Sockets

s20

Sockets

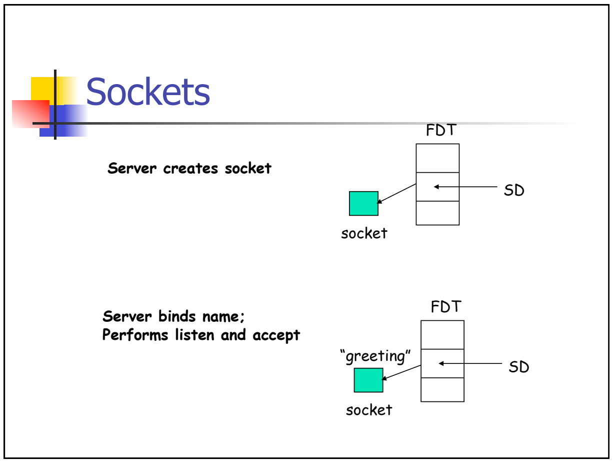
- Introduced by the Berkeley version of Unix
- Attempts to make IPC look like file I/O
- Use some of the same file operations e.g. read, write, close

Sockets

- Domains specify where two processes can communicate
- There are two domains
  - Unix domain
  - Internet domain

Sockets

- The socket type defines the communication access method
- There are three main types
  - Stream sockets
  - Datagram sockets
  - Raw sockets



Sockets

Server creates socket

FDT

SD

socket

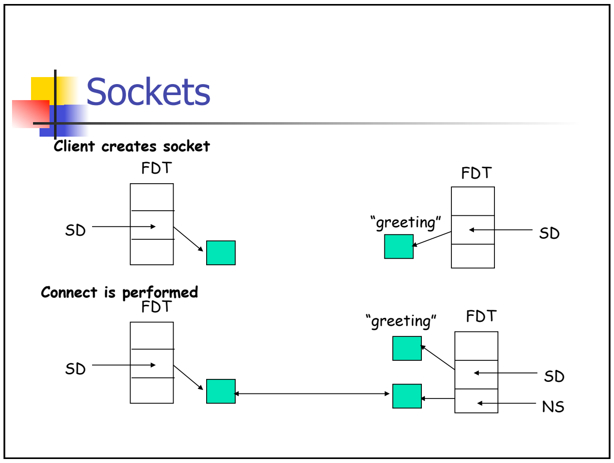
Server binds name;
Performs listen and accept

FDT

"greeting"

SD

socket



Sockets

Client creates socket

FDT

SD

FDT

"greeting"

SD

Connect is performed

FDT

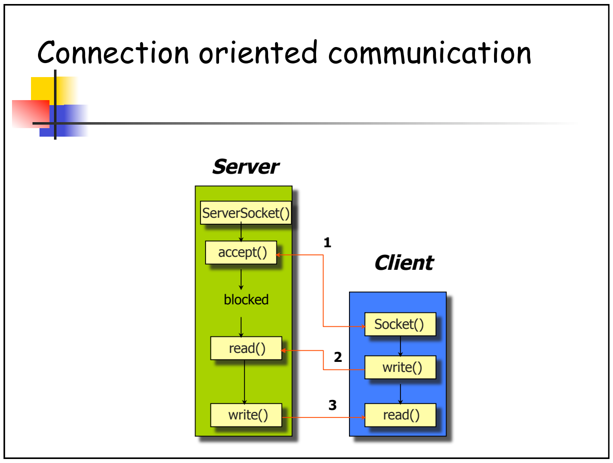
SD

"greeting"

FDT

SD

NS



Connection oriented communication

Server

ServerSocket()

accept()

blocked

read()

write()

Client

Socket()

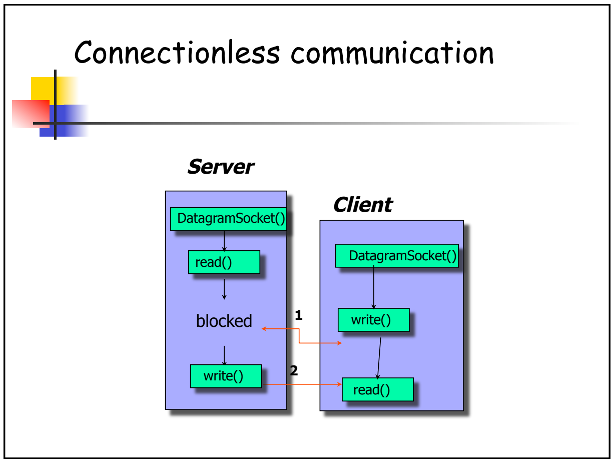
write()

read()

