A COMPLETE PREMIER RESCUE PROGRAM

A COMPLETE PREMIER RESCUE PROGRAM

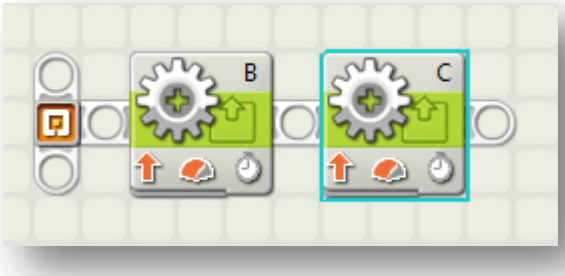
To successfully complete the Premier Rescue course a robot needs to be programmed in a manner that will allow it to sense the various challenges as they appear and react to them. In order to do this the program needs to be written so that it is waiting for a number of different sensor inputs to take place before reacting in the appropriate manner. For example, While the robot is navigating its way along the line it needs to be waiting for the possibility of the Water Bottle Obstacle being in the robots way. By whatever method the robot sensors/detects the Water Bottle, it then needs to stop following the line and move around the Obstacle.

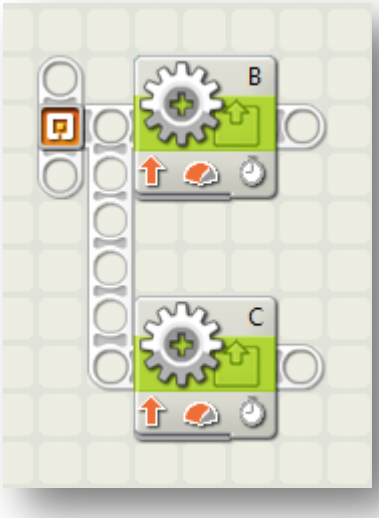
What we need is a way to be constantly checking the sensor assigned to detecting the Water Bottle Obstacle while following the line (as well as all the other things we need to be watching out for including Gridlock and the Chemical Spill). We need a way of multi-tasking our program so that it can watch for particular events, Water Bottle, Gridlock, Chemical Spill etc, while doing something else like following the line. Unlike RoboLab, Mindstorms NXT-G does not have an obvious Event Block or Task Block.

> Multitasking in Mindstorms NXT-G requires the use of the programming Blocks supplied, in a logical manner. NXT-G is much more like a modern structured programming language than Robolab, so you need to stop thinking GOTO (Jumps & Lands) and start thinking Structured Programming.

Mindstorms NXT-G is capable of simultaneously running parallel sequence bars so that, for example, two Motor Blocks controlled by time will operate at the same time.

Below is an example of what appears to be the same program.
The sequence on the left has the same two motors with the exact same properties as the one on the right.
However, if you run the left hand program, Motor B will rotate for 1 second and stop then Motor C will rotate for 1 second and stop because the Wait For Completion option is fixed when motors are in timer mode.
The program on the right will have both motors run simultaneously for 1 second as they are positioned in parallel on the sequence bars.





EVENTS, MULTI-TASKING & TASK SWAPPING

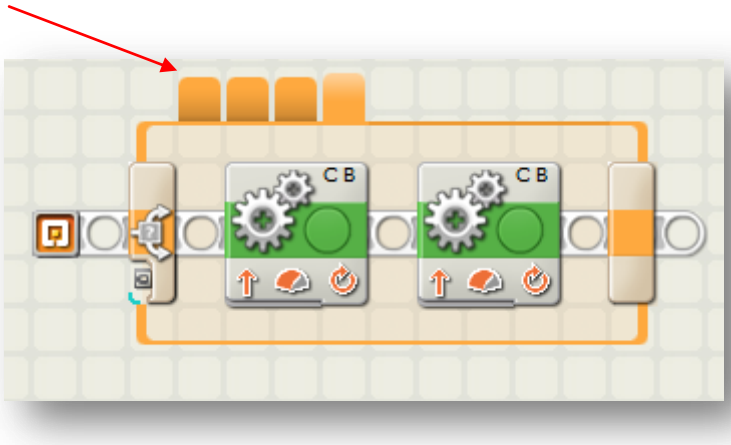
In order for our Premier Rescue robot to successfully navigate a complete course using the NXT-G programming language, it needs to be able to:

1. Wait for Events to be triggered, eg Touch Sensor detects the Water Bottle Obstacle.
2. Multi-task – Simultaneously watch for Events that will require the robot to change tasks, eg watch for the Water Bottle and the Gridlock and the Chemical Spill at the same time.
3. Task Swap from one task/behaviour to another as the course requires, eg change from line flowing to navigating around the Water Bottle Obstacle.

In order to do this we need a way for the program to know what mode it is in. If each mode is assigned a number we can store this value in a Variable that can be changed as Events are detected by the robots sensors.

Let's give a default value of;
0 (zero) to Line Following,
1 to the Water Bottle,
2 to the Gridlock,
3 to the Find the Chemical Spill and then the Victim,
4 to the Find the Platform.

We now need a control structure that can handle multiple events. The one that comes to mind is a CaseWhere Multi-way Selection control structure. Luckily NXT-G has a Multi-way Selection programming Block available. The NXT-G Switch Block can be configured to handle more than two selection events therefore turning it into a CaseWhere Block.
The program below has a Switch Block that has been placed into Flat View with the Control mode set to Value. It has four option tabs allowing the Block to perform four different actions.



