Worksheet 3 – Putting everything together

This lesson assumes a basic familiarity with programming in SCRATCH
This lesson assumes the Raspberry Pi has been configured with GPIO and Pibrella add-ons.

Learning objectives

- You will be able to apply the skills learned in the first 2 lessons to build a simple game using the Raspberry Pi

You will need

- Raspberry Pi
- Pibrella Interface board
- Mouse, Keyboard, Monitor
- 2 light sensors

Logging into your Raspberry Pi

Once your Pi has booted up, log in with the username "pi" and the password "raspberry"

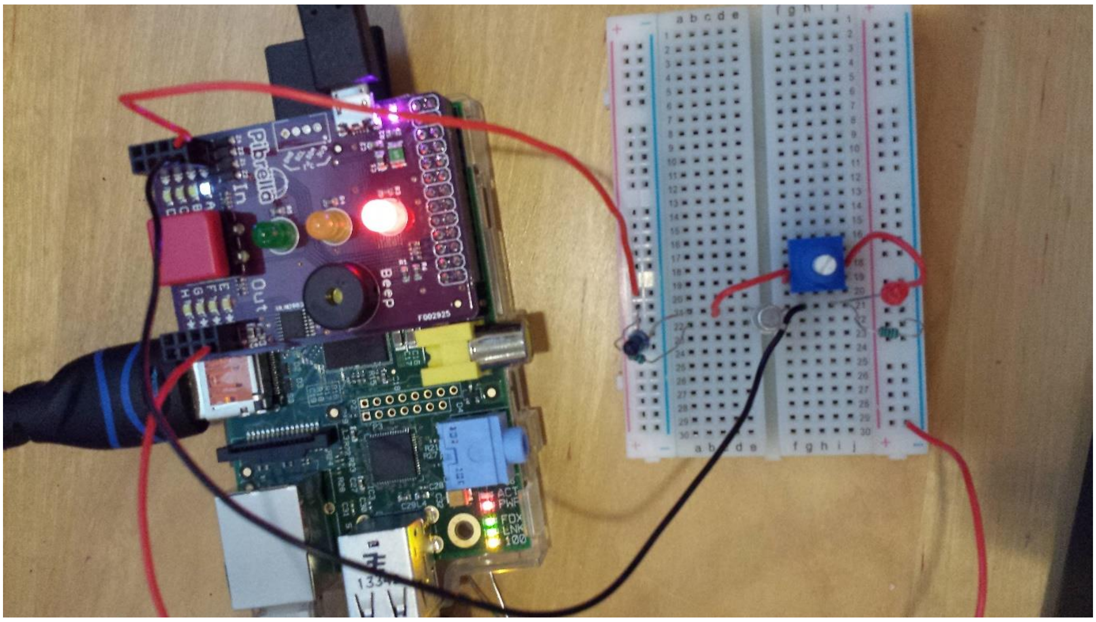
Type the command "startx" to open the desktop

Once the desktop has opened, double click on "SCRATCH GPIO4" to start writing your computer program.

Connecting your sensors to the Raspberry Pi

Plug the light sensor you made in lesson 1 into your Rasperry Pi. The light sensor requires 3 wires to be connect. If you have a second sensor, connect this to input B and output F.

- Input A +ve – connect to top rail of breadboard
- Input A –ve – connect to emitter of transistor
- Output E –ve – connect to bottom rail of breadboard



Calibrate your light sensor by adjusting the potentiometer so that when the Phototransitor is covered. The value for InputA should switch from 0 to 1 and back when covered.

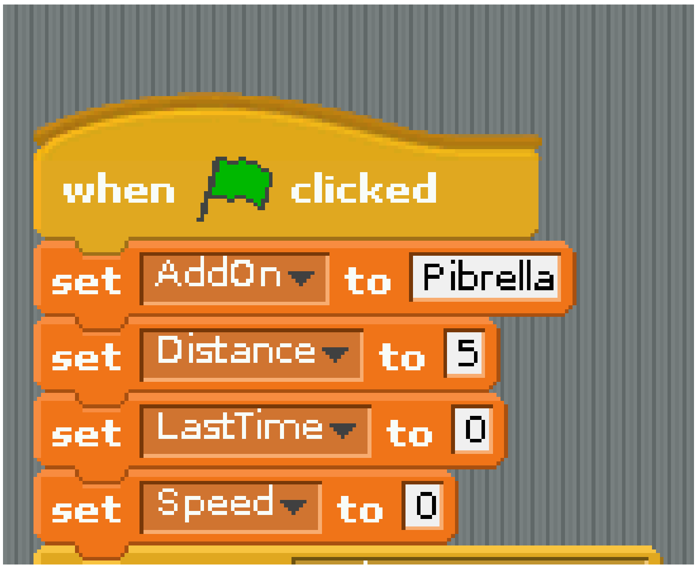
Experiment 1 – Using the light sensors to calculate the speed of a moving object

To calculate an object's speed, you need to know how far it travels in a given time.

Name a unit that can speed be speed measured in.

The raspberry pi can be used to measure the time taken for an object to pass between two light gates. We can measure how far apart the light gates are, and use this information to calculate the speed.

