CLAUDE PROJECTS · GUIDE 2 OF 3

Claude Projects Under the hood

What actually happens to your instructions, files and memory, and what the new Code-side Projects do in the cloud.

Checked against Anthropic's docs on 10 November 2026.

Alex Kerss

SECTION 1

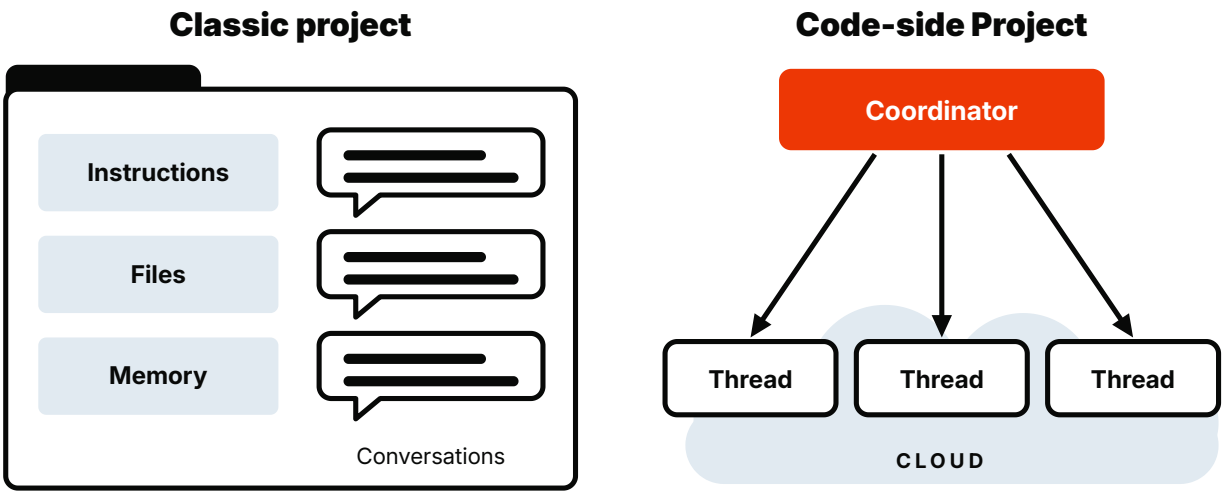
Two different features share one name

Claude has two kinds of Projects right now, and they work in completely different ways.

The first is the **classic project**. This is the one in the Claude app on the web, the desktop app and the mobile app, in Chat and in Cowork. A classic project groups conversations with a set of instructions, some knowledge files and its own memory. Each conversation inside it still runs on its own, one reply at a time.

The second is the **redesigned Project** on the Code side, which launched as a beta on 17 September 2026. It has a coordinator that plans the work and threads that carry it out, and the threads run in Anthropic's cloud. Anthropic describes the classic kind as grouping "conversations and reference files without threads or a coordinator".

The redesigned version hasn't reached Chat or Cowork. The docs say projects there "stay on the current version for now". So if you open Projects in the Claude tab, you're looking at a classic project, and everything in the first half of this guide applies.



	CLASSIC PROJECT	CODE-SIDE PROJECT
Where	Chat and Cowork (web, desktop, mobile)	Code on the web, the desktop Code tab, mobile
Who runs the work	You, one conversation at a time	Threads, directed by a coordinator
Runs on	Each conversation's own context	Cloud sessions, each with its own sandbox
Plans	All plans, Free included	Pro, Max and Ultra, in beta
Sharing	Team and Enterprise	Not shareable

SECTION 2

How a classic project fills the context window

When you start a conversation in a classic project, Claude doesn't look things up as it goes. By default the project uses what Anthropic calls in-context processing. Your project instructions and every knowledge file are loaded into the context window before your first message.

