

A POWER-USER'S GUIDE

GROK EDITION

Maximum Autonomy

Every setting, flag, config file, and token choice for
hands-off AI agents, tool by tool

Maximum Autonomy

Every setting, flag, config file, and token choice for hands-off AI agents, tool by tool

GROK EDITION

Concise edition, revised and fact-checked by Grok Bot in GrokAllManualsV1.

This guide is for one goal: letting your AI agents work **hands-off**, with as few approval prompts as possible, across Claude Code, the Claude apps, Cursor, VS Code and Copilot, Codex, Gemini CLI, GitHub, MCP servers, n8n, and Notion.

You've decided you accept the risk, so this guide doesn't argue with you. It gives you the exact switches. It also shows the one habit that makes maximum autonomy sustainable: **contain the agent instead of restricting it**. Give it a box where mistakes are cheap, then remove the guardrails inside that box. Each chapter ends with what *still* prompts, no matter what you set.

HOW TO READ THIS MANUAL

Tip — a shortcut or good habit.

Warning — easy to get wrong.

\$ **command** — text you type, exactly as shown.

Note — useful background.

Danger — can lose data, money, or access.

Settings > **Privacy** — a menu path to click.

New words appear in **bold** the first time and are collected in the Glossary at the back.

Settings, flags, and menu paths were checked against each vendor's official documentation in October 2026. These products change monthly, and safety prompts in particular are added and renamed often. If a setting name doesn't match what you see, check the vendor's current docs. You are responsible for what your agents do with the access you grant.

Contents

01 The Autonomy Playbook	5
1.1 Four levels of autonomy	5
1.2 Contain, don't restrict	5
1.3 Quick-start: maximum autonomy at a glance	6
02 Build a Safe Playground	7
2.1 Option 1: a dev container (recommended for coding agents)	7
2.2 Option 2: a dedicated machine or cloud box	8
2.3 Option 3: a separate user account on your computer	8
2.4 Accounts, tokens, and money	8
03 Claude Code (Terminal)	9
3.1 Level 3: auto mode by default	9
3.2 Level 4: bypass permissions	10
3.3 Fully unattended runs	11
3.4 A complete maximum-autonomy settings file	11
3.5 Reduce prompts without bypass	12
3.6 What still prompts in Claude Code, no matter what	13
04 Claude Desktop, Cowork, and claude.ai	14
4.1 The Code tab in Claude Desktop	14
4.2 Cowork	15
4.3 Connectors in claude.ai and the Claude apps	15
4.4 Claude in Chrome	15
05 Cursor	16
5.1 The editor's agent	16
5.2 The Cursor CLI	17
06 VS Code and GitHub Copilot	18
6.1 Copilot agent in VS Code	18
6.2 Copilot CLI	19
6.3 Copilot coding agent (on GitHub)	19
07 Codex, Gemini CLI, and Builders	20
7.1 OpenAI Codex	20
7.2 Gemini CLI	20
7.3 Browser app builders	21
08 GitHub: Tokens and Repository Settings	22
8.1 Authenticate agents broadly	22

8.2	Repository settings that remove friction	23
8.3	What still prompts on GitHub	24
09	MCP Servers and Connectors	25
9.1	Give servers enough power	25
10	n8n	26
10.1	Let AI agents build and run your workflows	26
10.2	Self-hosted: unlock everything	26
11	Notion	28
11.1	Notion MCP (simplest, full personal access)	28
11.2	An API connection with every capability	28
11.3	Notion AI agents	28
12	What Always Asks: The Master List	30
13	Claude Code Permission Modes and Settings Reference	31
13.1	The six permission modes	32
13.2	Choosing a mode	32
13.3	Every settings key, A-Z	33
14	Autonomy Recipes	43
14.1	Golden rules of autonomy	43
15	Incident Response: When an Agent Goes Wrong	44
16	Codex config.toml Reference	46
16.1	Command-line autonomy flags (current release)	46
16.2	config.toml keys (292)	47
16.3	requirements.toml keys (174)	61
17	Gemini CLI settings.json Reference	71
	Your Autonomy Setup Checklist	81
	Glossary	82

The Autonomy Playbook

IN THIS CHAPTER

- The four levels of agent autonomy
- “Contain, don’t restrict”: the principle behind this guide
- The quick-start setup for each tool

1.1 Four levels of autonomy

Every agent tool offers some version of these four levels:

LEVEL	WHAT HAPPENS	EXAMPLES
1. Ask	The agent asks before most actions	Claude Code <i>Manual</i> , Cursor <i>Allowlist</i> with an empty list
2. Allowlist	Pre-approved actions run; everything else asks	Allow rules, terminal allowlists
3. Classifier	A second AI reviews each action and blocks only risky ones	Claude Code <i>Auto</i> , Cursor <i>Auto-review</i>
4. Bypass	Everything runs. No checks except a few hard-coded ones	<code>--dangerously-skip-permissions</code> , Cursor <i>Run Everything</i> , Codex <code>--yolo</code>

Level 3 gets you about 95% of the hands-off feel while still catching disasters. Level 4 is true maximum autonomy. This guide gives you both, so you can choose per tool and per project.

1.2 Contain, don’t restrict

Prompts exist to stop an agent from damaging things that matter. You can get the same protection a better way: put the agent somewhere where **nothing it can reach matters too much**. Then you can safely turn the prompts off.

- **A disposable machine:** a container, a virtual machine, a cloud dev box, or a spare computer.

- **Separate accounts:** a dedicated GitHub account or bot for the agent, a separate Notion workspace or teamspace, a separate n8n instance.
- **Narrow-but-complete tokens:** full power *inside* the agent's projects, nothing outside them.
- **Budgets:** spending caps on every API and cloud account.
- **Backups and Git:** everything committed, everything restorable.

Chapter 2 shows how to set this up in an afternoon.

☆ IMPORTANT

The single biggest real-world risk of hands-off agents is **prompt injection**: instructions hidden in a web page, email, issue, or document that the agent reads and obeys. Containment limits the damage it can do. Bypass settings remove the last line of defense against it.

1.3 Quick-start: maximum autonomy at a glance

TOOL	HANDS-OFF WITH SAFETY REVIEW	FULL BYPASS
Claude Code (CLI)	<code>"defaultMode": "auto" in ~/.claude/settings.json</code>	<code>claude --dangerously-skip-permissions</code>
Claude Desktop, Code tab	Mode selector, then Auto	Settings > Claude Code > Allow bypass permissions mode , then Bypass permissions
Claude Cowork	Mode selector, then Automatically approve	(no separate bypass)
claude.ai connectors	Each tool set to Always allow	(same)
Cursor	Approvals & Execution: Auto-review	Approvals & Execution: Run Everything
VS Code + Copilot	Assisted permissions (experimental)	Allow all / Autopilot , <code>chat.tools.global.autoApprove</code>
Copilot CLI	Allow rules	<code>copilot --yolo</code> (alias of <code>--allow-all</code>)
OpenAI Codex	<code>--sandbox workspace-write</code> or <code>--approve-for-me</code>	<code>codex --yolo</code> , or <code>approval_policy = "never" + sandbox_mode = "danger-full-access"</code>
Gemini CLI	<code>--approval-mode=auto_edit</code>	<code>gemini --yolo</code>
GitHub	Bot account + fine-grained token	Broad token + Actions write permissions + auto-merge
n8n	Instance-level MCP + API key	Plus <code>NODES_EXCLUDE="[]"</code> on self-hosted
Notion	Notion MCP (OAuth)	Internal connection with all capabilities, shared at the top level

Build a Safe Playground

Fifteen minutes of setup here lets you run everything else in this guide at full throttle.

2.1 Option 1: a dev container (recommended for coding agents)

A **dev container** is a Docker container configured as a complete development environment. Anthropic publishes a reference dev container for Claude Code that runs as a **non-root user** with a network firewall, which is exactly the setup bypass mode is designed for.

RUN AGENTS IN A DEV CONTAINER

- 1 **Install Docker Desktop** (docker.com) and **VS Code** with the **Dev Containers** extension.
- 2 **Add a `.devcontainer/` folder** to your project. Copy Anthropic's reference configuration from the Claude Code repository (`anthropics/claude-code` , folder `.devcontainer`), or ask Claude Code: “set up a dev container for this project based on the official Claude Code devcontainer.”
- 3 **Open the project in the container.**
Command Palette, then **Dev Containers: Reopen in Container**.
- 4 **Run your agent inside it** in bypass mode. If it goes wrong, rebuild the container; your host machine is untouched.

NOTE

On Linux and macOS, Claude Code refuses `--dangerously-skip-permissions` when run as **root** or with `sudo`, except inside a recognized sandbox. Dev containers and a normal user account both satisfy this.

2.2 Option 2: a dedicated machine or cloud box

A spare laptop, a small cloud VM, or a cloud dev environment (GitHub Codespaces, a cloud VM, or Claude Code's own cloud sessions at claude.ai/code) all work. Give it its own accounts and tokens, and nothing personal.

2.3 Option 3: a separate user account on your computer

Create a second user account on your Mac or PC just for agents. Your personal files, browser logins, and SSH keys stay out of reach.

2.4 Accounts, tokens, and money

- ☐ A separate GitHub account (or GitHub App) for agents, added as a collaborator only on the repos they work on
- ☐ Fine-grained tokens scoped to those repos, with long-enough expiry
- ☐ A separate Notion teamspace or workspace for agent-writable content
- ☐ Hard monthly spend limits on Anthropic, OpenAI, Google, and cloud accounts
- ☐ Automatic backups (Time Machine, File History, or cloud backups) of anything the agent can write
- ☐ Everything in Git, pushed often, with `main` protected (Chapter 8)

Claude Code (Terminal)

Claude Code has the most complete autonomy controls of any tool. Two settings files matter:

- `~/ .claude/settings.json` : **user** settings. Your machine-wide defaults.
- `.claude/settings.json` (project) and `.claude/settings.local.json` (project, just you).

⚠ WARNING

`"defaultMode": "auto"` and `"defaultMode": "bypassPermissions"` are **ignored** in project files (`.claude/settings.json` and `.claude/settings.local.json`); this stops a cloned repository from switching your agent to bypass. Put them in `~/ .claude/settings.json`, in managed settings, in a `--settings` file, or pass the flag.

3.1 Level 3: auto mode by default

Recent versions of Claude Code already start interactive sessions in **auto mode**, where a classifier model reviews risky actions instead of asking you. To make it explicit and permanent:

~/ .CLAUDE/SETTINGS.JSON

```
{
  "permissions": {
    "defaultMode": "auto"
  }
}
```

Then reduce the classifier's false alarms:

- **Teach it your infrastructure.** Run `/auto-mode-setup` to draft `autoMode.environment` entries (trusted repos, buckets, domains, services) from your project and recent sessions, and save them.

- **Approve standing patterns.** Tell it in plain English with specifics (“you may force-push to branches starting with `agent/`”), or add rules under `autoMode.allow`.
- **Check the rules:** `claude auto-mode defaults` prints the built-in block/allow lists; `claude auto-mode config` shows your effective config.

NOTE

If the classifier blocks 3 actions in a row or 20 in total, auto mode pauses and asks you. These thresholds can't be changed. In `claude -p` runs with no prompt available, the blocked action is skipped and Claude keeps working.

3.2 Level 4: bypass permissions

One session:

TERMINAL

```
claude --dangerously-skip-permissions
```

(Equivalent: `claude --permission-mode bypassPermissions`.) The first time, you accept a warning; Claude Code then records `"skipDangerousModePermissionPrompt": true` in `~/ .claude/settings.json` so it doesn't ask again.

Every session, by default:

~/ .CLAUDE/SETTINGS.JSON

```
{
  "permissions": {
    "defaultMode": "bypassPermissions"
  },
  "skipDangerousModePermissionPrompt": true
}
```

Start in another mode but keep bypass one keypress away: `claude --allow-dangerously-skip-permissions` adds bypass to the `Shift` + `Tab` cycle without starting in it.

A shell alias for convenience:

~/ .ZSHRC OR ~/ .BASHRC

```
alias yolo='claude --dangerously-skip-permissions'
```

3.3 Fully unattended runs

For scripts, scheduled jobs, and CI, use print mode:

TERMINAL

```
claude -p "Fix the failing tests and commit the result" \  
  --dangerously-skip-permissions \  
  --permission-prompts none \  
  --max-turns 60 \  
  --max-budget-usd 10
```

- `--permission-prompts none` denies anything that would still need a human, instead of waiting forever.
- `--max-turns` and `--max-budget-usd` are your seat belts.
- `--output-format json` gives machine-readable results.
- `claude --bg "..."` starts a **background agent** that keeps running after you close the terminal; manage it with `claude agents`. Background sessions honor bypass only if your acceptance is recorded in `~/claude/settings.json`.
- In CI without bypass, use `--permission-mode dontAsk` with an exact `--allowedTools` list.

3.4 A complete maximum-autonomy settings file

~/CLAUDE/SETTINGS.JSON

```
{  
  "permissions": {  
    "defaultMode": "bypassPermissions",  
    "allow": [  
      "Bash",  
      "Edit",  
      "Write",  
      "WebFetch",  
      "WebSearch",  
      "mcp__github__*",  
      "mcp__notion__*"  
    ],  
    "additionalDirectories": ["~/projects"]  
  },  
  "skipDangerousModePermissionPrompt": true,  
  "enableAllProjectMcpServers": true,  
  "env": {  
    "CLAUDE_CODE_DISABLE_DANGEROUS_RM_TIMEOUT": "1"  
  }  
}
```

What each line buys you:

SETTING	EFFECT
<code>defaultMode: bypassPermissions</code>	Every session starts with no prompts
<code>allow rules</code>	Still useful when you drop into auto or manual mode: these tools stop asking. (Allow rules have no effect <i>in</i> bypass, which already allows everything; and broad rules like <code>Bash</code> are dropped while in auto mode.)
<code>mcp__server__*</code>	Approves every tool from that MCP server. Allow globs must name the server; <code>mcp__*</code> alone is ignored
<code>additionalDirectories</code>	Lets Claude read and edit outside the folder you started in without asking
<code>enableAllProjectMcpServers</code>	Auto-approves servers listed in projects' <code>.mcp.json</code> files
<code>CLAUDE_CODE_DISABLE_DANGEROUS_RM_TIMEOUT</code>	Removes the 2-minute countdown on critical deletions (the prompt itself stays)

TIP

Workspace trust: the first time you open a folder, Claude Code asks whether you trust it. Answer once per repository and it remembers. For automation, `-p` runs don't show the dialog, though project allow rules aren't applied there until a folder is trusted.

3.5 Reduce prompts without bypass

If you'd rather stay in Manual or Accept-edits mode for some projects:

- `/fewer-permission-prompts` scans your history and proposes an allowlist.
- `/sandbox`, then **auto-allow**, runs shell commands inside an OS-level sandbox *without prompting* (macOS, Linux, WSL2).
- Answer prompts with **Yes, and don't ask again** to save rules automatically.

3.6 What still prompts in Claude Code, no matter what

STILL ASKS, EVEN IN BYPASS	WHY / WORKAROUND
Tools matching an explicit ask rule	Remove the ask rule
<code>rm / rmdir</code> targeting a critical path: <code>/</code> , top-level folders, your home folder, the working directory or its parents	A deliberate circuit breaker. In bypass it asks (with a 2-minute countdown that auto-denies). Use literal subfolder paths instead
<code>AskUserQuestion</code> and MCP tools marked <code>requiresUserInteraction</code>	By design; they need an answer from a person
Connector tools your organization set to "ask"	Admin policy
Reads outside working directories if <code>blockReadsOutsideWorkingDirectories</code> is on	Leave it off, or add directories
Some cross-session messaging approvals	Only matters with multiple agent sessions messaging each other
<code>Remove-Item</code> on system paths or bare wildcards (PowerShell)	Denied outright in every mode
Cloud sessions (claude.ai/code)	Bypass isn't offered; Auto is the maximum

Claude Desktop, Cowork, and claude.ai

4.1 The Code tab in Claude Desktop

The Code tab uses the same engine and settings files as the CLI.

ENABLE BYPASS IN THE DESKTOP APP

- 1 **Open Settings** in Claude Desktop and go to **Claude Code**.
- 2 **Turn on “Allow bypass permissions mode.”** On Pro and Max plans this is a personal toggle; on Team and Enterprise, your organization’s policy controls it.
- 3 **Choose the mode.**
In a session, open the mode selector next to the send button (or press `Cmd` + `Shift` + `M`) and pick **Bypass permissions**, or **Auto** for classifier-reviewed autonomy.
- 4 **It sticks per folder.**
The desktop app remembers the mode you pick for each project folder (except Plan). A `defaultMode` in `~/ .claude/settings.json` sets the starting point for new folders.

Still asks in the desktop app: the Claude Code list from Chapter 3, plus **safety classifiers on actions Claude takes on external websites** in the built-in browser (any mode), and destructive desktop actions such as deleting a session. On websites, Claude won’t purchase things, create accounts, or solve CAPTCHAs without you. Answer a site’s first action card with **Always allow** to stop repeat prompts for that site.

4.2 Cowork

Cowork (a tab in Claude Desktop) does longer agentic work on your files, in your browser, and through connectors.

- In the Cowork mode selector, choose **Automatically approve**. On Team and Enterprise, an admin must leave **Allow “Automatically approve” mode** on (it’s on by default) in **Organization settings** > **Cowork** > **Permissions**.
- For connector **write** tools to stop asking every task, your admin must also turn on **Allow “Always allow” for connector tools** (off by default on Team and Enterprise). Then choose **Allow for all tasks** in the approval dialog.

4.3 Connectors in claude.ai and the Claude apps

Each connector’s tools can be set to **Always allow**, **Needs approval**, or **Blocked**.

MAKE CONNECTORS HANDS-OFF

- 1 Open **Settings** > **Connectors** (or **Customize**, then **Connectors**, in the desktop app).
- 2 Click a connector, then **Tool permissions**.
- 3 Set each category or tool to **Always allow**.
Do this for read and write tools you want the agent to use freely.
- 4 Repeat for each connector.

NOTE

Custom connectors’ tools usually lack “read-only” labels, so they default to asking. Set them to **Always allow** explicitly. Organization owners can cap what members may allow, and that cap can’t be overridden by individuals.

4.4 Claude in Chrome

The Chrome extension asks per site the first time Claude acts there. Choose **Always allow** for sites you want it to operate on freely. It will still stop for purchases, account creation, and CAPTCHAs, and its safety classifier can still raise a prompt on risky page actions.

Cursor

5.1 The editor's agent

TURN ON RUN EVERYTHING

- 1 **Open settings:**
`Settings` > `Agents` > `Approvals & Execution` .
- 2 **Set the Run Mode to Run Everything.**
Every tool call, including shell commands, MCP tools, and web fetches, runs automatically, with no sandbox and no classifier.
- 3 **Read Access** is hidden in Run Everything (nothing is checked). In other modes, set it to **System** to allow reads outside the workspace without asking.

Prefer a safety net? **Auto-review** runs allowlisted calls immediately, sandboxes shell commands where it can, and sends the rest to a classifier. Tune it in plain English in

`~/.cursor/permissions.json` :

`~/.CURSOR/PERMISSIONS.JSON`

```
{
  "autoRun": {
    "allow_instructions": [
      "Any git command in this project, including push to branches starting with agent/.",
      "Installing packages with npm, pnpm, or pip."
    ],
    "block_instructions": []
  }
}
```

The sandbox's network access and extra writable paths live in `~/.cursor/sandbox.json` .

5.2 The Cursor CLI

The command-line agent reads permissions from `~/.cursor/cli-config.json` (global) or `.cursor/cli.json` (project):

```
~/.CURSOR/CLI-CONFIG.JSON
```

```
{
  "permissions": {
    "allow": ["Shell(*)", "Read(**)", "Write(**)"],
    "deny": []
  }
}
```

For one-off unattended runs, the CLI's `--force` option runs commands without prompting.

Still asks in Cursor: on a team plan, admin policies and Cursor's built-in security rules sit on top of your local files and can't be weakened by them. In Auto-review or Allowlist modes, commands that need to leave the sandbox ask first, as do writes to protected paths like `.git/config`, `.git/hooks`, `.vscode`, and `.cursorignore`.

