# Prompt Engineering with GitHub Copilot

This guide is intended to shed light on the art of creating prompts with GitHub Copilot.

## On this page



# Prompt Engineering with GitHub Copilot

Welcome to the knowledgebase page dedicated to Prompt Engineering with GitHub Copilot. This guide is intended to shed light on the art of creating prompts, which is a fundamental aspect of interacting with AI models like GitHub Copilot.

## 1) What is a Prompt?

A **prompt** is an initial input or question given to an AI model, setting the stage for the AI to generate a response. When dealing with models like Copilot, which are based on the GPT (Generative Pre-trained Transformer) architecture, the prompt helps the model understand the context and desired outcome of the user.

# 2) What is Prompt Engineering?

**Prompt Engineering** is the process of designing, testing, and refining these input queries to elicit desired outputs from the AI model. It requires a combination of understanding the model's behavior and crafting prompts that guide the model effectively.

# 3) Why is Prompt Engineering important?

Prompt Engineering is vital for:

- **Precision**: To ensure the AI model generates accurate and contextually relevant responses.
- **Efficiency**: Reduces the back-and-forth between the user and the model, saving time.
- **Flexibility**: Allows users to guide the model in various directions based on evolving requirements.

# 4) Techniques

## Zero-Shot Prompting

This involves presenting a task to the AI model without providing any prior examples. The model relies entirely on its pre-trained knowledge.

*Example*: Asking Copilot to "Write a Python function to calculate factorial" without providing any related code or context.

## One-Shot Prompting

Here, you provide a single example to the model to guide its response.

*Example*: Showing Copilot a single example of a Python list comprehension and then asking it to generate another one based on new criteria.

## Few-Shot Prompting

This involves providing multiple examples to help the model grasp the task better and generate a more refined response.

*Example*: Giving Copilot several examples of SQL queries and then asking it to craft a new, complex query.

# 5) Cornerstones of Prompting

### Provide Examples

- Especially beneficial in **Few-Shot Prompting**.
- Examples guide the AI in understanding the pattern or format you're looking for.

### Give Direction

- Be clear and specific about what you want.
- For instance, instead of just asking for a "sorting function", specify "a sorting function in JavaScript that sorts an array of strings in reverse alphabetical order".

### Format your Responses

- If you need the output in a specific format, indicate this in the prompt.
- E.g., "Provide the answer in JSON format."

### Evaluate Quality

- Always review the AI-generated content for accuracy and relevance.
- Refine your prompts based on the outputs you receive.

### Chaining Prompts

- Use the output from one prompt as the input for another.
- This allows for sequential tasks and iterative development.

### Control Output Verbosity

- If you need a detailed explanation, you can ask the model explicitly.
- Conversely, if you require a concise answer, instruct the model to "summarize" or "provide a brief" response.

## 6) How to Learn More

### Resources

- Prompting Guide
- The Complete Prompt Engineering for AI Bootcamp (2023)
- Google Scholars -> Prompt Engineering Search.
    - Numerous techniques were discovered by reading peer-reviewed journals for the GitHub CfB training design such as Zero, One and Few shot programming as well as Let's think

step by step.
    - Many of these optimiziations are still being found in the wild and peer-reviewed journals.

## Conclusion

Prompt Engineering is both an art and a science, and mastering it can greatly enhance your experience with GitHub Copilot. By understanding and employing the techniques and cornerstones detailed above, you can communicate more effectively with the AI, making your coding sessions more productive and efficient.

*Note: Always consider the dynamic nature of AI models and be prepared to iterate on your prompts for optimal results. As AI models evolve, the techniques and strategies for prompting may also see shifts.*