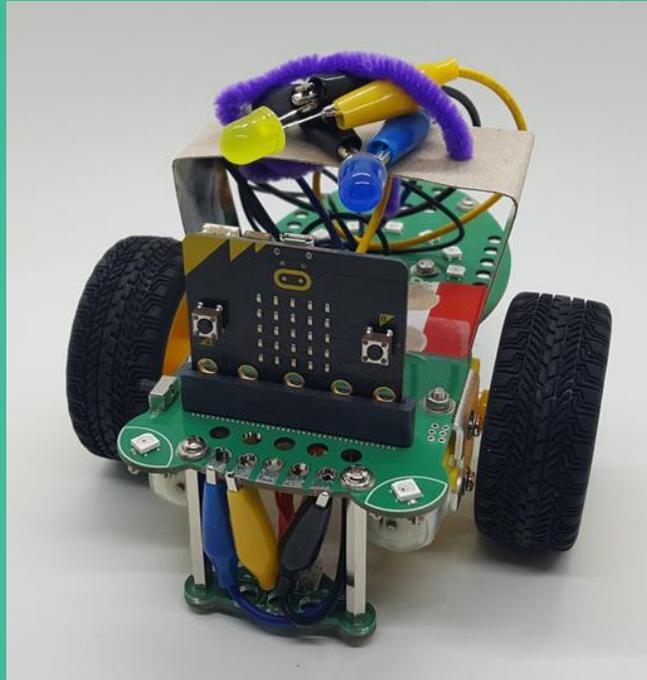


CAR DESIGN

MISSION 9



Giggle
Bot

TABLE OF CONTENTS

• CAR DESIGN	1
Your Role: Automotive Designer	1
Your Task: Upgrade It	1
Considerations.....	1
Materials.....	1
• LEARN: CONNECTING LEDS	2
Light Up an LED Using the Pins	2
Let There Be Light.....	3
Light Up Two LEDs Using a Parallel Circuit	4
Light Up Two LEDs Using Two Simple Circuits	6
• PLAN IT OUT	8
Build Your GiggleBot Car.....	9
Build Your Program.....	9
• ARE YOU STUCK??	10
Design and Build Your GiggleBot Tow Truck	10
Plan Your GiggleBot Program.....	11
Program Your GiggleBot	12
• TRY IT OUT	17
Iterate	17
Extension	17



> CAR DESIGN

YOUR ROLE: AUTOMOTIVE DESIGNER

📘 Automotive design is the process of developing the appearance of motor vehicles, including but not limited to automobiles.

YOUR TASK: UPGRADE IT

Transform the GiggleBot into a car with additional LEDs as headlights, blinkers and/or a backup light.

CONSIDERATIONS

- > Where will the blinkers / turn signals be located on the GiggleBot?
- > Where will the backup light be?
- > What will the lights blink like turn signals on current vehicles or will you use another light pattern?
- > Will the lights automatically turn on when it gets dark and turn off when it is bright?

MATERIALS

- > GiggleBot with good batteries
- > micro:bit and provided cable
- > Laptop / computer
- > LEDs
- > Alligator clips
- > Optional: craft supplies to transform the GiggleBot into a car (ex: cardboard, pipe cleaners, and construction paper)

> LEARN: CONNECTING LEDs

LIGHT UP AN LED USING THE PINS

In mission 7, you connected a speaker to the GiggleBot using pins on the front of the GiggleBot. We are now going to use those same pins to light up even more LEDs!

Attach an alligator clip to PO and another alligator clip to GND.



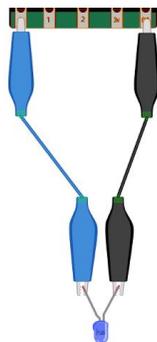
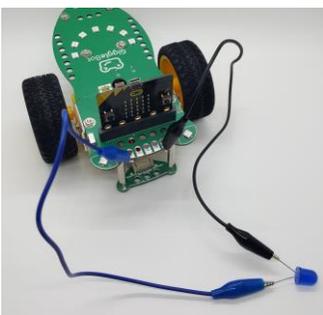
Take a look at one of your LEDs. The legs on it are two different sizes. The longer leg is the positive side (also known as *anode*) and the shorter leg is the negative side (also known as *cathode*).

