CS 580 Client-Server Programming
Spring Semester, 2009
Doc 3 Intro to Client-Server
Jan 29, 2009

Reading

Java

Java Network Programming, Harold 3rd Ed,

Chapter 4 & 10

Ruby

Programming Ruby, Thomas, 2'ed

Chapter 10 Basic Input & Output

Class IO documentation (pp 503-515)

IPSocket & TCPSocket in Appendix A

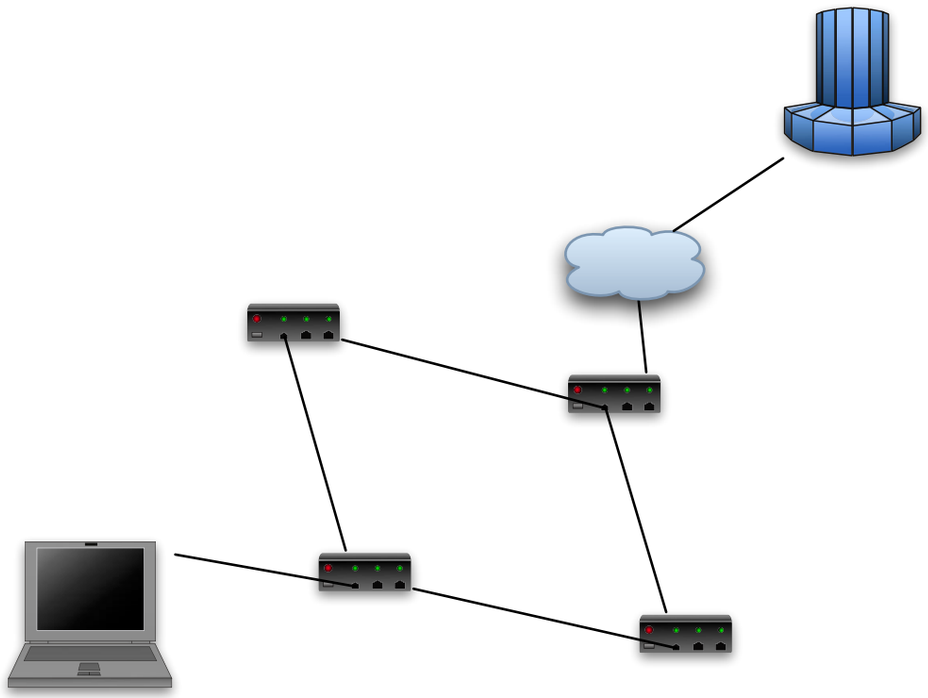
Smalltalk

BasicLibraries.pdf in VisualWorks docs directory

Chapter 9, Socket Programming,

Chapter 2, Streams, BasicLibraries.pdf

Network Overview



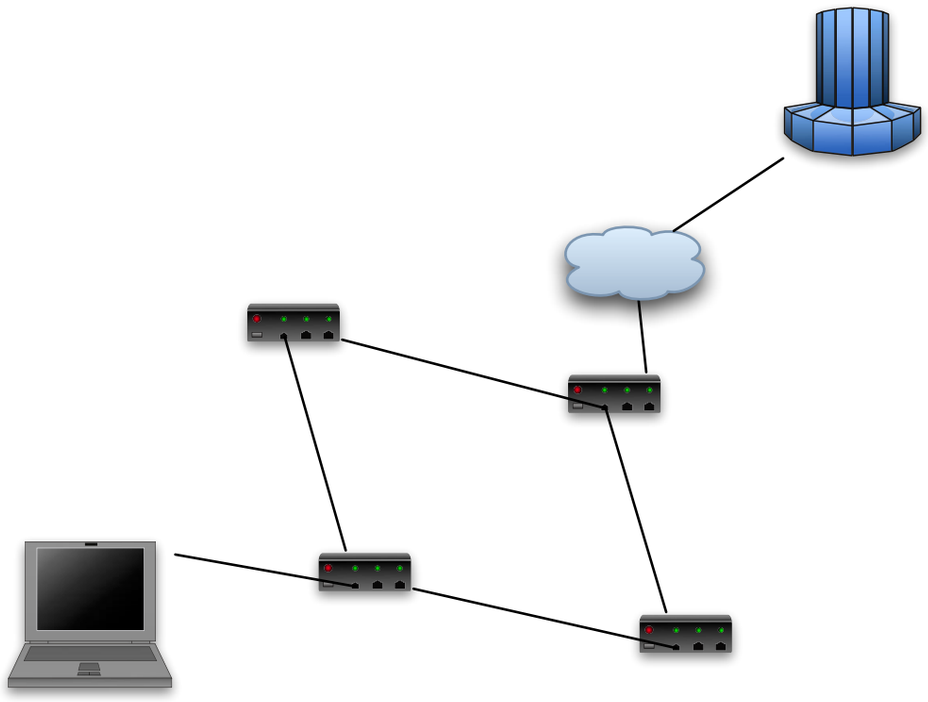
Messages divided into packets

Each packet routed separately

Routing Issues

Overhead issues

UDP



Fast

Packets are treated individually
Packets may arrive out of order
Packets may be lost

Client & Server must handle resulting problems

Used by:

Games
NFS

TCP

Handles lost packets

Handles packet order

TCP has delays
- Starting of connection
- Closing of connection
- Resending packets

Client & Sever don't have to deal with
- Packet order
- Packet loss

IP Addresses

IP address is currently a 32-bit number

130.191.3.100 (Four 8 bit numbers)

