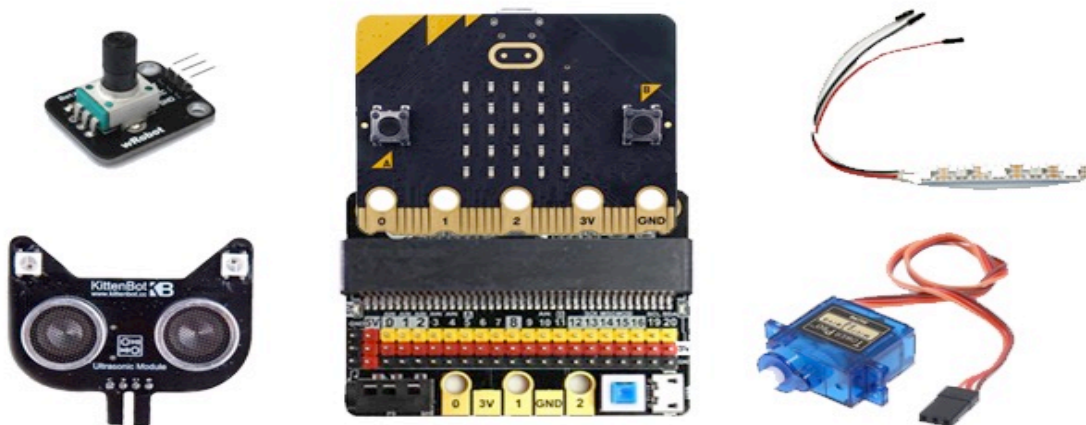# PHSS Hackathon Training 2025

Preface

This training programme is to guide learners to be ready participants of a Hackathon, to pick up micro:bit coding skills & develop innovation by integrating sensors into the micro:bit, to put into practice problem-solving skills in a real-world context following the theme of "Innovative Solution to Tomorrow's Problem"

Trainer Guide

Version 1.3

Last updated on 17 April 2025

1.1.1 What are Hackathons?

Hackathon, a mesh up of "hack" and "marathon," put them together and it is used to describe one of the most popular and powerful problem-solving techniques.

**to hack \ ˈhak **
transitive verb
1: to cut or sever with repeated irregular or unskillful blows
2: to gain illegal access to (a computer network, system, etc.)

**marathon \ ˈmer-ə-ˌthän **
noun
1: a footrace run on an open course usually of 26 miles / 42.2 kilometers
broadly: a long-distance race
2: a: an endurance contest
   b: something (such as an event, activity, or session) characterized by great
    length or concentrated effort

Source: Merriam-Webster

A hackathon a physical or digital event where participants get together for a short period of time to collaborate and generate some form of innovation such as a functioning software or a hardware prototype.

1.1.2 How to prepare a Hackathon

These are some of the things you should do the prepare for a hackathon:

Do your research

Research about the theme in advance so that you are prepared. Knowing the theme before going through the video training will help you appreciate the training more and allow you to be more focused on the parts that can help you with the final prototype.

Know your teammates

Appoint a team leader and get to know each other's strength and weaknesses. If someone is better at coding, let them work on the codes. If someone is better at the presentation, let them do the presentation preparation. Nothing will make a team more disorganised than having teammates who are doing things that they are not good at.

Understand the rules and ask questions

Not only should you research the theme ahead of time, make sure you know the hackathon rules and regulations. If something is unclear, ask the organizers. A lot of times there will be FAQ of the hackathon to make sure everyone is on the same page. Do not be the team that breaks the rules and gets disqualified.

Set your prototype baseline

Work with your team to determine your prototype baseline. Once you achieve your baseline, you can go and add all the enhancements you want, but the baseline ensures that you have met the minimum hackathon requirements.

Be sure to set realistic goals

Make sure that you set a goal that can be accomplished. You may not have the winning prototype at every hackathon, but it is nice to be able to present something you can be proud of for judging.

Do not stress

Hackathons are for fun and learning. If you finish your hack and prototype, great! If not, it is not the end of the world. You still accomplished something. Celebrate your wins!

1.1.3 Example of previous Hackathons

Commercial: Razer Fintech's Digital Hackathon

The Razer Fintech Digital Hackathon will encourage Singapore tertiary students, young professionals and the start-up community to think outside the box and address critical financial challenges that have been exacerbated by COVID-19. Winning participants will have the opportunity to implement and operationalize their original banking solutions within Razer Fintech's financial services ecosystem or with its partners, through securing full-time employment, internships, investments or commercial partnerships.
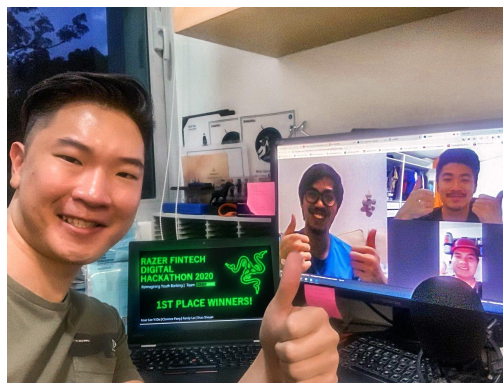
Team Razer Blazers, comprising of Clarence Pang, Randy Lai and Shao Shxuan from the School of Information Systems, and Isaac Lee Yi De from Lee Kong Chian School of Business SMU, have emerged as one of two winners at the inaugural Razer Fintech Digital Hackathon held from 15 to 17 May 2020 in Singapore.

https://press.razer.com/company-news/razer-fintech-launches-digital-hackathon-empowering-singapore-youth-to-create-and-implement-original-banking-solutions-amidst-covid-19-circuit-breaker/

https://news.smu.edu.sg/news/2020/05/20/smu-team-wins-razer-fintechs-inaugural-hackathon-singapore

Commercial: JunctionX Singapore 2019

Organized for developers by developers, JunctionX Singapore 2019 was the first hackathon of JunctionX Singapore team, and it was a massive success. The 48-hour hackathon was organized at the beautiful office of Rakuten Singapore in CBD. Participants came from a diverse range of backgrounds with more than half being international participants (from Algeria, Japan, etc.) who came to Singapore just to attend this hackathon.

u

JunctionX Asia 2020

JunctionX Asia is an online hackathon targeted at solving problems the global community is facing during this ongoing pandemic. Our goal lies in bringing developers, designers and other tech-minded individuals together to create exciting projects and solve intriguing challenges. You will work in a team within 4 days to build a project, realizing it from just an idea to an actual product.

Green Scanner

A gamified mobile application that encourages consumers and merchants to adopt conservative measures towards carbon emissions.

Green Scanner aims to solve the carbon emission problem through a gamified platform. Users of Green Scanner mobile application can accumulate reward points as they purchase items of low carbon emission. Green Scanner also helps to track the carbon emission produced through the user's consumption. Green Scanner aims to educate users on the importance of minimising carbon footprints while motivating merchants to play their part through corporate social responsibility by having operations that is minimal in carbon emissions. Green Scanner is developed in Flutter, hence available on android and iOS devices. Green Scanner leverages on Azure services for its back-end and REST APIs are hosted on Azure App Service. In the long run, Green Scanner aims to incorporate Data Analytics for its large data sets gathered from users' consumption of carbon emission products that can be used by corporates to improve their product standing.
This is our pitch deck: https://docs.google.com/presentation/d/1VBipHGvXUma2zHcfQqHGtp5HLkQIOgnr6RmbSNqz6U4/edit?usp=sharing

VIDEO
No video available

DEMO
https://www.youtube.com/watch?v=ZxUzI3PpCks

SOURCE CODE
https://github.com/geraldspacelim/green_scanner_flutter

CHALLENGES
Develop for Social Good (Microsoft)

https://singapore.hackjunction.com/JunctionxSingapore2019

https://asia.hackjunction.com/

Education: Hack&Roll 2021

Hack&Roll is a hackathon organized by NUS. Build whatever you want! You can even choose to build something absolutely useless, just for fun! Build something new. You may reuse code for specific components, but recycling an entire project is not okay. Projects will be judged on awesomeness. Decisions made by the judges are final.

https://hacknroll.nushackers.org/

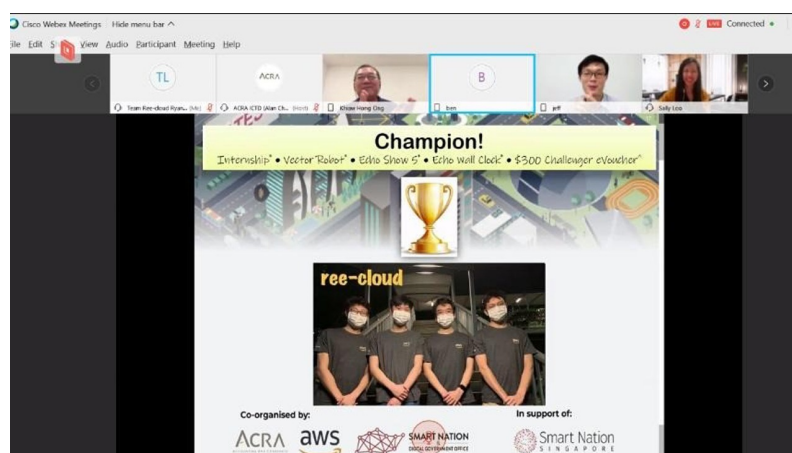**Government: Live Smart Singapore Hackathon (ACRA)**

The Accounting and Corporate Regulatory Authority (ACRA), Amazon Web Services (AWS) and Smart Nation and Digital Government Office (SNDGO) invite all polytechnic students to participate in the Live Smart Singapore Hackathon (ACRA). Showcase your tech ideas to develop innovative digital solutions to help make Singapore the best place for business.

Team of 4 students consisting of 3 SIT students and graduating student awaiting his NS. Won 1st prize and ACRA's Popular Choice Award in the inaugural Live Smart Singapore Hackathon (ACRA), held virtually on 6 Aug 2020. Competition jointly organized by ACRA, Amazon Web Services (AWS) and Singapore National Digital Government Office (SNDGO) from Jun - Aug 2020.The hackathon was aimed at harnessing tech ideas from polytechnic students to develop innovative digital solutions to help to solve real business challenges. The team created a solution to automate ACRA's appeal for waiver process using a suite of AWS technology to boost productivity and improve customer satisfaction.

https://aws-educate.wixsite.com/smartnationhackathon

https://www.nyp.edu.sg/schools/sit/achievements-and-awards/2020/live-smart-singapore-hackathon-2020.html

1.1.4 Tools and materials used in the Hackathon

1. micro:bit v2.2 kit
2. KittenBot IObit
3. 3V Servo Motor
4. Analog Rotation Sensor
5. Ultrasonic Sensor
6. NeoPixel LED Strip
7. Jumper Wires
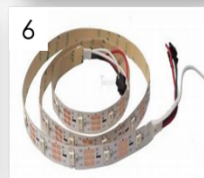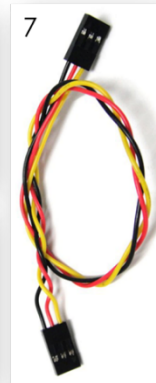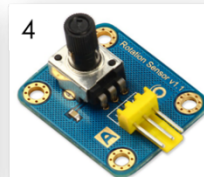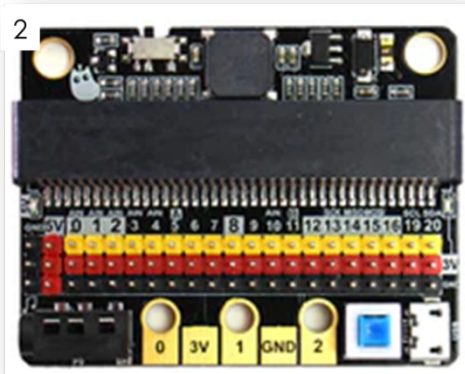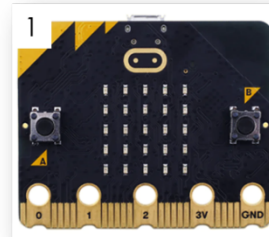
1.2.1 What Is the micro:bit v2.2?

The BBC micro:bit is a pocket-sized computer that introduces you to how software and hardware work together. It has an LED light display, buttons, sensors and many input and output features that, when programmed, let it interact with you and your world.

The new micro:bit with sound adds a built-in microphone and speaker, as well as an extra touch input button and a power button.
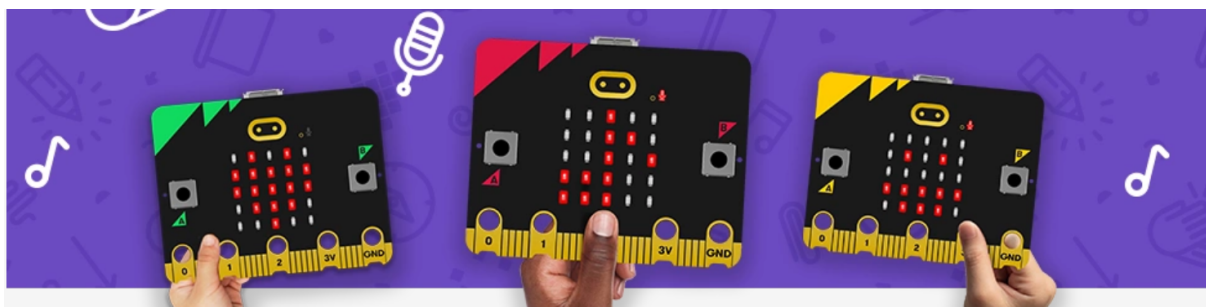
1.2.2 Features – Hardware
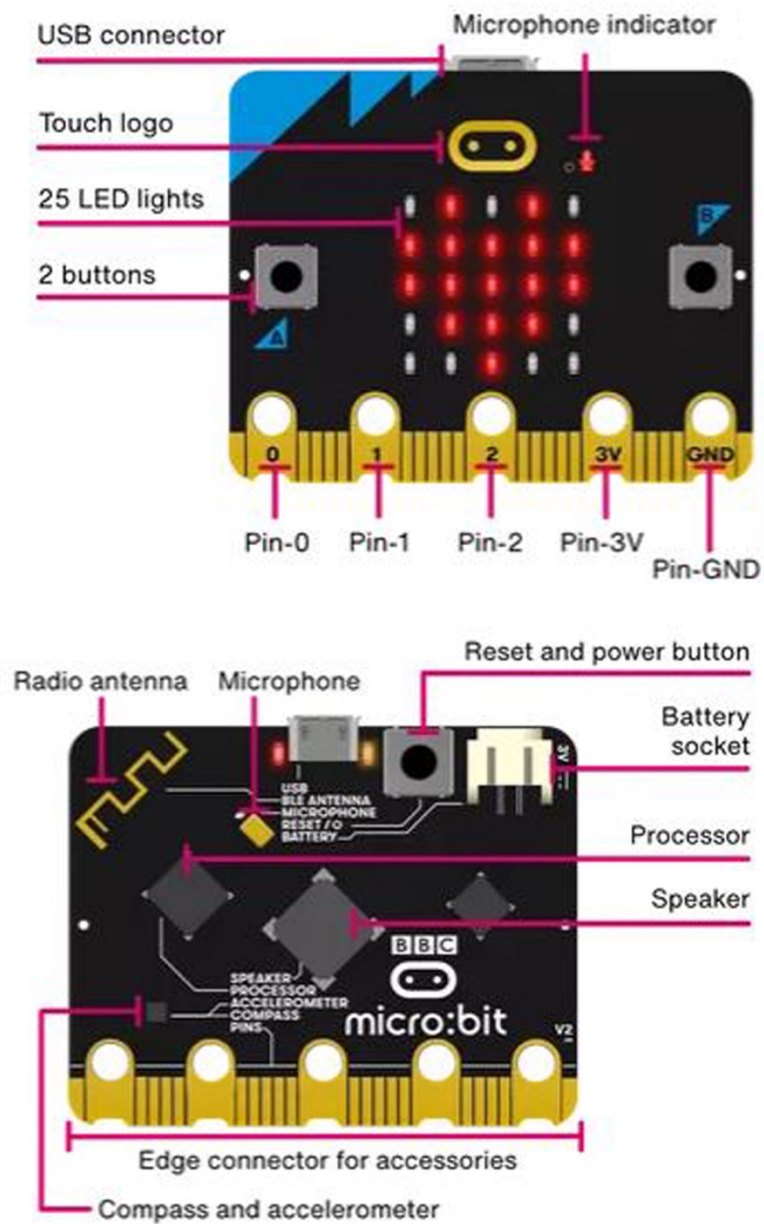
Display

An LED, or light-emitting diode is an output device that gives off light. Your BBC micro:bit has a display of 25 LEDs for you to program.

