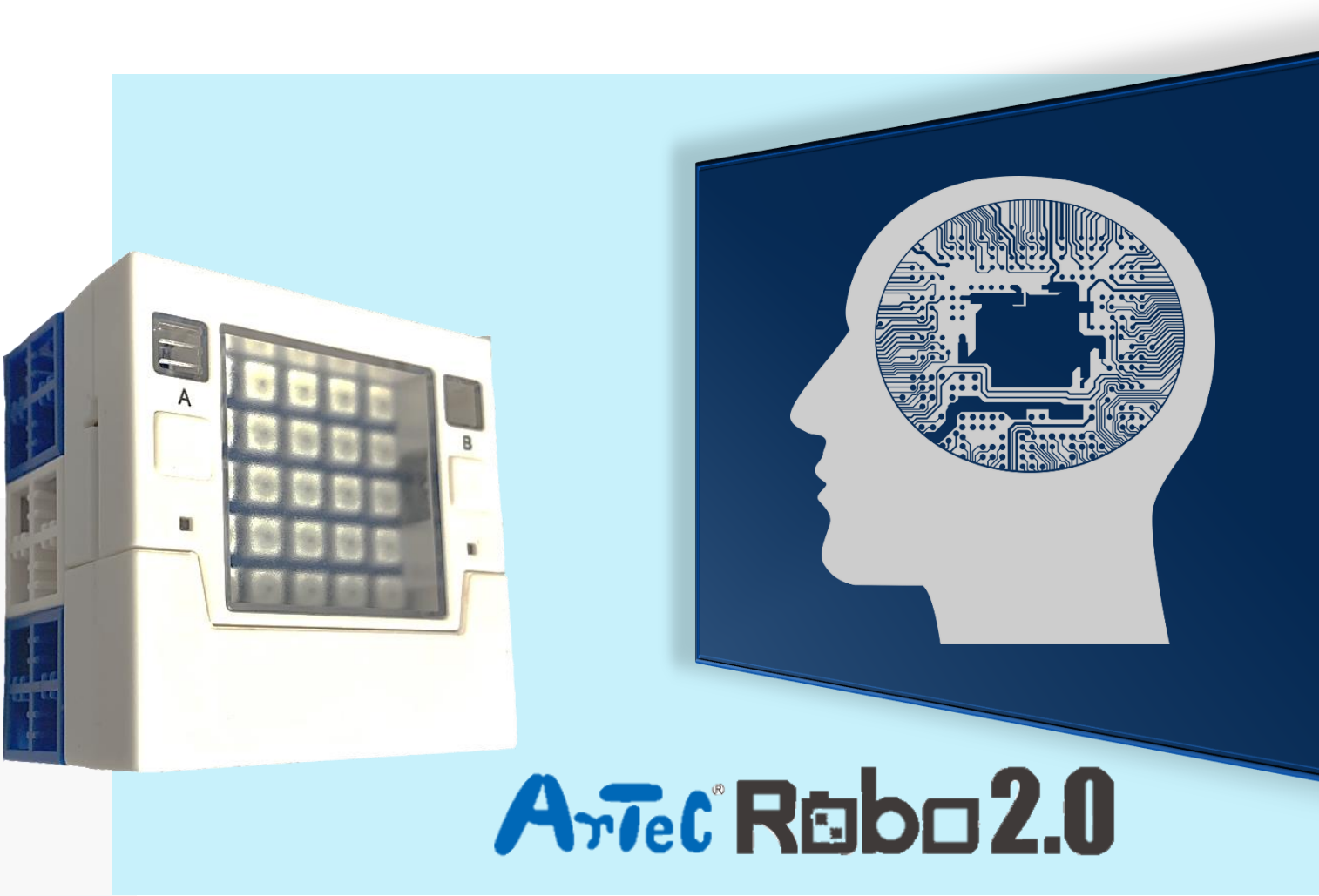
# Chapter 2: Talking to Robots




Name:

Grade:

Class:

# Table of Contents

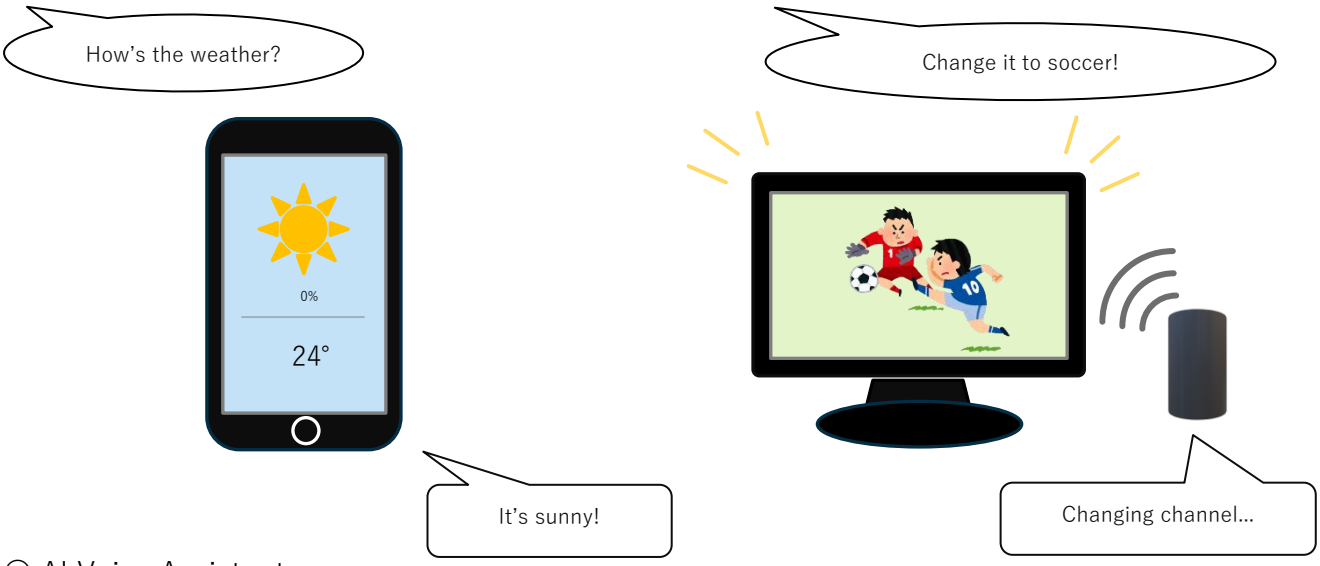
# Lesson 1

## 1 . Smart Lights

### 1 - 1 . AI Voice Assistants

AI voice assistants are a feature or service that can recognize and respond to what you say. You'll find them in smartphones, tablets, smart speakers, and all sorts of other devices. With these assistants, you can get today's whether forecast or the exact directions and time to the nearest trains station just by asking! You can also connect them to your home appliances and control each one with your voice. This includes turning them on or off, setting the temperature of the air conditioner, or even changing the channels on your TV!

The table below shows some of the most popular AI voice assistants. How many have you used before? AI voice assistants are everywhere around us, and it would be hard to imagine life without them!




○ AI Voice Assistants

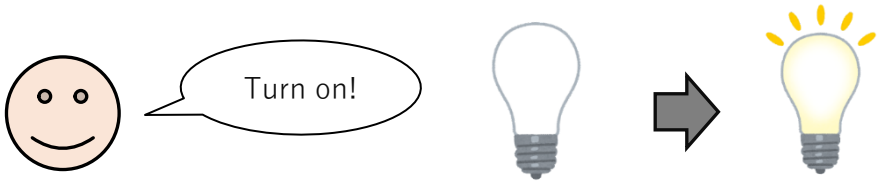
| Name | Siri® | Alexa | Google Assistant™ |
|---|---|---|---|
| Company | Apple® | Amazon | Google |
| Device | iPhone®, iPad®, etc. | Amazon Echo series | Android devices |

## 1 - 2 . What's a Smart Light?

It's time to make our own mini home appliance with an AI voice assistant! In this chapter we'll make a smart light that you can turn on or off just by using your voice.


It will be able to do the following:

1. Turn on when you say **turn on**.




2. Turn off when you say **turn off**.


Now let's make our smart light and add these features to it. The first step is to build it out of blocks!

# 2 . Making a Smart Light

## 2 - 1 . The Blocks

### You'll Need

Basic Cube (white) x 5

Basic Cube (blue) x 4

Core Unit x 1

If you use USB...

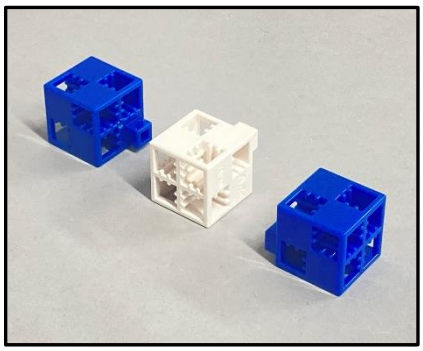
USB Cable x 1

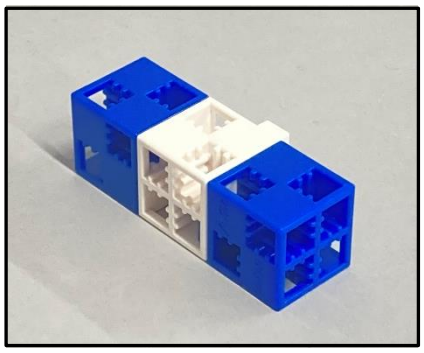
If you use Bluetooth...

Battery Box x 1

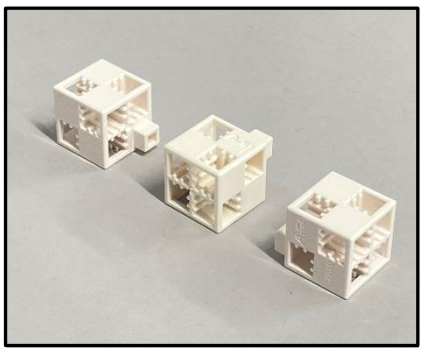

AA/LR6 Battery x 3

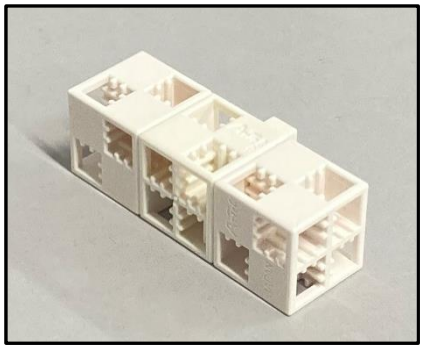
① Connect the blocks shown.



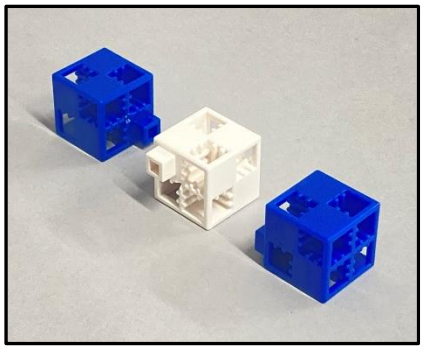


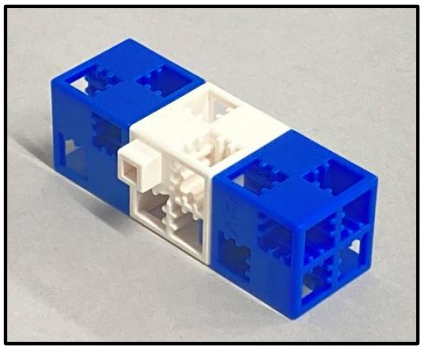
② Connect the blocks shown.