> Carefully spread the legs on the LED apart so that you can connect an alligator clip to each one without the alligator clips touching each other. Keep track of which one is the longer leg as it will be more difficult to figure out with the legs apart from each other.



> Connect the shorter leg (the *cathode*) to the GND alligator clip and the longer leg (the *anode*) to the PO alligator clip.

> Pull the plastic cover down over each clip so that the two clips do not touch one another.



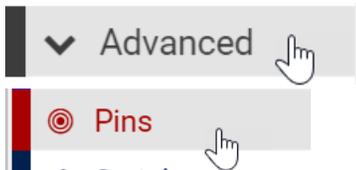
Here's the final circuit for our one LED project and the corresponding circuit diagram.

LET THERE BE LIGHT

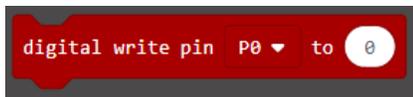
We are going to use button **A** to turn the LED on and button **B** to turn the LED off.



- > Move two **on button A pressed** blocks to the workspace. Change one of them to say **on button B pressed**.

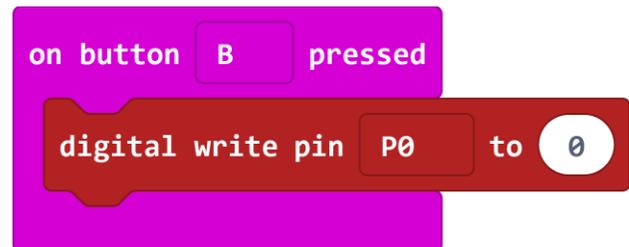
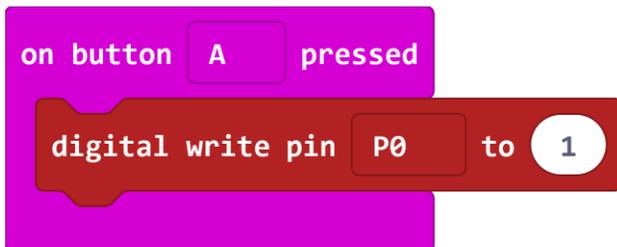


- > Next, go to **Advanced** and then **Pins**.



- > Connect a **digital write pin P0 to ___** block into the **on button A pressed** block and the **on button B pressed** block.

With this block you can set the pin value to 0 or 1. For the LED, 0 is off and 1 is on. Change the value for the block inside **on button A pressed** to 1. This means that when button **A** is pressed, the LED will turn on (pin value 1) and when button **B** is pressed, it will turn off (pin value 0).



Download and transfer the program to your GiggleBot. Test it out.

- > Does your LED turn on when button A is pressed?
- > Does it turn off when button B is pressed?

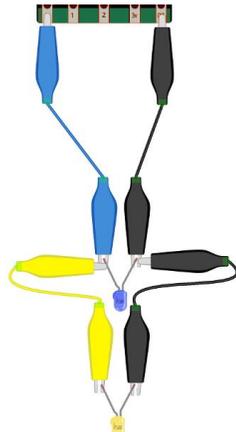
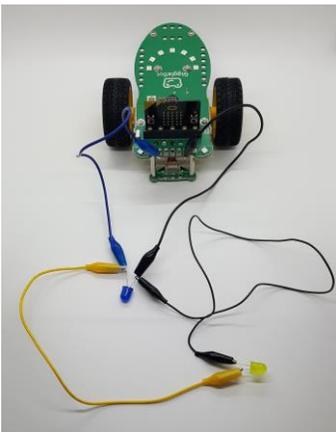
LIGHT UP TWO LEDs USING A PARALLEL CIRCUIT

Now we will connect two LEDs to your GiggleBot. There are two approaches to this, creating a series circuit or a parallel circuit.

- ❶ In a series circuit, the electric current flows in one path. This means that if there is an interruption in the path, both lights will turn off. Many Christmas lights are made with series circuits. That is why when one light goes out, they all go out.
- ❷ In a parallel circuit, the electric current flows in more than one path. We are going to create a parallel circuit so that if one light goes out, the other will stay lit.

