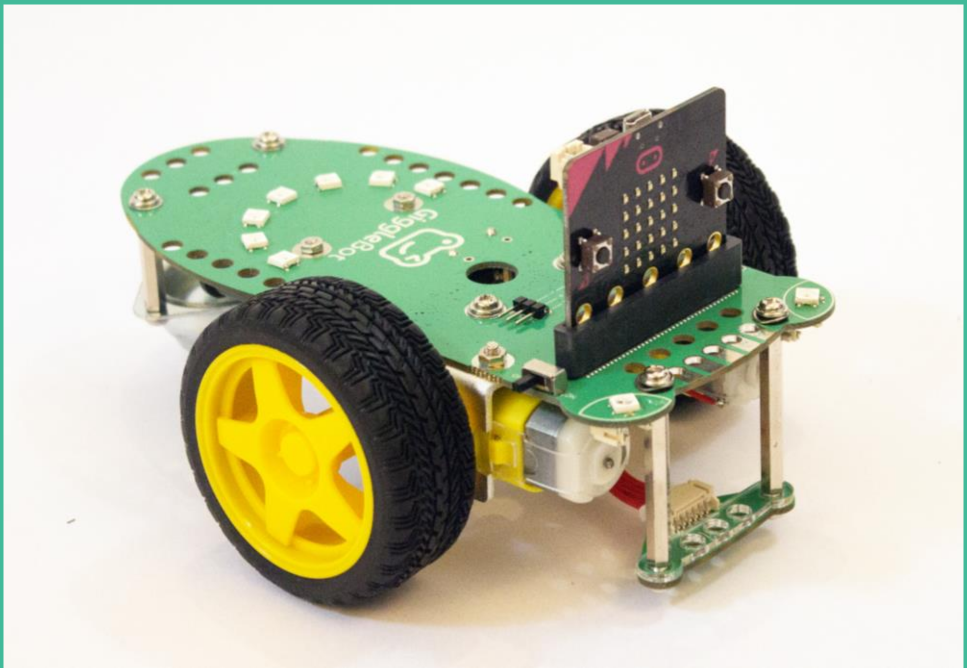
BUST A MOVE

MISSION 1



GiggleBot

TABLE OF CONTENTS

- **INTRODUCTION AND PREPARATION** 1

Your Role: Choreographer 1

Your Task: turn your robot into a dancing machine! 1

Considerations 1

Materials 1

- **LEARN: MAKE IT MOVE!** 2

Extension 2

Event blocks 3

Move blocks 3

Downloading code 5

More blocks 6

- **PLAN IT OUT** 9

Build Your Program 9

Build Your Costume 9

Are you stuck?? 10

- **TRY IT OUT** 11

Iterate 11

- **EXTENSION** 12

> INTRODUCTION AND PREPARATION

In this first mission, we're going to make our robot dance!

YOUR ROLE: CHOREOGRAPHER

> ⓘ A choreographer is a person who creates dance compositions and plans and arranges dance movements and patterns for dances.

YOUR TASK: TURN YOUR ROBOT INTO A DANCING MACHINE!

You will choreograph a routine for your robot to dance around the room to part of your favorite song. **Optional:** Design and build a costume for the GiggleBot to go along with your dance routine.

Your GiggleBot could groove out to a pop song such as "Can't Stop the Feeling" or Pharell William's "Happy". Perhaps your GiggleBot is going to transform into Batman and race into action. Maybe your robot will "fly" alongside Hedwig from Harry Potter. The options are endless!

CONSIDERATIONS

- How does the GiggleBot move? How are the movements of the GiggleBot similar / different from that of a human dancer?
- Look at the GiggleBot. What are some other things you could program the GiggleBot to do during the song?

MATERIALS

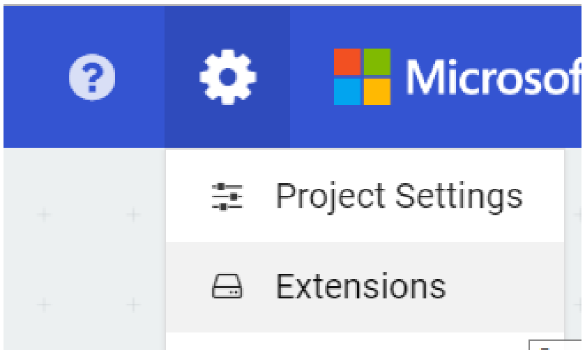
- GiggleBot with batteries, micro:bit with usb cable
- Your favorite music
- Laptop / computer
- Optional (for the costume): craft supplies to build the costume such as construction paper, pipe cleaners, cloth scraps