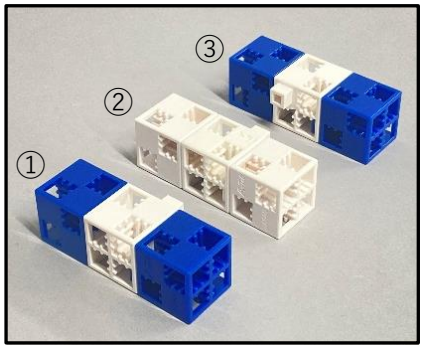


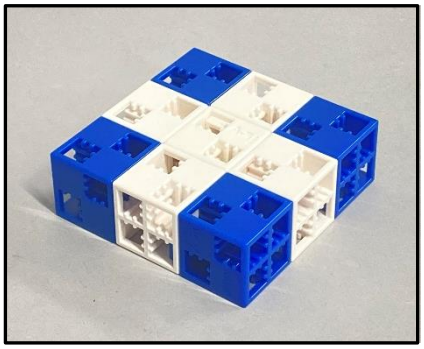
③ Connect the blocks shown.



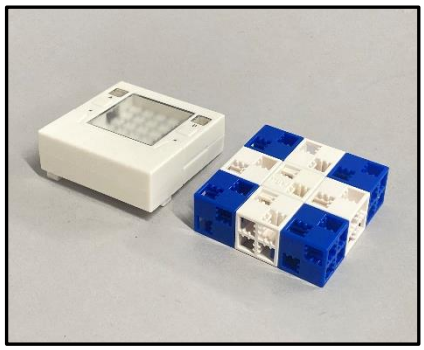


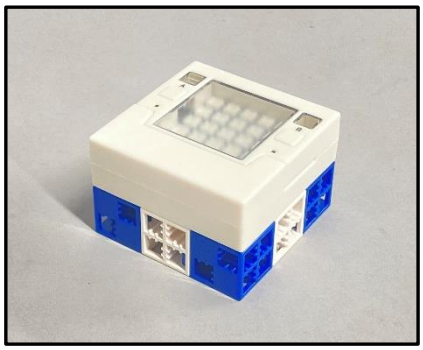
④ Connect ① through ③ as shown.



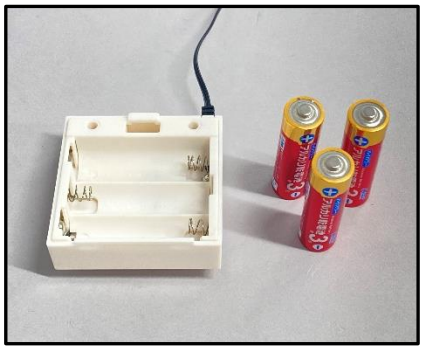



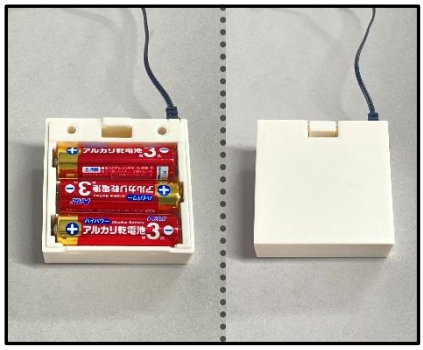
⑤ Add your Core Unit on top of ④. This is the last step if you're using USB!



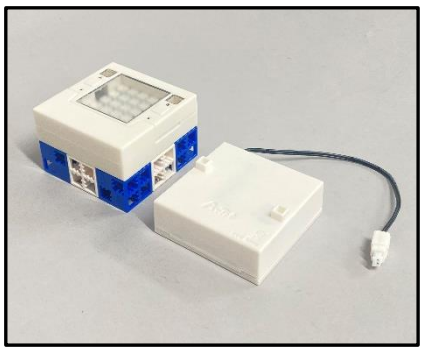


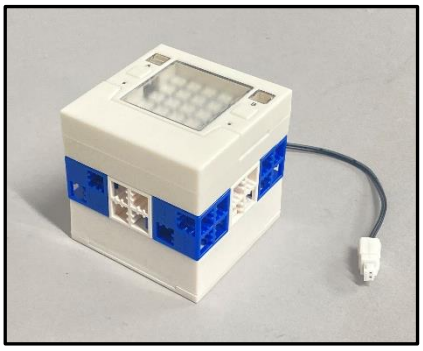
⑥ If you're using Bluetooth, put your batteries into the Battery Box and close it!



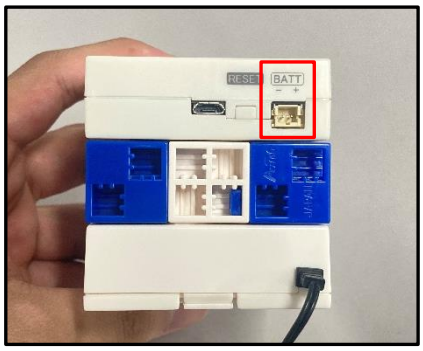


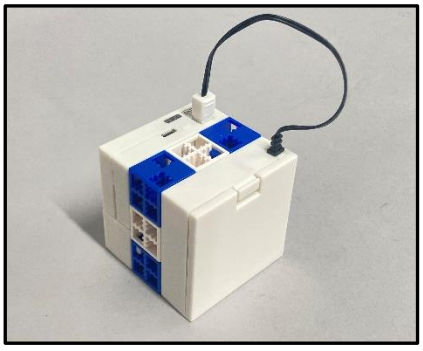
⑦ Add ⑤ on top of ⑥.





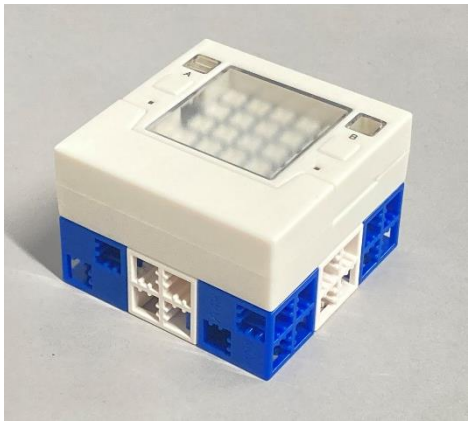
⑧ Plug your Battery Box into the BATT port.
(unplug it when you aren't using the light to keep your batteries from draining!)



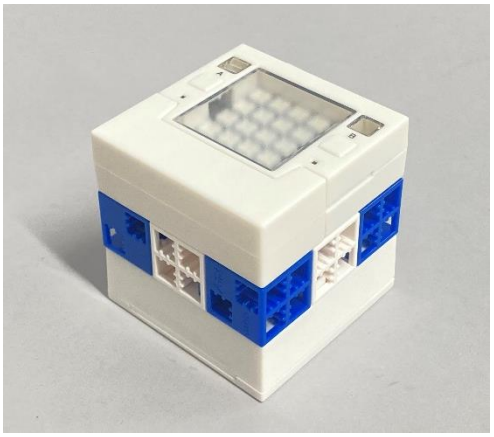



⑨ Finished!

USB



Bluetooth

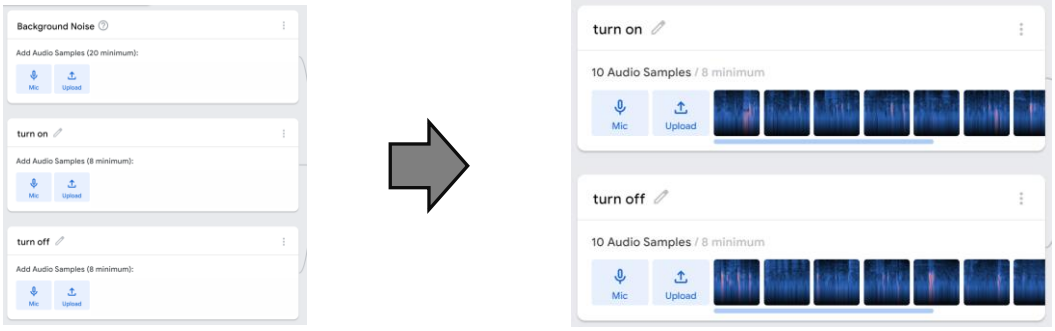


# 3 . Voice Commands

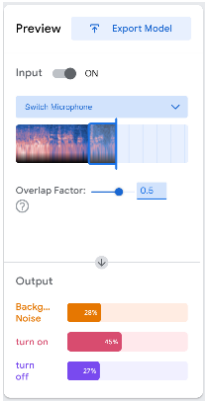
## 3 - 1 . Voice Training with Teachable Machine

Launch Teachable Machine and open a new audio project. If you need to review how to record samples, go back to Chapter 1! Follow the same steps as the last chapter to record your background noise.

① Add the classes **turn on** and **turn off**, then use all three classes to record your background noise and voice commands. Your turn on and turn off commands should have at least 10 samples.




② Now click the **Train Model** button to train your classification model! Once your model is trained, use the Preview window to see if it works.




③ If it does, export the model to get the URL for your project. It might be a good idea to paste the URL into a text file and save it! If your model is having trouble recognizing your voice, try adding more samples until it works correctly!

### ▶ Saving the Project

Let's save this project with the name project21

so we can use it in the next class!

## 3 - 2 . Connecting to TM2Scratch

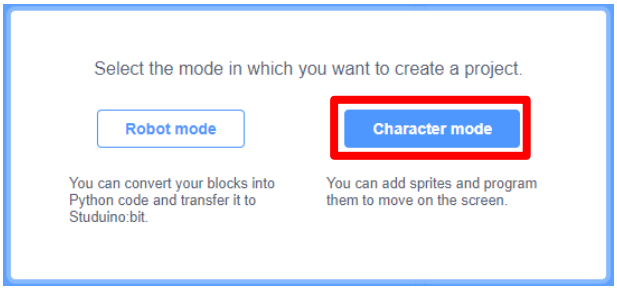
① Open Studuino:bit Online in Chrome version 89 or later and click Character Mode.

★ Studuino:bit Online won't work on Safari, Edge, Internet Explorer, Firefox, or other browsers.

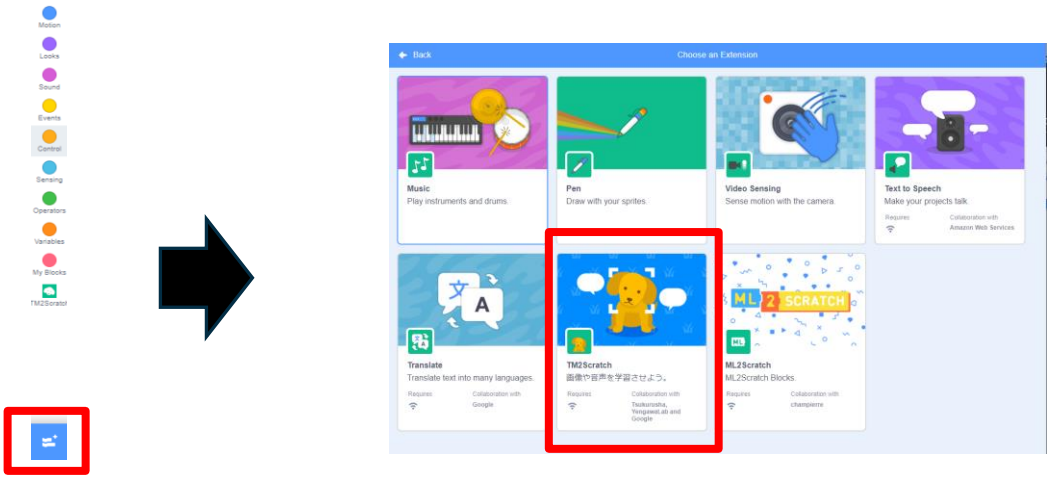
★ This version of the app isn't supported on tablets.

