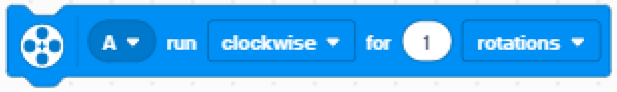
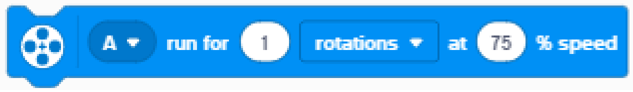
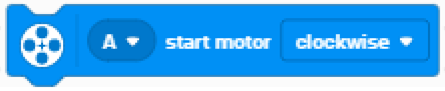
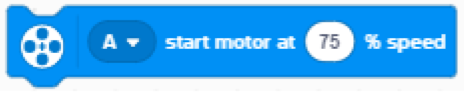
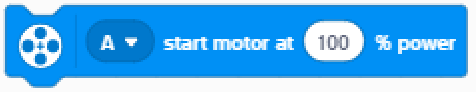
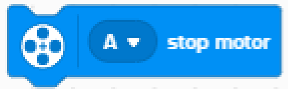
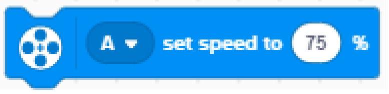
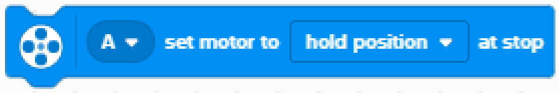
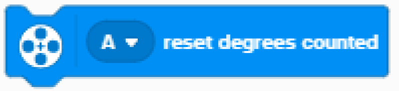
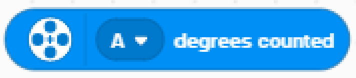
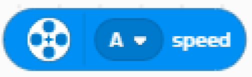
MOTOR BLOCKS MOTORS	Motor blocks make a single motor run or get information from a single motor.
RUN MOTOR FOR DURATION  A run clockwise for 1 rotations	Runs one motor clockwise or counterclockwise for a specified number of rotations, degrees, or seconds. Motor speed is set by the SET MOTOR SPEED block. The default speed is 75%.
RUN MOTOR FOR DURATION AT SPEED  A run for 1 rotations at 75 % speed	Runs one motor clockwise for a specified number of rotations, seconds, or degrees at a specified speed. A negative speed value runs the motor counterclockwise.
START MOTOR  A start motor clockwise	Runs one motor clockwise or counterclockwise until the program ends or until another block tells the motor do something else. The motor speed is set by the SET MOTOR SPEED block. The default speed is 75%.
START MOTOR AT SPEED  A start motor at 75 % speed	Runs one motor at a specified speed until the program ends or until another block tells the motor to do something else. A negative speed value runs the motor counterclockwise.
START MOTOR AT POWER  A start motor at 100 % power	Runs one motor at a specified power level until the program ends or until another block tells the motor to do something else. This block is different from blocks that use SPEED. Those blocks try to increase or decrease power to maintain the speed that is set. This block gives a fixed power level and will not adjust it. A negative power value runs the motor counterclockwise.
STOP MOTOR  A stop motor	Stops one motor from running. By default, the motor will brake so that it quickly comes to a complete stop. The motor will hold its position once stopped. You can change what happens when a motor stops with the SET MOTOR STOP ACTION block.

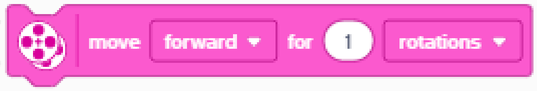
SET MOTOR SPEED  	Sets the speed of one motor. The speed range is -100 to 100. Negative values will reverse the direction of the motor. This will NOT turn on the motor. It must be used with other motor blocks. The default speed is 75%.
SET MOTOR STOP ACTION  	Sets the action that the motor will perform when a motor command completes or when the motor is stopped. It can be set to float (coast) or actively hold position (brake).
RESET MOTOR DEGREES  	Resets the degree count of a motor to zero. The degree count is equal to the motor's relative position from where it was at the start of the program or when it was connected. Turning the motor clockwise increased the count, while turning counterclockwise decreases the count.
READ MOTOR DEGREES COUNTED  	Reports the number of degrees a motor has turned since the start of the program or since it was last reset using the RESET MOTOR DEGREES block. Turning the motor clockwise increased the count, while turning counterclockwise decreases the count.
READ MOTOR SPEED  	Reports the current speed of the motor. The value given is the motor's actual speed, not the speed set by the SET MOTOR SPEED block.

