NCPCS

Codex Session Builder

Operational Manual

FLOW	**The working path** Idea -> Refine the instructions -> Configure the agent -> Launch -> Execute -> Verify

This manual replaces the earlier training-center manual.

The page is now designed for one job: build the command that opens the right Codex session for a refined PromptStudio instruction.

MAIN OPERATION

IDEA
-> PromptStudio.us: refine the instruction
-> ncpcs.com/cli.html: configure the Codex session
-> PowerShell: launch Codex
-> Codex: paste and execute the instruction
-> Verify: /status, diff, tests, review

Keven Crick

N.C. Programmable Cognitive Solutions

www.ncpcs.com/cli.html | www.promptstudio.us

Instructions checked: July 20, 2026

1. The Main Idea

A coding agent needs two clear inputs. They are different, and both matter.

1. THE INSTRUCTION	2. THE SESSION SETTINGS
Created or refined in PromptStudio. It tells Codex what result you want. *Example: “Create a responsive website and test the navigation.”*	Selected in cli.html. They tell Codex where it may work and how much access it receives. *Example: project folder, workspace-write, approval policy, images, and web search.*

KEY

Simple rule
PromptStudio defines the mission. cli.html defines the operating boundaries. Codex performs the work.

Quick start

1. Start with an idea for a program, website, repair, or change.
2. Use PromptStudio.us to refine the instruction.
3. Open ncpcs.com/cli.html.
4. Paste the refined instruction into the page.
5. Enter the project folder.
6. Choose a session profile.
7. Add only the optional operators you need.
8. Copy the generated command into PowerShell.
9. When Codex opens, paste the refined instruction.
10. Run /status and inspect the result before accepting it.

NOTE

Do not mix the two inputs
The PowerShell command opens and configures Codex. The refined instruction is pasted after Codex opens.

2. Refine the Instruction First

Start with the idea. Then use PromptStudio to turn the idea into a clear execution request. The goal is a high-signal prompt: everything Codex needs, with unrelated wording removed.

A strong instruction answers these questions

- What should Codex create, change, inspect, or repair?
- Which files, folders, languages, or technologies are involved?
- What must stay unchanged?
- What features are required?
- How should Codex test or verify the result?
- What should Codex report when it is finished?

Weak instruction

TOO MUCH IS LEFT TO GUESS

Make my website better.

Stronger instruction

CLEAR RESULT + LIMITS + VERIFICATION

Inspect the current single-file HTML website.
Simplify the navigation for a first-time visitor.
Preserve NCPCS and Keven Crick branding.
Do not add external libraries or CDNs.
Keep the page usable on phones and desktop computers.
Before editing, summarize the changes you plan to make.
After editing, check the HTML and JavaScript for errors.
Report every file changed and explain how to test it.

GOAL

One-shot goal
A strong first instruction can reduce extra back-and-forth. It does not remove the need to check the result.

What to paste into cli.html

Paste the finished PromptStudio instruction into the first box. cli.html keeps that instruction separate from the command used to launch Codex.

3. Configure the Codex Session

Open the session builder: www.ncpcs.com/cli.html

1

Paste the refined instruction

Place the completed PromptStudio instruction in the task box. This is what you will paste into Codex after it opens.

2

Enter the project folder

Use the full Windows path to the folder Codex should treat as the main workspace.

`C:\Path\To\Project`

3

Choose a starting profile

Pick the amount of access that matches the operation. The page generates the correct launch settings.

4

Add optional operators

Images, web search, another writable folder, a model, or automation should be selected only when the task needs them.

5

Copy the command

Paste the generated command into PowerShell or Windows Terminal. This starts Codex with the selected configuration.

The four session profiles

Profile	Main setting	Risk	Use it when
Inspect	read-only	Minimal	You want Codex to read and explain the project without normal editing.
Build or edit	workspace-write + on-request	Low	You want Codex to edit inside the project and ask before crossing normal boundaries.
Full access	--yolo	Critical	You intentionally want approvals and sandbox restrictions bypassed.
Custom	Your choices	Varies	You understand sandbox and approval policies and need a special combination.

4. Optional Operators

The starting profile handles the common settings. Add an operator only when the task needs it.

Operator	Plain name	What it changes
-C, --cd PATH	Main project folder	Sets the working directory before Codex processes the task.
--add-dir PATH	Another writable folder	Allows one additional directory without removing all workspace boundaries.
-i, --image PATH	Image input	Adds a screenshot, diagram, or other image to the starting context.
--search	Live model web search	Lets the model use live web-search results. This is not the same as unrestricted network access for every shell command.
-m, --model NAME	Model selection	Uses a named model when that model is available to the account and installed Codex version.
-p, --profile NAME	Saved profile	Loads a named Codex configuration profile.
-c, --config KEY=VALUE	One-run override	Changes one configuration value for this launch.
--no-alt-screen	Normal terminal scrollback	Keeps terminal output in the normal screen buffer.
--oss	Local open-source model	Uses a configured local model provider such as LM Studio or Ollama.
codex exec	Automation	Runs a task without opening the normal interactive composer.
--json	Machine-readable progress	Produces JSON Lines output for scripts or CI systems.
-o, --output-last-message PATH	Save final response	Writes the final assistant message to a file.
--ephemeral	Temporary run	Avoids saving normal session rollout files.

TIP

Keep it simple

More operators do not automatically produce a better result. Select the smallest set needed for the operation.

5. Launch and Execute

For an interactive session, cli.html gives you two separate items to copy.

A. Copy the launch command to PowerShell

EXAMPLE LAUNCH COMMAND

```
codex -C "C:\Websites\NCPCS" --sandbox workspace-write --ask-for-approval on-request
```

- Open PowerShell or Windows Terminal.
- Paste the generated command.
- Press Enter.
- Wait for the Codex interface to open and finish loading the selected session.