VS Code and GitHub Copilot

6.1 Copilot agent in VS Code

LEVEL	HOW
Allow all	Choose Allow all in the chat session's permission picker: "runs all tool calls without confirmation"
Autopilot	Choose Autopilot in the mode picker: auto-approves tools, retries errors, and answers blocking questions itself
Advanced Autopilot (preview)	<code>"chat.autopilot.advanced.enabled": true</code> , where a separate model decides when the task is complete
Every workspace	<code>"chat.tools.global.autoApprove": true</code>
This session only	Type <code>/yolo</code> (or <code>/autoApprove</code>); undo with <code>/disableYolo</code>

VS CODE SETTINGS.JSON

```
{
  "chat.tools.global.autoApprove": true,
  "chat.autopilot.advanced.enabled": true
}
```

Finer-grained alternatives: `chat.tools.terminal.autoApprove` (terminal command rules) and `chat.tools.urls.autoApprove` (web addresses). Terminal writes detected *outside* the workspace still ask by default; that's governed by `chat.tools.terminal.blockDetectedFileWrites`.

Running the **Claude agent harness inside VS Code**? Its equivalent of bypass is

`"github.copilot.chat.claudeAgent.allowDangerouslySkipPermissions": true`. The **Claude Code extension** has its own **Allow dangerously skip permissions** toggle in its settings, after which **Bypass permissions** appears in its mode indicator.

Still asks in VS Code: tools or terminal rules you've set to `false`; organization (device-management) policies; and, unless you change `blockDetectedFileWrites`, terminal writes it detects outside the workspace. An enabled sandbox still limits file and network access even in

Allow all.

6.2 Copilot CLI

TERMINAL

```
copilot --yolo
```

`--yolo` is an alias for `--allow-all`, which combines `--allow-all-tools`, `--allow-all-paths`, and `--allow-all-urls`. Inside a session, `/yolo` or `/allow-all` does the same; `/reset-allowed-tools` revokes it. Business and Enterprise admins can block these.

6.3 Copilot coding agent (on GitHub)

Assign an issue to Copilot and it works in its own GitHub Actions environment, then opens a pull request. It already runs without prompts. Widen what it can reach in the repository's **Settings** > **Copilot** > **Coding agent**: add a **Custom allowlist** for its firewall, or turn the firewall off entirely (maximum reach, maximum exfiltration risk), and add MCP servers.

CHAPTER 7

Codex, Gemini CLI, and Builders

7.1 OpenAI Codex

GOAL	HOW
Fewer prompts, sandboxed	<code>codex --sandbox workspace-write</code> (agent edits inside the workspace), or <code>codex --approve-for-me</code> (workspace-write sandbox plus automatic review of approval requests). The old <code>--full-auto</code> flag is gone from current releases.
No prompts, no sandbox	<code>codex --yolo</code> (alias of <code>--dangerously-bypass-approvals-and-sandbox</code>)
Make it the default	Put the two keys below in <code>~/.codex/config.toml</code>

~/.CODEX/CONFIG.TOML

```
approval_policy = "never"
sandbox_mode = "danger-full-access"
```

In the Codex app and IDE extension, choose **Full Access** from the permissions menu.

7.2 Gemini CLI

TERMINAL

```
gemini --yolo
# or
gemini --approval-mode=yolo
```

YOLO mode can only be turned on from the command line, so use an alias (`alias gem='gemini --yolo'`). The settings file `~/.gemini/settings.json` supports `general.defaultApprovalMode` of `"default"`, `"auto_edit"` (auto-approve file edits), or `"plan"`. YOLO also switches on Gemini's

sandbox by default, which doesn't add prompts. Leave it on as free protection.

Still asks: if `security.disableYoloMode` or the admin setting `admin.secureModeEnabled` is true, YOLO is refused. Untrusted folders (folder trust) restrict what runs until you trust them.

7.3 Browser app builders

TOOL	MAX-AUTONOMY SETTING
Replit Agent	In the Agent's tools panel, set Autonomy Level to Max for long, unsupervised runs (labeled beta)
Lovable, Bolt, v0, Base44	They act without per-step approvals already. Give them a connected GitHub repo and backend, and let them run; review via version history

CHAPTER 8

GitHub: Tokens and Repository Settings

GitHub is where agents push code, open pull requests, and trigger automation. Here's how to give them full power with minimal friction.

8.1 Authenticate agents broadly

The GitHub CLI (used by Claude Code, Codex, and others):

TERMINAL

```
gh auth login
gh auth refresh -h github.com -s repo,workflow,read:org,gist,delete_repo
```

The `workflow` scope lets agents edit `.github/workflows/` files; without it, pushes that touch workflows are rejected. `delete_repo` is only needed if you want agents to delete repositories.

A fine-grained personal access token (for MCP servers, n8n, and scripts): go to [Settings](#) > [Developer settings](#) > [Personal access tokens](#) > [Fine-grained tokens](#), then **Generate new token**:

OPTION	MAXIMUM-AUTONOMY CHOICE
Resource owner	Your account (or the organization the agent works in)
Expiration	As long as you're comfortable with, or custom
Repository access	All repositories (or every repo the agent should touch)
Contents	Read and write
Pull requests	Read and write
Issues	Read and write
Workflows	Read and write
Actions	Read and write
Administration	Read and write (only if agents create repos or change settings)
Metadata	Read (required)

Classic tokens are simpler but blunter: the `repo` and `workflow` scopes cover most agent work.

For teams: install the **Claude GitHub App** (run `/install-github-app` in Claude Code) on **All repositories** so `@cLaude` works everywhere.

8.2 Repository settings that remove friction

SETTING	WHERE	CHOOSE
Workflow permissions	<code>Settings</code> > <code>Actions</code> > <code>General</code>	Read and write permissions
Let Actions open and approve PRs	Same page	Tick Allow GitHub Actions to create and approve pull requests
Auto-merge	<code>Settings</code> > <code>General</code> , under Pull Requests	Tick Allow auto-merge ; agents can then run <code>gh pr merge --auto --squash</code>
Branch rules	<code>Settings</code> > <code>Rules</code> > <code>Rulesets</code>	Remove required reviews, or add the agent's account or app to the ruleset's Bypass list
Delete merged branches	<code>Settings</code> > <code>General</code>	Tick Automatically delete head branches

TIP

A powerful, still-sane setup: leave `main` protected by status checks only (tests must pass), put your agent's account on nothing else, and turn on auto-merge. Agents ship on their own, but only green code lands.

8.3 What still prompts on GitHub

- **Sudo mode:** sensitive actions in the web interface (deleting repos, changing security settings) re-ask for your password or 2FA. Tokens and the API aren't affected.
- **Push protection:** pushes containing detected secrets are blocked until bypassed for that push.
- **Claude's auto mode** blocks merging pull requests no human approved, and force pushes, by default. Use bypass mode, or grant a specific standing approval, if you want agents to do that.
- **Organization policies** (SSO token authorization, required reviews, Copilot policies) override your personal choices.

MCP Servers and Connectors

Every MCP tool call can trigger an approval prompt. Here's how to approve them up front in each host.

HOST	AUTO-APPROVE MCP TOOLS
Claude Code	Allow rules: <code>"mcp__github__*"</code> , <code>"mcp__notion__*"</code> , one per server (a bare <code>mcp__*</code> is ignored). Or bypass mode. <code>"enableAllProjectMcpServers": true</code> skips approving <code>.mcp.json</code> servers
Claude apps	Connector, then Tool permissions , then Always allow
Cursor	Run Everything covers MCP calls; Auto-review sends them to the classifier
VS Code	Allow all , Autopilot, or <code>chat.tools.global.autoApprove</code>
Codex	<code>approval_policy = "never"</code>
Gemini CLI	<code>--yolo</code> , or per server <code>"trust": true</code> in its <code>mcpServers</code> entry
n8n AI Agent	Don't add human-review steps to its tools

9.1 Give servers enough power

- **Remote OAuth servers** (GitHub, Notion, Linear...) act with *your* account's permissions. Approve every scope they request during sign-in.
- **Token-based servers:** create the token with the broad scopes from Chapter 8 (GitHub) or full capabilities (Notion, Chapter 11).
- **Local servers:** they run with your user's file and network permissions. Point filesystem servers at the folders you want writable.

Still asks: tools a server marks as needing user interaction (`requiresUserInteraction`) always prompt in Claude Code, and so do connector tools your organization set to "ask." ChatGPT's developer-mode connectors may still ask you to confirm write actions.

n8n

10.1 Let AI agents build and run your workflows

OPEN N8N TO AI CLIENTS

1 Create an API key:

Settings > **n8n API** > **Create an API key**. Copy it into scripts, Claude Code, or other tools that call n8n's REST API.

2 Enable instance-level MCP:

Settings > **Instance-level MCP**, then **Enable MCP access** (owner or admin). Use **Connect** to get the server URL (ending in `/mcp-server/http`) and connect Claude, Cursor, Codex, or ChatGPT with OAuth or an API key.

3 Expose workflows:

under **Workflows enabled**, click **Enable workflows** and add the ones agents may run, or in each workflow's **Settings** toggle **Available in MCP**. Clients can build and edit workflows (n8n 2.13+) as well as run them.

4 Agents in workflows:

in AI Agent nodes, attach the tools you want and *don't* add human-in-the-loop approval to them.

10.2 Self-hosted: unlock everything

TERMINAL

```
# Re-enable nodes n8n 2.0+ blocks by default (Execute Command, etc.)  
NODES_EXCLUDE="[]"
```

Set it in your Docker or Docker Compose environment and restart. **Execute Command** runs shell commands on the n8n host. It's powerful and dangerous, so keep self-hosted n8n in a container.

Still asks / limits: n8n Cloud doesn't offer Execute Command at all. The MCP server only lets clients run or see the full contents of workflows you enable for it (`search_workflows` previews everything you can access). Admins can disable MCP or Agents instance-wide.

Notion

11.1 Notion MCP (simplest, full personal access)

Connect `https://mcp.notion.com/mcp` from Claude (**Settings** > **Connectors** > **Notion**), Claude Code, Cursor, or ChatGPT. Signing in with OAuth gives the AI exactly **your** access: every page you can see and edit. Then set its tools to **Always allow** in the host app (Chapter 9).

11.2 An API connection with every capability

For n8n, Zapier, and scripts:

CREATE AN ALL-ACCESS CONNECTION

- 1 **Open** `notion.com/profile/integrations` (or **Settings** > **Connections** > **Develop or manage integrations**) and create a new **internal connection** for your workspace. You must be a workspace owner.
- 2 **Enable every capability:**
read content, update content, insert content, read comments, insert comments, and read user information including email addresses.
- 3 **Grant content access broadly.**
In the connection's **Content access** tab, add every top-level page in each teamspace you want covered. Child pages inherit access.
- 4 **Copy the API token** into your tools.

11.3 Notion AI agents

Custom Agents run on schedules or triggers. Give each agent the sources and tools it needs in the Agent Builder, and review its run log rather than approving each action. On Business and Enterprise plans, admins control who can create agents, and Custom Agents use paid Notion credits.

Still limited: a connection can never see pages you haven't shared with it. Workspace owners can restrict or revoke MCP clients in [Settings](#) > [Connections](#) . Agents run inside Notion's own permission system and can't exceed the access of the content shared with them.

What Always Asks: The Master List

These are the prompts and blocks that survive every setting in this guide. They're worth knowing so you're not surprised when a "fully autonomous" agent pauses.

TOOL	ALWAYS ASKS OR BLOCKS
Claude Code	Critical-path <code>rm / rmdir</code> (root, home, top-level dirs, working dir); explicit ask rules; <code>AskUserQuestion ; requiresUserInteraction</code> MCP tools; org "ask" connector policies; PowerShell <code>Remove-Item</code> on system paths; running bypass as root outside a sandbox
Claude Desktop / Chrome	Purchases, account creation, CAPTCHAs; classifier-flagged actions on external sites; deleting sessions
Cowork	Write connector tools when the admin hasn't allowed "Always allow"
Claude cloud sessions	No bypass; Auto is the ceiling
Cursor	Team admin policies; hard-coded security rules; (outside Run Everything) protected paths and commands leaving the sandbox
VS Code / Copilot	Tools set to <code>false</code> ; organization policies; enabled sandbox limits
Codex / Gemini CLI	Admin "secure mode" or disabled-YOLO settings; untrusted folders (Gemini)
GitHub	Sudo mode in the web UI; push protection for secrets; org SSO and policies
n8n	Workflows not enabled for MCP; Execute Command unavailable on Cloud
Notion	Pages not shared with a connection; owner-level MCP restrictions
ChatGPT agent	Confirmations before high-impact actions (purchases, sending); "watch mode" on sensitive sites

Claude Code Permission Modes and Settings Reference

This chapter is the exhaustive reference behind Chapter 3. The tables were generated from Anthropic's permission-modes page (code.claude.com/docs/en/permission-modes) and settings reference (code.claude.com/docs/en/settings-reference) in October 2026.

IN THIS CHAPTER

- The six permission modes, compared
- Which mode to choose for each situation
- Every documented `settings.json` key, A-Z

13.1 The six permission modes

MODE	WHAT RUNS WITHOUT ASKING	BEST FOR
<code>default</code>	Reads only	Reviewing every action yourself, sensitive work
<code>acceptEdits</code>	Reads, file edits, and common filesystem commands (<code>mkdir</code> , <code>touch</code> , <code>mv</code> , <code>cp</code> , etc.)	Iterating on code you're reviewing
<code>plan</code>	Reads, plus classifier-approved commands when auto mode is available	Exploring a codebase before changing it
<code>auto</code>	Everything, with background safety checks	Long tasks, reducing prompt fatigue
<code>dontAsk</code>	File reads inside your working directories and pre-approved tools; anything that would prompt is denied	Locked-down CI and scripts
<code>bypassPermissions</code>	Everything	Isolated containers and VMs only

13.2 Choosing a mode

YOU WANT TO...	START WITH	ISOLATION NEEDED
Review every action yourself	<code>claude --permission-mode default</code>	None
Explore before changing anything	<code>claude --permission-mode plan</code>	None
Work hands-off with background safety checks	<code>claude --permission-mode auto</code>	None required. A sandbox or container adds defence in depth
Run in CI with an exact allowlist	<code>claude -p "run the test suite" --permission-mode dontAsk --allowedTools "Bash(npm test)" "Read"</code>	Your CI runner
Run fully unattended	<code>claude -p "<prompt>" --dangerously-skip-permissions</code>	Required: a container, VM, or sandbox runtime

Set a default for every session with `permissions.defaultMode` in `~/.claude/settings.json`, or in a project's `.claude/settings.json`. According to the docs, project settings can't make `auto` or `bypassPermissions` the starting mode, which stops a cloned repository from quietly granting itself autonomy.

⚠ DANGER

`bypassPermissions` means **every** action runs with no checks. Only use it inside something you're happy to throw away: a container, a VM, or a cloud sandbox with no access to your real accounts or files (Chapter 2).

13.3 Every settings key, A–Z

Search this table when a guide mentions a setting. Many keys only matter to administrators. Run `/config` or ask Claude (`/update-config`) to change settings safely.

KEY	WHAT IT DOES
<code>advisorModel</code>	Pick which model answers when Claude calls the server-side advisor tool.
<code>alwaysThinkingEnabled</code>	Turn extended thinking off for every session by setting this to <code>false</code> .
<code>availableModels</code>	Restrict which models people can select for the main session, subagents, skills, and the advisor.
<code>availableModelsMatch</code>	Choose how <code>availableModels</code> entries match model IDs.
<code>deniedModels</code>	Block specific models, with or without an <code>availableModels</code> allowlist and even when that list permits them.
<code>effortLevel</code>	Set a default effort level for models you haven't saved a level for.
<code>enforceAvailableModels</code>	The <code>/model</code> picker has a Default option, and <code>default</code> model setting describes the model it resolves to.
<code>fallbackModel</code>	Name backup models for Claude Code to try, in order, when your primary model is overloaded or unavailable.
<code>fastMode</code>	Turn fast mode on for sessions where it's available, for interactive work like rapid iteration or live debugging where you want speed at...
<code>fastModePerSessionOptIn</code>	Normally, running <code>/fast</code> saves <code>fastMode</code> to a person's user settings, so fast mode is on at the start of every later session.
<code>language</code>	Have Claude respond in a language other than English by default.
<code>maxEffortLevel</code>	Cap the effort level a session can use, leaving lower levels available.
<code>model</code>	Set the model every new session uses, so you don't have to pick one with <code>/model</code> each time.
<code>modelOverrides</code>	Map Anthropic model IDs to provider-specific model IDs, such as Amazon Bedrock inference profile ARNs.
<code>modelPicker</code>	List the models the <code>/model</code> picker offers, in the order you write them and under labels you choose, so the picker lists the models your...
<code>modelPricing</code>	Report spend at the rates your organization pays instead of list price.
<code>modelSettings</code>	Save an effort level for each model you use.
<code>outputStyle</code>	Select an output style by name.
<code>promptCacheTtl</code>	Choose how long the prompt cache holds the main conversation.
<code>showThinkingSummaries</code>	See summaries of Claude's extended thinking in interactive sessions.
<code>subagentPromptCacheTtl</code>	Choose how long the prompt cache holds the requests Claude Code makes outside the main conversation.
<code>switchModelsOnFlag</code>	Choose what happens when a safety classifier flags a request: switch to the fallback model and continue, or pause so you can choose...
<code>ultracode</code>	Start sessions with ultracode on.
<code>allowManagedPermissionRulesOnly</code>	Make managed settings the only settings source of permission rules.

<code>autoMode</code>	Add your own rules to what the auto mode classifier blocks and allows.
<code>autoMode.classifyAllShell</code>	Send every Bash and PowerShell command through the auto mode classifier while auto mode is active.
<code>disableAutoMode</code>	Remove auto mode from the <code>Shift+Tab</code> cycle.
<code>permissions</code>	Control which tools Claude can use without asking, which ones always prompt, and which ones are blocked, and set the permission mode a...
<code>useAutoModeDuringPlan</code>	Choose whether Claude Code uses the auto mode classifier to review shell commands in plan mode.
<code>permissions.allow</code>	List the tool uses Claude Code approves without asking you.
<code>permissions.ask</code>	List the tool uses that prompt you for confirmation even in a permission mode that would otherwise approve them, such as <code>acceptEdits</code> ...
<code>permissions.deny</code>	List the tool uses Claude Code blocks.
<code>permissions.additionalDirectories</code>	Give Claude file access to directories outside the one you started in, as additional working directories.
<code>permissions.blockReadsOutsideWorkingDirectories</code>	Make Claude's file tools refuse reads outside your working directories in every permission mode, including <code>bypassPermissions</code> .
<code>permissions.defaultMode</code>	Set the permission mode new sessions start in.
<code>permissions.disableBypassPermissionsMode</code>	Prevent anyone from entering <code>bypassPermissions</code> mode.
<code>skipAutoPermissionPrompt</code>	Skip the one-time notice describing auto mode that Claude Code shows when you first enter auto mode yourself, for example through your...
<code>skipDangerousModePermissionPrompt</code>	Skip the confirmation dialog Claude Code shows before a session enters <code>bypassPermissions</code> mode, whether from...
<code>sandbox</code>	Isolate the Bash commands Claude runs from your filesystem and network with sandboxing.
<code>sandbox.enabled</code>	Turn on sandboxing for Bash commands.
<code>sandbox.failIfUnavailable</code>	Make Claude Code exit with an error at startup when <code>sandbox.enabled</code> is <code>true</code> but the sandbox can't start, because a dependency is...
<code>sandbox.autoAllowBashIfSandboxed</code>	Let Claude Code run sandboxed Bash commands without a permission prompt.
<code>sandbox.excludedCommands</code>	Name commands that Claude Code runs outside the sandbox, such as tools that don't work under it.
<code>sandbox.allowUnsandboxedCommands</code>	Let Claude retry a command outside the sandbox with the <code>dangerouslyDisableSandbox</code> parameter after the sandbox blocks it.
<code>sandbox.filesystem</code>	Control which paths sandboxed commands can read and write.
<code>sandbox.filesystem.allowWrite</code>	Add paths where sandboxed commands can write, beyond the working directory, the per-user temp directory, and the directories you've...

