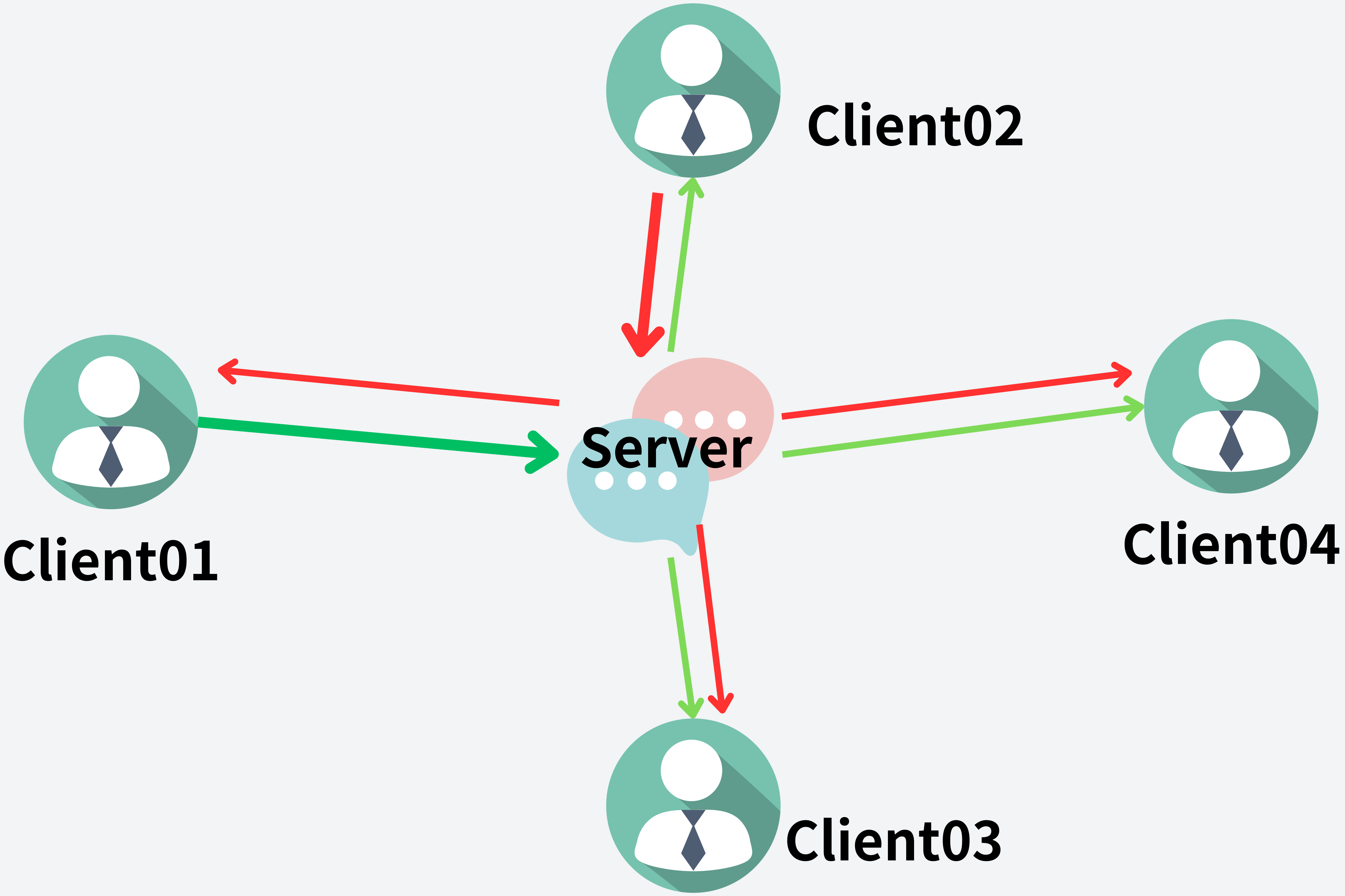
