Cursus Chatprogramma:

Inhoud

[1 TCP & UDP 3](#_Toc24116500)

[1.2 TCP 3](#_Toc24116501)

[1.3 UDP 3](#_Toc24116502)

[1.4 Waarom gebruiken we TCP 3](#_Toc24116503)

[1.5 Voorbeeld TCP 3](#_Toc24116504)

[2 Poorten 4](#_Toc24116505)

[2.2 Wat zijn poorten 4](#_Toc24116506)

[2.3 Welke poorten gaan we gebruiken 4](#_Toc24116507)

[2.4 Voorbeeld firewall error 4](#_Toc24116508)

[3 Mutlithreading 5](#_Toc24116509)

[3.2 Wat is multithreading 5](#_Toc24116510)

[3.3 Wat zijn threads 5](#_Toc24116511)

[3.4 Waarvoor gebruiken we threads 5](#_Toc24116512)

[3.5 Voorbeeld multithreading 5](#_Toc24116513)

[4 Sockets 6](#_Toc24116514)

[4.2 Wat zijn sockets 6](#_Toc24116515)

[Een *socket* is een communicatie-eindpunt in een netwerk. Deze worden geïdentificeerd door de combinatie van het IP-adres en de poortnummer gescheiden door een deelteken bv: 127.0.0.1:80 6](#_Toc24116516)

[4.3 Waarom gebruiken we sockets 6](#_Toc24116517)

[We gebruiken in de server de stream socket dit is een type socket waarmee een betrouwbare bytestroom tussen *hosts* kan worden opgezet. Dit is standaard bij TCP. 6](#_Toc24116518)

[5 Oefening Chatprogram 7](#_Toc24116519)

[5.1 Client: solution 7](#_Toc24116520)

[6 Bronnen: 8](#_Toc24116521)

INLEIDING

# TCP & UDP

## TCP

TCP of transmission control protocol is een connectie-georiënteerd protocol dat veel gebruikt wordt voor het maken van netwerkverbindingen.

TCP maakt ook gebruik van *Quality of service (QoS)* dit betekend dat pakketten voorrang op andere kunnen krijgen.

TCP maakt ook gebruik van de *Three-Way Handshake,* dit is voor de connectie te controleren of de data die je verstuurd zeker aankomt.

## UDP

UDP of *User Datagram Protocol* is sneller dan TCP maar gaat de connectie tussen *sender* en *reciever* niet controleren waardoor er data verloren kan gaan. Dit word gebruikt bij Livestreans, fps

## Waarom gebruiken we TCP

We gebruiken TCP omdat we zeker willen zijn dat er connectie is van de client tot de server. Zo kunnen we zeker zijn dat de verstuurde berichten aankomen.

## Voorbeeld TCP

HOE

# Poorten

# Wat zijn poorten

Een poort of een gate zien we vooral als een toegang tot iets. Er zijn 2 soorten poorten een *hardwarepoort* en een *netwerkpoort*.

*Hardwarepoorten* worden gebruikt voor het aansluiten van randapparatuur.

Grootste voorbeelden zijn: de parallelle poort, de seriële poort en de USB

Netwerkpoort is een poort waaraan een nummer word gegeven volgens een bepaald TCP/IP-protocol.

## Welke poorten gaan we gebruiken

We gaan netwerkpoorten gebruiken voor de verbinding tussen de server en de client. We gaan poort 443 gebruiken voor de connectie naar het internet.

## Voorbeeld firewall error

# Mutlithreading

SCREENSHOT

## Wat is multithreading

Bij multithreaden word de data verdeelt in kleinere delen ook wel threads genoemd. Deze worden dan allemaal verstuurd en op het einde terug in hun originele vorm geplaatst

Het zit Een virtuele adresruimte dat het procesbeeld bevat of in heeft een beveiligde toegang tot processoren, andere processen (voor interproces communicatie), bestanden en I/O bronnen (apparaten en kanalen).

## Wat zijn threads

Een *thread* is de kleinste opeenvolging van geprogrammeerde instructies.

## Waarvoor gebruiken we threads

We gebruiken threads omdat het programma terwijl andere clients verbonden zijn het ook nog kan kijken of er nieuwe clients willen verbinden.

## Voorbeeld multithreading

HOE

# Sockets

## Wat zijn sockets

## Een *socket* is een communicatie-eindpunt in een netwerk. Deze worden geïdentificeerd door de combinatie van het IP-adres en de poortnummer gescheiden door een deelteken bv: 127.0.0.1:80

## Waarom gebruiken we sockets

## We gebruiken in de server de stream socket dit is een type socket waarmee een betrouwbare bytestroom tussen *hosts* kan worden opgezet. Dit is standaard bij TCP.

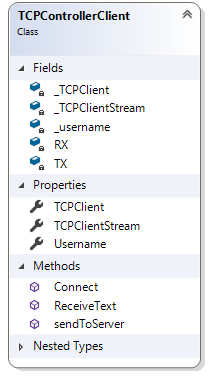
ALLEEN CLIENT LATEN MAKEN (server moet niet besproken worden)

# Oefening Chatprogram

Met deze oefening gaan we proberen een chatprogramma te maken dat kan gebruikt worden bij bijvoorbeeld een klein spel zodat de spelers met elkaar kunnen communiceren. Wat we voor deze oefening nodig hebben zijn 2 verschillende projects: Client en Server. Jullie gaan van deze 2 de client maken, de server hebben wij al voor jullie gemaakt en kan je vinden op onze github namelijk(moet nog gemaakt worden).

## 5.1 Client: solution

Maak een *Windows Forms Application* aan met de naam ***ChatClient.***

**Client: Domeinlaag**

* Maak 3 eigenschappen aan waarin de TCPClient, TCPCLientStream en Username kunnen bewaard worden.

## Bronnen:

<https://nl.wikipedia.org/wiki/TCP-_en_UDP-poorten>

<http://howtostartprogramming.com/vb-net/vb-net-tutorial-53-multithreading/>

<http://www.handel.broeders.be/wiki/index.php?title=Transport_protocol#Transmission_Control_Protocol>

<https://www.geeksforgeeks.org/tcp-and-udp-in-transport-layer/>