Coding for beginners - how to code your Chatter

Let's draw on the display!

Let's get down to business!

Before doing anything, you need to connect your Chatter to your computer's USB port and turn it on.



If everything is okay, CircuitBlocks should say, "Chatter connected".

Let's draw something!

We will kick things off as simply as possible.

The first thing we're going to learn is how to clear Chatter's display and draw on it!

You'll only need to use a Display block section to do that.

Firstly, drag and drop the "fill frame with black" block.

Search
🍫 Logic
C Loops
🖬 Math
A Text
Variables
Functions
🖵 Display
I /O
Č Time
_

We changed the background color to white, but you can choose any color you want.

You can choose different colors by clicking on the word "black".

Now, look for the block labeled "draw sprite ... x ... y".

We chose to draw the sword, but you can use any of the available sprites.

Both coordinates are set to 50.

Search				
🍄 Logic				
C Loops				
🖬 Math				
A Text				
Variables		fill frame with White - draw sprite sword - x 50 y: 60		
Functions				
🖵 Display				
I/O				
Č Time				
-				

To guarantee that your code executes successfully, you must always include the "push frame" block at the end of the "Display" codes.

🍫 Logic
C Loops
🖬 Math
A Text fill frame with White -
Variables draw sprite sword - x 50 y 60
Functions
🖵 Display
I/O
Ö Time

Now, click the Run button and the sword should appear on the Chatter's screen.

Click, click...

Now that you know a thing or two about CircuitBlocks, it's time for a bit more advanced sketch.

Firstly, we want to code what will happen first once we run the code.

We'd like to write the instructions on the screen for you to follow.

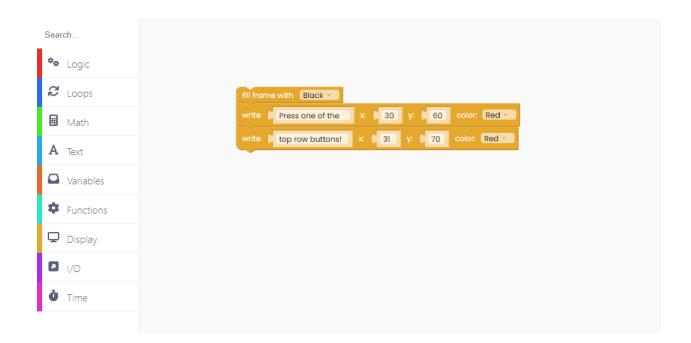
The "fill frame with black" code will be used first.

Search		
🍫 Logic		
C Loops	fill frame with Black	
🖬 Math		
A Text		
Variables		
Functions		
🖵 Display		
I/O		
Č Time		
_		

After we took care of the background, it's time to write the instructions.

Please note that you can choose any background color you want.

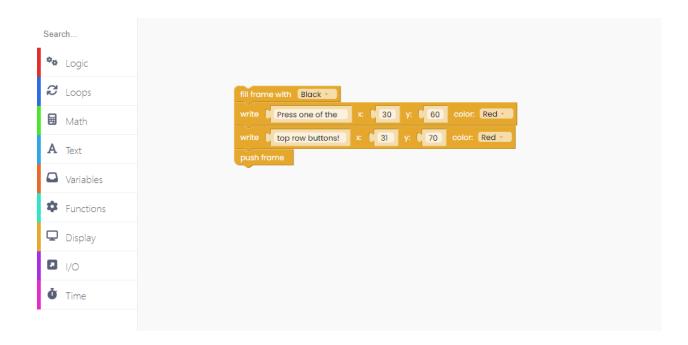
Now, in the "Display" block section, look for the "write" block.



The directions, as you can see, are written in red and say "Press one of the top row buttons!".

To fit the display, we divided the sentence into two blocks.

To guarantee that the code executes properly, add the "push frame" block at the end of any "Display" code.



Let's now code what happens when we press and release the buttons.

To do that, find the "When button pressed" button from the I/O block section.

Search				
🍫 Logic				
C Loops	fill frame with Black -			
🖬 Math	write Press one of the x: 30 y: 60 color: Red write top row buttons! x: 31 y: 70 color: Red			
A Text	push frame			
Variables				
Functions	when button Left - pressed			
🖵 Display				
I/○				
Č Time				
-				

So, within the purple block, we want to write the code that we want to run only once when we push that button.

We want the screen to go black and to show which button has been pressed.

