Görsel Programlama

DERS 12

Java Ağ İşlemleri (Java Networking)

Birbirleri ile ağ araçları ve kabloları ile bağlantılı bilgisayarlar bir ağ sistemi oluştururlar. İnternet, şirketlerin yerel bilgisayar sistemleri bu şekildedir. Ağ üzerinde bulunan bilgisayarlar birbirleri ile iletişim kurabilirler. Java programları bu iletişim ile ilgili tüm sınıfları "java.net" paketinde sağlamaktadır.

Java ağ üzerindeki işlemleri dosyalar gibi Stream tabanlı sınıflar ile gerçekleştirir. Bununla birlikte paket temelli çalışan sınıflarda bu pakette bulunmaktadır.

Java Ağ İşlemleri (Java Networking)

Ağ üzerindeki bir iletişimden bahsettiğimiz zaman istemci-sunucu (client-server) kavramı ortaya çıkmaktadır.
İstemci ağdaki sunucudan bazı bilgiler ister ve sunucuda bu bilgileri istemciye gönderir. Örneğin; www sunucuları ile tarayıcılar arasındaki iletişim bu şekilde gerçekleşmektedir. İstemci sunucuya görmek istediği sayfayı söyler, sunucuda sayfa var ise istemciye bu sayfayı gönderir.
İnternet ya da ağ üzerinde birbirleriyle haberleşen tüm uygulamalar *TCP(Transmission Control Protocol)*ve ya *UDP(User Datagram Protocol)* iletişim protokolünü kullanarak birbirleri ile haberleşirler.

Java Ağ İşlemleri (Java Networking)

TCP protokolü, en fazla kullanılan ve güvenli bir iletişim protokolüdür. Gönderilen veriler paket adı verilen küçük parçacıklar halinde karşı bilgisayar gönderilir ve ulaşıp ulaşmadığı ya da arızalı olup olmadığı denetlenir. Herhangi bir olumsuzlukta tekrar gönderilir.

UDP protokolünde veri sadece gönderilir. Paketlerin hangi sırada ulaştığı ya da ulaşıp ulaşmadığı garanti edilmez.

PORT Nedir?

Bir bilgisayarın genellikle ağ ile bir fiziksel bağlantı noktası vardır; ethernet kartı. Buna karşın bilgisayarımızda birden fazla program aynı anda ağa, internete erişmek ve veri alışverişi nde bulunmak isteyebilir. Bu işlemi tek bir çıkış ile sağlayabilmeleri için Port(Kapı) lar kullanılır.

Ağ üzerinde ve intenette taşınacak olan veri paketlerinde hangi adrese gideceğini belirten (IP-Adres) bilgileri bulunmaktadır. Bu adrese ek olarak mevcut adreste hangi kapıyı iletişim için kullanağı port numarası kullanılarak gösterilir. (IP+PORT) bilgisi veri paketinin ulaşması istenilen adres bilgisi olmuş olur.

PORT Nedir?

IP Adresi 32 bit veri ile,
PORT ise 16 bit veri ile gösterilir.

Port numaraları 0-65535 arasında değer alabilirler.

0-1024 arasındaki port numaraları belirli programlar ve iletişim protokolleri için ayrılmıştır. Örneğin; http (80), FTP (21) .

Sunucu Oluşturma

1) Öncelikle bir *ServerSocket* nesnesi oluşturulur.

ServerSocket sunucu = new ServerSocket(port,kuyrukBoyutu)

port=sunucunun dinleme yapacağı kapı numarası, kuyrukBoyutu = sunucuya maksimum bağlanabilecek istemci sayısıdır.

2) Sunucu tüm istemcileri *Socket* nesnesi ile yönetir. Sunucu ikinci adımda dinleme işlemi yapar.

Socket baglanti = sunucu.accept();

Bu dinleme işleminde bir istemci bağlanırsa geriye bağlantı nesnesi döndürülür. Bu nesne sunucunun istemci ile iletişim kurmasını sağlar. Bağlanan her istemci için ayrı nesne oluşturulur.

