



The 2020 Creative Class of the Creative Robotics Education Program

# Learning Robotics Programming with the Cheese Stick





## General Managers

Hyo Kyung Dong and Kyu Hwan Kim(Global Expansion Team Manager) at the Korea Institute for Robot Industry Advancement

## Acknowledgements

- Daegu Association of Teachers for Computing Jin Soo Kim (Daegu Yuga Elementary School)
- Jung Suh Lee (Daegu Hwanam Elementary School) Jun Hyuk Jang (Daegu Seodong Elementary School)
- Yong Yuk Jeon (Daegu Hansil Elementary School) Hyun Jae Jung (Daegu Dasa Elementary School)
- Robomation Co., Ltd.

## About This Book

1. This book is the work of the Creative Robotics Education Program within the Distribution and Expansion of Intelligent Robots Project.
2. This book aims to facilitate learning in creative classes that use the Cheese Stick Starter Kit, an educational tool developed by Robomation Co., Ltd. (Robomation was chosen as the Educational Robotics Company for the 2020 Creative Class of the Creative Robotics Education Program.)
3. This book aims to serve as a workbook for creative classes using the Cheese Stick Starter Kit. Please cite the source if you use any information in this book.
4. This book was put together with the planning of the Korea Institute of Robot Industry Advancement and the research of the Daegu Association of Teachers for Computing.
5. For any inquiries about this book, please contact the Global Expansion Team at the Korea Institute of Robot Industry Advancement at 053-210-9673.



# Contents



Cheese Stick

## 1. Playing the Piano

Cheese Stick

## 2. Making Yes or No Quiz Buttons

Cheese Stick

## 3. Creating Your Own Animation

Cheese Stick

## 4. Making an Scratch Cat Controller

Cheese Stick

## 5. Changing the Direction of a Pinwheel

Cheese Stick

## 6. Building an Emotion Expression Program

Cheese Stick

## 7. Making a Bouncing Ball Game

Cheese Stick

## 8. Adjusting the Brightness of a Lamp

Cheese Stick

## 9. Making Paper Cups Dance

Hamster

## 10. Learning about a Hamster Robot

Cheese Stick  
& Hamster

## 11. Making a Hamster Robot Controller

Cheese Stick  
& Hamster

## 12. Making a Landmine Detecting Robot

Appendix

## Decorating Paper Cups (9. Making Paper Cups Dance)

# 01 **Playing the Piano**

You can program the Cheese Stick to play the piano on your keyboard.



You can program the Cheese Stick to play the piano on your keyboard.  
Let's learn the [sound blocks] of the Cheese Stick.



Let's get ready.



Desktop or laptop



Cheese Stick



USB dongle



Let's learn the blocks for this chapter.

when  clicked

Clicking the  button runs the connected blocks.

when  key pressed

Pressing the assigned key runs the connected blocks.



play note   for  beats

This block plays the given note for the given number of beats.



rest for  beats

This block pauses the program for the given number of seconds.



change tempo by  BPM

This block changes the tempo by the given BPM.



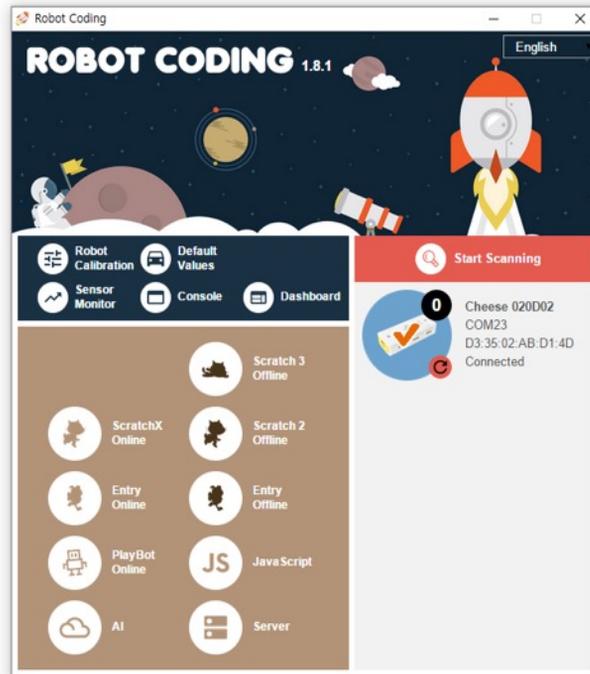
set tempo to  BPM

This block sets the tempo to the given BPM.

## 01. Playing the Piano



Now, let's install the Cheese Stick program.



Install and open the Robot Coding software on your computer. The Robot Coding software can be downloaded from [http://hamster.school/ko/download/robotcoding\\_preview.jsp](http://hamster.school/ko/download/robotcoding_preview.jsp)



Hamster Robot



Turtle Robot



The Robot Coding software allows you to control not only the Cheese Stick but also the Hamster Robot and the Turtle Robot.



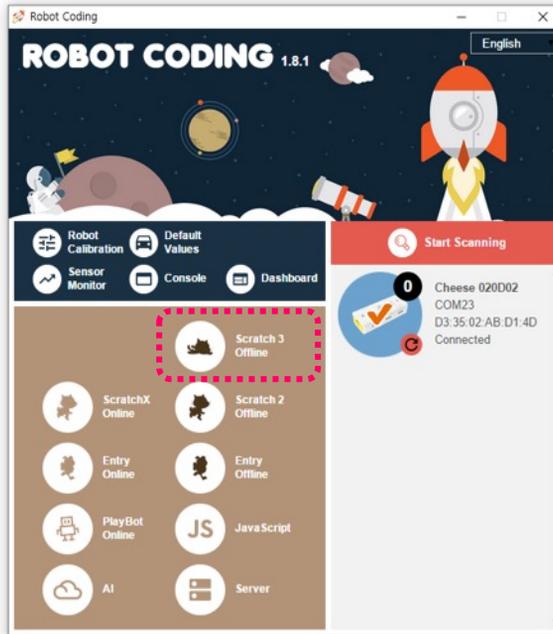
# 01. Playing the Piano



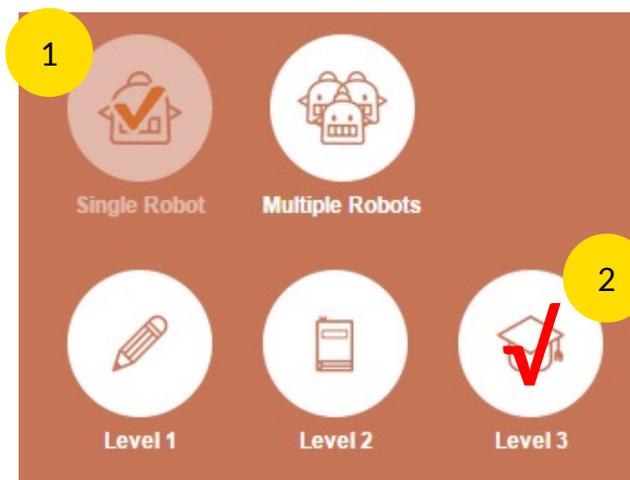
Let's start programming on the Robot Coding.



After connecting the Cheese Stick to your computer, click [Scratch 3 Offline].



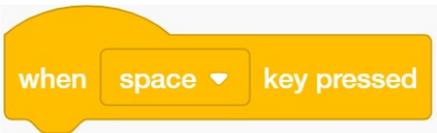
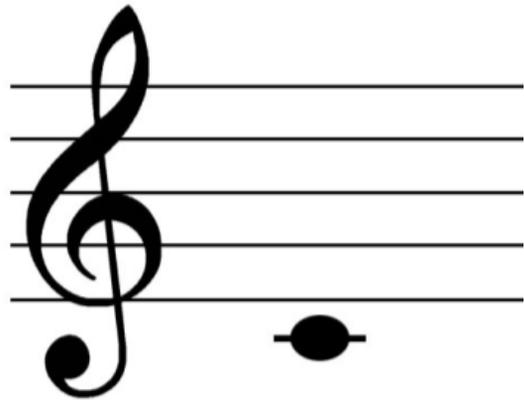
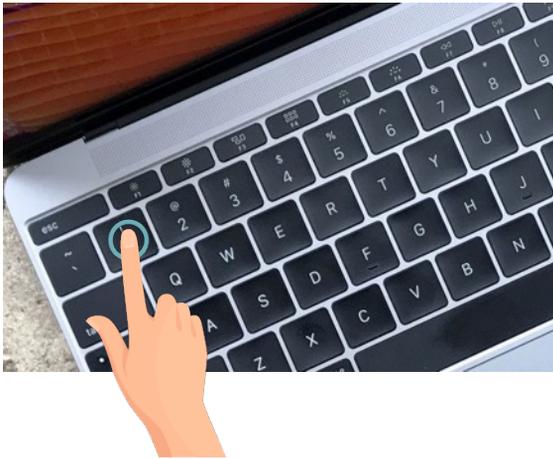
Click [Single-Robot ], then click [Level 3].



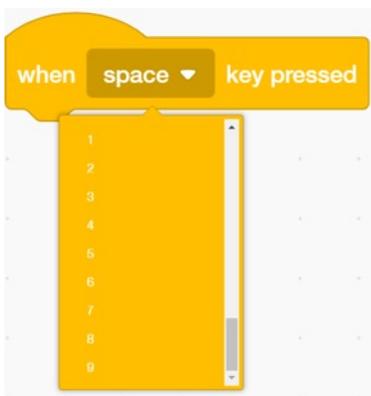
# 01. Playing the Piano



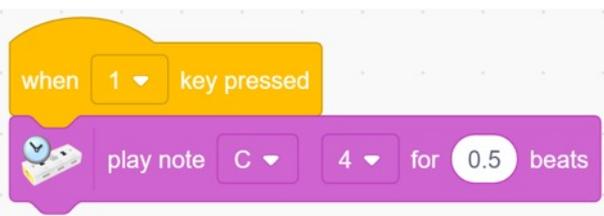
Let's program the Cheese Stick to play C when you press the 1 key on your keyboard.



Open [When clicking the space key] block under the [Events] tab.



Change [space] to [1].



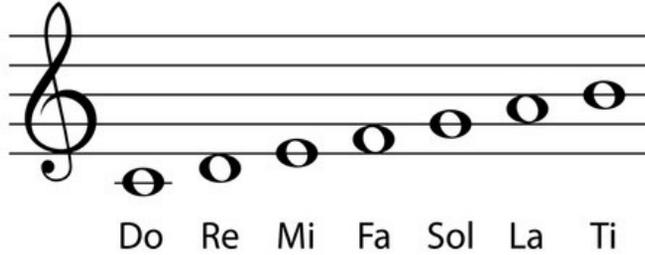
Connect [Play C4 for 0.5 beats] block under the [Cheese Stick] tab.

# 01. Playing the Piano



## [Practice Assignment 1]

Play C, D, E, F, G, A, B with the 1-7 keys on your keyboard.



## [Practice Assignment 2]

Play high C.



## Practice Assignment 1

when 1 key pressed play note C 4 for 0.5 beats

when 2 key pressed play note D 4 for 0.5 beats

when 3 key pressed play note E 4 for 0.5 beats

when 4 key pressed play note F 4 for 0.5 beats

when 5 key pressed play note G 4 for 0.5 beats

when 6 key pressed play note A 4 for 0.5 beats

when 7 key pressed play note B 4 for 0.5 beats



## Practice Assignment 2

when 8 key pressed play note C 5 for 0.5 beats

## 01. Playing the Piano



Let's program the Cheese Stick to play C, D, E, F, G, A, B, C when you click the start button.

A Scratch script starting with a yellow 'when green flag clicked' block. It is followed by eight purple 'play note' blocks. The notes and their durations are: C (4, 0.5), D (4, 0.5), E (4, 0.5), F (4, 0.5), G (4, 0.5), A (4, 0.5), B (4, 0.5), and C (5, 0.5).



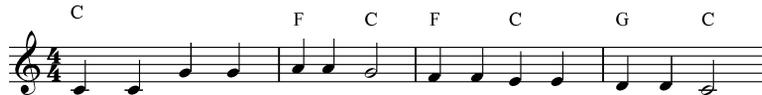
The following chart shows commonly used notes and rests.

Notes or Rests	Name	Beat(s)
	Eighth note	0.5 beats
	Quarter note	1 beat
	Dotted quarter note	1.5 beats
	Half note	2 beats
	Eighth rest	0.5 beats
	Quarter rest	1 beat

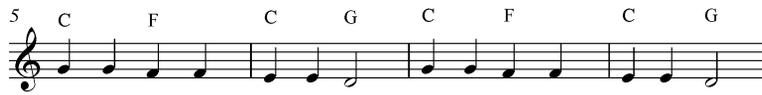


Review this chapter with the following assignment.

## Twinkle, Twinkle Little Star



1. Twin - kle, twin - kle lit - tle star, how I won - der what you are.  
 2. When the blaz - ing sun is gone, when he noth - ing shines up - on.  
 3. As your bright and ti - ny spark, lights the trav' ller in the dark.



Up a - bove the world so high, like a dia - mond in the sky.  
 Then you show your lit - tle light, twin - kle, twin - kle all the night.  
 Though I know not what you are, twin - kle, twin - kle lit - tle star.



Twin - kle, twin - kle lit - tle star, how I won - der what you are.



## [Practice Assignment 1]

Enter the following commands to play this song.

when 1 key pressed  
 play note C 4 for 0.5 beats

when 2 key pressed  
 play note D 4 for 0.5 beats

when 3 key pressed  
 play note E 4 for 0.5 beats

when 4 key pressed  
 play note F 4 for 0.5 beats

when 5 key pressed  
 play note G 4 for 0.5 beats

when 6 key pressed  
 play note A 4 for 0.5 beats

when 7 key pressed  
 play note B 4 for 0.5 beats

when 8 key pressed  
 play note C 5 for 0.5 beats

# 01. Playing the Piano



Here is another assignment.

I won - der    I won - der    What are the    same  
I won - der    I won - der    What are the    same

Two chop - sticks in    a set    they are the    same  
Four Yut - sticks in    a set    they are the    same



[Practice Assignment 2] Enter the following commands and program the Cheese Stick to play this song when you click the start button.

when clicked

set tempo to 120 BPM

play note C 4 for 1 beats

play note E 4 for 1 beats

play note G 4 for 1 beats

play note C 4 for 1 beats

play note E 4 for 1 beats

play note G 4 for 1 beats

play note A 4 for 1 beats

play note G 4 for 2 beats

You can also use [Repeat] block.

when clicked

set tempo to 120 BPM

repeat 2

play note C 4 for 1 beats

play note E 4 for 1 beats

play note G 4 for 1 beats

play note A 4 for 1 beats

play note A 4 for 1 beats

play note A 4 for 1 beats

play note G 4 for 2 beats

# 02 Making Yes or No Quiz Buttons

You can make Yes or No quiz buttons with a 5x5 LED Matrix.



You can make Yes or No quiz buttons with two buttons on a 5x5 LED Matrix. Let's learn about a 5x5 LED Matrix.



Let's get ready.



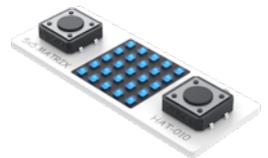
Desktop or laptop



Cheese Stick



USB dongle



5X5 LED Matrix

## 02. Making Yes or No Quiz Buttons



Let's learn the blocks for this chapter.



Clicking the  button runs the connected blocks.



This block infinitely repeats the blocks inside it.



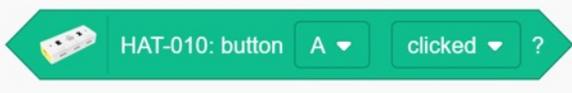
The blocks inside this block will run if the condition is true.



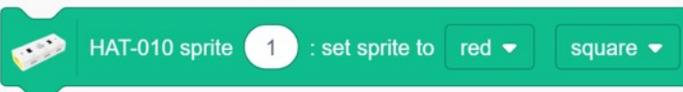
This block evaluates to true if both conditions are true.



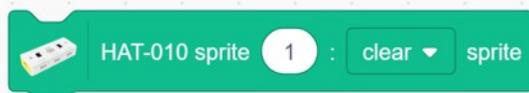
This block is needed to use the 5x5 LED Matrix.



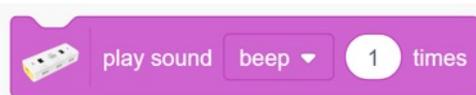
This block is needed to press the buttons on the 5x5 LED Matrix.



This block displays different figures on the 5x5 LED Matrix.



This block deletes the figures displayed on the 5x5 Matrix.



This block plays the given sound for the given number of times.

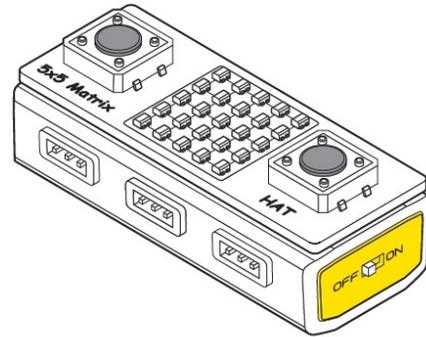
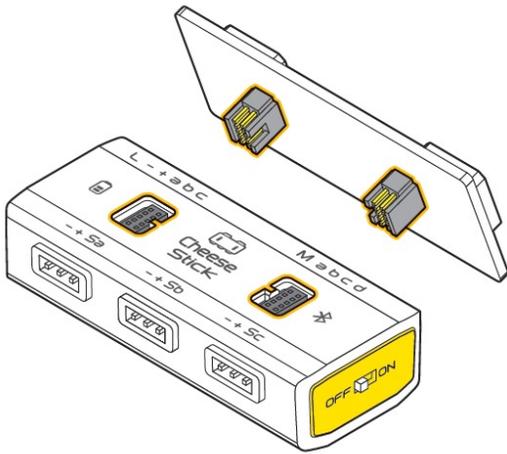
## 02. Making Yes or No Quiz Buttons



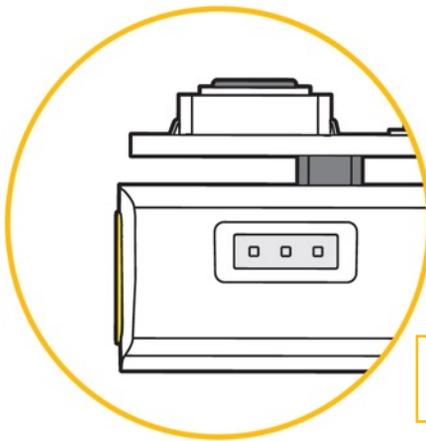
Let's connect the 5x5 LED Matrix.



Plug the 5x5 LED Matrix into the HAT port on the Cheese Stick.



The 5x5 LED Matrix can be plugged into the Cheese Stick in any direction.



<Side view of the LED Matrix plugged into the Cheese Stick>

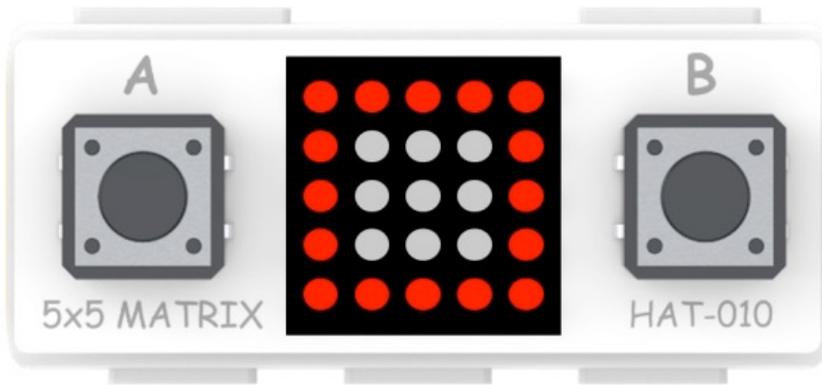


When the 5X5 LED Matrix is plugged into the HAT port, it is normal that there is a small gap between them.

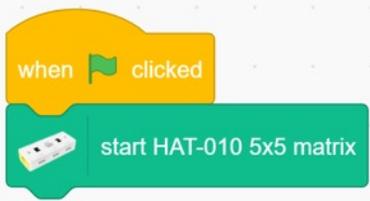
## 02. Making Yes or No Quiz Buttons



Let's program the 5x5 LED Matrix to display a square.



Open [When  clicked] block under the [Events] tab.



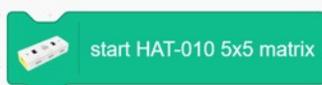
Connect [Start HAT-010 5x5 Matrix] block under the [HAT-010] tab.



Connect [Set HAT-010 figure 1 to red square] block under the [Hardware] tab.



You need



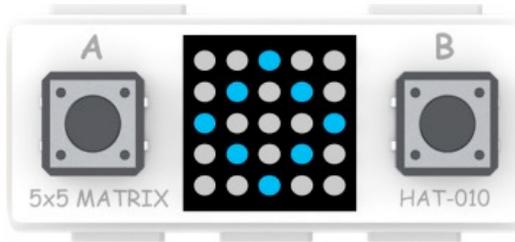
block to use the 5x5 LED Matrix.

## 02. Making Yes or No Quiz Buttons



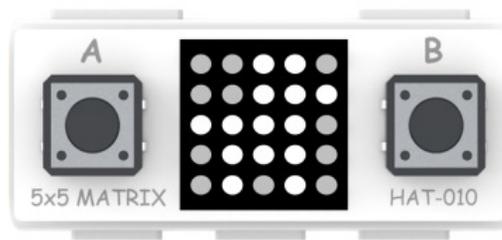
### [Practice Assignment 1]

Display a blue diamond on the 5x5 LED Matrix.

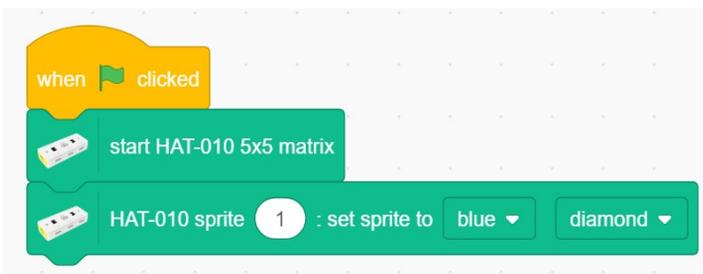


### [Practice Assignment 2]

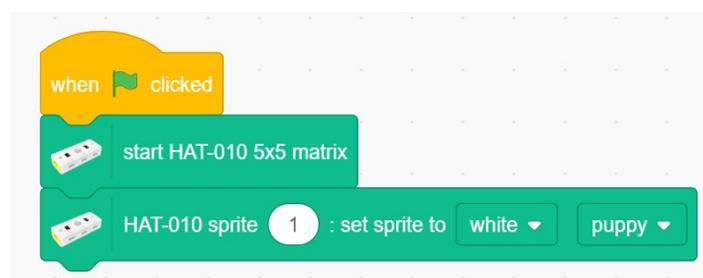
Display a white dog on the 5x5 LED Matrix.



### Practice Assignment 1



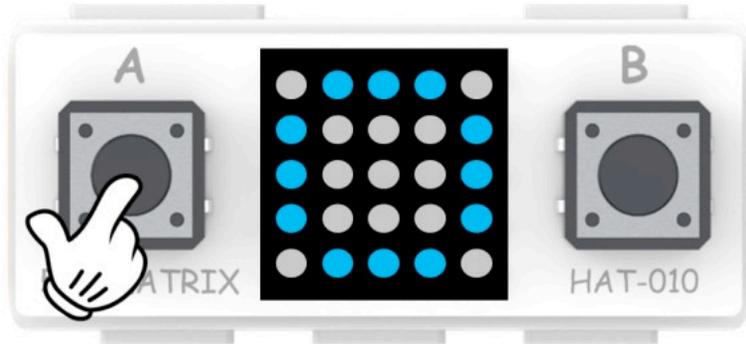
### Practice Assignment 2



## 02. Making Yes or No Quiz Buttons



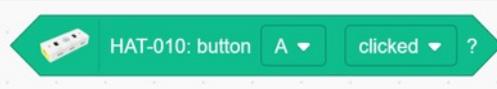
Let's program the LED Matrix to display a blue O when you press button A.



Use



and

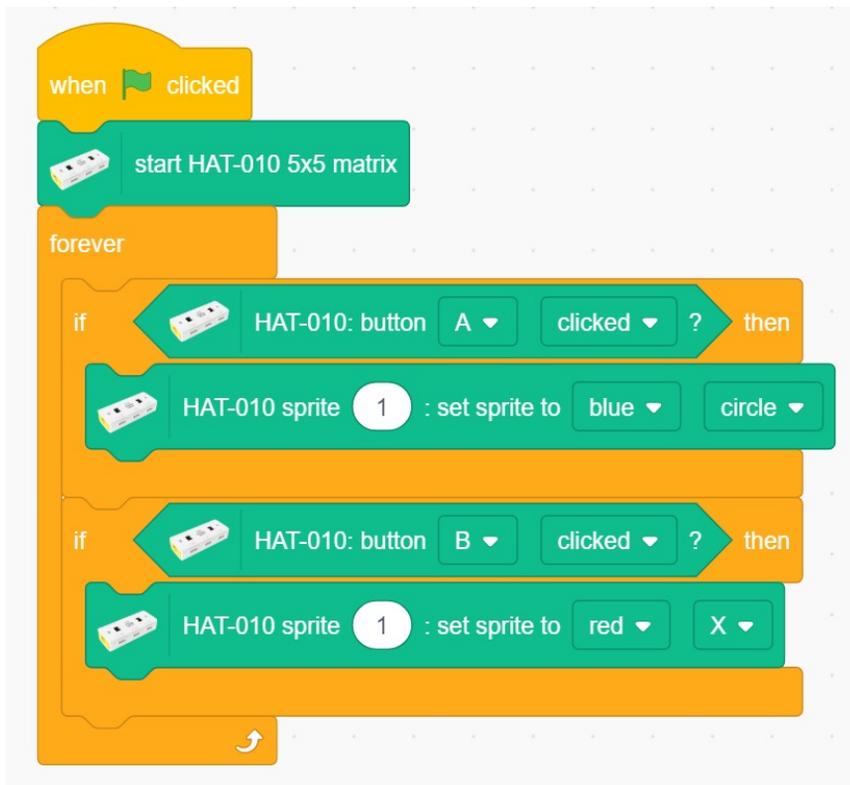
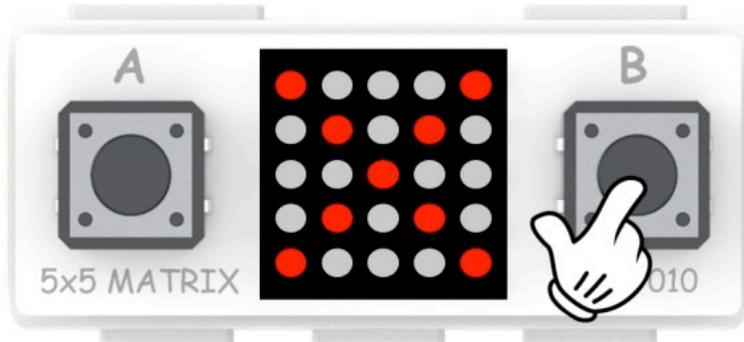


blocks.