<code>sandbox.filesystem.denyWrite</code>	Block sandboxed commands from writing to specific paths, including paths inside a directory that is otherwise writable.
<code>sandbox.filesystem.denyRead</code>	Block sandboxed commands from reading specific paths, such as credential files that the default read policy would otherwise expose.
<code>sandbox.filesystem.allowRead</code>	Re-open reading for specific paths inside a region that <code>denyRead</code> blocks, to build workspace-only read access.
<code>sandbox.filesystem.allowManagedReadPathsOnly</code>	Honor only the <code>allowRead</code> entries that come from managed settings, so developers can't re-open read access to paths your organization...
<code>sandbox.filesystem.disabled</code>	Skip filesystem isolation while keeping network isolation.
<code>sandbox.ignoreViolations</code>	Silence sandbox violation reports for paths you expect a command to probe and be refused, such as a tool that checks <code>/etc/hosts</code> on...
<code>sandbox.enableWeakerNestedSandbox</code>	Run the Linux sandbox inside an unprivileged Docker container, where bubblewrap can't mount a fresh <code>/proc</code> .
<code>sandbox.enableWeakerNetworkIsolation</code>	Let sandboxed commands on macOS reach the system TLS trust service, <code>com.apple.trustd.agent</code> .
<code>sandbox.allowAppleEvents</code>	Let sandboxed commands on macOS send Apple Events, which <code>open</code> , <code>osascript</code> , and tools that open URLs in a browser need; without it...
<code>sandbox.ripgrep</code>	Point the sandbox at a ripgrep binary of your own instead of the one Claude Code uses, for example when your platform needs a...
<code>sandbox.bwrapPath</code>	Point the sandbox at a bubblewrap binary installed outside <code>PATH</code> , such as a vendored copy on an air-gapped host.
<code>sandbox.socatPath</code>	Point the sandbox network proxy at a <code>socat</code> binary installed outside <code>PATH</code> .
<code>sandbox.credentials</code>	Declare the credential files and environment variables to protect from sandboxed commands.
<code>sandbox.credentials.files</code>	Protect credential files or directories from sandboxed commands.
<code>sandbox.credentials.envVars</code>	Protect environment variables from sandboxed commands.
<code>sandbox.credentials.allowPlaintextInject</code>	Allow <code>mask</code> substitution on plain HTTP requests as well as TLS-terminated HTTPS. On plain HTTP the upstream identity is unverified and...
<code>sandbox.credentials.awsPairs</code>	Group masked environment variables that form one AWS credential for SigV4 re-signing when your credential lives in variables with...
<code>sandbox.credentials.sigv4</code>	Choose what the sandbox proxy does with AWS request forms it can't re-sign: <code>streaming</code> for aws-chunked streaming uploads, <code>presigned</code> ...
<code>sandbox.network</code>	Control which hosts, ports, and sockets sandboxed commands can reach.
<code>sandbox.network.allowUnixSockets</code>	List the Unix socket paths sandboxed commands can connect to on macOS. Claude Code ignores this list on Linux and WSL2, where the...
<code>sandbox.network.allowAllUnixSockets</code>	Let sandboxed commands connect to every Unix socket.
<code>sandbox.network.allowLocalBinding</code>	Let sandboxed commands on macOS listen on network ports, for example to start a dev server, and connect to any port on localhost.
<code>sandbox.network.allowMachLookup</code>	List additional XPC and Mach service names the macOS sandbox may look up.

<code>sandbox.network.allowedDomains</code>	Pre-allow domains for outbound traffic from sandboxed commands, so the sandbox doesn't prompt for them.
<code>sandbox.network.deniedDomains</code>	Block domains for outbound traffic from sandboxed commands, using the same wildcard, port, and IPv6 syntax as <code>allowedDomains</code> .
<code>sandbox.network.strictAllowlist</code>	Deny sandboxed commands access to hosts outside the allowlist instead of prompting for approval.
<code>sandbox.network.allowManagedDomainsOnly</code>	Lock the network allowlist to what managed settings define.
<code>sandbox.network.httpProxyPort</code>	Point the sandbox at your own HTTP proxy instead of the one Claude Code runs.
<code>sandbox.network.socksProxyPort</code>	Point the sandbox at your own SOCKS5 proxy instead of the one Claude Code runs.
<code>sandbox.network.tlsTerminate</code>	Make the sandbox proxy terminate TLS so it can read the contents of HTTPS requests.
<code>autoCompactEnabled</code>	Have Claude Code compact the conversation automatically when context approaches the limit.
<code>autoCompactWindow</code>	Set how full the context window gets before Claude Code compacts automatically.
<code>autoMemoryDirectory</code>	Store auto memory in a directory of your choice instead of the per-project default.
<code>autoMemoryEnabled</code>	Turn auto memory on or off.
<code>bashOutputMaxChars</code>	Set how many characters of a successful Bash or PowerShell command's output Claude receives inline.
<code>claudeMd</code>	Inject CLAUDE.md-style instructions as organization-managed memory without deploying a separate file.
<code>claudeMdExcludes</code>	Skip specific <code>CLAUDE.md</code> files when Claude Code loads memory.
<code>env</code>	Set environment variables for every session and for the subprocesses Claude Code starts from it.
<code>fileCheckpointingEnabled</code>	Have Claude Code snapshot files before each edit so <code>/rewind</code> can restore them.
<code>idleCompaction</code>	Idle compaction compacts a long conversation that you've stepped away from, shortly before its prompt cache expires.
<code>plansDirectory</code>	Choose where Claude Code stores the plan files it writes in plan mode.
<code>skillListingBudgetFraction</code>	Each turn, Claude sees a listing of your skills with their descriptions, and Claude Code caps that listing at a share of the context window.
<code>skillListingMaxDescChars</code>	Each turn, Claude sees a listing of your skills that shows each skill's <code>description</code> and <code>when_to_use</code> text.
<code>taskOutputMaxChars</code>	Through v2.1.276, you set this key to the number of characters of a background task's output that Claude received inline when it read...
<code>askUserQuestionTimeout</code>	Let an unanswered <code>AskUserQuestion</code> dialog auto-continue after a period of idle time, submitting whatever options you had already selected.
<code>autoContinueAtUsageLimit</code>	After a claude.ai usage limit stops your session, wait in the open session and continue the task automatically after the reset.
<code>autoScrollEnabled</code>	Follow new output to the bottom of the conversation in fullscreen rendering.

<code>axScreenReader</code>	Render screen-reader friendly output: flat text without decorative borders or animations.
<code>bashEditDiffEnabled</code>	Choose whether Claude Code records which files changed in a Git repository while a Bash command runs.
<code>companyAnnouncements</code>	Show your organization's announcements to users at startup.
<code>defaultShell</code>	Choose whether Bash or PowerShell runs the shell commands you type with the <code>!</code> prefix in the input box, the ones Claude Code runs...
<code>dialogExpiry</code>	Set the deadline for dialogs Claude Code forwards to a remote client, such as a Remote Control or SDK host, and for the approval dialog...
<code>editorMode</code>	Choose the key binding mode for the input prompt.
<code>emojiCompletionEnabled</code>	Show emoji suggestions when you type <code>:</code> plus a shortcode in the prompt input, and replace a completed shortcode such as <code>:heart:</code> with...
<code>fileSuggestion</code>	Run your own command to supply <code>@</code> file path autocomplete instead of the built-in file suggestion.
<code>footerLinksRegexes</code>	Render extra clickable badges in the footer below the input box when a regex matches turn output: tool results, including file contents...
<code>keybindingFlavor</code>	In v2.1.238 through v2.1.260, setting it to <code>"readline"</code> made <code>Ctrl+W</code> delete back to the previous whitespace instead of only the...
<code>maxProseWidth</code>	Cap the width of the prose in Claude's responses so lines stay readable in a wide terminal.
<code>prefersReducedMotion</code>	Reduce or turn off interface animations such as the spinner, shimmer, and flash effects.
<code>promptSuggestionEnabled</code>	Show or hide prompt suggestions, the grayed-out predictions that appear in your prompt input.
<code>respectGitignore</code>	Control whether the <code>@</code> file picker leaves out files that match <code>.gitignore</code> patterns.
<code>respondToBashCommands</code>	Choose whether Claude responds after you run a shell command with the <code>!</code> prefix in the input box.
<code>showClearContextOnPlanAccept</code>	When Claude finishes a plan in plan mode, it shows an approval menu.
<code>showTurnDuration</code>	Show or hide the turn duration message after each response, such as "Cooked for 1m 6s · done 6:05 PM". The clock after "done" shows when...
<code>spellcheck</code>	Underline misspelled words in the prompt input as you type, using a spell checker you install.
<code>spinnerTipsEnabled</code>	While Claude works, the spinner line rotates through short tips about Claude Code features, such as "Use Plan Mode to prepare for a..."
<code>spinnerTipsOverride</code>	Add your own tips to the spinner tips that Claude Code shows while Claude works, or replace the built-in tips with yours.
<code>spinnerVerbs</code>	While a turn is in progress, the spinner shows a rotating verb such as "Accomplishing", "Architecting", or "Baking". Use this key to add...
<code>statusLine</code>	Run your own command to render a status line below the prompt with context such as the model, cost, or git branch.
<code>subagentStatusLine</code>	When Claude runs subagents, Claude Code lists them in a task display below the prompt, one row per subagent showing `name · description`...
<code>syntaxHighlightingDisabled</code>	Claude Code colors code by language in the diffs, code blocks, and file previews it shows in the terminal, with its built-in...
<code>terminalProgressBarEnabled</code>	Some terminals can show a progress indicator on the tab or in the taskbar for the program running in them.

<code>terminalTitleFromRename</code>	Claude Code sets your terminal tab's title.
<code>theme</code>	Pick the color theme for the interface.
<code>timeFormat</code>	Choose how Claude Code writes the times it shows in the interface, such as the <code>done 6:05 PM</code> at the end of each turn duration message...
<code>timeZone</code>	Show the times in the interface in a time zone other than your system's.
<code>tui</code>	Choose the terminal UI renderer.
<code>verbose</code>	By default, the transcript collapses each tool call to a short summary, such as the command Claude ran and a line count of its output,...
<code>viewMode</code>	Set the transcript view Claude Code starts in: <code>"default"</code> , <code>"verbose"</code> , or <code>"focus"</code> .
<code>vimInsertModeRemaps</code>	Map two-key INSERT-mode sequences to Escape in vim editor mode.
<code>voice</code>	Turn on voice dictation and choose how the dictation key behaves.
<code>voiceEnabled</code>	Turn voice dictation on with the single Boolean form that predates the <code>voice</code> object.
<code>wheelScrollAccelerationEnabled</code>	Accelerate mouse-wheel scroll speed during fast scrolls in fullscreen rendering.
<code>attribution</code>	Customize the attribution Claude Code adds to git commits and pull requests.
<code>includeCoAuthoredBy</code>	Use <code>attribution</code> instead, which replaces this key and lets you change or hide the commit trailer, the pull request text, and the...
<code>includeGitInstructions</code>	Claude Code gives Claude two git-related pieces of context: its built-in instructions for how to write commits and pull requests, in the...
<code>prUrlTemplate</code>	Point the PR links Claude Code renders, in the footer badge and in tool-result summaries, at an internal code-review tool instead of...
<code>attribution.commit</code>	Set the attribution text Claude Code adds to git commits, including any trailers.
<code>attribution.pr</code>	Set the attribution text Claude Code adds to pull request descriptions.
<code>attribution.sessionUrl</code>	Choose whether Claude Code appends the <code>claude.ai</code> session link when it commits or opens a pull request from a cloud or Remote Control...
<code>allowedHttpHookUrls</code>	Limit which URLs HTTP hooks can target.
<code>allowManagedHooksOnly</code>	Restrict hook execution to hooks your organization deploys.
<code>disableAllHooks</code>	Turn off hooks, any custom status line, and any custom file suggestion command.
<code>disableWorkflows</code>	Turn off dynamic workflows and the bundled workflow commands for everyone your settings reach, such as an organization through managed...
<code>enableWorkflows</code>	Turn dynamic workflows on or off for yourself when your plan's default isn't what you want.
<code>hooks</code>	Run your own commands, prompts, agents, HTTP requests, or MCP tools as hooks at points in Claude Code's lifecycle, such as before a tool...
<code>httpHookAllowedEnvVars</code>	An HTTP hook can put the value of an environment variable into a request header, for example an <code>Authorization: Bearer \$HOOK_TOKEN</code> ...
<code>workflowKeywordTriggerEnabled</code>	Choose whether typing the keyword <code>ultracode</code> in a prompt triggers a dynamic workflow.
<code>workflowSizeGuideline</code>	Set the agent count Claude aims for in the dynamic workflows it writes.
<code>disableBundledSkills</code>	Turn off the skills and workflows included with Claude Code.
<code>disableSkillShellExecution</code>	Turn off inline shell execution for <code>!`...`</code> and <code>```!</code> blocks in skills and custom commands from user, project, plugin, or...

<code>skillOverrides</code>	Hide or collapse a skill without editing its <code>SKILL.md</code> .
<code>syncClaudeAiSkills</code>	Turn off the download of the skills enabled for your claude.ai account.
<code>syncClaudeAiPlugins</code>	Turn off the download of the plugins enabled for your claude.ai account.
<code>allowedChannelPlugins</code>	Choose which channel plugins can push messages into sessions in your organization.
<code>blockedMarketplaces</code>	Block plugin marketplace sources for your organization.
<code>channelsEnabled</code>	Allow channels for your organization.
<code>disableCommandPluginSources</code>	Block the <code>command</code> plugin source, which installs a plugin by running a marketplace-declared command on the user's machine.
<code>pluginSuggestionMarketplaces</code>	Name the marketplaces whose plugins can appear as contextual install suggestions, in spinner tips and pinned at the top of the <code>/plugin</code> ...
<code>pluginTrustMessage</code>	Add your organization's own text to the plugin trust warning Claude Code shows before installation, for example to confirm that plugins...
<code>strictKnownMarketplaces</code>	Restrict which plugin marketplace sources people in your organization can add and install plugins from.
<code>strictPluginOnlyCustomization</code>	Block skills, agents, hooks, and MCP servers from user and project sources, so they can come only from plugins or managed settings.
<code>strictPluginOnlyCustomization.skills</code>	Lock the <code>skills</code> surface.
<code>strictPluginOnlyCustomization.agents</code>	Lock the <code>agents</code> surface.
<code>strictPluginOnlyCustomization.hooks</code>	Lock the <code>hooks</code> surface.
<code>strictPluginOnlyCustomization.mcp</code>	Lock the <code>mcp</code> surface.
<code>enabledPlugins</code>	Turn individual plugins on or off, keyed by <code>plugin-name@marketplace-name</code> .
<code>extraKnownMarketplaces</code>	Register additional plugin marketplaces by name, so that people who open the repository, or everyone your managed settings reach, get...
<code>pluginConfigs</code>	Store the non-sensitive answers you give a plugin's <code>userConfig</code> configuration dialog, keyed by plugin ID. Claude Code writes this key...
<code>prependPlugins</code>	List the managed plugins whose mods run before every mod a user installs, in the listed order.
<code>appendPlugins</code>	List the managed plugins whose mods run after every mod a user installs, in the listed order.
<code>allowAllClaudeAiMcps</code>	Load the claude.ai connectors Claude Code fetches itself alongside a deployed <code>managed-mcp.json</code> .
<code>allowClaudeInChromeWithManagedMcp</code>	Let the built-in Claude in Chrome server run alongside a deployed <code>managed-mcp.json</code> .
<code>allowedMcpServers</code>	Allowlist the MCP servers people can add.
<code>allowManagedMcpServersOnly</code>	Make the managed allowlist the only one that applies.
<code>deniedMcpServers</code>	Block specific MCP servers.
<code>disableClaudeAiConnectors</code>	Turn off the claude.ai MCP connectors Claude Code fetches itself, so it neither fetches nor connects them.
<code>disabledMcpjsonServers</code>	Reject specific servers defined in a project's <code>.mcp.json</code> file so Claude Code never connects them or asks you to approve them.

<code>enableAllProjectMcpServers</code>	Approve every MCP server defined in project <code>.mcp.json</code> files without a prompt.
<code>enabledMcpjsonServers</code>	Approve specific servers defined in project <code>.mcp.json</code> files so Claude Code connects them without asking.
<code>managedMcpServers</code>	Provide remote MCP servers to every user from managed settings.
<code>agent</code>	Run the main thread as a named subagent, so Claude Code applies that subagent's system prompt, tool restrictions, and model to your session.
<code>crossSessionInbound</code>	Choose what this session does with messages arriving from your other Claude Code sessions.
<code>disableAgentView</code>	Turn off background agents and agent view: <code>claude agents</code> , <code>--bg</code> , <code>/background</code> , and the on-demand supervisor.
<code>isolatePeerMachines</code>	Require your explicit approval before Claude's <code>SendMessage</code> reaches one of your sessions beyond this machine; see Require approval for...
<code>processWrapper</code>	On macOS and Linux, place a corporate launcher command in front of the background processes Claude Code starts.
<code>teammateMode</code>	Choose where Claude Code shows agent team teammates: inside your main terminal pane, or in split panes when your terminal supports them.
<code>worktree</code>	Configure how Claude Code creates and manages git worktrees for <code>--worktree</code> , the <code>EnterWorktree</code> tool, and isolated subagents and...
<code>worktree.baseRef</code>	Choose which ref new worktrees branch from. "fresh" branches from <code>origin/</code> for a clean tree matching the remote; "head" branches...
<code>worktree.symlinkDirectories</code>	Symlink directories from the main repository into each worktree so you don't duplicate large directories on disk.
<code>worktree.sparsePaths</code>	Check out only the listed directories in each worktree through git sparse-checkout.
<code>worktree.bgIsolation</code>	Choose how background sessions isolate their file edits.
<code>worktree.location</code>	Choose the folder on a remote machine where Desktop SSH sessions create their worktrees, instead of <code>/.claude/worktrees/</code> .
<code>agentPushNotifEnabled</code>	Allow Claude to send a push notification to your phone when it decides one is worth sending, for example when a long task finishes.
<code>awaySummaryEnabled</code>	Show a one-line session recap when you return to the terminal after a few minutes away.
<code>disableArtifact</code>	Use <code>enableArtifact</code> instead to turn off the Artifact tool, which publishes session output as a private web page on claude.ai.
<code>disableDeepLinkRegistration</code>	Stop Claude Code from registering the <code>claude-cli://</code> protocol handler with the operating system, which it otherwise does after you send...
<code>disableDesktopLocalSessions</code>	Turn off Code sessions that run on the device in the desktop app, for deployments where developers should work on remote machines over...
<code>disableRemoteControl</code>	Turn off Remote Control: Claude Code then refuses <code>claude remote-control</code> , the <code>--remote-control</code> flag, auto-start, and the in-session...
<code>enableArtifact</code>	Turn off the Artifact tool, which publishes session output as a private web page on claude.ai.
<code>inputNeededNotifEnabled</code>	Get a push notification on your phone when a permission prompt or question is waiting for your input.
<code>preferredNotifChannel</code>	Choose how Claude Code notifies you when a task completes or a permission prompt is waiting.
<code>remote.defaultEnvironmentId</code>	Pick the default cloud environment for cloud sessions you create from the CLI, such as with <code>claude --cloud</code> .

