Driving mBot

New out of the box (or reloaded with the Default Program), mBot has three pre-programmed modes of operation. You can select one of these modes by pressing the **A**, **B**, or **C** button on the Infrared (IR) remote control. You can also step through the three modes by pressing the on-board button near the front of mBot.

Mode A–Manual Drive In this mode, you can drive the robot forward or backward and turn it right or left by pressing the arrow keys on the IR remote control. Numbered keys correspond to sounds. When you switch to Mode A, mBot's LEDs are white.

Mode B–Obstacle Avoidance In this mode, mBot will drive forward until it detects an obstacle such as a wall or chair via its ultrasonic range finder. Then, it will back up, turn, and resume driving forward. When you switch to Mode B, mBot's LEDs are green.

Mode C–Line Following In this mode, mBot will follow a black line while driving forward. It keeps track of the black line using its line-following sensor, mounted to the front of the mBot. When you switch to Mode B, mBot's LEDs are blue.

Let's start learning to program by looking at the code used for Mode A.

Understanding Mode A–Manual Drive

Infrared (IR) radiation is all around us. But what is it? IR is a form of electromagnetic radiation, just like visible light. However, because IR radiation has a lower frequency than visible light, humans are not able to see it.

Many household electronics (e.g., TVs, audio systems, and games) come with remote controls that use IR to send signals between a transmitter and a receiver. The remote control for mBot uses IR in the same way. mBot remote control contains a transmitter. When you press one of the buttons on the remote control, the infrared LED, located at one end of the remote control, sends out a series of IR pulses (on and off). mBot has an onboard IR receiver near the front. The microprocessor in mBot interprets the pulses of the IR transmitter and determines which button was pressed.

Let's try to write a simple program that will allow us to manually control mBot with the IR remote control. In this first program, which we'll call the Drive Forward program, we will try to make mBot move forward when the up arrow button on the remote control is pressed.

To get started, open the mBlock program on your computer or Chromebook. We will write code in a mode in mBlock that is called Arduino mode. Choose Arduino mode from the Edit menu. You will see a new pane in the mBlock window. On the left are the block and the palettes, where you find the raw material from which you will build your program. In the middle pane, you will build your program. In the right pane, mBlock translates your program into Arduino code. This is the code that will be loaded to mBot. For the most part, when we are coding for theses activities, we will work in the left and middle panes.

Now let's build the Drive Forward program in mBlock:

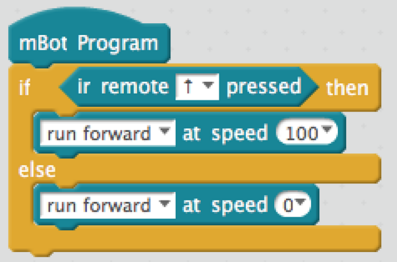

Figure 1 Drive Forward program

Notice that the pallets are color coded. The yellow blocks are from the Control pallet. Drag the "forever" and "if then else" blocks to the middle pane.

The blue-green blocks are from the Robots pallet. Drag the "mBot Program" block to the middle pane. (Note that this block has a rounded top; it is called a hat block. A hat block starts a program or a subroutine.) Add the "ir remote pressed" block and two copies of the "run forward" block to your program in the middle pane.

Move blocks around until you have all of the blocks in the proper positions. When you have the blocks in position, you may notice some things that need to be changed to match the program in Figure 1. Consider the "ir remote" block:

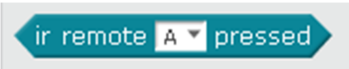

Figure 2 An "ir remote" block

By default, the "A" button is selected in the "ir remote" block. For the Drive Forward program, we want to control mBot with the "up arrow" button on the remote. So, choose the ↑ (up arrow) from the menu on the "ir remote" block.

