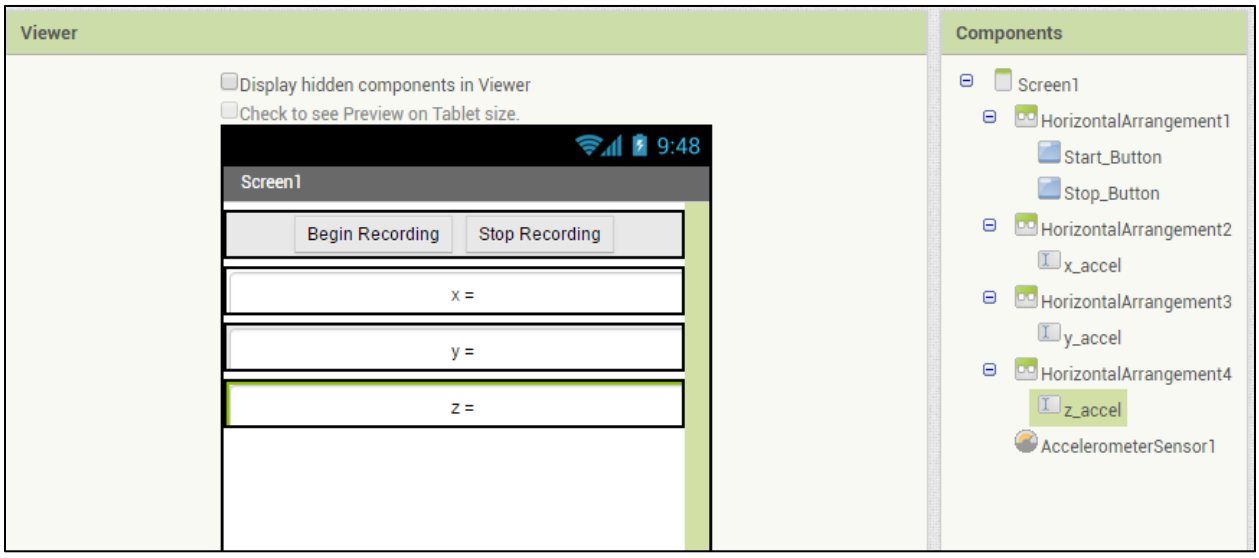
Tutorial 2 - Solutions

Accelerometer

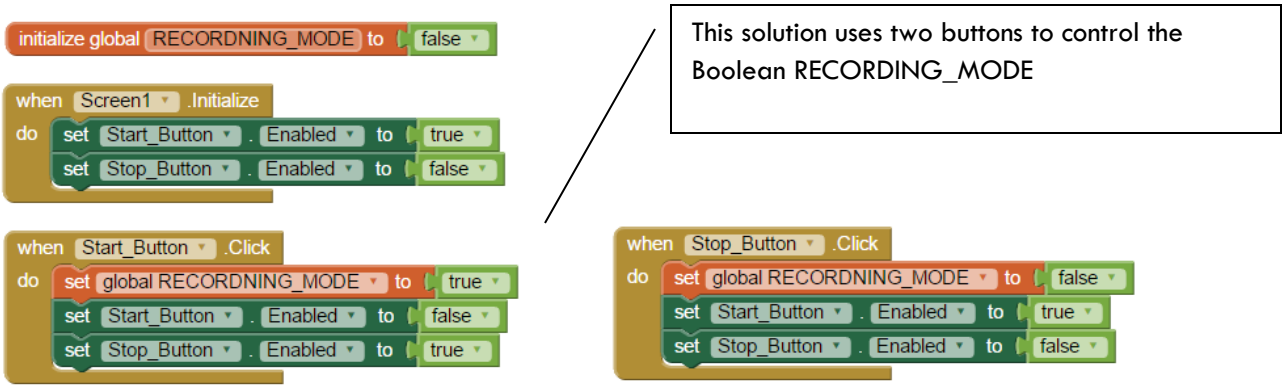
Have a go yourself!

Build an application which displays pure recordings from the x, y and z axis dimensions. The application should have a controller which will allow the recordings to begin and end.

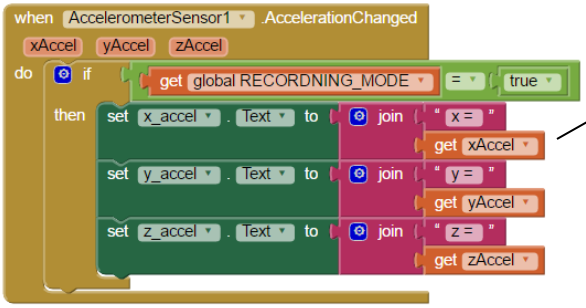
Designer



Blocks



This solution uses two buttons to control the Boolean RECORDING_MODE



We are then using a conditional block to define if the text fields should be updated or not.

Barcode Scanner

Have a go yourself!

In this exercises, we would like to introduce using conditional statements to provide feedback to a user based on expected inputs/outputs

Create a QR code image, and keep a record of the text which is embedded.

Set this text as a global variable in your App.

Use the barcode scanner to scan the image and check if the QR code text you are expecting match

If so, display a message informing the user of the successful identification.

If not, display a message informing the user of the unsuccessful identification.