Sunucu Oluşturma

3) Bu adımda istemciden veri alıp-göndermek için nesneler oluşturulur, Stream nesneleri oluşturulur. Sunucu veri göndermek için *"OutputStream"*, veri almak için *"InputStream"* nesnelerini kullanır. Sunucu bu nesneleri elde etmek için istemciye ait olan *bağlantı (Socket)* nesnesinin metotlarını kullanır.

baglanti.getOutputStream()
baglanti.getInputStream()

OutputStream nesnesinin *write()* metodu bilgi göndermek için InputStream nesnesinin *read()* metodu bilgi okumak için kullanılır.
4) Bağlantı bittiğinde *Socket.close()* metodu ile sonlandırılır. Socket nesnesi sayesinde ağ iletişimi I/O gibi yapılmaktadır.

İstemci(Client) Oluşturma

1) Sunucuya(Server) bağlanmak için Socket nesnesi oluşturulur.

Socket baglanti = new Socket(sunucuAdresi,port);

Eğer bağlantı başarılı olur ise Socket geri döndürülür. Bağlantı başarısız olursa *"IOException"* oluşturulur. Sunucu adresi bulunamıyorsa *"UnknownHostException"*, bağlanmada hata oluşursa *"ConnectException"* oluşturulur.

2) Sunucuda olduğu gibi Socket nesnesinin *getInputStream()* vet *getOutputStream()* metotları kullanılarak veri yazmak için ve okumak için kullandığımız nesneleri alırız.

3) *Socket.close()* ile bağlantıyı sonlandırırız.

SUNUCU

Sunucu.java

```
package ce.comu.edu.ders12;

import java.awt.BorderLayout;

public class Sunucu extends JFrame {
    private JTextField girdiAlani;
    private JTextArea ciktiAlani;
    private ObjectOutputStream cikti;
    private ObjectInputStream girdi;
    private ServerSocket sunucu;
    private Socket baglanti;
    private int sayac=1;
    public Sunucu() {
        super("Sunucu");
        createGUI();
    }
    private void createGUI() {
        Container con = this.getContentPane();
        girdiAlani = new JTextField();
        girdiAlani.setEditable(true);
        girdiAlani.addActionListener(new ActionListener(){

            public void actionPerformed(ActionEvent e) {
                String veri = e.getActionCommand();
                veriGonder(veri);
                girdiAlani.setText("");
            }
        });
        con.add(girdiAlani,BorderLayout.PAGE_START);
        ciktiAlani = new JTextArea();
        con.add(new JScrollPane(ciktiAlani),BorderLayout.CENTER);
        this.setSize(300,150);
    }
```

Sunucu.java

```
public void sunucuCalistir(){
    try {
        sunucu= new ServerSocket(12345,100);
        while (true) {
            try {
                baglantiBekle();
                streamleriAl();
                baglantiyiIsle();
            } catch (EOFException e) {
                System.err.println("Sunucu baglantiyi durdurdu");
            } finally{
                baglantiyiKapat();
                ++sayac;
            }
        }
    } catch (IOException e) {
        e.printStackTrace();
    }
}
private void baglantiBekle() throws IOException {
    mesajGoster("Baglanti Bekleniyor\n");
    baglanti=sunucu.accept();
    mesajGoster("Baglanti"+sayac+" kuruldu, Adres="+baglanti.getInetAddress().getHostName());
}
private void streamleriAl() throws IOException {
    cikti = new ObjectOutputStream(baglanti.getOutputStream());
    cikti.flush();
    girdi = new ObjectInputStream(baglanti.getInputStream());
    mesajGoster("\n IO Streamler olusturuldu \n");
}
private void baglantiyiIsle() throws IOException {...}
private void baglantiyiKapat() {...}
private void mesajGoster(String mesaj) {...}
protected void veriGonder(String veri) {...}
```

