

ADVANCED EV3 PROGRAMMING LESSON



EV3 Classroom: Menu System

By Sanjay and Arvind Seshan



EV3 CLASSROOM LESSON
BY EV3LESSONS.COM

Lesson Objectives

- Learn to use variables
- Learn to create a menu system that is not limited to a particular number of choices
- Learn to create a menu system that updates the menu view
- Prerequisites: Variables, Math Blocks, Brick Buttons

A Fancier Menu System

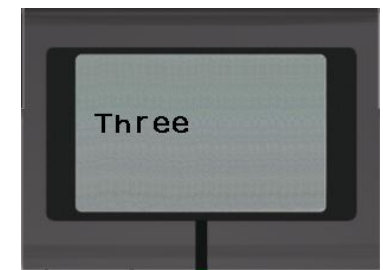
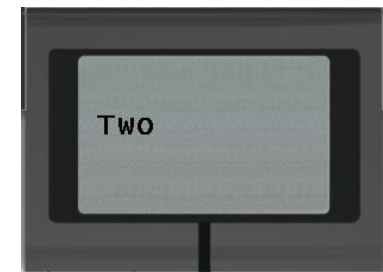
- In the “Using Brick Buttons as Sensors” Lesson in Intermediate, one of the challenges asked you to create a menu with 4 choices and a single screen display for the entire menu
- In this version, we build a menu system that updates the menu view each time you change your selection and lets you have a larger number of menu choices
- To make this menu, you will need to learn how to use variables



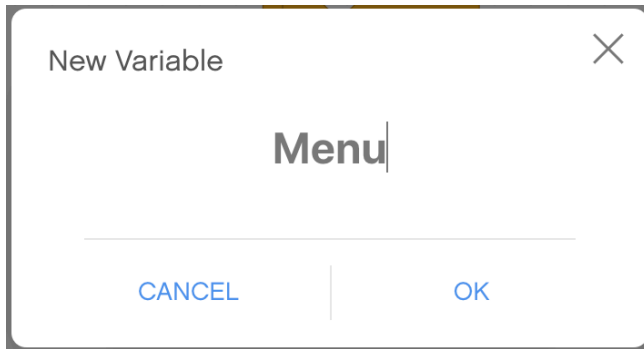
Menu Challenge

- **Challenge:** Make a menu system that lets you perform 3 actions (display and say the numbers 1, 2, and 3) based on the button pressed
- **Step 1:** Use a variable to store the current menu choice
- **Step 2:** Display the menu description for the current menu choice
- **Step 3:** Wait for the user to press a button (top, middle, down buttons)
- **Step 4:** Based on the button press: run the code for the menu choice (for middle button), or increase/decrease the menu choice variable (for up/down buttons)
- **Step 5:** Go back to 2...

