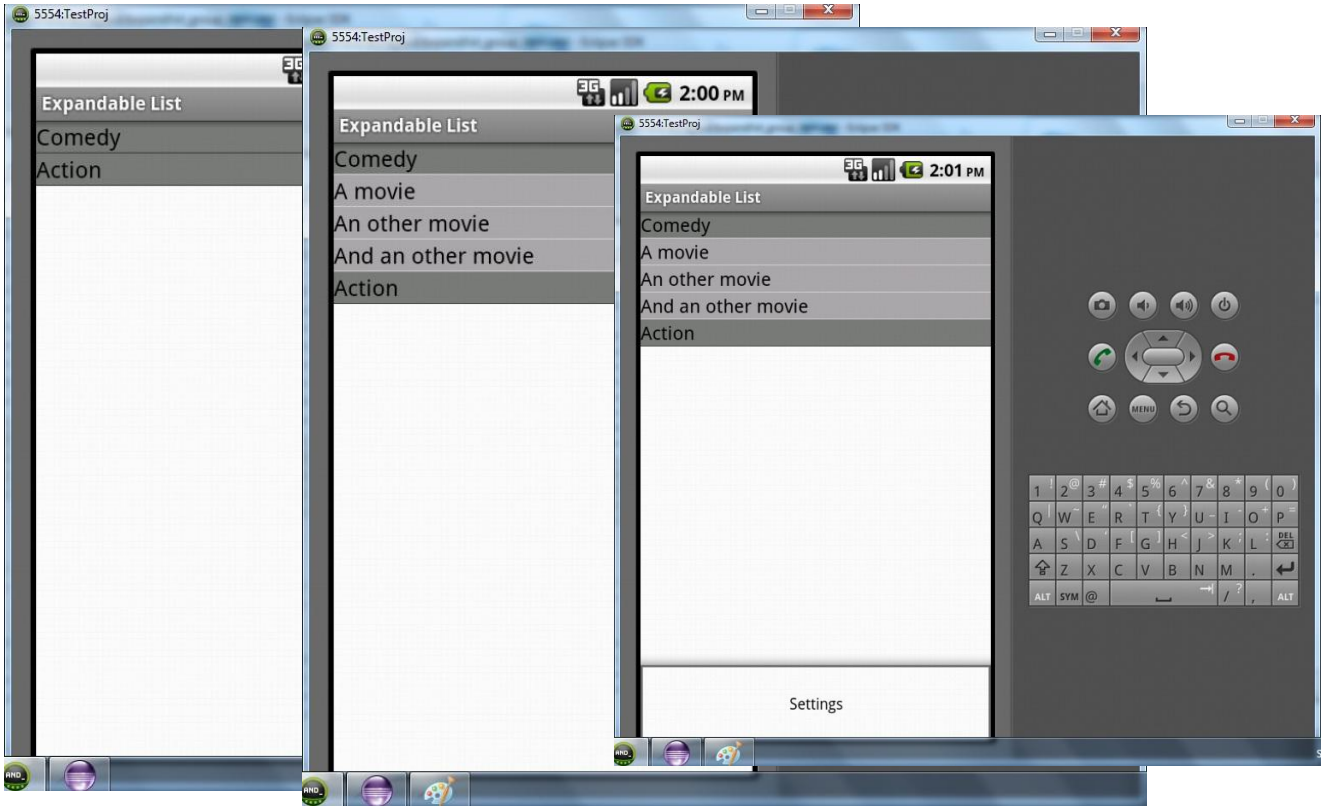
Appsolut Distribute

Application Client Architecture Suggestions

First of I've made a simple expandable list which will be used in the application after some modification to suit the MVC-package model which all members have agreed to use in the project.

