



North Texas FLL Coaches' Clinics Master Sequencers

Patrick R. Michaud
pmichaud@pobox.com

October 2016

Goals

Learn about master sequencer programs

Reduce errors and time spent in Base

Topics

MyBlocks

Master Sequencer Basics

Switch blocks

Loop blocks

Variables

Background

Hopefully you already know about...

Compiling and downloading programs to EV3

Motor / move blocks

Wait blocks

Touch sensors

MyBlocks

Master Sequencers

A “master sequencer” combines all missions in the Robot Game into a single program

This reduces time spent in Base by not requiring drivers to select the next program / mission to run

Master sequencers can also make it easy to repeat or skip missions

Basic concepts

Most FLL teams create separate programs for missions (or “trips”) out of Base

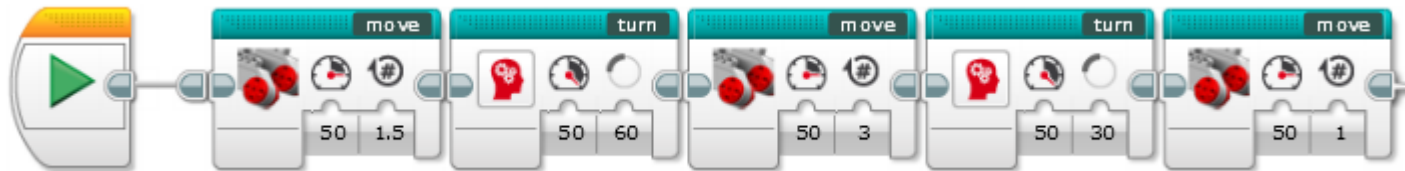
Steps:

1. Each program for the missions is converted into its own MyBlock
2. Master program calls all of these MyBlocks in the desired sequence

Basic terminology

FLL #27 Republic of Pi organizes its programs into “mission” and “trip” MyBlocks

A *mission* is the programming needed to solve a single mission combined into a single MyBlock



A *trip* is a sequence of *mission* blocks where the robot leaves Base and returns



The *master sequencer* allows the drivers to select the next *trip* to be run

It also automatically advances from one trip to the next

The simplest sequencer

Suppose our team has several programs for the robot game:

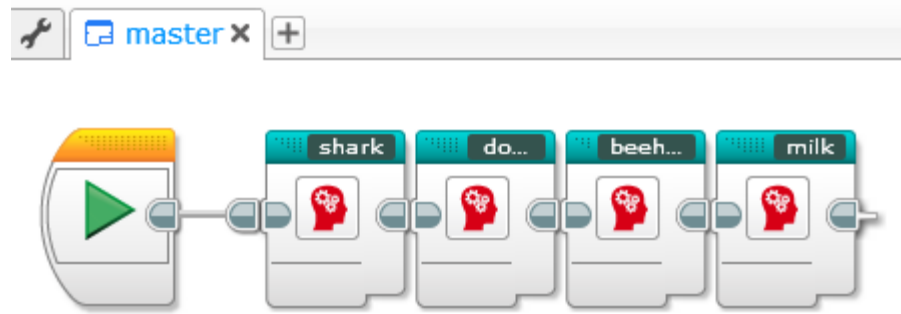
1. shark
2. dog-food
3. beehive
4. milk

The first step is to convert each program into its own MyBlock

Select the entire program, then use
Tools → My Block Builder

The simplest sequencer

Next, create a “master” program that calls each MyBlock in turn:

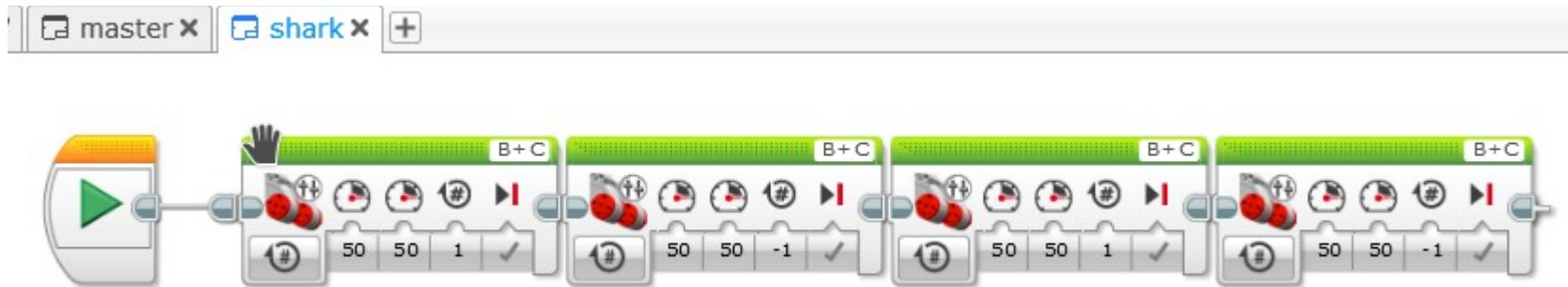


Of course, this will run all of the missions *without stopping* between each mission

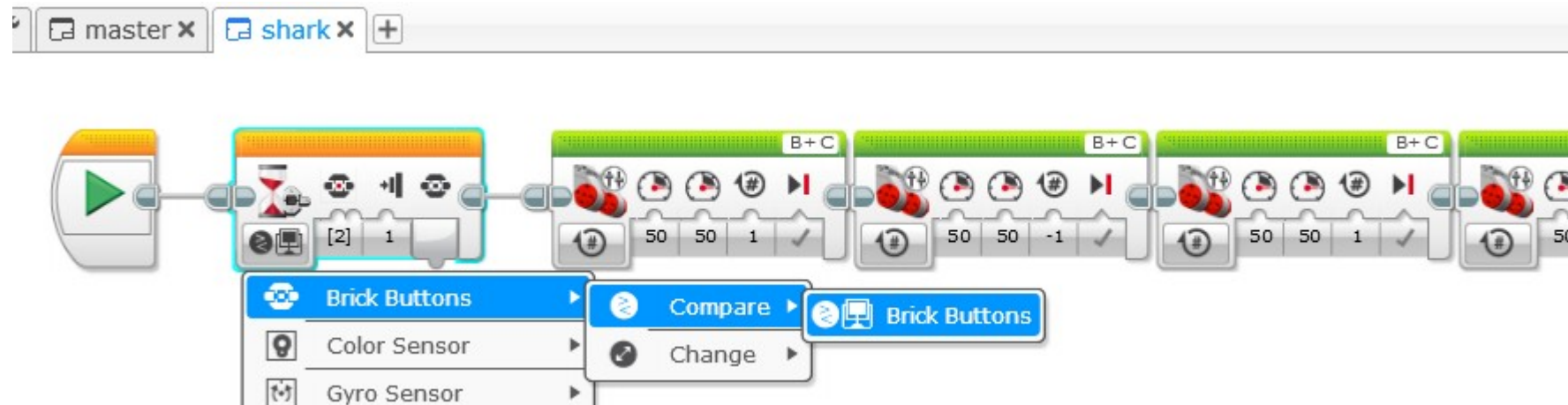
How to fix that?

Pausing between trips/missions

Add a Wait Block at the beginning of each trip/mission
MyBlock, so



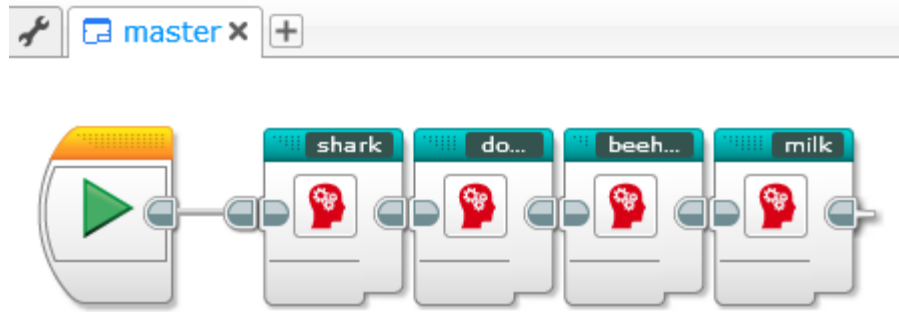
becomes



Now each mission will wait for start button to be pressed

The simplest sequencer

So, when “master” program is run, it runs each mission MyBlock in sequence



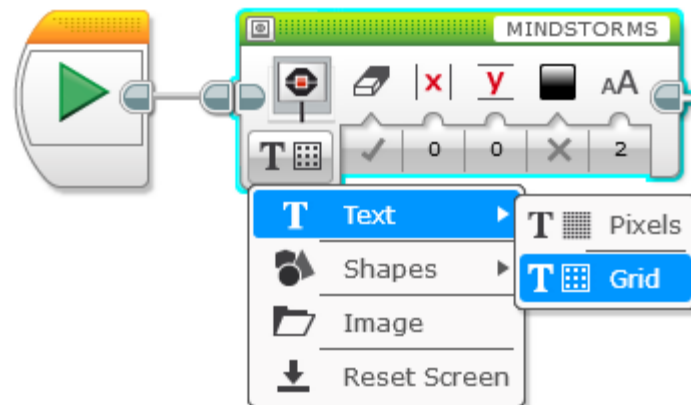
Each mission MyBlock waits for the Start button to be pressed before running