IPv6 uses 128 bit numbers for addresses

105.220.136.100.0.0.0.0.0.0.18.128.140.10.255.255

69DC:8864:0:0:0:1280:8C0A:FFFF

69DC:8864::1280:8C0A:FFFF

Machines on a network need a unique IP address

What is the difference between

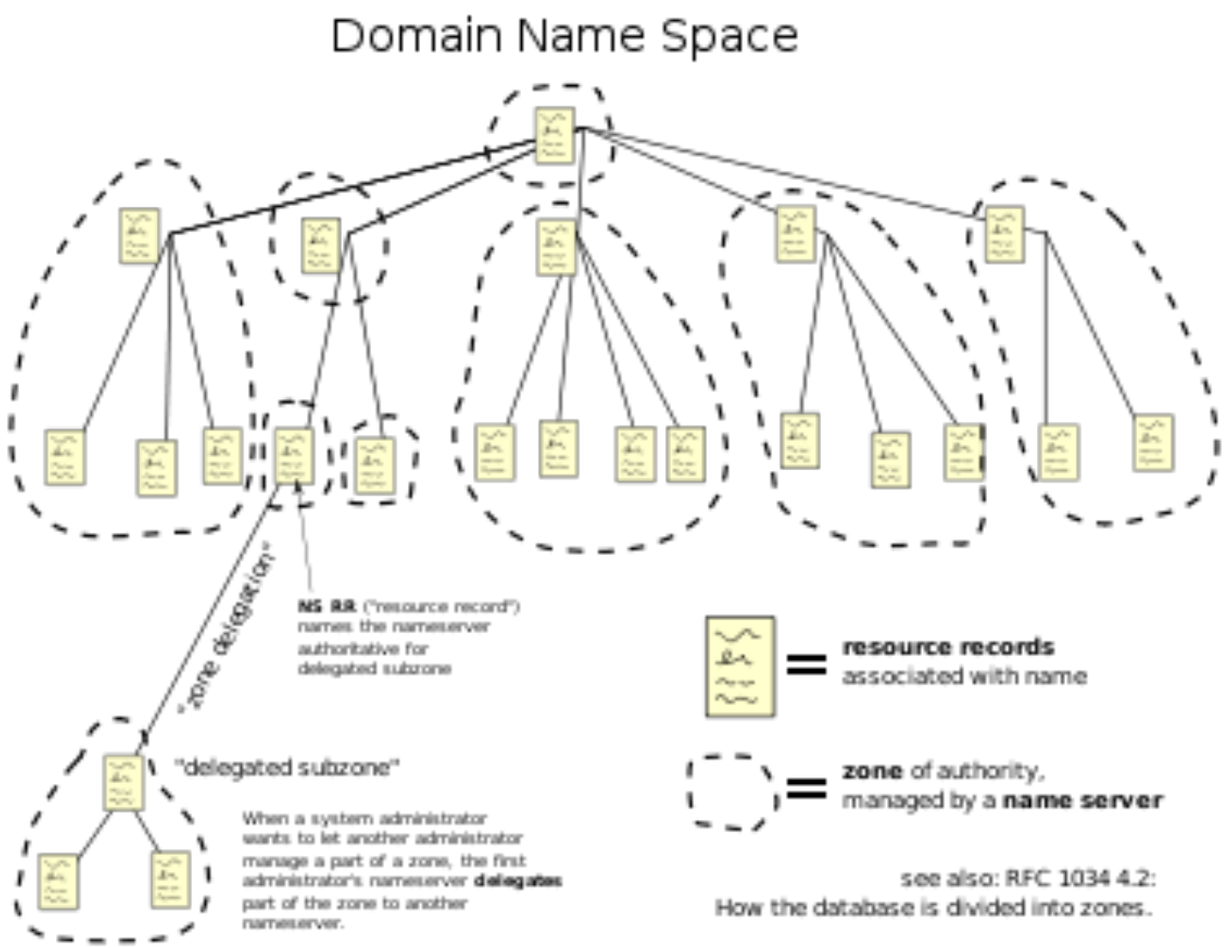
MAC address

IP address

Domain Name System (DNS)

Maps machine names to IP addresses

Internet Corporation for Assigned Names and Numbers (ICANN http://www.icann.org/) oversees assigning TLDs