You need to hover the mouse over each tab to see the value assigned to it, and by selecting each tab you can see the programming blocks that reside on each sequence bar of the Switch Block.

EVENTS, MULTI-TASKING & TASK SWAPPING (CONT..)

MULTI-TASKING ALGORTIHM

Lets look at the logic behind what we are trying to do.

BEGIN PremierRescue
Task0 (LineFollower)
Program to follow the line and naviage the intersections)
EndTask0

Task1 (waterBottleObstacle)
Program to navigate around the Obstacle
EndTask1

Task2 (gridLock)
Program to navigate the Gridlock)
EndTask2

Task3 (chemicalSpillDetect)
Program to detect the Chemical Spill
EndTask3

Task4 (findVictim)
Program to find the Victim
EndTask4

Task5 (placeVictimOnPlatform)
Program to place the victim on the platform)
EndTask5

TaskMain
CaseWhere TaskNo
0 : Task0 *'default task'*
1 : Task1
2 : Task2
3 : Task3
4 : Task4
5 : Task5
EndCaseWhere
ENDTaskMain
END

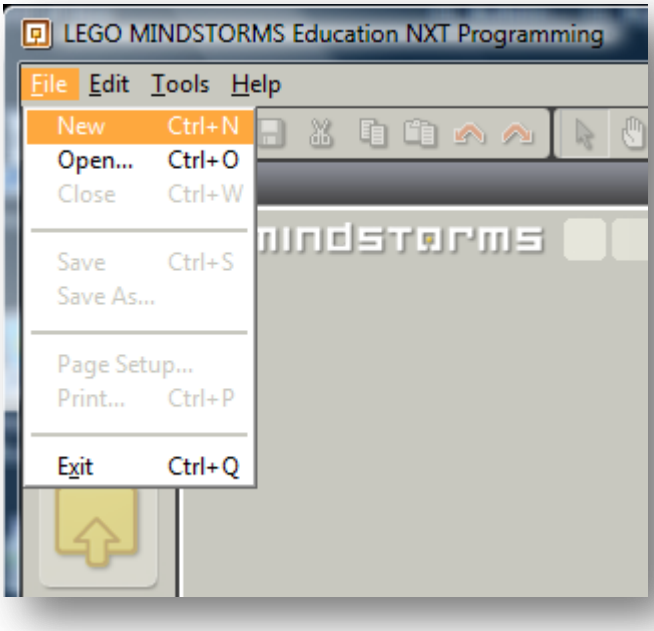
One advantage of this method is that a Premier Rescue team can have different members solving the various problems seperately. When a task is working it simply needs to be included into the Premier Rescue program.

The following section shows you how to do Mult-Task Swapping in the NXT-G Programming Language.

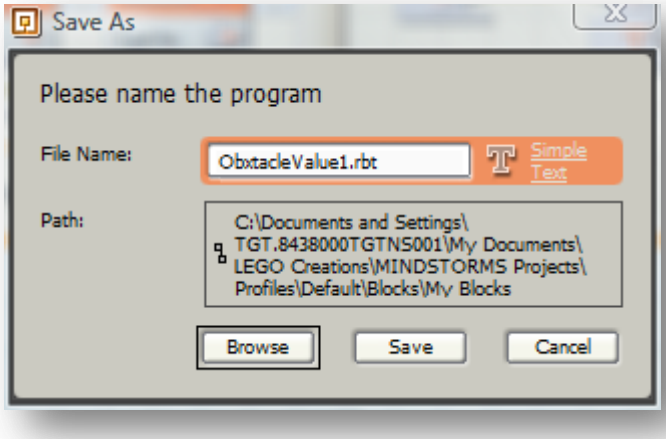
EVENTS, MULTI-TASKING & TASK SWAPPING (CONT..)

We need a way of generating and storing a value when a particular event occurs.
To store a value we can use a Variable.
To generate the value, we need a sequence of blocks that will constantly check for a event to take place and when it does generate the appropriate value and store it into the variable.
Exercise: We will create a program that will detect the Obstacle and set the TaskNo variable to a value of 1.

Firstly we need to declare (create) a variable that will allow us to store values.
Create a new program and call it ObstacleValue1.rbt

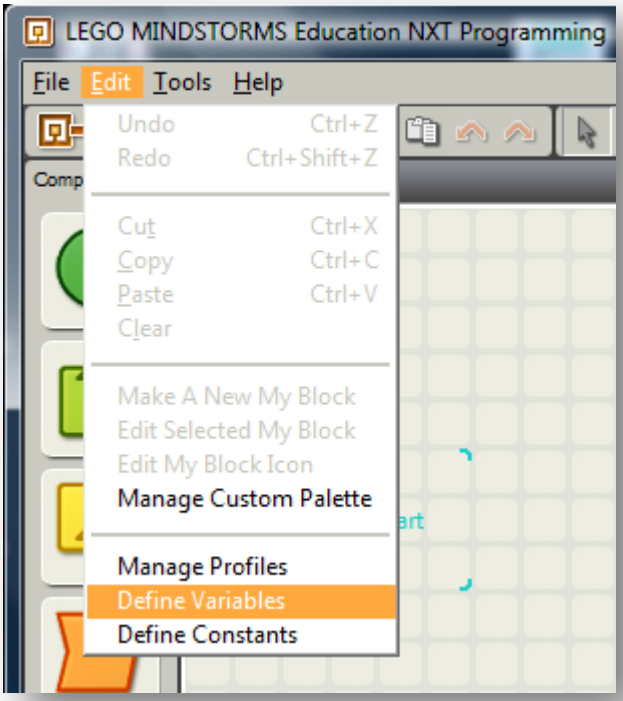


Select File and Save.

