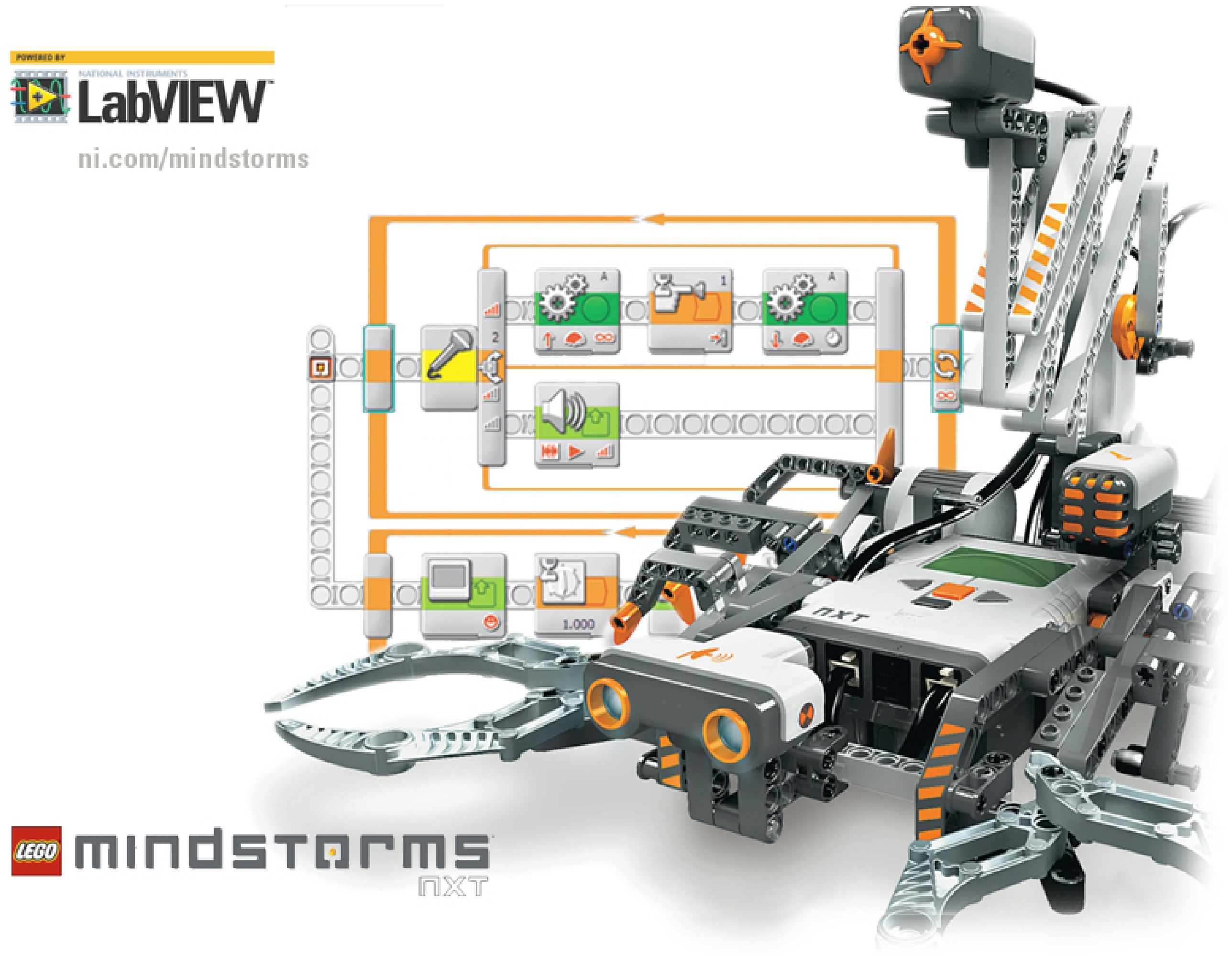
Learning through Problem Solving:

Implementing Lego Mindstorms in the Modern Classroom



Durham District School Board
G. Thompson, Programs Facilitator
Spring 2007

Table of Contents

Learning through Problem-Solving:
The Durham Robotics Initiative

> Research and successful classroom practice have shown that an investigative approach, with an emphasis on learning through problem solving and reasoning, best enables students to develop the conceptual foundation they need.
>
> - The Ontario Curriculum, Grades 1-8: Mathematics p. 24

Students often learn best when using manipulatives to plan, learn and explore their environment. Integrating the practical, logic-oriented Lego Mindstorms in schools provides students and teachers the opportunity to explore and search "for patterns and relationships and engage in logical inquiry" (The Ontario Curriculum: Mathematics p.24). Moreover, boys (as well as many girls) often learn best when engaged in practical, hands-on learning initiatives. The Government of Australia suggests that boys in particular learn best and "respond more positively to learning experiences that:

- have a practical focus and physical or hands-on dimension;
- they see as relevant and having a real world connection;
- use thinking skills focused on actual problems;
- challenge them by requiring higher order and conceptual thinking;
- have clear instructions and structured sessions in manageable chunks;
- enable them to work with others as well as individually;
- provide for a range of ways in which work can be presented; and
- provide them with a degree of involvement in decisions about content and opportunities to negotiate their learning as a valued stakeholder."
  - Australian Government, Department of Education, Science and Training[1]

Naturally, problem-solving learning scenarios permeate Ontario curriculum at both the elementary and secondary level. Moving from the abstract to the concrete across the subject disciplines, Ontario curriculum addresses the need for students of all ages and cultures to pervasively explore challenges using hands-on, tangible learning manipulatives. The goal of this project is, therefore, to create learning opportunities for students that are engaging, align with the curriculum and meet the needs of those learners who learn best in dynamic, real-world learning situations.

What follows is a brief summary of direct links between Robotics learning environments and the Ontario curriculum.

[1] *Australian Government, Department of Education, Science and Training.* 30 October 2006 <http://www.dest.gov.au/sectors/school_education/policy_initiatives_reviews/key_issues/boys_education/guiding_principles_in_educating_boys.htm#Guiding_principles_for_success_in_educating_boys>

Foundations: The Ontario Curriculum

Grades 1-8 : Mathematics

Through mathematical activities that are practical and relevant to their lives, students develop mathematical understanding, problem-solving skills, and related technological skills that they can apply in their daily lives and, eventually, in the workplace (p. 24).

When planning mathematics programs, teachers will provide activities and assignments that encourage students to search for patterns and relationships and engage in logical inquiry (p.24).

Students will investigate mathematical concepts using a variety of tools and strategies, both manual and technological. Manipulatives are necessary tools for supporting the effective learning of mathematics by all students (p. 25).

Common problem-solving strategies include the following: making a model, picture, or diagram; looking for a pattern; guessing and checking; making an organized list; making a table or chart; making a simpler problem; working backwards; using logical reasoning (p. 12).

Research and successful classroom practice have shown that an investigative approach, with an emphasis on learning through problem solving and reasoning, best enables students to develop the conceptual foundation they need (p. 24).

Grade 8 Science: Structures and Mechanisms – Mechanical Efficiency

- describe in quantitative terms the relationship between force, area, and pressure;
- investigate and measure forces that affect the movement of an object (e.g., friction);
- distinguish between velocity and speed (i.e., define velocity as speed in a given direction);
- predict the mechanical efficiency of using different mechanical systems (e.g., a winch).
- plan investigations for some of these answers and solutions, identifying variables that need to be held constant to ensure a fair test and identifying criteria for assessing solutions;
- produce technical drawings and layout diagrams of a structure or a mechanical system that they are designing, using a variety of resources.
- make informed judgements about products designed and made by others;
- evaluate their own designs against the original need, and propose modifications to improve the quality of the products.