What you will see on the EV3 Brick

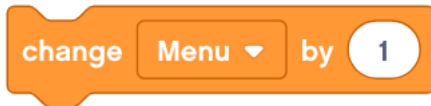


Main Component



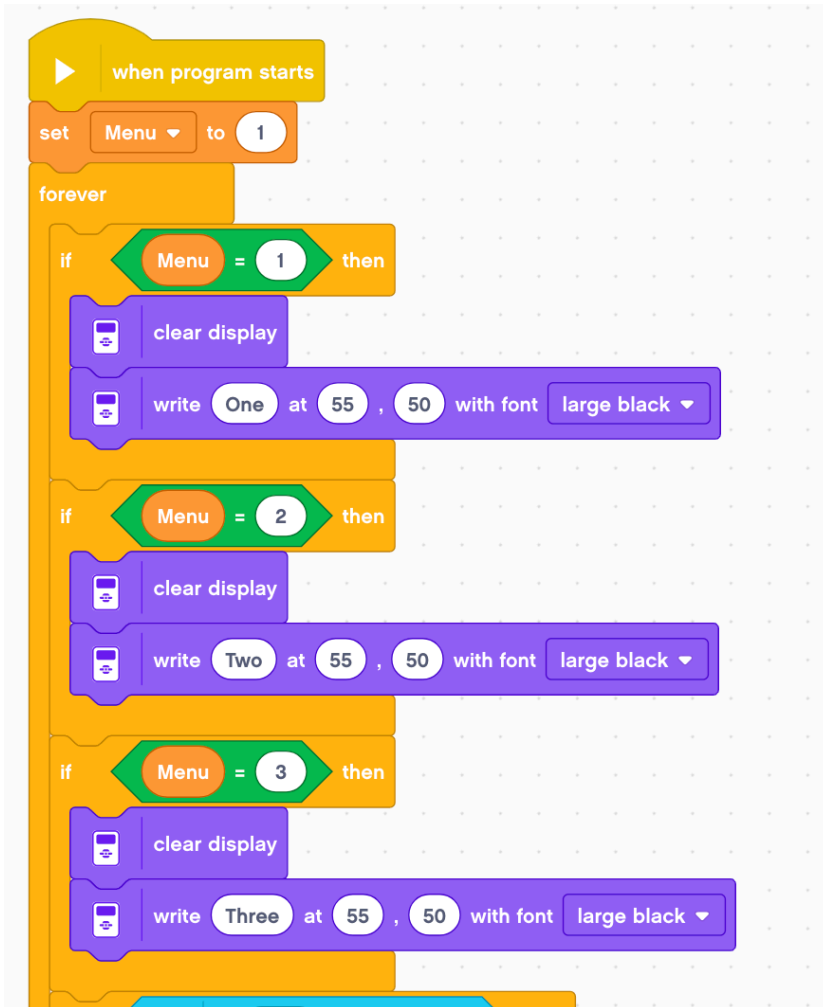
Variables can store values for later use. Make a variable and give it a name “Menu” before using it.

Variables



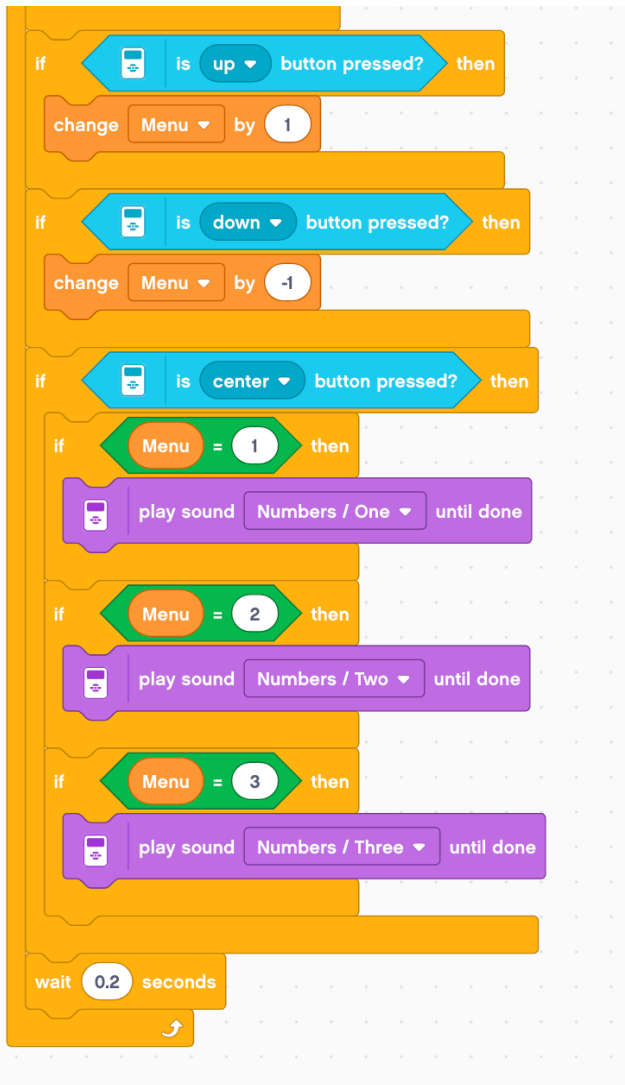
Once created, the variable shows up in the Block Palette under Variables. A Set “Menu” and Change “Menu” Block are also automatically created.