CREATE A PARALLEL CIRCUIT:

- > Connect a second alligator clip (yellow in the photo) to the alligator clip connected to the long leg of the first LED (blue in the photo).
- > Then, connect the other end of that yellow alligator clip to the long leg of a second LED.
- > Connect the short leg of the 2nd LED to a new alligator clip (black in the photo).
- > Last, connect the other end of the alligator clip to the alligator clip already connected to the short leg of the first LED (also black in the photo).



Check your connections against the photo and circuit diagram.

Run your program again.

- > Do both LEDs light up when you press button **A**?
- > Do they both turn off if you press button **B**?
 - If not, check your alligator clips to ensure that they are connected securely and in the correct order.
 - Check that the long and short legs of your second LED are connected to the correct alligator clips.

INVESTIGATE A PARALLEL CIRCUIT:

- > Disconnect one of the alligator clips attached to the second LED (for example the yellow one from the photo on the previous page).
- > Press button **A**.

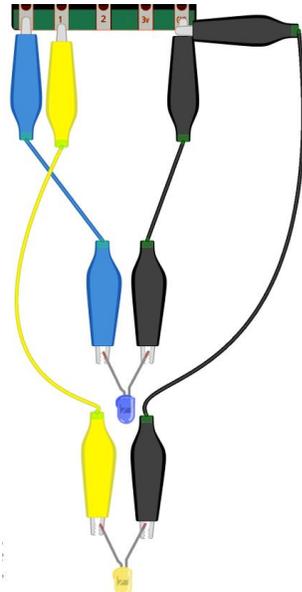
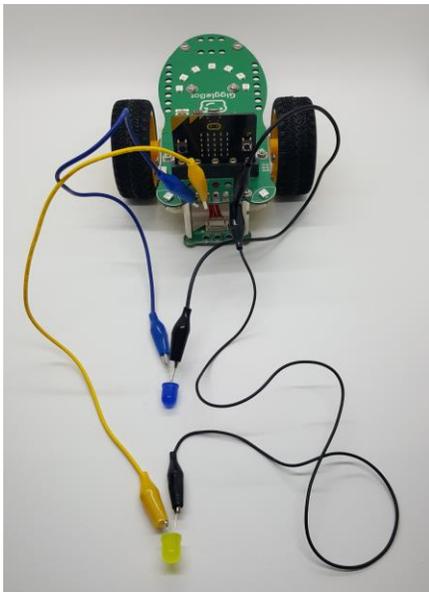
The first LED should still light up, but the second one will not. Since we created a parallel circuit, the path of electricity to the first LED is not broken, but the path to the second one is because we disconnected the alligator clip.

📍 Think about what benefits a parallel circuit could have in the creation of your GiggleBot car. How could a parallel circuit be helpful when you turn your GiggleBot into a car?

LIGHT UP TWO LEDs USING TWO SIMPLE CIRCUITS

What if you do not want the LEDs to turn off and on at the same time? What if you want them to be controlled independently of one another? Let's connect the LEDs in a different way!

- > Keep your first LED connected to the GiggleBot, but disconnect both wires going to the second LED.
- > Next, connect the yellow wire from the second LED to the GiggleBot **P1** and the black wire from the second LED to **GND**.



Check your connections against the photo and the circuit diagram.



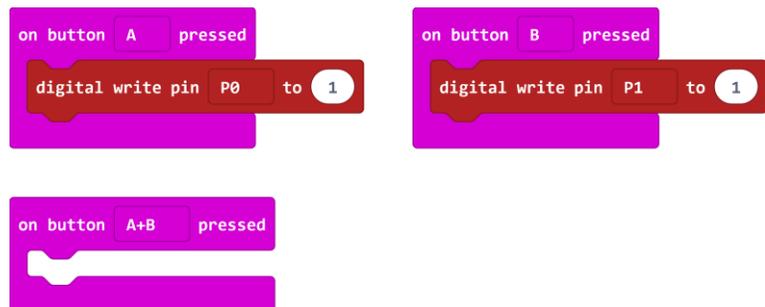
Now, we need to create a new program that allows us to control each LED independently. Let's start out with three input blocks - **on button A pressed**, **on button B pressed**, and **on buttons A+B pressed** (found under **Input**). Move these to your workspace.

