

AĞ PROGRAMLAMA

DERS 4: Java ile Ağ
Programlamaya Giriş

Giriş

- **java.net package** kullanılır.
- **Stream-based** veya **packed-based** haberleşme yapmaya izin verir.
- Kod örnekleri için: <http://carus.aces.shu.ac.uk/staff/cmsjg3/javanet.html>
- Bu derste:
 - IP adres bilgisi okuma
 - TCP socket açma
 - UDP socket açma
 - GUI sahibi ağ programları oluşturma
 - Bir makinadaki çalışan portların tespiti

Class: InetAddress

- Host adları ve IP adresleri ile ilgili işlemleri yapmamızı sağlayan sınıfır.
- Static methodu getByName DNS(Domain Name Server) üzerinden ilgili makine bilgisini ve IP adresini InetAddress objesi olarak bize verir.
- UnknownHostException: Bilinmeyen bir DNS kaydı istenirse dönecek hata mesajı.

Class: InetAddress

```
import java.net.*;
import java.util.*;

public class IPFinder
{
    public static void main(String[] args)
    {
        String host;
        Scanner input = new Scanner(System.in);

        System.out.print("\n\nEnter host name: ");
        host = input.next();
        try
        {
            InetAddress address =
                InetAddress.getByName(host);
            System.out.println("IP address: "
                + address.toString());
        }
        catch (UnknownHostException uhEx)
        {
            System.out.println("Could not find " + host);
        }
    }
}
```

Kendi IP adresini bulma örneği

```
import java.net.*;  
  
public class MyLocalIPAddress  
{  
    public static void main(String[] args)  
    {  
        try  
        {  
            InetAddress address =  
                InetAddress.getLocalHost();  
            System.out.println(address);  
        }  
        catch (UnknownHostException uhEx)  
        {  
            System.out.println(  
                "Could not find local address!");  
        }  
    }  
}
```

Javada Socket'ler

- TCP/IP sockets
- Datagram sockets (UDP sockets)
- Çoğunlukla her ikisinde de client/server ilişkisi bulunmaktadır.

TCP Sockets

- **connection-orientated** bağlantı sağlar.
- Bir taraf bağlantı sonlandırma işlemini başlatmadığı sürece kalıcı bağlantı sağlar.
- Server ve Client tarafı için farklı çalışan iki yapı bulunmaktadır.
- Server tarafının hazırlanması 5 adımı içerir.

TCP Sockets: Server Hazırlanması

- 1. ServerSocket nesnesinin oluşturulması.

```
ServerSocket servSock = new ServerSocket(1234);
```

- 2. Serverin gelen isteklerin dinleme moduna geçirilmesi.

```
Socket link = servSock.accept();
```

- 3. Giriş/Çıkış Streamlerinin hazırlanması

```
Scanner input = new Scanner(link.getInputStream());  
PrintWriter output =  
    new PrintWriter(link.getOutputStream(), true);
```

TCP Sockets: Server Hazırlanması

- 4. Veri gönderimi / alımı

```
output.println("Awaiting data...");  
String input = input.nextLine();
```

- 5. Haberleşme bittiğinde hattın kapatılması.

```
link.close();
```

- Örnek: TCPEchoServer: Gönderilen mesajı geri gönderen bir sunucu örneği.

TCP Sockets: Client Hazırlanması

- 1. Sunucuya bir bağlantı hazırlanması.
 - Sunucu IP bilgisi (InetAddress)
 - ilgili Sunucu Port bilgisi

```
Socket link =  
    new Socket(InetAddress.getLocalHost(), 1234);
```

- 2. Giriş Çıkış Streamlerinin ayarlanması.
- 3. Veri Gönderimi/Alımı
- 4. Hattın kapatılması.
- Örnek TCPEchoClient

Datagram (UDP) Sockets

- **Connectionless** (bağlantısız) iletişim kurar.
- İstemci / Sunucu arasında bilgi alışverişi olacak süre boyunca bir bağlantı tutulmaz.
- **Unreliable**(Güvenilmez) , TCP'ye göre daha hızlı iletişim sunar.
- Her paket (**datagram**) izole bir şekilde iletılır.
- TCP/IP Socket'den farkı sunucu tarafında **ServerSocket** yerine **DatagramSocket** nesnesi kullanıdır.

UDP Sockets: Sunucu Hazırlanması

- 1. DatagramSocket oluşturulması

```
DatagramSocket datagramSocket =  
    new DatagramSocket(1234);
```

- 2. Gelen datagramlar için incoming buffer oluşturulması.

```
byte[] buffer = new byte[256];
```

- Gelen datagramları kabul edecek DatagramPacket nesnesi oluşturulması.

```
DatagramPacket inPacket =  
    new DatagramPacket(buffer, buffer.length);
```

UDP Sockets: Sunucu Hazırlanması

- 4. Gelen verinin alınması

```
datagramSocket.receive(inPacket);
```

- 5. Gönderen Port ve IP bilgilerinin alımı.

```
InetAddress clientAddress = inPacket.getAddress();  
int clientPort = inPacket.getPort();
```

- 6. Bufferdan verinin okunması.

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

- 7. Cevap datagramının oluşturulması.

- (burada response cevabı tutan bir string'dir.)

```
DatagramPacket outPacket =  
    new DatagramPacket(response.getBytes(),  
                      response.length(),clientAddress, clientPort);
```

UDP Sockets: Sunucu Hazırlanması

- 8. Cevap Datagram'ın gönderimi.

```
datagramSocket.send(outPacket);
```

- 9. DatagramSocket'in kapatılması. (Normal bir süreç ilk 8 adımda bitecektir.)

```
datagramSocket.close();
```

- 4-8 arasındaki adımlar sonsuz bir döngüde sürekli bekleme aşamasında kalacak bir sunucu oluşturabilir.
- Örnek : UDPEchoServer

UDP Sockets: İstemci Hazırlanması

- 1. DatagramSocket nesnesinin hazırlanması.

```
DatagramSocket datagramSocket = new DatagramSocket();
```

- 2. Çıkış datagramının hazırlanması

```
DatagramPacket outPacket =
    new DatagramPacket(message.getBytes(),
                      message.length(), host, PORT);
```

- Sunucunun 7.maddesi ile aynıdır.
- 3. Datagram Mesajının gönderilmesi.

```
datagramSocket.send(outPacket);
```

- 4. Gelen datagramlar için buffer oluşturulması.

```
byte[] buffer = new byte[256];
```

UDP Sockets: İstemci Hazırlanması

- 5. Gelen datagramlar için DatagramPacket nesnesi oluşturulması.

```
DatagramPacket inPacket =  
    new DatagramPacket(buffer, buffer.length);
```

- 6. Datagramın okunması.

```
datagramSocket.receive(inPacket);
```

- 7. Bufferdan verinin okunması.

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

- DatagramSocket'in kapatılması.

```
datagramSocket.close();
```

- Örnek : UDPEchoClient

GUI örneği.

- DayTime Port 13 ile saat sunucusuna bağlanmaya çalışacak.
- Ör sunucu : ivy.shu.ac.uk
- Firewall engelleyebilir.
- Kendi DayTime sunucumuzu oluşturalım.
 - DaytimeServer
- Port Scanner Uygulaması.