User Buttons

Buttons are a common input device. Your micro:bit has two buttons you can program, and a reset button.

Accelerometer

An accelerometer is a motion sensor that measures movement. The accelerometer in your BBC micro:bit detects when you tilt it left to right, backwards and forwards and up and down.

Temperature sensor

A temperature sensor is an input device that measures temperature. Your BBC micro:bit has a temperature sensor inside the processor which can give you an approximation of the air temperature.

Light sensor

A light sensor is an input device that measures light levels. Your BBC micro:bit uses the LEDs to sense the levels of light and lets you program your micro:bit as a light sensor.

Compass

A digital compass is an input sensor that detects magnetic fields. Your BBC micro:bit has an inbuilt compass that can detect the direction in which it is facing.

Touch logo

The touch logo uses capacitive touch, sensing tiny changes in electrical fields to know when your finger is pressing it - just like your phone or tablet screen.

Speaker

The new micro:bit has built-in speaker, which makes it easy to add sound to your projects. Any micro:bit sound project will work with the speaker, but with the new micro:bit you can also express yourself with some new sounds: make your micro:bit giggle, greet you or let you know when it is sleepy or sad.

You can also mute the speaker and sound will still come out of the pins so you can still enjoy micro:bit music on headphones connected to GND and pin 0. In MakeCode, use the music block 'set on-board speaker off'.

Microphone

The new micro:bit has a built-in microphone. You can use it as a simple input - make your micro:bit turn the lights on when you clap. It can also measure the amount of sound, so you can make a noise level meter or disco lights that beat in time with music.

The microphone is on the back of the new micro:bit, and on the front, you'll find a new microphone LED next to the hole that lets the sound in. It lights up to show you when your micro:bit is measuring sound levels.

Radio

Radio is a way of sending and receiving messages and BBC micro:bits can use radio waves to communicate with each other.

Pins

On the bottom edge of your BBC micro:bit there are 25 gold strips, called pins. These pins allow you to really get creative. You can create circuits, connect external things like buzzers and motors and make your own fun projects.

1.2.3 Using MakeCode Blocks Editor

1. Using Google Chrome, key in the following website: *http://makecode.microbit.org/*
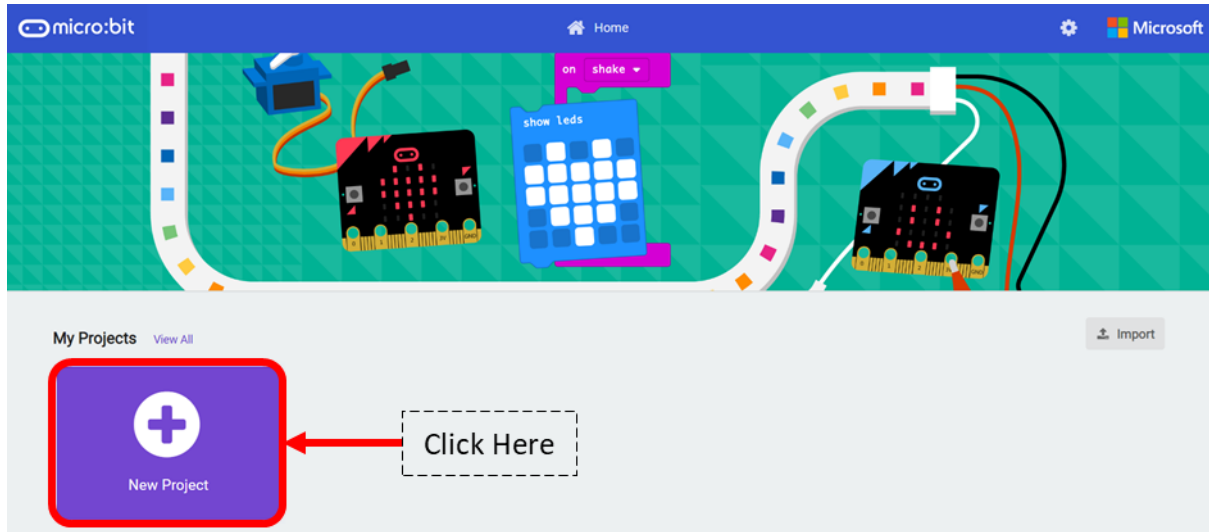
2. Click on [New Project] and give the project a name. Click on the [Create] button.
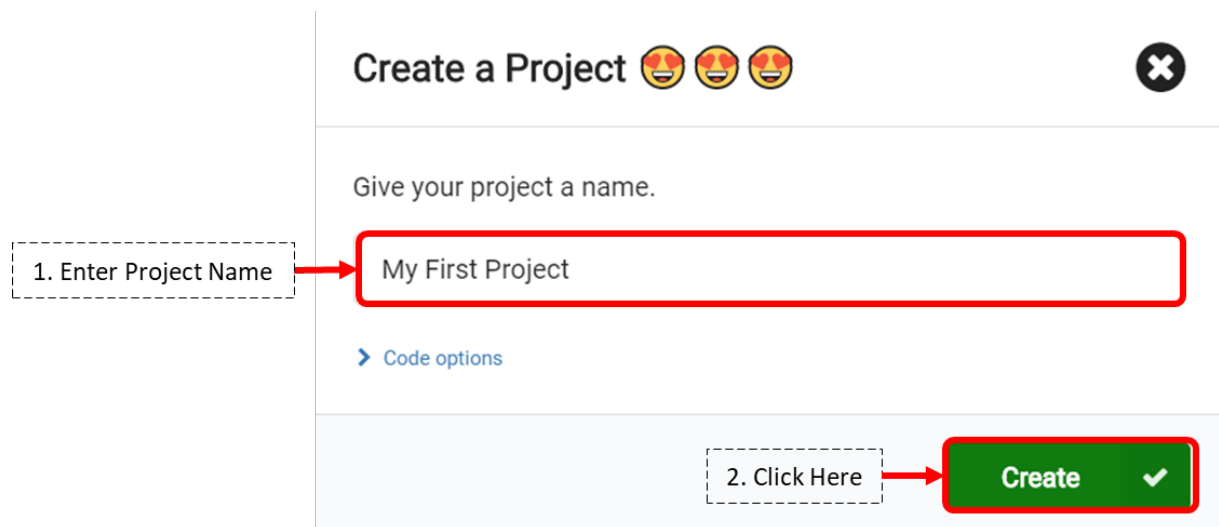
3. The different areas of the coding interface.
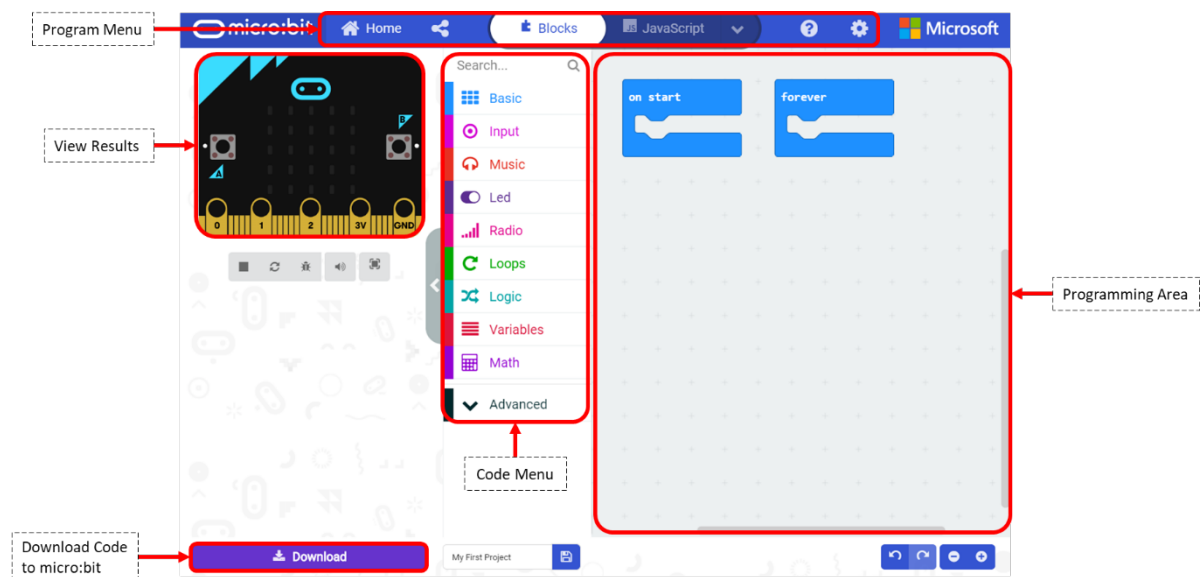
1.2.4   Getting A Program on the micro:bit