Grade 7 Science: Structures and Mechanisms – Structural Strength

- design and make a variety of structures, and investigate the relationship between the design and function of these structures and the forces that act on them;
- demonstrate an understanding of the factors (e.g., availability of resources) that must be considered
- in the designing and making of products that meet a specific need.
- demonstrate awareness that the position of the centre of gravity of a structure (e.g., bridge, building, tower) determines whether the structure is stable or unstable;
- describe, using their observations,ways in which different forces can affect the stability of a structure (e.g., certain forces may cause a structure to shear, twist, or buckle);
- identify forces within a structure that are affected by forces outside the structure (e.g., shear, torsion, tension, and compression within a bridge are affected by external forces such as high wind or ice);
- use appropriate techniques and materials (e.g., cutting and joining pieces of wood or plastic) while making structures that have mechanisms;
- formulate questions about and identify needs and problems related to the strength of structures, and explore possible answers and solutions (e.g., determine what caused structural failure and propose ways of supporting a specific load);
- plan investigations for some of these answers and solutions, identifying variables that need to be held constant to ensure a fair test and identifying criteria for assessing solutions;

- recognize that a solution to a problem may result in creating new problems in other areas, and that a solution to a problem may be found while one is working on solving a problem in another area;

Computer and Information Science: TIK2O

Problem Solving, Logic, and Design

TF1.01I
– use input, processing, and output correctly as a model for solving problems using a computer;
TF1.02I
– explain how clarity at each step in the problem-solving process determines the quality and effectiveness of the final product;
TF1.03I
– define a problem by identifying the required result, the necessary user inputs, and the steps required to produce the result.

Programming Concepts

TF3.01I
– use correct terminology to describe programming concepts;
TF3.02I
– describe the types of data that computers store, including numbers and characters;
TF3.03I
– define constants, variables, expressions, and assignment statements, including the order in which the operations are performed;
TF3.04I
– explain the need for decision and repetition structures, and how they can be expressed in different programming languages;
TF3.05I
– explain the difference between logic and syntax errors;
TF3.06I
– explain the role of internal documentation in ensuring program correctness and clarity.

Coded Expectations, Computer and Engineering Technology: TEE2O

Computer Logic

TF1.01E
– describe the relationship between the binary number system and computer logic;
TF1.02E
– define a standard way of representing characters in binary code;
TF1.03E
– describe the function of the fundamental logic gates, including the function of each pin: AND, NAND, OR, NOR, XOR, XNOR, and NOT.

Programming Concepts

TF3.01E
– define constants, variables, expressions, and assignment statements, including the order in which the operations are performed;
TF3.02E
– describe how computers store and work with different types of data, including numbers and characters.

