FLL Programming 101
NXT-G

September 2007
Version 1.1b
Legal Stuff

© 2006 INSciTE in agreement with, and permission from FIRST and the LEGO Group. This document is developed by INSciTE and is not an official FLL document from FIRST and the LEGO Group. This work is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike License. To view a copy of this license, visit

http://creativecommons.org/licenses/by-nc-sa/2.0/

or send a letter to Creative Commons, 559 Nathan Abbott Way, Stanford, California 94305, USA.

LEGO®, ROBOLAB, and MINDSTORMS™ are trademarks of the LEGO Group used here with special permission. FIRST™ LEGO® League is a trademark owned by FIRST (For Inspiration and Recognition of Science and Technology) and the LEGO Group used here with special permission. INSciTE™ is a trademark of Innovations in Science and Technology Education.

INSciTE
PO Box 41221
Plymouth, MN 55441
www.hightechkids.org
High Tech Kids is committed to making the best possible training material. Since HTK has such a dynamic and talented global community, the best training material and processes, will naturally come from a team effort.

Professionally, the open source software movement has shown that far flung software developers can cooperate to create robust and widely used software. The open source process is a model High Tech Kids wants to emulate for much of the material we develop. The open source software license is a key enabler in this process. That is why we have chosen to make this work available via a Creative Commons license. Your usage rights are summarized below, but please check the complete license at: http://creativecommons.org/licenses/by-nc-sa/2.0/.
This presentation was developed by Doug Frevert. It is based on the work of Fred Rose. The accompanying labs were originally done in RCX Code by Joel Stone and converted to ROBOLAB by Doug Frevert. A portion of the material is taken from “Building LEGO Robots for FIRST LEGO League” by Dean Hystad. Amy Harris defined the 10 programming steps. Eric Engstrom, Jen Reichow, and Ted Cochran reviewed ongoing drafts. Eric taught the first class and helped modify the content accordingly.
Objective
- Learn to program robots.
  - FLL, NXT, NXT-G

Structure
- Theory
- Examples specific to NXT-G
- Hands-on

This class is
- An approach to programming

This class is not
- Exhaustive reference on NXT-G
Class Agenda

- Computer Basics
- The Programming Environment
- Moving and Turning
  - Lab #1
- MyBlocks and Loops
  - Lab #2
- Sensors
  - Lab #3
- Advanced Topics
  - Problem Solving
Computer Basics
The Computer (Generic)

- The processor executes commands.
- Memory stores program and data.
- Inputs tell the computer about the world.
- Outputs tell the world about the computer.
Processor: 32 bit ARM Atmel AT91SAM256 running at 50 Mhz
Memory: 64K Static RAM, 256K Flash
Firmware Loaded?

- Firmware must be loaded onto an NXT so that the NXT can understand your programs.
- Only required to be loaded
  - To install a new firmware release
  - To restore lost firmware (was a problem for RCX).
Writing a Computer Program

- Specify the task
  - Inputs to be supplied
  - Outputs to be produced
- Devise an algorithm
- Express that algorithm in a computer language

From: Introduction to Pascal, Welsh and Elder
Running a computer program (NXT)

1. Write program on PC (NXT-G)
2. Program conv. to bytecodes (text description)
3. Download to the NXT
   - SetPower(A,3)

Bytecodes converted to ARM (NXT) machine code commands

NXT executes commands

Move X to register 001 100011...
NXT-G
The Programming Environment
NXT-G Opening Workspace

Profiles

Program Blocks

Select a Program Name

Block Settings

Pre-built Robots, Programs, and Challenges

Help and Zoom Panel
NXT-G Work Space

Add Program Blocks here

Download and Run

a. Common,
b. Complete, and
c. Custom tabs

View NXT. See the next slide.
NXT-G to NXT

Communications

When connected by USB cable or Bluetooth
Give your NXT a new name
Check Battery voltage
View available memory (in KiloBytes)
Firmware version
NXT-G to NXT

Memory

Select, then delete Programs, Sounds, Graphics, and Unused files.