Displaying missions to drivers

Now let's improve our sequencer to tell the driver what mission will be run next

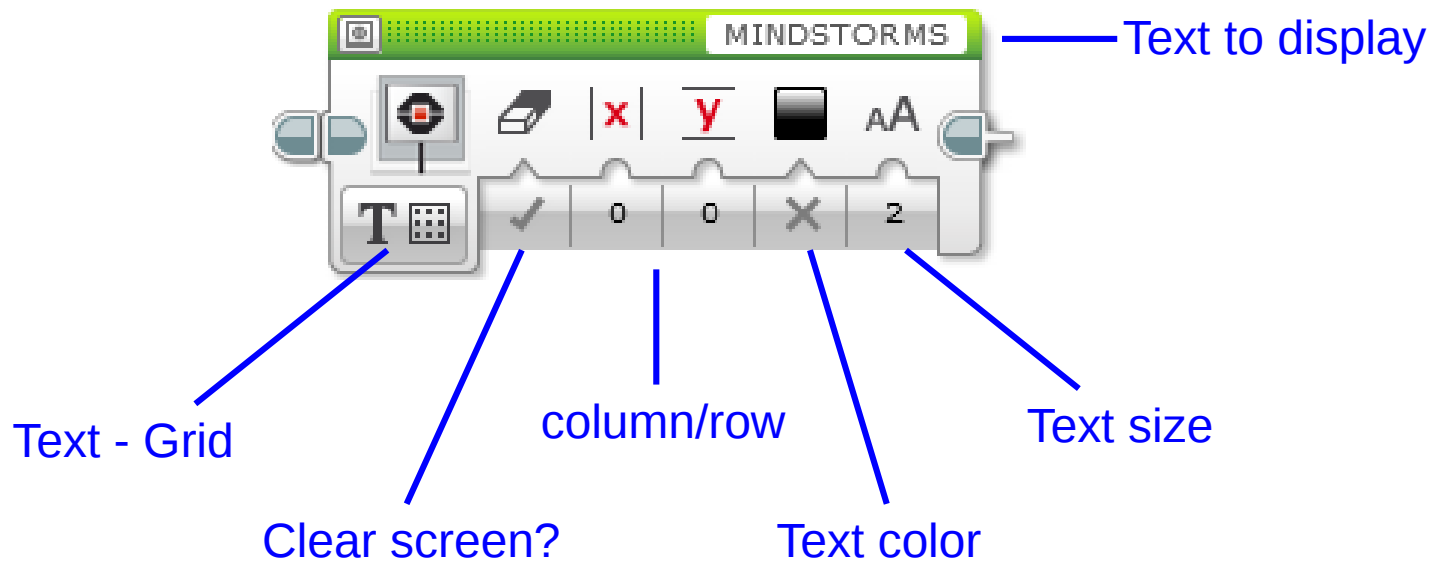
For this, we'll create a "tripstart" MyBlock

Create a new program, add a "Display" block:



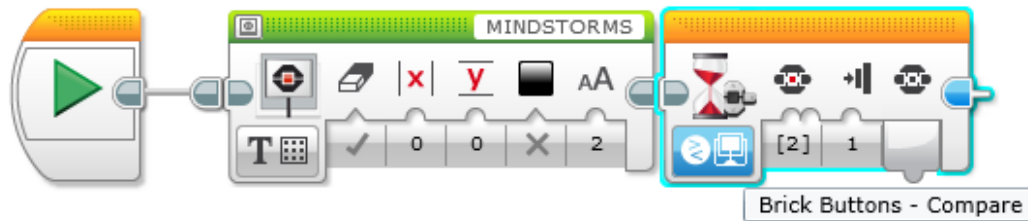
Display blocks

The Display block displays information on the EV3 screen:



Tripstart MyBlock - blocks

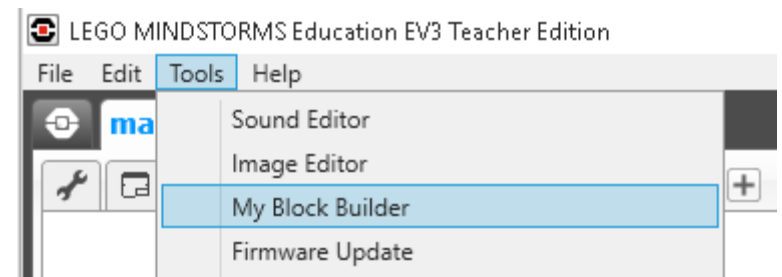
Add a Wait for Brick Button block:



Set state to “bumped” (2) instead of “pressed”:

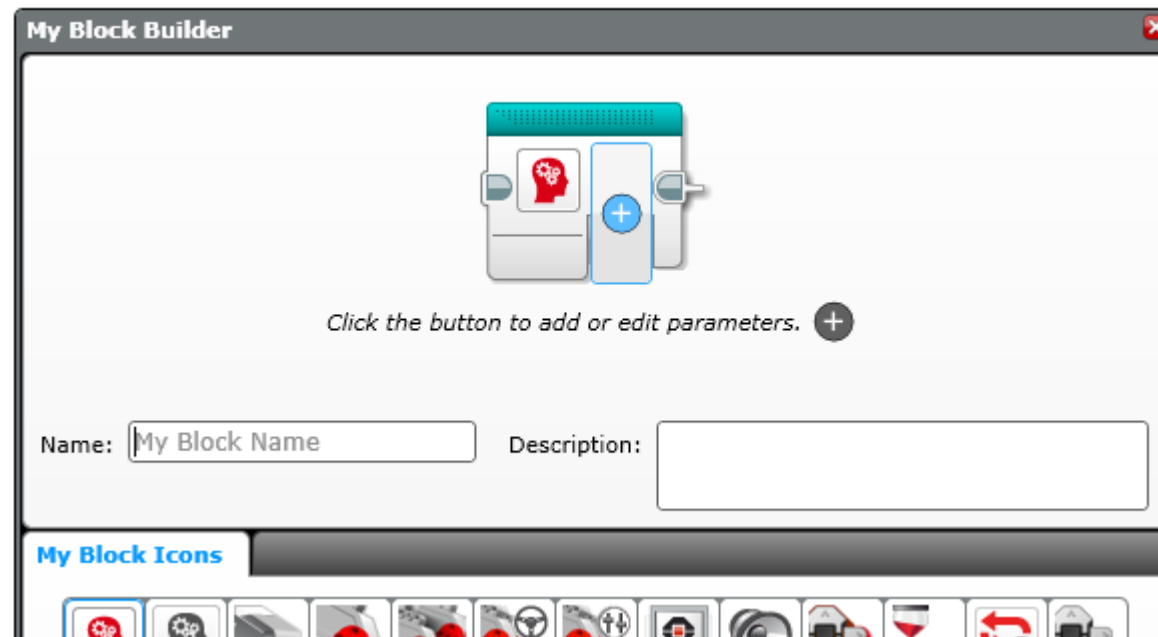


Select both blocks, then
Tools → My Block Builder



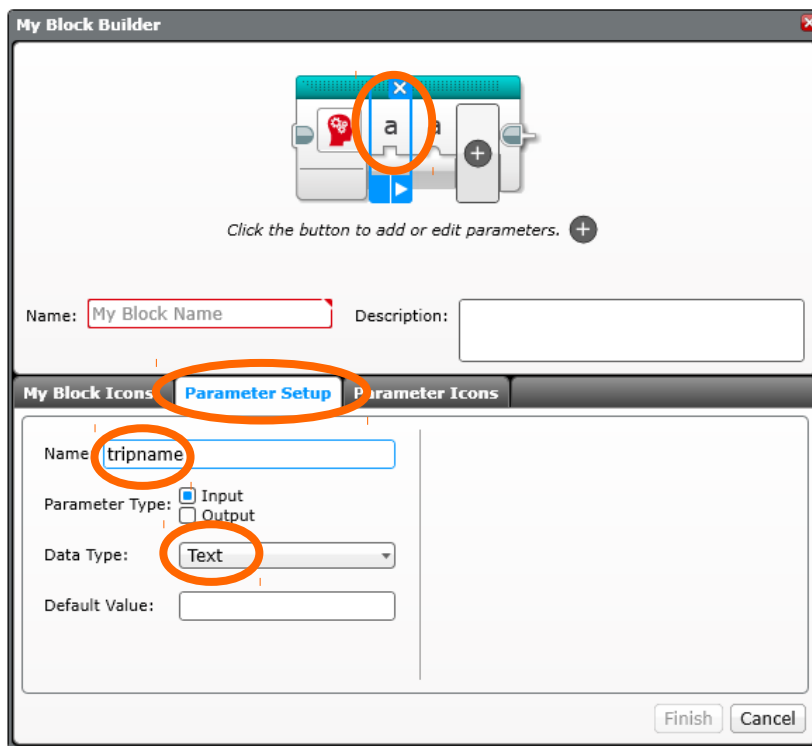
Tripstart MyBlock - create

1. Use the “+” button to add two parameters to the MyBlock

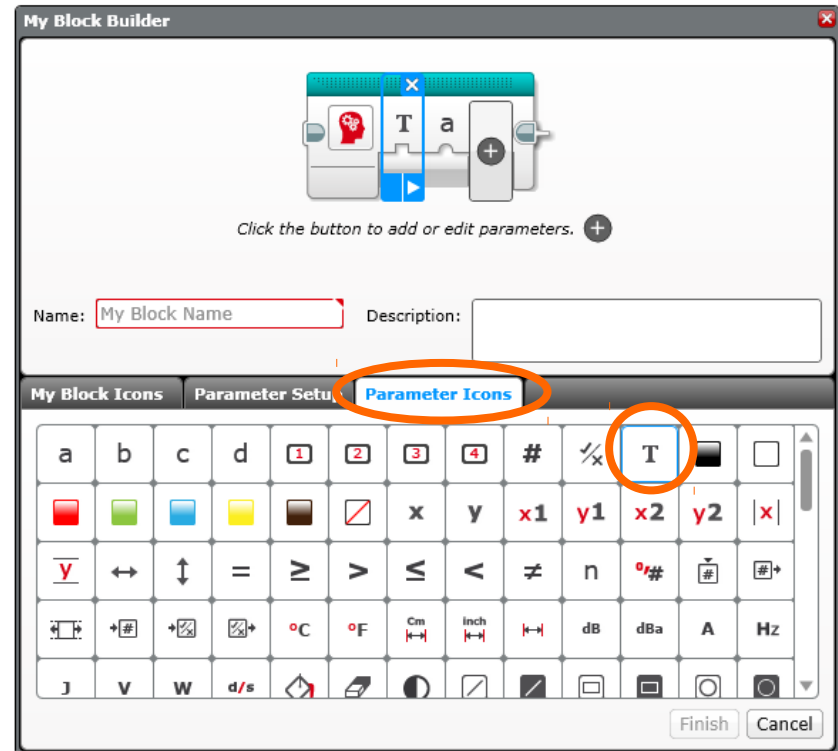


Tripstart MyBlock - parameters

2. Set input tripname parameter

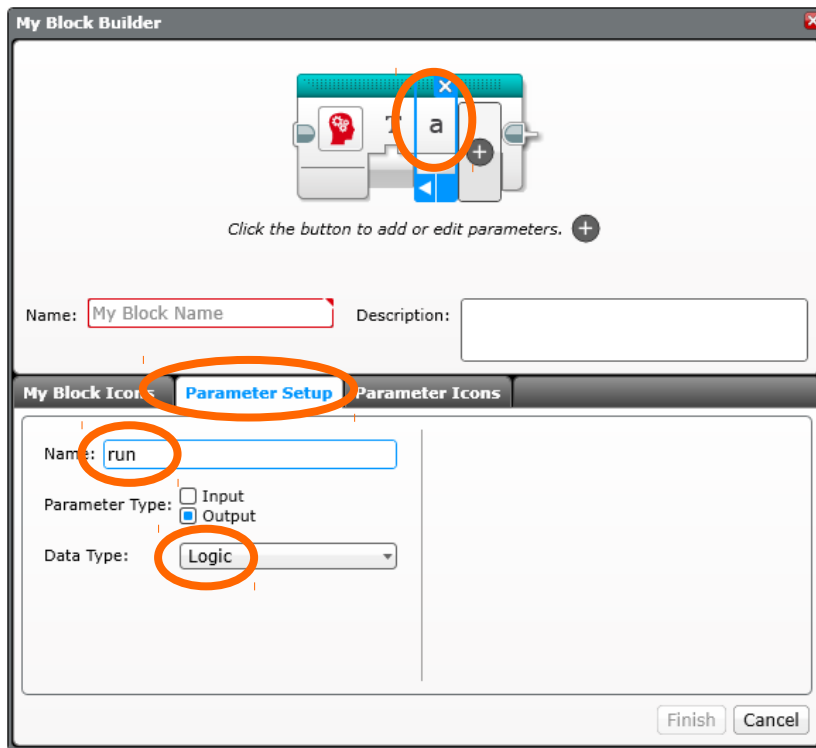


3. Set tripname icon

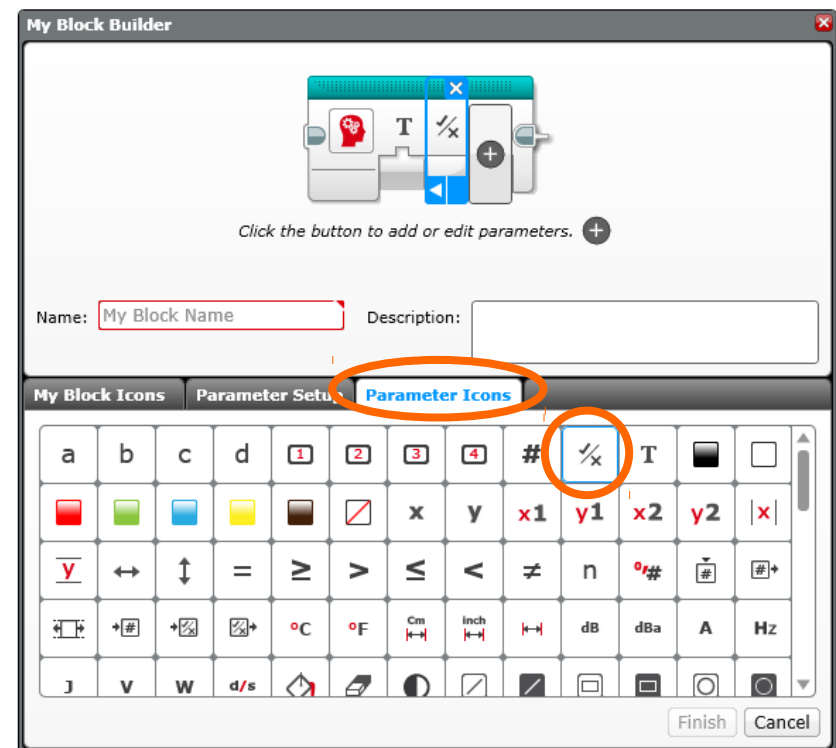


Tripstart MyBlock – “run” parameter

4. Set output “run” parameter

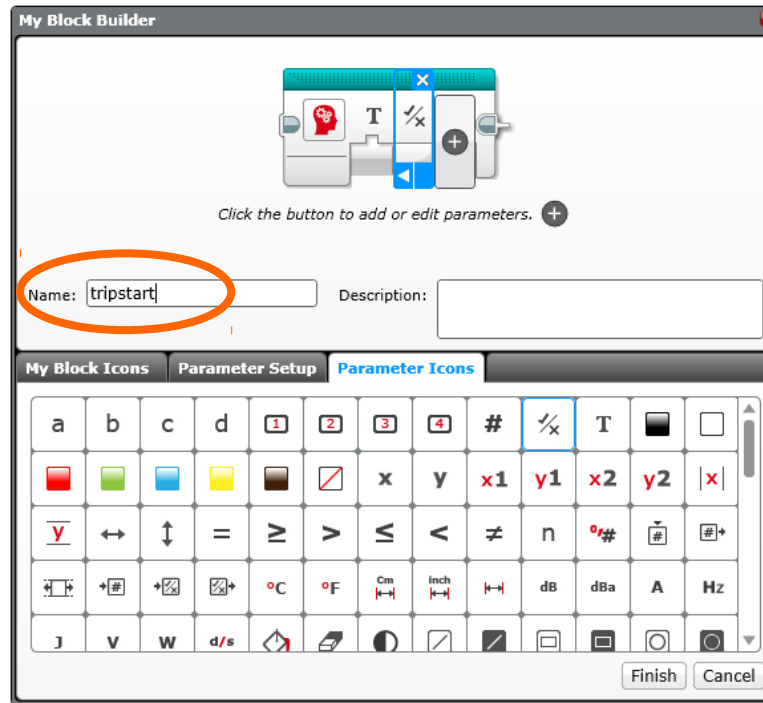


5. Set run icon



Tripstart MyBlock - finish

6. Give the MyBlock a name (“tripstart”)



7. Press “Finish”



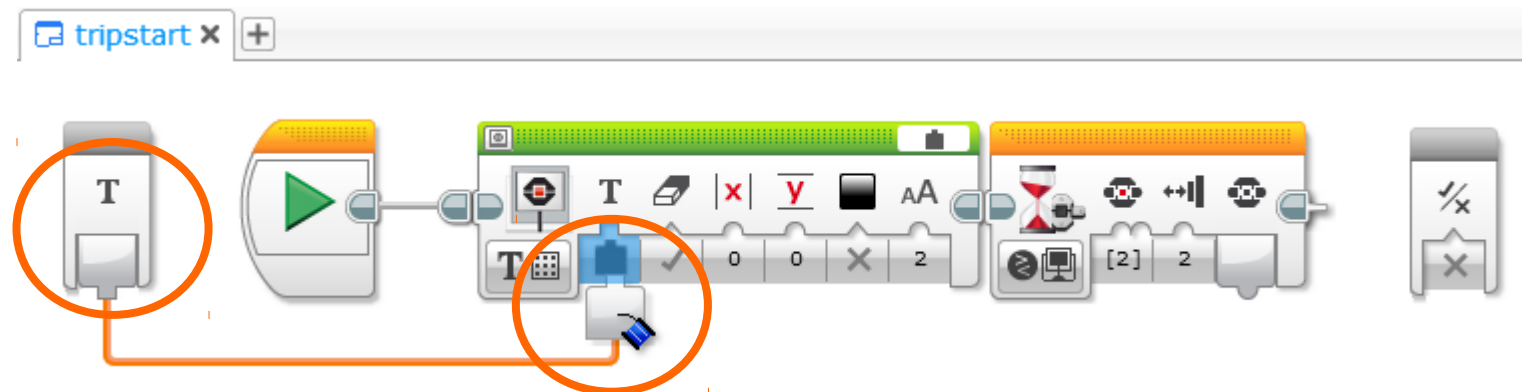
Tripstart – wiring text

8. Click on “MINDSTORMS” in display block and select “Wired”:



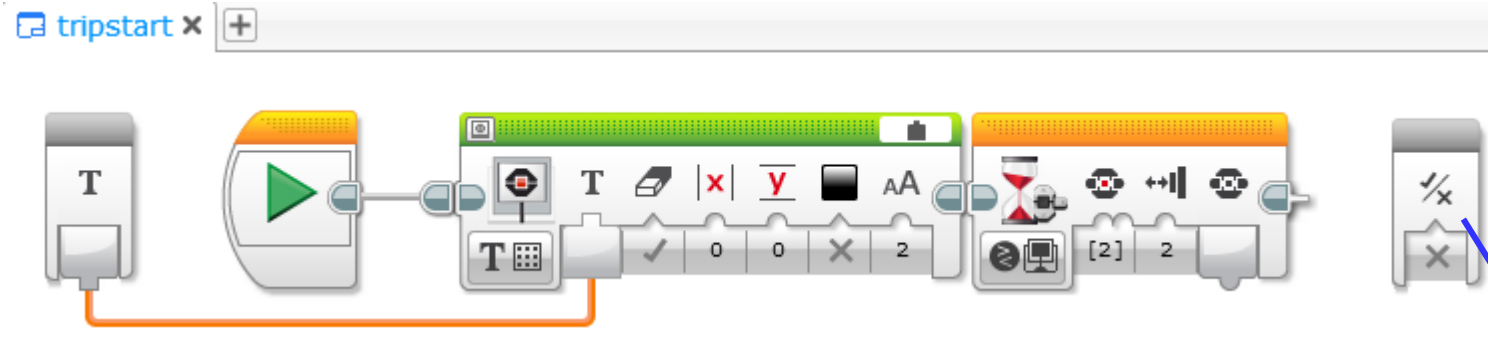
This adds a parameter to the display block.

9. Wire the input text to the display block:



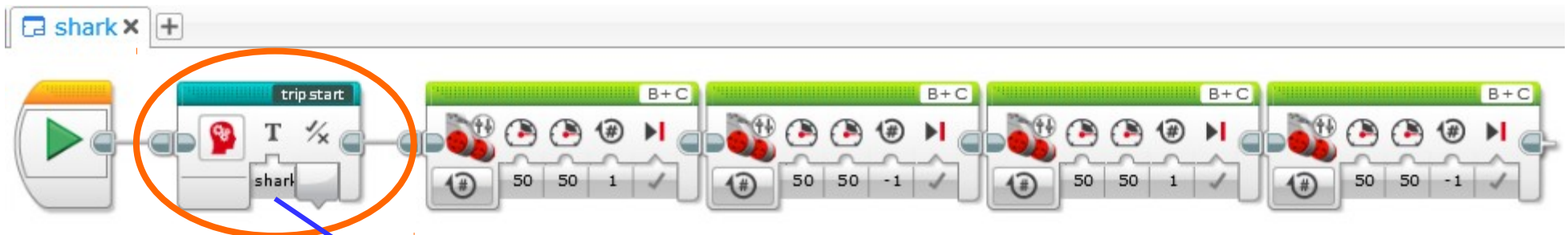
Tripstart – finished block

The “tripstart” block looks like this:



Ignore this block for now – we'll use it later.

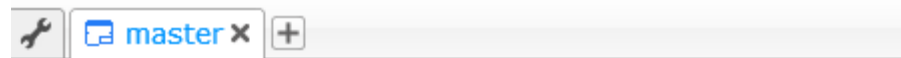
Instead of a Wait block at the beginning of each trip or mission, use the “tripstart” block:



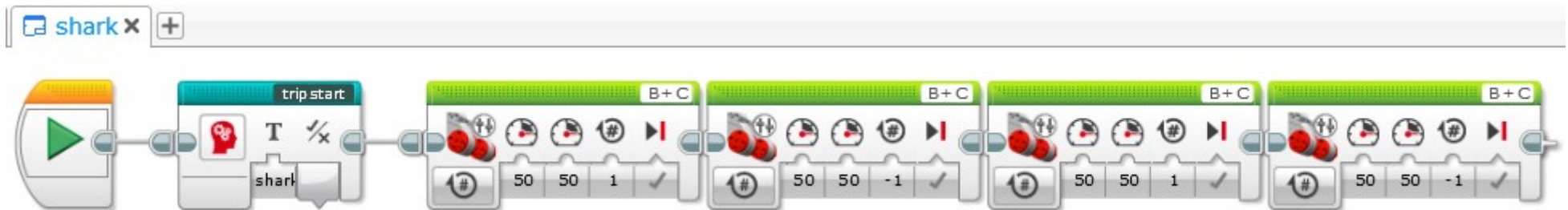
This is the text that will be displayed for the trip

A better sequencer

Now when “master” program is run, it runs each mission MyBlock in sequence



and each mission MyBlock uses tripstart block to display the mission to be run and wait for Start





Intermission

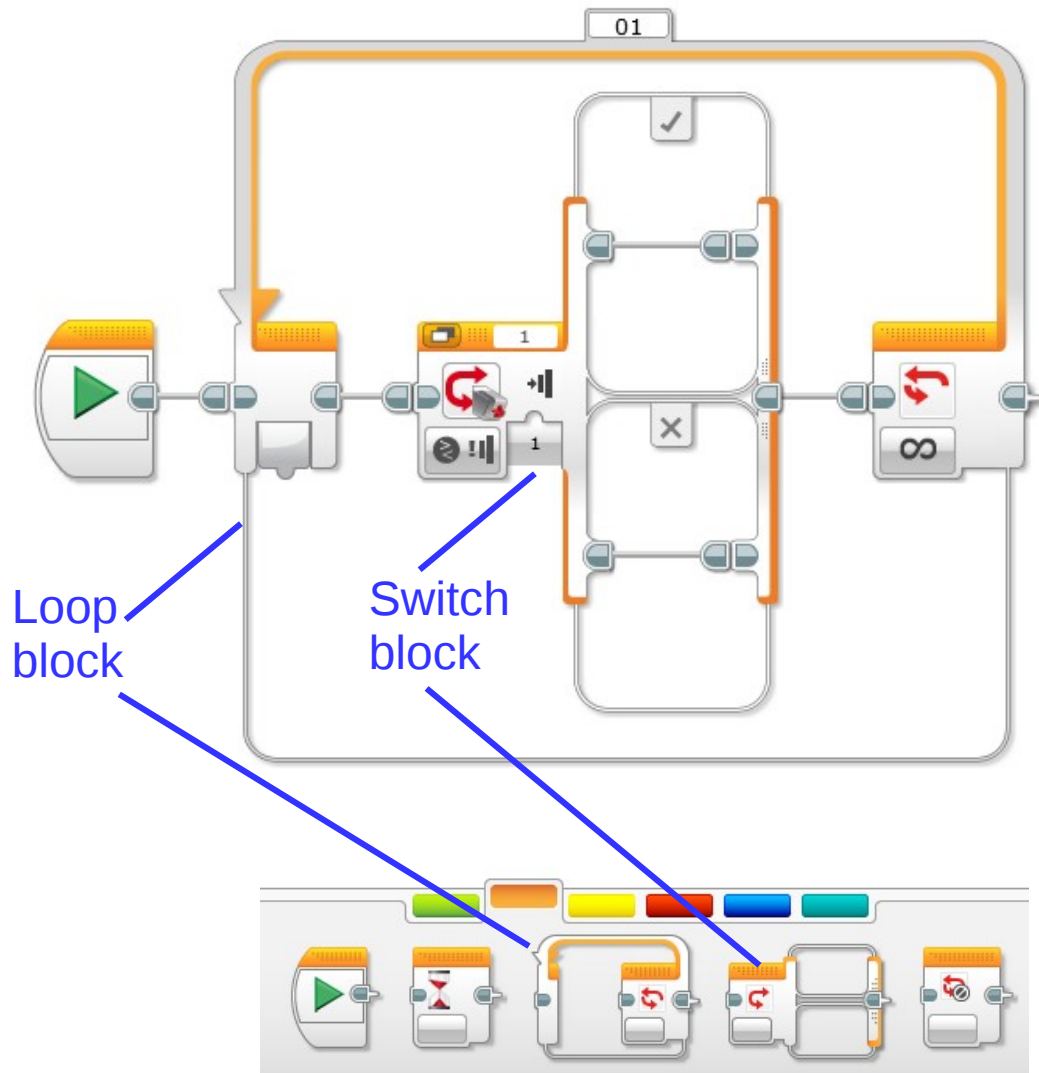


Q: What if we want to repeat or skip a trip?

A: We'll set up the left and right brick buttons
to select trip to run next

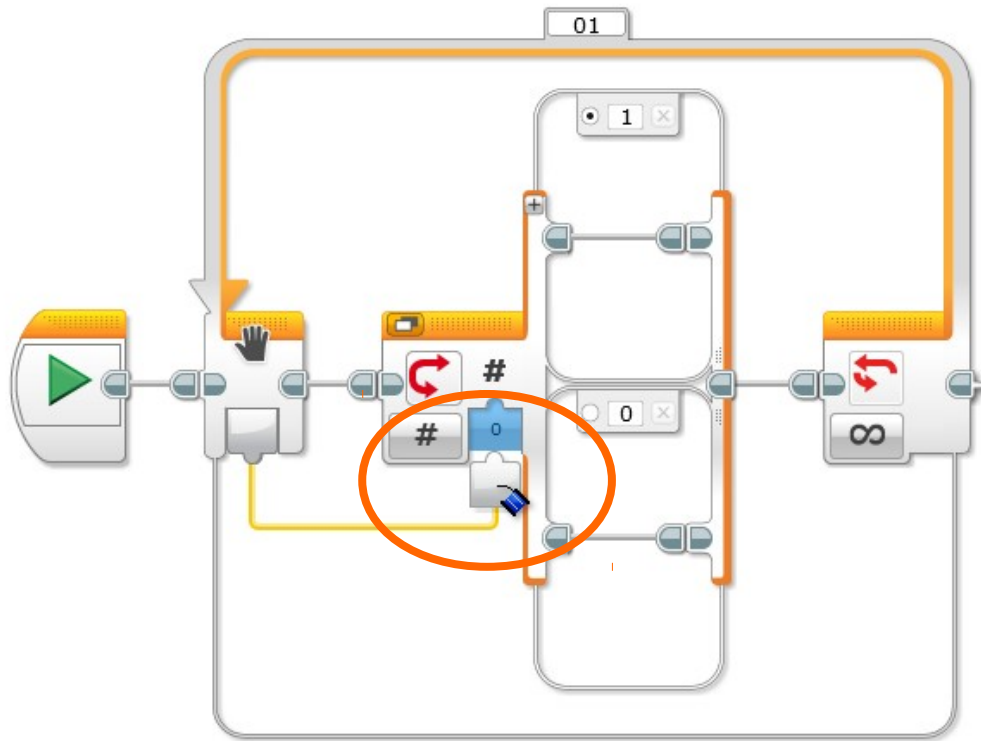
A new “master” program

Instead of a direct sequence, place missions to be run in a Loop block containing a Switch block



Master program using a loop/switch

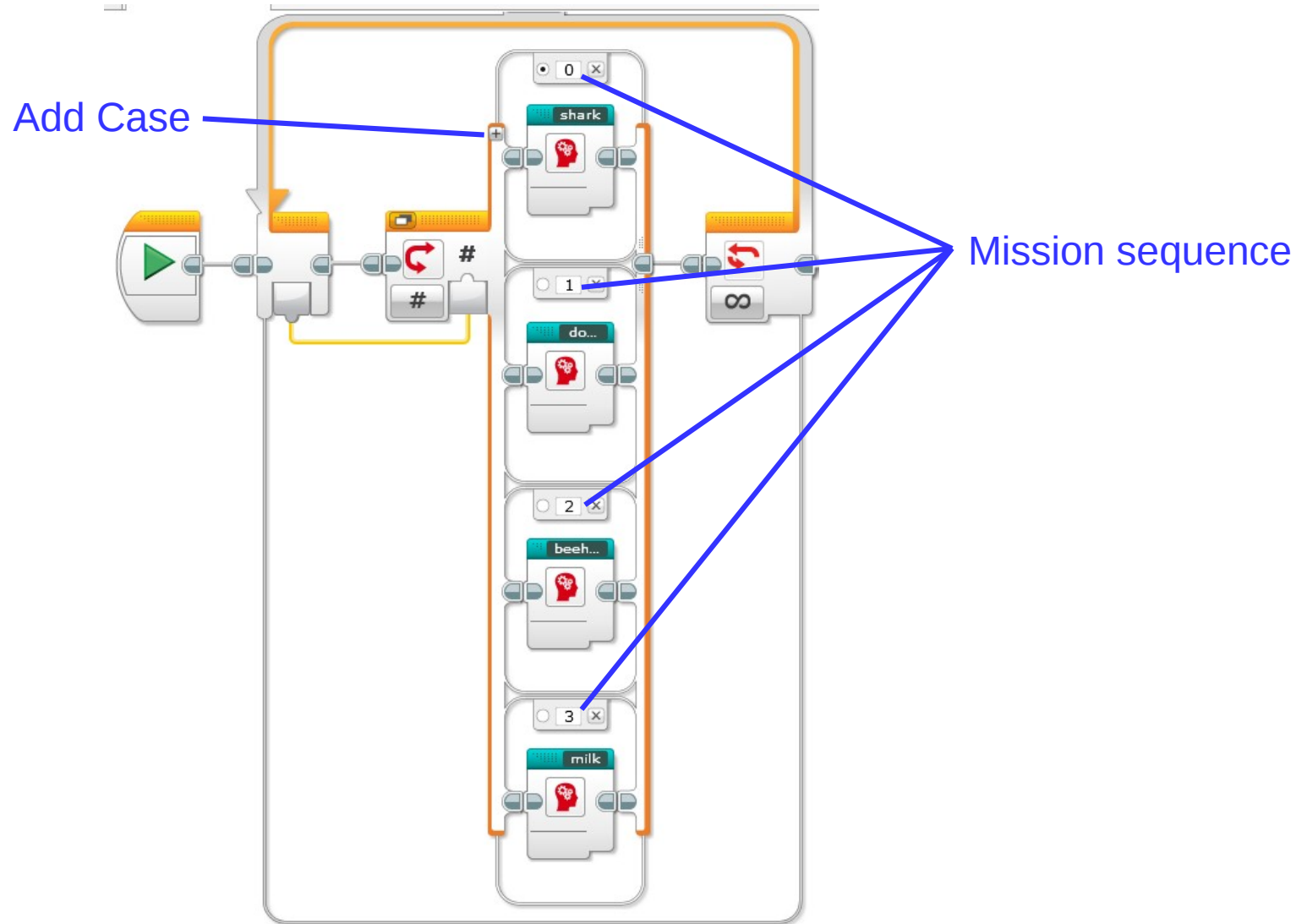
Change the Switch Block to use Numeric input, and wire the Loop Index to the switch:



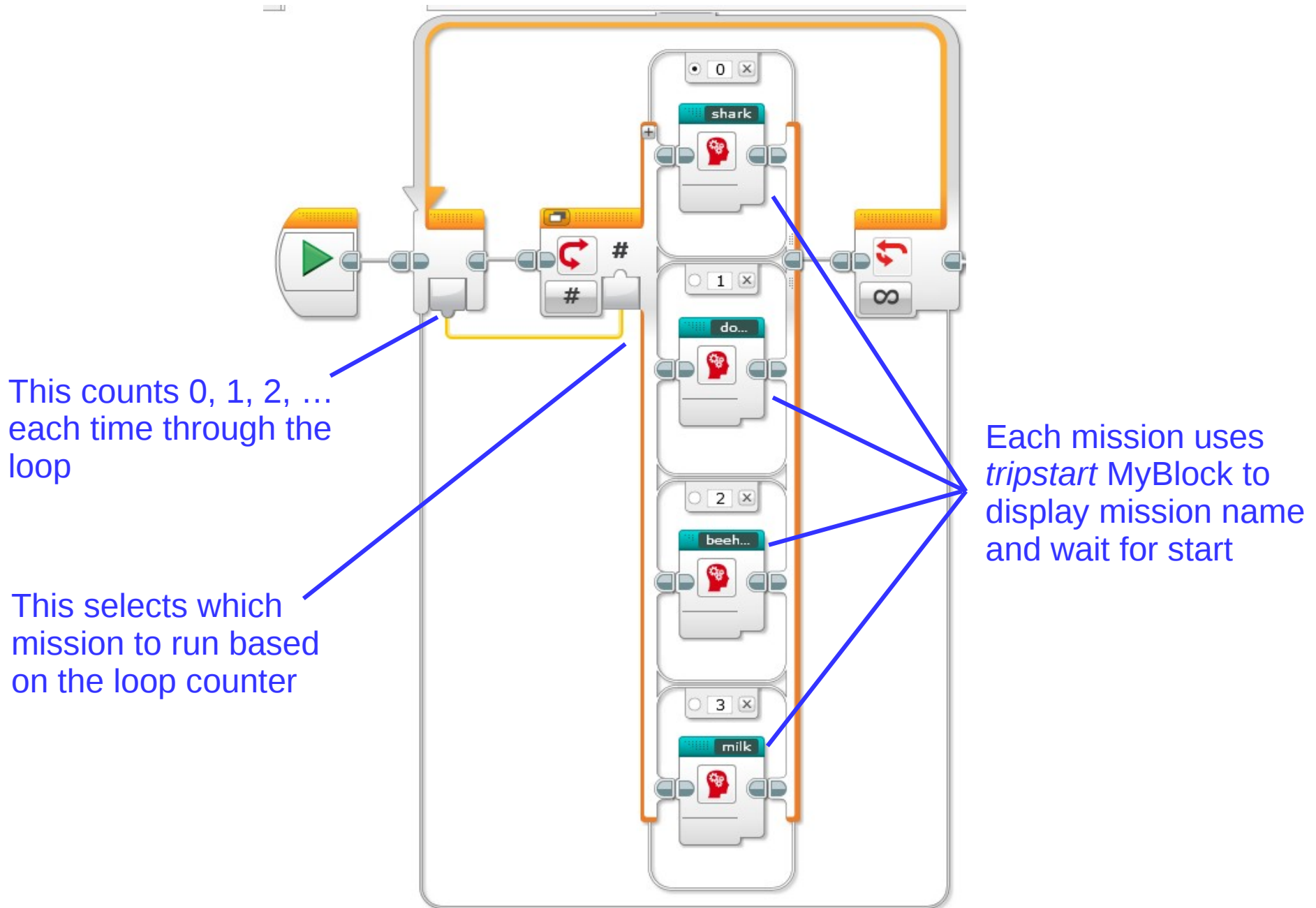
Each time through the loop will execute a different path of the Switch, starting with zero

Completing the loop/switch

Use the “+” button to add more options, then set the order to run missions:

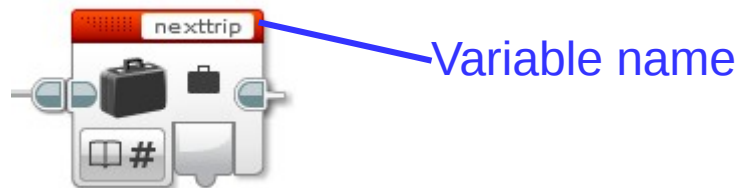


Review

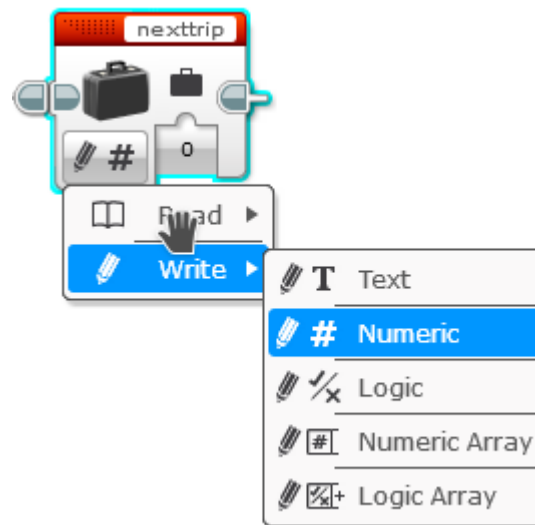


Variables

A *variable* is a place to store a value

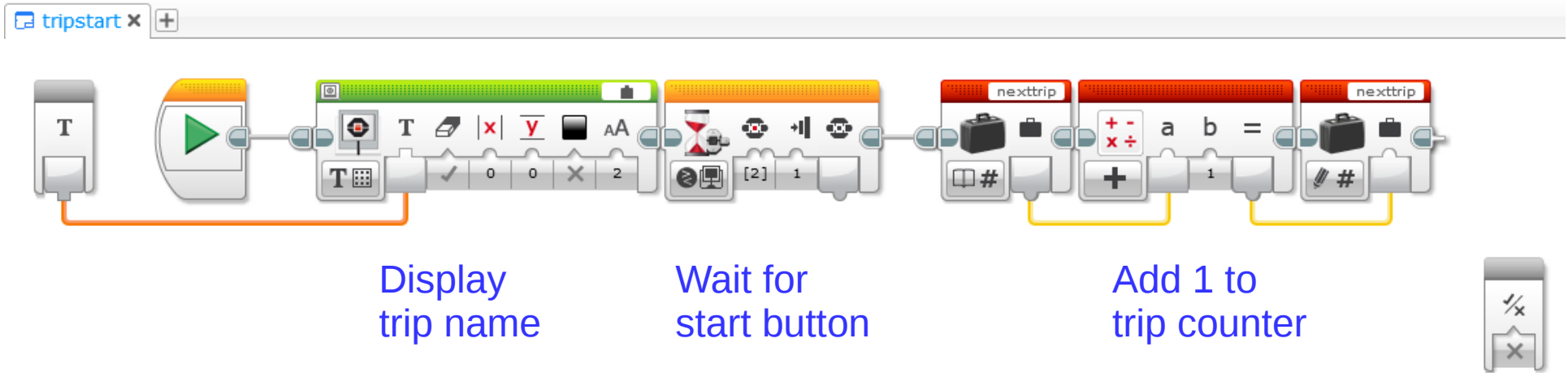


Each variable is given a name, a type, and whether it's being written or read



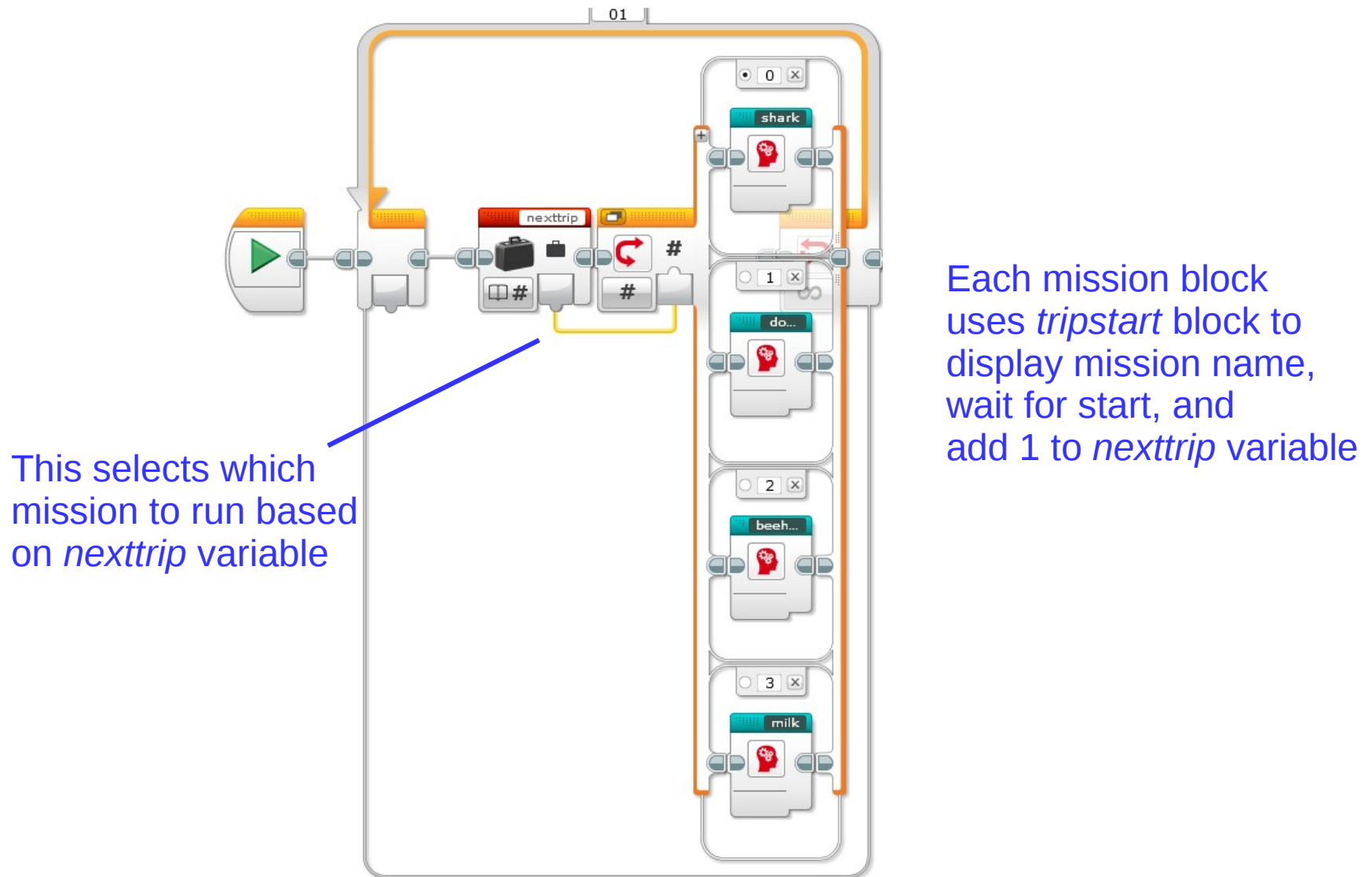
Trip counter

In the tripstart block, let's create a variable to keep track of the next trip to be run:



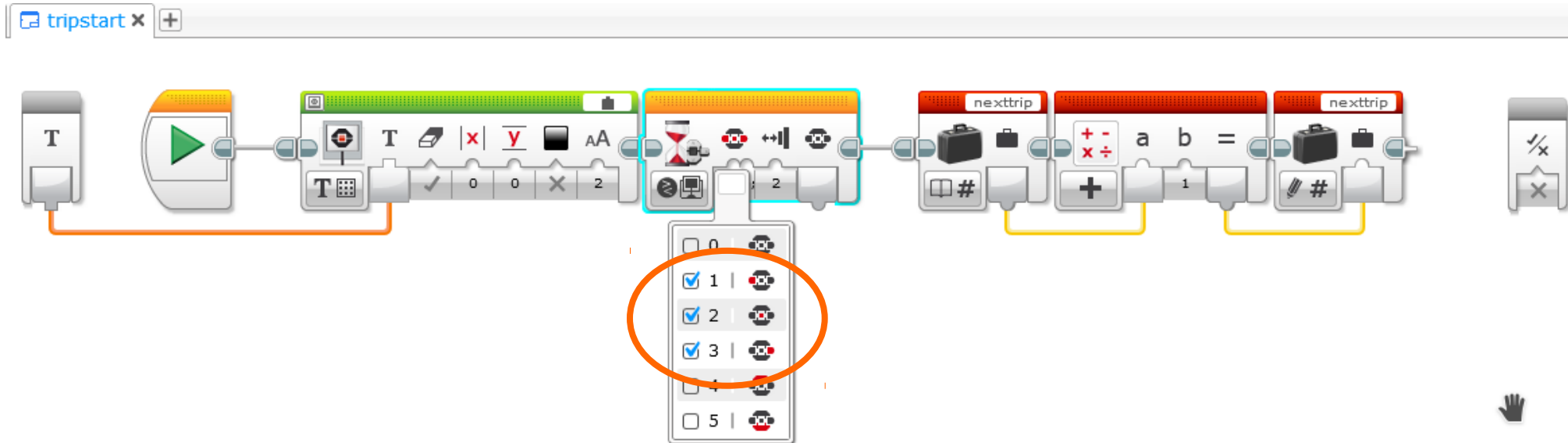
Rewiring the master loop

In the master loop, use the *nexttrip* variable to determine which mission to run next:



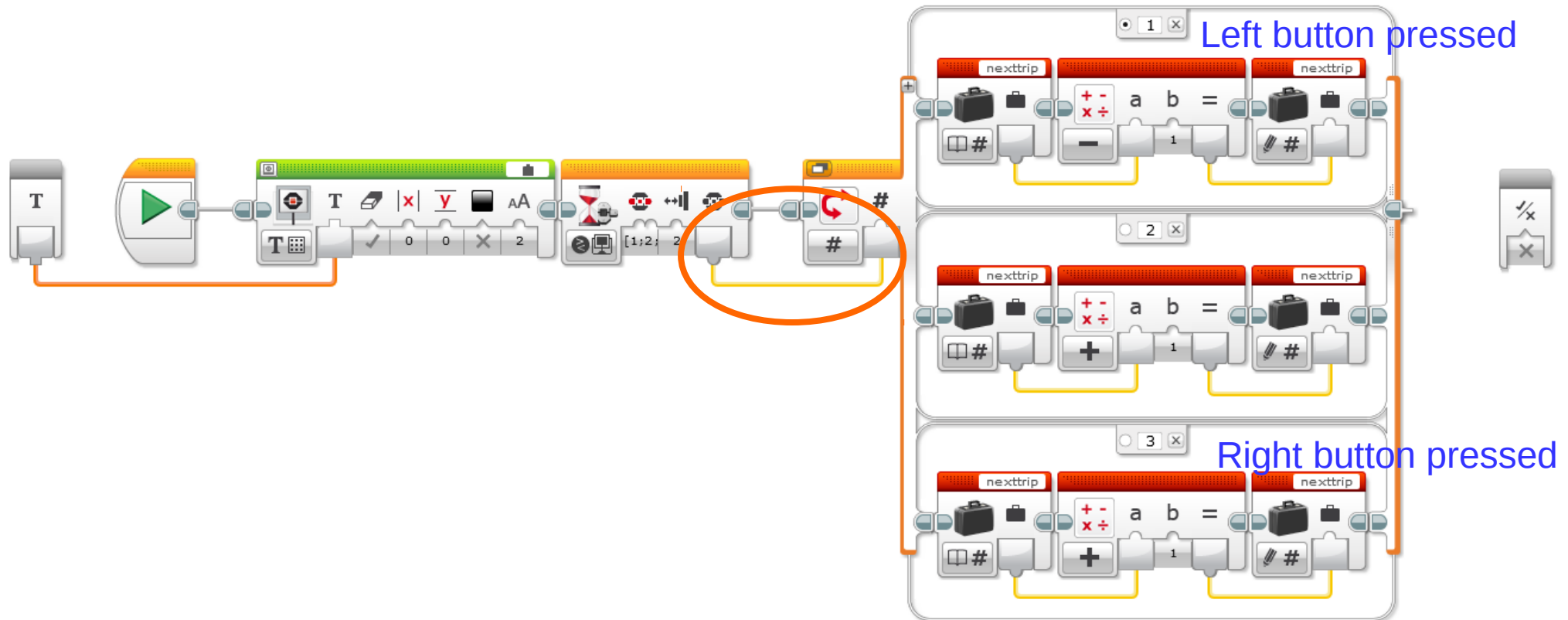
Enabling left/right buttons

In the tripstart block, change the Wait for Brick Button to accept left, center, and right buttons:



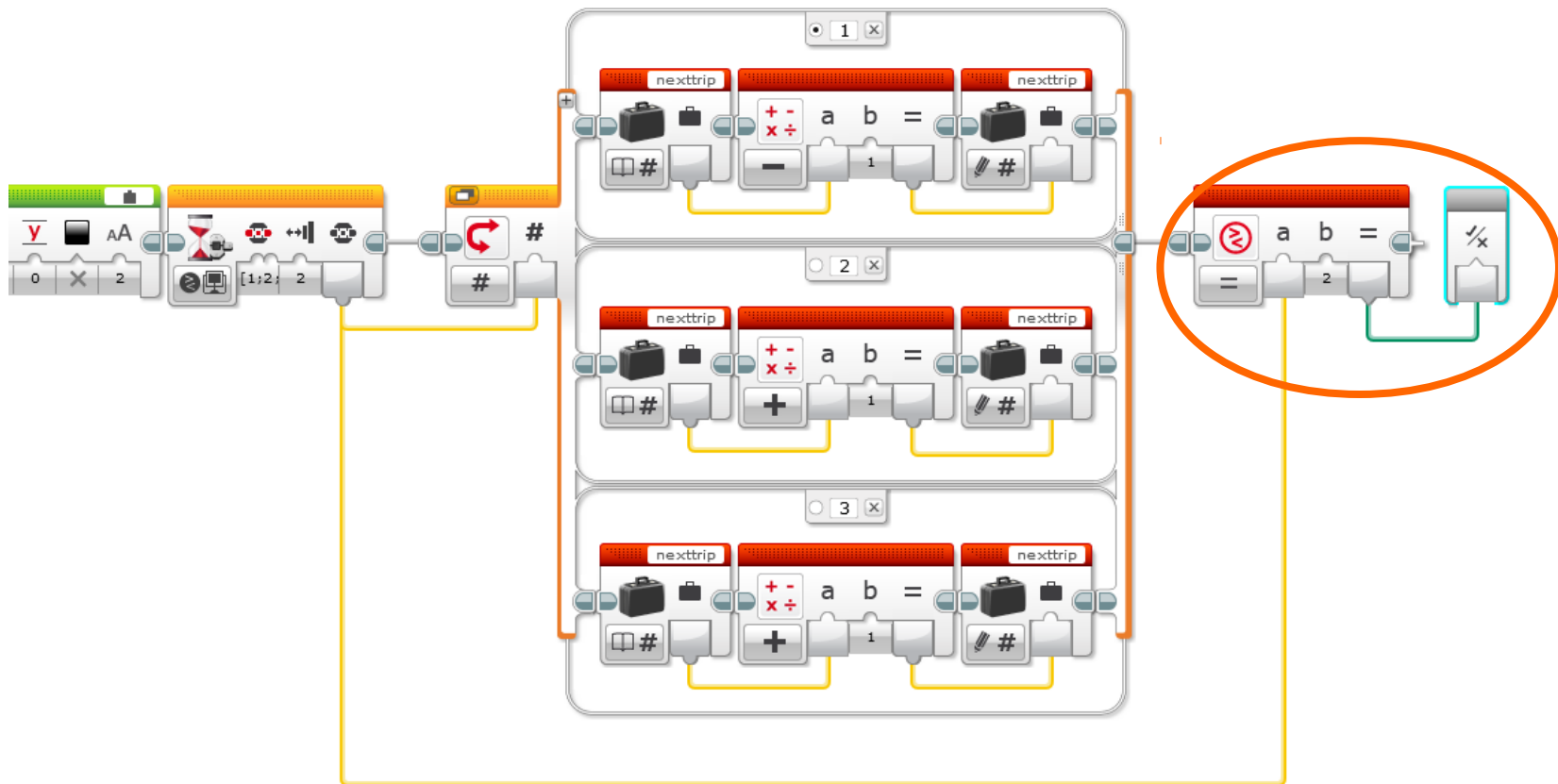
Enabling left/right buttons

If the button pressed is left (1), we want to reduce the trip counter; if it's center or right (2 or 3), we want to increase the trip counter

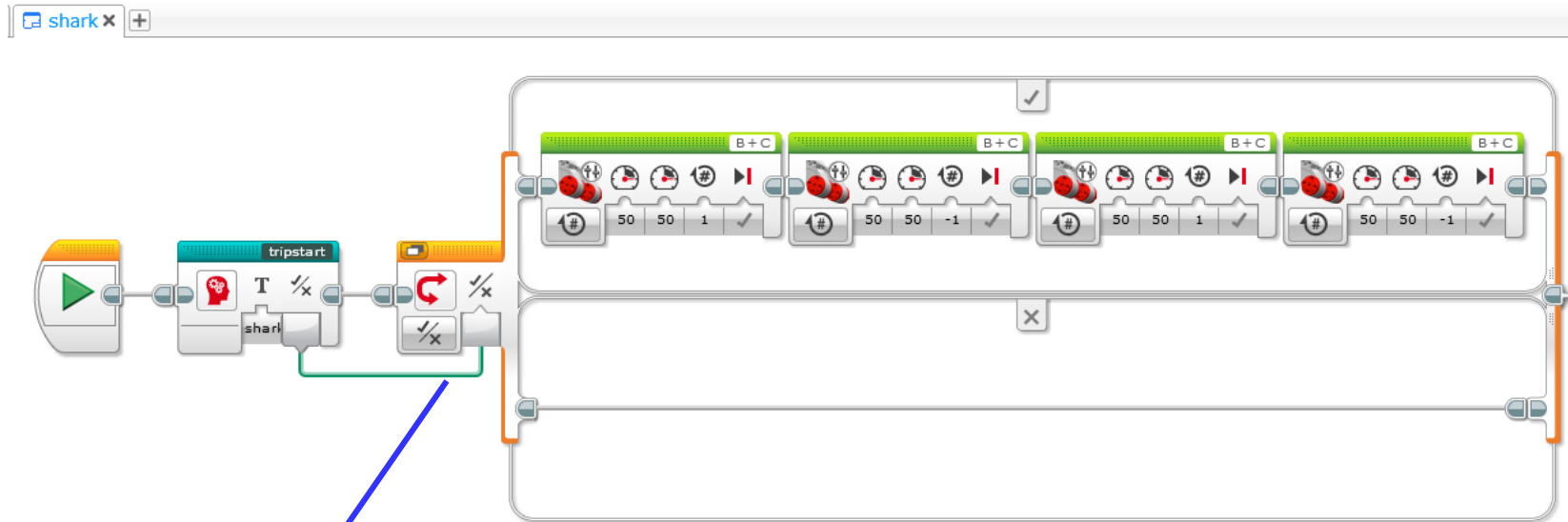


Only run mission if start is pressed

Finally, we want the mission to run only if the “start” button has been pressed:

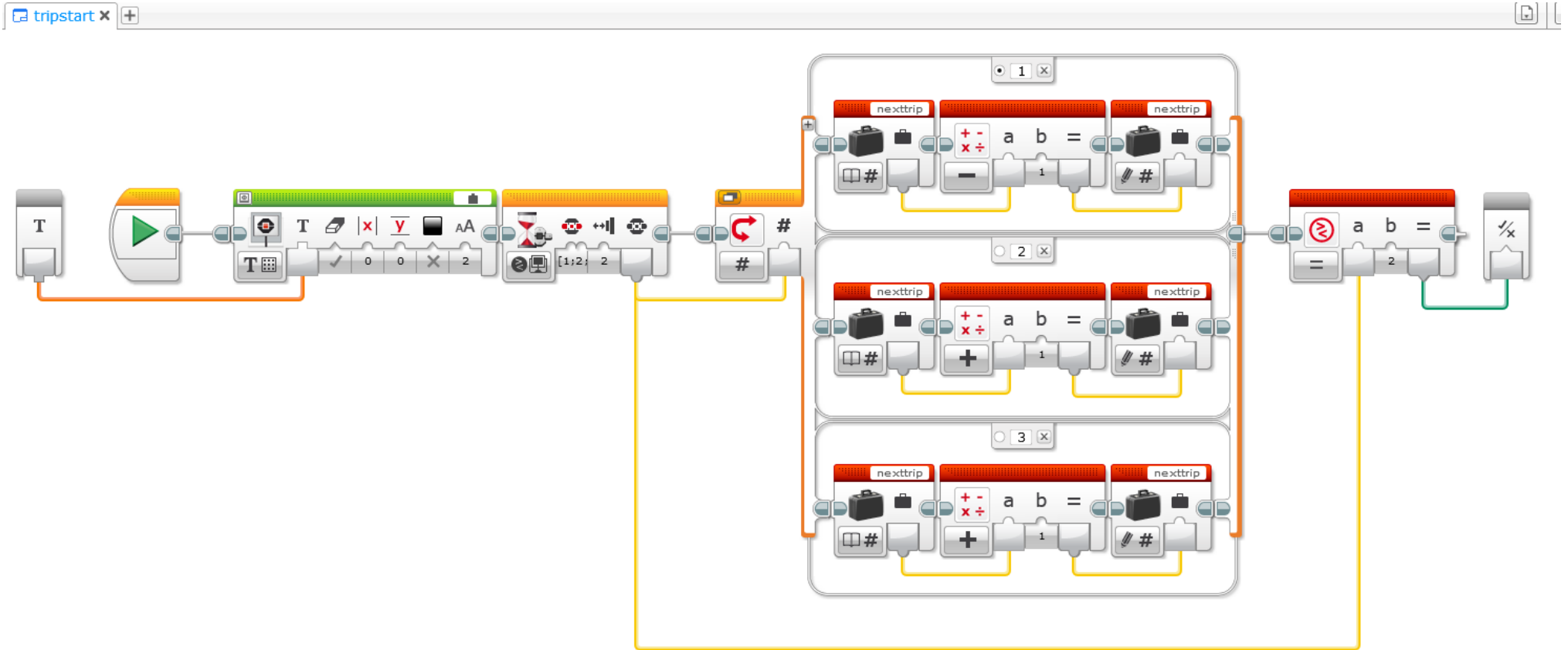


Only run mission if start is pressed

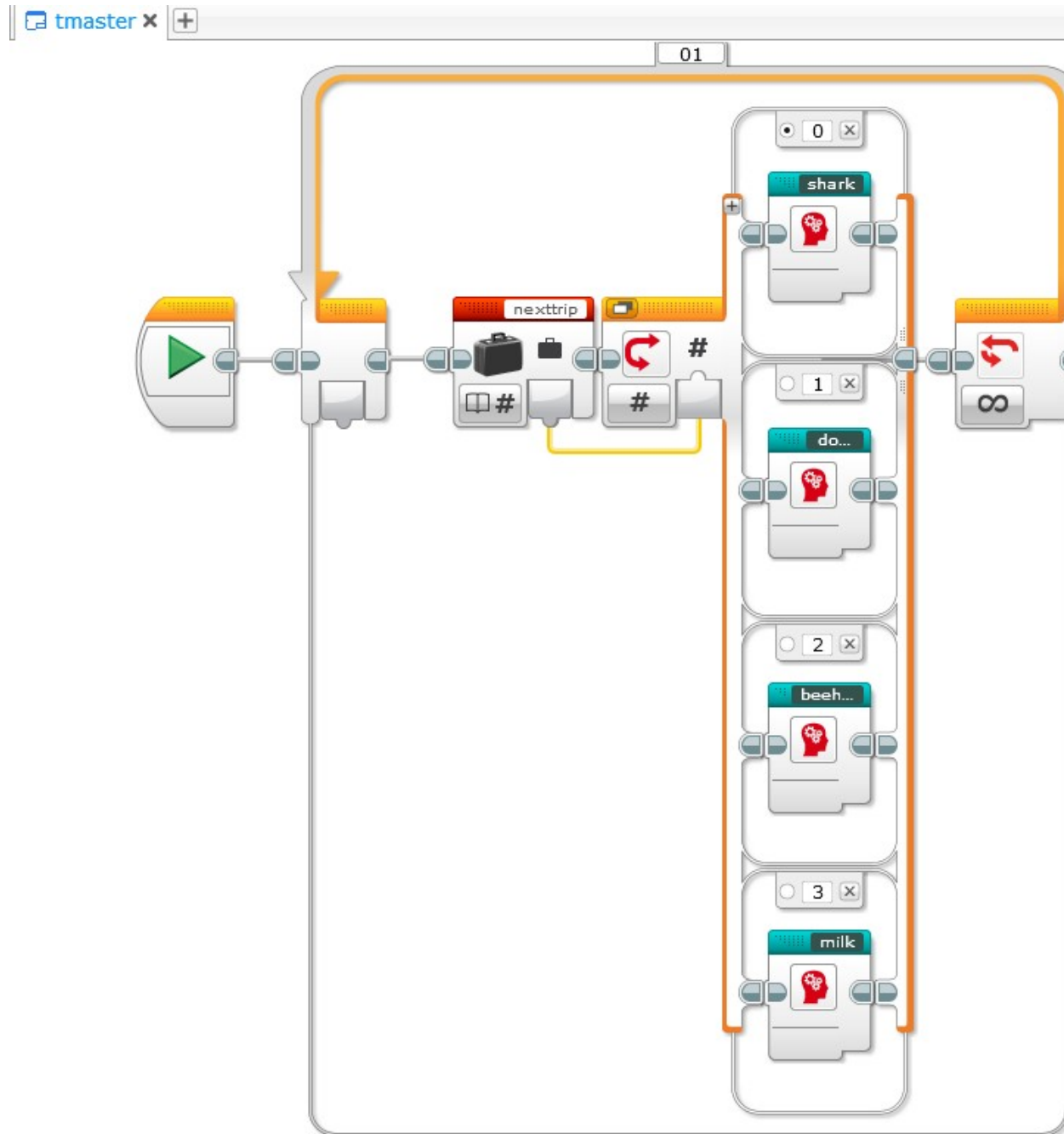


Add a "Logic" switch block to run mission steps only if "start" was pressed in tripstart

tripstart block



Master program



Thank you!

Questions?

Patrick R. Michaud
pmichaud@pobox.com
republicofpi.org

Join the NorthTexasFLL group!