How the example looks like:



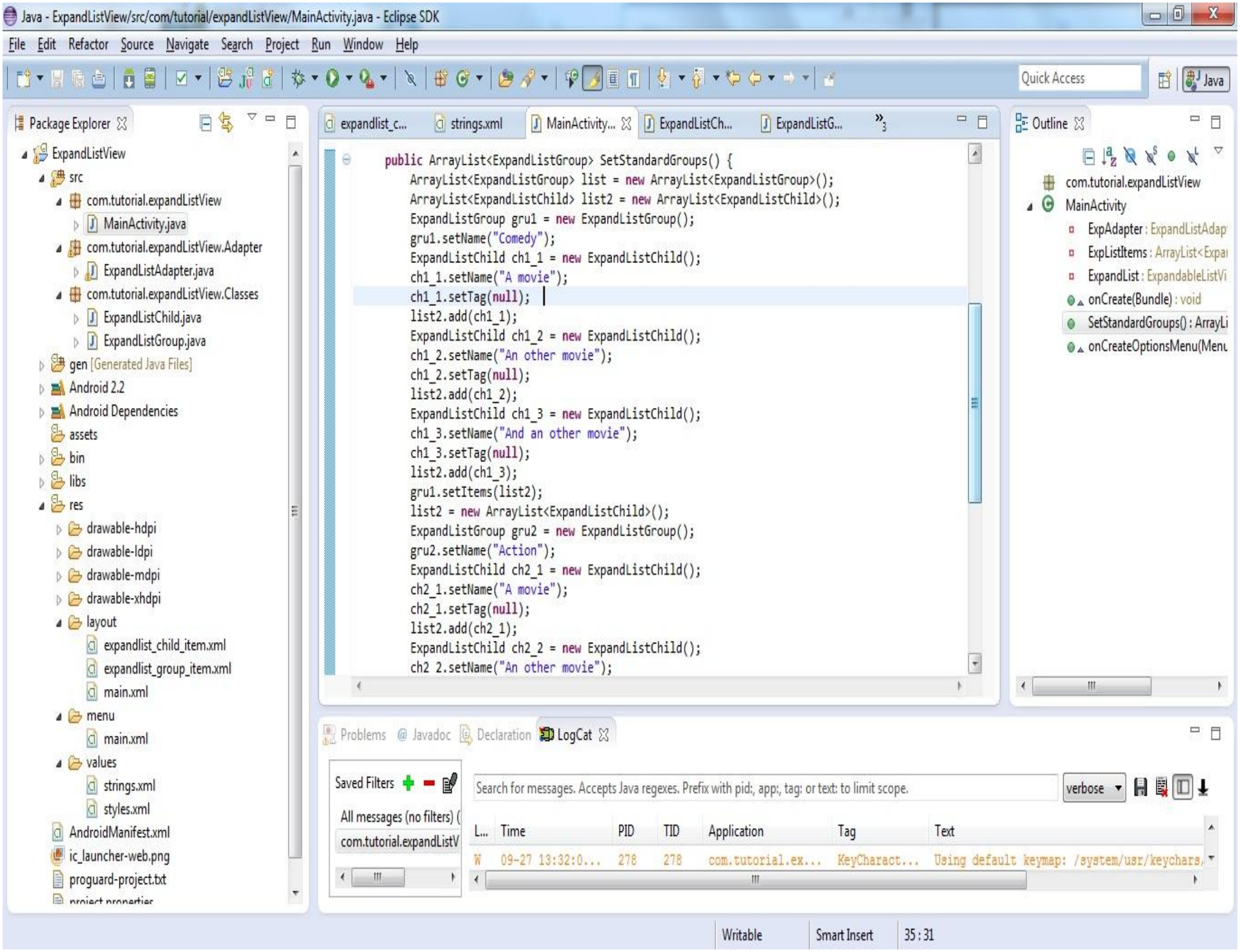
When pressing on a "group" choice from the groups list the list expands and gives room for the children list. It also has the default menu settings button. Nothing happens in this example if you press a child or the Settings.

