

2

Baseball batter

“Take me out to the ball game . . .”
—J. Norworth and A. Von Tilzer

The simple contraption you’re about to build is a baseball -playing robot. This project will introduce you to the three stages of any robotic system: input, processing, and output.

Input is data from the world outside the robot. In this case, the input comes through the LEGO sensors, which will

detect the ball when it comes close. *Processing* happens when we take that input and do something with it. Processing happens inside the Hub, which will run a program that we’ll write. The program will tell the robot to wait for the ball, then swing the bat.

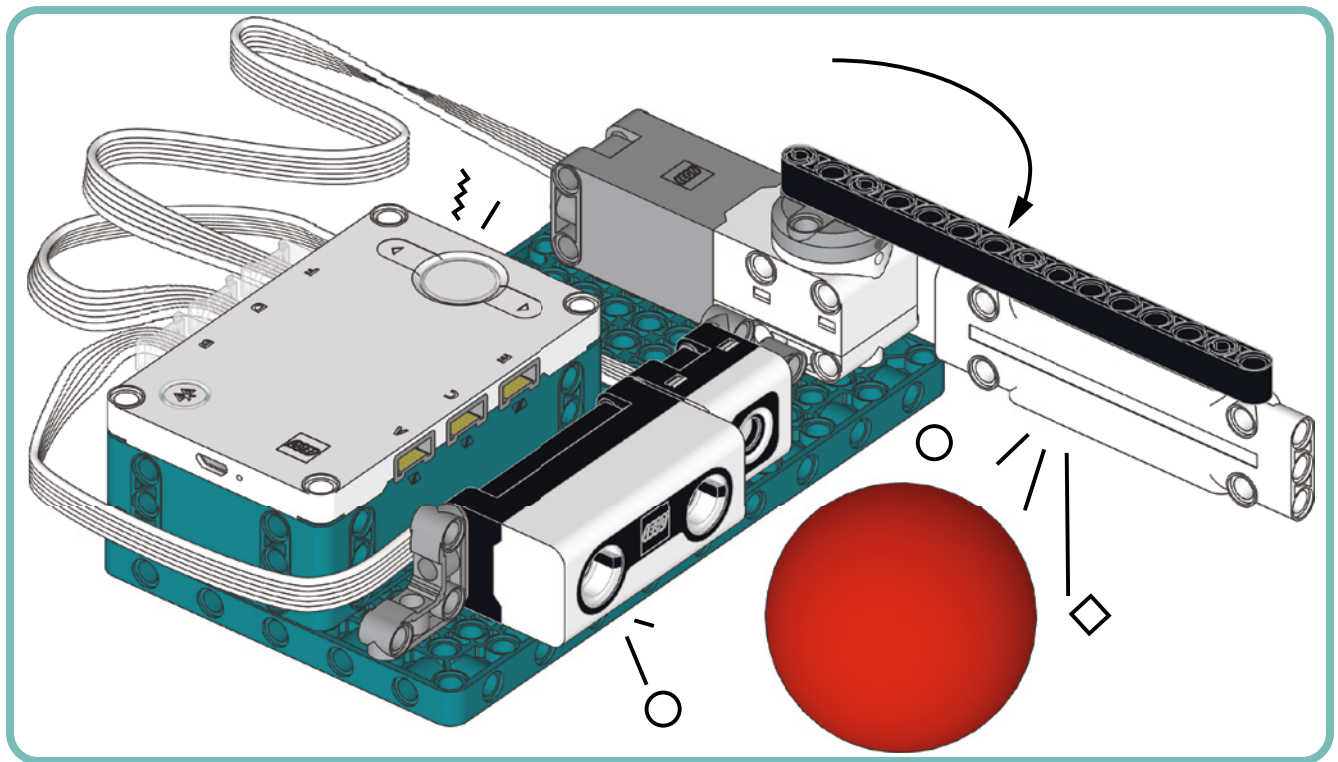


Figure 2-1: The Baseball Batter is a simple contraption that will help you take the first steps in robot programming.

Output is what the robot does as a result of the processing. Our output will be an action: the motor's swinging the bat to hit the ball. We can make the robot even more fun by adding other outputs, like playing a sound effect and showing a nice animation on the Hub's display.

building the baseball batter

In this section, you'll find step-by-step instructions for building the Baseball Batter. For each step, the first box shows the parts you'll need for that step.