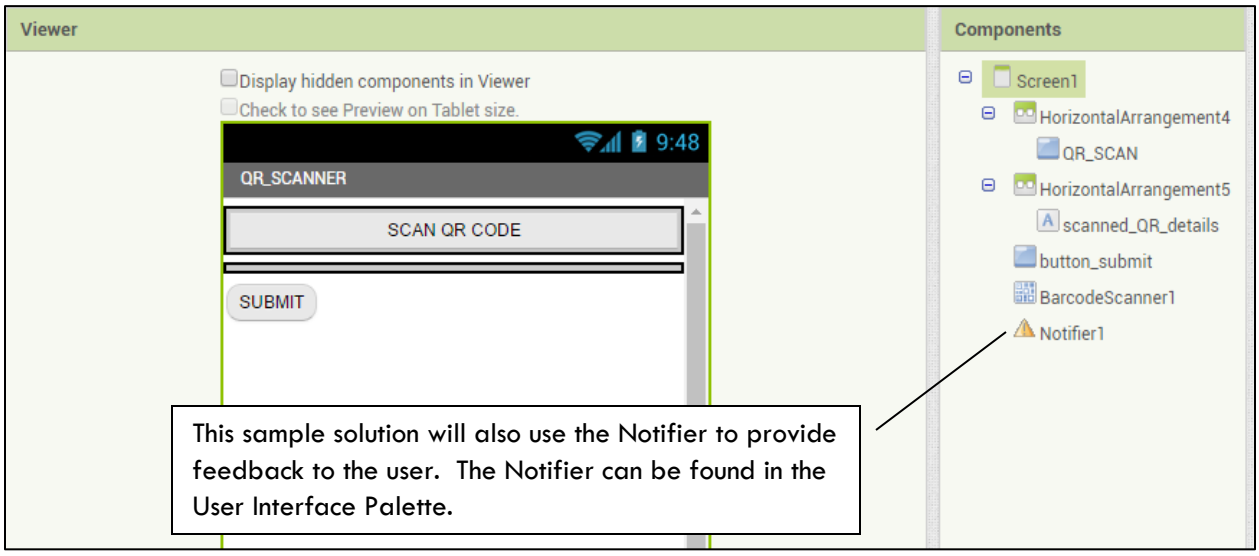
Hint: for proper testing of this application, you should test how the application behaves in both instances. You will need to generate another QR image which contains unexpected text.

QR Images:

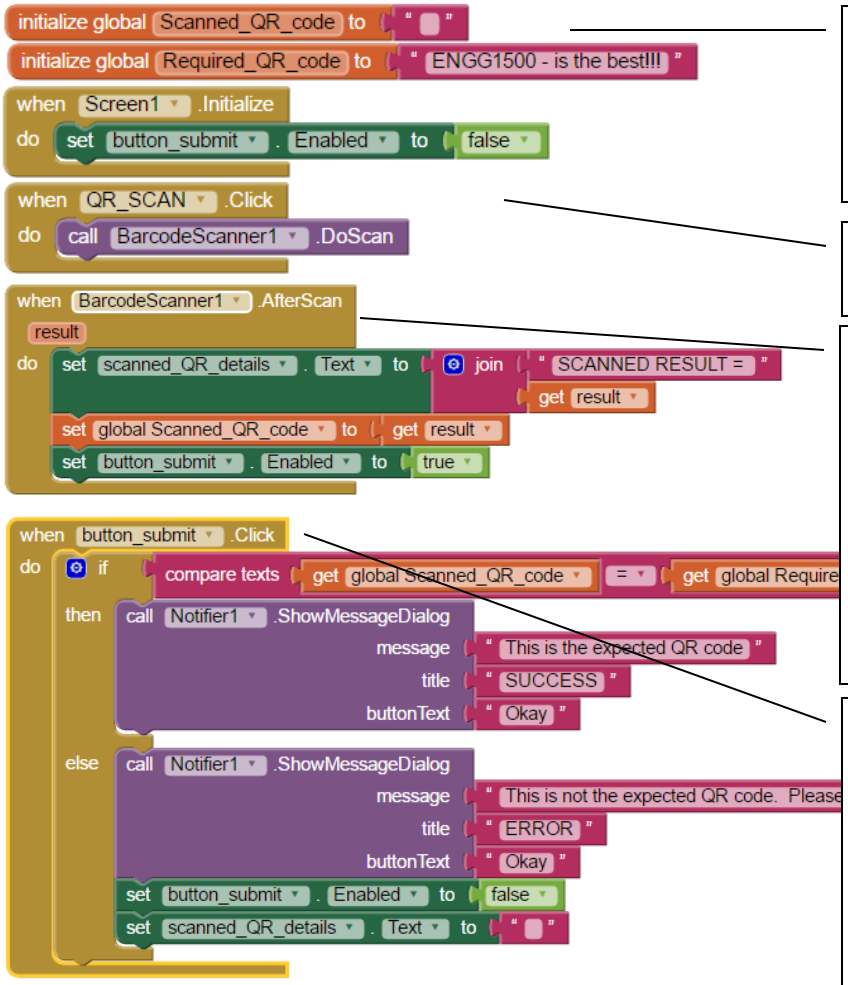
Correct:

Incorrect:

Designer



Blocks



Here we are setting some global variables. One is for the value which we have scanned (this is initialised to an empty string), the other is for the value we are expecting to read

When the screen initialises – we add some control by disabling the submit button

The AfterScan event provides us with the result of the scan. So that you can see this result during testing, we are display it to the screen in a label called scanned_QR_Details. We then store it as our global variable (for later comparison). Finally, we can enable the submit button which will allow us to compare the scanned result against the expected result.

Once we have submitted, we can make the comparison between the Scanned_QR_code, and the Required_Code. We then use the Notifier to show a message to the user, letting them know the outcome of the scan. If it was not the expected code, we disable the submit button, and set the scanned details to the empty string again. The user will be able to scan another QR code.

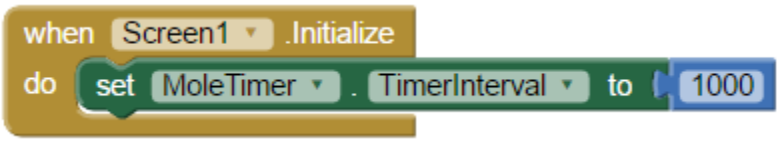
Clock

Have a go yourself!

Update the MoleMash Tutorial

Next to the reset button, include 2 new buttons for "faster" and "slower"
Hint: You may want to use a horizontal alignment for these.

When Screen1 is initialized, set the timerInterval to 1 second (or 1000 milli-seconds)
Hint: The initialize method is an event in Screen 1, and the timerInterval is a variable in the moleTimer. You will need to use an integer from the Math library to set the value of the variable.

