ChatGPT를 API에서 이용하기

3-3 입력과 출력의 길이 제한 및 요금에 영향을 주는 '토큰'

Tokenizer와 tiktoken 소개

In [1]:
```
!pip install tiktoken
```

```
Requirement already satisfied: tiktoken in c:\programdata\anaconda3\lib\site-packages (0.12.0)
Requirement already satisfied: regex>=2022.1.18 in c:\programdata\anaconda3\lib\site-packages (from tiktoken) (2024.11.6)
Requirement already satisfied: requests>=2.26.0 in c:\programdata\anaconda3\lib\site-packages (from tiktoken) (2.32.5)
Requirement already satisfied: charset_normalizer<4,>=2 in c:\programdata\anaconda3\lib\site-packages (from requests>=2.26.0->t
iktoken) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in c:\programdata\anaconda3\lib\site-packages (from requests>=2.26.0->tiktoken) (3.
7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\programdata\anaconda3\lib\site-packages (from requests>=2.26.0->tiktoke
n) (2.3.0)
Requirement already satisfied: certifi>=2017.4.17 in c:\programdata\anaconda3\lib\site-packages (from requests>=2.26.0->tiktoke
n) (2025.10.5)
```

In [2]:
```
import tiktoken

text = "It’s easy to make something cool with LLMs, but very hard to make something production-ready with them."

encoding = tiktoken.encoding_for_model("gpt-4.1-mini")
tokens = encoding.encode(text)
print(len(tokens))
```

```
23
```

In [3]:
```
text = "LLM을 사용해서 멋져 보이는 것을 만들기는 쉽지만, 프로덕션 수준으로 만들어 내기는 매우 어렵다."

encoding = tiktoken.encoding_for_model("gpt-4.1-mini")
tokens = encoding.encode(text)
print(len(tokens))
```

3-4 Chat Completions API를 접하는 환경 준비

OpenAI API 키 준비

In [4]:
```python
import os
from dotenv import load_dotenv
load_dotenv()
OPENAI_API_KEY = os.getenv("OPENAI_API_KEY")
```

3-5 Chat Completions API 사용해 보기

OpenAI 라이브러리

In [5]:
```
!pip install openai
```

```
Requirement already satisfied: openai in c:\programdata\anaconda3\lib\site-packages (2.15.0)
Requirement already satisfied: anyio<5,>=3.5.0 in c:\programdata\anaconda3\lib\site-packages (from openai) (4.7.0)
Requirement already satisfied: distro<2,>=1.7.0 in c:\programdata\anaconda3\lib\site-packages (from openai) (1.9.0)
Requirement already satisfied: httpx<1,>=0.23.0 in c:\programdata\anaconda3\lib\site-packages (from openai) (0.28.1)
Requirement already satisfied: jiter<1,>=0.10.0 in c:\programdata\anaconda3\lib\site-packages (from openai) (0.12.0)
Requirement already satisfied: pydantic<3,>=1.9.0 in c:\programdata\anaconda3\lib\site-packages (from openai) (2.10.3)
Requirement already satisfied: sniffio in c:\programdata\anaconda3\lib\site-packages (from openai) (1.3.0)
Requirement already satisfied: tqdm>4 in c:\programdata\anaconda3\lib\site-packages (from openai) (4.67.1)
Requirement already satisfied: typing-extensions<5,>=4.11 in c:\programdata\anaconda3\lib\site-packages (from openai) (4.15.0)
Requirement already satisfied: idna>=2.8 in c:\programdata\anaconda3\lib\site-packages (from anyio<5,>=3.5.0->openai) (3.7)
Requirement already satisfied: certifi in c:\programdata\anaconda3\lib\site-packages (from httpx<1,>=0.23.0->openai) (2025.10.
5)
Requirement already satisfied: httpcore==1.* in c:\programdata\anaconda3\lib\site-packages (from httpx<1,>=0.23.0->openai) (1.
0.9)
Requirement already satisfied: h11>=0.16 in c:\programdata\anaconda3\lib\site-packages (from httpcore==1.*->httpx<1,>=0.23.0->o
penai) (0.16.0)
Requirement already satisfied: annotated-types>=0.6.0 in c:\programdata\anaconda3\lib\site-packages (from pydantic<3,>=1.9.0->o
penai) (0.6.0)
Requirement already satisfied: pydantic-core==2.27.1 in c:\programdata\anaconda3\lib\site-packages (from pydantic<3,>=1.9.0->op
enai) (2.27.1)
Requirement already satisfied: colorama in c:\programdata\anaconda3\lib\site-packages (from tqdm>4->openai) (0.4.6)
```