If you're using USB... https://app.artec-kk.co.jp/studuinobit/usb/

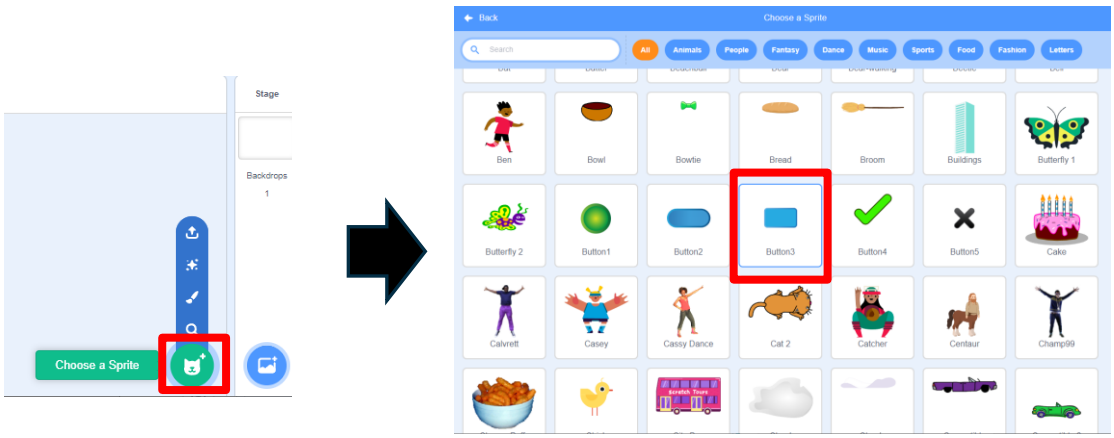
If you're using Bluetooth... https://app.artec-kk.co.jp/studuinobit/bt/



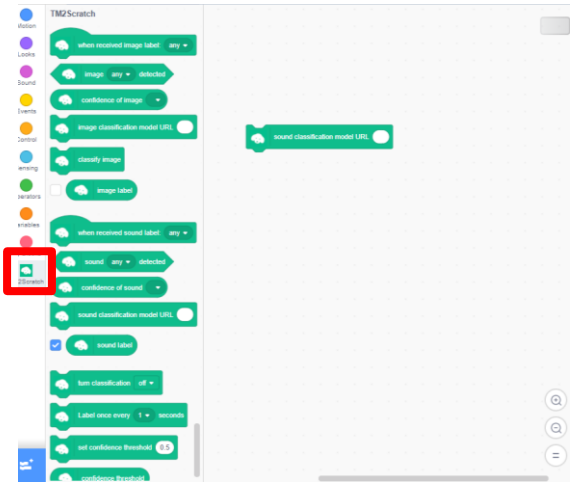

② Click the at the bottom left, then click TM2Scratch.



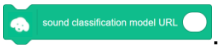

③ Click the **Choose a Sprite** button, then choose **Button3**.



④ Click the TM2Scratch icon.

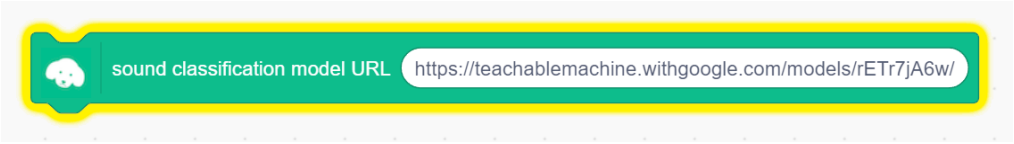



⑤ Drag a **sound classification model URL** middle of the screen and delete any URL inside of it.

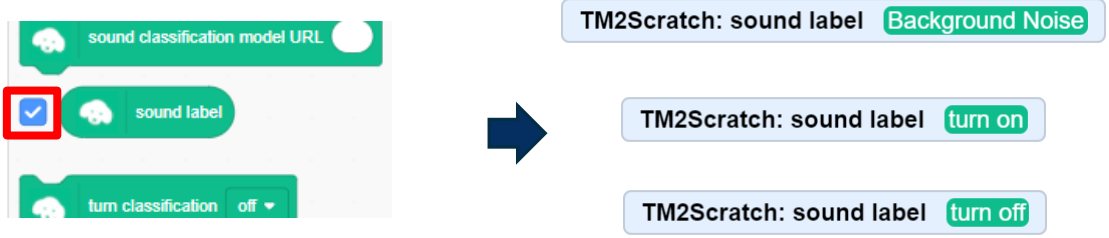

⑥ Paste the URL from 3-1 into 

.

⑦ The block will be highlighted in yellow and return to normal once your project has loaded!




⑧ Check the box next to the **sound label** block to see the labels for your project. Now try staying silent, then speak into your smart light to see how each label reacts!

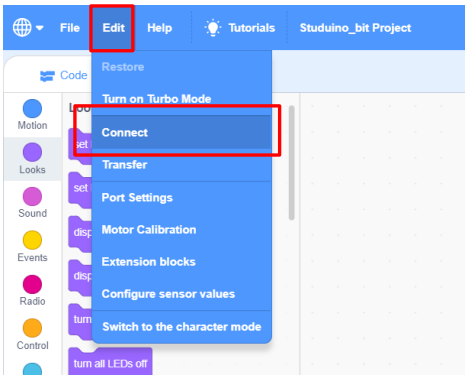


# 4 . Connecting the Core Unit

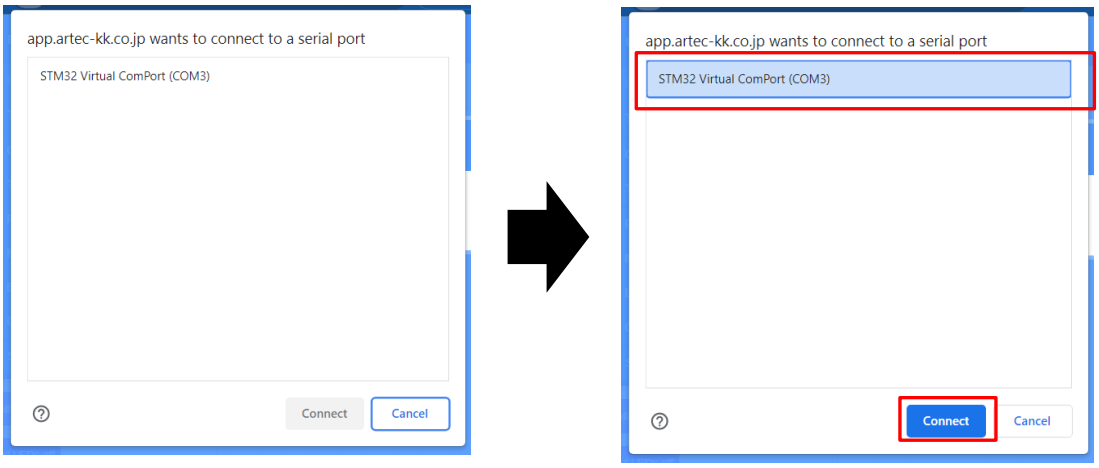
## 4 - 1 . USB

① Connect the USB cable to the PC and the Core Unit.

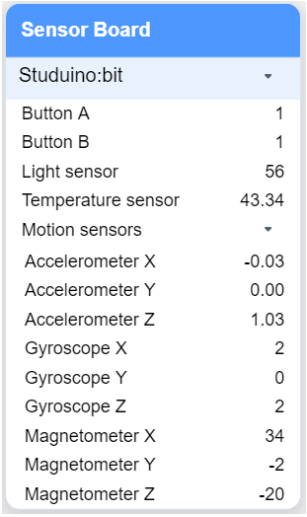
② Next, click the **Edit** menu and choose **Connect**.



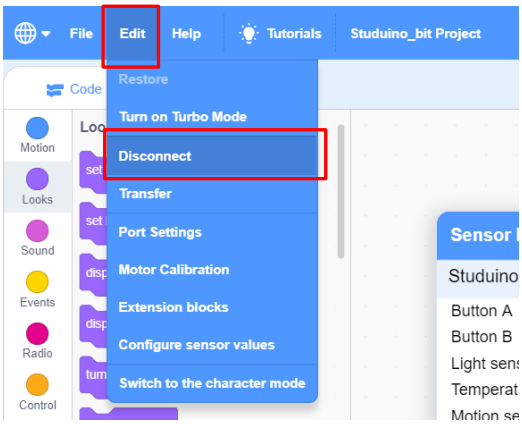

③ Choose **STM32**, then click the **Connect** button.



④ The Sensor Board will appear. You can use it to check how your sensor values change! The Sensor Board will stop updating if you unplug the USB cable while connected.



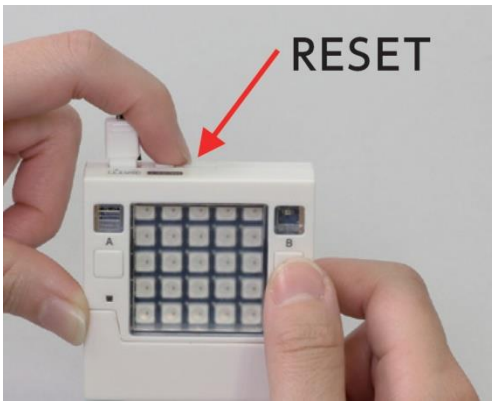

⑤ If this happens, click the **Edit** menu and choose **Disconnect**. Always remember to disconnect if your USB cable becomes unplugged. This will help prevent errors!



## 4 - 2 . Bluetooth

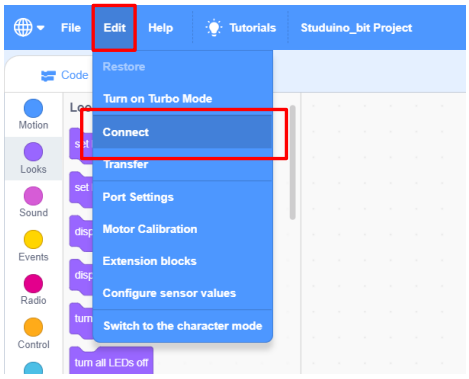
① Plug the Core Unit's Battery Box cable into the Battery Box.

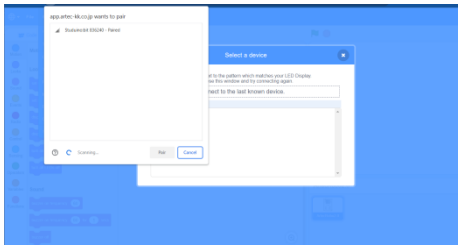

② Hold down the B button and press the Reset button.



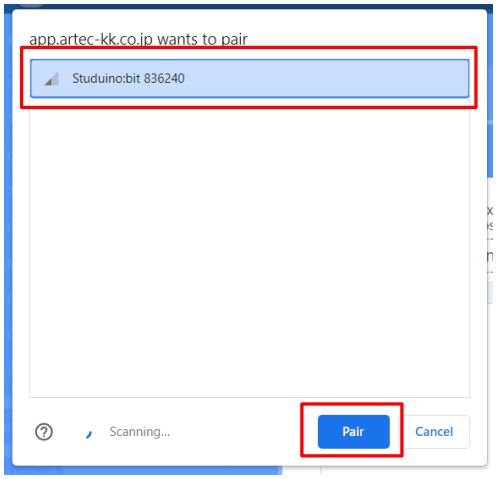

③ Keep holding B until you see the pattern on the LED display.

④ Next, click the **Edit** menu and choose **Connect**.

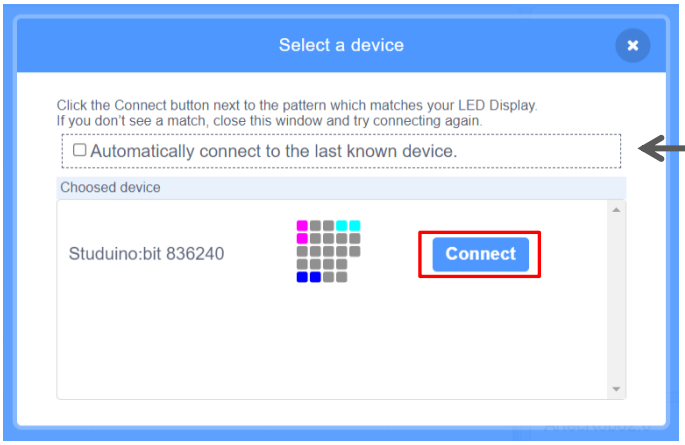


⑤ You'll see a list of Core Units in your area along with their patterns.
Choose **Studuino:bit XXXX** (XXXX should be four numbers) and click **Pair**.