## 02. Making Yes or No Quiz Buttons



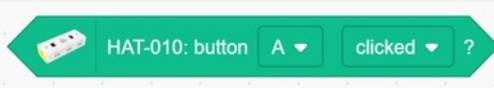
Let's program the LED Matrix to display a red X when you press button B.



Use



and

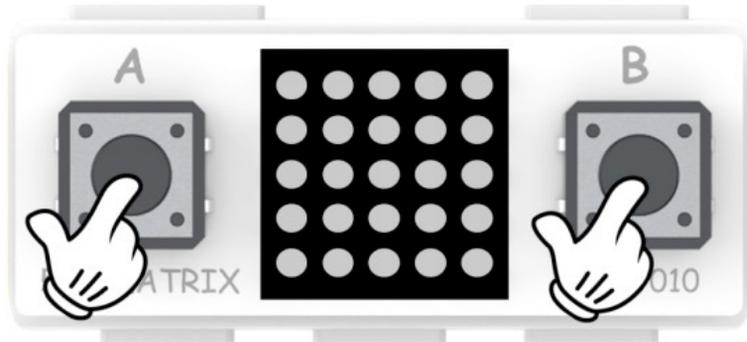


blocks.

## 02. Making Yes or No Quiz Buttons



Let's program the 5x5 LED Matrix to remove the O or X when you press both buttons(A and B) together.



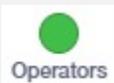
```
when clicked
  start HAT-010 5x5 matrix
  forever
    if HAT-010: button A clicked ? then
      HAT-010 sprite 1 : set sprite to blue circle
    if HAT-010: button B clicked ? then
      HAT-010 sprite 1 : set sprite to red X
    if HAT-010: button A clicked ? and HAT-010: button B clicked ? then
      HAT-010 sprite 1 : clear sprite
```



Use



block under the

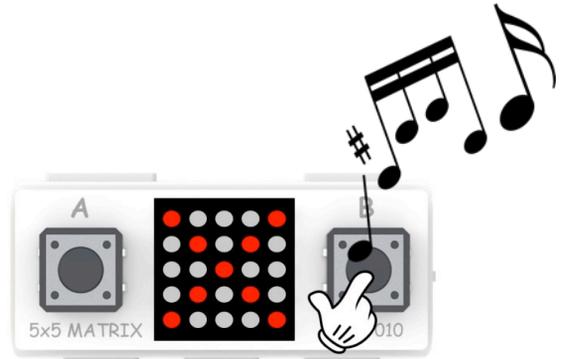
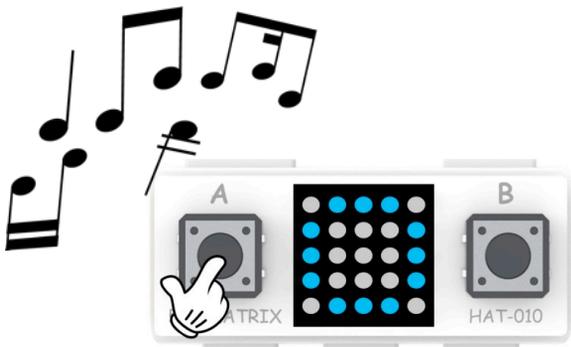


tab.

## 02. Making Yes or No Quiz Buttons



Let's program the LED Matrix to play the given sound in addition to displaying an O or X.



```
when clicked
  start HAT-010 5x5 matrix
  forever
    if HAT-010: button A clicked ? then
      play sound beep 1 times
      HAT-010 sprite 1 : set sprite to blue circle
    if HAT-010: button B clicked ? then
      play sound beep 1 times
      HAT-010 sprite 1 : set sprite to red X
    if HAT-010: button A clicked ? and HAT-010: button B clicked ? then
      play sound good job 1 times
      HAT-010 sprite 1 : clear sprite
```



Use



block under the



tab.

## 02. Making Yes or No Quiz Buttons



Let's review this chapter with the following assignment.



[Practice Assignment 1]

Play Yes or No quiz game with the Cheese Stick buttons.

<Learning Distancing Rules In Daily Life>

1. Go out when you have respiratory symptoms like fever, cough, mucus, and muscle pain.

**NO**

Stay home for three to four days.

2. Keep a distance of two arms' length between people.

**YES**

Keeping a distance of two meters between people reduces the infection rate as COVID-19 is mainly transmitted by droplets.

3. Wash your hands thoroughly and frequently and cough into your sleeve.

**YES**

Washing hands prevents the virus on your hands from entering your body, and coughing into your sleeve minimizes the transmission of COVID-19 through droplets.

4. There is no need to ventilate and disinfect often.

**NO**

Ventilate at least twice a day and disinfect regularly.

5. Maintain physical distancing but remain connected.

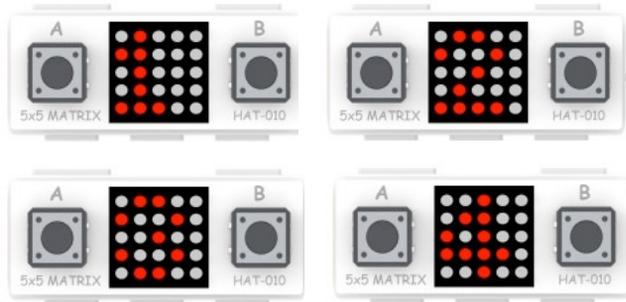
**YES**

Each and every one of us has a role to play in slowing down the spread of COVID-19.

## 02. Making Yes or No Quiz Buttons



Here is another assignment.



[Practice Assignment 2]

Enter the following commands to make multiple choice quiz buttons.

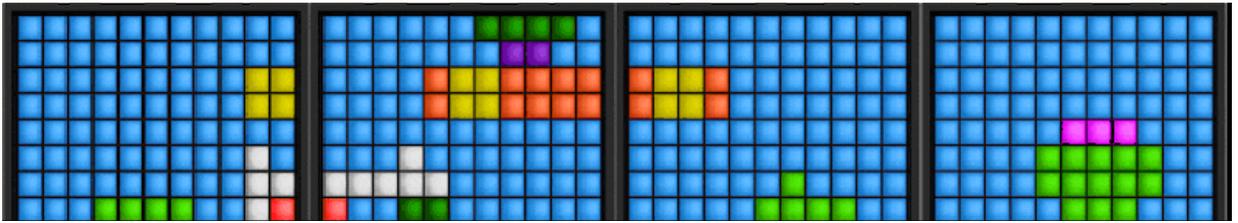
```
when clicked
  start HAT-010 5x5 matrix
  forever
    if key 1 pressed? then
      play sound beep 1 times
      HAT-010 sprite 1 : set sprite to red string 1
    if key 2 pressed? then
      play sound beep 1 times
      HAT-010 sprite 1 : set sprite to red string 2
    if key 3 pressed? then
      play sound beep 1 times
      HAT-010 sprite 1 : set sprite to red string 3
    if key 4 pressed? then
      play sound beep 1 times
      HAT-010 sprite 1 : set sprite to red string 4
    if key space pressed? then
      play sound good job 1 times
      HAT-010 sprite 1 : clear sprite
```

# 03 Creating Your Own Animation

You can create your own animation effects with the 5x5 LED Matrix.



You can create your own animation effects with the 5x5 LED Matrix.  
Let's learn how to program the 25 LEDs.



Let's get ready.



Desktop or laptop



Cheese Stick



USB dongle



5X5 LED Matrix

### 03. Creating Your Own Animation



Let's learn the blocks for this chapter.



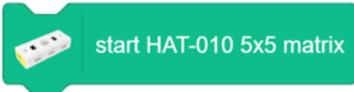
Clicking the  button runs the connected blocks.



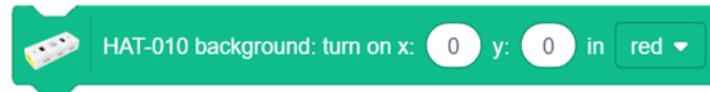
This block infinitely repeats the blocks inside it.



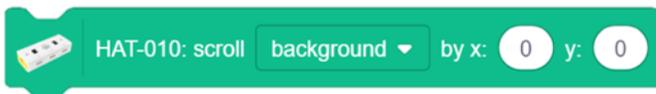
This block pauses the program for the given number of seconds before running the next block.



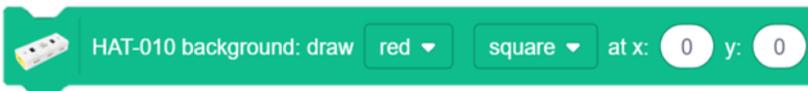
This block is needed to use the 5x5 LED Matrix.



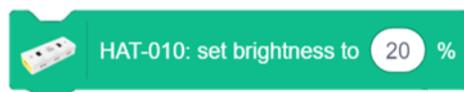
This block lights up the given x- and y-coordinates on the LED.



This block moves the background by the given x and y values.



This block displays the given background at the given x- and y-coordinates.



This block sets the brightness of the 5x5 LED Matrix to the given value.

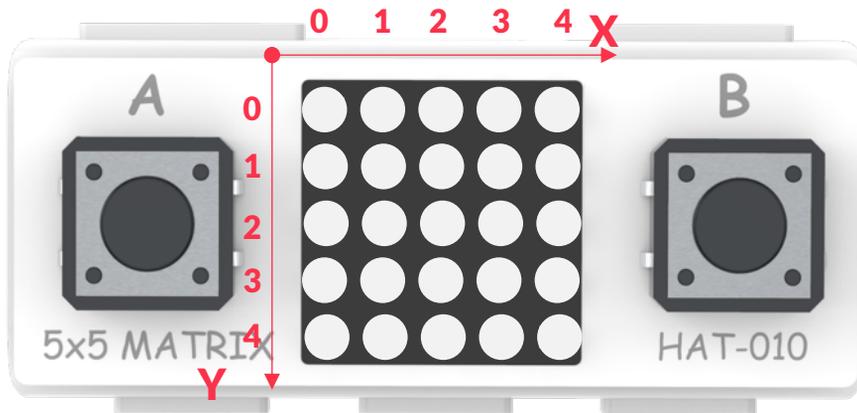
### 03. Creating Your Own Animation



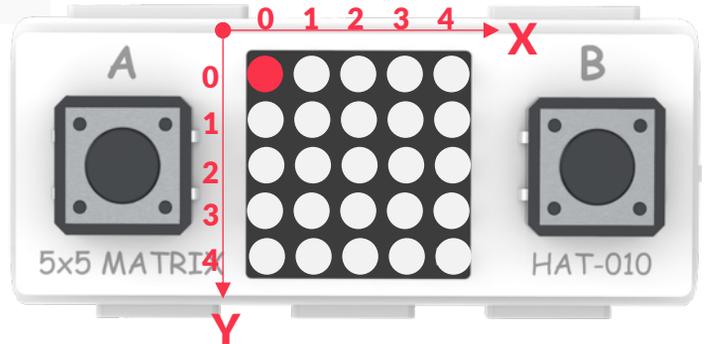
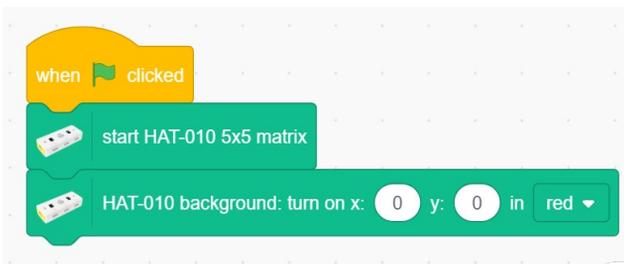
Let's program each LED individually.



The 5x5 LED Matrix has 25 LEDs.



You can control the brightness and color of each of the 25 LEDs individually with its x- and y-coordinates.



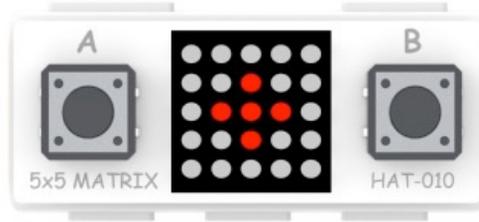
block can light up the given colour at the given

x- and y-coordinates.



#### [Practice Assignment 1]

Program the 5x5 LED Matrix to display the following red figure.



#### [Practice Assignment 2]

Program the 5x5 LED Matrix to display the following multi-colored figure.



#### [Practice Assignment 1]



#### [Practice Assignment 2]

when clicked

- start HAT-010 5x5 matrix
- HAT-010 background: turn on x: 1 y: 2 in red
- HAT-010 background: turn on x: 2 y: 1 in red
- HAT-010 background: turn on x: 2 y: 2 in red
- HAT-010 background: turn on x: 2 y: 3 in red
- HAT-010 background: turn on x: 3 y: 2 in red

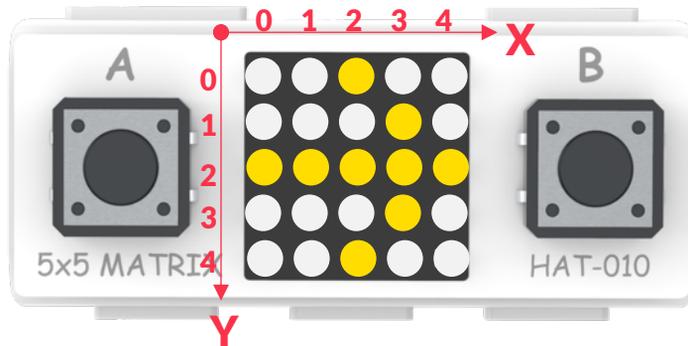
when clicked

- start HAT-010 5x5 matrix
- HAT-010 background: turn on x: 1 y: 2 in orange
- HAT-010 background: turn on x: 2 y: 1 in yellow
- HAT-010 background: turn on x: 2 y: 2 in green
- HAT-010 background: turn on x: 2 y: 3 in blue
- HAT-010 background: turn on x: 3 y: 2 in purple

### 03. Creating Your Own Animation



Let's display a right arrow on the 5x5 LED Matrix.

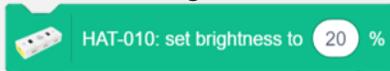


when clicked

- start HAT-010 5x5 matrix
- HAT-010: set brightness to 50 %
- HAT-010 background: turn on x: 0 y: 2 in yellow
- HAT-010 background: turn on x: 1 y: 2 in yellow
- HAT-010 background: turn on x: 2 y: 0 in yellow
- HAT-010 background: turn on x: 2 y: 2 in yellow
- HAT-010 background: turn on x: 2 y: 4 in yellow
- HAT-010 background: turn on x: 3 y: 1 in yellow
- HAT-010 background: turn on x: 3 y: 2 in yellow
- HAT-010 background: turn on x: 3 y: 3 in yellow
- HAT-010 background: turn on x: 4 y: 2 in yellow



The default brightness of the LED is 20%. You can adjust the brightness with

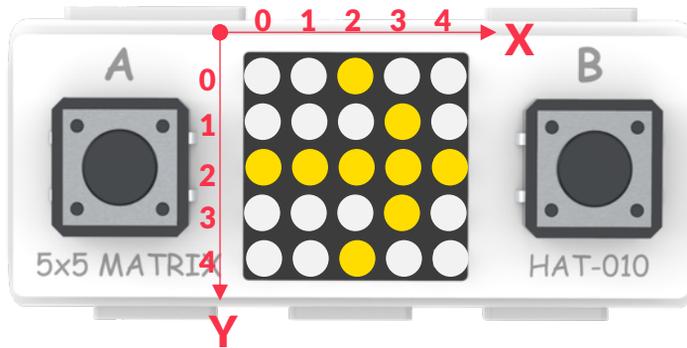


block. Make sure you are not turning up the brightness level too high. Exposures to high-intensity LED light can hurt your eyes.

### 03. Creating Your Own Animation



Let's learn another way of displaying a right arrow on the 5x5 LED Matrix.



when clicked

- start HAT-010 5x5 matrix
- HAT-010 background: draw yellow pattern 00100 at x: 0 y: 0
- HAT-010 background: draw yellow pattern 00010 at x: 0 y: 1
- HAT-010 background: draw yellow pattern 11111 at x: 0 y: 2
- HAT-010 background: draw yellow pattern 00010 at x: 0 y: 3
- HAT-010 background: draw yellow pattern 00100 at x: 0 y: 4



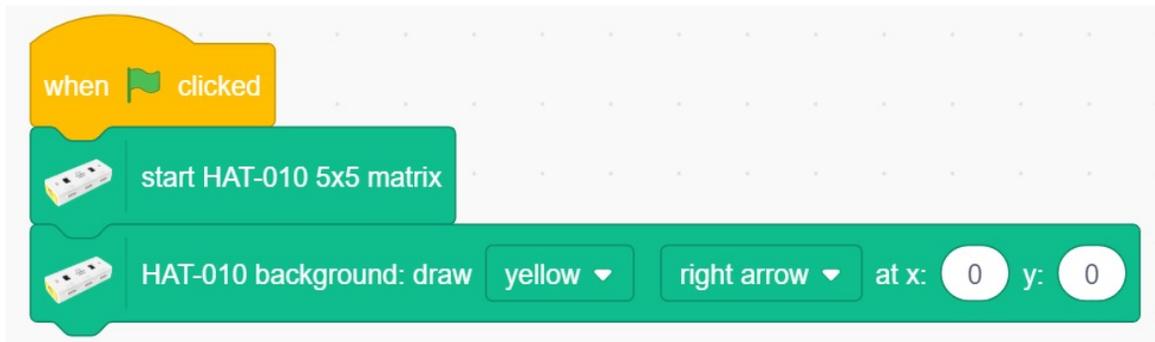
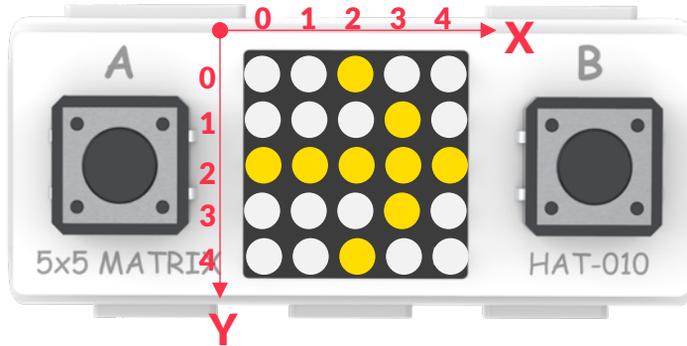
Pattern blocks can turn the LEDs on and off using the binary language of computers. Use five-bit binary numbers instead of x- and y-coordinates to display backgrounds.

HAT-010 background: draw yellow pattern 10010 at x: 0 y: 4

### 03. Creating Your Own Animation

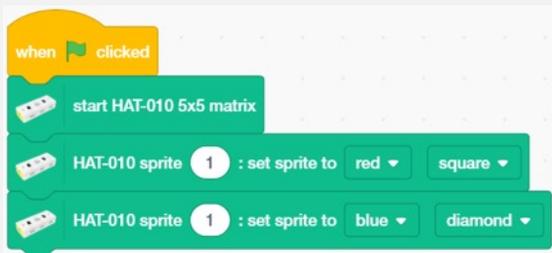


There is one more way of displaying a right arrow on the LED Matrix.

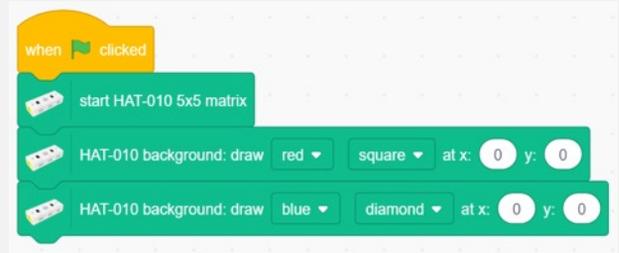


Unlike figure blocks, background blocks can overlap many backgrounds, or layers, and display them as a single image.

<figure blocks>



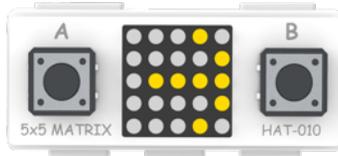
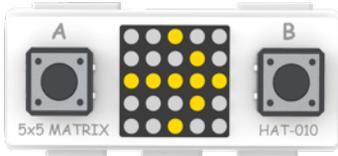
<background blocks>



### 03. Creating Your Own Animation



Let's program the 5x5 LED Matrix to display an emergency exit with a moving arrow.



```
when clicked
  start HAT-010 5x5 matrix
  forever
    HAT-010 background: draw yellow right arrow at x: 0 y: 0
    repeat 6
      wait 0.2 seconds
      HAT-010: scroll background by x: 1 y: 0
```



To program the arrow to move horizontally or vertically, Use

```
repeat 10, wait 2 seconds
```

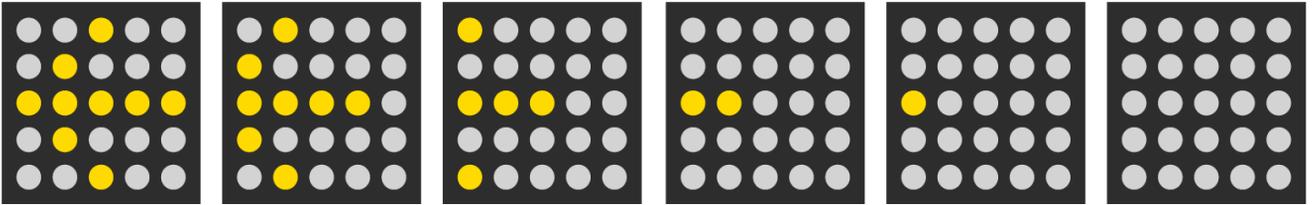
, and

```
HAT-010: scroll background by x: 0 y: 0
```

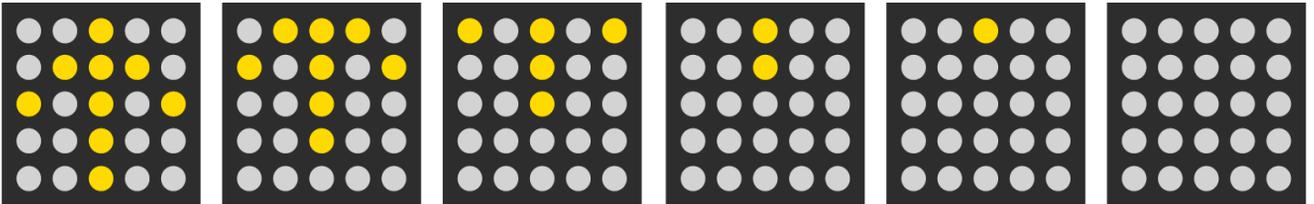
### 03. Creating Your Own Animation



[Practice Assignment 1] Create an arrow moving left.



[Practice Assignment 2] Create an arrow moving upward.



[Practice Assignment 1]



[Practice Assignment 2]

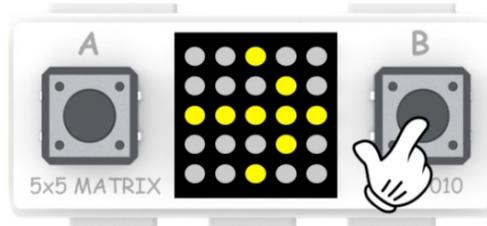
```
when clicked
  start HAT-010 5x5 matrix
  forever
    HAT-010 background: draw yellow left arrow at x: 0 y: 0
    repeat 6
      wait 0.2 seconds
      HAT-010: scroll background by x: -1 y: 0
```

```
when clicked
  start HAT-010 5x5 matrix
  forever
    HAT-010 background: draw yellow up arrow at x: 0 y: 0
    repeat 6
      wait 0.2 seconds
      HAT-010: scroll background by x: 0 y: -1
```

### 03. Creating Your Own Animation

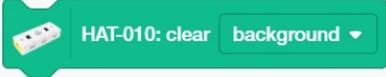


Let's program the 5x5 LED Matrix to move the arrow left when you press button A and right when you press button B.



```
when clicked
  start HAT-010 5x5 matrix
  forever
    if HAT-010: button A clicked ? then
      HAT-010 background: draw yellow left arrow at x: 0 y: 0
      repeat 6
        wait 0.2 seconds
        HAT-010: scroll background by x: -1 y: 0
      HAT-010: clear background
    if HAT-010: button B clicked ? then
      HAT-010 background: draw yellow right arrow at x: 0 y: 0
      repeat 6
        wait 0.2 seconds
        HAT-010: scroll background by x: 1 y: 0
      HAT-010: clear background
```



After displaying one arrow, use  block to display another one.

