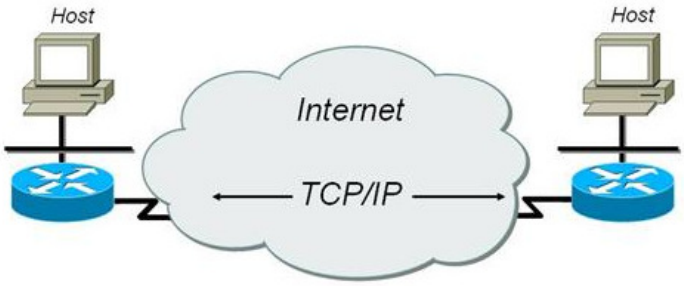
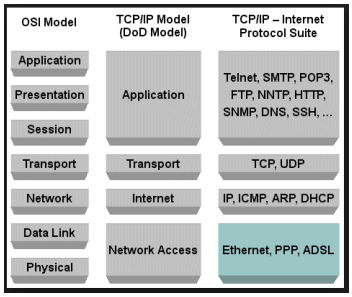
TCPIP Client/Server

The upper layers are client and server specific. The upper layers focus on the applications, which is entirely the responsibility of the developers. Common for all applications is the IP-address to which the application is connected and then opening the gate. Ports is enabled in all, 65536, of which the first 1024 are so called "Well Known Ports" (these ports need admin rights). The use of the ports try to organize the IANA, but everyone can select any virtual port for his needs without limitations.





Below an example of a program, which connects to a certain address and the port (socket):

The steps involved in establishing a socket on the *client* side are as follows:

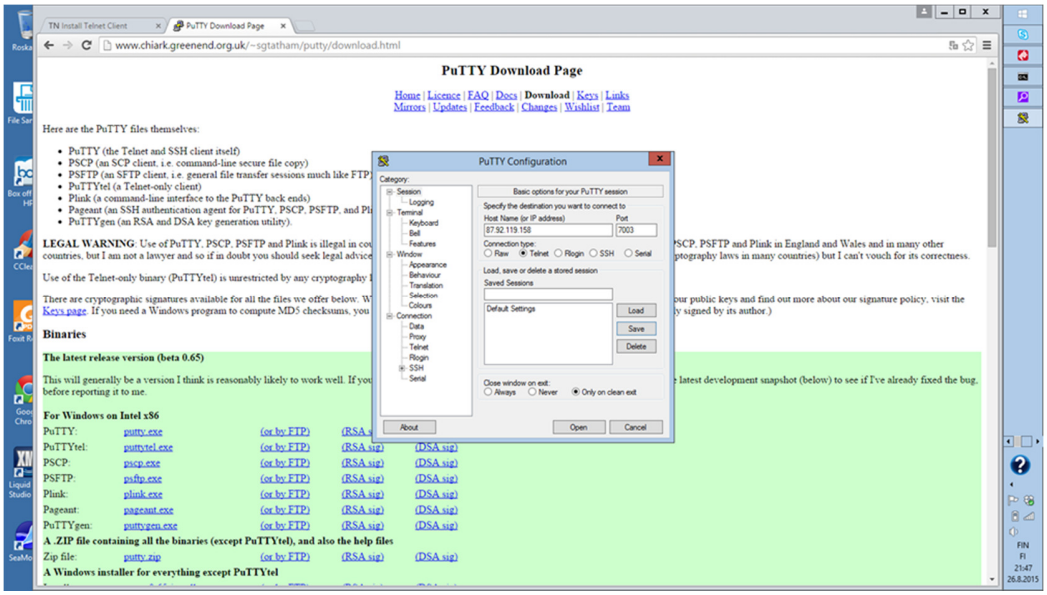
1. **Create a socket with the `socket()` system call**
2. **Connect the socket to the address of the server using the `connect()` system call**
3. **Send and receive data. There are a number of ways to do this, but the simplest is to use the `read()` and `write()` system calls.**

The steps involved in establishing a socket on the *server* side are as follows:

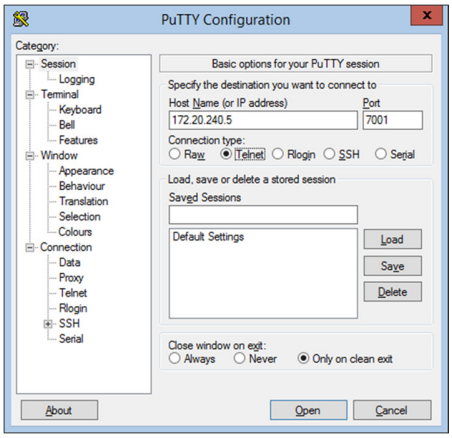
1. **Create a socket with the `socket()` system call**
2. **Bind the socket to an address using the `bind()` system call. For a server socket on the Internet, an address consists of a port number on the host machine.**
3. **Listen for connections with the `listen()` system call**
4. **Accept a connection with the `accept()` system call. This call typically blocks until a client connects with the server.**
5. **Send and receive data**

