Developer quickstart

Get up and running with the OpenAI API

Looking for ChatGPT? Head to chatgpt.com.

The OpenAI API provides a simple interface for developers to create an intelligence layer in their applications, powered by OpenAI's state of the art models. The Chat Completions endpoint powers ChatGPT and provides a simple way to take text as input and use a model like GPT-4 to generate an output.

Want to jump straight to the code?
Skip the quickstart and dive into the API reference.

This quickstart is designed to help get your local development environment setup and send your first API request. If you are an experienced developer or want to just dive into using the OpenAI API, the API reference of GPT guide are a great place to start. Throughout this quickstart, you will learn:

- How to setup your development environment
- How to install the latest SDKs
- Some of the basic concepts of the OpenAI API
- How to send your first API request

If you run into any challenges or have questions getting started, please join our developer forum.

Account setup

First, create an OpenAI account or sign in. Next, navigate to the API key page and "Create new secret key", optionally naming the key. Make sure to save this somewhere safe and do not share it with anyone.

Quickstart language selection

Select the tool or language you want to get started using the OpenAI API with.

curl [✓ Python] Node.js

Python is a popular programming language that is commonly used for data applications, web development, and many other programming tasks due to its ease of use. OpenAI provides a custom Python library which makes working with the OpenAI API in Python simple and efficient.

Step 1: Setting up Python

Install Python

To use the OpenAI Python library, you will need to ensure you have Python installed. Some computers come with Python pre-installed while others require that you set it up yourself. To test if you have Python installed, you can navigate to your Terminal or Command line:

- MacOS: **Open Terminal**: You can find it in the Applications folder or search for it using Spotlight (Command + Space).
- Windows: **Open Command Prompt**: You can find it by searching "cmd" in the start menu.

Next, enter the word `python` and then press return/enter. If you enter into the Python interpreter, then you have Python installed on your computer already and you can go to the next step. If you get an error message that says something like "Error: command python not found", you likely need to install Python and make it available in your terminal / command line.

To download Python, head to the official Python website and download the latest version. To use the OpenAI Python library, you need at least Python 3.7.1 or newer. If you are installing Python for the first time, you can follow the official Python installation guide for beginners.

Set up a virtual environment (optional)

Once you have Python installed, it is a good practice to create a virtual python environment to install the OpenAI Python library. Virtual environments provide a clean working space for your Python packages to be installed so that you do not have conflicts with other libraries you install for other projects. You are not required to use a virtual environment, so skip to step 3 if you do not want to set one up.

To create a virtual environment, Python supplies a built in venv module which provides the basic functionality needed for the virtual environment. Running the command below will create a virtual environment named "openai-env" inside the current folder you have selected in your terminal / command line:

```
python -m venv openai-env
```

Once you've created the virtual environment, you need to activate it. On Windows, run:

```
openai-env\Scripts\activate
```

On Unix or MacOS, run:

```
source openai-env/bin/activate
```

You should see the terminal / command line interface change slightly after you active the virtual environment, it should now show "openai-env" to the left of the cursor input section. For more details on working wit virtual environments, please refer to the official Python documentation.

Install the OpenAI Python library

Once you have Python 3.7.1 or newer installed and (optionally) set up a virtual environment, the OpenAI Python library can be installed. From the terminal / command line, run:

```
pip install --upgrade openai
```

Once this completes, running `pip list` will show you the Python libraries you have installed in your current environment, which should confirm that the OpenAI Python library was successfully installed.

Step 2: Set up your API key

Set up your API key for all projects (recommended)

The main advantage to making your API key accessible for all projects is that the Python library will automatically detect it and use it without having to write any code.

