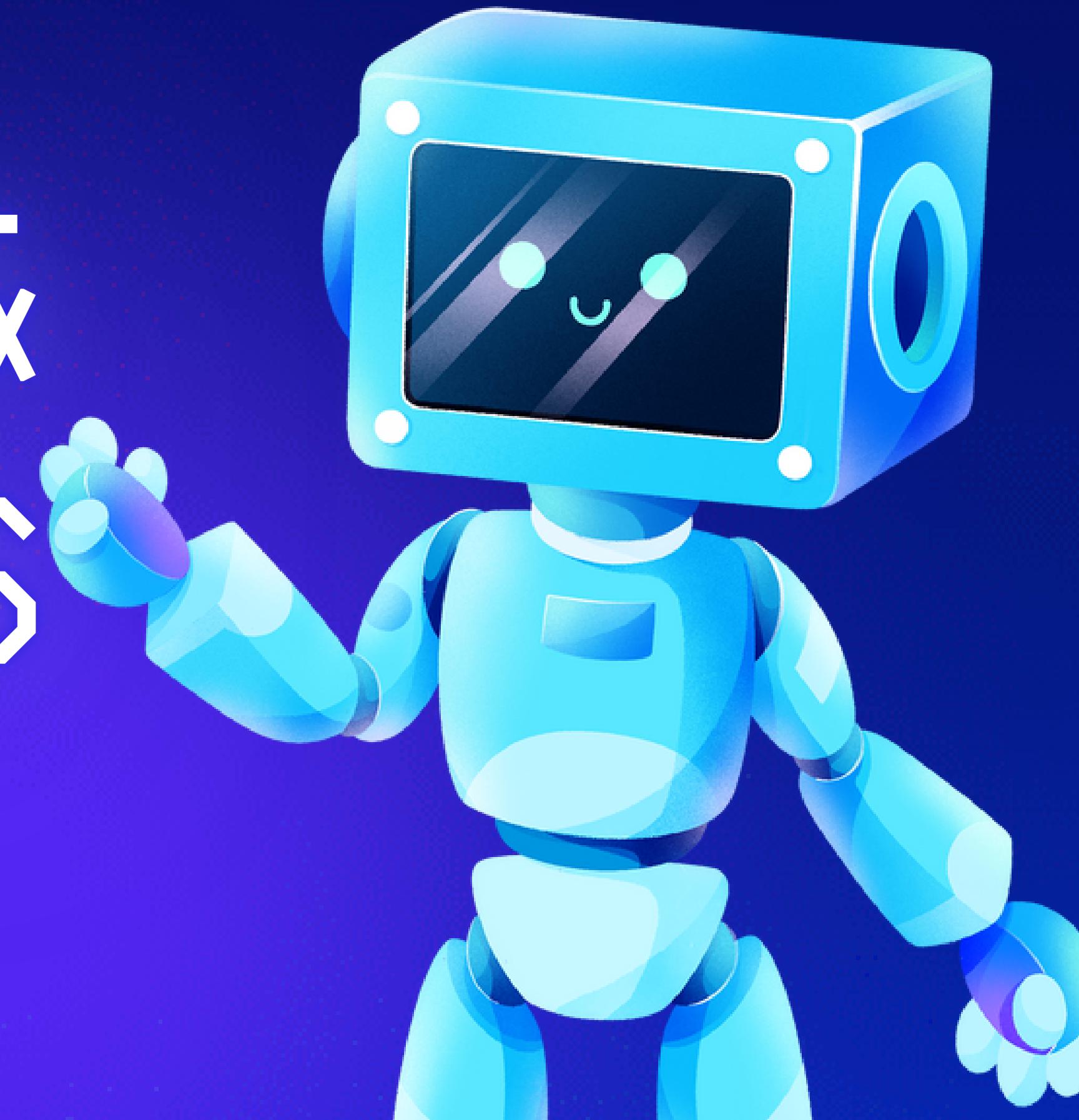




MILLBANK ACADEMY

CODING & ROBOTICS



17th January 2024

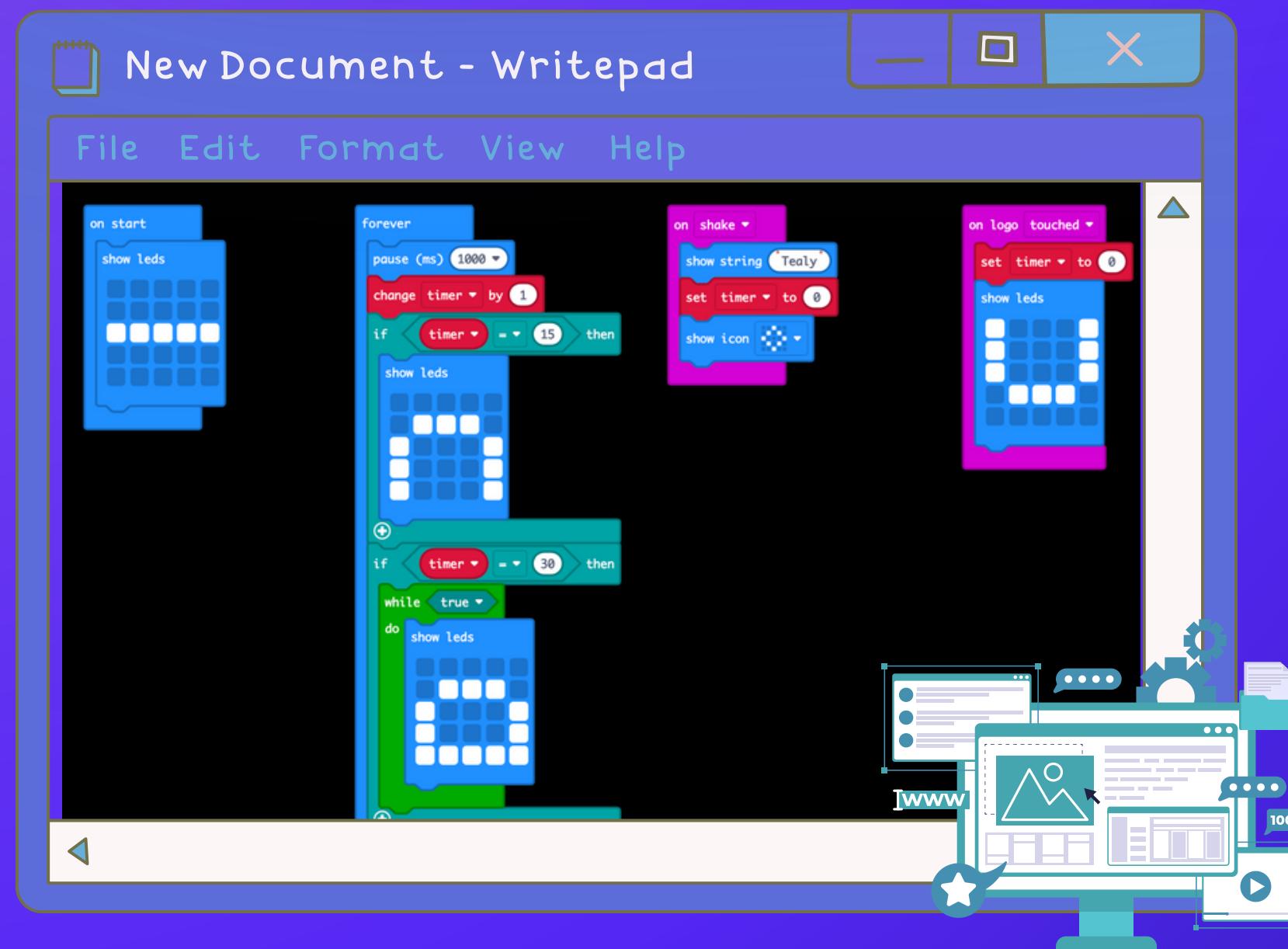
CODING </>



PROGRAMMING </>



PROGRAMMING LANGUAGES



The image shows a Node.js application running on a local server. The terminal window displays the application's logs, including messages about restarting due to changes and connecting to a MongoDB database. The code in the editor is for a 'Project' model, which extends a 'Schema' from 'mongoose'. It includes validation for 'picture' URLs and a 'ProjectSchema' definition with fields for 'name', 'location', 'description', 'picture', 'youtube', and 'timestamp'. The routes file ('choo.js') defines endpoints for creating, reading, updating, and deleting projects. On the right, there is an 'Image Editor' interface with an 'Adjustments' panel containing sliders for effects like 'Halftone', 'Emboss', 'Sharpen', 'Zoom Blur', 'Motion Blur', 'Gaussian Blur', 'Noise', 'Diffuse', and 'Mosaic'. There are also 'Filters', 'Effects', 'Resize', 'Export', and 'Upload' buttons.