Graphic is from http://en.wikipedia.org/wiki/Domain_name_system

Unix "host" command

Shows mapping between machine names and IP address

->host rohan.sdsu.edu
rohan.sdsu.edu has address 130.191.3.100

->host 130.191.3.100
100.3.191.130.IN-ADDR.ARPA domain name pointer rohan.sdsu.edu

Ports

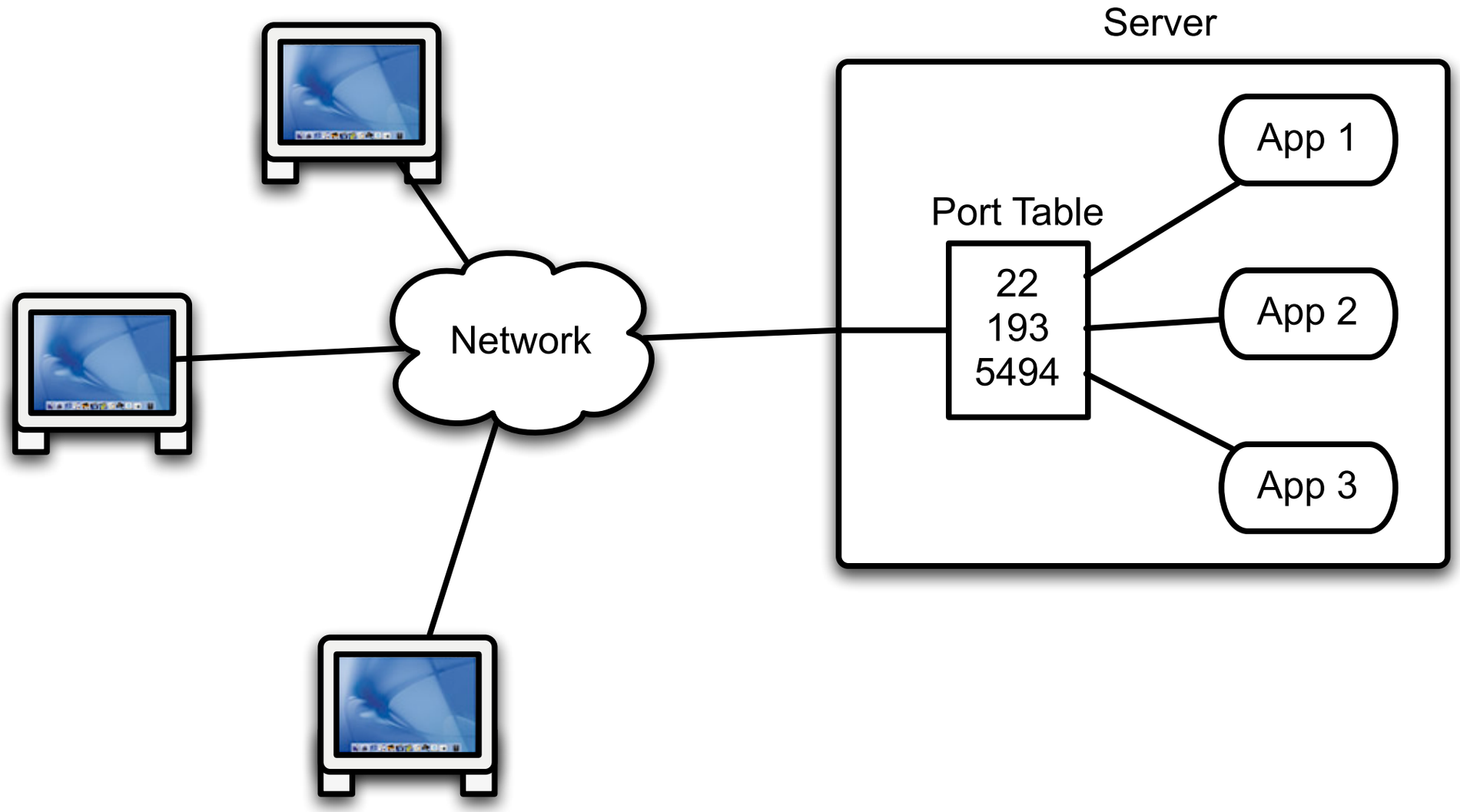
TCP/IP supports multiple logical communication channels called ports

