Documentation for the Nano-Mouse Application

By: Jason Nixon

1. Application Structure:

This part of the documentation will explain the layout of each method used to create the application for the Nano-Mouse:

- **ButtonStart.Click()**: This is the only method for the first screen of the application that control the start button used to enter into the second screen.
- **ConnectList.BeforePicking():** This method controls what happens before the device is connected to the mouse. Before its connected there will be a rectangular button that on the left side of the screen that we have the word “Connection” on it will a red background, this indicates that the android device has not paired with a Bluetooth module.
- **ConnectList.AfterPicking():** This method function is to display a listview of all the paired devices and when one is picked then the user will have control of the mouse. After the user has picked a paired device the “Connection” button’s background will change from red to green to indicate that a connection has been established.
- **ConnectButton.Click():** This method controls the connection between the devices. When there is not a connection already established then the user will be able to connect or if there is a connection already established then it disconnects.
- **Clocks.Timer():** This method uses four global variables that contain values that are sent over in bytes to the Nano-Mouse and this values are displayed on the android device screen with values ranging from 100 to -100.

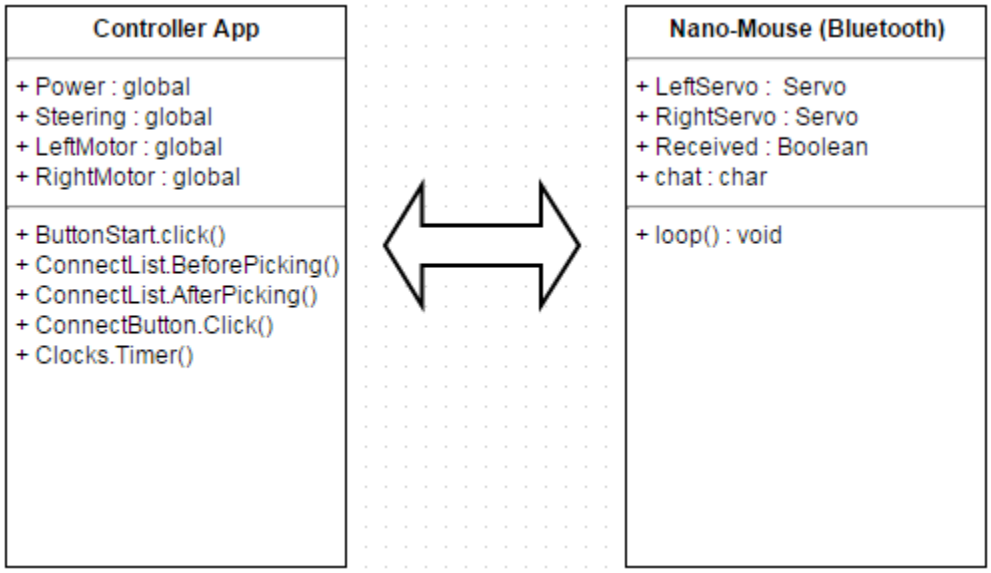
2. Arduino Structure:

This part will explain how and what is been sent over from the android device to the Nano-Mouse:

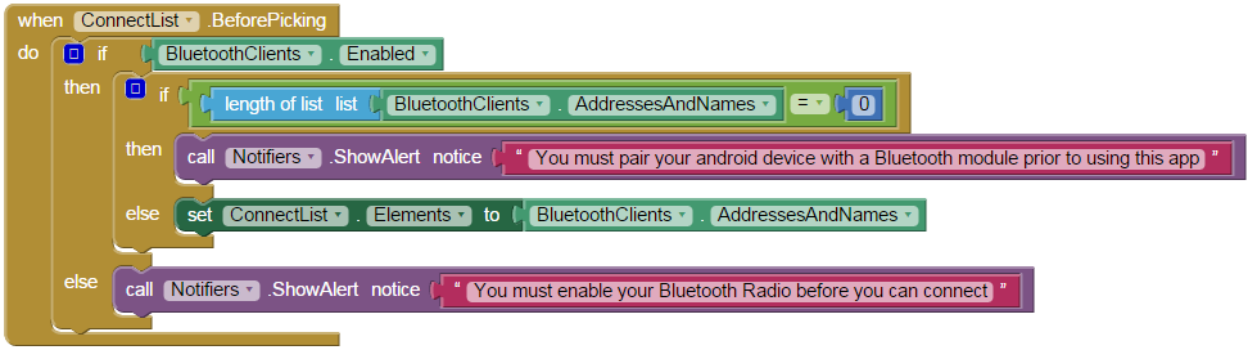
- **Messages_ino.ino:** This code is loaded on to the Arduino Nano, but when uploading the code it is necessary for the user to take out the RX/TX cable from the board or the code won’t load. The way the code works is that it detects if there has been a serial connection between both devices, when there is the values that are being displayed on the android device are actual the values being sent over to the Nano-Mouse in bytes in a Char array and the values determine the amount of power going into each servo motors.