Solution Part 1



- ➔ Set Variable Menu to start at 1
- ➔ Display the word ONE, TWO or THREE at the center of the screen using Display Blocks
- ➔ Clear the screen before each word is displayed

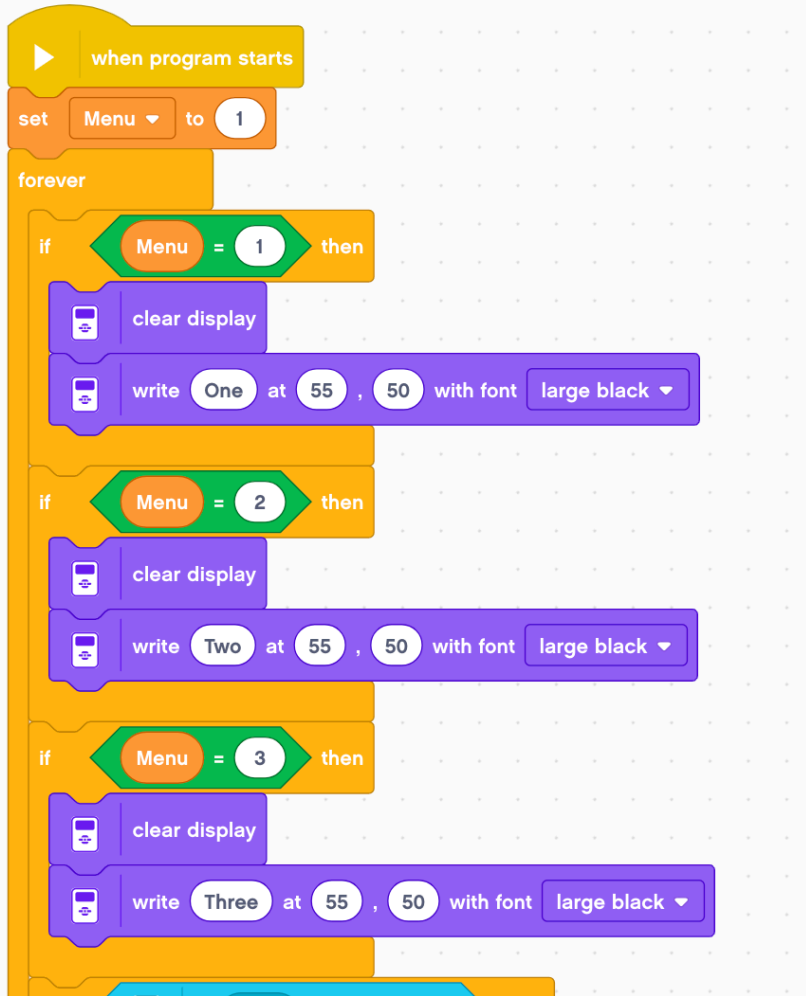
Solution Part 2



- ➔ When the Up or Down brick buttons are pressed, change the value of the variable Menu by 1
- ➔ Depending upon which value is picked, play a different sounds (“one”, “two”, or “three”)