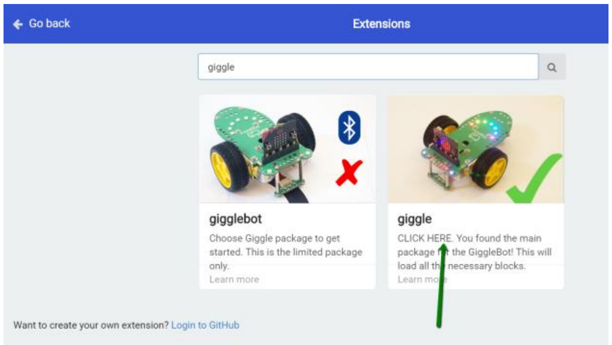
> LEARN: MAKE IT MOVE!

ADD THE EXTENSION

First, we need to learn how to program the GiggleBot. Go to https://makecode.microbit.org/. We now need to add blocks for the GiggleBot.

Click on the gear icon found top right and then **Extensions**. Type in *giggle* in the search area. There are two options that show up. For this mission and for almost every other project, add the *giggle* package.





gigglebot

Choose Giggle package to get started. This is the limited package only.

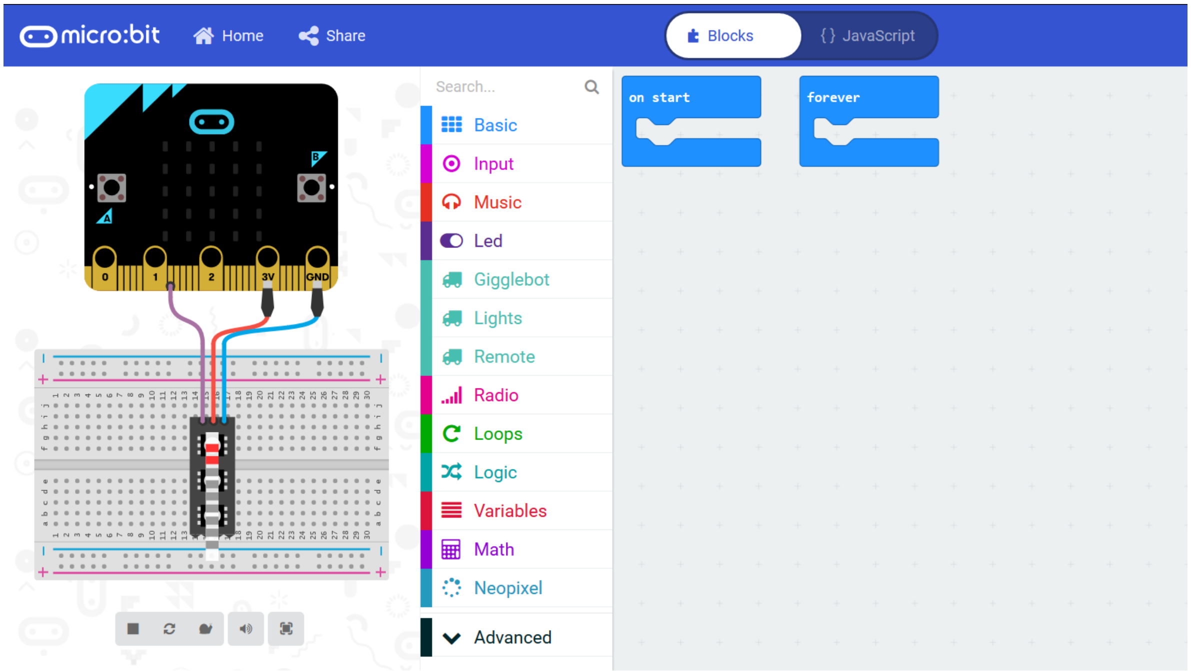
Learn more

giggle

CLICK HERE. You found the main package for the GiggleBot! This will load all the necessary blocks.

Learn more

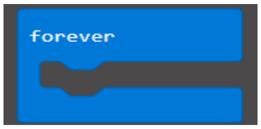
Want to create your own extension? Login to GitHub



EVENT BLOCKS

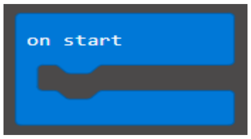
To create your program, you need to connect the movement blocks to another block that tells the GiggleBot when to start moving. These are called event-handler blocks.

Two event-handler block options can be found under Basic.



```
forever
```

One option is a forever block. Whatever is inside of a forever loop will repeat forever (or until the GiggleBot is turned off).



```
on start
```

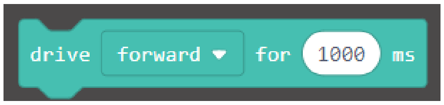
Another option is the “on start” block. If you use this block, your program will run one time right after the GiggleBot is turned on. We will look at some other options for starting your program a little later.