Throughout the book, sometimes you'll see a number inside a circle (for axles or panels) or inside a square (for beams) next to a part. This number tells you the part's length in LEGO units, or *modules*. You can measure a beam's length by counting its holes. You can measure axles by putting them next to a beam and counting the holes on the beam. For example, Figure 2-2 shows a 5-module (5M) axle next to a 9M beam.

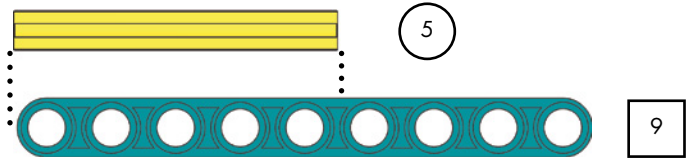
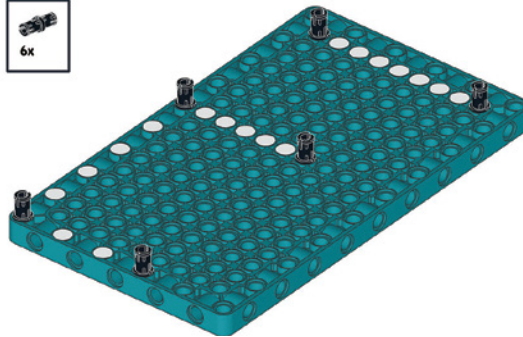


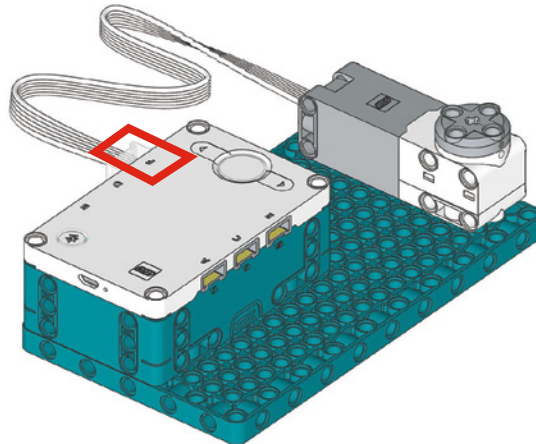
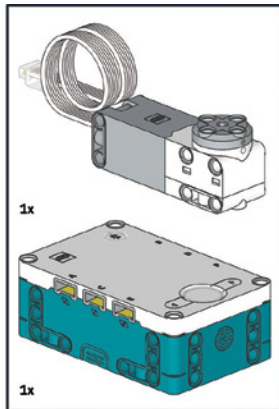
Figure 2-2: You can measure a beam's length by counting its holes. To find the length of an axle, place it next to a beam and count the holes on the beam.

The white dots help you put the black Technic pins in the right holes. The teal Technic baseplate is not shown in the parts list to save space.

1

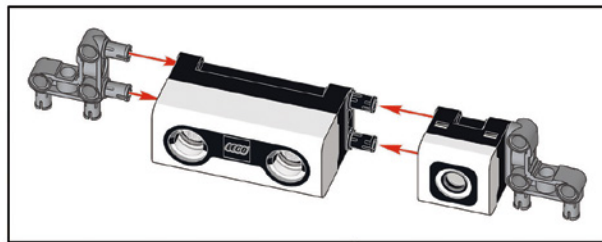
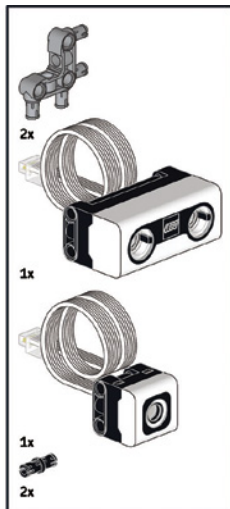


2

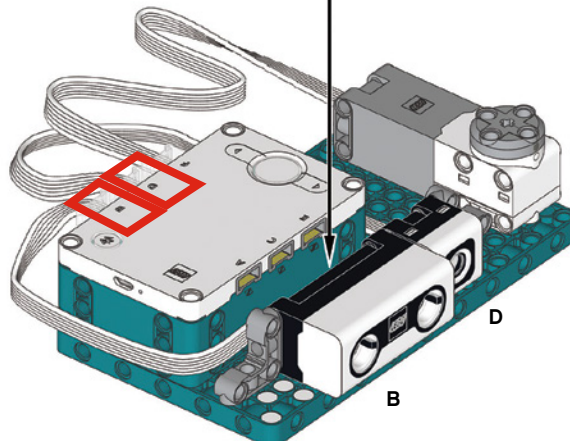


Attach the motor to port F of the Hub.