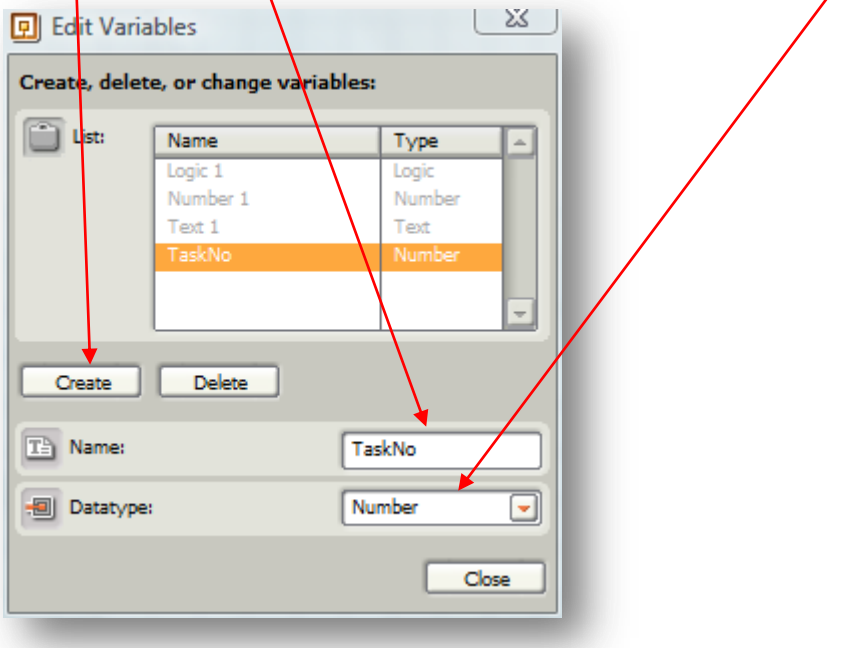


EVENTS, MULTI-TASKING & TASK SWAPPING (CONT..)

Now select Edit and then Define Variables.

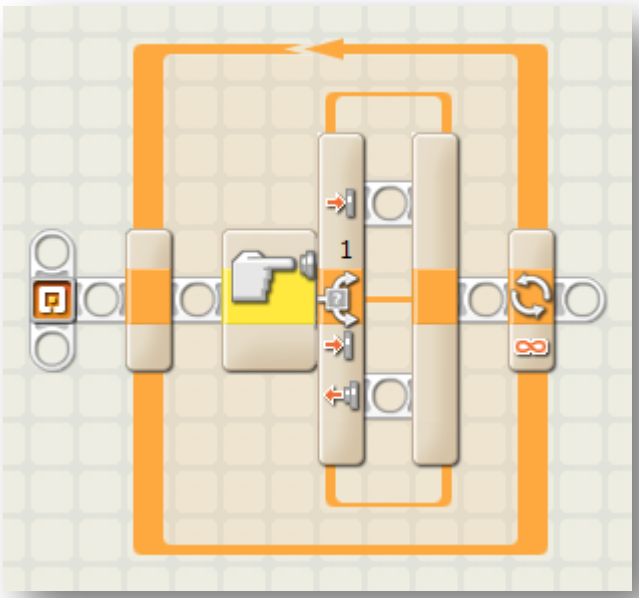


We will now create a variable that we can be used to store our task number.
Select Create. Type in TaskNo into the Name field and change the Data type to Number. Then select Close.



EVENTS, MULTI-TASKING & TASK SWAPPING (CONT..)

Firstly we need to place a Loop Block onto the Sequence Bar. Inside it, place a Switch Block.