Some visual improvements that I have been thinking about are changing the lists color (maybe even transparent color to see through to the background) and add a background image, maybe even make it possible for the user to alter the colors of the list and the background image.

The example structure (see next page for description):

There is no apparent architectural structure in the example, but there is a structure:

- View; is the xml files, is set to be content view of MainActivity.java. This makes the com.tutorial.expandListView the View package.
- Control; is the com.tutorial.expandListView.Adapter package where the ExpandListAdapter.java is located. It is here that one sets the action for "onClick" for the list.
- Model; is the com.tutorial. expandListView.Classes, this is where the structure of the Expandable List is defined.


Java - ExpandListView/src/com/tutorial/expandListView/MainActivity.java - Eclipse SDK
File Edit Refactor Source Navigate Search Project Run Window Help
Quick Access
Java
Package Explorer
ExpandListView
src
com.tutorial.expandListView
MainActivity.java
com.tutorial.expandListView.Adapter
ExpandListAdapter.java
com.tutorial.expandListView.Classes
ExpandListChild.java
ExpandListGroup.java
gen [Generated Java Files]
Android 2.2
Android Dependencies
assets
bin
libs
res
drawable-hdpi
drawable-ldpi
drawable-mdpi
drawable-xhdpi
layout
expandlist_child_item.xml
expandlist_group_item.xml
main.xml
menu
main.xml
values
strings.xml
styles.xml
AndroidManifest.xml
ic_launcher-web.png
proguard-project.txt
expandlist_c...
strings.xml
MainActivity...
ExpandListCh...
ExpandListG...
public ArrayList<ExpandListGroup> SetStandardGroups() {
ArrayList<ExpandListGroup> list = new ArrayList<ExpandListGroup>();
ArrayList<ExpandListChild> list2 = new ArrayList<ExpandListChild>();
ExpandListGroup gru1 = new ExpandListGroup();
gru1.setName("Comedy");
ExpandListChild ch1_1 = new ExpandListChild();
ch1_1.setName("A movie");
ch1_1.setTag(null);
list2.add(ch1_1);
ExpandListChild ch1_2 = new ExpandListChild();
ch1_2.setName("An other movie");
ch1_2.setTag(null);
list2.add(ch1_2);
ExpandListChild ch1_3 = new ExpandListChild();
ch1_3.setName("And an other movie");
ch1_3.setTag(null);
list2.add(ch1_3);
gru1.setItems(list2);
list2 = new ArrayList<ExpandListChild>();
ExpandListGroup gru2 = new ExpandListGroup();
gru2.setName("Action");
ExpandListChild ch2_1 = new ExpandListChild();
ch2_1.setName("A movie");
ch2_1.setTag(null);
list2.add(ch2_1);
ExpandListChild ch2_2 = new ExpandListChild();
ch2 2.setName("An other movie");
Outline
com.tutorial.expandListView
MainActivity
ExpAdapter : ExpandListAdap
ExpListItems : ArrayList<Expa
ExpandList : ExpandableListVi
onCreate(Bundle) : void
SetStandardGroups() : ArrayLi
onCreateOptionsMenu(Menu
Problems
Javadoc
Declaration
LogCat
Saved Filters
All messages (no filters) (
com.tutorial.expandListV
Search for messages. Accepts Java regexes. Prefix with pid:, app:, tag: or text: to limit scope.
verbose
L... Time PID TID Application Tag Text
W 09-27 13:32:0... 278 278 com.tutorial.ex... KeyCharact... Using default keymap: /system/usr/keychars/
Writable
Smart Insert
35 : 31

Weaknesses in the current structure:

The Control package and Model package resides in the View package.
The input of the list objects is inputted in the MainActivity class (se the picture above) it should be done in a different package (probably the database package). We want this to be done in a separate class, reading from a database what to put in the list and to make it dynamical. Meaning it can be reused to make (read in) a new menu to the expandable list and making it possible to reuse the same activity.