1. Connect the smaller end of the USB cable to the micro-USB port on the micro:bit and the other end to a USB port on your computer.
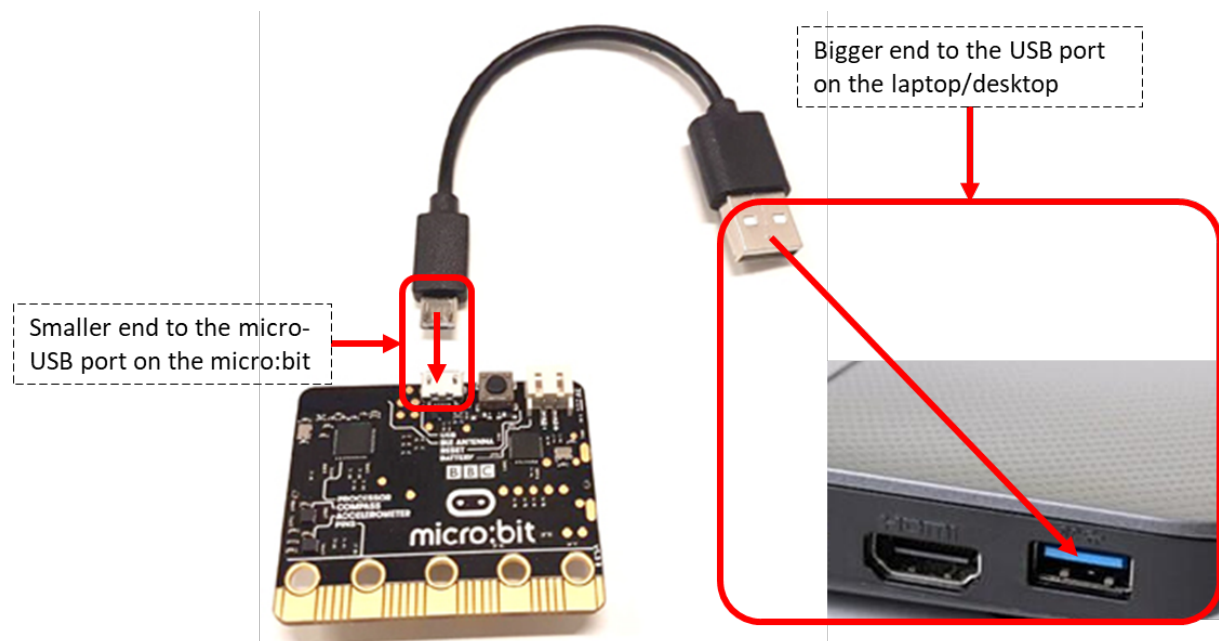
2. Your computer should recognize your micro:bit as a new drive and will appear as MICROBIT drive under Devices and drives.
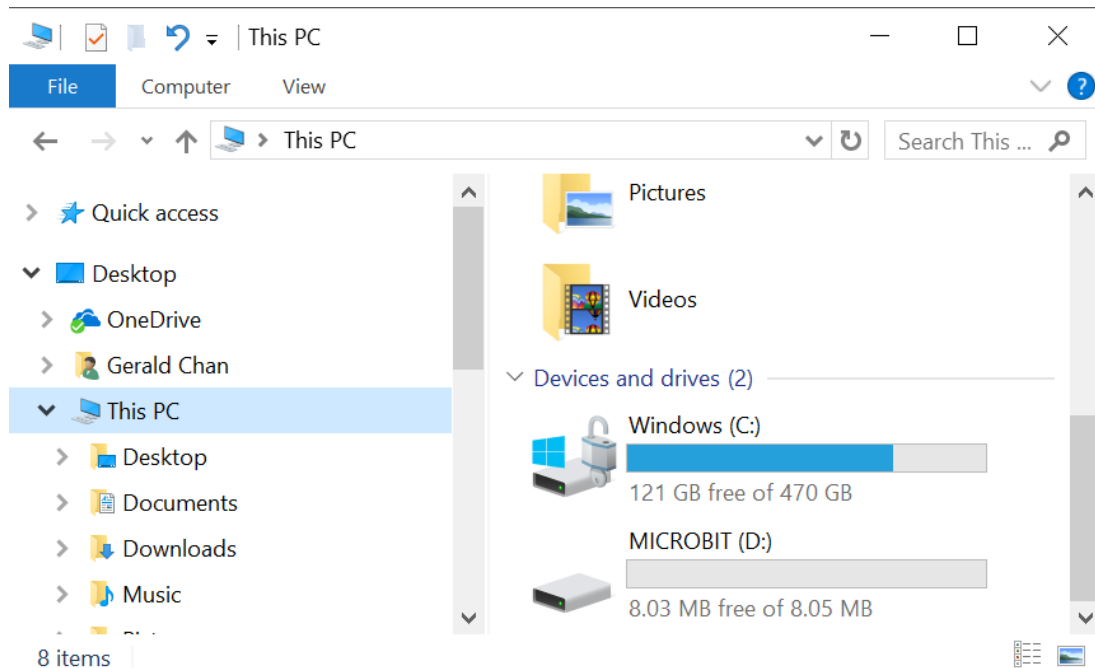
3. Download the program
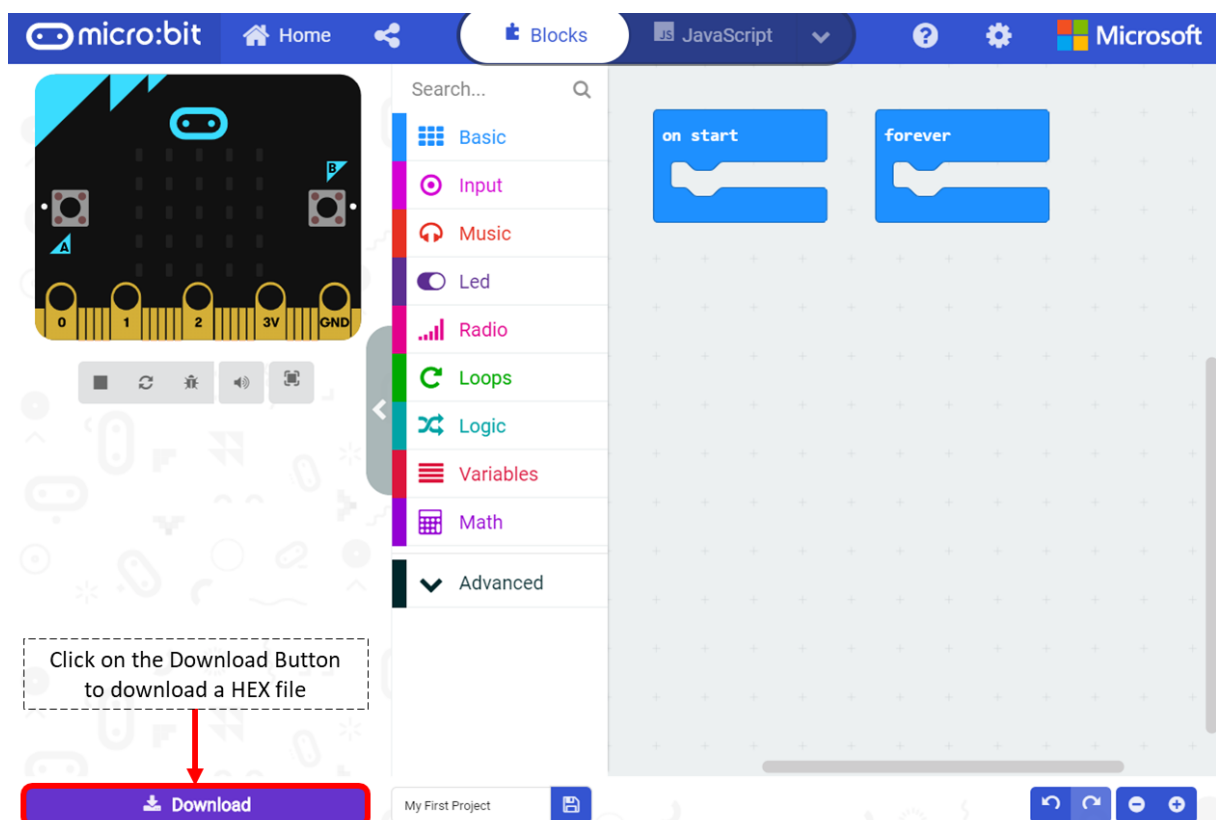
4. Transfer the program to the micro:bit
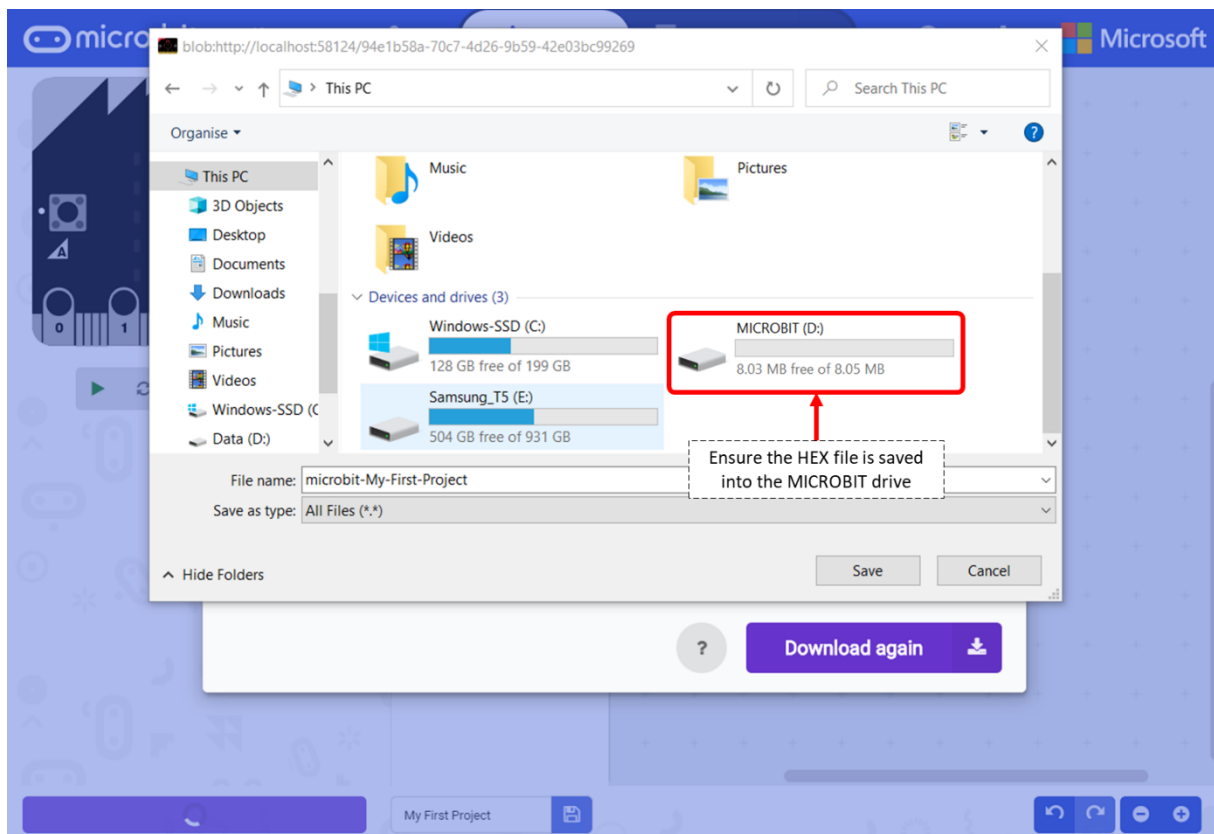
5. Once transferred, the code will run automatically on your micro:bit. To restart your program, press the reset button on the back of your micro:bit.

6. By copying the HEX file onto the MICROBIT drive, you have programmed it into the flash memory on the micro:bit, which means even after you unplug the micro:bit, your script will still run if the micro:bit is powered by battery.

Troubleshooting Note:

1. *You cannot drag and drop more than one hex file at once onto your micro:bit. If you try to drag and drop a second hex file onto your micro:bit before the first file has finished downloading, then the second file may fail in different ways.*

2. *When the first hex file has been written to the micro:bit, the drive will disengage. If you drag and drop a second hex file at this point it may not find the drive and the second write will fail.*

1.2.5 Blinking, Simple Animation, Scrolling Text & Display Number

Blinking Heart LED

1. Click on the Basic menu and drag the [SHOW LEDS] block to the programming area under the [FOREVER] block.
2. Choose the following LEDS to display a blinking heart animation
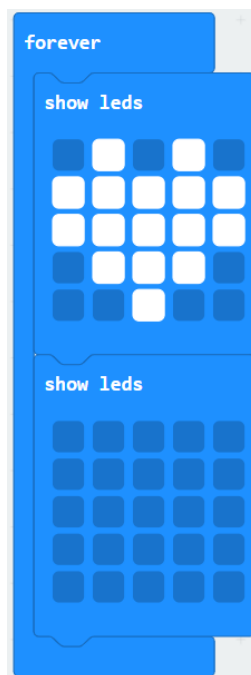
Scrolling Text LED

1. Click on the Basic menu and drag the [SHOW STRING] block to the programming area under the [FOREVER] block.
2. Choose the following LEDS to display a scrolling line of text "Hello"
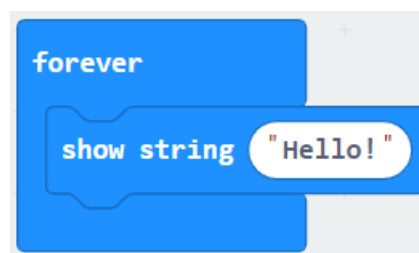
Display Number LED (1)

1. Click on the Basic menu and drag the [SHOW NUMBER] block to the programming area under the [FOREVER] block.

2. Choose the following LEDS to display a scrolling number "123"
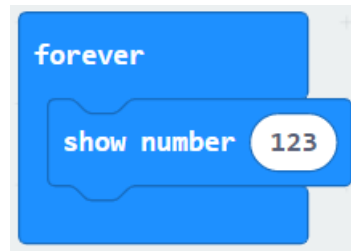
Display Number LED (2)

1. Click on the Basic menu and drag the [SHOW NUMBER] block to the programming area under the [FOREVER] block.

2. Choose the following LEDS to display individual numbers "1" followed by "2" followed by "3"
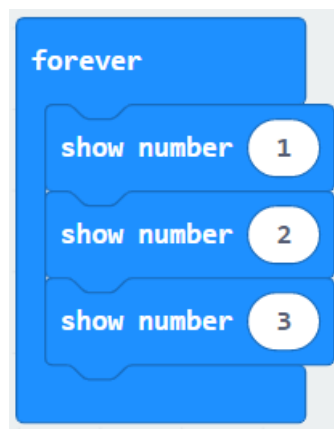
Display Simple Animation LED

1. Click on the Basic menu and drag the [SHOW LEDS] block to the programming area under the [FOREVER] block.

2. Choose the following LEDS to display a dancing man animation

1.2.6 Activity 1: Program a LED animation that shows chasing LED lights looping forever

Each LED is assigned an X and Y value (X,Y) corresponding to its position on the matrix.
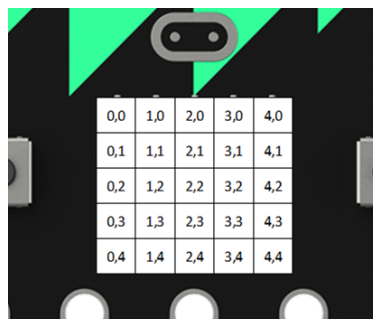
1. Click on the LED menu and drag the [TOGGLE] block to the programming area under the [FOREVER] block.

2. Click on the BASIC menu and drag the [PAUSE (MS)] block to the programming area under the [TOGGLE] block

3. Repeat Steps 1 & 2, 4 more times.
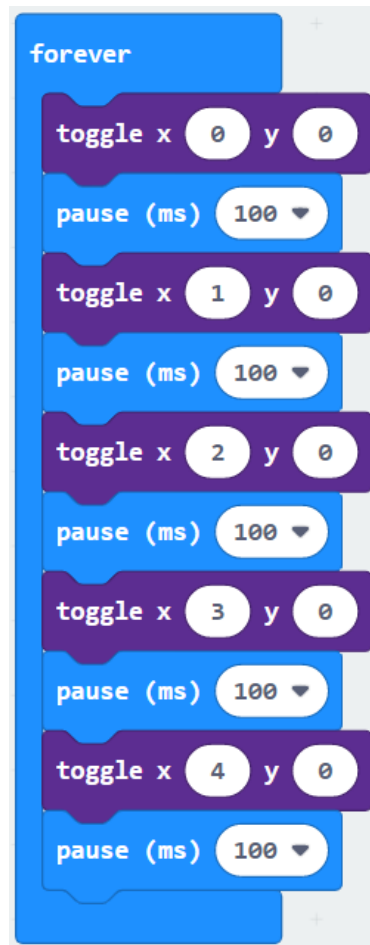
1.3.1 What is a Buzzer?

It is an output device that generates a buzzing noise. Typical uses of buzzers include alarm devices, timers, and confirmation of user input such as a mouse click or keystroke. It works when an electrical field is applied which causes the length of the piezoelectric material surface to change and converts electrical energy into mechanical energy that creates sound waves the human ear can detect.

1.3.2 Active & Passive Buzzer

A passive buzzer is an electromagnetic squeaker used to generate sound signals of different frequencies while an active buzzer is a module that produces a sound of about 2 kHz.

The main difference between the active buzzer and the passive buzzer is that the active buzzer generates sound independently. To do this, the user simply turn it on or off; in other words, by applying a voltage to the contacts or by de-energizing. On the other hand, a passive buzzer requires a signal source, which will set the sound signal parameters.

Active buzzer   Passive buzzer

1.3.3 Activity 2: Improve LED animation

1. Click on the LED menu and drag the [TOGGLE] block to the programming area under the [FOREVER] block.
2. Click on the BASIC menu and drag the [PAUSE (MS)] block to the programming area under the [TOGGLE] block
3. Repeat Steps 1 & 2, 4 more times.
4. Click on the MUSIC menu and drag the [PLAY SOUND UNTIL DONE] block to the programming area under the last [PAUSE (MS)] block.
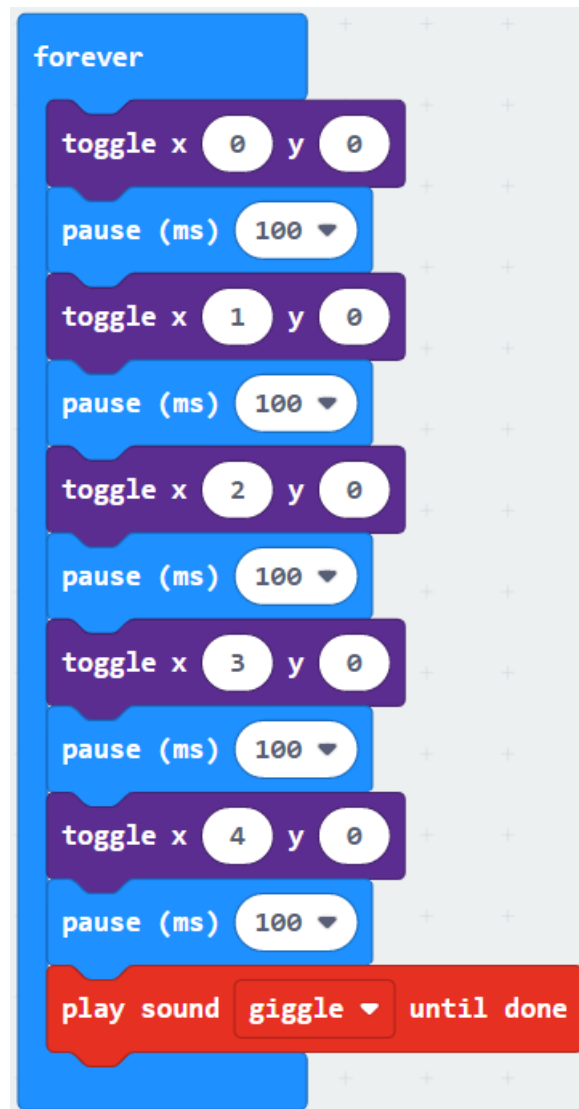
1.4.1 Button State and Event

What is a push button?

It is an input sensor that is pushed to operate an electrical device. Typical uses include calculators, doorbells and emergency stop buttons.

