Ağ Programlama
Soket Programlama

Konular

- ✓ Stream
- ✓ TCP Soketi
- ✓ UDP Soketi

STREAM

- ✓ Bilgi (objects, characters, images, or sounds) diskteki dosyada, ağ üzerinde bir kaynakta, bellekte ya da bir başka programda bulunabilir.
- ✓ Bilginin ardışık olarak taşınması için stream açılır.
- ✓ Java platformunda TCP istemci/sunucu soketleri veri gönderme/alma işlemleri için Stream kullanırlar.

java.io

✓ Byte Streams

Some important Byte stream classes.

Stream class	Description
BufferedInputStream	Used for Buffered Input Stream.
BufferedOutputStream	Used for Buffered Output Stream.
DataInputStream	Contains method for reading java standard datatype
DataOutputStream	An output stream that contain method for writing java standard data type
FileInputStream	Input stream that reads from a file
FileOutputStream	Output stream that write to a file.
InputStream	Abstract class that describe stream input.
OutputStream	Abstract class that describe stream output.
PrintStream	Output Stream that contain `print()` and `println()` method

These classes define several key methods. Two most important are

1. `read()` : reads byte of data.
2. `write()` : Writes byte of data.

java.io

✓ Character Streams (unicode character)

Some important Charcter stream classes.

Stream class	Description
BufferedReader	Handles buffered input stream.
BufferedWriter	Handles buffered output stream.
FileReader	Input stream that reads from file.
FileWriter	Output stream that writes to file.
InputStreamReader	Input stream that translate byte to character
OutputStreamReader	Output stream that translate character to byte.
PrintWriter	Output Stream that contain `print()` and `println()` method.
Reader	Abstract class that define character stream input
Writer	Abstract class that define character stream output

Java ile Socket (java.net) Uygulaması

Sunucu

```
serverSocket = new ServerSocket(port);

System.out.println("Sunucu baslatildi. Baglanti bekleniyor...");


clientSocket = serverSocket.accept(); // Bağlantı kabul ediliyor

// Giriş Çıkış Stream leri tanımlanıyor
out = new PrintWriter(clientSocket.getOutputStream(), true);
in = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));
```

İstemci

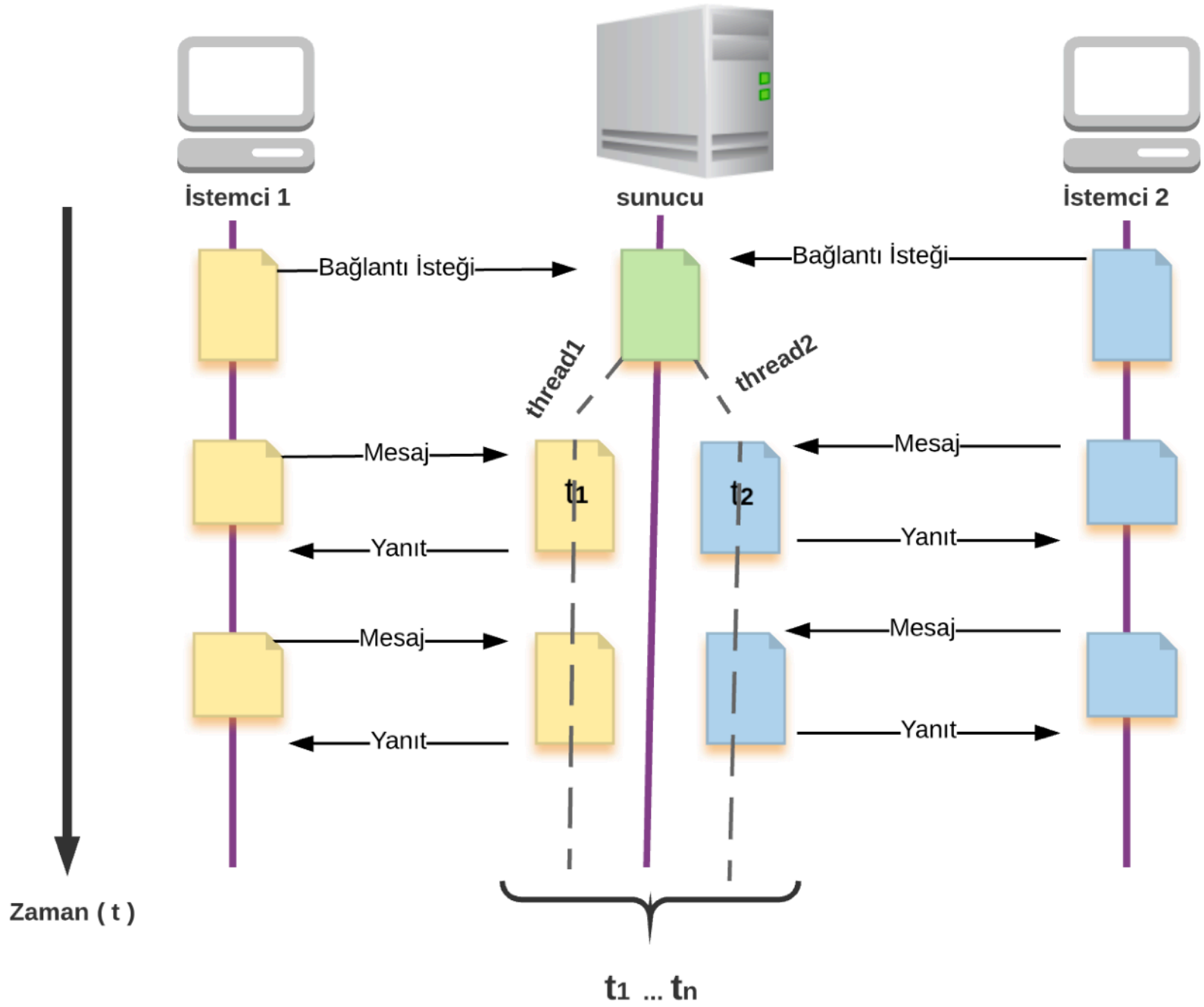
```
socket = new Socket(sERVER, pORT); // istemci soketi oluşturuluyor

// output stream ve input stream olusuyor
out = new PrintWriter(socket.getOutputStream(), false);
//Creates a new PrintWriter, without (if true then with) automatic line
//flushing, from an existing OutputStream.
in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
```