```
const mongoose = require('mongoose');
const Schema = mongoose.Schema;

const projectSchema = new Schema({
  projectName: { type: String, required: true },
  year: { type: Number, required: true },
  location: { type: Object, required: true },
  description: { type: String, required: true },
  picture: { type: String, required: true },
  youtube: { type: String }
}, {
  timestamps: true
});

projectSchema.path('picture').validate((val) => {
  if(val) {
    const urlRegex = /^(http|https)?:\/\/(www\.)?([a-zA-Z0-9]+\.[a-zA-Z]{2,})?([a-zA-Z0-9]+[^\s]*)?$/;
    return urlRegex.test(val);
  }
  return true;
});

const Project = mongoose.model('Project', projectSchema);

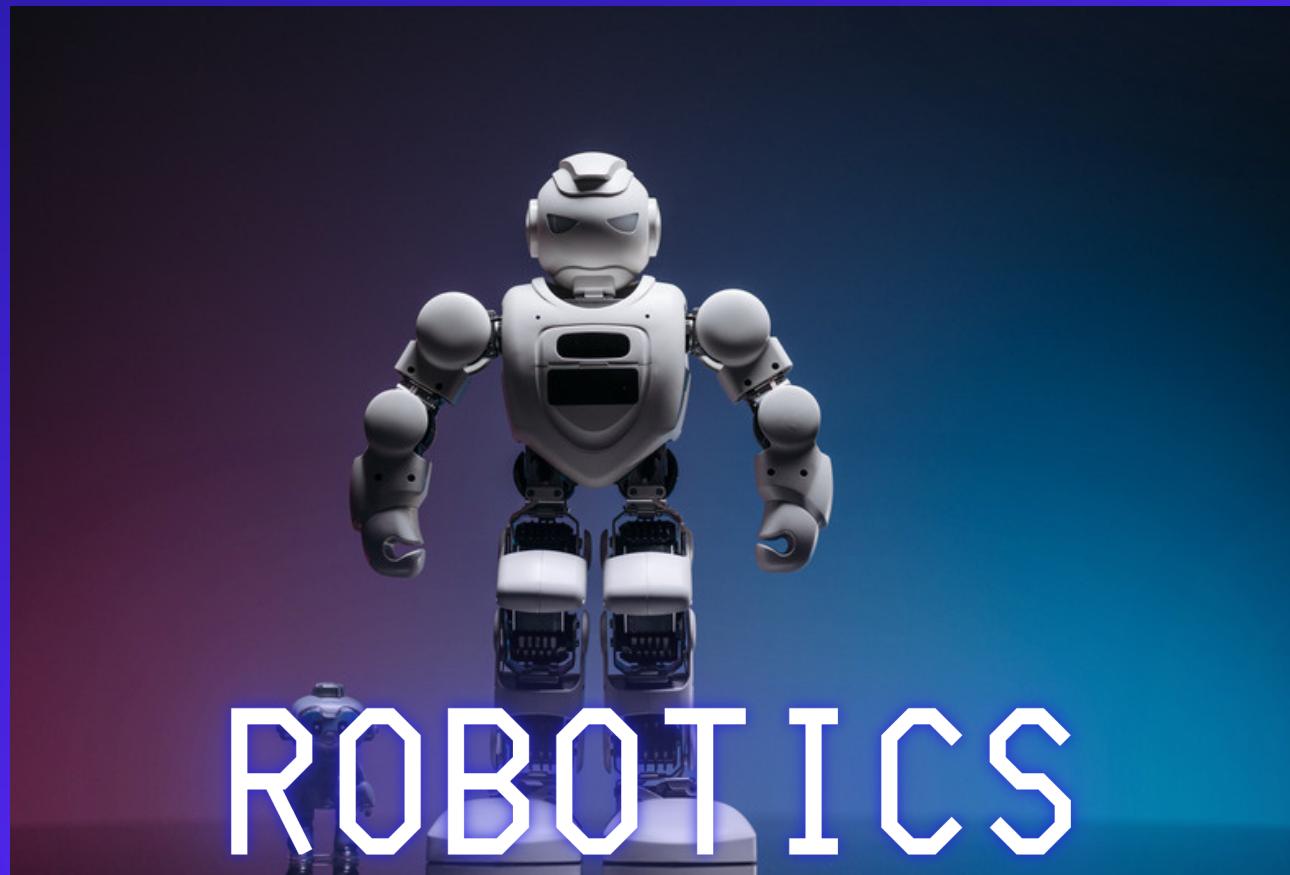
module.exports = Project;
```



VIDEO GAMES



WEBSITES



ROBOTICS



SMARTPHONE



01

17TH JAN
MICROPETS



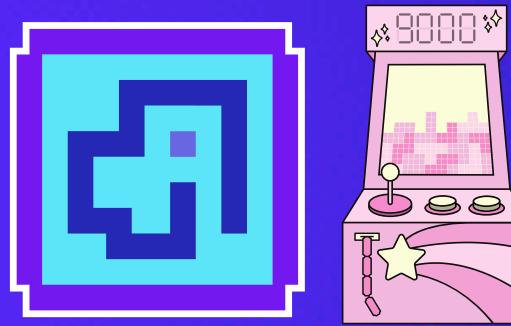
02

24TH JAN
MUSIC



03

31ST JAN
**SNAKE &
TETRIS**



04

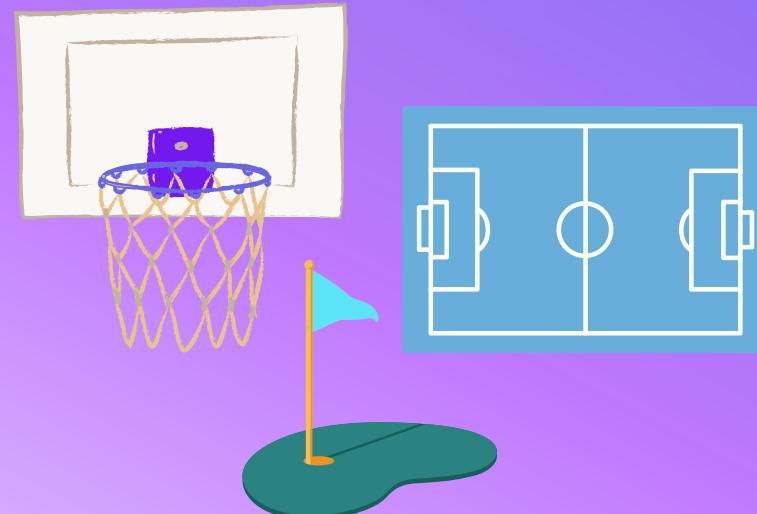
7TH FEB
**MAZE RUNNER,
BULLDOZER &
PICASSO**