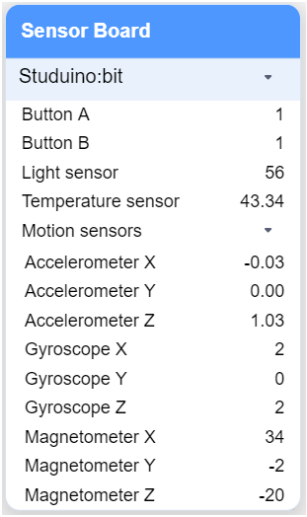



⑥ Make sure that the pattern matches the one on your Core Unit, then click **Connect**.
If the pattern doesn't match, click the x to close the window, then repeat step ⑤ and choose another device in step ⑥!

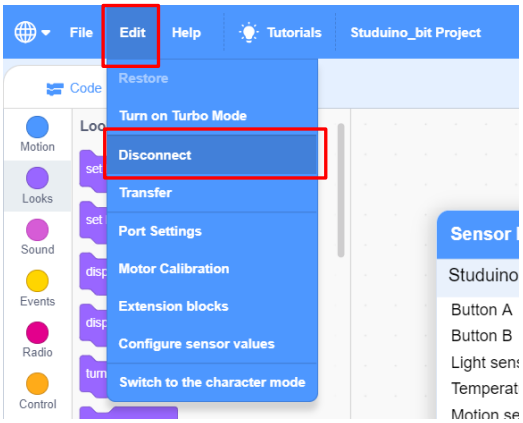



Check here to skip step ⑥ next time you connect!

⑦ The Sensor Board will appear. You can use it to check how your sensor values change!



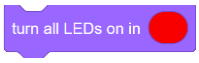

⑧ If this happens, click the **Edit** menu and choose **Disconnect**.



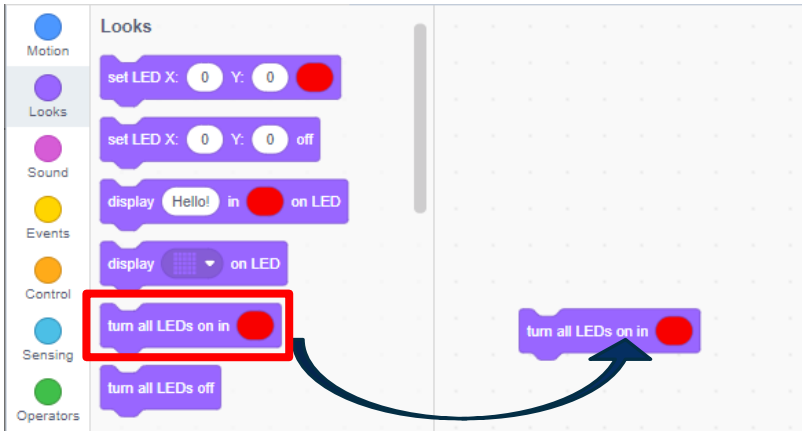

★ Click Edit and choose Disconnect if your connection is interrupted!

## 4 - 3 . Turning On

Now that we're connected, let's try turning on the smart light!

① Drag a 

block to the middle of the screen.




② Click the color in the red square to choose a color for your smart light. Setting every slider to 31 will cause the smart light to light up in white!


③ Click the 

block to turn on your light.

④ Now click the 

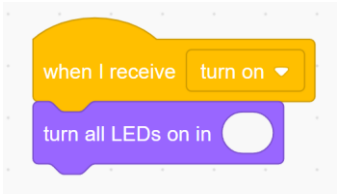
block to turn the light off!

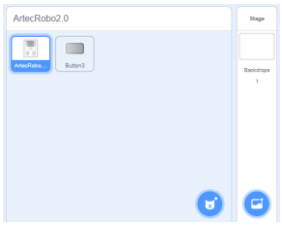
# 5 . Programming ①

## 5 - 1 . Making the Program

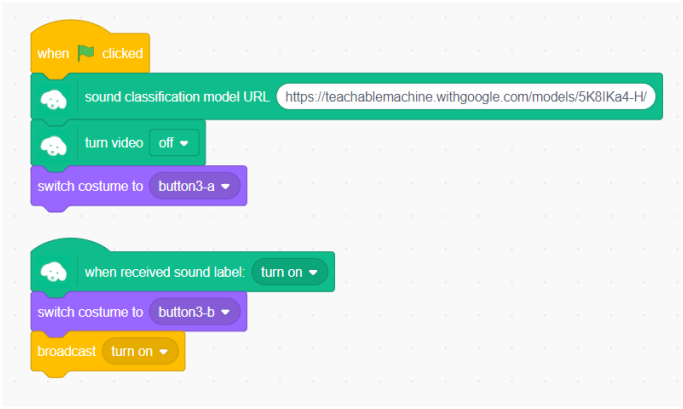
Make the sets of blocks you see below. You'll have to program your Core Unit and Button3 sprite separately. Click the  icon for each one to change screens and program them. The program you make here will turn on the light when you say **turn on**!

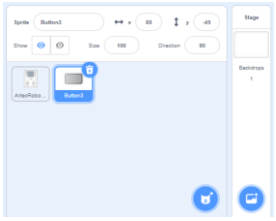
Core Unit

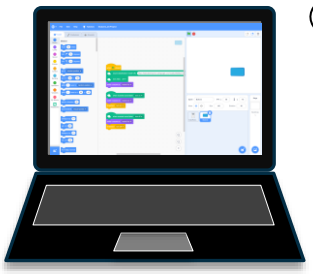


Button3



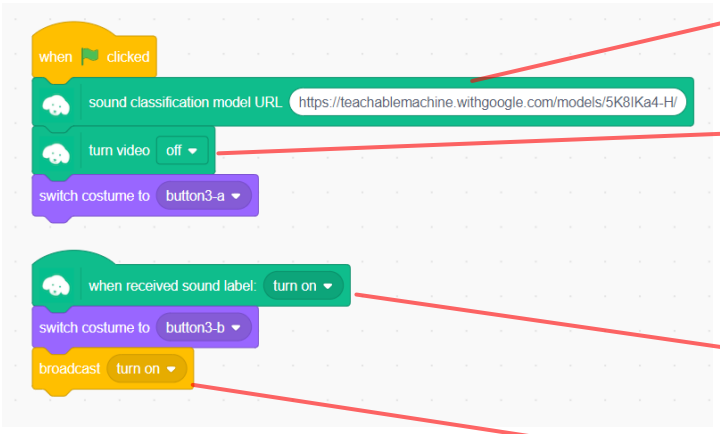






But why do we need to program each one separately? This is because the Core Unit can't recognize your voice all by itself! Voice recognition is handled by the sprite on the computer and the result is sent to your Core Unit. When the Core Unit receives the command from the computer, its program turns on the smart light!

## 5 - 2 . How the Program Works

### Button3



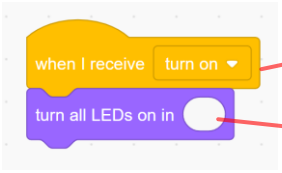

○ This block loads the sound classification model.

○ If you're using image recognition you'll see the image on the Stage. Since we aren't using image recognition, we use this block to turn the video off!

○ Blocks connected to this one run when the program recognizes the **turn on** command.

○ This block sends the message **turn on** to your Core Unit!

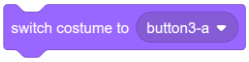
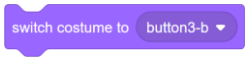
### Programming the Core Unit




○ Blocks connected to this one run when they receive the message.

○ This block turns on the smart light!

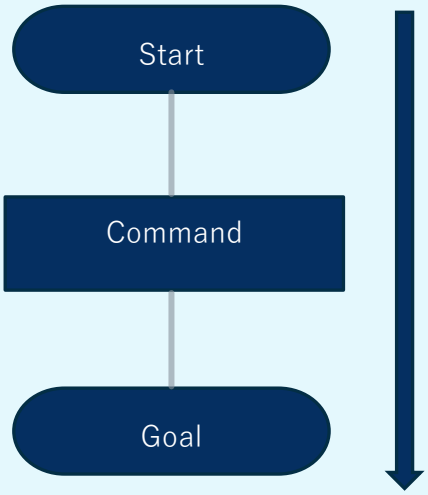
**The Costume Blocks**

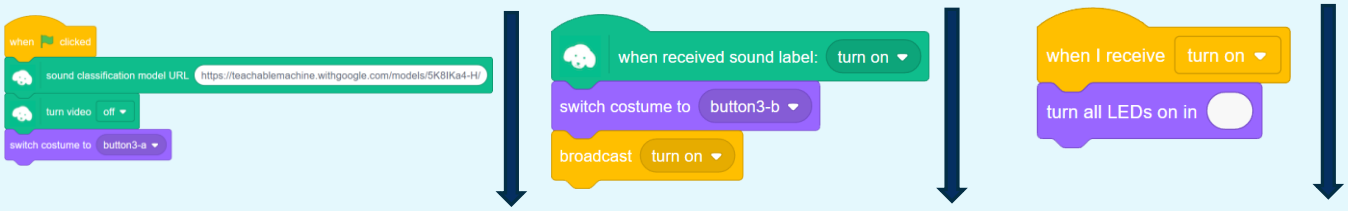
| | When the light is off... | When the light is on... |
|---|---|---|
| Light |  |  |
| Block |   |   |
| Sprite |  |  |

We're going to use costumes to tell us whether the smart light is on or off. We can do this by setting the costume to Button3-a when the light is off and Button3-b when the light is on!

## Sequential Processing

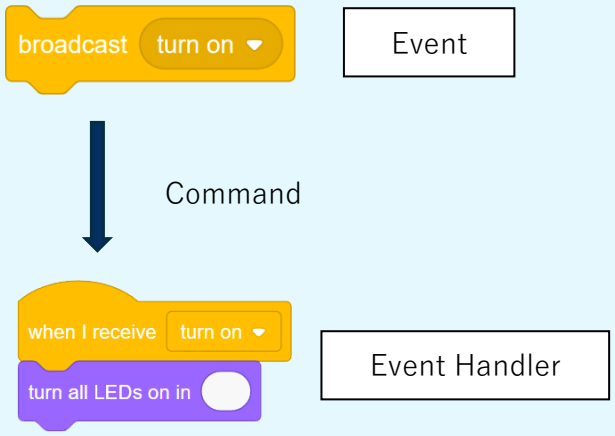
Sequential processing means running a series of commands in order. You can think of this like putting the commands of your program into a single-file line and letting them run one-by-one! This means that the sets of blocks in your program run one at a time in order!




## Event Handlers

Event handlers are used to run certain commands when an event happens. As we learned on page 16, your Core Unit turns on the smart light when it receives a command from the computer. You can use **broadcast** blocks on the computer to send commands as messages to the Core Unit! In this case, the **broadcast turn on** block is the **event**, and the **when I receive turn on** block is the **event handler**!