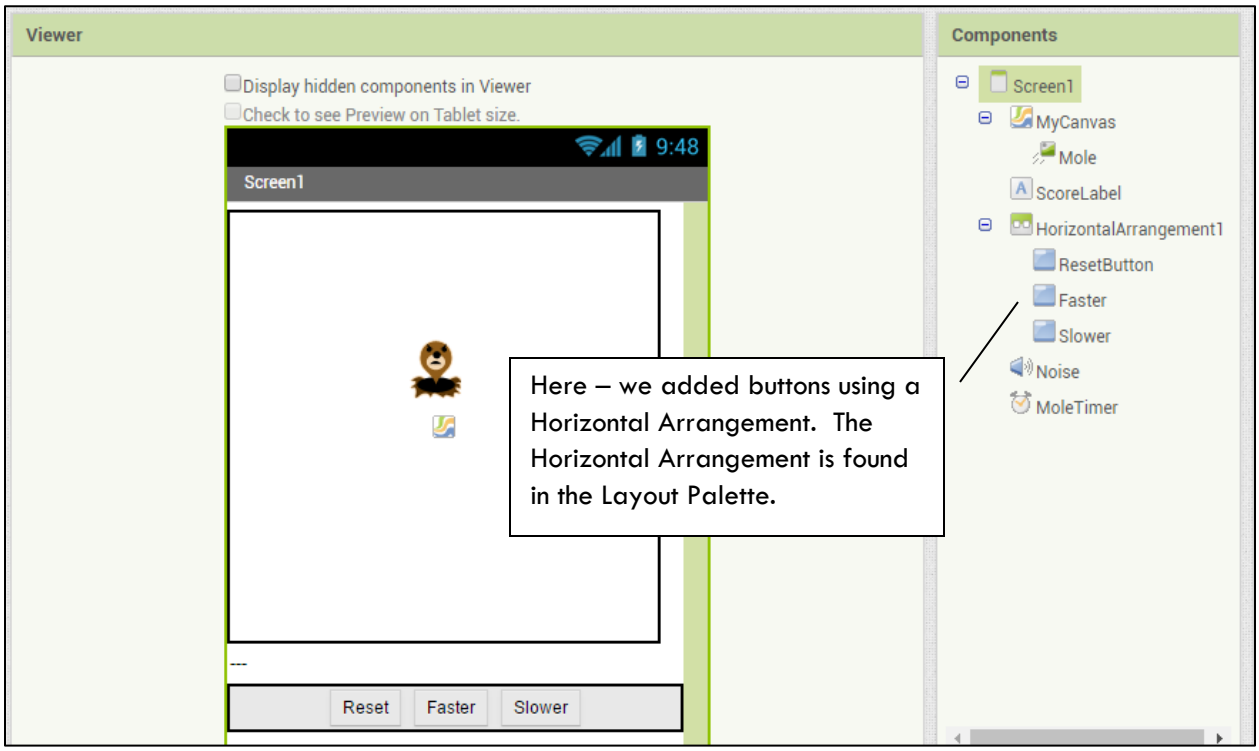


When the button faster is pushed, make the mole move twice as fast.
Hint: set the value of the timerInterval to its current value / 2

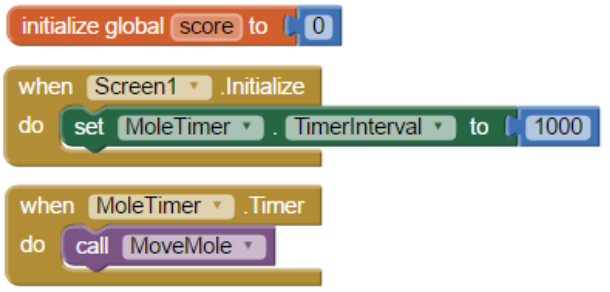
When the button slower is pushed, make the mole move twice as slow.

Make sure you update the reset button, so that the time is re-set to its initial value when this button is clicked.