B. Paste the instruction after Codex opens

EXAMPLE CODEX INSTRUCTION

Inspect the current single-file HTML website.
Simplify the navigation for a first-time visitor.
Preserve NCPCS branding and verify the final page.

- Paste the complete PromptStudio instruction into the Codex composer.
- Submit the instruction.
- Allow Codex to inspect the project and perform the work within the selected session boundaries.

ORDER

Why the order matters
Launch operators such as -C, --sandbox, --add-dir, and --search configure Codex before the task begins. They are not normal sentences to paste into an already-running prompt.

Interactive versus automated

Interactive mode opens Codex so you can continue communicating with it. Automation uses codex exec and passes the task through the launch command or standard input. Begin with interactive mode unless you specifically need a script or CI process.

6. Verify the Session and the Result

A good prompt and a correct launch command improve the first result. Verification confirms that the agent worked in the right place and produced what you expected.

First check the session

TYPE THIS INSIDE CODEX

```
/status
```

- Confirm the project directory.
- Confirm the active model.
- Confirm the sandbox and approval settings.
- Confirm that any extra directory or other selected capability is available.

Then check the work

Command	Purpose
!git status	Shows which files changed or are untracked.
/diff	Displays the current file changes so you can read them.
tests / lint / validation	Checks whether the program or website still works.
/review	Asks Codex to analyze the changes for problems. It supports, but does not replace, reading the diff.

When the result is not right

- Do not start over with a completely unrelated prompt.
- Tell Codex exactly what is wrong.
- Give one focused correction at a time.
- Rerun the relevant test or check.
- Use Git or your backup to restore unwanted changes.

CHECK

Professional result
Single-shot efficiency is a useful goal. Verification is still part of the operation.

7. Full Access and --yolo

The Full Access profile uses --yolo. This is a real operational choice, but it is not the same as a normal convenience setting.

STOP

Critical warning
--yolo bypasses Codex approvals and sandboxing. Codex may act without asking first and may reach beyond normal workspace limits.

Use Full Access only when

- You understand what the task may run or change.
- The project is backed up or committed to Git.
- Unrelated valuable files are not exposed.
- Sensitive credentials and secrets are protected.
- The machine or environment can be restored if something goes wrong.
- You intentionally accept the loss of approval and sandbox protection.

Safer everyday choice

BALANCED BUILD OR EDIT

```
codex -C "C:\Path\To\Project" --sandbox workspace-write --ask-for-approval on-request
```

This allows Codex to work inside the project while keeping approval checkpoints for boundary-crossing operations.

Keven’s personal preference versus the public default

Keven may choose --yolo for selected personal operations. The public page should still explain the difference and require the user to acknowledge the risk. A personal preference is not automatically the correct setting for every computer or every project.

8. Complete Example

Goal

Simplify an NCPCS webpage while preserving branding and checking the final code.

Step 1 - Refined PromptStudio instruction

```
Inspect the current single-file HTML page.
Simplify the first screen and navigation.
Preserve Keven Crick and NCPCS branding.
Do not add external libraries, fonts, or CDNs.
Keep the page responsive and keyboard accessible.
Before editing, explain the proposed changes.
After editing, validate the HTML and check for JavaScript errors.
Report all changes and explain how to test them.
```

Step 2 - Configure cli.html

- Project folder: C:\Websites\NCPCS
- Profile: Build or edit
- Optional operators: none unless an image or live search is needed

Step 3 - Launch command

```
codex -C "C:\Websites\NCPCS" --sandbox workspace-write --ask-for-approval on-request
```

Step 4 - Execute

After Codex opens, paste the PromptStudio instruction and submit it.

Step 5 - Verify

```
/status
!git status
/diff
Run the page checks or tests
/review
```

9. Common Problems

FIX	**The command opens the wrong folder** Check the project path. Use a full path with -C. Put paths containing spaces inside quotation marks.
FIX	**Codex cannot edit files** You may have selected Inspect/read-only. Use Build or edit when changes are expected.
FIX	**Codex asks for approval** That is normal with on-request when an operation crosses a boundary. Read the request before approving it.
FIX	**An image is missing** Confirm the image path exists and add it with -i or --image before launching Codex.
FIX	**Web search does not work** Confirm --search was selected. Remember that model web search is separate from unrestricted network access for shell commands.
FIX	**The model name fails** Model availability can differ by account and installed Codex version. Leave the model blank to use the configured default or check the installed options.
FIX	**The result is close but not correct** Give one focused correction. State what is wrong, what must remain, and how to verify the fix.
FIX	**You are unsure what is active** Run /status inside Codex. The installed codex --help, slash-command menu, and /status are authoritative for that installation.

10. Final Operating Flow

1 **IDEA**

Define the program, website, repair, change, or final result.

2 **REFINE**

Use PromptStudio.us to create a clear instruction with requirements, limits, and verification.

3 **CONFIGURE**

Use cli.html to select the project folder, access level, and needed CLI operators.

4 **LAUNCH**

Copy the generated command into PowerShell or Windows Terminal.

5 **EXECUTE**

After Codex opens, paste the refined PromptStudio instruction.

6 **VERIFY**

Check /status, review file changes, run tests, and confirm the final result.

FLOW

The complete method

Idea -> Refine the instructions with PromptStudio.us -> Configure the agent with cli.html -> Launch Codex -> Execute -> Verify

Official references

- OpenAI Codex CLI command-line reference
- OpenAI Codex slash commands
- OpenAI Codex best practices
- OpenAI Codex prompting guidance
- NCPCS Codex Session Builder
- PromptStudio

Important: Codex changes over time. The installed codex --help, slash-command menu, and /status are the final authority for the version running on that computer.