Against the Clock

Introduction

You are going to learn how to make a timer, so that you can use it to challenge your friends!

![Microbit display showing A+B button being pressed](image)

Resources

For this project, the MakeCode (PXT) microbit editor should be used.

Learning Objectives

- Repetition (while loop);
- not Boolean operator.

Challenges

- "Challenge your friends!" - use the timer to set challenges for others to complete.
- "Counting Down" - using a while loop to create a timer that counts down until it reaches 0.

Step 1: Setting up your timer

Let's set your timer to 0 when buttons A and B are pressed together.
**Activity Checklist**

- Go to [rpf.io/microbit-new](https://rpf.io/microbit-new) to start a new project in the MakeCode (PXT) editor. Name your project 'Timer'.
- Delete the **on start** and **forever** blocks, as you don't need them.
- Add a new **on button pressed event** and select **A+B**:

![Diagram of on button A+B pressed event](image)

- Click the 'Variables' then 'Make a variable', and create a new variable called `time`.

![Diagram showing creating a variable](image)

- When buttons A and B are pressed together, you want the `time` to be set to 0. To do this, drag a **set** block into your **on button A+B pressed** block:

![Diagram showing setting time to 0](image)

The default value of zero is what you need.
Let's use button A to start your timer, and button B to stop it.

You should also display the time. To do this, drag in a show number block and drag your time variable into it:

Click 'run' to test your code. Press the 'A+B' button (below the micro:bit) to set your timer to 0.

Step 2: Starting and stopping your timer

Let's use button A to start your timer, and button B to stop it.

**Activity Checklist**

- Your timer should start when button A is pressed. Add a new on button A pressed block to your script:

- The timer should count up as long as button B has not been pressed. To do this, first drag a while block into your new on button A pressed event.
Drag a **not** block, from 'Logic' to your **while** block:

You can then drag a **button B pressed** block after the **not** block.

Any code inside this **while** loop will be run repeatedly, **as long as button B has not been pressed**.

Next, you want to add 1 to your **time** variable every second (1 second = 1000 ms). Add a **pause** block to make your timer wait for 1 second.
To increase your `time` variable,

```
on button A pressed
while not button B is pressed
do
  pause (ms) 1000
  change time by 1
  show number time
```

Finally, you'll need to display the updated `time` variable. Here's how your code should look:

```
on button A pressed
while not button B is pressed
do
  pause (ms) 1000
  change time by 1
  show number time
```

Click 'run' to test your code.
- Press buttons A and B together to set your timer to 0
- Press button A to start your timer
- Press (and hold) button B to stop your timer
Challenge your friends!

Use the timer to challenge your friends. For example, you could see how long it takes them to say the alphabet backwards, or name 10 capital cities.

Challenge: Counting Down

Can you create a new timer, that counts down to 0? Here's how your new timer should work:

+ Pressing buttons A and B together should reset your timer to 0.

+ Pressing button B should add 1 to your timer. Press it 10 times to create a 10 second timer.

+ Pressing button A should take 1 from your time variable until it gets to 0. This means you'll need a while loop that runs as long as the time is greater than 0.

Challenge: Accurate timer

Have you noticed that the timer isn't very accurate! This is because it takes time to display and scroll numbers on the micro:bit. Try adjusting the pause to improve the timing. You can use an if/else block to have shorter delays for bigger numbers that take longer to scroll.