CHAPTER 4

Get Movin'

I would say that the MOVE block is probably one of the most important blocks when it comes to programming a robot. Without the MOVE block, you can still build bots, but they won't be able to do much. They can sit on a desk or table (just like SPOT), but they're not going to be very exciting to watch. Any robot that you design that uses one or more motors will use the MOVE block.

So, let's go over this very important block and see what it can do.

The MOVE block

Open your Mindstorms NXT software, and drag and drop a MOVE block from the Common Palette onto the beam. The configuration panel will appear in the lower-left corner of the screen (see Figure 4-1).

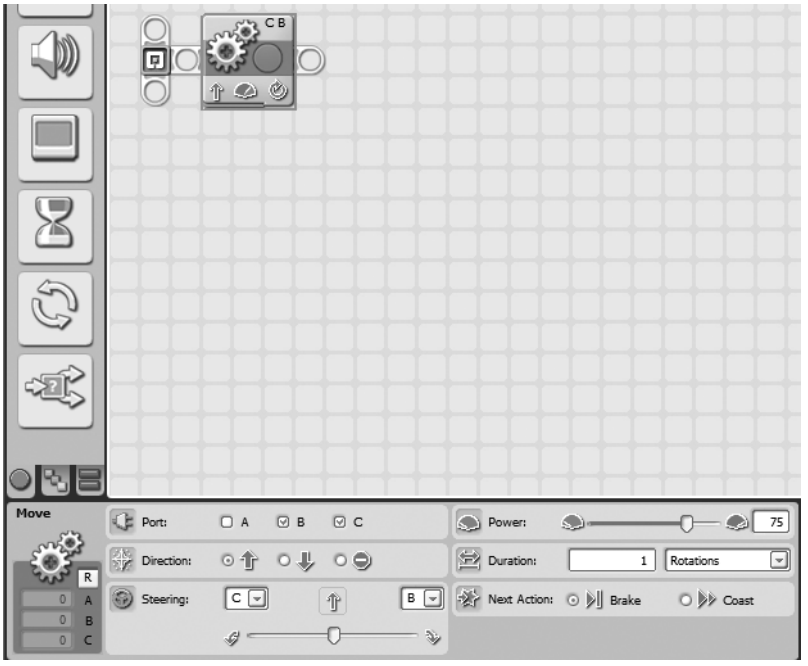

Figure 4-1. *The MOVE block and its configuration panel*

Your Brick has three ports for motors: Port A, Port B, and Port C. You must plug motors into these ports in order for them to work properly. Motors have numerous options including how fast they spin (Power) and how long or far they spin (Duration). By default, a `MOVE` block is configured to control Port B and Port C on the Brick. The other defaults include a Power setting of 75 and a Duration of 1 rotation, and the Next Action is set to Brake.

Moving Forward and Backward

Before I move on, I want to bring to your attention the subject of motor spin direction. Take a look at Figure 4-2. It shows a motor in two different orientations.