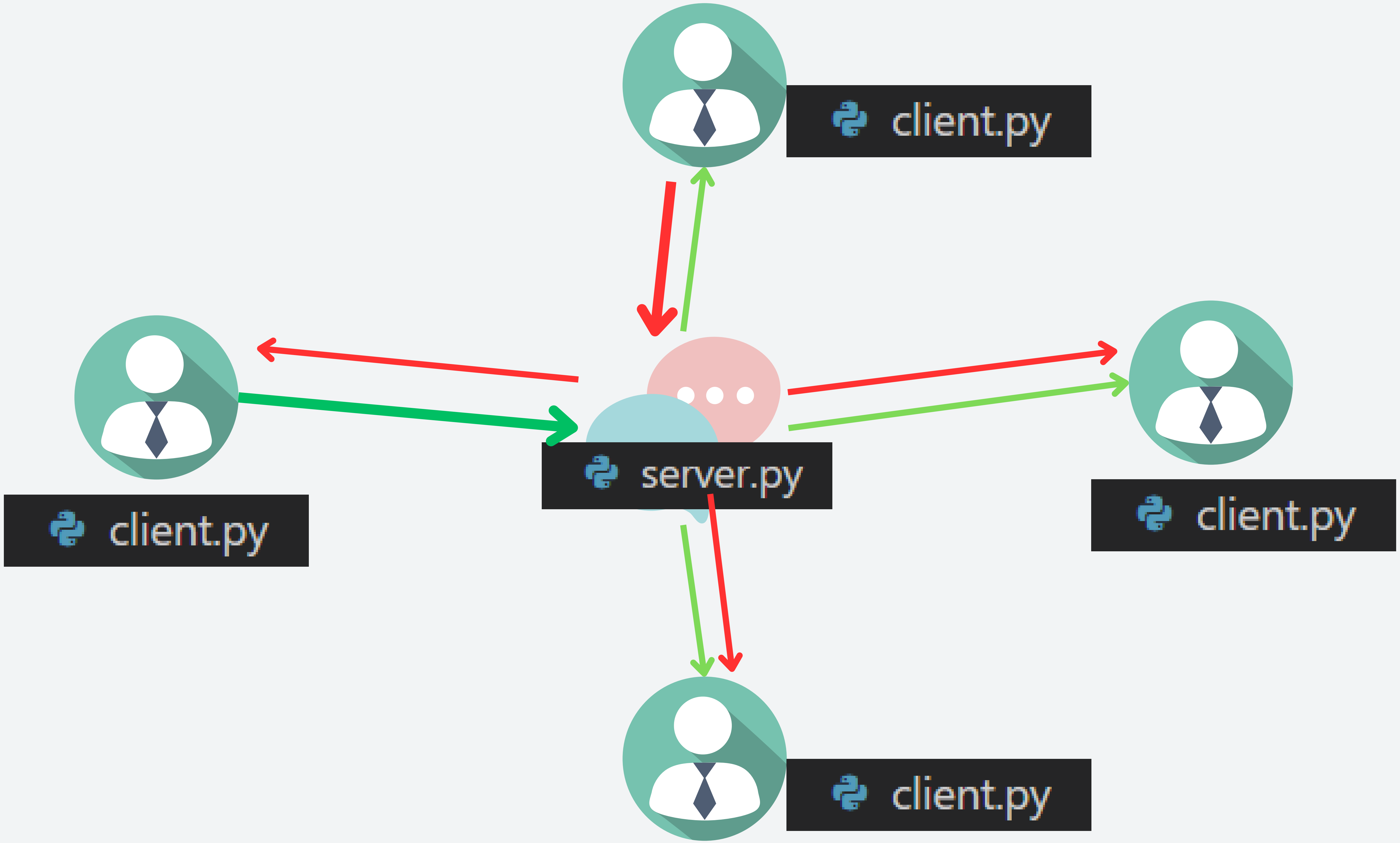
次世代無線網路概論

