Programming and Variables

Without some way to store values, programming a computer would be very difficult. Holding onto input (like sensor values) for future reference, comparison, or manipulation is necessary to make a computer do what you want.

Variables are the computer's way of storing values for later use. They function as temporary **containers or storage for values.**

Values such as the robot's sensor reading can be placed in a variable and retrieved at a later time.

A variable is simply a place to store a value. There are, however**, different types of values.** For instance, there are different types of numbers (*integers* versus *decimals*, to name just two), and there are values that aren't even numbers, like *words*.

Since there are different types of values we can store, we must "**declare" the type** of variable we want when we create it.

Automatic Threshold Values and Variables (cont.)

To declare a variable, simply call out its type, then its name, then end with a semicolon.

`int lightValue;` will create a new integer-type variable named lightValue.

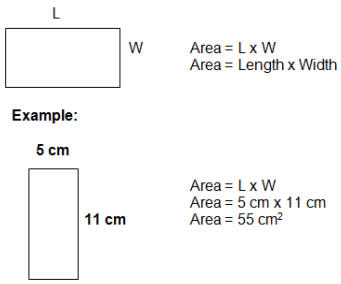
`bool isAwake;` will create a new true-or-false (Boolean) variable named isAwake.

Optionally, you can also **assign a value** to the variable at this point, but it is not necessary.

`int lightValue = 0;` will create a new integer-type variable named lightValue, with a starting value of 0.

`bool isAwake = true;` will create a new true-or-false (Boolean) variable named isAwake, with a starting value of true.

Data Type	Description	Example	Code
Integer	Positive and negative whole numbers, as well as zero.	-35, -1, 0, 33, 100, 345	`int`
Floating Point Decimal	Numeric values with decimal points.	-.123, 0.56, 3.0, 1000.07	`float`
String	A string of characters that can include numbers, letters, or typed symbols.	"Counter reached 4", "STOP", "time to eat!"	`string`
Boolean	True or False. Useful for expressing the outcomes of comparisons.	true, false	`bool`



Exercise sheet

Practice with Variables:

AREA FINDER

Create a program that collects two ***separate*** distances from the sonar sensor Multiplies them together, to get an AREA then displays the result on the NXT screen.

Essentially your are creating a scanner gun that can find the area of any rectangular box

Use the touch sensor as a button to initiate data collection

Here is simple **algorithm** that might help:

Create 3 variables:

dist1
dist2
Total

While touch sensor is <u>not</u> pressed Display "Waiting" on the screen

While touch senor is pressed – **store** Sonar sensor value **in *dist1***

While touch sensor is <u>not</u> pressed Display "Waiting"

While touch sensor is pressed – **store** Sonar Sensor value in ***dist2***
*Total = dist1 * dist2*
Display total on screen

The **while loops** here act to "hold" the program in one spot until the button is pressed each time by the user.

Helpful Variable syntax:

A) Declare variables ***above*** taskmain( )

```
int dist1;    (creating a integer variable called "dist1")
int dist2;
```

```
task main()
{

}
```

B) Storing Data in a Variable

```
dist1 = SensorValue(S1);
```
(stores the current sensor value of S1 in dist1)