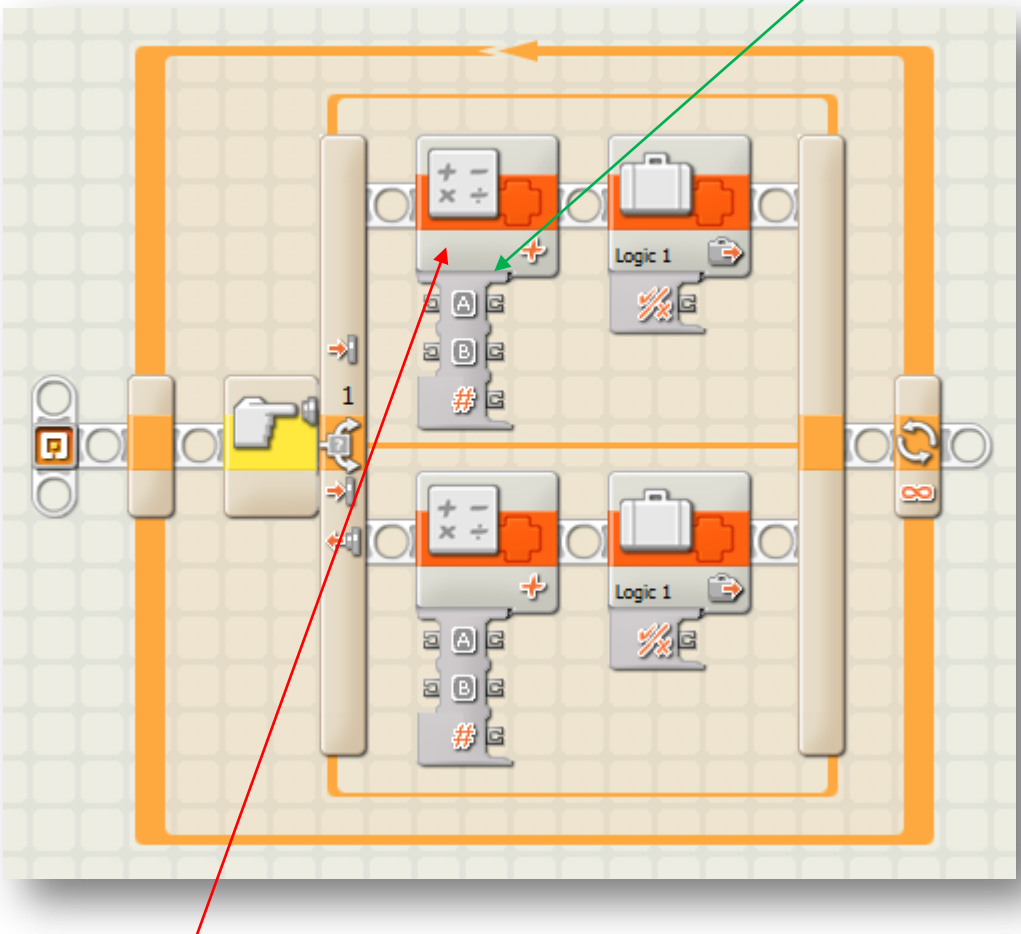


The Loop Block is left on Forever as we want this task to be constantly looking for the Obstacle. Remember that it is possible that more than one Obstacle maybe placed on a Rescue Course.

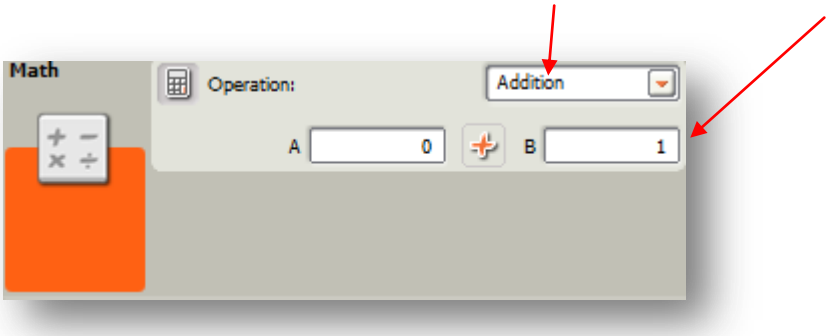
Set the Port Number of the Switch to the port that has a bump sensor set up to detect the presence of the Obstacle (Water Bottle).

EVENTS, MULTI-TASKING & TASK SWAPPING (CONT..)

Place a Maths Block and a Variable Block on both branches of the Switch Block. Get these blocks from the Data Pallet and then expand the Data Hubs of each of the blocks by hovering over the join/groove at the bottom of each block to allow for attaching the data wires between the blocks.



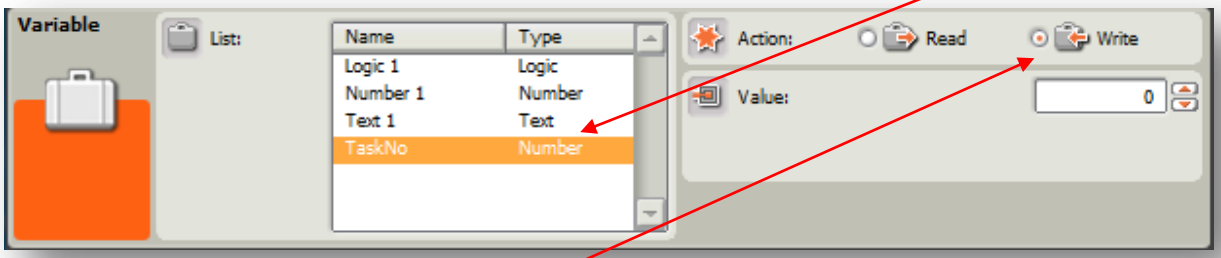
Select the top Maths Block and set the Operation to Addition and the B property to **1**.



Select the bottom Maths Block and set the Operation to Addition and the B property to **0**.