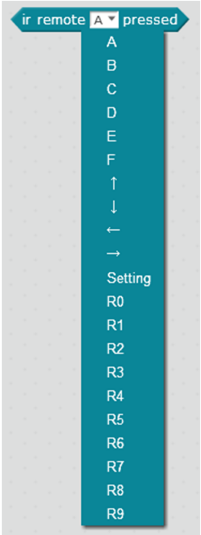

Figure 3 Buttons available in the "IR remote" block menu

Finally, take a moment to look at the two "run forward" blocks:

Figure 4 The "run forward" block

Each "run forward" block has two menus. One menu controls the speed and the other controls the direction. For the Drive Forward program, increase the speed of the first "run forward" block; leave the other menus with the default settings.

TRY IT OUT

Once you have built the Drive Forward program, and you have set the menus properly, try out the program with your mBot:

1. Connect mBot to the computer or Chromebook with a USB cable (included with mBot).
2. Turn on mBot.
3. Choose Serial Port from the Connect menu and select the port to which you've connected mBot. This is usually the last one on the list.
4. To upload the program, click the Upload to Arduino button in the right pane of mBlock.
5. Wait for the program to upload. If the program uploads successfully, you will see a message telling you the upload is complete. The program will start running as soon as the program has uploaded.
6. Disconnect mBot from the USB cable and try the program. Use the IR remote. Does mBot move forward on command?

Now, let's add some functionality to our program so we can make mBot go both forward and backward. We'll call this the Forward and Backward program. To build the Forward and Backward program, we need to add some blocks:

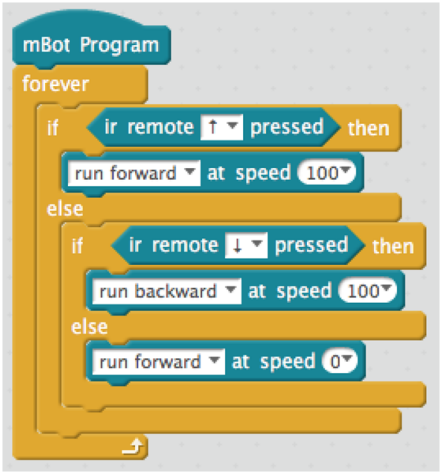

Figure 5 Forward and Backward program

Notice the structure of the Forward and Backward program. If the up arrow button is pressed on the IR remote control, mBot moves forward. If the up arrow button is continually pressed (if it is held down), mBot will continue to move forward. If the button is released, mBot stops moving and the program checks to see if the down arrow button is pressed. If the down arrow button has been pressed, the mBot will move backward. If neither button is pressed, the mBot runs forward at speed 0 (i.e., mBot doesn't move).

TRY IT OUT

Upload the Forward and Backward program (Figure 5) to mBot. Are you able to use the up arrow and down arrow buttons to make mBot move forward and backward?

When you have the program working, add a few more blocks to build the Four-Directional Movement program (see Figure 6). This more complicated program will allow you to have four-directional control of your mBot.

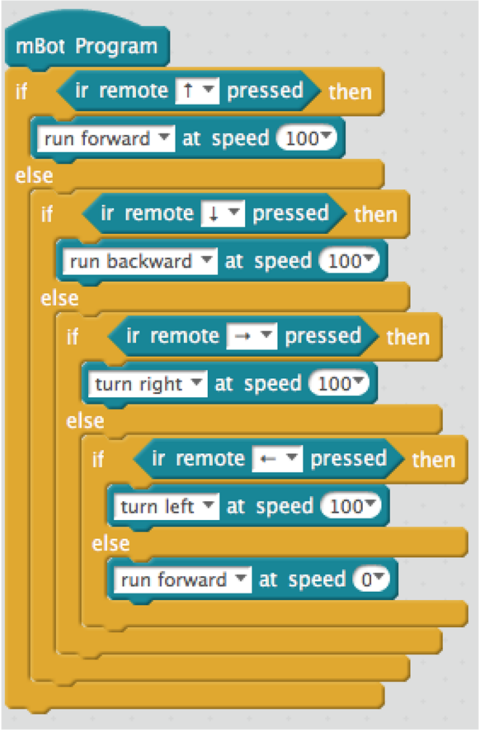

Figure 6 Four-Directional Movement program

TRY IT OUT

Upload the Four-Directional Movement program (Figure 6) to mBot. See if you can move mBot in all four directions using the buttons on the IR remote control.

