KINGY.AI / ILLUSTRATED FIELD GUIDE

Codex field handbook

Five revised companions in one printable reference.

COMPANION EDITION / OCTOBER 16, 2026

CONTENTS

Quick start 02

Your first local task, result checks and status key.

Advanced workflows 03

Worktrees, branches, Handoff and delivery checks.

Troubleshooting 04

Nine diagnostic cases, a prompt and careful retries.

Prompt cookbook 05-08

Twelve reusable recipes and a focused revision pattern.

Shortcuts and commands 09-12

Desktop, CLI and VS Code references.

Use it at your desk or on paper

Start with Quick start, or choose the section you need. The contents, bookmarks and companion links navigate within this PDF. Source links open online.

Print: choose Fit to printable area on US Letter or A4 paper.

Compiled from the five revised Kingy.ai companions. Their source links and checked dates are retained. Prompts are examples to adapt to your task.

Read the full illustrated guide on Kingy.ai

Codex quick start

One small task. A clear result you can inspect.

01 / START WITH A PRACTICE FOLDER

Open

Choose an empty practice folder as a local project.

Understand

Ask Codex to explain the folder and confirm its path without editing.

Make

Use the starter prompt below to create one small file.

Inspect

Open the saved file. Request a focused revision if needed.

02 / SEND THIS STARTER PROMPT

```
Create practice-note.txt in this practice folder.
Add the title 'My first Codex task' and three short bullets.
Work only in this folder. Show me the saved file and its path.
Check the filename, title and bullet count before finishing.
```

03 / CHECK THE RESULT BEFORE USING IT

File
Open it; confirm the path and requested content.

Code
Inspect the diff and checks actually run.

Research
Open the sources supporting the claims.

04 / READ THE SYMBOL AND ITS LABEL TOGETHER

Unread
Activity you have not marked as read.

Needs input
An approval, answer or decision is needed.

Ready (pet)
Completed, with unread activity to inspect.

Blocked (pet)
The chat failed or hit a system error.

Open PR
The pull request is still open.

Merged PR
Integrated into its target branch; check deployment separately.

Blue, green and purple are this reference's illustration key. Actual appearance varies by client and release. Read the object, label and tooltip; unread activity, task status and Git status are different.

If something goes wrong

Check the folder and pending input before retrying. Use Troubleshooting below for errors, or Prompts for your next task.

Read the full illustrated guide on Kingy.ai

Sources: OpenAI Prompting Notifications

Checked October 16, 2026

Codex advanced workflows

Isolate work. Integrate carefully. Verify the result.

01 / RUN A COORDINATED CHANGE

1 Define

Name the outcome, allowed files, acceptance checks and work to preserve.

2 Isolate

Use a worktree per independent editing stream; confirm the starting branch and setup.

3 Delegate

Give each lane an owner, scope and concise return. Parallel agents use more tokens.

4 Integrate

Bring changes together once. Resolve conflicts and test the combined result.

5 Review

Compare with the intended base. Check findings against the latest diff and tests.

6 Deliver

Choose the requested delivery step and inspect its evidence. Verify the live result after deployment.

02 / KEEP WORKTREES USABLE AND BRING WORK BACK

```
Confirm the starting ref, checkout path and existing edits.
Run setup in the worktree; ignored local files may be absent.
In an app-managed worktree, use Create branch here to keep work.
Use Hand off > Local to continue in your regular checkout.
A branch can be checked out in only one location at a time.
After a move, confirm the path and files, then rerun checks.
Publish or deploy only when that step is part of your task.
```

Copyable task briefs are in the Prompt cookbook linked below.

03 / MAKE REPEAT WORK PREDICTABLE

Package a focused skill

Put repository rules in AGENTS.md. Use a skill for one repeatable job; test its inputs, outputs and failure handling.

Schedule a bounded check

Set cadence, time zone, sources and notification rules. Try the prompt manually. For local files, keep the computer on and app running.

04 / KEEP THE DELIVERY STATES SEPARATE

Diff reviewed

Committed locally

Pushed to remote

PR opened

Merged to target

Live result verified

Worktrees isolate checkouts; subagents divide work. Check tools and permissions for each run.