MOVE BLOCKS

Now, look at the blocks under GiggleBot. What block options do you have to tell the GiggleBot to move?

To move the GiggleBot forward you have two block options:

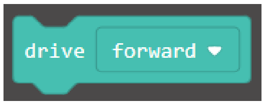
Block 1: timed



```
drive forward ▾ for 1000 ms
```

If you want the GiggleBot to drive forward for a specific amount of time, use block 1.

Block 2: unlimited

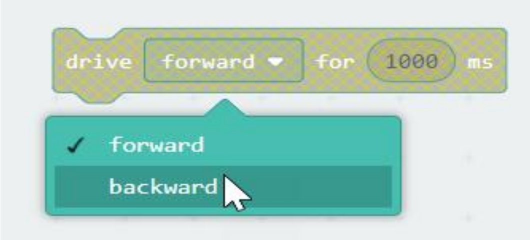


```
drive forward ▾
```

If you want it to drive forward forever or until told otherwise, use block 2 instead.

ⓘ ms stands for milliseconds. There are 1,000 milliseconds in one second. You can convert seconds to milliseconds by multiplying by 1,000. If you would like the GiggleBot to drive forward for 1.5 seconds, you will enter 1500 ms.

Using the drop-down on the blocks, you can change the block from forward to backward.



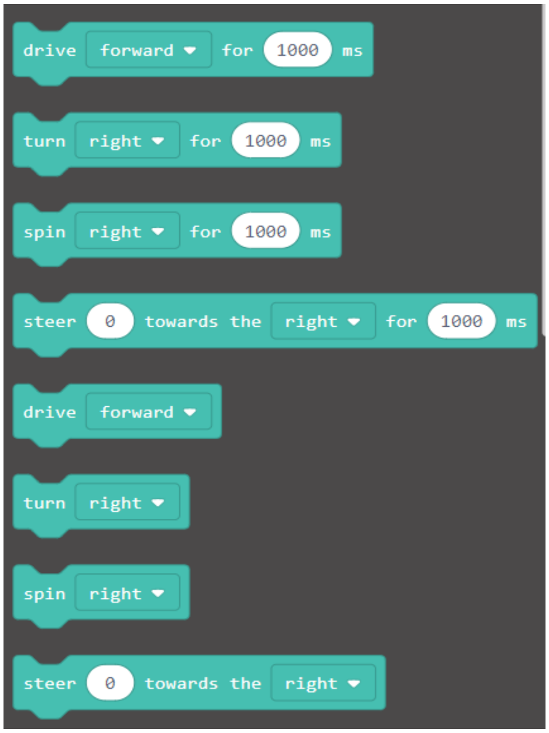
```
drive forward ▾ for 1000 ms
```

There are also several blocks to tell the GiggleBot how to turn.

A **turn** allows you to go around an object.

A **spin** tells the GiggleBot to turn around on the spot.

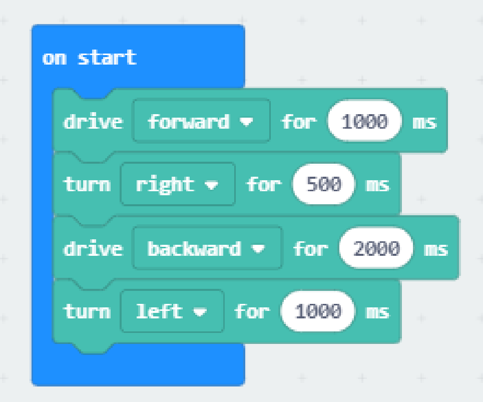
Steering can be used to program the GiggleBot to orbit around a "sun".



```
drive forward ▾ for 1000 ms
turn right ▾ for 1000 ms
spin right ▾ for 1000 ms
steer 0 towards the right ▾ for 1000 ms
drive forward ▾
turn right ▾
spin right ▾
steer 0 towards the right ▾
```

Let's try a quick program and get your GiggleBot moving. Connect a combination of forward, backward, left and right blocks (experiment with turn and spin!) to the "**on start**" block.