Can free up to 130Kb of Free Storage on the NXT.
NXT-G Work Space

- **Pointer Tool**
- **Comment**
- **Grab Hand to move the display**
Tips and Tricks (1)

- The NXT has memory to store many programs
  - NXT automatically powers down.

- Bluetooth Communications
  - The NXT has bluetooth communications. If enabled, PCs, NXTs and other bluetooth devices can talk to each other.
  - Disable bluetooth during competition.
Tips and Tricks (2)

- Direction of connecting wires
  - NXT wire connectors only fit one way. Can not be rotated.

- Batteries
  - AA
  - No worry about losing firmware.
  - NiMH rechargeable batteries work. NiCads don't.
  - Lithium rechargeable batteries come with the FLL Mindstorm kits.
  - Avoid stalling the motors, it drains batteries.
Common Blocks

- Common blocks are full featured actions
  - Like English statements
    - Move
    - Wait for an action
    - Display a value
  - With many modifiers
    - Move direction, steering, distance, motors used . . .
    - Wait for light sensor, light threshold, sensor port, . . .

Move Block

Move Block Settings
Common Blocks

- Move
- Sound
- Display
- Wait For Touch
- Switch (Touch)
Adding a Block to a Program

- Click on a Block
- Move cursor onto Program and drop it into place. NXT-G will make room.
- Change settings
WaitFor Blocks

- Click on the hourglass.
- Click on a WaitFor Block
- Time
- Touch
- Light
- Sound
- Distance (Ultrasonic sensor)
- Others (see settings)
Motors

- 9 volt geared motor
  - Making the motors turn is the output of your program. It makes your creation a robot!
  - Without load, motor shaft turns at about 150 rpm.
  - Servo sensitive to 1 degree.
  - With a typical robot, 3-4 hours on a set of batteries.
- FLL allows up to 3 motors.
Motor Details

- Motor can be set to different power settings
  - Power levels 0-100
  - Power is adjusted by Pulse Width Modulation
- Turning the power setting up higher essentially makes the shaft turn faster.
Using the Move Block

- **Ports** A, B, and/or C: Forward, Backward or Stationary
- **Power** 0-100%
- **Duration** Time, Degrees, Rotations, Forever
- **Steering** Spin, Pivot, Arc, Straight
- **Brake or Coast**
A sensor is built into each motor.

Measure by rotations or degrees.
Lab One

Task:
Given rectangle ABCD.
Move from A to B to C to D.
Extra Credit: A to C to D to A.
Lab 1  An Answer
Problem Solving
Generic Problem Solving Process

- Define the problem
- Brainstorm solutions
- Evaluate solutions  Pick one
- Try (implement) it
- Evaluate results

- Express the solution as an algorithm, then convert it into a computer program.
Pseudocode: Not software code, not English, but somewhere in between.

Count Eggs
Add Ingredients
Mix and Strain
Dip, Fry, Sprinkle
Serve

set counter to zero
for each egg
increment counter
crack open into bowl
next
do counter times
add ½ cup milk
add ¼ cup flour
end do
do counter/2 times
do 4 times
dip 2 slices of bread
fry, sprinkle, serve
end do
end do
Flowcharts: A graphic representation of logic. Convert from pseudocode. One step closer to software.

set counter to zero
for each egg
  increment counter
  crack into bowl
next
Debugging and Analysis

- Split into pieces. Test each piece.
- Do little pieces at a time
  - For example, get the robot to where it needs to be first, then work on getting it to do something
- Reuse pieces that work
  - For example, you know how to turn 90°
- Brainstorm a new solution
- Look outside your box.
  - Ask for help.
Keep It Simple Strategies

KISS #1: Subroutines
#2: Comments
#3: Loops
KISS #1: Subroutines

- Wrap a complicated process into a neat and tidy package.
- Once wrapped, just worry about the package.

- In NXT-G, Subroutines are MyBlocks
  - Select from the Custom Tab
Subroutines: When to Use

- To do the same thing from different places.
  - Reuse

- To divide a task into pieces.
  - Modules

- To hide complex details.
MyBlock Names

- Useful and informative
  - ClearSoccerField  *not* Csf_amy_3a
  - 12 characters visible on a MyBlock
  - 15 characters visible on the NXT

- Suggest using “action + to + target”:
  - Fwd2Wall or ForwardToWall or Forward_To_Wall
  - FwdDist
  - TurnRight

- Name the task accomplished, not how it was done.
  - FollowLine  *not* FollowLine1LightSensor
MyBlock Creation 1

- For each parameter, add a variable.