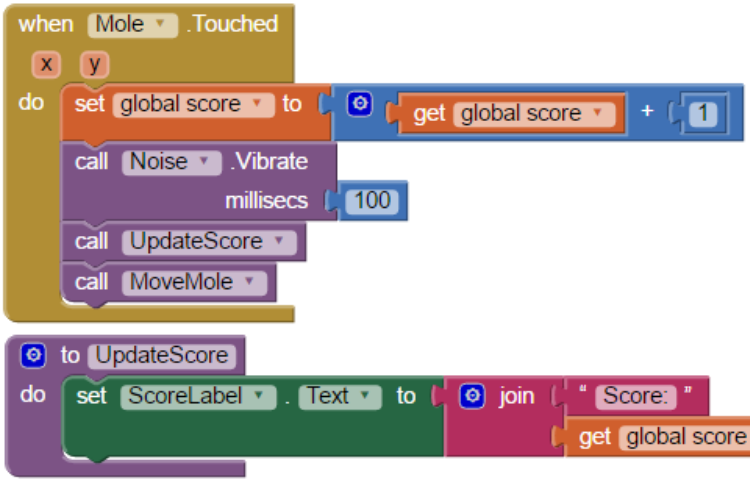
Designer

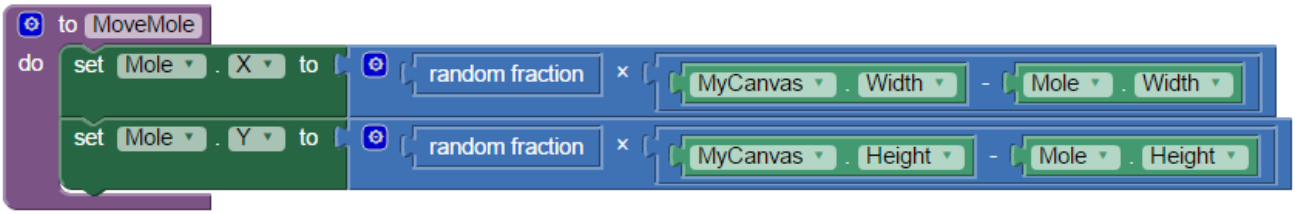


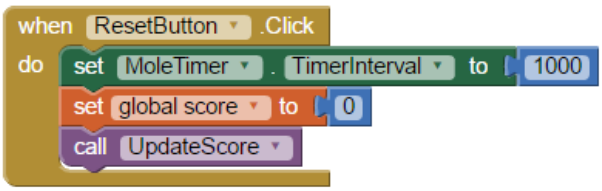
Blocks



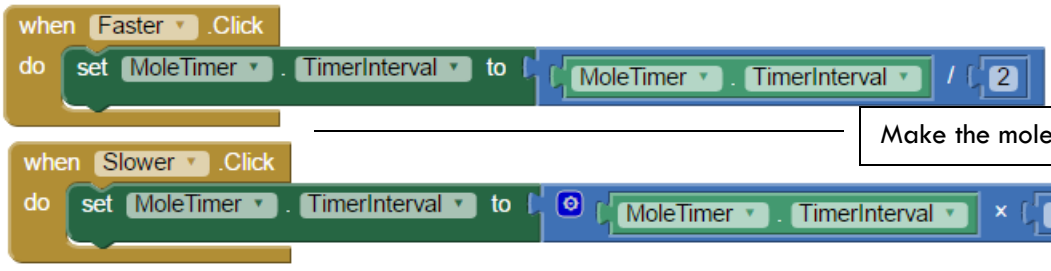
Initialise the value of the moleTimer when the application starts







Update the ResetButton to re-set the value of the MoleTimer



Make the mole move faster or slower.

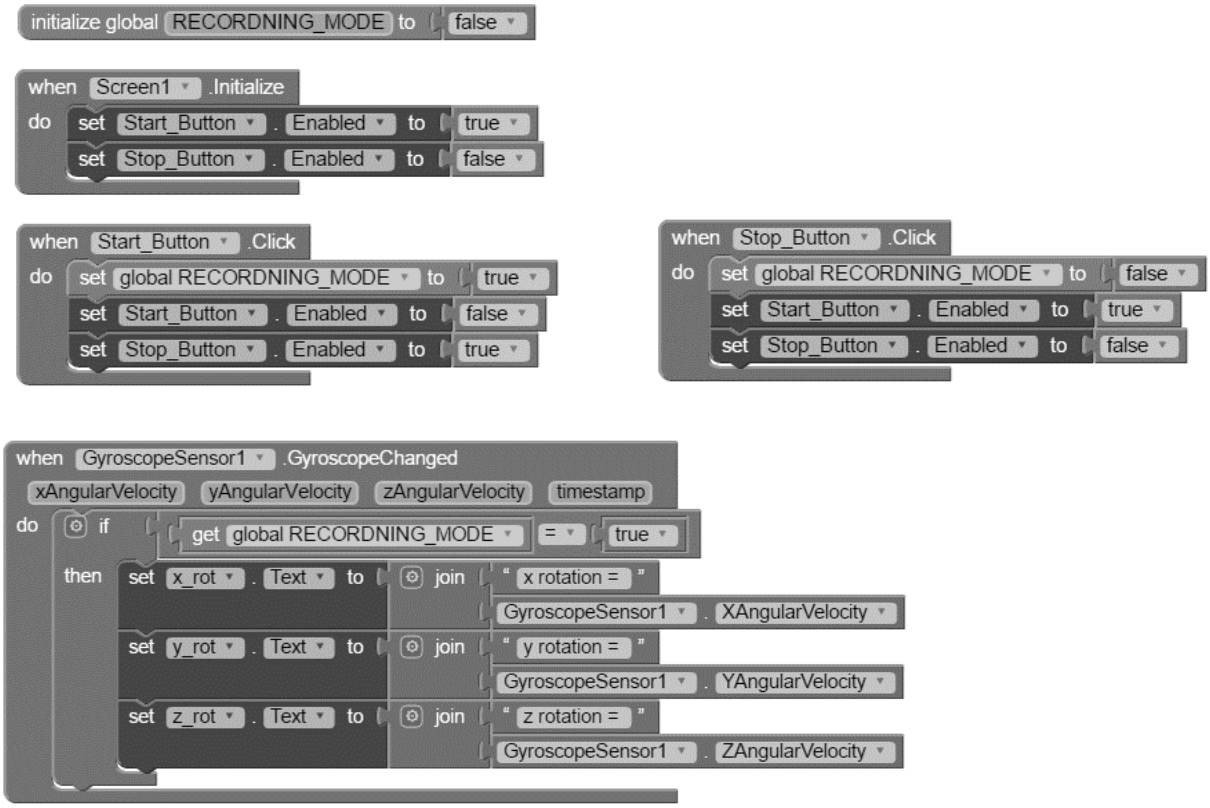
Gyroscope Sensor

Have a go yourself!

Create an app that displays the XAngularVelocity , YAngularVelocity and ZAngularVelocity at any point in time. Note that this application is similar to the one suggested for the Accelerometer.

Designer

Blocks



Location Sensor

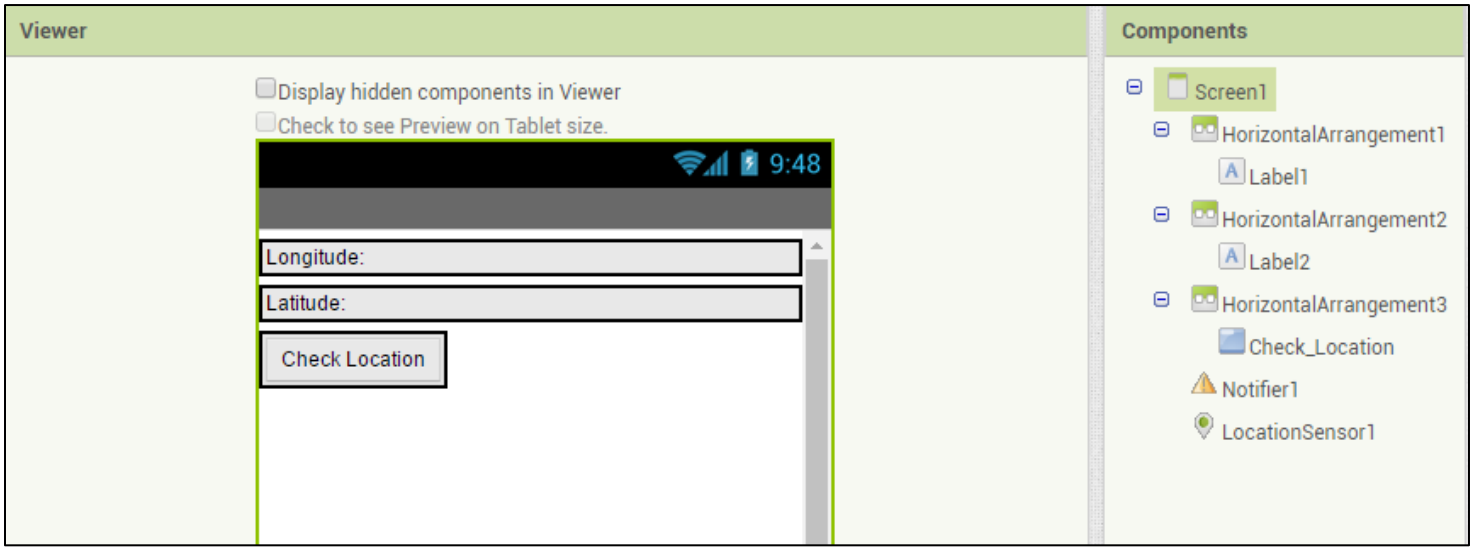
Create a small application that stores as global variables the longitude and latitude of a location.