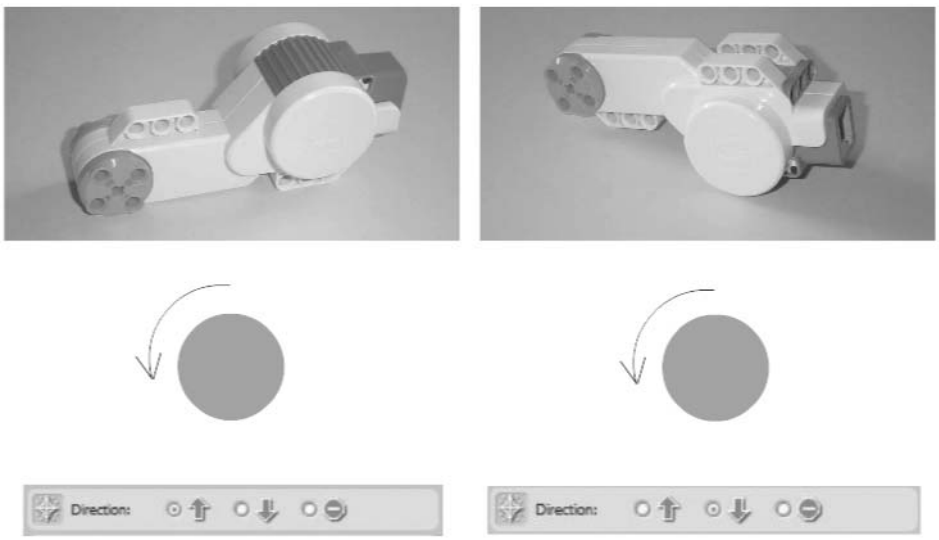

Figure 4-2. *A MOVE block can configure a motor to spin in two different directions (note the direction settings for each).*

All motors can spin forward and backward. But you need to be careful when describing a motor as "spinning forward" or "spinning backward," because the orientation of the motor also needs to be described. When we program, we have to take into consideration the orientation of the motor.

In Figure 4-2, the motor on the left has the up arrow selected in the Direction section on the `MOVE` block configuration panel. This up arrow corresponds to `FORWARD`, and this motor spins in the direction shown, counterclockwise. Now, if I flip this motor over (like the motor on the right side of Figure 4-2) but don't change the Direction arrow selection, the motor will spin in the opposite direction (clockwise). I have to change the Direction arrow to down (or `REVERSE`) to make the motor spin counterclockwise when it's position like the motor on the right.

Be sure to keep this in mind when building and programming your bots. Depending on whether you want your bot to move forward or backward, you'll have to select the proper Direction arrow (up or down) based on how the motors are attached to the bot.

Okay, now let's go over the rest of the `MOVE` block configuration panel.

First, we'll cover the ports. The MOVE block can control Ports A, B, and C and any motors attached to those ports. There is no rule for the way in which you connect motors to ports, but I would recommend that you decide on a "standard" method for connecting motors and stick with it for all robots you build.

As an example, look at Figure 4-3. When I build a robot I always connect the motor on the left side of the Brick (with the Brick's sensor ports on the bottom and motor ports on top) to Port B and the motor on the right side of the Brick to Port C. I always use Port A for my third motor. If I build a bot that doesn't need a third motor, Port A is always left open.

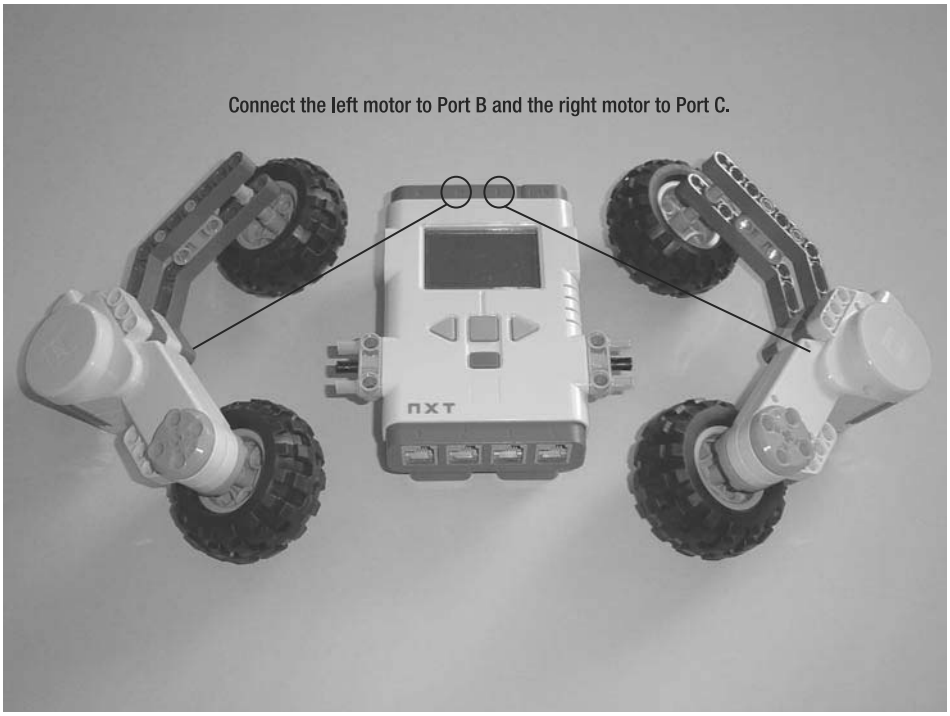

Figure 4-3. *Pick a method for connecting your motors to your ports, and always use it.*