1

2

3



Connectionless communication

Server

DatagramSocket()

read()

blocked

write()

Client

DatagramSocket()

write()

read()

1

2

Reading from sockets

- Object Streams
  - Data is sent and received as object streams
  - Easiest way of sending/receiving data
  - Platform dependent
  - Classes
    - java.io.ObjectInputStream
    - java.io.ObjectOutputStream

Reading from sockets

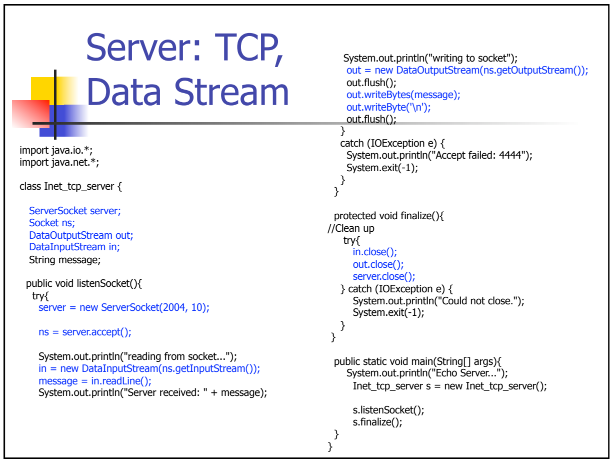
- Data Streams
  - Data is sent and received as data streams
  - Lower level for sending/receiving data
  - Platform independent
  - Classes
    - java.io.DataInputStream
    - java.io.DataOutputStream

Server: TCP, Object Streams

```
import java.io.ObjectOutputStream;
import java.io.ObjectInputStream;
import java.io.IOException;
import java.net.ServerSocket;
import java.net.Socket;

class Inet_tcp_server {

  ServerSocket  server;
  Socket  ns;
  ObjectOutputStream out;
  ObjectInputStream in;
  String message;

 public void listenSocket(){
  try{
    server = new ServerSocket(2004, 10);
    ns = server.accept();

    System.out.println("reading from socket...");
    in = new ObjectInputStream(ns.getInputStream());
    message = (String) in.readObject();
    System.out.println("Server received: " + message);
    System.out.println("writing to socket");
    out = new ObjectOutputStream(ns.getOutputStream());
    out.flush();
    out.writeObject(message);
    out.flush();
  }
  catch (IOException e) {
    System.out.println("Accept failed: 2004");
    System.exit(-1);
  }
  catch(ClassNotFoundException classNotFoundException) {
    System.out.println("Class not found error");
    System.exit(-1);

  }
 }

 protected void finalize(){
//Clean up
    try{
      in.close();
      out.close();
      server.close();
 } catch (IOException e) {
      System.out.println("Could not close.");
      System.exit(-1);
  }
 }

 public static void main(String[] args){
    System.out.println("Echo Server...");
      Inet_tcp_server s = new Inet_tcp_server();

      s.listenSocket();
      s.finalize();
 }
}
```

Client: TCP, Object Streams

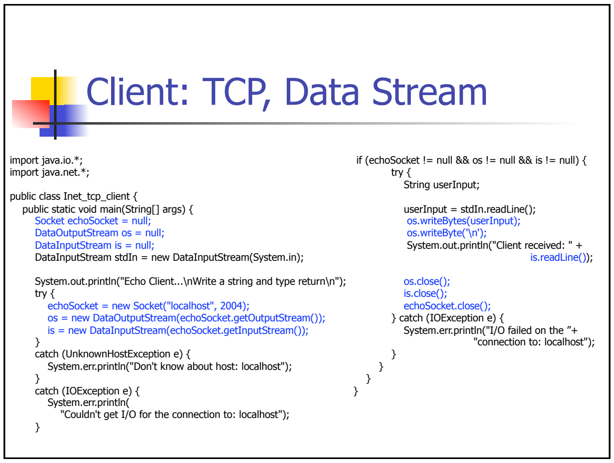
```
import java.io.ObjectOutputStream;
import java.io.ObjectInputStream;
import java.io.DataInputStream;
import java.io.IOException;
import java.net.ServerSocket;
import java.net.Socket;
import java.net.UnknownHostException;


public class Inet_tcp_client {
  public static void main(String[] args) throws IOException {
    Socket echoSocket = null;
    ObjectOutputStream os = null;
    ObjectInputStream is = null;
    DataInputStream stdIn = new DataInputStream(System.in);

    System.out.println("Echo Client...\nWrite a string and type return\n");
    try {
      String userInput = stdIn.readLine();

      echoSocket = new Socket("localhost", 2004);
      //echoSocket = new Socket("127.0.0.1", 2004); //also works
      os = new ObjectOutputStream(
                echoSocket.getOutputStream());
      os.flush();
      os.writeObject(userInput);
      os.flush();

      is = new ObjectInputStream(
                  echoSocket.getInputStream());
      System.out.println("Client received: " +
                        (String) is.readObject());

      os.close();
      is.close();
      echoSocket.close();
    } catch (UnknownHostException e) {
      System.err.println(
                "Don't know about host: localhost");
    } catch (IOException e) {
      System.err.println(
     "Couldn't get I/O for the connection to: localhost");
    }
    catch(ClassNotFoundException
                        classNotFoundException) {
     System.out.println("Class not found error");
     System.exit(-1);
    }
  }
}
```