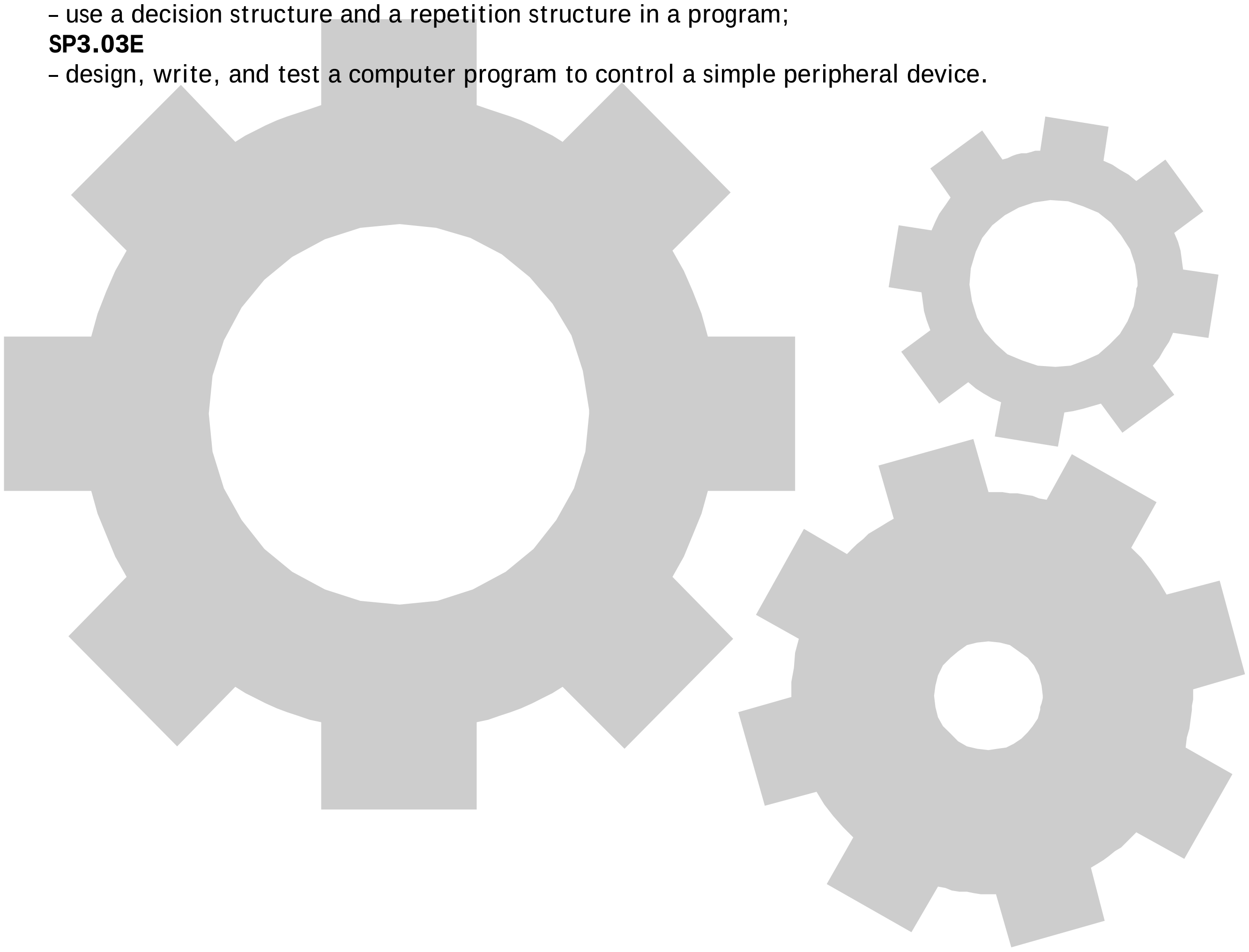
Skills and Processes

Overall Expectations

SPV.01E
– connect and use correctly a variety of computer components and peripherals;

SPV.02E
– demonstrate the use of an operating system, including a network;
SPV.03E
– use logic gates to construct simple circuits;
SPV.04E
– apply fundamental programming constructs to develop programs that interact with external components.

Programming Concepts
SP3.01E
– use input and output statements in a program;
SP3.02E
– use a decision structure and a repetition structure in a program;
SP3.03E
– design, write, and test a computer program to control a simple peripheral device.



An NXT Overview

The Mindstorms NXT kit comes with a series of 3 Servo motors, 4 Sensors and a CPU (typically referred to as the 'brick' or NXT). The brick connects to a computer via USB cable and can acquire software downloads from this machine. In order for these two machines to properly interface, the PC must have the following minimum specifications:

Technical specifications

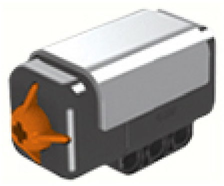
- Pentium class 4 Microprocessor
- 256 Kbytes FLASH, 64 Kbytes RAM
- 8-bit AVR microcontroller
- 256K Memory
- Bluetooth wireless communication (Bluetooth Class II V2.0 compliant)
- USB port
- 4 input ports, 6-wire cable digital platform (One port includes a IEC 61158 Type 4/EN 50 170 compliant expansion port for future use)
- 3 output ports, 6-wire cable digital platform
- 100 x 64 pixel LCD graphical display
- Loudspeaker - 8 kHz sound quality. Sound channel with 8-bit resolution and 2-16 KHz sample rate (Sound Blaster Compatible).
- Power source: 6 AA batteries [2]