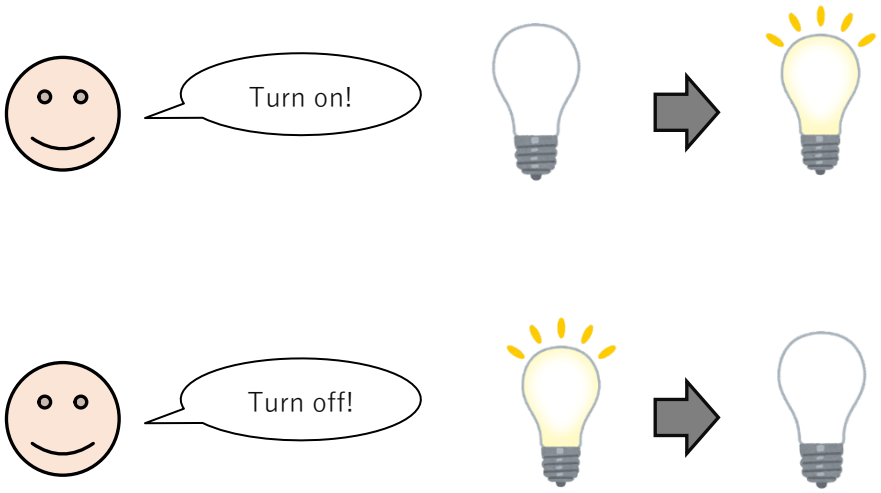



## Saving Your Program

Go ahead and save this program as **ai21**!

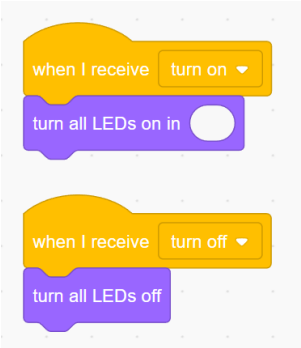
## 5 - 3 . The Challenge

Now let's try programming the smart light to turn off when you say **turn off**!

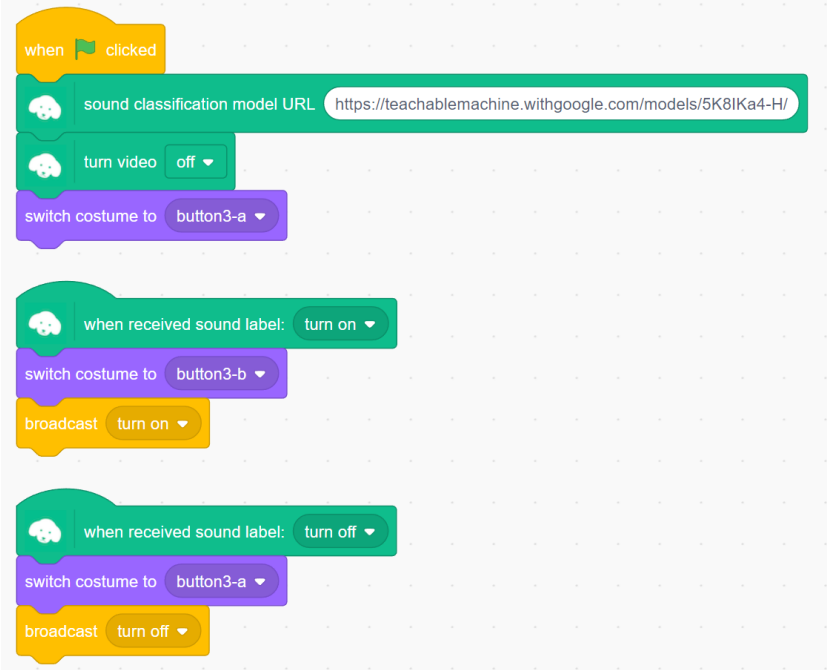


## Example Program

Core Unit




Button3




## Saving Your Program

Go ahead and save this program as **ai22**!

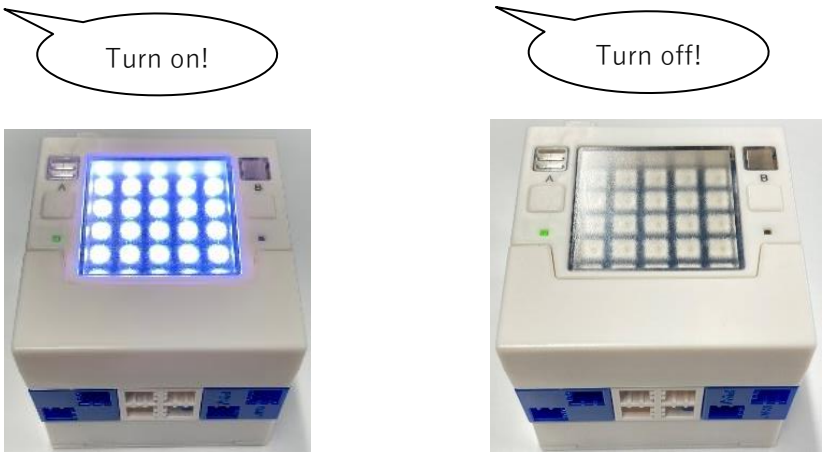
In Lesson 2 we're going to take on the challenge of adding a secret phrase to our smart light!

# Lesson 2

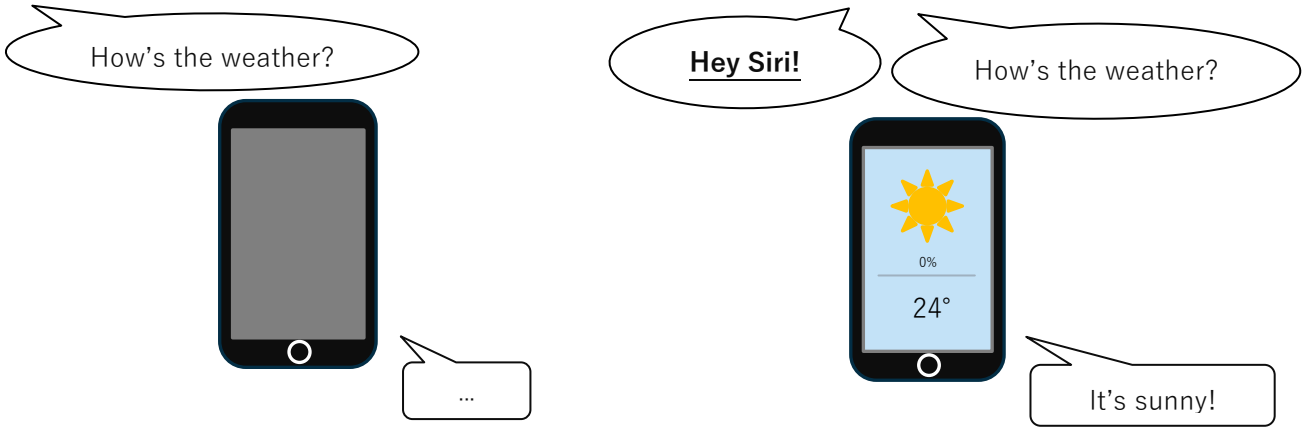
## 6 . Programming ②

### 6 - 1 . Upgrading the Smart Light

The smart light you made in the last lesson is a simple one that we can turn on or off using voice commands.



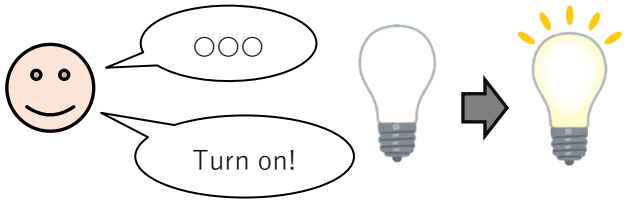

Real-world AI voice assistants, however, won't react if you just talk to them. You have to use a phrase like, "Hey Siri®" or, "Alexa" to get them to respond!

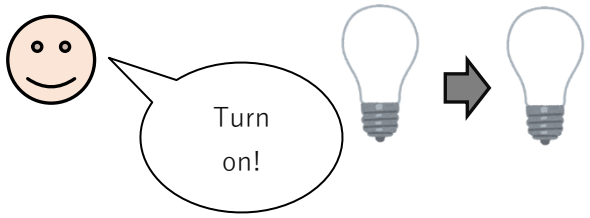



In this lesson we're going to add a phrase that you have to use before turning your smart light on or off. This can be any phrase you want, so try picking a unique one!

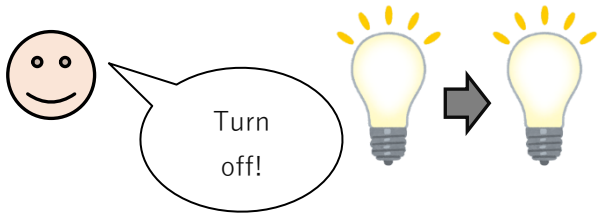
○ The Features

1. You have to say the phrase, then **turn on** to turn on the light.




2. You have to say the phrase, then **turn off** to turn off the light.

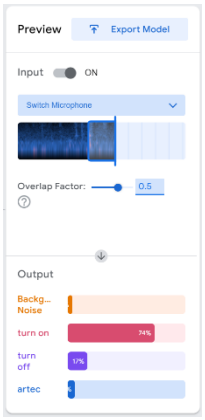



## 6 - 2 . Voice Training with Teachable Machine

Launch Teachable Machine and open project21 from the previous chapter. Now let's add a new label and use it to record your phrase! This textbook uses the phrase **artec**.

① Add the label for your phrase to your **turn on** and **turn off** labels, then record at least 10 samples of the phrase!


② Now click the **Train Model** button to train your classification model! Once your model is trained, use the Preview window to see if it works.




③ If it does, export the model to get the URL for your project. It might be a good idea to paste the URL into a text file and save it! If your model is having trouble recognizing your voice, try adding more samples until it works correctly!

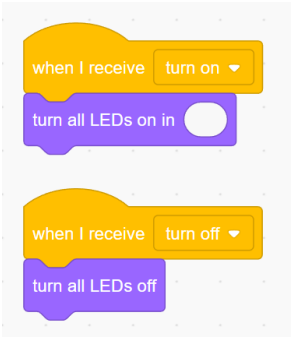
### ▶ Saving the Project

Let's save this project with the name **project22**!

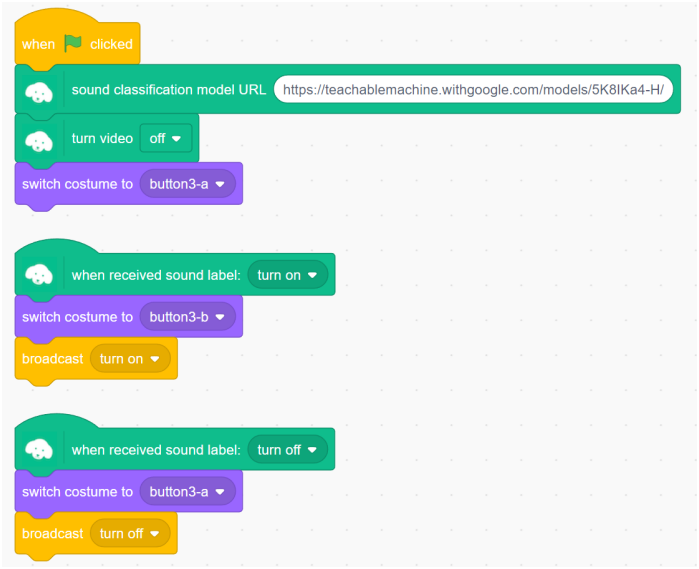
## 6 - 3 . Programming

① Open your ai22 program from the last class. Paste the URL for your new sound classification model into the block. Now let's modify our program to get the smart light to recognize the secret phrase!