Read the full illustrated guide on Kingy.ai

OpenAI: Worktrees Agents Review Skills Scheduling Checked Oct 16, 2026

Companions: Quick start Advanced Troubleshooting Prompts Shortcuts

Codex troubleshooting

Find the failed step. Check the state. Fix the cause.

01 / START WITH THE SYMPTOM

Stuck or waiting

Check Activity and the chat for a pending question, approval or tool run. Identify the active step before restarting.

Wrong files or folder

Confirm the project folder, worktree and execution host. Check that the needed files exist in that environment.

Access denied

Read the exact file, network or service error. Use the narrowest permission or connection that supports the task.

Command or setup fails

Capture the command, directory and error. Check dependencies and worktree setup; ignored files may need copying.

The answer is wrong

Point to the requirement, source or version it missed. Inspect the output, then request a focused correction and check.

Unfamiliar changes

Git review can include other edits. Select Last turn to inspect the latest assistant turn; preserve unrelated work.

Sign-in or connection

Confirm the account, workspace and sign-in method. External tools have their own connections. Use the sign-in flow; keep credentials out of prompts.

Usage limit

Read the limit message and check the usage view for reset times or available options. Wait if needed; repeated retries do not restore an allowance.

Missing feature

Check the client, version, model and workspace policy. App, CLI and editor releases can differ; a shortcut cannot enable an unavailable feature.

02 / ADAPT THIS DIAGNOSTIC PROMPT

```
Diagnose [symptom] before making changes.
Confirm the project path, worktree, host and exact error.
Check which files or external actions already completed.
Suggest the smallest fix and how to verify the result.
Avoid repeated writes; separate findings from assumptions.
```

Authored example. Replace the bracketed field before sending.

03 / RECOVER WITH A USEFUL RECORD

Before retrying

Inspect saved files and the external result. Determine whether a previous write completed, then repeat only the action still needed.

If you need to report a bug

Include expected vs actual behavior, client and OS versions, path, host, exact error and checks tried. Review logs for private data before sharing.

A fix is complete when the original task works and the result has been checked.

Read the full illustrated guide on Kingy.ai

Codex prompt cookbook

Explore and create

OUTCOME + CONTEXT + BOUNDARIES + A RESULT YOU CAN CHECK

Attach the relevant material. Replace every bracketed field before sending.

01 Understand an unfamiliar project

Get your bearings before choosing what to change.

Explore [project folder] and explain how [feature or request] works. Identify the entry points, key files, data flow and relevant checks, with file references I can follow. Read the project instructions. Do not edit files yet. Flag anything you could not inspect and finish with the next useful investigation.

Check: Follow two file references and confirm the described flow.

02 Research a decision

Compare current options using evidence you can revisit.

Research [decision] for [audience and use case]. Compare [options] on [criteria] within [budget or constraints]. Use current primary sources and link the facts that affect the decision. Separate published facts from your analysis. Deliver a short recommendation, a comparison table and unresolved questions. Do not purchase or contact vendors.

Check: Open the decisive sources and check dates, terms and assumptions.

03 Create a finished document

Turn supplied material into a file for a specific reader.

Use [source files] to create [document type] for [audience]. Lead with [main decision or message], keep it to [length], and match [style reference]. Preserve the supplied figures and cite their sources. Flag gaps instead of inventing details. Save the finished file in [folder], inspect its rendered layout and report any unresolved problems.

Check: Check the source figures and inspect the finished file.

Original Kingy.ai examples. Adapt the scope; tool access and permissions still apply.

Read the full illustrated guide on Kingy.ai

OpenAI prompting Codex best practices Checked Oct 16, 2026

Codex prompt cookbook

Build and repair

OUTCOME + CONTEXT + BOUNDARIES + A RESULT YOU CAN CHECK

Name the behavior, reproduction steps and constraints that matter.

04 Fix a reproducible bug

You can describe what fails and what should happen.

Fix [bug] in [project]. Reproduce it using [steps]; expected behavior is [expected], but I see [actual]. Inspect the relevant code and explain the cause briefly. Make a focused fix while preserving [API or behavior constraint]. Add a regression test if useful, rerun the reproduction and relevant checks, and report commands, results and remaining gaps.

Check: Repeat the original reproduction after the fix.

05 Implement a scoped feature

You know the intended behavior and acceptance checks.

