Taylor Pedretti, Jeremy Noble
CSE 461, Winter 2019
Lab 4 Report

Lab 4 Report

This lab is 100% complete and everything functions as it should. We ended up writing our own servers in C# that's running on a remote machine. One of the servers performs the simple calculator operations and the other generates the random numbers. We had a bit of trouble getting the client side to display the result because of timing issues. The client thread would make the remote call then return without getting the response from the server. We ended up time delaying the client 1.5 sec to allow the server time to finish. Other than that everything works great and the lab overall is a success.

Exercise 1: Simple Remote Calculator

Taylor Pedretti, Jeremy Noble
CSE 461, Winter 2019
Lab 4 Report

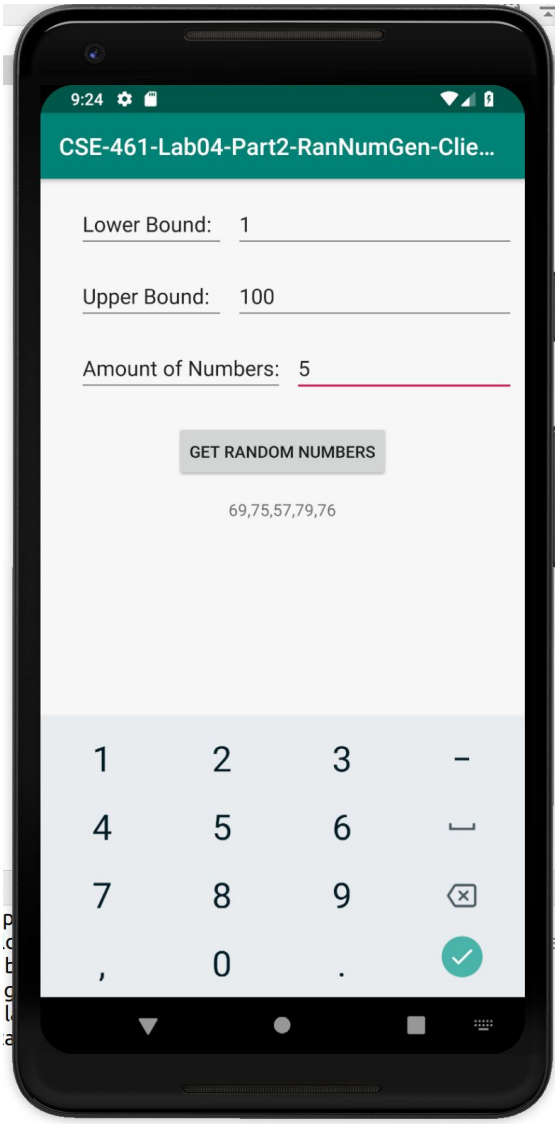
Partial Source Code:

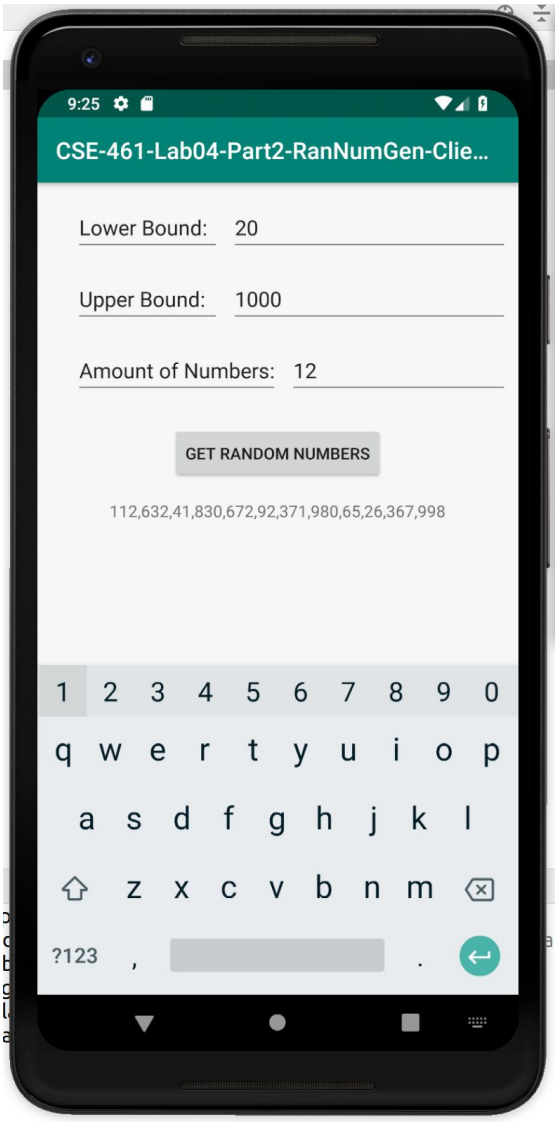
```
package com.example.cse_461_lab04_part1_simple_calculator_remote;

import android.app.Activity;
import android.os.AsyncTask;

import java.net.DatagramSocket;
import java.net.DatagramPacket;
import java.net.InetAddress;
import java.net.UnknownHostException;
import java.net.SocketException;
import java.io.IOException;
import android.os.Build;
import android.widget.TextView;

import com.example.cse_461_lab03_part2_simple_calculator.R;

public class Client
{
   private DatagramSocket socket;
   private InetAddress addr;
   private AsyncTask<Void, Void, Void> async_client;
   public String Message;
   String result = "";

   public Client()
   {
       try { socket = new DatagramSocket(); }
       catch(SocketException ex) { ex.printStackTrace(); }
   }

   public void SendMSG(Activity curAct)
   {
       TextView temp = curAct.findViewById(R.id.displayResult);
       async_client = new AsyncTask<Void, Void, Void>()
       {
           @Override
           protected Void doInBackground(Void... voids)
           {
               byte[] buf = Message.getBytes();

               try { addr = InetAddress.getByName("96.44.135.45"); }
               catch(UnknownHostException ex) { ex.printStackTrace(); }

               DatagramPacket packet = new DatagramPacket(buf, buf.length, addr, 1337);

               try
               {
                   socket.send(packet);
               }
               catch(IOException ex)
               {
                   ex.printStackTrace();
               }
```

```
                packet = new DatagramPacket(buf, buf.length);

                try
                {
                    socket.receive(packet);
                    result = new String(buf, 0, packet.getLength());

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

                return null;
            }
        };

        if(Build.VERSION.SDK_INT >= 11)
            async_client.executeOnExecutor(AsyncTask.THREAD_POOL_EXECUTOR);
        else
            async_client.execute();

        try {
            Thread.sleep(1500);
        }
        catch(InterruptedException ex) {
            ex.printStackTrace();
        }

        temp.setText(Message + "=" + result);
    }
}
```