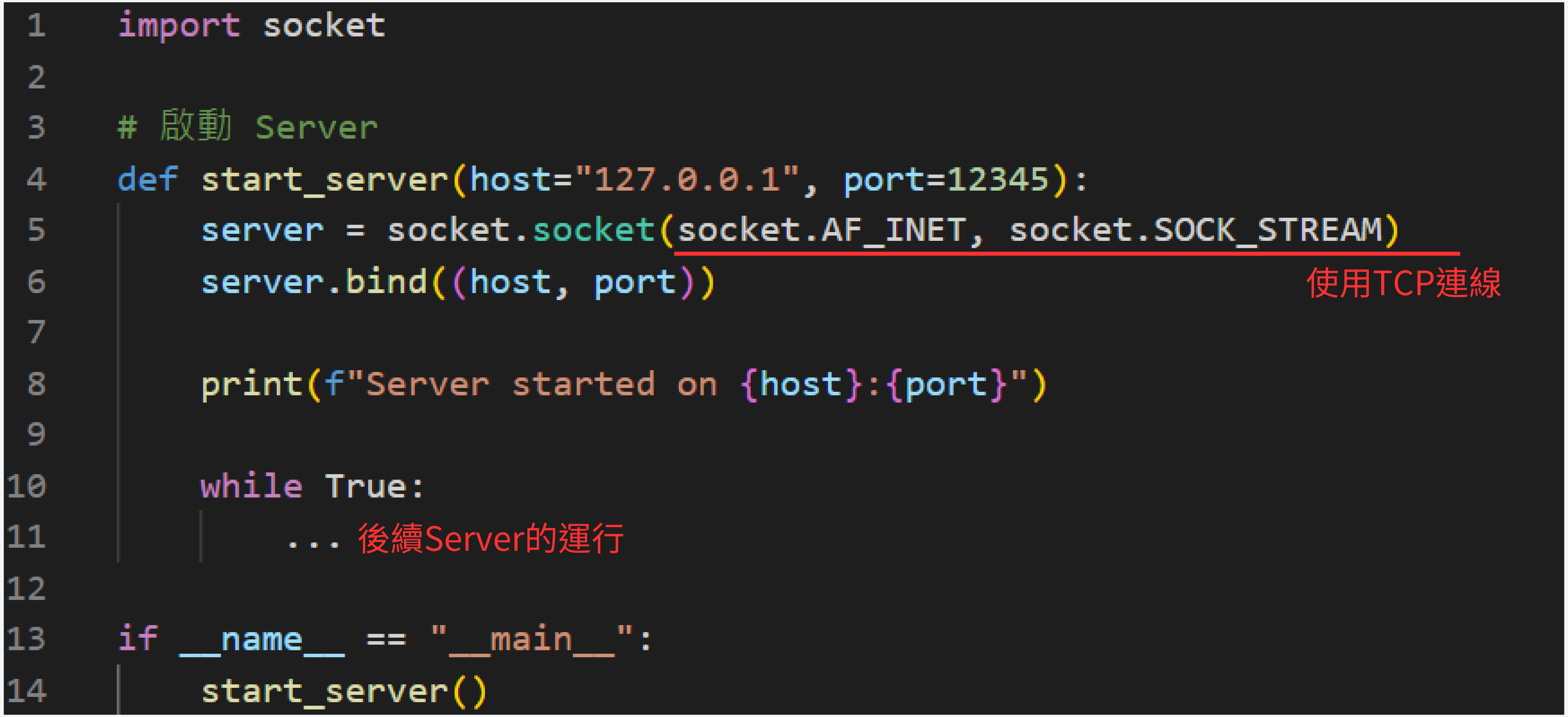
HW03-TUTORIAL

SOCKET PROGRAMMING - CHAT ROOM




client.py
client.py
server.py
client.py
client.py

Server Start

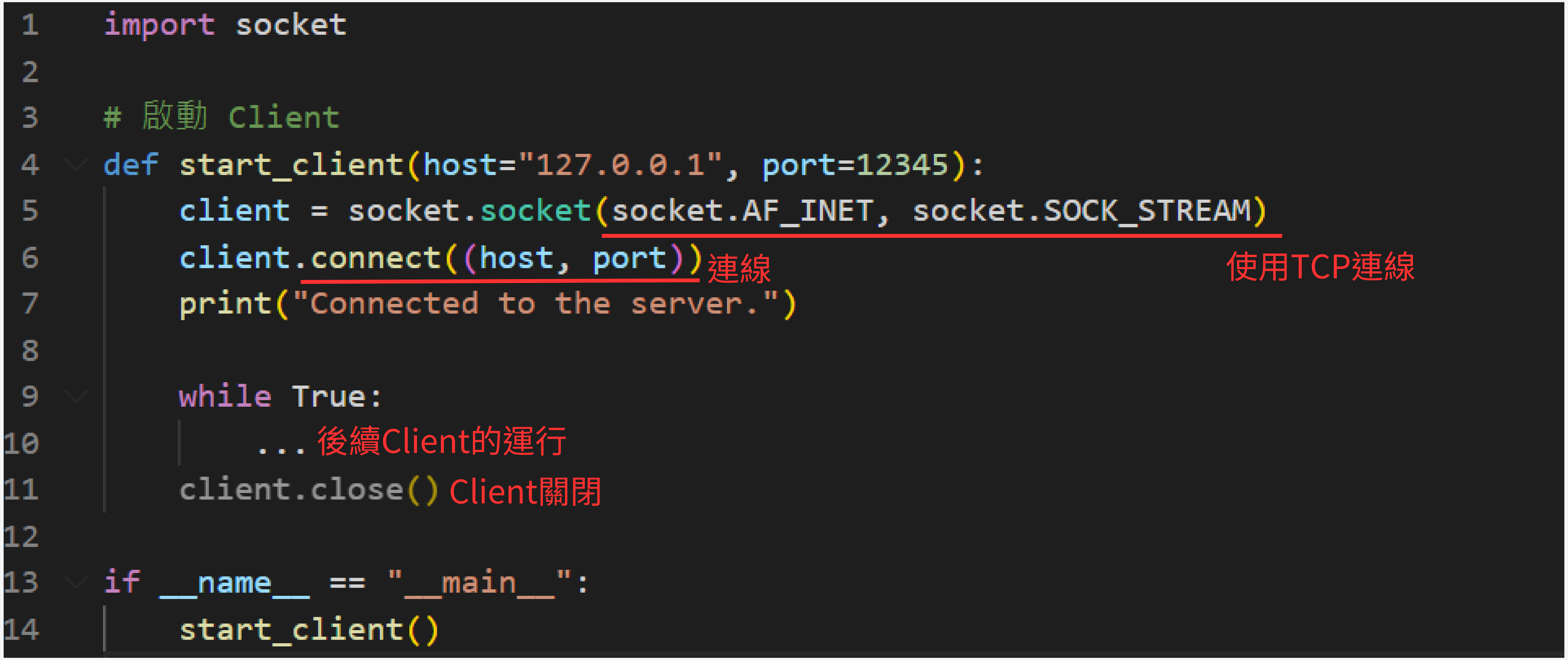


```python
import socket

# 啟動 Server
def start_server(host="127.0.0.1", port=12345):
    server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    server.bind((host, port))

    print(f"Server started on {host}:{port}")

    while True:
        ...

if __name__ == "__main__":
    start_server()
```

使用TCP連線

後續Server的運行

Client Connect



```python
import socket

# 啟動 Client
def start_client(host="127.0.0.1", port=12345):
    client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    client.connect((host, port))
    print("Connected to the server.")

    while True:
        ...
    client.close()

if __name__ == "__main__":
    start_client()
```

