Some things to consider while coding an Android application
AKA
OpenComm's Coding Rules

By Rahul Arora

1. Logging output puts a damper on application performance

2. Proper commenting. Everybody has his/her own style but for pointers check out logincontroller.java at the end of this document.

3. Don't ignore exceptions

```
void setServerPort(String value) {
        try {
            serverPort = Integer.parseInt(value);
      } catch (NumberFormatException e) { }
}
```

4. Also don't catch generic exceptions.

```
try {
      someComplicatedIOFunction();          // may throw IOException
      someComplicatedParsingFunction();     // may throw ParsingException
      someComplicatedSecurityFunction();    // may throw SecurityException
      // phew, made it all the way
} catch (Exception e) {                     // I'll just catch all exceptions
      handleError();                        // with one generic handler!
}
```

5. Fully qualified imports

```
import foo.Bar;
```

and not import foo.*;

6. Write short methods. If it is going beyond 40lines, then you need to shorten it down.

7. Limit variable scope {int i=0}

8. Use standard brace style{
      }

9. Limit line length to make the code legible and fit in the screen (80 – 100 ch.)

10. Use TODO comments

/* [TODO] : I have to finish up this coding convention on time */

11. Limit line length but do not obfuscate code because you have to do that.

12. Use enhanced For loop syntax (also called as For-each)

13. Use Static Final for Constants
    Only applies to primitive types and String constants.

14.Try to avoid floating point where possible

15. Instead of hard-coding values in the code, use a XML to do so.

16. Learn how to debug

LoginController:

```
/**
 * Controller called from LoginView. Takes care of login functionality.
 * It connects to the server and returns true if the login is a success.
 * If Login is successful it calls DashboardView.
 *
 * Issues [TODO]
 * - No login incorrect message on screen
 * - ProgressDialog is visible even if login is incorrect
 * - For any other issues search for string "TODO"
 *
 * @author rahularora[hcisec], vinaymaloo[ui]
 * */

package edu.cornell.opencomm.controller;

import java.util.concurrent.locks.ReentrantLock;

import org.jivesoftware.smack.XMPPException;

import android.content.Intent;
import android.os.AsyncTask;
import android.util.Log;
import android.view.View;
import android.widget.EditText;
import edu.cornell.opencomm.Values;
import edu.cornell.opencomm.network.Network;
import edu.cornell.opencomm.network.NetworkService;
import edu.cornell.opencomm.view.DashboardView;
import edu.cornell.opencomm.view.LoginView;
import edu.cornell.opencomm.view.NotificationView;

public class LoginController {
	private LoginView loginView;

	// Debugging
	private static final boolean D = Values.D;

	// Logs
	private static final String LOG_TAG = "LoginController";

	// Check successful login
```

```
	private boolean islogin;

	private enum ReturnState{SUCEEDED, COULDNT_CONNECT, WRONG_PASSWORD,
ALREADY_CLICKED};

	// Username and password strings
	private String username;
	private String password;

	private ReentrantLock loginLock = new ReentrantLock();

	// Instance of XMPP connection
	public static NetworkService xmppService;

	public LoginController(LoginView loginView) {
		this.loginView = loginView;
	}

	public void handleLoginButtonClick(final EditText usernameEdit, final EditText
passwordEdit) {
			new LoginTask().execute(usernameEdit.getText().toString(),
passwordEdit.getText().toString());

	}

	private class LoginTask extends AsyncTask<String, Void, ReturnState> {


			@Override
			protected void onPreExecute() {
				loginView.getLoginOverlay().setVisibility(View.VISIBLE);
			}

			@Override
			protected ReturnState doInBackground(String... strings) {
				if(loginLock.isLocked()) return ReturnState.ALREADY_CLICKED;
				loginLock.lock();
				try{
					if (D) {
						Log.d(LOG_TAG, "Android app is attempting to connect to
the server");
						username = Network.DEBUG_USERNAME;
```

```
                        password = Network.DEBUG_PASSWORD;
                    }
                    else{
                        //username = strings[0];

                        String[] temp = strings[0].split("@");
                        try{
                            username = temp[0]+temp[1];
                        }
                        catch (ArrayIndexOutOfBoundsException e) {
                            username = temp[0];
                        }
                        password = strings[1];
                    }

                    if (D) Log.d(LOG_TAG, "Got Here1");

                    xmppService = new NetworkService(Network.DEFAULT_HOST,
Network.DEFAULT_PORT);
                    if (xmppService.isConnected()){
                        if (D) {
                            Log.d(LOG_TAG, xmppService.toString());
                            Log.d(LOG_TAG, "XMPP Connection established");

                        }
                    }
                    else {
                        return ReturnState.COULDNT_CONNECT;
                    }

                    /** Check whether the login is successful or not
                     * In case it is, start DashboardView using Intent else, [TODO]
              / @author: rahularora, vinaymaloo **/
                    if (D) Log.d(LOG_TAG, "Got Here2");
                    try {
                        islogin = xmppService.login(username, password);
                    } catch(XMPPException e) {
                        return ReturnState.WRONG_PASSWORD;
                    }

                    if (D) Log.d(LOG_TAG, "Got Here3");
                    if (islogin){
```

```
                        Intent i = new Intent(loginView, DashboardView.class);
                        i.putExtra(Network.KEY_USERNAME, username);
                        i.setAction(Network.ACTION_LOGIN);

                        loginView.startActivity(i);
                        loginView.finish();
                        return ReturnState.SUCEEDED;
                  }

                  else{
                        return ReturnState.WRONG_PASSWORD;
                  }
            }
            catch (IllegalStateException e) {
                  // TODO Auto-generated catch block
                  e.printStackTrace();
            }
            catch (NullPointerException e) {
                  // TODO Auto-generated catch block
                  e.printStackTrace();
            } catch(Exception e) {
                  e.printStackTrace();
            }
            finally {
                  loginLock.unlock();
            }
            if (D) Log.d(LOG_TAG, "Got Here5");
            return ReturnState.COULDNT_CONNECT;
      }

      @Override
      protected void onPostExecute(ReturnState state) {
            if(state == ReturnState.WRONG_PASSWORD) {
                  NotificationView nv=new NotificationView(loginView);
                  nv.launch("incorrect username or password","RED", "WHITE",
true);
                  Log.v(LOG_TAG, "Login failed for username "+username+"
failed");
                  loginView.getLoginOverlay().setVisibility(View.INVISIBLE);
            } else if(state == ReturnState.COULDNT_CONNECT) {
                  NotificationView nv=new NotificationView(loginView);
                  nv.launch("Could not connect to server","RED", "WHITE", true);
```

```
                        Log.v(LOG_TAG, "Could not connect to server.");
                        loginView.getLoginOverlay().setVisibility(View.INVISIBLE);
                    }
                }
            }
        }
```