### 03. Creating Your Own Animation



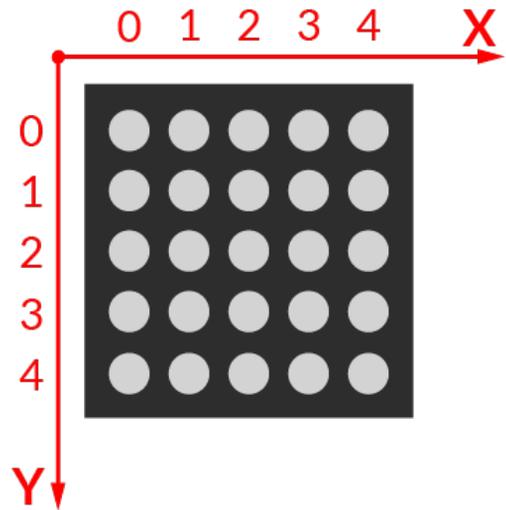
Let's review this chapter with the following assignments.



#### [Practice Assignment 1]

Enter the following commands and color the LEDs below.

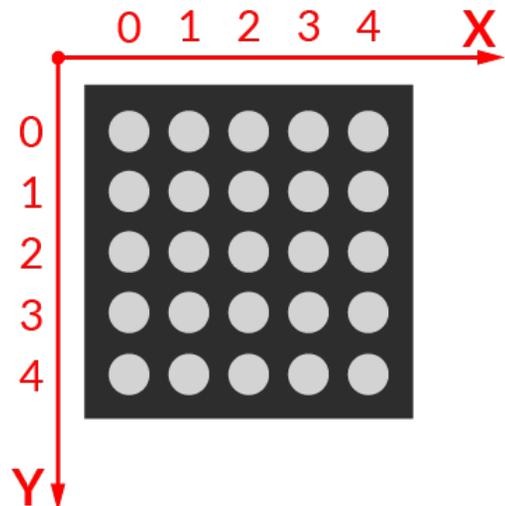
```
when clicked
  start HAT-010 5x5 matrix
  HAT-010 background: turn on x: 0 y: 3 in red
  HAT-010 background: turn on x: 1 y: 2 in red
  HAT-010 background: turn on x: 1 y: 3 in red
  HAT-010 background: turn on x: 2 y: 1 in red
  HAT-010 background: turn on x: 2 y: 3 in red
  HAT-010 background: turn on x: 3 y: 2 in red
  HAT-010 background: turn on x: 3 y: 3 in red
  HAT-010 background: turn on x: 4 y: 3 in red
```



#### [Practice Assignment 2]

Enter the following commands and color the LEDs below.

```
when clicked
  start HAT-010 5x5 matrix
  HAT-010 background: draw red square at x: 1 y: 1
```



### 03. Creating Your Own Animation



#### [Practice Assignment 3]

Enter the following commands to make a Tetris Game Dance.



when clicked

start HAT-010 5x5 matrix

forever

- HAT-010 background: draw  pattern  at x:  y:

wait  seconds

- HAT-010 background: draw  pattern  at x:  y:

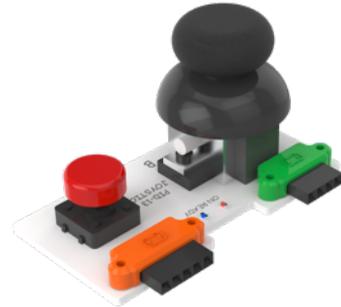
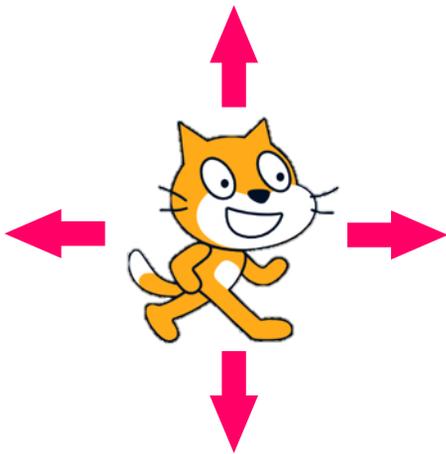
wait  seconds

# 04 Making a Scratch Cat Controller

You can move the Scratch sprite with a joystick.



You can move the Scratch sprite with a joystick. Let's learn about joysticks.



Let's get ready.



Cheese Stick

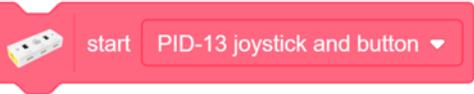


Joystick

## 04. Making a Scratch Cat Controller



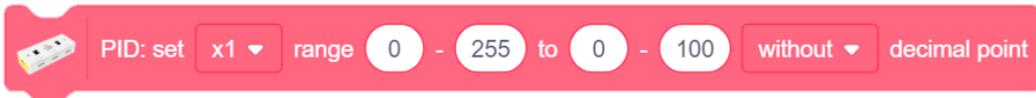
Let's learn the blocks for this chapter.



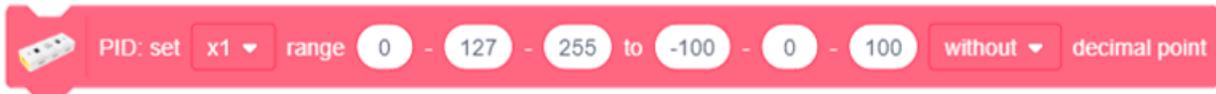
This block is needed to use a joystick.



This block sets the horizontal and vertical values of a joystick.



This block sets the range of joystick values as 0 to 100.



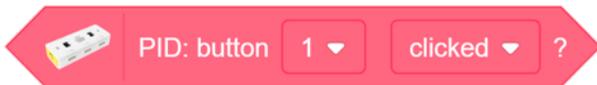
This block divides joystick values into three different ranges.



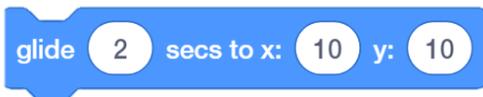
This block compares two values to run a command.



This block evaluates to true if one or both conditions are true.



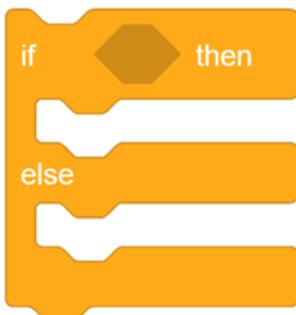
This block runs the command when the button is pressed.



This block moves the Sprites by the given x and y values for the given number of seconds.



This block changes the x-coordinate by the given value.



This block runs different commands depending on whether the condition is true or not.



Let's learn how to connect a joystick and check its value



Plug the joystick into the Cheese Stick as shown below.



You can use the joystick and its button to control the Scratch sprite's motions and movements.



The X-coordinate gradually decreases to 0.

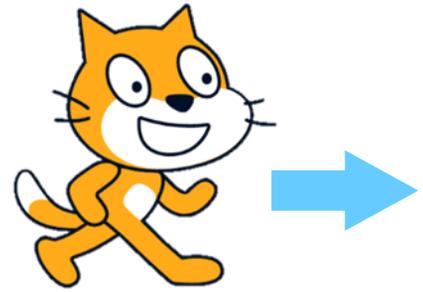
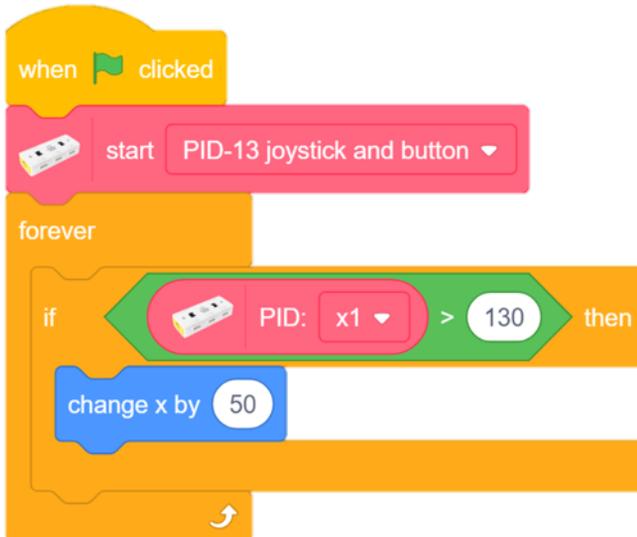


The X-coordinate gradually increases to 255.  
(The maximum value may vary for other devices.)

## 04. Making a Scratch Cat Controller



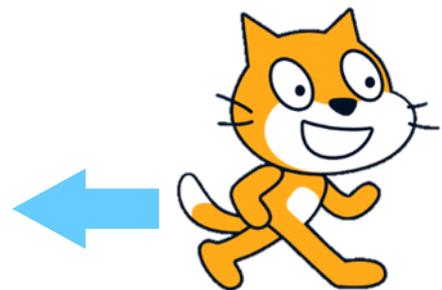
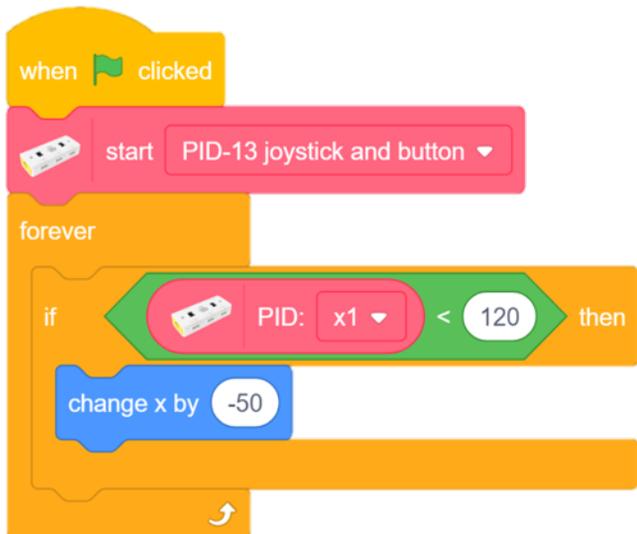
Let's move the Scratch Cat forward with a joystick.



move forward



Let's move the Scratch Cat backward with a joystick.



move backward



You need



block to use a joystick.

## 04. Making a Scratch Cat Controller



[Practice Assignment 1]



[Practice Assignment 2]



move upward



move downward



[Practice Assignment 1]



[Practice Assignment 2]

when clicked

start PID-13 joystick and button ▾

forever

if PID:  >  then

change y by

when clicked

start PID-13 joystick and button ▾

forever

if PID:  <  then

change y by



Make sure you know the difference between these two blocks.

change x by

change y by

If the condition is true, this block moves the sprite horizontally by the given x value and vertically by the given y value for the given number of seconds.

glide  secs to x:  y:

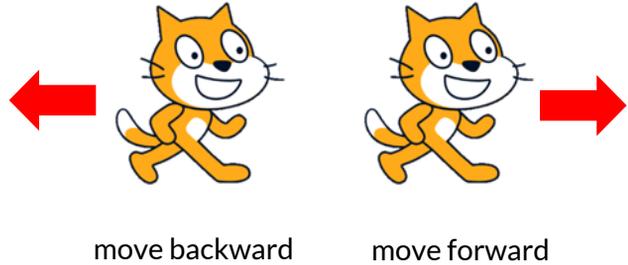
If the condition is true, this block moves the sprite to the given x- and y- coordinates for the given number of seconds.

## 04. Making a Scratch Cat Controller



Let's program the joystick to move the Scratch Cat back and forth.

```
when clicked
  start PID-13 joystick and button
  PID: set x1 range 0 - 255 to 0 - 100 without decimal point
  forever
    if PID: x1 > 50 or PID: y1 = 50 then
      change x by 10
    else
      change x by -10
```



Now, let's program the Scratch Cat to return to its starting position when it touches the screen borders.

```
when clicked
  start PID-13 joystick and button
  PID: set x1 range 0 - 255 to 0 - 100 without decimal point
  forever
    if PID: x1 > 50 or PID: y1 = 50 then
      change x by 10
    else
      change x by -10
    if touching edge ? then
      go to x: 0 y: 0
```



You need



block to use a joystick.

## 04. Making a Scratch Cat Controller



Let's program the joystick to move the Scratch Cat in three horizontal directions.

```
when clicked
  start PID-13 joystick and button
  PID: set x1 range 0 - 127 - 255 to -100 - 0 - 100 without decimal point
  forever
    if -10 < PID: x1 and PID: x1 < 10 then
      change x by 0
    else
      if PID: x1 > 10 then
        change x by 10
      else
        change x by -10
```



Let's program the joystick to move the Scratch Cat in three vertical directions.

```
when clicked
  PID: set y1 range 0 - 127 - 255 to -100 - 0 - 100 without decimal point
  forever
    if -50 < PID: y1 and PID: y1 < 10 then
      change y by 0
    else
      if PID: y1 > 50 then
        change y by 10
      else
        change y by -10
```

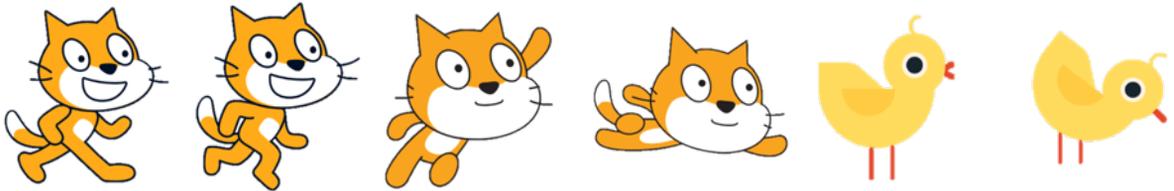


## 04. Making a Scratch Cat Controller



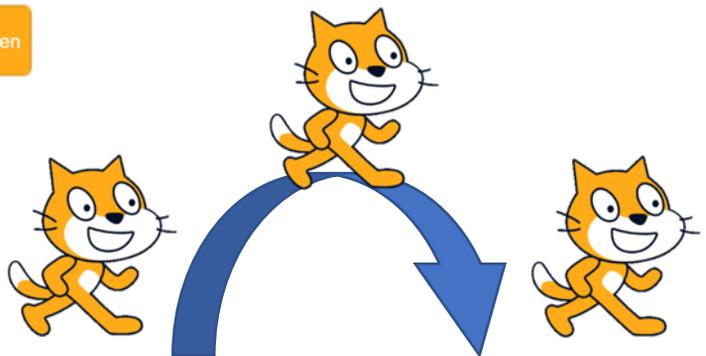
Let's program the joystick button to change the Scratch Cat's motions.

```
when green flag clicked
  start PID-13 joystick and button
  forever
    if PID: button 2 clicked ? then
      next costume
```



Let's program the joystick button to make the Scratch Cat jump.

```
when green flag clicked
  start PID-13 joystick and button
  forever
    if PID: button 2 clicked ? then
      repeat 10
        change x by 2
        change y by 2
      repeat 10
        change x by 2
        change y by -2
```



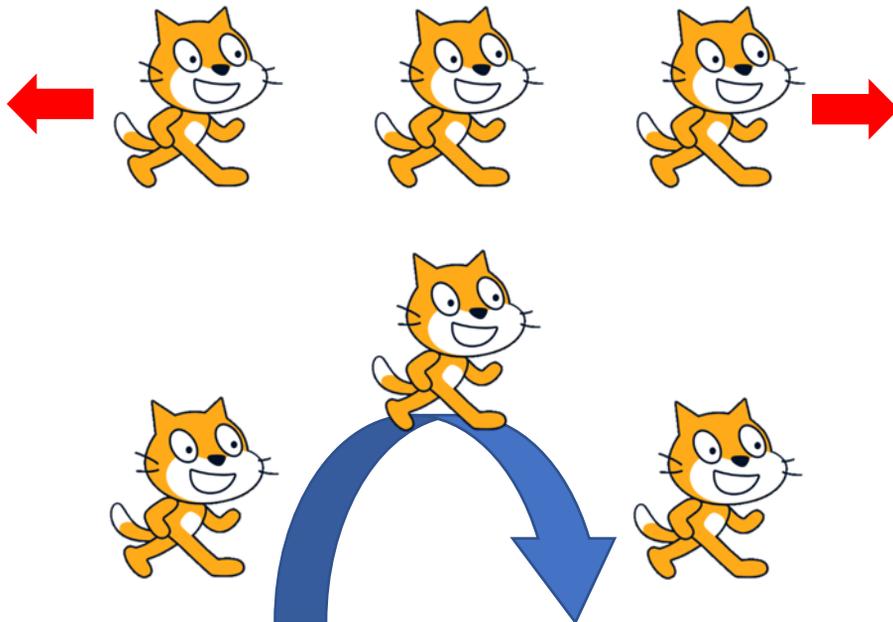
## 04. Making a Scratch Cat Controller



Let's program the joystick button to change the Scratch Cat's motions.

```
when clicked
  start PID-13 joystick and button
  PID: set x1 range 0 - 127 - 255 to -100 - 0 - 100 without decimal point
  forever
    if (-10 < PID: x1 and PID: x1 < 10) then
      change x by 0
    else
      if PID: x1 > 10 then
        change x by 10
      else
        change x by -10
```

```
when clicked
  start PID-13 joystick and button
  forever
    if PID: button 2 clicked ? then
      repeat 10
        change x by 2
        change y by 2
      repeat 10
        change x by -2
        change y by -2
```



To run two commands simultaneously, both command blocks need to be connected to



block.

## 04. Making a Scratch Cat Controller



Let's review this chapter with the following assignments.



[Assignment 1] Enter the following commands. Then, write how the Scratch Cat moves.

```
when clicked
  start PID-13 joystick and button
  PID: set x1 range 0 - 127 - 255 to -100 - 0 - 100 without decimal point
  forever
    if (-10 < PID: x1 and PID: x1 < 10) then
      change x by 0
    else
      if PID: x1 > 10 then
        change x by 10
      else
        change x by -10
```



[Assignment 2] Enter the following commands. Then, write how the Scratch Cat moves.

```
when clicked
  start PID-13 joystick and button
  forever
    if PID: button 2 clicked ? then
      glide 2 secs to x: 10 y: 10
```

## 04. Making a Scratch Cat Controller



[Assignment 3] You want to program the Scratch Cat to move back and forth and change its motions when you press the joystick button. Check the blocks you need to run these two commands simultaneously.

```
when clicked
  start PID-13 joystick and button
  PID set x1 range 0 127 255 to -100 0 100 without decimal point
  forever
    if <-10 < PID: x1 > and <PID: x1 < -10 > then
      change x by 0
    else
      if <PID: x1 > -10 > then
        change x by 10
      else
        change x by -10
```

( )

```
when clicked
  PID: set y1 range 0 127 255 to -100 0 100 without decimal point
  forever
    if <50 < PID: y1 > and <PID: y1 < -10 > then
      change y by 0
    else
      if <PID: y1 > 50 > then
        change y by 10
      else
        change y by -10
```

( )

```
when clicked
  start PID-13 joystick and button
  forever
    if <PID: button 2 > clicked > ? > then
      repeat 10
        change x by 2
        change y by 2
      repeat 10
        change x by -2
        change y by -2
```

( )

```
when clicked
  start PID-13 joystick and button
  forever
    if <PID: button 2 > clicked > ? > then
      next costume
```

( )

```
when clicked
  start PID-13 joystick and button
  PID: set x1 range 0 255 to 0 100 without decimal point
  forever
    if <PID: x1 > 50 > or <PID: y1 > = 50 > then
      change x by 10
    else
      change x by -10
    if <touching edge > ? > then
      go to x: 0 y: 0
```

( )

```
when clicked
  start PID-13 joystick and button
  PID: set x1 range 0 255 to 0 100 without decimal point
  forever
    if <PID: x1 > 50 > or <PID: y1 > = 50 > then
      change x by 10
    else
      change x by -10
```

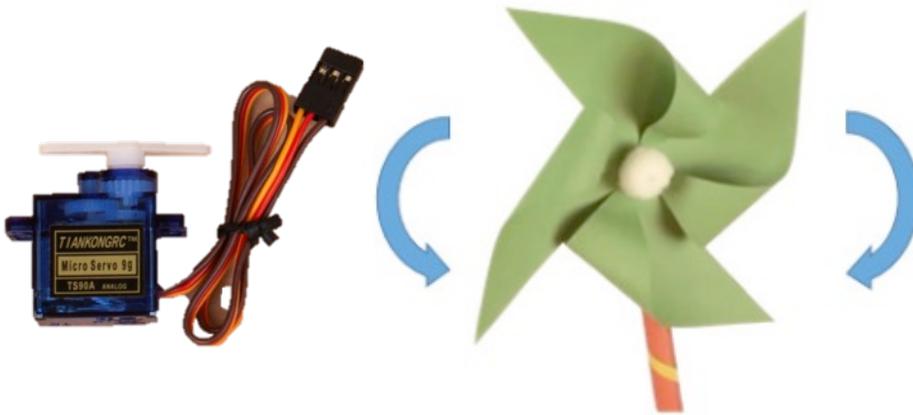
( )

# 05 Changing the Direction of a Pinwheel

You can control a pinwheel with a joystick, a servo motor, and colored paper.



You can change the direction of a pinwheel with a joystick and a servo motor. Let's make a pinwheel with colored paper and connect it to a servo motor. Then, you can control it with a joystick.



Let's get ready.



Joystick



Servo Motor



Cheese Stick

## 05. Changing the Direction of a Pinwheel



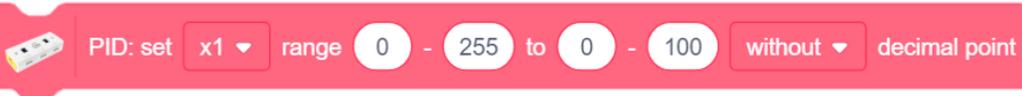
Let's learn the blocks for this chapter.



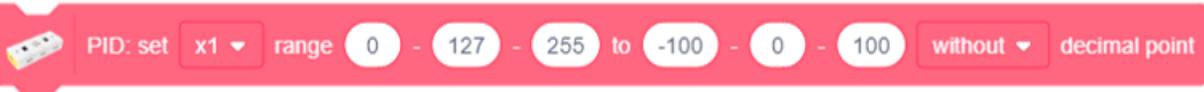
This block is needed to use a joystick.



This block sets the horizontal and vertical values of a joystick.



This block sets the range of joystick values as 0 to 100.



This block divides joystick values into three different ranges.



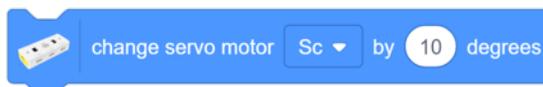
This block compares two values to run a command.



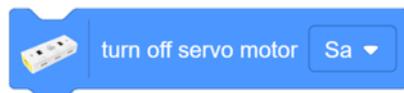
This block pauses the program for the given number of seconds before running the next block.



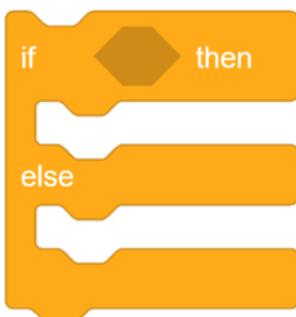
This block sets the angle of a servo motor.



This block changes the angle of a servo motor for the given number of seconds.



This block stops the running command given to a servo motor.



This block runs different commands depending on whether the condition is true or not.

## 05. Changing the Direction of a Pinwheel



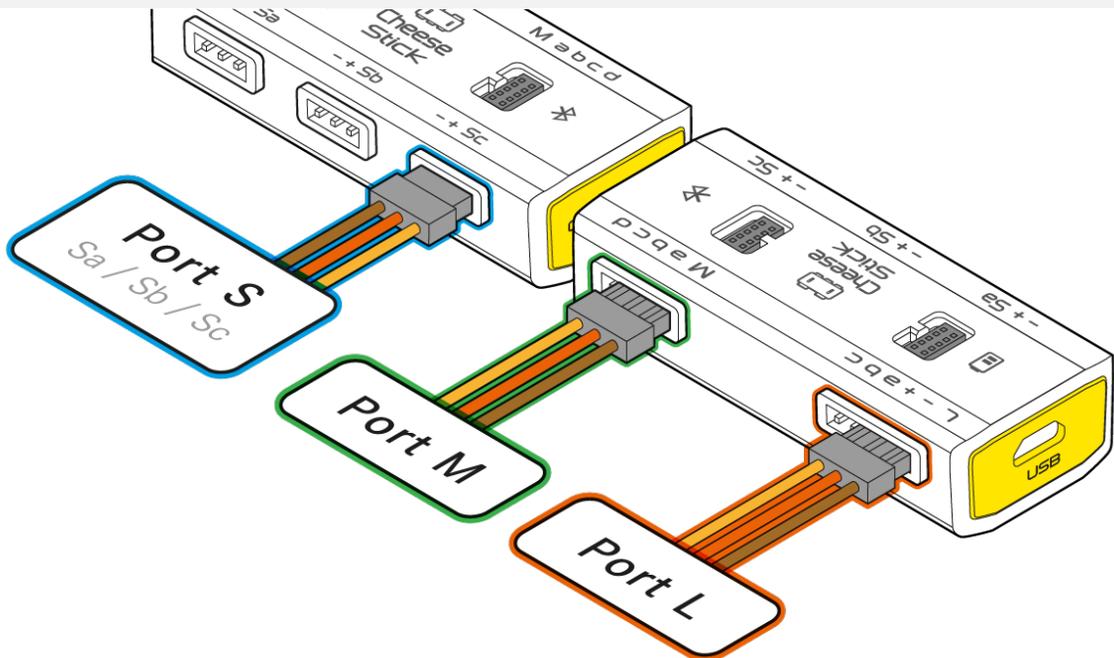
Let's connect a joystick and a servo motor to the Cheese Stick.



Plug the joystick into the Cheese Stick's L and M ports and the servo motor into any of its three ports (Sa, Sb, or Sc).