<code>remoteControlAtStartup</code>	Connect Remote Control automatically when each interactive session starts, instead of waiting for <code>/remote-control</code> .
<code>sshConfigs</code>	Add SSH connections to the Desktop environment dropdown.
<code>sshHostAllowlist</code>	Limit the hosts a Desktop SSH session can connect to.
<code>allowedProviders</code>	List the services a machine may reach Claude through, such as the Anthropic API, Amazon Bedrock, or an LLM gateway.
<code>apiKeyHelper</code>	Run your own command to produce the credential Claude Code sends with model requests.
<code>awsAuthRefresh</code>	Run your own command, such as <code>aws sso login</code> , to refresh the credentials in your <code>.aws</code> directory when the ones Claude Code has for...
<code>awsCredentialExport</code>	Run your own command that prints AWS credentials as JSON, so Claude Code can call Amazon Bedrock with credentials that don't live in...
<code>forceLoginMethod</code>	Restrict which kind of account people can log in with.
<code>forceLoginGatewayUrl</code>	Set the gateway URL the <code>/login</code> Cloud gateway screen connects to, so people reach your cloud gateway without typing its address.
<code>forceLoginOrgUUID</code>	From a managed source, require claude.ai account logins to belong to one Anthropic organization, given as a single UUID, or to any of...
<code>gatewayInternalNetworks</code>	Declare the public IPv4 blocks that your organization numbers its internal network from, so <code>/login</code> accepts a cloud gateway there.
<code>gcpAuthRefresh</code>	Run your own command to refresh Google Cloud Application Default Credentials when Claude Code finds they've expired or can't be loaded,...
<code>otelHeadersHelper</code>	Run your own command to generate the headers Claude Code sends with OpenTelemetry exports, for backends whose tokens rotate.
<code>autoUpdatesChannel</code>	Choose which release channel background auto-updates and <code>claude update</code> follow.
<code>minimumVersion</code>	Keep background auto-updates and <code>claude update</code> from installing any version below this one, so moving to the "stable" channel doesn't...
<code>requiredMaximumVersion</code>	Set the newest Claude Code version your organization allows to start.
<code>requiredMinimumVersion</code>	Set the oldest Claude Code version your organization allows to start.
<code>browserExternalPageTools</code>	Stop Claude from using its tools to read or act on external pages in the desktop app's Browser pane.
<code>disableBrowserExternalNavigation</code>	Turn off external browsing in the desktop app's Browser pane for people and Claude alike.
<code>disableMobileSimulatorTools</code>	Block Claude's tools for the desktop app's iOS Simulator pane.
<code>cleanupPeriodDays</code>	Set how many days Claude Code keeps session transcripts and other application data before deleting them.
<code>desktopSessionCleanupPeriodDays</code>	Set an age limit in days for the transcripts of sessions you started or most recently continued in Claude Desktop or Cowork.
<code>feedbackDrafts</code>	Control Claude-drafted feedback: whether Claude can queue feedback drafts for you to review, and whether Claude Code shows a card when...
<code>feedbackSurveyRate</code>	Set the probability that the session quality survey appears when a session is eligible for it.
<code>skipWebFetchPreflight</code>	Skip the WebFetch domain safety check, which sends each requested hostname to <code>api.anthropic.com</code> before fetching.

<code>disableSideLoadFlags</code>	Reject the <code>--plugin-dir</code> , <code>--plugin-url</code> , <code>--agents</code> , and <code>--mcp-config</code> CLI flags at startup, which users could otherwise pass to...
<code>forceRemoteSettingsRefresh</code>	Block CLI startup until Claude Code has freshly fetched server-managed settings.
<code>managedSourcesBehavior</code>	Choose whether Claude Code applies only the highest-priority managed source your organization delivers, or combines every admin source...
<code>parentSettingsBehavior</code>	Choose whether Claude Code applies managed settings supplied by an embedding host process, such as the Agent SDK or an IDE extension,...
<code>policyHelper</code>	Run an executable you deploy that computes managed settings at startup, so you can derive policy from device posture, identity, or a...
<code>policyHelper.path</code>	Name the helper executable Claude Code runs.
<code>policyHelper.timeoutMs</code>	Set how long Claude Code waits for the helper before treating the run as failed.
<code>policyHelper.refreshIntervalMs</code>	Have Claude Code re-run the helper in the background on an interval so policy changes reach a running session.
<code>wslInheritsWindowsSettings</code>	Have Claude Code on WSL read managed settings from the Windows policy chain, with HKLM and the Windows managed settings file taking...
<code>autoConnectIde</code>	Connect to a running IDE automatically when you start Claude Code from an external terminal.
<code>autoInstallIdeExtension</code>	Install the Claude Code IDE extension automatically when you run Claude Code from a VS Code terminal.
<code>claudeInChromeDefaultEnabled</code>	Start every interactive CLI session with Chrome integration on, without passing <code>--chrome</code> each time.
<code>copyFullResponse</code>	Make <code>/copy</code> copy the full response every time, without showing the picker.
<code>copyOnSelect</code>	Copy text to your clipboard automatically when you finish selecting it with the mouse in fullscreen rendering or agent view.
<code>defaultToAgentsView</code>	Open agent view instead of a new conversation when you run <code>claude</code> with no arguments.
<code>diffTool</code>	Choose where Claude Code shows the diff of an <code>Edit</code> or <code>Write</code> change it proposes when a VS Code or JetBrains IDE is connected:...
<code>externalEditorContext</code>	When you press <code>Ctrl+G</code> , Claude Code opens the prompt you're typing in your external editor.
<code>leftArrowOpensAgents</code>	Press <code>←</code> on an empty prompt to background the session and open agent view.
<code>permissionExplainerEnabled</code>	Through v2.1.256, you could press <code>Ctrl+E</code> on a Bash or PowerShell permission prompt to see a model-generated explanation of the...
<code>prStatusFooterEnabled</code>	Show a badge in the prompt footer for the current branch's open pull request or merge request, with a colored underline that shows its...
<code>teammateDefaultModel</code>	Through v2.1.233, you set this key to the model for agent team teammates your prompt didn't name a model for: an alias such as...

Autonomy Recipes

Copy-ready setups for the most common hands-off scenarios. Each recipe lists the risk level and the guardrails you must keep.

RECIPE	SETUP	RISK	MUST-KEEP GUARDRAILS
Overnight refactor in a container	Dev container or VM, <code>--dangerously-skip-permissions</code> , a git branch	Medium	No production credentials in the box. Review the diff in the morning
Daily docs or report job	<code>claude -p</code> from cron, <code>dontAsk</code> , an allowlist of exact commands	Low	Read-only tokens. Output to a file, not straight to publishing
PR auto-fix bot	CI runner, allowlist of test and lint commands, a scoped GitHub token	Medium	Branch protection on <code>main</code> . A human merges
n8n AI agent with tools	Agent node with minimal tools, approval step for sending or deleting	Medium	Error workflow, rate limits, logging
Notion assistant	Integration shared with one teamspace only	Low	No delete permission. Weekly audit
Browser agent	Separate browser profile, no saved passwords	High	Watch it live. Never let it handle payments unattended
Trading bot	Paper account only (Vol. 10)	High	Kill switch, loss limits, separate keys

14.1 Golden rules of autonomy

- ☐ Autonomy goes in a box: a container, VM, or separate account
- ☐ Credentials are scoped to the task and set to expire
- ☐ Anything that can't be undone gets a human approval step
- ☐ Everything is logged, and logs are reviewed
- ☐ There's a tested kill switch for every agent
- ☐ Start with a short leash and lengthen it only after a clean track record

Incident Response: When an Agent Goes Wrong

Sooner or later an agent will do something you didn't expect. A calm, practised routine turns a disaster into an inconvenience.

STOP, CONTAIN, RECOVER

- 1 Stop it.**
Press `Esc` or `Ctrl + C`, stop the container or VM, deactivate the n8n workflow, or use `/stop` for a background session.
- 2 Cut its access.**
Revoke or rotate any tokens it held: GitHub, cloud, Notion, MCP connectors, API keys.
- 3 Assess.**
Check `git status` and `git log`, cloud consoles, sent mail, and billing pages. What changed?
- 4 Restore.**
`git revert` or restore from a branch or backup. Use `/rewind` for local file edits. Restore databases from snapshots.
- 5 Notify.**
If other people's data or money was affected, tell them, and the relevant service.
- 6 Learn.**
Write down what permission made it possible, then tighten that rule or add a hook or deny rule.

Emergency card: Esc / Ctrl+C → kill container → revoke tokens → check git, billing, and sent mail → revert → tighten rules.

Codex config.toml Reference

This chapter lists every key in Codex's `~/.codex/config.toml` and the admin-only `requirements.toml`. It was generated on October 16, 2026 from the official reference at developers.openai.com/codex/config-reference, and descriptions are shortened to their first sentence. Check the official page for full details before you change a key you don't recognize.

- **User config:** `~/.codex/config.toml`. **Project config:** `.codex/config.toml`, which loads only when you trust the project.
- Project config can't override provider, auth, notification, profile or telemetry keys. Codex ignores them there.
- `approval_policy = "untrusted"` is **retired**. Use `on-request`, `never`, or a `granular` table.
- **Profiles** live next to config.toml as `$CODEX_HOME/<name>.config.toml`. Select one with `--profile <name>`.

16.1 Command-line autonomy flags (current release)

These come from the Codex source (`codex-rs/utils/cli/src/shared_options.rs`).

FLAG	ALIAS	WHAT IT DOES
<code>--sandbox <mode>, -s</code>	—	Sets the sandbox: <code>read-only</code> , <code>workspace-write</code> or <code>danger-full-access</code>
<code>--approve-for-me</code>	<code>--not-so-yolo</code>	Same as <code>approvals_reviewer="auto_review"</code> , <code>approval_policy="on-request"</code> and <code>sandbox_mode="workspace-write"</code>
<code>--dangerously-bypass-approvals-and-sandbox</code>	<code>--yolo</code>	No prompts and no sandbox. Use it only inside a sandbox you built yourself
<code>--dangerously-bypass-hook-trust</code>	—	Runs enabled hooks without persisted hook trust for this one run
<code>--profile <name>, -p</code>	—	Layers <code>\$CODEX_HOME/<name>.config.toml</code> on top of your config
<code>--cd <dir>, -C</code>	—	Sets the working root
<code>--add-dir <dir></code>	—	Adds another writable directory
<code>--worktree</code>	—	Runs the session in a new managed Git worktree

⚠ WARNING

`--full-auto` appears in older tutorials but no longer exists. Use `--sandbox workspace-write` or `--approve-for-me` instead.

16.2 config.toml keys (292)

KEY	TYPE	WHAT IT DOES
<code>model</code>	string	Model to use (e.g., <code>gpt-6.1-sol</code>).
<code>review_model</code>	string	Optional model override used by <code>/review</code> (defaults to the current session model).
<code>model_provider</code>	string	Provider id from <code>model_providers</code> (default: <code>openai</code>).
<code>openai_base_url</code>	string	Base URL override for the built-in <code>openai</code> model provider.
<code>model_context_window</code>	number	Context window tokens available to the active model.
<code>model_auto_compact_token_limit</code>	number	Token threshold that triggers automatic history compaction (unset uses model defaults).
<code>model_auto_compact_token_limit_scope</code>	total / body_after_prefix	Controls whether the auto-compaction threshold counts the full active context (<code>total</code> , the default) or only growth after the...
<code>model_catalog_json</code>	string (path)	Optional path to a JSON model catalog loaded on startup.
<code>oss_provider</code>	lmstudio / ollama	Default local provider used when running with <code>--oss</code> (defaults to prompting if unset).

<code>approval_policy</code>	<code>on-request / never / { granular = { s...</code>	Controls when Codex pauses for approval before executing commands.
<code>approval_policy.granular.sandbox_approval</code>	<code>boolean</code>	When <code>true</code> , sandbox escalation approval prompts are allowed to surface.
<code>approval_policy.granular.rules</code>	<code>boolean</code>	When <code>true</code> , approvals triggered by <code>execpolicy prompt</code> rules are allowed to surface.
<code>approval_policy.granular.mcp_elicitations</code>	<code>boolean</code>	When <code>true</code> , MCP elicitation prompts are allowed to surface instead of being auto-rejected.
<code>approval_policy.granular.request_permissions</code>	<code>boolean</code>	When <code>true</code> , prompts from the <code>request_permissions</code> tool are allowed to surface.
<code>approval_policy.granular.skill_approval</code>	<code>boolean</code>	When <code>true</code> , skill-script approval prompts are allowed to surface.
<code>approvals_reviewer</code>	<code>user / auto_review</code>	Who reviews eligible approval prompts under <code>on-request</code> or granular approval policies.
<code>auto_review.policy</code>	<code>string</code>	Local Markdown policy instructions for automatic review.
<code>auto_review.extra_policy</code>	<code>string</code>	Additional local Markdown policy for automatic review, included alongside the main policy.
<code>allow_login_shell</code>	<code>boolean</code>	Allow shell-based tools to use login-shell semantics.
<code>sandbox_mode</code>	<code>read-only / workspace-write / danger-...</code>	Sandbox policy for filesystem and network access during command execution.
<code>sandbox_workspace_write.network_access</code>	<code>boolean</code>	Allow outbound network access inside the workspace-write sandbox.
<code>sandbox_workspace_write.exclude_tmpdir_env_var</code>	<code>boolean</code>	Exclude <code>\$TMPDIR</code> from writable roots in workspace-write mode.
<code>sandbox_workspace_write.exclude_slash_tmp</code>	<code>boolean</code>	Exclude <code>/tmp</code> from writable roots in workspace-write mode.
<code>windows.sandbox</code>	<code>mxc / elevated / unelevated</code>	Native Windows sandbox implementation.
<code>browser_use.allow_history_access</code>	<code>boolean</code>	Set to <code>false</code> to restrict browser-history access.
<code>browser_use.default_origin_policy</code>	<code>table</code>	Fallback browser-origin restrictions.
<code>browser_use.origins.<origin></code>	<code>table</code>	Per-origin browser restrictions with the same fields as <code>browser_use.default_origin_policy</code> .
<code>computer_use.default_app_access</code>	<code>allow / deny</code>	Fallback native-app access policy for Computer Use.
<code>computer_use.macos.bundle_ids</code>	<code>map<string, allow / deny></code>	Native macOS app access keyed by bundle identifier.

<code>computer_use.windows.aumids</code>	<code>map<string, allow / deny></code>	Packaged Windows app access keyed by Application User Model ID (AUMID).
<code>computer_use.windows.exes</code>	<code>array<table></code>	Windows executable access rules.
<code>computer_use.windows.always_allowed_app_ids</code>	<code>array<string></code>	Windows app identifiers that Computer Use can open without prompting.
<code>notify</code>	<code>array<string></code>	Command invoked for notifications; receives a JSON payload from Codex.
<code>check_for_update_on_startup</code>	<code>boolean</code>	Check for Codex updates on startup (set to false only when updates are centrally managed).
<code>feedback.enabled</code>	<code>boolean</code>	Enable feedback submission via <code>/feedback</code> across local clients (default: true).
<code>analytics.enabled</code>	<code>boolean</code>	Enable or disable analytics for this machine/profile.
<code>instructions</code>	<code>string</code>	Reserved for future use; prefer <code>model_instructions_file</code> or <code>AGENTS.md</code> .
<code>developer_instructions</code>	<code>string</code>	Additional developer instructions injected into the session (optional).
<code>log_dir</code>	<code>string (path)</code>	Directory where Codex writes log files; defaults to <code>\$CODEX_HOME/log</code> .
<code>sqlite_home</code>	<code>string (path)</code>	Directory where Codex stores the SQLite-backed state DB used by agent jobs and other resumable runtime state.
<code>compact_prompt</code>	<code>string</code>	Inline override for the history compaction prompt.
<code>model_instructions_file</code>	<code>string (path)</code>	Replacement for built-in instructions instead of <code>AGENTS.md</code> .
<code>personality</code>	<code>none / friendly / pragmatic</code>	Deprecated compatibility setting. <code>friendly</code> and <code>pragmatic</code> no longer select a style. <code>none</code> omits the built-in Personality...
<code>service_tier</code>	<code>string</code>	Preferred service tier for new turns.
<code>experimental_compact_prompt_file</code>	<code>string (path)</code>	Load the compaction prompt override from a file (experimental).
<code>skills.max_context_tokens</code>	<code>integer (positive)</code>	Token budget for the available-skills catalog.
<code>skills.config</code>	<code>array<object></code>	Per-skill enablement overrides stored in config.toml.
<code>skills.config.<index>.path</code>	<code>string (path)</code>	Path to a skill folder containing <code>SKILL.md</code> .
<code>skills.config.<index>.enabled</code>	<code>boolean</code>	Enable or disable the referenced skill.
<code>apps.<id>.enabled</code>	<code>boolean</code>	Enable or disable a specific app/connector by id (default: true).
<code>apps._default.enabled</code>	<code>boolean</code>	Default app enabled state for all apps unless overridden per app.
<code>apps._default.destructive_enabled</code>	<code>boolean</code>	Default allow/deny for app tools with <code>destructive_hint = true</code> .

<code>apps._default.open_world_enabled</code>	<code>boolean</code>	Default allow/deny for app tools with <code>open_world_hint = true</code> .
<code>apps._default.approvals_reviewer</code>	<code>user / auto_review</code>	Default reviewer for app tool approval prompts unless overridden per app.
<code>apps._default.default_tools_approval_mode</code>	<code>auto / prompt / writes / approve</code>	Default approval behavior for app tools without per-app or per-tool overrides.
<code>apps.<id>.destructive_enabled</code>	<code>boolean</code>	Allow or block tools in this app that advertise <code>destructive_hint = true</code> .
<code>apps.<id>.open_world_enabled</code>	<code>boolean</code>	Allow or block tools in this app that advertise <code>open_world_hint = true</code> .
<code>apps.<id>.default_tools_enabled</code>	<code>boolean</code>	Default enabled state for tools in this app unless a per-tool override exists.
<code>apps.<id>.approvals_reviewer</code>	<code>user / auto_review</code>	Reviewer for this app's tool approval prompts.
<code>apps.<id>.default_tools_approval_mode</code>	<code>auto / prompt / writes / approve</code>	Default approval behavior for tools in this app unless a per-tool override exists.
<code>apps.<id>.tools.<tool>.enabled</code>	<code>boolean</code>	Per-tool enabled override for an app tool (for example <code>repos/list</code>).
<code>apps.<id>.tools.<tool>.approval_mode</code>	<code>auto / prompt / writes / approve</code>	Per-tool approval behavior override for a single app tool.
<code>features.apps</code>	<code>boolean</code>	Enable app (connector) integrations (stable; on by default).
<code>features.hooks</code>	<code>boolean</code>	Enable lifecycle hooks loaded from <code>hooks.json</code> or inline <code>[hooks]</code> config. <code>features.codex_hooks</code> is a deprecated alias.
<code>features.code_mode.enabled</code>	<code>boolean</code>	Enable code mode feature configuration.
<code>features.code_mode.excluded_tool_namespaces</code>	<code>array<string></code>	Tool namespaces code mode excludes from nested code-mode tool guidance and executor exposure.
<code>features.code_mode.direct_only_tool_namespaces</code>	<code>array<string></code>	Tool namespaces code mode can use only through direct tool calls.
<code>features.context_management.experimental_mode</code>	<code>boolean</code>	Experimental context-management setting.
<code>features.rollout_budget.enabled</code>	<code>boolean</code>	Enable rollout budget tracking.
<code>features.rollout_budget.limit_tokens</code>	<code>integer</code>	Positive token limit for rollout budget tracking.
<code>features.rollout_budget.reminder_interval_tokens</code>	<code>integer</code>	Positive token interval between rollout budget reminders.
<code>features.rollout_budget.sampling_token_weight</code>	<code>number</code>	Finite non-negative multiplier for sampled tokens in rollout budget accounting.
<code>features.rollout_budget.prefill_token_weight</code>	<code>number</code>	Finite non-negative multiplier for prefill tokens in rollout budget accounting.
<code>hooks</code>	<code>table</code>	Lifecycle hooks configured inline in <code>config.toml</code> .