Sunucu.java

```
    private void baglantiBekle() throws IOException {
        mesajGoster("Baglanti Bekleniyor\n");
        baglanti=sunucu.accept();
        mesajGoster("Baglanti"+sayac+" kuruldu, Adres="+baglanti.getInetAddress().getHostName());
    }
    private void streamleriAl() throws IOException {
        cikti = new ObjectOutputStream(baglanti.getOutputStream());
        cikti.flush();
        girdi = new ObjectInputStream(baglanti.getInputStream());
        mesajGoster("\n IO Streamler olusturuldu \n");
    }
    private void baglantiyiIsle() throws IOException {
        String mesaj = "Baglanti kuruldu";
        veriGonder(mesaj);
        girdiAlani.setEditable(true);
        do{
            try {
                mesaj = (String) girdi.readObject();
                mesajGoster("\n"+mesaj);
            } catch (ClassNotFoundException e) {
                mesajGoster("\nBilinmeyen Nesne\n");
            }
        }while(!mesaj.equals("IS>>>SON"));
    }
    private void baglantiyiKapat() {..
    private void mesajGoster(String mesaj) {..
    protected void veriGonder(String veri) {..

    public static void main(String[] args) {
        Sunucu sunucu = new Sunucu();
        sunucu.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        sunucu.setVisible(true);
        sunucu.sunucuCalistir();
    }
```

Sunucu.java

```
    private void baglantiyiIsle() throws IOException {
    private void baglantiyiKapat() {
        mesajGoster("\n Baglanti Sonlandiriliyor\n");
        girdiAlani.setEditable(false);
        try {
            cikti.close();
            girdi.close();
            baglanti.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
    private void mesajGoster(String mesaj) {
        ciktiAlani.append(mesaj);
        ciktiAlani.setCaretPosition(ciktiAlani.getText().length());
    }
    protected void veriGonder(String veri) {
        try {
            cikti.writeObject("SU>>>"+veri);
            cikti.flush();
            mesajGoster("\nSU>>>"+veri);
        } catch (IOException e) {
            ciktiAlani.append("\n Nesne yazmada hata");
        }
    }

    public static void main(String[] args) {
        Sunucu sunucu = new Sunucu();
        sunucu.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        sunucu.setVisible(true);
        sunucu.sunucuCalistir();
    }
}
```

İSTEMCİ

Istemci.java

```
import java.awt.BorderLayout;

public class Istemci extends JFrame {
    private JTextField girdiAlani;
    private JTextArea ciktiAlani;
    private ObjectOutputStream cikti;
    private ObjectInputStream girdi;
    private String mesaj="";
    private String sunucu;
    private Socket istemci;

    public Istemci(String host) {
        super("Istemci");
        sunucu=host;
        createGUI();
    }

    private void createGUI() {
        Container con = this.getContentPane();
        girdiAlani = new JTextField();
        girdiAlani.setEditable(false);
        girdiAlani.addActionListener(new ActionListener(){

            public void actionPerformed(ActionEvent e) {
                veriGonder(e.getActionCommand());
                girdiAlani.setText("");
            }
        });
        con.add(girdiAlani,BorderLayout.PAGE_START);
        ciktiAlani = new JTextArea();
        con.add(new JScrollPane(ciktiAlani),BorderLayout.CENTER);
        this.setSize(300,150);
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
```

Istemci.java

```
    private Socket istemci;
    public Istemci(String host) {..
    private void createGUI() {..
    public void istemciCalistir() {
        try {
            sunucuyaBaglan();
            streamleriAl();
            baglantiyiIsle();
        } catch (EOFException e) {
            System.err.println("istemci baglantiyi sonlandirdi");
        }catch (IOException e) {
            e.printStackTrace();
        }finally{
            baglantiyiKapat();
        }
    }
    private void sunucuyaBaglan() throws IOException, UnknownHostException {
        mesajGoster("Baglanti Kurulmaya Calisiliyor\n");
        istemci = new Socket(InetAddress.getByName(sunucu),12345);
        mesajGoster("Baglanildi:"+istemci.getInetAddress().getHostName());
    }
    private void streamleriAl() throws IOException {
        cikti = new ObjectOutputStream(istemci.getOutputStream());
        cikti.flush();
        girdi = new ObjectInputStream(istemci.getInputStream());
        mesajGoster("\n IO Streamler olusturuldu \n");
    }
    private void baglantiyiIsle() throws IOException {..
    private void baglantiyiKapat() {..
    private void mesajGoster(String mesaj) {..
    protected void veriGonder(String veri) {..
    public static void main(String[] args) {..
}
```