You can connect up to five servo motors to the Cheese Stick. With an expansion kit, a maximum of eight servo motors can be connected.

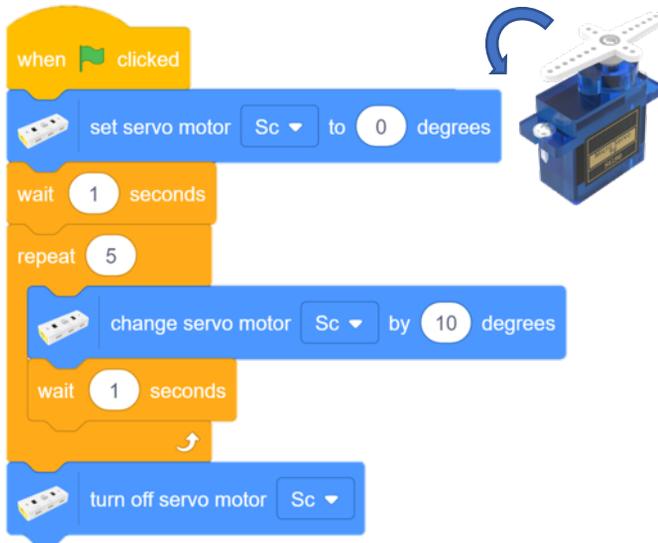


Plug the servo motor's brown cable into the negative side(-) of the Cheese Stick.

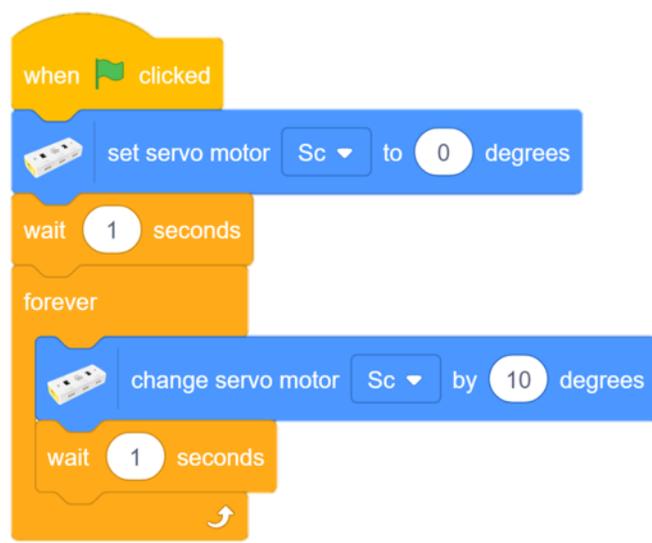
## 05. Changing the Direction of a Pinwheel



Let's program the servo motor to rotate five times.



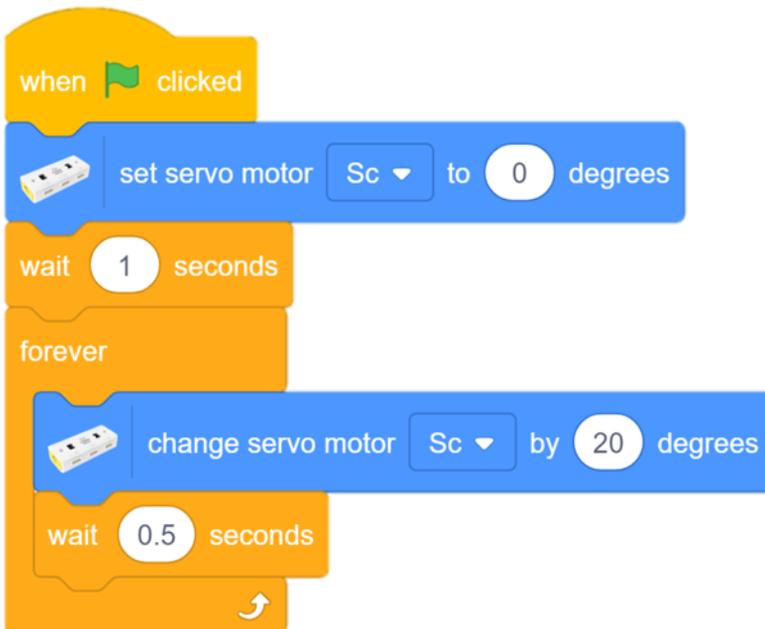
Let's program the servo motor to continue rotating.



A servo motor rotates from 0 to 180 degrees.



Let's program the servo motor to rotate faster.



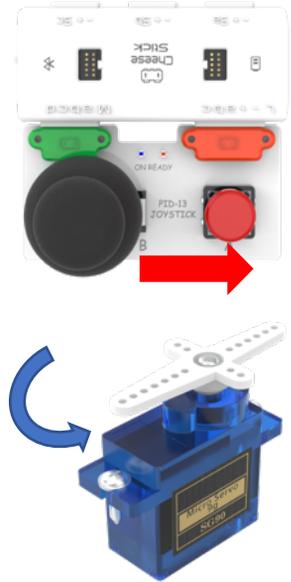
To increase the speed of the servo motor, you can either increase its angle or reduce waiting times between angle changes.

## 05. Changing the Direction of a Pinwheel



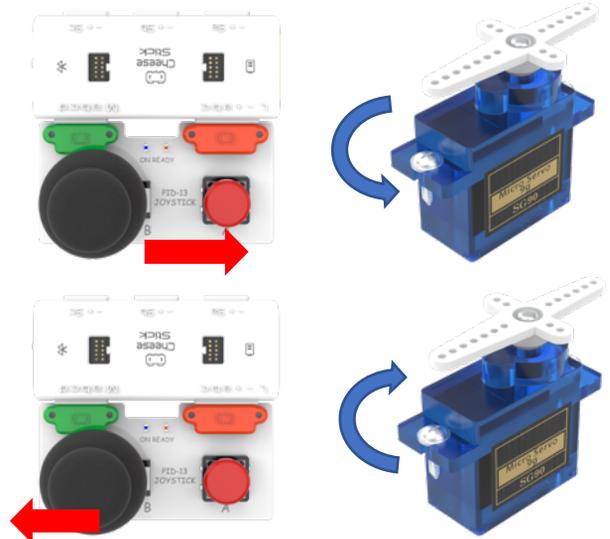
Let's program the servo motor to rotate in one direction with the joystick.

```
when clicked
  start PID-13 joystick and button
  PID: set x1 range 0 - 255 to 0 - 100 without decimal point
  forever
    if PID: x1 > 60 then
      change servo motor Sc by 10 degrees
      wait 0.5 seconds
```



Let's program the servo motor to rotate in both directions with the joystick.

```
when clicked
  start PID-13 joystick and button
  PID: set x1 range 0 - 255 to 0 - 100 without decimal point
  forever
    if PID: x1 > 60 then
      change servo motor Sc by 10 degrees
      wait 0.5 seconds
    else
      change servo motor Sc by -10 degrees
      wait 0.5 seconds
```

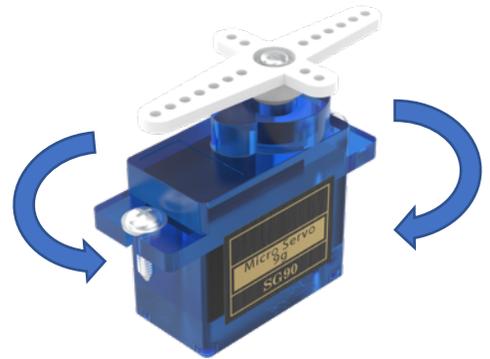


## 05. Changing the Direction of a Pinwheel



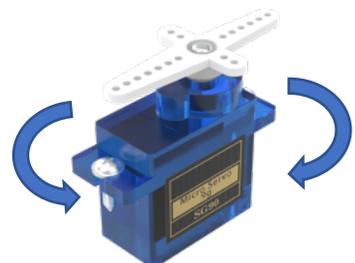
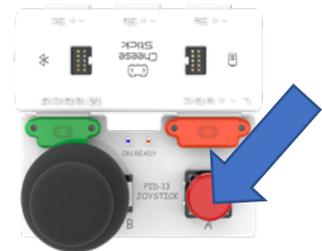
Let's program the servo motor to rotate in both directions without the joystick.

```
when clicked
  forever
    set servo motor Sc to 0 degrees
    wait 0.5 seconds
    change servo motor Sc by 10 degrees
    wait 0.5 seconds
```



Let's program the servo motor to rotate with the joystick button.

```
when clicked
  start PID-13 joystick and button
  forever
    if PID: button 2 clicked ? then
      change servo motor Sc by 10 degrees
      wait 0.5 seconds
```

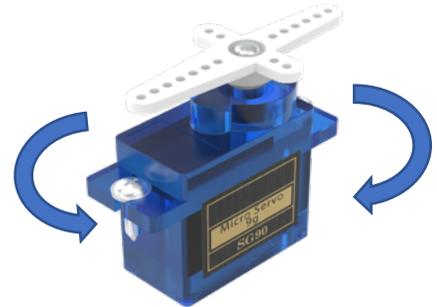


## 05. Changing the Direction of a Pinwheel

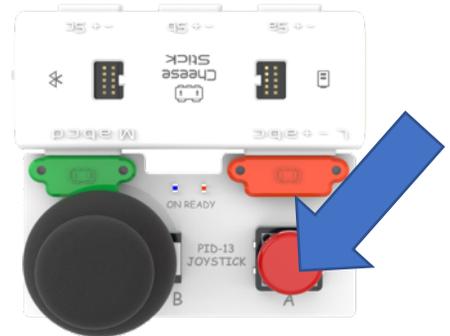


Let's program the servo motor to stop rotating when you press the joystick button.

```
when clicked
  start PID-13 joystick and button
  forever
    set servo motor Sc to 0 degrees
    wait 0.5 seconds
    change servo motor Sc by 180 degrees
    wait 0.5 seconds
```



```
when clicked
  start PID-13 joystick and button
  forever
    if PID: button 2 clicked ? then
      stop all
```



To run two commands simultaneously, both command blocks need to be connected to

when clicked block.

## 05. Changing the Direction of a Pinwheel

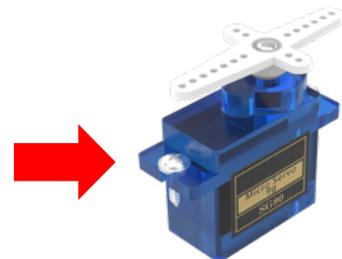
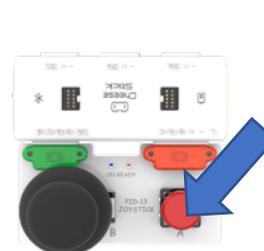


Let's connect the servo motor to a pinwheel. First, program the joystick to change the direction of the servo motor. Then, program the joystick button to stop the rotation of the servo motor. You can now control the pinwheel with the joystick and the joystick button.

```
when clicked
  start PID-13 joystick and button
  PID: set x1 range 0 - 255 to 0 - 100 without decimal point
  set servo motor Sc to 0 degrees
  forever
    if PID: x1 > 60 then
      change servo motor Sc by 10 degrees
      wait 0.5 seconds
    else
      change servo motor Sc by -10 degrees
      wait 0.5 seconds
```



```
when clicked
  start PID-13 joystick and button
  forever
    if PID: button 2 clicked ? then
      turn off servo motor Sc
```



## 05. Changing the Direction of a Pinwheel



Let's review this chapter with the following assignments.



[Assignment 1] Write how the servo motor would rotate when you enter the following commands.

```
when clicked
  start PID-13 joystick and button
  PID: set x1 range 0 - 255 to 0 - 100 without decimal point
  forever
    if PID: x1 > 60 then
      change servo motor Sc by 10 degrees
      wait 0.5 seconds
    else
      change servo motor Sc by -10 degrees
      wait 0.5 seconds
```



[Assignment 2] Write how the servo motor would rotate when you enter the following commands.

```
when clicked
  start PID-13 joystick and button
  PID: set x1 range 0 - 255 to 0 - 100 without decimal point
  set servo motor Sc to 0 degrees
  forever
    if PID: x1 > 60 then
      change servo motor Sc by 40 degrees
      wait 0.5 seconds
```

## 05. Changing the Direction of a Pinwheel



[Assignment 3] You want to change the rotation of the servo motor and also reset its value. Check the blocks you need to run these two commands simultaneously.

```
when clicked
  start PID-13 joystick and button
  PID: set x1 range 0 - 255 to 0 - 100 without decimal point
  set servo motor Sc to 0 degrees
  forever
    if PID: x1 > 60 then
      change servo motor Sc by 40 degrees
    wait 0.5 seconds
```

( )

```
when clicked
  start PID-13 joystick and button
  forever
    if PID: button 2 clicked? then
      stop all
```

( )

```
when clicked
  start PID-13 joystick and button
  PID: set x1 range 0 - 255 to 0 - 100 without decimal point
  forever
    if PID: x1 > 60 then
      change servo motor Sc by 10 degrees
      wait 0.5 seconds
    else
      change servo motor Sc by -10 degrees
      wait 0.5 seconds
```

( )

```
when clicked
  0: start PID-13 joystick and button
  forever
    if 0: PID: button 2 clicked? then
      0: set servo motor Sc to 0 degrees
```

( )

```
when clicked
  start PID-13 joystick and button
  forever
    if PID: button 2 clicked? then
      change servo motor Sc by 10 degrees
      wait 0.5 seconds
```

( )

```
when clicked
  forever
    set servo motor Sc to 0 degrees
    wait 0.5 seconds
    change servo motor Sc by 10 degrees
    wait 0.5 seconds
```

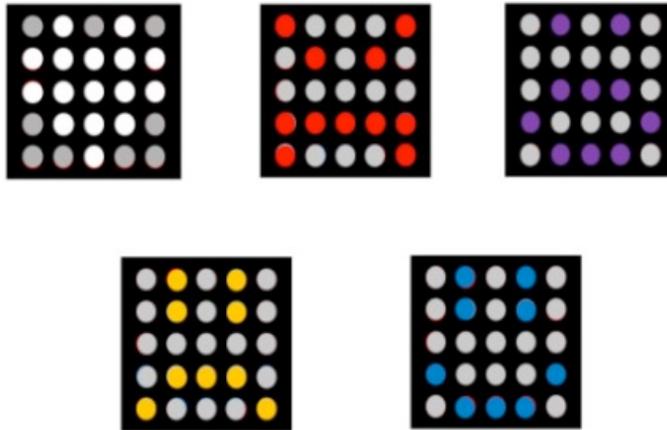
( )

# 06 Building an Emotion Expression Program

You can display different figures on the 5x5 LED Matrix with a joystick.



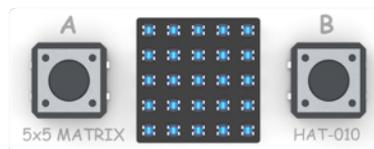
You can control the 5x5 LED Matrix with a joystick. Let's display different colors on the 5x5 LED Matrix.



Let's get ready.



Joystick



5x5 LED Matrix



Cheese Stick

## 06. Building an Emotion Expression Program



Let's learn the blocks for this chapter.



start PID-13 joystick and button ▾

This block is needed to use a joystick.



HAT-010 background: turn on x: 0 y: 0 in red ▾

This block controls the LEDs on the 5x5 LED Matrix.



start HAT-010 5x5 matrix

This block is needed to use the 5x5 LED Matrix.



HAT-010 background: draw red ▾ square ▾ at x: 0 y: 0

This block displays different figures on the 5x5 LED Matrix.



HAT-010: clear background ▾

This block removes all backgrounds on the 5x5 LED Matrix.



PID: x1 ▾



PID: y1 ▾

This block sets the horizontal and vertical values of a joystick.



This block compares two values to run a command.



PID: set x1 ▾ range 0 - 255 to 0 - 100 without ▾ decimal point

This block sets the range of joystick values as 0 to 100.



PID: set x1 ▾ range 0 - 127 - 255 to -100 - 0 - 100 without ▾ decimal point

This block divides joystick values into three different ranges.

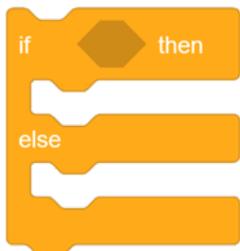


This block evaluates to true if one or both conditions are true.



PID: button 1 ▾ clicked ▾ ?

This block runs the command when the button is pressed.



This block runs different commands depending on whether the condition is true or not.

## 06. Building an Emotion Expression Program



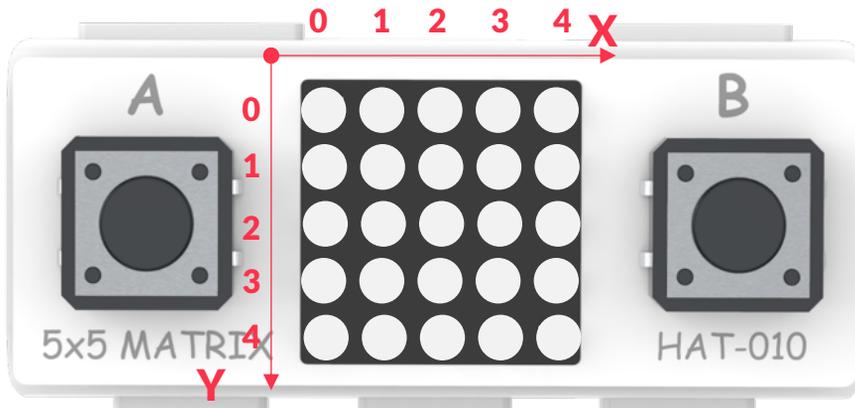
Connect a joystick and a 5x5 LED Matrix to the Cheese Stick.



Plug the joystick into the L and M ports on the Cheese Stick and the 5x5 LED Matrix into its top ports.



Push the 5X5 LED Matrix all the way into the Cheese Stick ports.



You can control the color and brightness of 25 LEDs using the 5x5 LED Matrix.

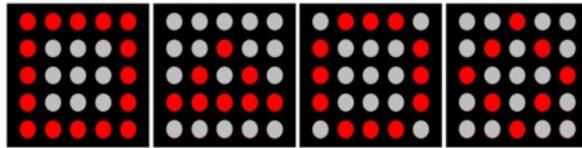
## 06. Building an Emotion Expression Program



Let's program the 5x5 LED Matrix to display different figures.

Four Scratch-style code snippets are shown, each starting with a "when clicked" event and a "start HAT-010 5x5 matrix" block. The "HAT-010 background: draw" blocks are configured as follows:

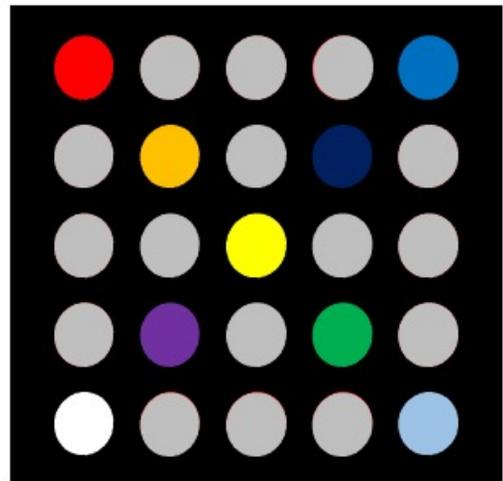
- Snippet 1: draw red square at x: 0 y: 0
- Snippet 2: draw red triangle at x: 0 y: 0
- Snippet 3: draw red circle at x: 0 y: 0
- Snippet 4: draw red diamond at x: 0 y: 0



Let's program the 5x5 LED Matrix to display different colors.

A Scratch-style code snippet is shown, starting with a "when clicked" event and a "start HAT-010 5x5 matrix" block. It is followed by nine "HAT-010 background: turn on" blocks, each with specific x and y coordinates and a color:

- Block 1: turn on x: 0 y: 0 in red
- Block 2: turn on x: 1 y: 1 in orange
- Block 3: turn on x: 2 y: 2 in yellow
- Block 4: turn on x: 3 y: 3 in green
- Block 5: turn on x: 4 y: 4 in sky blue
- Block 6: turn on x: 4 y: 0 in blue
- Block 7: turn on x: 3 y: 1 in violet
- Block 8: turn on x: 1 y: 3 in purple
- Block 9: turn on x: 0 y: 4 in white

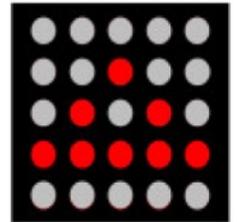


## 06. Building an Emotion Expression Program



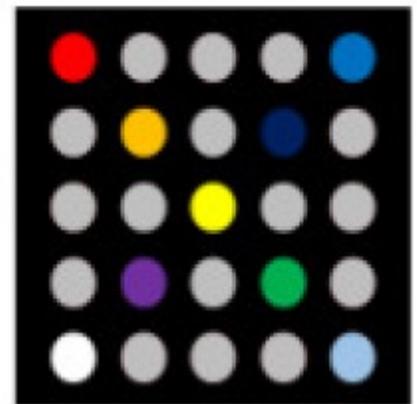
Let's program the joystick to display a triangle on the 5x5 LED Matrix.

```
when clicked
  start PID-13 joystick and button
  start HAT-010 5x5 matrix
  PID: set x1 range 0 - 255 to 0 - 100 without decimal point
  forever
    if PID: x1 > 60 then
      HAT-010 background: draw red triangle at x: 0 y: 0
```



Let's program the joystick to display different colors on the 5x5 LED Matrix.

```
when clicked
  start PID-13 joystick and button
  start HAT-010 5x5 matrix
  PID: set x1 range 0 - 255 to 0 - 100 without decimal point
  forever
    if PID: x1 > 60 then
      start HAT-010 5x5 matrix
      HAT-010 background: turn on x: 0 y: 0 in red
      HAT-010 background: turn on x: 1 y: 1 in orange
      HAT-010 background: turn on x: 2 y: 2 in yellow
      HAT-010 background: turn on x: 3 y: 3 in green
      HAT-010 background: turn on x: 4 y: 4 in sky blue
      HAT-010 background: turn on x: 4 y: 0 in blue
      HAT-010 background: turn on x: 3 y: 1 in violet
      HAT-010 background: turn on x: 1 y: 3 in purple
      HAT-010 background: turn on x: 0 y: 4 in white
```



To program the joystick to give commands to the 5x5 LED Matrix, you need both

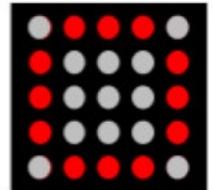
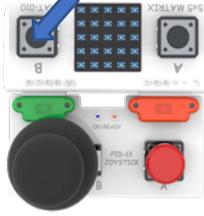
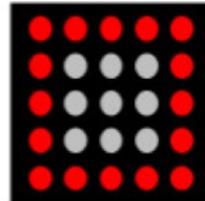
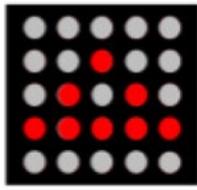
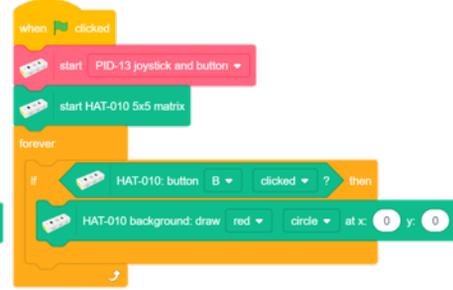
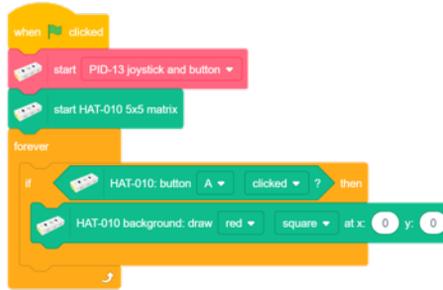


blocks.

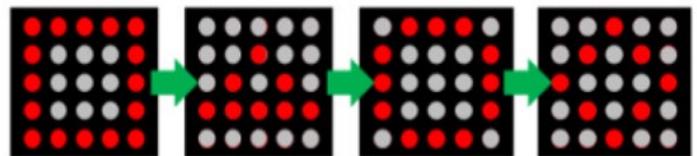
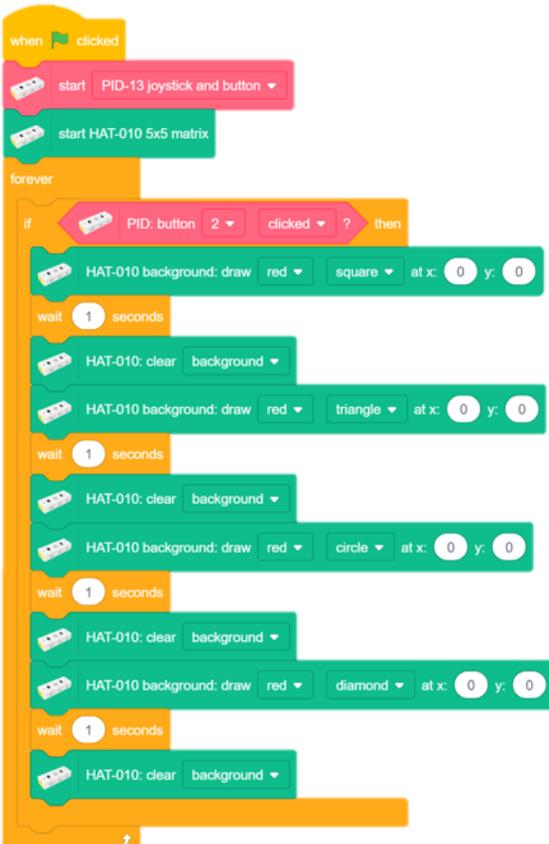
## 06. Building an Emotion Expression Program



Let's program the joystick button and the 5X5 LED Matrix buttons to display various figures.



Let's program the joystick button to display various figures in the given order on the 5x5 LED Matrix.