Next, we need to decide what each of these button presses will do. We can tell the GiggleBot to turn on the first LED when button **A** is pressed, to turn on the second LED when button **B** is pressed, and to turn them both off when both buttons are pressed together. This program is similar to our first program, however we need to program **P0** and **P1**.



- > Connect a **digital write pin __ to __** to the **on Button A pressed** block. Change the value to **1**. This tells the GiggleBot to turn on the LED connected to **P0**.
- > Do the same thing for the **on Button B pressed** block, but change the pin to **P1** and the value to **1**.

Your code should now look like this:

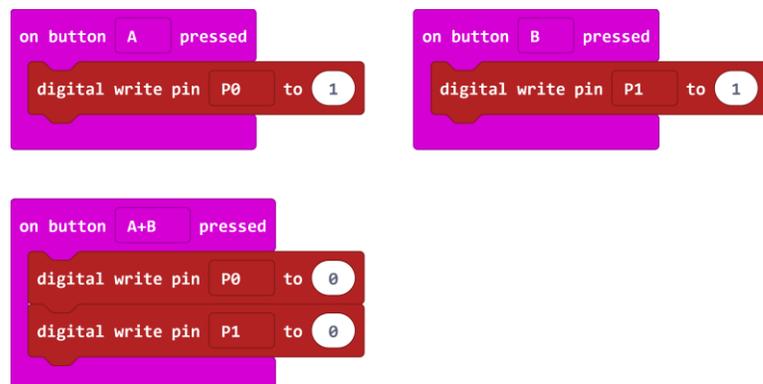


The last part is to program the GiggleBot to turn off both LEDs when button **A + B** are pressed simultaneously.



- > Connect two **digital write pin __ to __** blocks to the **on button A+B pressed** block.
- > Ensure that one of them says **P0** and the other says **P1**.

Download and transfer your program to your GiggleBot. You can now turn each LED on individually and then turn them both off. Test out your new program to see how it works.



 How could different simple circuits be helpful when you turn your GiggleBot into a car?

> PLAN IT OUT

WHAT TYPE OF VEHICLE IS YOUR GIGGLEBOT GOING TO BE?

What extra lights do you want your GiggleBot car to have?

- > Headlights?
- > Blinkers?
- > Back-up lights?
- > Emergency vehicle lights?

HOW WILL YOU CONNECT THE LEDS?

- > Will you use several simple circuits?
- > Will you use a parallel circuit?
- > Will you use a combination of the two?

Vehicle Type: _____

	Pin	Purpose
LED 1		
LED 2		
LED 3		

BUILD YOUR GIGGLEBOT CAR

- > Connect your LEDs to the pins on the GiggleBot using alligator clips.
- > Secure the alligator clips and wires to the GiggleBot body using pipe cleaners or tape.
- > Ensure that the alligator clips and wires do not interfere with the movement of the car.
You wouldn't want the wire to get stuck in the wheel or run over.

You can also build onto the GiggleBot using supplies such as recycleables, cardboard, construction paper, and tape to make it look more like a car.

BUILD YOUR PROGRAM

Once you are finished with the construction of your GiggleBot car and connecting all of the LEDs, write a program for the GiggleBot to drive around utilizing its new LED lights.

- > Maybe it drives around the room and every time that it turns, one of the LEDs blink to notify others that it is turning.
- > Perhaps your GiggleBot is a tow truck that has emergency vehicle lights to warn others of a potential hazard as it tows another vehicle or goes to rescue a broken-down car.

Remember to use the **digital write pin PO to ___** blocks to turn the LEDs on and off. You can utilize the extra lights in any way you choose!

> ARE YOU STUCK??

Let's build together!

First, we will decide what type of vehicle the GiggleBot will be transformed into and what each LED will do.

