Ministerul Educaţiei, Culturii şi Cercetării Republicii Moldova

Universitatea Tehnică a Moldovei

Facultatea Calculatoare Informatică și Microelectronica

Departamentul Ingineria Software şi Automatică

**Referat**

**Disciplina: Programarea în rețea**

**Laboratorul 4**

Tema: Aplicație Client-Server TCP

A elaborat:

Pavlenco Ecaterina, st. gr. TI-171

A verificat:

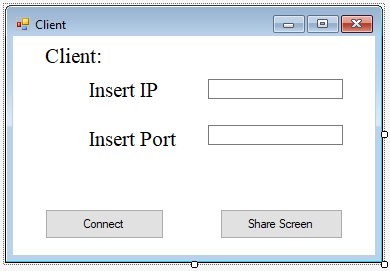
Buldumac Oleg

Chișinău 2020

Să se creeze o aplicație Client-Server TCP utilizând Sockets API

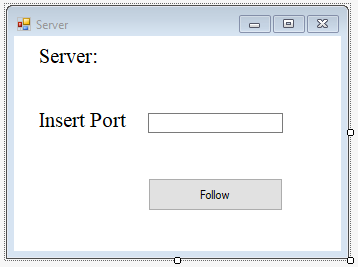
Laboratorul dat a fost efectuat prin crearea 2 aplicații , adică aplicația general condtă din 2 părți, partea Client și partea Server . Scopul acestea era comunicarea acestora prin protocolul TCP , și anume Serverul va lua pachetele de la Client și va face o partajere de ecran în formă de imagine .

Forms pentru părțile Client și Server :



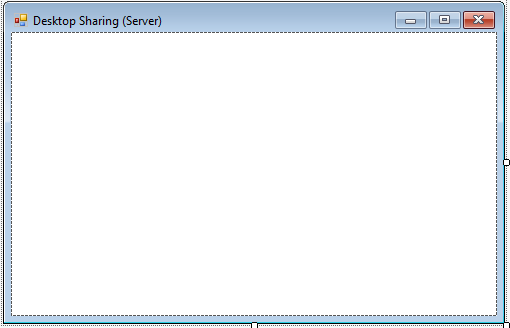
La Insert IP introducem IP local

Insert Port inserăm portul care va fi conectat cu Serverul și dăm la connect , în caz de eroare va fi afișată eroarea , dacă e correct va fi afișat boxul că e correct . Share Screen și Stop Screen deja pornește și oprește partajarea.



Creăm portul pentru a face conexiunea

Forms unde va fi partajat ecranul:



Pentru a primi imaginea screen Desktop noi utiluzăm metoda GrabDesktop() , care returnează imaginea . Această metodă este utilizată de clasa BinaryFormater care transmite imaginea la nivel de bytes.

Link:

https://github.com/katy198/PR-laboratorul4.git

Video:



Întrebări:

•Ce este un protocol orientat pe conexiune ?

Protocolul orientat pe conexiune este un protocol de rețea utilizat pentru a stabili o sesiune de comunicare între dispozitivele finale (client-server, client-client, server-server) în care aceste dispozitive utilizează protocoale preliminare de a stabili conexiunea ent-to-end și transmit secvențial datele.

• Ce tipuri de aplicații beneficiază în general de utilizarea protocolului TCP ?

Care au nevoie de mai mult timp pentru transmiterea datelor , aplicațiile pot fi Web Hostinguri, text chaturi, tranzacțiile bancare.

• Cum TCP garantează că datele vor fi transmise cu succes ?

Protocolul TCP numără secvență pentru a asigura că datele sunt furnizate în ordine și fără lipsuri.

• Diferența dintre blocking si non-blocking sockets

Socketurile non-blocking au efect ca la accept(), cînd apelăm accept și nu există deja client care să conecteze acesta returneaza Operation Would Block.Pe când la blocking este valabil precum ar fi la write() și/sau connect(), când le executăm conexiunea se blochează până când operațiunea este completă.

• Diferența dintre blocking multithreaded și non-blocking single thread socket

Într-un program cu un singur fir de execuție un server care dorește să prelucreze cereule concomitente de la mai mulți clienți, setează toate socketurile în modul de neblocare. În multithread sockets este altă abordare. În loc să setăm socketurile neblocate, serverul poate crea thread separat pentru a gestiona fiecare solicitare a clientului.

• Cum are loc procesul TCP Three Way Handshake ?