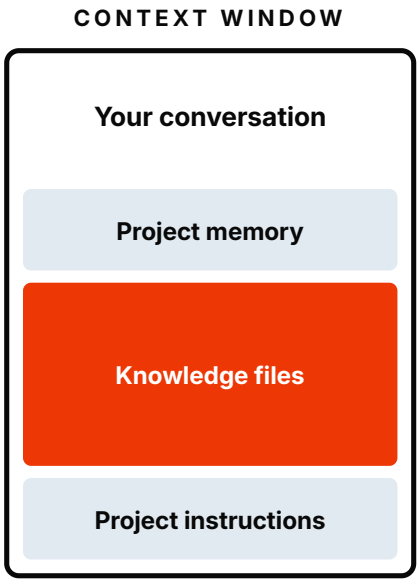
That has a few effects you can plan around.

First, Claude never sees the project's name or description. The docs are plain about this: "Claude will not have access to these details." Anything Claude needs to know has to go in the instructions or the files.

Second, the knowledge takes up room in every conversation in the project. A large knowledge base leaves less space for the conversation itself.

Third, conversations in the same project don't see each other. The only things they share are the knowledge files, the instructions and project memory. If one conversation produces a price list that a later one needs, add the price list to the project's knowledge.

The good news is that project content is cached. When Claude refers back to the same knowledge, it counts for less against your usage limits than pasting the same material into a fresh conversation would.



SECTION 3

File limits

Project knowledge has its own limits, and they're different from files you attach to an ordinary conversation.

	PROJECT KNOWLEDGE	ATTACHED TO A CONVERSATION
Size per file	30 MB	500 MB
Number of files	No fixed count, but the total has to fit the context window (until RAG takes over on paid plans)	20 per conversation
What Claude reads	Text only, except PDFs	
PDF pictures and charts	Read for PDFs of 100 pages or fewer	Up to 1,000 pages

So a 300-page PDF in project knowledge gets read as text only. Its charts and diagrams are skipped. If the visuals carry the meaning, split the PDF or pull the pages you need into a shorter file.

A few more limits apply:

- **Project count.** Free accounts can create up to five projects. No limit is documented for paid plans.
- **Instructions.** No character limit is documented for classic project instructions.
- **Google Drive.** You can add Drive files as knowledge, and Docs stay in sync, but only in private projects. Shared projects have Drive turned off.
- **GitHub.** Files from GitHub come from one branch, with no commit history and no pull requests. Press Sync to refresh them. Under one Claude (see below), adding from GitHub isn't supported.

SECTION 4

When knowledge outgrows the window: RAG

On a paid plan, a project can hold more than fits in the context window. When your knowledge gets close to the limit, Claude switches to RAG mode on its own. There's nothing to turn on. The docs say this expands a project's capacity "by up to 10x".

RAG stands for retrieval-augmented generation. Instead of loading every file, Claude searches the knowledge for each message and pulls in only the parts that look relevant.

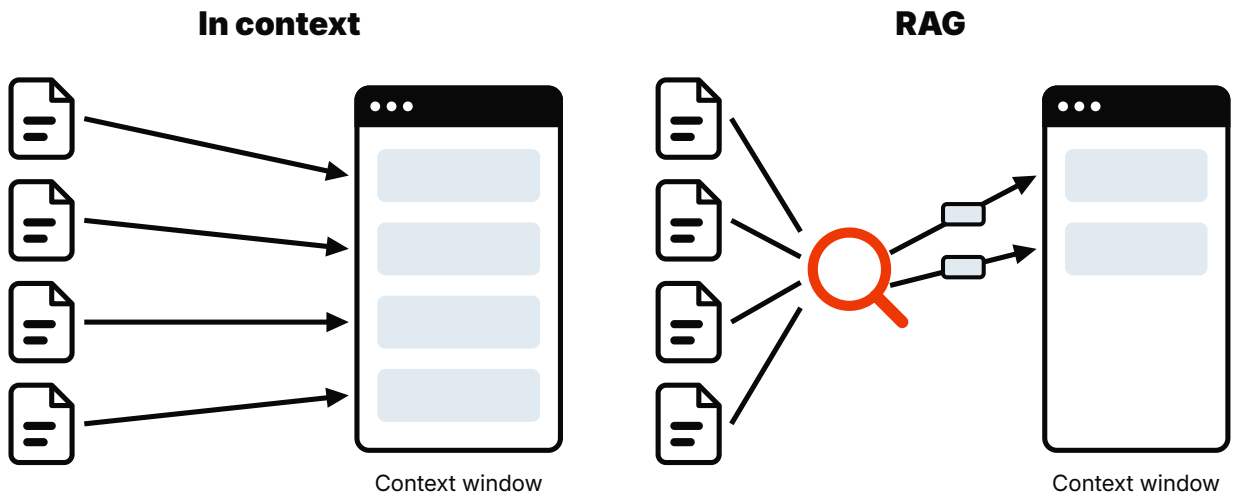
The switch changes how you should ask. With everything loaded, Claude can connect a line in one file with a table in another without being told. After the switch, Claude only sees what the search brings back. A question that doesn't point at the right document may be answered from the wrong one, or