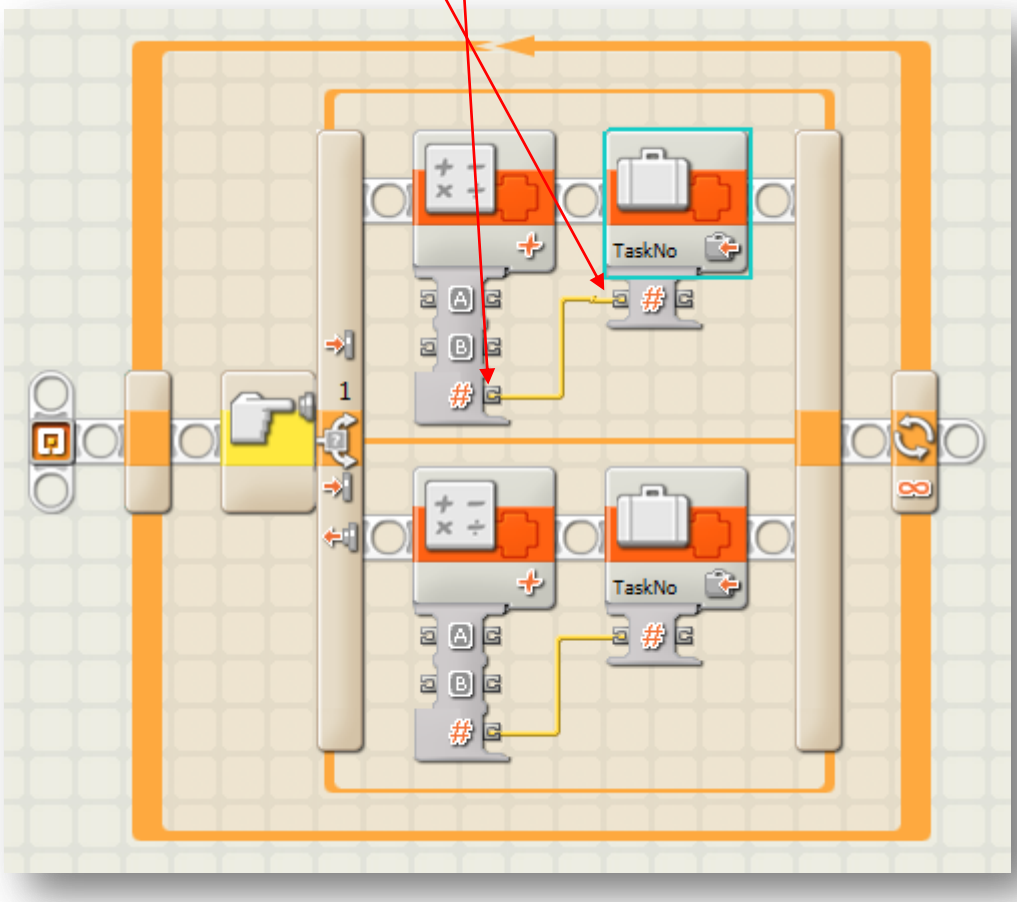
EVENTS, MULTI-TASKING & TASK SWAPPING (CONT..)

Now select one of the Variable Blocks and select from the List, the variable you created, **TaskNo**.



Also change the Action property to Write so that the Maths Block can change the value of the Variable.

Now hover the cursor over the Maths Block # datawire connector. It will turn into a wiring cursor.
Click and drag to the Variable Block # datawire connector and let go. A yellow datawire should link the two.
Do this for both sides of the Switch Block

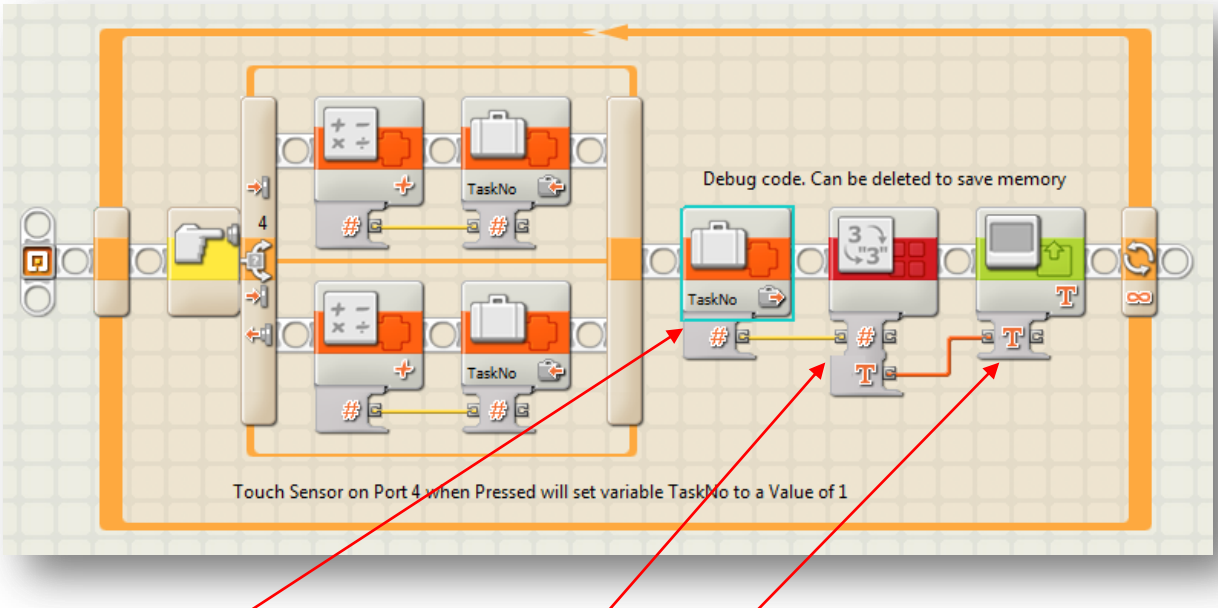


EVENTS, MULTI-TASKING & TASK SWAPPING (CONT..)

When programming, it is a good idea to be able to see what the program is doing. With our TaskNo program we want the variable to change to 1 if the Touch Sensor is pressed. However, how do we know if it's working?

