

A decorative border of colorful, abstract shapes surrounds the central text. The shapes include various colors like blue, green, yellow, pink, teal, purple, red, and light blue, some with patterns like polka dots or stripes. The shapes are irregular and organic in form, resembling stylized leaves or clouds.

4

Quick-fire Microbit projects

In this booklet

Hello my name is Andy, and in this booklet you will find 4 “quick-fire” coding projects to get you started with the BBC Microbit.

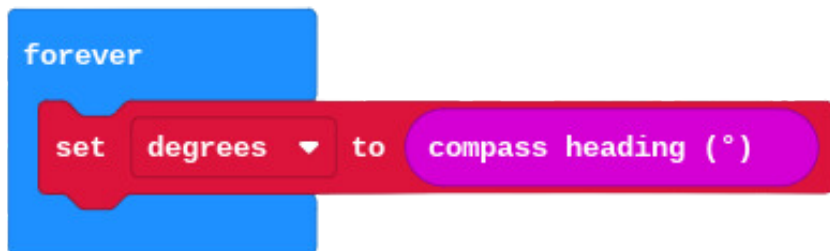
These projects focus on tools you can use your Microbit for and, I may be bias, they will be incredibly entertaining to complete.

Have fun!

1- Compass

This one is slightly misleading, and could get me in trouble for false advertising. I'm going to show you how to get your compass to tell you where north is so as long as you want to travel north this will work for you! I'm of course joking, we're making a full compass.- what a waste of words.

STEP 1: Create a variable called “**degrees**”. And use the forever loop and compass heading (which you'll find in input) together to make the below code.



Now for some logic inclusion to the “compass”... I wish I had some logic... I'd be unstoppable if I had any logic inside of me!

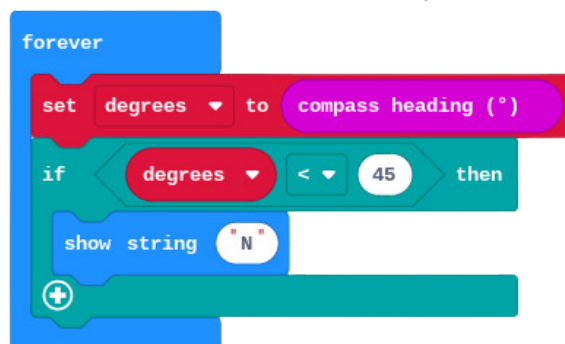
STEP 2: Head across to the **logic** menu and get yourself an **if then** block. I like to think of this block a bit like a door guard who needs a password for you to get in. If you get the right password you get through the door... if you don't have the password the door will never open.

If you don't meet the requirements of the **if then** block... the bit inside will never happen. You know what else will never happen? Me, writing some instructions without waffling...

Anyway, once you have your **if then** block, grab a **less than** block from the logic menu *arrow pointing at the left hand side* and pop it into that nice blank space next to where it says if. (Make sure you have your sound on so you can hear the satisfying clicking sound as the code slots together!)

Next grab a **degrees** from the variables menu and pop it into the blank space on the left of the **less than** block and type in the number **45** on the right hand side.

To finish off step 2... which rhymes with poo... go into **basic** and get **show string**. A **string** in coding terms means **letters**. Add that into the **if then** sandwich and type in **N** (stands for North) into the empty box next to it.



STEP 3: The more observant amongst you will have noticed that + at the bottom of the if block. The good news is, you can press it now... in fact, you can press it twice. After 2 clicks you should have an **else if** section appear.

Press the - button at the bottom to get rid of the bit that just has **else** written on it.

You can then repeat the steps we should have done in **Step 2**. (You know, add the < block and the **degrees variable** to the left side and instead of 45 write the number as **135**. So the block will read if **degrees < 135**.

Underneath that add **show string** and add the letter **E**.

We now have North and East coded! I suppose that's technically half a compass... unless you want to count north east, north west, south east etc. If you do then I we have 1/8 of a compass.

Whilst we're on the topic, North, East, South, West are known as the compasses cardinal directions (NE, NW etc. are known as inter-cardinal directions) so you could also say that we have 50% of the compasses cardinal directions done.

Quick fire coding project eh?

Step 4- Things are going south now... which is why we're going to code our south compass point next. I'll be honest, my hands are getting tired from all the typing... a lot of this step is repeating the process. The angle needs to **225** and show string is **N**.

Don't forget to press the + twice and then the - on the bit that says **else**.

Step 5- On to the final step and again, I'm not going to ruin my fingers typing as it's repetitive. I will, however, treat you to a compass related joke...

