Python] Быстрое создание простого HTTP-сервера (http.server)

вводная

http.server - это подкласс socketserver.TCPServer, который создает и прослушивает HTTP-сокеты и отправляет запросы обработчикам. Эта статья представляет собой краткое руководство по созданию простого HTTP-сервера с помощью блока модулей http.server в Python.

монтаж

Стандартный модуль, встроенный в Python3, не требующий установки. (В предыдущих версиях Python2 он назывался SimpleHTTPServer).

учебники

Создается в командной строке

Поддерживается прямой вызов http.server с аргументом -m в интерпретаторе Python.

Создайте минимальный HTTP-сервер, выполнив следующую команду:

python -m http.server

Порт прослушивания сервера по умолчанию - 8000. Поддерживаются пользовательские номера портов:

python -m http.server 9000

По умолчанию сервер привязывается ко всем интерфейсам, и вы можете указать адрес, например, локального хоста, с помощью команды -b/--bind:

python -m http.server --bind 127.0.0.1

По умолчанию рабочим каталогом сервера является текущий каталог. Вы можете указать рабочий каталог с помощью параметра -d/--directory:

python -m http.server --directory /tmp/

Кроме того, вы можете передать аргумент --cgi, чтобы включить  [CGI](https://baike.baidu.com/item/CGI/607810) обработчик запросов:

python -m http.server --cgi

Напишите код для создания

http.server также поддерживается для вызова из кода, просто импортируйте соответствующие классы и функции.

from http.server import SimpleHTTPRequestHandler

from http.server import CGIHTTPRequestHandler

from http.server import ThreadingHTTPServer

from functools import partial

import contextlib

import sys

import os

class DualStackServer(ThreadingHTTPServer):

def server\_bind(self):

# suppress exception when protocol is IPv4

with contextlib.suppress(Exception):

self.socket.setsockopt(socket.IPPROTO\_IPV6, socket.IPV6\_V6ONLY, 0)

return super().server\_bind()

def run(server\_class=DualStackServer,

handler\_class=SimpleHTTPRequestHandler,

port=8000,

bind='127.0.0.1',

cgi=False,

directory=os.getcwd()):

"""Run an HTTP server on port 8000 (or the port argument).

Args:

server\_class (\_type\_, optional): Class of server. Defaults to DualStackServer.

handler\_class (\_type\_, optional): Class of handler. Defaults to SimpleHTTPRequestHandler.

port (int, optional): Specify alternate port. Defaults to 8000.

bind (str, optional): Specify alternate bind address. Defaults to '127.0.0.1'.

cgi (bool, optional): Run as CGI Server. Defaults to False.

directory (\_type\_, optional): Specify alternative directory. Defaults to os.getcwd().

"""

if cgi:

handler\_class = partial(CGIHTTPRequestHandler, directory=directory)

else:

handler\_class = partial(SimpleHTTPRequestHandler, directory=directory)

with server\_class((bind, port), handler\_class) as httpd:

print(

f"Serving HTTP on {bind} port {port} "

f"(http://{bind}:{port}/) ..."

)

try:

httpd.serve\_forever()

except KeyboardInterrupt:

print("\nKeyboard interrupt received, exiting.")

sys.exit(0)

if \_\_name\_\_ == '\_\_main\_\_':

run(port=8000, bind='127.0.0.1')

server\_class: класс сервера

handler\_class: класс обработчика запроса

порт: порт

привязка: IP

cgi: включать или не включать обработчики CGI-запросов

каталог: рабочий каталог

реальный пример

Теперь, когда мы знаем, что http.server позволяет быстро создать HTTP-сервер, в каких сценариях проекта его можно применить?

Предварительный просмотр небольшого веб-проекта в локальной сети

* Каталог проектов

web:.

├─index.html

* index.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">.

<meta http-equiv="X-UA-Compatible" content="IE=edge">.

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>**Документ**</title

</head>

<body>

**мир приветствий**

</body>

</html>

**Перейдите в каталог cd web, выполните команду python -m http.server и введите localhost:8000 в адресную строку браузера, чтобы отобразить его:**

**мир приветствий**

**Другие пользователи локальной сети могут получить доступ к нему через IP-адрес хоста + номер порта, например, если ваш IP-адрес 192.168.0.1, то отправьте URL 192.168.0.1:8000 своим коллегам или одноклассникам, и они также смогут увидеть содержимое, отображаемое в файле index.html.**

**Сопоставление портов для доступа к удаленному серверу в локальном браузере**

**Если вы подключаетесь к удаленному серверу через VSCode, открытие порта с помощью http.server автоматически сопоставит его с локальным, чтобы вы могли просматривать и загружать ресурсы удаленного сервера в своем локальном браузере. (Другие инструменты, помимо VSCode, также должны уметь сопоставлять порты между удаленным и локальным).**

### **принимать к сведению**

http.server реализует только самые базовые проверки безопасности, поэтому не используйте его в производственной среде.

### **консультация**

<https://docs.python.org/3/library/http.server.html>