Coding for beginners - how to code your Nibble

Changing screen color

Now that you're accustomed to every type of block, it's time to learn how to use them!

Let's begin!

Example #1

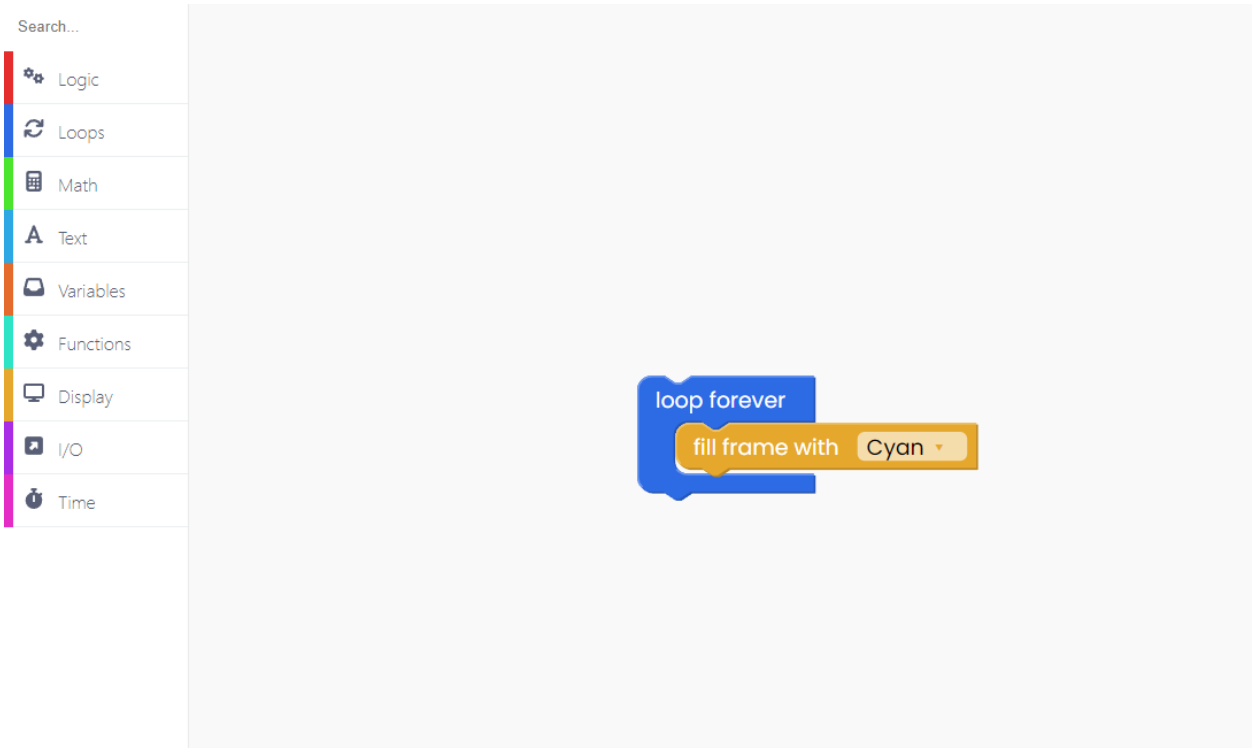
Let's kick things off as simply as possible.

The main component on Nibble is definitely the screen since working without one would be pretty much impossible.

What we want to do as a test is to change the screen color of our program.

The default screen color is black, or as it is known in our library, TFT_BLACK.

All colors from this library are labeled with TFT_ prefix because they are made to be used on TFT screens.



Now let's move to the code generated by the blocks.

The first thing we need to take is the "Loop forever" block from the "Loops" block section. This block will ensure your code is run all the time.

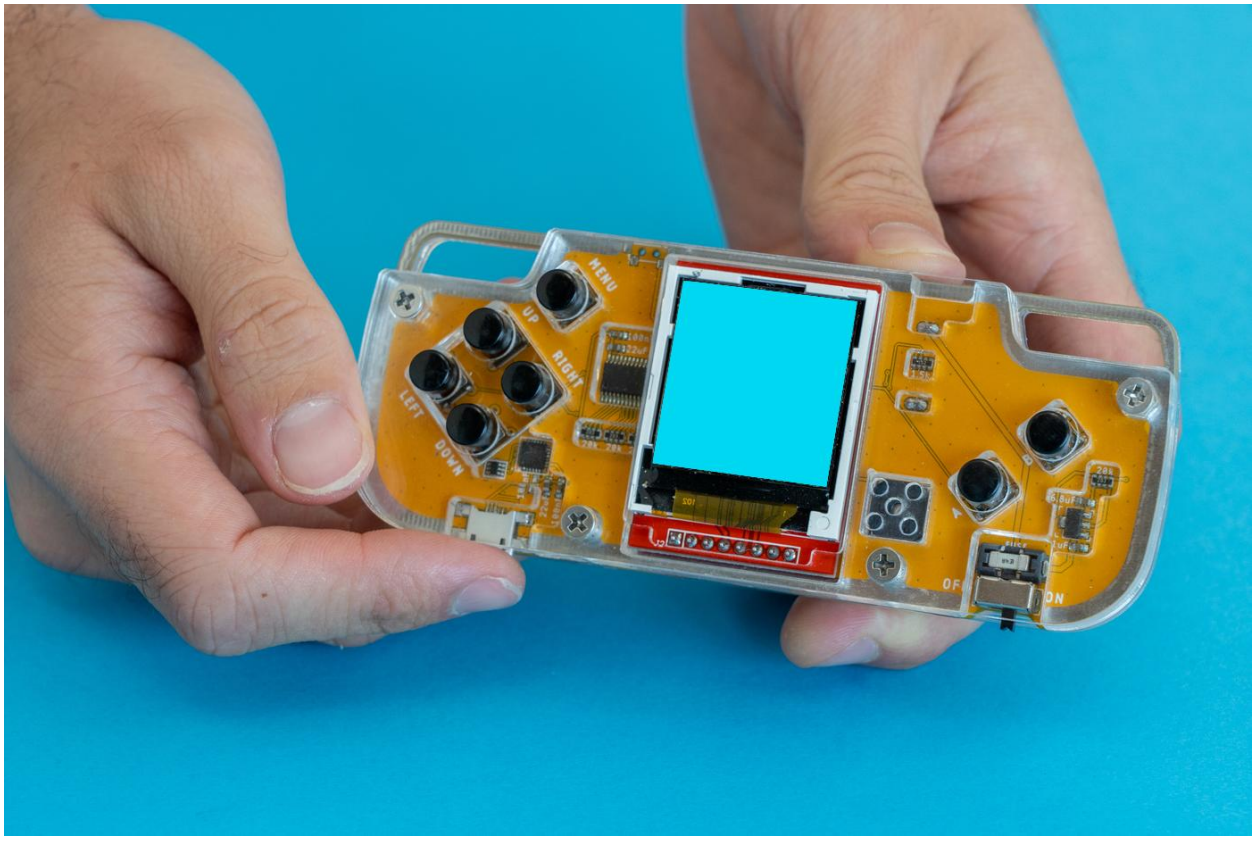
In the "Display" block section find the "Fill frame with" block and place it inside the purple block. You can choose any color you want by clicking on the color part of the block.

For this example, we chose cyan.

Click on the Run button.

Nibble's screen should now be cyan. It's really as simple as that.

This is just a beginning but you can imagine that the possibilities of this screen are countless.