*What do you get if you cross a compass with an alligator?
A Navigator!*

Anyway, have you added another **else if degrees <** with the number of **315** and a show string of **W** yet? I wouldn't be surprised if you've been laughing so hard at my compass joke, that you've forgotten what you're making.

We're making a compass... actually we've made a compass!

I'll whack the full code on the next page... which you probably could have just copied in the first place. But at least you got a good joke out of reading this waffle.

```
forever
  set degrees to compass heading (°)
  if degrees < 45 then
    show string "N"
  else if degrees < 135 then
    show string "E"
```

```
forever
  set degrees to compass heading (°)
  if degrees < 45 then
    show string "N"
  else if degrees < 135 then
    show string "E"
  else if degrees < 225 then
    show string "S"
```

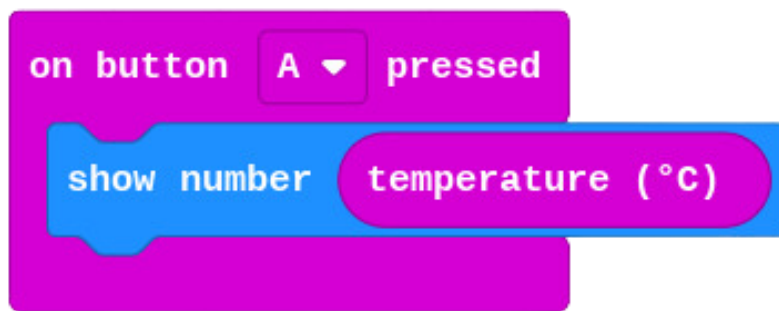
```
forever
  set degrees to compass heading (°)
  if degrees < 45 then
    show string "N"
  else if degrees < 135 then
    show string "E"
  else if degrees < 225 then
    show string "S"
  else if degrees < 315 then
    show string "W"
  +
```

2- Thermometer

This really is a quickfire coding project for the Microbit as you only need 3 blocks of code. I'd tell you a thermometer world but I can't think of one.... We may as well make one of these projects quick-fire I suppose.

The blocks of code you need can be found in **input, basic and input ...** why did I write input twice? To boost the word count.

The blocks you need are **on button A pressed, temperature (C)** and **show number**. You need to then assemble them like Thanos assembling the infinity stones like this...



And that's the thermometer done!

Told you it was quick-fire.

3- Dice

You know what I enjoy? Rolling a dice. You know what I enjoy more? Rolling a dice on a Microbit...

Okay, that's a lie- but in this quick-fire coding project we'll turn the humble Microbit into a dice... I suppose we can turn the arrogant Microbits into a dice as well.

STEP 1- It's time to shake... your Microbit. Head across to **input** and get an **on shake** block. Whatever we put inside this code block will activate when the Microbit is shaken... not stirred.

Speaking of putting things inside it, go to **basic** and add **show number 0** to the code.

I know what you're thinking... what time is lunch... I mean- dice don't have the number 0 on them. Well, the beauty of a Microbit dice is that they can have whatever number you like on them. But, we will code a traditional dice (1-6) so I don't get told off by the dice police.

To be a proper dice we need our Microbit to choose a random number every time we roll it. For this, we are going to the **math** section. Sorry to anyone reading this from Britain... we, quite rightly, call it maths but on makecode... it's math. G-R-O-S-S.

In the **math** 🤖 section you will see a **pick random 1-10** - that's the block we need. Drag it into where your code currently reads **0**.

To finish off this step, we need to change the numbers in the **pick random** to 1 and 6... so actually, just change the 10 to 6.



Expecting more code? Well there isn't any... we've just built a dice. Give yourself a pat on the eyebrows.

4- Stopwatch

In my opinion one of the most useful tools is a stopwatch as you can use it for so many different things. You can time how long you wee for, how long your fart lasts, how quickly you can run away from a plague of butterflies... the list for needing a stopwatch is potentially endless. What's the saying? The limit is your imagination.

Step 1- We need a way to activate. Find your way over to the **input menu** and get the very attractive looking **on button A pressed**. Then we need to create an equally attractive **variable** and call it **start**.

Go ahead and whack a **set start to 0** from the **variable menu** into the **on button A pressed** block. Then go back to the **input menu** and click on **more** to find **running time (ms)** and pop it over the **0**.

So we can master our time keeping, **ms** stands for **milliseconds**. There are 1000 milliseconds in a second.