Improving Manual Drive with Variables

As you work with mBot, you might want to make it move more quickly or slowly. If you were to make this adjustment in the Four-Directional Movement program; you would have to change the setting in four different places. However, you can write a more elegant program, and one that is more easily modified, if you include a variable that can be used to control speed.

In mBlock, select the Data & Blocks palette, and then click the Make a Variable button. Enter a name for your variable, select For all sprites, and click OK.

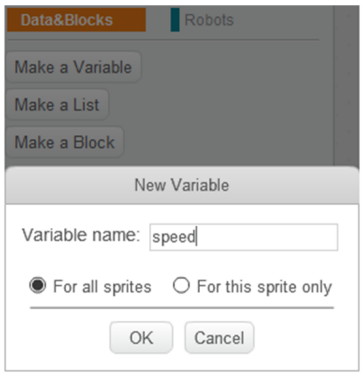

Figure 7 Creating a new variable

Once you click OK, you will have access to several different commands regarding your new variable, including the ability to set its value.

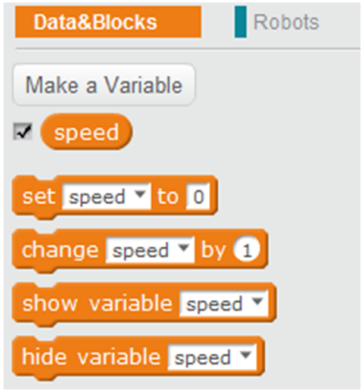

Figure 8 Commands available when defining a variable

After you have created a speed variable, you can add it to your program.

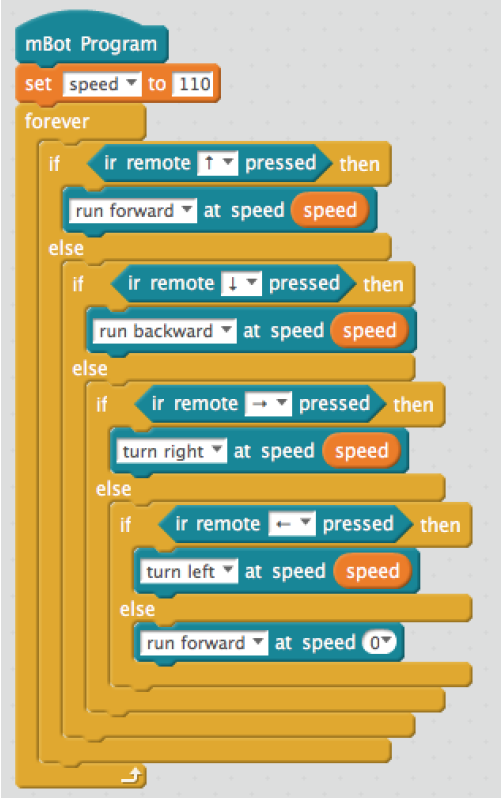

Figure 9 Four Directions and Speed program

TRY IT OUT

In mBlock, create the Four Directions and Speed program (Figure 9), and upload the program to mBot. Does mBot drive like it did before? Change your value for the "speed" variable and upload your updated program to mBot. Does mBot drive faster? How low can you set the "speed" variable and still get mBot to move?

Tip The motor speed can be in the range of –255 to 255. If you set the speed above 255, the value will default to 255. If you set the speed below –255, the value will default to –255.