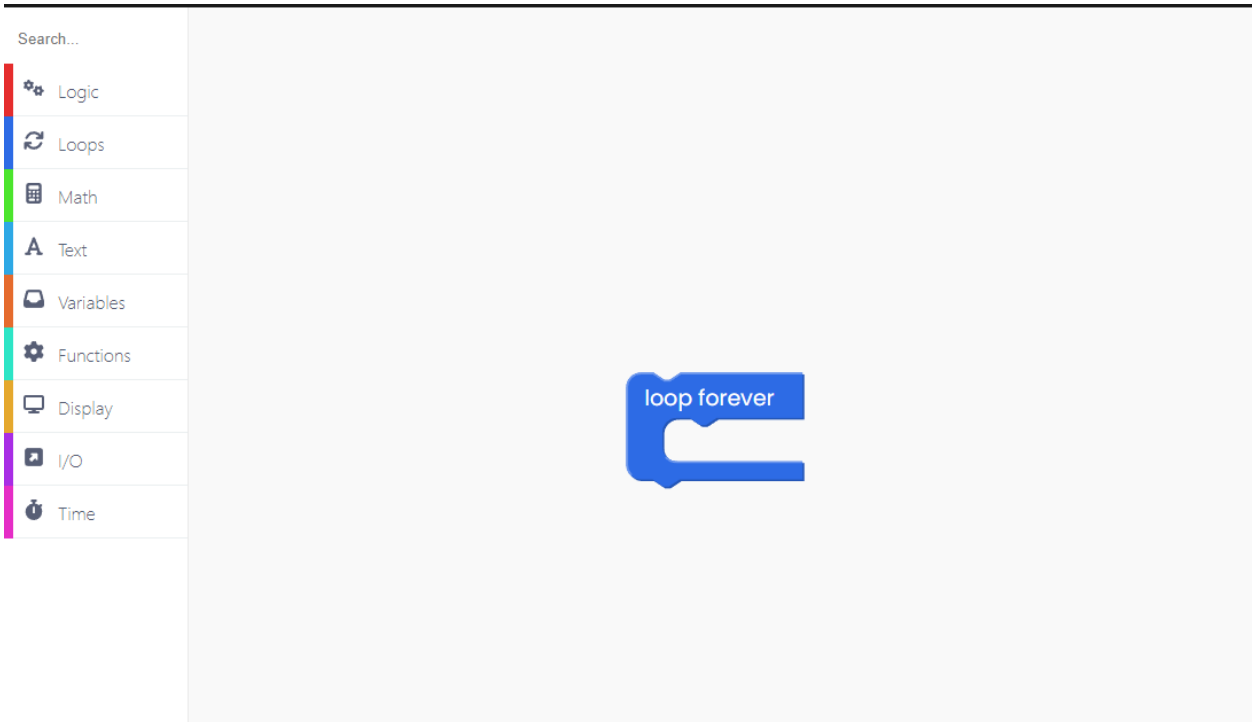


Controlling the display

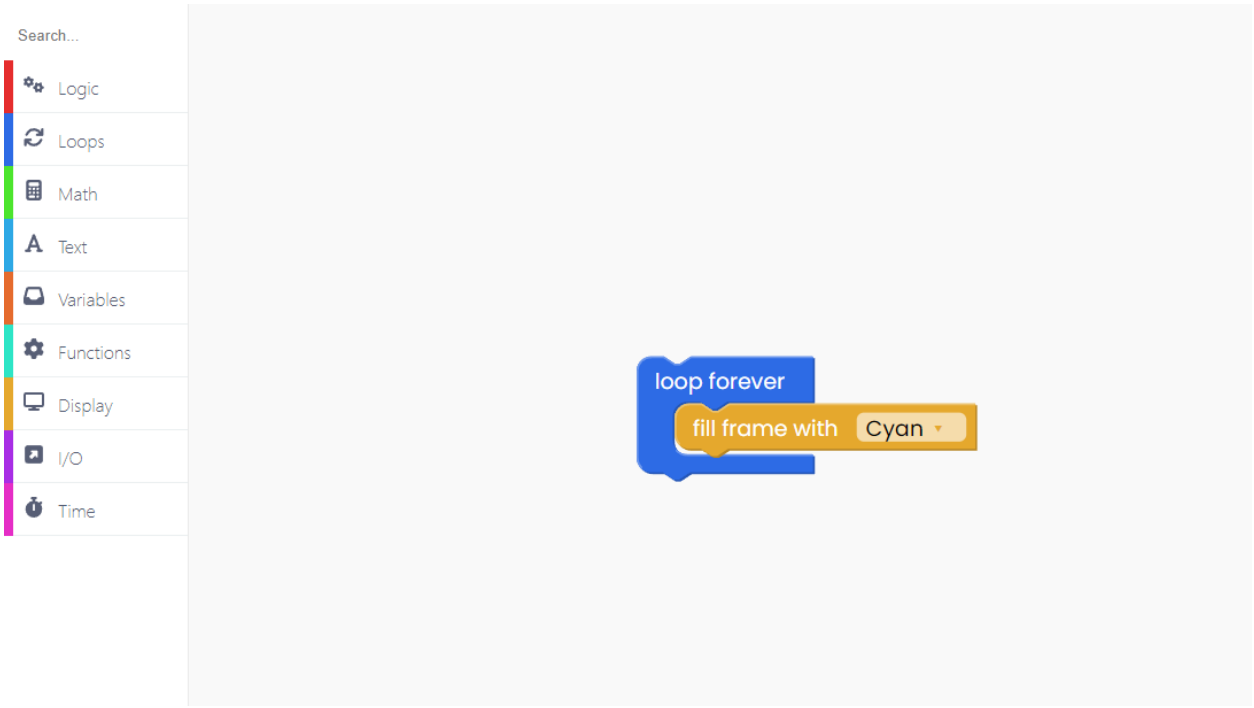
Scrolling colors

We'll use this code to have the display light up in different colors every 500 ms.

Firstly, use the "Loop forever" block from the "Loops" block section.



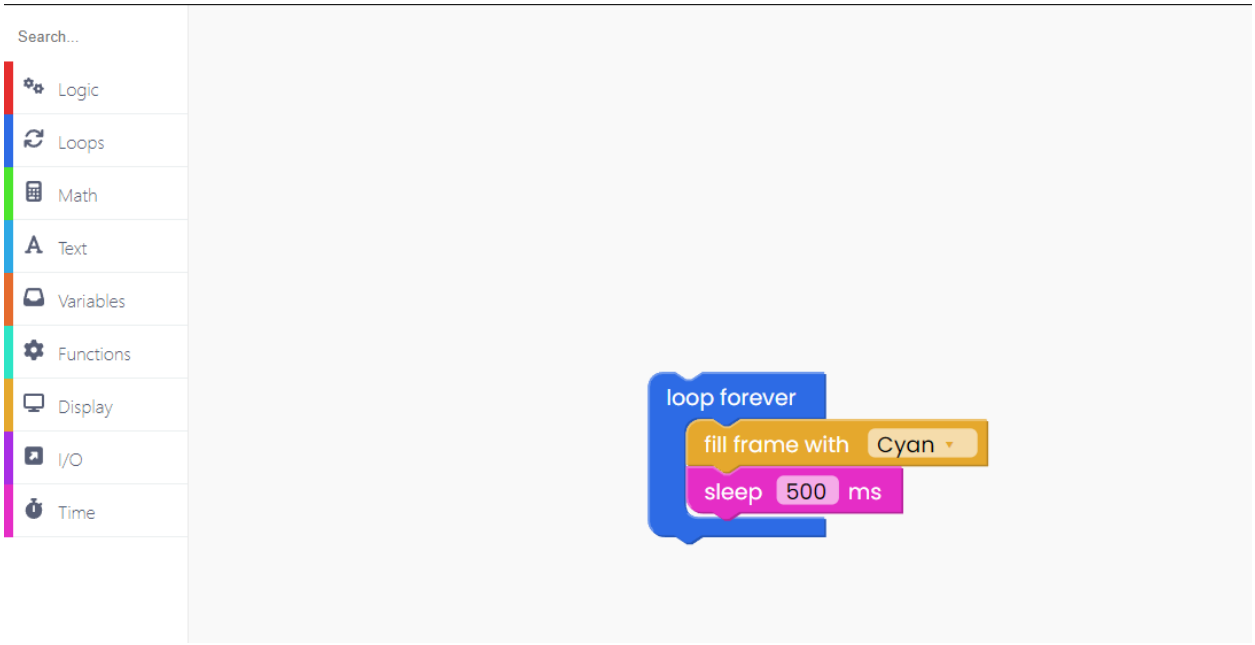
Now, look for the "Fill frame with" block inside the "Display" block section, and place it inside the blue block.