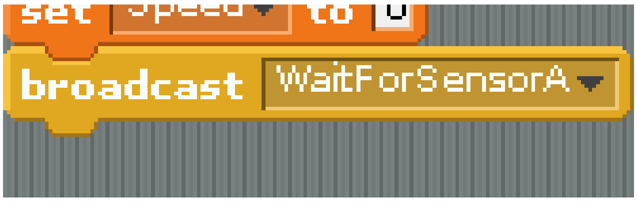
Create 3 variables in SCRATCH for Speed, Distance and Time.



```
when ⚑ clicked
set AddOn to Pibrella
set Distance to 5
set LastTime to 0
set Speed to 0
```

Set the values to those shown above. Later, we'll set the distance by measuring the distance between our 2 light gates with a ruler.

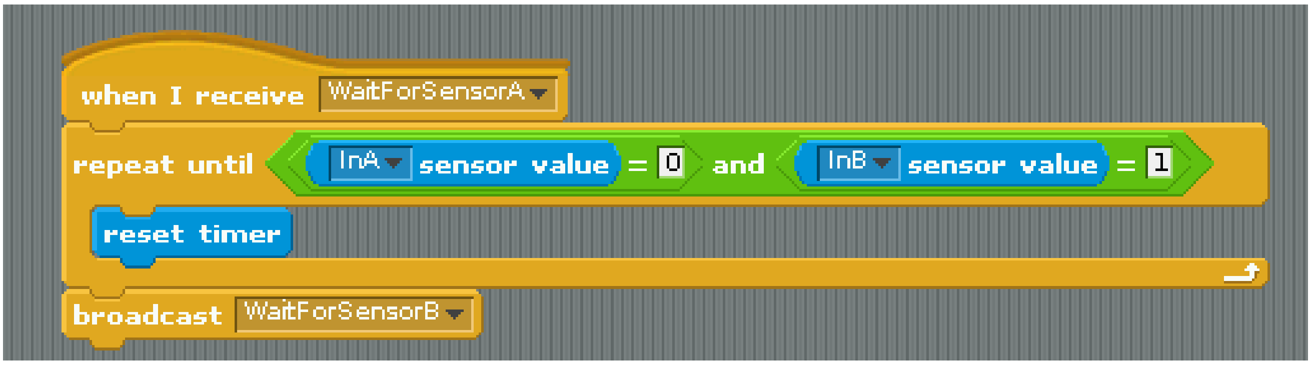
Once these variables have been created. Broadcast the message "WaitForSensorA". This is used to call another section of code that is listening for this message.



```
broadcast WaitForSensorA
```

What do you think will happen when you run your program?

Now add the following code under the header “WaitForSensorA”



```
when I receive WaitForSensorA
repeat until < < InA sensor value = 0 > and < InB sensor value = 1 > >
    reset timer
broadcast WaitForSensorB
```

Run your program. What happens to the timer?

Cover the light gate connected to Input A. What happens now?

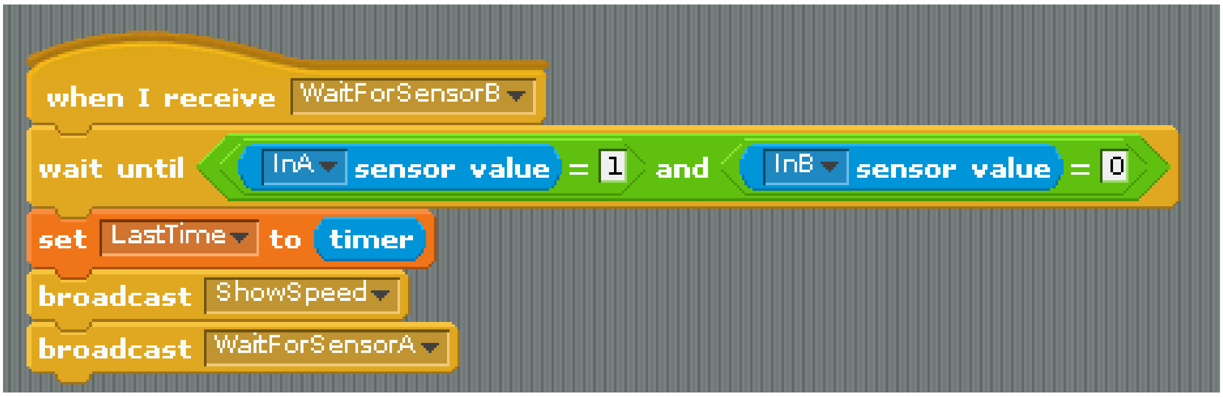
Stop the program. When neither light gate is covered, the timer should reset to 0. When light gate A is activated, the timer should start counting up.

If this does not happen, you will need to calibrate your light gates and make sure they’re connected properly.

Now we can add code which waits for input B.

Create a block of code that responds starts running when it receives “WaitForSensorB”

Add the following code.



```
when I receive WaitForSensorB
wait until InA sensor value = 1 and InB sensor value = 0
set LastTime to timer
broadcast ShowSpeed
broadcast WaitForSensorA
```

Describe in your own words what happens when this code is run.

The broadcast “ShowSpeed” method is used to calculate the speed using the distance and time.

What is the formula to calculate speed given a distance and a time?

Add the code which uses the distance and time to calculate the speed of an object. Write it below.

Finally, measure the distance between your 2 light gates and set this value when the applications starts.