Ports are numbered from 0 - 65535

A connection between two machines is uniquely defined by:

Protocol (TCP or UDP)
IP address of local machine
Port number used on the local machine
IP address of remote machine
Port number used on the remote machine

How Ports Work



When a client connects to a server is has to specify a machine and a port. The OS on the server keeps a table of port numbers and applications (sockets from the program) associated with each port number. When a client request comes in the OS will forward the request to the socket associated with the port number if one is associated (connected) with the port. A similar thing happens on the client side. When you open a socket on the client to connect to the server, the client socket is assigned a port on the client machine. When the server responds to the client it sends the response to that port on the client machine.

Some Port Numbers

Well known Ports	1-1023
Registered Ports	1024-49151
Dynamic/Private Ports	49152-65535

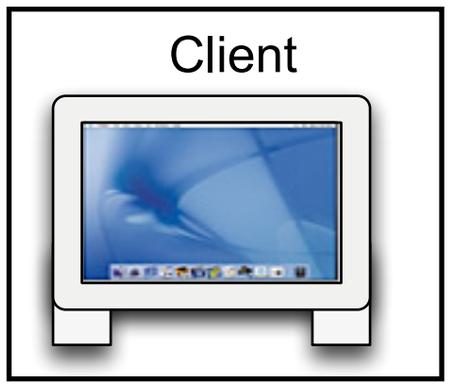
For a local list of services
file://rohan.sdsu.edu/etc/services

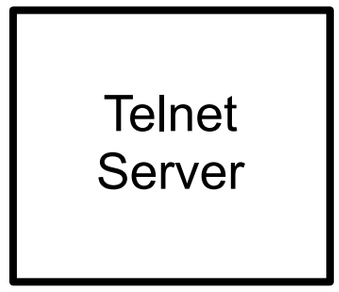
For a complete list see:
http://www.iana.org/assignments/port-numbers

See IANA numbers page http://www.iana.org/numbers.html for more information about protocol numbers and assignment of services

Service	Port
echo	7
discard	9
ftp	21
ssh	22
telnet	23
smtp	25
time	37
http	80
pop	110
https	443
doom	666
mysql	3306
postgresql	5432
gnutella	6346 6347

What is Telnet?





Protocol

Send text between client & server

Server

Requests login

Sends text to shell to be executed

Returns result of commands

Client

Transfers text between user and server

Telnet & Other Text-based Protocols

rohan 37 -> **telnet www.eli.sdsu.edu 80**
GET /courses/spring06/cs580/index.html   HTTP/1.0 <CR>
<CR>

Note <CR> indicates were you need to hit return

rohan 38->**telnet cs.sdsu.edu 110**
Trying 130.191.226.116...
Connected to cs.sdsu.edu.
Escape character is '^]'.
+OK QPOP (version 3.1.2) at sciences.sdsu.edu starting.
USER whitney
+OK Password required for whitney.
PASS typeYourPasswordHere
+OK whitney has 116 visible messages (0 hidden) in 640516 octets.