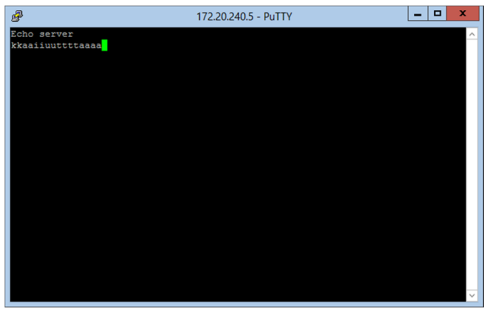
Exercise 3.1 Test the Telnet connection to Echoserver,

Get first Putty.exe (or similar) for Windows:



Test the connection to EchoServer at IP address: 85.23.10.191 and at port **7003**, use TCP connection (telnet). The server is ready to echo every keypress after TCP connection. (http://www.nodejs.org/.)

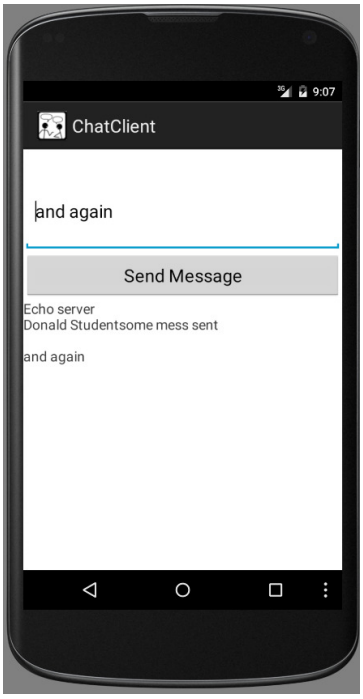




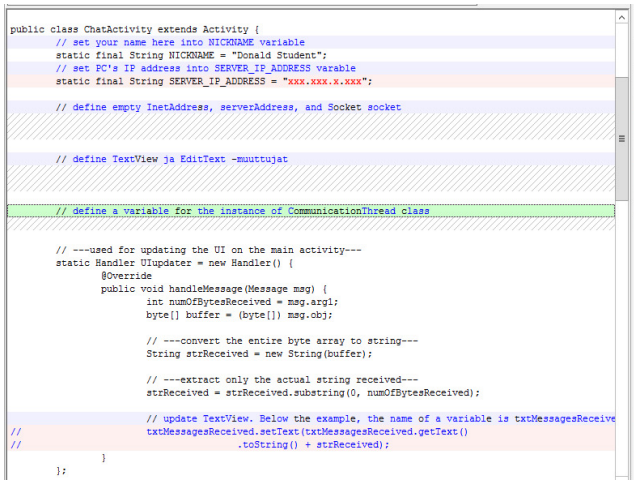
Exercise 3.2 Make an Android application ChatClient (the base application is in the source folder) and test on the application the same thing as earlier using EchoServer.

1. Open the project ChatClient by Android Studio.
2. Add permission for Internet in AndroidManifest.xml file.
3. Adjust activity's layout file (res/layout/activity_main.xml):
   - add EditText for writing the message
   - add Button for sending
   - add listener for button click (android:onClick="onClickSend")

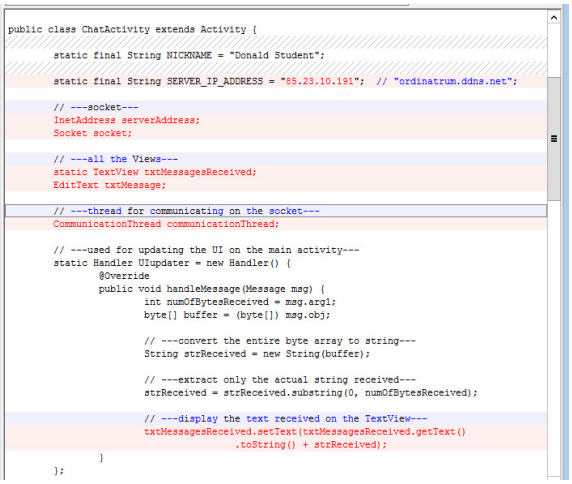
   TextView for echoing the messages is already done. Use LinearLayout.
4. Add a text for the button (res/values/strings.xml)
5. Add all the variables needed in Activity class (InetAddress, Socket, class variables ..).