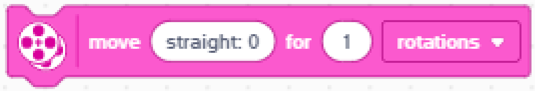
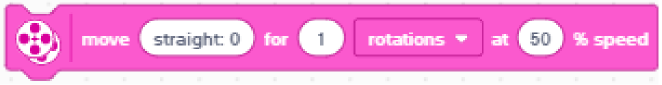
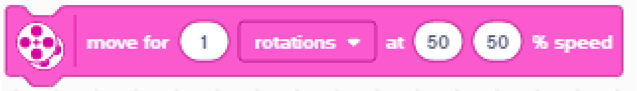
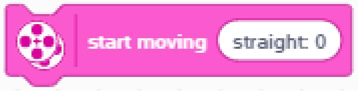
MOVEMENT BLOCKS

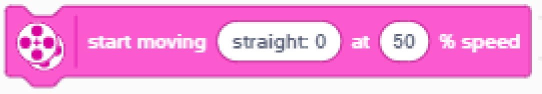
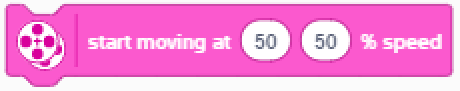
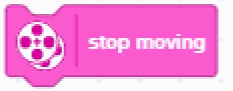
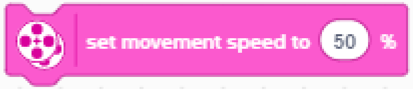
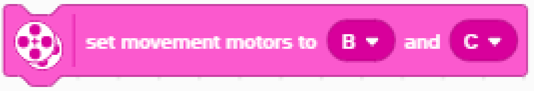
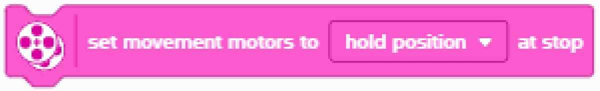
MOVEMENT

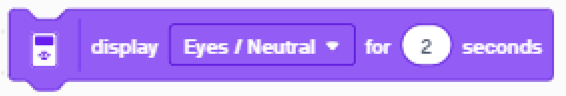
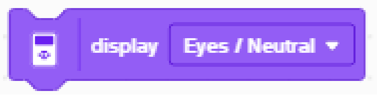
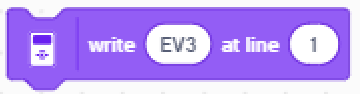
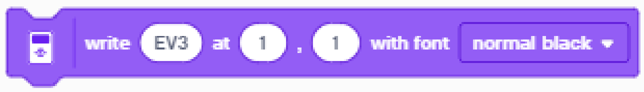
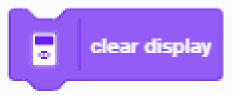
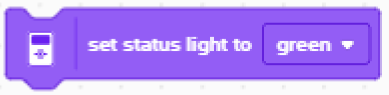
Movement blocks allow you to run two motors in a synchronized motion.

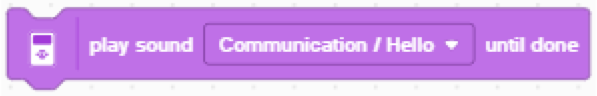
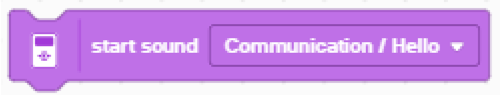
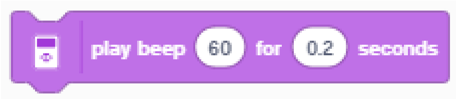
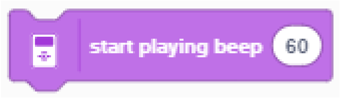
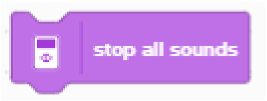
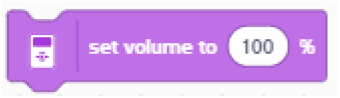
(Only motors of the same type can be synchronized.)