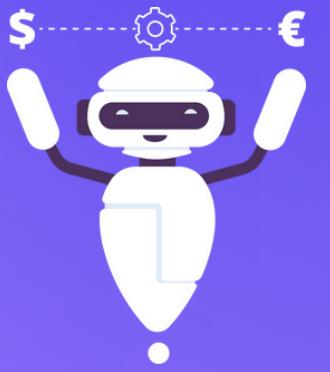
05



21ST FEB
SPORTS DAY



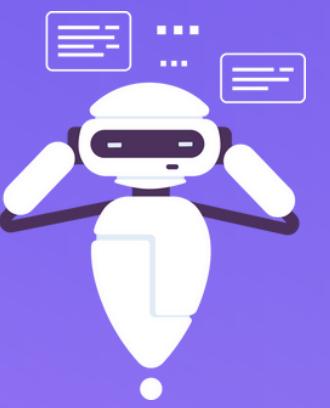
06



28TH FEB
MAKE YOUR
OWN SCRATCH
GAME



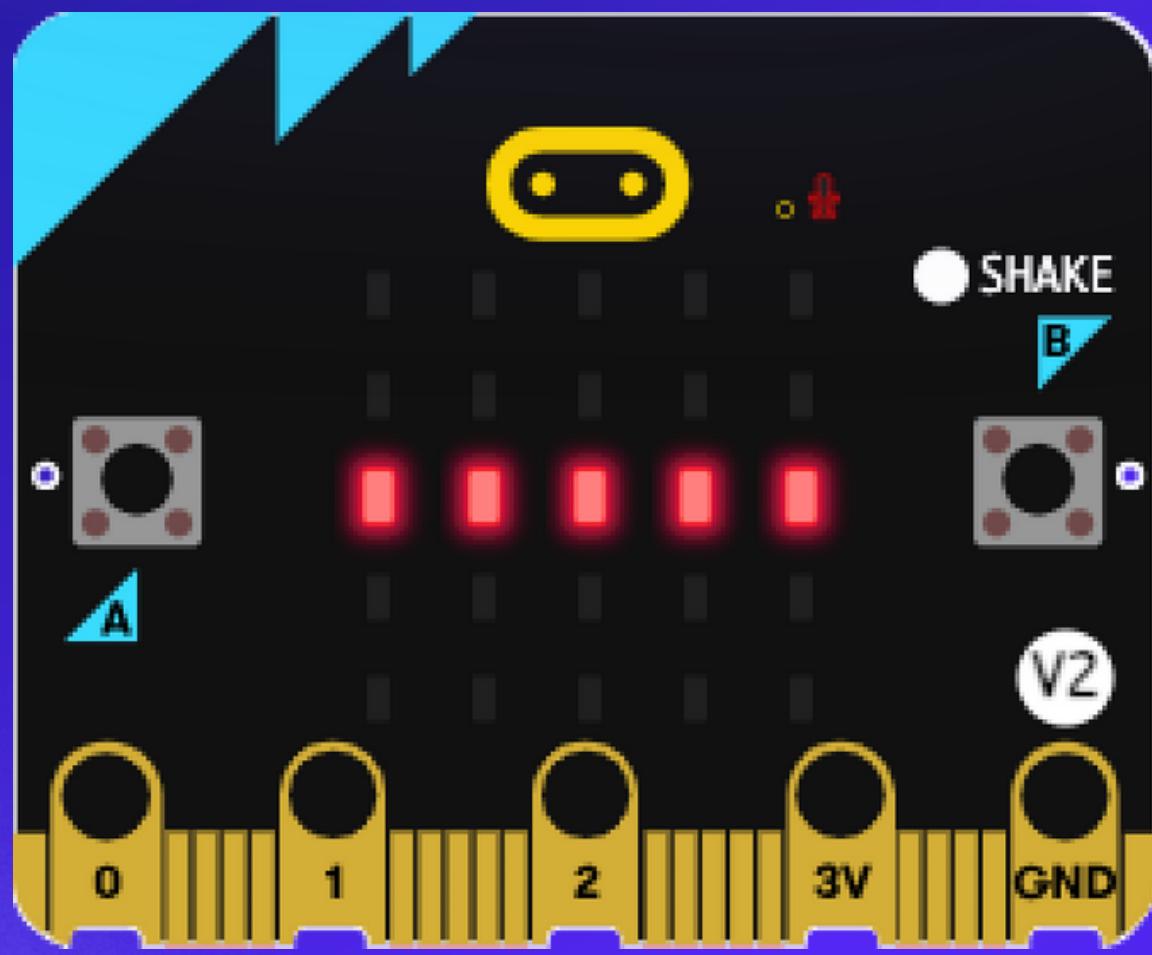
07



6TH MARCH
OTTOBOT
PARTY TIME

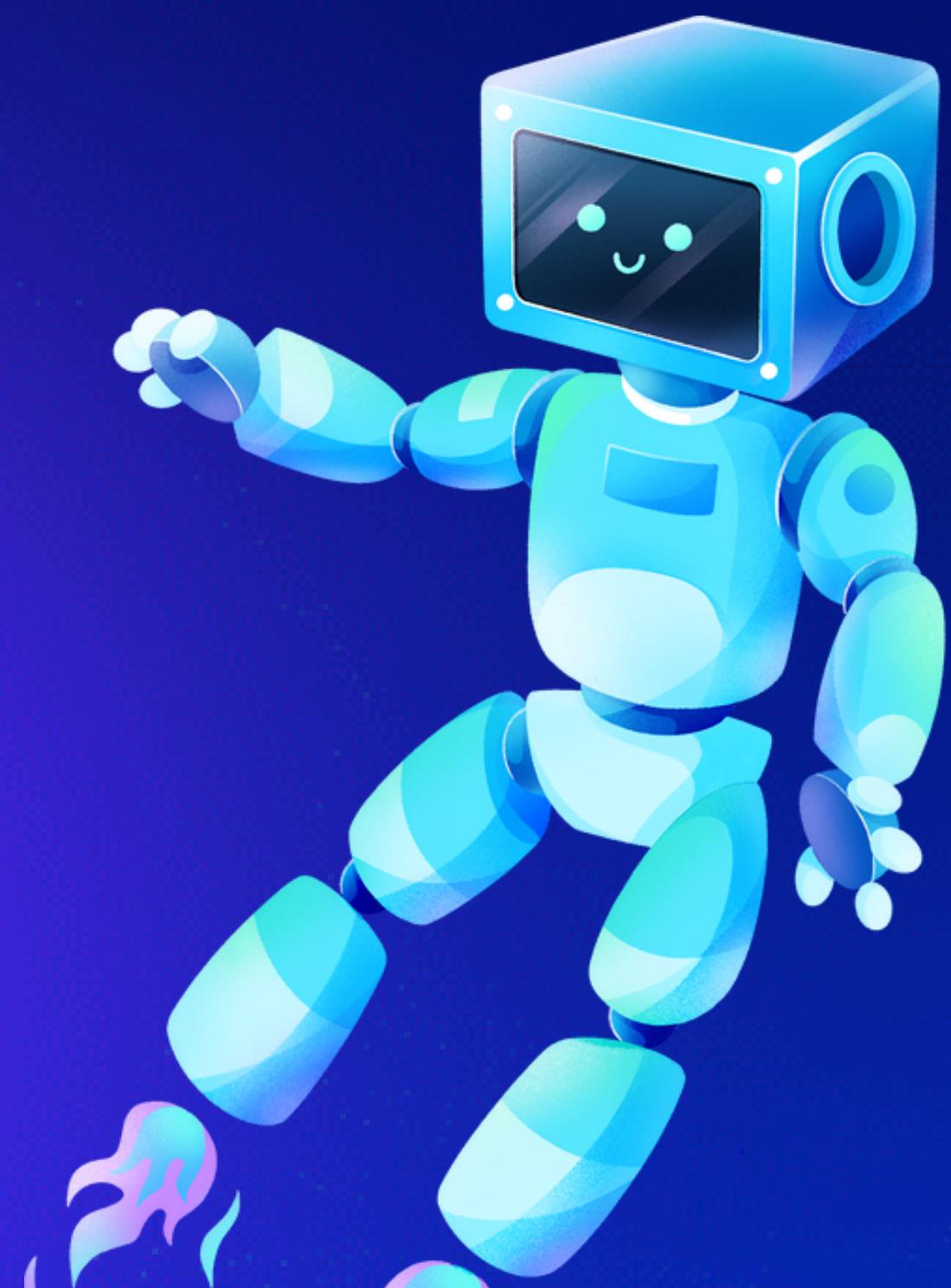


MICROBIT



VARIABLES