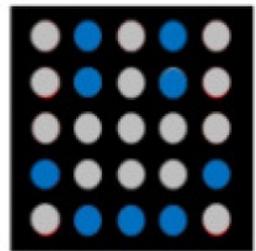
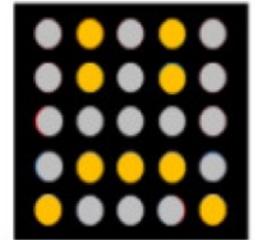
## 06. Building an Emotion Expression Program



Let's program the joystick to display three facial expressions on the 5x5 LED Matrix.

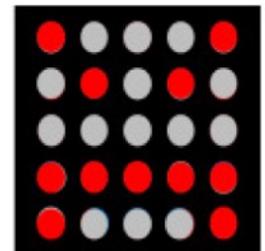
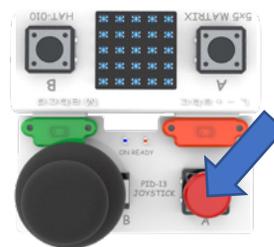
Scratch code for joystick-based facial expressions:

- when green flag clicked
- start PID-13 joystick and button
- start HAT-010 5x5 matrix
- PID: set x1 range 0 - 255 to 0 - 100 without decimal point
- forever loop:
  - if PID: x1 > 60 then:
    - HAT-010 background: draw blue like at x: 0 y: 0
    - wait 1 seconds
  - else:
    - if PID: x1 < 40 then:
      - HAT-010 background: draw orange dislike at x: 0 y: 0
      - wait 1 seconds
    - else:
      - HAT-010: clear background



Scratch code for button-based facial expression:

- when green flag clicked
- start PID-13 joystick and button
- start HAT-010 5x5 matrix
- forever loop:
  - if PID: button 2 clicked ? then:
    - HAT-010 background: draw red angry at x: 0 y: 0
    - wait 0.4 seconds
    - HAT-010: clear background

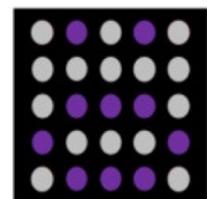
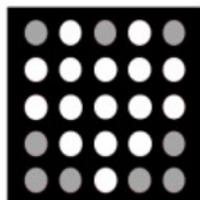
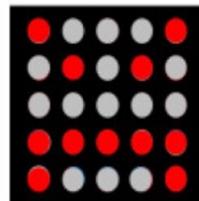
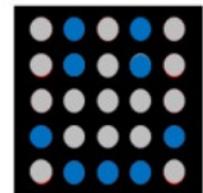
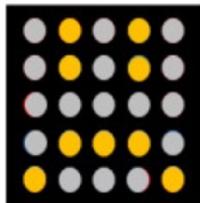


## 06. Building an Emotion Expression Program



Let's give additional commands to the joystick to display five facial expressions on the 5x5 LED Matrix.

```
when clicked
  start PID-13 joystick and button
  start HAT-010 5x5 matrix
  forever
    if HAT-010: button A clicked ? then
      HAT-010 background: draw white heart at x: 0 y: 0
      wait 1 seconds
      HAT-010: clear background
    if HAT-010: button B clicked ? then
      HAT-010 background: draw purple open mouth at x: 0 y: 0
      wait 1 seconds
      HAT-010: clear background
```



## 06. Building an Emotion Expression Program



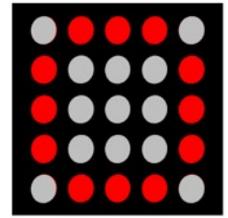
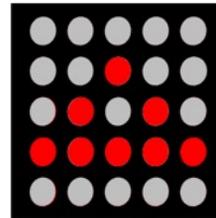
Let's review this chapter with the following assignments.



[Assignment 1] Enter the following commands. Check the figure displayed on the 5x5 LED Matrix.

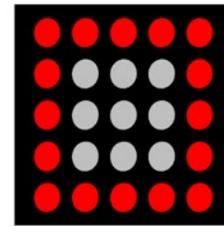
```

when clicked
  start PID-13 joystick and button
  start HAT-010 5x5 matrix
  PID: set x1 range 0 - 255 to 0 - 100 without decimal point
  forever
    if PID: x1 > 60 then
      HAT-010 background: draw red triangle at x: 0 y: 0
  
```



( )

( )



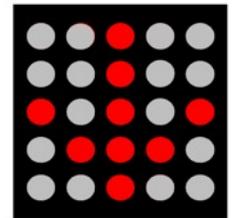
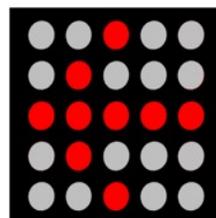
( )



[Assignment 2] Enter the following commands and press the joystick button. Check the figure displayed on the 5x5 LED Matrix.

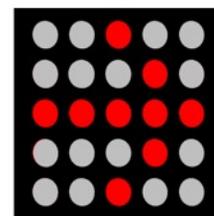
```

when clicked
  start PID-13 joystick and button
  start HAT-010 5x5 matrix
  forever
    if PID: button 2 clicked ? then
      HAT-010 background: draw red left arrow at x: 0 y: 0
  
```



( )

( )



( )

## 06. Building an Emotion Expression Program



[Assignment 3] You want to program the joystick and the 5x5 LED Matrix buttons to display five facial expressions on the LED Matrix. Check the command blocks you need for this.

```
when clicked
  start PID-13 joystick and button
  start HAT-010 5x5 matrix
  PID set x1 range 0 255 to 0 100 without decimal point
  forever
    if PID: x1 > 80 then
      HAT-010 background: draw blue smile at x: 0 y: 0
      wait 1 seconds
    else
      PID: x1 < 80 then
      HAT-010 background: draw orange smile at x: 0 y: 0
      wait 1 seconds
      HAT-010 clear background
```

( )

```
when clicked
  start PID-13 joystick and button
  start HAT-010 5x5 matrix
  PID set x1 range 0 255 to 0 100 without decimal point
  forever
    PID: x1 > 80 then
      start HAT-010 5x5 matrix
      HAT-010 background: turn on x: 0 y: 0 in red
      HAT-010 background: turn on x: 1 y: 0 in orange
      HAT-010 background: turn on x: 2 y: 0 in yellow
      HAT-010 background: turn on x: 3 y: 0 in green
      HAT-010 background: turn on x: 4 y: 0 in sky blue
      HAT-010 background: turn on x: 0 y: 1 in blue
      HAT-010 background: turn on x: 1 y: 1 in violet
      HAT-010 background: turn on x: 2 y: 1 in purple
      HAT-010 background: turn on x: 3 y: 1 in white
```

( )

```
when clicked
  start PID-13 joystick and button
  start HAT-010 5x5 matrix
  forever
    if HAT-010 button A clicked ? then
      HAT-010 background: draw white heart at x: 0 y: 0
      wait 1 seconds
      HAT-010 clear background
    if HAT-010 button B clicked ? then
      HAT-010 background: draw purple open mouth at x: 0 y: 0
      wait 1 seconds
      HAT-010 clear background
```

( )

```
when clicked
  start PID-13 joystick and button
  start HAT-010 5x5 matrix
  forever
    PID: button 2 clicked ? then
      HAT-010 background: draw red square at x: 0 y: 0
      wait 1 seconds
      HAT-010 clear background
      HAT-010 background: draw red square at x: 0 y: 0
      wait 1 seconds
      HAT-010 clear background
      HAT-010 background: draw red circle at x: 0 y: 0
      wait 1 seconds
      HAT-010 clear background
      HAT-010 background: draw red diamond at x: 0 y: 0
      wait 1 seconds
      HAT-010 clear background
```

( )

```
when clicked
  start PID-13 joystick and button
  start HAT-010 5x5 matrix
  forever
    if PID: button 2 clicked ? then
      HAT-010 background: draw red angry at x: 0 y: 0
      wait 0.4 seconds
      HAT-010 clear background
```

( )

```
when clicked
  start HAT-010 5x5 matrix
  HAT-010 background: turn on x: 0 y: 0 in red
  HAT-010 background: turn on x: 1 y: 1 in orange
  HAT-010 background: turn on x: 2 y: 2 in yellow
  HAT-010 background: turn on x: 3 y: 3 in green
  HAT-010 background: turn on x: 4 y: 4 in sky blue
  HAT-010 background: turn on x: 4 y: 0 in blue
  HAT-010 background: turn on x: 3 y: 1 in violet
  HAT-010 background: turn on x: 1 y: 3 in purple
  HAT-010 background: turn on x: 0 y: 4 in white
```

( )

# 07 Making a Bouncing Ball Game

You can use the Cheese Stick and a potentiometer to make a bouncing ball game.



A potentiometer is a type of sensor that changes its value when you turn its knob. In this chapter, we will learn about a potentiometer and use it to program a bouncing ball game.



Let's get ready.



Potentiometer



Cheese Stick

## 07. Making a Bouncing Ball Game



Let's learn the blocks for this chapter.



This block sets input and output devices of Sa, Sb, and Sc ports.



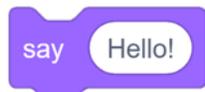
This block saves the input and output device values of Sa port.



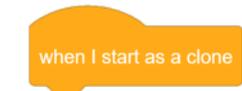
This block changes the range of input and output device values of Sa port to the given range.



This block shows or hides the sprite.



This block allows the sprite to speak through speech bubbles.



These blocks create and delete the sprite duplicates.



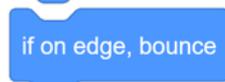
This block moves the sprite in the given direction.



This block rotates the sprite in the given direction.



This block moves the sprite to the given x- and y- coordinates.



This block bounces the sprite off the screen borders when it touches them.



This block generates a random number from the given range.



This block checks whether an sprite/Sprite touches the screen borders or other sprites.



This block compares two values to run a command.

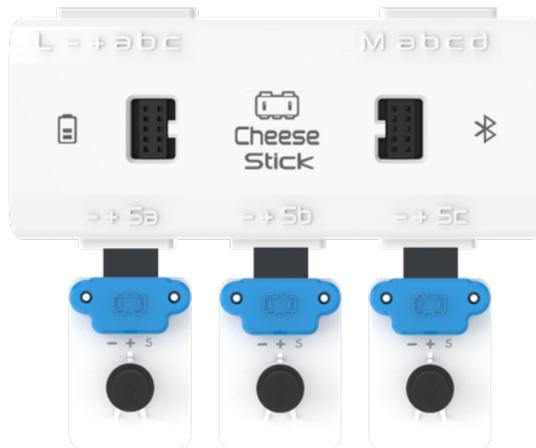
## 07. Making a Bouncing Ball Game



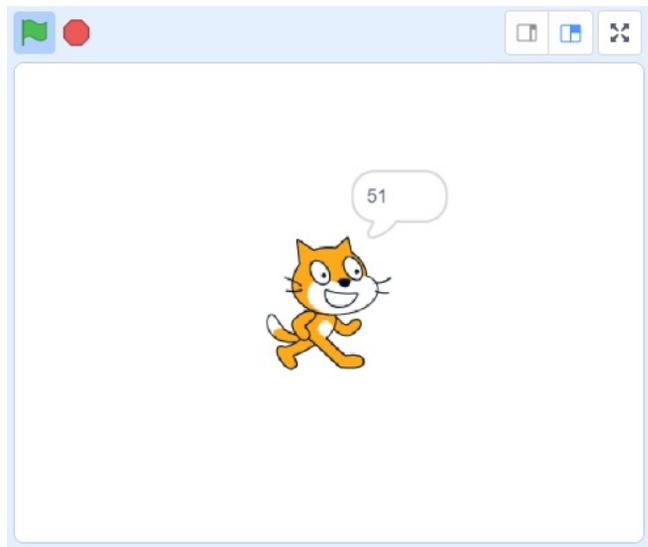
Let's connect a potentiometer to the Cheese Stick.



Plug the potentiometer into any of the three ports (Sa, Sb, or Sc) on the Cheese Stick. You can also plug it into L port in the following terminal order: `-+a`.



You can change the value of the potentiometer by turning its knob. Let's program the Scratch Cat to display the potentiometer value. Play with the knob and observe changes in the value.



A potentiometer is an analog input device. To use this device, you need



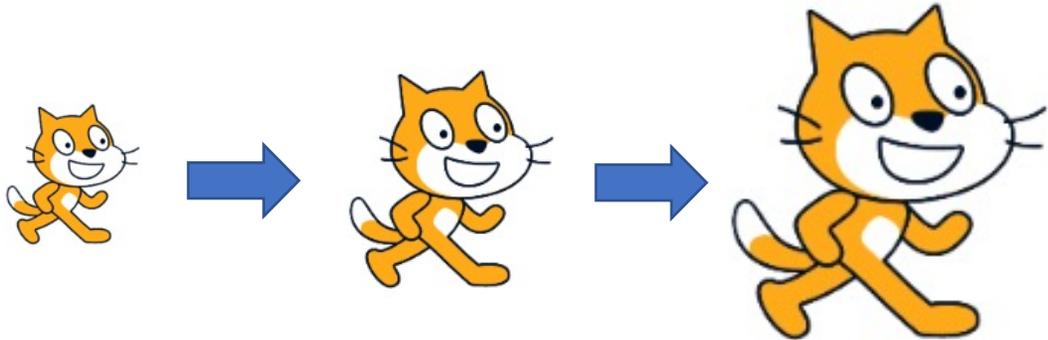
block.

## 07. Making a Bouncing Ball Game



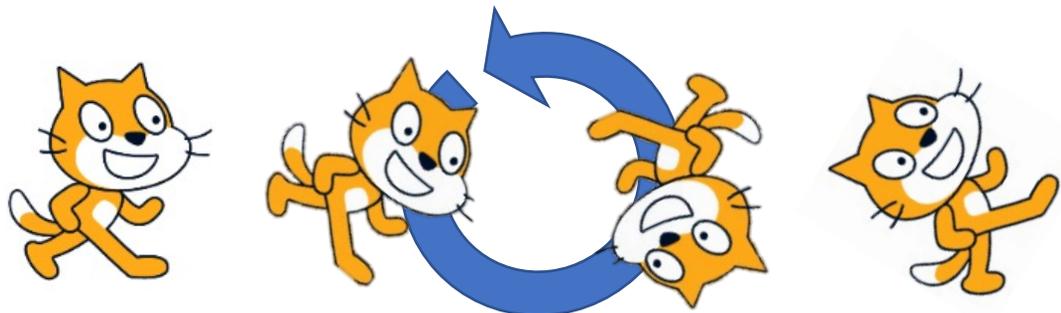
### [Practice Assignment 1]

Program the potentiometer to enlarge and reduce the size of the Scratch Cat.



### [Practice Assignment 2]

Program the potentiometer to rotate the Scratch Cat.



### [Practice Assignment 1]

```
when green flag clicked
  set port Sa to analog input
  forever
    set size to input Sa %
```



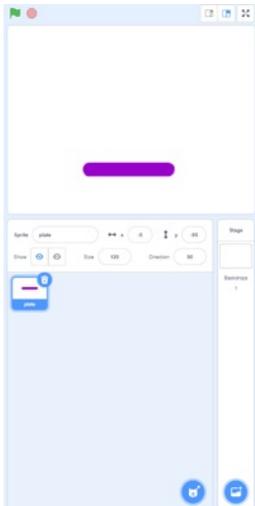
### [Practice Assignment 2]

```
when green flag clicked
  set port Sa to analog input
  forever
    point in direction input Sa
```

## 07. Making a Bouncing Ball Game



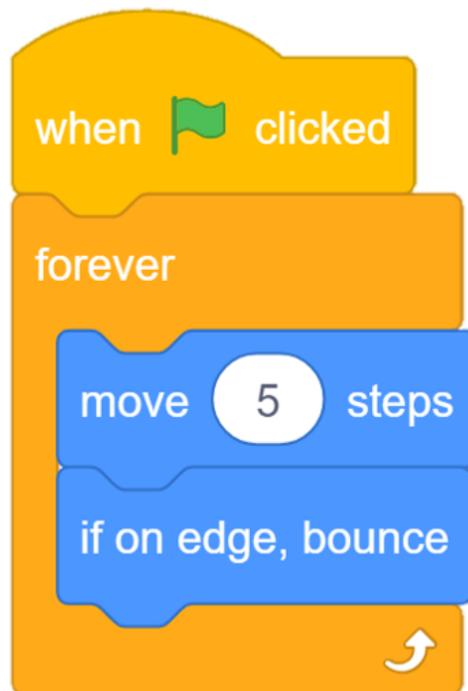
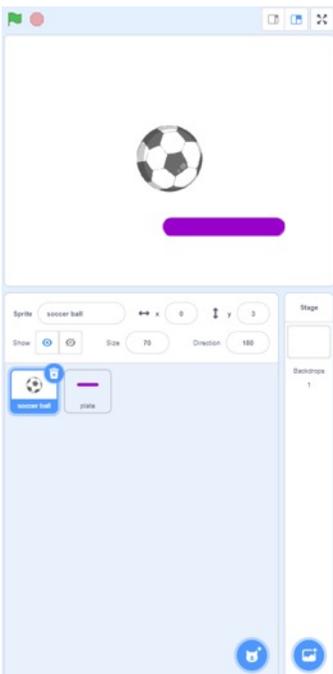
Let's program a bouncing ball game. First, make a bar that moves as you turn the knob of the potentiometer. Add the [board] sprite and enter additional commands.



The potentiometer values are between 0 and 1023. But to make a ball bounce, the x-coordinate has to be between -240 and 240. Therefore, you need



Add the [ball] sprite. Then, let's program it to move with the sprite. Let's also program it to bounce off the screen borders when it touches them.



## 07. Making a Bouncing Ball Game



It's not much fun if the ball only bounces up and down, is it? Let's program it to bounce in random directions when it touches the [Board] sprite.

```
when green flag clicked
  forever loop
    move 5 steps
    if on edge, bounce
    if touching plate ? then
      point in direction pick random -50 to 50
```

The code consists of a 'when green flag clicked' block followed by a 'forever' loop. Inside the loop, there are three blocks: 'move 5 steps', 'if on edge, bounce', and an 'if touching plate ? then' block. The 'if touching plate ? then' block contains a 'point in direction pick random -50 to 50' block. A red box highlights the 'if touching plate ? then' block and its contents.



Let's first program the timer to start. Then, program the timer to stop and the ball to disappear when the ball touches the screen borders.

```
when green flag clicked
  show
  forever loop
    move 5 steps
    if on edge, bounce
    if touching plate ? then
      point in direction pick random -50 to 50
    if touching Line ? then
      hide
      stop all

when green flag clicked
  set timer to 0
  forever loop
    wait 1 seconds
    change timer by 1
```

The image shows two separate Scratch code snippets. The left snippet is a continuation of the previous code, with a red box highlighting an 'if touching Line ? then' block containing 'hide' and 'stop all' blocks. The right snippet starts with 'when green flag clicked' and contains a 'set timer to 0' block followed by a 'forever' loop with 'wait 1 seconds' and 'change timer by 1' blocks. A red box highlights the 'set timer to 0' block and the 'forever' loop.

## 07. Making a Bouncing Ball Game



Let's add a little challenge to the game. Program the ball to move slowly in the first ten seconds then increase its speed.

```
when clicked
  forever
    if timer < 10 or timer = 10 then
      move 5 steps
    if timer > 10 then
      move 10 steps
    if on edge, bounce
    if touching plate? then
      point in direction pick random -50 to 50

when clicked
  set timer to 0
  forever
    wait 1 seconds
    change timer by 1
```



You can make the timer variable to change the speed of the ball. Let's program the ball to move at the speed of 5 in the given direction for the first 10 seconds and double its speed after 10 seconds. Try changing the time and the speed of the ball.

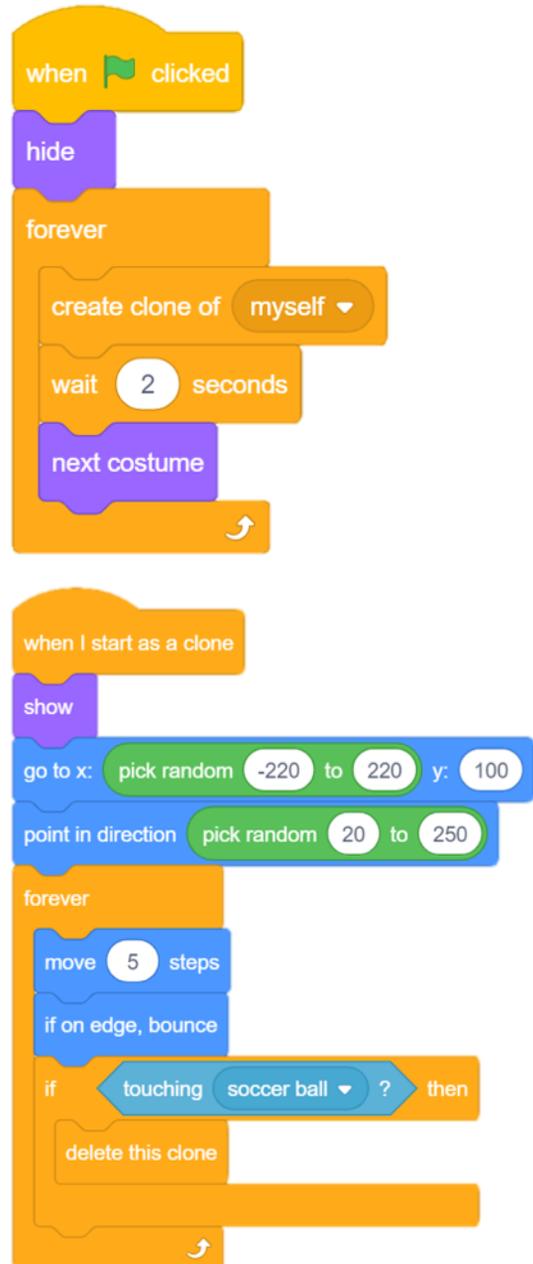
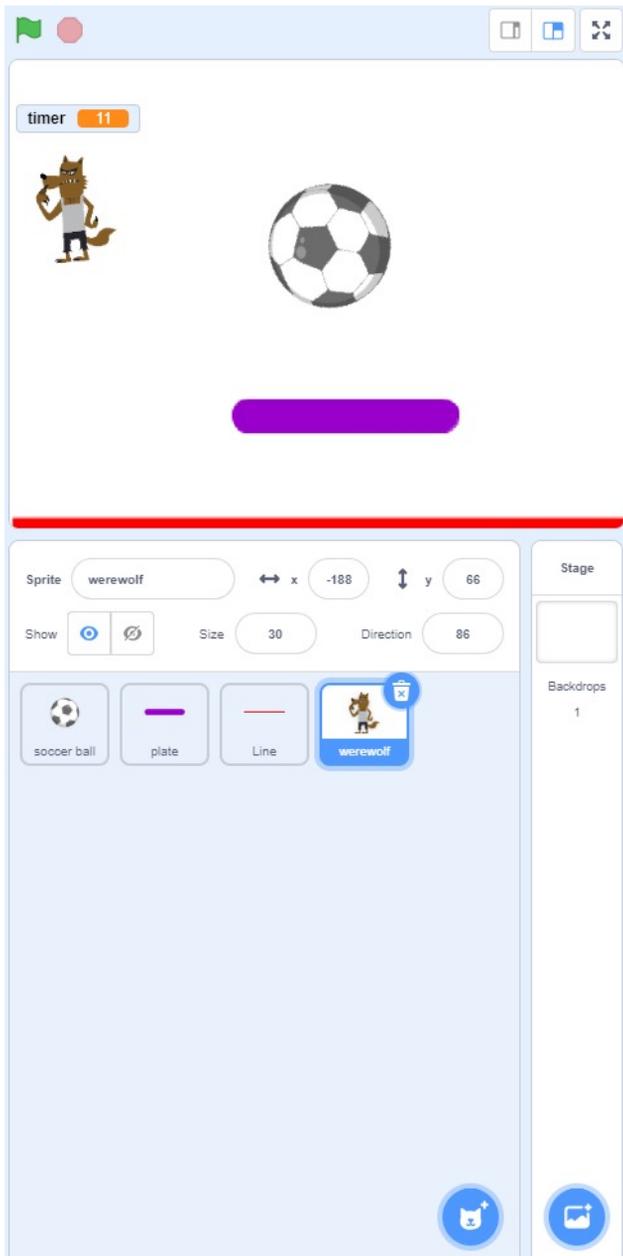
**my variable** block is under the **Variables** tab,

and **10 > 10** block is under the **Operators** tab.

## 07. Making a Bouncing Ball Game



Catching a monster with the bouncing ball would add some extra fun. Add the [Werewolf] sprite. Then, program the werewolf to disappear when the bouncing ball touches it.



You need `create clone of myself` block to program the monster to pop up on the screen.

Think about the function of each command block.



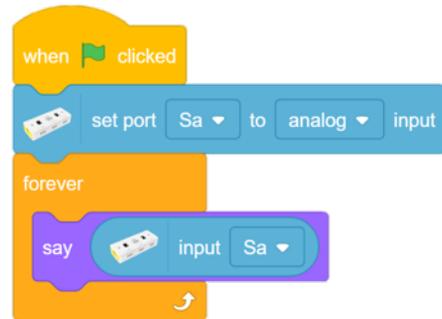
Let's review this chapter with the following assignments.



[Assignment 1] Connect the potentiometer as shown below and enter the following commands. Check what the ScratchCat displays on the screen.



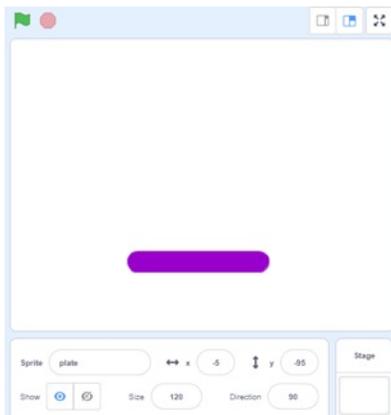
- ① Potentiometer value
- ② Scratch Cat's angle of rotation
- ③ Size of the Scratch Cat



- ④ Battery of the Cheese Stick



[Assignment 2] You programmed the [Board] sprite to move as you change the potentiometer value. Check the problem that might occur when you enter the commands as shown below.



