看rustTcpServer/src/main.rs

//说明：rustTCPServer
//图0:rustTCPServer专案
//图1:在rustTCPServer下跑：cargo run
//图2:建立一个简单的rustTCPClient
//图3:在rustTCPClient下跑：cargo run后rustTCPServer terminal的情况，有印出"New connection:...“

//参考：
https://riptutorial.com/rust/example/4404/a-simple-tcp-client-and-server-application--echo

```
use std::thread;
use std::net::{TcpListener, TcpStream, Shutdown};
use std::io::{Read, Write};

//handle_client function，main()会呼叫
fn handle_client(mut stream: TcpStream) {
    let mut data = [0 as u8; 50]; // using 50 byte buffer
    //while loop，模式匹配
    while match stream.read(&mut data) {

        Ok(size) => {
            // echo everything!
            stream.write(&data[0..size]).unwrap();
            true
        },
        //有Err时print出error并shut down
        Err(_) => {
            println!("An error occurred, terminating connection with {}",
stream.peer_addr().unwrap());
            stream.shutdown(Shutdown::Both).unwrap();
            false
        }
    } {}
}

fn main() {
    //listener
    let listener = TcpListener::bind("0.0.0.0:3333").unwrap();
    // accept connections and process them, spawning a new thread for each one
    println!("Server listening on port 3333");
    //for loop，模式匹配
    for stream in listener.incoming() {
        match stream {
            //ok时print出"New connection:...“,并呼叫handle_client
            Ok(stream) => {
```

```
                println!("New connection: {}", stream.peer_addr().unwrap());
                thread::spawn(move|| {
                    // connection succeeded
                    handle_client(stream)
                });
            }
            //error时print出error
            Err(e) => {
                println!("Error: {}", e);
                /* connection failed */
            }
        }
    }
    // close the socket server
    drop(listener);
}
```