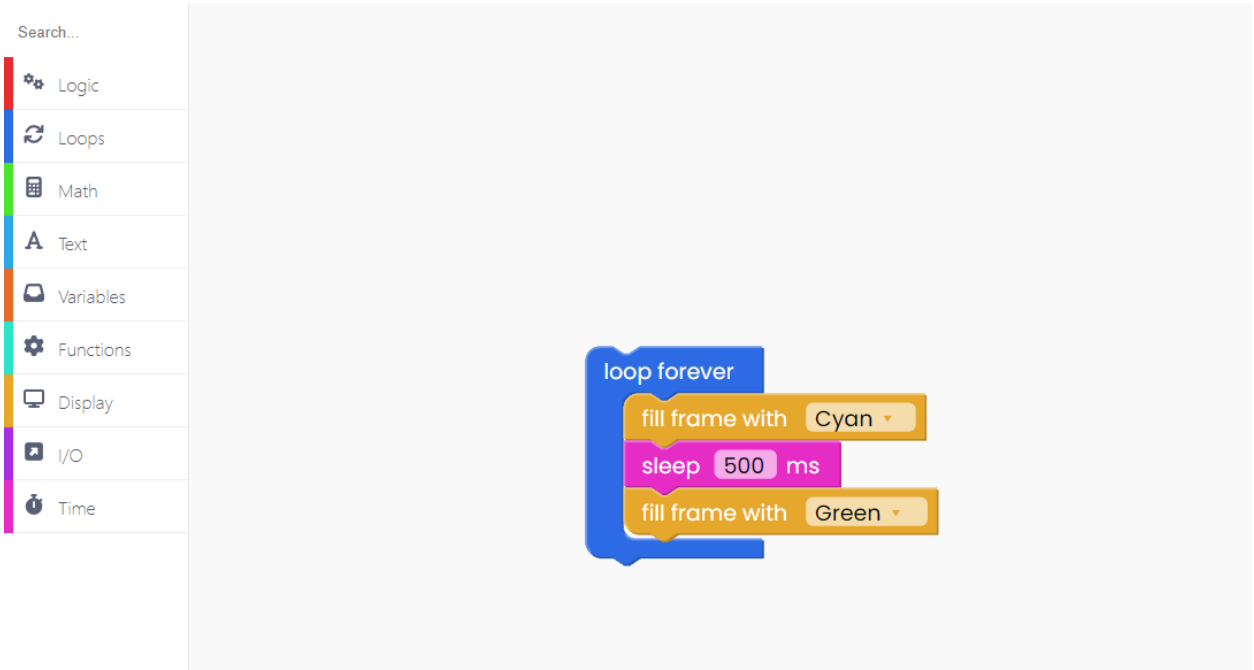
We'll start with cyan, but you can choose whatever color you wish.

Now, under the "Time" block section, find the "sleep 0 ms" block and place it below the "Display" block.

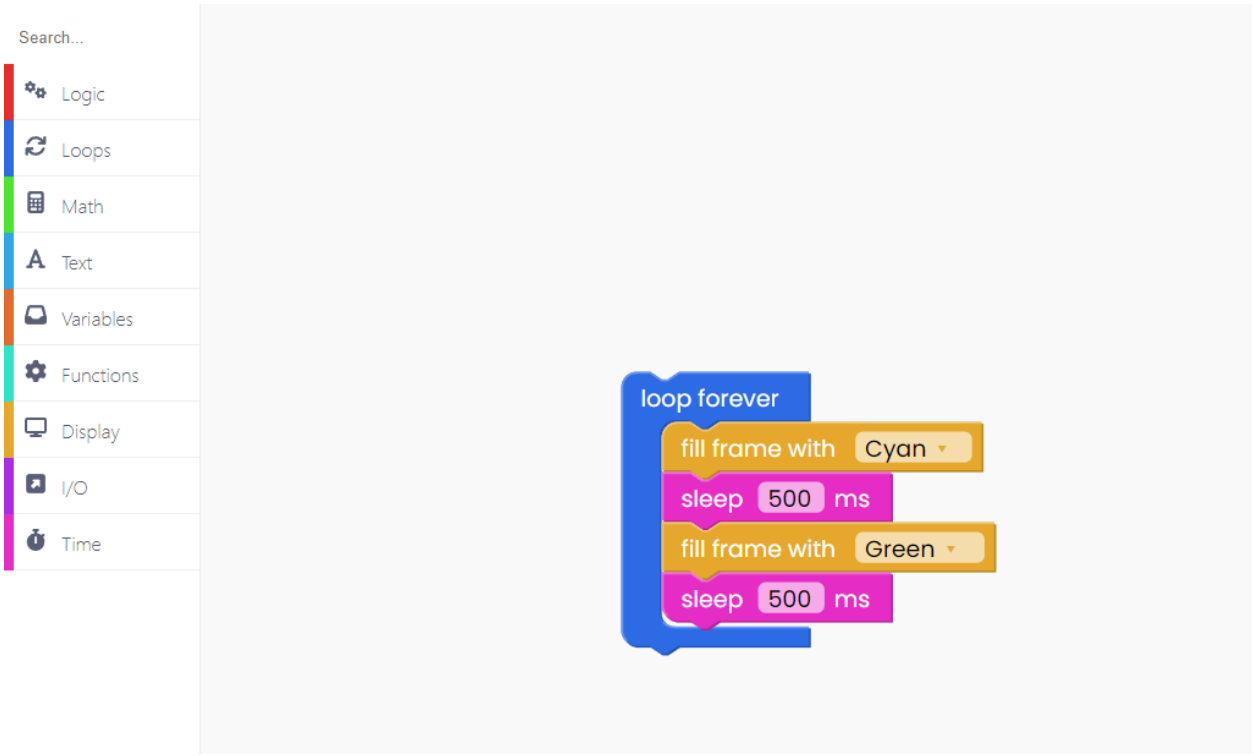
Instead of 0, enter 500. This indicates that it will take exactly 500 milliseconds for the display color to change.



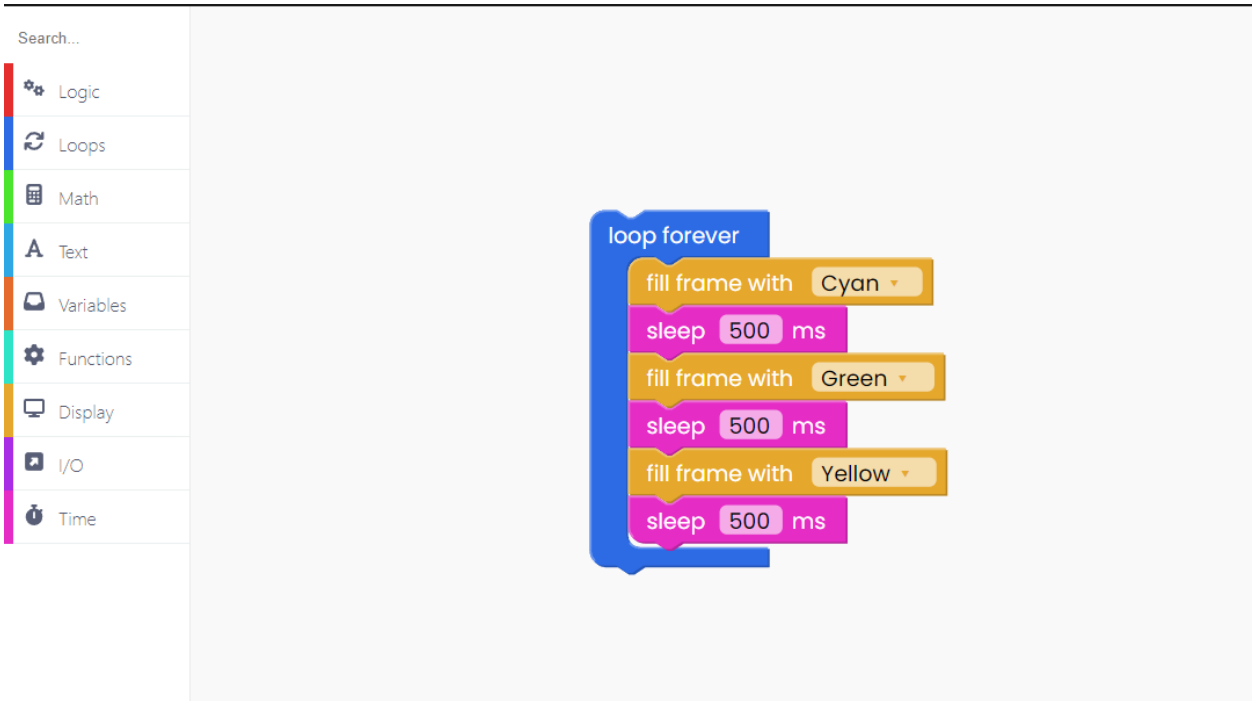
Now, we'll go back to the "Display" block section and change the color of the display to green.