Server: TCP, Data Stream

```
import java.io.*;
import java.net.*;

class Inet_tcp_server {

  ServerSocket server;
  Socket ns;
  DataOutputStream out;
  DataInputStream in;
  String message;

 public void listenSocket(){
  try{
   server = new ServerSocket(2004, 10);

   ns = server.accept();

   System.out.println("reading from socket...");
   in = new DataInputStream(ns.getInputStream());
   message = in.readLine();
   System.out.println("Server received: " + message);
```

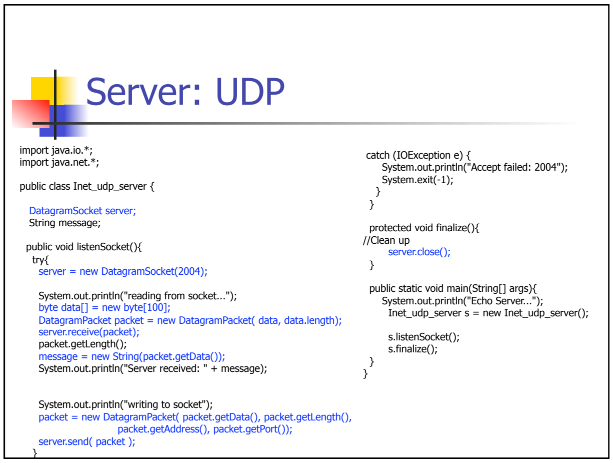
```
   System.out.println("writing to socket");
    out = new DataOutputStream(ns.getOutputStream());
    out.flush();
    out.writeBytes(message);
    out.writeByte('\n');
    out.flush();
   }
   catch (IOException e) {
    System.out.println("Accept failed: 4444");
    System.exit(-1);
   }
  }

  protected void finalize(){
 //Clean up
    try{
      in.close();
      out.close();
      server.close();
   } catch (IOException e) {
      System.out.println("Could not close.");
      System.exit(-1);
   }
 }

  public static void main(String[] args){
    System.out.println("Echo Server...");
     Inet_tcp_server s = new Inet_tcp_server();

     s.listenSocket();
     s.finalize();
  }
 }
```



Client: TCP, Data Stream

```
import java.io.*;
import java.net.*;

public class Inet_tcp_client {
   public static void main(String[] args) {
      Socket echoSocket = null;
      DataOutputStream os = null;
      DataInputStream is = null;
      DataInputStream stdIn = new DataInputStream(System.in);

      System.out.println("Echo Client...\nWrite a string and type return\n");
      try {
         echoSocket = new Socket("localhost", 2004);
         os = new DataOutputStream(echoSocket.getOutputStream());
         is = new DataInputStream(echoSocket.getInputStream());
      }
      catch (UnknownHostException e) {
         System.err.println("Don't know about host: localhost");
      }
      catch (IOException e) {
         System.err.println(
            "Couldn't get I/O for the connection to: localhost");
      }
```

