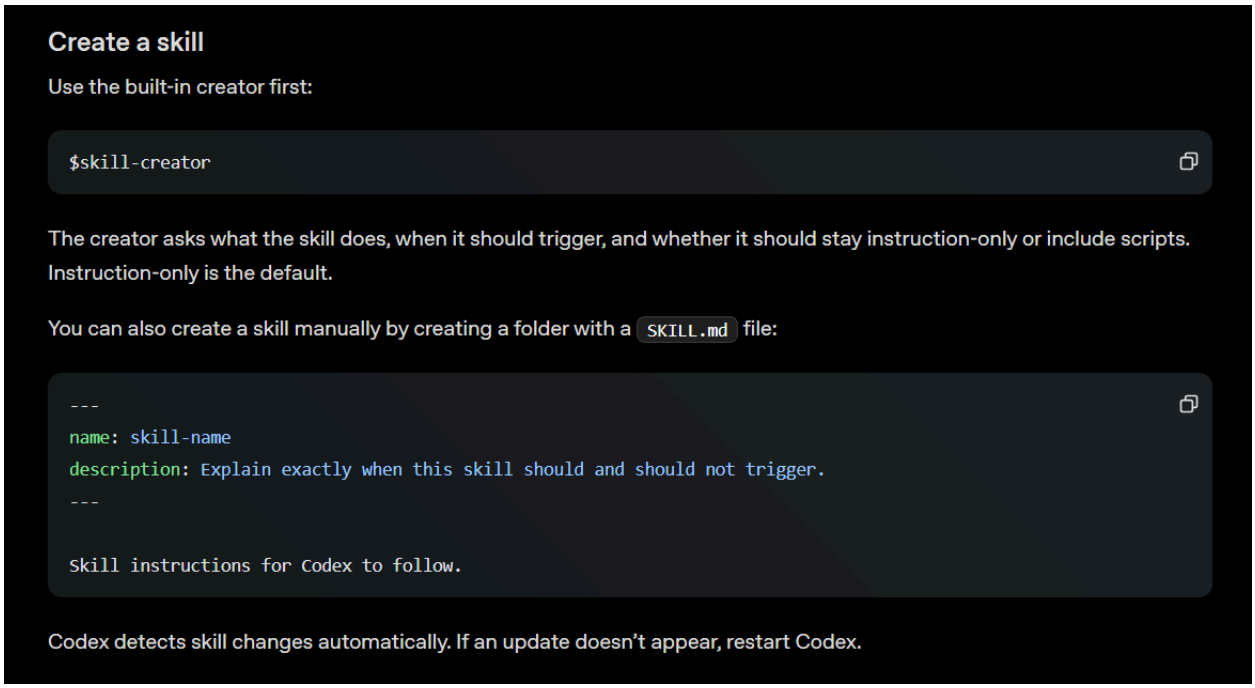
Codex Notes

- Codex is best used when we have a premade AGENTS.md file which gives a list of conventions we would want in our code base, how we want our UI to look, etc. This is like persistent memory the model can access of what we want our code to look like.
- We should not follow a specific template or a off-the-shelf AGENTS.md , but should keep those design principles in mind when we make ours.
- The main thing we should keep in mind for our project is utilizing the ready MCP's that can allow codex to view the browser, access figma for designing an interesting UI
- We should not overwhelm codex with MCP's as too many mcps just become cumbersome in the long run
- In-app browser – Codex app | OpenAI Developers This is useful for giving Codex a preview of what the website looks like in it's current state. This could be done without an mcp, we can just take a screenshot of our ui and ask it to fix specific features.
- Useful @playwright/mcp - npm playwright mcp, allows the model to look at browsers. ChromeDevTools/chrome-devtools-mcp: Chrome DevTools for coding agents useful for if our browser is chrome.
- github/github-mcp-server: GitHub's official MCP Server allows codex to add pull requests to our repo, essentially fully automating our workflow.
- Agent skills: basically allows our model to become an expert in one particular thing
- Full documentation Agent Skills – Codex | OpenAI Developers
- Use agent skills to extend Codex with task-specific capabilities. A skill packages instructions, resources, and optional scripts so Codex can follow a workflow reliably. Skills build on the open agent skills standard.
- Skills use progressive disclosure to manage context efficiently: Codex starts with each skill's metadata (name, description, file path, and optional metadata from agents/openai.yaml). Codex loads the full SKILL.md instructions only when it decides to use a skill.

- A skill is a directory with a SKILL.md file plus optional scripts and references. The SKILL.md file must include name and description.



Create a skill

Use the built-in creator first:

```
$skill-creator
```

The creator asks what the skill does, when it should trigger, and whether it should stay instruction-only or include scripts. Instruction-only is the default.

You can also create a skill manually by creating a folder with a `SKILL.md` file:

```
---
name: skill-name
description: Explain exactly when this skill should and should not trigger.
---

Skill instructions for Codex to follow.
```

Codex detects skill changes automatically. If an update doesn't appear, restart Codex.