An example is shown here:

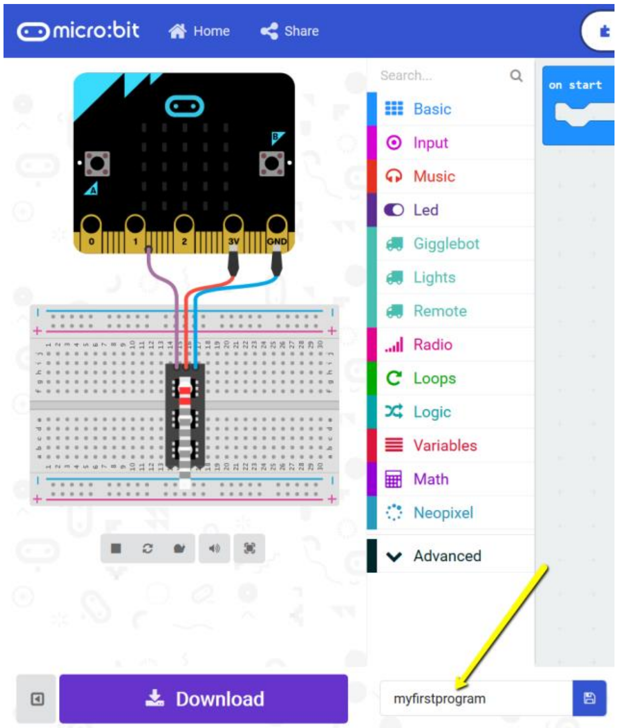


```
on start
    drive forward ▾ for 1000 ms
    turn right ▾ for 500 ms
    drive backward ▾ for 2000 ms
    turn left ▾ for 1000 ms
```

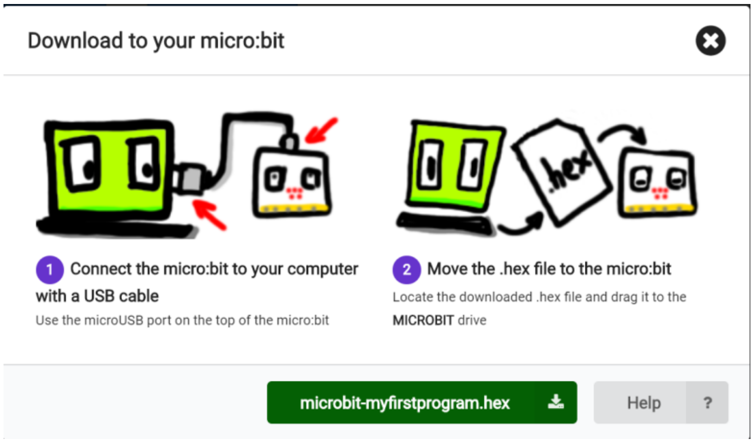
DOWNLOADING CODE

The micro:bit can only hold one program at a time so you have to transfer your program to it each time you make a change.

> When you are finished writing your program, connect your micro:bit to the computer using the provided USB cable. The micro:bit will be detected by your computer and show up as an external drive.



- Name your program and click the big purple Download button.
- You will be prompted to download the **.hex** file onto the micro:bit (a .hex file is a techy word for the code you wrote). Follow the directions on the screen to transfer the program to your microbit.



Download to your micro:bit

1 Connect the micro:bit to your computer with a USB cable
Use the microUSB port on the top of the micro:bit

2 Move the .hex file to the micro:bit
Locate the downloaded .hex file and drag it to the **MICROBIT** drive

The micro:bit will flash yellow while the file is being transferred (check the tiny light at the back of the micro:bit). When it stops flashing, the file has been transferred and you can plug the micro:bit back onto the GiggleBot.

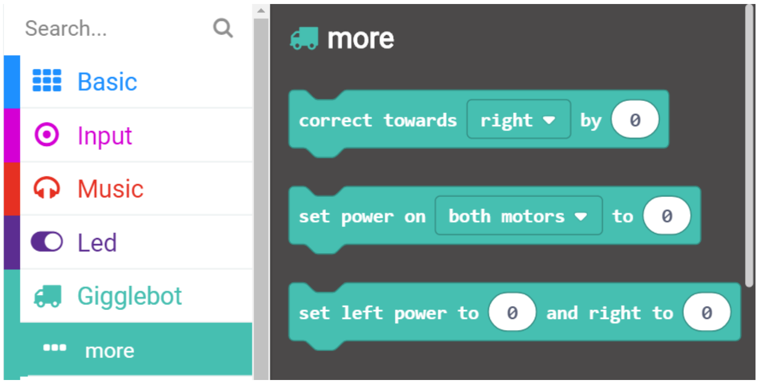
MORE BLOCKS

Jot down notes about how the GiggleBot moved and any changes you want to make for your next program.

Let's learn about a few more blocks before you program your dance routine.