Add a button on the screen to test if you are currently at the location.

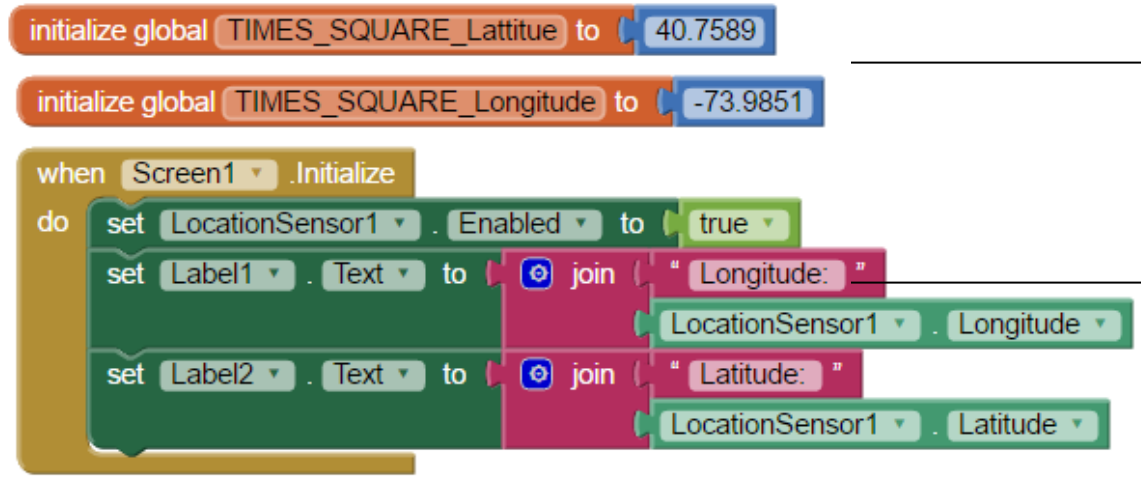
If you are at that location, provide a message to the user stating you are at this place.

If you are not, provide a message to the user stating that you are not detected at this place.

Designer

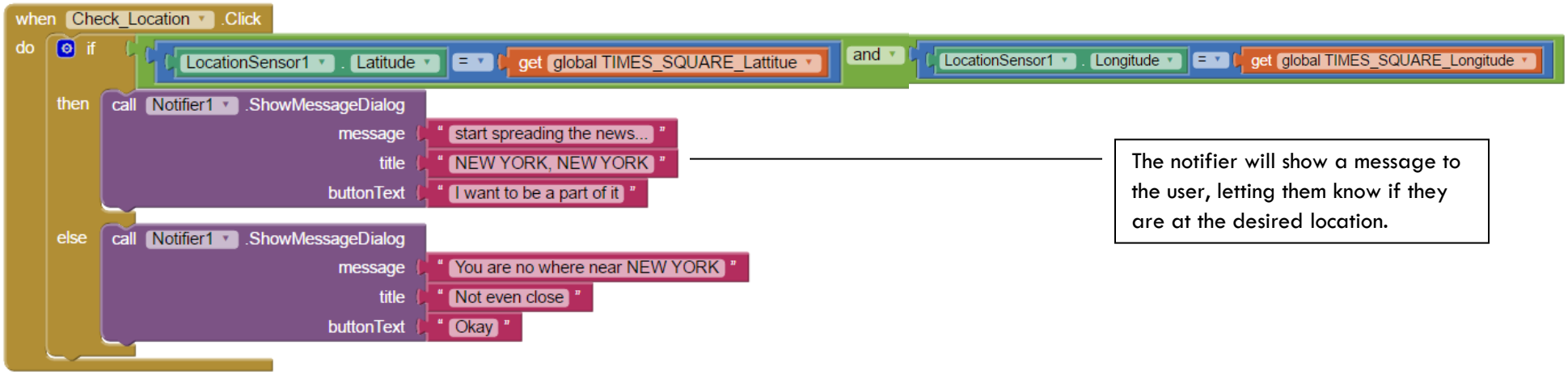


Blocks



We are keeping track of the Latitude and Longitude for TIMES SQAURE – NEW YORK.

When the screen starts up, we will get the Longitude and Latitude of the user's current location.

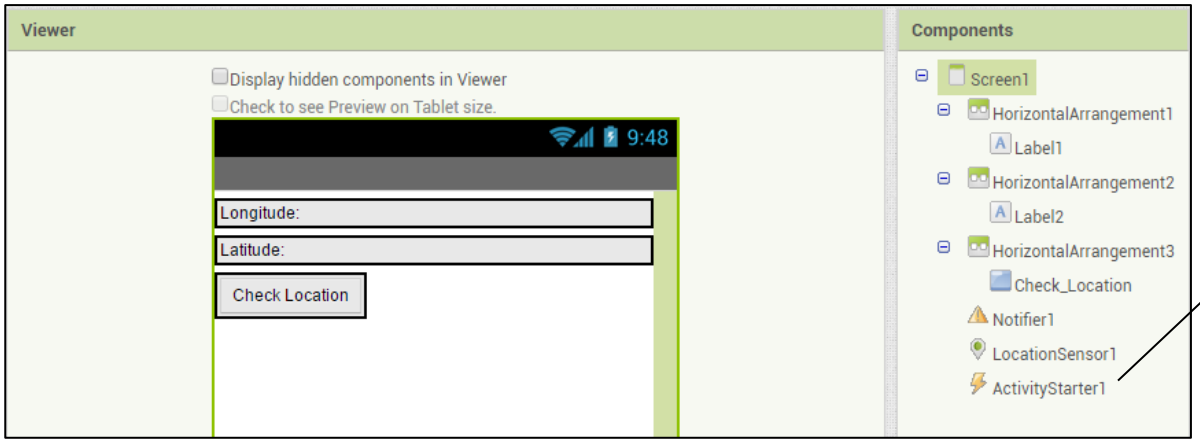


The notifier will show a message to the user, letting them know if they are at the desired location.