3

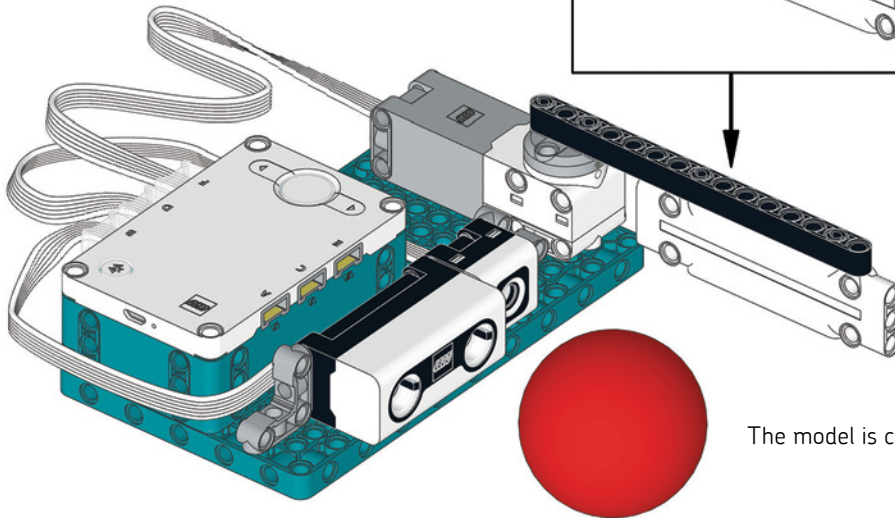
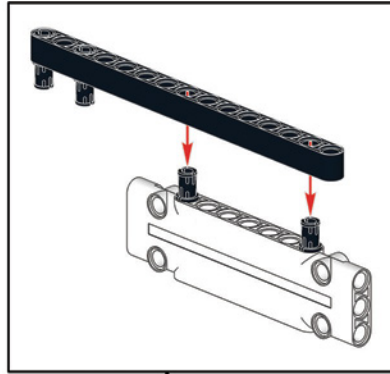
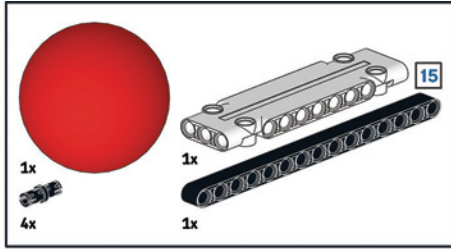


When you see a submodel in its own box like this, always assemble it *before* attaching it to the main model.



Attach the Distance Sensor to port B and the Color Sensor to port D.

4



The model is complete!

programming the batter

Now that you've built the robot, it's time to tell it what to do. In other words, you need to program it. You might know programming by another name: *coding*. When you program, you're writing code with instructions that a machine like a robot can understand. I'll guide you step-by-step through the process of programming the Baseball Batter so you can see how to write code in the MINDSTORMS App. Remember, you can also download all the code used in this book at URL.

NOTE If you're an expert MINDSTORMS App user, you can just take a look at the complete program in Figure 2-5 before going straight to the section "Understanding the Program."

getting around the app

Launch the LEGO MINDSTORMS App by double-clicking or tapping its icon, depending on the device you're using. After a few moments, the lobby screen should show up. On a computer, it should look like Figure 2-3. On smaller screens, you'll only see one robot at a time. Either way, you should see four buttons at the bottom of the screen, and one at the right top of the screen. The Home button brings you to the lobby, Community shows a list of inventions made by LEGO fans, Projects shows you a list of the projects you made, and Code brings you straight to the programming environment. The Settings button, the gear shape at the top-right corner of the screen, shows the settings and help.

