

Gemini 3 Flash Preview

Our agentic workhorse model, bringing near Pro agentic, coding and multimodal intelligence, with more balanced cost and speed.


Otwórz w Vertex AI Studio

Wyświetl kod

Latest

Overview

Documentation

 Udostępniona przepustowość jest dostępna dla Gemini 3 Flash Preview

Zadbaj o zadania generatywnej AI dla obsługiwanych modeli w Vertex AI. [Więcej informacji](#)

[Zarządzaj zamówieniami](#)

[Złóż zamówienie](#)

Identyfikator model
publishers/google/r
3-flash-preview

Nazwa wersji
google/gemini-3-flas

Latest

Gemini 3 Flash Preview is designed to deliver strong agentic capabilities (near-Pro level) at substantial speed and value. Making it perfect for engaging multi-turn chats, and collaborating back and forth with your coding agent without getting out of flow. Compared to 2.5 Flash it delivers significant improvements across the board.

Gemini 3 models are thinking models, capable of reasoning through their thoughts before responding, resulting in enhanced performance and improved accuracy.

The Gemini 3 Flash Preview model is now available on [Vertex AI](#).

Gemini 3 Flash Preview is also available through our [Gen AI SDK](#) that provides a unified interface for Google AI Studio and Vertex AI in Python, Go.

For more information about differences with the previous model version, see [Model versions and lifecycle](#).

Tagi

Działanie

Generowanie

Wyodrębnianie

Rozpoznawanie

Overview

Gemini 3 Flash Preview is our most powerful agentic and coding model. It features a 1M token context window with the best multimodal understanding capabilities.

Model details

The model details are as follows:

Property	Description
Model name	gemini-3-flash-preview
Supported data types	Inputs: audio, images, video, text, and PDF Output: text
Token limits	Input token limit: 1M Output token limit: 64k

Feature support

These capabilities are available in the Gemini 3 Flash Preview model:

- Thinking mode with thinking levels minimal, low, medium or high

- Grounding with Google Search (Search as a Tool)
- Vertex AI RAG Engine
- URL Context
- Code Execution as a Tool
- Structured Output
- Context Caching
- Implicit Caching
- Batch Prediction
- Provisioned Throughput

The following capability isn't available in the Gemini 3 Flash Preview model:

- Tuning

Main API changes compared to Gemini 2.5 Flash

Gemini 3 Flash is launched with the following API changes compared to Gemini 2.5 Flash:

- Missing [thought signature](#) in the first function calling part of multi-step calls results in 400 errors. This stricter validation is introduced to ensure model maintains context across turns like remembering why it called the tool. Note that thought signature must be returned irrespective of the thinking level used.
- Replacing thinking budget with [thinking level](#) parameter offering 4 states:
 - **MINIMAL (Gemini 3 Flash only):** Constrains the model to use as few tokens as possible for thinking and is best used for low-complexity tasks that wouldn't benefit from extensive reasoning. MINIMAL is as close as possible to a zero budget for thinking but still requires thought signatures.
 - **LOW:** Constrains the model to use fewer tokens for thinking and is suitable for simpler tasks where extensive reasoning is not required. LOW is ideal for high-throughput tasks where speed is essential.
 - **MEDIUM (Gemini 3 Flash only):** Offers a balanced approach suitable for tasks of moderate complexity that benefit from reasoning but don't require deep, multi-step planning. It provides more reasoning capability than LOW while maintaining lower latency than HIGH**.**
 - **HIGH:** Allows the model to use more tokens for thinking and is suitable for complex prompts requiring deep reasoning, such as multi-step planning, verified code generation, or advanced function calling scenarios. This is the default level for Gemini 3 Flash. Use this configuration when replacing tasks you might have previously relied on specialized reasoning models for.

Note: You cannot use both `thinking_level` and the legacy `thinking_budget` parameter in the same request. Doing so will return a 400 error

- Introducing [Multimodal function response](#) and [Streaming function calling](#)
- Returning PDF token counts under modality.IMAGE instead of modality.DOCUMENT in [usage metadata](#) of the model response
- Changing [media resolution](#) defaults and mapping of enums (low, medium, high) across [image](#), [video](#) and [document](#) modalities

	Tokens		
	Image	Video	PDF
MEDIA_RESOLUTION_UNSPECIFIED (DEFAULT)	1120	70	560
MEDIA_RESOLUTION_LOW	280	70	280
MEDIA_RESOLUTION_MEDIUM	560	70	560
MEDIA_RESOLUTION_HIGH	1120	280	1120
MEDIA_RESOLUTION_ULTRA_HIGH	2240		

Other models

The Gemini model family has multiple model sizes and capabilities. View the other Gemini model options:

Model name	Input data	Output data	Launch stage	Description
Gemini 3 Pro Preview	Audio, images, video, text, and PDF	Text	Preview	Our most powerful agentic and coding model, with the best multimodal capabilities.

Gemini 2.5 Pro	Audio, images, video, text, and PDF	Text	GA	Strongest 2.5 model, especially suitable for code and world knowledge
Gemini 2.5 Flash	Audio, images, video, text, and PDF	Text	GA	Best for balancing reasoning and speed.
Gemini 2.5 Flash-Lite	Audio, images, video, text, and PDF	Text	GA	Our cost effective offering to support high throughput.