Chat Completions API 호출

In [6]:

```python
from openai import OpenAI

client = OpenAI()

response = client.chat.completions.create(
  model="gpt-4.1-mini",
  messages=[
    {"role": "system", "content": "You are a helpful assistant."},
    {"role": "user", "content": "Hello! I'm John."}
  ]
)

print(response)
```

```
ChatCompletion(id='chatcmpl-CxlgfESqM8uM7Tspp9ZxbYCwRujW1', choices=[Choice(finish_reason='stop', index=0, logprobs=None, messa
ge=ChatCompletionMessage(content='Hello John! How can I assist you today?', refusal=None, role='assistant', annotations=[], aud
io=None, function_call=None, tool_calls=None))], created=1768360697, model='gpt-4.1-mini-2025-04-14', object='chat.completion',
service_tier='default', system_fingerprint='fp_376a7ccef1', usage=CompletionUsage(completion_tokens=10, prompt_tokens=22, total
_tokens=32, completion_tokens_details=CompletionTokensDetails(accepted_prediction_tokens=0, audio_tokens=0, reasoning_tokens=0,
rejected_prediction_tokens=0), prompt_tokens_details=PromptTokensDetails(audio_tokens=0, cached_tokens=0)))
```

```
In [7]: print(response.model_dump_json(indent=2))
```

```
{
  "id": "chatcmpl-CxlgfESqM8uM7Tspp9ZxbYCwRujW1",
  "choices": [
    {
      "finish_reason": "stop",
      "index": 0,
      "logprobs": null,
      "message": {
        "content": "Hello John! How can I assist you today?",
        "refusal": null,
        "role": "assistant",
        "annotations": [],
        "audio": null,
        "function_call": null,
        "tool_calls": null
      }
    }
  ],
  "created": 1768360697,
  "model": "gpt-4.1-mini-2025-04-14",
  "object": "chat.completion",
  "service_tier": "default",
  "system_fingerprint": "fp_376a7ccef1",
  "usage": {
    "completion_tokens": 10,
    "prompt_tokens": 22,
    "total_tokens": 32,
    "completion_tokens_details": {
      "accepted_prediction_tokens": 0,
      "audio_tokens": 0,
      "reasoning_tokens": 0,
      "rejected_prediction_tokens": 0
    },
    "prompt_tokens_details": {
      "audio_tokens": 0,
      "cached_tokens": 0
    }
  }
}
```

대화 이력을 바탕으로 한 응답 얻기

In [8]:
```python
response = client.chat.completions.create(
  model="gpt-4.1-mini",
  messages=[
        {"role": "system", "content": "You are a helpful assistant."},
        {"role": "user", "content": "Hello! I'm John."},
        {"role": "assistant", "content": "Hello John! How can I assist you today?"},
        {"role": "user", "content": "Do you know my name?"}
    ]
)

print(response.model_dump_json(indent=2))
```

```
{
  "id": "chatcmpl-CxlggiklDhumMCU7arBjT90RVm1Wk",
  "choices": [
    {
      "finish_reason": "stop",
      "index": 0,
      "logprobs": null,
      "message": {
        "content": "Yes, you mentioned that your name is John. How can I help you today, John?",
        "refusal": null,
        "role": "assistant",
        "annotations": [],
        "audio": null,
        "function_call": null,
        "tool_calls": null
      }
    }
  ],
  "created": 1768360698,
  "model": "gpt-4.1-mini-2025-04-14",
  "object": "chat.completion",
  "service_tier": "default",
  "system_fingerprint": "fp_376a7ccef1",
  "usage": {
    "completion_tokens": 19,
    "prompt_tokens": 46,
    "total_tokens": 65,
    "completion_tokens_details": {
      "accepted_prediction_tokens": 0,
      "audio_tokens": 0,
      "reasoning_tokens": 0,
      "rejected_prediction_tokens": 0
    },
    "prompt_tokens_details": {
      "audio_tokens": 0,
      "cached_tokens": 0
    }
  }
}
```