<code>hooks.<Event></code>	<code>array<table></code>	Matcher groups for hook events such as <code>PreToolUse</code> , <code>PermissionRequest</code> , <code>PostToolUse</code> , <code>PreCompact</code> , <code>PostCompact</code> , ...
<code>hooks.<Event>[].hooks</code>	<code>array<table></code>	Hook handlers for a matcher group.
<code>hooks.<Event>[].hooks[].async</code>	<code>boolean</code>	Run a command hook in the background without delaying the triggering operation.
<code>hooks.<Event>[].hooks[].additionalContextLimit</code>	<code>integer</code>	Approximate per-handler token threshold for saving oversized <code>additionalContext</code> to disk and showing the model a shorter preview.
<code>hooks.<Event>[].hooks[].commandWindows</code>	<code>string</code>	Windows-only command override for command hooks.
<code>features.memories</code>	<code>boolean</code>	Enable Memories (off by default).
<code>mcp_optional_startup_grace_ms</code>	<code>integer</code> (milliseconds)	Shared wait for optional MCP servers when building the initial tool catalog.
<code>mcp_servers.<id>.command</code>	<code>string</code>	Launcher command for an MCP stdio server.
<code>mcp_servers.<id>.args</code>	<code>array<string></code>	Arguments passed to the MCP stdio server command.
<code>mcp_servers.<id>.env</code>	<code>map<string,string></code>	Environment variables forwarded to the MCP stdio server.
<code>mcp_servers.<id>.cwd</code>	<code>string</code>	Working directory for the MCP stdio server process.
<code>mcp_servers.<id>.url</code>	<code>string</code>	Endpoint for an MCP streamable HTTP server.
<code>mcp_servers.<id>.auth</code>	<code>oauth / chatgpt</code>	Authentication fallback for an MCP HTTP server after configured bearer tokens and authorization headers. <code>oauth</code> (default)...
<code>mcp_servers.<id>.oauth.client_id</code>	<code>string</code>	Pre-registered OAuth client ID used for authorization and token exchange with this MCP server.
<code>mcp_servers.<id>.oauth.client_secret</code>	<code>string</code>	OAuth client secret for token exchange with a pre-registered client.
<code>mcp_servers.<id>.oauth.callback_url</code>	<code>string</code>	Server-specific OAuth callback.
<code>mcp_servers.<id>.oauth.callback_port</code>	<code>integer</code>	Fixed OAuth callback listener port for this MCP server.
<code>mcp_servers.<id>.bearer_token_env_var</code>	<code>string</code>	Environment variable sourcing the bearer token for an MCP HTTP server.
<code>mcp_servers.<id>.http_headers</code>	<code>map<string,string></code>	Static HTTP headers included with each MCP HTTP request.
<code>mcp_servers.<id>.http_headers_helper</code>	<code>string (command)</code>	Local command that prints a JSON object of HTTP header names and values.
<code>mcp_servers.<id>.env_http_headers</code>	<code>map<string,string></code>	HTTP headers populated from environment variables for an MCP HTTP server.
<code>mcp_servers.<id>.enabled</code>	<code>boolean</code>	Disable an MCP server without removing its configuration.
<code>mcp_servers.<id>.required</code>	<code>boolean</code>	When true, fail startup/resume if this enabled MCP server cannot initialize.

<code>mcp_servers.<id>.startup_timeout_sec</code>	number	Override the default 10s startup timeout for an MCP server.
<code>mcp_servers.<id>.startup_timeout_ms</code>	number	Alias for <code>startup_timeout_sec</code> in milliseconds.
<code>mcp_servers.<id>.tool_timeout_sec</code>	number	Override the default 60s per-tool timeout for an MCP server.
<code>mcp_servers.<id>.enabled_tools</code>	array<string>	Allow list of tool names exposed by the MCP server.
<code>mcp_servers.<id>.disabled_tools</code>	array<string>	Deny list applied after <code>enabled_tools</code> for the MCP server.
<code>mcp_servers.<id>.default_tools_approval_mode</code>	auto / prompt / writes / approve	Default approval behavior for MCP tools on this server unless a per-tool override exists.
<code>mcp_servers.<id>.tools.<tool>.approval_mode</code>	auto / prompt / writes / approve	Per-tool approval behavior override for one MCP tool on this server.
<code>mcp_servers.<id>.tools.<tool>.output_token_limit</code>	integer (positive)	Token budget for one MCP tool's output, before the standard 20% serialization allowance.
<code>mcp_servers.<id>.scopes</code>	array<string>	OAuth scopes to request when authenticating to that MCP server.
<code>mcp_servers.<id>.oauth_resource</code>	string	Optional RFC 8707 OAuth resource parameter to include during MCP login.
<code>mcp_servers.<id>.experimental_environment</code>	local / remote	Experimental placement for an MCP server. <code>remote</code> starts stdio servers through a remote executor environment; streamable HTTP...
<code>agents</code>	table	Multi-agent settings and custom role declarations.
<code>agents.enabled</code>	boolean	Enable or disable multi-agent tools (default: true).
<code>agents.max_concurrent_threads_per_session</code>	number	Maximum number of spawned-agent threads that can be open concurrently, excluding the primary thread.
<code>agents.max_threads</code>	number	Legacy alias for <code>agents.max_concurrent_threads_per_session</code> .
<code>agents.default_subagent_model</code>	string	Default model for spawned agents.
<code>agents.default_subagent_reasoning_effort</code>	string	Default reasoning effort for spawned agents.
<code>agents.interrupt_message</code>	boolean	Record a model-visible message when an agent turn is interrupted (default: true).
<code>agents.<name>.description</code>	string	Role guidance shown to Codex when choosing and spawning that agent type.
<code>agents.<name>.config_file</code>	string (path)	Path to a TOML config layer for that role; relative paths resolve from the config file that declares the role.
<code>memories.generate_memories</code>	boolean	When <code>false</code> , newly created threads are not stored as memory-generation inputs.
<code>memories.use_memories</code>	boolean	When <code>false</code> , Codex skips injecting existing memories into future sessions.

<code>memories.disable_on_external_context</code>	boolean	When <code>true</code> , threads that use external context such as MCP tool calls, web search, or tool search are kept out of memory...
<code>memories.max_raw_memories_for_consolidation</code>	number	Maximum recent raw memories retained for global consolidation.
<code>memories.max_unused_days</code>	number	Maximum days since a memory was last used before it becomes ineligible for consolidation.
<code>memories.max_rollout_age_days</code>	number	Maximum age of threads considered for memory generation.
<code>memories.max_rollouts_per_startup</code>	number	Maximum rollout candidates processed per startup pass.
<code>memories.min_rollout_idle_hours</code>	number	Minimum idle time before a thread is considered for memory generation.
<code>memories.min_rate_limit_remaining_percent</code>	number	Minimum remaining percentage required in Codex rate-limit windows before memory generation starts.
<code>memories.extract_model</code>	string	Optional model override for per-thread memory extraction.
<code>memories.consolidation_model</code>	string	Optional model override for global memory consolidation.
<code>features.prefer_mxc</code>	boolean	Prefer MXC for local Windows execution when native capabilities and policy allow it; otherwise retain the configured legacy...
<code>features.unified_exec</code>	boolean	Use resumable command execution (stable; enabled on all platforms).
<code>features.shell_snapshot</code>	boolean	Snapshot shell environment to speed up repeated commands (stable; on by default).
<code>features.multi_agent</code>	boolean	Enable multi-agent collaboration tools (<code>spawn_agent</code> , <code>send_input</code> , <code>resume_agent</code> , <code>wait_agent</code> , and <code>close_agent</code>) (stable;...
<code>features.goals</code>	boolean	Enable persisted goals and automatic continuation (stable; on by default).
<code>features.remote_plugin</code>	boolean	Enable the remote plugin catalog (stable; on by default).
<code>features.personality</code>	boolean	Removed feature flag.
<code>features.network_proxy</code>	boolean / table	Start the network proxy for sandboxed commands (experimental; off by default).
<code>features.network_proxy.enabled</code>	boolean	Start the sandboxed-command network proxy when command network access is enabled.
<code>features.network_proxy.domains</code>	map<string, allow / deny>	Domain policy for sandboxed networking.
<code>features.network_proxy.unix_sockets</code>	map<string, allow / deny>	Unix socket policy for sandboxed networking.
<code>features.network_proxy.allow_local_binding</code>	boolean	Allow broader local/private-network access.
<code>features.network_proxy.enable_socks5</code>	boolean	Expose SOCKS5 support.

<code>features.network_proxy.enable_socks5_udp</code>	boolean	Allow UDP over SOCKS5. Defaults to <code>true</code> .
<code>features.network_proxy.allow_upstream_proxy</code>	boolean	Allow chaining through an upstream proxy from the environment.
<code>features.network_proxy.dangerously_allow_non_loopback_proxy</code>	boolean	Permit non-loopback listener addresses.
<code>features.network_proxy.dangerously_allow_all_unix_sockets</code>	boolean	Permit arbitrary Unix socket destinations instead of allowlist-only access.
<code>features.web_search</code>	boolean	Deprecated legacy toggle; prefer the top-level <code>web_search</code> setting.
<code>features.shell_tool</code>	boolean	Enable command execution tools (stable; on by default).
<code>features.enable_request_compression</code>	boolean	Compress streaming request bodies with zstd when supported (stable; on by default).
<code>features.skill_mcp_dependency_install</code>	boolean	Allow prompting and installing missing MCP dependencies for skills (stable; on by default).
<code>features.fast_mode</code>	boolean	Enable Fast mode controls advertised by the active model (stable; on by default), independently of <code>features.ultrafast_mode</code> .
<code>features.ultrafast_mode</code>	boolean	Enable Ultrafast mode controls (stable; on by default), independently of <code>features.fast_mode</code> .
<code>features.prevent_idle_sleep</code>	boolean	Prevent the machine from sleeping while a turn is actively running (experimental; off by default).
<code>suppress_unstable_features_warning</code>	boolean	Suppress the warning that appears when under-development feature flags are enabled.
<code>model_providers.<id></code>	table	Custom provider definition.
<code>model_providers.<id>.name</code>	string	Display name for a custom model provider.
<code>model_providers.<id>.base_url</code>	string	API base URL for the model provider.
<code>model_providers.<id>.model_catalog_url</code>	string	Full URL for a Codex-native model catalog.
<code>model_providers.<id>.capabilities.external_web_access</code>	boolean	Whether the provider's hosted web search can access the live web.

<code>model_providers.<id>.capabilities.remote_compaction</code>	<code>unsupported / v2</code>	Provider compaction protocol. <code>unsupported</code> uses local compaction; <code>v2</code> sends compaction triggers through Responses.
<code>model_providers.<id>.env_key</code>	<code>string</code>	Environment variable supplying the provider API key.
<code>model_providers.<id>.env_key_instructions</code>	<code>string</code>	Optional setup guidance for the provider API key.
<code>model_providers.<id>.experimental_bearer_token</code>	<code>string</code>	Direct bearer token for the provider (discouraged; use <code>env_key</code>).
<code>model_providers.<id>.requires_openai_auth</code>	<code>boolean</code>	The provider uses OpenAI authentication (defaults to false).
<code>model_providers.<id>.wire_api</code>	<code>responses</code>	Protocol used by the provider. <code>responses</code> is the only supported value, and it is the default when omitted.
<code>model_providers.<id>.query_params</code>	<code>map<string,string></code>	Extra query parameters appended to provider requests.
<code>model_providers.<id>.http_headers</code>	<code>map<string,string></code>	Static HTTP headers added to provider requests.
<code>model_providers.<id>.env_http_headers</code>	<code>map<string,string></code>	HTTP headers populated from environment variables when present.
<code>model_providers.<id>.request_max_retries</code>	<code>number</code>	Retry count for HTTP requests to the provider (default: 4).
<code>model_providers.<id>.stream_max_retries</code>	<code>number</code>	Retry count for SSE streaming interruptions (default: 5).
<code>model_providers.<id>.stream_idle_timeout_ms</code>	<code>number</code>	Idle timeout for SSE streams in milliseconds (default: 300000).
<code>model_providers.<id>.supports_websockets</code>	<code>boolean</code>	Whether that provider supports the Responses API WebSocket transport.
<code>model_providers.<id>.supports_standalone_web_search</code>	<code>boolean</code>	Advertise support for a compatible standalone web search endpoint (default: false).
<code>model_providers.<id>.auth</code>	<code>table</code>	Command-backed bearer token configuration for a custom provider.
<code>model_providers.<id>.auth.command</code>	<code>string</code>	Command to run when Codex needs a bearer token.
<code>model_providers.<id>.auth.args</code>	<code>array<string></code>	Arguments passed to the token command.
<code>model_providers.<id>.auth.timeout_ms</code>	<code>number</code>	Maximum token command runtime in milliseconds (default: 5000).
<code>model_providers.<id>.auth.refresh_interval_ms</code>	<code>number</code>	How often Codex proactively refreshes the token in milliseconds (default: 300000).
<code>model_providers.<id>.auth.cwd</code>	<code>string (path)</code>	Working directory for the token command.
<code>model_providers.amazon-bedrock.aws.profile</code>	<code>string</code>	AWS profile name used by the built-in <code>amazon-bedrock</code> provider.
<code>model_providers.amazon-bedrock.aws.region</code>	<code>string</code>	AWS region used by the built-in <code>amazon-bedrock</code> provider.

<code>model_reasoning_effort</code>	<code>string</code>	Reasoning effort advertised by the selected model, such as <code>low</code> , <code>medium</code> , <code>high</code> , <code>xhigh</code> , <code>max</code> , or <code>ultra</code> .
<code>plan_mode_reasoning_effort</code>	<code>string</code>	Plan-mode-specific reasoning override using a level supported by the selected model.
<code>model_reasoning_summary</code>	<code>auto / concise / detailed / none</code>	Select reasoning summary detail or disable summaries entirely.
<code>model_verbosity</code>	<code>low / medium / high</code>	Optional GPT-5 Responses API verbosity override; when unset, the selected model/preset default is used.
<code>model_supports_reasoning_summaries</code>	<code>boolean</code>	Force Codex to send or not send reasoning metadata.
<code>shell_environment_policy.inherit</code>	<code>all / core / none</code>	Baseline environment inheritance when spawning subprocesses.
<code>shell_environment_policy.ignore_default_excludes</code>	<code>boolean</code>	Keep variables containing KEY, SECRET, or TOKEN before other filters run (default: true).
<code>shell_environment_policy.filters</code>	<code>map<string, include / exclude></code>	Canonical case-insensitive environment-variable pattern filters.
<code>shell_environment_policy.exclude</code>	<code>array<string></code>	Legacy environment-variable exclusion patterns.
<code>shell_environment_policy.include_only</code>	<code>array<string></code>	Legacy allowlist of environment-variable patterns.
<code>shell_environment_policy.set</code>	<code>map<string, string></code>	Explicit environment values injected after exclusions; include filters can still remove them.
<code>shell_environment_policy.experimental_use_profile</code>	<code>boolean</code>	Use the user shell profile when spawning subprocesses.
<code>project_root_markers</code>	<code>array<string></code>	List of project root marker filenames; used when searching parent directories for the project root.
<code>project_doc_max_bytes</code>	<code>number</code>	Maximum bytes read from <code>AGENTS.md</code> when building project instructions.
<code>project_doc_fallback_filenames</code>	<code>array<string></code>	Additional filenames to try when <code>AGENTS.md</code> is missing.
<code>history.persistence</code>	<code>save-all / none</code>	Control whether Codex saves session transcripts to history.jsonl.
<code>tool_output_token_limit</code>	<code>number</code>	Token budget for storing individual tool/function outputs in history.

<code>background_terminal_max_timeout</code>	number	Maximum poll window in milliseconds for empty <code>write_stdin</code> polls (background terminal polling).
<code>history.max_bytes</code>	number	If set, caps the history file size in bytes by dropping oldest entries.
<code>file_opener</code>	<code>vscode</code> / <code>vscode-insiders</code> / <code>windsurf</code> / ...	URI scheme used to open citations from Codex output (default: <code>vscode</code>).
<code>otel.environment</code>	string	Environment tag applied to emitted OpenTelemetry events (default: <code>dev</code>).
<code>otel.exporter</code>	<code>none</code> / <code>otlp-http</code> / <code>otlp-grpc</code>	Select the OpenTelemetry exporter and provide any endpoint metadata.
<code>otel.trace_exporter</code>	<code>none</code> / <code>otlp-http</code> / <code>otlp-grpc</code>	Select the OpenTelemetry trace exporter and provide any endpoint metadata.
<code>otel.metrics_exporter</code>	<code>none</code> / <code>statsig</code> / <code>otlp-http</code> / <code>otlp-grpc</code>	Select the OpenTelemetry metrics exporter (defaults to <code>statsig</code>).
<code>otel.log_user_prompt</code>	boolean	Opt in to exporting raw user prompts with OpenTelemetry logs.
<code>otel.exporter.<id>.endpoint</code>	string	Exporter endpoint for OTEL logs.
<code>otel.exporter.<id>.protocol</code>	<code>binary</code> / <code>json</code>	Protocol used by the OTLP/HTTP exporter.
<code>otel.exporter.<id>.headers</code>	map<string, string>	Static headers included with OTEL exporter requests.
<code>otel.trace_exporter.<id>.endpoint</code>	string	Trace exporter endpoint for OTEL logs.
<code>otel.trace_exporter.<id>.protocol</code>	<code>binary</code> / <code>json</code>	Protocol used by the OTLP/HTTP trace exporter.
<code>otel.trace_exporter.<id>.headers</code>	map<string, string>	Static headers included with OTEL trace exporter requests.
<code>otel.exporter.<id>.tls.ca-certificate</code>	string	CA certificate path for OTEL exporter TLS.
<code>otel.exporter.<id>.tls.client-certificate</code>	string	Client certificate path for OTEL exporter TLS.
<code>otel.exporter.<id>.tls.client-private-key</code>	string	Client private key path for OTEL exporter TLS.
<code>otel.trace_exporter.<id>.tls.ca-certificate</code>	string	CA certificate path for OTEL trace exporter TLS.
<code>otel.trace_exporter.<id>.tls.client-certificate</code>	string	Client certificate path for OTEL trace exporter TLS.
<code>otel.trace_exporter.<id>.tls.client-private-key</code>	string	Client private key path for OTEL trace exporter TLS.
<code>desktop.custom_file_handlers.<id></code>	table	User-level only.
<code>desktop.custom_file_handlers.<id>.label</code>	string	Display name shown in Open in menus.
<code>desktop.custom_file_handlers.<id>.icon</code>	string	Bundled asset path, Base64-encoded <code>data:image/...</code> URL, file URI, or absolute local path for the handler icon.

<code>desktop.custom_file_handlers.<id>.command</code>	<code>string</code>	Executable path or command name to detect and launch.
<code>desktop.custom_file_handlers.<id>.args</code>	<code>array<string></code>	Arguments inserted between the command and file input (default: <code>[]</code>).
<code>desktop.custom_file_handlers.<id>.input</code>	<code>path</code> / <code>json_argument</code> / <code>json_stdin</code>	How the app sends file input to the handler (default: <code>path</code>).
<code>desktop.custom_file_handlers.<id>.supports_ssh</code>	<code>boolean</code>	Offer the handler for files in SSH workspaces (default: <code>false</code>).
<code>tui</code>	<code>table</code>	TUI-specific options such as enabling inline desktop notifications.
<code>tui.notifications</code>	<code>boolean</code> / <code>array<string></code>	Enable TUI notifications; optionally restrict to specific event types.
<code>tui.notification_method</code>	<code>auto</code> / <code>osc9</code> / <code>bel</code>	Notification method for terminal notifications (default: <code>auto</code>).
<code>tui.notification_condition</code>	<code>unfocused</code> / <code>always</code>	Control whether TUI notifications fire only when the terminal is unfocused or regardless of focus.
<code>tui.animations</code>	<code>boolean</code>	Enable terminal animations (welcome screen, shimmer, spinner) (default: <code>true</code>).
<code>tui.disable_paste_burst</code>	<code>boolean</code>	Disable burst-paste detection in the TUI (default: <code>false</code>).
<code>tui.alternate_screen</code>	<code>auto</code> / <code>always</code> / <code>never</code>	Control alternate screen usage for the TUI (default: <code>auto</code> ; <code>auto</code> skips it in Zellij to preserve scrollback).
<code>tui.resume_cwd</code>	<code>current</code> / <code>session</code>	Working directory to use when resuming or forking a session.
<code>tui.vim_mode_default</code>	<code>boolean</code>	Start the composer in Vim normal mode instead of insert mode (default: <code>false</code>).
<code>tui.raw_output_mode</code>	<code>boolean</code>	Start the TUI in raw scrollback mode for copy-friendly terminal selection (default: <code>false</code>).
<code>tui.show_tooltips</code>	<code>boolean</code>	Show onboarding tooltips in the TUI welcome screen (default: <code>true</code>).
<code>tui.auto_recap</code>	<code>boolean</code>	Generate automatic conversation recaps while the terminal is unfocused (default: <code>true</code>).
<code>tui.show_server_version_notice</code>	<code>boolean</code>	Show informational notices about connected app-server version differences (default: <code>true</code>).
<code>tui.status_line</code>	<code>array<string></code> / <code>null</code>	Ordered list of TUI footer status-line item identifiers. <code>null</code> disables the status line.
<code>tui.theme</code>	<code>string</code>	Syntax-highlighting theme override (kebab-case theme name).
<code>tui.keymap.<context>.<action></code>	<code>string</code> / <code>array<string></code>	Keyboard shortcut binding for a TUI action.
<code>tui.keymap.<context>.<action> = []</code>	<code>empty array</code>	Unbind the action in that keymap context.
<code>marketplaces.<name>.source_type</code>	<code>git</code> / <code>local</code>	Source kind for a configured plugin marketplace.