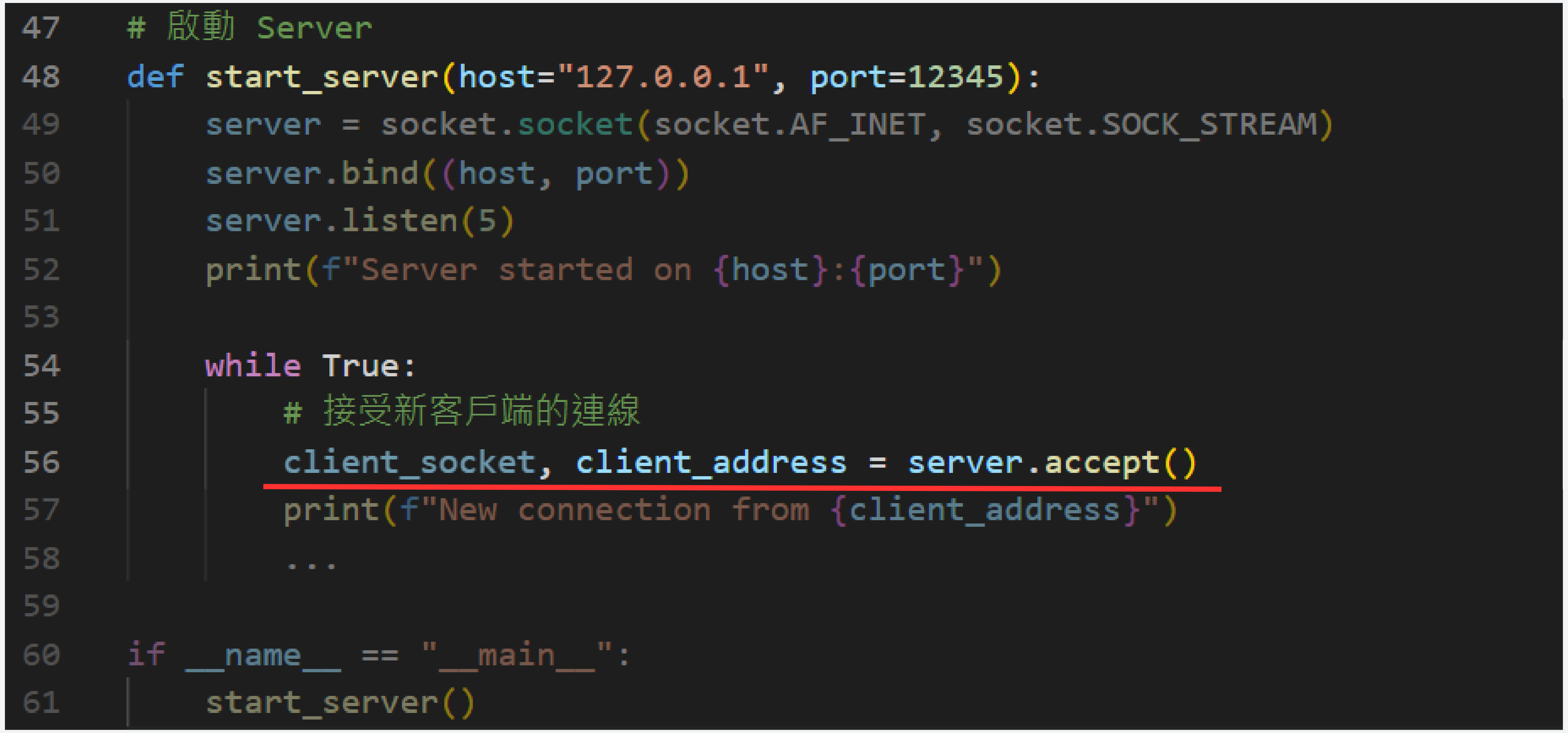
使用TCP連線

連線

後續Client的運行

Client關閉

Server accept Client



```python
# 啟動 Server
def start_server(host="127.0.0.1", port=12345):
    server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    server.bind((host, port))
    server.listen(5)
    print(f"Server started on {host}:{port}")

    while True:
        # 接受新客戶端的連線
        client_socket, client_address = server.accept()
        print(f"New connection from {client_address}")
        ...

if __name__ == "__main__":
    start_server()
```

accept function 會返回 client_socket & client_address

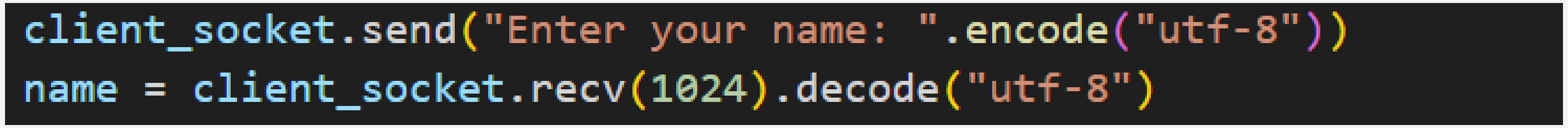
client_socket: 為client與server建立的socket ，可以用來 send() & recv()
client_address: 為tuple: (ip_address, port)

