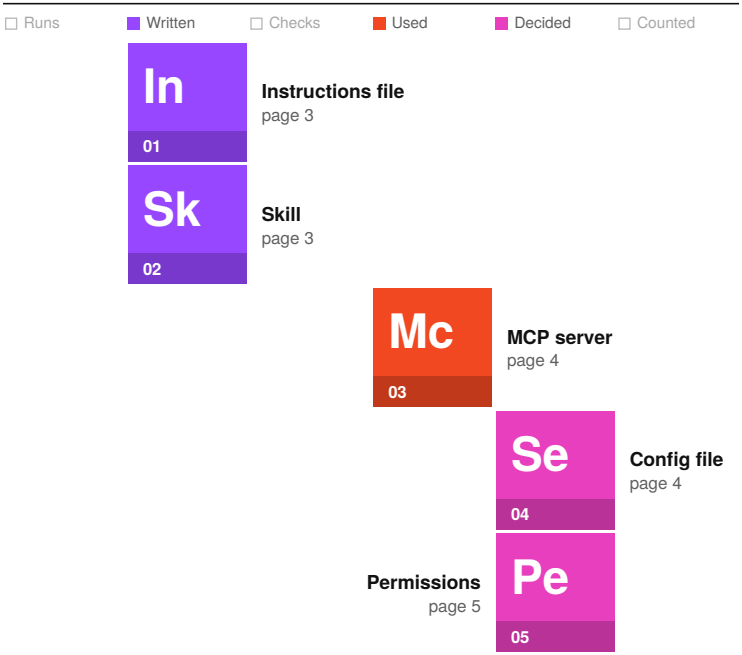
OpenAI Codex

A pocket-sized guide to where Codex keeps its setup, and which copy wins, as of October 2026.



My friend has a lot of questions.

What can you set up in Codex?

You can give Codex standing instructions, skills, MCP servers, settings and permissions. Each one is a plain file, and the Codex CLI, the IDE extension and the ChatGPT desktop app all read the same files.

Most of these files live in your Codex home folder, `~/.codex` unless `CODEX_HOME` says otherwise, or in your project. An Accidental Guide to: Agents, Harnesses, Skills, and Tools explains the basics.

Instructions file	Guidance Codex reads at the start of every session.
Skill	Steps for one kind of task, read when the task comes up.
MCP server	A program that gives Codex tools for another system.
Config file	The file that holds settings such as the model and MCP servers.
Permissions	What commands may change, and when Codex stops to ask you.

Instructions file

In
01

Identify by `AGENTS.md`, in `~/.codex/` or anywhere in a project.

What **Codex reads your `AGENTS.md`, then each one from the project's root down to where you work.** An Accidental Guide to: Agents, Harnesses, Skills, and Tools explains them.

So what **Put guidance in the folder it's about.** The file closest to your work has the last word.

Field marks

Loads	Add with
At the start of a session.	`/init`.

Confused with Sk **Skill**
Codex reads an instructions file every session. A skill's steps are read only when the skill is used.

Skill

Sk
02

Identify by A folder with a `SKILL.md` file.

What **Codex reads the rest of a skill when your task matches its description, or when you type $ and its name.** An Accidental Guide to: Agent Skills explains skills.

So what A project's skills in `.agents/skills/` reach everyone who works on it.

Field marks

Loads	Check with
Description first.	`/skills`.

Confused with **Plugin**
A plugin packages skills, MCP servers and hooks to install together, using `/plugins` in the CLI.

MCP server

Mc
03

Identify by A `[mcp_servers.docs]` table in `config.toml`.

What **Each server's entry gives a command that starts it on your computer, or the url of a remote one.** An Accidental Guide to: Model Context Protocol explains MCP servers.

So what Set a server up once and it works in the CLI, the IDE extension and the desktop app.

Field marks

Add with	Check with
`codex mcp add`.	`/mcp`.

Confused with **Tool**
The MCP server is the program. The tools it offers are what Codex calls.

Config file

Se
04

Identify by `config.toml`, in `~/.codex/` or in a project's `.codex/` folder.

What **Settings are the choices that control how Codex runs, such as the model.** Codex keeps them in `config.toml`, along with MCP servers and permissions.

So what Your own `config.toml` applies everywhere, and a project's `.codex/config.toml` changes settings for that project.

Field marks

Loads	Override with
When Codex starts.	`-c` **for one run.**

Confused with **Requirements file**
Your `config.toml` sets defaults you can change. An administrator's `requirements.toml` sets limits you can't.

Permissions

Pe
05

Identify by `sandbox_mode` and `approval_policy` in `config.toml`.