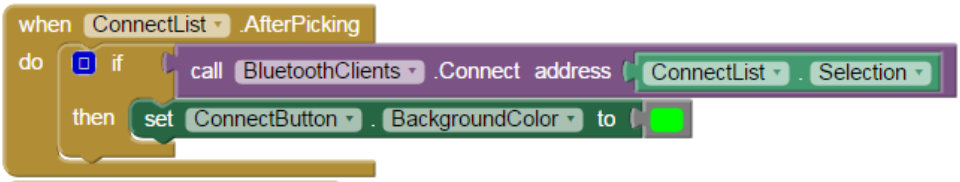
3. UML Diagram:

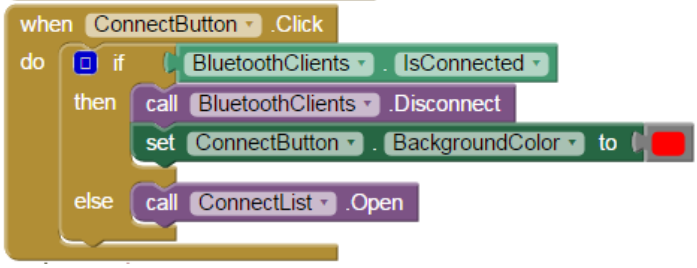
This UML Diagram to show the relationship between the application and the Nano-Mouse:

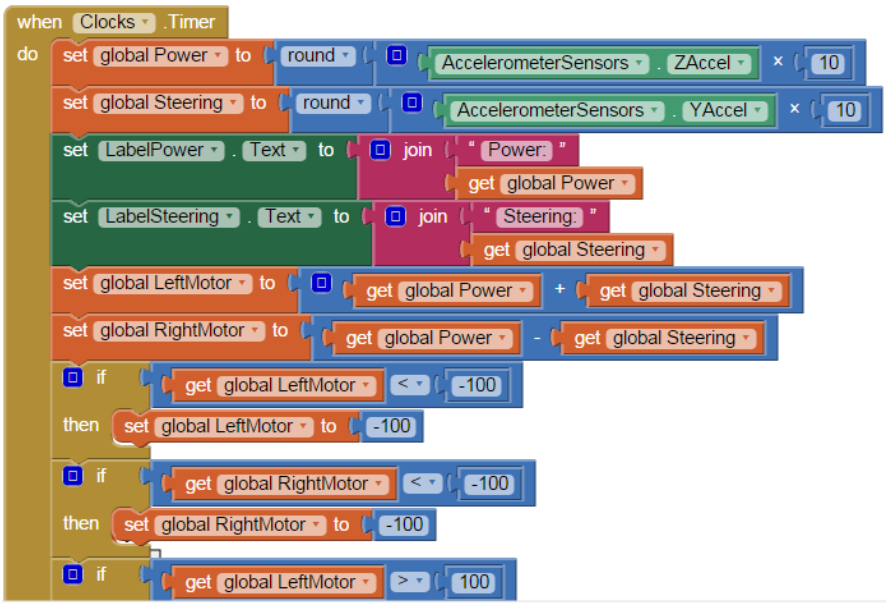


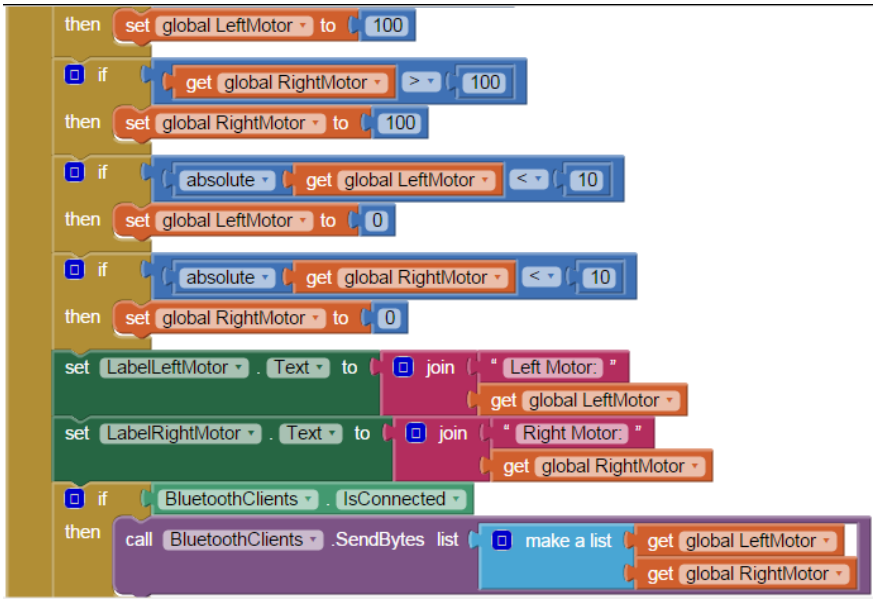
4. Application Source Code:








when Clocks .Timer
do set global Power to round AccelerometerSensors . ZAccel × 10
set global Steering to round AccelerometerSensors . YAccel × 10
set LabelPower . Text to join " Power: "
get global Power
set LabelSteering . Text to join " Steering: "
get global Steering
set global LeftMotor to get global Power + get global Steering
set global RightMotor to get global Power - get global Steering
if get global LeftMotor < -100
then set global LeftMotor to -100
if get global RightMotor < -100
then set global RightMotor to -100
if get global LeftMotor > 100


then set global LeftMotor to 100
if get global RightMotor > 100
then set global RightMotor to 100
if absolute get global LeftMotor < 10
then set global LeftMotor to 0
if absolute get global RightMotor < 10
then set global RightMotor to 0
set LabelLeftMotor . Text to join " Left Motor: "
get global LeftMotor
set LabelRightMotor . Text to join " Right Motor: "
get global RightMotor
if BluetoothClients . IsConnected
then call BluetoothClients .SendBytes list make a list get global LeftMotor
get global RightMotor