Button State

The buttons on the micro:bit has 2 states; [PRESSED] & [NOT PRESSED]. It can be applied on button A, button B and button A+B.
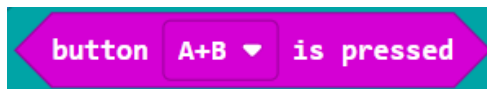
Button Event

The buttons on the micro:bit supports 3 events; [ON BUTTON A PRESSED], [ON BUTTON B PRESSED] & [ON BUTTON A+B PRESSED]
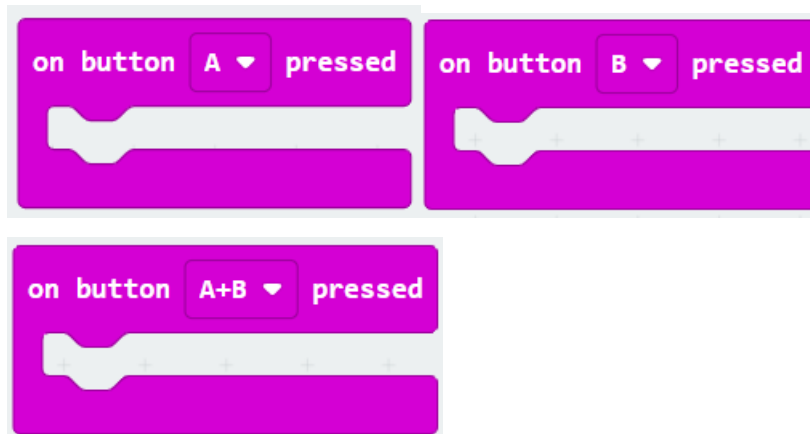
1.4.2 Selection/Conditional Statements

If . . . else Statements

If...else statement is a set of programming instructions that is performed at a defined point of the program once. It allows the programmer to execute different sets of instructions when different defined conditions are met. For example, IF the input condition is "Raining?", the program will execute [Open Umbrella] when the result is TRUE and execute [Keep Umbrella] when the result is FALSE.
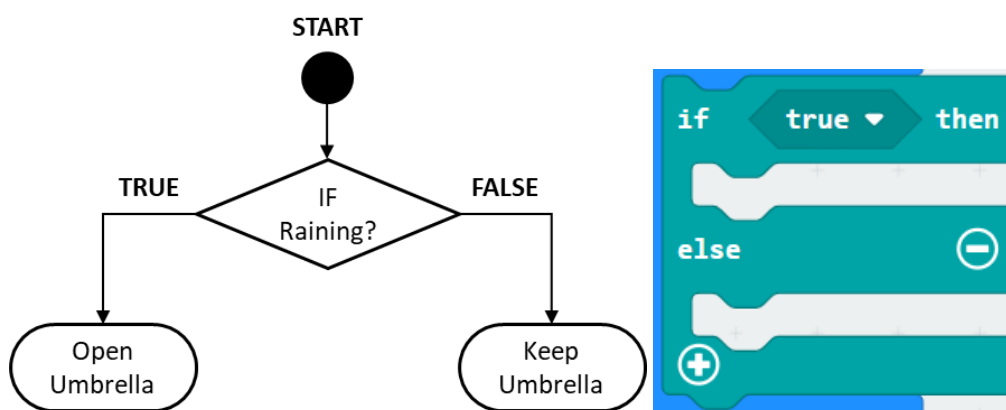
Conditional Loops

A conditional loop is a set of programming instructions that is performed at a defined point of the program repeatedly the counter number is reached. For example, WHILE the input condition is "Raining?", the program will execute [Use Umbrella] when the result is TRUE and keep repeating the same input condition. Only when the input condition result is FALSE then will the program execute [Keep Umbrella].
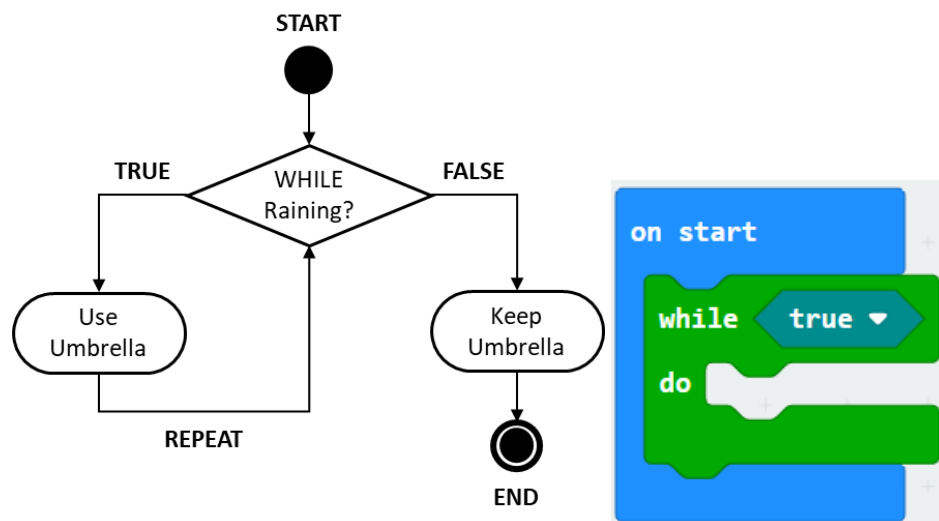
1.4.3 Variables

A variable is a virtual storage location for data in a computer program. This data can be retrieved subsequently by the computer program for use.
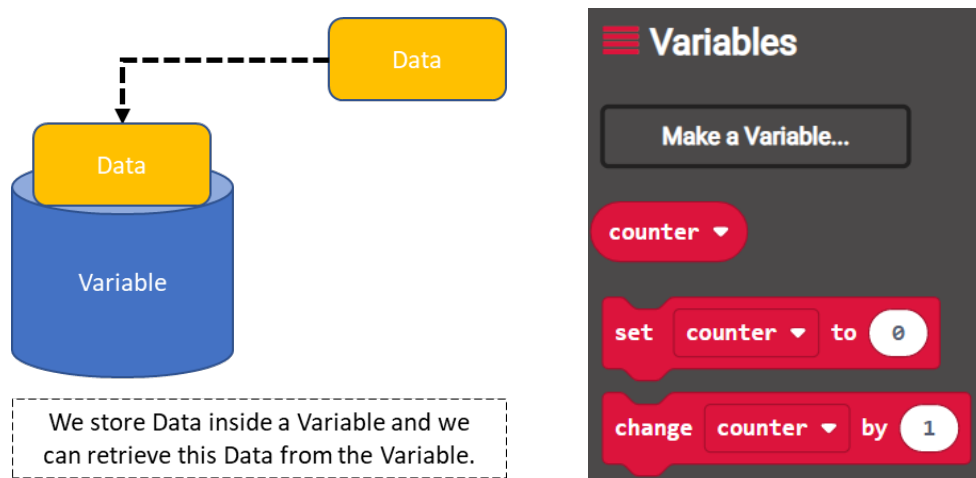
1.4.4 Activity 3: Counter (Keeping Score)

1. Click on the BASIC menu and drag the [SHOW NUMBER] block to the programming area under the [FOREVER] block.

2. Click on the VARIABLE menu and click on the [MAKE A VARIABLE] button. Give it a name [COUNTER]

3. Click on the VARIABLE menu and drag the variable [COUNTER] block to the programming area inside the [SHOW NUMBER] block

4. Click on the INPUT menu and drag the [ON BUTTON A PRESSED] block to the programming area

5. Click on the INPUT menu and drag the [ON BUTTON B PRESSED] block to the programming area

6. Click on the INPUT menu and drag the [ON BUTTON A+B PRESSED] block to the programming area

7. Click on the VARIABLE menu and drag the [CHANGE BY 1] block to the programming area inside the [ON BUTTON A PRESSED] block

8. Click on the VARIABLE menu and drag the [CHANGE BY -1] block to the programming area inside the [ON BUTTON B PRESSED] block

9. Click on the VARIABLE menu and drag the [SET TO 0] block to the programming area inside the [ON BUTTON A+B PRESSED] block
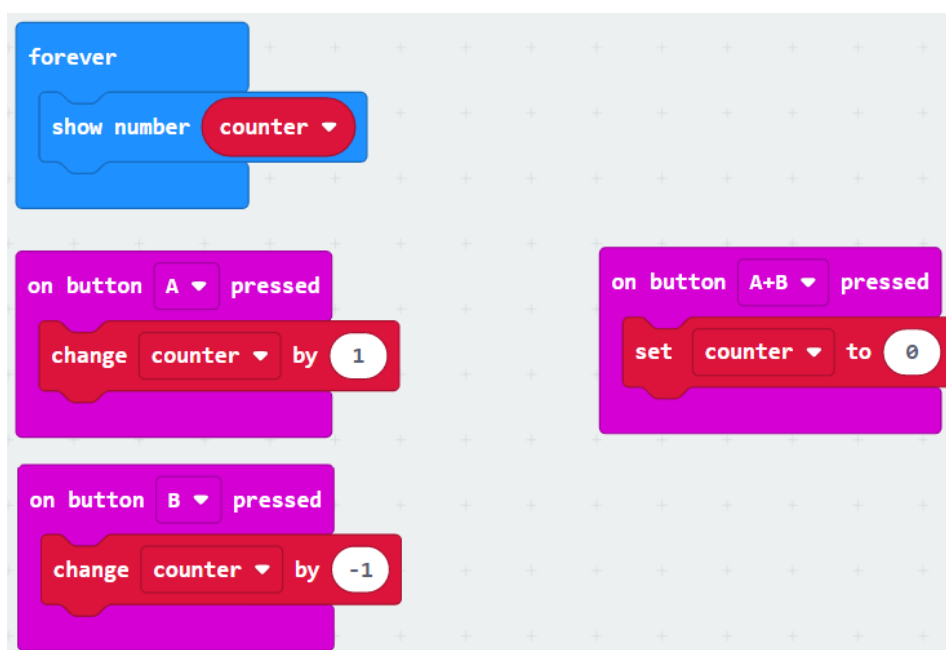
1.5.1 What Is an Accelerometer?

An accelerometer is a device used to measure force and acceleration. It is useful for sensing vibrations and orientation. The accelerometer reports values that describe the changes in acceleration along the 3 axes of the coordinate system (X, Y & Z axis).
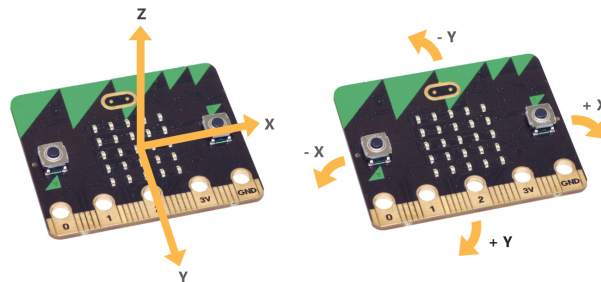
1.5.2 Tilting - Measuring Pitch & Roll

The accelerometer can measure the pitch and roll of the micro:bit, rotation along the x-axis or y-axis, in degrees

This program will show a smiley when the micro:bit is level.
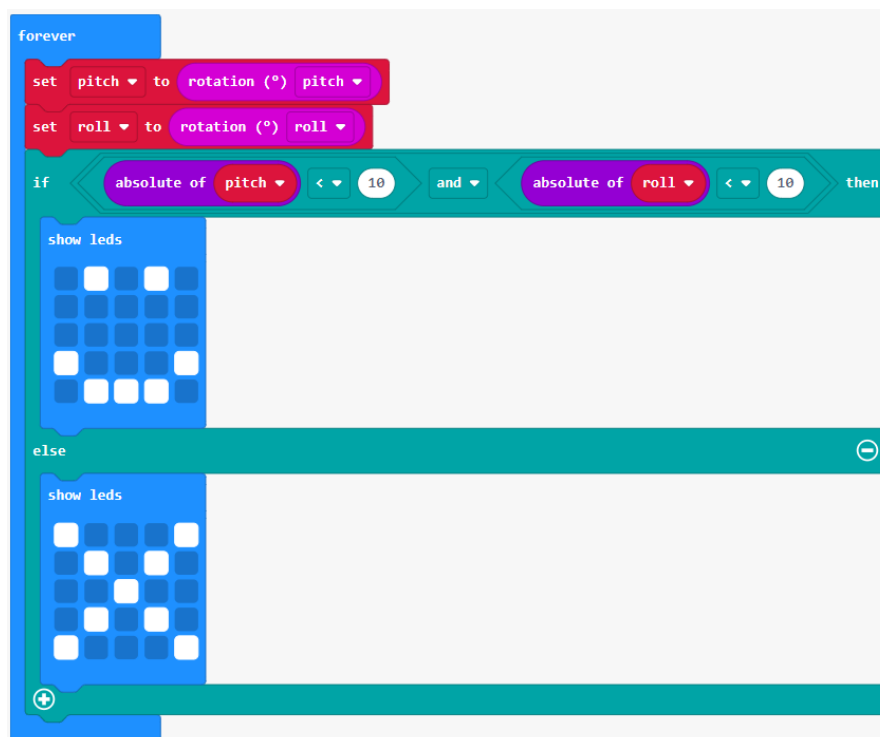
1.5.3 Shaking

The gesture event can also detect the way you hold or move the micro:bit. One of the ways is shaking. This program will show a number from 2 to 9 when you shake the micro:bit.
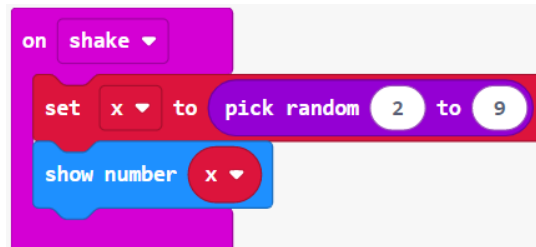
1.5.4 Activity 4: Step Tracker

1. Click on the VARIABLE menu and drag the [SET TO 0] block to the programming area under the [ON START] block.
2. Click on the BASIC menu and drag the [SHOW NUMBER] block to the programming area under the [FOREVER] block.
3. Click on the VARIABLE menu and drag the variable [COUNTER] block to the programming area inside the [SHOW NUMBER] block
4. Click on the INPUT menu and drag the [ON SHAKE] block to the programming area
5. Click on the VARIABLE menu and drag the [CHANGE BY 1] block to the programming area inside the [ON SHAKE] block
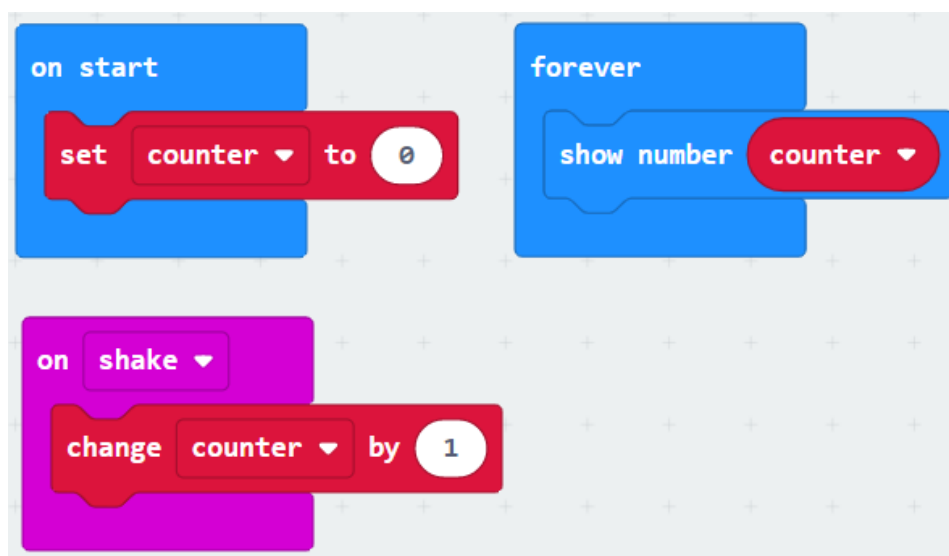
2.1.1 Programming - External Sensors (Digital/Analog Read)

To be able to read in values from a connected external sensor, we need to use the Digital Read or Analog Read blocks. Digital Read will read a digital (0 or 1) signal from any of the pins P0 to P16 on the micro:bit board.
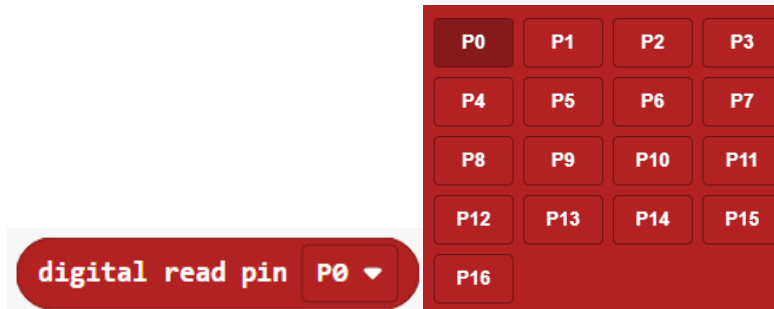
Analog Read will read an analog signal (0 through 1023) from any of the pins P0 to P4 and P10 on the micro:bit board.

