

Robot C: Starting Guide

16-311 Intro to Robotics
Spring 2010

Content

- Hello World
- Similarities & Differences to C
- Motors & Encoders
- Sensors
- Printing to the Screen
- Downloading Code
- Debugging
- Advanced Topics
- Tips

Hello World

- For main use “task”

```
task main()  
{  
    nxtDisplayTextLine(0, "Hello World!");  
}
```

Similarities & Differences to C

Similarities

- Loops and conditionals work the same.
- Same primitives with the addition of bool
- Functions work the same
- Preprocessor works the same

Differences

- Missing of the standard libraries (stdio, stdlib)
 - No memory allocation
- No pointers
- Typically (for us) the whole program is in one file

Motors & Encoders

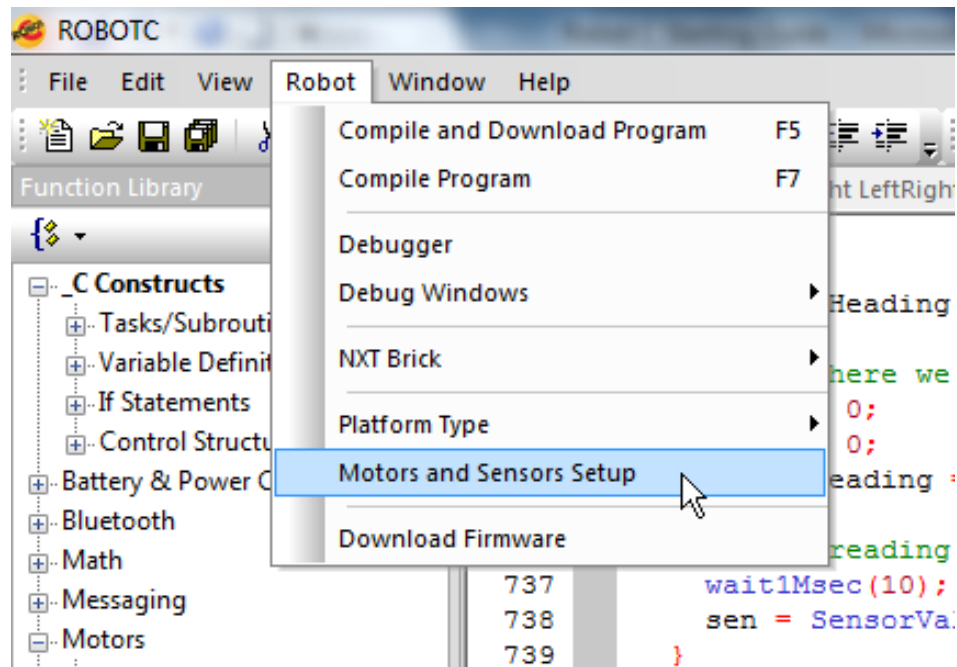
- At the heart of your programming will be driving the motors and getting encoder information from them.
- Motors
 - -100 = full reverse
 - 0 = stop
 - 100 = full forward
 - 3 motor outputs (A, B, and C)
- Encoders
 - Measures number of turns
 - One per motor
 - Can only be set to 0
 - 32bit integer

```
motor[motorA] = -100;  
motor[motorB] = 0;  
motor[motorC] = 100;
```

```
nMotorEncoder[motorA] = 0;  
while (nMotorEncoder[motorB] < 10);
```