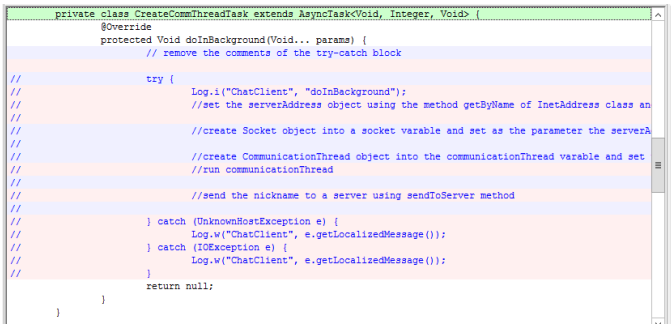
Solution tips:



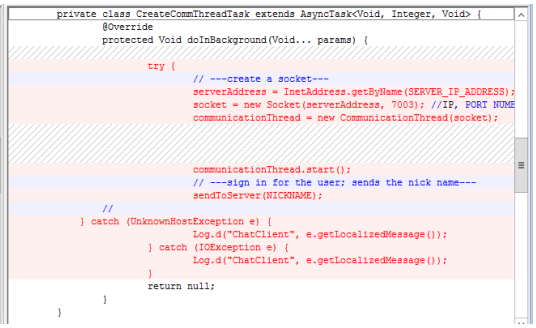
```
public class ChatActivity extends Activity {
        // set your name here into NICKNAME variable
        static final String NICKNAME = "Donald Student";
        // set PC's IP address into SERVER_IP_ADDRESS varable
        static final String SERVER_IP_ADDRESS = "xxx.xxx.x.xxx";

        // define empty InetAddress, serverAddress, and Socket socket



        // define TextView ja EditText -muuttujat


        // define a variable for the instance of CommunicationThread class


        // ---used for updating the UI on the main activity---
        static Handler UIupdater = new Handler() {
                @Override
                public void handleMessage(Message msg) {
                        int numOfBytesReceived = msg.arg1;
                        byte[] buffer = (byte[]) msg.obj;

                        // ---convert the entire byte array to string---
                        String strReceived = new String(buffer);

                        // ---extract only the actual string received---
                        strReceived = strReceived.substring(0, numOfBytesReceived);

                        // update TextView. Below the example, the name of a variable is txtMessagesReceive
//                      txtMessagesReceived.setText(txtMessagesReceived.getText()
//                                      .toString() + strReceived);
                }
        };
```



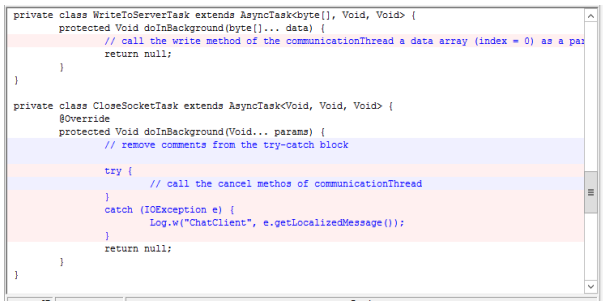
```
public class ChatActivity extends Activity {
        static final String NICKNAME = "Donald Student";
        static final String SERVER_IP_ADDRESS = "85.23.10.191";  // "ordinatrum.ddns.net";

        // ---socket---
        InetAddress serverAddress;
        Socket socket;

        // ---all the Views---
        static TextView txtMessagesReceived;
        EditText txtMessage;

        // ---thread for communicating on the socket---
        CommunicationThread communicationThread;

        // ---used for updating the UI on the main activity---
        static Handler UIupdater = new Handler() {
                @Override
                public void handleMessage(Message msg) {
                        int numOfBytesReceived = msg.arg1;
                        byte[] buffer = (byte[]) msg.obj;

                        // ---convert the entire byte array to string---
                        String strReceived = new String(buffer);

                        // ---extract only the actual string received---
                        strReceived = strReceived.substring(0, numOfBytesReceived);

                        // ---display the text received on the TextView---
                        txtMessagesReceived.setText(txtMessagesReceived.getText()
                                        .toString() + strReceived);
                }
        };
```



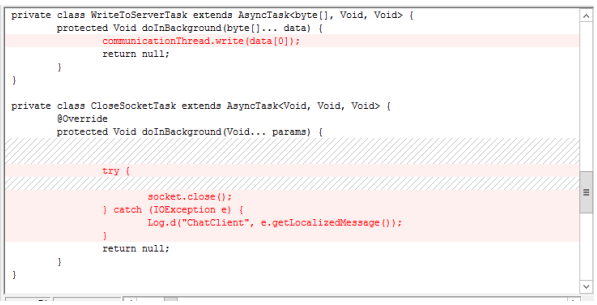
```
        private class CreateCommThreadTask extends AsyncTask<Void, Integer, Void> {
                @Override
                protected Void doInBackground(Void... params) {
                        // remove the comments of the try-catch block

//                      try {
//                              Log.i("ChatClient", "doInBackground");
//                              //set the serverAddress object using the method getByName of InetAddress class an
//
//                              //create Socket object into a socket varable and set as the parameter the serverA
//
//                              //create CommunicationThread object into the communicationThread varable and set
//                              //run communicationThread
//
//                              //send the nickname to a server using sendToServer method
//
//                      } catch (UnknownHostException e) {
//                              Log.w("ChatClient", e.getLocalizedMessage());
//                      } catch (IOException e) {
//                              Log.w("ChatClient", e.getLocalizedMessage());
//                      }
                        return null;
                }
        }
```