2.1.2 Connecting External Sensors to the Breakout Board

The breakout board allows you to utilize all the pins on the micro:bit and opens up some previously inaccessible communication ports, like I2C and SPI.
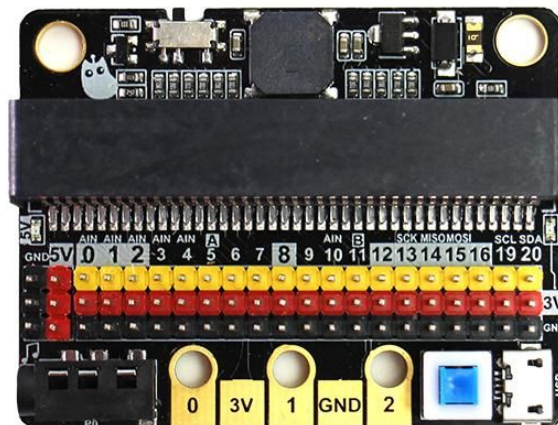
To connect external sensor modules to the breakout board, we match the pins on the external modules to the pins on the breakout board.

The 3 coloured pins are:

1. Yellow is the signal pin that transfers data between the micro:bit and the external sensor

2. Red is the voltage pin that is used to deliver electricity to the external sensor

3. Black is the ground pin that is used to complete the circuit between the micro:bit and the external sensor
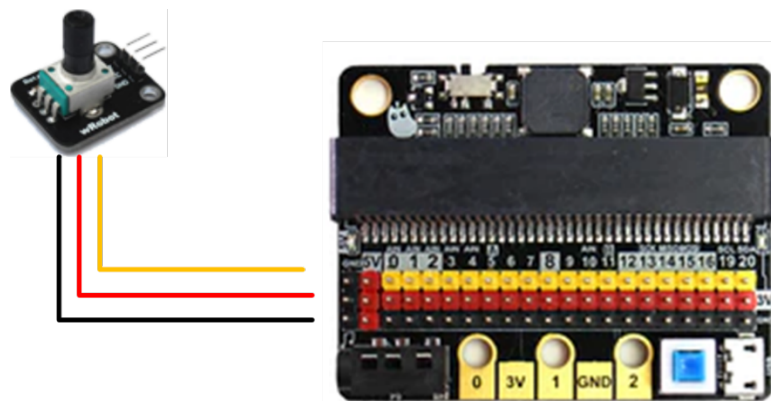
2.1.3 Analog Rotation Sensor and How it Works

The Analog Rotation Sensor is an analog potentiometer mounted onto a handy module. It is based on multi-turn precision potentiometer. It can measure the amount of rotation on the potentiometer.

1. Click on the VARIABLE menu and click on the [MAKE A VARIABLE] button. Give it a name [VALUE]

2. Click on the VARIABLE menu and drag the [SET TO] block to the programming area under the [FOREVER] block

3. Click on the ADVANCED > PINS menu and drag the [ANALOG READ PIN] block to the programming area inside the [SET TO] block

4. Click on the LED menu and drag the [PLOT BAR GRAPH] block to the programming area under the [SET TO] block

5. Set the [UP TO] value of the [PLOT BAR GRAPH] block to 1023

6. Click on the VARIABLE menu and drag the variable [VALUE] block to the programming area inside the [PLOT BAR GRAPH] block

7. Click on the INPUT menu and drag the [ON SHAKE] block to the programming area

8. Click on the VARIABLE menu and drag the [CHANGE BY 1] block to the programming area inside the [ON SHAKE] block
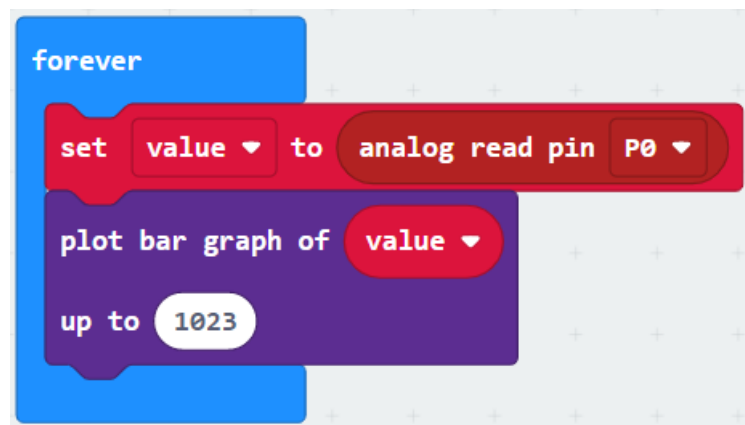
2.1.4 Activity 5: Tuning Music

1. Click on the VARIABLE menu and click on the [MAKE A VARIABLE] button. Give it a name [VALUE]

2. Click on the VARIABLE menu and drag the [SET TO] block to the programming area under the [FOREVER] block

3. Click on the ADVANCED > PINS menu and drag the [ANALOG READ PIN] block to the programming area inside the [SET TO] block

4. Change the analog read pin to P1

5. Click on the LOGIC menu and drag the [IF ELSE] block to the programming area under the [SET TO] block

6. Click on the [+] button at the bottom of the [IF ELSE] block to get 3 more [ELSE IF] extensions

7. Click on the [-] button at the bottom to remove the [ELSE] block

8. Click on the LOGIC menu and drag the [=] comparison block to the programming area inside all the comparison area of the [IF ELSE] block

9. Click on the VARIALBE menu and drag the [VALUE] variable to the programming area inside all the [=] comparison blocks.

10. Set the right comparison values of the [=] comparison blocks to 256, 512, 768 and 1023 respectively

11. Click on the MUSIC menu and drag the[REST] & [PLAY TONE] blocks to the programming area inside the [IF ELSE] block

12. Set the [TONE] of the [PLAY TONE] blocks to "Middle C", "Middle D", "Middle E" and "Middle F" respectively

2.2.1 Programming - Radio Control

The Radio function allows the micro:bit to send and receive data between micro:bits using radio packets. Radio group will need to be set up before the micro:bits can start to send and receive data. A group is like a channel (a micro:bit can only send or receive in one group at a time). A group ID is like the channel number. The group ID can be between 0 to 255.

To send and receive data, the following pair of blocks will have to be used together.

Radio data sent through this block	Can only be received using this block
radio send number 0	on radio received receivedNumber
radio send value "name" = 0	on radio received name value
radio send string " "	on radio received receivedString

2.2.2 Activity 6: Radio Doorbell

1. Click on the RADIO menu and drag the [RADIO SET GROUP] block to the programming area under the [ON START] block
2. Set the [GROUP] value to 1
3. Click on the INPUT menu and drag the [ON BUTTON A PRESSED] block to the programming area

4. Click on the RADIO menu and drag the [RADIO SEND NUMBER] block to the programming area under the [ON BUTTON A PRESSED] block

5. Set the [NUMBER] value of the [RADIO SEND NUMBER] block to 1

6. Click on the INPUT menu and drag the [ON BUTTON B PRESSED] block to the programming area

7. Click on the RADIO menu and drag the [RADIO SEND NUMBER] block to the programming area under the [ON BUTTON B PRESSED] block

8. Set the [NUMBER] value of the [RADIO SEND NUMBER] block to 2

9. Click on the RADIO menu and drag the [ON RADIO RECEIVED NUMBER] block to the programming area

10. Click on the LOGIC menu and drag the [IF ELSE] block to the programming area under the [ON RADIO RECEIVED NUMBER] block

11. Click on the [+] button at the bottom of the [IF ELSE] block to get 1 more [ELSE IF] extension

12. Click on the LOGIC menu and drag the [=] comparison block to the programming area inside all the comparison area of the [IF ELSE] block

13. Click on the [RECEIVEDNUMBER] variable on the [ON RADIO RECEIVED NUMBER] block to the programming area inside all the [=] comparison blocks.

14. Set the right comparison values of the [=] comparison blocks to 1 and 2 respectively

15. Click on the MUSIC menu and drag the[REST] & [START MELODY] blocks to the programming area inside the [IF ELSE] block

16. Set the [MELODY] of the [START MELODY] blocks to "ba ding" and "power up" respectively
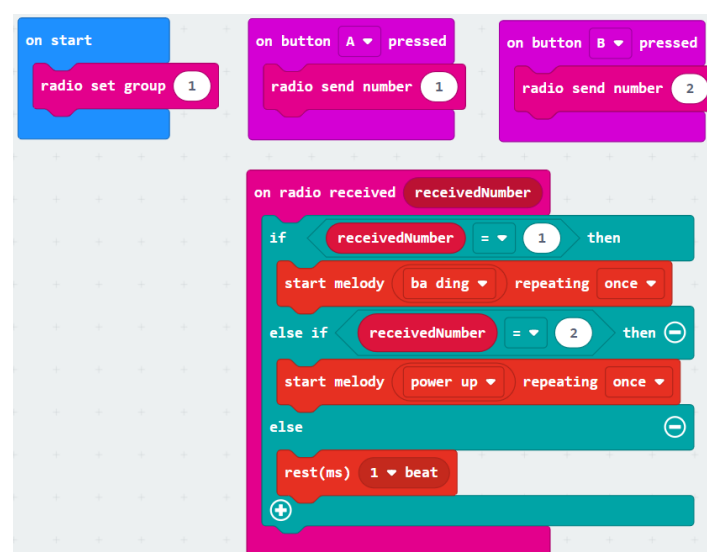
2.3.1 What is an Ultrasonic Sensor and how it works

An ultrasonic sensor is an output device that uses sound to accurately measure distance. It is often used to detect objects.

Connection to the breakout board
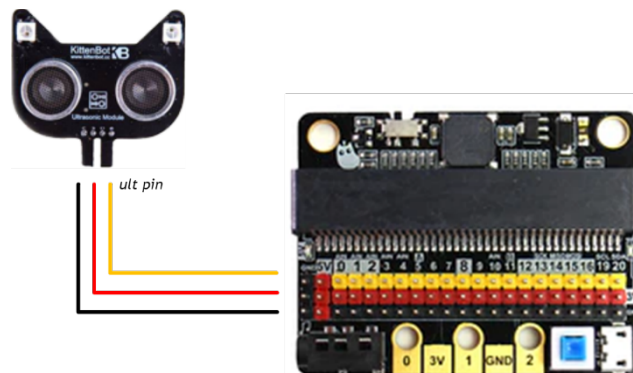
Adding the ultrasonic sensor extension

1. Click on the gear icon on the top right corner of the program menu.
2. Click on the [EXTENSIONS] link
3. The default list of extensions will be displayed
4. Click on the [ROBOTBIT] extension
5. MakeCode will download the [ROBOTBIT] extension
6. When done, the [ROBOTBIT] extension will be found on the code menu
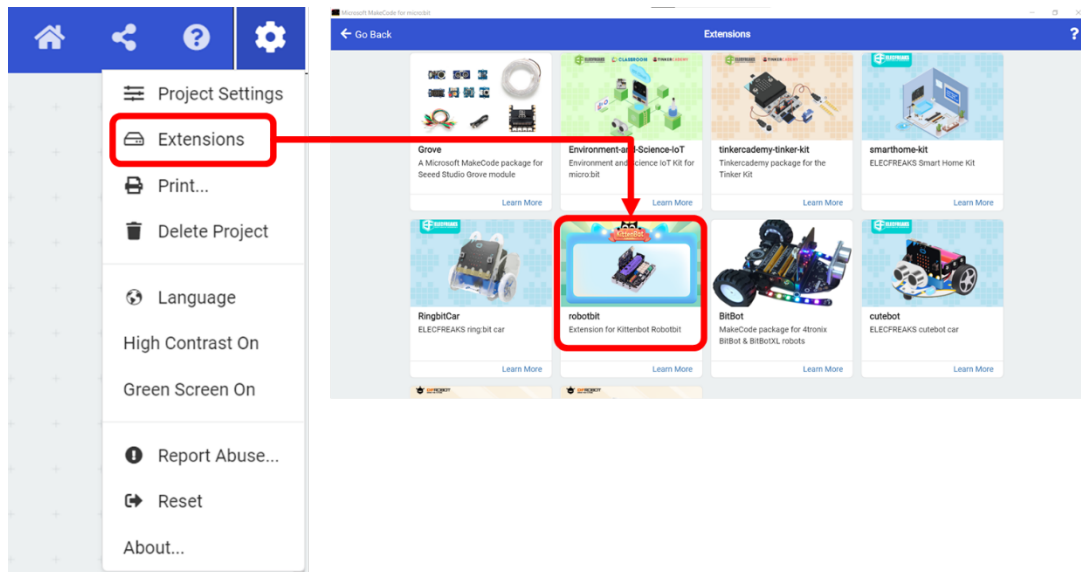
Sample code

Use the [PLOT BAR GRAPH] block to visualize the distance reported by your sensor.
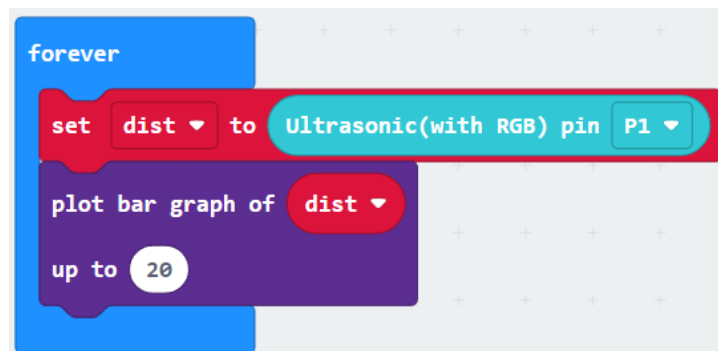
2.3.2 Understanding the values of the Ultrasonic Sensor

The values returned by the ultrasonic sensor is the distance to an obstacle detected by the ultrasonic sensor. The unit of measurement is in "cm"