When you connect a motor to a port, you must make certain to check the box for that Port in the MOVE block's configuration panel (see Figure 4-4). In Figure 4-4, you'll also notice that the motor ports you select (in this example, Ports A and B) are listed on the MOVE block in the upper right corner. This can be helpful for troubleshooting and to remind you which motors will be used.

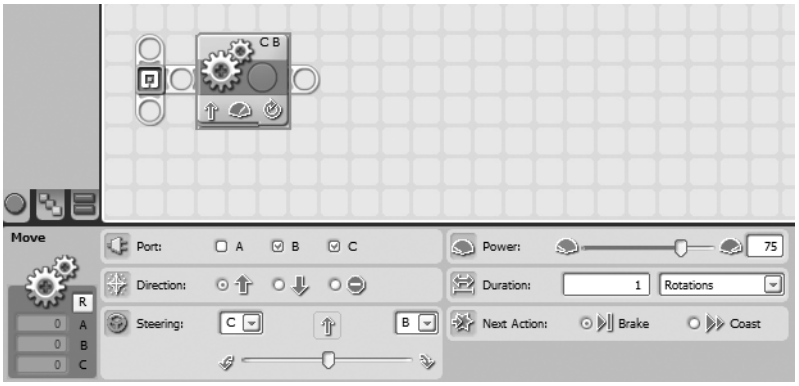

Figure 4-4. *Use the MOVE block's configuration panel to select motor ports.*

Stopping

I've already mentioned the Direction control using the up and down arrows (or `FORWARD` and `REVERSE`). The other option shown in Figure 4-5 is `STOP`. If you select the `STOP` option, check the Port boxes for the motor(s) you want to stop.

Figure 4-5. *Select the STOP option in the MOVE configuration panel to stop selected motors.*

Steering

The next item to discuss is steering. Figure 4-6 shows the Steering section that is available if you have selected either `FORWARD` or `REVERSE` for the Direction and have two motor ports checked.

■ **Note** If you select all three motor ports, the Steering control is turned off.

Figure 4-6. *The MOVE block's Steering control in the configuration panel*

The Steering control can be very useful if you know how to use it. Depending on its setting, you can configure a bot to move in a small or large circle or just spin in place.

If you have two motors configured for your bot's movement (Ports B and C, for example), you can make your bot spin in place by dragging the Steering control all the way to the left or right (the direction you drag the control will determine if the bot spins clockwise or counterclockwise). Try it! Drag the slider all to the way to the left. Save your program, upload it to your bot, and run the program. Which direction did the bot spin? Now let's change it. Drag the slider all the way to the right. Save, upload, and run the program again. Did the bot spin in the opposite direction?

You can also program your bot to drive in a circle; the size of the circle depends on how far you drag the Steering control left or right: dragging it closer to either the far right or far left will make the circle smaller. You'll have to play around with the Steering control to get the size of the circle just the way you want it. Go ahead and try this, too. Drag the Steering slider to the left but not all the way. Upload the program, and run it. Did the bot move in a small or large circle? Try it again, but this time, move the Steering slider to a different location before you upload and run the program. Did the bot move in a smaller or larger circle?

Power Settings

Next on the configuration panel is the Power section (see Figure 4-7). The Power setting range is 0 to100. You can type a value into the Power text box or drag the sliding bar to the right (to decrease power) or to the left (to increase power).