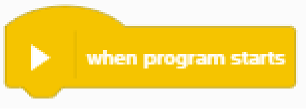
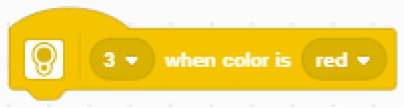
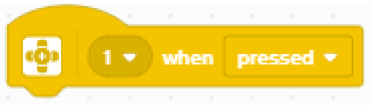
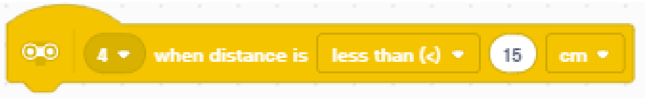
MOVE FOR DURATION  	Moves synced motors forward or backward a specified number of rotations, degrees, or seconds. Use the SET MOVEMENT MOTORS block to change which motors are controlled. The default ports are B (left) and C (right). Use the SET MOVEMENT SPEED block to change the speed of the motors. The default speed is 50%.

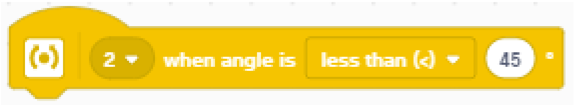
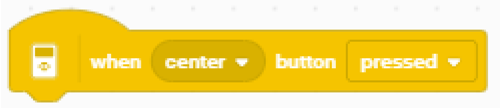
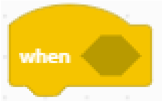
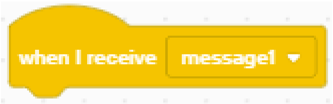
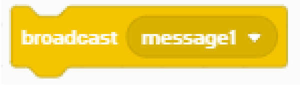
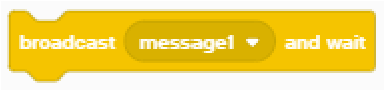
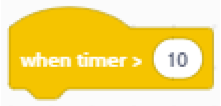
MOVE WITH STEERING FOR DURATION  move straight: 0 for 1 rotations ▾	Moves synced motors forward the specified number of rotations, degrees, or seconds with the specified steering. A value of "0" goes in a straight line. A value of "50" turns one motor off and the other one on (for a pivot turn). A value of "100" turns one motor forward and the other one backward (for a spin turn). Any other values will have the two motors spinning different speeds to drive in an arc. Use the SET MOVEMENT SPEED block to change the speed of the motors. The default speed is 50%.
MOVE WITH STEERING FOR DURATION AT SPEED  move straight: 0 for 1 rotations ▾ at 50 % speed	Moves synced motors forward the specified number of rotations, degrees, or seconds with the specified steering at the specified speed. (See steering values above.)
MOVE WITH TANK FOR DURATION AT SPEED  move for 1 rotations ▾ at 50 50 % speed	Moves synced motors forward the specified number of rotations, degrees, or seconds with the specified steering at the specified speeds. The difference between STEERING and TANK is that you can control the two motor speeds independently. The first speed value sets the speed of the left motor and the second speed value sets the speed of the right motor. Two matching values will go in a straight line. Any value paired with a "0" turns one motor off and the other one on (for a pivot turn). Any positive value paired with the same negative value turns one motor forward and the other one backward (for a spin turn). Any other values will have the two motors spinning different speeds to drive in an arc.
START MOVING WITH STEERING  start moving straight: 0	Turns the synced motors on with the specified steering until the program ends or until another block tells the motors do something else. (See steering values above.)