2.3.3 Activity 7: Invisible Guitar

1. Click on the VARIABLE menu and click on the [MAKE A VARIABLE] button. Give it a name [DIST]

2. Click on the VARIABLE menu and drag the [SET TO] block to the programming area under the [FOREVER] block

3. Click on the ROBOTBIT menu and drag the [ULTRASONIC] block to the programming area inside the [SET TO] block

4. Click on the LOGIC menu and drag the [IF ELSE] block to the programming area under the [SET TO] block

5. Click on the [+] button at the bottom of the [IF ELSE] block to get 3 more [ELSE IF] extension

6. Click on the LOGIC menu and drag the [<] comparison block to the programming area inside all the comparison area of the [IF ELSE] block

7. Click on the VARIABLE menu and drag the [DIST] variable to the programming area inside all the [<] comparison blocks.

8. Set the right comparison values of the [<] comparison blocks to 5, 10, 15 and 20 respectively

9. Click on the MUSIC menu and drag the[REST] & [PLAY TONE] blocks to the programming area inside the [IF ELSE] block

10. Set the [TONE] of the [PLAY TONE] blocks to "Middle C", "Middle D", "Middle E" and "Middle F" respectively

```
forever
  set dist ▼ to Ultrasonic(with RGB) pin P1 ▼
  if      dist ▼ < ▼ 5      then
    play tone Middle C for 1 ▼ beat
  else if  dist ▼ < ▼ 10    then ⊖
    play tone Middle D for 1 ▼ beat
  else if  dist ▼ < ▼ 15    then ⊖
    play tone Middle E for 1 ▼ beat
  else if  dist ▼ < ▼ 20    then ⊖
    play tone Middle F for 1 ▼ beat
  else                           ⊖
    rest(ms) 1 ▼ beat
  ⊕
```

3.1.1 What are NeoPixel LEDs and how they work

NeoPixel LEDs are individually addressable LEDs all housed on a string that can be controlled from a single pin on the micro:bit. This means that one pin can control all of the LEDs colours and which LEDs are on at any given time.
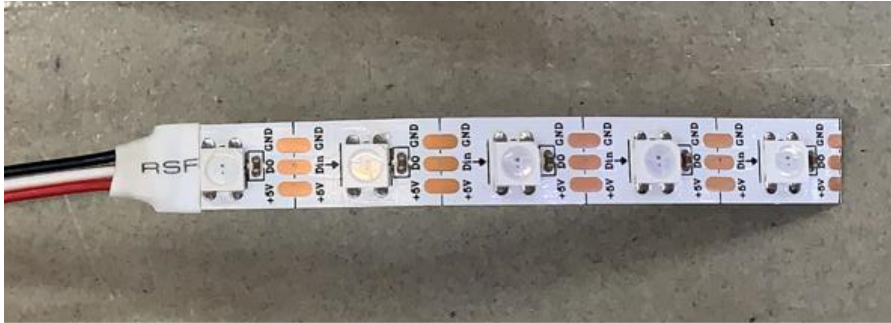
Connection to the breakout board
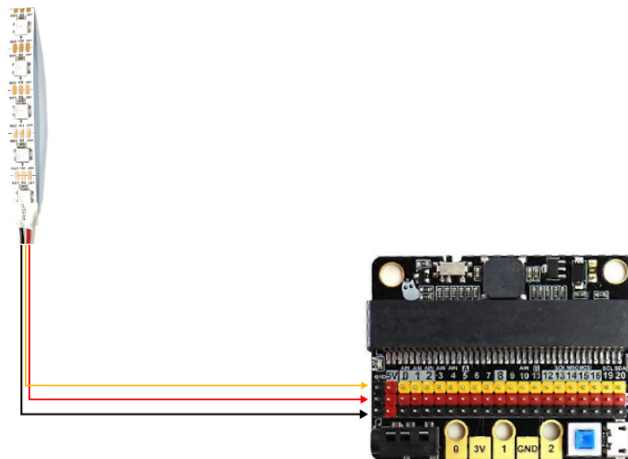
Adding the NeoPixel extension

1. Click on the gear icon on the top right corner of the program menu.
2. Click on the [EXTENSIONS] link
3. The default list of extensions will be displayed
4. Click on the [NEOPIXEL] extension
5. MakeCode will download the [NEOPIXEL] extension
6. When done, the [NEOPIXEL] extension will be found on the code menu
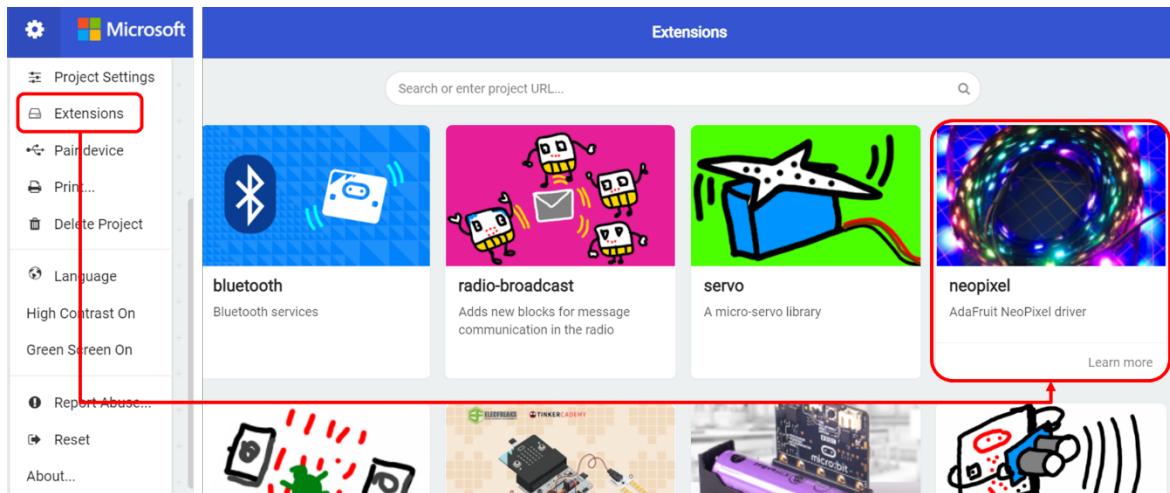
Sample code

Use the [SHOW COLOR] blocks to control the NeoPixel LEDs.
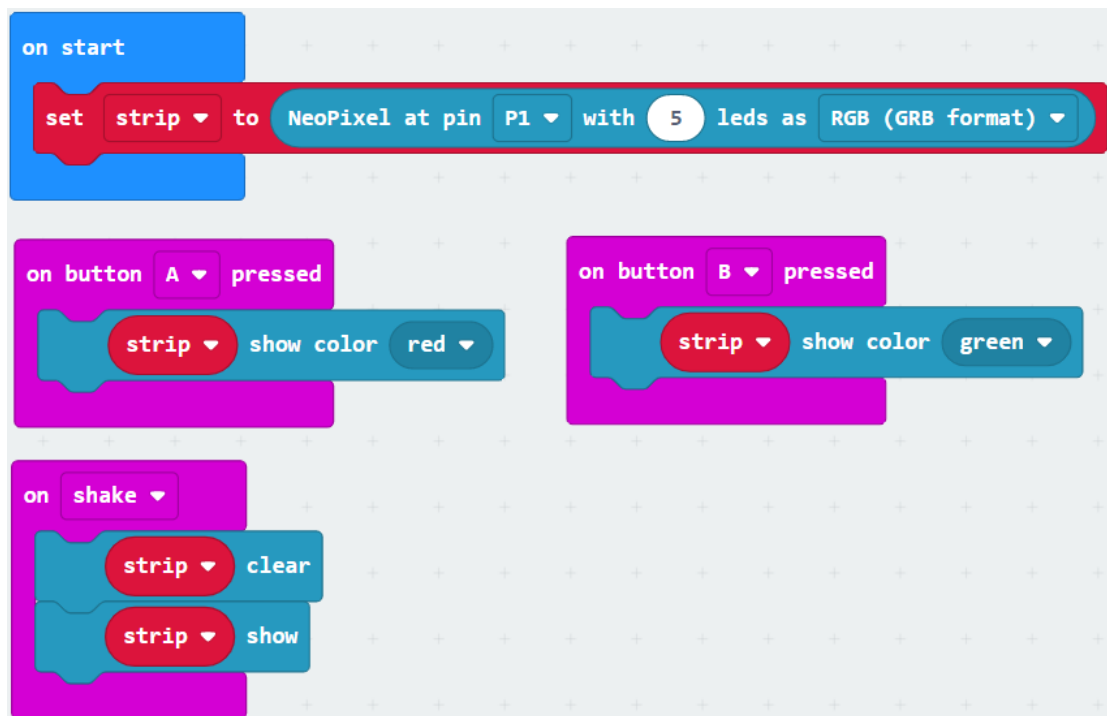
3.1.2 Activity 8: Clap-O-Meter

1. Click on the VARIABLE menu and click on the [MAKE A VARIABLE] button. Give it a name [COUNTER]

2. Click on the VARIABLE menu and click on the [MAKE A VARIABLE] button. Give it a name [LED]

3. Click on the NEOPIXEL menu and drag the [SET STRIP TO] block to the programming area under the [ON START] block
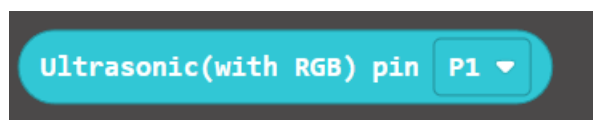
4. Click on the VARIABLE menu and drag the [SET COUNTER TO] block to the programming area under [ON START] under the [SET STRIP TO] block

5. Set the [SET COUNTER TO] value to 0

6. Click on the LOGIC menu and drag the [IF ELSE] block to the programming area under the [FOREVER] block

7. Click on the LOGIC menu and drag the [>] comparison block to the programming area inside the [IF ELSE] block

8. Click on the INPUT menu and drag the [SOUND LEVEL] variable to the programming area inside the [>] comparison blocks

9. Set the right comparison values of the [>] comparison blocks to 100

10. Click on the VARIABLE menu and drag the [CHANGE COUNTER BY] block to the [IF] area under [IF ELSE] block

11. Set the [CHANGE COUNTER BY] value to 1

12. Click on the VARIABLE menu and drag the [CHANGE COUNTER BY] block to the [ELSE] area under [IF ELSE] block

13. Set the [CHANGE COUNTER BY] value to -1

14. Click on the LOGIC menu and drag the [IF] block to the [ELSE] area under [IF ELSE] block

15. Click on the LOGIC menu and drag the [<] comparison block to the programming area inside the [IF] block

16. Click on the VARIABLE menu and drag the [COUNTER] variable to the programming area inside the [<] comparison blocks

17. Click on the VARIABLE menu and drag the [SET COUNTER TO] block to the [IF] area under the [IF] block

18. Click on the VARIABLE menu and drag the [SET LED TO] block to the programming area under the [IF ELSE] block

19. Click on the MATH menu and drag the [TRUNCATE] block to the programming area inside the [SET LED TO] block

20. Click on the MATH menu and drag the [÷] block to the programming area inside the [TRUNCATE] block

21. Click on the VARIABLE menu and drag the [COUNTER] variable to the programming area inside the [÷] comparison block

22. Set the right comparison values of the [÷] comparison block to 10

23. Click on the BASIC menu and drag the [PAUSE (MS)] block to the programming area under the [SET LED TO] block

24. Set the [MS] value to 100ms

25. Click on the BASIC menu and drag another [FOREVER] block to the programming area

26. Click on the NEOPIXEL menu and drag the [STRIP SET PIXEL COLOR] block to the programming area under the new [FOREVER] block

27. Click on the VARIABLE menu and drag the [LED] variable to the programming area inside the [STRIP SET PIXEL COLOR] block

28. Click on the NEOPIXEL menu and drag the [STRIP SHOW] block to the programming area under the [STRIP SET PIXEL COLOR] block
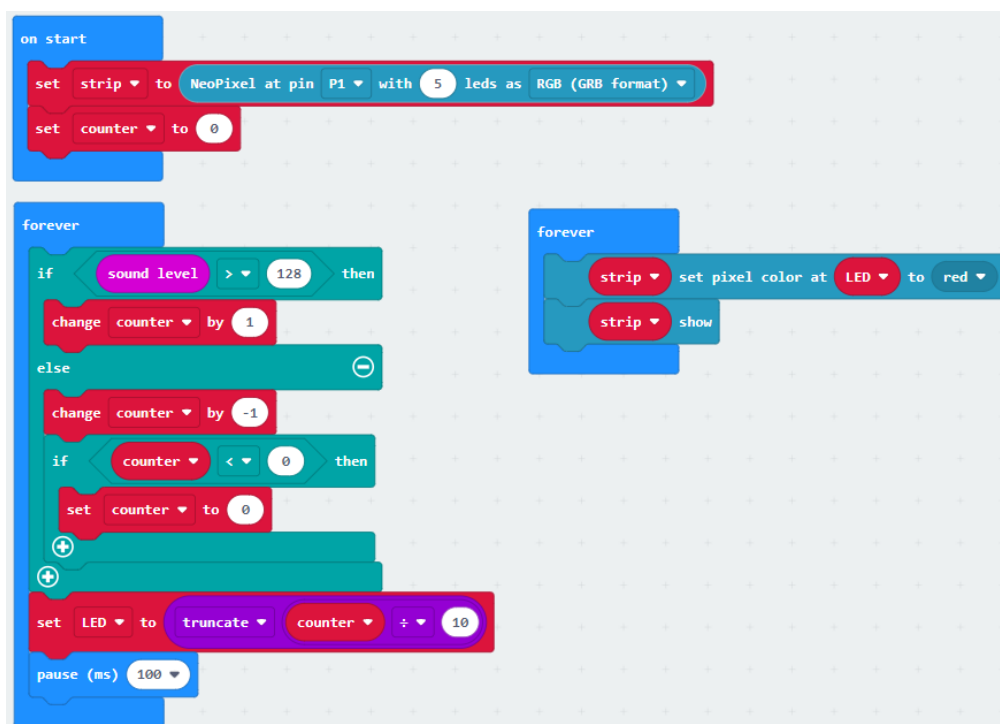
3.2.1 What is a Servo Motor and how it work

A servo motor is an output device with integrated gears and a shaft that can be precisely controlled. Standard servos allow the shaft to be positioned at various angles, usually between 0 and 180 degrees.

Typical uses of servo motors include operating remote-controlled or radio-controlled toy cars, robots and airplanes. Servo motors are also used in industrial applications, robotics, in-line manufacturing, pharmaceutics and food services.