Testing the Sensors

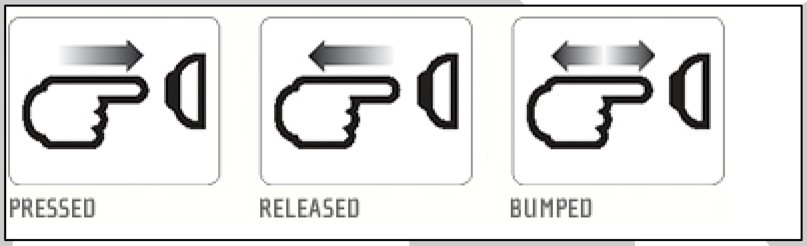
Each of the sensors can operate as part of a program or independently when connected to the NXT. For example, the Sound Sensor, when attached to the NXT can determine the decibel level of various noises in a room. Or, as part of a program, the Sound Sensor could direct the NXT to initiate a Servo Motor when the decibel level of the room reaches a certain level. The following 'Knowing the Sensors' section provides detailed information on how the sensors can operate apart from a program.

[2] *Mindstorms – Home – Overview*. October 28 2006 <http://mindstorms.lego.com/Overview/The_NXT.aspx>

Knowing the 5 Sensors:



The Touch Sensor gives your robot a sense of touch. The Touch Sensor detects when it is being pressed by something and when it is released again.



Suggestions for use
You can use the touch Sensor to make your robot pick up things: a robotic arm equipped with a Touch Sensor lets the robot know whether or not there is something in its arm to grab. Or you can use a Touch Sensor to make your robot act on a command. For example, by pressing the Touch Sensor you can make your robot walk, talk, close a door, or turn on your TV.

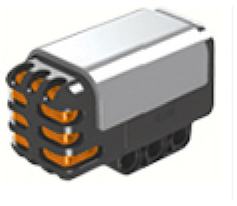
Try Me

The NXT comes with a Try Me function. Connect a Touch Sensor to port 1 of the NXT and select the Try Me submenu on the NXT to test your Touch Sensor. You'll get a fun reaction.[3]

[3] *Mindstorms – Home – Overview*. October 28 2006 <http://mindstorms.lego.com/Overview/Touch_Sensor.aspx>

SOUND SENSOR

The Sound Sensor makes your robot hear!



The Sound Sensor can detect both decibels [dB] and adjusted decibel [dBA]. A decibel is a measurement of sound pressure.

dBA: in detecting adjusted decibels, the sensitivity of the sensor is adapted to the sensitivity of the human ear. In other words, these are the sounds that your ears are able to hear.

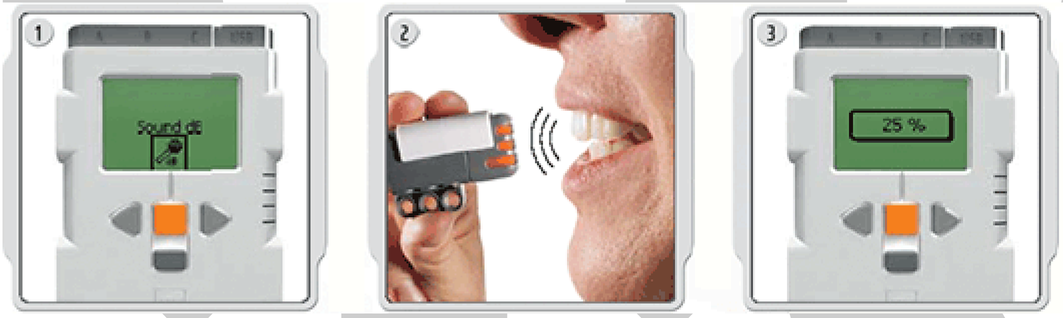
dB: in detecting standard [unadjusted] decibels, all sounds are measured with equal sensitivity. Thus, these sounds may include some that are too high or too low for the human ear to hear.