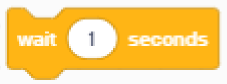
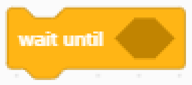
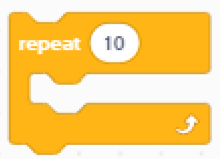
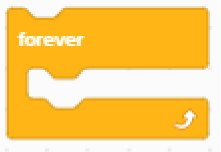
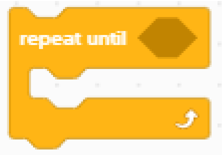
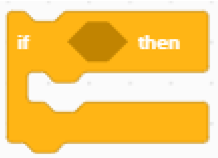
START MOVING WITH STEERING AT SPEED  start moving straight: 0 at 50 % speed	Turns the synced motors on with the specified steering at the specified speed until the program ends or until another block tells the motors do something else. (See steering values above.)
START MOVING WITH TANK AT SPEED  start moving at 50 50 % speed	Turns the synced motors on with the specified speeds until the program ends or until another block tells the motors do something else. The difference between STEERING and TANK is that you can control the two motor speeds independently. The first speed value sets the speed of the left motor and the second speed value sets the speed of the right motor. (See tank values above.)
STOP MOVING  stop moving	Stops the synced motors from running. By default, the motors will brake so that they quickly come to a complete stop. The motors will hold their position once stopped. You can change what happens when move motors stop with the SET MOVEMENT STOP ACTION block.
SET MOVEMENT SPEED  set movement speed to 50 %	Sets the speed of the synced motors. The speed range is -100 to 100. Negative values will reverse the direction of the movement. This will NOT turn on the motors. It must be used with other movement blocks. The default speed is 50%.
SET MOVEMENT MOTORS  set movement motors to B ▾ and C ▾	Sets which two motors will be synced for all movement blocks. The first port sets the left motor and the second port sets the right motor. The default ports are B (left) and C (right).
SET MOVEMENT STOP ACTION  set movement motors to hold position ▾ at stop	Sets the action that the synced motors will perform when a movement command completes or when the moving is stopped. It can be set to float (coast) or actively hold position (brake).

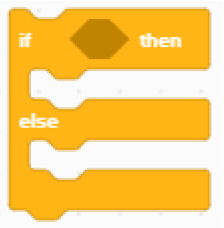
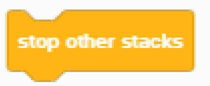
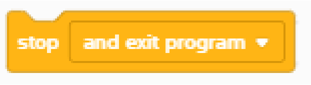
DISPLAY BLOCKS DISPLAY	Display blocks allow you to show something on the EV3 screen or change the status lights.
DISPLAY IMAGE FOR SECONDS  display Eyes / Neutral ▾ for 2 seconds	Shows the selected image on the EV3 display screen for the specified amount of time.
DISPLAY IMAGE  display Eyes / Neutral ▾	Shows the selected image on the EV3 display screen. The image stays on the screen until it is overwritten by another block, the display is cleared, or the program ends.
WRITE AT LINE  write EV3 at line 1	Shows the specified text at the specified line on the EV3 display screen. The line height is 10 pixels. The text stays on the screen until it is overwritten by another block, the display is cleared, or the program ends.
WRITE AT COORDINATES WITH FONT  write EV3 at 1 , 1 with font normal black ▾	Shows the specified text, in the specified font, at the specified XY coordinates on the EV3 display screen. The line height is 10 pixels. The text stays on the screen until it is overwritten by another block, the display is cleared, or the program ends.
CLEAR DISPLAY  clear display	Clears the EV3 display screen.
SET STATUS LIGHT  set status light to green ▾	Sets the EV3 brick status light to the specified color and pulse option. Light can be solid or pulse (flash).

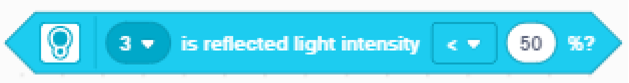
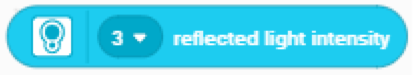
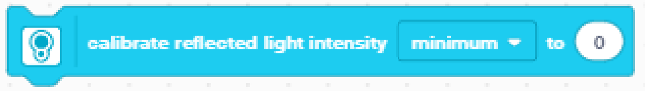
SOUND BLOCKS ● SOUND	Sound blocks play sounds through the EV3 speaker.
PLAY SOUND UNTIL DONE  play sound Communication / Hello ▾ until done	Plays the selected sound on the EV3 brick and waits until the sound is finished before the program continues.
START SOUND  start sound Communication / Hello ▾	Starts playing the selected sound on the EV3 brick and immediately continues with the next command in the program stack.
PLAY NOTE FOR SECONDS  play beep 60 for 0.2 seconds	Plays the selected musical note for the specified number of seconds before the program continues.
START PLAYING NOTE  start playing beep 60	Starts playing the selected musical note. The note keeps playing until the speaker is told to do something else or the program ends.
STOP ALL SOUNDS  stop all sounds	Stops all sounds that are currently playing on the EV3 brick.
SET VOLUME  set volume to 100 %	Sets the volume of the sound. The default volume is 100%.