from nothing.

Anthropic's own advice for RAG projects:

- Give files clear names that say what's in them.
- Keep related documents together.
- Name the document you mean in your question ("in the 2026 pricing sheet, what's the discount for...").

RAG is available on Pro, Max, Ultra, Team and Enterprise. On Free, the whole knowledge base has to fit the context window.



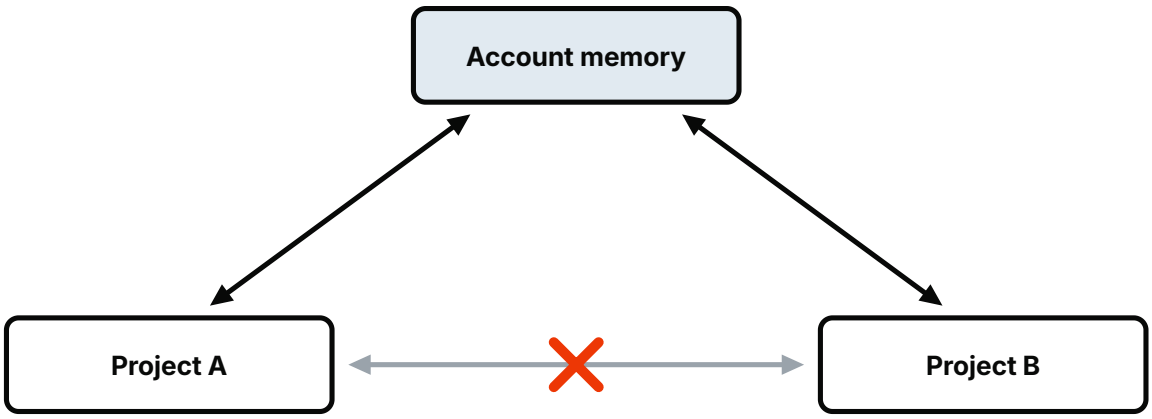
SECTION 5

Project memory and account memory

Each classic project keeps its own memory, separate from other projects. But it isn't sealed off from the rest of your account, and the connection runs in both directions.

Into the project. Inside a project, Claude reads the project's memory plus your account memory, the memory built from conversations outside any project. It doesn't read the memory of other projects.

Out of the project. New memory is saved to the project. If memory is on for your account, Claude can also refer to that memory in conversations outside projects. So a detail from a sensitive project can turn up in an unrelated conversation.



To control this:

1. On the project page, turn off **Use account memory**. The docs give this as the way to keep a project's memory to itself.
2. To keep one conversation out of memory altogether, turn off **Memory** in the **+** menu before you send the first message. That conversation then neither reads nor adds to project memory.

Memory is on by default on Free, Pro, Max and Ultra, on the web, desktop and mobile. On Team and Enterprise, the organisation's owners decide whether it's on.