- ① The board would not move.
- ② The board would go out of the screen.



- ③ The board would move up and down.
- ④ The board would move slowly.

## 07. Making a Bouncing Ball Game



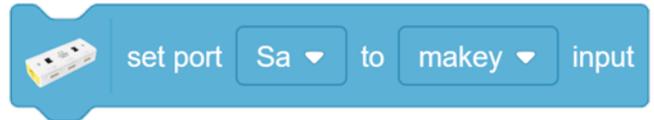
[Assignment 3] You want to change the range of your potentiometer value. Check the block you need for this.



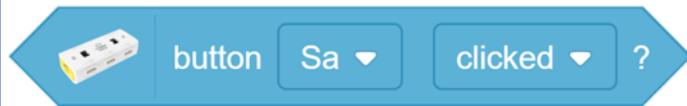
(                    )



(                    )



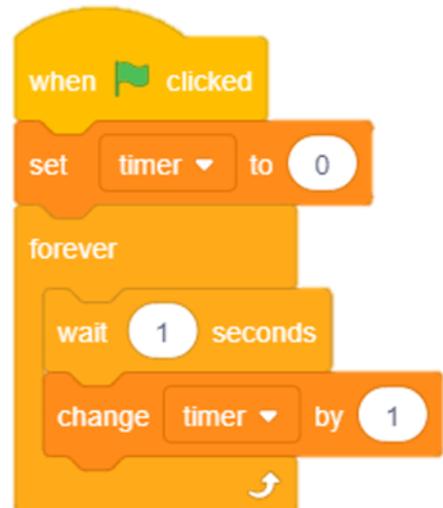
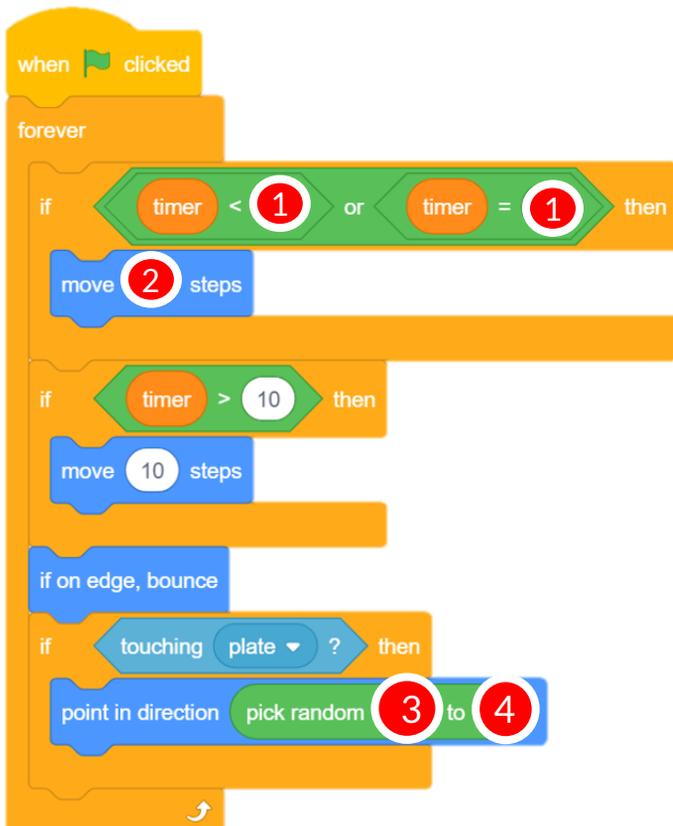
(                    )



(                    )



[Assignment 4] You want to control the speed of the ball in a bouncing ball game. Check the block you need to change for this.



# 08 Adjusting the Brightness of a Lamp

With a potentiometer and an LED, you can make a lamp and adjust its brightness.



Let's program a potentiometer and an LED to make a lamp with adjustable brightness.



Let's get ready.



Potentiometer



LED

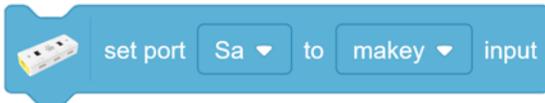


Cheese Stick

## 08. Adjusting the Brightness of a Lamp



Let's learn the blocks for this chapter.



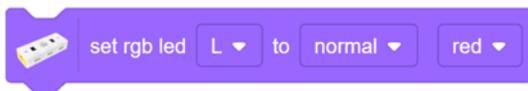
This block sets input and output devices of Sa, Sb, and Sc ports.



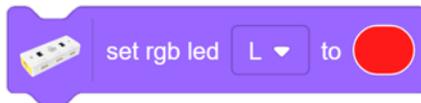
This block saves the input and output device values of Sa port.



This block changes the range of input and output device values of Sa port to the given range.



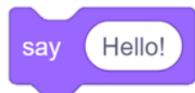
This block sets the color and brightness(dark, standard, and bright) of the LED.



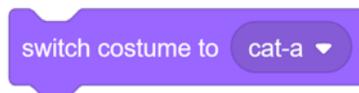
This block sets the LED color by adjusting the brightness(luminance), saturation, and color(hue).



This block changes the LED color by adjusting the RGB value.



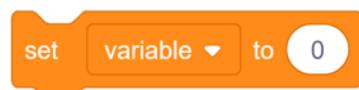
This block allows the sprite to speak through speech bubbles.



This block changes the Sprite's motions.



This block compares two values to run a command.

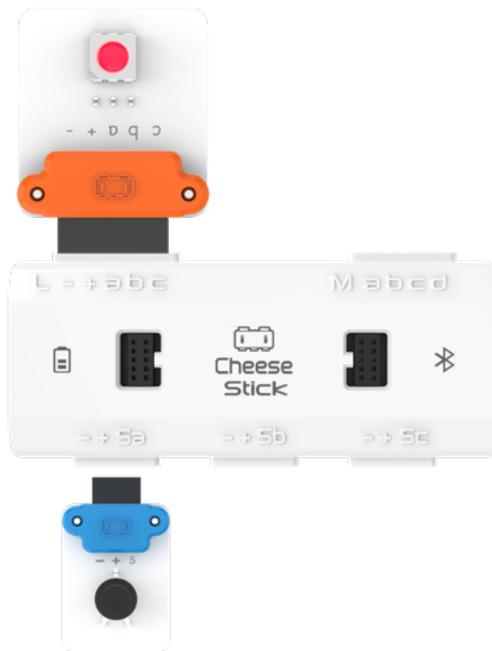
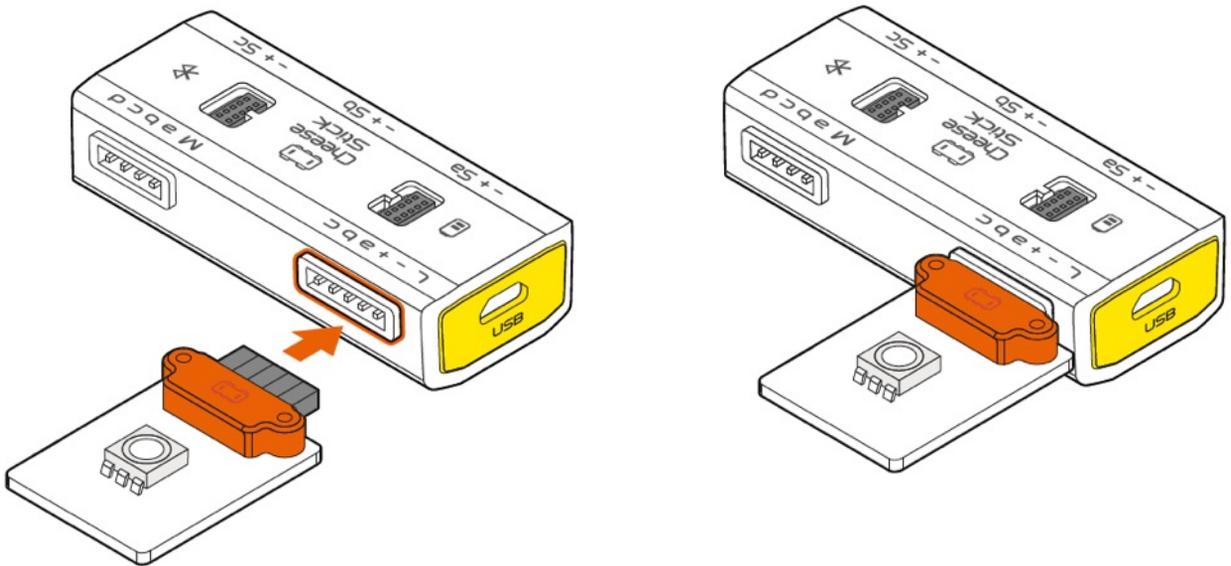


This block assigns a value to the variable.

## 08. Adjusting the Brightness of a Lamp



Let's connect the potentiometer and the LED to the Cheese Stick. Plug the potentiometer into Sa port and the LED into L port.



Exposures to high-intensity LED light can hurt your eyes. Keep the LED at low intensity or cover the LED with a paper cup.

## 08. Adjusting the Brightness of a Lamp



[Practice Assignment 1] Let's program the LED to light up white.



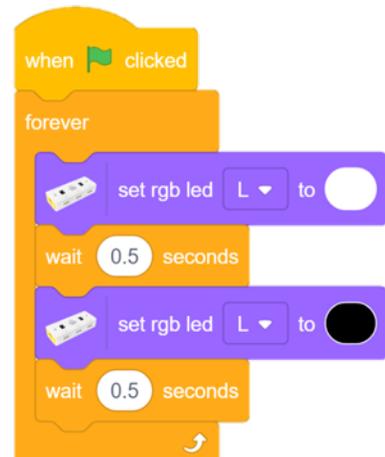
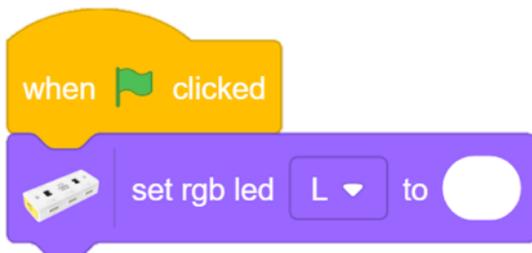
[Practice Assignment 2] Let's program the LED to blink white.



Practice Assignment 1



Practice Assignment 2



## 08. Adjusting the Brightness of a Lamp



Let's learn different ways of programming the LED to light up white.

set rgb led L to normal white

- red
- orange
- yellow
- green
- sky blue
- blue
- violet
- purple
- ✓ white

when green flag clicked

set rgb led L to

Color 0

Saturation 100

Brightness 100

set rgb led L to r: 255 g: 255 b: 255



Adjust the brightness of the R, G, and B to create different colors. Entering the same value for R, G, and B produces white. Each of the three colors has a minimum value of 0 and a maximum value of 255.

## 08. Adjusting the Brightness of a Lamp



Let's enter commands to set the LED color to white and adjust its brightness.

when  clicked

set rgb led  to r: 10 g: 10 b: 10

when  clicked

set rgb led  to r: 50 g: 50 b: 50

when  clicked

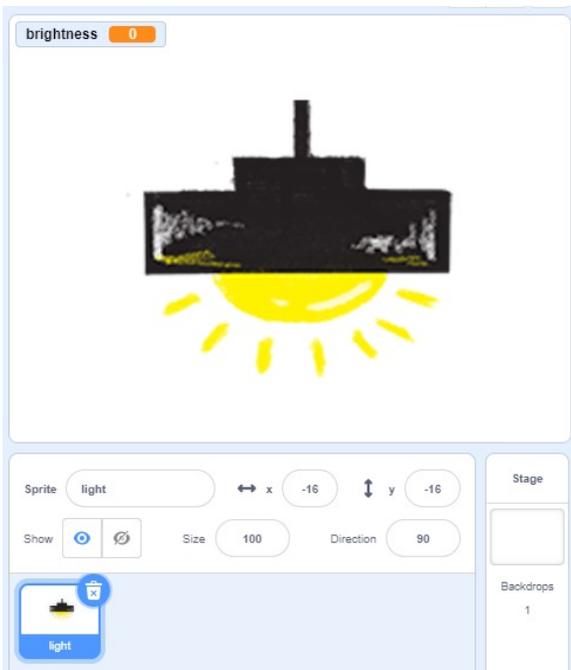
set rgb led  to r: 255 g: 255 b: 255



Try entering the same value for R, G, and B and observe how the same white light gets brighter as you enter greater numbers.



Let's add the [Lamp] sprite and program the LED to light up white.



brightness 0

Sprite: light x: -16 y: -16

Show Size: 100 Direction: 90

Stage

Backdrops: 1

when  clicked

set rgb led  to r: 255 g: 255 b: 255

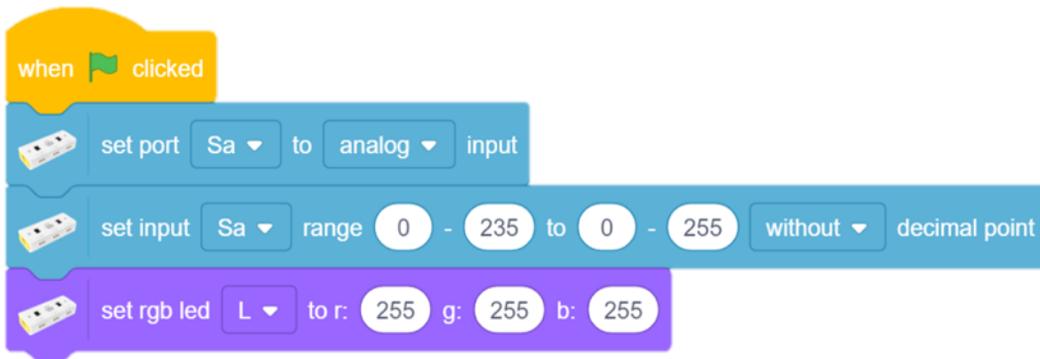
## 08. Adjusting the Brightness of a Lamp



Let's program the LED to change its light intensity when you turn the potentiometer's knob. Give commands to the LED to take the potentiometer values.



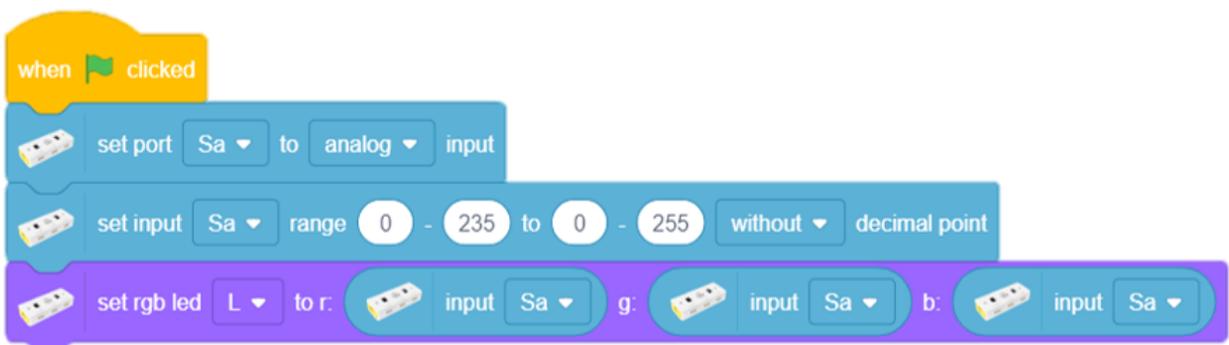
Change the range of the potentiometer values to the range of the RGB values.



Potentiometer and RGB values range from 0 to 1023 and from 0 to 255, respectively. Therefore, the range of the potentiometer values has to be adjusted to fit the range of the RGB values.



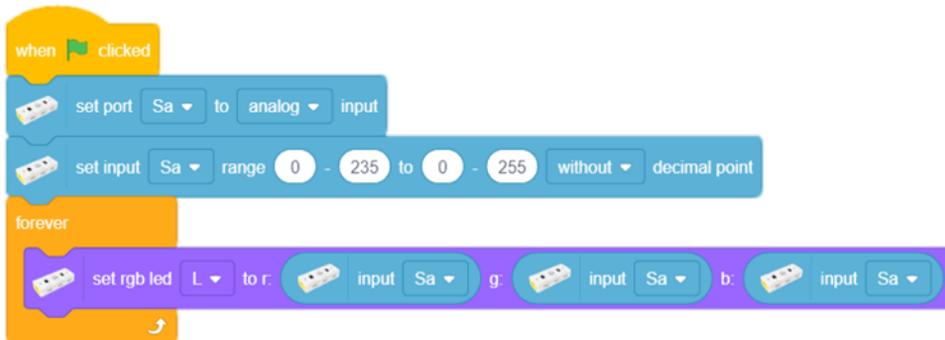
Program the potentiometer value to be entered in the RGB LED block.



## 08. Adjusting the Brightness of a Lamp



Let's use [Repeat] block to continue adjusting the brightness.

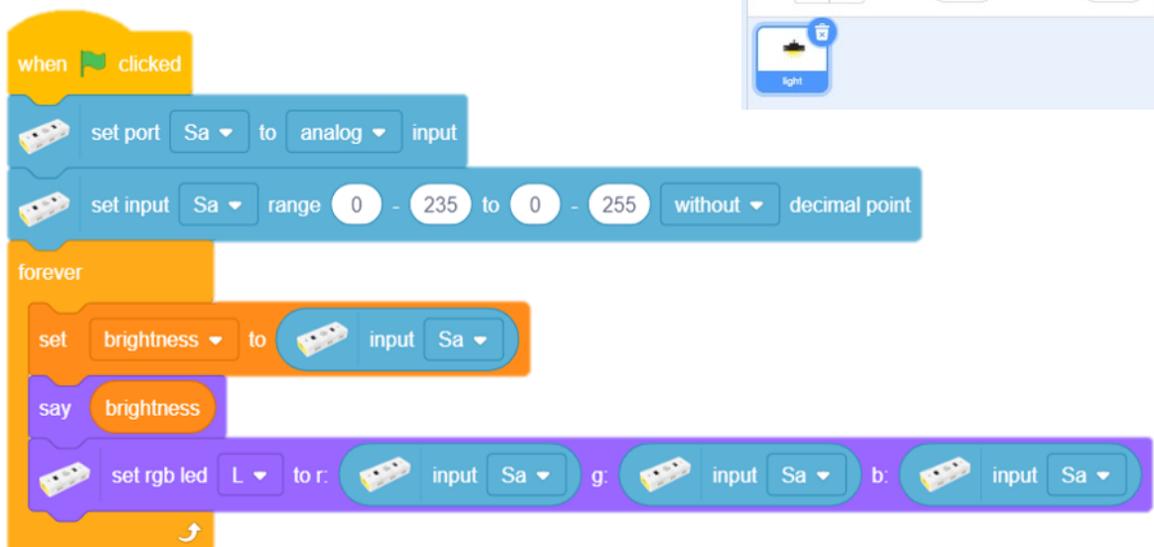


Let's display the brightness level on the screen. Create a variable and name it brightness.

The Variables palette shows 'Make a Variable' highlighted with a red box. Below it are various variable blocks like 'set my variable to 0', 'change my variable by 1', 'show variable my variable', and 'hide variable my variable'. The 'Variables' category is highlighted with a red box at the bottom.

The 'New Variable' dialog box shows 'New variable name:' with 'brightness' entered in the text field. The 'For all sprites' radio button is selected. 'Cancel' and 'OK' buttons are at the bottom, with 'OK' highlighted by a red box.

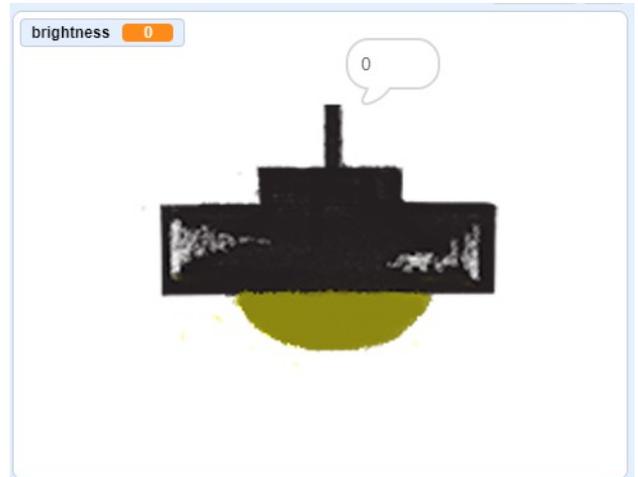
The stage shows a 'light' sprite with a yellow sun-like glow. A 'brightness 0' variable monitor is visible at the top left. The sprite's properties are shown at the bottom: Sprite: light, x: -16, y: -16, Size: 100, Direction: 90.



## 08. Adjusting the Brightness of a Lamp



Let's program the lamp to turn on and off on the screen.



```
when green flag clicked
  set port Sa to analog input
  set input Sa range 0 - 235 to 0 - 255 without decimal point
  forever loop
    set brightness to input Sa
    say brightness
    set rgb led L to r: input Sa g: input Sa b: input Sa
    if input Sa = 0 then
      switch costume to light_off
    else
      switch costume to light_on
```



See how the lamp brightens and dims depending on the value of the potentiometer. The lamp is off when the potentiometer value is 0.

## 08. Adjusting the Brightness of a Lamp

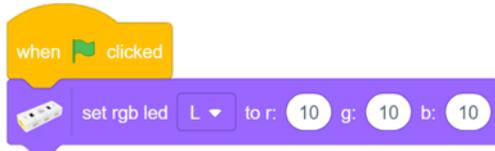


Let's review this chapter with the following assignments.

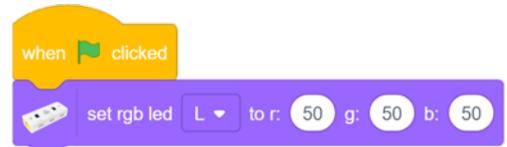


[Assignment 1] Check the command that does not program the LED to light up white.

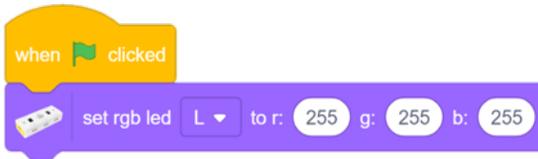
①



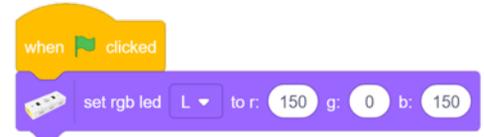
③



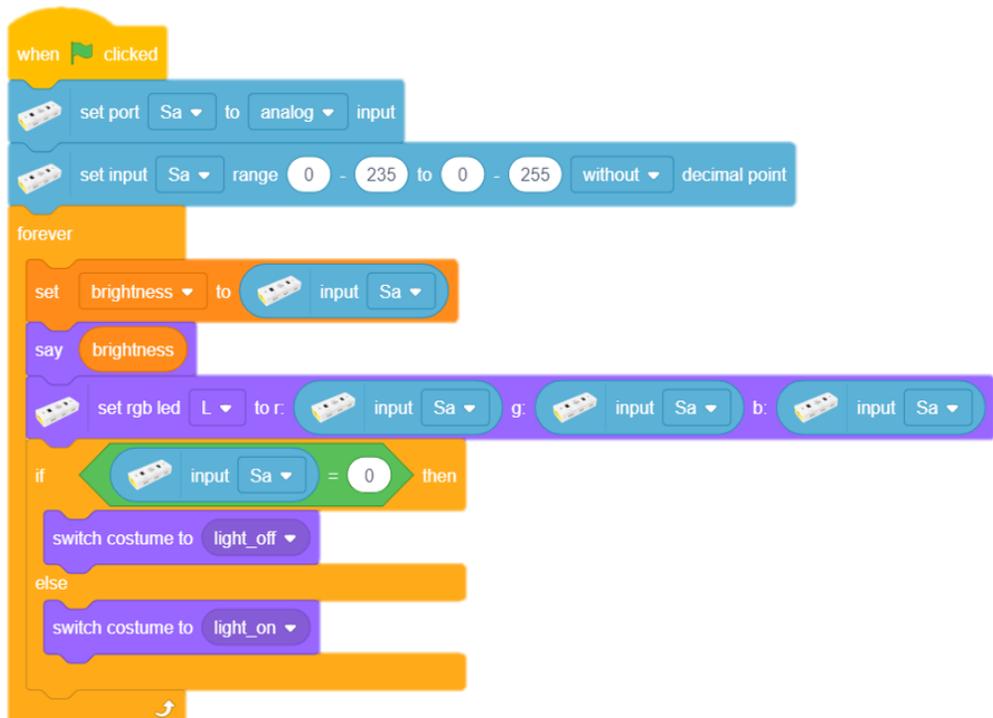
③



④



[Assignment 2] You want to program the LED to light up white and adjust its brightness with the potentiometer. You ran the program as shown below but were unable to adjust the brightness. Figure out which block you need to change to fix this problem.



# 09 Making Paper Cups Dance

You can use a potentiometer, a servo motor, and paper cups to make dancing paper cups.



You can make fun toys with a potentiometer and a servo motor. Let's program paper cup characters to dance.



Source : <http://sblog.i-scream.co.kr/platon1026/168?sso=ok#none>



Let's get ready.



