

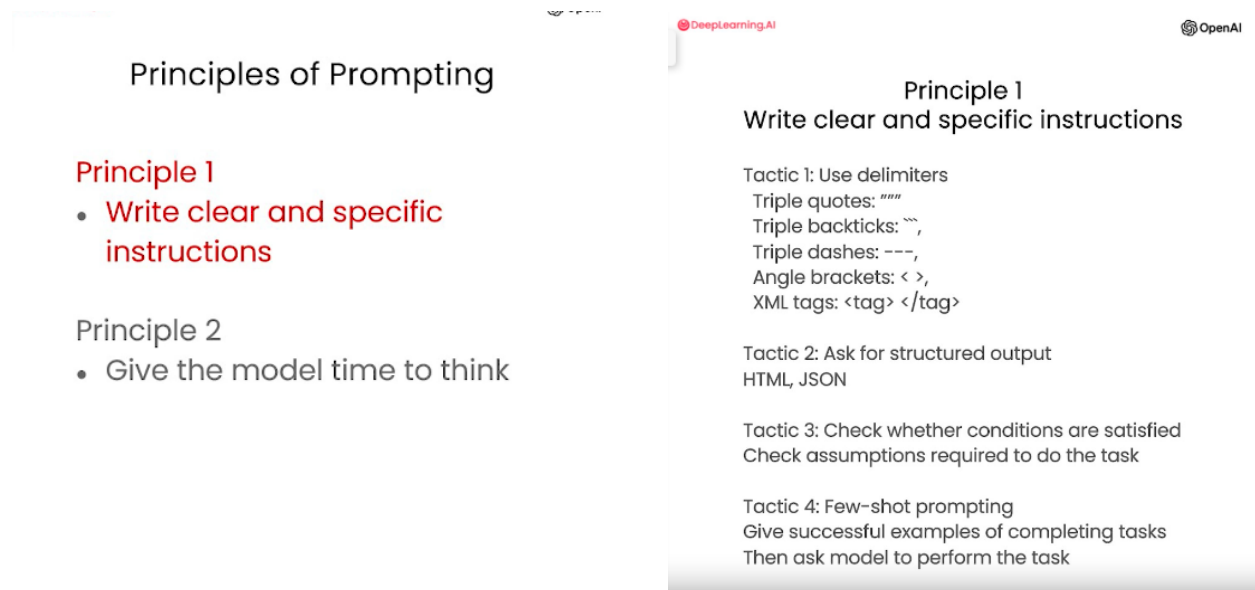
ChatGPT Prompt Engineering for Developers

<https://learn.deeplearning.ai/chatgpt-prompt-eng/lesson/1/introduction>

OpenAI API Keys: <https://platform.openai.com/account/api-keys>

Intro

Guidelines



Principles of Prompting

Principle 1

- Write clear and specific instructions

Principle 2

- Give the model time to think

Principle 1

Write clear and specific instructions

Tactic 1: Use delimiters

Triple quotes: `"""`

Triple backticks: `````

Triple dashes: `---`

Angle brackets: `< >`

XML tags: `<tag> </tag>`

Tactic 2: Ask for structured output

HTML, JSON

Tactic 3: Check whether conditions are satisfied

Check assumptions required to do the task

Tactic 4: Few-shot prompting

Give successful examples of completing tasks

Then ask model to perform the task

Deeplearning.AI OpenAI

Principle 2

Give the model time to think

Tactic 1: Specify the steps to complete a task

- Step 1: ...
- Step 2: ...
- ...
- Step N: ...

Tactic 2: Instruct the model to work out its own solution before rushing to a conclusion

Deeplearning.AI OpenAI

Model Limitations

Hallucination
Makes statements that sound plausible but are not true

Reducing hallucinations:
First find relevant information, then answer the question based on the relevant information.

Iterative

Summarizing

Inferring

Transforming

Expanding

Chatbot

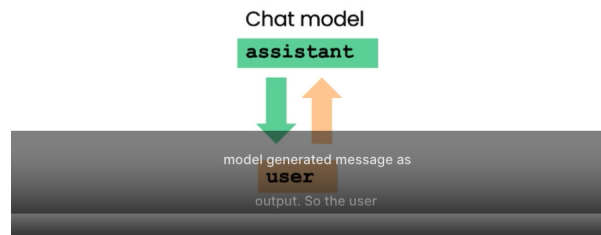
OpenAI API call

```
def get_completion(prompt,
                    model="gpt-3.5-turbo"):
    messages = [{"role": "user",
                  "content": prompt}]

    response = openai.ChatCompletion.create(
        model=model,
        messages=messages,
        temperature=0)
```

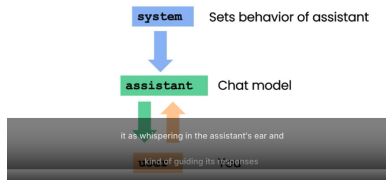
OpenAI API call

```
def get_completion(prompt,
                   model="gpt-3.5-turbo"):
    messages = [{"role": "user",
                  "content": prompt}]
    response = openai.ChatCompletion.create(
        model=model,
        messages=messages,
        temperature=0)
```



Role

```
messages = [
    {"role": "system",
     "content": "You are an assistant... "},
    {"role": "user",
     "content": "tell me a joke "},
    {"role": "assistant",
     "content": "Why did the chicken... "},
    ...
]
```



Adding to the Context

```
messages = [
    system
    user
    assistant
    user
    assistant
    ...
]
```

System message

```
context = [
    {'role': 'system',
     'content': ""
     You are OrderBot, an automated service
     to collect orders for a pizza restaurant.
     You first greet the customer,
     then collect the order,
     and then ask if it's a pickup or delivery.
     You wait to collect the entire order,
     then summarize it and check for a final time
     if the customer wants to add anything else.
     If it's a delivery, you ask for an address.
     Finally you collect the payment.
     Make sure to clarify all options, extras and sizes
     to uniquely identify the item from the menu.
     You respond in a short, very conversational
     friendly style.
     The menu includes
     ...
     style. The
     menu includes, and then, here
```

Conclusion

Summary

- Principles:
 - Write clear and specific instructions
 - Give the model time to “think”
 - Iterative prompt development
 - Capabilities: Summarizing, Inferring, Transforming, Expanding
 - Building a chatbot
-