<code>marketplaces.<name>.source</code>	<code>string</code>	Git repository location or local marketplace root directory.
<code>marketplaces.<name>.ref</code>	<code>string</code>	Optional Git branch, tag, or commit for the marketplace.
<code>marketplaces.<name>.sparse_paths</code>	<code>array<string></code>	Optional sparse checkout paths for a Git marketplace.
<code>plugins.<plugin>.enabled</code>	<code>boolean</code>	Enable or disable a local-marketplace plugin using a <code>plugin-name@marketplace-name</code> key.
<code>plugins.<plugin>.mcp_servers.<server>.enabled</code>	<code>boolean</code>	Enable or disable an MCP server bundled by an installed plugin without changing the plugin manifest.
<code>plugins.<plugin>.mcp_servers.<server>.default_tools_approval_mode</code>	<code>auto / prompt / writes / approve</code>	Default approval behavior for tools on a plugin-provided MCP server.
<code>plugins.<plugin>.mcp_servers.<server>.enabled_tools</code>	<code>array<string></code>	Allow list of tools exposed from a plugin-provided MCP server.
<code>plugins.<plugin>.mcp_servers.<server>.disabled_tools</code>	<code>array<string></code>	Deny list applied after <code>enabled_tools</code> for a plugin-provided MCP server.
<code>plugins.<plugin>.mcp_servers.<server>.tools.<tool>.approval_mode</code>	<code>auto / prompt / writes / approve</code>	Per-tool approval behavior override for a plugin-provided MCP tool.
<code>tui.model_availability_nux.<model></code>	<code>integer</code>	Internal startup-tooltip state keyed by model slug.
<code>hide_agent_reasoning</code>	<code>boolean</code>	Suppress reasoning events in both the TUI and <code>codex exec</code> output.
<code>show_raw_agent_reasoning</code>	<code>boolean</code>	Surface raw reasoning content when the active model emits it.
<code>disable_paste_burst</code>	<code>boolean</code>	Legacy fallback for <code>tui.disable_paste_burst</code> .
<code>windows_wsl_setup_acknowledged</code>	<code>boolean</code>	Track Windows onboarding acknowledgement (Windows only).
<code>chatgpt_base_url</code>	<code>string</code>	Override the base URL used during the ChatGPT login flow.
<code>cli_auth_credentials_store</code>	<code>file / keyring / auto / ephemeral</code>	Control where the CLI stores cached credentials.
<code>mcp_oauth_credentials_store</code>	<code>auto / file / keyring</code>	Preferred store for MCP OAuth credentials.
<code>mcp_oauth_callback_port</code>	<code>integer</code>	Optional global fixed port for the local HTTP callback server used during MCP OAuth login.
<code>mcp_oauth_callback_url</code>	<code>string</code>	Optional base callback URL for MCP OAuth login, such as a devbox ingress URL. Newly added pre-registered clients use this URL...
<code>experimental_use_unified_exec_tool</code>	<code>boolean</code>	Legacy name for enabling unified exec; prefer <code>[features].unified_exec</code> or <code>codex --enable unified_exec</code> .
<code>tools.view_image</code>	<code>boolean</code>	Enable the local-image attachment tool <code>view_image</code> .

<code>default_permissions</code>	string	Name of the default permissions profile to apply to sandboxed tool calls.
<code>permissions.<name>.description</code>	string	Human-readable description for this named profile.
<code>permissions.<name>.extends</code>	string	Optional parent profile applied before this named profile.
<code>permissions.<name>.workspace_roots</code>	table	Profile-defined workspace roots that receive <code>:workspace_roots</code> filesystem rules alongside the session's runtime workspace roots.
<code>permissions.<name>.workspace_roots.<path></code>	boolean	Opt a path into the profile's workspace root set when <code>true</code> .
<code>permissions.<name>.filesystem</code>	table	Named filesystem permission profile.
<code>permissions.<name>.filesystem.glob_scan_max_depth</code>	number	Maximum depth for expanding deny-read glob patterns on platforms that snapshot matches before sandbox startup.
<code>permissions.<name>.network.enabled</code>	boolean	Enable network access for commands in this permission profile.
<code>permissions.<name>.network.proxy_url</code>	string	HTTP listener URL used when this permissions profile enables sandboxed networking.
<code>permissions.<name>.network.enable_socks5</code>	boolean	Expose SOCKS5 support when this permissions profile enables sandboxed networking.
<code>permissions.<name>.network.socks_url</code>	string	SOCKS5 proxy endpoint used by this permissions profile.
<code>permissions.<name>.network.enable_socks5_udp</code>	boolean	Allow UDP over the SOCKS5 listener when enabled.
<code>permissions.<name>.network.allow_upstream_proxy</code>	boolean	Allow sandboxed networking to chain through another upstream proxy.
<code>permissions.<name>.network.dangerously_allow_non_loopback_proxy</code>	boolean	Permit non-loopback bind addresses for sandboxed networking listeners.
<code>permissions.<name>.network.dangerously_allow_all_unix_sockets</code>	boolean	Allow arbitrary Unix socket destinations instead of the default restricted set.
<code>permissions.<name>.network.mode</code>	limited / full	Network proxy mode used for subprocess traffic.
<code>permissions.<name>.network.domains</code>	table	Domain rules for sandboxed commands.
<code>permissions.<name>.network.domains.<pattern></code>	allow / deny	Allow or deny an exact host or scoped wildcard pattern such as <code>*.example.com</code> or <code>**.example.com</code> .
<code>permissions.<name>.network.unix_sockets</code>	table	Unix socket allowlist overrides for sandboxed networking.
<code>permissions.<name>.network.unix_sockets.<path></code>	allow / deny	Add an absolute Unix socket path to the effective allowlist with <code>allow</code> , or reject it with <code>deny</code> .
<code>permissions.<name>.network.allow_local_binding</code>	boolean	Permit broader local/private-network access through sandboxed networking.

<code>notice.hide_full_access_warning</code>	<code>boolean</code>	Track acknowledgement of the full access warning prompt.
<code>notice.hide_world_writable_warning</code>	<code>boolean</code>	Track acknowledgement of the Windows world-writable directories warning.
<code>notice.hide_rate_limit_model_nudge</code>	<code>boolean</code>	Track opt-out of the rate limit model switch reminder.
<code>notice.hide_gpt5_1_migration_prompt</code>	<code>boolean</code>	Track acknowledgement of the GPT-5.1 migration prompt.
<code>notice.hide_gpt-5.1-codex-max_migration_prompt</code>	<code>boolean</code>	Track acknowledgement of the gpt-5.1-codex-max migration prompt.
<code>notice.model_migrations</code>	<code>map<string,string></code>	Track acknowledged model migrations as old->new mappings.
<code>forced_login_method</code>	<code>chatgpt / api</code>	Restrict Codex to a specific authentication method.
<code>forced_chatgpt_workspace_id</code>	<code>string (uuid)</code>	Limit ChatGPT logins to a specific workspace identifier.

16.3 requirements.toml keys (174)

Admins use `requirements.toml` to enforce policy that users can't override. For example, `allow_managed_hooks_only = true` ignores user, project and session hooks, and works only in this file.

KEY	TYPE	WHAT IT DOES
<code>sqlite_home</code>	<code>string (path)</code>	Enforce the directory where Codex stores SQLite-backed runtime state.
<code>log_dir</code>	<code>string (path)</code>	Enforce the directory where Codex writes local log files.
<code>model_catalog_json</code>	<code>string (path)</code>	Enforce the JSON model catalog Codex uses at startup.
<code>check_for_update_on_startup</code>	<code>boolean</code>	Enforce whether Codex checks for updates when it starts.
<code>allow_login_shell</code>	<code>boolean</code>	Enforce whether shell tools can start a login shell.
<code>allowed_login_methods</code>	<code>array<string></code>	Allow <code>chatgpt</code> , <code>api</code> , or both.
<code>allowed_chatgpt_workspaces</code>	<code>array<string></code>	Restrict ChatGPT login, including Codex access tokens, to the listed workspace IDs.
<code>cli_auth_credentials_store</code>	<code>file / keyring / auto / ephemeral</code>	Enforce the CLI credential store before authentication loads. <code>file</code> uses <code>CODEX_HOME/auth.json</code> ; <code>keyring</code> requires the OS...
<code>chatgpt_base_url</code>	<code>string</code>	Enforce the ChatGPT service base URL before authentication and cloud-policy retrieval.
<code>feedback</code>	<code>table</code>	Managed feedback settings.
<code>feedback.enabled</code>	<code>boolean</code>	Enforce whether users can submit feedback across Codex clients.
<code>allowed_approval_policies</code>	<code>array<string></code>	Allowed approval policies, such as <code>on-request</code> , <code>never</code> , and <code>granular</code> .

<code>allowed_approvals_reviewers</code>	<code>array<string></code>	Allowed values for <code>approvals_reviewer</code> , such as <code>user</code> and <code>auto_review</code> .
<code>guardian_policy_config</code>	<code>string</code>	Managed Markdown policy instructions for automatic review.
<code>guardian_extra_policy</code>	<code>string</code>	Additional managed Markdown policy for automatic review, included alongside the main policy.
<code>additional_developer_instructions</code>	<code>string</code>	Managed developer instructions added as a separate developer message.
<code>auto_review</code>	<code>table</code>	Managed automatic-review requirements.
<code>auto_review.required_on_models</code>	<code>array<string></code>	Model slugs that must use automatic review.
<code>auto_review.ignore_rules</code>	<code>array<string></code>	Full model slugs for which Codex ignores <code>allow</code> prefix rules in command execution policy.
<code>allowed_permission_profiles</code>	<code>table<boolean></code>	Complete list of allowed permission profiles.
<code>allowed_permission_profiles.<name></code>	<code>boolean</code>	Allow or deny a built-in or custom permission profile defined in a loaded config or requirements source.
<code>default_permissions</code>	<code>string</code>	Managed default permission profile.
<code>enforce_residency</code>	<code>string</code>	Require Codex service traffic to use a supported data residency.
<code>model_provider</code>	<code>string</code>	Enforce the model provider ID, overriding local and session configuration.
<code>model_providers</code>	<code>map<string, table></code>	Managed model provider definitions.
<code>model_providers.<id></code>	<code>table</code>	Complete managed provider definition.
<code>models</code>	<code>table</code>	Contains the <code>[models.new_thread]</code> table.
<code>models.new_thread</code>	<code>table</code>	Optional defaults to apply when a new local thread starts.
<code>models.new_thread.model</code>	<code>string</code>	Default model for new threads.
<code>models.new_thread.model_reasoning_effort</code>	<code>string</code>	Default reasoning effort for new threads.
<code>models.new_thread.service_tier</code>	<code>string</code>	Default service tier for new threads.
<code>permissions</code>	<code>table</code>	Admin-defined permission profiles keyed by profile name.
<code>permissions.<name></code>	<code>table</code>	Admin-defined permission profile.
<code>allowed_sandbox_modes</code>	<code>array<string></code>	Allowed values for <code>sandbox_mode</code> .
<code>windows</code>	<code>table</code>	Native Windows sandbox requirements.
<code>windows.allowed_sandbox_implementations</code>	<code>array<string></code>	Allowed legacy native Windows sandbox implementations (<code>elevated</code> and <code>unelevated</code>).
<code>windows.allow_mxc</code>	<code>boolean</code>	Set to <code>false</code> to prohibit both explicit MXC selection and automatic selection through <code>features.prefer_mxc</code> .
<code>remote_sandbox_config</code>	<code>array<table></code>	Host-specific sandbox requirements.

<code>remote_sandbox_config[].hostname_patterns</code>	<code>array<string></code>	Case-insensitive host name patterns.
<code>remote_sandbox_config[].allowed_sandbox_modes</code>	<code>array<string></code>	Allowed sandbox modes to apply when this host-specific entry matches.
<code>allowed_web_search_modes</code>	<code>array<string></code>	Allowed values for <code>web_search</code> (<code>disabled</code> , <code>cached</code> , <code>indexed</code> , <code>live</code>). <code>disabled</code> is always allowed; an empty list...
<code>allow_managed_hooks_only</code>	<code>boolean</code>	When <code>true</code> , Codex skips user, project, session, and plugin hooks while still allowing managed hooks from <code>requirements.toml</code> ...
<code>allow_appshots</code>	<code>boolean</code>	Set to <code>false</code> to disable Appshots for managed users.
<code>allow_remote_control</code>	<code>boolean</code>	Set to <code>false</code> to disable device remote control for managed users.
<code>allow_browser_and_computer_use</code>	<code>boolean</code>	Set to <code>false</code> to block both agent-driven Browser Use and native-app Computer Use.
<code>features.plugin_sharing</code>	<code>boolean</code>	Set to <code>false</code> in cloud-managed <code>requirements.toml</code> to disable workspace sharing for locally built plugins.
<code>features</code>	<code>table</code>	Pinned feature values.
<code>features.<name></code>	<code>boolean</code>	Require a documented runtime or app feature to stay enabled or disabled.
<code>features.apps</code>	<code>boolean</code>	Pin Apps integration availability on or off for managed users.
<code>features.in_app_updates</code>	<code>boolean</code>	Set to <code>false</code> in <code>requirements.toml</code> to disable in-app updates.
<code>features.in_app_browser</code>	<code>boolean</code>	Set to <code>false</code> in <code>requirements.toml</code> to disable the built-in browser pane that users open and control directly.
<code>features.in_app_chat</code>	<code>boolean</code>	Set to <code>false</code> to hide ChatGPT and ChatGPT Work conversation screens and related cloud automation UI in the ChatGPT desktop app.
<code>features.in_app_dictation</code>	<code>boolean</code>	Set to <code>false</code> to disable in-app dictation in the desktop app.
<code>features.in_app_local_automation</code>	<code>boolean</code>	Set to <code>false</code> to disable local scheduled tasks in the desktop app.
<code>features.browser_use</code>	<code>boolean</code>	Set to <code>false</code> in <code>requirements.toml</code> to disable agent-driven Browser Use.
<code>features.browser_use_external</code>	<code>boolean</code>	Set to <code>false</code> in <code>requirements.toml</code> to prevent Codex from operating supported browsers through the ChatGPT browser...
<code>features.browser_use_full_cdp_access</code>	<code>boolean</code>	Set to <code>false</code> in <code>requirements.toml</code> to disable full Chrome DevTools Protocol access in the local runtime, including Browser...
<code>features.fast_mode</code>	<code>boolean</code>	Pin the canonical <code>fast_mode</code> feature on or off for managed users.
<code>features.guardian_approval</code>	<code>boolean</code>	Pin Guardian approval availability on or off for managed users.

<code>features.memories</code>	<code>boolean</code>	Pin Memories availability on or off for managed users.
<code>features.multi_agent</code>	<code>boolean</code>	Pin multi-agent availability on or off for managed users.
<code>features.plugins</code>	<code>boolean</code>	Pin plugin availability on or off for managed users.
<code>features.remote_plugin</code>	<code>boolean</code>	Pin remote plugin catalog availability on or off for managed users.
<code>features.realtime_conversation</code>	<code>boolean</code>	Set to <code>false</code> to disable the experimental <code>/voice</code> command in the Codex CLI. Do not rely on this setting to block ChatGPT...
<code>features.computer_use</code>	<code>boolean</code>	Set to <code>false</code> in <code>requirements.toml</code> to disable Computer Use, Record & Replay, and related install or enablement flows.
<code>features.workspace_dependencies</code>	<code>boolean</code>	Pin bundled workspace-dependency runtime availability on or off for managed users.
<code>in_app_browser</code>	<code>table</code>	Requirements for the built-in browser pane.
<code>in_app_browser.allow_external_browser_settings_import</code>	<code>boolean</code>	Set to <code>false</code> to prevent users from importing settings or browsing data from an external browser into the built-in browser.
<code>browser_use</code>	<code>table</code>	Managed requirements for agent-driven Browser Use.
<code>browser_use.allow_webmcp</code>	<code>boolean</code>	Set to <code>true</code> to opt the desktop app's built-in browser into site tools (WebMCP).
<code>browser_use.allow_history_access</code>	<code>boolean</code>	Set to <code>false</code> to prevent Browser Use from reading browser history.
<code>browser_use.disable_auto_review</code>	<code>boolean</code>	Set to <code>true</code> to skip automatic review for Browser Use and ask the user for approval instead.
<code>browser_use.allow_global_persistent_approval</code>	<code>boolean</code>	Set to <code>false</code> to prevent Browser Use from creating or honoring <code>Always allow</code> approvals that cover every site, such as...
<code>browser_use.default_origin_policy</code>	<code>table</code>	Fallback for each Browser Use setting when no matching entry under <code>browser_use.origins</code> defines it.
<code>browser_use.default_origin_policy.access</code>	<code>allow / deny</code>	Use <code>deny</code> to block Browser Use on origins that use the fallback.
<code>browser_use.default_origin_policy.downloads</code>	<code>allow / deny</code>	Use <code>deny</code> to block Browser Use downloads on origins that use the fallback. <code>allow</code> only lets normal approval and policy...
<code>browser_use.default_origin_policy.uploads</code>	<code>allow / deny</code>	Use <code>deny</code> to block Browser Use uploads on origins that use the fallback. <code>allow</code> only lets normal approval and policy checks...
<code>browser_use.default_origin_policy.full_cdp_access</code>	<code>allow / deny</code>	Use <code>deny</code> to block full Chrome DevTools Protocol (CDP) access on origins that use the fallback. <code>allow</code> only lets normal...

<code>browser_use.default_origin_policy.auto_review</code>	<code>allow / deny</code>	Use <code>deny</code> to skip automatic review on origins that use the fallback and ask the user for approval instead. <code>allow</code> leaves...
<code>browser_use.default_origin_policy.persistent_approval</code>	<code>boolean</code>	Set to <code>false</code> to prevent Browser Use from saving or honoring an <code>Always allow</code> approval on origins that use the fallback.
<code>browser_use.default_origin_policy.access_approval_lifetime</code>	<code>turn / thread</code>	Set how long a non-persistent site-access approval lasts: <code>turn</code> limits it to the current turn, and <code>thread</code> keeps it for the...
<code>browser_use.origins.<pattern></code>	<code>table</code>	Policy for origins matching this pattern.
<code>browser_use.origins.<pattern>.access</code>	<code>allow / deny</code>	Use <code>deny</code> to block Browser Use on matching origins.
<code>browser_use.origins.<pattern>.downloads</code>	<code>allow / deny</code>	Use <code>deny</code> to block Browser Use downloads on matching origins. <code>allow</code> only lets normal approval and policy checks continue.
<code>browser_use.origins.<pattern>.uploads</code>	<code>allow / deny</code>	Use <code>deny</code> to block Browser Use uploads on matching origins. <code>allow</code> only lets normal approval and policy checks continue.
<code>browser_use.origins.<pattern>.full_cdp_access</code>	<code>allow / deny</code>	Use <code>deny</code> to block full Chrome DevTools Protocol (CDP) access on matching origins. <code>allow</code> only lets normal opt-in and...
<code>browser_use.origins.<pattern>.auto_review</code>	<code>allow / deny</code>	Use <code>deny</code> to skip automatic review on matching origins and ask the user for approval instead. <code>allow</code> leaves automatic review...
<code>browser_use.origins.<pattern>.persistent_approval</code>	<code>boolean</code>	Set to <code>false</code> to prevent Browser Use from saving or honoring an <code>Always allow</code> approval on matching origins.
<code>browser_use.origins.<pattern>.access_approval_lifetime</code>	<code>turn / thread</code>	Set how long a non-persistent site-access approval for matching origins lasts: <code>turn</code> limits it to the current turn, and...
<code>computer_use</code>	<code>table</code>	Managed requirements for agent-driven work in native desktop apps.
<code>computer_use.allow_locked_computer_use</code>	<code>boolean</code>	Set to <code>false</code> to prevent users from enabling Locked Use on a managed macOS device.
<code>computer_use.allow_persistent_approval</code>	<code>boolean</code>	Set to <code>false</code> to remove the option to save app approvals across sessions.