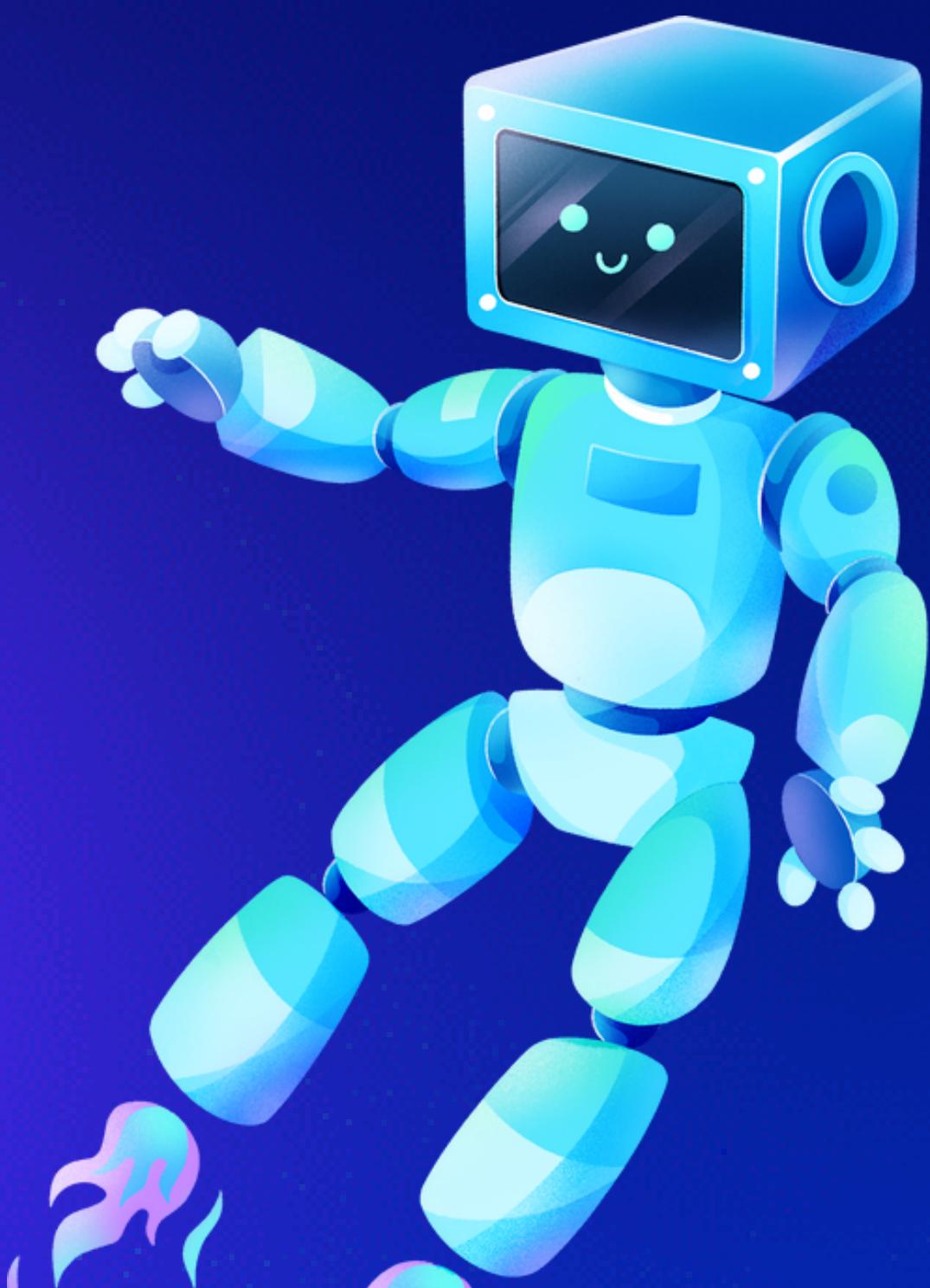
timer ▾



VARIABLES

timer ▾

set **timer** ▾ to 0

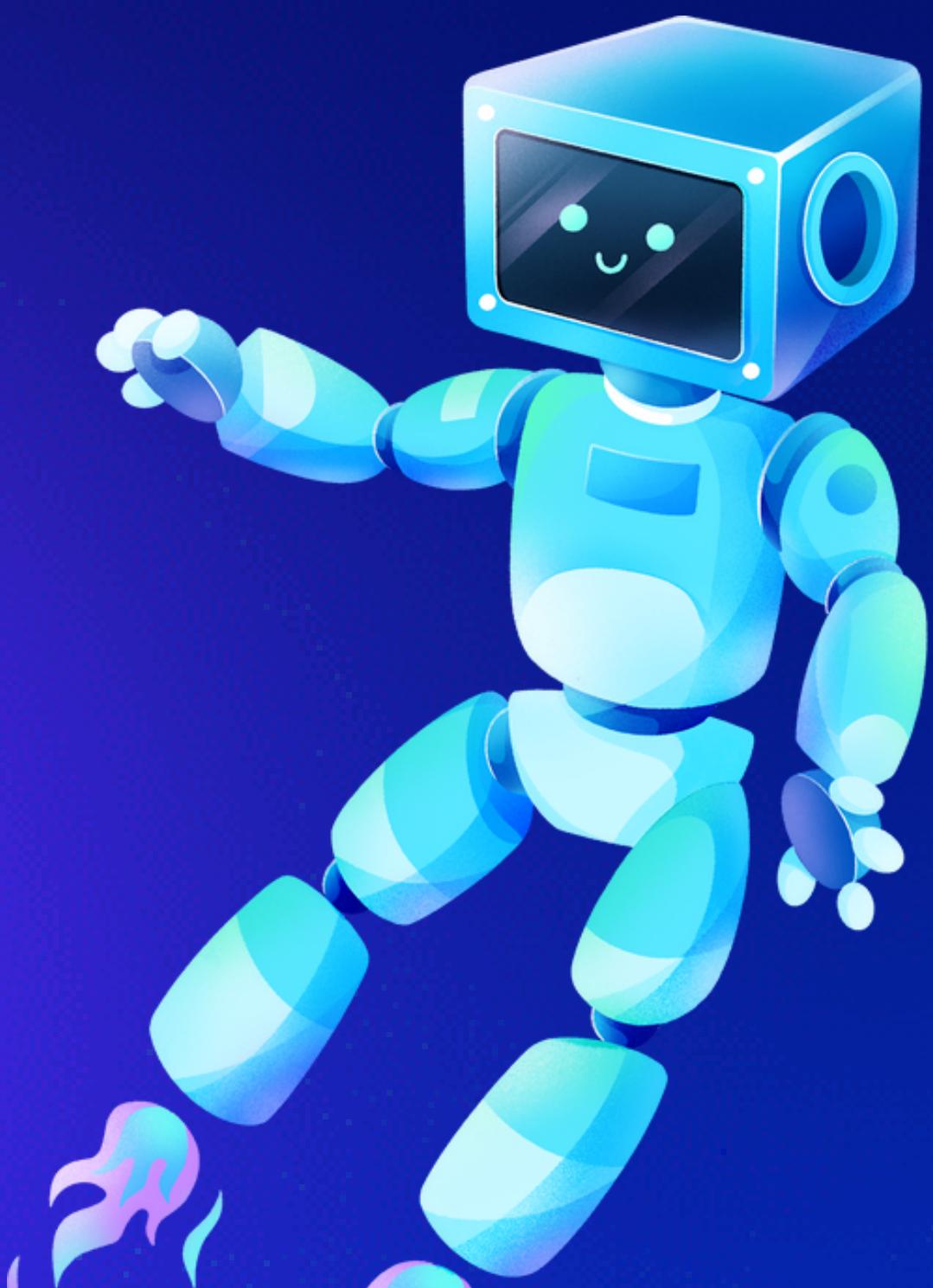


VARIABLES

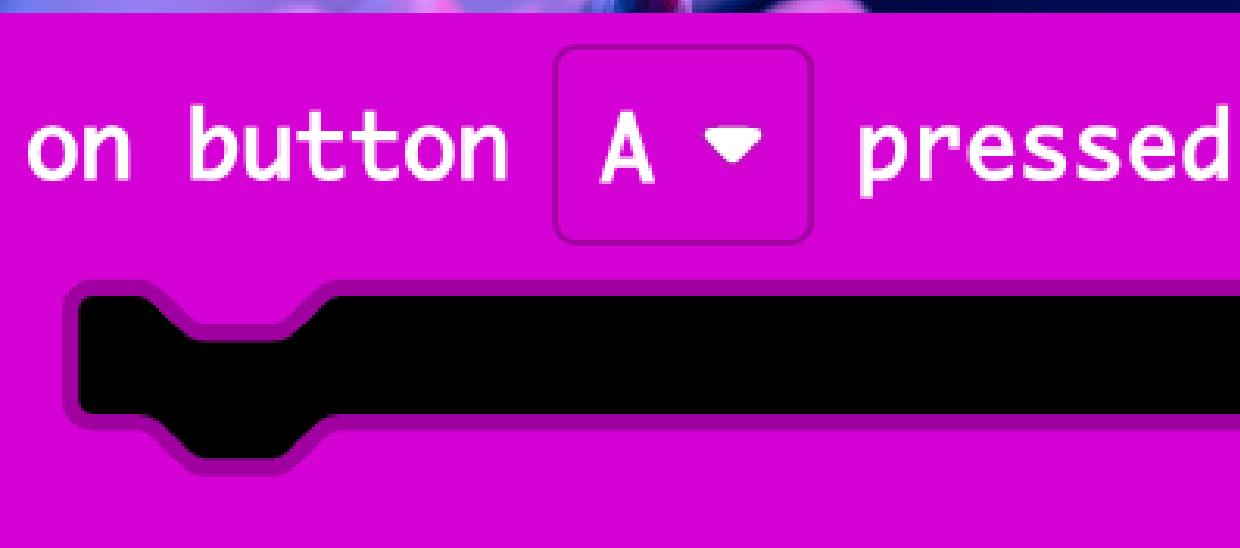
timer ▾

set timer ▾ to 0

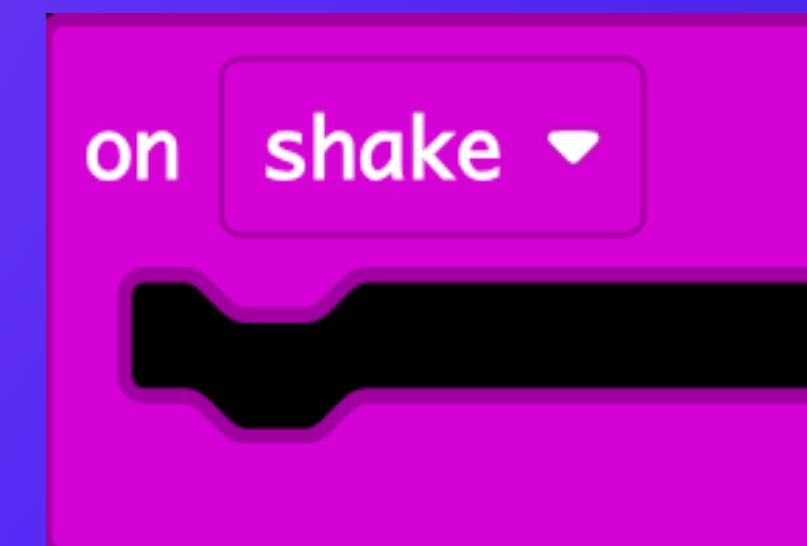
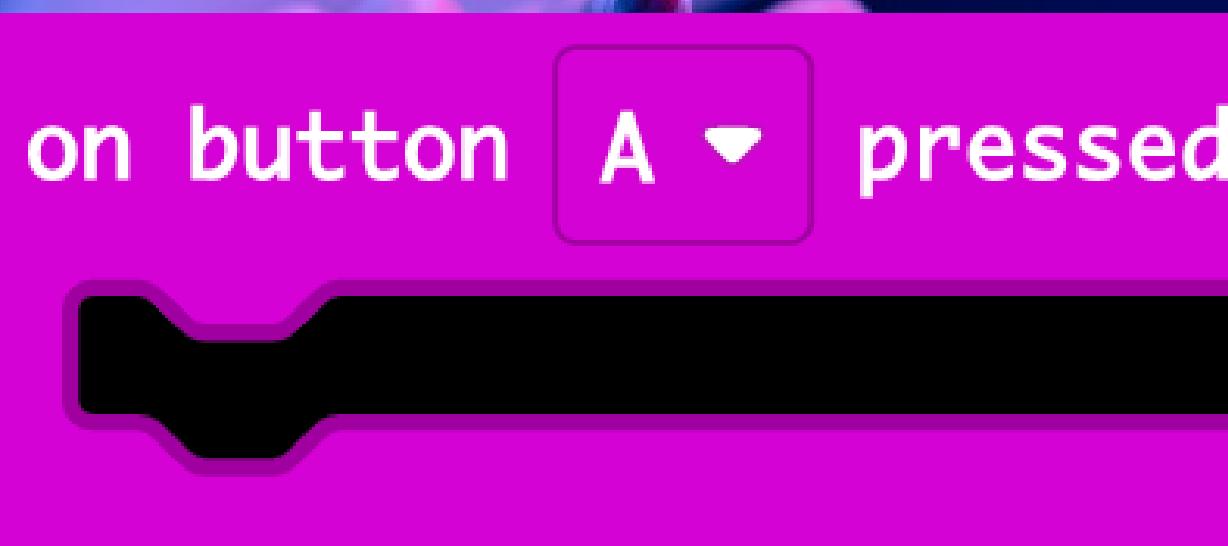
change timer ▾ by 1