*Acesta reprezintă crearea conexiunii între 2 puncte și are loc din 3 stări.*

*Starea 1 Transmiterea SYN- clientul dorește să stabilească o conexiune cu serverul*

*Starea 2 SYN+ACK - Serverul răspunde la solicitarea clientului cu biți de semnal SYN-ACK setați*

*Starea 3 ACK - În partea finală, clientul recunoaște răspunsul serverului și amândoi stabilesc o conexiune fiabilă cu care va începe transferul efectiv de date*

• Numiti cele 4 apeluri de sistem necesare pentru a crea un server TCP

*Socket – crearea socketului*

*Bind – asignarea adresei*

*Listen – ascultă adresa*

*Accept – acceptă conexiunea*

• Care este rolul metodei bind() ?

Are rolul de a asocia socketul la adresa locala, adică adresa IP, portul, și familia de adrese.

• Care este rolul metodei accept() ?

Accept() are funcția de accepta conexiunea a unui socket, o conexiune de intrare este recunoscută și asociată cu un socket imediat.

• Ce se întîmplă cînd apelați mai întîi connect() apoi bind() ?

Se va afișa eroare deoarece connect face conectarea clientului să facă listen la server pe când bind alocă port de către sistem pentru un socket , pentru server . Deci conexiunea nu va fi posibilă.

• Ați avea vreodată nevoie să implementați un timeout într-un client sau server care utilizează TCP?

În cazul în care pentru fiecare conexiune, „timeout” al utilizatorului, care specifică cantitatea maximă de timp în care datele transmise pot rămâne necunoscute înainte ca TCP să închidă cu forță conexiunea corespunzătoare.

• Într-o conexiune TCP, clientul sau serverul trimite mai întâi datele ?

Secvența de transmisii depinde complet de programarea „clientului” și a „serverului”.De fapt, cele două părți ar putea fi „colegi” cu fiecare trimitere de date către cealaltă simultan.

Conexiunea TCP prevede două conducte unidirecționale (A -> B & B -> A) care pot fi utilizate oricât de dorit și cu orice coordonare între conducte este dorită.

• Care este adresa de loopback IPv4 și care este rolul ei ?

Loopback este un canal de comunicare cu un singur sfarsit. Retelele TCP/IP prezintă un loopback care permite software-ului client sa comunice cu software-ul server pe același computer. Intervalul de adrese pentru funcționalitatea loopback este intervalul de la 127.0.0.0 pana la 127.255.255.255. Similar cu ping, loopback permite utilizatorului sa-si testeze propria rețea pentru a se asigura ca stack-ul IP funcționează corect.

• De unde știe un sistem de operare ce aplicație este responsabilă pentru un pachet primit din rețea ?

• Datele primite prin recv() au întotdeauna aceeași dimensiune cu datele trimise cu send() ?

De obicei da, dar recv () nu primește întotdeauna datele din bucățile pe care le-au fost trimise de send (), de fapt - se întâmplă foarte rar - din cauza tamponării (de exemplu, trimiteți 256 de octeți primiți două tampoane de 128 de octeți fiecare)

• Este acceptabil să închei executia programului dacă este detectată o eroare de rețea ?

Fiecare eroare de rețea are o fază, care descrie în ce fază a apărut eroarea fiecare cu soluții aparte de soluționare a erorii apărute :

dns - eroarea a apărut în timpul rezoluției DNS

connection - eroarea s-a produs în timpul stabilirii conexiunii sigure

application - eroarea a apărut în timpul transmiterii cererii și răspunsului

• Puteți îmbunătăți performanța aplicației prin dezactivarea algoritmului Nagle ?

Algoritmul Nagle ar trebui de dezactivat în cazul când algoritmul duce la deprecierea preformanțelor ca mesaje scurte, segvențe escape ș.a. pentru ca programul să fie mai rapid , în caz contrar algoritmul propus din contra încearcă să îmbunătățească aplicația.

• Ce instrumente listează socket-urile TCP deschise în sistemele de operare Windows și Linux ?

• Tehnici de sincronizare a firelor de execuții

 Fiecare fir de executie are o viata proprie si nu este interesat de ceea ce fac celelalte fire de executie . Sincronizarea se bazeaza pe conceptul de **monitor.** Daca un fir ocupa un monitor, un alt fir care incearca ocuparea aceluiasi monitor, este blocat si asteapta pana la eliberarea monitorului. Pentru a realiza aceasta comunicare, clasa Thread pune la dispozitie metodele wait, notify, notifyAll.