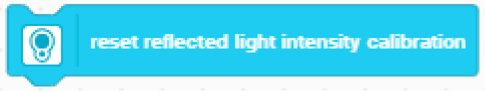
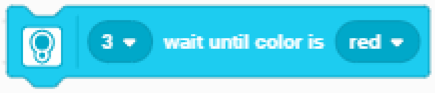
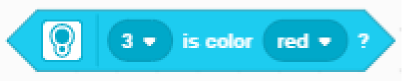
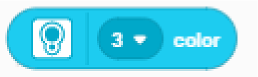
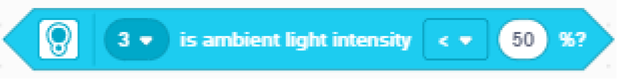
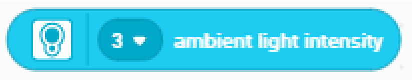
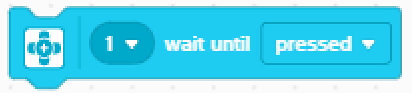
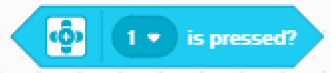
EVENT BLOCKS ● EVENTS	Event blocks are always the first block in a program stack. They allow you to run actions based on when certain events happen. (Events can be when the program starts or when a sensor or timer reaches a certain value.)
WHEN PROGRAM STARTS 	When the program starts, it runs all commands attached to it from top to bottom. The program can be started by clicking the RUN button (▶) in EV3 Classroom, which is dangerous if your robot has wheels and can roll off the table, or the DOWNLOAD button (↓) then selecting and running the program from the EV3 brick.
WHEN COLOR 	When the specified color is detected, it runs all commands attached to it from top to bottom. The sensor can detect: • no color • black • blue • green • yellow • red • white • brown • changed The color detected must change before this block is triggered again.
WHEN TOUCH 	It runs all commands attached to it from top to bottom when the touch sensor is pressed or released. The state of the touch sensor must change before this block is triggered again.
WHEN DISTANCE 	It runs all commands attached to it from top to bottom when the ultrasonic sensor's distance is less than, greater than, equal to, or changes more than the specified distance. The distance must change before this block is triggered again.

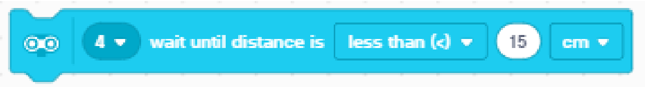
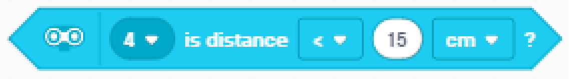
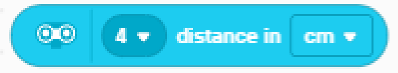
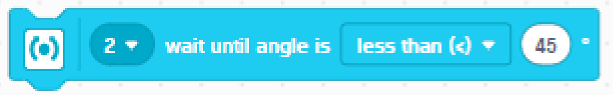
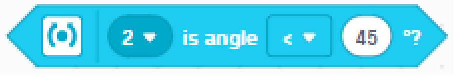
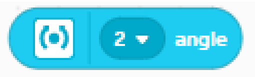
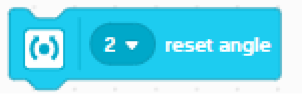
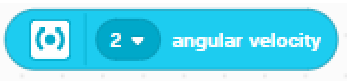
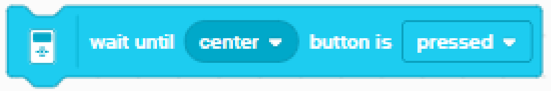
WHEN ANGLE  	It runs all commands attached to it from top to bottom when the gyro sensor's angle is less than, greater than, equal to, or changed more than the specified angle. The angle must change before this block is triggered again.
WHEN BRICK BUTTON  	It runs all commands attached to it from top to bottom when the specified brick button is pressed or released. The state of the button must change before this block is triggered again.
WHEN  	It runs all commands attached to it from top to bottom when the specified Boolean condition placed inside it becomes TRUE. The condition must change to FALSE and back to TRUE before this block is triggered again.
WHEN MESSAGE RECEIVED  	It runs all commands attached to it from top to bottom when the specified message is broadcast from another part of the program (using the BROADCAST MESSAGE block or the BROADCAST MESSAGE AND WAIT block).
BROADCAST MESSAGE  	Broadcasts the specified message to any WHEN MESSAGE RECEIVED blocks in the program. This block sends the message and immediately proceeds to the next command in the program without waiting.
BROADCAST MESSAGE AND WAIT  	Broadcasts the specified message to any WHEN MESSAGE RECEIVED stacks in the program. This block sends the message and waits until all commands attached to the WHEN MESSAGE RECEIVED stack have finished before proceeding to the next command in the program.
WHEN TIMER  	It runs all commands attached to it from top to bottom when the timer exceeds the specified value. The timer would have to be reset and exceed the value again before this block is triggered again.