Connection to the breakout board
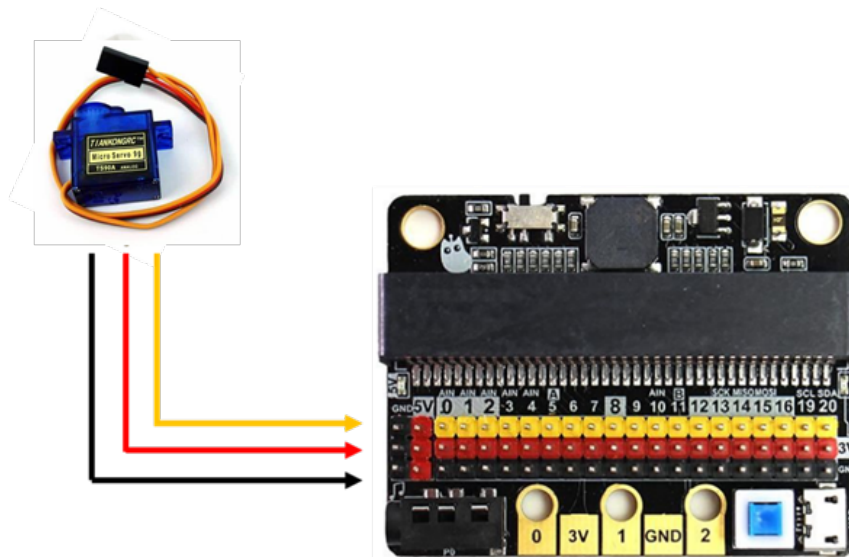
**Adding the Servo extension**

1. Click on the gear icon on the top right corner of the program menu.
2. Click on the [EXTENSIONS] link

3. The default list of extensions will be displayed

4. Click on the [SERVO] extension

5. MakeCode will download the [SERVO] extension

6. When done, the [SERVO] extension will be found on the code menu
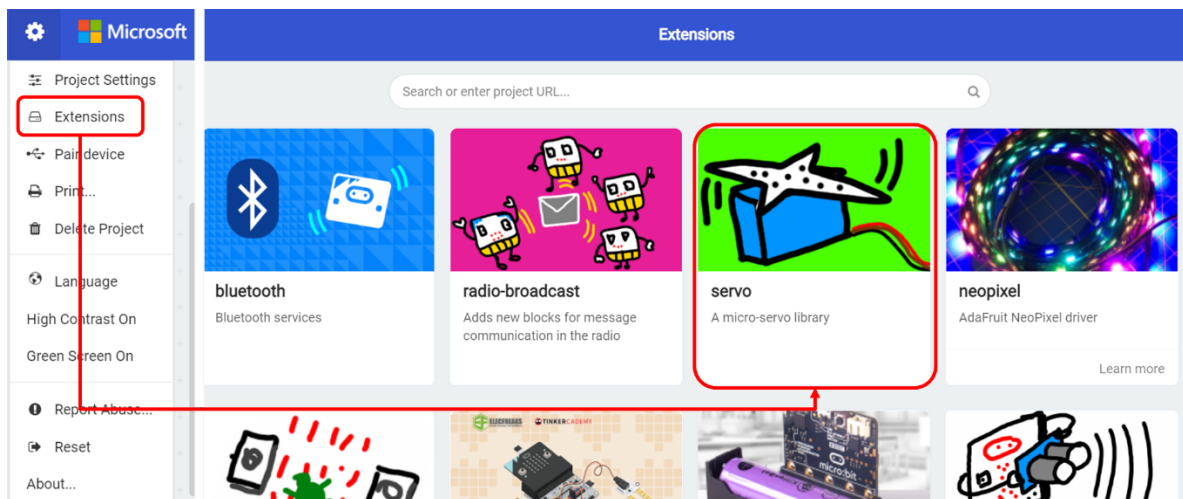
Sample code

Use the [SET SERVO] blocks to control the Servo Motors.
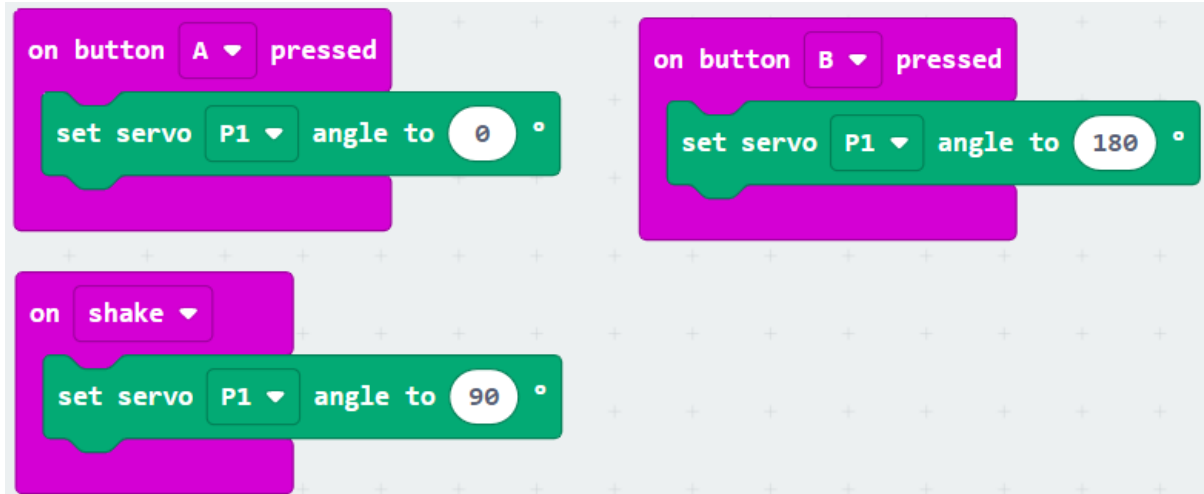
3.2.2 Activity 9: Windscreen Wiper

1. Click on the VARIABLE menu and click on the [MAKE A VARIABLE] button. Give it a name [STARTSTOP]

2. Click on the VARIABLE menu and click on the [MAKE A VARIABLE] button. Give it a name [SPEED]

3. Click on the VARIABLE menu and click on the [MAKE A VARIABLE] button. Give it a name [ANGLE]

4. Click on the INPUT menu and drag the [ON BUTTON A PRESSED] block to the programming area

5. Click on the VARIABLE menu and click on the [SET STARTSTOP TO] block to the programming area under [ON BUTTON A PRESSED] block

6. Set the [SET STARTSTOP TO] value to 1

7. Click on the INPUT menu and drag the [ON BUTTON B PRESSED] block to the programming area

8. Click on the VARIABLE menu and click on the [SET STARTSTOP TO] block to the programming area under [ON BUTTON B PRESSED] block

9. Set the [SET STARTSTOP TO] value to 0

10. Click on the INPUT menu and drag the [ON SHAKE] block to the programming area

11. Click on the VARIABLE menu and click on the [CHANGE SPEED BY] block to the programming area under [ON SHAKE] block

12. Set the [CHANGE SPEED BY] value to 1

13. Click on the LOGIC menu and drag the [IF] block to the programming area under [ON SHAKE] block

14. Click on the LOGIC menu and drag the [>] comparison block to the programming area inside the [IF] block

15. Click on the VARIABLE menu and drag the [SPEED] variable to the programming area inside the [>] comparison blocks

16. Set the right comparison values of the [>] comparison blocks to 3

17. Click on the VARIABLE menu and click on the [SET SPEED TO] block to the programming area under the [IF] block

18. Set the [SET SPEED TO] value to 1

19. Click on the LOGIC menu and drag the [IF ELSE] block to the programming area under the [FOREVER] block

20. Click on the LOGIC menu and drag the [=] comparison block to the programming area inside the [IF ELSE] block

21. Click on the VARIABLE menu and drag the [STARTSTOP] variable to the programming area inside the [=] comparison blocks

22. Set the right comparison values of the [=] comparison blocks to 1

23. Click on the LOGIC menu and drag the [IF ELSE] block to the programming area under the [IF] area under the [IF ELSE] block

24. Click on the [+] button at the bottom of the [IF ELSE] block to get 1 more [ELSE IF] extension

25. Click on the LOGIC menu and drag the [=] comparison block to the programming area inside all the comparison area of the [IF ELSE] block

26. Click on the VARIALBE menu and drag the [SPEED] variable to the programming area inside all the [=] comparison blocks.

27. Set the right comparison values of the [=] comparison blocks to 1 and 2 respectively

28. Click on the SERVO menu and drag 2 [SET SERVO] blocks to the programming area inside [IF] area of the [IF ELSE] block

29. Set the [ANGLE] of the [SET SERVO] blocks to 180 and 0 respectively

30. Click on the BASIC menu and drag 2 [PAUSE (MS)] block to the programming area under each [SET SERVO] block

31. Set the [MS] value to 2000ms

32. Click on the SERVO menu and drag 2 [SET SERVO] blocks to the programming area inside [ELSE IF] area of the [IF ELSE] block

33. Set the [ANGLE] of the [SET SERVO] blocks to 180 and 0 respectively

34. Click on the BASIC menu and drag 2 [PAUSE (MS)] block to the programming area under each [SET SERVO] block

35. Set the [MS] value to 1000ms

36. Click on the SERVO menu and drag 2 [SET SERVO] blocks to the programming area inside [ELSE] area of the [IF ELSE] block

37. Set the [ANGLE] of the [SET SERVO] blocks to 180 and 0 respectively

38. Click on the BASIC menu and drag 2 [PAUSE (MS)] block to the programming area under each [SET SERVO] block

39. Set the [MS] value to 500ms

40. Click on the SERVO menu and drag the [SET SERVO] block to the programming area inside [ELSE] area of the [IF ELSE] block

41. Set the [ANGLE] of the [SET SERVO] block to 0

```
on start
  set StartStop ▼ to 0
  set speed ▼ to 1
  set angle ▼ to 0

on button A ▼ pressed
  set StartStop ▼ to 1

on button B ▼ pressed
  set StartStop ▼ to 0

on shake ▼
  change speed ▼ by 1
  if speed ▼ > ▼ 3 then
    set speed ▼ to 1

forever
  if StartStop ▼ = ▼ 1 then
    if speed ▼ = ▼ 1 then
      set servo P1 ▼ angle to 180 °
      pause (ms) 2000 ▼
      set servo P1 ▼ angle to 0 °
      pause (ms) 2000 ▼
    else if speed ▼ = ▼ 2 then
      set servo P1 ▼ angle to 180 °
      pause (ms) 1000 ▼
      set servo P1 ▼ angle to 0 °
      pause (ms) 1000 ▼
    else
      set servo P1 ▼ angle to 180 °
      pause (ms) 500 ▼
      set servo P1 ▼ angle to 0 °
      pause (ms) 500 ▼
  else
    set servo P1 ▼ angle to 0 °
```

3.3.1 Introduction to Design Thinking

Design thinking is an approach towards solving real world design problems by understanding users' needs and developing key insights to solve those needs.

Problem Scoping

Problem scoping involves the critical process of identifying a good design problem. Identification can be through interviewing and asking questions and brainstorming and coming out with multiple issues and problem areas.

Concept Generation

Concept generation is the creative process of generating, developing, and communicating new ideas, where an idea is understood as a basic element of thought that can be either visual, concrete, or abstract. Concept generation comprises all stages of a thought cycle, from innovation, to development, to actualization.

Concept Selection

Concept selection arrives at the winning solution by comparing the relative strengths and weaknesses of the concepts generated.

Prototyping

Prototyping process involves development of an early representation of the final solution.

Testing

Testing is the final stage of design thinking. This is where the prototype solution is tested in real life and in real time by the actual users.
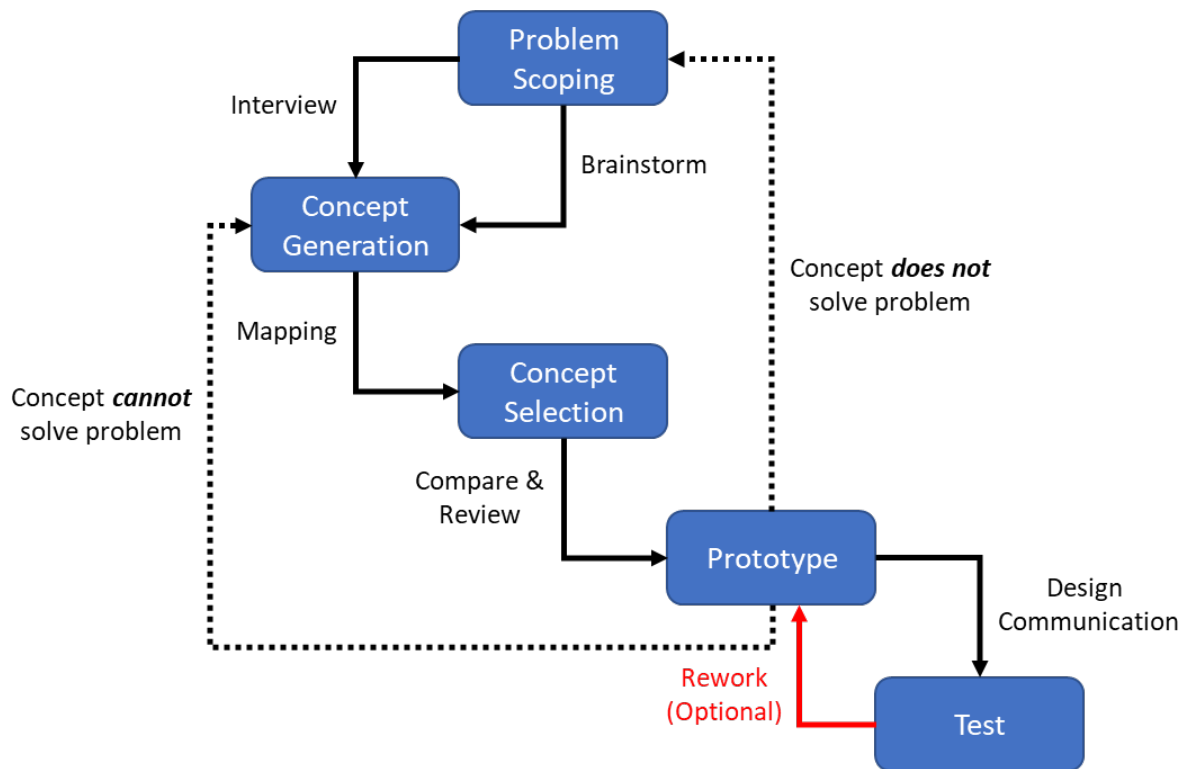
Theme

The theme for this hackathon is all about finding creative and innovative solutions to the problems we might face in the future. As students, you are the future leaders of our world, and we believe that you can come up with brilliant ideas to solve the problems of tomorrow.

We want you to think outside the box and come up with solutions that are sustainable, efficient, and scalable. Some areas of focus could be climate change, healthcare, transportation, education, and more.

During this hackathon, you will have the opportunity to work in teams and collaborate with other students from different schools. You will be given access to mentors who will guide you and help you develop your ideas.

At the end of the hackathon, you will present your solutions to a panel of judges who will evaluate your projects based on innovation, feasibility, and impact.

(https://phss.epc-education.com/hackathon-2023/)

Activity 1: Problem Scoping – Interview

Here are some sample problems that primary school students could potentially solve during the hackathon:

1. Climate change: Develop a solution to reduce carbon emissions in your school or community. This could include ways to conserve energy, reduce waste, or promote sustainable transportation.

2. Healthcare: Create a technology or tool to help people manage their health better. This could be an app that reminds people to take their medication, a device that monitors vital signs, or a platform that connects patients with doctors remotely.

3. Education: Find a way to make learning more engaging and accessible for students. This could include a game that teaches a subject, a platform that connects students with mentors, or a tool that helps students track their progress.

4. Transportation: Create a solution to make transportation more efficient and sustainable. This could include a carpooling app for parents, a platform that encourages walking or biking to school, or a device that tracks carbon emissions from transportation.

5. Community building: Develop a solution to bring people together and promote social cohesion. This could include an app that connects volunteers with local organizations, a platform that helps people organize community events, or a tool that promotes cultural understanding and diversity.

These are just a few sample problems, but there are many other challenges that students can tackle during the hackathon. The key is to identify a problem that resonates with you and your team, and to come up with a creative and innovative solution that can make a real difference in the world.