<code>computer_use.default_app_access</code>	<code>allow / deny</code>	Fallback access for native apps that do not match a platform-specific rule. <code>deny</code> blocks access. <code>allow</code> only lets normal...
<code>computer_use.macos</code>	<code>table</code>	Computer Use app rules for macOS.
<code>computer_use.macos.bundle_ids</code>	<code>map<string, allow / deny></code>	Map exact macOS bundle identifiers to <code>allow</code> or <code>deny</code> .
<code>computer_use.macos.bundle_ids.<bundle-id></code>	<code>allow / deny</code>	Use <code>deny</code> to block the exact bundle identifier. <code>allow</code> overrides only this policy source's default and still requires any...
<code>computer_use.windows</code>	<code>table</code>	Computer Use app rules for packaged and unpackaged Windows apps.
<code>computer_use.windows.aumids</code>	<code>map<string, allow / deny></code>	Map exact, registered Application User Model IDs (AUMIDs) for signed packaged apps to <code>allow</code> or <code>deny</code> .
<code>computer_use.windows.aumids.<aumid></code>	<code>allow / deny</code>	Use <code>deny</code> to block the exact packaged-app identity. <code>allow</code> overrides only this policy source's default and still requires...
<code>computer_use.windows.exes</code>	<code>array<table></code>	Rules for signed, unpackaged Windows executables.
<code>computer_use.windows.exes[].publisher_name</code>	<code>string</code>	Required exact publisher name from the executable's trusted signing certificate, formatted as a Windows X.500 distinguished name.
<code>computer_use.windows.exes[].product_name</code>	<code>string</code>	Required exact <code>ProductName</code> from the executable's signed version information.
<code>computer_use.windows.exes[].binary_name</code>	<code>string</code>	Optional <code>OriginalFilename</code> from the executable's signed version information.
<code>computer_use.windows.exes[].access</code>	<code>allow / deny</code>	Required access decision for matching executables. <code>deny</code> blocks access. <code>allow</code> overrides only this policy source's default...
<code>application</code>	<code>table</code>	Managed desktop application requirements.
<code>application.network</code>	<code>table</code>	Destination restrictions for desktop app network requests, separate from command networking and browser origin rules.
<code>application.network.enabled</code>	<code>boolean</code>	Defaults to true when the table is present.
<code>application.network.domains</code>	<code>table<allow / deny></code>	Exact domain rules for desktop app requests.
<code>application.network.domains.<domain></code>	<code>allow / deny</code>	Allow or deny an exact domain.
<code>experimental_network</code>	<code>table</code>	Administrator-managed network requirements for sandboxed local commands, enforced from <code>requirements.toml</code> .
<code>experimental_network.enabled</code>	<code>boolean</code>	Enable sandboxed networking requirements.
<code>experimental_network.http_port</code>	<code>integer</code>	Loopback HTTP listener port to use for <code>[experimental_network]</code> requirements.
<code>experimental_network.socks_port</code>	<code>integer</code>	Loopback SOCKS5 listener port to use for <code>[experimental_network]</code> requirements.

<code>experimental_network.allow_upstream_proxy</code>	<code>boolean</code>	Allow sandboxed networking to chain through an upstream proxy from the environment.
<code>experimental_network.dangerously_allow_non_loopback_proxy</code>	<code>boolean</code>	Permit non-loopback listener addresses for <code>[experimental_network]</code> requirements.
<code>experimental_network.dangerously_allow_all_unix_sockets</code>	<code>boolean</code>	Permit arbitrary Unix socket destinations instead of allowlist-only access.
<code>experimental_network.domains</code>	<code>map<string, allow / deny></code>	Map-shaped administrator domain policy for sandboxed networking.
<code>experimental_network.domains.<pattern></code>	<code>allow / deny</code>	Allow or deny sandboxed network access for the matching domain pattern.
<code>experimental_network.allowed_domains</code>	<code>array<string></code>	Administrator allow rules for sandboxed-command networking while the managed network proxy is enabled.
<code>experimental_network.denied_domains</code>	<code>array<string></code>	List-shaped administrator deny rules for sandboxed networking.
<code>experimental_network.managed_allowed_domains_only</code>	<code>boolean</code>	When networking requirements are enabled and this is true, ordinary user configuration and per-domain approvals cannot expand...
<code>experimental_network.unix_sockets</code>	<code>map<string, allow / deny></code>	Administrator-managed Unix socket allowlist for sandboxed networking on macOS. Paths must be absolute.
<code>experimental_network.unix_sockets.<path></code>	<code>allow / deny</code>	On macOS, <code>allow</code> adds an absolute Unix socket path to the allowlist; <code>deny</code> leaves it out.
<code>experimental_network.allow_local_binding</code>	<code>boolean</code>	Permit broader local/private-network access for sandboxed networking.
<code>hooks</code>	<code>table</code>	Admin-enforced managed lifecycle hooks.
<code>hooks.managed_dir</code>	<code>string (absolute path)</code>	Directory containing managed hook scripts on macOS and Linux.
<code>hooks.windows_managed_dir</code>	<code>string (absolute path)</code>	Directory containing managed hook scripts on Windows.
<code>hooks.<Event></code>	<code>array<table></code>	Matcher groups for a hook event such as <code>PreToolUse</code> , <code>PermissionRequest</code> , <code>PostToolUse</code> , <code>PreCompact</code> , <code>PostCompact</code> ,...

<code>hooks.<Event>[].hooks</code>	<code>array<table></code>	Hook handlers for a matcher group.
<code>hooks.<Event>[].hooks[].async</code>	<code>boolean</code>	Run a command hook in the background without delaying the triggering operation.
<code>hooks.<Event>[].hooks[].additionalContextLimit</code>	<code>integer</code>	Approximate per-handler token threshold for saving oversized <code>additionalContext</code> to disk and showing the model a shorter preview.
<code>hooks.<Event>[].hooks[].commandWindows</code>	<code>string</code>	Windows-only command override for command hooks.
<code>permissions.filesystem.deny_read</code>	<code>array<string></code>	Admin-enforced filesystem read denials.
<code>mcp_servers</code>	<code>table</code>	Allowlist of MCP servers that may be enabled.
<code>mcp_servers.<id>.identity</code>	<code>table</code>	Identity rule for a single MCP server.
<code>mcp_servers.<id>.identity.command</code>	<code>string / table</code>	Allow an MCP stdio server by exact command string, or use a matcher table to require an exact executable and ordered argument...
<code>mcp_servers.<id>.identity.command.executable</code>	<code>string</code>	Executable that the stdio server's configured <code>command</code> must match exactly.
<code>mcp_servers.<id>.identity.command.args</code>	<code>array<table></code>	Ordered argument matchers for a stdio server.
<code>mcp_servers.<id>.identity.command.args[].match</code>	<code>exact / prefix / regex</code>	Match operation for this argument position.
<code>mcp_servers.<id>.identity.command.args[].value</code>	<code>string</code>	Value used by an <code>exact</code> or <code>prefix</code> argument matcher.
<code>mcp_servers.<id>.identity.command.args[].expression</code>	<code>string</code>	Regular expression used by a <code>regex</code> argument matcher.
<code>mcp_servers.<id>.identity.url</code>	<code>string / table</code>	Allow an MCP streamable HTTP server by exact URL string, or use an <code>exact</code> , <code>prefix</code> , or <code>regex</code> value matcher table.
<code>mcp_servers.<id>.identity.url.match</code>	<code>exact / prefix / regex</code>	Match operation for the configured MCP server URL.
<code>mcp_servers.<id>.identity.url.value</code>	<code>string</code>	Value used by an <code>exact</code> or <code>prefix</code> URL matcher.
<code>mcp_servers.<id>.identity.url.expression</code>	<code>string</code>	Regular expression used by a <code>regex</code> URL matcher.
<code>plugins</code>	<code>table</code>	Plugin-specific MCP server allowlists keyed by plugin identifier.
<code>plugins.<plugin>.mcp_servers</code>	<code>table</code>	Allowlist for MCP servers bundled with one plugin.
<code>plugins.<plugin>.mcp_servers.<server>.identity</code>	<code>table</code>	Identity rule for one plugin-bundled MCP server.
<code>plugins.<plugin>.mcp_servers.<server>.identity.command</code>	<code>string / table</code>	Allow a plugin's stdio MCP server by exact command string, or use a matcher table to require an exact executable and ordered...
<code>plugins.<plugin>.mcp_servers.<server>.identity.command.executable</code>	<code>string</code>	Executable that the plugin-bundled stdio server's configured command must match exactly.

<code>plugins.<plugin>.mcp_servers.<server>.identity.command.args</code>	<code>array<table></code>	Ordered argument matchers for a plugin-bundled stdio server.
<code>plugins.<plugin>.mcp_servers.<server>.identity.command.args[].match</code>	<code>exact / prefix / regex</code>	Match operation for this argument position.
<code>plugins.<plugin>.mcp_servers.<server>.identity.command.args[].value</code>	<code>string</code>	Value used by an <code>exact</code> or <code>prefix</code> argument matcher.
<code>plugins.<plugin>.mcp_servers.<server>.identity.command.args[].expression</code>	<code>string</code>	Regular expression used by a <code>regex</code> argument matcher.
<code>plugins.<plugin>.mcp_servers.<server>.identity.url</code>	<code>string / table</code>	Allow a plugin's streamable HTTP MCP server by exact URL string, or use an <code>exact</code> , <code>prefix</code> , or <code>regex</code> value matcher table.
<code>plugins.<plugin>.mcp_servers.<server>.identity.url.match</code>	<code>exact / prefix / regex</code>	Match operation for the plugin-bundled MCP server URL.
<code>plugins.<plugin>.mcp_servers.<server>.identity.url.value</code>	<code>string</code>	Value used by an <code>exact</code> or <code>prefix</code> URL matcher.
<code>plugins.<plugin>.mcp_servers.<server>.identity.url.expression</code>	<code>string</code>	Regular expression used by a <code>regex</code> URL matcher.
<code>marketplaces</code>	<code>table</code>	Admin requirements for plugin marketplace sources.
<code>marketplaces.restrict_to_allowed_sources</code>	<code>boolean</code>	When <code>true</code> , require configured marketplace sources to match <code>allowed_sources</code> for marketplace add, plugin install, refresh,...
<code>marketplaces.allowed_sources</code>	<code>table</code>	Allowed marketplace sources keyed by administrator-chosen rule name.
<code>marketplaces.allowed_sources.<name></code>	<code>table</code>	One allowed source rule.
<code>marketplaces.allowed_sources.<name>.source</code>	<code>git / host_pattern / local</code>	Marketplace source matcher type.
<code>marketplaces.allowed_sources.<name>.ref</code>	<code>string</code>	Optional exact Git ref for a <code>git</code> rule.
<code>apps</code>	<code>table</code>	Managed app requirements keyed by app identifier.
<code>apps.<id>.enabled</code>	<code>boolean</code>	Set to <code>false</code> to disable an app.
<code>apps.<id>.tools.<tool>.approval_mode</code>	<code>auto / prompt / writes / approve</code>	Set the managed approval mode for one app tool.
<code>rules</code>	<code>table</code>	Admin-enforced command rules merged with <code>.rules</code> files.
<code>rules.prefix_rules</code>	<code>array<table></code>	List of enforced prefix rules.
<code>rules.prefix_rules[].pattern</code>	<code>array<table></code>	Command prefix expressed as pattern tokens.
<code>rules.prefix_rules[].pattern[].token</code>	<code>string</code>	A single literal token at this position.
<code>rules.prefix_rules[].pattern[].any_of</code>	<code>array<string></code>	A list of allowed alternative tokens at this position.

<code>rules.prefix_rules[].decision</code>	<code>prompt / forbidden</code>	Required.
<code>rules.prefix_rules[].justification</code>	<code>string</code>	Optional non-empty rationale surfaced in approval prompts or rejection messages.

 TIP

You rarely need most of these keys. A practical starting point is to set only the approval policy, the sandbox mode and a few prefix rules for commands you run all the time, then add other settings when a specific need comes up. Re-check the official Codex configuration reference after each Codex update, because keys are added and renamed.

Gemini CLI settings.json Reference

This chapter lists every setting in the Gemini CLI's `settings.json`. It was generated on October 16, 2026 from `docs/reference/configuration.md` in the `google-gemini/gemini-cli` repository, and descriptions are shortened.

- **User:** `~/.gemini/settings.json` . **Project:** `.gemini/settings.json` . System and admin files override both.
- `general.defaultApprovalMode` accepts `"default"`, `"auto_edit"` or `"plan"` only. **YOLO can only be turned on from the command line** (`--yolo` or `--approval-mode=yolo`).
- `security.disableYoloMode` and `admin.secureModeEnabled` both block YOLO.

SETTING	TYPE	DEFAULT	WHAT IT DOES
<code>policyPaths</code>	array	<code>[]</code>	Additional policy files or directories to load.
<code>adminPolicyPaths</code>	array	<code>[]</code>	Additional admin policy files or directories to load.
<code>general.preferredEditor</code>	enum	<code>undefined</code>	The preferred editor to open files in.
<code>general.openEditorInNewWindow</code>	boolean	<code>false</code>	Open VS Code-family editors in a new window when editing files.
<code>general.vimMode</code>	boolean	<code>false</code>	Enable Vim keybindings.
<code>general.defaultApprovalMode</code>	enum	<code>"default"</code>	The default approval mode for tool execution. 'default' prompts for approval, 'auto_edit' auto-approves edit tools,...
<code>general.devtools</code>	boolean	<code>false</code>	Enable DevTools inspector on launch.
<code>general.enableAutoUpdate</code>	boolean	<code>true</code>	Enable automatic updates.
<code>general.enableAutoUpdateNotification</code>	boolean	<code>true</code>	Enable update notification prompts.
<code>general.enableNotifications</code>	boolean	<code>false</code>	Enable terminal run-event notifications for action-required prompts and session completion.
<code>general.notificationMethod</code>	enum	<code>"auto"</code>	How to send terminal notifications.

<code>general.checkpointing.enabled</code>	boolean	<code>false</code>	Enable session checkpointing for recovery.
<code>general.plan.enabled</code>	boolean	<code>true</code>	Enable Plan Mode for read-only safety during planning.
<code>general.plan.directory</code>	string	<code>undefined</code>	The directory where planning artifacts are stored.
<code>general.plan.modelRouting</code>	boolean	<code>true</code>	Automatically switch between Pro and Flash models based on Plan Mode status.
<code>general.retryFetchErrors</code>	boolean	<code>true</code>	Retry on "exception TypeError: fetch failed sending request" errors.
<code>general.maxAttempts</code>	number	<code>10</code>	Maximum number of attempts for requests to the main chat model.
<code>general.debugKeystrokeLogging</code>	boolean	<code>false</code>	Enable debug logging of keystrokes to the console.
<code>general.sessionRetention.enabled</code>	boolean	<code>true</code>	Enable automatic session cleanup.
<code>general.sessionRetention.maxAge</code>	string	<code>"30d"</code>	Automatically delete chats older than this time period (e.g., "30d", "7d", "24h", "1w").
<code>general.sessionRetention.maxCount</code>	number	<code>undefined</code>	Alternative: Maximum number of sessions to keep (most recent).
<code>general.sessionRetention.minRetention</code>	string	<code>"1d"</code>	Minimum retention period (safety limit, defaults to "1d").
<code>general.topicUpdateNarration</code>	boolean	<code>true</code>	Enable the Topic & Update communication model for reduced chattiness and structured progress reporting.
<code>general.logRagSnippets</code>	boolean	<code>false</code>	Log full Code Customization (RAG) retrieved snippets to a local file for debugging.
<code>output.format</code>	enum	<code>"text"</code>	The format of the CLI output.
<code>ui.debugRainbow</code>	boolean	<code>false</code>	Enable debug rainbow rendering.
<code>ui.theme</code>	string	<code>undefined</code>	The color theme for the UI. See the CLI themes guide for available options.
<code>ui.autoThemeSwitching</code>	boolean	<code>true</code>	Automatically switch between default light and dark themes based on terminal background color.
<code>ui.terminalBackgroundPollingInterval</code>	number	<code>60</code>	Interval in seconds to poll the terminal background color.
<code>ui.customThemes</code>	object	<code>{}</code>	Custom theme definitions.
<code>ui.hideWindowTitle</code>	boolean	<code>false</code>	Hide the window title bar.
<code>ui.inlineThinkingMode</code>	enum	<code>"off"</code>	Display model thinking inline: off or full.
<code>ui.showStatusInTitle</code>	boolean	<code>false</code>	Show Gemini CLI model thoughts in the terminal window title during the working phase.
<code>ui.dynamicWindowTitle</code>	boolean	<code>true</code>	Update the terminal window title with current status icons (Ready:  , Action Required:  , Working: ).
<code>ui.showHomeDirectoryWarning</code>	boolean	<code>true</code>	Show a warning when running Gemini CLI in the home directory.
<code>ui.showCompatibilityWarnings</code>	boolean	<code>true</code>	Show warnings about terminal or OS compatibility issues.
<code>ui.hideTips</code>	boolean	<code>false</code>	Hide helpful tips in the UI.

<code>ui.escapePastedAtSymbols</code>	boolean	<code>false</code>	When enabled, @ symbols in pasted text are escaped to prevent unintended @path expansion.
<code>ui.showShortcutsHint</code>	boolean	<code>true</code>	Show the “? for shortcuts” hint above the input.
<code>ui.compactToolOutput</code>	boolean	<code>true</code>	Display tool outputs (like directory listings and file reads) in a compact, structured format.
<code>ui.hideBanner</code>	boolean	<code>false</code>	Hide the application banner.
<code>ui.hideContextSummary</code>	boolean	<code>false</code>	Hide the context summary (GEMINI.md, MCP servers) above the input.
<code>ui.footer.items</code>	array	<code>undefined</code>	List of item IDs to display in the footer.
<code>ui.footer.showLabels</code>	boolean	<code>true</code>	Display a second line above the footer items with descriptive headers (e.g., /model).
<code>ui.footer.hideCWD</code>	boolean	<code>false</code>	Hide the current working directory in the footer.
<code>ui.footer.hideSandboxStatus</code>	boolean	<code>false</code>	Hide the sandbox status indicator in the footer.
<code>ui.footer.hideModelInfo</code>	boolean	<code>false</code>	Hide the model name and context usage in the footer.
<code>ui.footer.hideContextPercentage</code>	boolean	<code>true</code>	Hides the context window usage percentage.
<code>ui.hideFooter</code>	boolean	<code>false</code>	Hide the footer from the UI.
<code>ui.collapseDrawerDuringApproval</code>	boolean	<code>true</code>	Whether to collapse the UI drawer when a tool is awaiting confirmation.
<code>ui.showMemoryUsage</code>	boolean	<code>false</code>	Display memory usage information in the UI.
<code>ui.showLineNumbers</code>	boolean	<code>true</code>	Show line numbers in the chat.
<code>ui.showCitations</code>	boolean	<code>false</code>	Show citations for generated text in the chat.
<code>ui.showModelInfoInChat</code>	boolean	<code>false</code>	Show the model name in the chat for each model turn.
<code>ui.showUserIdentity</code>	boolean	<code>true</code>	Show the signed-in user’s identity (e.g. email) in the UI.
<code>ui.useAlternateBuffer</code>	boolean	<code>false</code>	Use an alternate screen buffer for the UI, preserving shell history.
<code>ui.renderProcess</code>	boolean	<code>true</code>	Enable Ink render process for the UI.
<code>ui.terminalBuffer</code>	boolean	<code>false</code>	Use the new terminal buffer architecture for rendering.
<code>ui.useBackgroundColor</code>	boolean	<code>true</code>	Whether to use background colors in the UI.
<code>ui.incrementalRendering</code>	boolean	<code>true</code>	Enable incremental rendering for the UI. This option will reduce flickering but may cause rendering artifacts.
<code>ui.showSpinner</code>	boolean	<code>true</code>	Show the spinner during operations.
<code>ui.loadingPhrases</code>	enum	<code>"off"</code>	What to show while the model is working: tips, witty comments, all, or off.
<code>ui.errorVerbosity</code>	enum	<code>"low"</code>	Controls whether recoverable errors are hidden (low) or fully shown (full).
<code>ui.customWittyPhrases</code>	array	<code>[]</code>	Custom witty phrases to display during loading.
<code>ui.accessibility.enableLoadingPhrases</code>	boolean	<code>true</code>	@deprecated Use ui.loadingPhrases instead.