Optional: Open up a google maps view pinned to the desired location (i.e; the global longitude and latitude

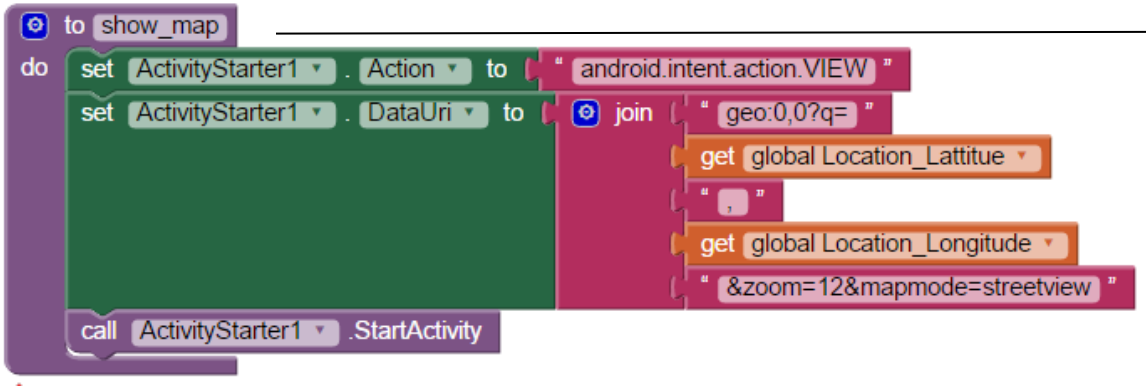
Hint: for this you will need to use an Activity Starter.

Designer



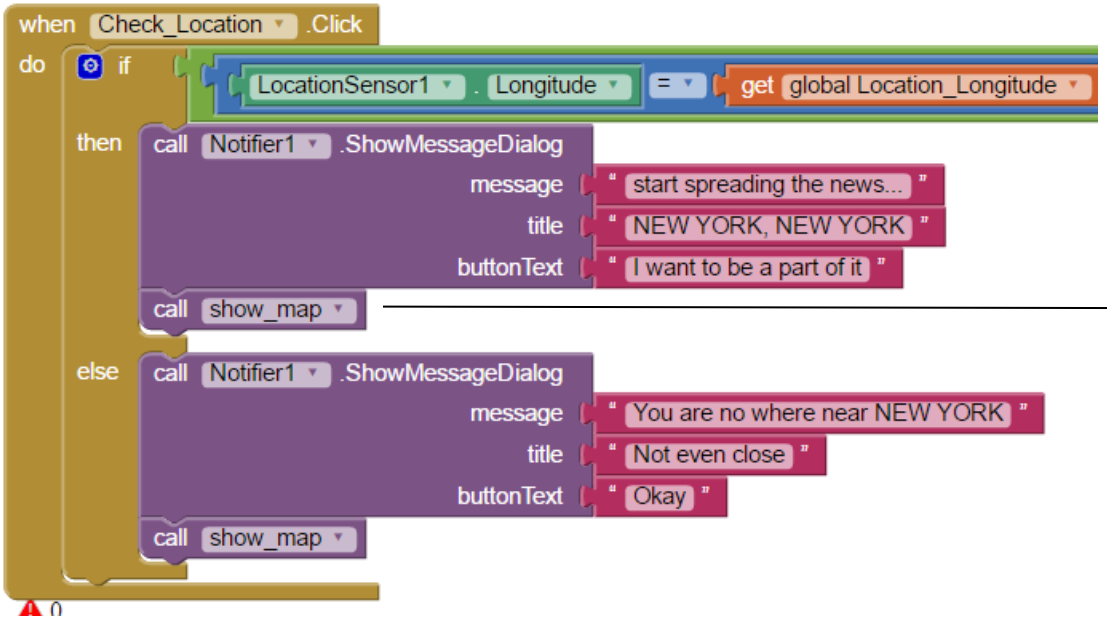
In this example, we include the Activity Starter

Blocks



Then, we make our own method show_ map. This will set the details of the activity starter, so that it will open the maps application, for the location we have specified.

More details about formatting location calls for google maps can be found here. Don't worry if you don't understand all of this yet, feel free to ask your tutor for more help.

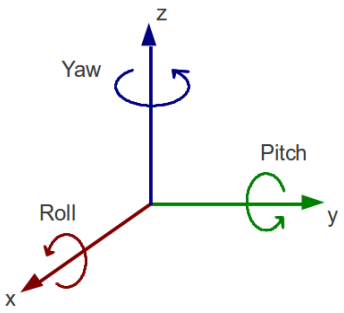


Finally, we need to make sure we update the program – so that it makes a call to show_map.

Orientation Sensor

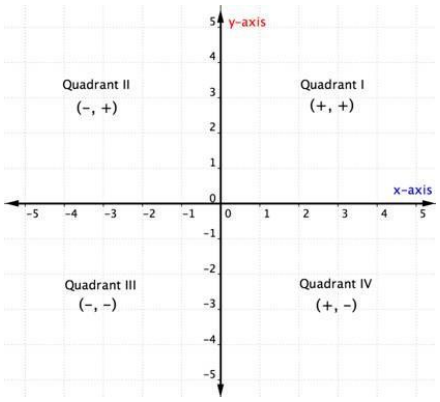
Have a go yourself

The orientation sensor uses **pitch** and **roll** for some of its measurements. The terms pitch and roll come from flight dynamics, where pitch, roll and yaw are used to determine the orientation of an aircraft. You can think of **roll as the rotation around the x axis**, and **pitch as the rotation of the y axis**.