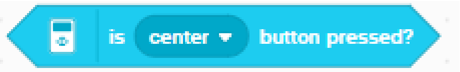
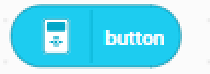
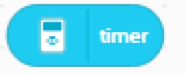
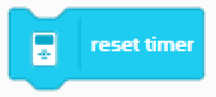
CONTROL BLOCKS ● CONTROL	Control blocks change the flow of the program allow you to play sounds through the robot.
WAIT FOR SECONDS 	This block pauses the program stack for a specified number of seconds. You can use whole numbers and decimals. WAIT does not mean do nothing. For example, motors that have been started before this command will continue to run during the wait.
WAIT UNTIL 	This block pauses the program stack until the specified Boolean condition becomes TRUE. WAIT does not mean do nothing. For example, motors that have been started before this command will continue to run during the wait.
REPEAT LOOP 	All the blocks held inside this block will be repeated the specified number of times before the stack continues running any blocks below it.
FOREVER LOOP 	All the blocks held inside this block will be repeated forever. The only way to stop this loop is to end the program with the buttons on the EV3 brick or having another program stack call the STOP OTHER STACKS block or the STOP block.
REPEAT UNTIL LOOP 	All the blocks held inside this block will be repeated until the specified Boolean condition becomes TRUE. Then any blocks below it will play.
IF THEN 	This block will check if the specified Boolean condition is TRUE. If it is true, any blocks inside will be played. If it is FALSE, all blocks inside will be ignored.

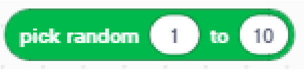
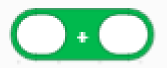
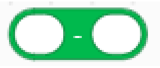
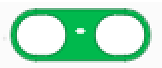
IF THEN ELSE  	This block will check if the specified Boolean condition is TRUE. If it is true, any blocks inside the first space will be played and any blocks inside the second space will be ignored. If it is FALSE, any blocks inside the first space will be ignored and all blocks inside the second space will be played.
STOP OTHER STACKS  	This block stops all program stacks except its own.
STOP  	This block can stop all running program stacks, or its own program stack, or end the program.

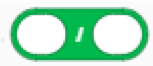
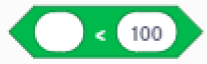
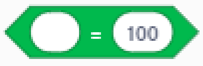
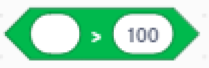
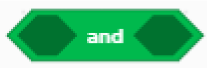
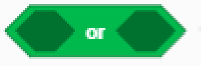
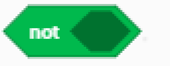
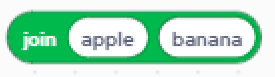
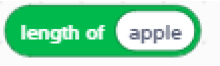
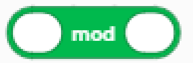
SENSOR BLOCKS SENSORS	Sensor blocks read or compare values from the different sensors. (color/light, touch, distance, gyro, brick buttons)
IS REFLECTED LIGHT  	Returns TRUE if the color sensor's reflected light intensity is greater than, equal to, or less than the specified percentage. Use PORT VIEW on the EV3 brick to find reflected light values. CALIBRATING your sensor will change the values returned.
READ REFLECTED LIGHT  	Returns the current value of the color sensor's reflected light intensity as a percentage. CALIBRATING your sensor will change the values returned.
CALIBRATE REFLECTED LIGHT  	This block adjusts the sensitivity of the color sensors. When this block is run, you can set the maximum and minimum to any value you choose. The most practical application is to use a READ REFLECTED LIGHT block in the value window. Use maximum when the sensor is over the brightest color you will see. Use minimum when the sensor is over the darkest color you will see.