İş Parçacığı - Thread

http://docs.oracle.com/javase/7/docs/api/java/lang/Thread.html
http://www.tutorialspoint.com/java/java_multithreading.htm

Çok İş parçacıklı Soket Sunucu



Çok İşparçacıklı Soket Sunucu

Her istek için bir iş parçacığı oluşturuluyor

```
ServerSocket serverSocket = new ServerSocket(PORT);
System.out.println("Listening");
try {
    while (true)
    {
        Socket clientSocket = serverSocket.accept();
        System.out.println(clientSocket.getRemoteSocketAddress() + " baglandi.");

        // Her bağlantıı için yeni bir iş parçacığı (thread) oluşturuluyor...
        Thread task = new Handler(clientSocket);
        task.start();
    }
```

DoS- Hizmet engelleme saldırılarına karşı, oluşturulacak iş parçacığı sayısı sınırlanıyor

```
// DOS saldırılarına karşı, oluşturulacak thread miktarını sınırlamak için ExecutorService kullanılır
ExecutorService executor = Executors.newFixedThreadPool(2);
ServerSocket serverSocket = new ServerSocket(PORT);
System.out.println("Listening");
try {
    while (true)
    {
        Socket clientSocket = serverSocket.accept();
        System.out.println(clientSocket.getRemoteSocketAddress() + " baglandi.");

        // Her bağlantıı için yeni bir thread oluşturuluyor...
        executor.execute(new Handler(clientSocket));
        //Thread task = new Handler(clientSocket);
        //task.start();
    }
```

UDP Soket İstemcisi

```
//Log dosyası belirleniyor...

FileHandler handler = new FileHandler("Logs/client.log", 100000, 10000);
Logger.getLogger("gunluk").addHandler(handler);

// Soket oluşturuluyor
socketClient = new DatagramSocket();

// veri gönderildikten sonra yanıtın gelmesini bekleme süresi ayarlanıyor. UDP, TCP gibi
//bağlantı yönelimli olmadığı için bu kadar süre sonra yanıt gelmez ise verinin gitmediği düşünülebilir...

socketClient.setSoTimeout(tIMEOUT);

sunucuIPAdresi = InetAddress.getByName(sUNUCU);

// klavyeden girdi: stdIn
BufferedReader stdIn = new BufferedReader(new InputStreamReader(System.in));
String userInput=null;

// Giriş ve çıkışlar için oluşturulacak DatagramPacket içerisinde kullanılmak üzere tampon bellek oluşturuluyor.
// TCP deki stream yerine DatagramPacket kullanılıyor.
byte[] in = new byte[1024];
byte[] out  = new byte[1024];

userInput = new String(stdIn.readLine());
out=userInput.getBytes(); // kullanıcının girdiği string byte dizisine dönüştürülüyor.

// Sunucuya veri göndermek üzere DatagramPacket oluşturuluyor, içerisine kullanıcının klavyeden girdiği
// mesaj yazılıyor.
DatagramPacket gonderilecekPaket = new DatagramPacket(out, out.length, sunucuIPAdresi, pORT);
socketClient.send(gonderilecekPaket);
// Bilgi mesajı olarak günlüğe ekleniyor
audit.info("Paket gönderildi. Hedef soket adresi:"+sunucuIPAdresi+" : "+pORT);


// Sunucudan gelen veriyi almak üzere DatagramPacket oluşturuluyor
DatagramPacket gelenPaket = new DatagramPacket(in, in.length);

socketClient.receive(gelenPaket);
```

UDP Soket Sunucusu

```
socketServer = new DatagramSocket(pORT);

// Giriş ve çıkışlar için oluşturulacak DatagramPacket içerisinde kullanılmak üzere tampon bellek oluşturuluyor.
// TCP deki stream yerine DatagramPacket kullanılıyor.

byte[] in = new byte[1024];
byte[] out  = new byte[1024];

DatagramPacket gelenPaket = new DatagramPacket(in, in.length);

socketServer.receive(gelenPaket);

String inputLine = new String(gelenPaket.getData());
InetAddress IPAddress = gelenPaket.getAddress();

outputLine = inputLine.toUpperCase();
out= outputLine.getBytes();

DatagramPacket gonderilecekPaket = new DatagramPacket(out, out.length, gelenPaket.getAddress(), gelenPaket.getPort());
socketServer.send(gonderilecekPaket);
audit.info("Adres:"+IPAddress);
```

Kaynaklar

- http://www.studytonight.com/java/java-io-stream.php
- http://www.tutorialspoint.com/java/java_multithreading.htm