**TCP  NEDİR?**

TCP, Transmission Control Protocol (İletim Denetimi Protokolü) sözcüklerinin kısaltmasıdır**.**Uygulama programlarının veri alışverişinde bulunabileceği bir ağ iletişiminin nasıl kurulacağını ve sürdürüleceğini tanımlayan bir standarttır.TCP, bilgisayarların birbirlerine nasıl veri paketleri göndereceğini tanımlayan İnternet Protokolü (IP) ile çalışır. TCP ve IP birlikte İnterneti tanımlayan temel kurallardır.

**TCP NASIL ÇALIŞIR?**

TCP, bağlantı odaklı bir protokoldür, yani her iki uçtaki uygulama programları mesaj alışverişini bitirene kadar bağlantı kurulur ve sürdürülür. Uygulama verilerinin ağların teslim edebileceği paketlere nasıl bölüneceğini belirler, paketleri ağ katmanına gönderir ve bu katmanlardan gelen paketleri kabul eder, akış kontrolünü yönetir ve hatasız veri iletimi sağlaması amaçlandığı için bırakılan veya bozuk olanların yeniden iletimini yönetir paketler ve gelen tüm paketleri onaylar. Açık Sistemler Ara Bağlantısı (OSI) iletişim modelinde, TCP, Katman 4'ün bölümlerini, taşıma katmanını ve oturum katmanı olan Katman 5'in bölümlerini kapsar.

Önek verirsek, bir web sunucusu bir istemciye bir HTML dosyası gönderdiğinde, bunu yapmak için köprü metni aktarım protokolünü (HTTP) kullanır. HTTP program katmanı, TCP katmanından bağlantıyı kurmasını ve dosyayı göndermesini ister. TCP yığını, dosyayı veri paketlerine böler, numaralandırır ve ardından dağıtım için ayrı ayrı IP katmanına iletir. İletimdeki her paket aynı kaynak ve hedef IP adresine sahip olsa da, paketler birden çok yol üzerinden gönderilebilir. İstemci bilgisayardaki TCP program katmanı, tüm paketler gelene kadar bekler, daha sonra aldıklarını onaylar ve eksik paket numaralarına bağlı olarak almayanların yeniden iletilmesini ister. TCP katmanı daha sonra paketleri bir dosyada birleştirir ve dosyayı alıcı uygulamaya teslim eder.

**UDP NEDİR**

UDP (Kullanıcı Datagram Protokolü), öncelikle internetteki uygulamalar arasında düşük gecikmeli ve kayba toleranslı bağlantılar kurmak için kullanılan bir iletişim protokolüdür. Alıcı taraf tarafından bir anlaşma yapılmadan önce veri aktarımını etkinleştirerek aktarımları hızlandırır. Sonuç olarak, UDP, Internet Protokolü üzerinden ses (VoIP), alan adı sistemi (DNS) araması ve video veya ses çalma gibi zamana duyarlı iletişimlerde faydalıdır. UDP, İletim Kontrol Protokolü’ne (TCP) bir alternatiftir.

**UDP NASIL ÇALIŞIR?**

UDP protokolü, TCP'ye benzer şekilde çalışır, ancak tüm hata denetimi öğelerini atar. Tüm ileri geri iletişim, işleri yavaşlatan gecikmeye neden olur.

Bir uygulama UDP kullandığında, paketler yalnızca alıcıya gönderilir. Gönderen, alıcının paketi aldığından emin olmak için beklemez, yalnızca sonraki paketleri göndermeye devam eder. Alıcı, arada sırada birkaç UDP paketini kaçırırsa, kaybolur - gönderen bunları yeniden göndermez. Tüm bu ek yükü kaybetmek, cihazların daha hızlı iletişim kurabileceği anlamına gelmektedir.

UDP, hız istendiğinde ve hata düzeltme gerekmediğinde kullanılır. Örneğin, UDP genellikle canlı yayınlar ve çevrimiçi oyunlar için kullanılır.