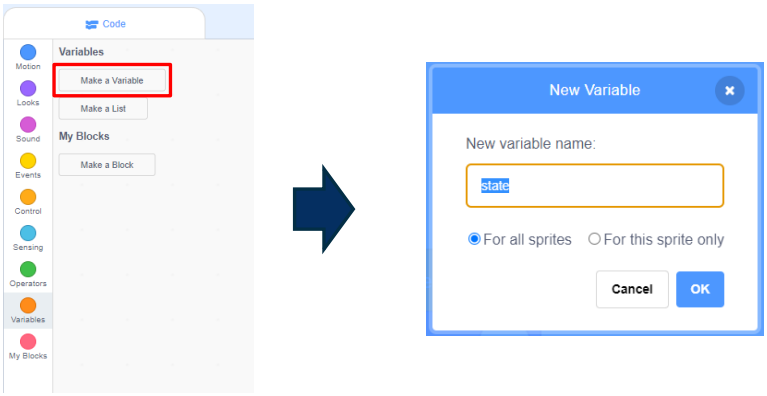
Core Unit




Button3



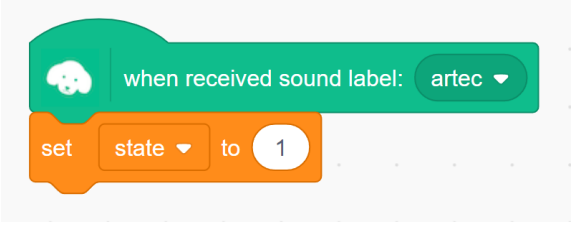

② Make a new variable called **state**. We're going to use the state variable to control the state of our smart light! Click **Make a Variable** and type **state** into the New Variable dialog box.




You'll see the following blocks appear when you make a new variable. A box showing the value of the variable will also appear on the Stage! All variables start with a value of **0**.

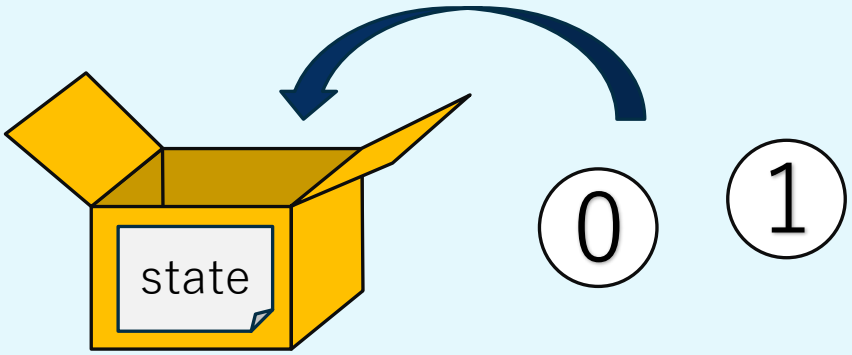
Make the set of blocks you see below.

Button3



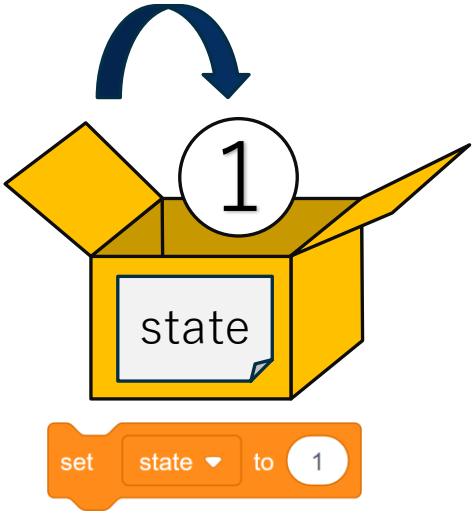

**Variables**

Think of variables as a kind of box that lets you store numbers. Give that box a name and you can pull those numbers out and check them pretty much anywhere! Naming the box **state** and placing a **0** or **1** inside of it lets us check the state of the smart light just by looking inside of the box. Remember to pick a good name for any variable you make!

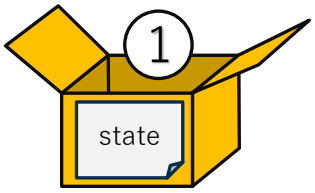
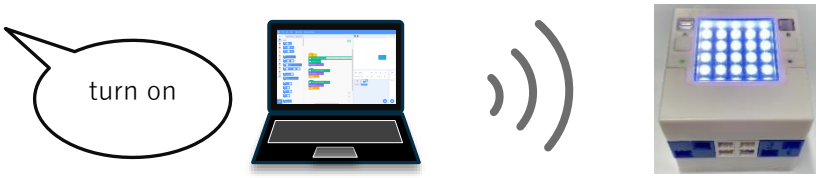
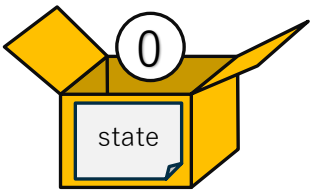



Hear the phrase...


And put 1 in **state**!

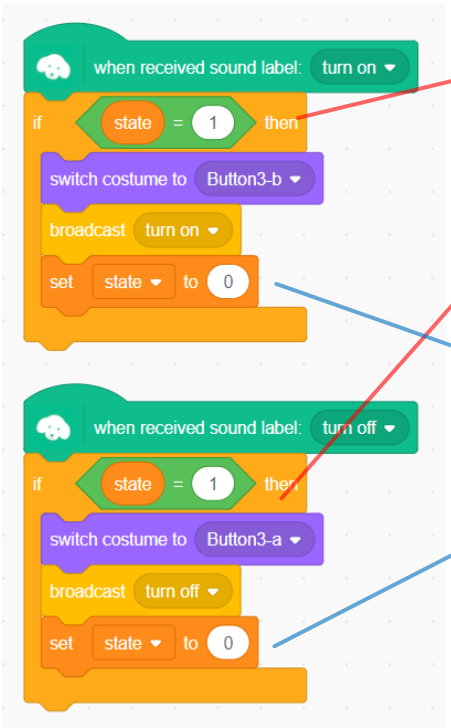


③ Now let's change our program to send the messages **turn on** and **turn off** only when **state** is **1**. That means that the light has already heard the phrase!

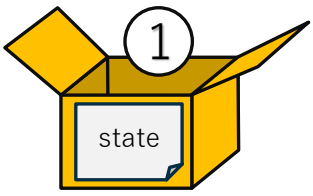
| When state is... | The smart light... |
|---|---|
| 1   | Takes commands to turn on or off after it hears the phrase   |
| 0   | Won't take commands it if it doesn't hear the phrase   |

Make the sets of blocks you see below:

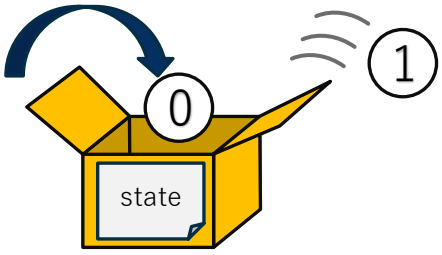
Button3

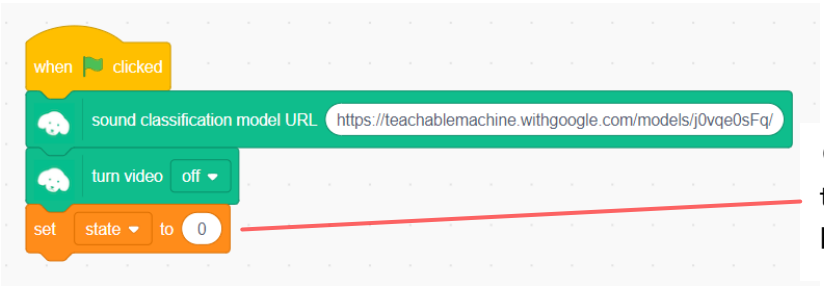



○ These blocks run when the smart light hears the phrase.




○ This resets the smart light after it receives a **turn on** or **turn off** command.

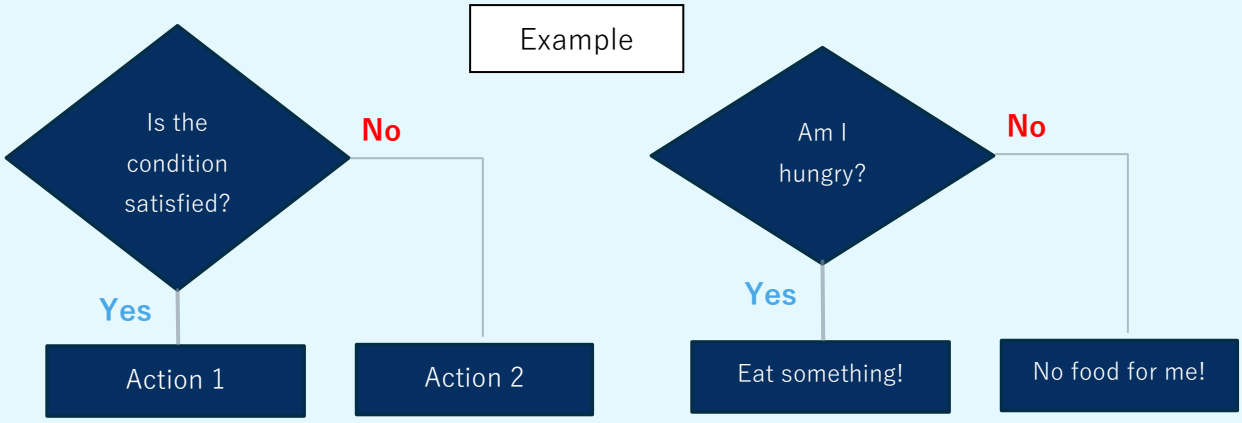


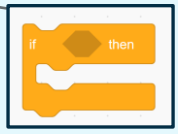

○ This resets state to 0. That's the state of the smart light before it hears the phrase!

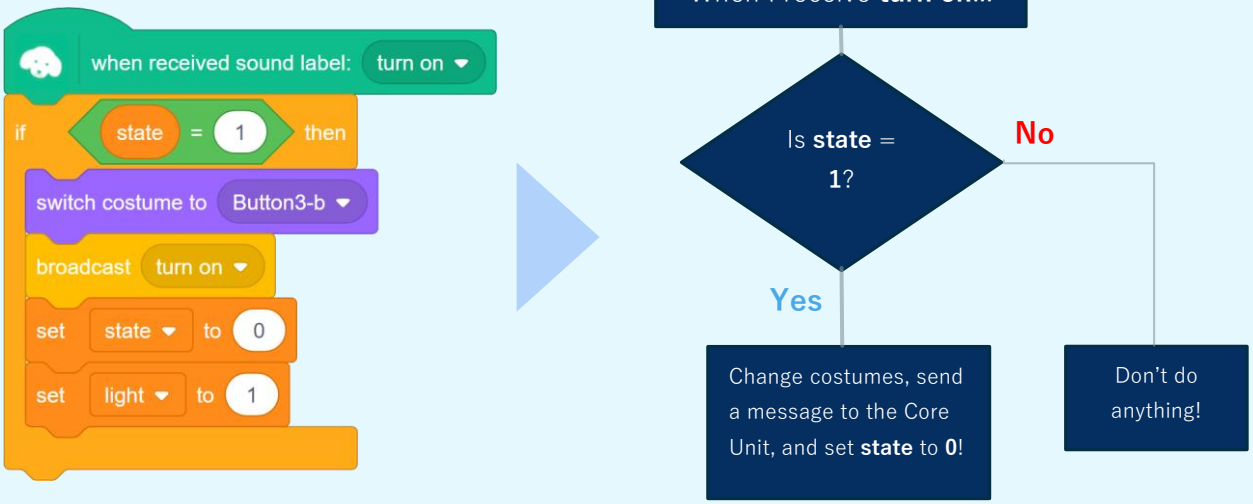
## Conditions

Conditions tell your program to do different things if certain conditions are met, just like a person might change their plans depending on what happens that day! If you're hungry, for example, you'll eat something. If not, you won't eat anything. This is something you do every day! To make this a condition, you would ask yourself: Am I hungry? What you do next would depend on if the answer is yes or no! You can also map conditions using flowcharts like the one below:




Let's take a closer look at our program on page 25 which receives the message **turn on**. The condition asks whether the value of **state** is **1**. This part uses an **if, then** block to set the condition. If the answer is yes, the blocks inside of this orange block will run!