INPUT



INPUT

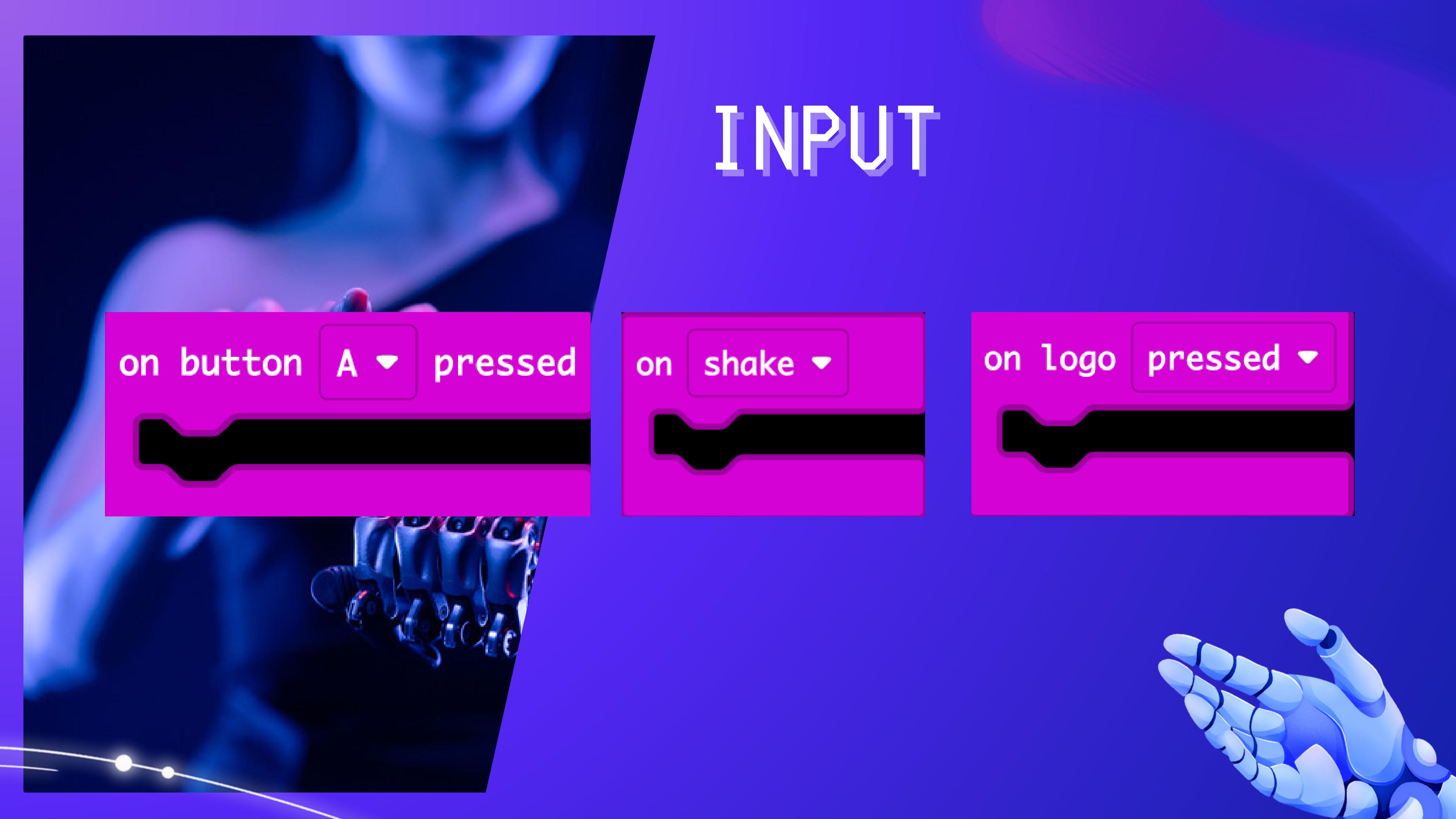


INPUT

on button A pressed

on shake

on logo pressed



OUTPUT

show number 0



OUTPUT

show number 0

show icon



OUTPUT

show number 0

show icon

show string "Hello!"



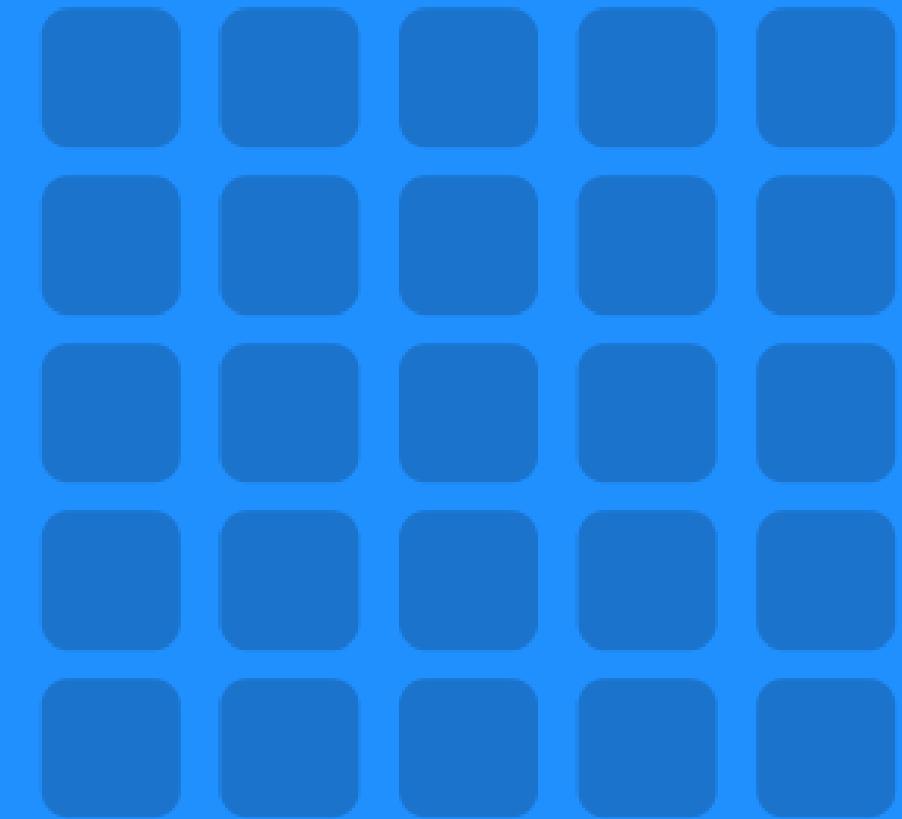
OUTPUT

show number 0

show icon

show string "Hello!"

show leds



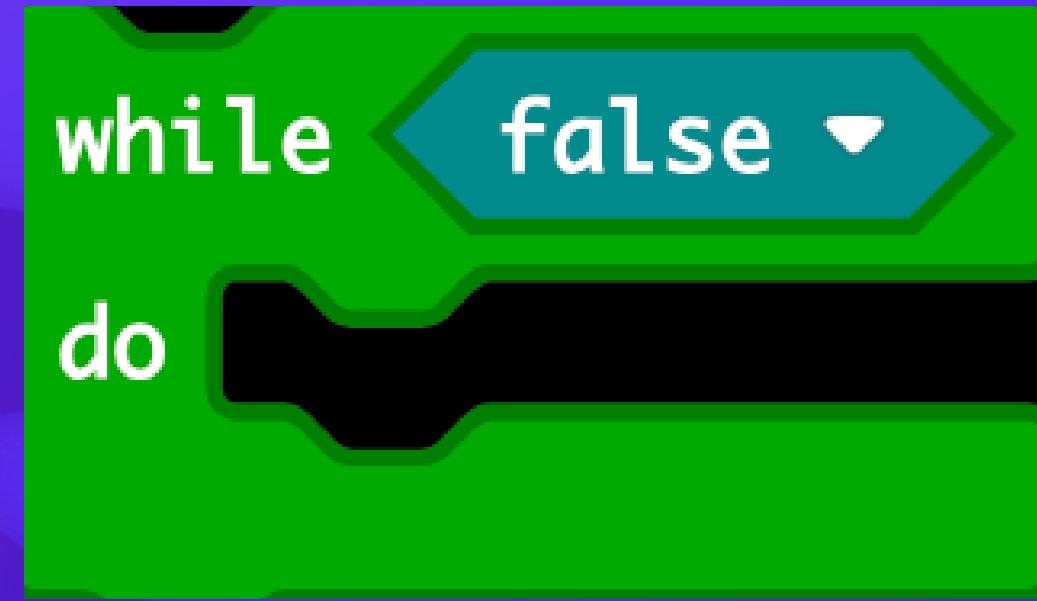
CONDITIONALS

01

02



LOOPS



TIME TO CODE!