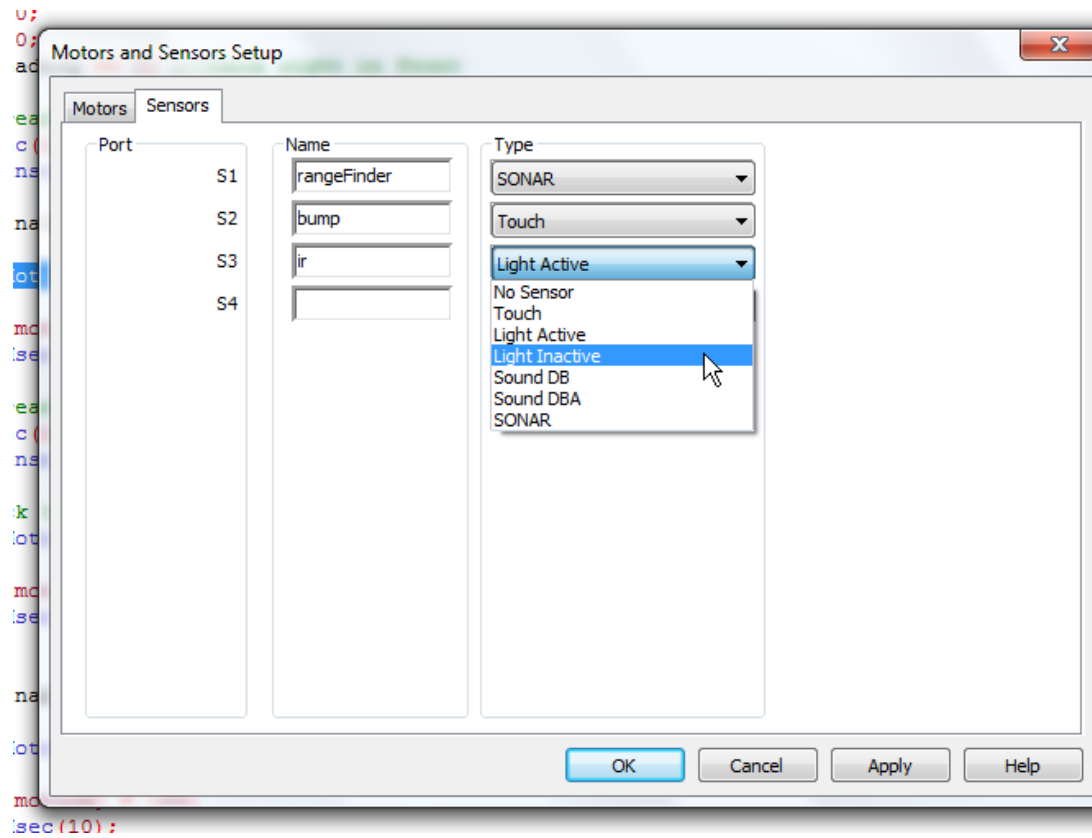
Sensors

- Touch, sound, IR, sonar, among others.
- Must first declare the sensors



Sensors

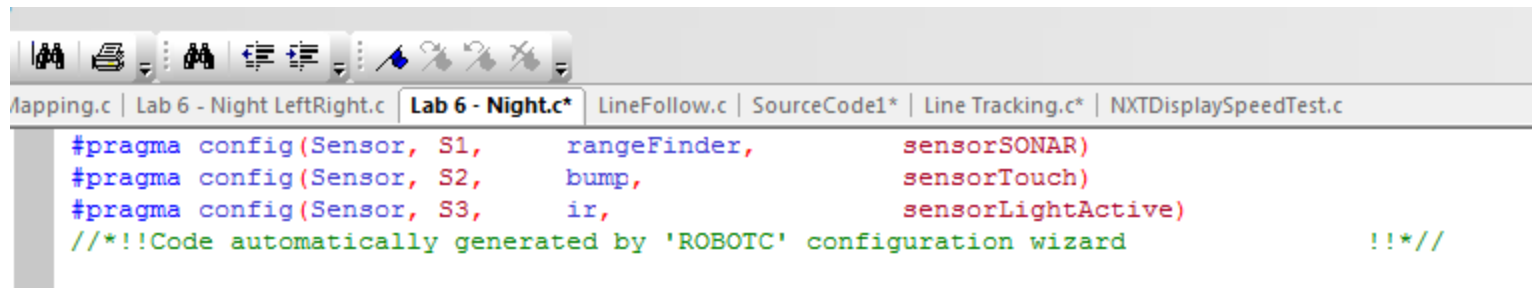
- Select and name the desired sensors



- 4 sensor inputs (S1, S2, S3, and S4)

Sensors

- Pragas generated at the top of the file by the setup:

A screenshot of a code editor window. The title bar shows several tabs: 'mapping.c', 'Lab 6 - Night LeftRight.c', 'Lab 6 - Night.c*' (selected), 'LineFollow.c', 'SourceCode1*', 'Line Tracking.c*', and 'NXTDisplaySpeedTest.c'. The editor displays the following C code:

```
#pragma config(Sensor, S1, rangeFinder, sensorSONAR)
#pragma config(Sensor, S2, bump, sensorTouch)
#pragma config(Sensor, S3, ir, sensorLightActive)
/**!!Code automatically generated by 'ROBOTC' configuration wizard **!
```

