**InetServer.java**

// Importing Java Input Output and networking libraries

import java.io.\*;

import java.net.\*;

// Worker class

class Worker extends Thread {

// sock contains the current client connection

Socket sock;

// Worker Constructor

Worker (Socket s) {sock = s;}

// Run method will run when start() is called

public void run(){

// Defining I/O streams in/out to null

PrintStream out = null;

BufferedReader in = null;

// Trying to assign in/out to socket's(client) Input/Output stream and returning error if not possible

try {

in = new BufferedReader(new InputStreamReader(sock.getInputStream()));

out = new PrintStream(sock.getOutputStream());

// Trying to read client request and if it cannot be read returning an error

try {

String name;

// Reading client request with the help of in

name = in.readLine ();

System.out.println("Looking up " + name);

printRemoteAddress(name, out);

} catch (IOException x) {

System.out.println("Server read error");

x.printStackTrace ();

}

// Closing the connection with client, though server is still running

sock.close();

} catch (IOException ioe) {System.out.println(ioe);}

}

// This static method will print the output on client side with the help of out

static void printRemoteAddress (String name, PrintStream out) {

try {

out.println("Looking up " + name + "...");

InetAddress machine = InetAddress.getByName (name);

out.println("Host name : " + machine.getHostName ());

out.println("Host IP : " + toText (machine.getAddress ()));

} catch(UnknownHostException ex) {

out.println ("Failed in atempt to look up " + name);

}

}

static String toText (byte ip[]) {

StringBuffer result = new StringBuffer ();

for (int i = 0; i < ip.length; ++ i) {

if (i > 0) result.append (".");

result.append (0xff & ip[i]);

}

return result.toString ();

}

}

// Server class

public class InetServer {

// Main method

public static void main(String a[]) throws IOException {

// Number of request at a time is stored in a queue of length 6

int q\_len = 6;

int port = 1565;

Socket sock;

// Creating a server socket

ServerSocket servsock = new ServerSocket(port, q\_len);

System.out.println("Vatsal Parikh's Inet Server is starting up, listening at port 1565.\n");

// Waiting for a client connection

while (true) {

// sock will contain the client connection

sock = servsock.accept();

// Starting the thread(worker) to handle the request

new Worker(sock).start();

}

}

}

**InetClient.java**

// Importing Java Input Output and networking libraries

import java.io.\*;

import java.net.\*;

// Client class

public class InetClient{

// Main method

public static void main (String args[]) {

String serverName;

if (args.length < 1) serverName = "localhost";

else serverName = args[0];

System.out.println("Vatsal Parikh's Inet Client\n");

System.out.println("Using server: " + serverName + ", Port: 1565");

// Taking input from the user using InputStreamReader

BufferedReader in = new BufferedReader(new InputStreamReader(System.in));

// Trying to get all user inputs until user passes quit as input

try {

String name;

do {

System.out.print("Enter a hostname or an IP address, (quit) to end: ");

System.out.flush ();

name = in.readLine ();

// If user input is valid, getRemoteAddress method is called with user input and server name

if (name.indexOf("quit") < 0)

getRemoteAddress(name, serverName);

} while (name.indexOf("quit") < 0);

System.out.println ("Cancelled by user request.");

} catch (IOException x) {x.printStackTrace ();}

}

static String toText (byte ip[]) {

StringBuffer result = new StringBuffer ();

for (int i = 0; i < ip.length; ++ i) {

if (i > 0) result.append (".");

result.append (0xff & ip[i]);

}

return result.toString ();

}

// This static method will fetch the output from server side with the help of fromServer

static void getRemoteAddress (String name, String serverName){

// sock contains the current client connection

Socket sock;

// Defining I/O streams fromServer/toServer to null

BufferedReader fromServer;

PrintStream toServer;

String textFromServer;

// Trying to make a connection with server with the server name and port number

try{

sock = new Socket(serverName, 1565);

// Assigning fromServer/toServer to socket's(server) Input/Output stream

fromServer = new BufferedReader(new InputStreamReader(sock.getInputStream()));

toServer = new PrintStream(sock.getOutputStream());

// Sending machine name and ip address

toServer.println(name); toServer.flush();

// Reading the first 2-3 lines from server

for (int i = 1; i <=3; i++){

textFromServer = fromServer.readLine();

// If the line read is not null, it is printed on client side

if (textFromServer != null) System.out.println(textFromServer);

}

sock.close();

} catch (IOException x) {

System.out.println ("Socket error.");

x.printStackTrace ();

}

}

}