Add another time block below this one.



Repeat this step once more.



Amazing!

Now, you can click on the Run button.

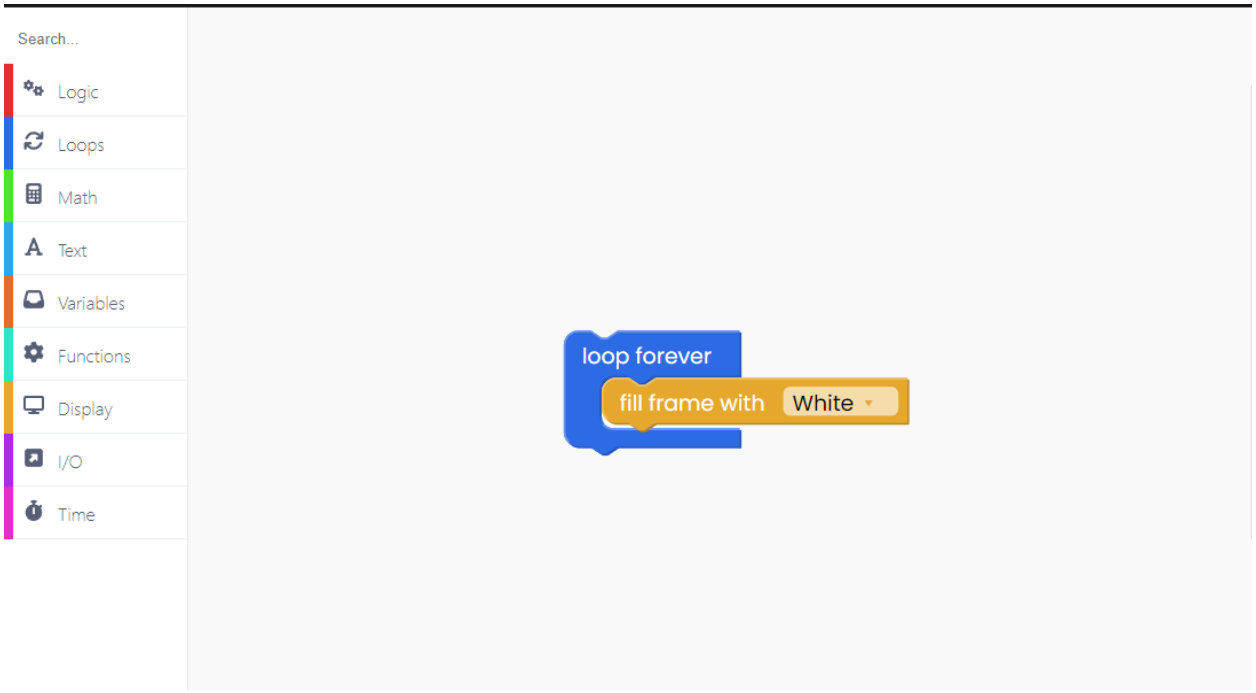
Every 500 ms, the display will begin to change colors.

Writing out some text

Now that we know how to change the background color, it's time to start writing something on that canvas.

Find the "Loop forever" block, and place the "Fill frame with" block within it.

We'll use white to fill the frame.



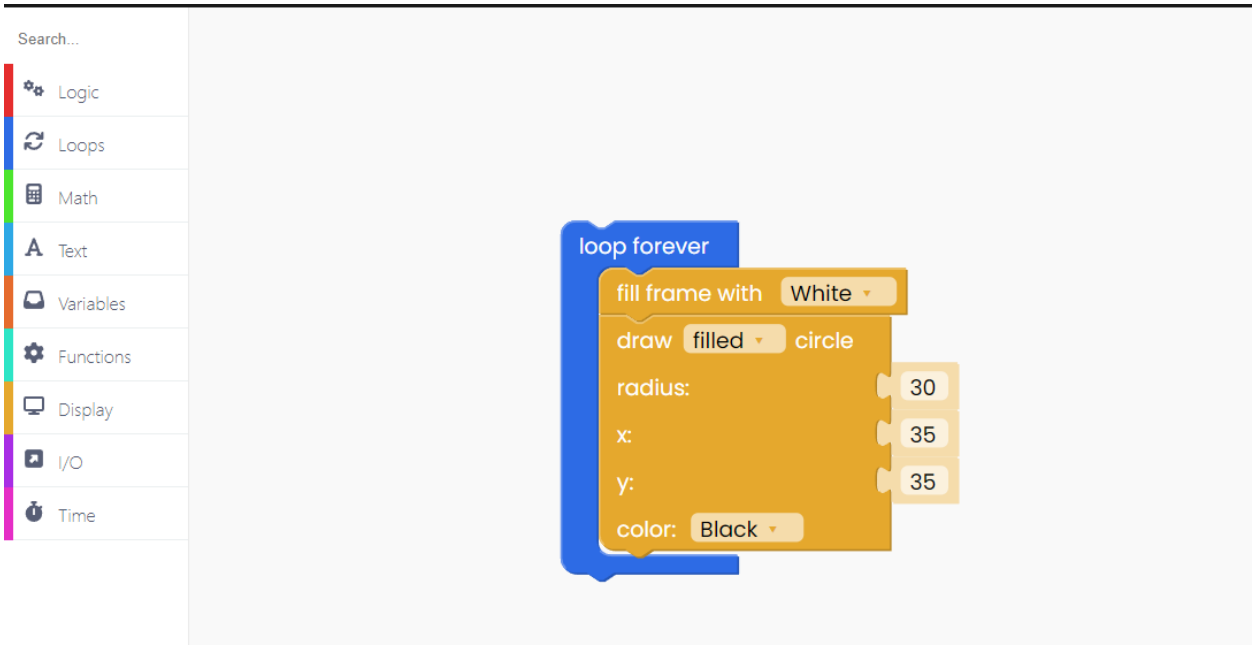
After the display is filled with white, you'll try to code the filled circle on it.

To do so, find this block and insert it beneath the first one.

When you click on the filled, you'll find that you have the option of making a filled or outlined circle.