Add Comments to Your mBot Program

As a programmer, it is important to frequently add comments to your code. When you are writing a program, you think you know exactly what is going on, and it may feel as though comments are unnecessary. However, if you close the program and do not think about it for a few weeks, you may not remember the situation as well as you do now. Also, you may want to share your program with other people so they can use it, and you want them to be able to understand your code. Comments allow other people to interpret your program and be able to build upon it.

mBlock makes it easy to add comments to you program. To create a comment, right-click (Windows or ChromeOS) or CTRL-click (macOS) a block and choose "add comment".

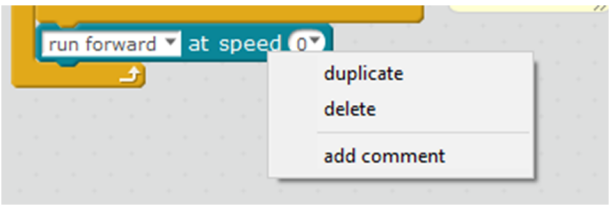

Figure 10 Adding comments in mBlock

Here is our program with some comments:

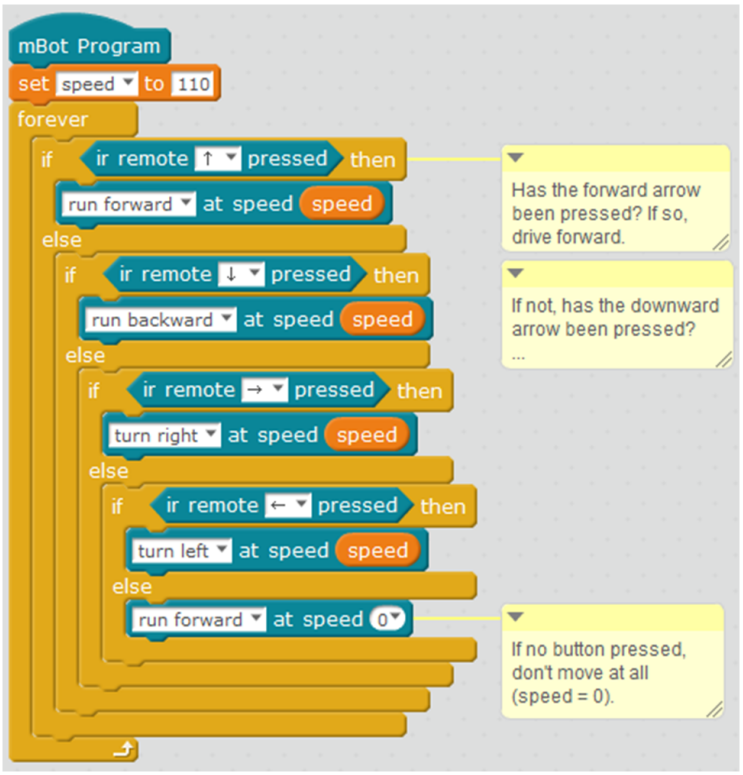

Figure 11

Comments become increasingly helpful as your programs become more and more complex. Know that it takes time to learn how to write comments that will be useful to you and others. Practice adding comments to the programs you write for these activities so you can do it more easily and effectively in the future.

Note When you upload a program to mBot, comments do not impact how the program runs, but they are saved with the program, so they will always be available in mBlock.

Using Lights on mBot

If you watch the mBot operating in Mode A of the Default Program, you may notice that there is more going on than we have incorporated in our program. One significant difference is that the

LEDs on the front of mBot light up as follows:

Motion	LED color and state
Forward	Both LEDs are green.
Backward	Both LEDs are red.
Left	The left LED is green.
Right	The right LED is green.

In the Robots palette of mBlock, there is a block for controlling the LEDs on mBot, "set led on board":

Figure 12 The "set led on board" block from the Robots palette

Use the first menu to select which LED(s) you want to control (all, led left, or led right) and the other menus to set the *intensity*, or brightness, of the three RGB (red, green, blue) components of light from the LED. The range for each component is 0 to 255.