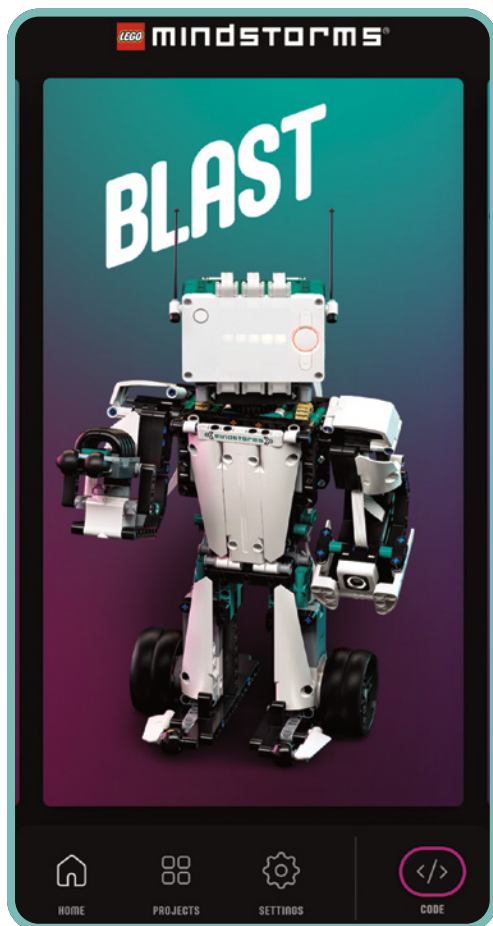


Figure 2-3: The LEGO MINDSTORMS App lobby. From here, you can start the activities for the five robots included in the box, manage your projects and settings, or start a new program. The app's look may vary depending on the device it's running on.


Click the **Code** button at the bottom right of the screen to open the programming area. It should look like Figure 2-4.

the word blocks


The LEGO MINDSTORMS App lets you program your robots using Scratch 3.0, a block-based visual programming language. On the left of your screen, you'll see all the *word blocks*, the building blocks of your programs. Each word block represents one instruction for your robot to carry out. The blocks are grouped into *palettes* based on their function. For example, one palette has blocks just for controlling the motors, another has blocks for the sensors, and so on.

You can drag blocks from the palette to the programming area on the right. By combining blocks into stacks, you can build programs that tell your robot to do all sorts of things. Each stack of blocks reads from top to bottom.

connecting to the hub

To control your robot, you must connect your computer, phone, or tablet to the Hub. You can use a USB cable or Bluetooth. When the Hub is connected to your computer via USB, the app finds the Hub automatically. If you want to connect via Bluetooth, click the **Open Hub Connection** button  and follow the onscreen instructions.

NOTE If you have trouble connecting to the Hub through Bluetooth, see the official LEGO support page: <https://education.lego.com/en-us/product-resources/spike-prime/troubleshooting/bluetooth-connectivity/>. This page refers to SPIKE Prime, a different LEGO robotics kit. The SPIKE Prime Hub is exactly the same as the Robot Inventor Hub, except it's yellow.

Once the Hub is connected, the dot on the Hub icon at the top right of the app turns green, and real-time sensor readings appear beside that icon.  These readings tell you exactly what each sensor is picking up at any given moment. Try moving your hand back and forth in front of the Distance Sensor and you should see the numbers change on the screen.

creating a program from scratch

When you open the programming area or create a new project, you should already see one block on the programming canvas: when program starts. This is an example of a *hat block*. Like a hat, a hat block goes on top of other blocks. Every block stack must start with a hat block, which tells the robot when to run that stack. For now, you can find all the hat blocks in the yellow Events palette. (Later, when we create a remote controller, more hat blocks will appear in the Remote Control palette.)

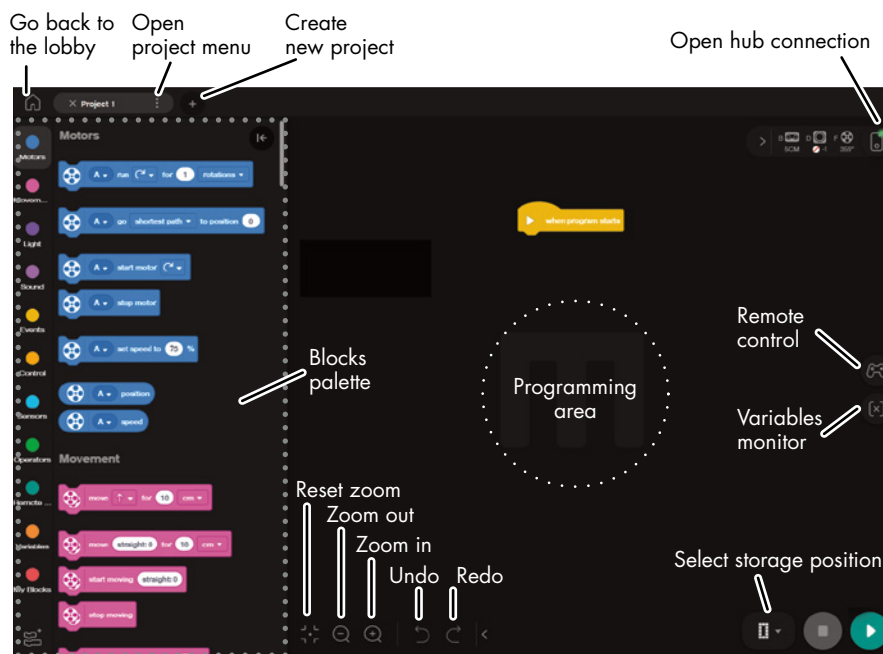


Figure 2-4: The programming area shown on a laptop screen. The LEGO MINDSTORMS App lets you create programs by stacking blocks, just like you stack LEGO parts.