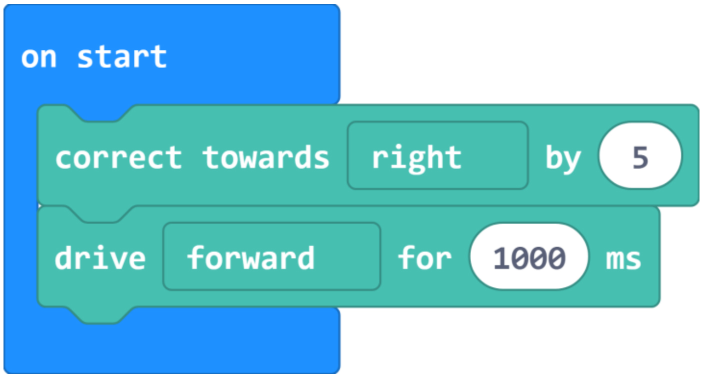
CORRECTING THE DIRECTION

Did your GiggleBot drive in a straight line? If not, you can use the **correct** block to correct the motors on the GiggleBot. If it drives to the left, then correct towards the right. If it drives to the right, then correct towards the left. The trim correction value can be between 0 and 100. Likely the value will be below 10. You will find that block under "Gigglebot/more…"



```
correct towards right by 0
set power on both motors to 0
set left power to 0 and right to 0
```

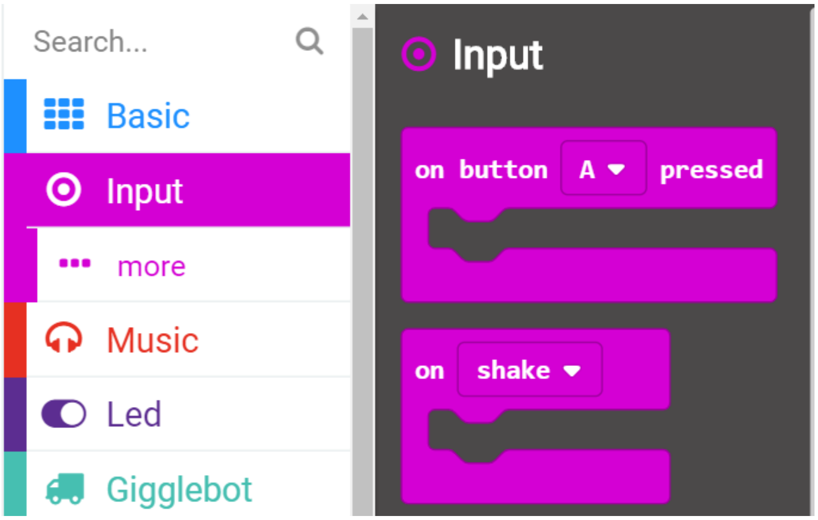
For example, if your GiggleBot is driving to the left slightly then you could use this program to help correct it.



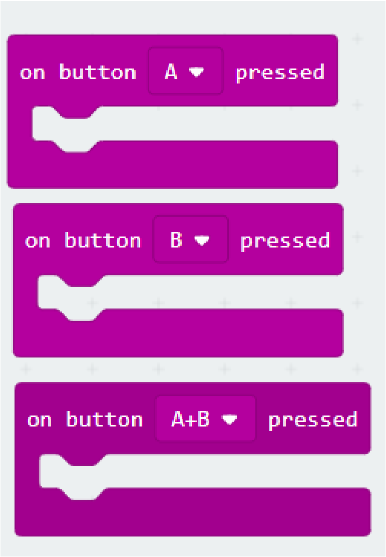
```
on start
    correct towards right by 5
    drive forward for 1000 ms
```

BUTTON BLOCKS

There are other ways to initiate your program. There are two buttons on the micro:bit, labeled A and B. You can use these to start your program. You can find the block "**on button A pressed**" under **Input**. Using the drop-down menu, you can change this to "**on button B pressed**" or "**on button A+B pressed**."

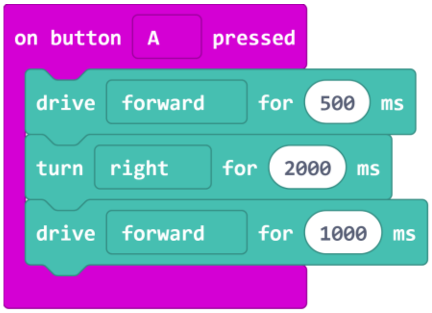


```
on button A ▾ pressed

on shake ▾
```