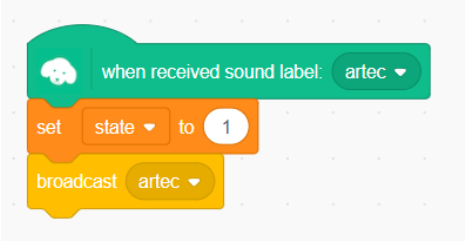


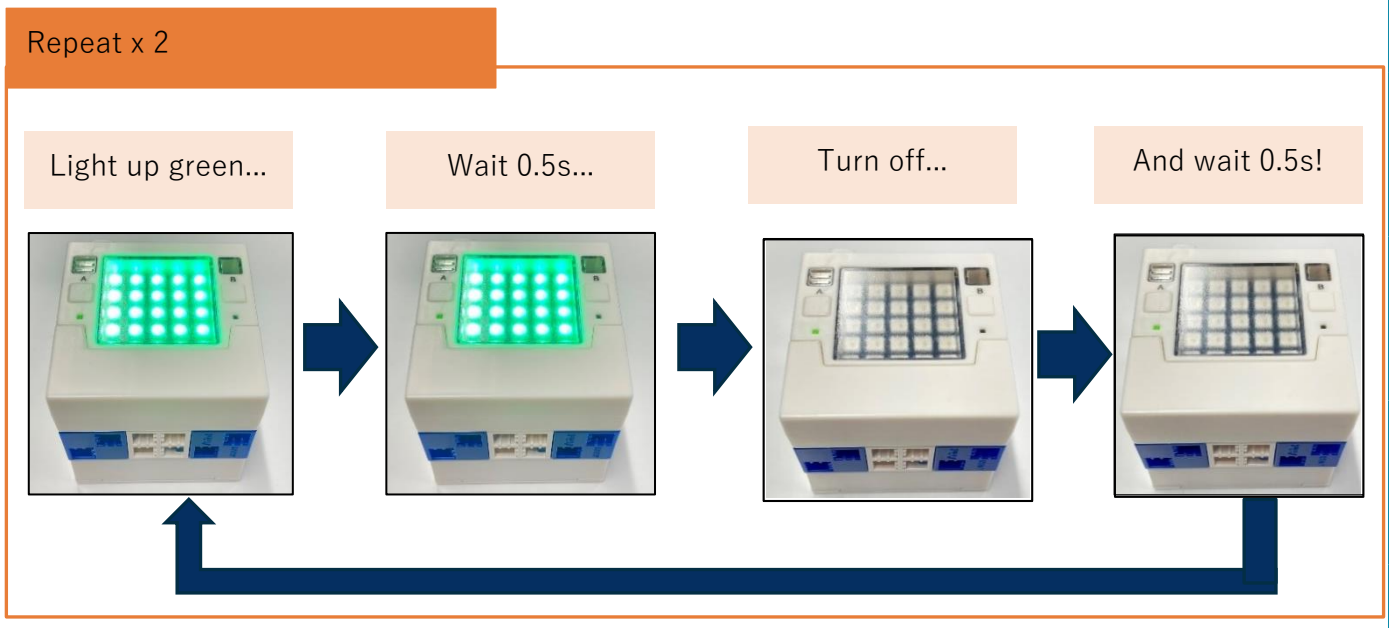
④ Now let's program the smart light to recognize the phrase. We just learned how to change the state of the smart light by using the value of the **state** variable, but that doesn't tell us whether the light can actually recognize the phrase. To solve this, we're going to make the smart light even easier to use by making it light up in green when the value of **state** is **1**!

Let's start by adding a block which sends a message to the Core Unit when the computer hears the phrase.

Button3




Now let's program the light to flash green twice when it hears the phrase!



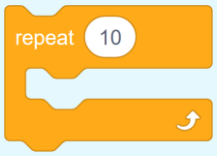

Make the set of blocks below and click them to see if your Core Unit flashes green!

Core Unit

**Iterative Processing**

Iterative processing means repeating the same action multiple times. It's used all the time in programming to repeat the same thing with regularity! Think about making a really long program to make your smart light repeat 100 times. All of that programming and editing would leave a human feeling pretty tired!

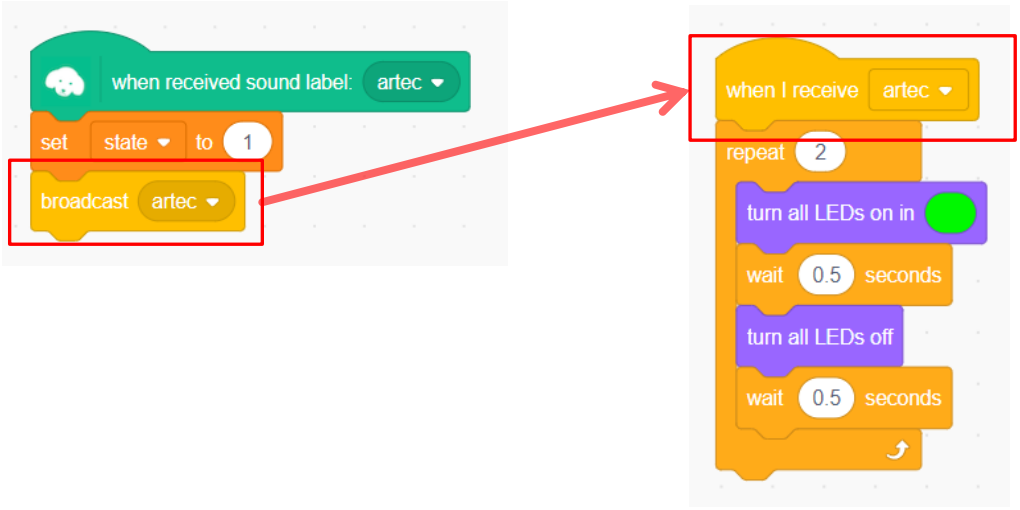

The good news is that you don't have to make the same program over and over again since there's a very handy block that lets us repeat parts of our program very easily. Just wrap your blocks in a **repeat x times** block, set a number, and those blocks will repeat that number of times!




⑤ Now let's add blocks that send and receive the message to make our smart light blink when you say the phrase.

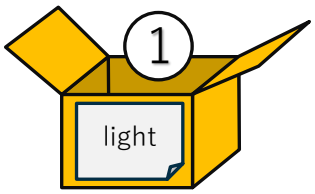
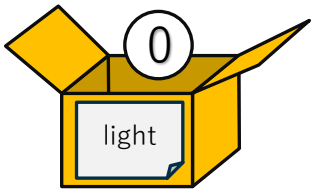
Now say it and see if it works!

| Button3 | Core Unit |
|---|---|

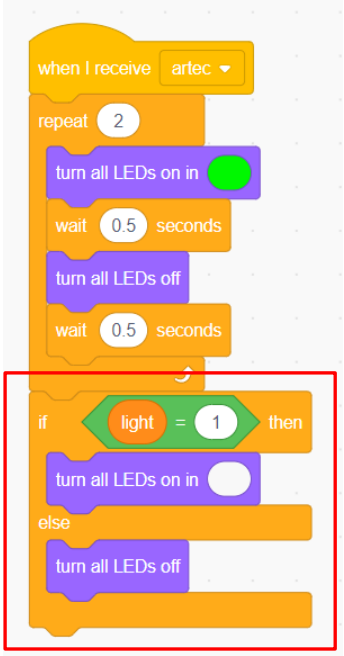


⑥ Now let's program the smart light to go back to its original state after you say the phrase. We just made the smart light flash green, what do you think it should do after that?

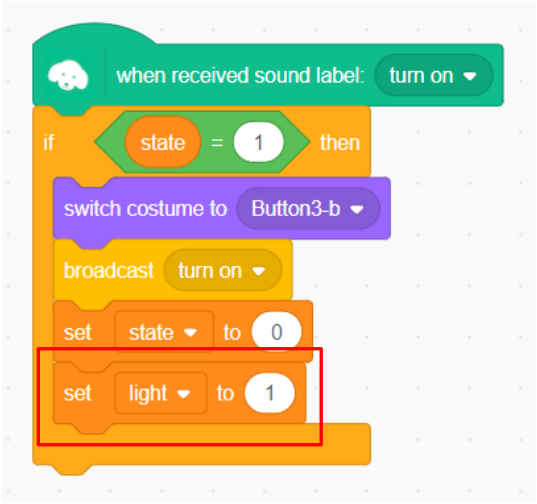
Let's make a new **light** variable and use it to reset the light back to its original state!

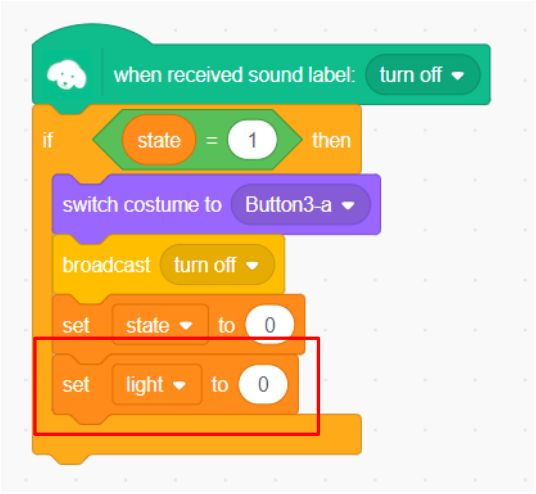
| When light is... | The smart light... |
|---|---|
| 1   | Lights up!  |
| 0   | Turns off!  |

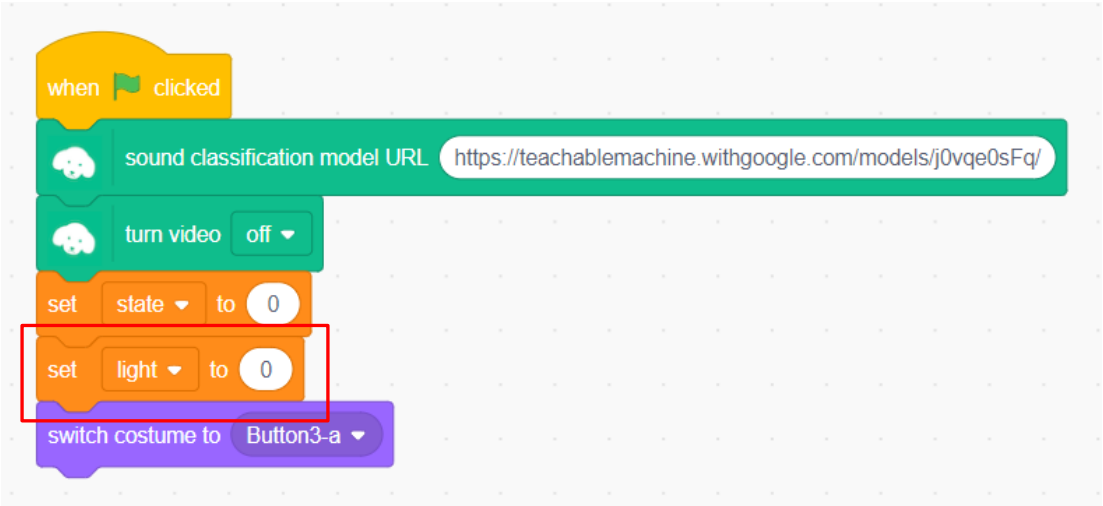
Core Unit



Button3

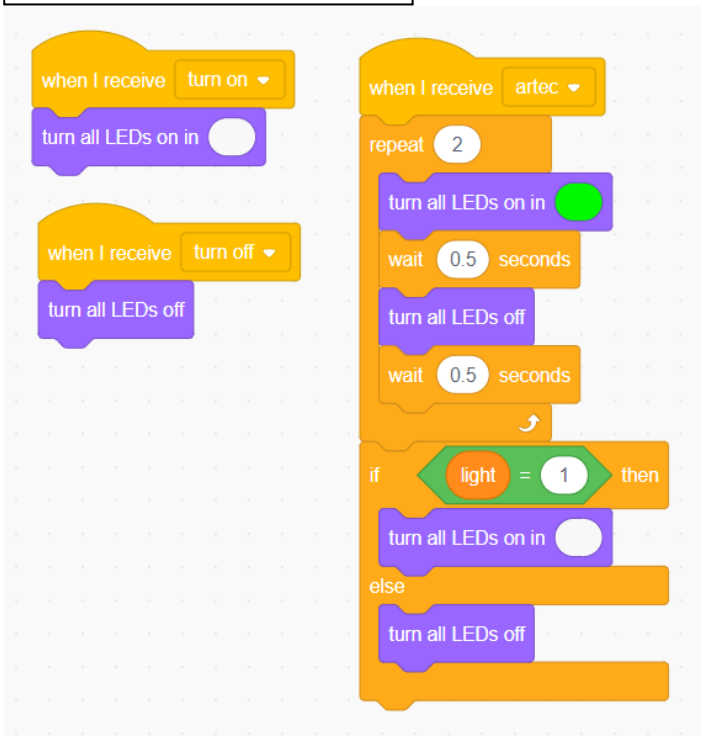

when received sound label: turn on
if state = 1 then
switch costume to Button3-b
broadcast turn on
set state to 0
set light to 1


when received sound label: turn off
if state = 1 then
switch costume to Button3-a
broadcast turn off
set state to 0
set light to 0