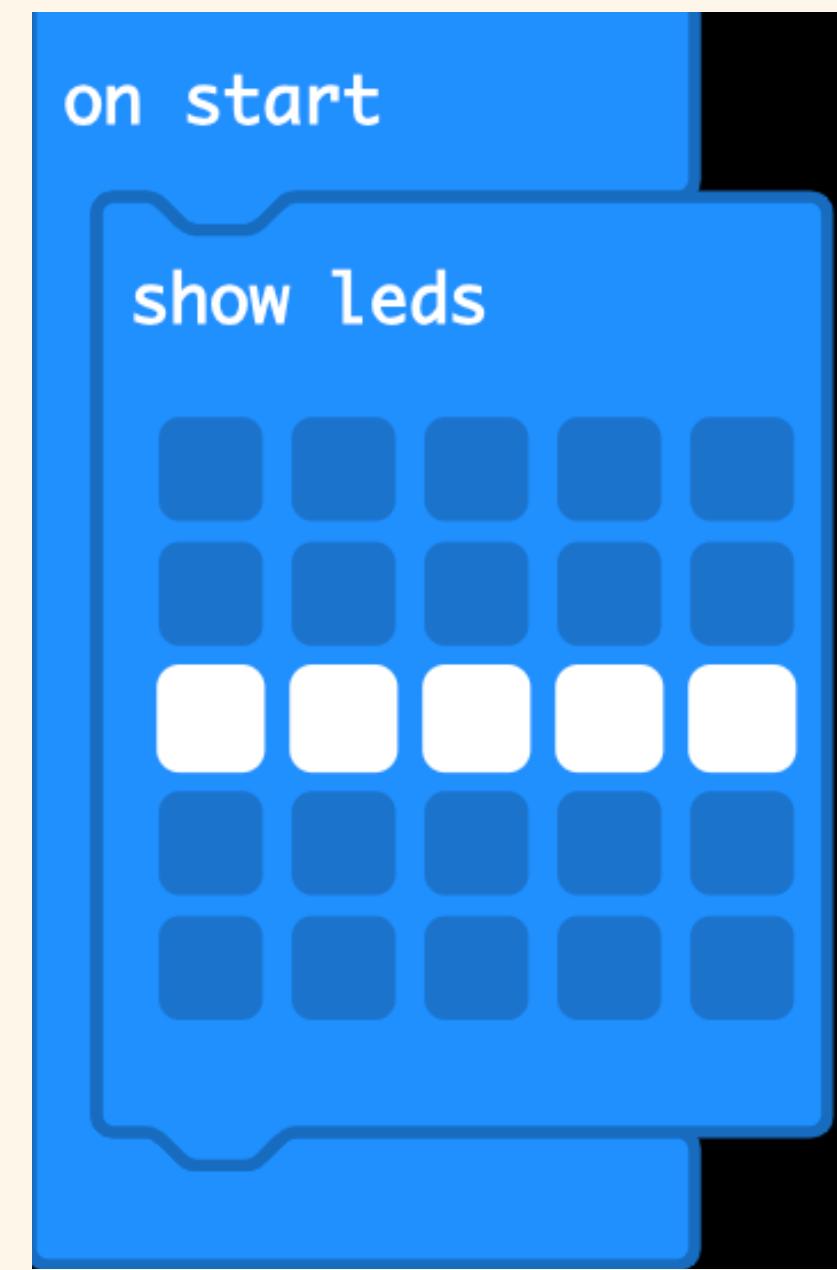
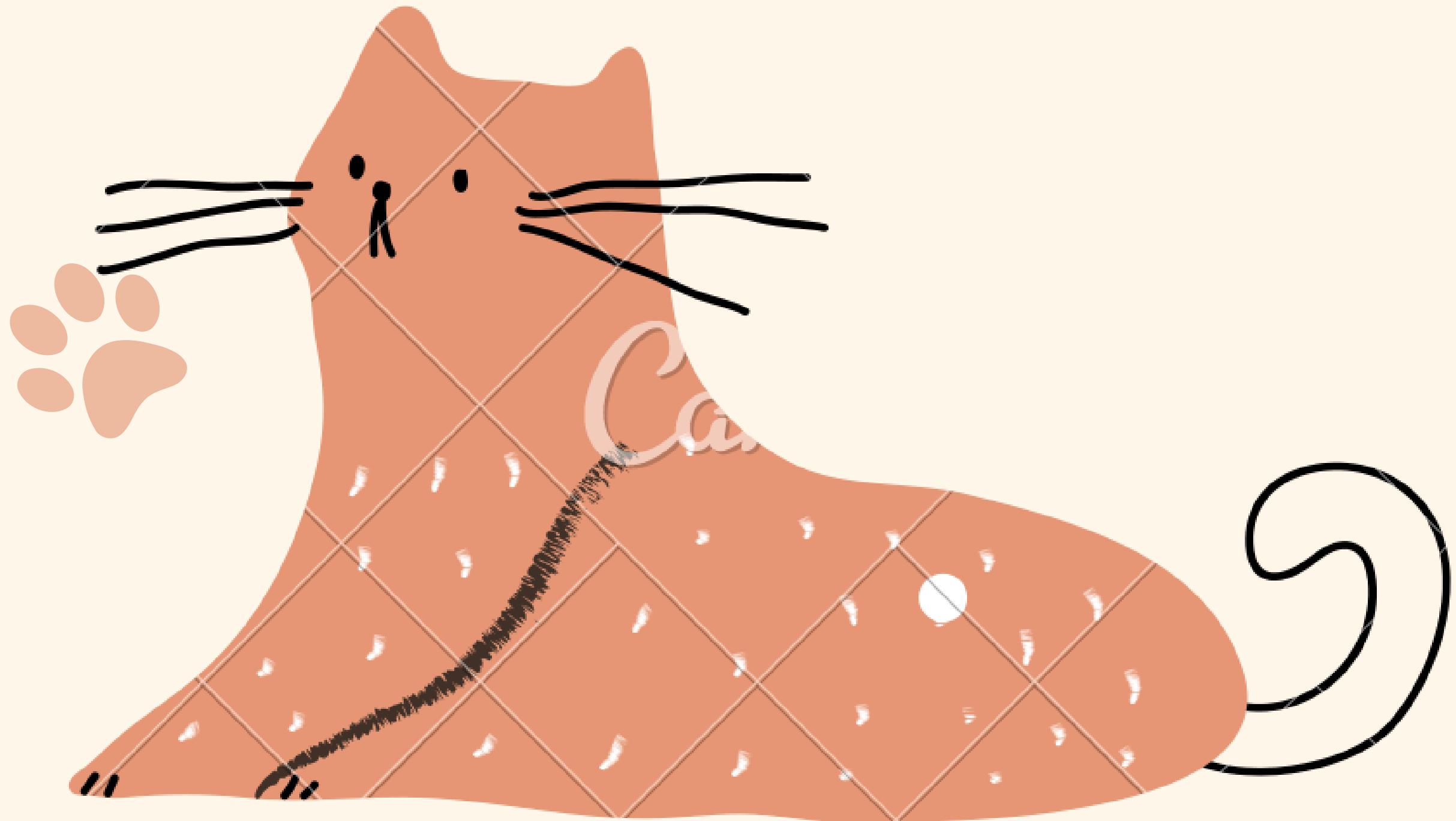