- Add and wire Variable Blocks to your program.

Create, name, and select the datatype.

Click here to reveal wire connections.
MyBlock Creation 2

- Select blocks to include.

- Click the Create My Block button.
MyBlock Creation 3

- Name the MyBlock.
- Give it a description.
- Click “Next” to add an icon.
MyBlock Creation 4

- MyBlock replaces the selected Blocks!

- To add the new MyBlock to a program, select it from the Custom Tab.

Can only set the duration.
KISS #2: Comments

- Explain the program.
- Programmers forget.
- Teams compete in FLL. More than one person will be working on the program.

Enter comments with the balloon text tool
- Add who, when, how to use, assumptions taken, and expected results.
KISS #3: Loops

• Loops are a control structure
  • In other programming languages:
    For … Next    Do loop n times
    Do ... Until    Do it. Unless some test, do it again.

• There are loops for
  • Forever
  • Every sensor (including time)
  • Logic
  • Count
Simple Loop

- Convert this
- to something simpler using a loop
Lab Two

Task:
Make Forward and Right Turn MyBlocks.

Move around the rectangle twice, ending at A.
Lab 2 Answer

One possible answer.
Data Input

Sensors
Sensors

- Allow your robot to detect the real world.
  - Touch
    - Has your robot made contact with something?
  - Light
    - Is the surface light or dark?
  - Sound (Microphone)
  - Ultrasound (Distance)
  - Rotation
    - Embedded in the motors
  - Time
    - Internal sensor, keeps track of time
  - Battery Voltage
Sensor #1: Touch

- To detect touching or bumping into something.
- Good for detecting robot arm movements. The sensor activates when the arm moves far enough to push in the touch sensor.
Pressed, Released, Bumped

Bumped: pressed and released in any order.
Touch Sensor WaitFor Block

Robot waits until sensor responds

Pressed, released, or bumped
Touch Sensor Switch Block

- Waiting for a touch sensor can be useful, but many times you want to do different things based on the current value.
Touch Sensor Loop Commands

- Loop until a touch sensor is pressed. Useful if the loop contains commands that may be repeated.

For instance, a routine that beeps until a bumper hits something.
Sensor #2: Light

- Operates in "percent" mode
  - 0 to 100
  - Higher number = more light. A lighter surface reflects more light.
- Calibrate the sensor.
- Light can be turned off.
- Shines a red light.
Light Sensor Spectrum

- Most sensitive to red/IR light.
Light Sensor Readings

- Lowest likely reading 5%
- Highest likely reading 100% (pointing at a light)
- Readings also depend on the color of the surface
  - See “Building LEGO Robots for FIRST LEGO League” by Dean Hystad.
- Sensitive to the distance between the sensor and the reflecting surface. Variations can make the readings unusable. Keep the sensor close to the surface, but not too close.
- Shield the sensor from other light sources.
Light Sensor Readings

- The light sensor averages its readings over roughly a circular area.
- Cross a line too fast and you may miss the line.
- Test and recalibrate on competition day.
Light Sensor WaitFor Block

Use WaitFor Blocks when watching only one sensor.

Turn light on/off
Light Sensor Switch Block

- If brighter than 50, turn motor B on. Stop C.
- Otherwise, turn motor C on. Stop B.

What is this?

Switch a.k.a. If/then/else

Outside Loop
Calibrate Light Sensor

- Move your robot over light and dark areas.
- Resets light sensor percentages.
- Can also be done on the NXT.
Sensor #3: Rotation

- Measures how far a rotating axle has turned. As the axle turns, a counter in the NXT is incremented or decremented.
- 360 counts per rotation.
- Each motor has an embedded rotation sensor.
Rotation Sensor Loop Block

- Possible to make a rotation loop.
  - Select any loop. Pick “Sensor” type and then “Rotations.”
The rotation sensor also brings in the possibility of doing some real math! We’ll leave that as an exercise for the reader! Of course, trial and error also works.