Documentation

Get started

You can use Gemini 3 Flash Preview in Vertex AI Studio or use the API to integrate the model in your application.

Before you begin

Enable the [Vertex AI API](#).

For more information on getting set up on Google Cloud, see [Get set up on Google Cloud](#).

Try Gemini 3 Flash Preview in Vertex AI Studio (console)

To use Gemini 3 Flash Preview in Vertex AI Studio, click [Open Vertex AI Studio](#). In Vertex AI Studio, you can write a prompt then click Submit to view the output generated by Gemini 3 Flash Preview.

Try Gemini 3 Flash Preview (curl)

These instructions don't apply if you're using express mode for Google Cloud. If you're using express mode, follow the instructions for express mode instead.

To use Gemini 3 Flash Preview with the command line interface (CLI), do the following:

- 1. Open [Cloud Shell](#) or a local terminal window with the [gcloud CLI](#) installed.
- 2. Configure environment variables by entering the following. Replace YOUR_PROJECT_ID with the ID of your Google Cloud project.

```
MODEL_ID="gemini-3-flash-preview"
PROJECT_ID="YOUR_PROJECT_ID"
```

- 3. Send a prompt request by entering the following curl command:

```
curl \
-X POST \
-H "Authorization: Bearer $(gcloud auth application-default print-access-token)" \
-H "Content-Type: application/json" \
https://aiplatform.googleapis.com/v1/projects/${PROJECT_ID}/locations/global/publishers/google/models/${
  "contents": {
    "role": "user",
    "parts": [
      {
        "fileData": {
          "mimeType": "image/png",
          "fileUri": "gs://generativeai-downloads/images/scones.jpg"
        }
      },
      {
        "text": "Describe this picture."
      }
    ]
  }
}
```

For more information, see the [Gemini API reference](#).

Try Gemini 3 Flash Preview while using express mode (curl)

Follow these instructions only if you're using express mode for Google Cloud.

To use Gemini 3 Flash Preview with the command line interface (CLI), do the following:

1. Open [Cloud Shell](#) or a local terminal window with the [gcloud CLI](#) installed.
2. Configure environment variables by entering the following. Replace YOUR_API_KEY with the API key that you created for using express mode.

```
MODEL_ID="gemini-3-flash-preview"
API_KEY="YOUR_API_KEY"
```



3. Send a prompt request by entering the following curl command:

```
curl \
-X POST \
-H "Content-Type: application/json" \
https://aiplatform.googleapis.com/v1/publishers/google/models/${MODEL_ID}:streamGenerateContent?key=${API_KEY} \
${'{"contents": [{"role": "user", "parts": [{"fileData": {"mimeType": "image/png", "fileUri": "gs://generativeai-downloads/images/scones.jpg"}], "text": "Describe this picture."}]}'}'
```



For more information, see the [Gemini API reference](#).

Try Gemini 3 Flash Preview (Python)

These instructions don't apply if you're using express mode for Google Cloud. If you're using express mode, follow the instructions for express mode instead.

Before trying this sample, follow the Python setup instructions in the [Google Gen AI SDK quickstart using client libraries](#).

To authenticate to Vertex AI, set up Application Default Credentials. For more information, see [Set up authentication for a local development environment](#).

1. Install or update the [Google Gen AI SDK](#). You can also install it in a virtual environment.

```
pip3 install --upgrade --user google-genai
```



2. Send a prompt request. Replace YOUR_PROJECT_ID with your Google Cloud project ID.

```
from google import genai
from google.genai import types
client = genai.Client(
    vertexai=True, project="YOUR_PROJECT_ID", location="global",
)
# If your image is stored in Google Cloud Storage, you can use the from_uri class method to create a Part
IMAGE_URI = "gs://generativeai-downloads/images/scones.jpg"
model = "gemini-3-flash-preview"
response = client.models.generate_content(
    model=model,
    contents=[
        "What is shown in this image?",
        types.Part.from_uri(
            file_uri=IMAGE_URI,
            mime_type="image/png",
        ),
    ],
)
print(response.text, end="")
```



For more information, see the [Gemini SDK reference](#).

Try Gemini 3 Flash Preview while using express mode (Python)

Follow these instructions only if you're using express mode for Google Cloud.

Before trying this sample, follow the Python setup instructions in the [Google Gen AI SDK quickstart using client libraries](#).

1. Install or update the [Google Gen AI SDK](#). You can also install it in a virtual environment.

```
pip3 install --upgrade --user google-genai
```



2. Send a prompt request. Replace YOUR_API_KEY with your API key.

```
from google import genai
from google.genai import types
client = genai.Client(
    vertexai=True, api_key="YOUR_API_KEY"
)
# If your image is stored in Google Cloud Storage, you can use the from_uri class method to create a Part
IMAGE_URI = "gs://generativeai-downloads/images/scones.jpg"
model = "gemini-3-flash-preview"
response = client.models.generate_content(
    model=model,
    contents=[
        "What is shown in this image?",
        types.Part.from_uri(
            file_uri=IMAGE_URI,
            mime_type="image/png",
        ),
    ],
)
print(response.text, end="")
```



For more information, see the [Gemini SDK reference](#).