Send & Recv Message

Concept

1.透過 socket 對象進行 send(), recv()

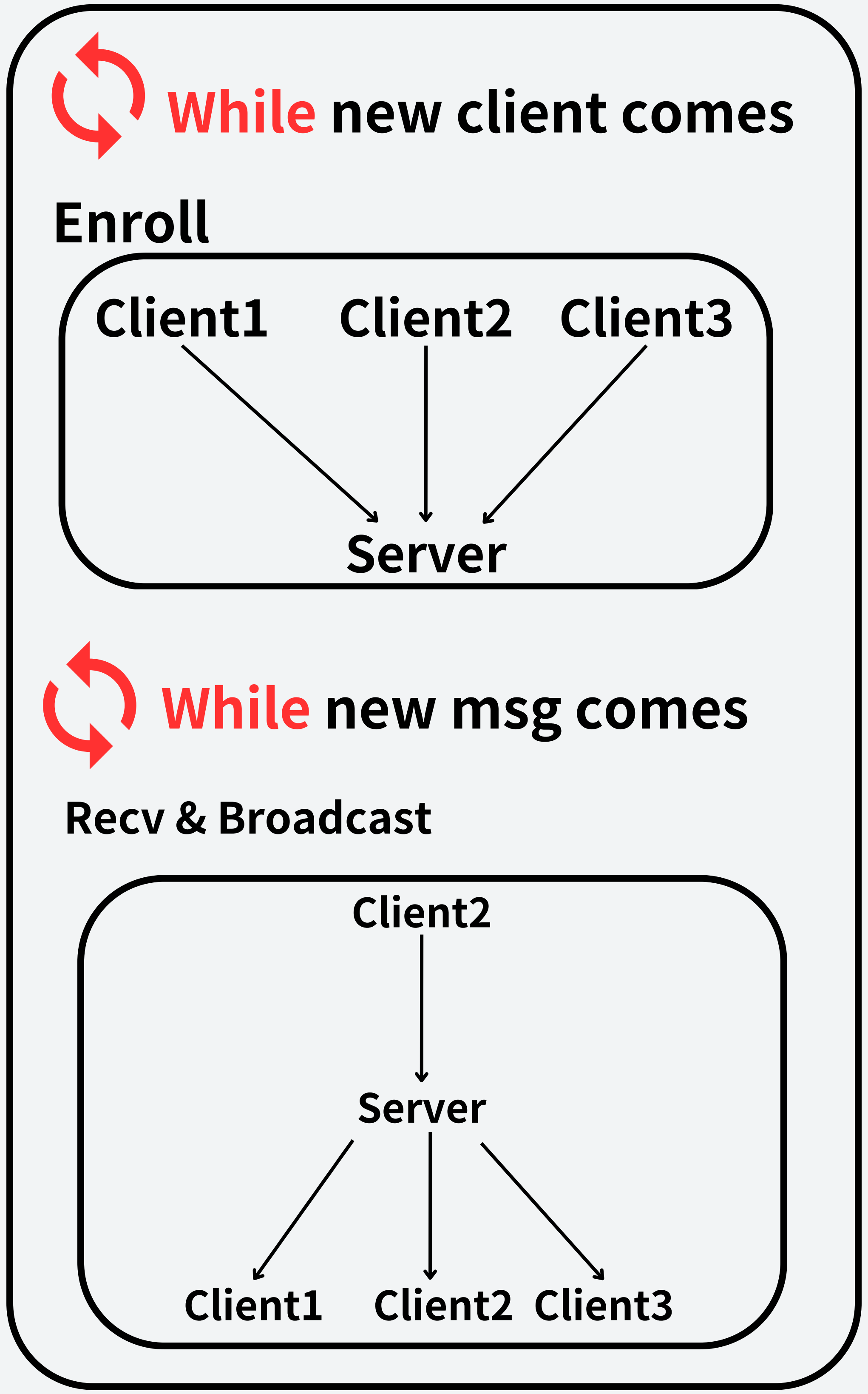
2.send: 先encode 後send
recv: 先recv 後decode



```
client_socket.send("Enter your name: ".encode("utf-8"))
name = client_socket.recv(1024).decode("utf-8")
```

3.Server運行至recv function時，會一直等待接收到訊息後才會繼續run下去
(阻塞 I/O)