Vehicle Type: Tow truck

	Pin	Purpose
LED 1	P0	Emergency vehicle light (on top)
LED 2	P1	Emergency vehicle light (on top)
LED 3	P2	Back-up Light

DESIGN AND BUILD YOUR GIGGLEBOT TOW TRUCK

Create a structure for the emergency vehicle lights. These are the lights that flash on top of a tow truck, fire truck, police car, ambulance, or any other emergency vehicle. Use whatever materials you have on hand. You could use straws, cardboard, toy building bricks, or anything else you have.

- > Add any additional structures onto your GiggleBot to make it look more like a tow truck. This could include a tow arm or winch.

CONNECT THE LEDS

Next, we will connect the LEDs and secure them to the GiggleBot using three simple circuits. Go back to the "Light Up Two LEDs Using Two Simple Circuits" section and follow the directions to connect the first two LEDs to create the emergency vehicle lights. Then, follow the same process to connect one more LED using P2 and GND to create the backup light.

PLAN YOUR GIGGLEBOT PROGRAM

We will now program the GiggleBot tow truck. Here a possible scenario for the program to follow:

- 1 The GiggleBot tow truck is driving down the road when it receives a call that there is a broken down car.
- 2 It stops, turns around, and then speeds off to help retrieve the broken down car (emergency vehicle lights flashing).
- 3 When it arrives at the broken down car, it drives backwards (back-up light is on) to hook up to the broken down car.
- 4 Last, the GiggleBot tow truck drives away (emergency vehicle lights flashing).

If you want, you could build a model town with roads and buildings for the GiggleBot to drive around. Since this GiggleBot is going to be a tow truck that rescues a broken down vehicle, you could even place a toy car or something else on one of the roads for the GiggleBot to go pick up.

PROGRAM YOUR GIGGLEBOT



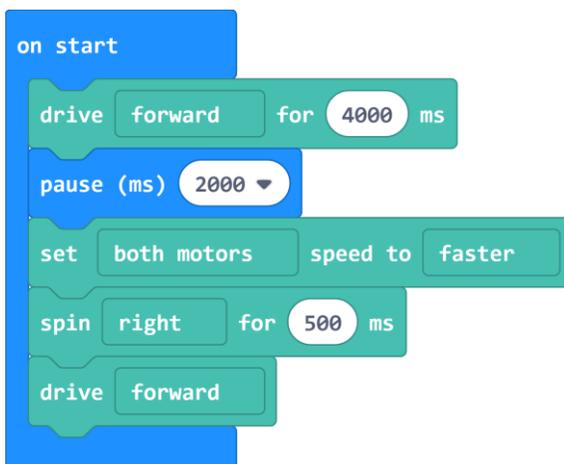
First, we need to move an event-handler block onto the workspace to start the program. For this program, we will use an block (found under Basic).



Decide how long the GiggleBot tow truck will drive before it is sent to “rescue” a broken down vehicle. Add movement blocks to the **on start** block to tell the GiggleBot to drive around. This can be one block or multiple blocks.



Add a pause block to represent how long the tow truck stops to when it finds out about the broken down vehicle.



Connect blocks to show the movement of the GiggleBot to go rescue the vehicle at high speed. One option is to turn around and GO!

> Consider using a **drive forward** block instead of a block that includes a specified amount of time to drive forward. This will allow the GiggleBot to drive forward and do other things (such as flash LED lights). We will need to tell the GiggleBot when to stop.

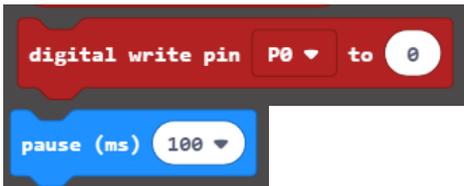
>If you use a **drive forward for __ ms** block, the GiggleBot will drive forward for the specified amount of time and then move on to the next action.

PROGRAMMING THE FLASHING LIGHTS

Next, we will program the GiggleBot's emergency vehicle lights to blink on and off. We need to turn on the LED connected to P0, turn it off, turn on the LED connected to P1, and then turn it off. Since it will take a long time to write that program over and over again so that the lights will continue flashing on and off, we will use a loop instead. A loop will tell the GiggleBot to do whatever is inside of the loop a certain number of times.