Ask questions like:

1. Have you heard about people suffering from mental health issues during the pandemic?

2. Who were they referring to?

3. What kind of problems/issues did they face?

4. What could be done to help these people?

5. Are these issues only due to the pandemic or do they also occur without the pandemic?

6. Do you have any personal stories or heard about your friends having mental health issues?

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

Activity 2: Problem Scoping –Brainstorm

Identify issues and problem areas that we see/hear/read about locally that are linked with the problems that occurred due to the mental health issue you selected.

Describe the problem:

1. What is the impact of the pandemic on the mental health issue?

2. How does it affect people/environment/things?

3. Who is affected by it?

4. Why do you think these problem areas are important to address?

5. Where does these problem areas normally occur?
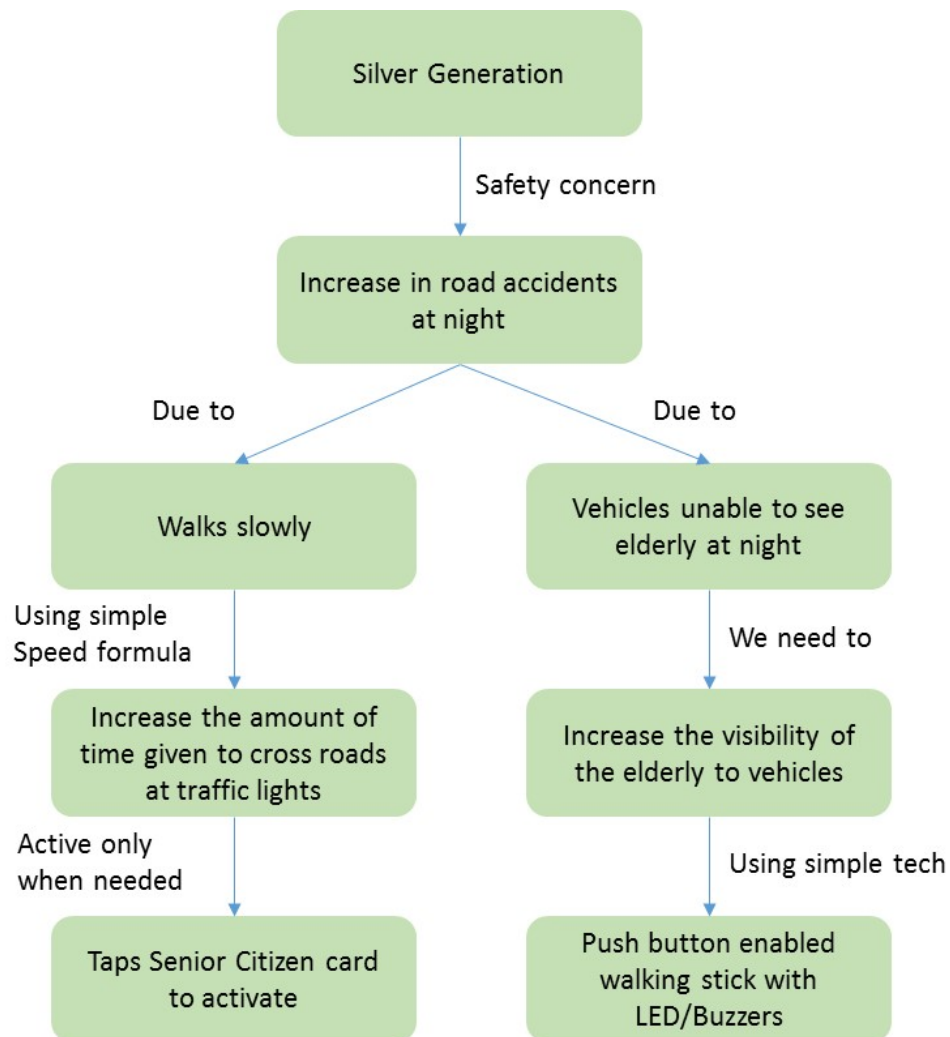
_____
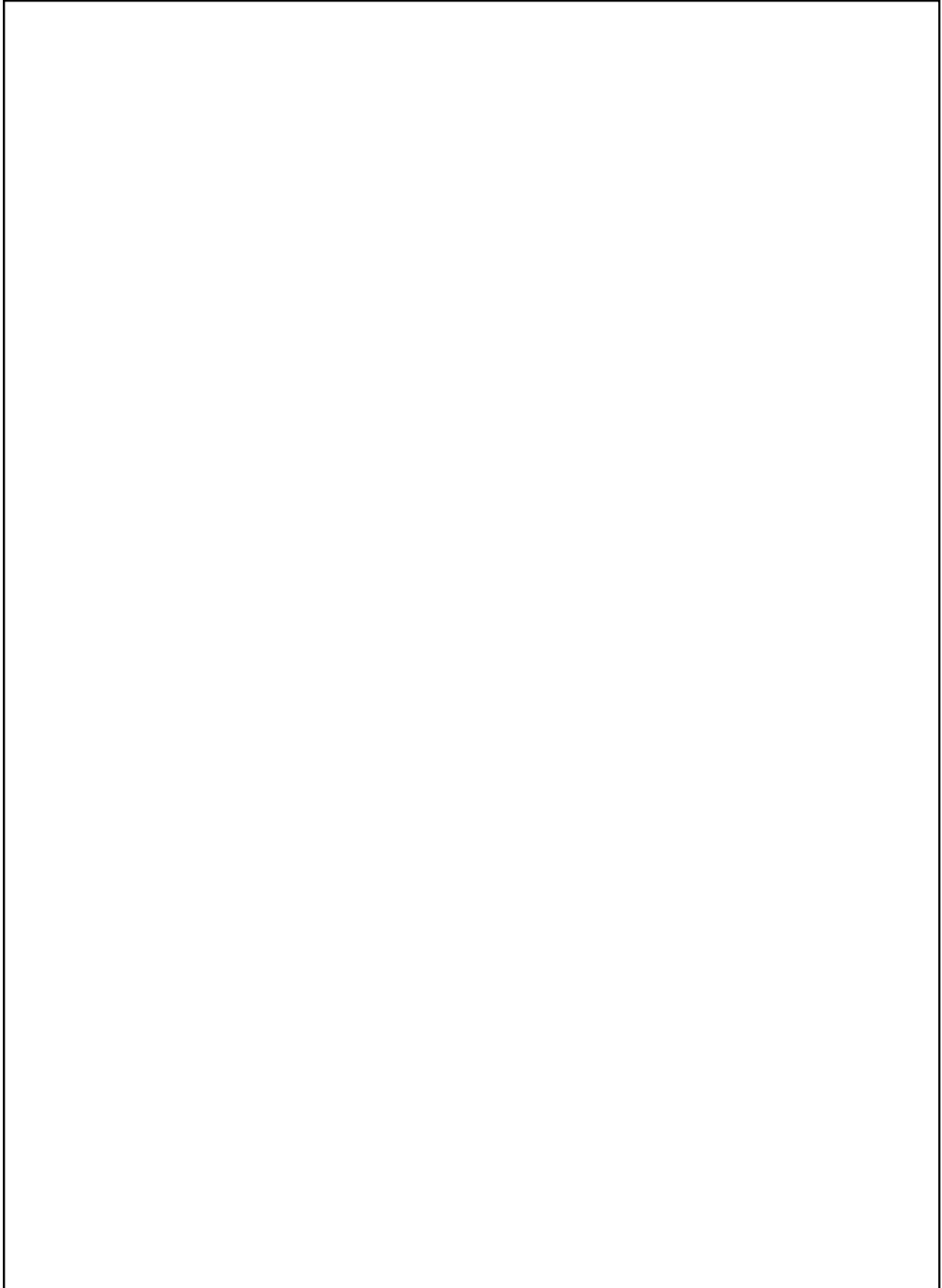
_____

_____

_____

_____

_____

_____

_____

_____

Activity 3: Concept Generation – Mapping

Concept generation maps are graphical tools for organizing and representing knowledge, usually enclosed in circles or boxes of some type, and relationships between concepts indicated by a connecting line linking two concepts.

```
                    ┌─────────────────────┐
                    │  Silver Generation  │
                    └─────────────────────┘
                              │ Safety concern
                              ▼
                    ┌─────────────────────┐
                    │ Increase in road    │
                    │ accidents at night  │
                    └─────────────────────┘
            Due to    ╱                ╲   Due to
                     ▼                  ▼
          ┌──────────────┐      ┌──────────────────┐
          │ Walks slowly │      │ Vehicles unable  │
          │              │      │ to see elderly   │
          │              │      │ at night         │
          └──────────────┘      └──────────────────┘
  Using simple  │                         │  We need to
  Speed formula ▼                         ▼
  ┌────────────────────┐      ┌────────────────────┐
  │ Increase the amount│      │ Increase the       │
  │ of time given to   │      │ visibility of the  │
  │ cross roads at     │      │ elderly to vehicles│
  │ traffic lights     │      │                    │
  └────────────────────┘      └────────────────────┘
  Active only   │                         │  Using simple tech
  when needed   ▼                         ▼
  ┌────────────────────┐      ┌────────────────────┐
  │ Taps Senior Citizen│      │ Push button enabled│
  │ card to activate   │      │ walking stick with │
  │                    │      │ LED/Buzzers        │
  └────────────────────┘      └────────────────────┘
```

Concepts are represented in a hierarchical fashion with the most inclusive, most general concepts at the top of the map and the more specific, less general concepts at the bottom of the map.

Can you generate a concept map to address the issues and problem areas you identified?

Activity 4: Concept Selection –Comparison and Review

Share your concept map and solution with your team. Like the problem scoping phase, list out suggestions and comments from your team about your solution.

1. Take note of likes/dislikes and builds on the idea, but also listen for new insights.

2. Spend the time listening to your teammates' reactions and questions.

3. Consider what you have learned both about your teammates, and about the solutions you generated.

4. Do you need to return to Problem Scoping or Concept Generation Phase?

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

Activity 5: Prototyping – Design Communication

In this phase you will need to create a small but crucial representation of the final solution.

This prototype can be either virtual or physical.

1. What are the characteristics of your solution?

2. How does it work? What are the functional requirements?

3. What does it look like?

4. How will people use it?

5. How must it make the user feel?

_____

_____

_____

_____

_____

_____

_____

_____

_____
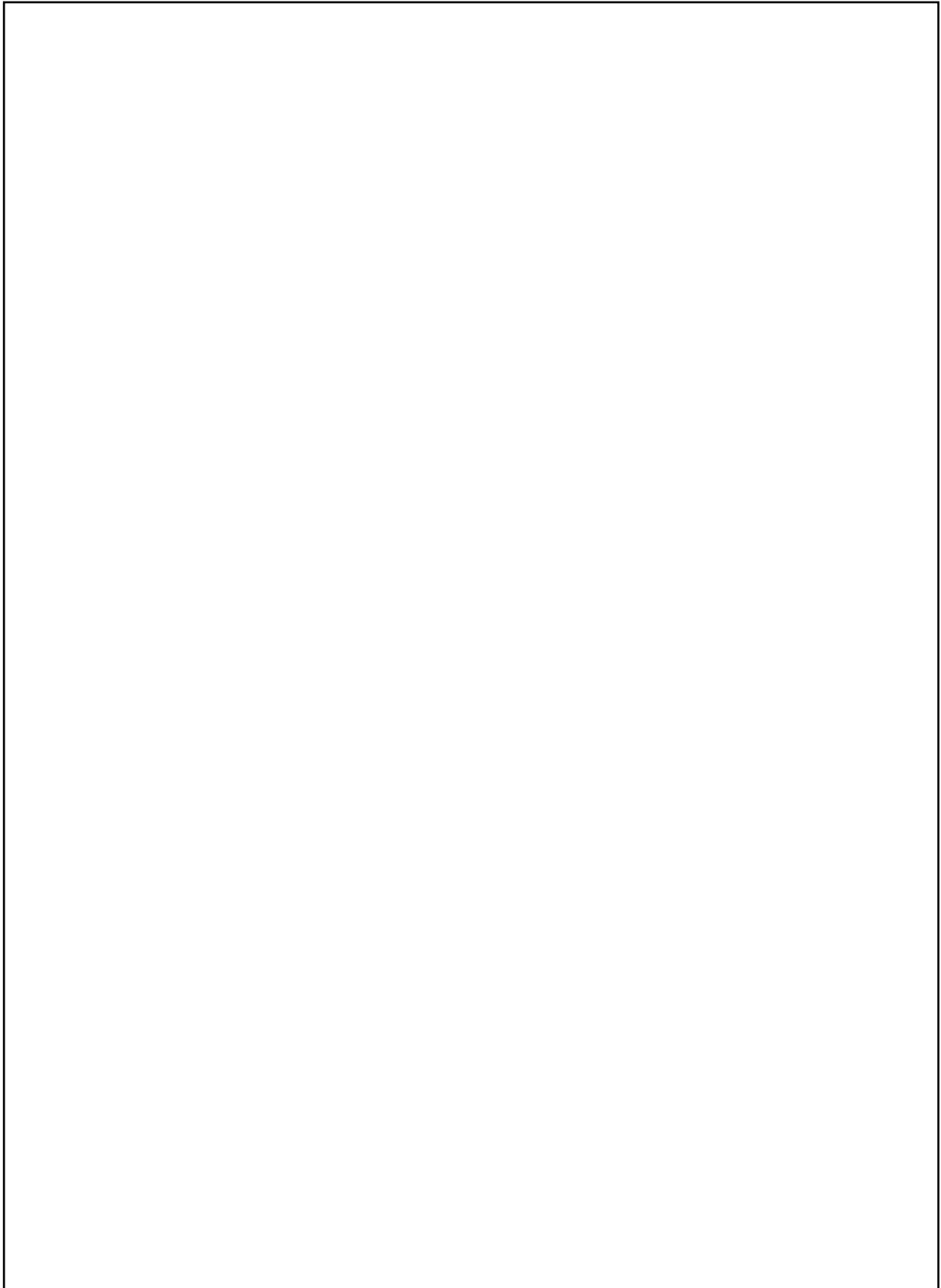
Draw out how you want the prototype to look like and list down what materials/tools/components do you need to complete the prototype.

Activity 6: Testing

In this phase you will need to test the prototype according to the previously listed design communication.

1. Does it display the characteristics of your solution?

2. Does it work according to the functional requirements?

3. Does it look like the way it was designed?

4. Can people use it the way it was designed?

5. How does the user feel after testing?

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

Final Points to Note

Design thinking is a human-centred approach to problem-solving that focuses on understanding people's needs, exploring creative solutions, and testing and iterating on ideas. Here are some of the core values of design thinking that primary school students can consider for this hackathon:

- Empathy: Design thinking starts with empathy, which means understanding the needs, wants, and challenges of the people you are designing for. During the hackathon, students should aim to empathize with the people who will benefit from their solution, whether it's their classmates, their community, or the world at large.

- Creativity: Design thinking encourages students to think creatively and come up with novel solutions to problems. Students should feel free to think outside the box and explore a wide range of possibilities during the hackathon.

- Collaboration: Design thinking is a collaborative process that involves working with others to generate and refine ideas. Students should work closely with their teammates, mentors, and other participants to build on each other's strengths and come up with the best possible solution.

- Experimentation: Design thinking involves prototyping and testing ideas to see what works and what doesn't. Students should be willing to try out different approaches and be open to learning from their mistakes.

- Iteration: Design thinking is an iterative process that involves refining and improving ideas based on feedback. Students should be prepared to iterate on their ideas and make changes based on what they learn from testing and user feedback.

By embracing these core values of design thinking, students can develop innovative and effective solutions to the problems of tomorrow.