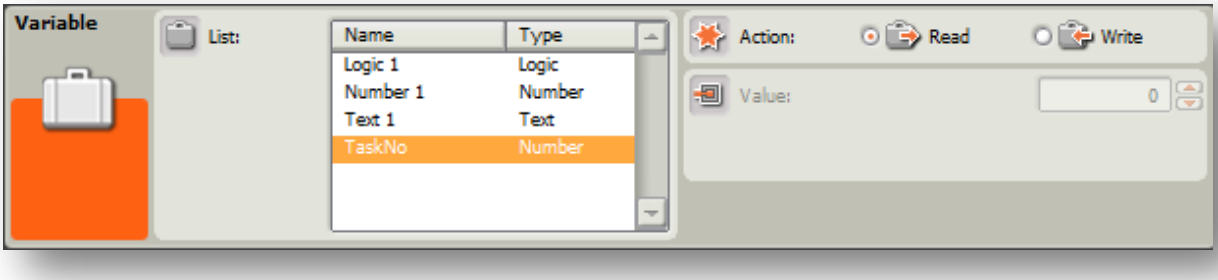
Lets put some debug blocks into the program so that we can see that the value of the variable changes when we press the Touch Sensor.

The NXT has the ability to display text and graphics on its screen. Therefore we will use this to check if the program is writing the value one (1) to the variable TaskNo.



Place a Variable Block, a NumberToText Block and a Display Block onto the Sequence Bar after the Switch Block as shown above.

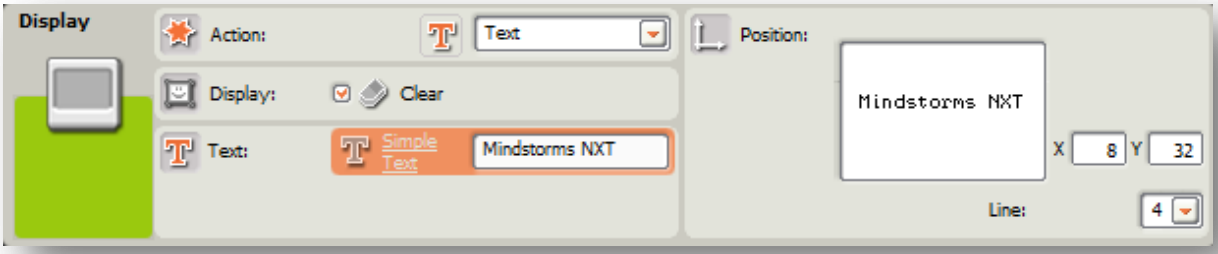
Set the Variable Block properties to; List = TaskNo, Action to Read. This Variable Block will now read the value that has been set inside the Switch Block. If the Touch Sensor has been pressed then the value should be 1. If not, then the value should be 0.



EVENTS, MULTI-TASKING & TASK SWAPPING (CONT..)

Leave the property setting of the Number to Text Block as default.

Change the Display Block property, Action = Text, leave everything else.



Wire the blocks up as shown in the main program screen and then download and run the program on your robot. Make sure you have a Touch Sensor attached to your robot on the port you have set and when the program is running, depress the switch. If everything is working the screen should display a 1 on the screen. When you let go of the switch, the screen should display a zero.

NOTE

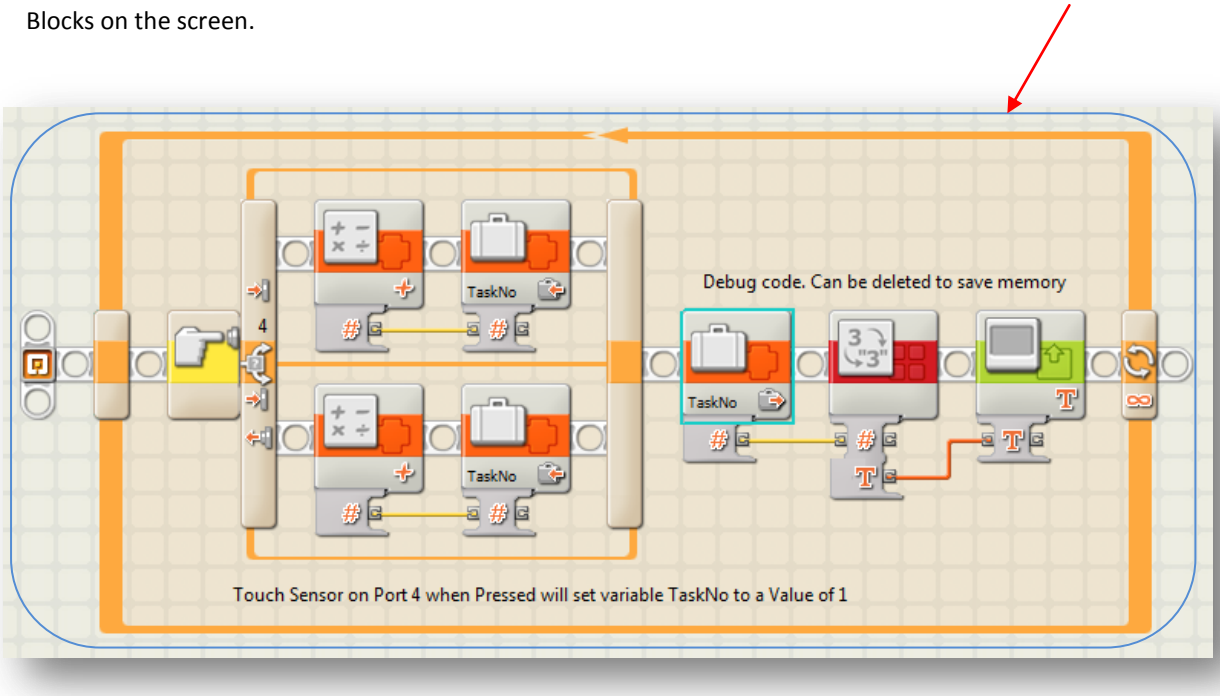
When you have completed all your programming you can remove the debug code from each of the MyBlocks. The less a program needs to read as it is executing/running the more efficient it will be and the less memory the program will need on your NXT.