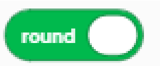
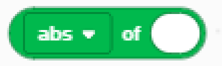
RESET REFLECTED LIGHT 	This block resets the reflected light sensor readings back to their default values. Use this if you no longer want to use the values set with the CALIBRATE REFLECTED LIGHT block.
WAIT UNTIL COLOR IS 	Pauses the program stack until the color sensor detects the specified color. The sensor can detect black, white, blue, brown, green, yellow, red, and no color, and changed color. These are calibrated to the colors of LEGO bricks.
IS COLOR 	Returns TRUE if the color sensor detects the specified color. Use PORT VIEW on the EV3 brick to find color values.
READ COLOR 	Returns the current color detected by the color sensor expressed as a number. • 0 = no color • 1 = black • 2 = blue • 3 = green • 4 = yellow • 5 = red • 6 = white • 7 = brown
IS AMBIENT LIGHT 	Returns TRUE if the color sensor's ambient light intensity is greater than, equal to, or less than the specified percentage. Ambient light reads light from the room, not the light generated by the sensor itself. Use PORT VIEW on the EV3 brick to find ambient light values.
READ AMBIENT LIGHT 	Returns the current value of the color sensor's ambient light intensity as a percentage.
WAIT UNTIL TOUCH PRESSED 	Pauses the program stack until the touch sensor is pressed or released.
IS TOUCH PRESSED 	Returns TRUE if the touch sensor is currently pressed.

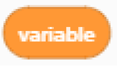
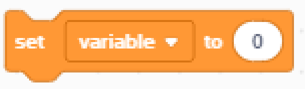
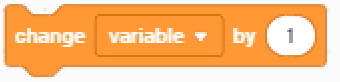
WAIT UNTIL DISTANCE IS  4 ▾ wait until distance is less than (<) ▾ 15 cm ▾	Pauses the program stack until the ultrasonic sensor's distance to an object is less than, greater than, equal to, or changed more than the specified distance in centimeters or inches.
IS DISTANCE  4 ▾ is distance < ▾ 15 cm ▾ ?	Returns TRUE if the ultrasonic sensor's distance to an object is less than, greater than, or equal to the specified distance in centimeters or inches.
READ DISTANCE  4 ▾ distance in cm ▾	Returns the ultrasonic sensor's current distance from an object in centimeters or inches. The sensor's range is 0-255 centimeters or 0-100 inches.
WAIT UNTIL ANGLE IS  2 ▾ wait until angle is less than (<) ▾ 45 °	Pauses the program stack until the gyro sensor's angle is less than, greater than, equal to, or changed more than the specified angle.
IS ANGLE  2 ▾ is angle < ▾ 45 °?	Returns TRUE if the gyro sensor's angle is greater than, less than, or equal to the specified angle.
READ ANGLE  2 ▾ angle	Returns the current angle read by the gyro sensor in degrees.
RESET ANGLE  2 ▾ reset angle	Resets the gyro sensor's angle to zero.
READ TURNING VELOCITY  2 ▾ angular velocity	Returns the gyro sensor's current angular velocity (speed of turning) in degrees per second.
WAIT UNTIL BRICK BUTTON IS PRESSED  wait until center ▾ button is pressed ▾	Pauses the program stack until the specified brick button is pressed or released.