Örneğin, genellikle TCP yerine UDP kullanılarak yayınlanan bir canlı video akışını izlediğinizi varsayalım. Sunucu sadece izleyen bilgisayarlara sabit bir UDP paketleri akışı gönderir. Bağlantınızı birkaç saniye için kaybederseniz, video donabilir veya bir an için titreyebilir ve ardından yayının o anki bitine atlayabilir. Küçük bir paket kaybı yaşarsanız, video eksik veriler olmadan oynatılmaya devam ettiği için video veya ses bir an için bozulabilmektedir..

Bu, çevrimiçi oyunlarda benzer şekilde çalışır. Bazı UDP paketlerini kaçırırsanız, siz yeni UDP paketlerini alırken oyuncu karakterleri harita boyunca ışınlanıyormuş gibi görünebilir. Oyun siz olmadan devam ettiği için, eski paketleri kaçırdıysanız istemenin bir anlamı yok. Önemli olan tek şey şu anda oyun sunucusunda olup bitenlerdir, birkaç saniye önce olanlar değil. TCP’nin hata düzeltmesini ortadan kaldırmak, oyun bağlantısını hızlandırmaya ve gecikmeyi azaltmaya yardımcı olur.

|  |  |
| --- | --- |
| **TCP** | **UDP** |
| Baglantı tabanlı protokoldür | Baglantı tabanlı protokol degildir. |
| TCP yavaştır. | UDP hızlıdır. |
| Başlık boyutu 20bytedir. | Başlık boyutu 8 bytedır. |
| TCP, verileri bayt akışları olarak okur ve mesaj, segment sınırlarına iletilir. | UDP mesajları, tek tek gönderilen paketleri içerir. Ayrıca varış saatinde bütünlüğü kontrol eder. |
| TCP mesajları internette bir bilgisayardan diğerine geçer. | Bağlantı tabanlı değildir, bu nedenle bir program diğerine çok sayıda paket gönderebilir. |
| TCP, veri paketlerini belirli bir sırayla yeniden düzenler. | Tüm paketler birbirinden bağımsız olduğu için UDP protokolünün sabit bir sırası yoktur. |
| TCP ağırdır. Herhangi bir kullanıcı verisi gönderilmeden önce bir soket bağlantısı kurmak için TCP’nin üç pakete ihtiyacı vardır. | UDP hafiftir. İzleme bağlantısı, mesaj sırası vb. yoktur. |
| Onay segmentleri vardır. | Onay segmentleri yoktur. |
| El sıkışma protokolü kullanır. SYN, SYN-ACK, ACK | El sıkışma olmaz. (Bu yüzden bağlantı tabanlı değildir) |
| TCP, hedef yönlendiriciye veri iletimini garanti ettiği için güvenilirdir. | Verinin hedefe ulaştırılması UDP’de garanti edilemez. |
| TCP, akış denetimi ve verilerin onaylanmasını sağladığı için kapsamlı hata kontrol mekanizmaları sunar | UDP’nin checksum için kullanılan tek bir hata kontrol mekanizması vardır. |

**PROGRAMIN İÇERİĞİ**

Chat programı server ve client ten oluşur. Programda tcp ve udp seçeneği vardır. Tcp de server kullanıcının girdiği bir portu dinlemeye alır. Client programı o port a bağlantı yaptığında chat başlar.

Bağlantı sağlamadan program bir alt satırdaki kod parçasına geçmez

clientSocket = serverSocket.accept();

Client'a veri gönderimi için kullandığımız PrintWriter nesnesi oluşturulur

PrintWriter out = new PrintWriter(clientSocket.getOutputStream(), true);

out.println(“”);

Client'dan gelen verileri tutan BufferedReader nesnesi oluşturulur

BufferedReader in = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));

clientGelen = in.readLine()

udp seçildiğinde server ile client arasında bağlantı sağlanmaz. Verinin doğru bir şekilde iletilmesi kontrol edilmez. Ilk olarak client server in portuna bir datagram paket yollar. Server bu paketten source port çıkarır ve oraya veri gönderebilir.