Server



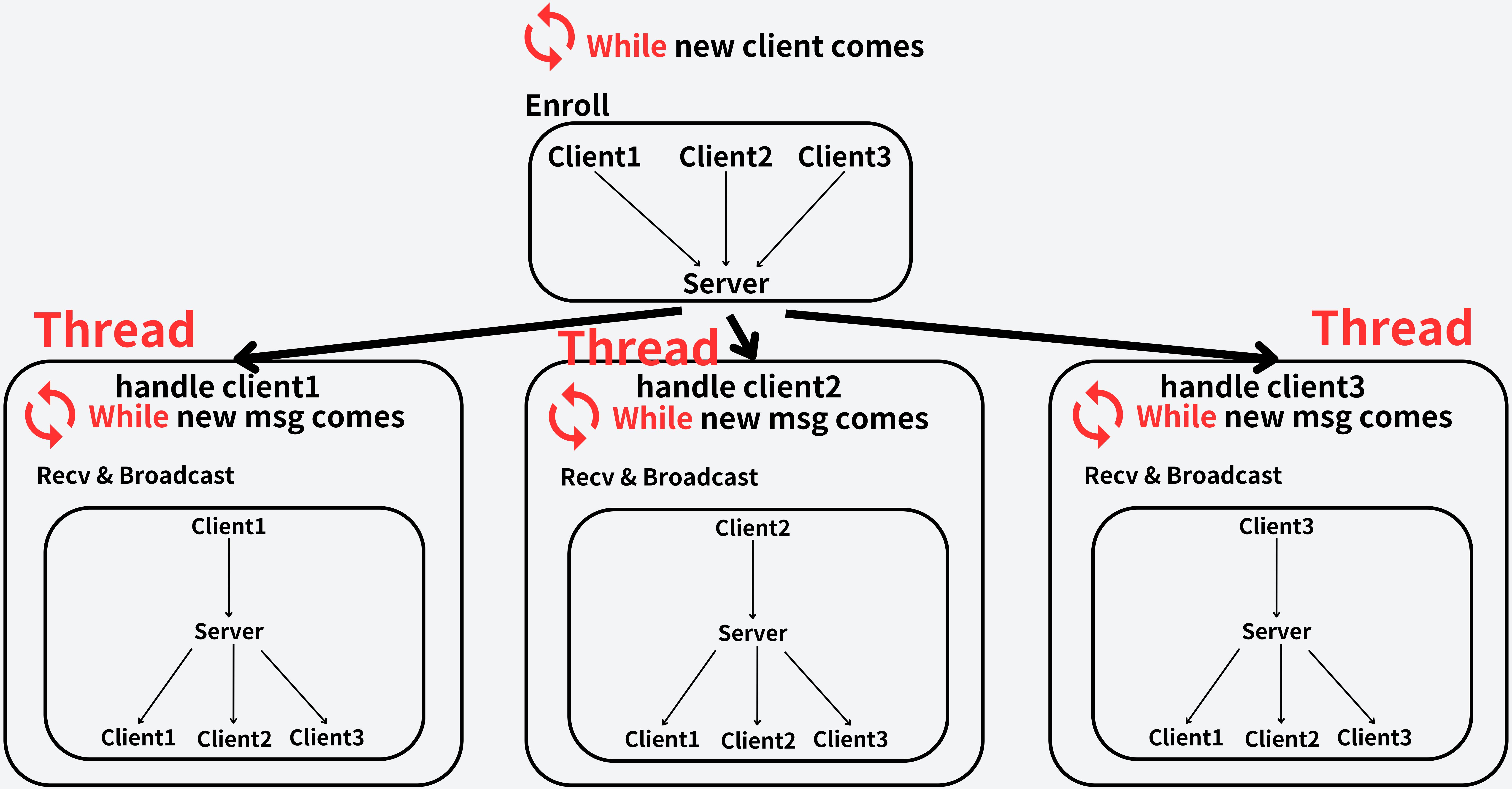
While new client comes

Enroll

While new msg comes

Recv & Broadcast

Server



While new client comes

Enroll

Client1 Client2 Client3

Server

Thread

handle client1

While new msg comes

Recv & Broadcast

Client1

Server

Client1 Client2 Client3

Thread

handle client2

While new msg comes

Recv & Broadcast

Client2

Server

Client1 Client2 Client3

Thread

handle client3

While new msg comes

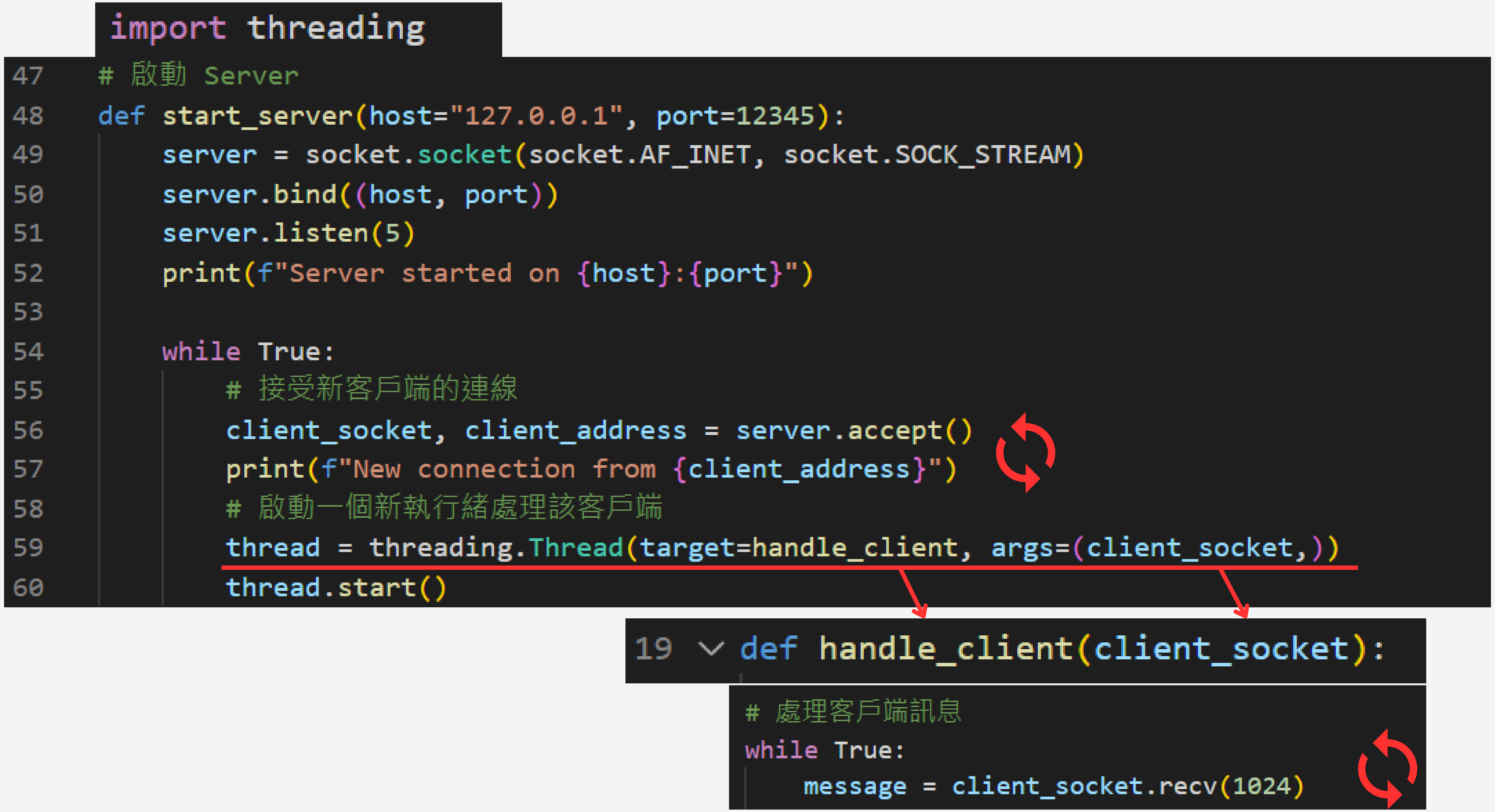
Recv & Broadcast

Client3

Server

Client1 Client2 Client3