The plan to modify the structure:

Current example structure:

-src
  -view
    MainActivity.java
    -controller
      ExpandListAdapter.java
    -model
      ExpandListChild.java
      ExpandListGroup.java

Other stuff …

-res
  +drawable mappar….
  -Layout
    expandlist_child.xml
    expandlist_group.xml
    main.xml
  -menu
    main.xml
  -Values
    strings.xml
    styles.xml

Other stuff …

Some new classes and xml files may appear in the modified version.

The Modified structure package and class structure:

-src
- -view(should these be empty and only contain other packages?)
  - MainActivity.java
  - Fragment.java
  - possible some other activity and fragments classes…
- -controller(should these be empty and only contain other packages?)
  - ExpandListAdapter.java
  - Other user input classes…
- -model(should these be empty and only contain other packages?)
  - ExpandListChild.java
  - ExpandListGroup.java
  - Client.java
  - other model classes…
- -database
  - ExpandListInput.java (read only, could end up in model.pkg)
  - Read.java
  - write.java
  - ClientIO.java

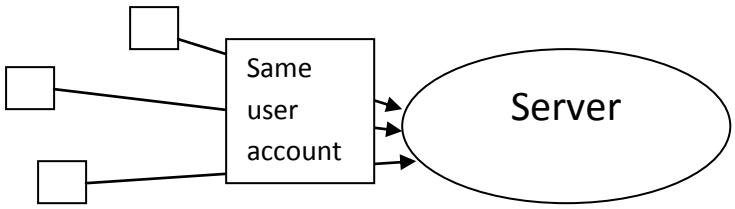
Other stuff …

-res
- +drawable mappar….
- -Layout
  - expandlist_child.xml
  - expandlist_group.xml
  - main.xml
  - other xml files…(for fragments etc.)
- -menu
  - main.xml
- -Values
  - strings.xml
  - styles.xml

Other stuff …

Note: all classes in the packages view, controller, model and database are probably put in “sub packages” inside of the four mentioned above. We as a group must decide on a package name and

The suggestion of having an internal group for the user account:



Theory:
An account holder on a server can connect from several devices (where the software is useable, ex Smartphone, tablets etc) and exchange information between the units.

How:
When a user has created an account an internal (or a private, I have not decided what to call it) "group" is created. The only user that can use that "group" (I call it group because it may work similar to the groups that are described in user story, difference **no** group admin or moderator) is the account holder. This group will not be shown to other account and should be easy to access.

Note:

The database structure for the individual/private group will be different, mostly because of the group hierarchy (admin, moderator) but also the connectivity.

If an account holder is able to connect to a server from several devices, it could mean that one can do everything an accountholder can do normally from several devices (for instance accessing the groups lobby etc). If this becomes a problem or security issue one can limit the account access rights while several devices are connected with the same account.

Pros/cons for developer:

+Could be created before we make the structure of the other groups and add the missing elements (group admin, moderator etc).

+If we develop the other group's structure first, we can just make it a modified group that is created with the user account.

- Will take some time (unknown how much). It isn't in the user story. Unknown problems may happen, but on the other hand that may occur for lots of things in our project.

Pros/cons for users:

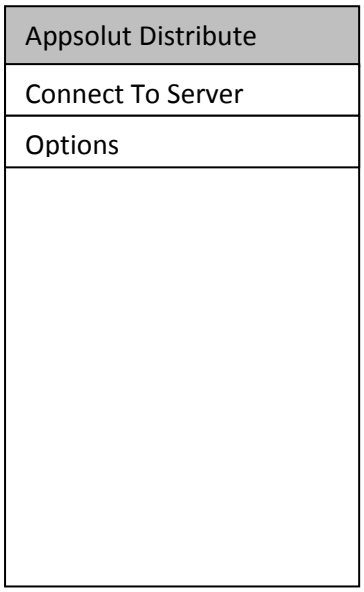
+Can share different kind of information between devices with just one account.
+Can create a network with just one account (ex. one account for a project group if they don't like administrators and moderators etc).