Potentiometer



Servo Motor



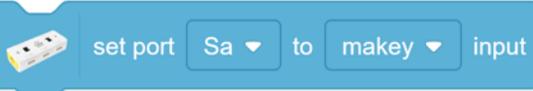
Paper Cup



Cheese Stick



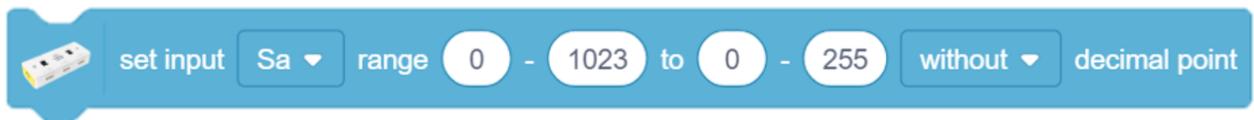
Let's learn the blocks for this chapter.



This block sets input and output devices of Sa, Sb, and Sc ports.



This block saves the input and output device values of Sa port.



This block changes the range of input and output device values of Sa port to the given range.



This block allows the sprite to speak through speech bubbles.



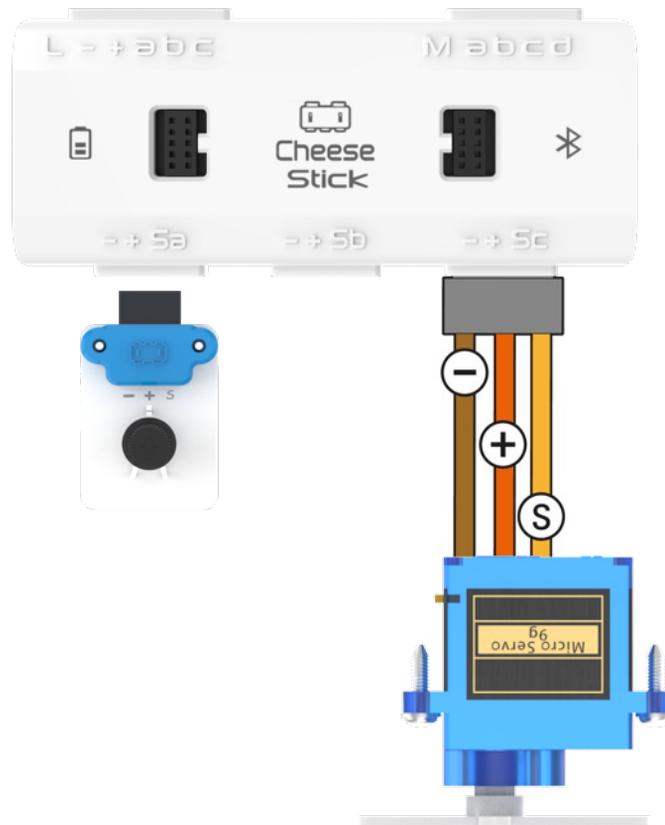
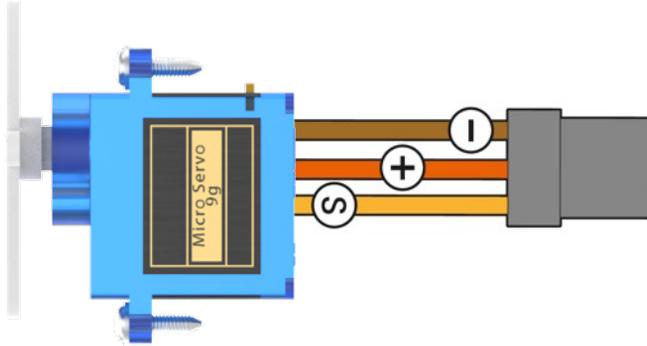
This block sets the angle of a servo motor.



This block pauses the program for the given number of seconds before running the next block.



Let's connect the potentiometer and servo motor to the Cheese Stick.

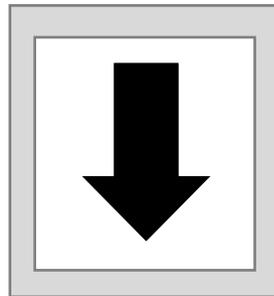
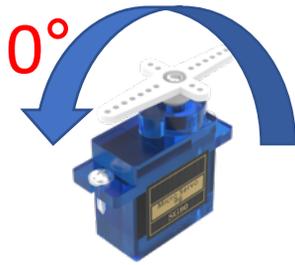
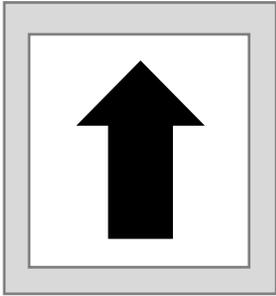


Connect the servo motor to one of the three S ports. Check the servo motor's cable colors before plugging it into the Cheese Stick. Using multiple servo motors can lead to a power shortage and cause the servo motor to malfunction.

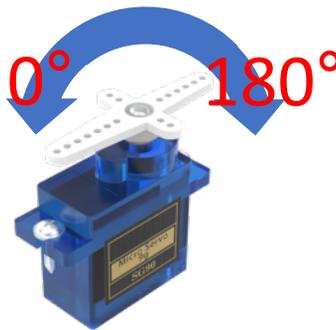
## 09. Making Paper Cups Dance



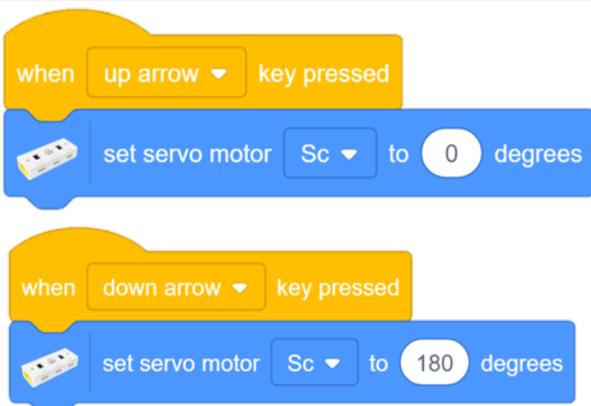
[Practice Assignment 1] Let's rotate the servo motor. Program it to rotate by 0 degree when you press the upward arrow and by 180 degrees when you press the downward arrow.



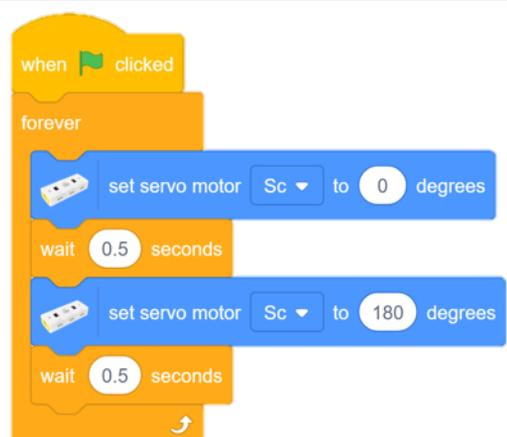
[Practice Assignment 2] Let's program the servo motor to repeat its rotation between 0 and 180 degrees.



[Practice Assignment 1]



[Practice Assignment 2]



## 09. Making Paper Cups Dance



Change the range of the potentiometer values to the range of the servo motor values.

```
when clicked
  set port Sa to analog input
  set input Sa range 0 - 1023 to 0 - 180 without decimal point
```



Enter the following commands to display the changed value of the potentiometer.

```
when clicked
  set port Sa to analog input
  set input Sa range 0 - 1023 to 0 - 180 without decimal point
  forever
    say input Sa
```

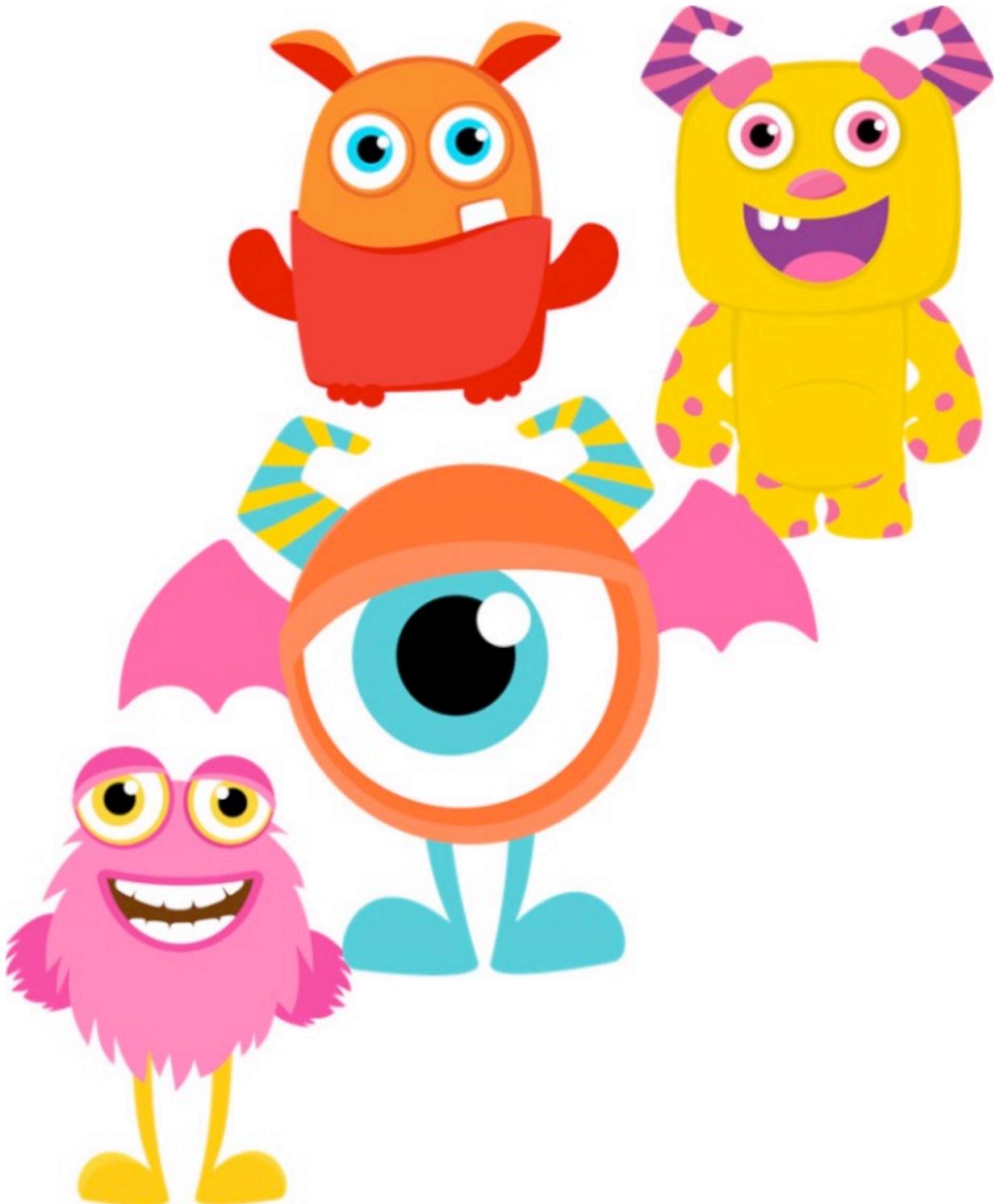


Enter the following commands to set the servo motor's angle to the potentiometer value.

```
when clicked
  set port Sa to analog input
  set input Sa range 0 - 1023 to 0 - 180 without decimal point
  forever
    say input Sa
    set servo motor Sc to input Sa degrees
```



Let's make toys with paper cups. Feel free to use the images in the <Appendix>.



## 09. Making Paper Cups Dance



Let's connect the servo motor to the paper cup toy and run the program.



Let's program the servo motor to have small or large rotation angles depending on the potentiometer value.

```
when clicked
  set port Sa to analog input
  set input Sa range 0 - 235 to 0 - 180 without decimal point
  forever
    say input Sa
    set servo motor Sc to input Sa degrees
    wait 0.2 seconds
    set servo motor Sc to 0 degrees
    wait 0.2 seconds
```

## 09. Making Paper Cups Dance



Let's review this chapter with the following assignments.

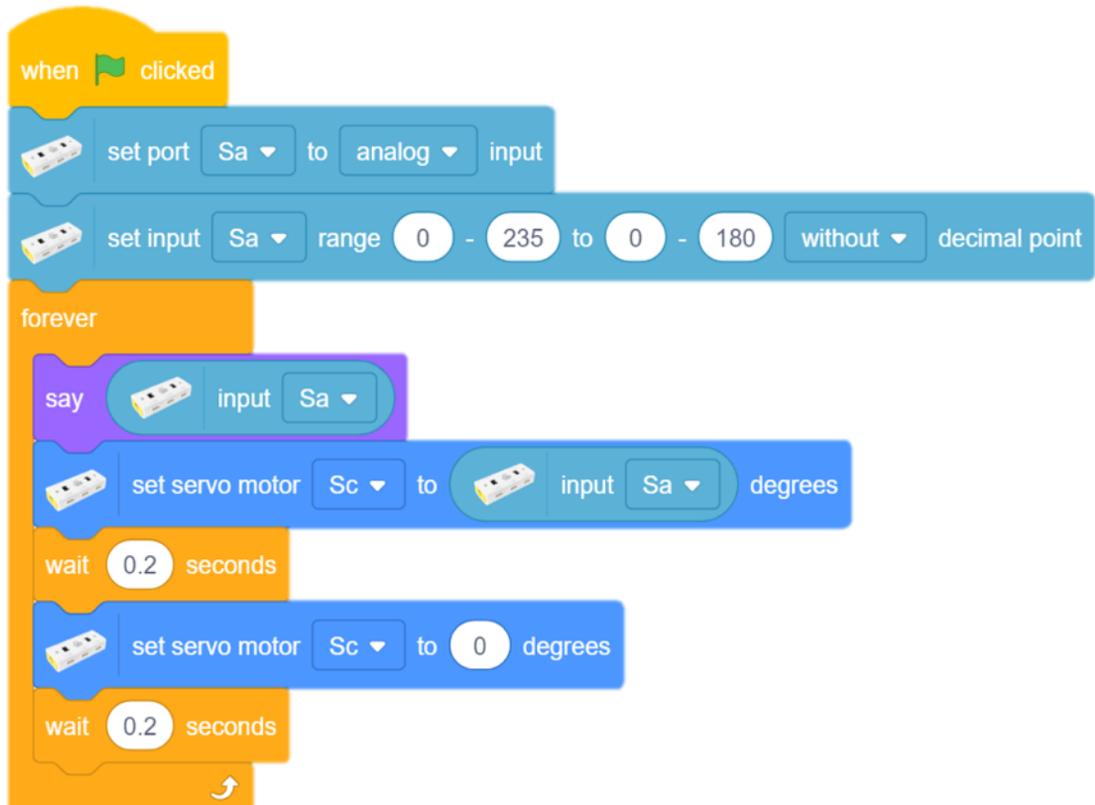


[Assignment 1] You want to change the range of your input sensor values to a different range. Check the command block you need.

- ①  rotate step motor 100 steps with velocity 300 step/sec
- ②  set input Sa range 0 - 255 to 0 - 100 without decimal point
- ③  set servo motor Sc to 0 degrees
- ④  set port Sa to analog input



[Assignment 2] You want to increase the rotational speed of the servo motor. See the following commands and figure out which block needs to be changed.



```
when clicked
  set port Sa to analog input
  set input Sa range 0 - 235 to 0 - 180 without decimal point
  forever
    say input Sa
    set servo motor Sc to input Sa degrees
    wait 0.2 seconds
    set servo motor Sc to 0 degrees
    wait 0.2 seconds
```

The script starts with a 'when clicked' event. It then sets port Sa to analog input and configures the input range from 0-235 to 0-180 without a decimal point. A 'forever' loop follows, containing: a 'say' block for input Sa, a 'set servo motor Sc to input Sa degrees' block, a 'wait 0.2 seconds' block, a 'set servo motor Sc to 0 degrees' block, and another 'wait 0.2 seconds' block.

# 10 Learning about the Hamster Robot

You can connect the Hamster Robot to your computer and use it to program.



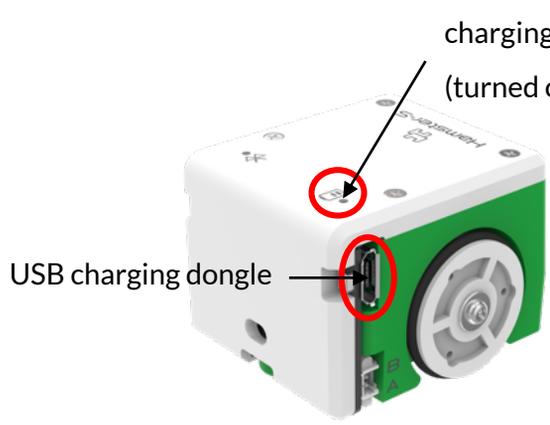
Let's learn about the Hamster Robot.



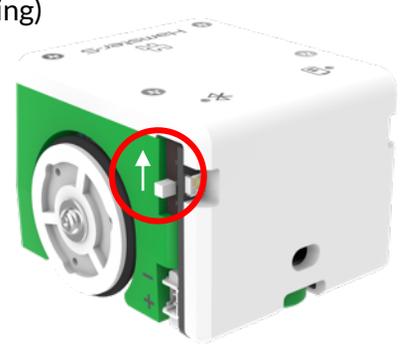
It can program 10 sensors, 2 LEDs, and 2 DC motors.



Charge the Hamster Robot and turn on the power button.



Use a micro USB cable to charge.

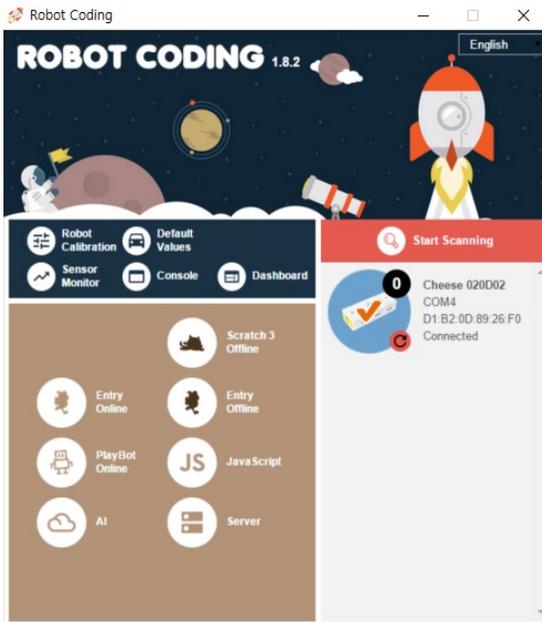


Turn on the power by flipping up the switch and off by flipping down.

## 10. Learning about the Hamster Robot



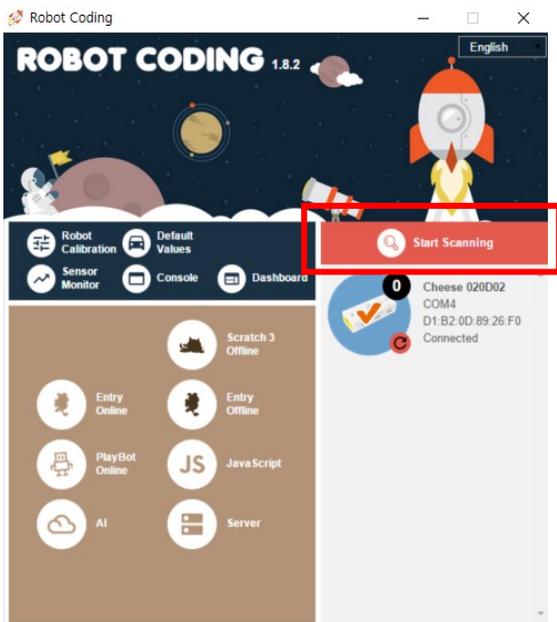
Let's connect the Cheese Stick and the Hamster Robot to the Scratch.



Connect the Cheese Stick to your computer using the Robot Coding software.

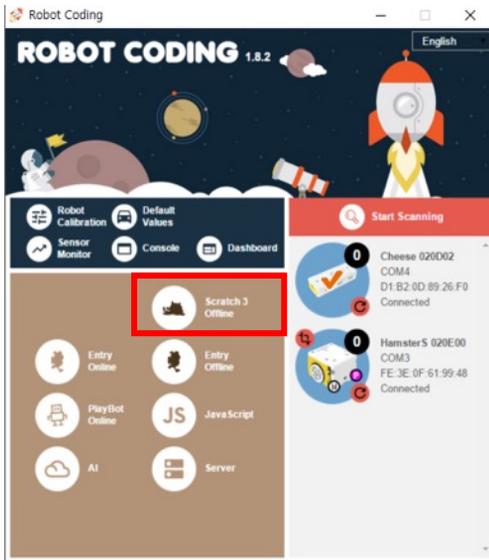


Plug another USB dongle into your computer and turn on the Hamster Robot.

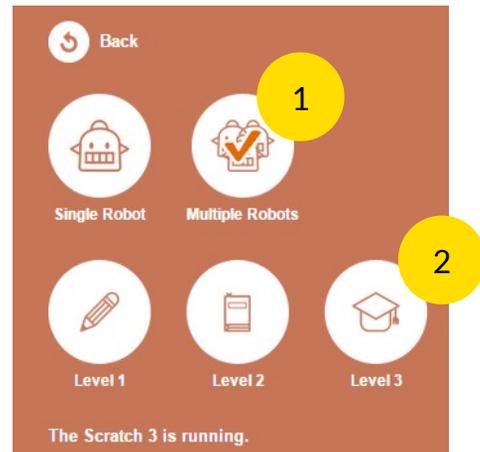
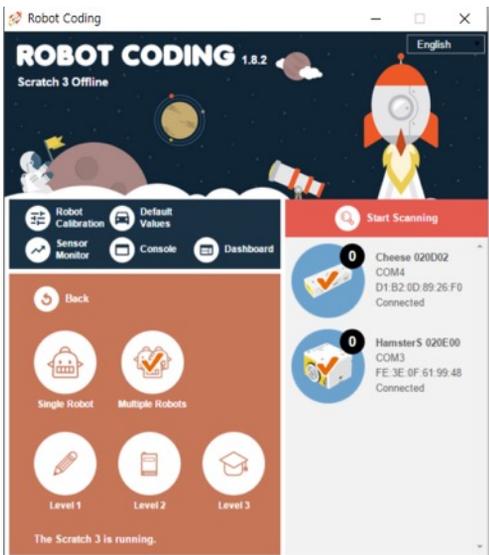


Click [Start Scanning].

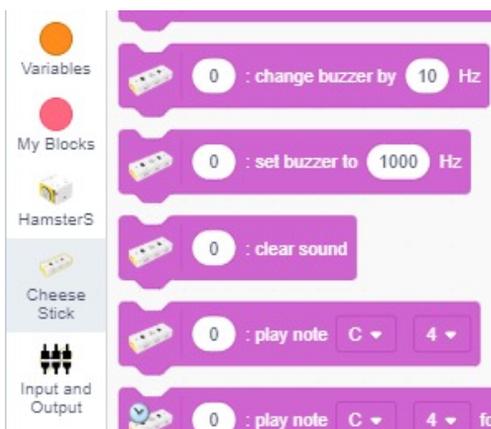
## 10. Learning about the Hamster Robot



Once the Cheese Stick and the Hamster Robot are connected to the Scratch Cat, click [Scratch 3 Offline].



Click [Multi-Robots] then [Level 3].

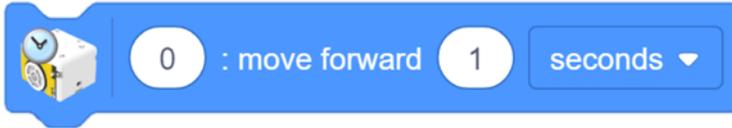


Check that both [Cheese Stick] and [HamsterS] tabs are created.

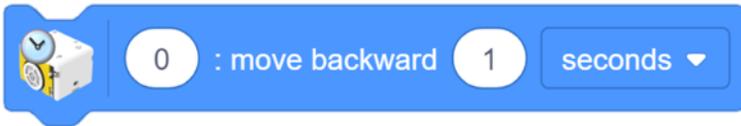
## 10. Learning about the Hamster Robot



Let's learn the blocks for this chapter.



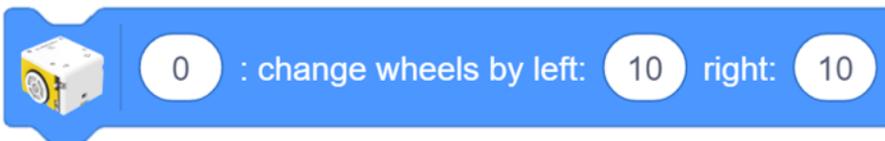
This block moves the Hamster forward for 1 second.



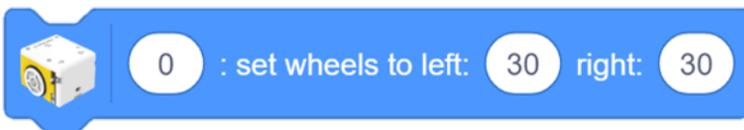
This block moves the Hamster backward for 1 second.



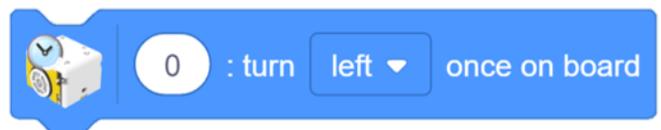
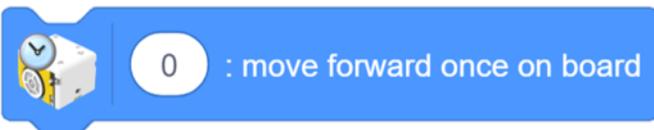
This block turns the Hamster left(right) for 1 second.



This block changes the speed of the Hamster's wheels. Positive numbers move the Hamster forward and negative numbers backward.



This block sets the speed of the Hamster's wheels. Positive numbers move the Hamster forward and negative numbers backward.



The Hamster is programmed to recognize one block as the distance between two dots on black grid lines. You can draw a grid with a marker or sharpie.