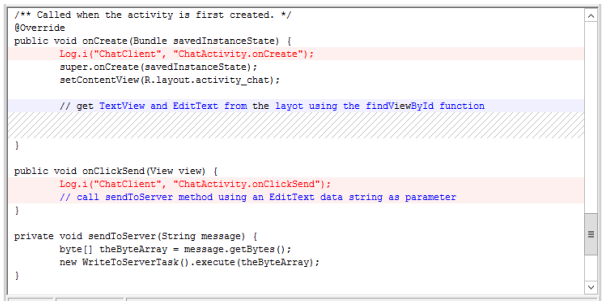
```
        private class CreateCommThreadTask extends AsyncTask<Void, Integer, Void> {
                @Override
                protected Void doInBackground(Void... params) {

                        try {
                                // ---create a socket---
                                serverAddress = InetAddress.getByName(SERVER_IP_ADDRESS);
                                socket = new Socket(serverAddress, 7003); //IP, PORT NUME
                                communicationThread = new CommunicationThread(socket);



                                communicationThread.start();
                                // ---sign in for the user; sends the nick name---
                                sendToServer(NICKNAME);
                        //
                } catch (UnknownHostException e) {
                                Log.d("ChatClient", e.getLocalizedMessage());
                        } catch (IOException e) {
                                Log.d("ChatClient", e.getLocalizedMessage());
                        }
                        return null;
                }
        }
```



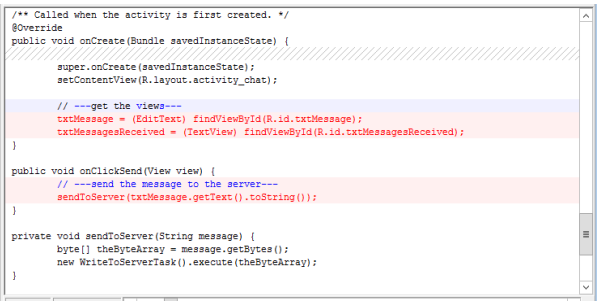
```
private class WriteToServerTask extends AsyncTask<byte[], Void, Void> {
        protected Void doInBackground(byte[]... data) {
                // call the write method of the communicationThread a data array (index = 0) as a par
                return null;
        }
}

private class CloseSocketTask extends AsyncTask<Void, Void, Void> {
        @Override
        protected Void doInBackground(Void... params) {
                // remove comments from the try-catch block

                try {
                        // call the cancel methos of communicationThread
                }
                catch (IOException e) {
                        Log.w("ChatClient", e.getLocalizedMessage());
                }
                return null;
        }
}
```



```
private class WriteToServerTask extends AsyncTask<byte[], Void, Void> {
        protected Void doInBackground(byte[]... data) {
                communicationThread.write(data[0]);
                return null;
        }
}

private class CloseSocketTask extends AsyncTask<Void, Void, Void> {
        @Override
        protected Void doInBackground(Void... params) {


                try {
                        socket.close();
                } catch (IOException e) {
                        Log.d("ChatClient", e.getLocalizedMessage());
                }
                return null;
        }
}
```



```
/** Called when the activity is first created. */
@Override
public void onCreate(Bundle savedInstanceState) {
        Log.i("ChatClient", "ChatActivity.onCreate");
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_chat);

        // get TextView and EditText from the layot using the findViewById function


}

public void onClickSend(View view) {
        Log.i("ChatClient", "ChatActivity.onClickSend");
        // call sendToServer method using an EditText data string as parameter
}

private void sendToServer(String message) {
        byte[] theByteArray = message.getBytes();
        new WriteToServerTask().execute(theByteArray);
}
```



```
/** Called when the activity is first created. */
@Override
public void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_chat);

        // ---get the views---
        txtMessage = (EditText) findViewById(R.id.txtMessage);
        txtMessagesReceived = (TextView) findViewById(R.id.txtMessagesReceived);
}

public void onClickSend(View view) {
        // ---send the message to the server---
        sendToServer(txtMessage.getText().toString());
}

private void sendToServer(String message) {
        byte[] theByteArray = message.getBytes();
        new WriteToServerTask().execute(theByteArray);
}
```