Blokkoló socket szerver

Nézzük meg hogyan implementálja a Python a socketeket.

Egyszerű blokkoló TCP szerver és a hozzá tartozó kliens létrehozásával szemléltetjük a blokkoló socket megközelítés különbségét Pythonban. Ez a szerver egyszerre egy kapcsolatot kezel blokkoló módon, ami azt jelenti, hogy vár (vagy blokkol) az I/O műveletekre, mint például új kapcsolatok elfogadása vagy adatok fogadása.

```
import socket

HOST = '127.0.0.1'  # Standard loopback interfész cím (localhost)
PORT = 65432        # Figyelt port (nem privilegizált portok > 1023)

# Socket létrehozása
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as server_socket:
    server_socket.bind((HOST, PORT))
    server_socket.listen()
    print(f"A szerver figyel a következőn: {HOST}:{PORT}")

    while True:
        # Új kapcsolat elfogadása
        conn, addr = server_socket.accept()
        with conn:
            print(f"Kapcsolódva {addr} címről")
            while True:
                data = conn.recv(1024)
                if not data:
                    break  # Nincs több adat a klienstől, kapcsolat bontása
                print(f"Fogadott adat: {data.decode()} innen: {addr}")
                response = "Válasz a szervertől.".encode()
                conn.sendall(response)
```

Blokkoló kliens

```
import socket

HOST = '127.0.0.1'  # A szerver hosztneve vagy IP címe
PORT = 65432        # A szerver által használt port

# Socket létrehozása
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
    s.connect((HOST, PORT))
    print("Kapcsolódva a szerverhez")

    # Adat küldése
```

```
    message = 'Helló, szerver'.encode()
    s.sendall(message)
    print("Üzenet elküldve a szervernek")

    # Várakozás válaszra
    data = s.recv(1024)
    print("Válasz fogadva a szervertől")

print(f"Fogadott adat: {data.decode()}")
```

Nem blokkoló szerver

Egy nem blokkoló TCP socket szerver létrehozása Pythonban magában foglalja egy olyan socket beállítását, amely kapcsolatokra figyel anélkül, hogy blokkolná a program fő végrehajtási szálát. Íme egy egyszerű példa egy nem blokkoló TCP szerverre, amely több klienskapcsolatot fogad és aszinkron módon kezeli őket. Ez a szerver a select metódust használja, amely lehetővé teszi több kapcsolat kezelését anélkül, hogy bármelyiken is blokkolódna.

```
import socket
import select

HOST = '127.0.0.1'  # Standard loopback interfész cím (localhost)
PORT = 65432        # Figyelt port (nem privilegizált portok > 1023)

# Socket létrehozása
server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_socket.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)

# A socket címhez és porthoz kötése
server_socket.bind((HOST, PORT))

# Kapcsolatokra való figyelés
server_socket.listen()

print(f"Figyelés itt: {HOST}:{PORT}")

# A szerver socket beállítása nem blokkoló módba
server_socket.setblocking(0)

# Bemeneti socketek nyomon követése
inputs = [server_socket]
outputs = []

while inputs:
    # Várakozás legalább egy socket készenlétére a feldolgozáshoz
```

```
    readable, writable, exceptional = select.select(inputs, outputs, inputs)

    for s in readable:
        if s is server_socket:
            # Új kapcsolat elfogadása
            connection, client_address = s.accept()
            print(f"Új kapcsolat innen: {client_address}")
            connection.setblocking(0)
            inputs.append(connection)
        else:
            data = s.recv(1024)
            if data:
                # Olvasható kliens socketnek van adata
                print(f"Fogadott adat: {data} innen: {s.getpeername()}")
                # Válaszcsatorna hozzáadása a válaszhoz
                if s not in outputs:
                    outputs.append(s)
            else:
                # Üres eredmény mint a kapcsolat bontása
                print(f"Kapcsolat bontása: {client_address}")
                if s in outputs:
                    outputs.remove(s)
                inputs.remove(s)
                s.close()

    for s in writable:
        response = b'Válasz a szervertől.'
        s.send(response)
        # Miután a választ elküldtük, már nem kell írnunk többet
        outputs.remove(s)

    for s in exceptional:
        print(f"Kivételes állapot kezelése itt: {s.getpeername()}")
        # Bemenet figyelésének leállítása a kapcsolaton
        inputs.remove(s)
        if s in outputs:
            outputs.remove(s)
        s.close()
```

Python nem blokkoló kliens

A nem blokkoló TCP szerver teszteléséhez létrehozhatunk egy egyszerű klienst, amely kapcsolódik a szerverhez, üzenetet küld, majd vár egy válaszra. Ez egy alap TCP kliens példa Pythonban, amely interakcióba lép a nem blokkoló szerverünkkel. Látunk különbséget a blokkoló klienssel összehasonlítva?

```
import socket
```

```
HOST = '127.0.0.1'  # A szerver hosztneve vagy IP címe
PORT = 65432        # A szerver által használt port

# Socket létrehozása
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
    # Kapcsolódás a szerverhez
    s.connect((HOST, PORT))
    print("Kapcsolódva a szerverhez")

    # Adat küldése
    message = 'Helló, szerver'.encode()
    s.sendall(message)
    print("Üzenet elküldve a szervernek")

    # Várakozás válaszra
    data = s.recv(1024)
    print("Válasz fogadva a szervertől")

print(f"Fogadott adat: {data.decode()}")
```

From:
https://edu.iit.uni-miskolc.hu/ - **Institute of Information Science - University of Miskolc**

Permanent link:
https://edu.iit.uni-miskolc.hu/tanszek:oktatas:informacios_rendszerek_integralasa:python_sockets

Last update: **2024/02/25 20:43**