MacOS

1. **Open Terminal**: You can find it in the Applications folder or search for it using Spotlight (Command + Space).
2. **Edit Bash Profile**: Use the command `nano ~/.bash_profile` or `nano ~/.zshrc` (for newer MacOS versions) to open the profile file in a text editor.
3. **Add Environment Variable**: In the editor, add the line below, replacing `your-api-key-here` with your actual API key:

```
export OPENAI_API_KEY='your-api-key-here'
```

1. **Save and Exit**: Press Ctrl+O to write the changes, followed by Ctrl+X to close the editor.
2. **Load Your Profile**: Use the command `source ~/.bash_profile` or `source ~/.zshrc` to load the updated profile.
3. **Verification**: Verify the setup by typing `echo $OPENAI_API_KEY` in the terminal. It should display your API key.

Windows

1. **Open Command Prompt**: You can find it by searching "cmd" in the start menu.
2. **Set environment variable in the current session**: To set the environment variable in the current session, use the command below, replacing `your-api-key-here` with your actual API key:

```
setx OPENAI_API_KEY "your-api-key-here"
```

```
This command will set the OPENAI_API_KEY environment variable for the cur
```

1. **Permanent setup**: To make the setup permanent, add the variable through the system properties as follows:
   - Right-click on 'This PC' or 'My Computer' and select 'Properties'.

  - Click on 'Advanced system settings'.
  - Click the 'Environment Variables' button.
  - In the 'System variables' section, click 'New...' and enter OPENAI_API_KEY as the variable name and your API key as the variable value.

2. **Verification**: To verify the setup, reopen the command prompt and type the command below. It should display your API key: `echo %OPENAI_API_KEY%`

Set up your API key for a single project

If you only want your API key to be accessible to a single project, you can create a local `.env` file which contains the API key and then explicitly use that API key with the Python code shown in the steps to come.

Start by going to the project folder you want to create the `.env` file in.

> In order for your **.env** file to be ignored by version control, create a **.gitignore** file in the root of your project directory. Add a line with **.env** on it which will make sure your API key or other secrets are not accidentally shared via version control.

Once you create the `.gitignore` and `.env` files using the terminal or an integrated development environment (IDE), copy your secret API key and set it as the `OPENAI_API_KEY` in your `.env` file. If you haven't created a secret key yet, you can do so on the API key page.

The `.env` file should look like the following:

```
# Once you add your API key below, make sure to not share it with anyon

OPENAI_API_KEY=abc123
```

The API key can be imported by running the code below:

```python
from openai import OpenAI

client = OpenAI()
```

```
# defaults to getting the key using os.environ.get("OPENAI_API_KEY")
# if you saved the key under a different environment variable name, you can
# client = OpenAI(
#   api_key=os.environ.get("CUSTOM_ENV_NAME"),
# )
```

Step 3: Sending your first API request

Making an API request

After you have Python configured and set up an API key, the final step is to send a request to the OpenAI API using the Python library. To do this, create a file named `openai-test.py` using th terminal or an IDE.

Inside the file, copy and paste one of the examples below:

ChatCompletions

```
from openai import OpenAI
client = OpenAI()

completion = client.chat.completions.create(
  model="gpt-3.5-turbo",
  messages=[
    {"role": "system", "content": "You are a poetic assistant, skilled in explaining co
    {"role": "user", "content": "Compose a poem that explains the concept of recursion
  ]
)

print(completion.choices[0].message)
```

To run the code, enter `python openai-test.py` into the terminal / command line.

The Chat Completions example highlights just one area of strength for our models: creative ability. Explaining recursion (the programming topic) in a well formatted poem is something both the best developers and best poets would struggle with. In this case, `gpt-3.5-turbo` does it effortlessly.

Next steps

Now that you have made you first OpenAI API request, it is time to explore what else is possible:

- For more detailed information on our models and the API, see our GPT guide.
- Visit the OpenAI Cookbook for in-depth example API use-cases, as well as code snippets for common tasks.
- Wondering what OpenAI's models are capable of? Check out our library of example prompts.
- Want to try the API without writing any code? Start experimenting in the Playground.
- Keep our usage policies in mind as you start building.