The docs don't say how often project memory is updated. They only say new memory is saved to the project, and Claude may tell you it has saved something partway through a conversation.

SECTION 6

Sharing and privacy

Sharing classic projects is for Team and Enterprise only. On those plans, a new project is either private or shared with the organisation.

When you share a project, you pick one of two permission levels:

- **Can view.** The person sees the knowledge and instructions and can hold conversations in the project.
- **Can edit.** The person can also change the instructions and knowledge, manage who's in the project and, on Enterprise, add tags.

People you share with get an email, and the project appears in their **Shared with you** tab. A few details are worth knowing:

- Owners can turn project sharing off for the organisation. That also turns off public projects, but shares already made stay in place.
- You can't change sharing settings on mobile.

- On Enterprise, project tags are visible across the whole organisation, so keep anything sensitive out of tag values.

How Anthropic treats your data depends on the plan, not on the project. On Free, Pro, Max and Ultra, your conversations are used for training only if that setting is on, and they're kept for five years with it on or 30 days with it off. On Team, Enterprise and the API, Anthropic doesn't train on your data unless your organisation opts in.

SECTION 7

Cowork projects and one Claude

On 16 September 2026, Chat and Cowork started merging into one Claude. Once the merge reaches your account, Chat projects and Cowork projects appear as one list.

The rollout depends on your plan:

- **Pro, Max and Ultra** get it gradually.
- **Team** has it in beta.
- **Enterprise** turns it on with the "Chat and Cowork unified" setting, and it becomes the default for Enterprise on 16 November 2026.
- Organisations with zero data retention, HIPAA-ready setups or customer-managed keys can't move to one Claude.

The switch is one-way. Once it's on, you can't go back.

Projects made in Cowork are saved to your Claude account, and any local folders you attach stay on your computer. That split decides what still works with your computer off. Uploaded knowledge is on Anthropic's servers, so it's always there. A local folder needs the Claude desktop app open.

Folder access is different between the two modes:

- **Cowork** can read and write the files in a project's local folders.
- **Chat** in a project can read folders added after 5 October 2026, but can't change them.
- A project on claude.ai in the browser can't hold local folders at all.

Cowork projects can also carry their own scheduled runs, which use the project's context. If you still have an older Chat project, Cowork can import it, which copies its files and instructions. Projects come across one at a time, with no bulk import.

