatic lookup via Models API:** Rather than hardcoding window sizes, query the Anthropic Models API, which returns exact `max_input_tokens`, `max_tokens`, and a `capabilities` object for every model ID.
- **Explicit persistence layers (Projects & System Prompts):** Beyond claude.ai opt-in memory summaries and Managed Agents memory stores/Dreaming, use **Claude Projects** (persistent documents and instructions attached to every thread in a project) or **API system prompts** when you need deterministic cross-session state without relying on inferred memory extraction.

Frequently Asked Questions

What is Claude's context window in 2026?

Fable 5.2, Opus 5.5, Sonnet 5.5, and Haiku 5.5 all have 1,000,000-token (1M) context windows and 128K max output as of November 13, 2026. Legacy Haiku 4.5 has a 200,000-token context window (64K output).

How many pages can Claude read at once?

At 1M tokens, Claude can hold roughly 750,000 words of English text — equivalent to approximately 3,000 average pages. In practice, a typical 20-page PDF is roughly 10,000-15,000 tokens, so you could load 60-100 such documents in a single session before approaching the limit.

Does the context window reset between messages?

No — the context window accumulates across an entire conversation session. Every message you send and every response Claude gives adds to the total. The window doesn't reset between individual messages; it resets when you start a new conversation.

What happens when Claude hits the context window limit?

When a conversation reaches the context window limit, earlier messages begin to drop out of the active context. Claude can no longer reference information from those earlier messages — it effectively forgets that part of the conversation. In the claude.ai interface, you'll see a notification when you're approaching the limit.

Is the 1M context window available on the free plan?

The model available to free plan users has access to the 1M context window. However, free plan usage limits mean long-context sessions hit rate limits faster than paid plans. The window is technically available, but sustained heavy use of it is more practical on paid tiers.

What's the difference between Claude Opus 5.5 and Sonnet 5.5 context windows?

Both Opus 5.5 and Sonnet 5.5 (as well as Fable 5.2 and Haiku 5.5) now support the 1M-token input context window and 128,000-token max output. Only legacy Sonnet 5 and legacy Haiku 4.5 capped output at 64,000 tokens.

■ Deploying Claude or AI Infrastructure in Your Business?

At Tygart Media, we engineer custom Model Context Protocol (MCP) servers, multi-model content pipelines, and AI operational systems. Explore our Claude AI Team Implementation Services or check out our complete Restoration Operations & AI Kit.

Primary & Reference Links Cited on This Page

- Anthropic models overview — `https://platform.claude.com/docs/en/about-claude/models/overview`
- API pricing — `https://platform.claude.com/docs/en/about-claude/pricing`
- Message Batches API — `https://platform.claude.com/docs/en/build-with-claude/batch-processing`
- Tygart Media: Claude AI Team Implementation Services
- Tygart Media: Restoration Operations & AI Kit