Getting Started

The Hitchhiker's Guide to Grok

Welcome! In this guide, we'll walk you through the basics of using the xAI API.

Step 1: Create an xAI Account

First, you'll need to create an xAI account to access xAI API. Sign up for an account here.

Once you've created an account, you'll need to load it with credits to start using the API.

Step 2: Generate an API Key

Create an API key via the API Keys Page in the xAI API Console.

After generating an API key, we need to save it somewhere safe! We recommend you export it as an environment variable in your terminal or save it to a `.env` file.

```bash
export XAI_API_KEY="your_api_key"
```

Step 3: Make your first request

With your xAI API key exported as an environment variable, you're ready to make your first API request.

Let's test out the API using `curl`. Paste the following directly into your terminal.

```bash
curl https://api.x.ai/v1/chat/completions \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $XAI_API_KEY" \
```

```
-m 3600 \
-d '{
    "messages": [
        {
            "role": "system",
            "content": "You are Grok, a highly intelligent, helpful AI assistant."
        },
        {
            "role": "user",
            "content": "What is the meaning of life, the universe, and everything?"
        }
    ],
    "model": "grok-4",
    "stream": false
}'
```

Step 4: Make a request from Python or Javascript

As well as a native xAI Python SDK, the majority our APIs are fully compatible with the OpenAI and Anthropic SDKs. For example, we can make the same request from Python or Javascript like so:

python (xAI SDK)

```python
# In your terminal, first run:
# pip install xai-sdk

import os

from xai_sdk import Client
from xai_sdk.chat import user, system

client = Client(
    api_key=os.getenv("XAI_API_KEY"),
    timeout=3600,  # Override default timeout with longer timeout for reasoning models
)

chat = client.chat.create(model="grok-4")
chat.append(system("You are Grok, a highly intelligent, helpful AI assistant."))
chat.append(user("What is the meaning of life, the universe, and everything?"))

response = chat.sample()
print(response.content)
```

Certain models also support Structured Outputs, which allows you to enforce a schema for the LLM output.

For an in-depth guide about using Grok for text responses, check out our Chat Guide.

Step 5: Use Grok to analyze images

Certain grok models can accept both text AND images as an input. For example:

```python
import os

from xai_sdk import Client
from xai_sdk.chat import user, image

client = Client(
    api_key=os.getenv("XAI_API_KEY"),
    timeout=3600,  # Override default timeout with longer timeout for reasoning models
)

chat = client.chat.create(model="grok-4")
chat.append(
    user(
        "What's in this image?",
        image("https://science.nasa.gov/wp-content/uploads/2023/09/web-first-images-release.png")
    )
)

response = chat.sample()
print(response.content)
```

And voila! Grok will tell you exactly what's in the image:

> *"This image is a photograph of a region in space, specifically a part of the Carina Nebula, captured by the James Webb Space Telescope. It showcases a stunning view of interstellar gas and dust, illuminated by young, hot stars. The bright points of light are stars, and the colorful clouds are composed of various gases and dust particles. The image highlights the intricate details and beauty of star formation within a nebula."*

To learn how to use Grok vision for more advanced use cases, check out our Image Understanding Guide.

Monitoring usage

As you use your API key, you will be charged for the number of tokens used. For an overview, you can monitor your usage on the xAI Console Usage Page.

If you want a more granular, per request usage tracking, the API response includes a usage object that provides detail on prompt (input) and completion (output) token usage.

```
json

"usage": {
    "prompt_tokens":37,
    "completion_tokens":530,
    "total_tokens":800,
    "prompt_tokens_details": {
        "text_tokens":37,
        "audio_tokens":0,
        "image_tokens":0,
        "cached_tokens":8
    },
    "completion_tokens_details": {
        "reasoning_tokens":233,
        "audio_tokens":0,
        "accepted_prediction_tokens":0,
        "rejected_prediction_tokens":0
    },
    "num_sources_used":0
}
```

If you send requests too frequently or with long prompts, you might run into rate limits and get an error response. For more information, read Consumption and Rate Limits.

Next steps

Now you have learned the basics of making an inference on xAI API. Check out Models page to start building with one of our latest models.