Any blocks attached to when `program starts` will automatically run as soon as the program begins. You can have more than one when `program starts` block in your program. In that case, each of those stacks will start at the same time.

Let's create the program for the Baseball Batter step-by-step. All we want the robot to do is swing the bat when it sees the ball coming. Figure 2-5 shows the complete program. We'll talk through building it one block at a time.



Figure 2-5: The complete program for the Baseball Batter uses the color sensor to detect the ball.

1. Open the **Motors** palette, drag the **set motor speed** block to the programming canvas and attach it under the yellow hat block. Notice how the block snaps into place even if you just drop it near the yellow block.

This block doesn't actually turn any motors on. It just tells them how fast to go once they're turned on. You can choose any speed from -100% to 100% . Negative numbers make the motors spin backward. Without this block, the motors will try to run at 75% speed by default.

If only one motor is attached to the Hub and the Hub is connected to the app, the `set motor speed` block should already show the port the motor is attached to. If you don't already see port `F` selected on the block, choose it using the port selector drop-down menu, as shown in Figure 2-6. Then set the speed to **100%**. Now the motor should run at full speed when the program is run.

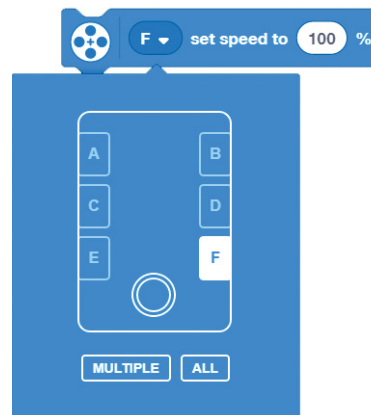


Figure 2-6: Each motor block has a port input menu where you can specify which port the motor you want to control is attached to. You can select a single port or multiple ports.

2. From the same palette, drag and drop the **motor go to position** block below the previous blue block and set the port to `F`. This block tells the motor to spin until the mark on the shaft reaches a certain angle relative to the round mark on the motor body. An angle of `0` aligns with the mark. Angles then increase clockwise, up to `359`.

To reach the specified angle, you can tell the motor to spin clockwise, counterclockwise, or whichever direction makes the shortest path, as shown in Figure 2-7. In our case, choose **shortest path** and set the position to `0`. We do this because it's possible for the motor shaft to move away from the zero-position while the program isn't running. Setting the position to `0` at the start of the program puts the LEGO Technic beam into the best position for hitting the ball.

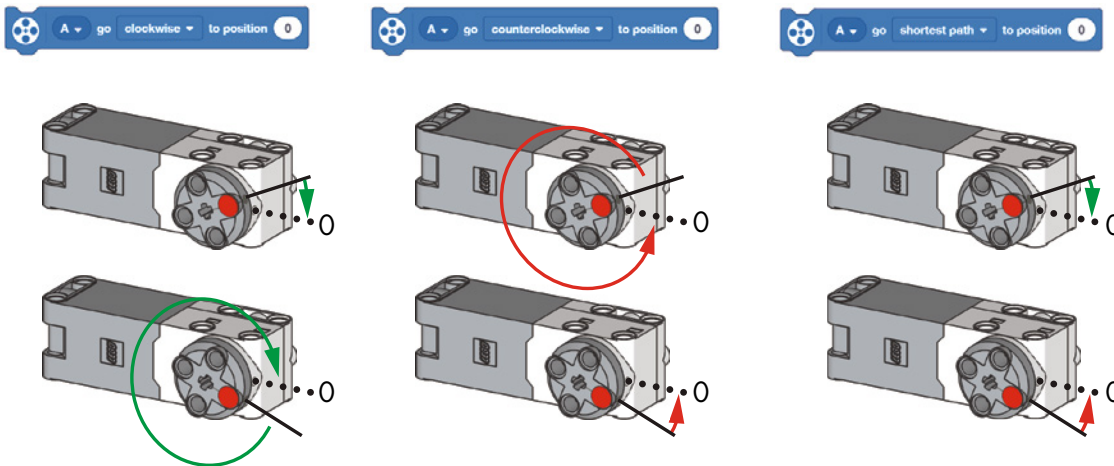


Figure 2-7: The motor *go to position* block spins the motor until it reaches a certain position. You can make it spin clockwise or counterclockwise, or whichever way is shorter.