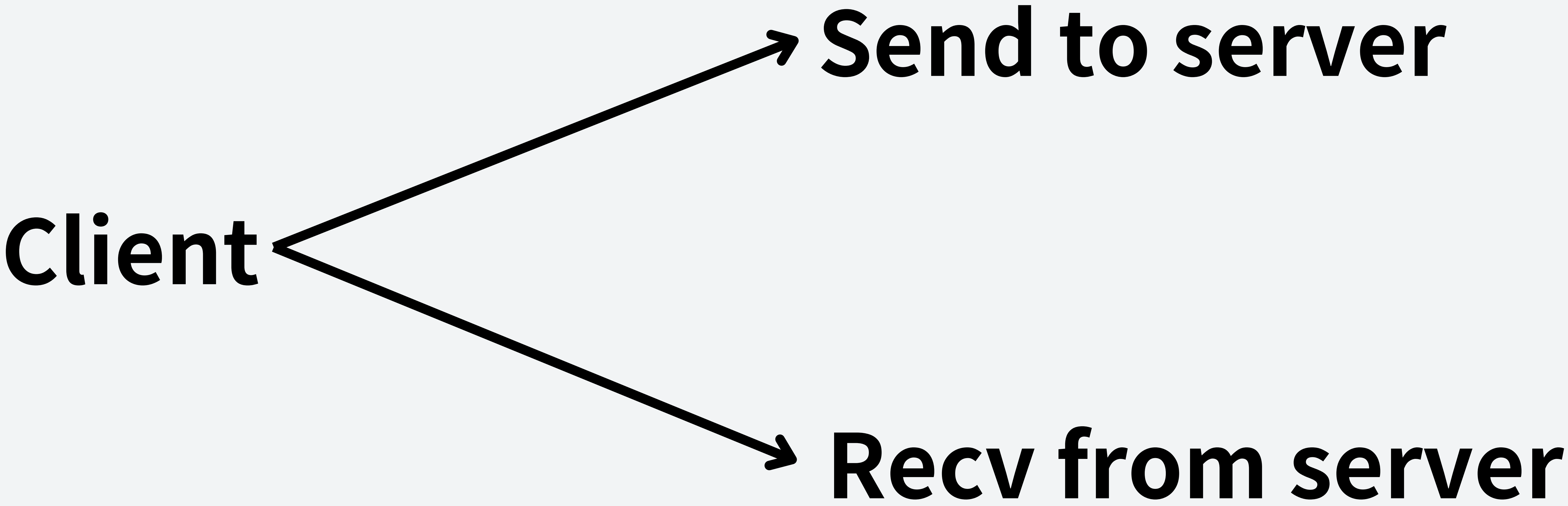
Thread



```python
import threading
```

```python
# 啟動 Server
def start_server(host="127.0.0.1", port=12345):
    server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    server.bind((host, port))
    server.listen(5)
    print(f"Server started on {host}:{port}")

    while True:
        # 接受新客戶端的連線
        client_socket, client_address = server.accept()
        print(f"New connection from {client_address}")
        # 啟動一個新執行緒處理該客戶端
        thread = threading.Thread(target=handle_client, args=(client_socket,))
        thread.start()
```

