## Function calling

Give models access to new functionality and data they can use to follow instructions and respond to prompts.

**Function calling** (also known as **tool calling**) provides a powerful and flexible way for OpenAI models to interface with external systems and access data outside their training data. This guide shows how you can connect a model to data and actions provided by your application. We'll show how to use function tools (defined by a JSON schema) and custom tools which work with free form text inputs and outputs.

### How it works

Let's begin by understanding a few key terms about tool calling. After we have a shared vocabulary for tool calling, we'll show you how it's done with some practical examples.

Tools - functionality we give the model

A **function** or **tool** refers in the abstract to a piece of functionality that we tell the model it has access to. As a model generates a response to a prompt, it may decide that it needs data or functionality provided by a tool to follow the prompt's instructions.

You could give the model access to tools that:

- Get today's weather for a location
- Access account details for a given user ID
- Issue refunds for a lost order

Or anything else you'd like the model to be able to know or do as it responds to a prompt.

When we make an API request to the model with a prompt, we can include a list of tools the model could consider using. For example, if we wanted the

model to be able to answer questions about the current weather somewhere in the world, we might give it access to a get_weather tool that takes location as an argument.

Tool calls - requests from the model to use tools

A **function call** or **tool call** refers to a special kind of response we can get from the model if it examines a prompt, and then determines that in order to follow the instructions in the prompt, it needs to call one of the tools we made available to it.

If the model receives a prompt like "what is the weather in Paris?" in an API request, it could respond to that prompt with a tool call for the get_weather tool, with Paris as the location argument.

Tool call outputs - output we generate for the model

A **function call output** or **tool call output** refers to the response a tool generates using the input from a model's tool call. The tool call output can either be structured JSON or plain text, and it should contain a reference to a specific model tool call (referenced by call_id in the examples to come).

To complete our weather example:

- The model has access to a get_weather **tool** that takes location as an argument.
- In response to a prompt like "what's the weather in Paris?" the model returns a **tool call** that contains a location argument with a value of Paris
- Our **tool call output** might be a JSON structure like {"temperature": "25", "unit": "C"}, indicating a current temperature of 25 degrees.

We then send all of the tool definition, the original prompt, the model's tool call, and the tool call output back to the model to finally receive a text response like:

The weather in Paris today is 25C.

Functions versus tools

- A function is a specific kind of tool, defined by a JSON schema. A function definition allows the model to pass data to your application, where your code can access data or take actions suggested by the model.
- In addition to function tools, there are custom tools (described in this guide) that work with free text inputs and outputs.
- There are also built-in tools that are part of the OpenAI platform. These tools enable the model to search the web, execute code, access the functionality of an MCP server, and more (if you're having difficulties with the links, click here).

### The tool calling flow

Tool calling is a multi-step conversation between your application and a model via the OpenAI API. The tool calling flow has five high level steps:

1. Make a request to the model with tools it could call
2. Receive a tool call from the model
3. Execute code on the application side with input from the tool call
4. Make a second request to the model with the tool output
5. Receive a final response from the model (or more tool calls)