응답을 스트리밍으로 받기

In [9]:
```
response = client.chat.completions.create(
  model="gpt-4.1-mini",
  messages=[
        {"role": "system", "content": "You are a helpful assistant."},
        {"role": "user", "content": "Hello! I'm John."}
  ],
  stream=True
)

for chunk in response:
  choice = chunk.choices[0]
  if choice.finish_reason is None:
    print(choice.delta.content)
```

```
Hello
 John
!
 How
 can
 I
 assist
 you
 today
?
```

(칼럼) Completions API

In [10]:
```
from openai import OpenAI

client = OpenAI()

response = client.chat.completions.create(
  model="gpt-4.1-mini",
  messages=[
    {"role": "user", "content": "Hello! I'm John."}
  ]
)

print(response.model_dump_json(indent=2))
```

```
{
  "id": "chatcmpl-CxlghMjgMEFIsOvBXLPhZsrbCiVjK",
  "choices": [
    {
      "finish_reason": "stop",
      "index": 0,
      "logprobs": null,
      "message": {
        "content": "Hello John! How can I assist you today?",
        "refusal": null,
        "role": "assistant",
        "annotations": [],
        "audio": null,
        "function_call": null,
        "tool_calls": null
      }
    }
  ],
  "created": 1768360699,
  "model": "gpt-4.1-mini-2025-04-14",
  "object": "chat.completion",
  "service_tier": "default",
  "system_fingerprint": "fp_376a7ccef1",
  "usage": {
    "completion_tokens": 10,
    "prompt_tokens": 12,
    "total_tokens": 22,
    "completion_tokens_details": {
      "accepted_prediction_tokens": 0,
      "audio_tokens": 0,
      "reasoning_tokens": 0,
      "rejected_prediction_tokens": 0
    },
    "prompt_tokens_details": {
      "audio_tokens": 0,
      "cached_tokens": 0
    }
  }
}
```

In [11]:
```python
messages = [
    {"role": "user", "content": "Hello! I'm John."},
    {"role": "assistant", "content": "Nice to meet you, John!"},
    {"role": "user", "content": "Do you know my name?"}
]

response = client.chat.completions.create(
    model="gpt-4.1-mini",
    messages=messages
)
print(response.model_dump_json(indent=2))
```

```
{
  "id": "chatcmpl-CxlgiSBESB3VNHOBDm3LOxJywyciO",
  "choices": [
    {
      "finish_reason": "stop",
      "index": 0,
      "logprobs": null,
      "message": {
        "content": "Yes, you mentioned that your name is John. How can I assist you today?",
        "refusal": null,
        "role": "assistant",
        "annotations": [],
        "audio": null,
        "function_call": null,
        "tool_calls": null
      }
    }
  ],
  "created": 1768360700,
  "model": "gpt-4.1-mini-2025-04-14",
  "object": "chat.completion",
  "service_tier": "default",
  "system_fingerprint": "fp_376a7ccef1",
  "usage": {
    "completion_tokens": 17,
    "prompt_tokens": 33,
    "total_tokens": 50,
    "completion_tokens_details": {
      "accepted_prediction_tokens": 0,
      "audio_tokens": 0,
      "reasoning_tokens": 0,
      "rejected_prediction_tokens": 0
    },
    "prompt_tokens_details": {
      "audio_tokens": 0,
      "cached_tokens": 0
    }
  }
}
```

3-6 Function calling

Function calling의 샘플 코드

OpenAI 공식 문서를 바탕으로 일부 수정한 코드입니다.