Server e veri alınırken

byte[] inBuf = new byte[256];

DatagramPacket inPacket = new DatagramPacket(inBuf, inBuf.length);

datagramSocket.receive(inPacket);

String data = new String(inPacket.getData(), 0, inPacket.getLength());

Bu gelen veri den source port ve address bulunur.

int source\_port = inPacket.getPort();

InetAddress source\_address = inPacket.getAddress();

Server cliente veri gönderirken bu source port ve source address i kullanarak cliente veri gönderir.

byte[] outBuf = msg.getBytes();

DatagramPacket outPacket = new DatagramPacket(outBuf, 0, outBuf.length, source\_address, source\_port);

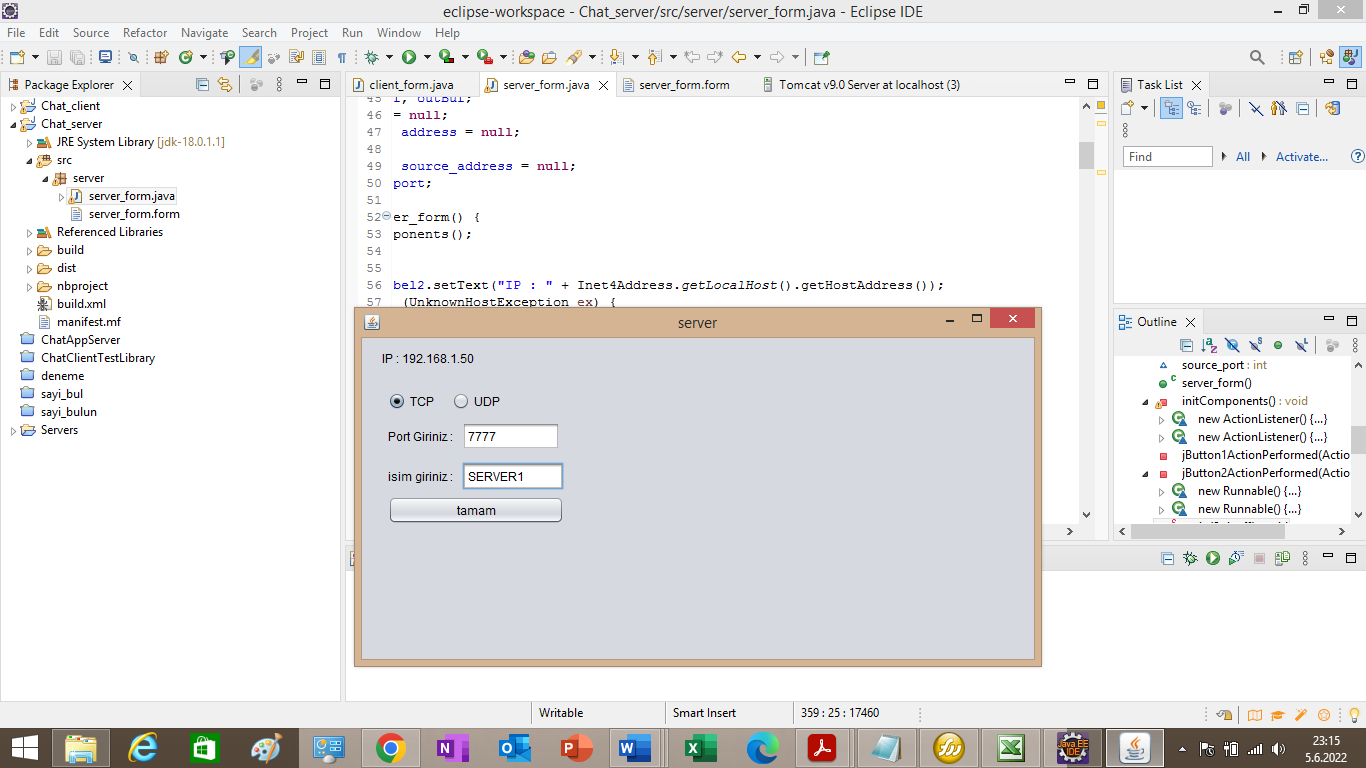
datagramSocket.send(outPacket);

**JAVA CHAT AKTİF EDİLMESİ**

Server\_form.java çalıştırılır. Aşağıdaki resimde olduğu gibi server’ın kurulacağı IP otomatik ekrana gelir.