Search ... 🍫 Logic fill frame with Black • 8 Loops Press one of the x 30 y: 60 color: Red -Math write 🕻 top row buttons! 🗴 🕻 31 y: 🕻 70 color: Red 🔹 A Text Variables Functions when button Left 📩 pressed fill frame with Black • Display write Left x: 10 y: 60 color: Red -I/O Ŏ Time

Don't forget to include the "push frame" block at the end, or your code will fail.

Let's make a copy of this I/O block.

Change left to right, and feel free to play with the coordinates.

 Logic Coops Math Math
write Press one of the × 0 30 y: 0 60 color: Red
Matha Matha
write top row buttons! × t 31 y: t 70 color: Red
A Text push frame
Variables
Functions when button Left - pressed fill frame with Black -
Display write Left x 10 y. 60 color: Red
I/O push frame
Ů Time

Duplicate it once more for when you press enter.

Search	
🏘 Logic	
C Loops	fill frame with Black
🖬 Math	write Press one of the x 0 30 y: 0 60 color: Red write top row buttons: x 0 31 y: 0 70 color: Red
A Text	push frame
Variables	
Functions	when button Left · pressed when button Right · pressed
🖵 Display	fill frame with Black · fill frame with Black · write 0 Left × 0 10 y; 0 60 color: Red · write 0 Right × 0 120 y; 1 60 color: Red ·
I /O	push frame
Č Time	
	when button Enter + pressed fill frame with Black +
	write 0 Enter press x: 0 45 y: 0 90 color: Red -
	push frame

We want the text to change to "Enter release" when we release the enter key.

That block can also be found in the I/O block section.

 Loops Math Math Text Variables Functions Display Mode Math Mat	 ↓ Loops ↓ Math ↓ Math ↓ Math ↓ Math ↓ Variables ↓ Variables ↓ Display ↓ Vo ↓ Math Hand ↓ When button Enter · pressed ↓ y · @0 color: Red · ↓ y · @0 color: Red · ↓ when button Enter · pressed ↓ y · @0 color: Red · ↓ When button Enter · pressed ↓ y · @0 color: Red · ↓ When button Enter · pressed ↓ y · @0 color: Red · ↓ When button Enter · pressed ↓ y · @0 color: Red · 	Search	
 Loops Math Math Itext Variables Functions Display Display I/O Time When button Enter = pressed If frame with Black = if frame with	<pre> Loops Loops Loops Write @ress one of the x @ 30 y; @ 60 color: Red # write @ cop row buttons! x & 3 y; @ 70 color: Red # write @ cop row buttons! x & 3 y; @ 70 color: Red # push frame Variables functions when button @ft @ pressed fill frame with Black # write @ left x @ 0 y; @ 60 color: Red # write @ Right & pressed fill frame with Black # write @ left x @ 0 y; @ 60 color: Red # write @ Right x @ 20 y; @ 0 color: Red # push frame write @ left x @ 0 y; @ 60 color: Red # write @ Right x @ 20 y; @ 0 color: Red # write @ left x @ 0 y; @ 0 color: Red # write @ left x @ 0 y; @ 0 color: Red # write @ left x @ 0 y; @ 0 color: Red # write @ left x @ 0 y; @ 0 color: Red # write @ left x @ 0 y; @ 0 color: Red # write @ left x @ 0 y; @ 0 color: Red # write @ left x @ 0 y; @ 0 color: Red # write @ left x @ 0 y; @ 0 color: Red # write @ left x @ 0 y; @ 0 color: Red # write @ left x @ 0 y; @ 0 color: Red # write @ left x @ 0 y; @ 0 color: Red # write @ left x @ 0 y; @ 0 color: Red # write @ left y; @ 0 color</pre>	🍄 Logic	
Math Math Math Write top row buttonst x 13 y. 70 color: Red Push frame Variables Functions fill frame with Black U/O I/O Time When button Eft: pressed write t Left x 10 y: 60 color: Red + When button Eft: pressed fill frame with Black write t Left x 10 y: 60 color: Red + When button Eft: pressed fill frame with Black write t Left x 10 y: 60 color: Red + When button Eft: released fill frame with Black write t Efter release x 45 y. 90 color: Red -	Math Math Wite top row buttonst x 13 y. 70 color: Red Push frame Variables Functions Men button Left pressed fil frame with Black push frame Vo Time When button Inter - pressed fil frame with Black when button Inter - pressed fil frame with Black when button Inter - pressed fil frame with Black when button Inter - pressed fil frame with Black when button Inter - pressed fil frame with Black when button Inter - pressed fil frame with Black when button Inter - pressed fil frame with Black when button Inter - pressed fil frame with Black when button Inter - pressed fil frame with Black when button Inter - pressed fil frame with Black when button Inter - pressed fil frame with Black when button Inter - pressed fil frame with Black when button Inter - pressed fil frame with Black y y g color Red -	2 Loops	
 Variables Functions Display I/O Time When button Enter - pressed <pre>file frame with Black -</pre>	 Variables Variables Functions Display I/O I/O Time When button filer pressed When button filer pressed If frame with Black = upsh frame When button filer pressed If frame with Black = upsh frame 	Math	
Functions Pisplay I/O Time when button left - pressed fill frame with Black - write left x 10 y: 00 color: Red - write Right - pressed write left x 10 y: 00 color: Red - write Right x 120 y: 00 color: Red - write Right - pressed write left x 10 y: 00 color: Red - write Right - pressed write left x 10 y: 00 color: Red - write Right - pressed write left x 10 y: 00 color: Red - write Right - pressed write left x 10 y: 00 color: Red - write Right - pressed write left x 10 y: 00 color: Red - write Right - pressed write left x 10 y: 00 color: Red - write Right - pressed	<pre>* Functions Functions Init forme with Black init form</pre>		push frame
 Volume vite black * Display Vo I vo i	<pre> Functions fill frame with Black -</pre>		
I VO Image: Time Push frame Image: Time Push f	I VO UO push frame push frame U Time when button Enter - pressed when button Enter - pressed fill frame with Black - fill frame with Black - when button Enter - pressed		fill frame with Black -
when button Enter - pressed when button Left - released fill frame with Black - fill frame with Black - write Enter press x 45 yr. 90 color. Red -	when button Enter - pressed when button Left - released fill frame with Black - fill frame with Black - write Enter press x 145 y: 190 color: Red -		
fill frame with Black • write 0 Enter press x \$\$ y: \$ 90 color: Red •	fill frame with Black • write 0 Enter press x 0 45 y: 0 90 color: Red • fill frame with Black • write 0 Enter press x 0 45 y: 0 90 color: Red •	Ŭ Time	
			fill frame with Black write fill frame with Black write if if arme with Black write fill frame with Black write