The Sound Sensor can measure sound pressure levels up to 90 dB – about the level of a lawnmower. Sound pressure levels are extremely complicated, so the Sound Sensor readings on the MINDSTORMS NXT are displayed in percent [%]. The lower the percent the quieter the sound. For example:

• 4-5% is like a silent living room
• 5-10% would be someone talking some distance away
• 10-30% is normal conversation close to the sensor or music played at a normal level
• 30-100% are people shouting or music being played at a high volume

Test it!

Test the Sound Sensor's ability to read sound volume:



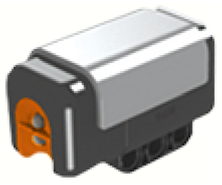
Connect the Sound Sensor to the NXT.
1. Select the View submenu on the NXT's display. Select the Sound Sensor icon and the port where you have connected the sensor.
2. Make some sounds into the Sound Sensor and watch the readings displayed on the NXT. Use the sensor to read some sounds around you.
3. How loud do your parents speak? How loud is your front doorbell?

Try Me

The NXT comes with a Try Me function. Connect a Sound Sensor to port 2 of the NXT and select Try Me submenu on the NXT to test your Sound Sensor. You'll get a fun reaction.[4]

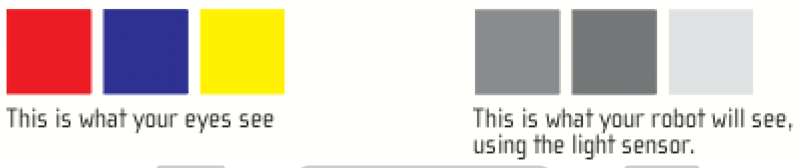
[4] *Mindstorms – Home – Overview*. October 28 2006 <http://mindstorms.lego.com/Overview/Sound_Sensor.aspx>

LIGHT SENSOR



The Light Sensor is one of the two sensors that give your robot vision (the Ultrasonic Sensor is the other). The Light Sensor enables your robot to distinguish between light and dark. It can read the light intensity in a room and measure the light intensity of colored surfaces.

This is what your eyes see


This is what your eyes see
This is what your robot will see, using the light sensor.

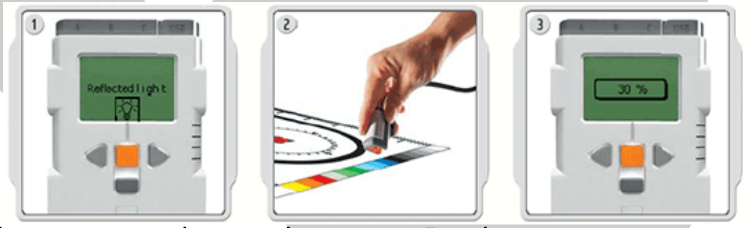
This is what your robot will see, using the light sensor.

Suggestions for use
You can use the Light Sensor to make a burglar alarm robot: when an intruder turns on the light in your room the robot can react to defend your property. You can also use the Light Sensor to make a line-following robot or a robot that can sort things by color.

Detecting ambient [surrounding] light
Test the Light Sensor's ability to read ambient light by measuring the light level in different locations of the room. For example, first hold the sensor against the window. Then hold it under the table. Watch how the readings differ.

Test it!

Test Light Sensor readings. Here's how:



1. Connect the Light Sensor to the NXT.

2. Select the View submenu on the NXT display. Select the Light Sensor icon and the port where you have connected the sensor, and press the orange Run button.

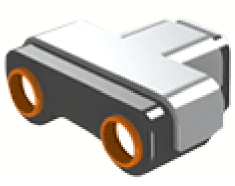
3. Hold the Light Sensor up to the different colors on the test pad that came with your kit and see the different readings.

Try Me

The NXT comes with a Try Me function. Connect a Light Sensor to port 3 of the NXT and select the Try Me submenu on the NXT to test your Light Sensor. You'll get a fun reaction.[5]

[5] *Mindstorms – Home – Overview*. October 28 2006 <http://mindstorms.lego.com/Overview/Light_Sensor.aspx>

ULTRASONIC SENSOR



The Ultrasonic Sensor is one of the two sensors that give your robot vision (the Light Sensor is the other). The Ultrasonic Sensor enables your robot to see and detect objects. You can also use it to make your robot avoid obstacles, sense and measure distance, and detect movement.