Figure 4-7. *The MOVE block's Power setting has a range of 0 to 100.*

Most uses of the Power setting will involve increasing or decreasing the spin speed of a motor. But there is one additional consideration, that is, lifting or pushing strength. If your robot is lifting a heavy object, for example, you might need to set the Power setting to a higher value. The motor will not spin as fast as it would if there were no resistance, but you may find that you need that extra power for the motor to successfully lift the object. The same goes for pushing. To push an object, your bot might need a higher Power setting than it will if it's not pushing anything. Surface conditions also affect power; climbing a hill will take more power and possibly slow the robot. Likewise, going down a hill won't take as much power. Also, whether a surface is smooth or rough can affect power; for example, you need more power to move over carpet than wood flooring. This is one of those settings where you'll just have to experiment. Change the Power setting, and play around with the Steering slider. See how fast or slow you can program your bot to make a circle. This will give you a better understanding of how the Power setting will affect your future bots.

Duration Settings

The Duration section of the configuration panel offers the most control of the `MOVE` block. There are four options in the Duration drop-down menu: Unlimited, Degrees, Rotations, and Seconds (see Figure 4-8).

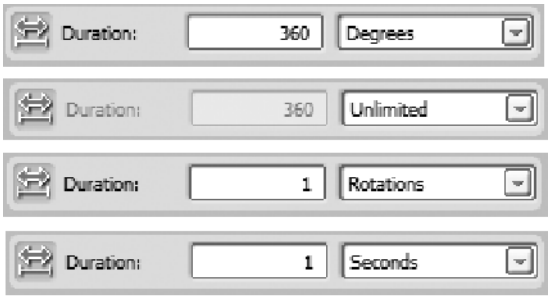

Figure 4-8. *There are four options for the Duration section of a MOVE block.*

From the Duration section, you can choose to have your motors spin forever by clicking the drop-down menu and choosing Unlimited. When the Duration is set to Unlimited, a single MOVE block will continue to spin its motors until the program ends or until you stop it. (There are other ways to stop a MOVE block such as using a LOOP block; I'll cover the LOOP block in Chapter 11.)

If you set the MOVE block Duration to Degrees, you must enter a value in the text box for the number of degrees for the motor(s) to spin. The value must be 0 or greater; it cannot be negative, but this limitation is simple to fix. If you wish for your motor to spin –90 degrees, for example, you simply type **90** in the text box and change the MOVE block Direction to its opposite setting (if it's set to spin FORWARD, just change it to REVERSE). If you've experimented with other programming environments, this may be unusual; it's possible you may have learned to use negative numbers to represent counterclockwise spinning of motors. Don't worry; you'll get used to the NXT-G method of simply changing the motor spin direction in the configuration panel. Just experiment with this concept, and it will start to make sense. Remember that your bot can spin a positive number of degrees, but you have to tell it whether to spin clockwise or counterclockwise by using the FORWARD or REVERSE directional controls.

If the Duration is set to Rotations, the same rules apply. You cannot enter a negative value for rotations, but any value of 0 and higher is acceptable. In order to spin the motor(s), a negative number of rotations, just change the Direction to its opposite setting (FORWARD or REVERSE). One thing you *can* do with Rotations is use fractional or decimal values. For example, you could configure a motor to spin 2.3 rotations or 50.9 rotations.

You may be wondering why you would ever want to configure a motor to spin 2.3 rotations. Well, I'll be covering that in this book's appendix when I show you how to program MOVE blocks for specific distances. For now, just keep in mind that your bots have the ability to move very small distances or very large distances with good accuracy, and it all depends on your ability to figure out exactly how many degrees or rotations to spin the motors (feel free to skip ahead to the appendix if you just can't wait).

The last option in the Duration section is Seconds. When you choose this option, you must specify the number of seconds for the MOVE block to spin a motor (or motors). For obvious reasons, you can't configure it for a negative value (say, –5 seconds). Just type in the number of seconds you want the motor(s) to spin, and you're finished.

Braking and Coasting

Now, take a look at Figure 4-9, which shows the last option you can configure for the MOVE block—the Next Action section. There are two options: Brake and Coast. If you select the Brake option, any motors connected to the ports you've configured will be stopped *fast* when the Duration you set expires (for example, after 10 seconds). Braking is useful if you need your bot to stop quickly and accurately at a specific point. However, keep in mind that this takes battery power.