EVENTS, MULTI-TASKING & TASK SWAPPING (CONT..)

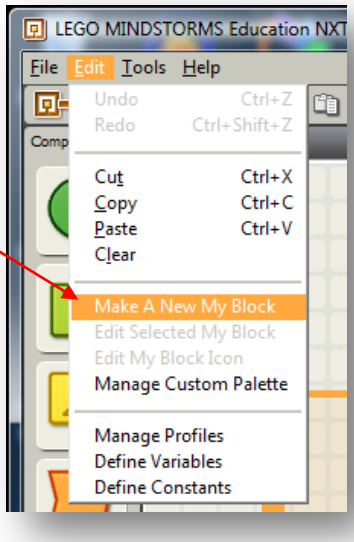
MY BLOCK CREATION

In order to make our overall program compact on the main PremierRescue program screen, one very good tool available in the NXT-G programming environment is the ability to create My Blocks. These are packaged programs that you can open and modify if needed. One other advantage of My Blocks is that they execute their code more efficiently than the standard program, and any extra speed we can get when programming our robot will be warmly greeted.

To create the My Block we need to select our completed Obstacle Detection program by lassoing all of the Blocks on the screen.

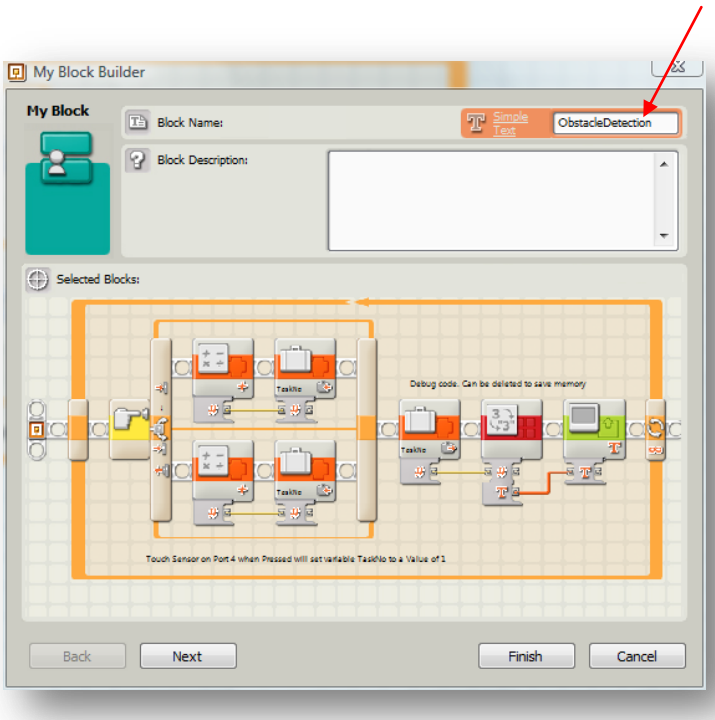


From the Edit menu, select *Make a new My Block*.

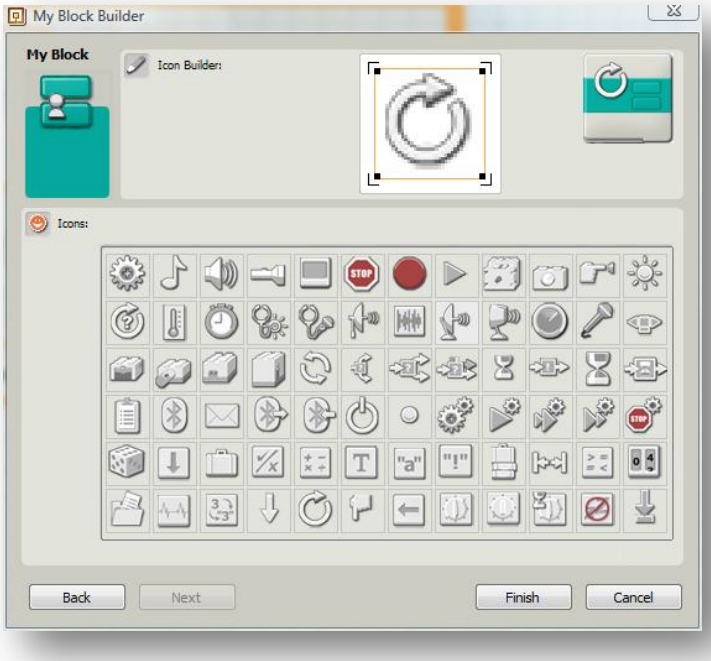


EVENTS, MULTI-TASKING & TASK SWAPPING (CONT..)

The following window will appear. Name the block something meaningful like ObstacleDetection.