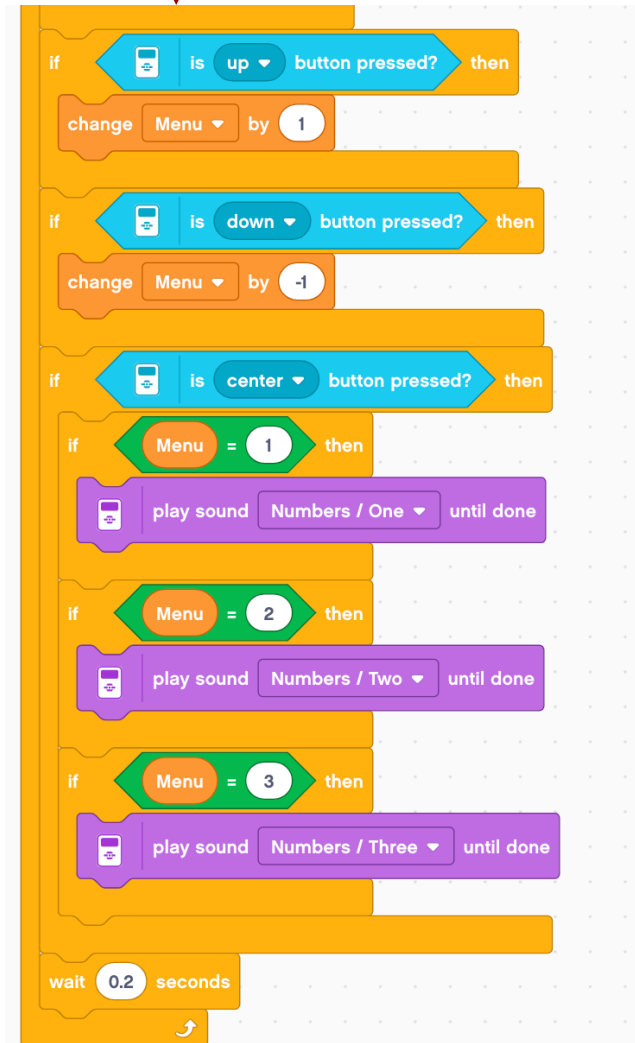
In this example, sounds are played based on the menu item picked. You can replace this with any code you wish.

CHALLENGE SOLUTION



```
when program starts
  set Menu to 1
  forever loop
    if Menu = 1 then
      clear display
      write One at 55, 50 with font large black
    if Menu = 2 then
      clear display
      write Two at 55, 50 with font large black
    if Menu = 3 then
      clear display
      write Three at 55, 50 with font large black
```

The code on the left is a Scratch script. It begins with a yellow 'when program starts' block. This is followed by an orange 'set Menu to 1' block. Below that is a large orange 'forever' loop block. Inside the loop, there are three conditional blocks: 'if Menu = 1 then', 'if Menu = 2 then', and 'if Menu = 3 then'. Each 'if' block contains a 'clear display' block and a 'write' block. The 'write' blocks are purple and contain the text 'One', 'Two', and 'Three' respectively, positioned at coordinates (55, 50) with a 'large black' font.



```
if is up button pressed? then
  change Menu by 1
if is down button pressed? then
  change Menu by -1
if is center button pressed? then
  if Menu = 1 then
    play sound Numbers / One until done
  if Menu = 2 then
    play sound Numbers / Two until done
  if Menu = 3 then
    play sound Numbers / Three until done
wait 0.2 seconds
```

The code on the right is a Scratch script that handles button events. It starts with three blue 'if is [button] pressed? then' blocks: 'is up button pressed?', 'is down button pressed?', and 'is center button pressed?'. The 'up' block is followed by an orange 'change Menu by 1' block. The 'down' block is followed by an orange 'change Menu by -1' block. The 'center' block is followed by a nested 'if' structure. This structure has three 'if Menu = [1, 2, 3] then' blocks, each followed by a purple 'play sound' block. The sound names are 'Numbers / One', 'Numbers / Two', and 'Numbers / Three'. At the bottom of the script is a yellow 'wait 0.2 seconds' block. A red arrow points from the top of the right script to the bottom of the left script, indicating that the right script is the continuation of the program.

Next Steps

- The ideas in this lesson can be adapted to help you build a mission sequencer for FIRST LEGO League. Sequencers are useful because they:
 - Allow you to skip missions if you are short of time
 - Allow you to repeat failed missions
 - Allow you access missions quickly (find them easily)
- If your Menu Action code is long (not just a display and sound), consider creating My Blocks out of your code

Credits

- This tutorial was created by Sanjay Seshan and Arvind Seshan
- More lessons at www.ev3lessons.com



This work is licensed under a [Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License](https://creativecommons.org/licenses/by-nc-sa/4.0/).