- Now let's make the robot sense the ball. From the **Events** palette, drag and drop the *when color is* block. Set the port letter to **D** and the color to **red** (Figure 2-8).

The block *when color is* is another hat block. It will run the stack of blocks beneath it when the Color Sensor detects a certain color. As you can see in Figure 2-8, the sensor can look for several colors besides red. When the sensor sees the right color, it only runs the block stack once. If the sensor still sees the same color when the stack is finished, it won't run the code again.

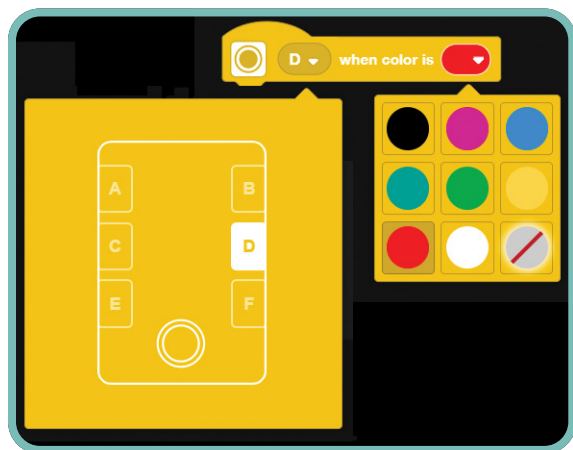


Figure 2-8: Setting the *when color is* hat block to look for red with the Color Sensor attached to port D

- From the **Sound** palette, drag and drop a *start sound* block under the *when color is* hat block. Then, following the numbered steps in Figure 2-9, choose the sound **Hit** from the **Tricky** tab of the Sound Library.

This block starts playing a sound of your choice and lets the program continue running as the sound plays. The app has a library of sounds, including Hit, already stored in the Hub. These will play from the Hub speaker. You can also select sounds from your own device or even record new sounds, but these will play from your device's speakers, not from the Hub.

- Add another *motor go to position* block below the sound block. Set the port to **F**, choose *shortest path*, and set the position to **40**. This block will make the motor rotate to an angle 40 degrees clockwise from the zero-position, moving the bat to hit the ball.
- Add a *start animation* block from the **Lights** palette and follow the steps in Figure 2-10 to select the **Celebrate** animation from the Animation Library. This block starts an animation on the Hub's display. The program will continue to run while the animation plays.

The *start animation* block also lets you design your own animations with the Animation Editor. Each animation is a sequence of patterns on the Hub's 5x5 grid of lights. You can choose which lights to light up and how bright they should be.