Server1’in aktif kurulacağı sanal port adı girilir. Örneğin 7777 gibi.

Server’in adı yazılır.



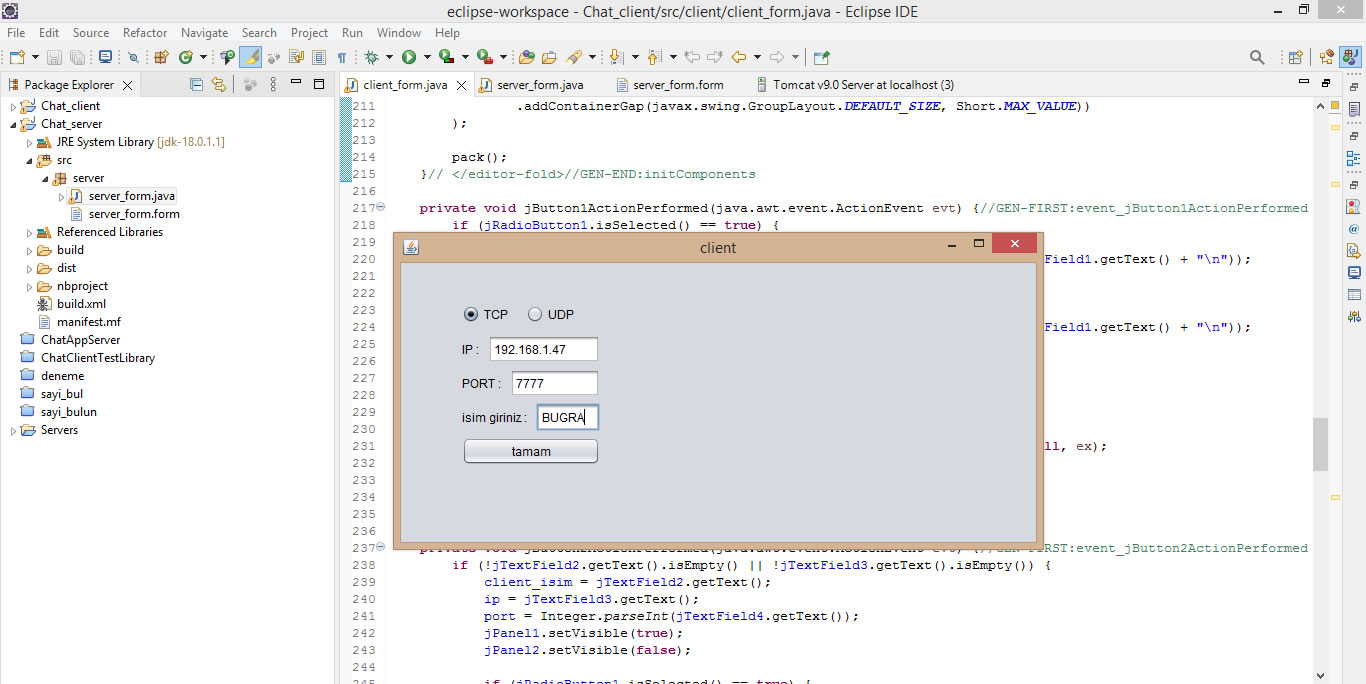
SONRA

Client\_form.java çalıştırılır.

Protokol seçilir. Karşı bağlanacak kişinin IP si belirlenir.

