Index

Python types

If you need a refresher about how to use Python type hints, check the first part of FastAPI's Python types intro [↪].

You can also check the mypy cheat sheet [↪].

In short (very short), you can declare a function with parameters like:

```
from typing import Optional

def type_example(name: str, formal: bool = False, intro: Optional[str] =
None):
    pass
```

And your editor (and **Typer**) will know that:

- `name` is of type `str` and is a required parameter.
- `formal` is a `bool` and is by default `False`.
- `intro` is an optional `str`, by default is `None`.

These type hints are what give you autocomplete in your editor and several other features.

Typer is based on these type hints.

Intro

This tutorial shows you how to use **Typer** with all its features, step by step.

Each section gradually builds on the previous ones, but it's structured to separate topics, so that you can go directly to any specific one to solve your specific CLI needs.

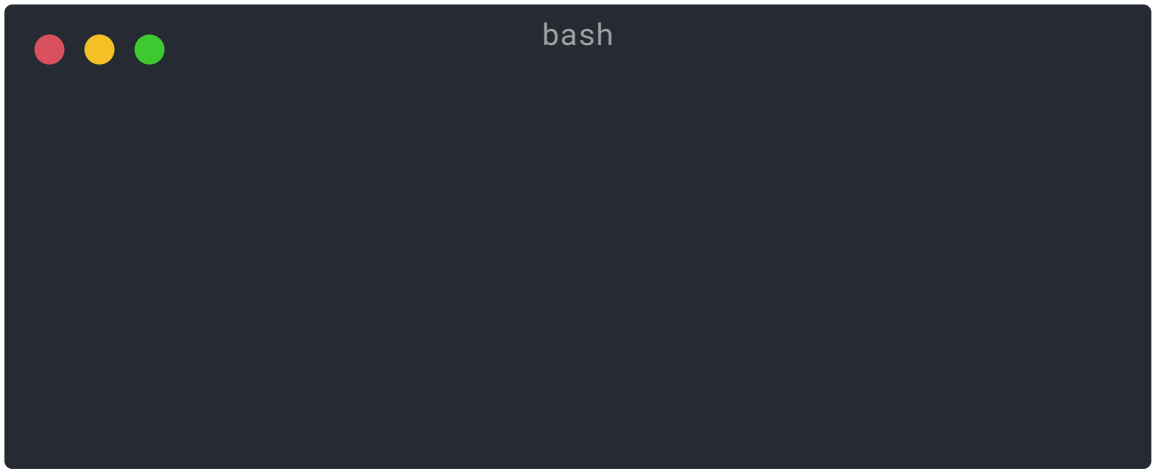
It is also built to work as a future reference.

So you can come back and see exactly what you need.

Run the code

All the code blocks can be copied and used directly (they are tested Python files).

To run any of the examples, copy the code to a file `main.py`, and run it:



It is **HIGHLY encouraged** that you write or copy the code, edit it and run it locally.

Using it in your editor is what really shows you the benefits of **Typer**, seeing how little code you have to write, all the type checks, autocompletion, etc.

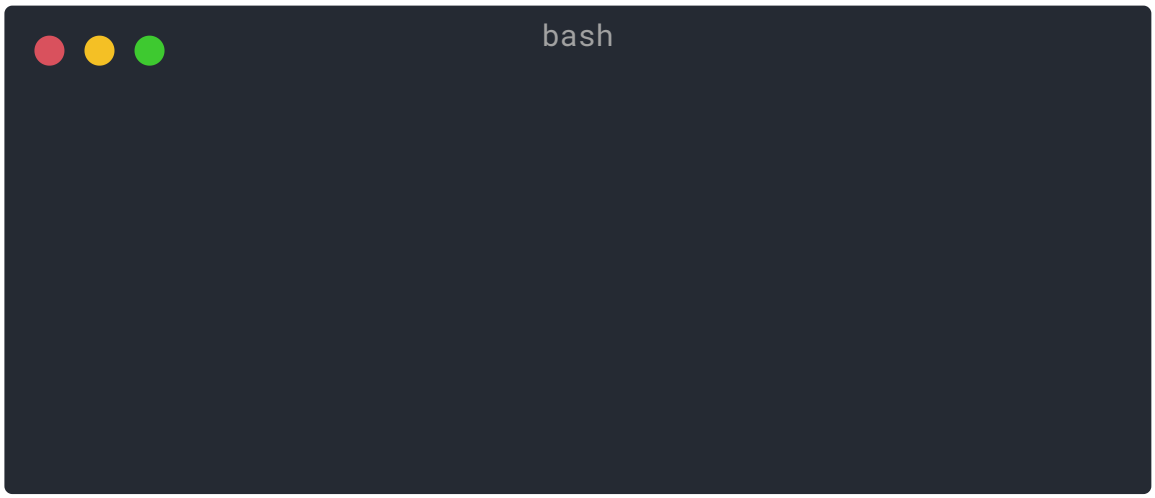
And running the examples is what will really help you understand what is going on.

You can learn a lot more by running some examples and playing around with them than by reading all the docs here.

Install **Typer**

The first step is to install **Typer**.

For the tutorial, you might want to install it with all the optional dependencies and features:



...that also includes `rich` and `shellingham`.