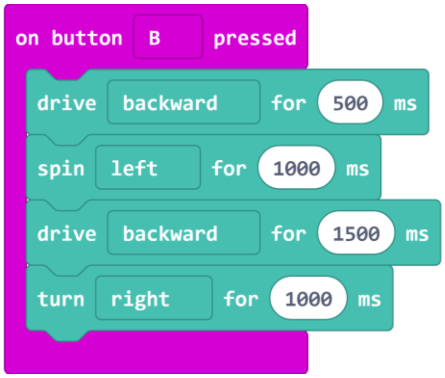


```
on button A ▾ pressed

on button B ▾ pressed

on button A+B ▾ pressed
```

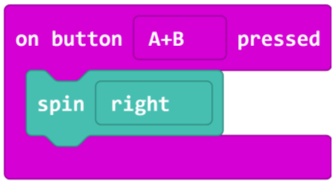
Using these blocks, you can create three different scripts for your GiggleBot to perform. Create a new program that uses the buttons to tell the GiggleBot when to perform each task. An example is shown next. When you are finished with your new program, save it and transfer it to your GiggleBot to test it out.



```
on button A pressed
    drive forward for 500 ms
    turn right for 2000 ms
    drive forward for 1000 ms
```



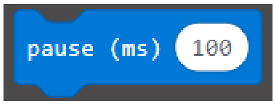
```
on button B pressed
    drive backward for 500 ms
    spin left for 1000 ms
    drive backward for 1500 ms
    turn right for 1000 ms
```



```
on button A+B pressed
    spin right
```

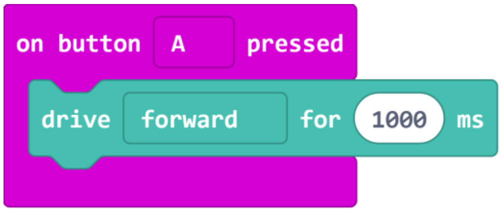
TIME BLOCKS

Now that you can make the GiggleBot do several different movements, let's learn about a few more helpful blocks.

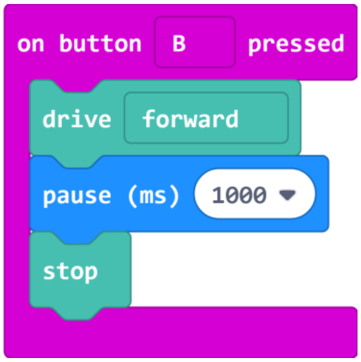


```
pause (ms) 100
```

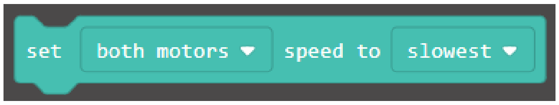
The **pause** block, found under **Basic**, tells the GiggleBot to keep doing whatever it was doing for a certain amount of time. For example, the **drive forward for 1000 ms** block does the same thing as **drive forward** + **pause 1000 ms** + **stop**. This means that the following two sequences do the same thing.



```
on button A pressed
    drive forward for 1000 ms
```



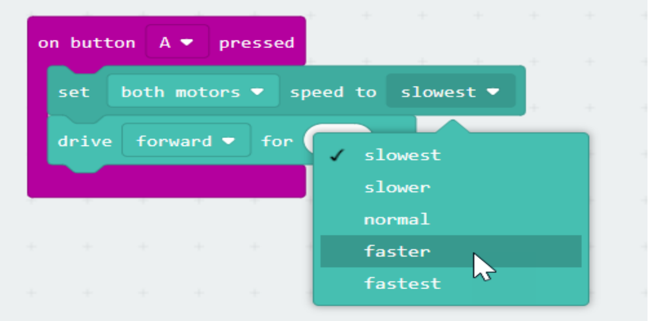
```
on button B pressed
    drive forward
    pause (ms) 1000 ▾
    stop
```



```
set both motors ▾ speed to slowest ▾
```

You can also control the speed of the GiggleBot. Under the GiggleBot tab, find the "**set <motor> speed**" block.

Using the drop down menu you can change the motor speed to slowest, slow, normal, fast, or fastest. This block needs to be used in conjunction with a block that tells the robot how to move such as forward or backward.



```
on button A ▾ pressed
    set both motors ▾ speed to slowest ▾
    drive forward ▾ for
```

MODIFY YOUR LAST PROGRAM

Include some pauses and changes in the motor speed. Download the program and transfer it to your GiggleBot.

- Did it move as you anticipated?
- How might you incorporate these moves into the GiggleBot's dance routine?

> PLAN IT OUT

Now it is time for you to plan out your robot's dance moves. What do you want the GiggleBot to do? Listen to the song you chose while you plan out the dance moves for your robot. It does not need to dance to the entire song. Choose a snippet of the song to start out with.

