Learn to Code Book 1 “Adding a New Command” Lesson 002

Learning goals: algorithms, invoking/calling/using methods or functions, program

Introduction: At this point, your students are going to be chomping-at-the-bit to do more of these playgrounds Let’s not get in their way too much.

We are adding a new behavior to Byte’s repertoire: `turnLeft()`

Note: If possible, try to use the words **behaviors** and **commands** when you discuss things with the students. The **commands** are the code in the program that make Byte do his **behaviors**. The word “behavior” will be important later when we introduce types and object-oriented programming concepts. So, introducing it early will familiarize students with the idea that Byte is an object and objects have behaviors associated with them.

Ask the students: “In this playground, what behaviors does Byte have?”

Byte currently has the behaviors:

`moveForward()`

`collectGem()`

`turnLeft()`

Before you start:

- Have the students pinch-out and pinch-in on the right side of the screen to see that they can zoom in and zoom out if they want to see detail of the scene.
- Remind the students that they can swipe left and right on the right side of the screen to see that Byte’s world rotates.
- Have students READ the instructions!
- Tap any red words in the instructions. These will define any new and important terms. In this lesson, the word ***command*** is linked and a definition is provided.
- Again, note the commands on the bottom of the screen to the students: `moveForward()`, `collectGem()` and `turnLeft()` should be all the commands in that are needed in this program. Later on, we will use a keyboard to type commands ourselves, but for these first few lessons, students should get used to tapping the shortcuts listed at the bottom.

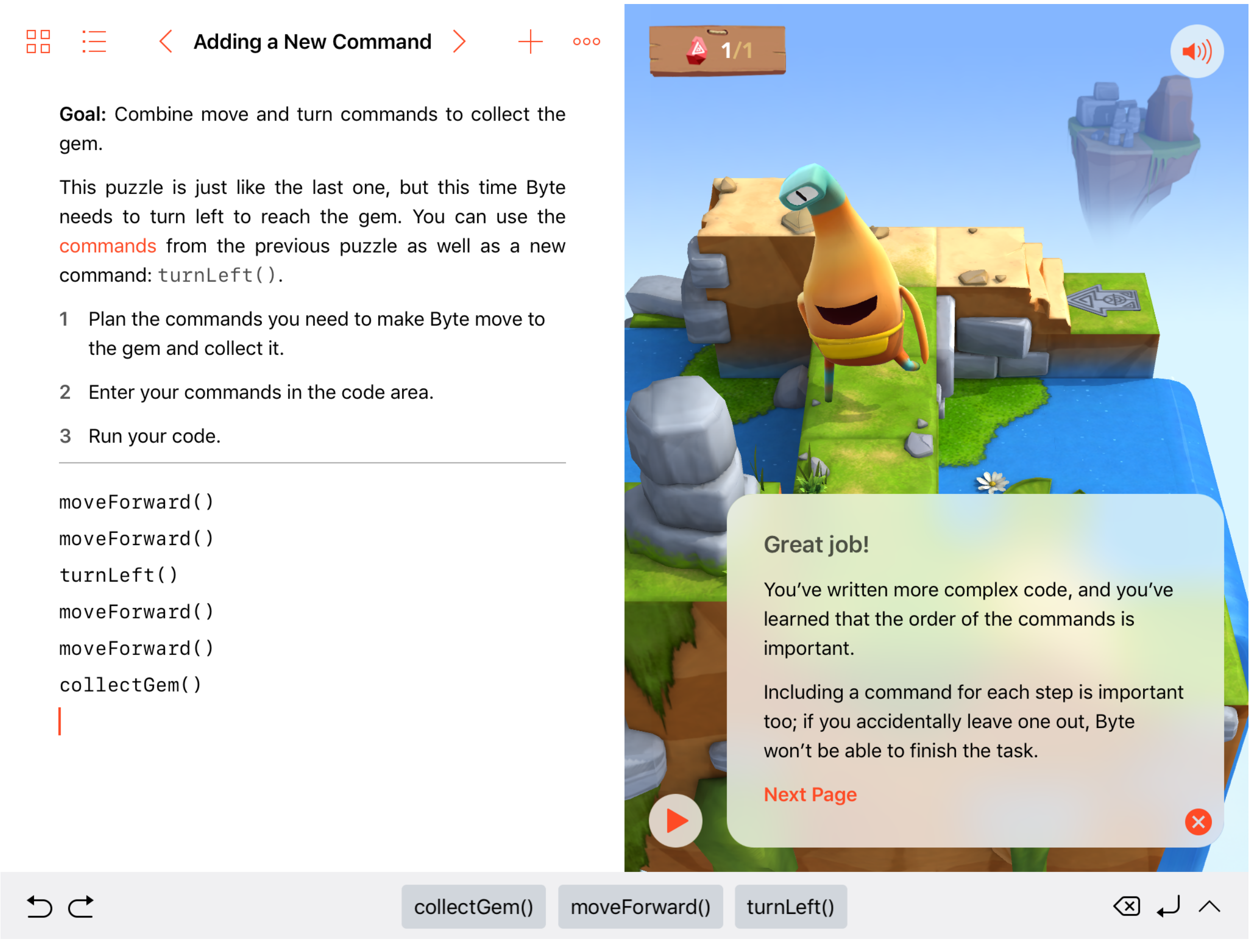
Activity: Have students write their code by tapping on the commands in the correct order and test it by tapping the *Run* button to make Byte perform their program. It’s OK to make mistakes. Students can fix their mistakes and press *Run* again to test their program.

Give students sufficient time to explore this playground but students should not move ahead to the next activity until it is introduced. (See solution below)

Follow-up activities:

- To students: “Does anyone want to share the statements they wrote in their program?” “Did your program work?”
- “Did anyone make a mistake in their program the first time they wrote it?” “What was the mistake?” "How did you fix it?”

Solution:



Goal: Combine move and turn commands to collect the gem.

This puzzle is just like the last one, but this time Byte needs to turn left to reach the gem. You can use the commands from the previous puzzle as well as a new command: `turnLeft()`.

1. Plan the commands you need to make Byte move to the gem and collect it.
2. Enter your commands in the code area.
3. Run your code.

```
moveForward()
moveForward()
turnLeft()
moveForward()
moveForward()
collectGem()
```

Great job!

You’ve written more complex code, and you’ve learned that the order of the commands is important.

Including a command for each step is important too; if you accidentally leave one out, Byte won’t be able to finish the task.