- Can now read data from these sensors:

```
int val = SensorValue[rangeFinder];
while (SensorValue[ir] > 0);
```


Printing to the Screen

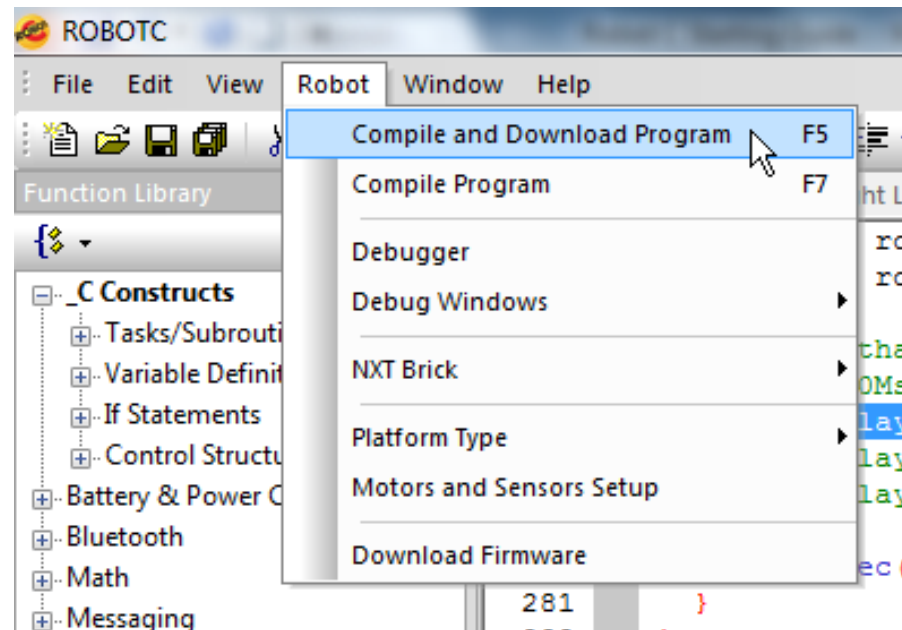
- Primarily line printing:

```
nxtDisplayTextLine(0, "X value = %f", robot_X);  
nxtDisplayTextLine(1, "Y value = %f", robot_Y);  
nxtDisplayTextLine(2, "Theta (degrees) = %d", robot_Y);
```