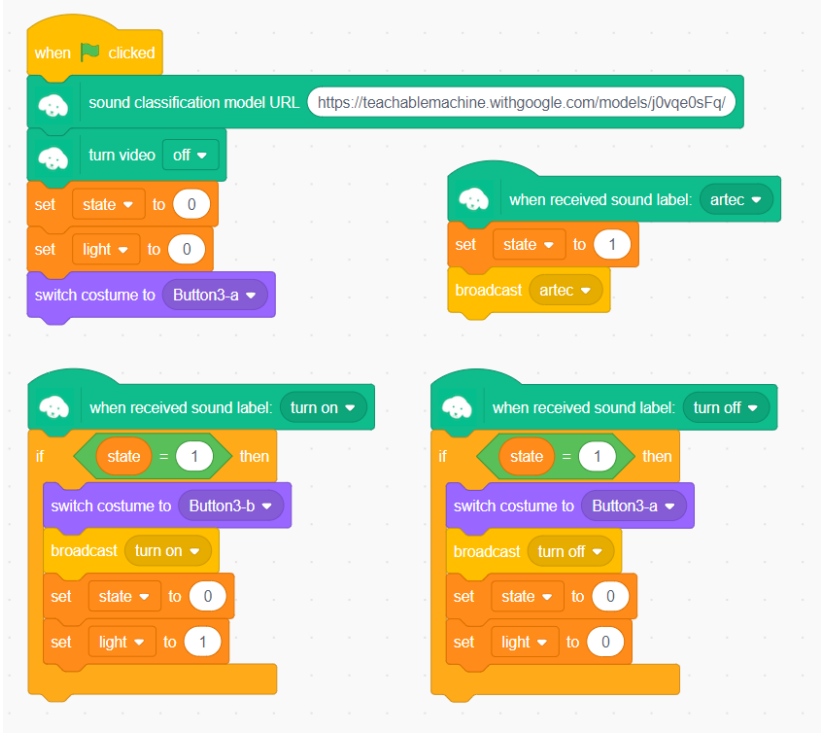

when clicked
sound classification model URL https://teachablemachine.withgoogle.com/models/j0vqe0sFq/
turn video off
set state to 0
set light to 0
switch costume to Button3-a

## The Finished Program

Core Unit




Button3

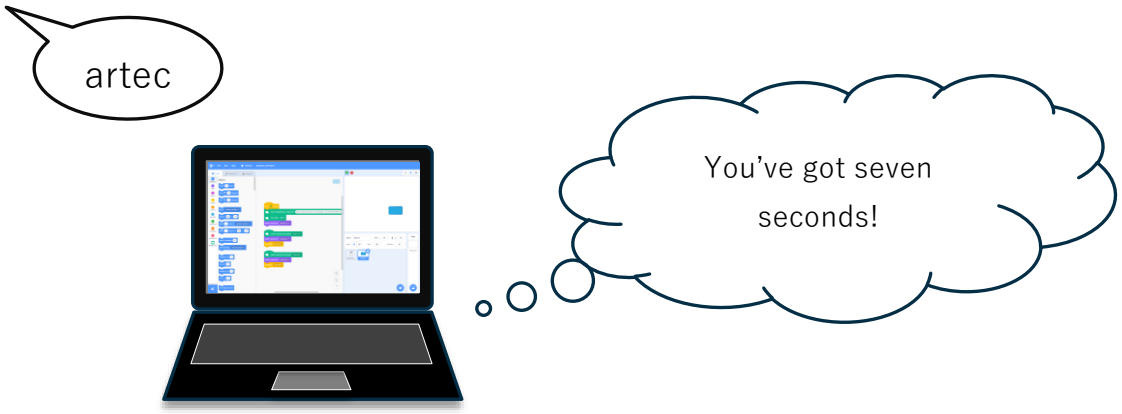



## Saving Your Program

Go ahead and save this program as **ai23**!

## 6 - 4 . The Challenge

The ai23 program we made in 6-3 leaves the value of **state** as **1** after you say the secret phrase. This means that you can still use voice commands to control your light one hour or even one day later. Even for a robot, that seems like a pretty long time to keep a conversation going! In this challenge, you'll program your smart light to stop recognizing commands seven seconds after you say your secret phrase!

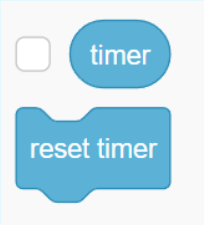



**Using Timers**

Click the **Sensing** icon and you'll find a special **timer** block.

Timer blocks count the amount of time that has passed since you opened the ByteCode app.

You can use a **reset timer** block to reset the timer to 0!

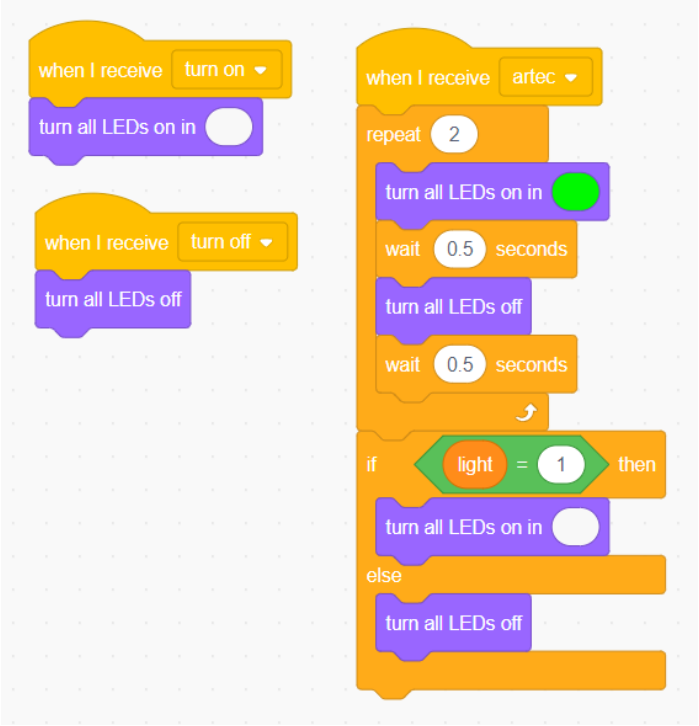




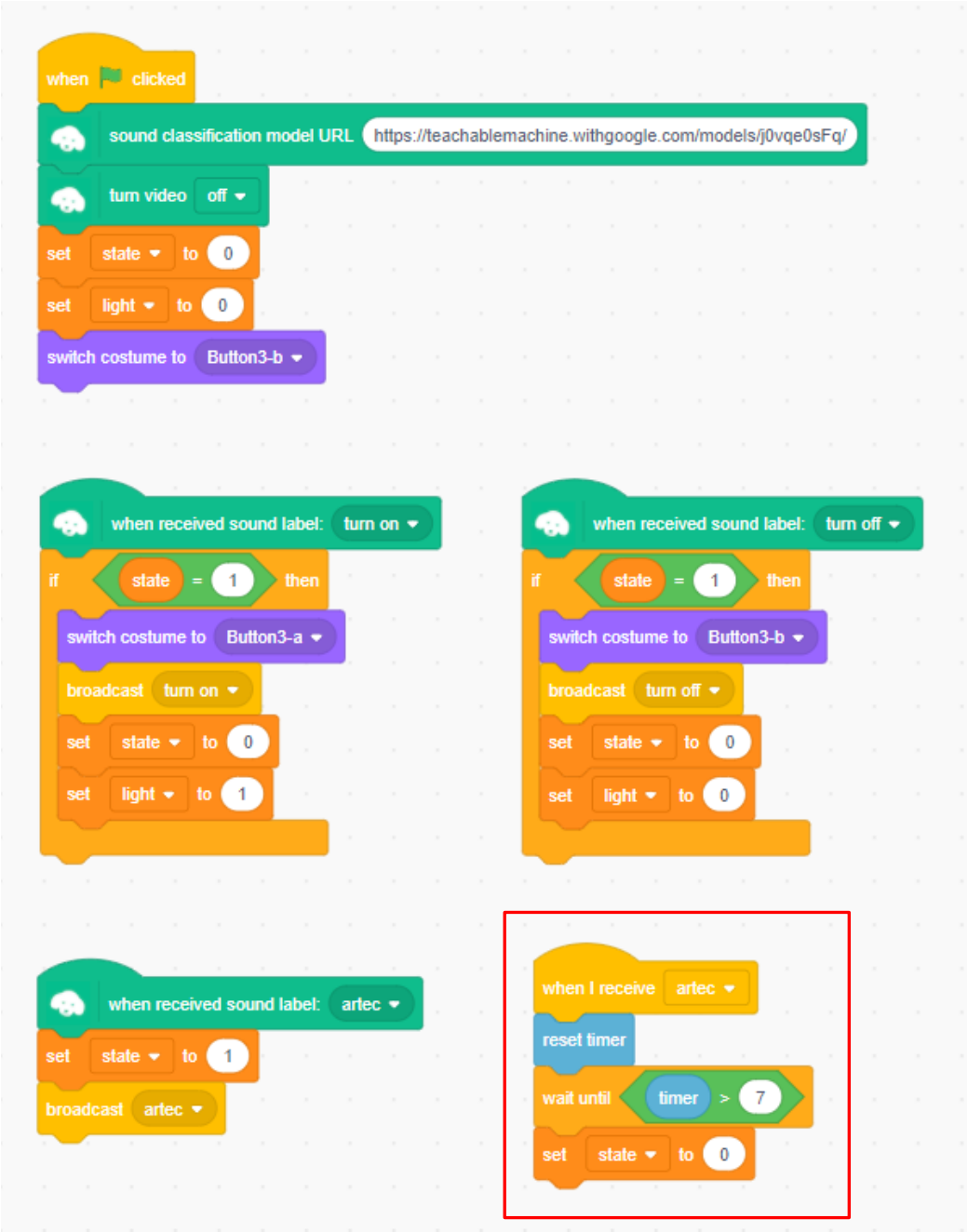
You can also check here to show the current value of the timer!

**Example Program**

| Core Unit (this is the same as Programming ②) |
|---|



Button3




This program resets the timer when it hears the secret phrase, then waits seven seconds.

Once seven seconds have passed, state gets automatically reset to **0** and your voice commands will no longer work!

## Saving Your Program

Go ahead and save this program as **ai24**!

# Notes