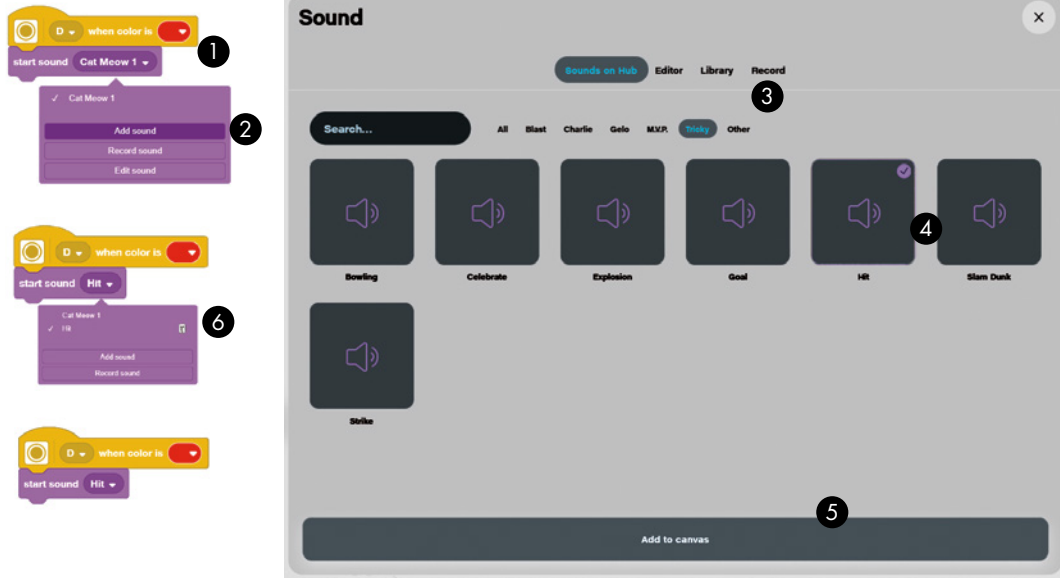


Figure 2-9: The `start sound` block plays sounds stored on the Hub or on your device.

7. Finally, add another `motor go to position` block below the animation block. Set the port to `F` and the position to `0`. Then choose `shortest path`. This block will bring the motor back to the zero-position, ready to hit the next ball you throw.

saving and running the program

Now you're ready to save the program and try it out! Here's how:

1. In case your Hub went to sleep while you created the program, turn it on again by pressing the round button and reconnect it to the app.

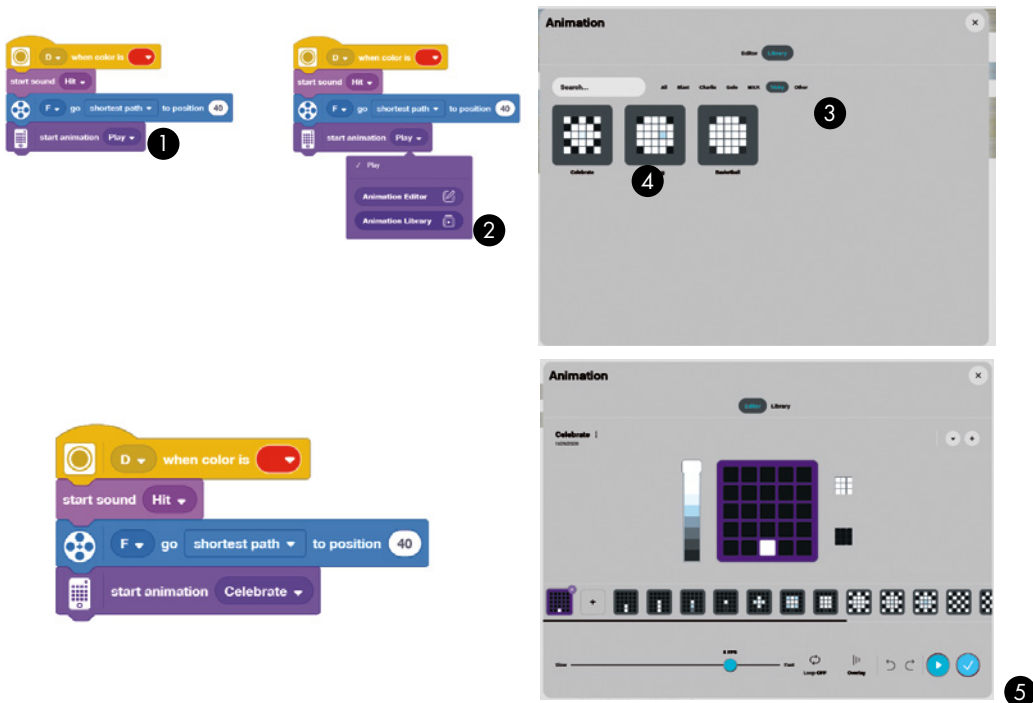


Figure 2-10: The `start animation` block plays animations on the Hub's display.

- Using the **Project** menu, rename the project `baseball_batter`.
- To run the program, click the **Select Storage Position** button in the bottom right corner of the programming area, beside the gray Stop button. Select **Download** mode, choose program slot number **0**, and click the **Play** button, as shown in Figure 2-12. The app will download and start the program on the Hub.



Figure 2-11: Select Rename Project from the Project menu.

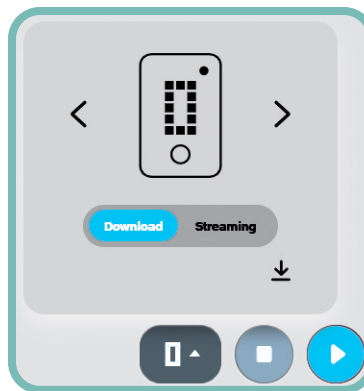



Figure 2-12: Select Download mode by clicking the Select Storage Position button before running your program.