## 10. Learning about the Hamster Robot



Let's program the Hamster to move.

when green flag clicked : move forward 0 1 seconds

when green flag clicked : move backward 0 1 seconds

when green flag clicked

forever

: move forward 0 1 seconds

: move backward 0 1 seconds

when green flag clicked : turn left 0 1 seconds in place

when green flag clicked : turn right 0 1 seconds in place



You can move the Hamster with following blocks.

: move forward 0 cm

: move backward 0 cm

: turn left 0 degrees in place

: turn right 0 degrees in place

However, you can't control the speed of the Hamster with this block.

## 10. Learning about the Hamster Robot

when  clicked

 0 : set wheels to left: 30 right: 30

when  clicked

 0 : set wheels to left: -30 right: -30

when  clicked

forever

 0 : set wheels to left: 30 right: 30

wait 1 seconds

 0 : set wheels to left: -30 right: -30

wait 1 seconds

Use  0 : set wheels to left: 30 right: 30 button to change the Hamster's speed.

To control the time, use  wait  seconds block.

when  clicked

forever

 0 : change wheels by left: 30 right: 30

when  clicked

forever

 0 : set wheels to left: 30 right: 30



 0 : set wheels to left: 30 right: 30 and  0 : change wheels by left: 10 right: 10 blocks

have different functions.

 0 : set wheels to left: 30 right: 30 block maintains the Hamster's speed at the given value,

while  0 : change wheels by left: 10 right: 10 block moves the Hamster at a steadily increasing speed that is given. Play with both commands and learn how they are different.

## 10. Learning about the Hamster Robot

when  clicked

 : set wheels to left:  right:

rotate left in place about the center

when  clicked

 : set wheels to left:  right:

rotate right in place about the center

when  clicked

 : set wheels to left:  right:

rotate in place about the left wheel

when  clicked

 : set wheels to left:  right:

rotate in place about the right wheel

when  clicked

 : set wheels to left:  right:

rotate right while moving forward

when  clicked

 : set wheels to left:  right:

rotate left while moving forward



Play with



: set wheels to left:  right:

block and learn different ways of

rotating the Hamster.

## 10. Learning about the Hamster Robot



Let's review this chapter with the following assignments.



[Assignment 1] Write how the Hamster would move when you run the following commands.

①

```
when clicked clicked
forever
  0 : set wheels to left: 30 right: 30
  wait 1 seconds
  0 : set wheels to left: -30 right: -30
  wait 1 seconds
```

②

```
when clicked clicked
forever
  0 : change wheels by left: 30 right: 30
```

③

```
when clicked clicked
forever
  0 : set wheels to left: 30 right: 30
```

## 10. Learning about the Hamster Robot



Here is another assignment.



[Assignment 2] Write how the Hamster would move when you run each of the following commands.

①

when  clicked

 : set wheels to left:  right:

②

when  clicked

 : set wheels to left:  right:

③

when  clicked

 : set wheels to left:  right:

# 11 Making a Hamster Robot Controller

You can program the Hamster Robot with a joystick.



You can move the Hamster in different directions with a joystick.



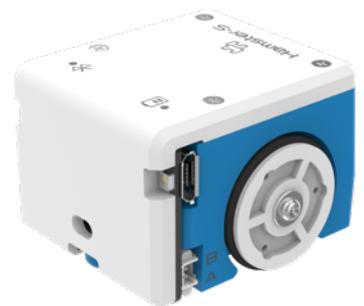
Let's get ready.



joystick



Cheese Stick



Hamster Robot

## 11. Making a Hamster Robot Controller



Let's learn the blocks for this chapter.



0 : start PID-13 joystick and button ▾

This block is needed to use a joystick.



0 : PID: x1 ▾

This block sets the horizontal and vertical values of a joystick.

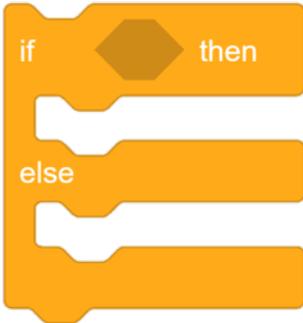


0 : PID: set x1 ▾ range 0 - 255 to 0 - 100 without ▾ decimal point

This block sets the range of joystick values as 0 to 100.



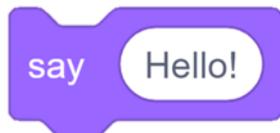
This block compares two values to run a command.



This block runs different commands depending on whether the condition is true or not.



This block maintains the running command for the given number of seconds.

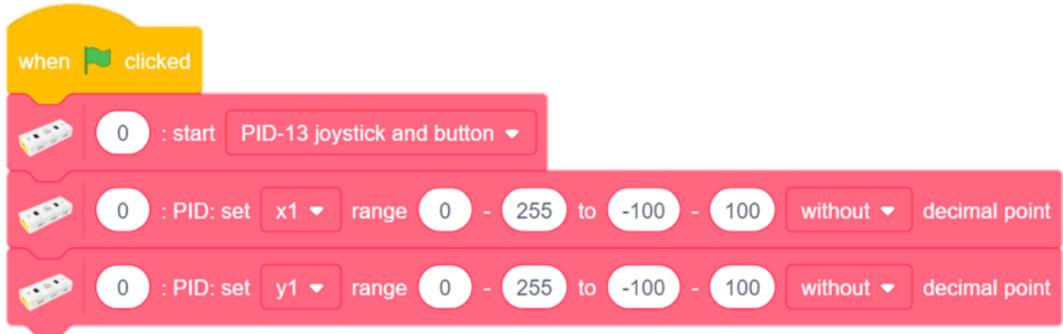


This block allows the sprite to speak through speech bubbles.

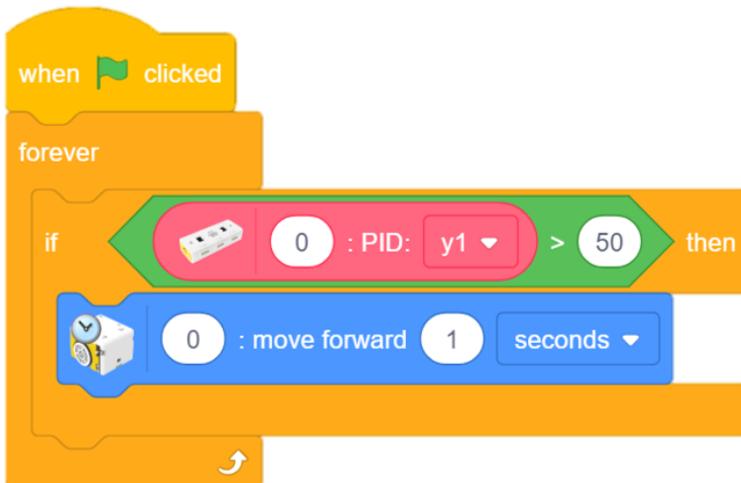
## 11. Making a Hamster Robot Controller



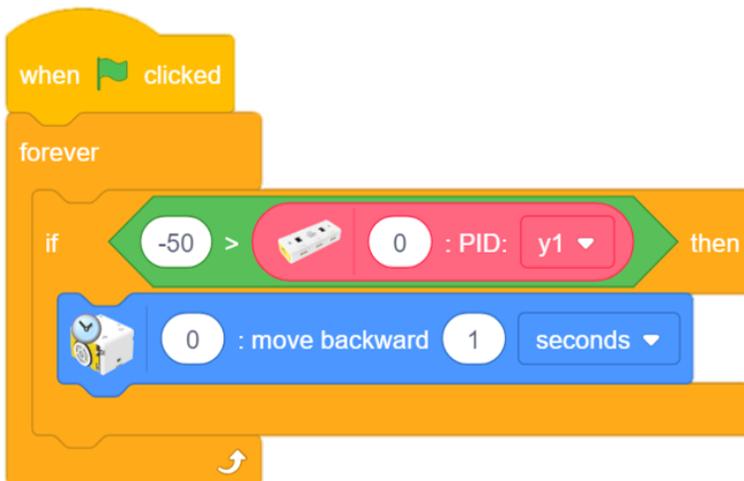
Let's use the joystick to move the Hamster.



x1 and y1 each refers to the horizontal and vertical range of joystick values.



Let's program the Hamster to move forward for one second when you move the joystick up.



Let's program the Hamster to move backward for one second when you move the joystick down.

## 11. Making a Hamster Robot Controller

```
when clicked
  0 : start PID-13 joystick and button
  0 : PID: set x1 range 0 - 255 to -100 - 100 without decimal point
  0 : PID: set y1 range 0 - 255 to -100 - 100 without decimal point
  forever
    if 0 : PID: x1 > 50 then
      0 : turn left 1 seconds in place
```

Let's program the Hamster rotate left for one second when you move the joystick left.

```
when clicked
  0 : start PID-13 joystick and button
  0 : PID: set x1 range 0 - 255 to -100 - 100 without decimal point
  0 : PID: set y1 range 0 - 255 to -100 - 100 without decimal point
  forever
    if 0 : PID: x1 > 50 then
      0 : turn right 1 seconds in place
```

Let's program the Hamster to rotate right for one second when you move the joystick right.

## 11. Making a Hamster Robot Controller



Use the joystick to move the Hamster left, right, forward and backward.

```
when green flag clicked
  start PID-13 joystick and button
  PID: set x1 range 0 - 255 to -100 - 100 without decimal point
  PID: set y1 range 0 - 255 to -100 - 100 without decimal point
  wait 1 seconds
  forever loop
    if PID: y1 > 50 then
      move forward 1 seconds
    if -50 > PID: y1 then
      move backward 1 seconds
    if PID: x1 > 50 then
      turn right 1 seconds in place
    if -50 > PID: x1 then
      turn left 1 seconds in place
```

The code starts with a 'when green flag clicked' event. It initializes two PID controllers for the joystick's x1 and y1 axes, both with a range of 0 to 255 and a target range of -100 to 100. After a 1-second wait, it enters a 'forever' loop. Inside the loop, four conditional blocks check the joystick's position: if y1 is greater than 50, move forward 1 second; if y1 is less than -50, move backward 1 second; if x1 is greater than 50, turn right 1 second in place; if x1 is less than -50, turn left 1 second in place.



When you click the start button, some machines calculate the joystick's value as -100 for a brief moment. In order to prevent the Hamster from moving at this speed, use

wait 1 seconds

block before adding other blocks.

## 11. Making a Hamster Robot Controller

```
when clicked
  0 : start PID-13 joystick and button
  0 : PID: set x1 range 0 - 255 to -100 - 100 without decimal point
  0 : PID: set y1 range 0 - 255 to -100 - 100 without decimal point
  wait 1 seconds
  forever
    if 0 : PID: y1 > 50 then
      0 : set wheels to left: 30 right: 30
    if -50 > 0 : PID: y1 then
      0 : set wheels to left: -30 right: -30
    if 0 : PID: x1 > 50 then
      0 : set wheels to left: 30 right: 0
    if -50 > 0 : PID: x1 then
      0 : set wheels to left: 0 right: 30
    if 0 : PID: button 1 clicked? then
      0 : set wheels to left: 0 right: 0
```



Add a command to stop the Hamster before changing the speed of its wheels. You can stop the Hamster by setting the speed of both wheels to 0.



If you program the Hamster to move for the given time, it can only run a single command at a time. However, if you program it to move at the given speed, you can run multiple commands.

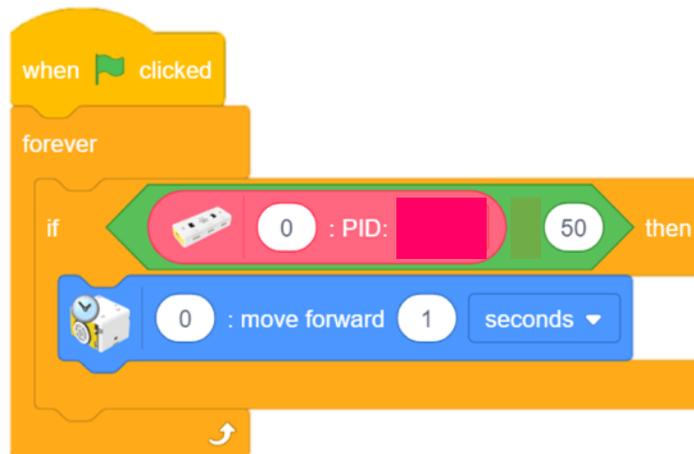
## 11. Making a Hamster Robot Controller



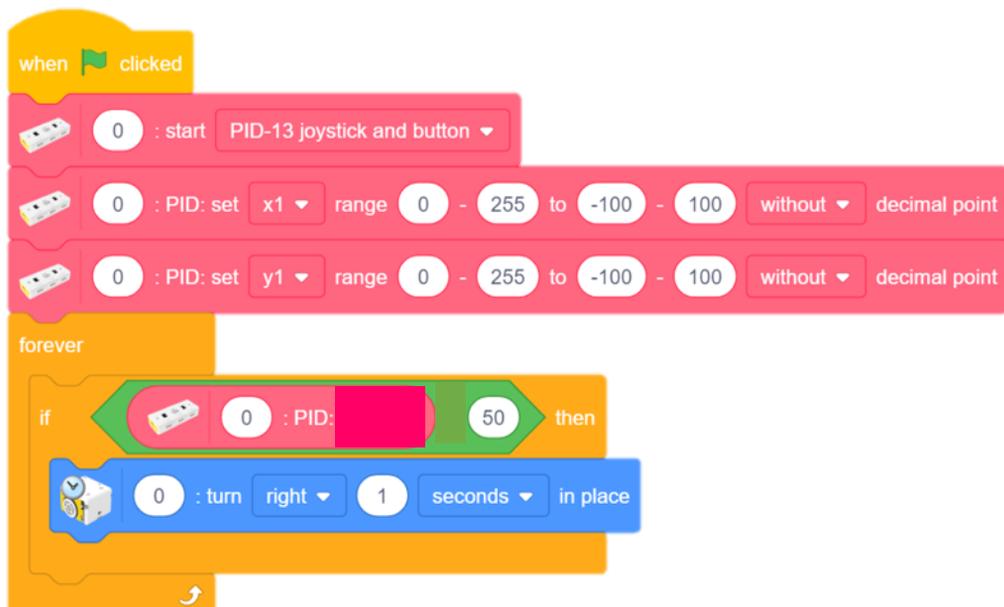
Let's review this chapter with the following assignments.



[Assignment 1] You want to program the Hamster to move forward when you move the joystick up. Fill in the blank with the command you need.



[Assignment 2] You want to program the Hamster to rotate right for one second when you move the joystick right. Fill in the blank with the command you need.



## 11. Making a Hamster Robot Controller



Here is another assignment.



[Assignment 3] When you click the start button, some machines calculate the joystick's value as -100 for a brief moment. Because of this, the Hamster might move at this speed. Figure out how to resolve this issue.

```
when green flag clicked
  start PID-13 joystick and button
  PID: set x1 range 0 - 255 to -100 - 100 without decimal point
  PID: set y1 range 0 - 255 to -100 - 100 without decimal point
  wait 1 seconds
  forever
    if PID: y1 > 50 then
      set wheels to left: 30 right: 30
    if -50 > PID: y1 then
      set wheels to left: -30 right: -30
    if PID: x1 > 50 then
      set wheels to left: 30 right: 0
    if -50 > PID: x1 then
      set wheels to left: 0 right: 30
    if PID: button 1 clicked ? then
      set wheels to left: 0 right: 0
```

The image shows a Scratch script for a Hamster robot controller. The script starts with a 'when green flag clicked' event. It then performs three 'PID: set' blocks for x1, y1, and y1, each with a range of 0 to 255 and a target range of -100 to 100. After a 1-second wait, it enters a 'forever' loop. Inside the loop, there are five 'if' blocks. The first 'if' block checks if PID: y1 is greater than 50, and if true, sets wheels to left: 30 and right: 30. The second 'if' block checks if -50 is greater than PID: y1, and if true, sets wheels to left: -30 and right: -30. The third 'if' block checks if PID: x1 is greater than 50, and if true, sets wheels to left: 30 and right: 0. The fourth 'if' block checks if -50 is greater than PID: x1, and if true, sets wheels to left: 0 and right: 30. The fifth 'if' block checks if PID: button 1 is clicked, and if true, sets wheels to left: 0 and right: 0.

# 12 Making a Landmine Detecting Robot

The Hamster Robot has floor and proximity sensors. You can turn the Hamster into a landmine detecting robot.



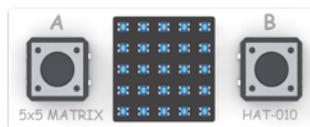
Let's control the Hamster Robot with the Cheese Stick and a potentiometer.



Let's get ready.



joystick



5x5 LED Matrix



Cheese Stick

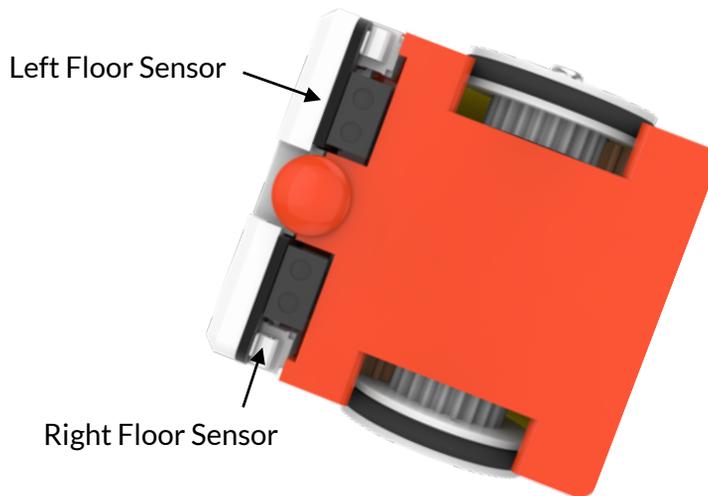


Hamster Robot

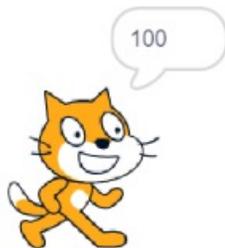
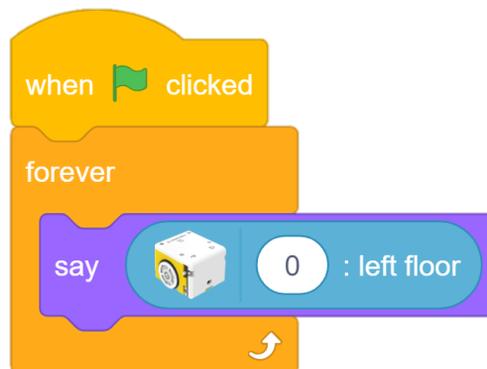
## 12. Making a Landmine Detecting Robot



Let's learn about the Hamster's floor sensors.



The Hamster's floor sensors use the infrared(IR) light to detect levels from 0 to 255. On a piece of paper, draw landmines with a black marker and see how the floor sensors detect the landmines and the blank space on the paper differently.



On the blank space



On the landmines

## 12. Making a Landmine Detecting Robot



Let's build a program to control the Hamster with a joystick.

```
when clicked
  0 : start PID-13 joystick and button
  0 : PID: set x1 range 0 - 255 to -100 - 100 without decimal point
  0 : PID: set y1 range 0 - 255 to -100 - 100 without decimal point
```

```
when clicked
  wait 1 seconds
  forever
    if 0 : PID: y1 > 50 then
      0 : set wheels to left: 30 right: 30
    if -50 > 0 : PID: y1 then
      0 : set wheels to left: -30 right: -30
    if 0 : PID: x1 > 50 then
      0 : set wheels to left: 30 right: 0
    if -50 > 0 : PID: x1 then
      0 : set wheels to left: 0 right: 30
    if 0 : PID: button 1 clicked ? then
      0 : set wheels to left: 0 right: 0
```

## 12. Making a Landmine Detecting Robot



Let's program the Hamster to detect landmines.

The image shows two identical Scratch code blocks. Each block is an orange 'if-then-else' structure. The 'if' block contains a green comparison block with the number '20' in a white circle, a greater-than sign '>', a Hamster robot icon, the number '0' in a white circle, and the text ': left floor'. The 'then' block is empty. The 'else' block is also empty.

The Hamster should be able to run commands even when only one of its wheels detects landmines. Enter the commands as shown above.

The image shows a single Scratch code block. It is an orange 'if-then-else' structure. The 'if' block contains a green comparison block with the number '20' in a white circle, a greater-than sign '>', a Hamster robot icon, the number '0' in a white circle, and the text ': left floor'. This is followed by a green 'or' block, another Hamster robot icon, the number '0' in a white circle, and the text ': right floor'. The 'then' block is empty. The 'else' block is also empty.

You can combine two commands into one with  or  block under the

 Operators tab.

## 12. Making a Landmine Detecting Robot

```
when clicked
  0 : start HAT-010 5x5 matrix
  forever
    if <math>20 > \text{left floor}</math> or <math>20 > \text{right floor}</math> then
      0 : HAT-010 background: draw red X at x: 0 y: 0
    else
      0 : HAT-010: clear background
```

```
when clicked
  0 : start HAT-010 5x5 matrix
  0 : start PID-13 joystick and button
  0 : PID: set x1 range 0 - 255 to -100 - 100 without decimal point
  0 : PID: set y1 range 0 - 255 to -100 - 100 without decimal point
```



To program the LED Matrix to display an X when the Hamster detects landmines, you need

```
0 : start HAT-010 5x5 matrix
```

 block.

If you want a warning sound with the O and X on the LED Matrix, use

```
0 : set buzzer to 1000 Hz
```

 block.

Use 

```
0 : clear sound
```

 block to turn the sound off when you are away from the landmines.

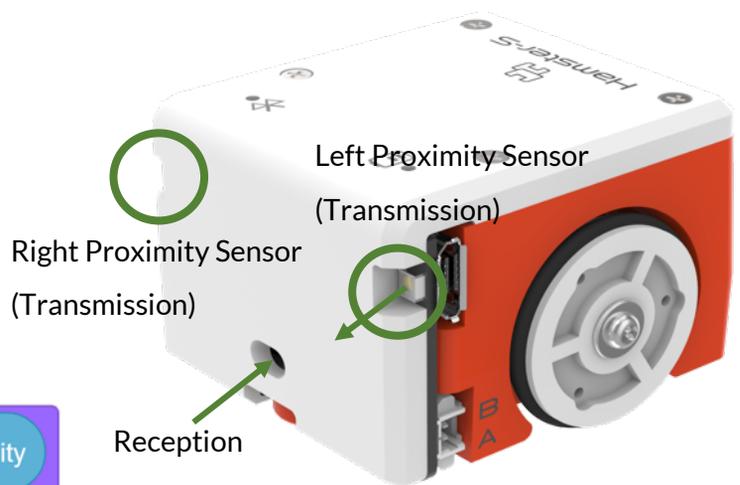
## 12. Making a Landmine Detecting Robot

```
when clicked
  start HAT-010 5x5 matrix
  forever
    if (20 > left floor) or (20 > right floor) then
      HAT-010 background: draw red X at x: 0 y: 0
      set buzzer to 1000 Hz
    else
      HAT-010: clear background
      clear sound
```



Let's learn about the Hamster's proximity sensors.

```
when clicked
  forever
    say left proximity
```



The proximity sensors can detect obstacles with the IR lights within the distance of 1-30mm. Play with the proximity sensors by bringing your hand close to them and learn how they detect obstacles.

## 12. Making a Landmine Detecting Robot



When obstacles are far away



When obstacles are near



When obstacles are too close



Let's build a program to prevent collision.

```
when clicked
  0 : start HAT-010 5x5 matrix
  forever
    if 0 : left proximity > 40 then
      0 : HAT-010 background: draw red dislike at x: 0 y: 0
    else
      0 : HAT-010: clear background
```

Program the 5x5 LED Matrix to display a hatred facial expression when the obstacle is close to the Hamster. You can also use

```
0 : set buzzer to 1000 Hz
```

block.

## 12. Making a Landmine Detecting Robot



Let's review this chapter with the following assignments.



[Assignment 1] Write the difference between running these two command blocks.

```
when clicked
  0 : start HAT-010 5x5 matrix
  forever
    if <20 > [0 : left floor] or <20 > [0 : right floor] then
      0 : HAT-010 background: draw red X at x: 0 y: 0
    else
      0 : HAT-010: clear background
      0 : clear sound
```

```
when clicked
  0 : start HAT-010 5x5 matrix
  forever
    if <20 > [0 : left floor] and <20 > [0 : right floor] then
      0 : HAT-010 background: draw red X at x: 0 y: 0
    else
      0 : HAT-010: clear background
      0 : clear sound
```

## 12. Making a Landmine Detecting Robot



[Assignment 2] You want to program the 5x5 LED Matrix to display a hatred facial expression when the Hamster detects an obstacle. Fill in the blanks with the commands you need.

```
when clicked
  0 : start HAT-010 5x5 matrix
  forever
    if 10 [Hamster] 0 [ ] then
      0 : HAT-010 background: draw red dislike at x: 0 y: 0
    else
      0 : HAT-010: clear background
```

The code consists of the following blocks:

- when clicked** (yellow)
- 0 : start HAT-010 5x5 matrix** (green)
- forever** (orange) loop containing:
  - if** (orange) block with:
    - Condition: **10** (green circle) **[Hamster]** (robot icon) **0** (white circle) **[ ]** (blue oval)
    - then** (green) block: **0 : HAT-010 background: draw** (green) **red** (dropdown) **dislike** (dropdown) **at x: 0 y: 0** (white circles)
  - else** (orange) block: **0 : HAT-010: clear** (green) **background** (dropdown)



[Assignment 3] You want program the Cheese Stick to make a warning sound when the Hamster detects landmines or obstacles. Check the block you need.

0 : set sound output to **internal speaker** (dropdown)

0 : set buzzer to **1000** (white circle) **Hz**

0 : set tempo to **60** (white circle) **BPM**

0 : clear sound