- Parameters: row, format_string, vargs

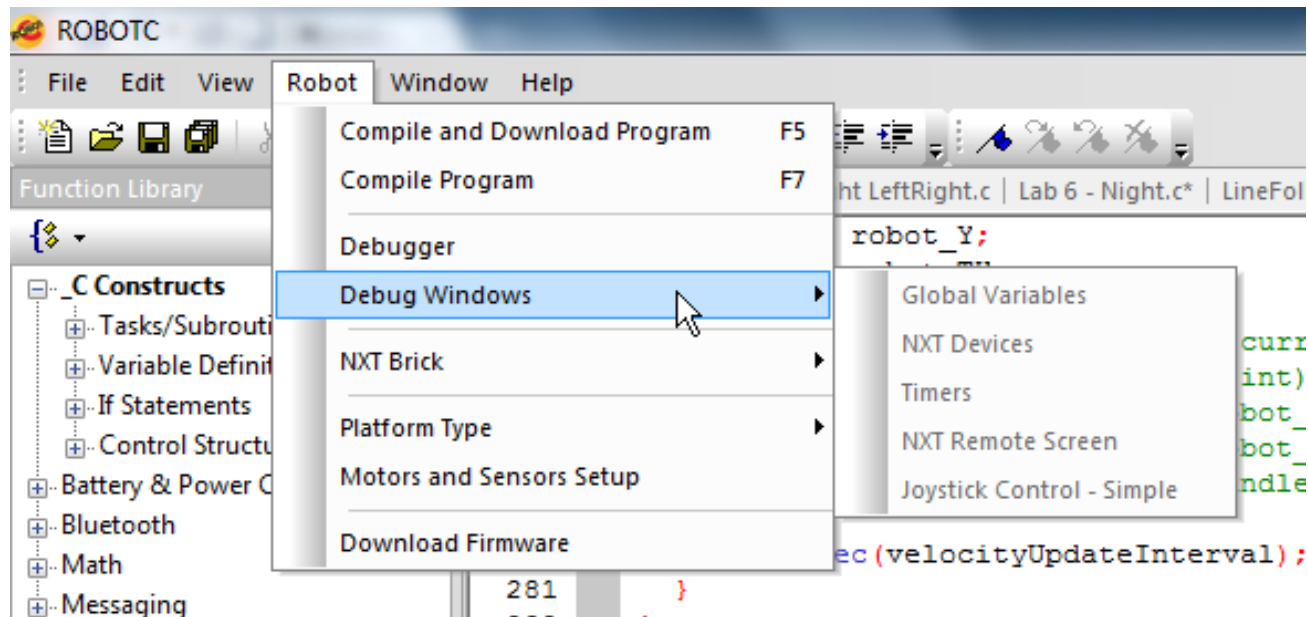
Downloading Code

- Must first establish the link with the NXT
 - USB cable (fast & easy)
 - Bluetooth (slower & longer to setup, but convenient!)
- Compile program (F7)
- Compile & Download (F5)



Debugging

- Robot C offers very good debugging tools. Use them!
- These include
 - Global variable viewer
 - Remote screen viewer
 - Joystick control



Advanced Topics

- “Threading”
 - Uniprocessor concurrency

```
task dead_reckoning()  
{  
    /* Read sensor data */  
}  
  
task main()  
{  
    StartTask(dead_reckoning);  
  
    /* Plan path */  
}
```

Advanced Topics

- Timers
 - Can keep track of 1, 10, or 100 milliseconds
 - 4 timers: T1, T2, T3, and T4

```
int oneMsTime = time1[T1];  
int tenMsTime = time10[T2];  
int oneHundredMsTime = time100[T3];
```

- Can wait 1 or 10 milliseconds

```
wait1Msec(num_millis);  
wait10Msec(num_ten_millis);
```

Advanced Topics

- PID Controller
 - Applied to individual motors

```
nMotorPIDSpeedCtrl[motorA] = mtrSpeedReg;  
nMotorPIDSpeedCtrl[motorB] = mtrSpeedReg;  
nPidUpdateIntervalNxt = update_interval_in_millis;
```

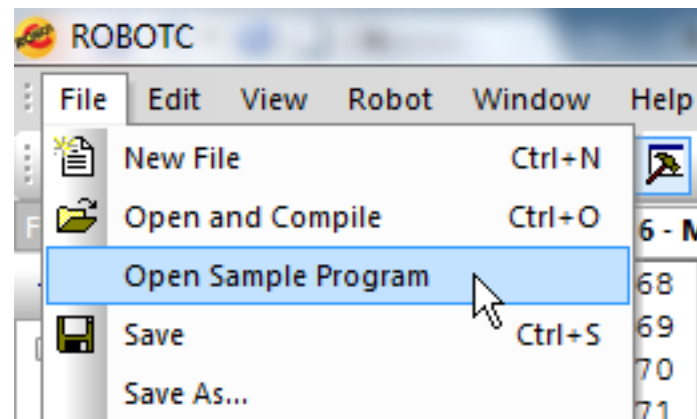
- **NOTE:** `mtrSpeedReg` is a global constant, not something defined by you.

Advance Topics

- [Insert your hack here]
- Robot C and the NXT allow you to do some pretty powerfull stuff if you invest the time:
 - Your own sensors/drivers
 - I2C messaging
 - File IO
 - Pixel drawing on the screen

Tips

- Keep it simple
 - Easier to debug issues
 - More likely to work effectively
- Check out the Sample Programs:



License for Robot C

License ID: XXXXXXXX

Password: (See Howie)

Customer ID: XXXXXXXX

Company Name: CARNEGIE MELLON

Contact Name: HOWIE CHOSET

Questions?

- generalrobotics.org
- 16311-s10-tas@lists.andrew.cmu.edu