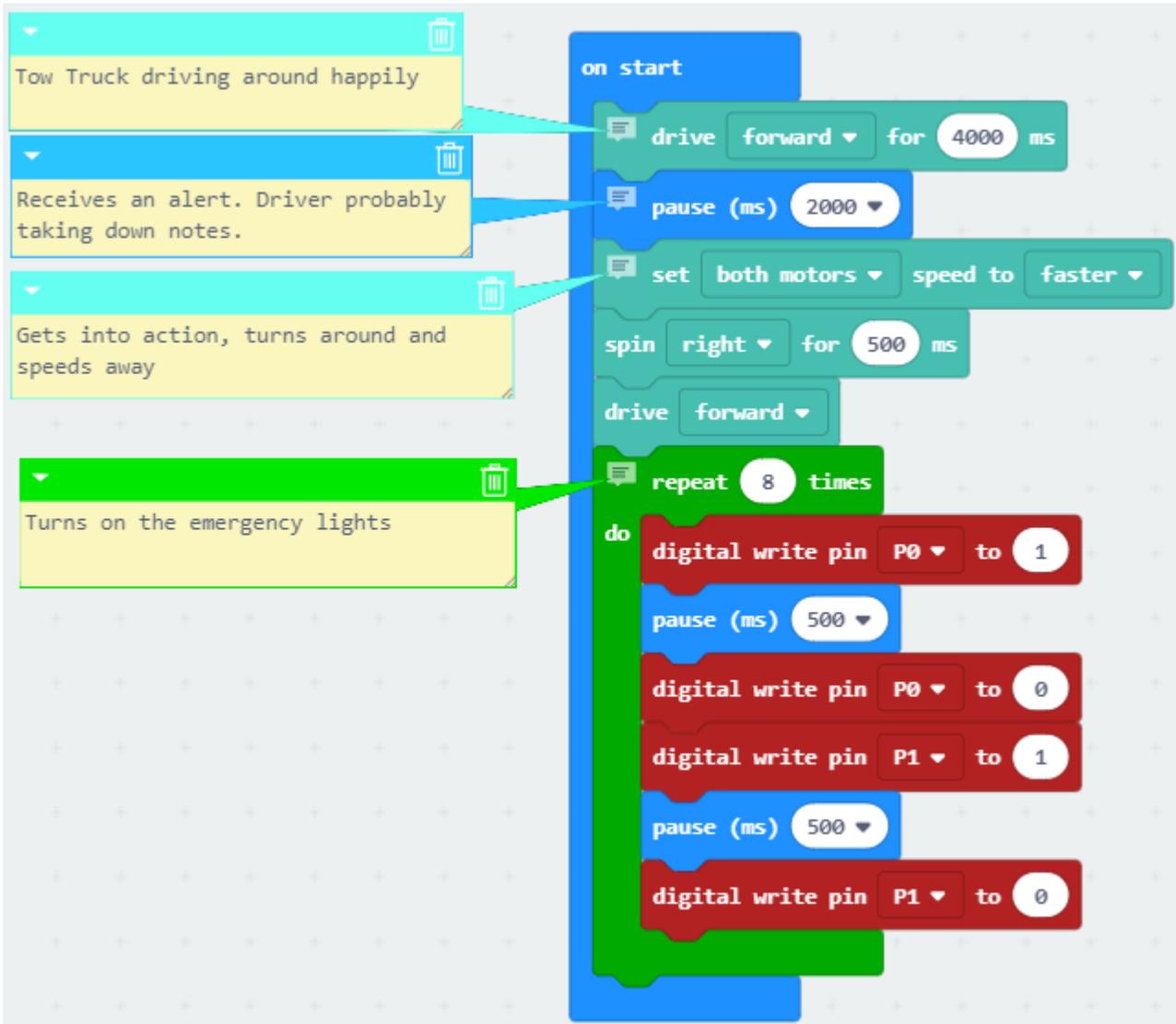
The **repeat 4 times** block can be found under **Loops**. Connect this block to the last movement block.