What **Codex has two permission settings.** The sandbox decides what commands can touch: read-only, workspace-write or danger-full-access. The approval policy, on-request or never, decides when Codex stops to ask. An Accidental Guide to: Agents, Harnesses, Skills, and Tools explains permissions.

So what **The sandbox keeps a wrong command away from everything outside your work.** In a Git folder Codex starts with workspace-write and on-request, and elsewhere read-only.

Field marks

Network	Change with
Off by default.	`/permissions`.

Confused with **Rules**

Rules files in `~/.codex/rules/` say whether a command may run outside the sandbox: allow, prompt or forbidden.

Where things live

Your own files apply in every project. A project's files are kept in its repository.

Your home folder

In **Instructions**	`~/.codex/AGENTS.md`, or `AGENTS.override.md` beside it
Sk **Skills**	`~/.agents/skills/`
Se **Config file**	`~/.codex/config.toml`, which also holds MCP servers and permissions, with profiles beside it
Hooks, rules, agents	`~/.codex/hooks.json`, `~/.codex/rules/`, `~/.codex/agents/`

This project

In **Instructions**	`AGENTS.md` in any folder
Sk **Skills**	`.agents/skills/` in any folder
Se **Config file**	`.codex/config.toml`
Hooks, rules, agents	`.codex/hooks.json`, `.codex/rules/`, `.codex/agents/`

Your organization

Se **Config file**	`/etc/codex/config.toml`, or defaults from the admin console
Sk **Skills**	`/etc/codex/skills/`
Requirements	`/etc/codex/requirements.toml`, or the admin console

Codex Cloud

Sk **Skills**	Only the repository's skills. Personal skills aren't synced.

Note. Codex reads a project's `.codex` folder only after you trust the project, so a repository you haven't trusted can't change your settings, hooks or rules.

How do the places combine?

Config file	Where the value comes from, strongest first
1	A flag or `-c` on the command line
2	The project's `.codex/config.toml` closest to where you work
3	A profile picked with `--profile`
4	Your `~/.codex/config.toml`
5	Your organization's defaults
6	Codex's built-in defaults

Other setup	When two places set it
In **Instructions**	Codex joins them, yours first, then the project's from the root down, so the closest file has the last word.
Sk **Skills**	Codex offers all of them. Two with the same name both appear.
Mc **MCP servers**	Servers can be in your `config.toml` or a project's, and a setting for the same server follows the numbered list.
Hooks	Codex keeps all of them.
Rules	The strictest match wins: forbidden, then prompt, then allow.

Same thing elsewhere

Codex	Claude Code	Claude Desktop	Copilot CLI	VS Code and Visual Studio
In **Instructions file**	`CLAUDE.md`	Instructions	`AGENTS.md`	`AGENTS.md`
Sk **Skill**	Skills	Skills	Skills	Skills
Mc **MCP server**	`.mcp.json`	Connectors	`.mcp.json`	`.mcp.json`
Se **Config file**	`settings.json`	Settings	`settings.json`	Settings
Pe **Permissions**	`settings.json`	Settings	Settings	Settings

Each of these tools has its own guide: An Accidental Guide to: Claude Code, An Accidental Guide to: Claude Desktop, An Accidental Guide to: GitHub Copilot CLI, and An Accidental Guide to: VS Code and Visual Studio.

Sources

These public sources were checked on 22 October 2026.

Config basics
OpenAI · learn.chatgpt.com/docs/config-file/config-basic

Advanced Configuration
OpenAI · learn.chatgpt.com/docs/config-file/config-advanced

Custom instructions with AGENTS.md
OpenAI · learn.chatgpt.com/docs/agent-configuration/agents-md

Build skills
OpenAI · learn.chatgpt.com/docs/build-skills

Model Context Protocol
OpenAI · learn.chatgpt.com/docs/extend/mcp

Agent approvals and security
OpenAI · learn.chatgpt.com/docs/agent-approvals-security

Package your plugin
OpenAI · developers.openai.com/plugins/build/plugins

Hooks
OpenAI · learn.chatgpt.com/docs/hooks

Rules
OpenAI · learn.chatgpt.com/docs/agent-configuration/rules

Cloud environments
OpenAI · learn.chatgpt.com/docs/environments/cloud-environments

Managed configuration
OpenAI · learn.chatgpt.com/docs/enterprise/managed-configuration

In Sk Mc Se Pe

An Accidental Guide to…
OpenAI Codex
A pocket-sized guide to where Codex keeps its setup, and which copy wins, as of October 2026.
Written and reviewed by a person. AI does the grunt work. Set in TeX Gyre Heros and Newsreader.
Leave a star: accidental.info/star

My friend has a lot of questions.