Finally, we'll add a press block for the back button.



When coding with the buttons, it's critical to include the "loops forever" and "scan buttons" blocks at the end to guarantee the buttons are always scanned and the code runs properly.

Click the Run button and start playing with the buttons.

Buzzzzer

Let's learn what to do with the Piezo buzzer you soldered onto your Chatter.

As the word itself says, the buzzer is used for making buzzing sounds.

We'll make a very similar sketch to the last one, but this time, pressing the buttons will trigger a particular sound to come out of the buzzer.

Let's start!

Let's start by coding what will happen when we run the code.

We want the instructions to appear on the screen.

We'll use the same code as in the last example.

Search					
🍫 Logic					
€ Loops	fill frame w	ith Purple -			
🖬 Math	write 🖡 P	ress one of the	× (30	у: 🕻 60	color: Yellow *
A Text			× (31	y: 0 71	color: Yellow -
Variables	push frame				
Functions					
🖵 Display					
I/O					
Č Time					

So we want the display to turn purple and show the message "Press one of the top row buttons!".

Now let's see what happens with the buzzer when a certain button gets pressed.

Like in the previous sketch, we'll use I/O blocks to identify what happens when specific buttons are pressed.

Search	
🍫 Logic	
8 Loops	fill frame with Purple -
🖬 Math	write Press one of the x 0 30 y: 0 60 color: Yellow -
A Text	write b top row buttons! × b 31 y: b 71 color: Yellow • push frame
Variables	pasi name
Functions	when button Left - pressed
🖵 Display	when ballon tert pressed
I/O	
Č Time	

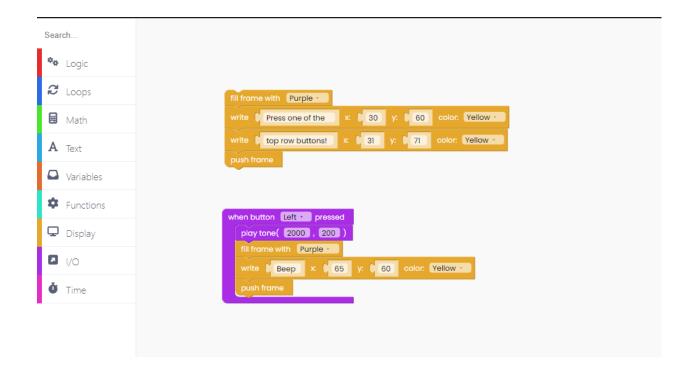
Let's introduce you to a new block called "Play tone (2000, 200)".