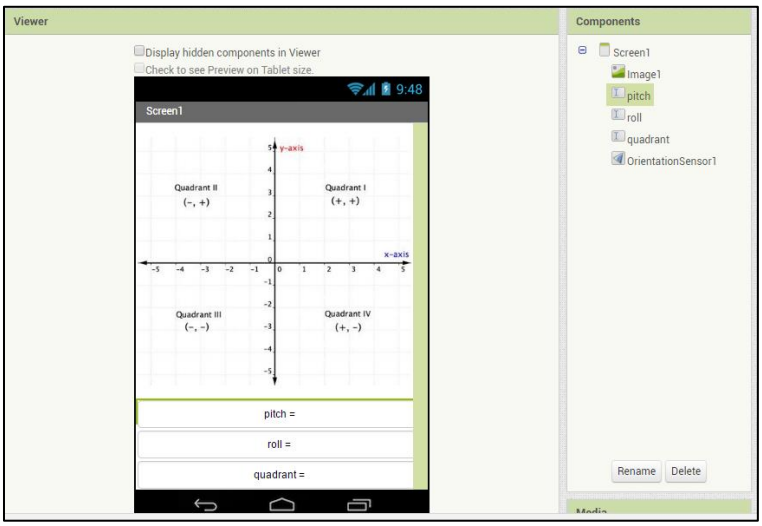


Make an application that displays the raw data for the roll and pitch.

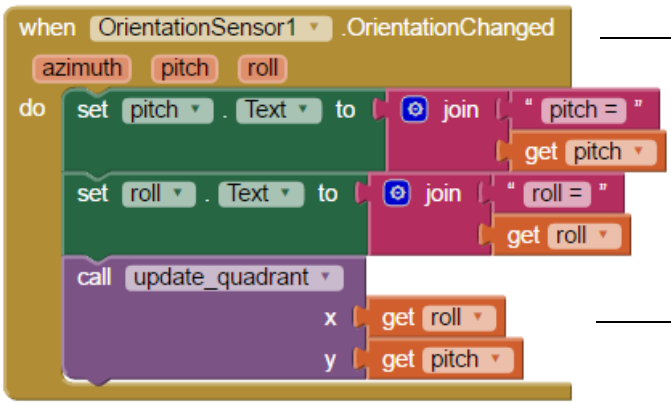
Add some conditional statements to this application. Using the values of roll for the x-plane, and pitch for the y-plane, add a label which displays which Cartesian quadrant the phone current maps to:



Designer



Blocks



When the orientation sensor detects a change, the text labels for pitch and roll will be updated.

We will then make a call to our own method: update_quadrant. Here, we take a copy of the value of roll (to become x), and a copy of the value of pitch (to become y).

This method will use the x and y values (which are copies of roll and pitch) to update the quadrant label.

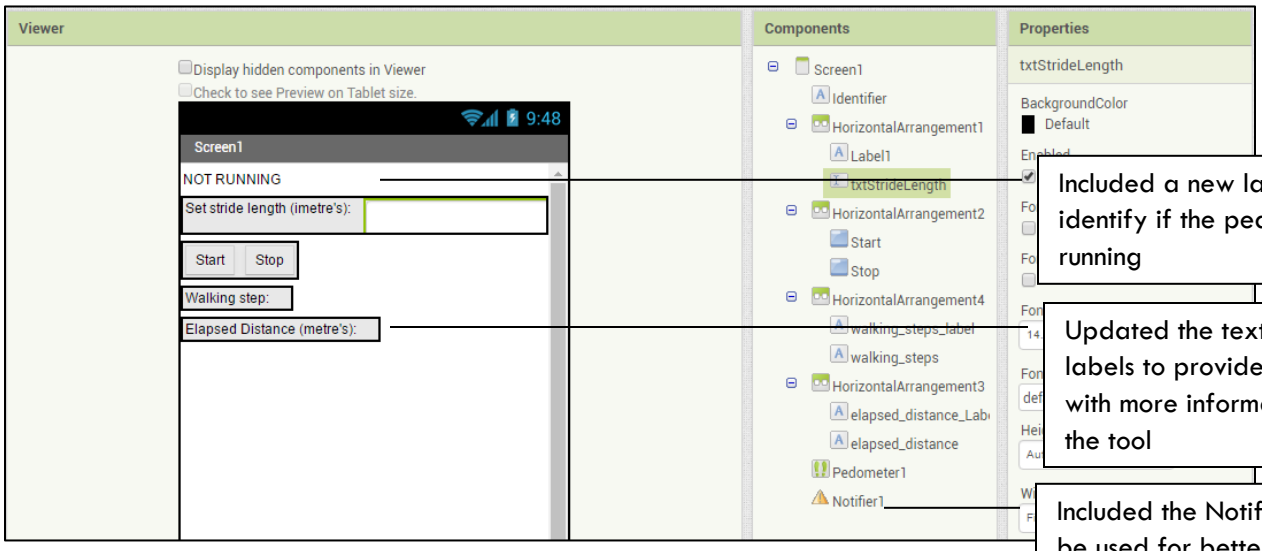
Pedometer

Have a go yourself!

At the moment, the program created in the tutorial above is not so great. It allows negative input for a stride length, and gives little feedback to the user about the program status. To make the program better, and more user friendly, we want to add some conditional statements and more feedback to user.

- If the txtStringLength text field has no value, display the message "Please enter a stride length"
- If the txtStringLength text field has a negative value, display the message "Stride length cannot be negative"
- Include the unit of measure (metre's) for the stride length input text label, and Elapsed Distance.
- Create an identifier that informs the user if the pedometer is running
- Add control to the start and stop buttons, so that the user cannot select to start the pedometer when it is already running, or stop the pedometer when it is not.