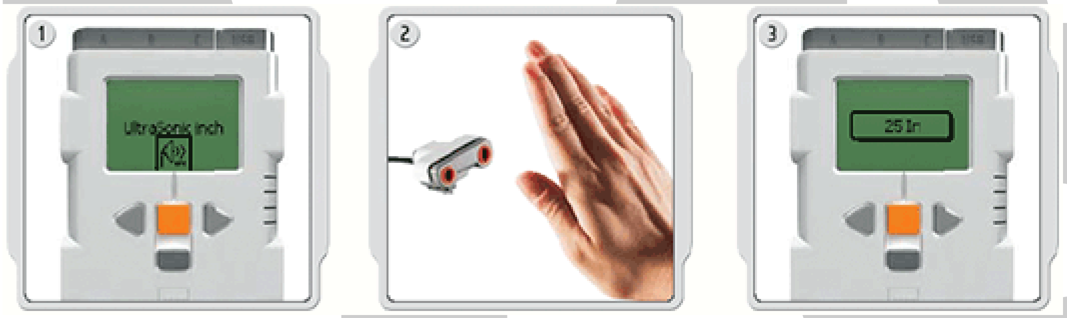
The Ultrasonic Sensor measures distance in centimeters and in inches. It is able to measure distances from 0 to 255 centimeters with a precision of +/- 3 cm.

The Ultrasonic Sensor uses the same scientific principle as bats: it measures distance by calculating the time it takes for a sound wave to hit an object and return – just like an echo.

Large sized objects with hard surfaces return the best readings. Objects made of soft fabric or that are curved [like a ball] or are very thin or small can be difficult for the sensor to detect.

*Note that two or more Ultrasonic Sensors operating in the same room may interrupt each other's readings.

Test it!
Test the Ultrasonic Sensor's ability to measure distance:



1. Connect the Ultrasonic Sensor to the NXT

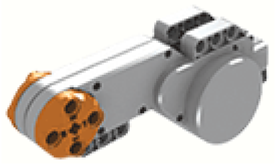
2. Select the View submenu on the NXT. Select the Ultrasonic Sensor icon and the port where you've connected the sensor.

3. Measure the distance to an object. For example, move your hand closer to the sensor and watch the readings change.

Try Me
The NXT comes with a Try Me function. Connect an Ultrasonic Sensor to port 4 of the NXT and select the Try Me submenu on the NXT to test your Ultrasonic Sensor. You'll get a fun reaction. [6]

[6] *Mindstorms – Home – Overview*. October 28 2006 <http://mindstorms.lego.com/Overview/Ultrasonic_Sensor.aspx>

SERVO MOTORS

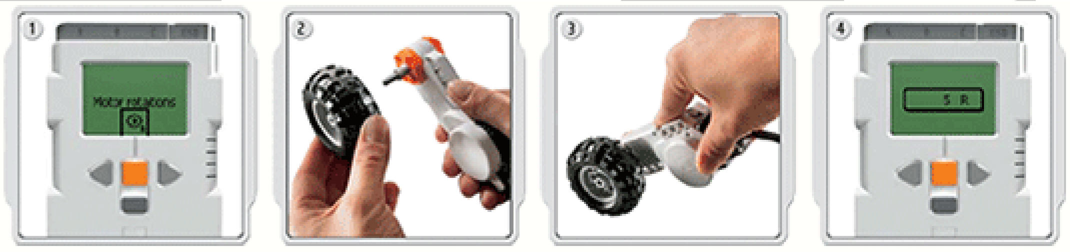


The three Servo Motors give your robot the ability to move. If you use the Move block in the LEGO MINDSTORMS NXT software to program your motors, the two motors will automatically synchronize, so that your robot will move in a straight line.