Sonra port girilir. Örneğin 7777 gibi.

Sonra chat yapacak kişinin adı girilir.

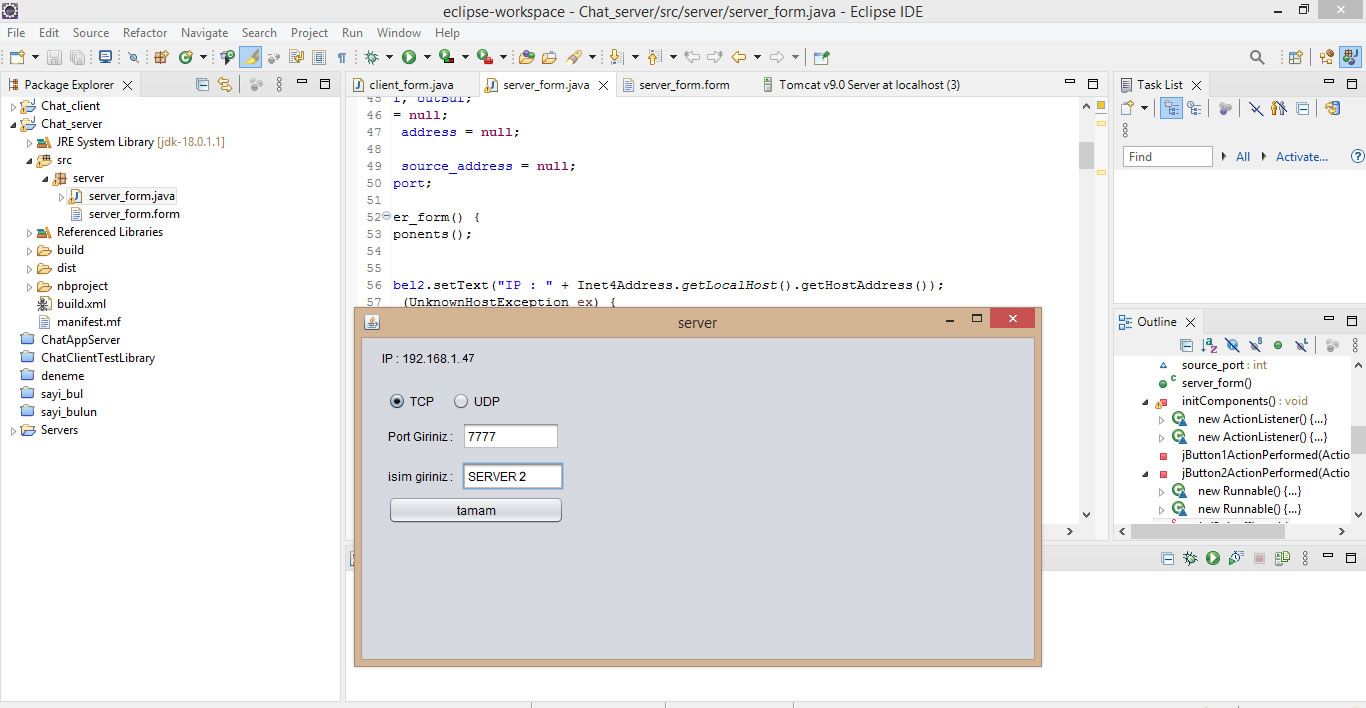


**DİĞER BİLGİSAYARDA CHAT KURULMASI**

Server\_form.java çalıştırılır. Aşağıdaki resimde olduğu gibi server2’nın kurulacağı IP otomatik ekrana gelir.

Server2’in aktif kurulacağı sanal port adı girilir. Örneğin 7777 gibi.

Server’in adı yazılır.



SONRA

Client\_form.java çalıştırılır.

Protokol seçilir. Karşı bağlanacak kişinin IP si belirlenir.

Sonra port girilir. Örneğin 7777 gibi.

Sonra chat yapacak kişinin adı girilir.