<code>ui.accessibility.screenReader</code>	boolean	<code>false</code>	Render output in plain-text to be more screen reader accessible.
<code>ide.enabled</code>	boolean	<code>false</code>	Enable IDE integration mode.
<code>ide.hasSeenNudge</code>	boolean	<code>false</code>	Whether the user has seen the IDE integration nudge.
<code>privacy.usageStatisticsEnabled</code>	boolean	<code>true</code>	Enable collection of usage statistics.
<code>billing.overageStrategy</code>	enum	<code>"ask"</code>	How to handle quota exhaustion when AI credits are available. 'ask' prompts each time, 'always' automatically uses...
<code>billing.vertexAi.requestType</code>	enum	<code>undefined</code>	Sets the X-Vertex-AI-LLM-Request-Type header for Vertex AI requests.
<code>billing.vertexAi.sharedRequestType</code>	enum	<code>undefined</code>	Sets the X-Vertex-AI-LLM-Shared-Request-Type header for Vertex AI requests.
<code>model.name</code>	string	<code>undefined</code>	The Gemini model to use for conversations.
<code>model.maxSessionTurns</code>	number	<code>-1</code>	Maximum number of user/model/tool turns to keep in a session. -1 means unlimited.
<code>model.summarizeToolOutput</code>	object	<code>undefined</code>	Enables or disables summarization of tool output.
<code>model.compressionThreshold</code>	number	<code>0.5</code>	The fraction of context usage at which to trigger context compression (e.g. 0.2, 0.3).
<code>model.disableLoopDetection</code>	boolean	<code>false</code>	Disable automatic detection and prevention of infinite loops.
<code>model.skipNextSpeakerCheck</code>	boolean	<code>true</code>	Skip the next speaker check.
<code>modelConfigs.aliases</code>	object		Named presets for model configs.
<code>modelConfigs.customAliases</code>	object	<code>{}</code>	Custom named presets for model configs.
<code>modelConfigs.customOverrides</code>	array	<code>[]</code>	Custom model config overrides.
<code>modelConfigs.overrides</code>	array	<code>[]</code>	Apply specific configuration overrides based on matches, with a primary key of model (or alias).
<code>modelConfigs.modelDefinitions</code>	object		Registry of model metadata, including tier, family, and features.
<code>modelConfigs.modelIdResolutions</code>	object		Rules for resolving requested model names to concrete model IDs based on context.
<code>modelConfigs.classifierIdResolutions</code>	object		Rules for resolving classifier tiers (flash, pro) to concrete model IDs.
<code>modelConfigs.modelChains</code>	object		Availability policy chains defining fallback behavior for models.
<code>agents.overrides</code>	object	<code>{}</code>	Override settings for specific agents, e.g. to disable the agent, set a custom model config, or run config.
<code>agents.browser.sessionMode</code>	enum	<code>"persistent"</code>	Session mode: 'persistent', 'isolated', or 'existing'.
<code>agents.browser.headless</code>	boolean	<code>false</code>	Run browser in headless mode.
<code>agents.browser.profilePath</code>	string	<code>undefined</code>	Path to browser profile directory for session persistence.

<code>agents.browser.visualModel</code>	string	<code>undefined</code>	Model for the visual agent's <code>analyze_screenshot</code> tool.
<code>agents.browser.allowedDomains</code>	array		A list of allowed domains for the browser agent (e.g., <code>["github.com", "*.google.com"]</code>).
<code>agents.browser.disableUserInput</code>	boolean	<code>true</code>	Disable user input on browser window during automation.
<code>agents.browser.maxActionsPerTask</code>	number	<code>100</code>	The maximum number of tool calls allowed per browser task.
<code>agents.browser.confirmSensitiveActions</code>	boolean	<code>false</code>	Require manual confirmation for sensitive browser actions (e.g., <code>fill_form</code> , <code>evaluate_script</code>).
<code>agents.browser.blockFileUploads</code>	boolean	<code>false</code>	Hard-block file upload requests from the browser agent.
<code>context.fileName</code>	string	<code>string[]</code>	<code>undefined</code>
<code>context.importFormat</code>	string	<code>undefined</code>	The format to use when importing memory.
<code>context.includeDirectoryTree</code>	boolean	<code>true</code>	Whether to include the directory tree of the current working directory in the initial request to the model.
<code>context.discoveryMaxDirs</code>	number	<code>200</code>	Maximum number of directories to search for memory.
<code>context.memoryBoundaryMarkers</code>	array		File or directory names that mark the boundary for GEMINI.md discovery.
<code>context.includeDirectories</code>	array	<code>[]</code>	Additional directories to include in the workspace context.
<code>context.loadMemoryFromIncludeDirectories</code>	boolean	<code>false</code>	Controls how <code>/memory reload</code> loads GEMINI.md files.
<code>context.fileFiltering.respectGitIgnore</code>	boolean	<code>true</code>	Respect <code>.gitignore</code> files when searching.
<code>context.fileFiltering.respectGeminiIgnore</code>	boolean	<code>true</code>	Respect <code>.geminiignore</code> files when searching.
<code>context.fileFiltering.enableFileWatcher</code>	boolean	<code>false</code>	Enable file watcher updates for <code>@</code> file suggestions (experimental).
<code>context.fileFiltering.enableRecursiveFileSearch</code>	boolean	<code>true</code>	Enable recursive file search functionality when completing <code>@</code> references in the prompt.
<code>context.fileFiltering.enableFuzzySearch</code>	boolean	<code>true</code>	Enable fuzzy search when searching for files.
<code>context.fileFiltering.customIgnoreFilePaths</code>	array	<code>[]</code>	Additional ignore file paths to respect.
<code>tools.sandbox</code>	string	<code>undefined</code>	Legacy full-process sandbox execution environment.
<code>tools.sandboxAllowedPaths</code>	array	<code>[]</code>	List of additional paths that the sandbox is allowed to access.
<code>tools.sandboxNetworkAccess</code>	boolean	<code>false</code>	Whether the sandbox is allowed to access the network.

<code>tools.shell.enableInteractiveShell</code>	boolean	<code>true</code>	Use node-pty for an interactive shell experience.
<code>tools.shell.backgroundCompletionBehavior</code>	enum	<code>"silent"</code>	Controls what happens when a background shell command finishes. 'silent' (default): quietly exits in background....
<code>tools.shell.pager</code>	string	<code>"cat"</code>	The pager command to use for shell output.
<code>tools.shell.showColor</code>	boolean	<code>true</code>	Show color in shell output.
<code>tools.shell.inactivityTimeout</code>	number	<code>300</code>	The maximum time in seconds allowed without output from the shell command.
<code>tools.shell.enableShellOutputEfficiency</code>	boolean	<code>true</code>	Enable shell output efficiency optimizations for better performance.
<code>tools.core</code>	array	<code>undefined</code>	Restrict the set of built-in tools with an allowlist.
<code>tools.allowed</code>	array	<code>undefined</code>	Tool names that bypass the confirmation dialog.
<code>tools.confirmationRequired</code>	array	<code>undefined</code>	Tool names that always require user confirmation.
<code>tools.exclude</code>	array	<code>undefined</code>	Tool names to exclude from discovery.
<code>tools.discoveryCommand</code>	string	<code>undefined</code>	Command to run for tool discovery.
<code>tools.callCommand</code>	string	<code>undefined</code>	Defines a custom shell command for invoking discovered tools.
<code>tools.useRipgrep</code>	boolean	<code>true</code>	Use ripgrep for file content search instead of the fallback implementation.
<code>tools.truncateToolOutputThreshold</code>	number	<code>40000</code>	Maximum characters to show when truncating large tool outputs.
<code>tools.disableLLMCorrection</code>	boolean	<code>true</code>	Disable LLM-based error correction for edit tools.
<code>mcp.serverCommand</code>	string	<code>undefined</code>	Command to start an MCP server.
<code>mcp.allowed</code>	array	<code>undefined</code>	A list of MCP servers to allow.
<code>mcp.excluded</code>	array	<code>undefined</code>	A list of MCP servers to exclude.
<code>useWriteTodos</code>	boolean	<code>true</code>	Enable the write_todos tool.
<code>security.toolSandboxing</code>	boolean	<code>false</code>	Tool-level sandboxing.
<code>security.disableYoloMode</code>	boolean	<code>false</code>	Disable YOLO mode, even if enabled by a flag.
<code>security.disableAlwaysAllow</code>	boolean	<code>false</code>	Disable "Always allow" options in tool confirmation dialogs.
<code>security.enablePermanentToolApproval</code>	boolean	<code>false</code>	Enable the "Allow for all future sessions" option in tool confirmation dialogs.
<code>security.autoAddToPolicyByDefault</code>	boolean	<code>false</code>	When enabled, the "Allow for all future sessions" option becomes the default choice for low-risk tools in trusted...

<code>security.blockGitExtensions</code>	boolean	<code>false</code>	Blocks installing and loading extensions from Git.
<code>security.allowedExtensions</code>	array	<code>[]</code>	List of Regex patterns for allowed extensions.
<code>security.folderTrust.enabled</code>	boolean	<code>true</code>	Setting to track whether Folder trust is enabled.
<code>security.environmentVariableRedaction.allowed</code>	array	<code>[]</code>	Environment variables to always allow (bypass redaction).
<code>security.environmentVariableRedaction.blocked</code>	array	<code>[]</code>	Environment variables to always redact.
<code>security.environmentVariableRedaction.enabled</code>	boolean	<code>false</code>	Enable redaction of environment variables that may contain secrets.
<code>security.auth.selectedType</code>	string	<code>undefined</code>	The currently selected authentication type.
<code>security.auth.enforcedType</code>	string	<code>undefined</code>	The required auth type.
<code>security.auth.useExternal</code>	boolean	<code>undefined</code>	Whether to use an external authentication flow.
<code>security.enableConseca</code>	boolean	<code>false</code>	Enable the context-aware security checker.
<code>advanced.autoConfigureMemory</code>	boolean	<code>true</code>	Automatically configure Node.js memory limits.
<code>advanced.dnsResolutionOrder</code>	string	<code>undefined</code>	The DNS resolution order.
<code>advanced.excludedEnvVars</code>	array		Environment variables to exclude from project context.
<code>advanced.ignoreLocalEnv</code>	boolean	<code>false</code>	Whether to ignore generic .env files in the project directory.
<code>advanced.bugCommand</code>	object	<code>undefined</code>	Configuration for the bug report command.
<code>experimental.gemma</code>	boolean	<code>true</code>	Enable access to Gemma 4 models via Gemini API.
<code>experimental.voiceMode</code>	boolean	<code>false</code>	Enable experimental voice dictation and commands (/voice, /voice model).
<code>experimental.voice.activationMode</code>	enum	<code>"push-to-talk"</code>	How to trigger voice recording with the Space key.
<code>experimental.voice.backend</code>	enum	<code>"gemini-live"</code>	The backend to use for voice transcription.
<code>experimental.voice.whisperModel</code>	enum	<code>"ggml-base.en.bin"</code>	The Whisper model to use for local transcription.
<code>experimental.voice.stopGracePeriodMs</code>	number	<code>4000</code>	How long to wait for final transcription after stopping recording.

<code>experimental.adk.agentSessionNoninteractiveEnabled</code>	boolean	false	Enable non-interactive agent sessions.
<code>experimental.adk.agentSessionInteractiveEnabled</code>	boolean	false	Enable the agent session implementation for the interactive CLI.
<code>experimental.adk.agentSessionSubagentEnabled</code>	boolean	false	Route subagent invocations through the AgentSession protocol instead of legacy executors.
<code>experimental.enableAgents</code>	boolean	true	Enable local and remote subagents.
<code>experimental.worktrees</code>	boolean	false	Enable automated Git worktree management for parallel work.
<code>experimental.extensionManagement</code>	boolean	true	Enable extension management features.
<code>experimental.extensionConfig</code>	boolean	true	Enable requesting and fetching of extension settings.
<code>experimental.extensionRegistry</code>	boolean	false	Enable extension registry explore UI.
<code>experimental.extensionRegistryURI</code>	string		The URI (web URL or local file path) of the extension registry.
<code>experimental.extensionReloading</code>	boolean	false	Enables extension loading/unloading within the CLI session.
<code>experimental.useOSC52Paste</code>	boolean	false	Use OSC 52 for pasting.
<code>experimental.useOSC52Copy</code>	boolean	false	Use OSC 52 for copying.
<code>experimental.taskTracker</code>	boolean	false	Enable task tracker tools.
<code>experimental.modelSteering</code>	boolean	false	Enable model steering (user hints) to guide the model during tool execution.
<code>experimental.directWebFetch</code>	boolean	false	Enable web fetch behavior that bypasses LLM summarization.
<code>experimental.dynamicModelConfiguration</code>	boolean	false	Enable dynamic model configuration (definitions, resolutions, and chains) via settings.
<code>experimental.gemmaModelRouter.enabled</code>	boolean	false	Enable the Gemma Model Router (experimental).
<code>experimental.gemmaModelRouter.autoStartServer</code>	boolean	false	Automatically start the LiteRT-LM server when Gemini CLI starts and the Gemma router is enabled.
<code>experimental.gemmaModelRouter.binaryPath</code>	string	""	Custom path to the LiteRT-LM binary.
<code>experimental.gemmaModelRouter.classifier.host</code>	string		The host of the classifier.
<code>experimental.gemmaModelRouter.classifier.model</code>	string		The model to use for the classifier.
<code>experimental.stressTestProfile</code>	boolean	false	Significantly lowers token limits to force early garbage collection and distillation for testing purposes.
<code>experimental.autoMemory</code>	boolean	false	Automatically extract memory patches and skills from past sessions in the background.
<code>experimental.generalistProfile</code>	boolean	false	Suitable for general coding and software development tasks.

<code>experimental.powerUserProfile</code>	boolean	<code>false</code>	Less cache friendly version of the generalist profile.
<code>experimental.contextManagement</code>	boolean	<code>false</code>	Enable logic for context management.
<code>experimental.topicUpdateNarration</code>	boolean	<code>false</code>	Deprecated: Use <code>general.topicUpdateNarration</code> instead.
<code>skills.enabled</code>	boolean	<code>true</code>	Enable Agent Skills.
<code>skills.disabled</code>	array	<code>[]</code>	List of disabled skills.
<code>hooksConfig.enabled</code>	boolean	<code>true</code>	Canonical toggle for the hooks system.
<code>hooksConfig.disabled</code>	array	<code>[]</code>	List of hook names (commands) that should be disabled.
<code>hooksConfig.notifications</code>	boolean	<code>true</code>	Show visual indicators when hooks are executing.
<code>hooks.BeforeTool</code>	array	<code>[]</code>	Hooks that execute before tool execution.
<code>hooks.AfterTool</code>	array	<code>[]</code>	Hooks that execute after tool execution.
<code>hooks.BeforeAgent</code>	array	<code>[]</code>	Hooks that execute before agent loop starts.
<code>hooks.AfterAgent</code>	array	<code>[]</code>	Hooks that execute after agent loop completes.
<code>hooks.Notification</code>	array	<code>[]</code>	Hooks that execute on notification events (errors, warnings, info).
<code>hooks.SessionStart</code>	array	<code>[]</code>	Hooks that execute when a session starts.
<code>hooks.SessionEnd</code>	array	<code>[]</code>	Hooks that execute when a session ends.
<code>hooks.PreCompress</code>	array	<code>[]</code>	Hooks that execute before chat history compression.
<code>hooks.BeforeModel</code>	array	<code>[]</code>	Hooks that execute before LLM requests.
<code>hooks.AfterModel</code>	array	<code>[]</code>	Hooks that execute after LLM responses.
<code>hooks.BeforeToolSelection</code>	array	<code>[]</code>	Hooks that execute before tool selection.
<code>contextManagement.historyWindow.maxTokens</code>	number	<code>150000</code>	The number of tokens to allow before triggering compression.
<code>contextManagement.historyWindow.retainedTokens</code>	number	<code>40000</code>	The number of tokens to always retain.
<code>contextManagement.messageLimits.normalMaxTokens</code>	number	<code>2500</code>	The target number of tokens to budget for a normal conversation turn.

<code>contextManagement.messageLimits.retainedMaxTokens</code>	number	12000	The maximum number of tokens a single conversation turn can consume before truncation.
<code>contextManagement.messageLimits.normalizationHeadRatio</code>	number	0.25	The ratio of tokens to retain from the beginning of a truncated message (0.0 to 1.0).
<code>contextManagement.tools.distillation.maxOutputTokens</code>	number	10000	Maximum tokens to show to the model when truncating large tool outputs.
<code>contextManagement.tools.outputMasking.protectLatestTurn</code>	boolean	true	Ensures the absolute latest turn is never masked, regardless of token count.
<code>admin.secureModeEnabled</code>	boolean	false	If true, disallows YOLO mode and "Always allow" options from being used.
<code>admin.extensions.enabled</code>	boolean	true	If false, disallows extensions from being installed or used.
<code>admin.mcp.enabled</code>	boolean	true	If false, disallows MCP servers from being used.
<code>admin.mcp.config</code>	object	{}	Admin-configured MCP servers (allowlist).
<code>admin.mcp.requiredConfig</code>	object	{}	Admin-required MCP servers that are always injected.
<code>admin.skills.enabled</code>	boolean	true	If false, disallows agent skills from being used.
<code>GOOGLE_APPLICATION_CREDENTIALS</code>	string		The path to your Google Application Credentials JSON file.
<code>SEATBELT_PROFILE</code>	macOS specific	.	

Your Autonomy Setup Checklist

- ☐ Agents run in a dev container, VM, cloud box, or separate user account
- ☐ Separate GitHub account or app for agents; fine-grained token with full repo permissions
- ☐ Hard spend limits set on every AI and cloud account
- ☐ Everything in Git; `main` protected by status checks; auto-merge allowed
- ☐ `~/.claude/settings.json`: `defaultMode` set to `auto` or `bypassPermissions`, plus `skipDangerousModePermissionPrompt`
- ☐ Claude Desktop: **Allow bypass permissions mode** on; Cowork: **Automatically approve**
- ☐ Every connector tool set to **Always allow**
- ☐ Cursor: **Run Everything** (or Auto-review with allow instructions)
- ☐ VS Code: `chat.tools.global.autoApprove` on, or Autopilot
- ☐ Codex `config.toml`: `approval_policy = "never"`, `sandbox_mode = "danger-full-access"`; Gemini alias with `--yolo`
- ☐ n8n: API key, instance-level MCP on, workflows enabled for MCP; `NODES_EXCLUDE="[]"` if self-hosted
- ☐ Notion: MCP connected; API connection with all capabilities and top-level pages shared
- ☐ Unattended runs have `--max-turns` and `--max-budget-usd` limits

Glossary

Allowlist	A list of pre-approved actions an agent may take without asking.
Auto mode	Claude Code's permission mode in which a classifier reviews actions instead of a person.
Bypass permissions	A mode that skips approval prompts entirely (<code>--dangerously-skip-permissions</code> in Claude Code).
Classifier	A second AI model that judges whether an agent's action is safe and matches the request.
Containment	Limiting what an agent can reach (a container, separate accounts, budgets) instead of limiting what it may do.
Critical path	A folder, such as the root, your home folder, or the working directory, that Claude Code protects from deletion in every mode.
Dev container	A Docker container configured as a development environment, used to isolate agents.
Fine-grained token	A GitHub access token limited to chosen repositories and permissions.
Headless / print mode	Running an agent non-interactively from a script, such as <code>claude -p</code> .
Prompt injection	Instructions hidden in content an agent reads, meant to hijack its behavior.
Ruleset	GitHub settings that protect branches, with an optional bypass list.
Sandbox	An isolated environment limiting the files and network a command can reach.
Sudo mode	GitHub's re-authentication step for sensitive actions in the web interface.
YOLO mode	Informal name for fully automatic approval in tools like Gemini CLI, Codex, and Copilot CLI.