Simple Date Example - Protocol

Client Commands	Server Response
"date" ended by line feed "date\n"	current date ended by line feed "January 30, 2007\n"
"time" ended by line feed "time\n"	Current time ended by line feed "6:58 pm\n"

Server listens for an incoming request

On request

- reads command
- returns response
- closes connection

On client errors - action not specified

Beware

Can only send bytes across network

Client & server maybe different hardware platforms

What is a newline?

End-of-file indicates connection is closed

Sample Java Client

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

class DateClient {
        String server;
        int port;

        public DateClient(String serverAddress, int port) {
                server = serverAddress;
                this.port = port;
        }

        public String date() {
                return send("date\n");
        }

        public String time() {
                return send("time\n");
        }
```

Java Client Continued

```
private String send(String text) {
        try {
                Socket connection = new Socket(server, port);
                OutputStream rawOut = connection.getOutputStream();
                PrintStream out = new PrintStream(new BufferedOutputStream(rawOut));
                InputStream rawIn = connection.getInputStream();
                BufferedReader in = new BufferedReader(new
InputStreamReader(rawIn));

                out.print(text);
                out.flush();
                String answer = in.readLine();
                out.close();
                in.close();
                return answer;
            }
            catch (IOException e) {
                return "Error in connecting to server";
            }
        }
    }
```


Bad very bad - using PrintStream for network code.

Running the Client

```
System.out.println("hi");
DateClient client = new DateClient("127.0.0.1", 4444);
System.out.println( client.date());
System.out.println( client.time());
```

Issue - Avoid Small Packets

```
OutputStream rawOut = connection.getOutputStream();
PrintStream out = new PrintStream(new BufferedOutputStream(rawOut));
```


Use buffers

Issue - Actually Send the request

```
out.flush();
```

Issue - Client will not work on all platforms

String answer = in.readLine();

What ends a line?

Don't Do this

```
String answer = in.readLine();
```


I did it to keep the example small. One can not get much code on a slide using 24 point font. Plus the Ruby example is sorter than this.

Issue - Close the connection when done

```
out.close();
in.close();
```

Issue - Testing

How does one test the client?

Issue - Background material

Java

Streams

Read Chapter 4

Sockets

Read Chapter 10

Java Network Programming, Harold 3rd Ed

Ruby Client

```
require 'socket'

class DateClient
    def initialize(serverAddress, port)
      @server = serverAddress
      @port = port
    end

    def date()
      send("date\n")
    end

    def time()
      send("time\n")
    end

  private
    def send(text)
      connection = TCPSocket.new(@server, @port)
      connection.send(text, 0)
      answer = connection.gets("\n")
      connection.close
      answer
    end
end
```

Running the client

```
client = DateClient.new("127.0.0.1", 4444)
puts client.date
puts client.time
```

Issues - Using Standard IO Methods

```
def send(text)
    connection = TCPSocket.new(@server, @port)
    connection.print(text)
    connection.flush
    answer = connection.gets("\n")
    connection.close
    answer
  end
```

Ruby Background

Sockets

Read IPSocket & TCPSocket in Appendix A

IO

Chapter 10 Basic Input & Output
Class IO documentation (pp 503-515)

Programming Ruby, Thomas, 2'ed

Smalltalk Client

```
Smalltalk defineClass: #NonGUITimeDateClient
        superclass: #{Core.Object}
        instanceVariableNames: 'port server '

                                Class Method
server: aMachineAddress port: anInteger
        ^self new setServer: aMachineAddress port: anInteger

                                Instance Methods
setServer: aMachineAddress port: anInteger
        server := aMachineAddress.
        port := anInteger

date
                ^self send: 'date'.

time
        ^self send: 'time'
```

Smalltalk Client

```
send: message
        | input timeSocket response |
        timeSocket := SocketAccessor
          newTCPclientToHost:  server
          port:  port asNumber.
        input := timeSocket readAppendStream.
        input
              nextPutAll: message;
              cr;
              commit.
    response := input upTo: Character lf.
    input close.
    ^response
```

Smalltalk Background

Sockets

Chapter 9, Socket Programming, BasicLibraries.pdf
VisualWorks docs directory

IO

Chapter 2, Streams, BasicLibraries.pdf

Server

Basic Algorithm