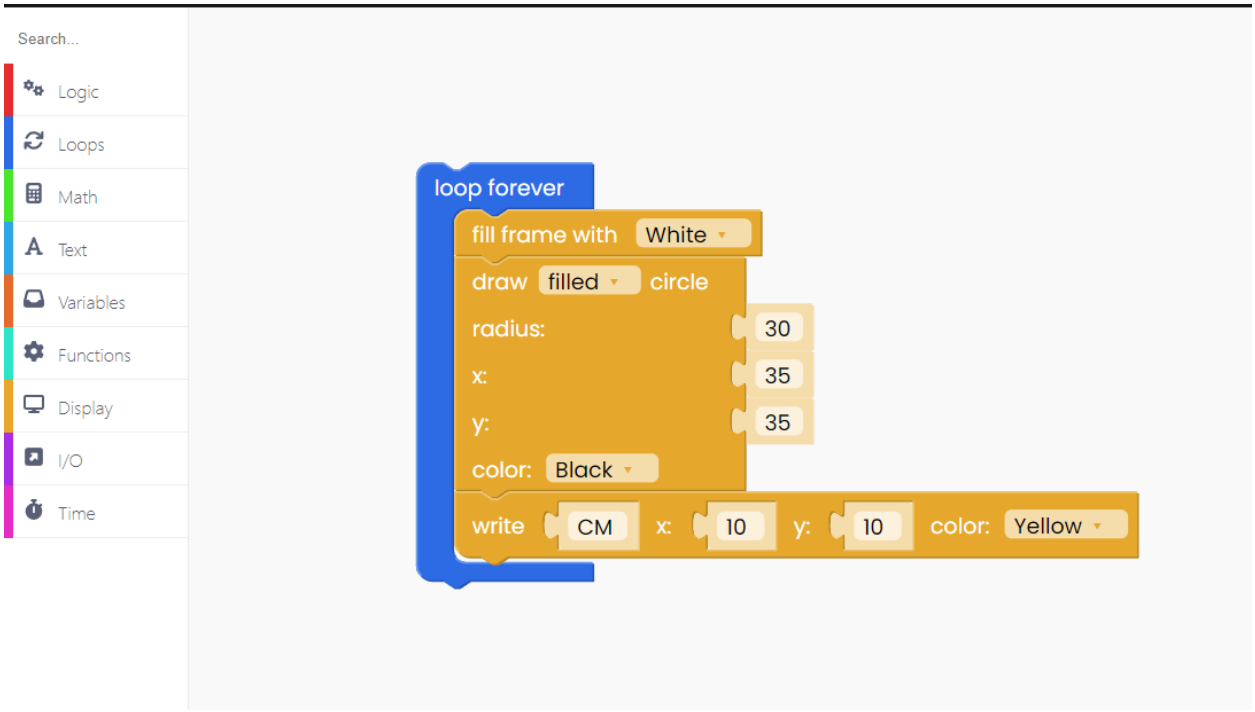
You can also customize the circle's color.

We chose 30 for the radius and 35 for the x and y coordinates (where the circle will show on the screen).



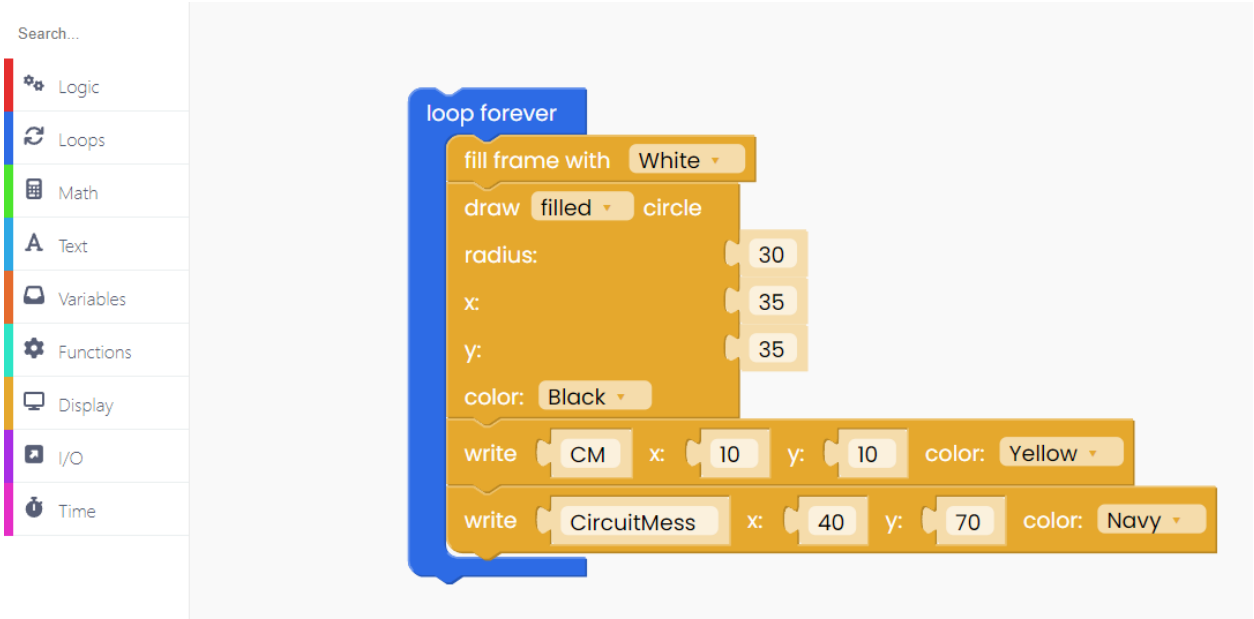
After this, we'll write something on the screen.

To do that, find the "Write" block from the "Display" block section.



We'll put "CM" in yellow lettering at the coordinates x:10, y:10.

Duplicate the previous block, and do this with it:



Click on the Run button and check it out!

Shapes and animations

Shapes

There are plenty of blocks that are still unexplained but offer so many different possibilities.

For now, let's focus on the ones from the Display section.

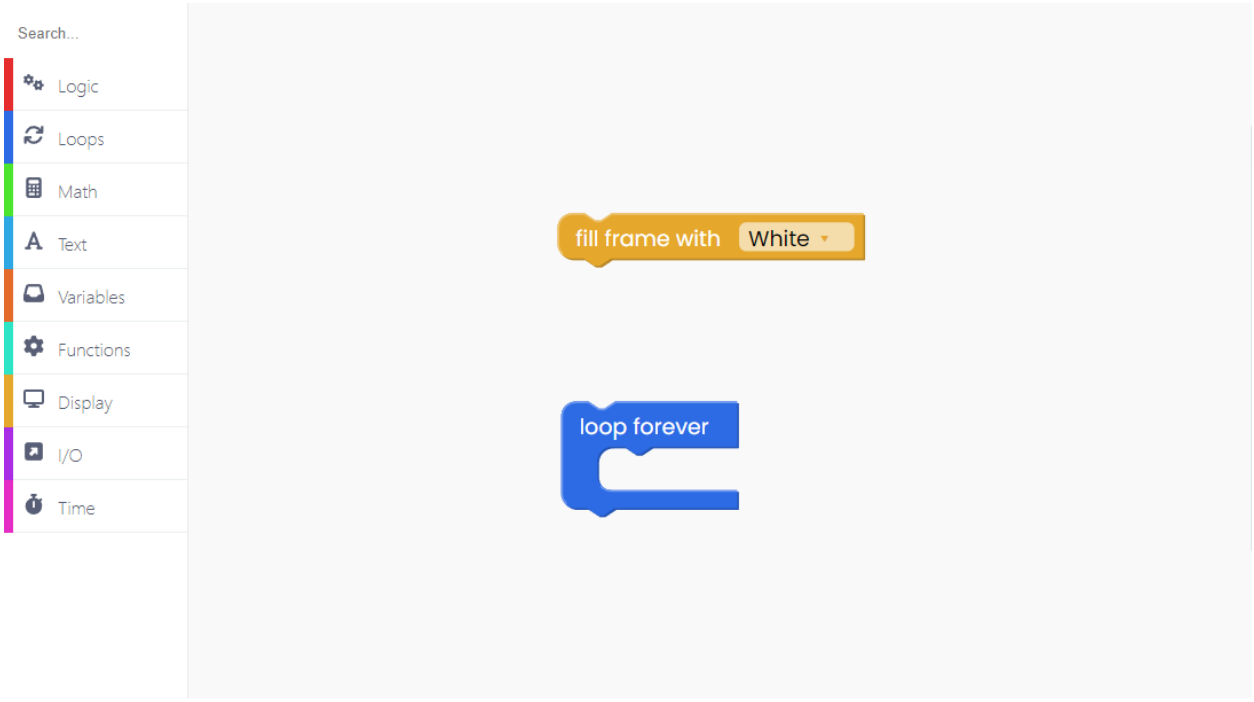
We know how to paint the entire screen with one color, but what about only part of it?

