ExamplesGemini

September 25, 2025

1 Using Google Gemini models through the Gemini API

[ ]:
```python
#!pip install -q -U google-genai
```

[10]:
```python
import os
from google import genai
from dotenv import load_dotenv, find_dotenv

# Read the local .env file, containing the Gemini secret API key.
_ = load_dotenv(find_dotenv())

client = genai.Client(api_key = os.environ["GEMINI_API_KEY"])
```

1.1 Conversational API Example

[11]:
```python
query = "Justin sits next to Razvan. One of them is happy and one of them is␣
↪grumpy. " \
                "The person sitting next to Justin is grumpy. Who is happy?"

response = client.models.generate_content(
    model = "gemini-2.5-flash",
    contents = query
)
print(response.text)
```

```
Let's break it down:

1.  Justin sits next to Razvan.
2.  The person sitting next to Justin is grumpy.
3.  Since Justin sits next to Razvan, the person sitting next to Justin *is*
Razvan.
4.  Therefore, **Razvan is grumpy**.
5.  We know one of them is happy and one is grumpy. Since Razvan is grumpy,
**Justin must be happy**.

**Justin is happy.**
```

1.2 Simple two-turn conversation

[12]:
```python
chat = client.chats.create(model = "gemini-2.5-flash")

response = chat.send_message("Who received the Nobel prize in medicine for␣
  ↪cancer immunotherapy?")
# print(response.text)

response = chat.send_message("Where did they do their research?")
# print(response.text)

for message in chat.get_history():
    print(f'role - {message.role}:', end = '\n')
    print('\t')
    print(message.parts[0].text, end = '\n')
    print('\n')
```

```
role - user:

Who received the Nobel prize in medicine for cancer immunotherapy?


role - model:

The Nobel Prize in Physiology or Medicine in 2018 was awarded to **James P.
Allison** and **Tasuku Honjo** for their discovery of cancer therapy by
inhibition of negative immune regulation. This is the groundbreaking work behind
cancer immunotherapy, specifically immune checkpoint blockade.


role - user:

Where did they do their research?


role - model:

They conducted their Nobel Prize-winning research primarily at the following
institutions:

*   **James P. Allison:** Much of his foundational work on **CTLA-4** and
demonstrating its potential for cancer treatment (through immune checkpoint
blockade) was performed at the **University of California, Berkeley** in the
late 1980s and 1990s. He later moved to Memorial Sloan Kettering Cancer Center
and then to MD Anderson Cancer Center, where he continued to advance the field
and work on its clinical application.

*   **Tasuku Honjo:** He discovered **PD-1** and elucidated its function as an
```

```
inhibitory receptor on T-cells at **Kyoto University** in Japan, starting in the
early 1990s. He has remained a prominent figure at Kyoto University throughout
his career.
```

1.3 Sequence Completion Example

[13]:
```python
question = "Provide the next number in the sequence 1, 2, 3, 5, 5, 8, 7, 11, 9,␣
 ↪..."

response = client.models.generate_content(
    model = "gemini-2.5-flash",
    contents = question
)
print(response.text)
```

```
This sequence is actually two interleaved arithmetic progressions:

*   **Odd positions (1st, 3rd, 5th, 7th, 9th):** 1, 3, 5, 7, 9, …
    This sequence adds 2 each time. (1+2=3, 3+2=5, 5+2=7, 7+2=9)

*   **Even positions (2nd, 4th, 6th, 8th):** 2, 5, 8, 11, …
    This sequence adds 3 each time. (2+3=5, 5+3=8, 8+3=11)

We need the 10th number in the sequence, which is an even position. So, we
continue the second sequence:
11 + 3 = **14**

The next number in the sequence is **14**.
```

1.4 Image Understanding example: captioning

[14]:
```python
from google.genai import types

with open('roxby.jpg', 'rb') as f:
    image_bytes = f.read()

    response = client.models.generate_content(
        model = 'gemini-2.5-flash',
        contents = [
            types.Part.from_bytes(
            data = image_bytes,
            mime_type = 'image/jpeg'),
          'Caption this image.'
        ])

print(response.text)
```

```
A young girl with her hair in a high ponytail and wearing a Hello Kitty tank top
is engrossed in coloring an activity sheet at a restaurant. She holds a blue
crayon and has a blue smiley face sticker on her arm as she focuses on an
underwater scene with a shark, turtle, and mermaid. Crayons and restaurant
menus, including one for "Boardwalk Billy's," are scattered across the wooden
table in front of her.
```

1.5 Image Understanding example: conversation

[16]:
```python
from google.genai import types

with open('pump.jpg', 'rb') as f:
    image_bytes = f.read()

    response1 = client.models.generate_content(
        model = 'gemini-2.5-pro',
        contents = [
            types.Part.from_bytes(data = image_bytes, mime_type = 'image/jpeg'),
            'What is in this image?'
        ])

print(response1.text)
```

```
This is a close-up image of a gas pump at a gas station.

The central feature is the digital display, which shows a total cost of
**$27.17** and **10.403 gallons** of fuel dispensed. Reflected in the screen is
the person taking the picture, a man with glasses and a beard holding a phone.

The pump is covered in various stickers and advertisements:
*   **Promotions:** A large blue and yellow ad at the top promotes a deal to
"Save 10¢ a gallon with earnify & Amazon Prime." Smaller stickers for "earnify"
```

```
with QR codes are also visible.
*   **Identification:** The pump is marked with a large green number **5**.
Small BP and Amoco logos are present.
*   **Official Seal:** On the left, there is a 2025 approval seal from the North
Carolina Department of Agriculture & Consumer Services.
*   **Warnings and Instructions:** Several notices are posted, including a
prominent sticker that reads "DO NOT USE PHONE WHILE REFUELING" and a yellow
"WARNING" sign with pictograms prohibiting smoking, leaving the car running, and
using a cell phone. There is also information about refueling services for
motorists with disabilities.
*   **Payment:** Logos for credit cards like Discover, Visa, Mastercard, and
American Express are shown near the card slot area.
```