Sources of error in calculation - dirt on surface, using a skid rather than a wheel, backlash (poor fitting gears).
More on Rotation Sensor

- Rotation sensor **counts forward and backwards.**
- Live updates (Bluetooth or USB connected).

![Diagram of Rotation Sensor](image.png)
Debugging and Analysis

- Common problems
  - Programming: *reset the sensor to zero* before use.
  - Design: inadequate sensor resolution (trying to measure something very accurately, when the sensor is not that accurate).
  - Control: starting, stopping, turning *too fast*.
  - Variations in the initial conditions: not putting everything in the same place before pushing the run button.
Sensor #4: Timer

- 3 Timers
  - Time in thousandths of a second
  - Greater or Less than test
  - Reset
  - Trigger Point
  - Timer Number
  - Yes/No
  - Time as a number
Other Sensors

- **Sound (microphone)**
  - Is the FLL competition too loud?

- **Ultrasonic (distance)**
  - Interference with other NXTs?

- **NXT Buttons**
  - One touch running of the next program.

- **Received Messages**
Keep it Simple Strategies

KISS #4: Variables
#5: Parallel Sequence
What’s a Variable?

A value that you can change during your program.
  - This value is “variable”, hence the name.

For example, your program may store a light sensor reading in a variable called *LightBright*. Use that value later.

Use a meaningful name.

Useful to pass values to MyBlocks.
KISS #4: Variables

- Creating a variable
  - `<Edit>`<Define Variables>
  - Create
  - Name the variable
  - Select a datatype
    - Number
    - Text
    - Logic
Using a Variable

Use the Write action

Pick a variable

Set the value
Using a Variable

Pick a variable

Use the Read action

Wire variable value to motor duration
KISS #5: Parallel Sequences

- Make two tasks that run independently. (Can you walk and chew gum?)
- One task lifts the arm. The other task heads for home.
Lab Three

Task:
Use a light sensor to detect the lines.
Move from A to B to C to D.
Extra Credit: While moving, touch the touch sensor to stop.
Lab 3  An Answer
Advanced Topics

Debugging Tools
Switch between Cases
File Operations
Bluetooth Communication Setup
Additional Resources
Debugging Tools

- **Music**
  - Use music to identify sections of code.
  - One or two quick notes, a good ear can hear the difference.

- **LCD**
  - Write text, even graphics, to the NXT LCD panel.

- **NXT Live updates**
  - From your PC (or MAC), use NXT-G to view the values of variables and sensors.
Switch Between Cases

One switch block with 3 tabs, one for each case.

Click here to add cases.

Text or number type.

Must not use flat view.
Reset Motor/Move Block

- Motor and Move Blocks are servo controls.
  - Corrects momentum over runs
  - For several blocks in a series, the correction is based on a total rotation target.
  - Even applies across programs
- Use the Reset Motor Block to reset the target.
Arithmetic

Operation

Wired Input

Output

Operation

Output
Combining Structures

Outside Loop runs forever.

Light Sensor Switch runs inside.
Comparing Algorithms

- Compare that line follower to this one:

  Better
  - Simpler. One motor changes speed. Faster.

  Worse
  - Tight corners?
Watching 3 Sensors

Outside Loop (Logical)

Logical ORs combine results into a single True or False
Optimizing Code

- Which is faster? more reliable? best?

- Use the one that makes sense to you, the programmer.
File Operations

- Read
- Write
- Close
- Delete

- Upload/Download
Bluetooth Communication Setup

- On the NXT, select Bluetooth -> On/off -> ON
  Bluetooth -> Visibility -> Visible

- In NXT-G, select a NXT and click “Connect”

- Enter a passkey, like “mysecret” and click “OK.”

- On the NXT a screen will appear. Re-enter the same passkey. (Be quick, you've got 30 sec.)

- After NXT-G shows “connected”, on the NXT, Bluetooth -> Visibility -> Invisible
Lab Four

Task:
Move exactly one lap around an oval.
(Black 2cm line on white paper)
Lab 4  An Answer

- Did you start by copying? Why not?
- Is building a robot like taking a test?