CHAPTER 2

Program Structure

I don't really like using technical terms like "program structure," but it is a very useful concept that will benefit you as you begin to program your robots. So bear with me for this short chapter.

What Do I Mean by Structure?

Back in Chapter 1, I gave you some examples of real-world programs. Would the following example have made any sense?

Teacher: Class, open your books to page 55.

[*The class looks confused.*]

Teacher: Class, I want you to get out your history books.

[*Giving the teacher confused looks, the students get out their books.*]

How can you read page 55 if you haven't yet been told which book to open? You might answer, "Yes, but I'm in history class, and the teacher said turn to page 55. So I'm sure the teacher means my history book!"

That's true. As a human, you are able to figure out certain instructions on your own. But remember—robots aren't that smart! They need to be given very strict and specific instructions. And those instructions need to be given in a specific order. That order is another way of saying "program structure."

Let's get out SPOT for another example. He's still doing his one and only trick—sit. We're not quite ready to upload an NXT-G program yet, but let's do some preplanning at this stage. I want you to use something that computer programmers call *pseudo-code*. What is pseudo-code? Well, the definition of "pseudo" is fake (as in pretend, simulated, virtual—get the idea?); it's not real. And "code" is simply another word for program. So put it all together and one way of looking at pseudo-code is this: fake program.

Our fake program isn't going to be written using NXT-G. The best way I can tell you to start creating a fake program is to pretend that SPOT has ears and tell SPOT what you want him to do. Let's try writing some pseudo-code using a numbered list:

1. SPOT, move forward until your Touch sensor is pressed and released; then stop.
2. Okay, SPOT, I want you to turn left 90 degrees.

3. Good job, SPOT. Now move backward until your Light sensor detects something black; then stop.

4. Now, SPOT, do a little dance.

That's pseudo-code? Well, it's a form of pseudo-code. Remember how I told you there are different programming languages? People write pseudo-code differently, too. The point I want you to understand is that before you can really program your robot using NXT-G, you need to have an idea of exactly what your robot will be doing. And the easiest way to do this is to simply write down, in simple language, instructions for your robot. That's the beginning of a good structure for the future NXT-G program.

When you write pseudo-code, you are accomplishing two things:

- You are gaining a better understanding of the tasks your robot will perform.
- You are creating an ordered set of instructions (structure) for your robot to follow.

You will use this pseudo-code to assist you when you begin to create your program with NXT-G. One final thing I want to mention about pseudo-code is that each instruction you give the robot should be as simple as possible. Take a look at the next two examples and tell me which one has the simpler instructions:

- *Example 1*: SPOT, move forward about 10 inches; turn left 90 degrees, and start moving forward; then start looking for a black object with your Ultrasonic sensor, because I want you to stop when you find a black object; then turn right 90 degrees, and move backward 2 feet, OK?
- *Example 2*:
  - SPOT, move forward 10 inches and stop.
  - Now turn left 90 degrees.
  - Starting moving forward, and turn on your Ultrasonic sensor.
  - Stop when you find a black object.
  - Turn right 90 degrees and stop.
  - Now move backwards 2 feet and stop.

Which example is less complicated to read? If you said Example 2, you are right. Let's be honest—some humans would be confused if you gave them the instructions in Example 1! When writing pseudo-code, break down your instructions into short and simple statements for your robot. This will make it easier for you to convert your pseudo-code to an NXT-G program.

Are you wondering how you convert pseudo-code to a real NXT-G program? Let me give you a small preview of what's to come in the chapters ahead.

Take a look back at my original pseudo-code for SPOT and read step 3, "Now move backward until your Light sensor detects something black; then stop."

If I am programming in NXT-G and am familiar with all the tools it contains, I would realize that there are tools (called *blocks*) that match up to my pseudo-code. Let me explain briefly.

When I want SPOT to move backward, he's going to use his motors, right? Well, I'll be using something called a MOVE block. The MOVE block will allow me to program my bot to spin the motors (and wheels) in reverse, so SPOT moves backward.

I only want SPOT to back up until his Light sensor detects the color black. To do this, I'll use something called a SENSOR block to monitor the Light sensor. The SENSOR block will be programmed to look for the color black.

Finally, I want SPOT to stop when the SENSOR block detects the color black. For this, I can use another MOVE block that tells the motors to stop spinning.

You will use these blocks and many more to properly program your robot to follow your instructions. This book will teach you about all the different NXT-G blocks, so you'll know which ones to use when converting your pseudo-code to an NXT-G program.

Okay, I told you this chapter wouldn't be long, and I meant it. If you can remember one thing from this chapter, it should be this: Programming your robot will be much easier if you take the time to write down the pseudo-code. If it helps, pretend your robot has ears, and tell it what you want it to do. Write down these instructions, and keep them short and simple.

In the next chapter, we're going to actually write some pseudo-code and convert it to an NXT-G program. The key to writing excellent NXT-G programs is understanding how the NXT-G programming blocks work; when you know how the blocks work, you'll know which blocks to use when converting your pseudo-code.

Chapter 3 is going to demonstrate the DISPLAY block, a very useful block that gives your robot the ability to write things to its LCD screen for others to read.