[17]:
```python
response2 = client.models.generate_content(
        model = 'gemini-2.5-pro',
        contents = [
        genai.types.Content(
            role = "user",
            parts = [
                    types.Part.from_bytes(data = image_bytes, mime_type =␣
  ↪'image/jpeg'),
                    types.Part(text = "What is in this image?"),
                    # genai.types.Part(inline_data=image_data)
                ]),
        genai.types.Content(
            role = "model",
            parts = [types.Part(text = response1.text)]),
        genai.types.Content(
            role = "user",
            parts = [types.Part(text = "What is the price per gallon at this␣
  ↪gas station?")])
    ]
)
print(response2.text)
```

```
Based on the information on the gas pump's digital display, we can calculate the
price per gallon.

*   **Total Cost:** $27.17
*   **Gallons Dispensed:** 10.403

To find the price per gallon, you divide the total cost by the number of
gallons:

$27.17 / 10.403 gallons = **$2.612 per gallon** (approximately)
```

[22]:
```python
with open('horse.png', 'rb') as f:
    image_bytes = f.read()
    response = client.models.generate_content(
        model = 'gemini-2.5-pro',
        contents = [
            types.Part.from_bytes(data = image_bytes, mime_type = 'image/png'),
            'Provide a short, humorous caption for this image.'
        ])

print(response.text)
```

```
Here are a few short, humorous captions for the image:

*   **The original self-driving vehicle.**
*   **1 horsepower autopilot.**
*   **Don't worry, the horse knows the way home.**
*   **Best designated driver ever.**
```

1.6 Image Understanding + Reasoning example

For large files or to be able to use the same image file repeatedly, we can upload images using the File upload API.

[18]:
```
math_file = client.files.upload(file = "math.png")

response3 = client.models.generate_content(
    model = "gemini-2.5-flash",
    contents = [math_file, "Extract the text from this image."],
)

print(response3.text)
```

```
A stock went down 60% to $17.80. What percent does it need to rise to get back
to it's original price?
```

[19]:
```
from google.genai import types

response4 = client.models.generate_content(
        model = 'gemini-2.5-pro',
        contents = response3.text,
#        config = types.GenerateContentConfig(
#            temperature = 0,
#            candidate_count = 1, # default
#            max_output_tokens = 500)
)

print(response4.text)
```

```
Excellent question! This is a classic trick question where the answer isn't
simply 60%.
```

```
The stock needs to rise **150%** to get back to its original price.

Here is the step-by-step breakdown:

### Step 1: Find the original price.
Let the original price be "P". A 60% decrease means the stock retained 40% of
its original value (100% - 60% = 40%).

*   Original Price (P) × 0.40 = $17.80
*   P = $17.80 / 0.40
*   P = **$44.50**

So, the original price of the stock was $44.50.

### Step 2: Calculate the required increase.
To get from the new price ($17.80) back to the original price ($44.50), it needs
to increase by:

*   $44.50 - $17.80 = **$26.70**

### Step 3: Calculate the percentage increase.
The percentage increase is calculated based on the *current price* ($17.80), not
the original price.

*   (Amount of Increase / Current Price) × 100
*   ($26.70 / $17.80) × 100
*   1.5 × 100 = **150%**

**Why isn't it 60%?**
The 60% decrease was calculated from a higher original price ($44.50). The
percentage increase required to get back to that price is calculated from a much
lower starting point ($17.80), so a much larger percentage is needed.
```

1.7 Code execution tool

[20]:
```python
from google import genai
from google.genai.types import (
#    HttpOptions,
    Tool,
    ToolCodeExecution,
    GenerateContentConfig,
)

#client = genai.Client(http_options = HttpOptions(api_version="v1"))
model_id = "gemini-2.5-flash"
```

```
code_execution_tool = Tool(code_execution = ToolCodeExecution())
response = client.models.generate_content(
    model = model_id,
    contents = "Calculate 20th fibonacci number. Then find the nearest␣
 ↪palindrome to it.",
    config = GenerateContentConfig(
        tools = [code_execution_tool],
        temperature = 0,
    ),
)
print("# Code:")
print(response.executable_code)
print("# Outcome:")
print(response.code_execution_result)
```

```
# Code:
def fibonacci(n):
    a, b = 0, 1
    for _ in range(n - 1):
        a, b = b, a + b
    return b

fib_20 = fibonacci(20)
print(f"The 20th Fibonacci number is: {fib_20}")

# Outcome:
The 20th Fibonacci number is: 6765
```

[ ]: