FLL Workshop – Session 2
Beginning FLL Programming

Patrick R. Michaud
pmichaud@pobox.com

University of Texas at Dallas
Summer 2018

Goals

Learn basics of Mindstorms programming

Be able to accomplish some missions

Provide tips that have helped our team

Point out traps that cause frustration

Topics

EV3 programming environment

Programming basics

Motor blocks – moving forward and backward

Detecting lines, borders, colors

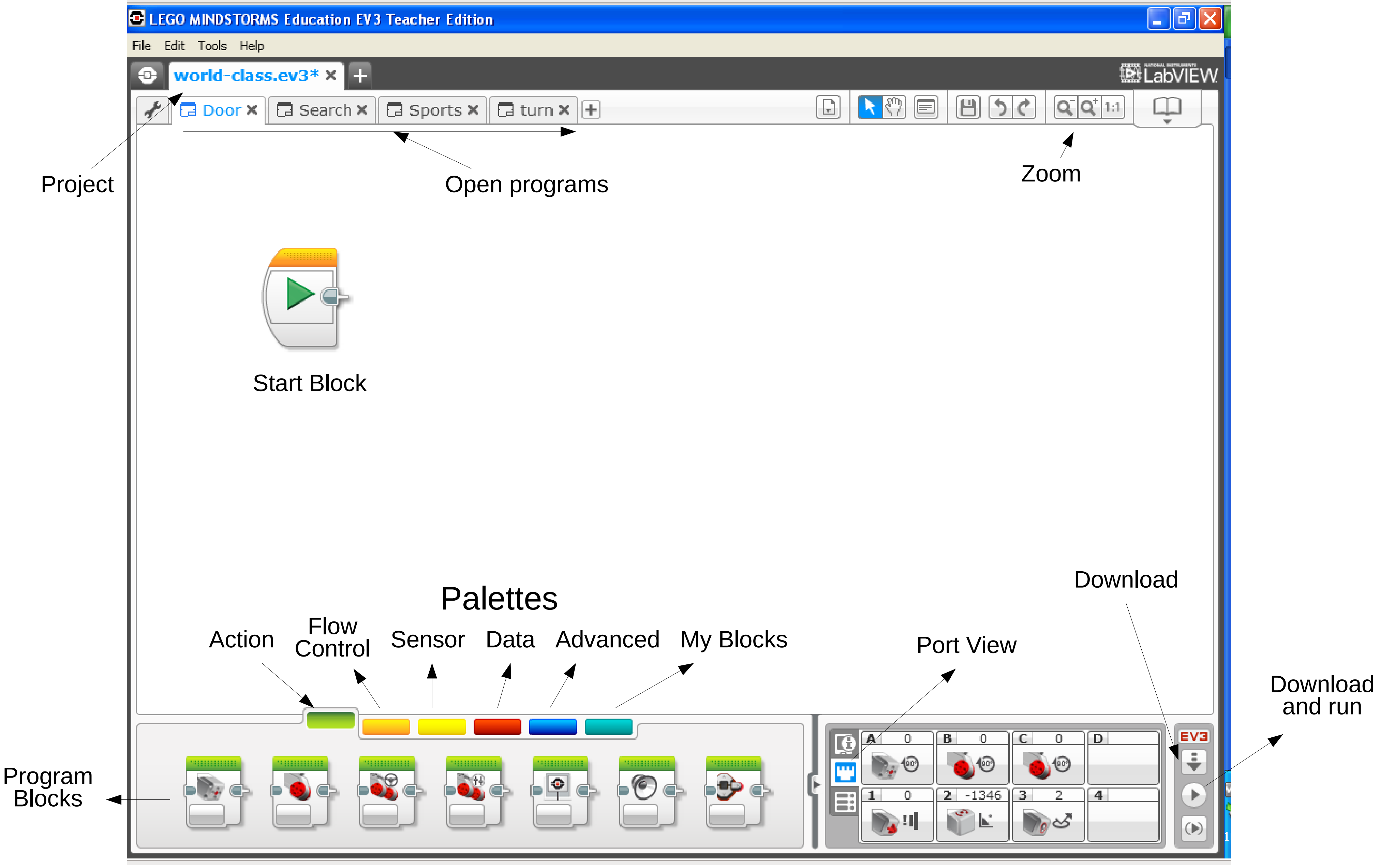
Port view and view mode

Turning basics

Turns using gyro sensor

Simple My Blocks

EV3 Programming Environment



Project

Open programs

Zoom

Start Block

Palettes

Action

Flow Control

Sensor

Data

Advanced

My Blocks

Port View

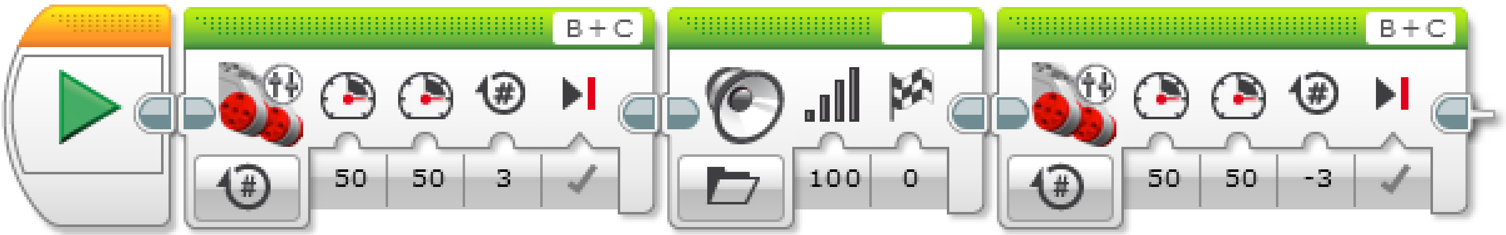
Download

Download and run

Program Blocks

Programming basics

A program is a sequence of actions to perform



Start Move forward 3 Play sound Move backward 3

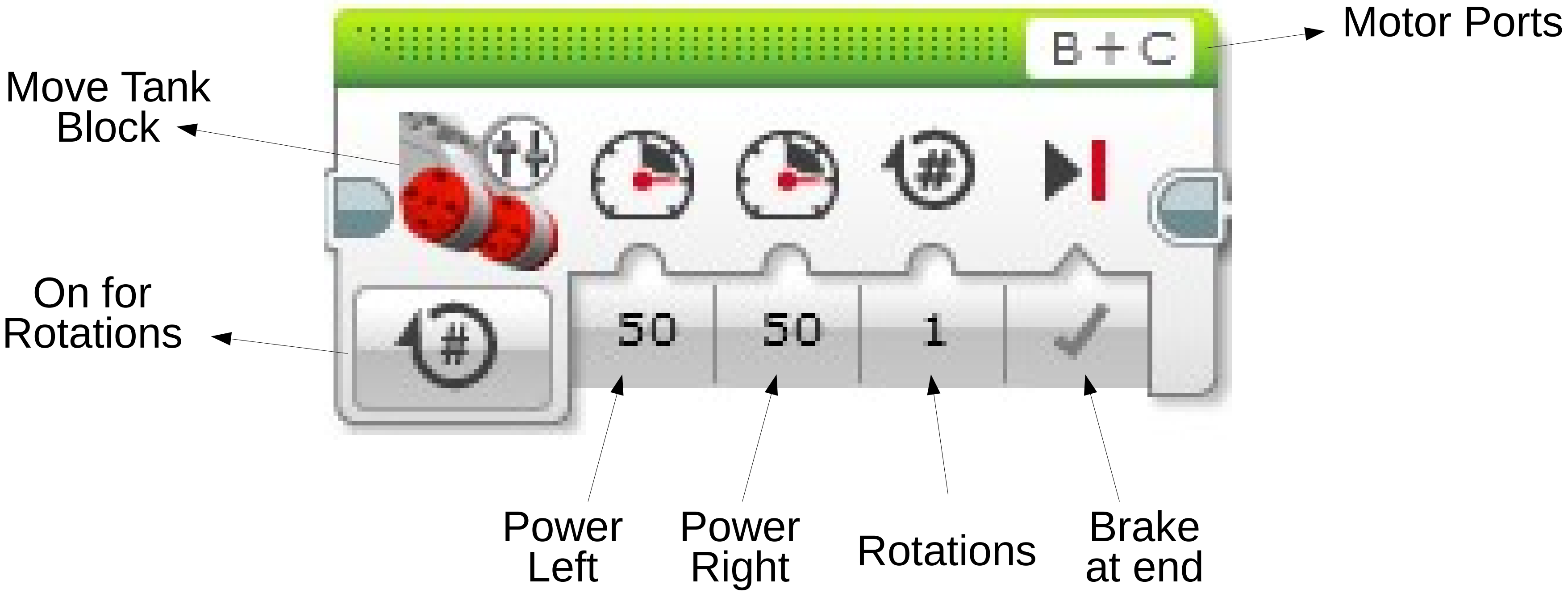
Each action is represented by a “block”

Icons indicate the type and values of block

Robot does *exactly* what you tell it to do
and *only* what you tell it to do

Programmer determines what actions to perform

Motor blocks – moving the robot



Motor Ports

Move Tank Block

On for Rotations

Power Left

Power Right

Rotations

Brake at end

Motor ports: which motors to turn

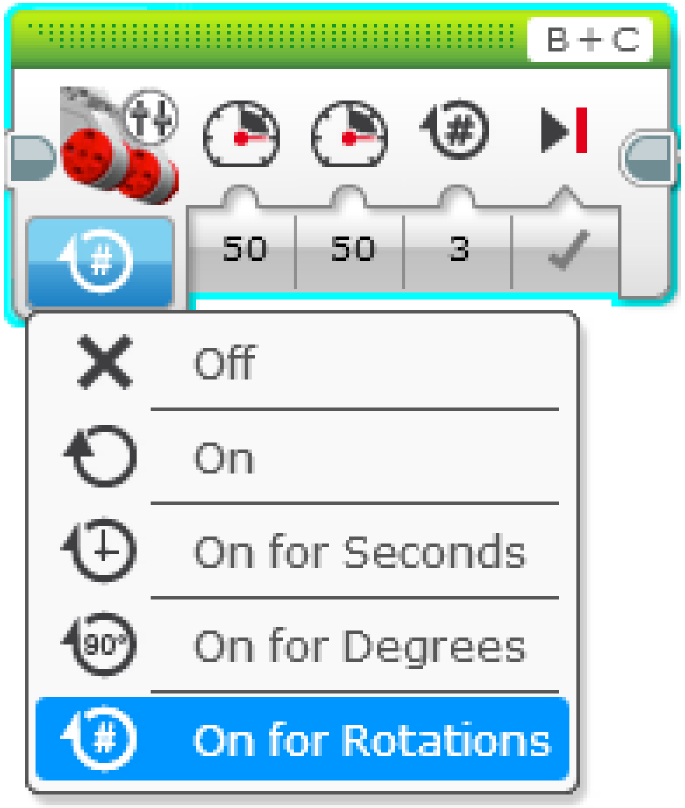
Power left/right: how fast to turn each motor

Rotations: how far to turn each motor

Brake at end: whether to apply brakes when done

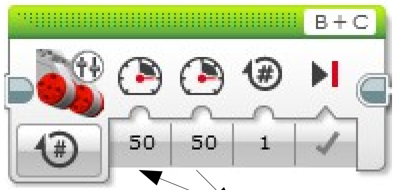
Motor blocks

Motor commands



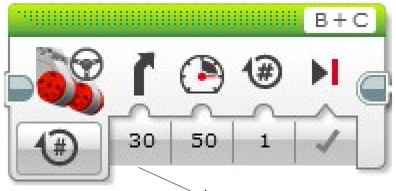
Types of moves

Move tank



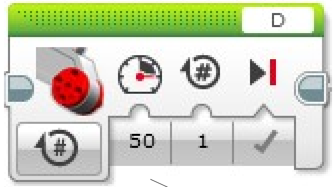
left/right motor speeds

Move steering



turn direction / sharpness

Large motor (one)



motor speed

Programming:
Move forward (and backward?)

Fundamentals of turns

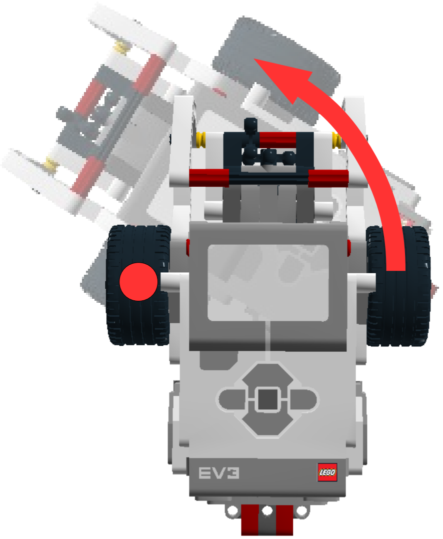
The robot turns when driving wheels move at different speeds

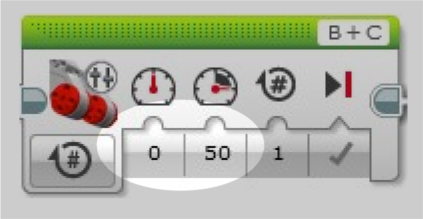
The robot turns towards the *slower* wheel

The greater the difference in speeds, the tighter the turn

Turning basics – types of turns

Pivot turn





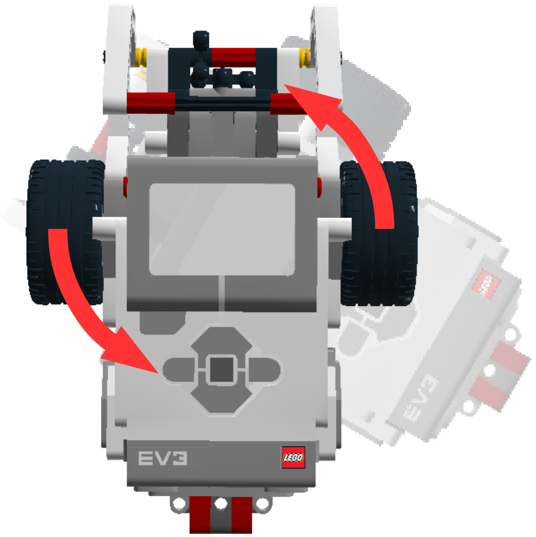
B+C

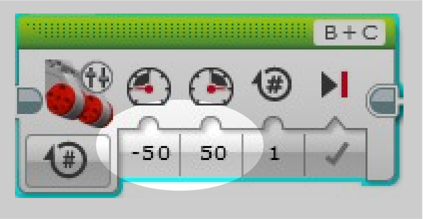
0 50 1

One wheel moves, other wheel brakes

This seems most consistent

Point turn



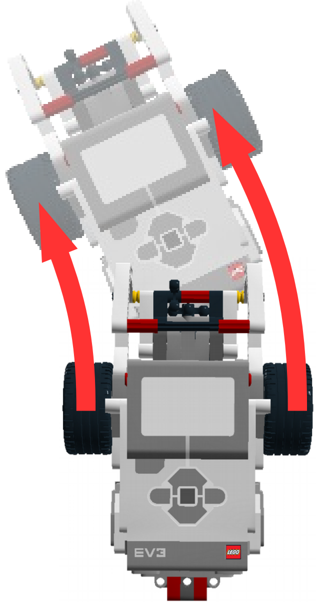


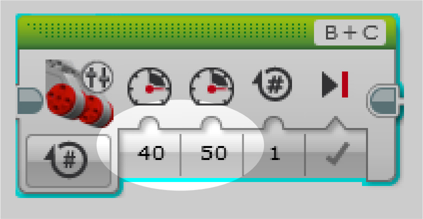
B+C

-50 50 1

Wheels move in opposite directions

Wide turn





B+C

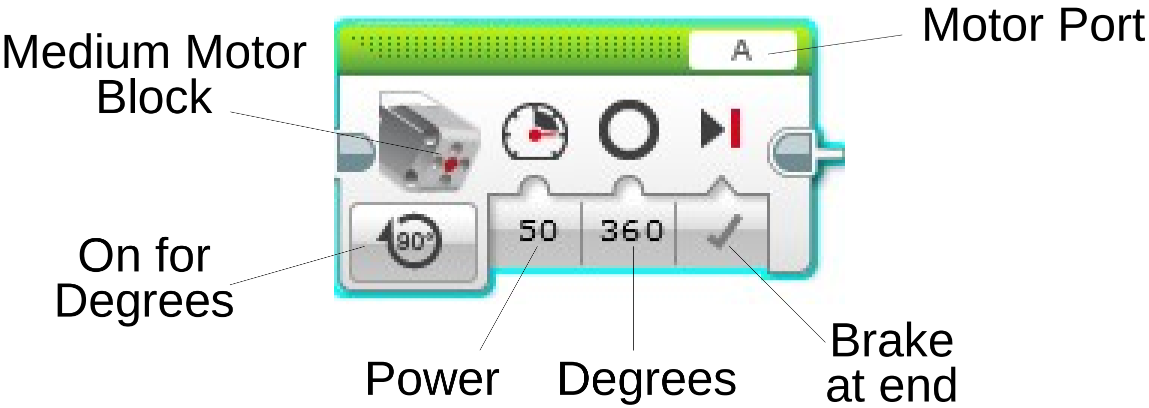
40 50 1

One wheel moves slower than other

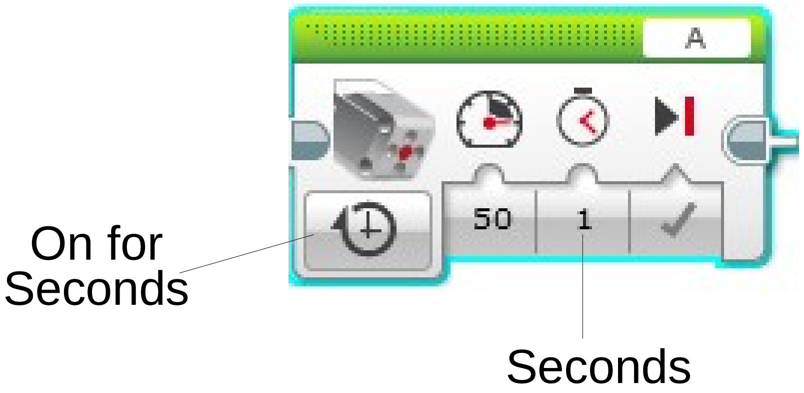
Programming:
Making the robot turn

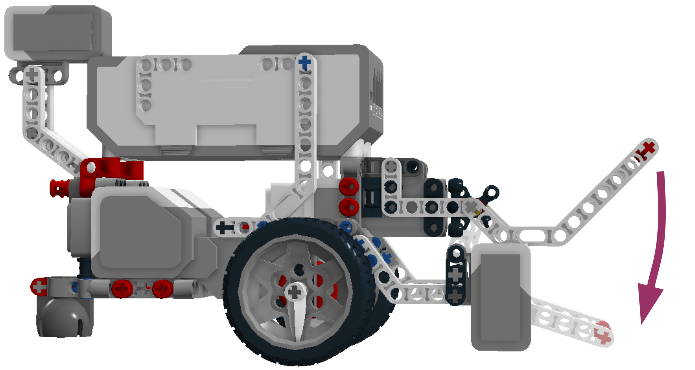
Using the medium motor

The medium motor has a different block



Use "On for Seconds" when you want to go to a limit (to avoid stalls)





Arm travel angle unknown

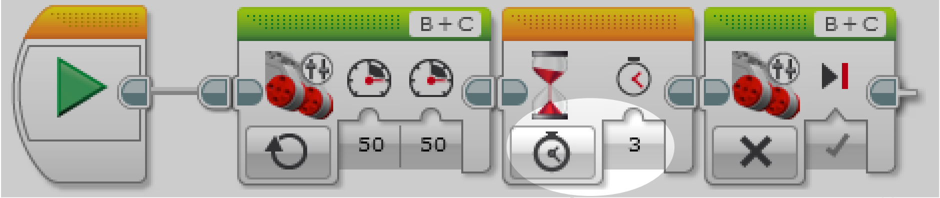
Wait blocks

Program executes blocks in sequence

Wait blocks makes program wait for something to happen before continuing to next block

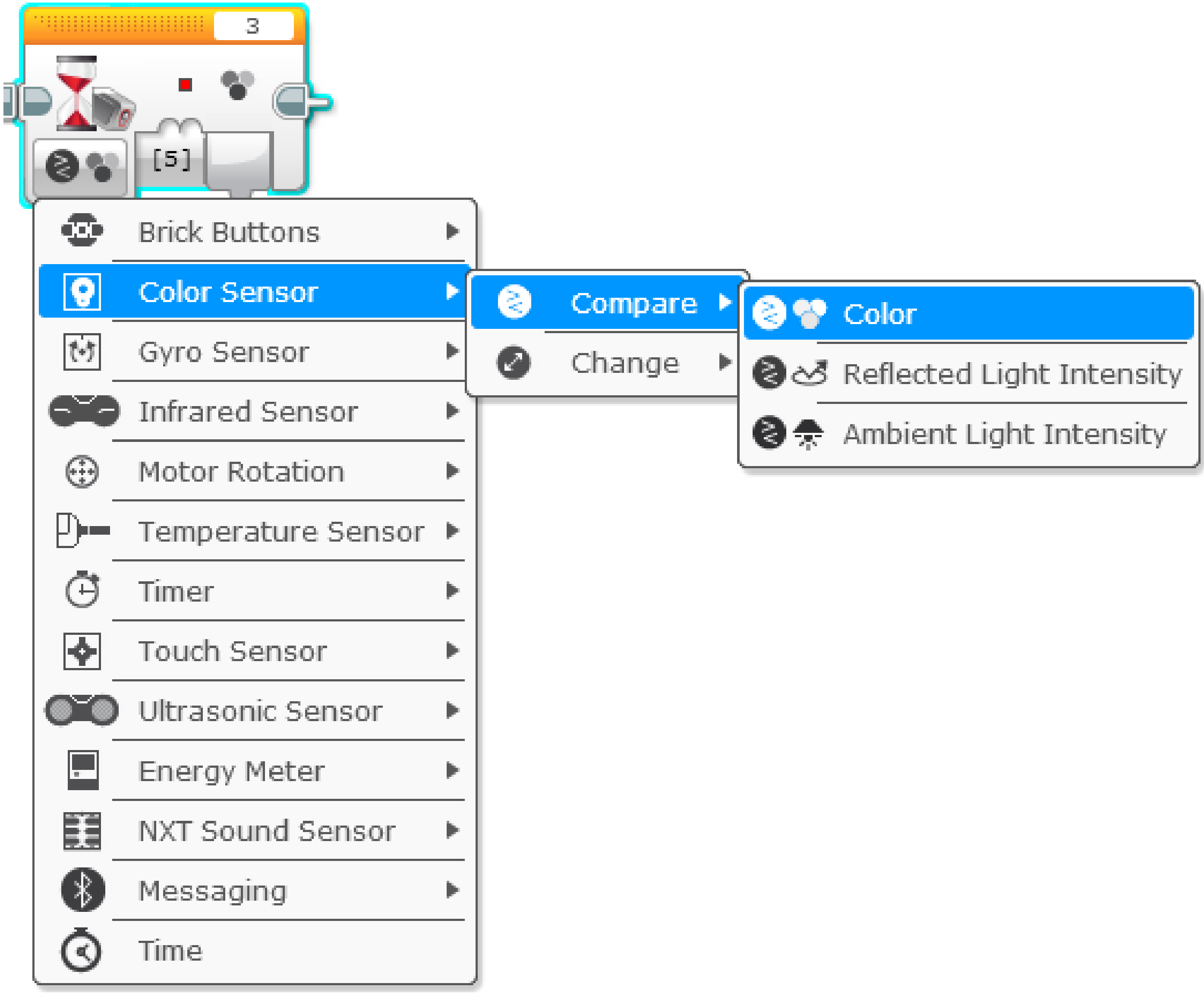
- Wait for seconds
- Wait for sensor to read a certain value
- Wait for sensor to change

Example:

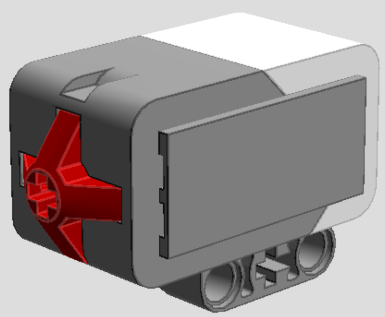


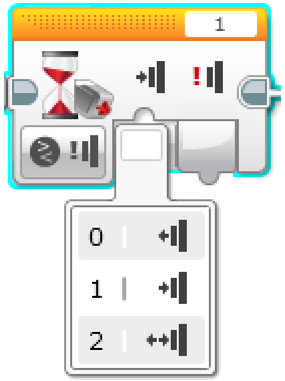
Start motors on wait 3 sec motors off

Wait block options

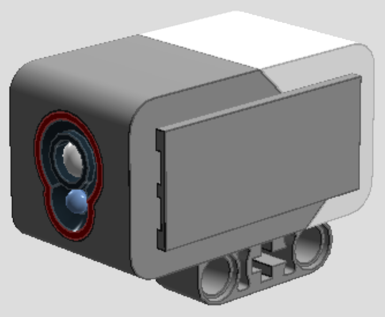


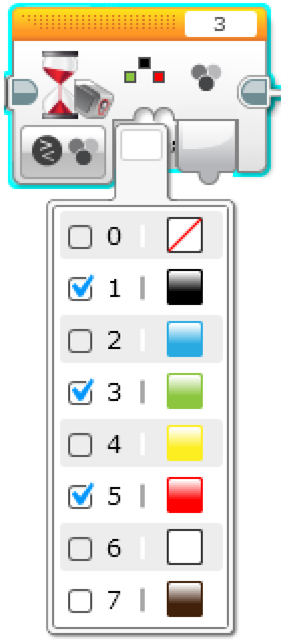
Touch and color sensors





Touch sensor detects pressure



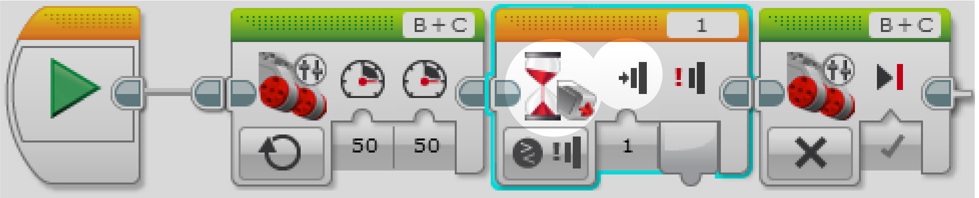


Color sensor can detect selected colors

Sensor should be 0.5 to 2.0 cm from color to be detected

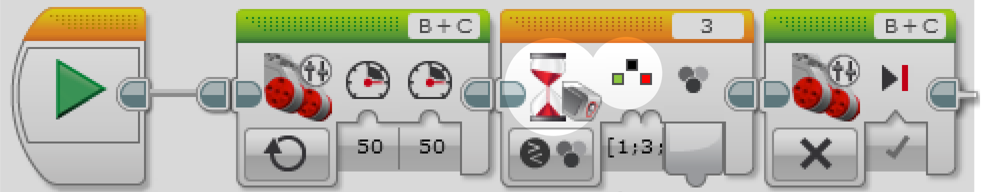
Moving to a border, line, or color

Move until touch sensor pressed



Start | motors on | wait for touch press | motors off

Move until black, green, or red detected



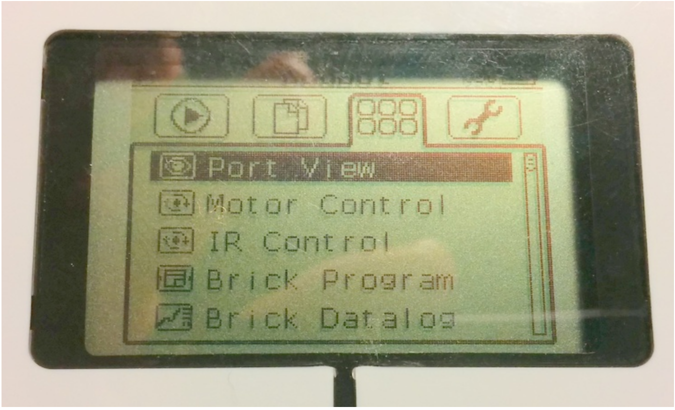
Start | motors on | wait for black, green or red | motors off

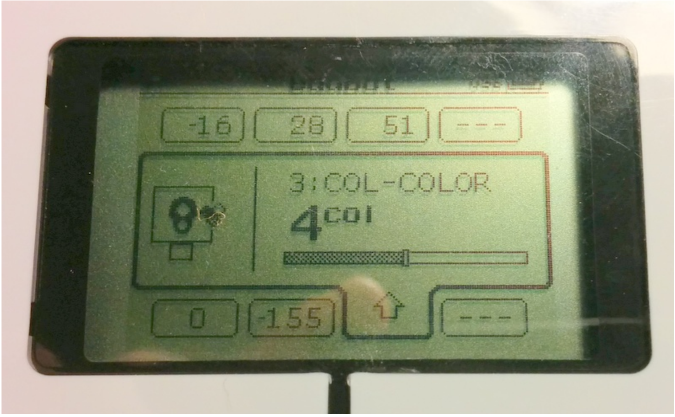
Programming:
Wait for color / touch

Use “port view” to see sensor changes

Software view:

EV3 view:

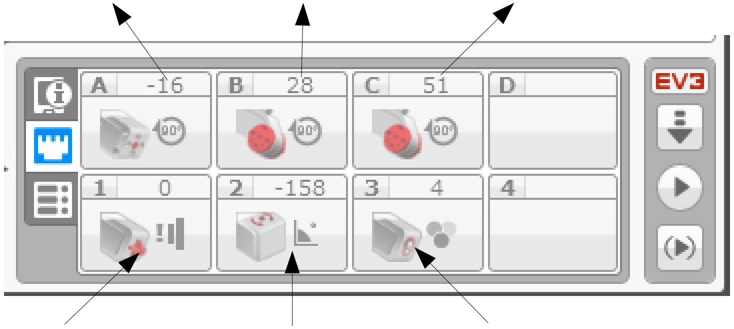




Motor A
-16 degrees

Motor B
28 degrees

Motor C
51 degrees



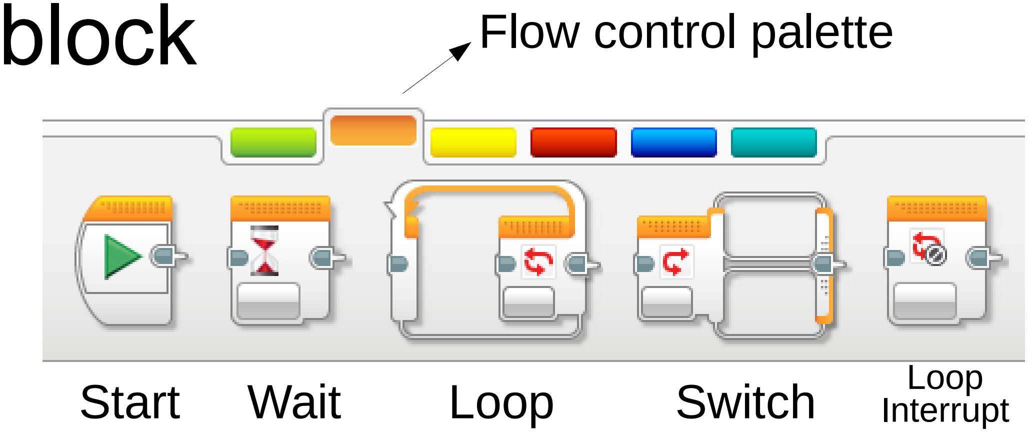
Touch 1
not pressed

Gyro 2
-158 degrees

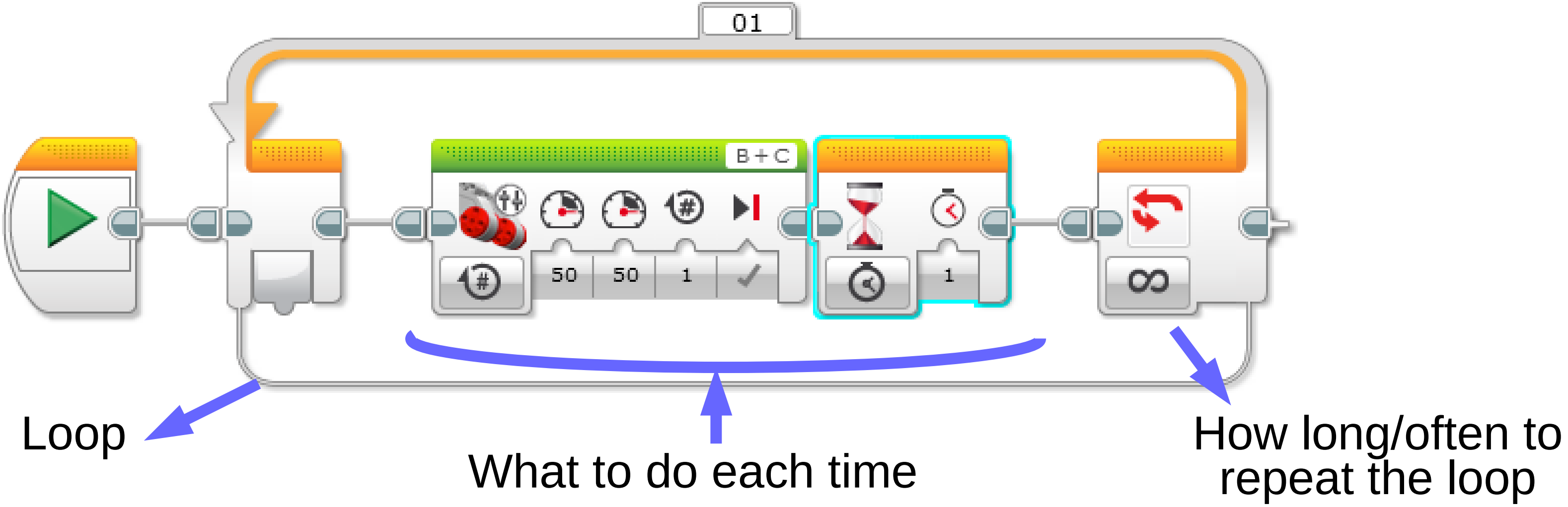
Color sensor 3
4 (yellow)

Loops

To do something repeatedly (like steering), use a "loop" block



A basic loop block



More about turning

Quiz: If a wheel turns 90 degrees, does the robot turn 90 degrees?

Answer: No

There *is* a formula to convert wheel turn angles to robot turn angles (advanced programming)

But many teams just use trial and error

Or, use a gyro sensor

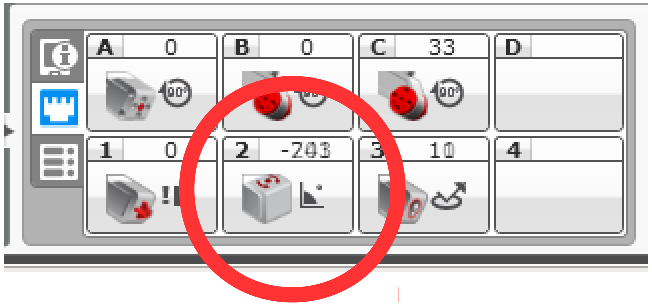
Gyro sensor

Gyro sensor detects rotation about an axis

Can measure rotation rate or rotation angle

Sensor has *bias* and *drift*

sometimes shows movement even when still

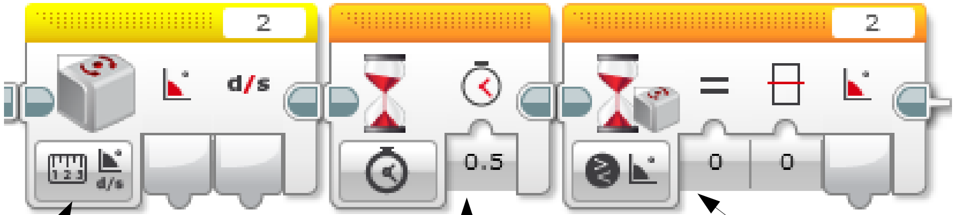


A 0 | B 0 | C 33 | D

1 0 | 2 -203 | 3 10 | 4

Reducing gyro drift

The following block sequence recalibrates the gyro sensor to eliminate drift:



Measure – Angle and Rate

Wait 0.5 sec

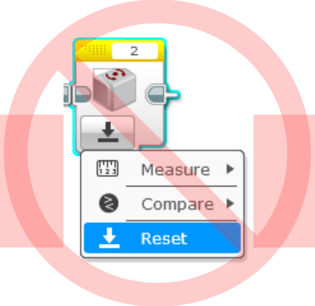
Gyro sensor – Compare – Angle equal to 0

Perform this once at beginning of program

Requires 2-3 seconds to complete

Gyro must be stationary while calibrating

Trap: “Gyro reset” block doesn't recalibrate gyro!



