The first thing I did was to import all the buttons and text views so they they could be used in java, I did this using findViewById() and referencing the android:id set In the xml. Then I desided to set the buttons up. To do this I originaly thort the best way would be to use OnClicklistener’s but the problem with this is that with the directional buttons I need to know if they are presed on not, not when they are presed. So after some resurch I came across this: <http://stackoverflow.com/questions/13409271/android-how-to-know-if-button-is-held-down>. I used the method stated in this (OnTouchlistener) for the direction buttons. I also desided to again use object oriented programing. So I have put the on touch listeners in another class that implements OnTouchListener. This means that I can have one method for all the direction buttons, simplifying the code, this is the UpdateMovment class, in a method called OnTouch() this sets up the liseners and the onTouch method will handle when one is touched. I have used a switch and case to tell whitch button is presed, I have one for eeach actions(up and down) with the switch v.getid and the the case is the ID used in XML.

I have also moved all the findviewbyid()’s to another method in the MainActivity class so the OnCreate method is less clutered.

I am also using Public int’s in the MainActivity for each data to be sent to the server. This way my plan is that the client will send data a set amount of times a second, and it will construct the array to be sent from those 3 int’s, this means they can be updated any time and the current versions of them will be sent.

I ran in to an error when I tried to use “this”(in java refers to the current instance of a activity) when making an OnTouchLisener . in a class that the view(button) did not belong to. So I have moved the On Touch Liseners to the MainActivity and made that implement OnTouchListener and created a method that takes a view and motion event so that it straght away and returs a boolian that the onTouch will return. I will use regual OnClick’s for the 2 other buttons.

30-7-14: I started work on the networking side of the app, I made a new class called Connections, and included simmiler code to what is on the server but with the basic layout of [this](http://www.java2s.com/Tutorial/Java/0320__Network/EchoClient.htm) and then to test all it dose is connects, sends a string and disconnects. But opon running this the app crashes after some resurch it seems this is because I was connecting to a network on the main UI thread and as of API level 11 this will cause the app to crash, this makes sence as if the server dose not respond this could casue the app to not respond. So instead I need to do it in a Async runner, this is a prosses that runs in the background, seperiout from the main thread, so I looked up example Async runner projects, one that I found very use full is in the example folder under Async. So I moved the networking over to a AsyncRunner class, but still keeping the Connections class and referencing this.

31-7-14: With the Async class at a point where I could test its connection to the server I need a way to show data from the Async class, so I set the textviews in MainActivity to public so they could be acsesed from the AsyncRunner class. But this would not work because the DoInBackground task can not modify a view. So after some resurch I realised that that is what the publishProgress is for, so made it so that the first parameter was the text to set a textview to (0 or 1 where 0 is the top one) and the second is the text.  
When attempting to run the app I first ran in to a problem because I had not declared the INTERNET premition in the manifest, and secondly because I had set up the socket outside of the Async runnerclass.

11-8-14: When attempting to send the array over a Object stream writer, it would not work when the array was made in another class, eventhow I could print to the TextView the value of it. So I made a new array that would be send by using "int[] Arrayout1 = {MainActivity.command, MainActivity.gofront, MainActivity.goside};"

I also did as lot of work on Error catching today, since this uses the internet the are a lot of things that can go wrong, to try to stop this I have used a Try and catch to stop the app crashing, and everywhere where I found nessasery I have made it so it stops looping (sending the array), Disconnects and sets a text view as the error.  
I also ran in to an error where it would not let me turn(to send the turning values) after looking in the UpdateMovment class this was because I had one switch and case for both turning and going forward, this means that it would only go to one case, the one that was first. So I split in to 2 seperiout switch and case statement one for front and back one for sideways.

App Milestone 1 11-8-14: the app is now at a point I would call mile stone 1, it is mostly functional, and more importantly the networking works and catches errors.  
But TODO I have the connect button changes text disconnects you once pressed. I have to get it so it takes the response the server sent, as seen in the protocol. And the art for the app so it looks good. Also im its current state it out puts position 1 and 2 to the 2 text view’s rather than what should be there.

16-8-14: I have added a global boolean to MainActivity class called running, it is set to true if the aSyncTask is running and false if it is not. And when the connect button is pressed depending on the state of running it either starts the Async task or sets command to 0 so it will disconnect.

17-8-14: The last thing left for the app to be feature complete is to add a way to set the IP, I already have a button to do this but it currently dose nothing, as said in the planning document. I have used [this](http://www.javacodegeeks.com/2011/01/android-quick-preferences-tutorial.html) tutorial as the basis for the implementation. I was originaly going to use [this](https://www.youtube.com/watch?v=zJ9qzvOOjAM) but this is depreciated as of API level 11. I also had to change my minSDKver in the android manifest to 11 to use this method. I have made it so that when the preference button is pressed it setts command to 0 so it will disconnect(if connected) and waits 100ms so it has time to disconnect, then goes to the ShowPrefrence class.  
I have made it so that in MainActivity just before it start the AsyncRunner it creates a new SharedPreferences and gets the stored ip and passes it in to the AsyncRunner. I have then replaced the line “String hostname = “192.168.1.105””(ip of my PC) with String hostname = params[0].   
I am also yet to lock the orientation of the main activity to landscape, so I look up how to do it and find [this](http://stackoverflow.com/questions/4675750/lock-screen-orientation-android) so in the android manifest I use that.

26-8-14: I now need to rewrite the Asyncrunner class to use UDP as explained in the RCserver proration of code document, to do this is used [this tutorial](http://systembash.com/content/a-simple-java-udp-server-and-udp-client/) and the [java doc](http://docs.oracle.com/javase/7/docs/api/java/net/DatagramSocket.html). After running this and the server, nothing happens, nothing is received on the server. After using [Wire Shark](https://www.wireshark.org/) to monitor the network I belive it is a problem with the app not sending the packets. To test this I made a small class that send a UDP packet to the server, and that was recived by the server( the test class is under code and UDP\_test.java). So I deside to rater than edit the Async runner class, so completely rewirte it as it is very messy.

29-8-14: I changed the Lables for the left and right buttons so they are the correct way around and changed the UpdateMovment class so they turned the correct way.

6-9-14: Currently The app will set commands with out checking there is a server there So now what I will do is every 1S from the client I will send a packet containing an int “6” when it receives this it will reply with an int “1” if it dose not do this with in 1s it will time out. To show an error message for this is will be doing “return 0;” this will end the async runner and pass 0 to the onPostExecute sub where it will display an appropriate error. To do the time out I am using .setSoTimeout(int); when I first added the “catch (SocketTimeoutException e) {“ I got an error because above it I had “catch (IOException e) {“ and later will catch the time out exception as well as others. I have made it so that if there is a time out exception then it will not continue looping(keepgoing = false) and it will also do one when the program first runs(test if there is a server). I have run in to an error where sometimes there will be a time out error even if the server exists and receives the number “6” so I have go it so that only if it fails 4 times it will quit this is because packets are not guarantied to be delivered using UDP. I will also be using a dedicated textview for the state and one for errors, I have also added another for instructions.

Now that the app is feature complete I need to make it more “professional”. So to do this I will need to do some of the following:

App icon

Change the Back Ground

Add proper instructions.

Make it so that the text changes in the connect buttons

Add a splash screen

Possibly add another screen just for instructions to show up the first time the app is run.

The first thing I will do is to set the text of the connect button appropriately. I will approach this and adding proper instructions at once, So I Add to the onPostExecute so it changes the text button and resets the Instructions I added a settext() sub in the Main\_activity that sets the text back to defult. And is run when the app first runs and when appropriate. I have also changed it so that it runs the test to see if the server is there as soon as it start to communicate with the server and it changes the instruction text.

9-7-14: I will start by making the graphics for the splash screen. For this I will be using [GIMP](http://www.gimp.org/). I have desided to use Roboto font found on the [developer.android](http://developer.android.com/design/style/typography.html) website. After testing all of the Roboto fonts I will use the Roboto-light.ttf as this look the best and not block on a small screen. All the raw .xfc files are in the Assets/RAW and the exported .png’s are in the assets folder. I have made the splash. I have also removed the title bar on the MainActivity as I think it look beater.

I made a new activity and lay out and used android:background="@drawable/splashscreen" in the xml to set the background and in the java I used [this](http://stackoverflow.com/a/9926866) and [this](http://stackoverflow.com/a/2591311) to hide the nav bar and status bar for this one activity. It then waits 5s then lantches Main\_activity. Next I made a app icon, I wanted this to be simple but recognisable. Due to time constraints I have only been able to make the assets splash screen and logo) be optimised for XHDIP specifically for 1080P if I had more time I would re make then for smaller resolutions.

12-9-14: I spent a lot of time commenting code. This is a very important practice because this allows other people to view your code and understand it very easily. This is very good practice for working on group coding projects in the furture as I will have to do at University. This also makes it easier to debug my code as I can quickly see what each part is meant to do.

//make different size once for diffrend DPI