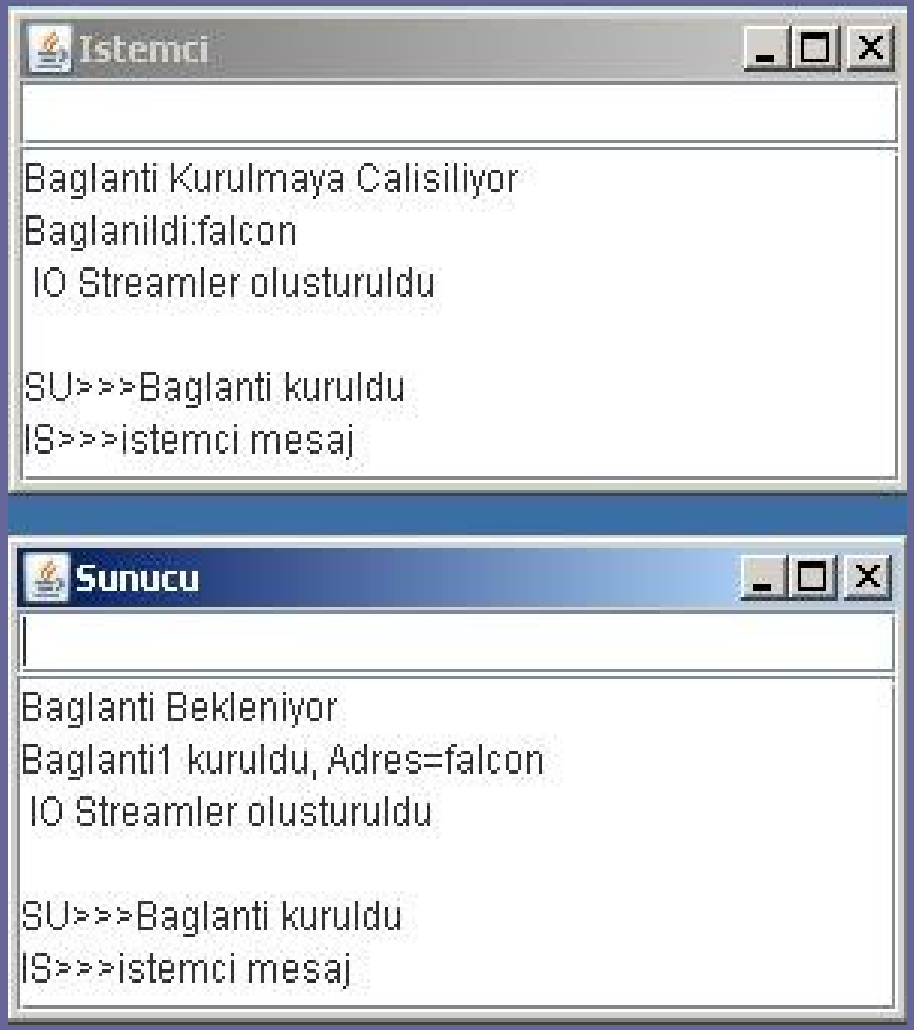
Istemci.java

```
    private void baglantiyiIsle() throws IOException {
        girdiAlani.setEditable(true);
        do {
            try {
                mesaj = (String)girdi.readObject();
                mesajGoster("\n"+mesaj);
            } catch (ClassNotFoundException e) {
                mesajGoster("\nBilinmeyen Nesne Tipi Alındı\n");
            }
        } while (!mesaj.equals("SU>>>SON"));
    }
    private void baglantiyiKapat() {
        mesajGoster("\nBaglanti Kapaniyor");
        girdiAlani.setEditable(false);
        try {
            cikti.close();
            girdi.close();
            istemci.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
    private void mesajGoster(String mesaj) {
        ciktiAlani.append(mesaj);
        ciktiAlani.setCaretPosition(ciktiAlani.getText().length());
    }
    protected void veriGonder(String veri) {
        try {
            cikti.writeObject("IS>>>"+veri);
            cikti.flush();
            mesajGoster("\nIS>>>"+veri);
        } catch (IOException e) {
            ciktiAlani.append("\n Mesaj Göndermede Hata");
        }
```

```
public class Istemci extends JFrame {
    private JTextField girdiAlani;
    private JTextArea ciktiAlani;
    private ObjectOutputStream cikti;
    private ObjectInputStream girdi;
    private String mesaj="";
    private String sunucu;
    private Socket istemci;
    public Istemci(String host) {
    private void createGUI() {
    public void istemciCalistir() {
    private void sunucuyaBaglan() throws IOException,
    private void streamleriAl() throws IOException {
    private void baglantiyiIsle() throws IOException {
    private void baglantiyiKapat() {
    private void mesajGoster(String mesaj) {
    protected void veriGonder(String veri) {
    public static void main(String[] args) {
        String sunucuIP="127.0.0.1";

        Istemci istemci = new Istemci(sunucuIP);
        istemci.setVisible(true);
        istemci.istemciCalistir();

    }
}
```