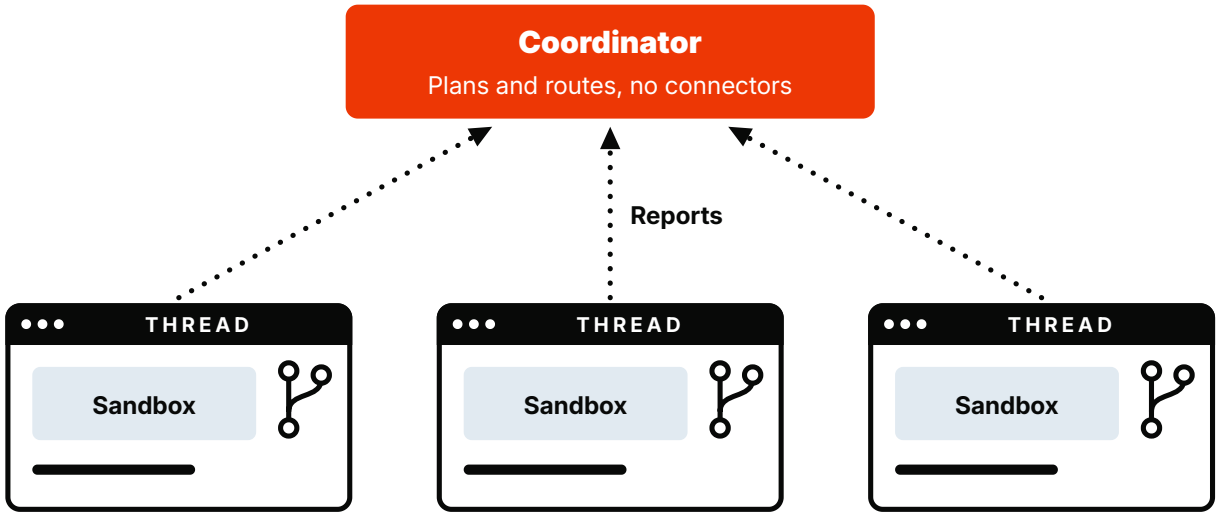
SECTION 8

Code-side Projects: the coordinator and its threads

A Code-side Project starts from a goal. You talk to the coordinator, and the coordinator doesn't do the work itself. It answers quick questions, sends new work to a new or existing thread, and may post suggested threads or setup recommendations for you to accept.

Each thread is a full Claude Code session in the cloud, with its own sandbox and its own copy of your code. Threads keep going after you close your laptop.

The coordinator sees what threads report back, not every step they take. Results stay in the threads, so open a thread to see its work. You can also talk to any thread directly.



A few rules about how the pieces connect:

- **Connectors.** The coordinator has none. Threads get the connectors on your claude.ai account.
- **Approvals.** Threads run in auto mode, and anything that needs your approval waits inside that thread. Saying "go ahead" to the coordinator doesn't reach it.
- **Instructions.** Project instructions go to the coordinator and to every new thread, up to 16,000 characters. They're under Project settings > Memory > Project instructions.
- **Memory.** Project memory is a set of notes Claude keeps as files, with a `MEMORY.md` index. It's separate from the auto memory Claude Code keeps on your own computer, even though both have a file with that name.
- **Overview.** The Threads tab sorts threads into Ready for review, Waiting on you, Working, Landing, Idle and Resolved. Other tabs hold the Library, Pull requests and Scheduled tasks.

Most threads run in the cloud, but you can send one thread at a time to your own computer with Remote Control. That thread only runs while the computer is awake. It needs Claude Code v2.1.280 or later, it's blocked if **Require trusted devices** is on, and it gets the project instructions but not the memory files.

SECTION 9

What spends your plan without you asking

Several defaults in a Code-side Project run work on their own. Each one uses your plan.

Opus at high effort. A new project runs Opus everywhere, with high effort for threads and low effort for the coordinator. Anthropic notes this draws on your plan fastest. Change both under Project settings > General.

Auto-continue. When your usage limit resets, project threads carry on automatically and start spending the new window. This is on by default. Turn it off if you'd rather restart work yourself. Going past your limits needs usage credits.

Threads that wake up. Threads open pull requests and then watch them. A CI failure or a review comment wakes an idle thread so it can respond.

Your first project moves first. When you create your first project, Claude takes a turn on its own straight away.

Reviving an idle thread. On Pro, Max and Ultra the prompt cache lasts one hour. A thread idle for longer re-reads its whole conversation when it starts again.

Thread limits. Telling the coordinator "use at most two threads" is a preference it follows, not a hard limit. The only enforced cap is 500 new threads a day across all your projects.