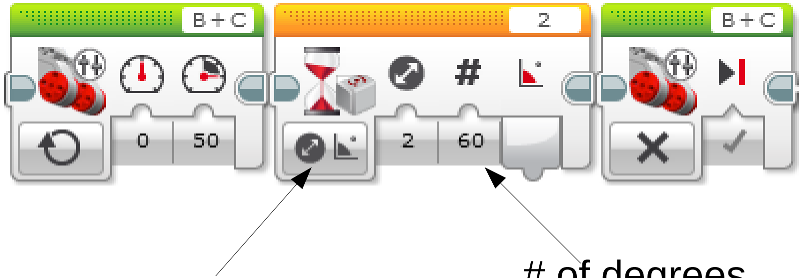
Turns using gyro sensor

Once gyro reads correctly, turning an angle is:

turn on
one motor

wait for gyro
to change 60°

turn off
motors



Gyro – Change – Angle

# of degrees
to change

Gyro sensor is *not* precise

Robot may turn slightly more or less than 60°

My Blocks

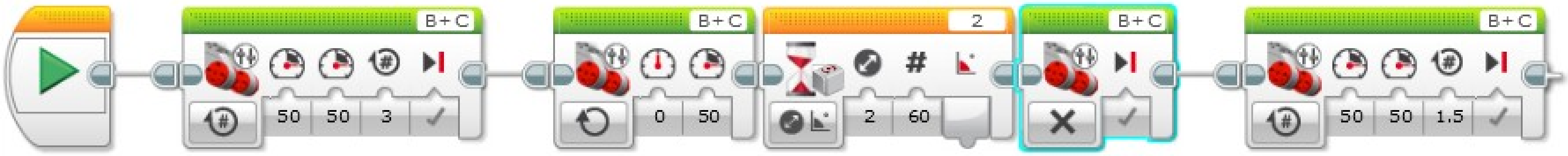
Simple My Blocks - turning

A program to move, turn, then move:

move fwd 3 rot

turn robot 60° to left

move fwd 1.5 rot

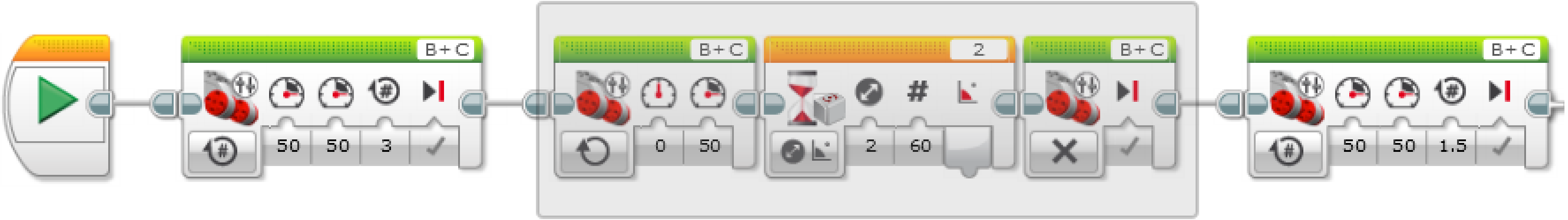


Adding three blocks for every turn will get tedious

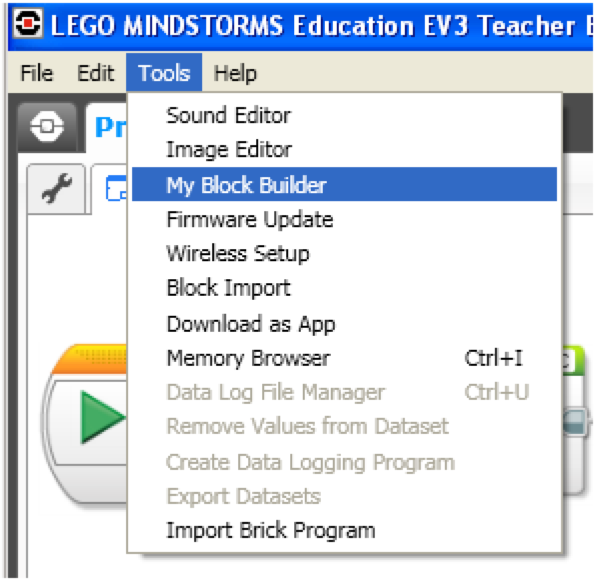
We can create custom blocks called “My Blocks”

Creating a My Block

Start by drag-selecting the blocks to be used

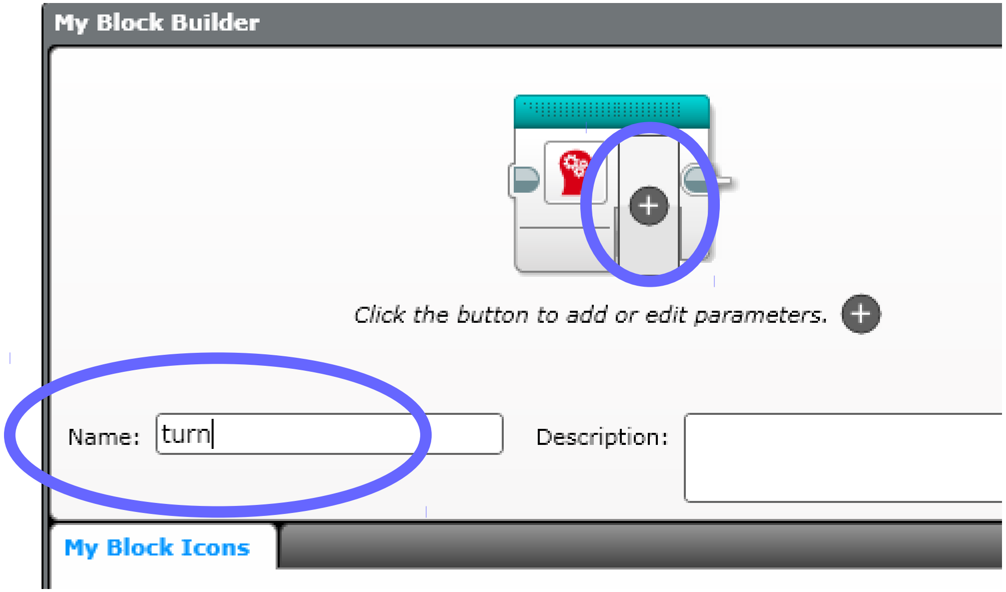


Select "My Block Builder" from the Tools menu



Creating a My Block

- Give the My Block a name
- Click the “+” to add a parameter
- Click “Parameter Setup”
  - Name parameter “Power”
  - Give it a default of 50

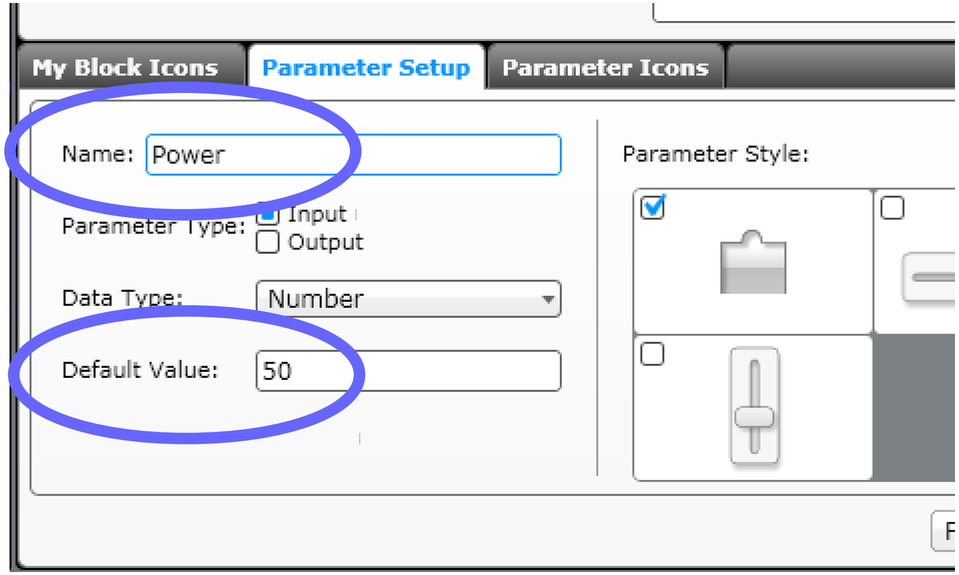


My Block Builder

Click the button to add or edit parameters. +

Name: turn

Description:



Name: Power

Parameter Type: ☑ Input □ Output

Data Type: Number

Default Value: 50

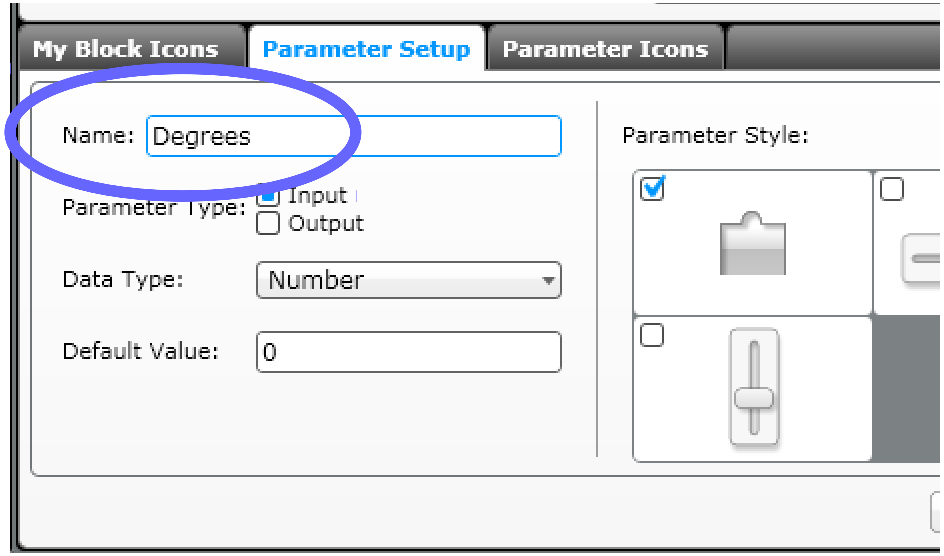
Parameter Style: ☑ □ □

Creating a My Block

Click "+" again to create another Parameter

Name it "degrees"

Default value 0



Name: Degrees

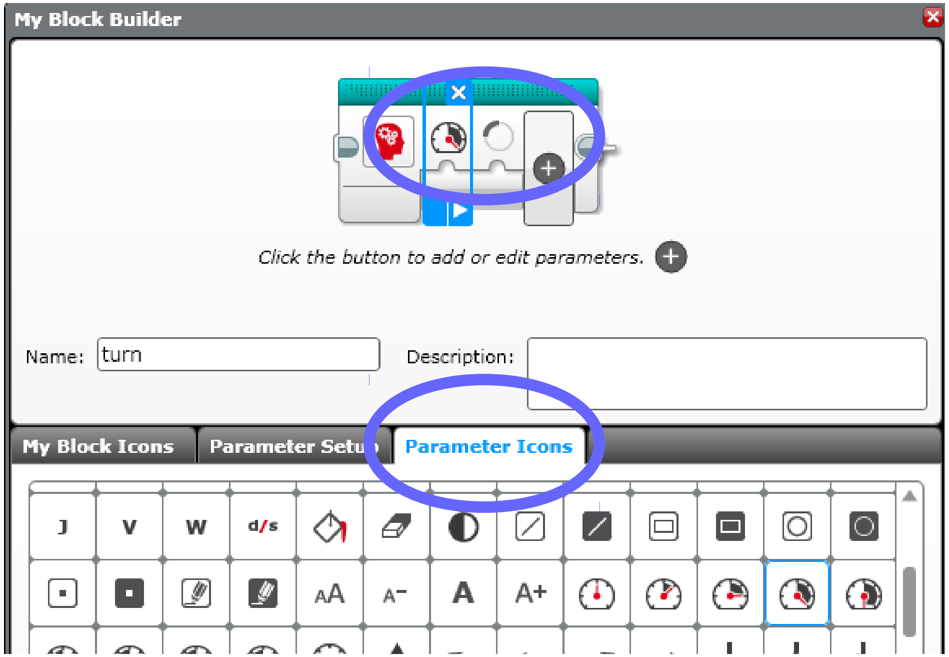
Parameter Type: ☑ Input ☐ Output

Data Type: Number

Default Value: 0

You can change the parameter icons to be more meaningful

Click "Finish"



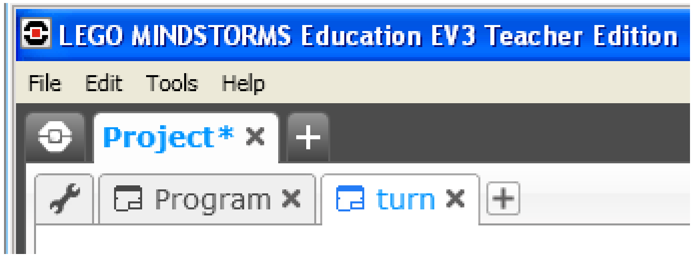
Click the button to add or edit parameters.

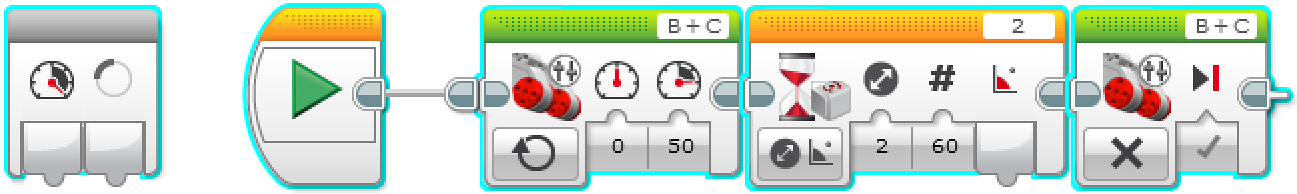
Name: turn

Description:

Creating a My Block

A new program is created for the My Block



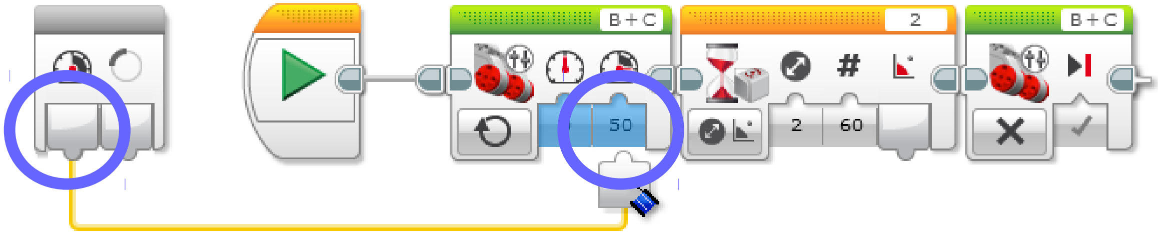


The grey block on the left has our “parameter inputs” for power and degrees

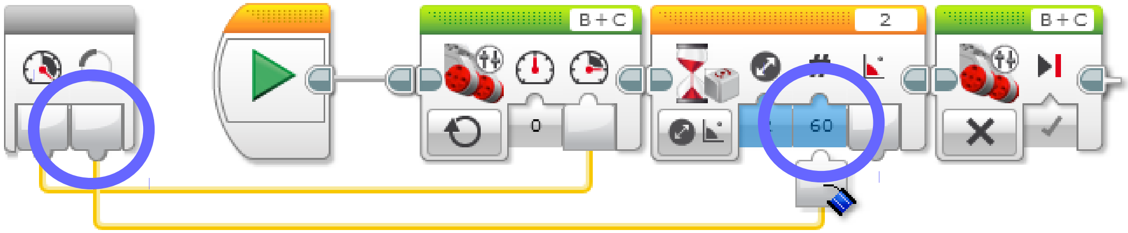
We have to wire them to the appropriate blocks

Creating a My Block

“Drag” a wire from the “Power” parameter to the C motor power input

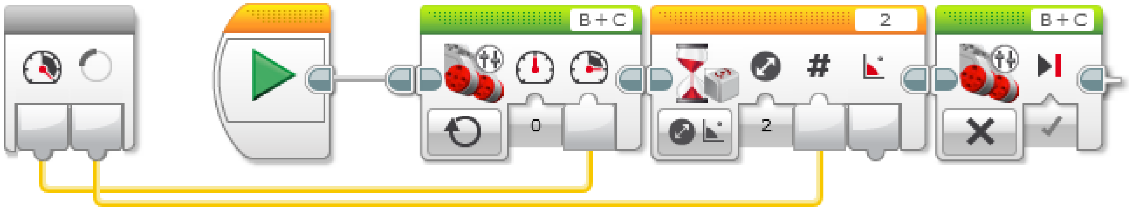


Drag a wire from the “Degrees” parameter to the degrees input of the wait block



Creating a My Block

We now have a "My Block" for turns

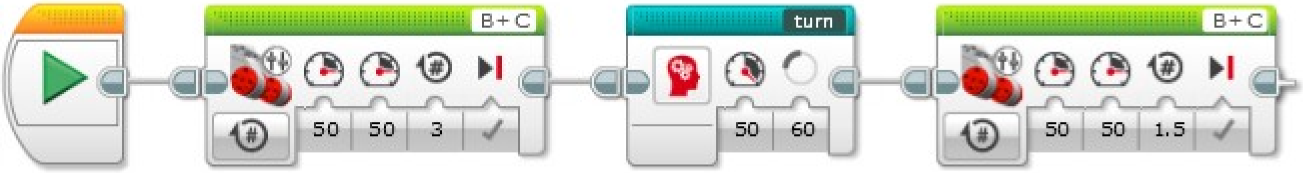


Our original program has the three-block sequence replaced with our "turn" block

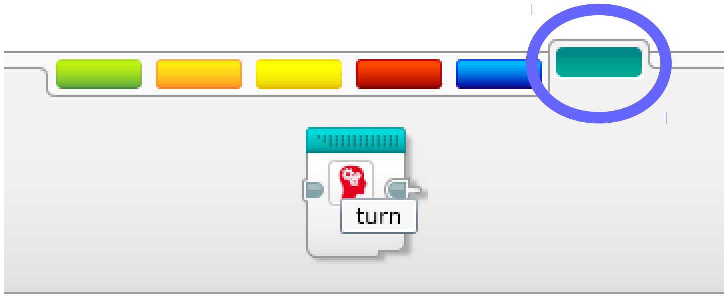
move fwd 3 rot

turn robot
60° to left

move fwd 1.5 rot

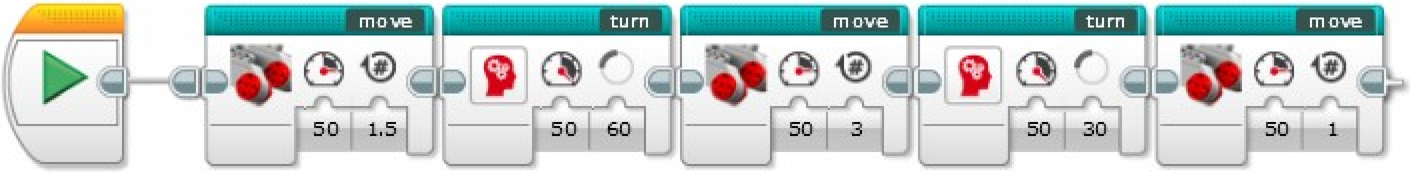


Our turn block appears in the My Blocks palette



My Blocks

Easier to read and build program



Suggestions:

- Create a My Block for each "mission"
- Combine mission My Blocks into "trip" My Blocks
- Create standard My Blocks for
  - Moving forward and backward, turning
  - Stopping
  - Motorized attachments
  - Initialization

Stop stop stop

Trap: Be sure the robot comes to a full stop between moves

When Move blocks complete, they brake the motors

Inertia carries the robot further, so the motors have to back up a bit

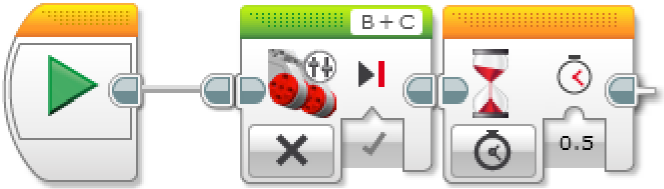
This takes a little time

If your program immediately goes to next action, robot will be inconsistent

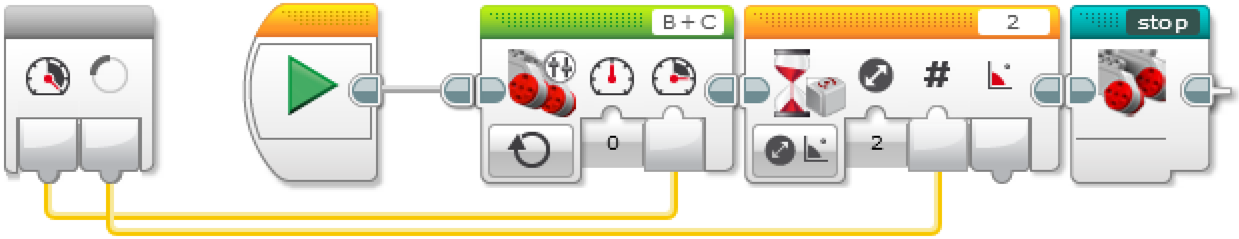
Create a "stop" My Block

Add a short wait after every movement

A simple "stop" My Block:



Can use this whenever you want to be sure the robot is stopped:



Thank you!

Questions?

Patrick R. Michaud
pmichaud@pobox.com

Join the NorthTexasFLL group!