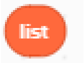
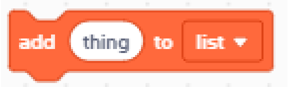
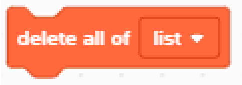
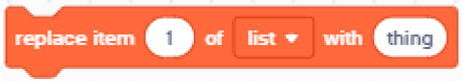
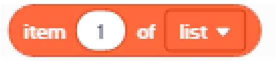
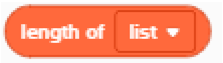
IS BRICK BUTTON PRESSED  	Returns TRUE if the specified brick button is pressed.
READ BRICK BUTTON  	Returns the current brick button being pressed expressed as a number. - 0 = no button - 1 = left - 2 = center - 3 = right - 4 = up - 5 = down
READ TIMER  	Returns the time, in seconds, since the program started or since the RESET TIMER block was last run.
RESET TIMER  	Resets the timer to zero.

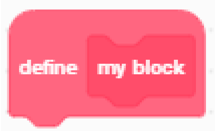
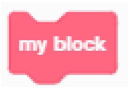
OPERATOR BLOCKS ● OPERATORS	Operator blocks allow you to complete math and logic operations.
PICK RANDOM NUMBER  	Generates a random number in the specified range (including both endpoints). For example, entering 2 to 5 could return the values of 2, 3, 4, or 5.
ADD  	Adds the two specified values and returns the result.
SUBTRACT  	Subtracts the second value from the first value and returns the result.
MULTIPLY  	Multiplies the two specified values and returns the result.

DIVIDE 	Divides the first value by the second value and returns the result
IS LESS THAN 	Returns TRUE if the first value is less than the second value.
IS EQUAL TO 	Returns TRUE if the first value is equal to the second value.
IS GREATER THAN 	Returns TRUE if the first value is greater than the second value.
AND 	Returns TRUE if both specified conditions are TRUE.
OR 	Returns TRUE if at least one of the specified conditions are TRUE.
NOT 	Returns TRUE if the specified condition is FALSE.
JOIN 	Joins (concatenates) two values and returns the combined result. For example, entering “apple” and “banana” would return “applebanana”. It is helpful to include a space after “apple” or before “banana”.
LENGTH 	Returns the number of characters (including spaces) in a specified string. For example, entering “apple” would return a 5.
MOD 	Returns the remainder resulting from dividing the first value by the second value. For example, entering 10 and 3 would return a 1 because 10 divided by 3 equals 3 with a remainder of 1.

ROUND 	Rounds the specified number to the nearest integer (positive or negative whole number). This block follows the standard rule of .5 or higher being rounded up, and less than .5 rounded down.
MATH FUNCTIONS 	Performs a math function on the specified value and returns the result. • abs = absolute value • floor = rounds down to next lower integer • ceiling = rounds up to next higher integer • sqrt = square root • sin = sine • cos = cosine • tan = tangent • asin = inverse sine • acos = inverse cosine • atan = inverse tangent • ln = natural logarithm • log = logarithm • e^ = exponential (base e) • 10^ = exponential (base 10)

VARIABLE BLOCKS ● VARIABLES	Variable blocks allow you to create your own variables and lists to store data.
VARIABLE 	Returns the value stored in the variable. Whenever a variable is created, a version of this block appears with the specified name on it.
SET VARIABLE TO 	Stores the specified value in the variable. Variables can be a string (characters) or numbers.
CHANGE VARIABLE BY 	Changes the value stored in the variable by the specified value. For example, if “variable” currently had a value of 5, entering a 3 in this block would increase “variable” to a value of 8. (Negative numbers will decrease the variable’s value.) If the variable is a string, this block will change to the specified number.

LIST 	Returns all the values stored in the list. Whenever a list is created, a version of this block appears with the specified name on it.
ADD ITEM TO LIST 	Adds an item containing the specified text or number to the list.
DELETE ITEMS IN LIST 	Deletes all items stored in the specified list.
REPLACE ITEM IN LIST 	Replaces the specified item’s content with the specified value.
READ ITEM IN LIST 	Returns the value of the specified item in the specified list.
LENGTH OF LIST 	Returns the number of items contained in the specified list.

MY BLOCKS ● MY BLOCKS	My blocks allow you to define your own repeatable functions (sets of instructions). (These can be called at any time from anywhere in your program.)
DEFINE BLOCK 	Allows you to create your own function (set of instructions). All blocks attached to this program stack will run whenever the RUN BLOCK is triggered.
RUN BLOCK 	Runs a user-defined function (set of instructions). This can be called at any time from anywhere in your program. The blocks run are the ones attached to the specified DEFINE BLOCK.