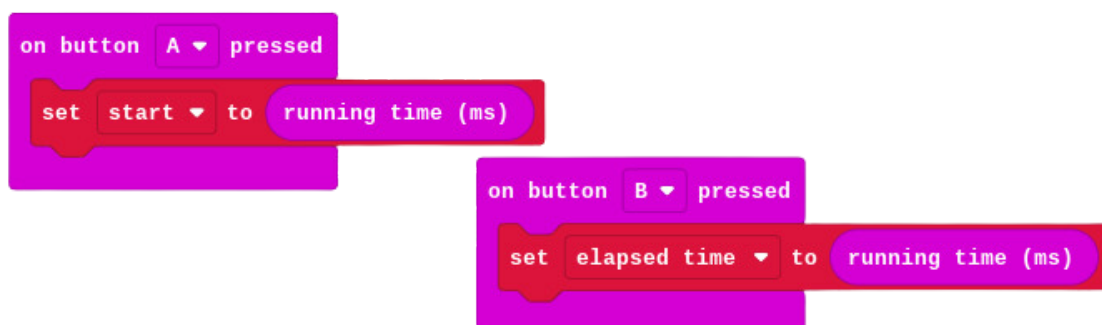


Step 2- We know by now that step 2 is always my favourite as 2 rhymes with poo. This step 2 is no different, we need to add an **on button B pressed** to our code. This one is sneaky though, because the button B pressed is hiding (like a Ditto) as a button A pressed in the **input menu**.

Grab yourself **on button A pressed** from the menu and click on the small arrow next to the A to change the letter to B.

Once you've done that, make a **new variable** called **elapsed time**. If you can spell that right on the first attempt then you've done better than I did... it took me 4 attempts. With the variable created, add **set start to running time (ms)** to the **on button B pressed** block.

We've made twins! #twinning



Step 3- For step 3 we're going back to basics with our need to go to the **basic menu**. From there, grab yourself a **show number 0**. Add it to the **on button B pressed** block.

Time for **math!** Go to the maths menu and get yourself a **square root 0** block. We're going to transform it though. Click on the arrow next to **square root** and change it to **integer /**. wow... what on earth does that mean?

An integer is a whole number. For full information if a number has a decimal point it is called a float. Don't worry too much about it, but if you wanted to do more coding with other languages, integers and floats will come up again.

So integer means whole number. The **/** is a lazy way of writing **divide**. So this code block (which I've already talked about for too long) means **divide this whole number by**.

Okay, add a **variable** block of **elapsed time** into the left hand 0 and change the right hand 0 to 1000. Then drop the whole block over the 0 in **show number 0**.

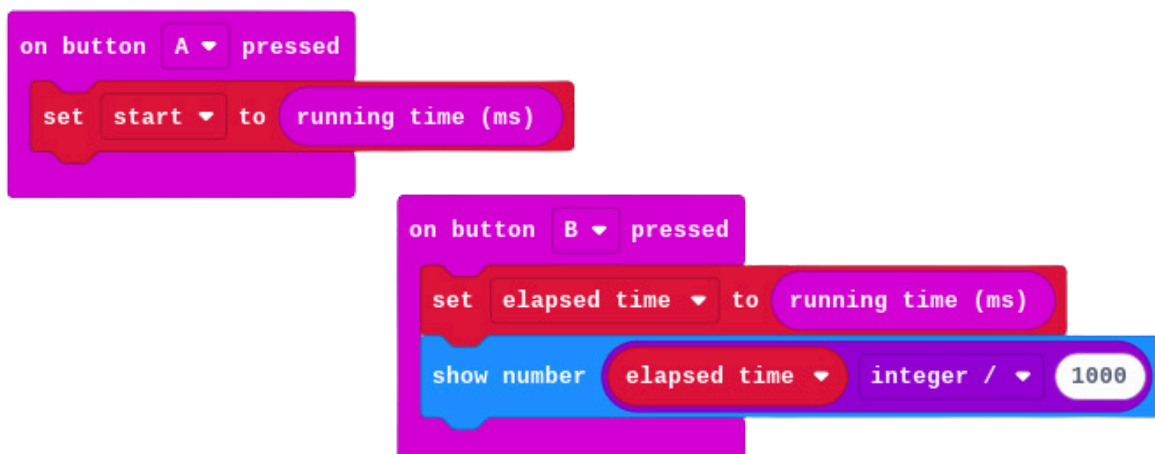


```
on button B pressed
  set elapsed time to running time (ms)
  show number elapsed time integer / 1000
```

And there you have it. Your very own stopwatch. You press A to start the stopwatch and B to get the time taken. You can press B as many times as you like.

If you fancy extending your stopwatch then you could work out a way to make pressing **A+B** at the same time, resets the time to 0... I'm not going to tell you how to do it though.

Have fun timing random things!



```
on button A pressed
  set start to running time (ms)

on button B pressed
  set elapsed time to running time (ms)
  show number elapsed time integer / 1000
```