micro:pets

17th January 2024

start - neutral

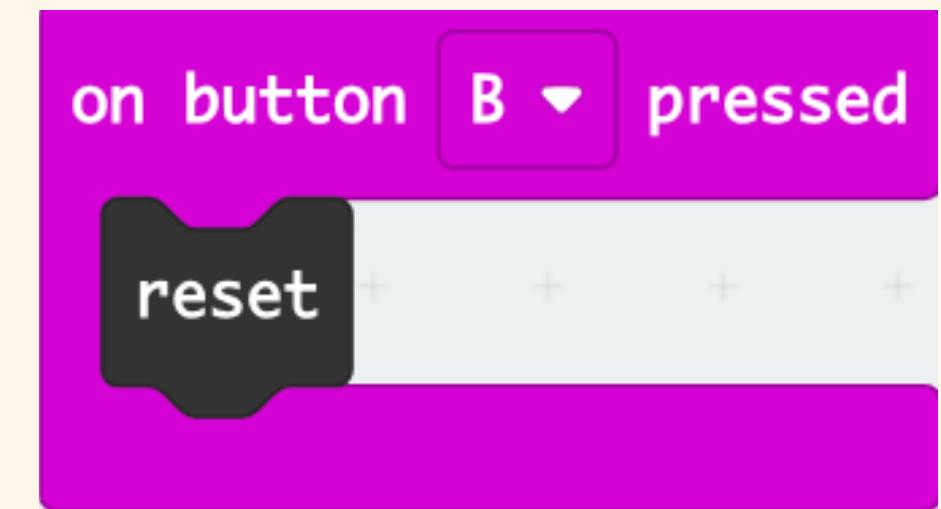


button A → happy

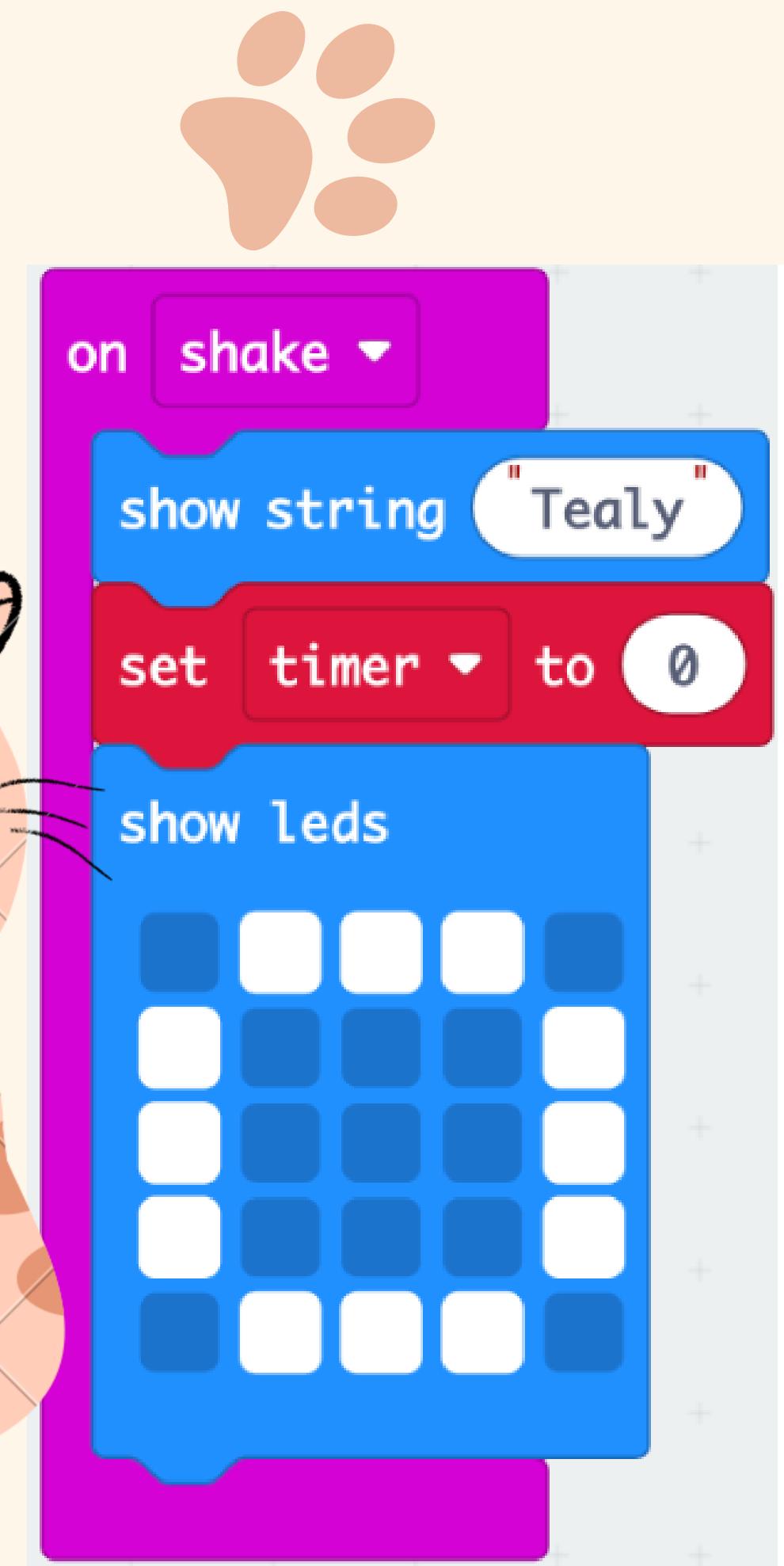




button B → reset

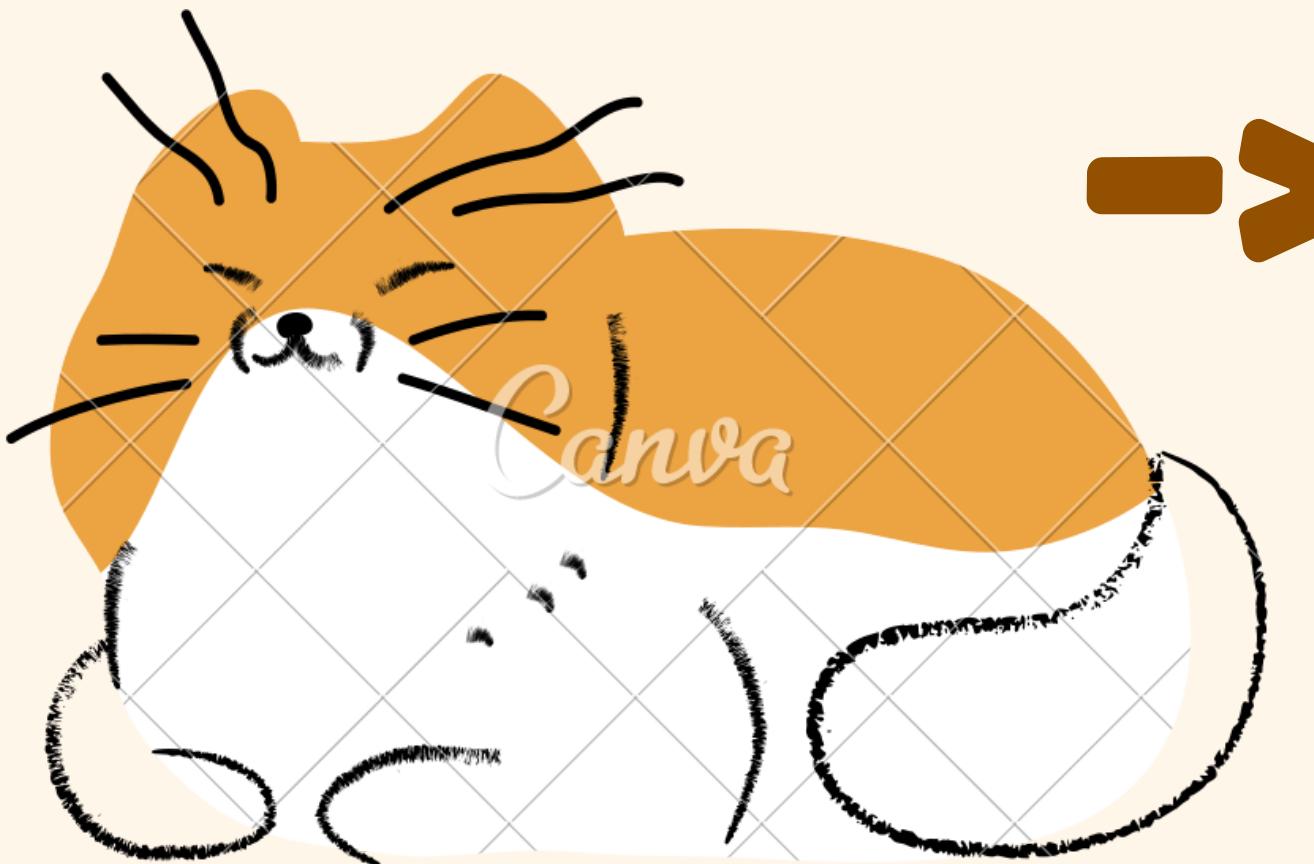


**shake →
show name**



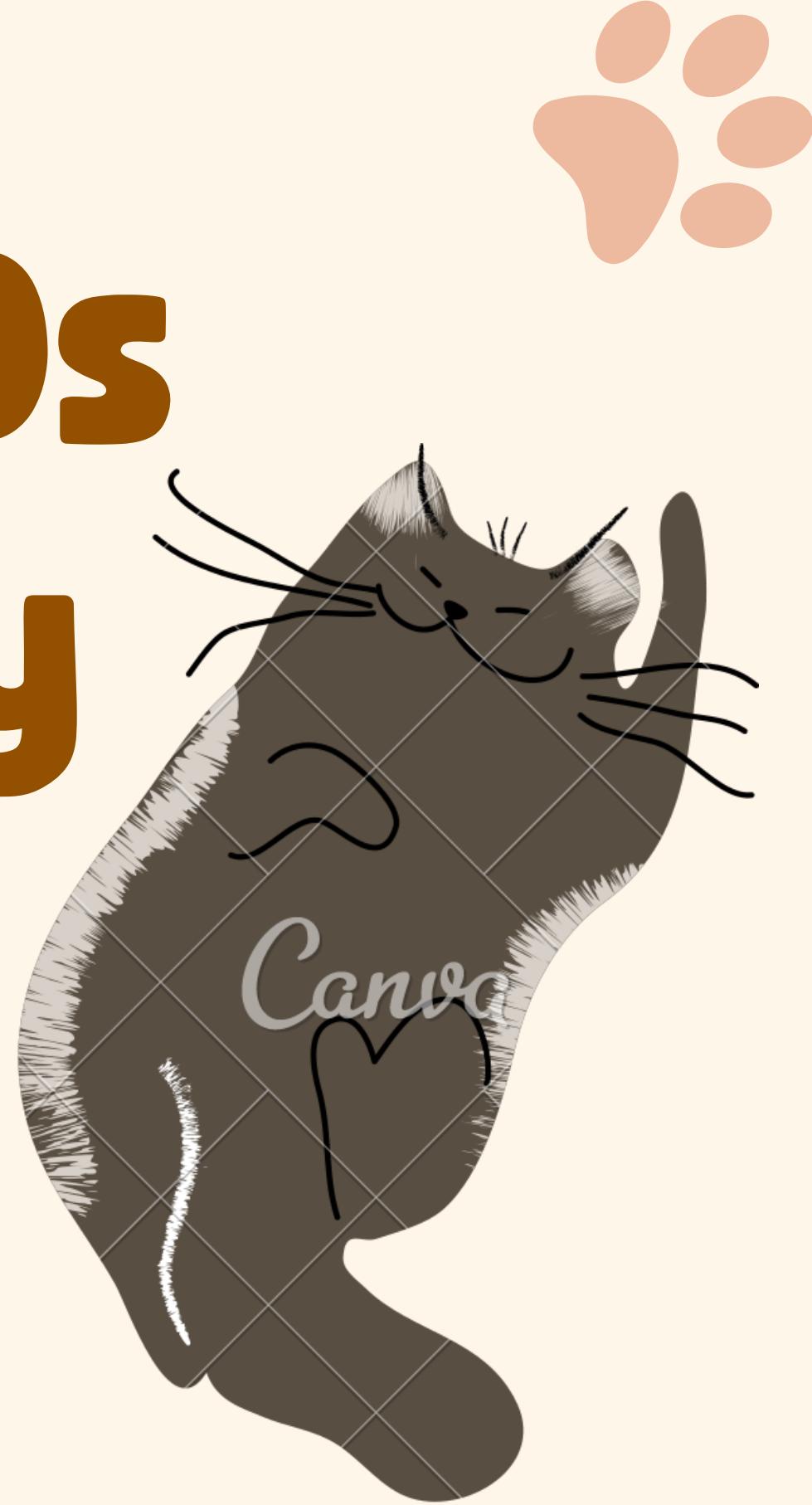
after 15s

→ sad



```
forever
  pause (ms) [1000 v]
  change [timer v] by [1 v]
  if [timer v] = [15 v] then
    show leds [grid v]
  end
  if [timer v] = [30 v] then
    while [true v]
      do
        show leds [grid v]
      end
    end
  end
end
```