In [12]:
```
import json

def get_current_weather(location, unit="celsius"):
    weather_info = {
        "location": location,
        "temperature": "25",
        "unit": "celsius",
        "forecast": ["sunny", "windy"],
    }
    return json.dumps(weather_info)
```

In [13]:
```
functions = [
    {
        "name": "get_current_weather",
        "description": "Get the current weather in a given location",
        "parameters": {
            "type": "object",
            "properties": {
                "location": {
                    "type": "string",
                    "description": "The city and state, e.g. Seoul",
                },
                "unit": {"type": "string", "enum": ["celsius", "fahrenheit"]},
            },
            "required": ["location"],
        },
    }
]
```

In [14]:
```
messages = [{"role": "user", "content": "What's the weather like in Seoul?"}]
```

```python
response = client.chat.completions.create(
    model="gpt-4.1-mini",
    messages=messages,
    functions=functions
)

print(response.model_dump_json(indent=2))
```

```
{
  "id": "chatcmpl-CxlgjyvZfVduUJuWCRdPJo2OvAsEG",
  "choices": [
    {
      "finish_reason": "function_call",
      "index": 0,
      "logprobs": null,
      "message": {
        "content": null,
        "refusal": null,
        "role": "assistant",
        "annotations": [],
        "audio": null,
        "function_call": {
          "arguments": "{\"location\":\"Seoul\",\"unit\":\"celsius\"}",
          "name": "get_current_weather"
        },
        "tool_calls": null
      }
    }
  ],
  "created": 1768360701,
  "model": "gpt-4.1-mini-2025-04-14",
  "object": "chat.completion",
  "service_tier": "default",
  "system_fingerprint": "fp_376a7ccef1",
  "usage": {
    "completion_tokens": 21,
    "prompt_tokens": 76,
    "total_tokens": 97,
    "completion_tokens_details": {
      "accepted_prediction_tokens": 0,
      "audio_tokens": 0,
      "reasoning_tokens": 0,
      "rejected_prediction_tokens": 0
    },
    "prompt_tokens_details": {
      "audio_tokens": 0,
      "cached_tokens": 0
    }
    }
}
```

In [15]:
```python
response_message = response.choices[0].message

available_functions = {
    "get_current_weather": get_current_weather,
}
function_name = response_message.function_call.name
fuction_to_call = available_functions[function_name]
function_args = json.loads(response_message.function_call.arguments)

function_response = fuction_to_call(
    location=function_args.get("location"),
    unit=function_args.get("unit"),
)

print(function_response)
```

```
{"location": "Seoul", "temperature": "25", "unit": "celsius", "forecast": ["sunny", "windy"]}
```

In [16]:
```python
messages.append(response_message)
messages.append(
    {
        "role": "function",
        "name": function_name,
        "content": function_response,
    }
)
```

In [17]:
```python
print(messages)
```

```
[{'role': 'user', 'content': "What's the weather like in Seoul?"}, ChatCompletionMessage(content=None, refusal=None, role='assi
stant', annotations=[], audio=None, function_call=FunctionCall(arguments='{"location":"Seoul","unit":"celsius"}', name='get_cur
rent_weather'), tool_calls=None), {'role': 'function', 'name': 'get_current_weather', 'content': '{"location": "Seoul", "temper
ature": "25", "unit": "celsius", "forecast": ["sunny", "windy"]}'}]
```

In [18]:
```python
second_response = client.chat.completions.create(
    model="gpt-4.1-mini",
    messages=messages,
)
```

```
print(second_response.model_dump_json(indent=2))
```

```
{
  "id": "chatcmpl-CxlgjqxPcjfPwUPTNIbdzdmeB7nge",
  "choices": [
    {
      "finish_reason": "stop",
      "index": 0,
      "logprobs": null,
      "message": {
        "content": "The weather in Seoul is currently sunny and windy, with a temperature of 25°C.",
        "refusal": null,
        "role": "assistant",
        "annotations": [],
        "audio": null,
        "function_call": null,
        "tool_calls": null
      }
    }
  ],
  "created": 1768360701,
  "model": "gpt-4.1-mini-2025-04-14",
  "object": "chat.completion",
  "service_tier": "default",
  "system_fingerprint": "fp_376a7ccef1",
  "usage": {
    "completion_tokens": 18,
    "prompt_tokens": 76,
    "total_tokens": 94,
    "completion_tokens_details": {
      "accepted_prediction_tokens": 0,
      "audio_tokens": 0,
      "reasoning_tokens": 0,
      "rejected_prediction_tokens": 0
    },
    "prompt_tokens_details": {
      "audio_tokens": 0,
      "cached_tokens": 0
    }
  }
}
```

```
In [ ]:
```