The program will now be visible in the Hub Program view, as shown in Figure 2-13. You can get there by clicking the Hub Monitor button at the top right of the screen (see Figure 2-4).

- Throw the ball at the robot and see how it reacts. Was it a strike, a ball, or a home run? How far from the Color Sensor can the ball be and still be detected?

- To stop the program, press the round button on the Hub or press the **Stop** button  in the app.

understanding the program

The cool thing about word blocks is that they let you read your programs like you would read ordinary sentences. Here's how

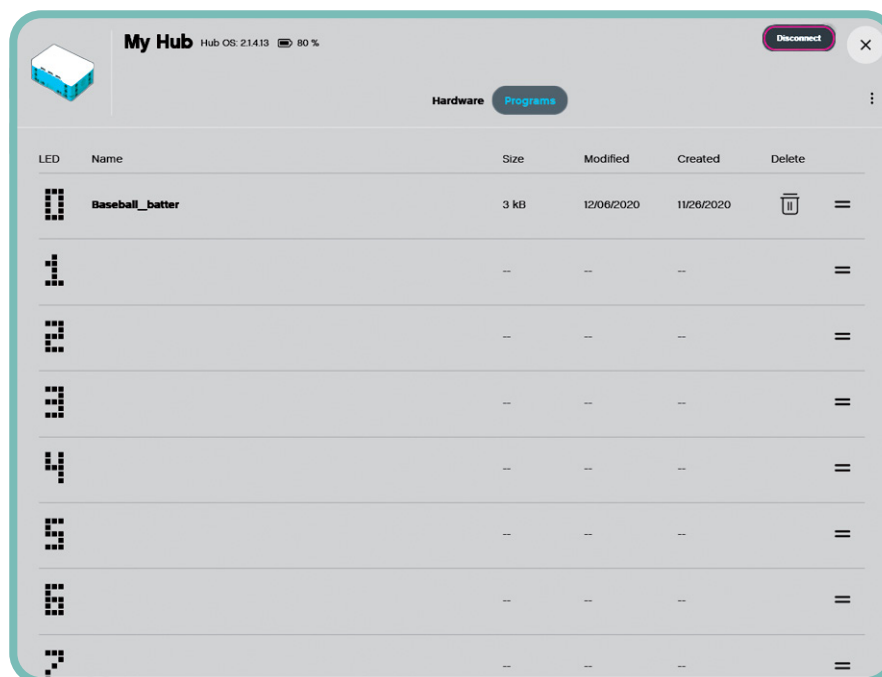


Figure 2-13: The Hub Program menu, accessible from the Hub Monitor button, shows the programs stored on your Hub and lets you reorder and delete them.

you might read the stack attached to the when program starts hat block. Each line represents one of the word blocks.

When the program starts,
set motor F's speed to 100% and then
tell motor F to follow the shortest path to position 0.

Here's how to read the other stack in our program:

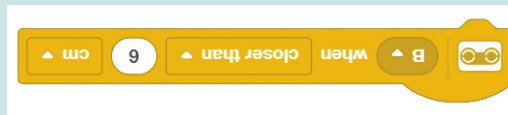
When the Color Sensor attached to port D reads RED,
start the Hit sound,
tell motor F to follow the shortest path to position 40,
start the Celebrate animation, and
tell motor F to follow the shortest path to position 0.

The first stack runs only once at the beginning of the program, while the other stack is triggered every time the Color Sensor detects red. In other words, detecting the color red is the *event* that triggers the sequence of actions.

NOTE When a program is written with ordinary words, it's called *pseudocode*. Pseudocode is a really important programming tool. In fact, before you create an actual program for a robot, it's important to plan out what you want your program to do using pseudocode. Once you have a plan, you're ready to start writing your actual code.

EXERCISE 2-1

Try using the Distance Sensor instead of the Color Sensor to detect the ball. Which block should you change? What should you change it to? Does the robot work better or worse than before?



ANSWER KEY: Replace the when color is hat block with another block from the Events palette: when closer than. Set the port to B, the threshold to 6, and the units to cm.

what you've learned

We've covered a lot in this chapter! By building and programming the Baseball Batter, you learned the key steps in any robotic system: reading sensor inputs, processing them, and producing outputs. You learned how to use event blocks, how motor blocks can rotate a motor to a precise angle, how to play sounds from the Hub, and how to show animations on the Hub display.

In the next chapter, you'll build and program the Gobbler, a robot that can help you cope with bad thoughts and answer the important questions of life.