Taylor Pedretti, Jeremy Noble
CSE 461, Winter 2019
Lab 4 Report

Exercise 2: Remote Number Generator





Partial Source Code:

```
package com.example.cse_461_lab04_part2_rannumgen_client;

import android.os.AsyncTask;
import android.os.Handler;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.text.TextUtils;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageButton;
import android.widget.TextView;
```

```
import java.io.IOException;
import java.lang.InterruptedException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.net.SocketException;
import java.net.UnknownHostException;
import java.nio.charset.Charset;
import java.util.Objects;

public class MainActivity extends AppCompatActivity implements View.OnClickListener
{
   EditText lowerB;
   EditText upperB;
   EditText amtNums;
   TextView ranNums;
   Button getNums;

   @Override
   protected void onCreate(Bundle savedInstanceState)
   {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.activity_main);

       lowerB = (EditText)findViewById(R.id.lowerBound);
       upperB = (EditText)findViewById(R.id.upperBound);
       amtNums = (EditText)findViewById(R.id.amtNumbers);
       ranNums = (TextView)findViewById(R.id.randomNumbers);
       getNums = (Button)findViewById(R.id.getRanNums);

       getNums.setOnClickListener(this);
   }

   public void onClick( View view )
   {
       Client rClient = new Client();
       rClient.Message += (lowerB.getText() + "," + upperB.getText() + "," + amtNums.getText());
       rClient.GetRandomNums(this);
   }
}

// Server
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

using System.Net;
using System.Net.Sockets;

namespace CSE_461_Lab04_Part2_RandomNumGen_Server
{
        class Program
```

```
        {
        private const int port = 1338;

        private static void Server()
        {
        UdpClient listner = new UdpClient(port);
        IPEndPoint groupEP = new IPEndPoint(IPAddress.Any, port);

        try
        {
                while(true)
                {
                Console.WriteLine("Waiting for info from client");
                byte[] clientRev = listner.Receive(ref groupEP);
                Console.WriteLine($"Received broadcast from {groupEP} :");
                Console.WriteLine($" {Encoding.UTF8.GetString(clientRev, 0, clientRev.Length)}");

                listner.Send(ranNumbers(Encoding.UTF8.GetString(clientRev, 0, clientRev.Length)),
                                Encoding.UTF8.GetString(clientRev, 0, clientRev.Length).Length,
groupEP);
                Console.WriteLine("Randoms numbers sent to client!");
                }
        }
        catch(SocketException ex)
        {
                Console.WriteLine(ex);
        }
        finally
        {
                listner.Close();
        }

        }

        private static byte[] ranNumbers(string input)
        {
        byte[] ranNums = null;
        string[] inputBounds = input.Split(',');
        List<String> ranList = new List<String>();
        string ranNumsString = "";
        Random r = new Random();

        for(int i = 0; i < Convert.ToInt32(inputBounds[2]); i++)
        {
                ranList.Add(r.Next(Convert.ToInt32(inputBounds[0]),
Convert.ToInt32(inputBounds[1])).ToString());
        }

        foreach(String t in ranList)
        {
                if(t == ranList.Last())
                {
                ranNumsString += t;
                }
                else
```

```
                {
                ranNumsString += (t + ",");
                }
        }

        Console.WriteLine("Random Numbers: " + ranNumsString);

        return (ranNums = Encoding.UTF8.GetBytes(ranNumsString.ToString()));
        }

        static void Main(string[] args)
        {
        Server();
        }
        }
}
```