Inside of the loop, use **digital write pin __ to __** blocks (found under **Pins**) and **pause** blocks to program the LEDs connected to **P0** and **P1** to blink on and off. Remember:

- > When the value is **0**, the LED will be off
- > When the value is **1**, the LED will be on.
- > If you want the LEDs to blink faster, then choose a lower value for the **pause** block.
- > If you want the LEDs to blink slower, choose a higher value for the **pause** block.

Your code so far will look like this:



The code is organized as follows:

- on start** (blue block)
 - drive forward** (green block) for 4000 ms. Callout: "Tow Truck driving around happily"
 - pause (ms)** (blue block) 2000. Callout: "Receives an alert. Driver probably taking down notes."
 - set both motors** (green block) speed to faster. Callout: "Gets into action, turns around and speeds away"
 - spin right** (green block) for 500 ms
 - drive forward** (green block)
 - repeat 8 times** (green block)
 - do** (red block)
 - digital write pin P0** (red block) to 1
 - pause (ms)** (blue block) 500
 - digital write pin P0** (red block) to 0
 - digital write pin P1** (red block) to 1
 - pause (ms)** (blue block) 500
 - digital write pin P1** (red block) to 0

Callout for the repeat loop: "Turns on the emergency lights"

HELPING OUT THE BROKEN VEHICLE

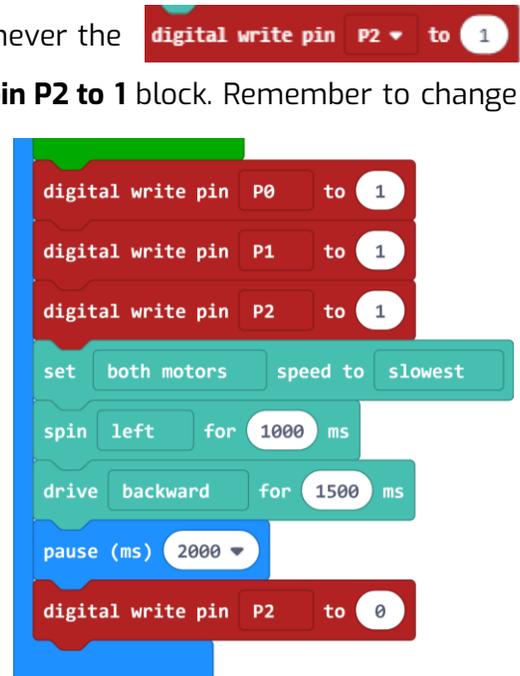
Now we will program the GiggleBot tow truck to turn around and back up to help out the broken down vehicle. First, decide if you want to keep the emergency vehicle lights on or off. In the program below, blocks were added to keep both LEDs turned on. Then, decide what your GiggleBot tow truck needs to do to help out the broken down vehicle. Does it need to slow down? Turn around? Back up?

Be sure to turn on the back up light connected to **P2** whenever the GiggleBot drives backwards by connecting a **digital write pin P2 to 1** block. Remember to change the pin to **P2** and the value to **1** for on and **0** for off

Add this code after the **repeat** loop.

The GiggleBot will:

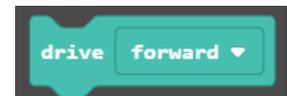
- > "Shift gears" to go slowly
- > Turn around
- > Back up towards the broken down vehicle.
- > Pause for 2 seconds (the driver is probably hooking up cables)
- > Turn the backup light off
- > Last, your GiggleBot tow truck needs to drive away towing the broken down vehicle (for real or you can imagine this part).



```

digital write pin P0 to 1
digital write pin P1 to 1
digital write pin P2 to 1
set both motors speed to slowest
spin left for 1000 ms
drive backward for 1500 ms
pause (ms) 2000
digital write pin P2 to 0

