OpenAI

Aug 7, 2025

GPT-5 prompting guide

1. Prompt for less eagerness

Prompting for less eagerness

GPT-5 is, by default, thorough and comprehensive when trying to gather context in an agentic environment to ensure it will produce a correct answer. To reduce the scope of GPT-5's agentic behavior—including limiting tangential tool-calling action and minimizing latency to reach a final answer—try the following:

- Switch to a lower `reasoning_effort`. This reduces exploration depth but improves efficiency and latency. Many workflows can be accomplished with consistent results at medium or even low `reasoning_effort`.
- Define clear criteria in your prompt for how you want the model to explore the problem space. This reduces the model's need to explore and reason about too many ideas:

```
<context_gathering>
Goal: Get enough context fast. Parallelize discovery and stop as soon as you can a
Method:
- Start broad, then fan out to focused subqueries.
- In parallel, launch varied queries; read top hits per query. Deduplicate paths a
- Avoid over searching for context. If needed, run targeted searches in one paralle
Early stop criteria:
- You can name exact content to change.
- Top hits converge (~70%) on one area/path.
Escalate once:
- If signals conflict or scope is fuzzy, run one refined parallel batch, then proce
Depth:
- Trace only symbols you'll modify or whose contracts you rely on; avoid transitive
Loop:
- Batch search → minimal plan → complete task.
- Search again only if validation fails or new unknowns appear. Prefer acting over
</context_gathering>
```

2. Prompt for more eagerness

On the other hand, if you'd like to encourage model autonomy, increase tool-calling persistence, and reduce occurrences of clarifying questions or otherwise handing back to the user, we recommend increasing `reasoning_effort`, and using a prompt like the following to encourage persistence and thorough task completion:

```
<persistence>
- You are an agent - please keep going until the user's query is completely
- Only terminate your turn when you are sure that the problem is solved.
- Never stop or hand back to the user when you encounter uncertainty — rese
- Do not ask the human to confirm or clarify assumptions, as you can always
</persistence>
```

Generally, it can be helpful to clearly state the stop conditions of the agentic tasks, outline safe versus unsafe actions, and define when, if ever, it's acceptable for the model to hand back to the user. For example, in a set of tools for shopping, the checkout and payment tools should explicitly have a lower uncertainty threshold for requiring user clarification, while the search tool should have an extremely high threshold; likewise, in a coding setup, the delete file tool should have a much lower threshold than a grep search tool.

3. One shot prompt to build apps

Zero-to-one app generation

GPT-5 is excellent at building applications in one shot. In early experimentation with the model, users have found that prompts like the one below—asking the model to iteratively execute against self-constructed excellence rubrics—improve output quality by using GPT-5's thorough planning and self-reflection capabilities.

```
<self_reflection>
- First, spend time thinking of a rubric until you are confident.
- Then, think deeply about every aspect of what makes for a world-class o
- Finally, use the rubric to internally think and iterate on the best pos
</self_reflection>
```

Matching codebase design standards

When implementing incremental changes and refactors in existing apps, model-written code should adhere to existing style and design standards, and "blend in" to the codebase as neatly as possible. Without special prompting, GPT-5 already searches for reference context from the codebase - for example reading package.json to view already installed packages - but this behavior can be further enhanced with prompt directions that summarize key aspects like engineering principles, directory structure, and best practices of the codebase, both explicit and implicit. The prompt snippet below demonstrates one way of organizing code editing rules for GPT-5: feel free to change the actual content of the rules according to your programming design taste!

4. Set system prompts reminders

Perhaps unsurprisingly, we recommend prompting patterns that are similar to GPT-4.1 for best results. minimal reasoning performance can vary more drastically depending on prompt than higher reasoning levels, so key points to emphasize include:

1. Prompting the model to give a brief explanation summarizing its thought process at the start of the final answer, for example via a bullet point list, improves performance on tasks requiring higher intelligence.
2. Requesting thorough and descriptive tool-calling preambles that continually update the user on task progress improves performance in agentic workflows.
3. Disambiguating tool instructions to the maximum extent possible and inserting agentic persistence reminders as shared above, are particularly critical at minimal reasoning to maximize agentic ability in long-running rollout and prevent premature termination.
4. Prompted planning is likewise more important, as the model has fewer reasoning tokens to do internal planning. Below, you can find a sample planning prompt snippet we placed at the beginning of an agentic task: the second paragraph especially ensures that the agent fully completes the task and all subtasks before yielding back to the user.

5. Use markdown formatting

Markdown formatting

By default, GPT-5 in the API does not format its final answers in Markdown, in order to preserve maximum compatibility with developers whose applications may not support Markdown rendering. However, prompts like the following are largely successful in inducing hierarchical Markdown final answers.

```
- Use Markdown **only where semantically correct** (e.g., `inline co
- When using markdown in assistant messages, use backticks to format
```

Occasionally, adherence to Markdown instructions specified in the system prompt can degrade over the course of a long conversation. In the event that you experience this, we've seen consistent adherence from appending a Markdown instruction every 3-5 user messages.

Found this helpful?

↓

Repost it

to help one person
in your network.