Designer

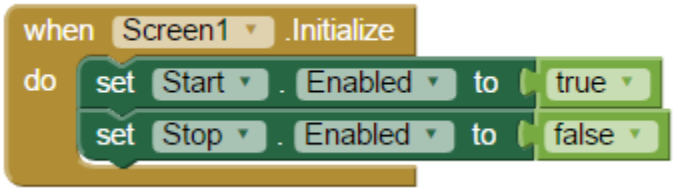


Included a new label to identify if the pedometer is running

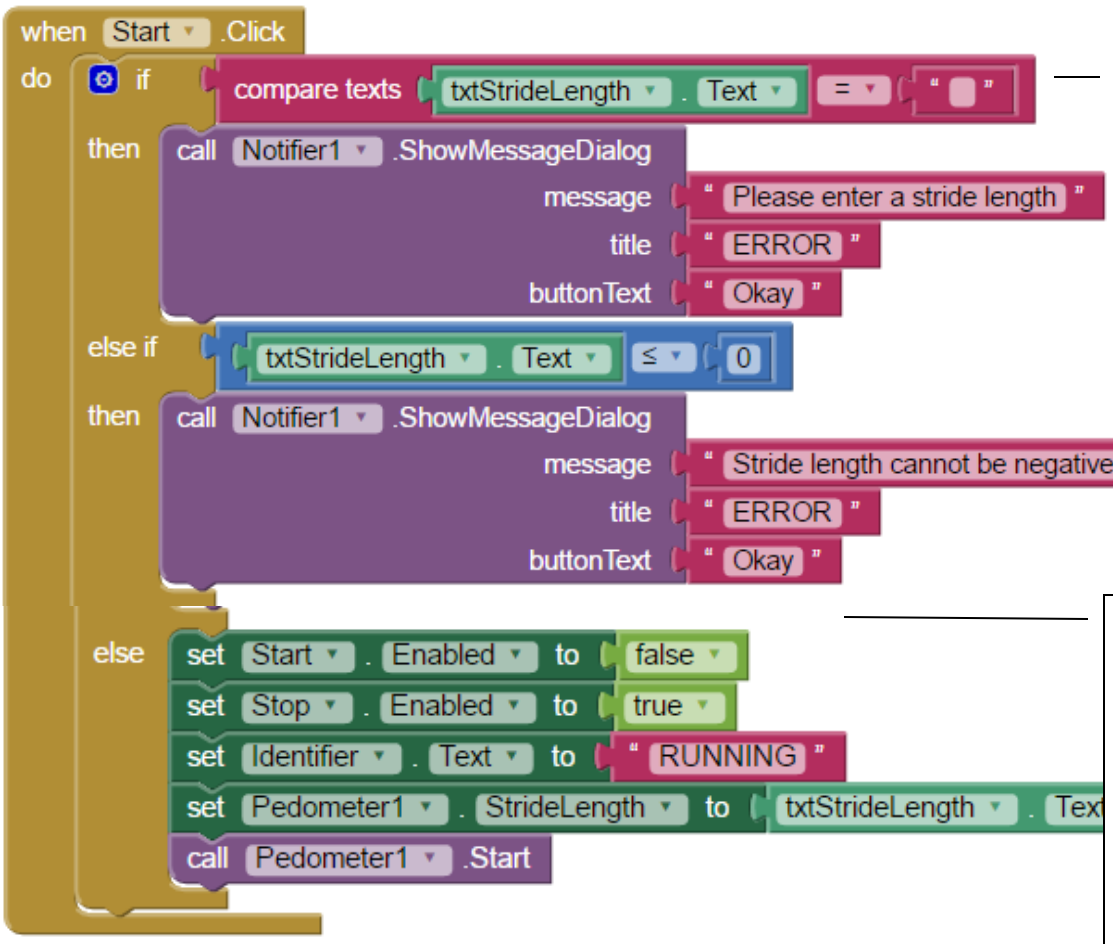
Updated the text on the labels to provide the user with more information about the tool

Included the Notifier, this will be used for better user interaction

Blocks

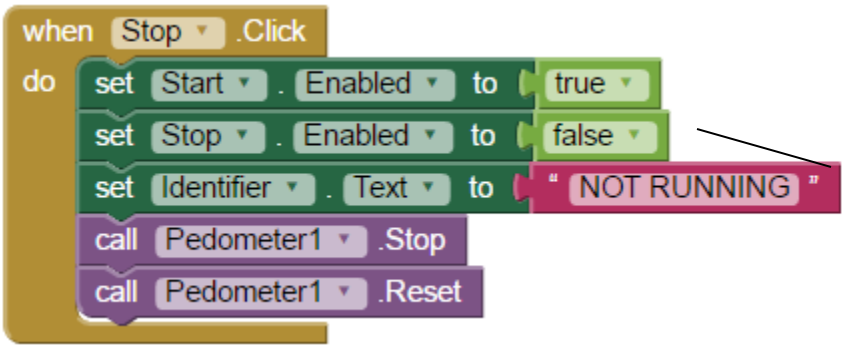


Added some button control when the screen is initialised, so that only the start button can be selected

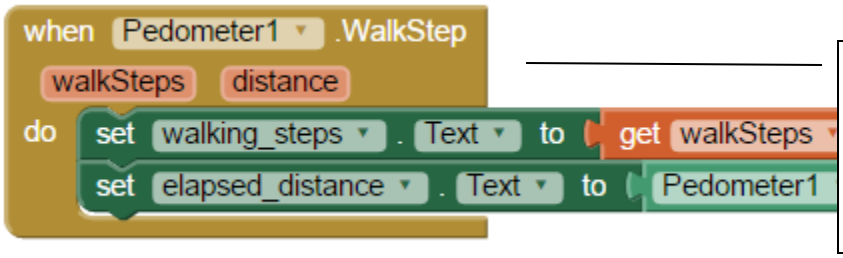


When the start button is clicked, we have included conditional verification of the users input, and provided messages to the user to inform them of incorrect input.

When the users input is correct (the "else" condition), we change which button is enabled. We update the identifier to "Running", we update the pedometer's stride length, and then we start the pedometer



When the stop button is clicked, we change which button is enabled, and update the identifier. We then stop the pedometer and reset it for later use.



When the pedometer is running the walkstep even can be used to update the text boxes, displaying information to the user about the number of steps walked, and the elapsed distance.