```

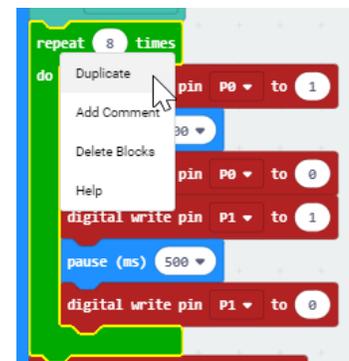


```

drive forward

```

Program the GiggleBot to drive forward once again flashing its emergency vehicle lights by using a **drive forward** block followed by a **loop** to control the LEDs. You can use the same loop as before if you would like to. Right click on the **repeat** block and click **duplicate**



```

repeat 8 times
do
Duplicate
Add Comment
Delete Blocks
Help
pin P0 to 1
pin P0 to 0
digital write pin P1 to 1
pause (ms) 500
digital write pin P1 to 0

```



Move the loop to the end of your program. After the loop, add a **stop** block to ensure that your GiggleBot does not drive forward forever.

And here is the full program!

Download and transfer the program to your GiggleBot. It is now a tow truck with working emergency vehicle lights and a backup light!

Can you identify the following sections?

- > The tow truck is driving normally
- > The tow truck is receiving a distress signal
- > The tow truck flies to the rescue, all lights blazing
- > The tow truck approaches the broken vehicle slowly
- > The tow trucks pretends to hook up the broken vehicle for towing
- > The tow truck tows the broken vehicle to a garage, with its lights flashing to warn other vehicles that it is currently towing a car.

```

on start
  drive forward for 4000 ms
  pause (ms) 2000
  set both motors speed to faster
  spin right for 500 ms
  drive forward
  repeat 8 times
  do
    digital write pin P0 to 1
    pause (ms) 500
    digital write pin P0 to 0
    digital write pin P1 to 1
    pause (ms) 500
    digital write pin P1 to 0
  do
    digital write pin P0 to 1
    digital write pin P1 to 1
    digital write pin P2 to 1
  set both motors speed to slowest
  spin left for 1000 ms
  drive backward for 1500 ms
  pause (ms) 2000
  digital write pin P2 to 0
  set both motors speed to normal
  drive forward
  repeat 8 times
  do
    digital write pin P0 to 1
    pause (ms) 500
    digital write pin P0 to 0
    digital write pin P1 to 1
    pause (ms) 500
    digital write pin P1 to 0
  stop
  
```

> TRY IT OUT

Be sure that all of the connections are correct. If you connect the alligator clips to the wrong leg of the LED, it will not light up. Run your program and see if your GiggleBot car and lights work correctly.

ITERATE

Did your GiggleBot car's lights turn on and off 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.

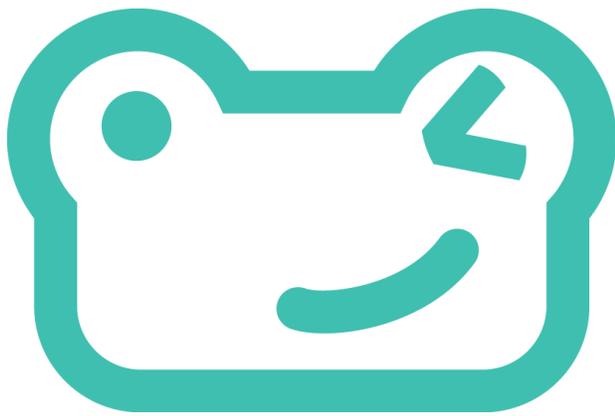
As you modify and revise your program, save each new program with a version number – you never know when you might want to take another look at an older version (for example: LEDs_V1 where “V” stands for version).

EXTENSION

What else can you add to your car?

- > Could you add a speaker for a back up signal?
- > Could you add a distance sensor and speaker so that your GiggleBot car makes a sound if it gets too close to something (or something gets too close to it)?
- > What other features do cars have that you could include?
- > What features do you want a car to have?

How could you turn your GiggleBot into a police car, fire truck, or ambulance using LEDs, a speaker, and servos?



Giggle Bot

Copyright Dexter Industries 2019. All rights reserved. Reproduction and distribution of the Mission without written permission of Dexter Industries is prohibited. GiggleBot is a registered Trademark of Dexter Industries.

Contact dexterred@dexterindustries.com for permissions and questions.