SECTION 10

What a cloud thread loads from your repo

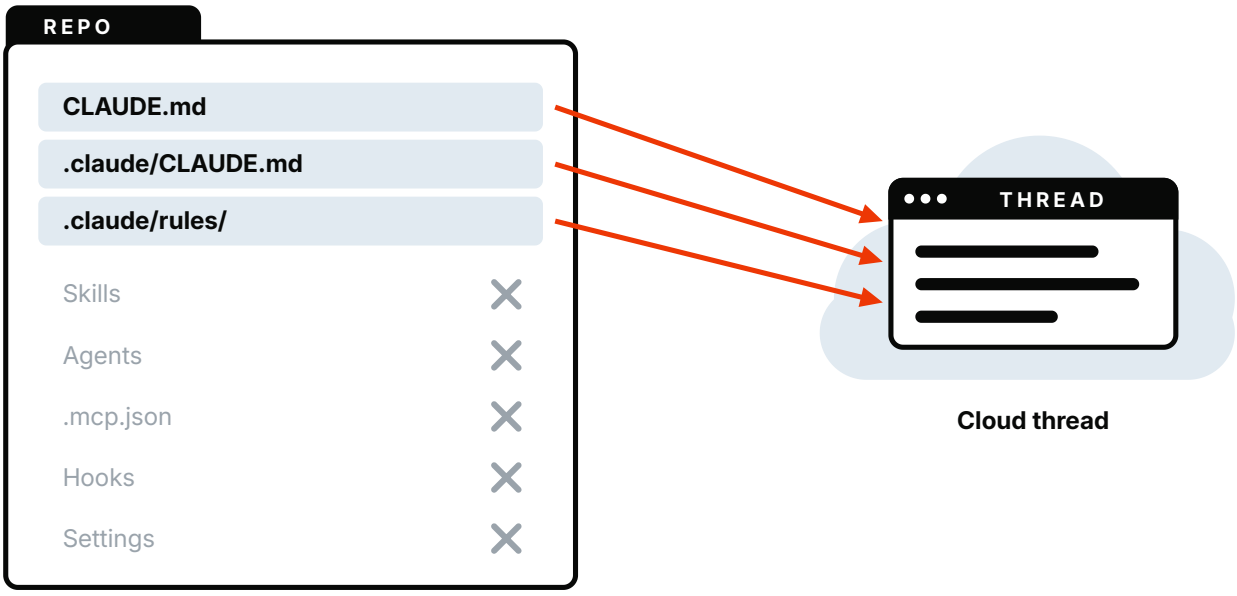
A cloud thread doesn't load your repository the way Claude Code on your computer does. Only three things come in automatically:

- `CLAUDE.md`
- `.claude/CLAUDE.md`
- the `.claude/rules/` folder at the root

Everything else in the repo's Claude setup is left out.

LOADED IN CLOUD THREADS	NOT LOADED
`CLAUDE.md` and `.claude/CLAUDE.md`	Repo skills, agents and commands
Root `.claude/rules/`	`.mcp.json` (MCP servers)
Skills turned on in your claude.ai account	Hooks
Plugins added in Project settings	Permission rules and `env` in settings

This changes what you can rely on. A rule enforced by a hook on your computer isn't enforced in a cloud thread. An MCP server in `.mcp.json` isn't there either. Put rules that must reach threads in `CLAUDE.md` or `.claude/rules/`. Turn on the skills you need in your claude.ai account, under **Plugins** with the skills filter, and add plugins in Project settings.



SECTION 11

Repos, folders and files

GitHub only. Code-side Projects need your code on github.com, push access to the repo and the Claude GitHub App installed. The token from `/web-setup` isn't enough. GitHub Enterprise Server, GitLab and Bitbucket aren't supported.

How a project starts. You start from scratch (a name, an optional goal, GitHub repos, files and Google Drive folders), or you move an existing cloud session in with **Continue in project**.

Folders are copies. When you add a folder, the project gets a copy of its first 100 files, up to 200 MB. Files over 30 MB and hidden files are skipped. Edits you make on your computer afterwards don't sync, so add the folder again if it changes.

LIMIT	VALUE
Files per upload	100 files or 2 GB
Size per uploaded file	480 MB (the New project dialog skips files over 30 MB)
Folder copy	First 100 files, up to 200 MB
Folders and Drive folders	10 per project

LIMIT	VALUE
New threads	500 a day

The sandbox can reset. A thread's cloud sandbox can be lost between turns. Have threads commit and push often, so nothing sits only in the sandbox.

Projects are fixed containers. Threads can't move between projects, and two projects can't be merged. Deleting a project can't be undone. It removes the threads, memory and files, but leaves your GitHub branches and pull requests alone.