Write down the steps on a piece of paper using words from the blocks such as **forward**, **backward**, **spin left**, **turn right**, **stop**, **slow**, and **fast**.

BUILD YOUR PROGRAM

To build your program, go back to https://makecode.microbit.org/ and follow the steps in your plan. Don't forget to include a block to tell the GiggleBot when to start running the program. If you need help, go back to the Learn section above or scroll down to the Are You Stuck? section to get step by step help.

When you are finished with the program, download it and then transfer it to your micro:bit.

BUILD YOUR COSTUME

Use materials around your house or classroom to create a costume for the GiggleBot that matches the song.

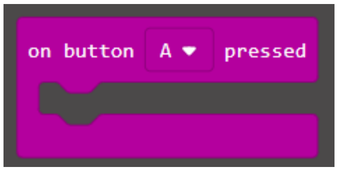
ARE YOU STUCK??

Let's build a program together! The music I chose for the GiggleBot to dance to is "Happy".

Here are my notes for what I want the GiggleBot to do during the song:

- Spin around (right, 4 seconds)
- Turn the other way (left, 4 seconds)
- Move fast
- Forward 2 seconds
- Move slowly
- Backwards 3 seconds
- Spin quickly (right, 4 seconds)

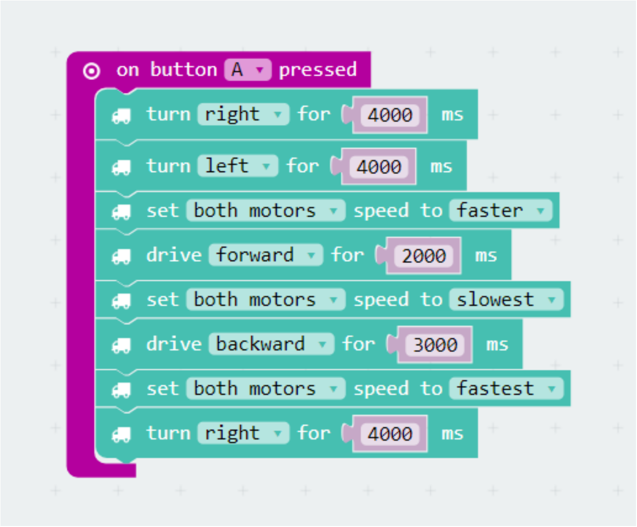
Now, let's translate that into blocks that the GiggleBot understands and determine what block to use to start the program.



```
on button A pressed
```

If we use a "**on button A pressed**" block, we can control when the GiggleBot starts to dance. The GiggleBot will wait to run the program until button A is pressed.

Next, we will follow the notes above and connect blocks to our "**on button A pressed"** block.



```
on button A pressed
    turn right for 4000 ms
    turn left for 4000 ms
    set both motors speed to faster
    drive forward for 2000 ms
    set both motors speed to slowest
    drive backward for 3000 ms
    set both motors speed to fastest
    turn right for 4000 ms
```

> TRY IT OUT

Ensure that you have put the micro:bit back into your GiggleBot. Turn on the GiggleBot and start your program. Reminder: if you program began with an **on start** block, it will run right after the GiggleBot is turned on. If the program starts with an **on button ____ pressed**, you will need to press the appropriate button to start the program.

ITERATE

Did your GiggleBot's dance routine go exactly as you had anticipated? Do you see any areas that you would like to modify or improve upon? In engineering, we try a solution, think about what needs to change, and we iterate.

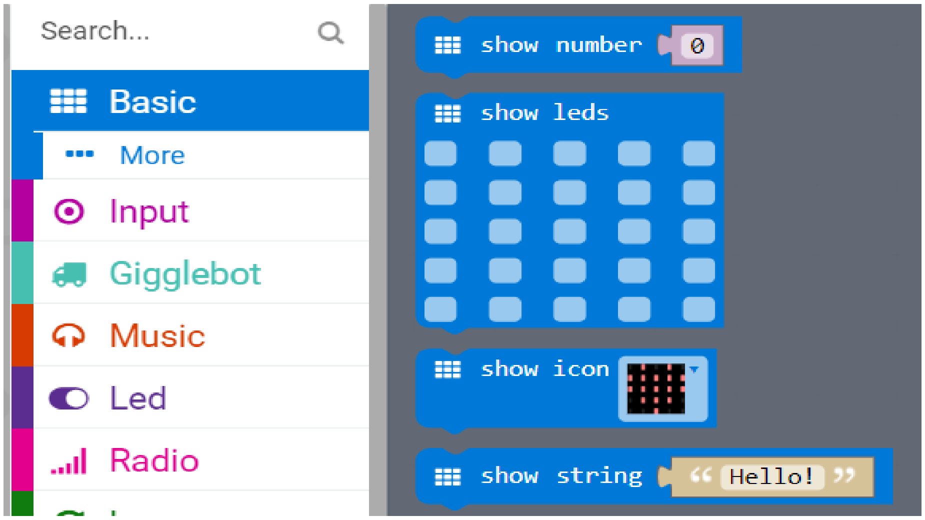
As you modify and revise your program, save each new program with a version number – you never know when you might want to take another look at an older version (for example: Dance_V1 where "V" stands for version).