Istemci
Baglanti Kurulmaya Calisiliyor
Baglanildi:falcon
IO Streamler olusturuldu
SU>>>Baglanti kuruldu
IS>>>istemci mesaj
Sunucu
Baglanti Bekleniyor
Baglanti1 kuruldu, Adres=falcon
IO Streamler olusturuldu
SU>>>Baglanti kuruldu
IS>>>istemci mesaj

ÖRNEK -2

SayiBulOyunu.java

```
public class SayiBulOyunu {
    private int sayi;
    private int durum = 0;
    public SayiBulOyunu() {
        //0-100 arası bir sayi
        sayi= (int) (Math.random()*101);
        durum=0;
    }
    public String girdiyiIsle(String girdi){
        String cikti = null;
        int tahmin =0;
        try {
            tahmin = Integer.valueOf(girdi);
        } catch (Exception e) {
            e.printStackTrace();
        }
        if (durum==0){ cikti="1-100 arasi bir sayi dusundum " +
            "tahmin etmeyi dene CIKIŞ=0";
            durum=1;
        }else{
            if (tahmin==0){
                cikti="CIK";
            }else if (tahmin<sayi){
                cikti="daha büyük";
            }else if(tahmin>sayi){
                cikti="daha küçük";
            }else{
                cikti="Tebrikler";
            }
        }
        return cikti;
    }
}
```

SAYI BUL SUNUCU

SayiBulSunucu.java

```
public class SayiBulSunucu {
    public static void main(String[] args) {
        ServerSocket sunucu = null;
        int port=4444;
        try {
            sunucu=new ServerSocket(port);
        } catch (IOException e) {
            System.err.println("4444 portu dinlenemiyor\n");
            System.exit(1);
        }
        Socket istemci=null;
        try {
            istemci = sunucu.accept();
        } catch (IOException e) {
            System.err.println("Sunucu istemci kabul hatasi\n");
            System.exit(1);
        }
        PrintWriter cikti = null;
        BufferedReader girdi =null;
        try {
            cikti = new PrintWriter(istemci.getOutputStream(),true);
            girdi = new BufferedReader(
                    new InputStreamReader(istemci.getInputStream()));
            String girdiSatiri,ciktiSatiri;
            SayiBulOyunu oyun = new SayiBulOyunu();
            ciktiSatiri = oyun.girdiyiIsle(null);
            cikti.println(ciktiSatiri);
            while((girdiSatiri=girdi.readLine())!=null){
```

SAYI BUL SUNUCU

```
            while((girdiSatiri=girdi.readLine())!=null){
                ciktiSatiri = oyun.girdiyiIsle(girdiSatiri);
                cikti.println(ciktiSatiri);
                if (ciktiSatiri.equals("CIK")) break;
            }
        } catch (IOException e) {
            e.printStackTrace();
        }finally{
            try {
                cikti.close();
                girdi.close();
                istemci.close();
                sunucu.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }

    }
}
```