Implement [feature] in [project or route]. It should [user-visible behavior] and handle [important edge cases]. Follow existing patterns and preserve [contracts or files]. Done means [acceptance checks]. Make reasonable implementation choices, run the relevant checks and show the working result. Report what changed and what you could not verify. Keep deployment outside this task.

Check: Exercise the main path and the named edge cases.

06 Refactor without changing behavior

Improve structure while keeping the public contract stable.

Refactor [module] to [specific structural goal]. Preserve [public API, outputs and compatibility]. Read the callers and existing tests before editing. Keep unrelated work intact and avoid broad cleanup. Verify the preserved behavior with the relevant checks, inspect the final diff and explain any compatibility risk or verification gap.

Check: Compare the public contract and behavior before and after.

Original Kingy.ai examples. Adapt the scope; tool access and permissions still apply.

Read the full illustrated guide on Kingy.ai

OpenAI prompting Codex best practices Checked Oct 16, 2026

Codex prompt cookbook

Check and improve

OUTCOME + CONTEXT + BOUNDARIES + A RESULT YOU CAN CHECK

Choose a comparison or success condition before asking for a review.

07 Review a change for real defects

Inspect a diff before you accept or share it.

Review [working changes, commit or PR] against [base or intended behavior]. Focus on correctness, regressions, data handling and missing coverage. For each actionable finding, give severity, file and line, trigger and impact. Separate confirmed defects from questions. Do not edit files. If you find no actionable defects, say so and list the checks you could not perform.

Check: Confirm each finding against the current diff and trigger.

08 Add tests that catch a failure

A behavior needs protection against a meaningful regression.

Add focused tests for [behavior] in [file or module], following this project's conventions. Cover [normal case], [edge case] and [failure case]. Test observable behavior rather than copying implementation details. Run the relevant suite and explain which regression each test would catch. Do not alter production behavior merely to make a failing expectation pass.

Check: Check that the tests exercise the stated behavior and edge cases.

09 Verify a browser journey

Check a real interface, including its mobile layout.

Verify [journey] at [URL] on desktop and mobile. Use [safe test data] and follow [steps]. Check the expected result, keyboard access, errors, visual layout and horizontal overflow at [widths]. Report observations and retain useful evidence. Do not send real messages, submit payments or change live records; stop before any such step and describe what remains untested.

Check: Match the evidence to the stated widths and expected result.

To revise: Keep [what works]. Change [specific issue]. Recheck [success condition].

Read the full illustrated guide on Kingy.ai

OpenAI prompting Codex best practices Checked Oct 16, 2026

Codex prompt cookbook

Coordinate and reuse

OUTCOME + CONTEXT + BOUNDARIES + A RESULT YOU CAN CHECK

Use advanced tools only when available and useful for the task.

10 Coordinate a larger implementation

Independent work can help a clearly defined change.

Deliver [change] in [repository] with [acceptance checks]. Inspect the starting state and preserve existing edits. Plan the dependencies, then implement. Where tools support it, use scoped subagents for independent work and separate worktrees for parallel editing. Give each lane clear ownership, integrate once, and verify the combined result. Report delivery state and remaining risks; do not deploy.

Check: Verify the integrated result, not only each lane's output.

11 Package a workflow as a skill

A task has worked manually and you need to repeat it.

Create a local skill for [repeatable job] in [skill folder]. Use [successful workflow] as the reference. Define when to use it, required inputs, steps, outputs, checks and failure handling. Keep the instructions concise and add scripts only where they improve reliability. Try [representative task], report the result and leave account connections, scheduling and publication outside this task.

Check: Try a realistic input and inspect the output and failure behavior.

12 Prepare a reliable continuation

Another session needs to continue from the actual state.

Prepare a continuation note for [task] in [file]. Record the objective, decisions, project path, branch or worktree, changed files, completed checks, failures, external actions and current delivery state. Link the supporting evidence and list the next concrete steps. Distinguish verified facts from assumptions. Keep secrets out. Do not perform new external writes while preparing the note.

Check: Confirm the note matches current files and external state.

Original Kingy.ai examples. Adapt the scope; tool access and permissions still apply.