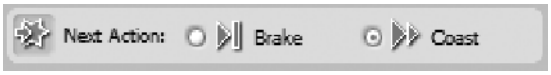

Figure 4-9. *You can configure motors to brake or coast.*

But what if you're not concerned about your bot stopping accurately? Then choose the Coast option. When you choose this option, any motors connected to the ports you've configured will stop receiving power, but the bot's momentum may carry it a little further; that's why it's labeled Coast—your bot will be coasting for a little while. You definitely want to try this out! So let's do that with some pseudo-code: SPOT, move forward 10 rotations at a Power of 75 and then Brake.

The MOVE block I've configured for SPOT is shown in Figure 4-10. Notice that I've configured Duration for 10 Rotations and Power at 75, and I've selected the Brake option. SPOT is also using Ports B and C for its motors, and I want it to travel in a straight line, so I've left the Steering control alone.

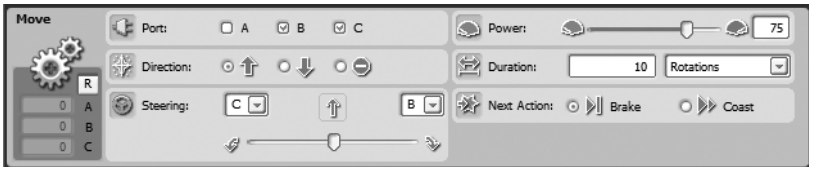

Figure 4-10. *The braking pseudo-code has been converted to a NXT-G program for SPOT.*

Now, when I upload and run this program, SPOT rapidly moves forward 10 rotations (about 6 feet) and comes to a quick stop.

Let's try a different test: SPOT, move backward 720 degrees at a Power of 100, and then Coast.

This time, I want SPOT to move in reverse, and I want the motors in Ports B and C to spin for 720 degrees. I want him to move *very fast,* so I've set the Power setting to 100. I don't need him to stop at a specific point, so I'll let him coast to a stop. See Figure 4-11 for the block programming of the pseudo-code.

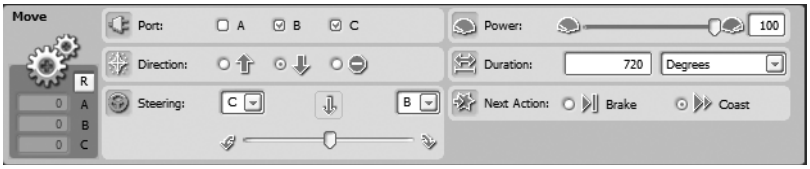

Figure 4-11. *A new coasting program for SPOT*

After I uploaded and ran this program, I watched as SPOT moved even faster in reverse for 720 degrees (about 15 inches). But this time, he didn't stop right away; he continued to roll for a few more inches, because I programmed him to coast to a stop.

Now, it's your turn. Using what you've learned in this chapter, create some different programs for your bot.

When you're finished, I have a final test for you to run before continuing on to Chapter 5: Program your bot to move forward for 10 rotations at a Power setting of 50, and set it to Brake. Place a piece of tape on the floor, and make this your bot's starting position. Now, run the program, and mark its stopping position with tape as well.

Next, run the same program, but change the Brake option to Coast. How far did it go beyond the previous stopping position?

Finally, reduce the Power setting a little bit, and run the program again (with the Coast option). How far did it go beyond the stopping position this time?

Keep reducing the Power setting and running the program until the bot stops at the original stopping position.

Why am I asking you to do this? Recall that I told you that the Brake option uses up battery power. This test shows you that you can save battery power by reducing the Power setting and keeping the Coast option. Running tests like this will help you to figure out how best to program your bot to save battery power and to correctly perform its programmed actions!

Okay, that's it for the `MOVE` block. Feel free to play around with the `MOVE` block until you're comfortable with it. Then, continue on to Chapter 5, where I cover the `RECORD/PLAY` block.