> EXTENSION

> What else can you choreograph?

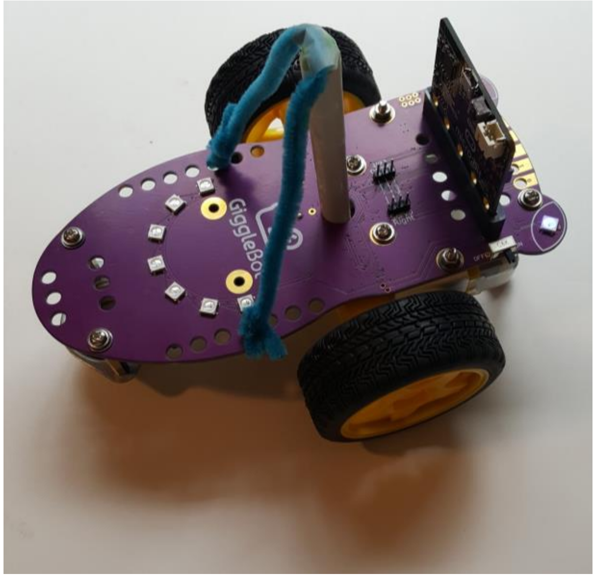
- Can you choreograph the GiggleBot to conduct a ceremony?
- Act out a historical event?
- A wedding?
- A gymnastics routine?
- A marching band show?
- A bank heist?

> The GiggleBot and micro:bit have numerous onboard LEDs! Click on the Basic tab. The blocks below allow you to control what is displayed on the LEDs on the micro:bit aboard the GiggleBot. You can show numbers, create an image by controlling individual LEDs, show an icon, or display text.



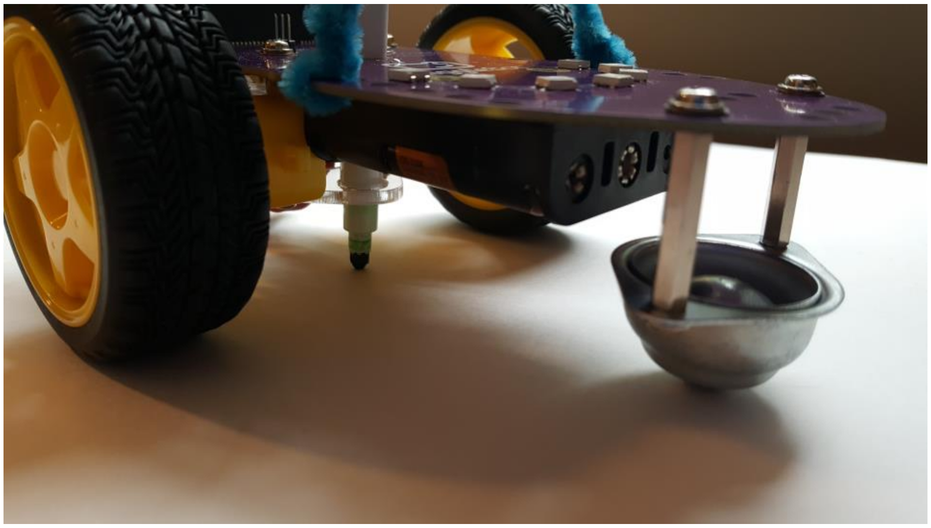
```
show number 0
show leds
show icon
show string "Hello!"
```

> Challenge your GiggleBot to a dance battle or have a dance battle with another GiggleBot.



- Draw with the GiggleBot. Get a large piece of paper (or tape several pieces of paper together) to make a space for the GiggleBot to draw on. Then, place a pencil or marker into the hole in the middle of the GiggleBot's body. Make sure that the pencil or marker is making good contact with the paper. Secure the pencil or marker with a rubber band or pipe cleaner.

Ensure that the pencil or marker has good contact with the paper so that the GiggleBot will draw nice lines.



Your GiggleBot is now an artist! Run your dance program or write a new one to tell the GiggleBot to draw a picture.

Giggle Bot