```
while (true) {
        Wait for an incoming request;
        Perform whatever actions are requested;
      }
```

Basic Server Issues

How to wait for an incoming request?
How to know when there is a request?
What happens when there are multiple requests?
How do clients know how to contact server?
How to parse client request?
How do we know when the server has the entire request?

Java Date Server

```
public class DateServer {
        private static Logger log = Logger.getLogger("dateLogger");

        public static void main (String args[]) throws IOException {
                ProgramProperties flags = new ProgramProperties( args);
                int port = flags.getInt( "port" , 8765);
                new DateServer().run(port);
        }

        public void run(int port) throws IOException {
                ServerSocket input = new ServerSocket( port );
                log.info("Server running on port " + input.getLocalPort());

                while (true) {
                        Socket client = input.accept();
                        log.info("Request from " + client.getInetAddress());
                        processRequest(
                                client.getInputStream(),
                                client.getOutputStream());
                        client.close();
                }
        }
```

Java Date Server Continued

```
void processRequest(InputStream in,OutputStream out)
        throws IOException {

        BufferedReader parsedInput =
                new BufferedReader(new InputStreamReader(in));

        boolean autoflushOn = true;
        PrintWriter parsedOutput = new PrintWriter(out,autoflushOn);

        String inputLine = parsedInput.readLine();

                if (inputLine.startsWith("date")) {
                        Date now = new Date();
                        parsedOutput.println(now.toString());
                }
        }
}
```

This server needs work

Starting the Server

```
rohan 16-> java -jar DateServer.jar
Feb 19, 2004 10:56:59 AM DateServer run
INFO: Server running on port 8765
```

Ruby Date Server

```
require 'socket'

class DateServer
  def initialize(port)
    @port = port
  end

  def run()
    server = TCPServer.new( @port)
    puts("start " + @port.to_s)
    while (session = server.accept)
      Thread.new(session) do |connection|
        process_request_on(connection)
        connection.close
      end
    end
  end
```

```
  private
    def process_request_on(socket)
      request = canonical_form( socket.gets("\n") )
      now = Time.now
      answer = case request
        when 'time'
          now.strftime("%X")
        when 'date'
          now.strftime("%x")
        else
          "Invalid request"
      end
      socket.send(answer + "\n",0)
    end

    def canonical_form(string)
      string.lstrip.rstrip.downcase
    end
  end
```

Text

Smalltalk Date Server

```
Smalltalk defineClass: #SimpleDateServer
        superclass: #{Core.Object}
        instanceVariableNames: 'serverSocket activeProcess port isRunning '
```

Class Method

```
port: anInteger
        ^super new setPort: anInteger
```

Instance Methods

```
setPort: anInteger
        port :=anInteger.
        isRunning :=false.
        connectionCount := 0.

isRunning
        ^isRunning
```

Smalltalk Date Server

```
start
        serverSocket := SocketAccessor newTCPserverAtPort: port.
        serverSocket
                listenFor: 4;
                soReuseaddr: true.
        isRunning :=true.
        connectionCount :=0.
        activeProcess := [self run] forkAt: Processor lowIOPriority

stop
        isRunning
                ifTrue:
                        [activeProcess terminate.
                        serverSocket close.
                        activeProcess := nil.
                        serverSocket := nil].
        isRunning := false
```

Smalltalk Date Server

```
run
    | childSocket clientIOStream |

    [childSocket := serverSocket accept.
    clientIOStream := childSocket readAppendStream.
    clientIOStream lineEndTransparent.
    [self processRequestOn: clientIOStream]
        forkAt: Processor userBackgroundPriority ] repeat
```

Smalltalk Date Server

```
processRequestOn: anReadAppendStream
        | clientRequest answer dateAndTime |
        connectionCount :=connectionCount + 1.
        clientRequest := (anReadAppendStream next: 4) asLowercase.
        answer := ''.
        dateAndTime := Time dateAndTimeNow.
        (clientRequest sameAs: 'date')
                ifTrue: [answer := dateAndTime first printString].
        (clientRequest sameAs: 'time')
                ifTrue: [answer := dateAndTime last printString].
    anReadAppendStream
        nextPutAll: answer;
        nextPut: Character lf;
        cr;
        commit;
        close
```

Issue - Date Format

What format does the server use for time and date?

Clients need to know so can parse them