Read the full illustrated guide on Kingy.ai

OpenAI prompting Codex best practices Checked Oct 16, 2026

Codex shortcuts & commands

Desktop app / documented defaults

Selected defaults: Cmd = Command; Alt = Mac Option; Ctrl = Control. Some actions need a feature.
Find, rebind or reset keys in Settings > Keyboard Shortcuts. Focus and keyboard layout matter.

NAVIGATION AND CONTROLS

Action	macOS	Windows / Linux
Command menu	Cmd+Shift+P	Ctrl+Shift+P
Settings	Cmd+,	Ctrl+,
Shortcut reference	Cmd+/	Ctrl+/
Open folder	Cmd+O	Ctrl+O
Sidebar	Cmd+B	Ctrl+B
Bottom panel	Cmd+J	Ctrl+J
Terminal	Ctrl+`	Ctrl+`
Model picker	Ctrl+Shift+M	Ctrl+Shift+M
Activity (when available)	Cmd+Alt+U	Ctrl+Alt+U

CHATS AND ATTENTION

Action	macOS	Windows / Linux
New chat	Cmd+N	Ctrl+N
Find in this chat	Cmd+F	Ctrl+F
Archive chat	Cmd+Shift+A	Ctrl+Shift+A
Mark chat unread	Cmd+Shift+U	Ctrl+Shift+U
Pin / unpin	Cmd+Alt+P	Ctrl+Alt+P
Rename chat	Cmd+Alt+R	Ctrl+Alt+R
Next chat needing attention	Cmd+Alt+A	Ctrl+Alt+A
Clear all unread indicators	Shift+Esc	Shift+Esc

FILES, REVIEW AND BROWSER

Action	macOS	Windows / Linux
Search files	Cmd+P	Ctrl+P
File tree	Cmd+Shift+E	Ctrl+Shift+E
Review tab	Ctrl+Shift+G	Ctrl+Shift+G
New browser tab (if available)	Cmd+T	Ctrl+T
Line / browser address bar	Cmd+L	Ctrl+L
Reload focused browser	Cmd+R	Ctrl+R
Full workspace view	Cmd+Shift+F	Ctrl+Shift+F
Cycle workspace layout	Cmd+Shift+B	Ctrl+Shift+B

Focus matters. Cmd+K / Ctrl+K also opens the command menu, but clears a focused terminal. Cmd+L / Ctrl+L goes to a line in an editor or the address bar in a browser.

Full illustrated guide on Kingy.ai

OpenAI app shortcuts

Checked Oct 16, 2026

Codex shortcuts & commands

Desktop app / all 28 slash commands in the current reference

Type / in the composer and choose a listed action. Type $ to invoke a skill.
Some commands need a local project, Git repository, cloud access or an eligible account.

Plan, inspect and configure

`/plan`
Plan a multi-step task.

`/goal`
Set a persistent objective.

`/review`
Review local changes or a base diff.

`/status`
Show chat ID, context and limits.

`/model`
Choose the chat model.

`/reasoning`
Choose reasoning effort.

`/fast`
Toggle a Fast tier if offered.

`/compact`
Summarize context to free room.

`/init`
Scaffold project AGENTS.md.

`/mcp`
Inspect connected MCP servers.

`/memories`
Configure memory, where available.

`/ide-context`
Toggle shared editor context.

`/feedback`
Open feedback; logs are optional.

`/approve`
Allow one retry after auto-review denial.

Choose a workspace or manage chats

`/project`
Choose a project for new chats.

`/chat`
Choose a projectless new chat.

`/new`
Blank chat in the same workspace, if offered.

`/local`
Use the selected local project.

`/worktree`
Use a new Git worktree.

`/cloud`
Use cloud execution, if available.

`/cloud-environment`
Choose a cloud environment.

`/fork`
Copy a local chat to a chat or worktree.

`/side`
Open a temporary side chat.

`/archive`
Archive this chat.

`/pin`
Toggle pinning, if available.

`/rename`
Change the chat name.

`/pet`
Show or hide the desktop pet.

`/usage`
Open usage/billing, if available.

Settings > General controls Enter vs Cmd+Enter (Mac) / Ctrl+Enter (Windows/Linux), and steer vs queue. Use the goal row to pause, resume or edit an active goal.