Type in a description if you want to and then select Next.
Select a meaningful icon to place on your MyBlock icon and then select Finish



EVENTS, MULTI-TASKING & TASK SWAPPING (CONT..)

We now need to use the Obstacle Detection Block in our main Premier Rescue program

Here we have a Premier Rescue program in development.

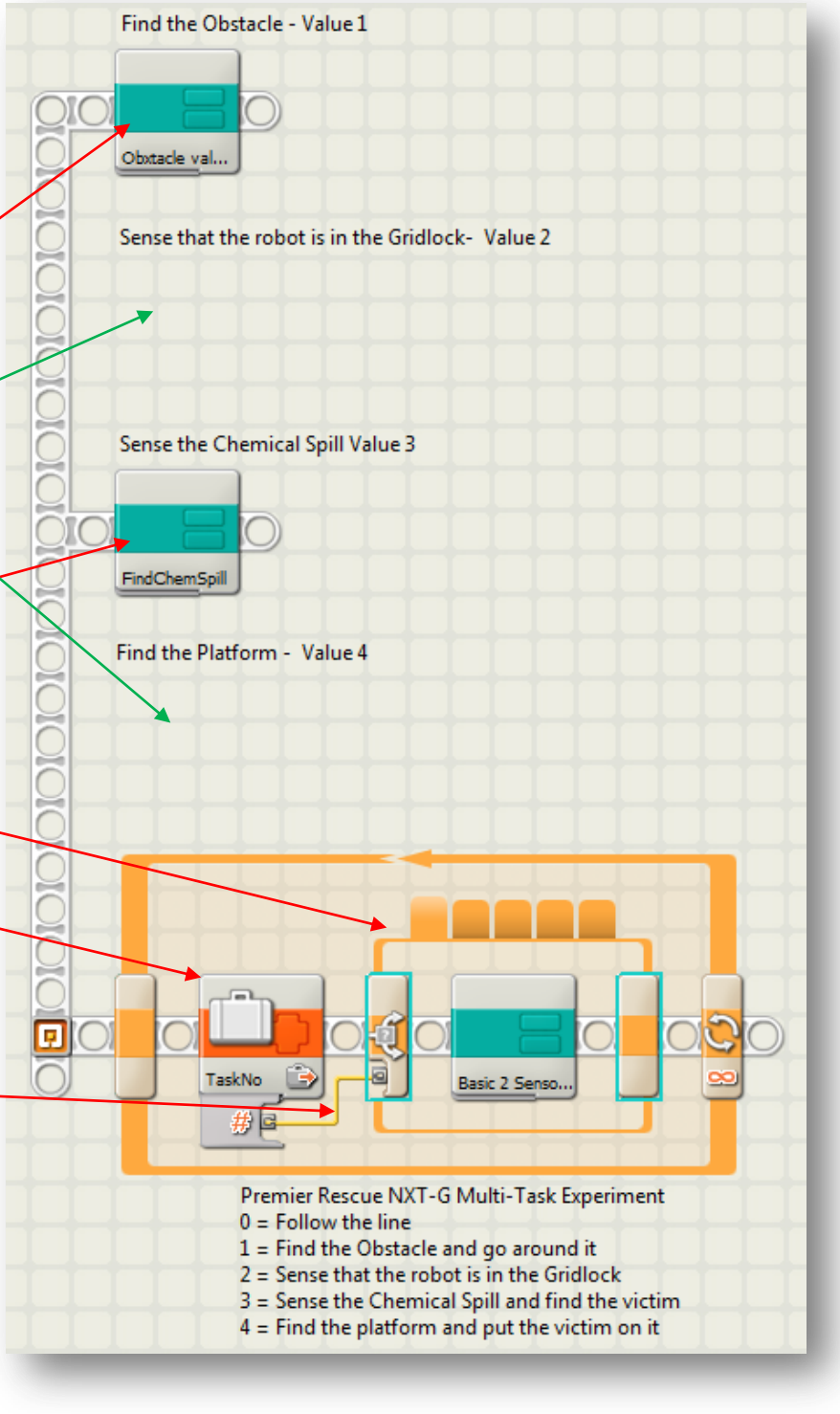
The Obstacle Detection My Block has been placed on a Parallel Sequence Bar so that the program is multi-tasking.

There are place holders for Gridlock and Find the Platform that have not been developed in this example.

The Chemical Spill My Block has been developed and placed in position so that it can also be detected when reached.

The Casewhere Main Program is setup to constantly check for the variable TaskNo value as the first Block in the main Forever Loop Block. The variable Block is set to TaskNo and Read. The value is then passed on via the wire to the CaseWhere Switch Block and depending on what the value is, executes the appropriate program.

Case 0 = Line Following
Case 2 = Navigate around the Obstacle. Etc.



EVENTS, MULTI-TASKING & TASK SWAPPING (CONT..)

END OF SECTION

So we now have a technique that allows us to change a variable to whatever number we want. The above example sets the variable TaskNo to the value of 1. In order to have it generate a different number simply change the value in the Maths Block to the required number eg 2.

You can now insert the My Blocks that solve each of the challengers into its appropriate section of the CaseWhere Switch Block.

Modify your Task My Blocks so that they set a number value appropriate for the problem to be solved, eg. Sense when the robot enters the Gridlock, or detects the Victim. Place them in a position parallel to the main program like the FindTheObstacle Block to solve the Premier Rescue Challenge.