SAYI BUL İSTEMCİ

SayiBulIstemci.java

```
public class SayiBulIstemci {
    public static void main(String[] args) {
        Socket sunucu = null;
        String sunucuAdres="127.0.0.1";
        BufferedReader girdi = null;
        PrintWriter cikti = null;
        try {
            sunucu = new Socket(sunucuAdres,4444);
            cikti = new PrintWriter(sunucu.getOutputStream(),true);
            girdi = new BufferedReader(
                    new InputStreamReader(sunucu.getInputStream()));
        } catch (UnknownHostException e) {
            System.err.println("Bilinmeyen Sunucu adı");
            System.exit(1);
        } catch (IOException e) {
            System.err.println("IO baglantisi sunucu ile kurulamadi");
            System.exit(1);
        }
        BufferedReader konsoleGirdi = new BufferedReader(
                new InputStreamReader(System.in));
        String sunucudan;
        String kullanicidan;
        try {
            while((sunucudan=girdi.readLine())!=null){
```

SAYI BUL İSTEMCİ

```
            while((sunucudan=girdi.readLine())!=null){
                System.out.println("Sunucudan:"+sunucudan);
                if (sunucudan.equals("CIK")) break;
                System.out.println("sayi:");
                kullanicidan = konsoleGirdi.readLine();
                if (kullanicidan!=null){
                    System.out.println("İstemci:"+kullanicidan);
                    cikti.println(kullanicidan);
                }
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
        try {
            cikti.close();
            girdi.close();
            konsoleGirdi.close();
            sunucu.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

ÖRNEK-3 ÇOKLU İSTEMCİ DESTEĞİ

CokluSayiBulSunucu.java

```
package ce.comu.edu.ders12;
import java.io.IOException;
public class CokluSayiBulSunucu {
    private boolean dinleme = true;
    public void sunucuBasla(){
        int port = 4444;
        ServerSocket sunucu = null;

        try {
            sunucu = new ServerSocket(port,10);
        } catch (IOException e) {
            System.err.println(port+" dinlenemiyor");
            System.exit(-1);
        }
        try {
            while (dinleme) {
                Socket istemci = sunucu.accept();
                new SunucuIsParcasi(istemci).start();
            }
            sunucu.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
    public static void main(String[] args) {
        CokluSayiBulSunucu sunucu = new CokluSayiBulSunucu();
        sunucu.sunucuBasla();
    }
}
```

ÖRNEK-3 ÇOKLU İSTEMCİ DESTEĞİ

SunucuIsParcasi.java

```
import java.io.BufferedReader;
public class SunucuIsParcasi extends Thread {
    private Socket istemci = null;
    private static int sayac =0;
    public SunucuIsParcasi(Socket istemci) {
        super("Sunucu İs Parcacigi:"+(++sayac));
        this.istemci= istemci;
    }
    @Override
    public void run() {
        PrintWriter cikti = null;
        BufferedReader girdi = null;
        try {
            cikti = new PrintWriter(istemci.getOutputStream(),true);
            girdi = new BufferedReader(
                    new InputStreamReader(istemci.getInputStream()));
            String girdiSatiri,ciktiSatiri;
            SayiBulOyunu oyun = new SayiBulOyunu();
            ciktiSatiri = oyun.girdiyiIsle(null);
            cikti.println(ciktiSatiri);
            while((girdiSatiri=girdi.readLine())!=null){
                ciktiSatiri = oyun.girdiyiIsle(girdiSatiri);
                cikti.println(ciktiSatiri);
                if (ciktiSatiri.equals("CIK")){ --sayac;break;}
            }
            cikti.close();
            girdi.close();
            istemci.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

UDP ile İletişim

Bu iletişim protokolü ile gönderilen paketlere “Datagram” adı verilir. Bu paketin ulaşması, ulaşma zamanı, içeriği garanti edilmez.
Kullanımı:

1) *DatagramSocket socket = new DatagramSocket(port);*
//istemci ve sunucu bu socket i iletişim için kullanır.

2) Paket okuma

```
try{
   byte[] veri = new byte[100];
   DatagramPacket paket = new DatagramPacket(veri,veri.length);
   socket.receive(paket);
}catch(IOException e){
    e.printStackTrace();
}
```

UDP ile İletişim

3) Paket gönderme

```
String mesaj=”örnek mesaj”;
byte[] veri = mesaj.getBytes();
DatagramPacket gonderilenPaket = new
DatagramPacket(veri,veri.length,”gideceği adres”,”port”);

socket.send(gonderilenPaket);
```

Görsel Programlama

DERS 12