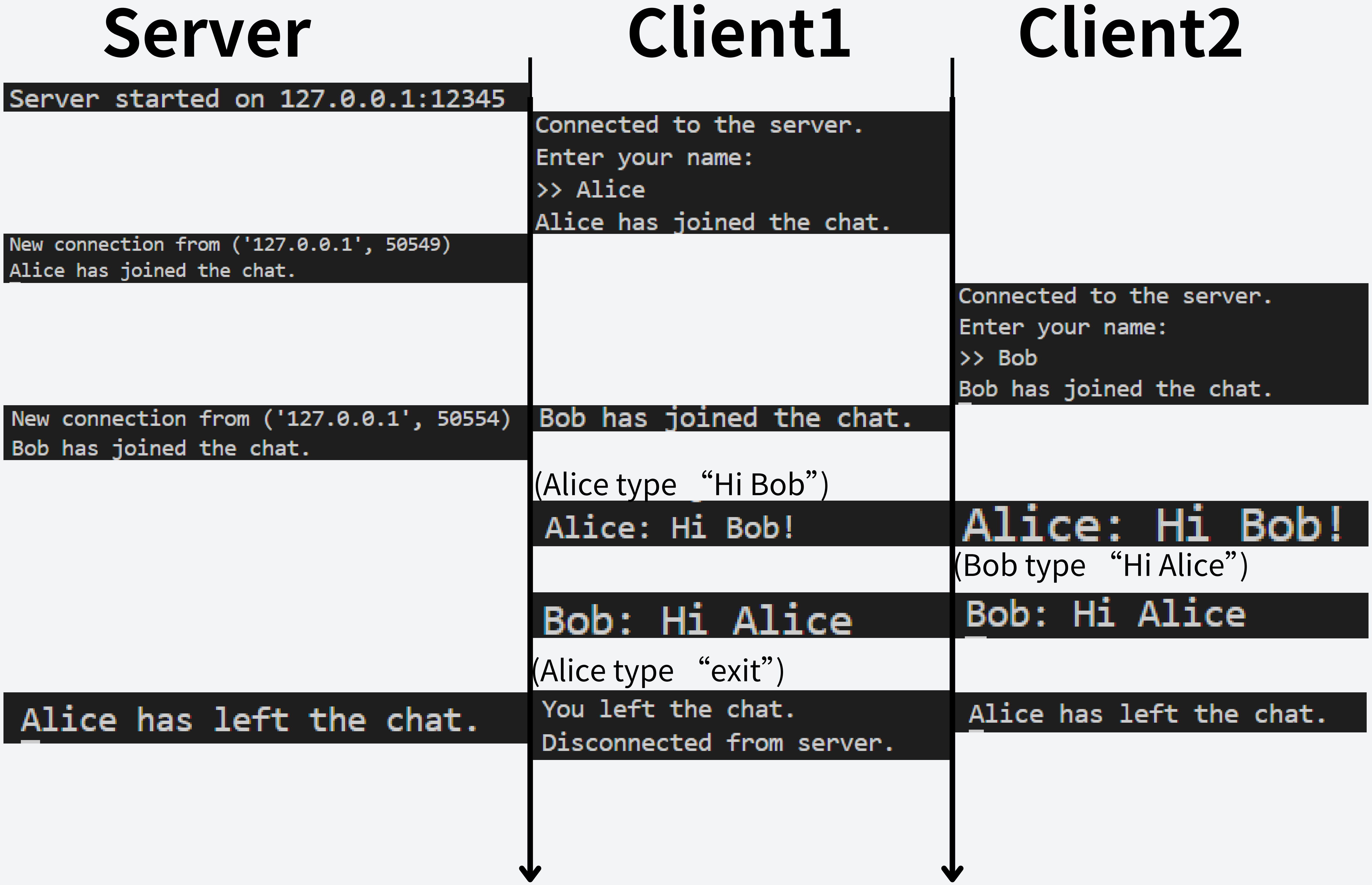
```python
def handle_client(client_socket):
    # 處理客戶端訊息
    while True:
        message = client_socket.recv(1024)
```



Client

Send to server

Recv from server



Server

Client1

Client2

```
Server started on 127.0.0.1:12345
```

```
Connected to the server.
Enter your name:
>> Alice
Alice has joined the chat.
```

```
New connection from ('127.0.0.1', 50549)
Alice has joined the chat.
```

```
Connected to the server.
Enter your name:
>> Bob
Bob has joined the chat.
```

```
New connection from ('127.0.0.1', 50554)
Bob has joined the chat.
```

```
Bob has joined the chat.
```

(Alice type "Hi Bob")

```
Alice: Hi Bob!
```

```
Alice: Hi Bob!
```

(Bob type "Hi Alice")

```
Bob: Hi Alice
```

```
Bob: Hi Alice
```

(Alice type "exit")

```
Alice has left the chat.
```

```
You left the chat.
Disconnected from server.
```

```
Alice has left the chat.
```