There is a function for drawing some of the most used shapes - rectangle, circle, and line.

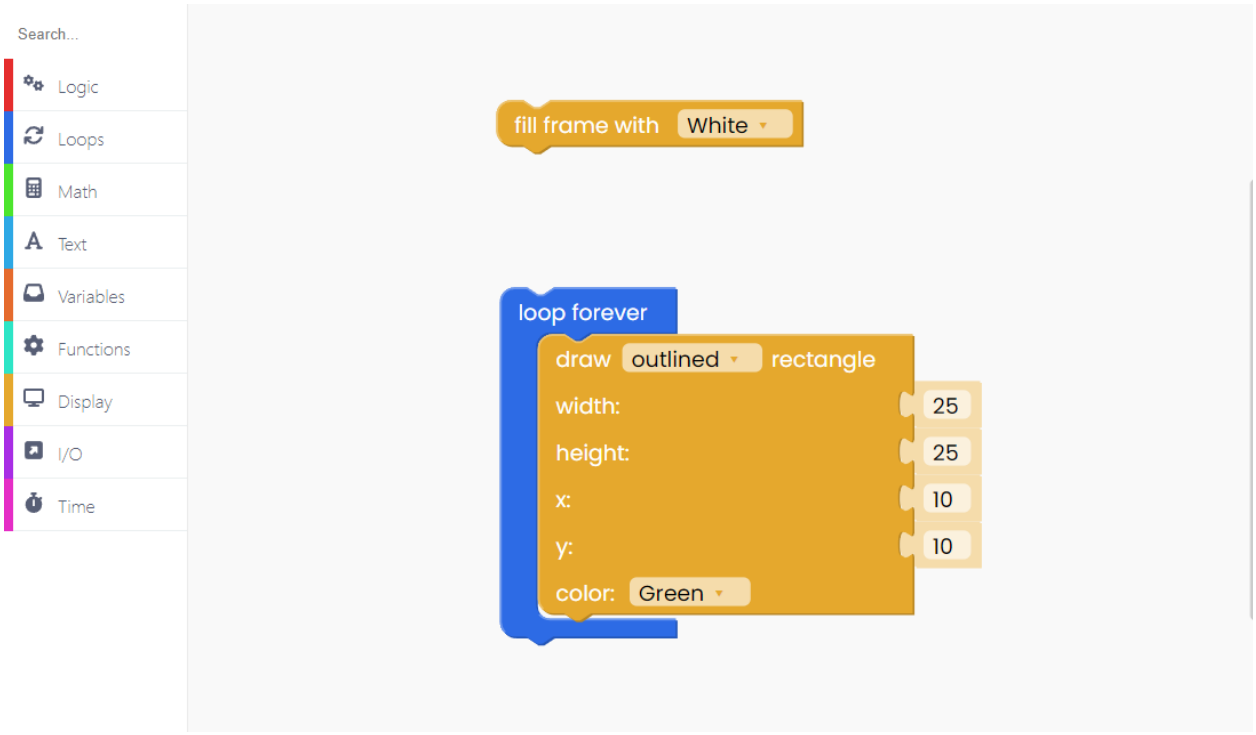
All of these can be either filled or not while using the same colors as for the screen.

Here is an example of one such code.

The first thing we want to do is fill the screen with white and put the "Loop forever" block in the drawing section.



Let's draw an outlined rectangle.

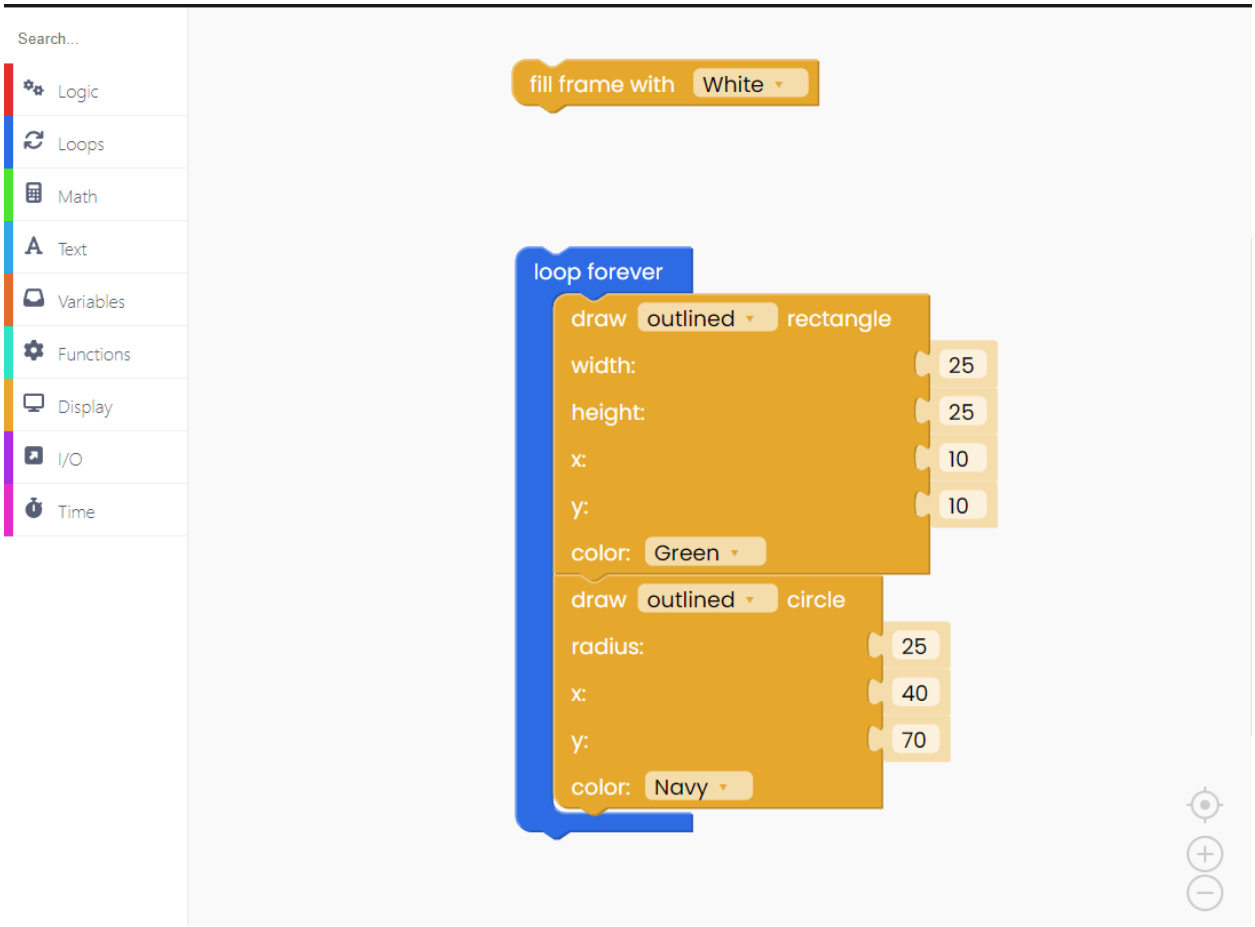


We chose a green outline. Both the width and height are set to 25. Also, both coordinates are set to 10.

The color of the rectangle is green.

We'll also draw an outlined circle with a radius of 25 and coordinates of 40 and 70.

The color of the circle is navy.



Click on the Run button and check it out.