One person per project. A Code-side Project belongs to one user. You can't share it or its threads, and there are no organisation controls during the beta.

SECTION 12

Plans and rollout

Code-side Projects are in public beta on Pro, Max and Ultra, and rolling out gradually. Team and Enterprise don't have them yet, and Anthropic hasn't given a date.

If Projects isn't in your Code sidebar, check under **More**. If it isn't there either, the rollout hasn't reached your account yet, and you can join the waitlist at claude.com/form/projects.

The rollout started with accounts that had used cloud sessions and had no existing projects in Chat or Cowork. If you have lots of classic projects, that's the most likely reason you don't see it yet.

Some existing Chat and Cowork projects on Pro, Max and Ultra are being moved into the new experience. The **Get started** button lists the claude.ai projects involved and asks you to upgrade or archive each one. Anthropic says nothing is lost or deleted in the move, and project scheduled runs come across too.

Code-side Projects work on Code on the web, in the desktop app's Code tab and in the mobile app. They aren't in the terminal CLI, VS Code or JetBrains, or through Bedrock, Vertex or Foundry.

SECTION 13

Nested sessions in the Code tab

Since 15 October 2026, a Code-tab session can start another session that sits underneath it. The new session has its own context, so the work it does doesn't fill up the original one. You can click into a nested session and steer it like any other, and it reports back to the session that started it.

Nested sessions only work in local and SSH sessions. They run wherever the parent runs, so they get the same setup as any local session: your skills, `.mcp.json`, hooks and settings. That's the opposite of a cloud thread.

In **Settings > Claude Code**, you choose what happens when you ask a session for something unrelated to what it's doing, such as starting a nested session for it.

No separate plan rule is documented for nested sessions. They come with the Code tab, which needs Pro, Max, Ultra, Team or Enterprise. That makes them the closest thing Team and Enterprise have to Code-side Projects for now.

Separately, a Code-tab session can list, read and message your other local, SSH and WSL sessions, by default the 20 most recent. It can't see cloud sessions or sessions in the terminal CLI.

	NESTED SESSIONS	CODE-SIDE PROJECTS
Runs on	Your computer or an SSH host	Anthropic's cloud
Loads	Your full local setup	Only `CLAUDE.md`, `.claude/CLAUDE.md` and root `.claude/rules/`
Shared instructions and memory	No, each session gets a brief from its parent	Yes, project instructions and memory
Keeps going with the laptop shut	No (an SSH host keeps running)	Yes
Plans	Any plan with the Code tab	Pro, Max, Ultra (beta)

Nested sessions aren't threads. The docs keep the word "thread" for work inside a Code-side Project.

AT A GLANCE

Quick reference

••• QUICK REFERENCE

Classic projects (Chat and Cowork)

- Instructions and all knowledge load into every conversation. Claude never sees the name or description.
- Knowledge: 30 MB per file. PDF visuals read only up to 100 pages. Free: up to five projects.
- Paid plans switch to RAG near the context limit, up to 10x capacity. Name the document you mean.
- Memory flows both ways with account memory. Turn off **Use account memory** to keep a project to itself, or **Memory** in the **+** menu before the first message.
- Sharing: Team and Enterprise, **Can view** or **Can edit**. Drive only in private projects.
- One Claude is one-way. Cowork reads and writes folders; Chat reads only.

Code-side Projects

- Coordinator routes, threads work in the cloud. Approvals happen inside threads. The coordinator has no connectors.
- Defaults: Opus, high effort for threads. Auto-continue on. Idle threads wake on CI failures and review comments. Your first project takes a turn straight away.
- Cloud threads load only `CLAUDE.md`, `.claude/CLAUDE.md` and root `.claude/rules/`.
- Needs github.com, push access and the Claude GitHub App.
- Folders: copies of the first 100 files, up to 200 MB, no sync.
- One user, no sharing. Pro, Max and Ultra beta, rolling out.

Nested sessions

- Local and SSH only, full local setup, any plan with the Code tab.