Let's add the LED behavior to our program to create our final program for this activity. We'll call this the Manual Drive program.

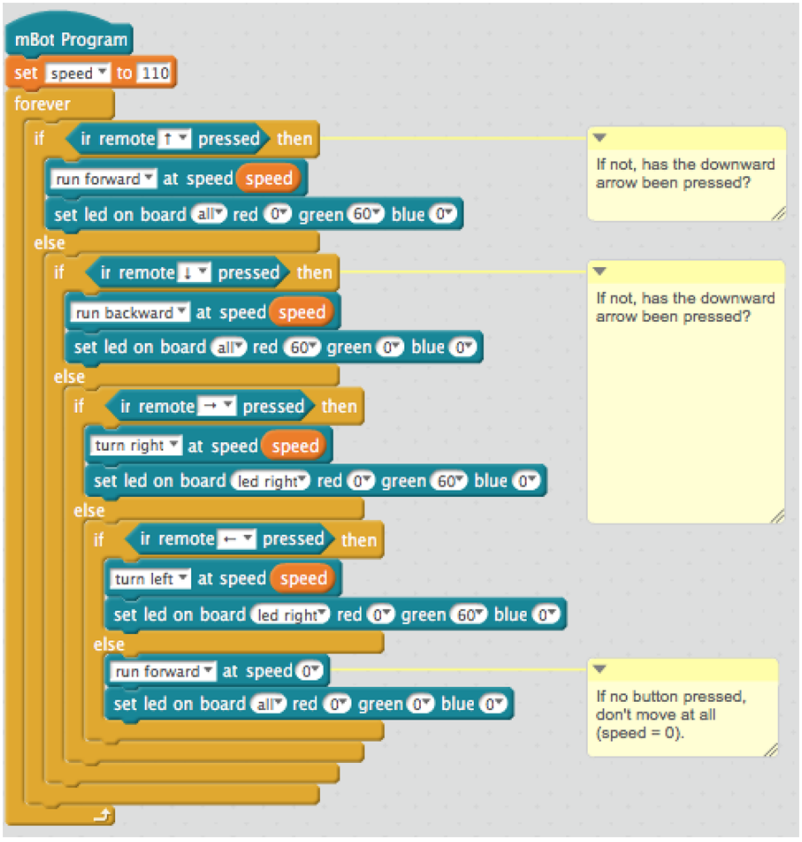

Figure 13 Manual Drive program with coordinated on-board lights

TRY IT OUT

Try out the Manual Drive program. Are you able to control the LEDs on your mBot? How similar is the program to Mode A of the Default Program?

Challenge Extensions

Extension 1 Change Speed with Numeric Buttons

Many vehicles, such as Segways and drones, have slow, medium, and fast speed settings; some hobby cars have a turbo speed setting. The program we've been writing specifies a constant speed of 110. Modify the Manual Drive program to allow the operator to choose one of three different speeds before driving. Be careful that you don't set the speed too low or mBot may not move at all.

Extension 2 Backup Warning

Most construction vehicles make a warning sound when they move backward. Can you add a back-up warning to the Manual Drive program? mBot has a small buzzer located near the front. You can use this buzzer to make sound with the "play tone" block in the Robots palette:

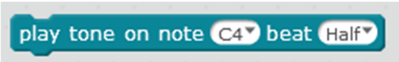

Figure 14 The "play tone" block

The sound will not be very loud, but you have a lot of control over the frequency and duration:

- Select a note from C2 (low) to D8 (high)
- Set the duration to any of the following options:
  - Zero (no sound)
  - Eighth (plays sound for 1/8 second)
  - Quarter (plays sound for 1/4 second)
  - Half (plays sound for 1/2 second)
  - Whole (plays sound for 1 second)
  - Double (plays sound for 2 seconds)

Experiment with the different sounds until you are able to make a sound that is similar to what you hear when a vehicle is backing up. Add the sound to the Manual Drive program so that the sound plays when mBot is moving backward.