The first number represents the frequency, while the second number determines how long the tone will play.

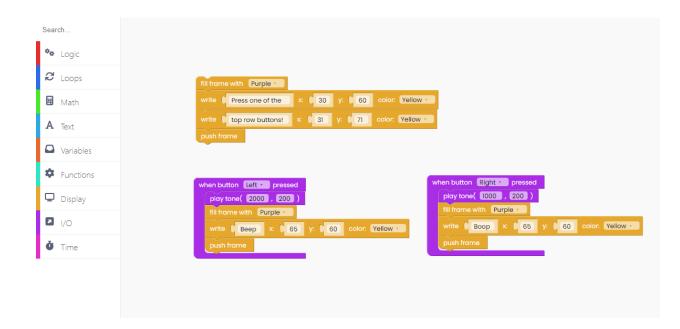
Aside from that, we want the screen to change color and to write what sound the buzzer makes.

That will be "Beep" for the first one.

This is what your code should look like by now:



Let's duplicate the I/O block and see what happens when the right button is pressed.



Duplicate it once more for the enter.

Search	
🍫 Logic	
€ Loops	fill frame with Purple
🖬 Math	write b Press one of the x: b 30 y: b 60 color: Yellow -
A Text	write to top row buttons! x: 1 31 y: 71 color: Yellow -
Variables	positions
Functions	when button Left - pressed when button Right - pressed
🖵 Display	play tone(2000 , 200)
I/O	fill frame with Purple - write b Beep x 1 65 y: 0 60 color: Yellow - write b Boop x 1 65 y: 0 60 color: Yellow -
Č Time	push frame
	when button Enter pressed play tone(1000 , 150) fill frame with Purple -
	write Buzz x: 0 65 y. 0 60 color: Yellow -
	push frame

And, for the back button.



As noted in a previous example, you must include the "loop forever" and "scan buttons" blocks at the end of any code involving buttons.



Your code is ready!

Click on the Run button and check it out.

Write on the screen

And the final sketch will be something simple and enjoyable.

That is, after the code is executed, something should be written on the screen.

Try to code the text by yourself before checking our code below.

Were you successful?

All of the blocks can be found in the "Display" block section.

This is what our code looks like:

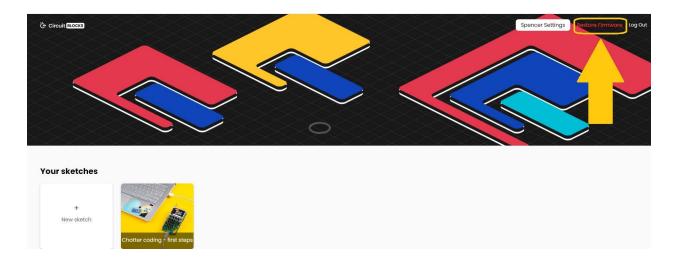
arch	
🍫 Logic	
C Loops	
Math	
A Text	
Variables	fill frame with Green
Functions	write Hello world x 10 y. 20 color: Navy
🖵 Display	push frame
I/O	
Č Time	

Click on the Run code and check it out.

Restore Chatter's firmware

If you want, for any reason, to restore Chatter's firmware, follow these steps.

This is quite simple, just connect your Chatter to the USB port of your computer and press the "Restore firmware" button on the top right.



You will be prompted with a window to choose the device you are restoring the firmware for.

Choose Chatter, of course.

XX			
R	estore Stock Firmware		
	Chatter		
	Chatter 2.0		
	Batmobile™		
	Batcontroller		
	Armstrong	•	
	CircuitPet		
	Wheelson		
	Jay-D		
•	ByteBoi		
	Synthia		
	Spencer		
	Nibble		
ite Do	Ringo		whe

Wait for a few seconds, and your Chatter will be back and running like usual.

What's next?

You've reached the end of our first Chatter coding tutorial, congratulations!



I hope you're as excited as we are about Chatter's future since there are so many cool things we want to do with it in future firmware and CircuitBlocks updates.

In the meantime, continue exploring on your own and show us what you've done with your Chatter by sharing it on the CircuitMess community forum or via 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!