-This function can only be used if the user has two units available. + (Can login on a friend's device)

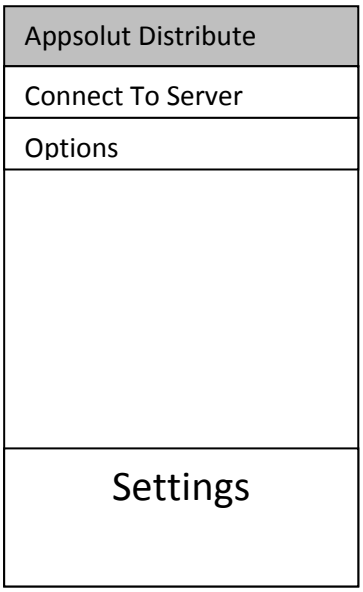
GUI suggestion:

These are sketches to roughly describe the wanted layout for the GUI and how the users will navigate in the application. These sketches only show the options available in the star screen, what happens when you select the different options and how it will look like when the user has logged in.
Note: I think that the settings should be set by using the menu button on android. That way it can be accessed when needed. Settings will at most in our application give a fragment to set background picture and (if there is any time) set color theme.

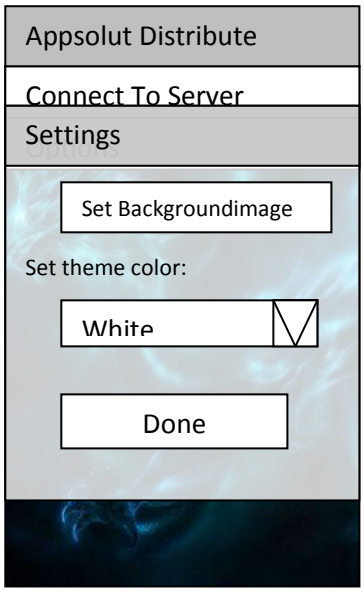
Start screen



Pressed the menu button



Pressed “Settings”



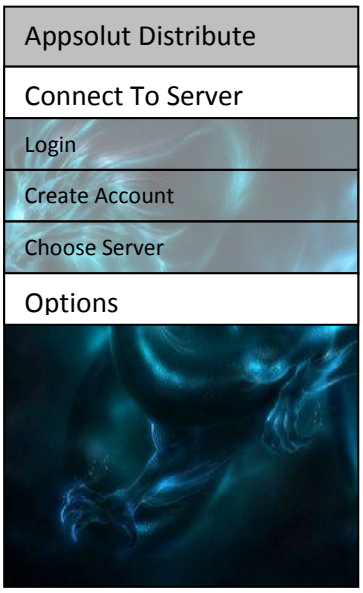
When “Settings” was pressed a fragment appeared, in this fragment the user will be able to change the looks of the environment.

Note:
I made the fragment background transparent to show where the fragment is. I only do this the first time. Though I have the thought of making it completely transparent and only see the choices, but could be too confusing.
I changed the Background image.
The fragment in itself might be reusable.
I use a “Spinner “(a class in java that I’m going to try out) in the fragment.
In the Fragment (and maybe in the Activity) I may need to set in the xml-file android:isScrollContainer="true". This could make the Activity or Fragment scrollable.
I should also research to save state when back is pressed during some situations (such as going back from the start screen), doesn’t appear to be a problem if home is pressed.
The Fragment should have a scrollable View to put the content in and a “Done” button.