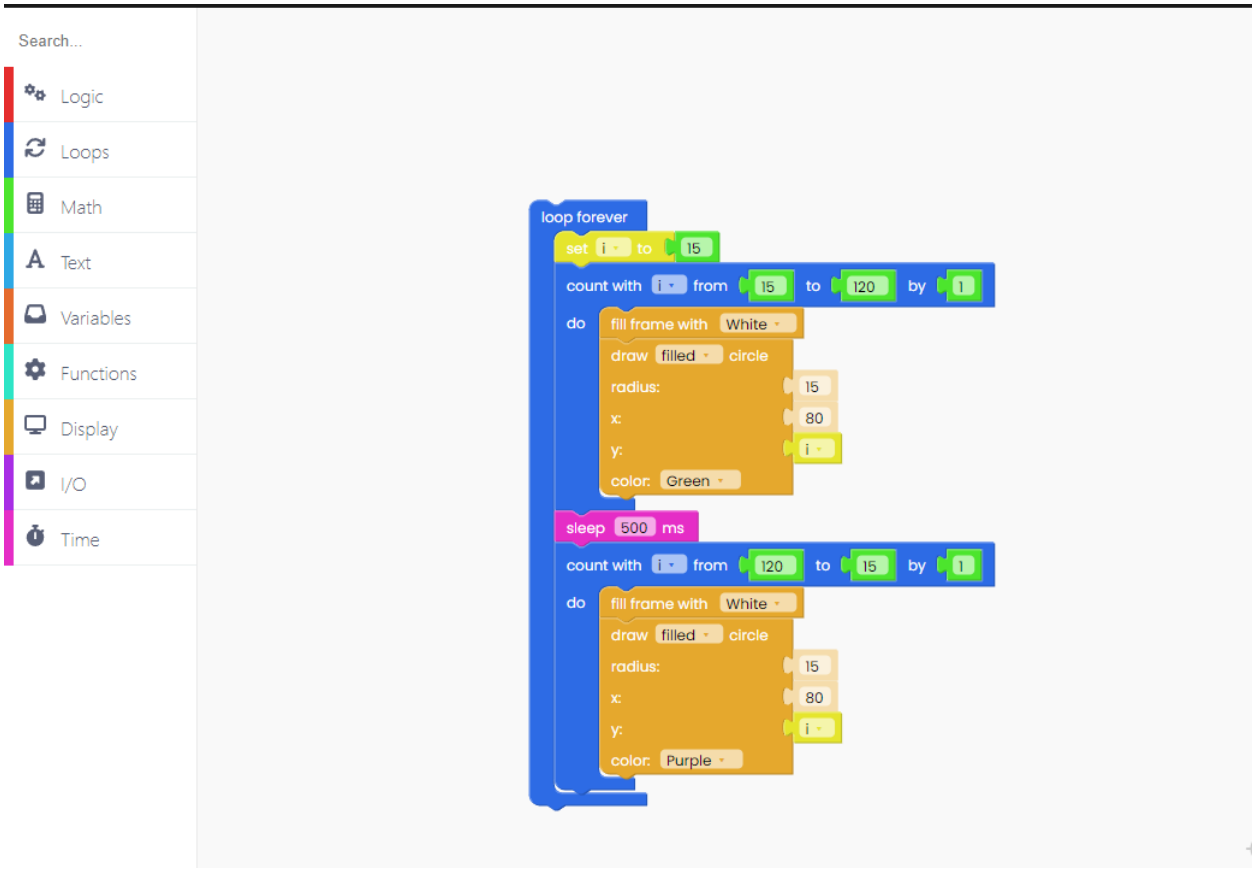
Animations

The screen refreshes 25 times in a second, which is fast enough to create the effect of a moving image.

Animations are just that - shapes and images moved so slightly that they have a smooth movement.

It's best that we show it by example.

We're going to take a circle and move it from the top of the screen to the bottom.
When it reaches the bottom, it will change its color and direction in which it moves.



What exactly does "Loop forever" do? It repeats whatever is inside it a specific amount of times.

As we can see in the example, we set i variable to 15 and increase it by 1 every time the loop loops.

When it gets to 120, the for loop breaks, and the program continues.

But what does this exactly have to do with animation? Well, we used that same i as one of the coordinates for the circle.

Every time the loop ran, the circle went down by 1 frame, but the loop ran so fast that we only saw the smooth movement of the circle.

Pressing the buttons

Nibble has 7 main clickable buttons, 1 reset button, and 1 power switch. It's quite simple - you press the button, and something happens.

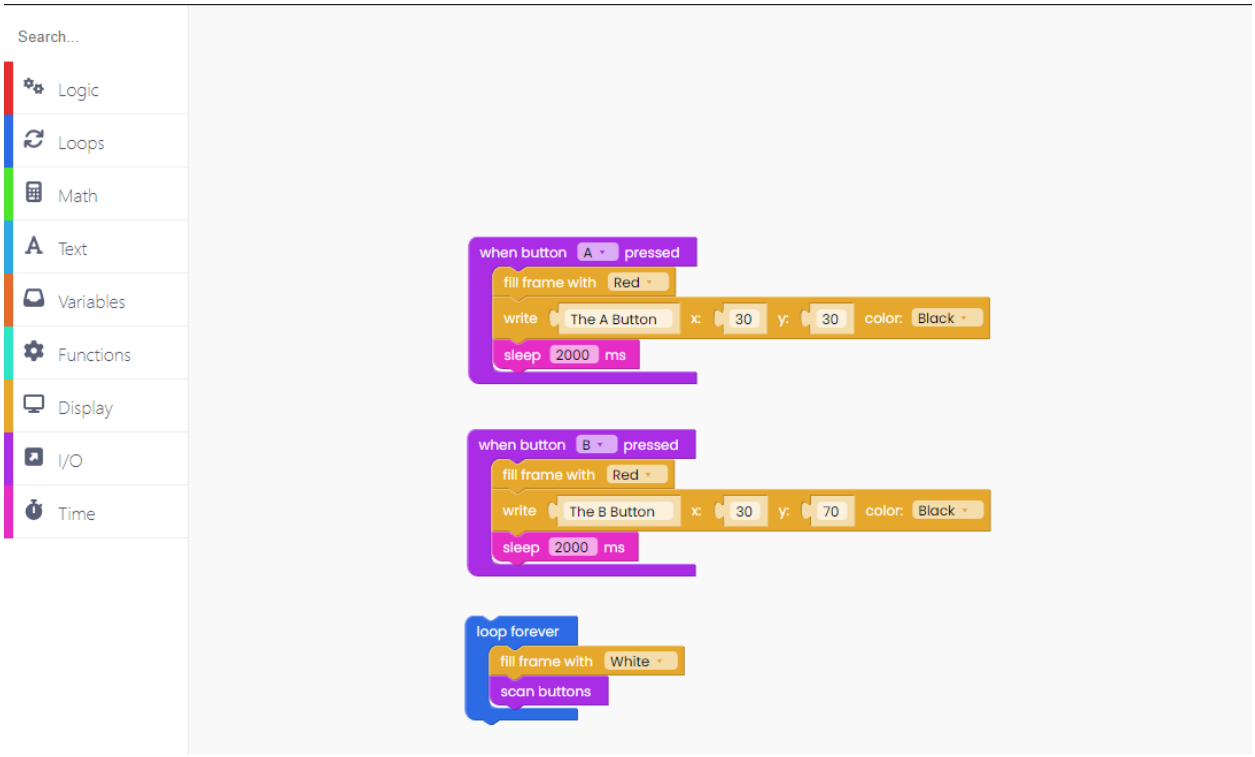
Only the main 7 buttons are programmable, while the remaining two always have the same function.

The buttons have tiny white text next to them, so you know which is which.

On the left side of the console, there are directional buttons and the menu button, while on the right side A and B buttons.

We're going to focus here on using the functions of drawing on the screen, but this time we're controlling it with a simple press of a button.

This gives us much more flexibility and dramatically improves the possibilities of a program.



This program is really simple - it changes the color of the screen and writes a simple text when a button has been pressed.

The screen color and the text are different depending on the button that has been pressed.

The system that is used on Nibble is called the callback system, meaning that the button presses are small events that trigger whenever a button press has been detected.

That way, there is no checking whether the button has been pressed or not in the loop section, but rather only when a button has been pressed a button press function is called. Remember to include the "Time" block inside these buttons to specify how much time will pass between each action.

Another thing you must never forget while using the buttons is to include the "Scan buttons" inside the "Loop forever" block to ensure that your buttons are being scanned at all times.

