Variables

Programming for beginners with the BBC micro:bit

John Davies

Bearsden and Milngavie u3a

2026 May 13

Introduction

Variables are data that change while a program runs. Examples are:

- the number of steps counted by a pedometer
- the number 'thrown' by a dice
- the temperature recorded by a thermometer
- a text message received from another micro:bit by radio

Variables can be:

- **numbers**, the simplest example, which we shall use in this class
- **text strings**, like a message sent from another micro:bit
- **tables**, such as successive readings of the temperature (a data logger)

As an introduction, we will count the number of times that a button is pressed. This clearly needs a variable. The second example will be a digital dice.

Create a variable for the button counter

MakeCode knows what blocks are available but it has no idea what variables we wish to use so we have to create them.

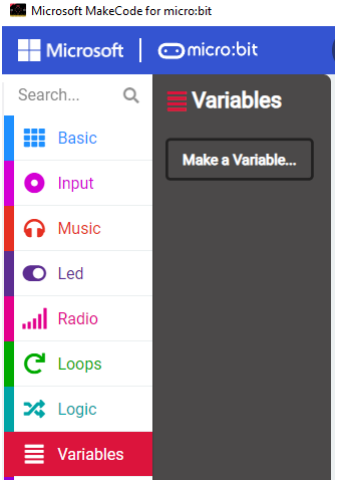
Each variable must have a unique name to distinguish it.

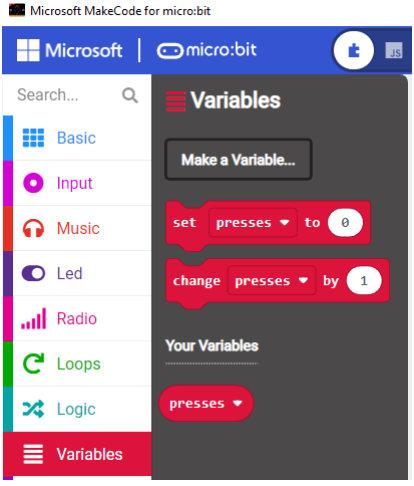
1. Click on the Variables section of the Toolbox. A pane appears with a button Make a Variable. . . .
2. Type the name of the variable into the box that appears. I called it presses. Click OK.
3. The Variables pane now shows the blocks that you can use with your variable.
   - set puts a given value into the variable (zero by default)
   - change adds or subtracts a number to the variable
   - the variable itself can be used as a number in another block

   We shall use all three in our simple program.

The next slide shows screenshots.

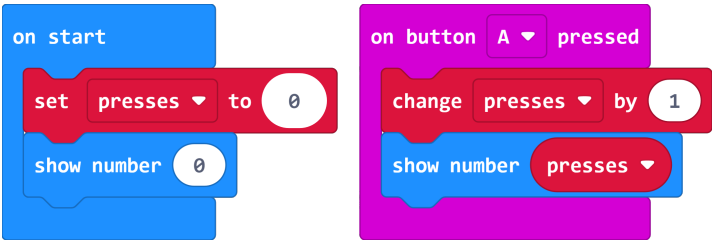
Variables pane before and after creating the ‘presses’ variable





Construct the program

Here is a program to count presses of button A, using two event blocks.



Why is the on start block needed? The answer is really important:

You must never use a variable until it has been given a value.

This **initialization** step is painfully easy to forget. . . . I have also set the display to zero for consistency.

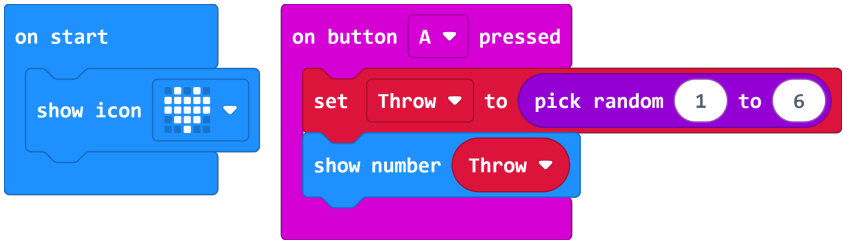
Extend the program

Here are some suggestions for extending the program.

1. Use button B to reset the counter (set `presses` back to zero and don't forget to update the display as well).
2. Turn the button counter into a pedometer. This is not difficult at all! How well does it work?
3. If you are dissatisfied with the basic pedometer, try the micro:bit project Sensitive step counter. This reads the accelerometer directly, which allows you to select how large a signal is required to indicate a new step.

Simple digital dice

A digital dice needs to generate a random number between 1 and 6. Conveniently, MakeCode provides a pick random function to do exactly this. Here is a simple dice.

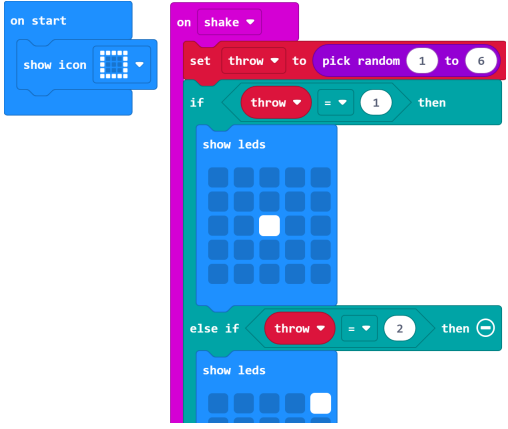


- The pick random function is in the Math section of the toolbox.
- You don't really need the on start block but I like to show something.
- We don't need to initialise Throw in the on start block because it is always set to an allowed value before we use it.

Traditional dice I

Here is the start of something more like a traditional dice, which

- shows a new value when shaken
- displays the traditional patterns of dots



Traditional dice II

An extended if . . . then . . . else if . . . then . . . block selects the pattern for each value. Use ⊕ to extend the block. The if . . . statements must be constructed **from the outside inward**.

1. Each if . . . then starts with an empty lozenge for the condition.

2. Drag a comparison block from the Logic section into the empty lozenge.

3. Drag your variable from the Variables section to replace the left-hand 0. Be careful that it doesn't replace the whole comparison block!

4. Replace the right-hand zero with the appropriate value for the dice.

Extend the program

Here are some suggestions for extending or adapting the program.

1. Add a short interval with a blank screen after the dice is thrown. This avoids confusion if the same number is thrown twice in a row.
2. Perhaps you would prefer to select a random suit of cards for whist. Again you can draw the patterns in show leds blocks (some are in show icon already).
3. Make a 'rock, paper, scissors' game. Use patterns in show icon or draw your own in show leds. Your micro:bit could play against another one if the game is triggered by a clap! The Make it: code it page has an example.
4. A magic 8-ball gives a random answer to a simple question. The example is a good starting point.
5. An activity picker gives a random suggestion from a list, useful when you are bored. This requires a list of strings, which is a new feature for us, so take a look at the example.