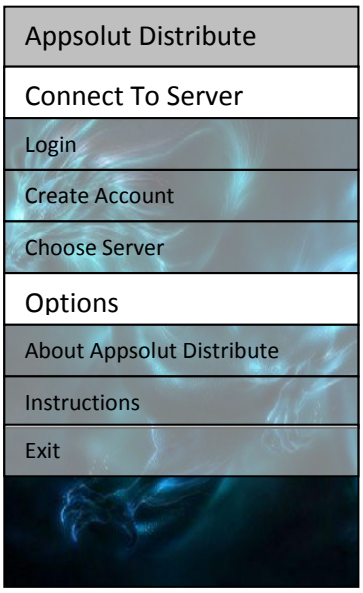
When the user is finished with the changes and pressed done the user will get back to the screen “main screen”, se next page. My suggestion here is to make the “children” in the expandable list transparent.

Back to start screen

Pressed “Connect To Server”

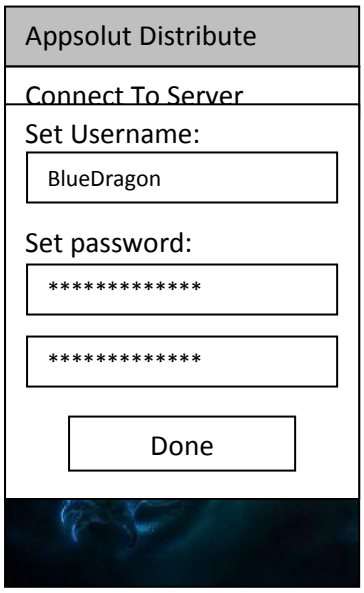


Pressed “Options”

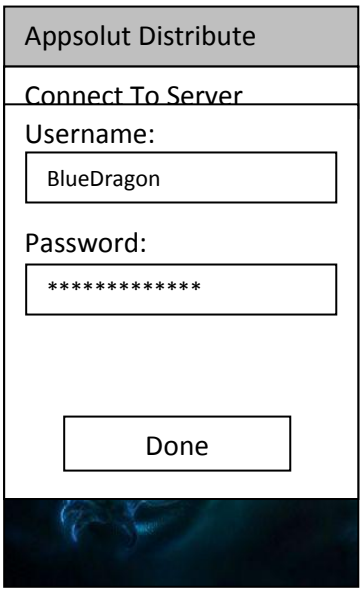


The server has a default value for now.

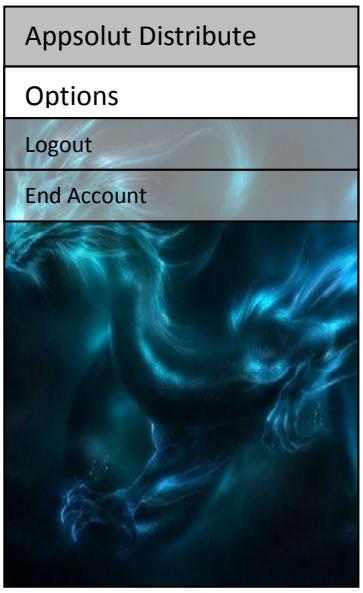
Pressed “Create Account”



Pressed “Login”



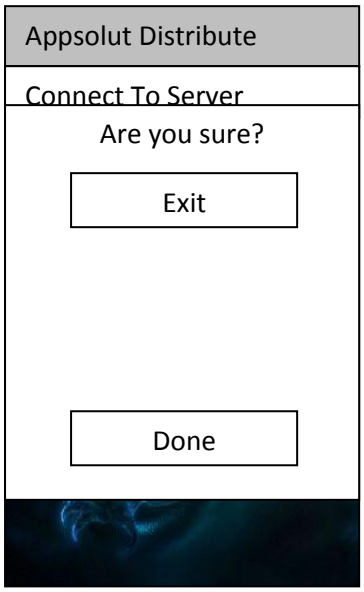
Account view, not yet decided



When an account is created should we have a server password (optional)?

Note:
Might make some sort of frame around the content of the Fragment(not the done button).

Pressed “Exit”



Other notes:

We should test our application on a tablet.

Should the application be online in the background, e.g. Skype? If so check how to get an icon in the “ongoing” field.