Holding the buttons and multiple functions

Now we go a bit more advanced.

Besides activating the button on press and release, you can also activate it after holding it down for a certain amount of time.

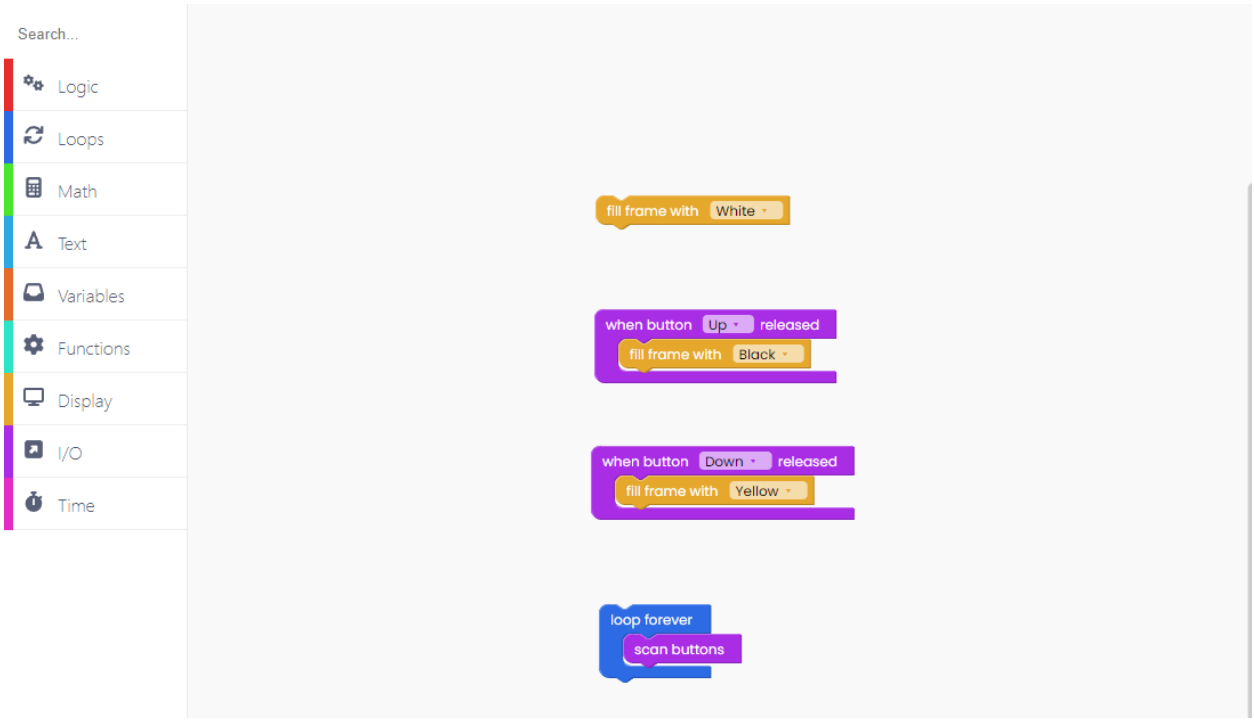
Firstly, we fill the frame with white.

Then, we look for the "When the button pressed/released" blocks and drag it into the drawing section.

What will happen when a certain event is triggered? The display will change the color.

Also, don't forget to include the "Scan buttons" block at the end.

Click the Run button and check it out.



Functions

Now that you've been introduced to every component, it is a good time to meet some functions, which are going to be crucial for making our first app.

Functions are basically sections of the code that receive and return some values and can be used in many ways.

They are crucial if you want to keep your code clean and fast. Also, they allow for some quick upgrades without having to redo half of the code.

Every program is made out of two main functions that we've already mentioned - void setup and void loop.

Other functions work pretty much the same way.

How do functions work?

You can create a new function in the Functions section, where you're setting its name, type, and variables
Once called, the program will automatically jump to the code in the function and will run that before continuing with the rest of the program.

You can see that the code will execute the same way whether it has been called from the function or whether it was directly inserted inside the void loop.

Whatever function returns can be caught in another variable or in comparison.

For example, you can call a function and, depending on what it returns, decide whether to enter the loop or not.

There are numerous ways you can use functions, but the premise is the same every time.

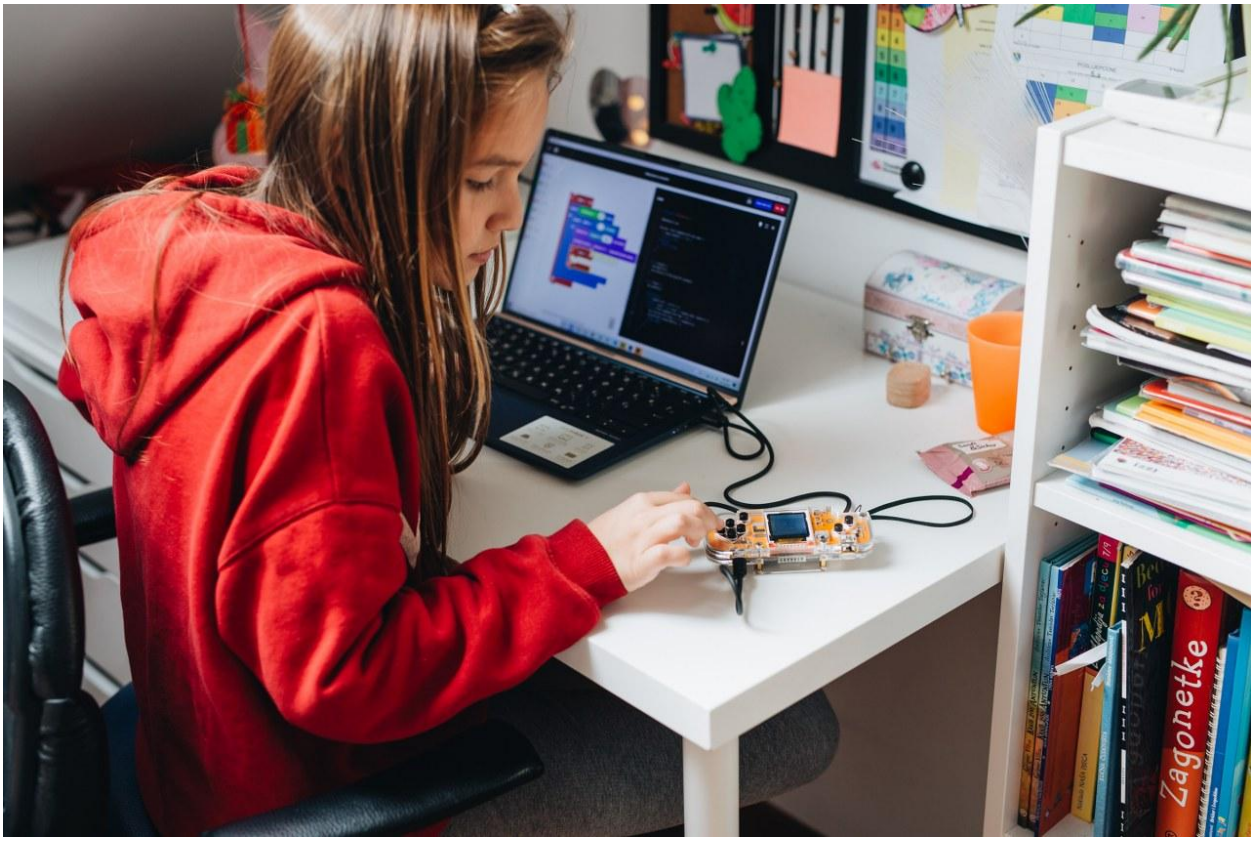
You can experiment a bit more with it or continue with the tutorial.

What's next?

You've reached the end of the Nibble coding tutorial; congratulations!

Continue exploring on your own and show us what you've done with your Nibble by sharing it on the CircuitMess community forum or our Discord channel.

If you need any help with your device, as always, reach out to us via contact@circuitmess.com and we'll help as soon as we can.



Thank you, and keep making!