Full illustrated guide on Kingy.ai

OpenAI app slash commands Settings Checked Oct 16, 2026

Codex shortcuts & commands

CLI / terminal keys, slash commands and shell commands

INTERACTIVE TERMINAL KEYS

`@`	Find and add a workspace file path.
`Up / Down`	Recall draft history.
`Ctrl+R`	Search prompt history; Enter selects, Esc cancels.
`Ctrl+O`	Copy the latest completed output.
`Enter while running`	Steer the active turn with new instructions.
`Tab while running`	Queue a message, slash command or shell line.
`Esc twice, empty input`	Edit the previous message and fork there.
`! at line start`	Run a shell command under current permissions.

SELECTED CLI SLASH COMMANDS / TYPE INSIDE CODEX

`/status`	Session setup and context use.
`/model`	Model and available effort.
`/plan`	Plan mode.
`/goal`	Manage a persistent goal.
`/permissions`	Change approval/access controls.
`/keymap`	Inspect or remap TUI shortcuts.
`/diff`	Inspect Git changes and new files.
`/review`	Review the working tree.
`/mention`	Attach a file.

`/copy`	Copy the last completed output.
`/resume`	Pick a saved chat.
`/fork`	Fork the current chat.
`/new`	Start a fresh chat.
`/compact`	Summarize earlier context.
`/ps`	Inspect background terminals.
`/stop`	Stop this chat's background terminals.
`/pwd`	Show the working directory.
`/exit`	Leave the CLI; save important work first.

SHELL COMMANDS / TYPE AT YOUR TERMINAL PROMPT

`codex --help`	Installed command reference.
`codex --version`	Installed CLI version.
`codex`	Start an interactive CLI session.
`codex --cd /path/to/repo`	Start in a chosen directory.
`codex resume`	Pick a saved CLI chat.
`codex fork`	Pick a saved chat to fork.
`codex exec "TASK"`	Run a non-interactive task; may edit files.
`codex review --uncommitted`	Review the working changes.

Use /keymap for custom CLI bindings and codex --help for your installed release. /stop stops background terminals; /exit leaves the CLI. Shell commands run under the current permissions.

Full illustrated guide on Kingy.ai

OpenAI CLI reference

Checked Oct 16, 2026

Codex shortcuts & commands

VS Code extension / commands and custom key bindings

Open the Command Palette: Cmd+Shift+P on Mac or Ctrl+Shift+P on Windows/Linux.
Run Preferences: Open Keyboard Shortcuts, then search Codex or a command ID below.

EXTENSION COMMAND IDS / BIND THESE IN THE EDITOR

`chatgpt.addToThread`	Add selected lines; unassigned.
`chatgpt.addFileToThread`	Add the whole file; unassigned.
`chatgpt.newChat`	New chat; Cmd+N / Ctrl+N.
`chatgpt.newCodexPanel`	New panel; unassigned.
`chatgpt.openCommandMenu`	Codex command menu; unassigned.
`chatgpt.openSidebar`	Codex sidebar; unassigned.

EXTENSION SLASH COMMANDS / TYPE IN ITS COMPOSER

Work and execution

`/approve`
Retry an automatic-review denial.

`/cloud`
Use cloud execution if available.

`/cloud-environment`
Choose cloud setup.

`/compact`
Compact chat context.

`/fast`
Toggle an offered Fast tier.

`/feedback`
Feedback and optional logs.

`/fork`
Copy a local chat.

`/goal`
Set a persistent goal.

`/ide-context`
Toggle automatic editor context.

`/init`
Scaffold AGENTS.md.

`/local`
Use the local workspace.

Tools and chat controls

`/mcp`
View connected servers.

`/memories`
Memory controls, if available.

`/model`
Choose the chat model.

`/personality`
Response style, if supported.

`/plan`
Toggle planning.

`/project`
Project for new chats.

`/reasoning`
Choose effort.

`/review`
Review edits or a base diff.

`/side`
Temporary side chat.

`/status`
Chat ID, context and limits.

`/worktree`
New Git worktree.

Defaults apply in the extension context; editor or custom bindings may intercept them. Desktop app keys and CLI-only commands do not automatically apply in VS Code.

Full illustrated guide on Kingy.ai

OpenAI extension commands

Checked Oct 16, 2026