after 30s
→ sleepy

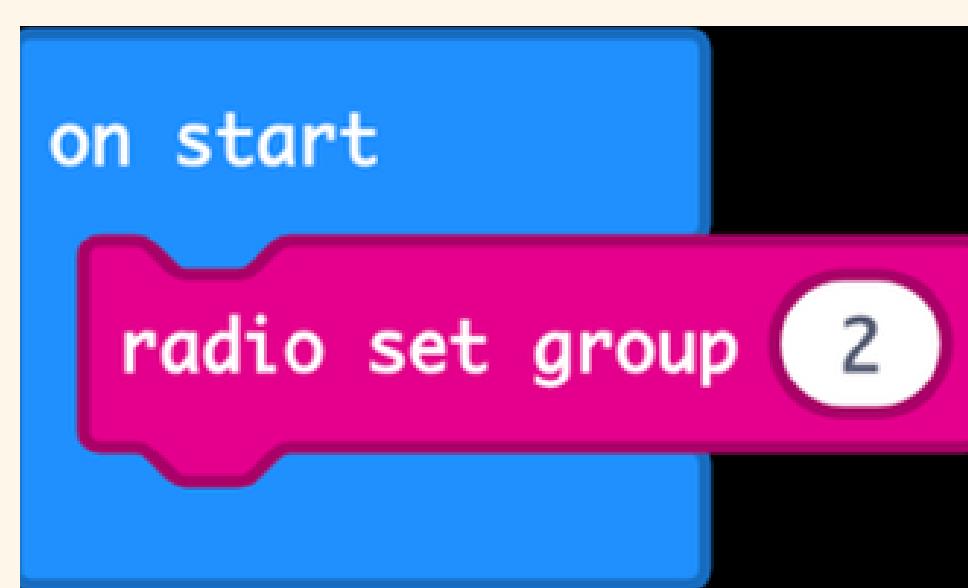


```
forever
  pause (ms) [1000 v]
  change [timer v] by [1 v]
  if [timer v] = [15] then
    show leds [grid v]
  end
  if [timer v] = [30] then
    while [true v]
      do
        show leds [grid v]
      end
    end
  end
end
```

A Scratch script titled "forever". It contains a "pause (ms) [1000 v]" block, a "change [timer v] by [1 v]" block, and an "if [timer v] = [15]" then block. Inside the "if" block is a "show leds [grid v]" block. Below the "forever" loop is another "if [timer v] = [30]" then block. Inside this "if" block is a "while [true v] do" loop, which contains a "show leds [grid v]" block. There are also three orange paw prints on the right side of the script area.

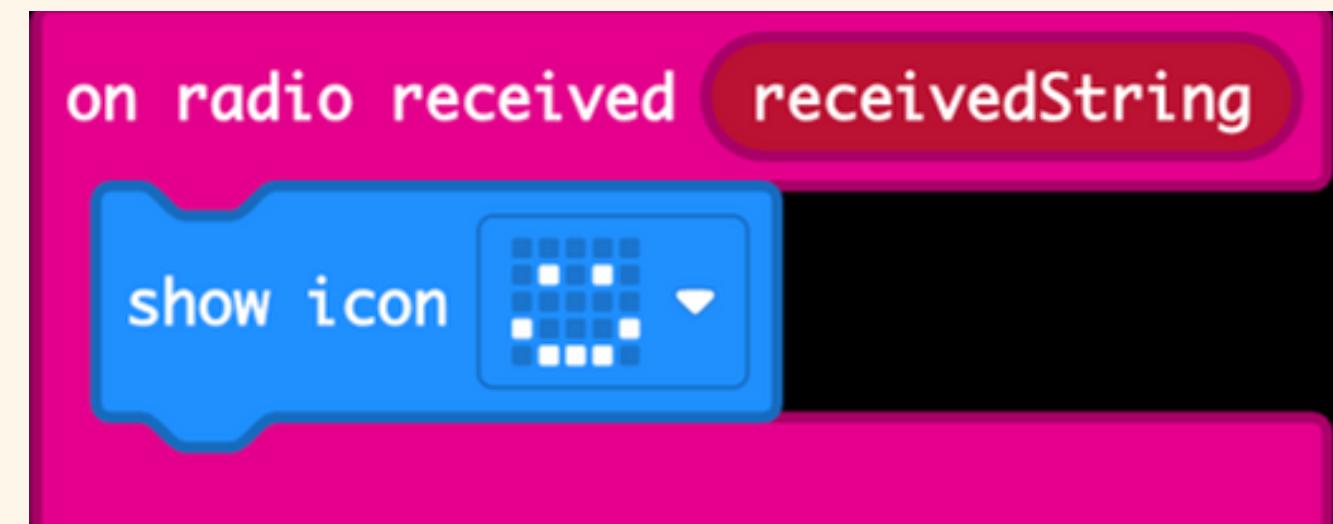


spread a smile :)

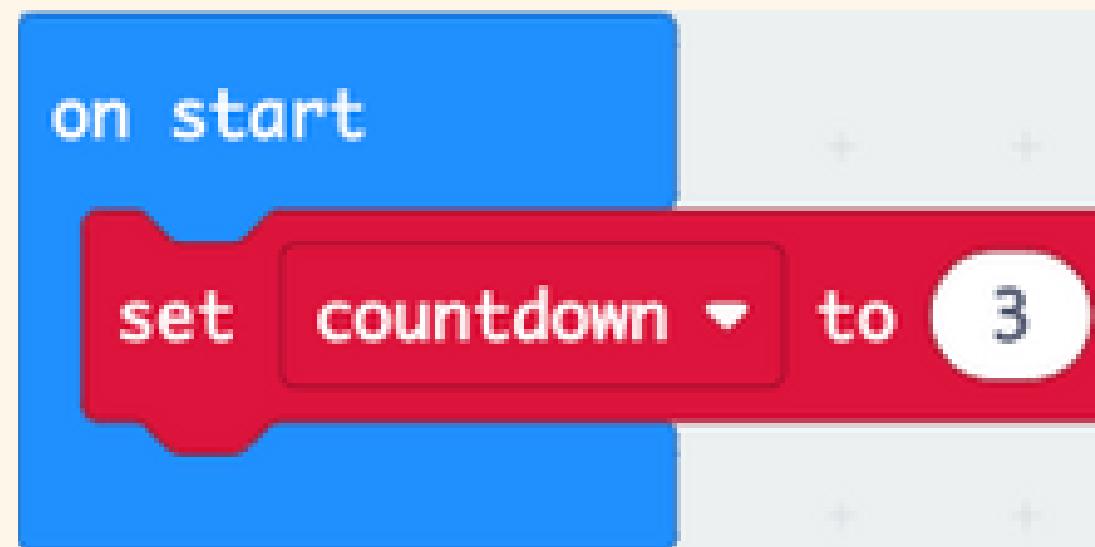
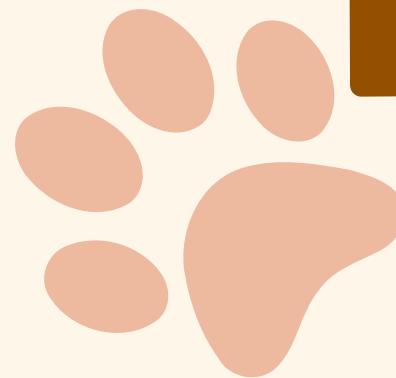


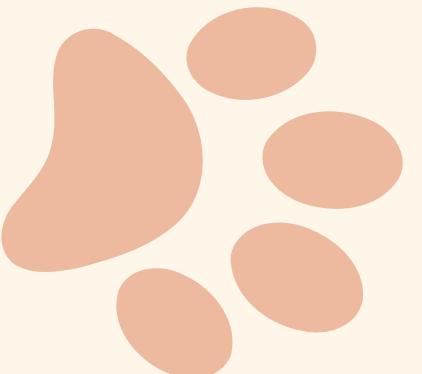
```
on button A pressed
  clear screen
  radio send string "smile"
```





rock paper scissors





```
on shake
repeat (3) times
  do
    show number countdown
    change countdown by (0) - (1)
  end
  set tool to pick random (0) to (2)
  if tool = (0) then
    show icon grid
  else if tool = (1) then
    show icon grid
  else
    show icon grid
  end
  pause (ms) (2000)
reset
```



A cartoon illustration of a ginger cat with a white belly. The cat has a grid pattern on its back and tail, resembling a map or graph paper. The word "Canva" is written in a stylized font across the grid. The cat is looking towards the right. There are several orange paw prints scattered around the cat, some above and some below it.

**see you
next week!**