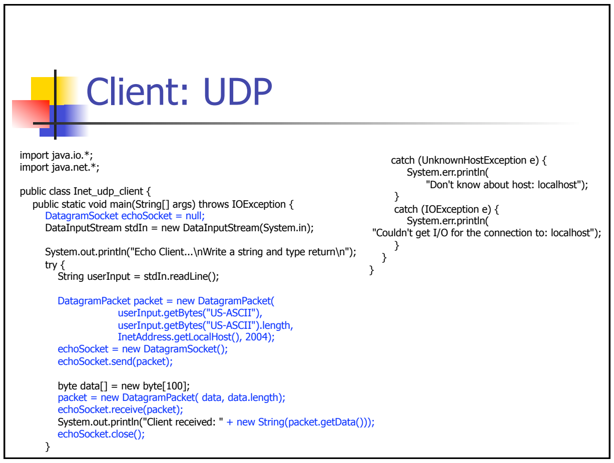
```
if (echoSocket != null && os != null && is != null) {
         try {
            String userInput;

            userInput = stdIn.readLine();
             os.writeBytes(userInput);
             os.writeByte('\n');
             System.out.println("Client received: " +
                                            is.readLine());

            os.close();
            is.close();
            echoSocket.close();
         } catch (IOException e) {
            System.err.println("I/O failed on the "+
                                 "connection to: localhost");
         }
      }
   }
}
```



Server: UDP

```
import java.io.*;
import java.net.*;

public class Inet_udp_server {

  DatagramSocket server;
  String message;

 public void listenSocket(){
  try{
   server = new DatagramSocket(2004);

   System.out.println("reading from socket...");
   byte data[] = new byte[100];
   DatagramPacket packet = new DatagramPacket( data, data.length);
   server.receive(packet);
   packet.getLength();
   message = new String(packet.getData());
   System.out.println("Server received: " + message);


   System.out.println("writing to socket");
   packet = new DatagramPacket( packet.getData(), packet.getLength(),
                        packet.getAddress(), packet.getPort());
   server.send( packet );
  }
```

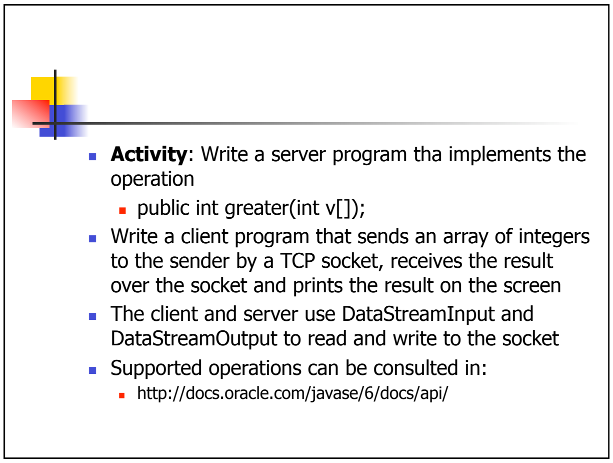
```
  catch (IOException e) {
     System.out.println("Accept failed: 2004");
     System.exit(-1);
   }
  }

  protected void finalize(){
 //Clean up
       server.close();
  }

  public static void main(String[] args){
     System.out.println("Echo Server...");
       Inet_udp_server s = new Inet_udp_server();

       s.listenSocket();
       s.finalize();
  }
 }
```



Client: UDP

```
import java.io.*;
import java.net.*;

public class Inet_udp_client {
   public static void main(String[] args) throws IOException {
      DatagramSocket echoSocket = null;
      DataInputStream stdIn = new DataInputStream(System.in);

      System.out.println("Echo Client...\nWrite a string and type return\n");
      try {
         String userInput = stdIn.readLine();

         DatagramPacket packet = new DatagramPacket(
                        userInput.getBytes("US-ASCII"),
                        userInput.getBytes("US-ASCII").length,
                        InetAddress.getLocalHost(), 2004);
         echoSocket = new DatagramSocket();
         echoSocket.send(packet);

         byte data[] = new byte[100];
         packet = new DatagramPacket( data, data.length);
         echoSocket.receive(packet);
         System.out.println("Client received: " + new String(packet.getData()));
         echoSocket.close();
      }
```

```
      catch (UnknownHostException e) {
         System.err.println(
            "Don't know about host: localhost");
      }
      catch (IOException e) {
         System.err.println(
   "Couldn't get I/O for the connection to: localhost");
      }
   }
}
```



- **Activity**: Write a server program tha implements the operation
  - public int greater(int v[]);
- Write a client program that sends an array of integers to the sender by a TCP socket, receives the result over the socket and prints the result on the screen
- The client and server use DataStreamInput and DataStreamOutput to read and write to the socket
- Supported operations can be consulted in:
  - http://docs.oracle.com/javase/6/docs/api/