Installation

Fast API works only on and above python 3.6

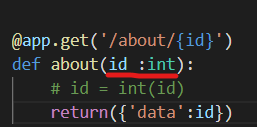
1. Pip install fastAPI
2. Pip install uvicorn

Setup

1. Create main.py file
2. Import FastAPI
3. Create the instance of FastAPI
   1. App = FastAPI()
4. Write a decorator
   1. @app.get(‘/’)
      1. app is a variable name act as FastAPI instance
      2. get is the operation that we need to perform on the function
      3. (‘/’) is the route on which we need to execute the function
   2. Slash part here is the URL address or the path
5. Write the function
6. To run server , In terminal write
   1. uvicorn main:app –reload
      1. this reload command is optional. It basically tracks every time we do some changes in the file and show those changes in server automatically

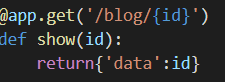
Path Parameters

We can defind id:int to define the id will get the integer type of value in the function. Example:

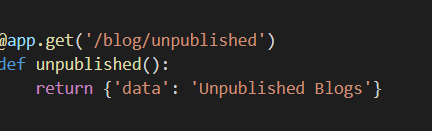


Make sure that static routing will come before dynamic routing for same path.

Example:

 here {id} is the dynamic part

This part will come after

 here unpublished is the static part

API Docs

SwaggerUI Doc :

<http://127.0.0.1:8000/docs#/>

ReDoc

<http://127.0.0.1:8000/redocs#/>

Request Body

We can use a lot of Request body like Get, Post.

We use Post request when we need to create something. Here to create the blog we will be using Post request

A **request** body is data sent by the client to your API. A **response** body is the data your API sends to the client

When you need to send data from a client (let's say, a browser) to your API, you send it as a **request body**.

To declare a **request** body, you use [Pydantic](https://pydantic-docs.helpmanual.io/" \t "_blank) models with all their power and benefits.

## Import Pydantic's BaseModel[¶](https://fastapi.tiangolo.com/tutorial/body/#import-pydantics-basemodel)

## First, you need to import BaseModel from pydantic:

from typing import Optional

from fastapi import FastAPI

from pydantic import BaseModel

class Item(BaseModel):

name: str

description: Optional[str] = None

price: float

tax: Optional[float] = None

app = FastAPI()

@app.post("/items/")

async def create\_item(item: Item):

return item

This is just like the view in Django. We are basically following the ORM model