Built-in Rotation Sensor
Each motor has a built-in Rotation Sensor. This lets your control your robot's movements precisely. The Rotation Sensor measures motor rotations in degrees or full rotations (accuracy of +/- one degree). One rotation is equal to 360 degrees, so if you set a motor to turn 180 degrees, its output shaft will make half a turn.

The built-in Rotation Sensor in each motor also lets you set different speeds for your motors [by setting different power parameters in the software]. Try running the motors at different speeds.

Test it!
Test the built-in Rotation Sensor's ability to measure distance:



1. Connect a motor to the NXT
2. Select the VIEW submenu on the NXT
3. Select the Motor Rotations icon.
4. Select the port where you have connected the motor
5. Attach a wheel to the motor and measure the rotations as you roll the wheel along the floor

Try Me

The NXT comes with a Try Me function. Select the Try Me submenu on the NXT display and test your motors. You'll get a fun reaction.[7]

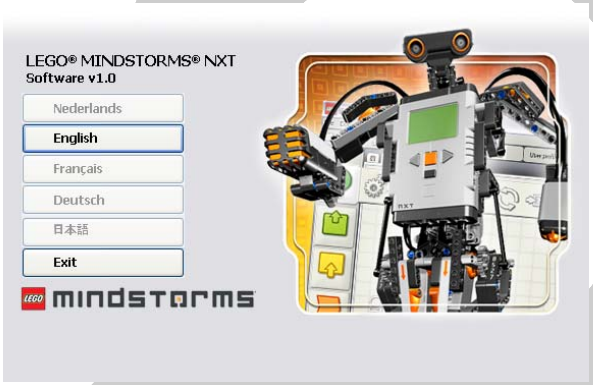
[7] *Mindstorms – Home – Overview*. October 28 2006 <http://mindstorms.lego.com/Overview/Servo_Sensor.aspx>

Installing Mindstorms NXT Software

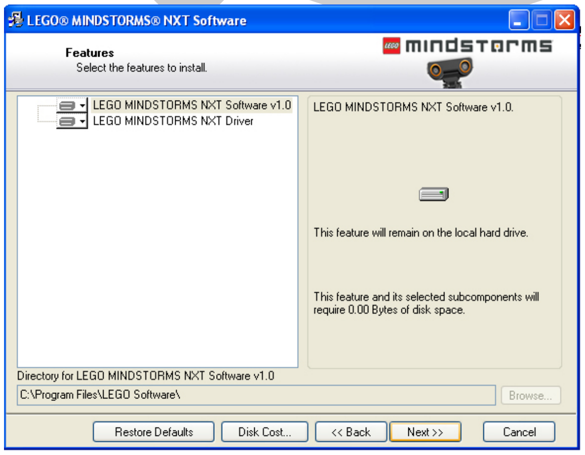
The LEGO MINDTSORMS NXT Software is powered by NI LabVIEW, an intuitive graphical programming software used by scientists and engineers worldwide to design, control and test consumer products and systems such as MP3 and DVD players, cell phones, and vehicle air bag safety systems. Applications include helping to control the NASA Mars Pathfinder exploration to testing the Microsoft Xbox.[8]

To program the NXT with any degree of complexity, it is necessary to install the Mindstorms NXT software. To preview this software before installing to review functionality, visit an online demo at http://www.ni.com/academic/mindstorms/.

When you are ready to install the Mindstorms NXT software, simply insert the CD, 'Lego Mindstorms NXT Software' into the computer's CDROM drive and wait for the screen prompt illustrated below.



At this point, select your preferred language and wait for the next install box to appear. You may receive a warning that your CPU speed is too slow, but this can often be ignored if you're certain your computer contains a faster processor than is being reported.


LEGO® MINDSTORMS® NXT Software

Features
Select the features to install.

LEGO MINDSTORMS NXT Software v1.0
LEGO MINDSTORMS NXT Driver

LEGO MINDSTORMS NXT Software v1.0.

This feature will remain on the local hard drive.

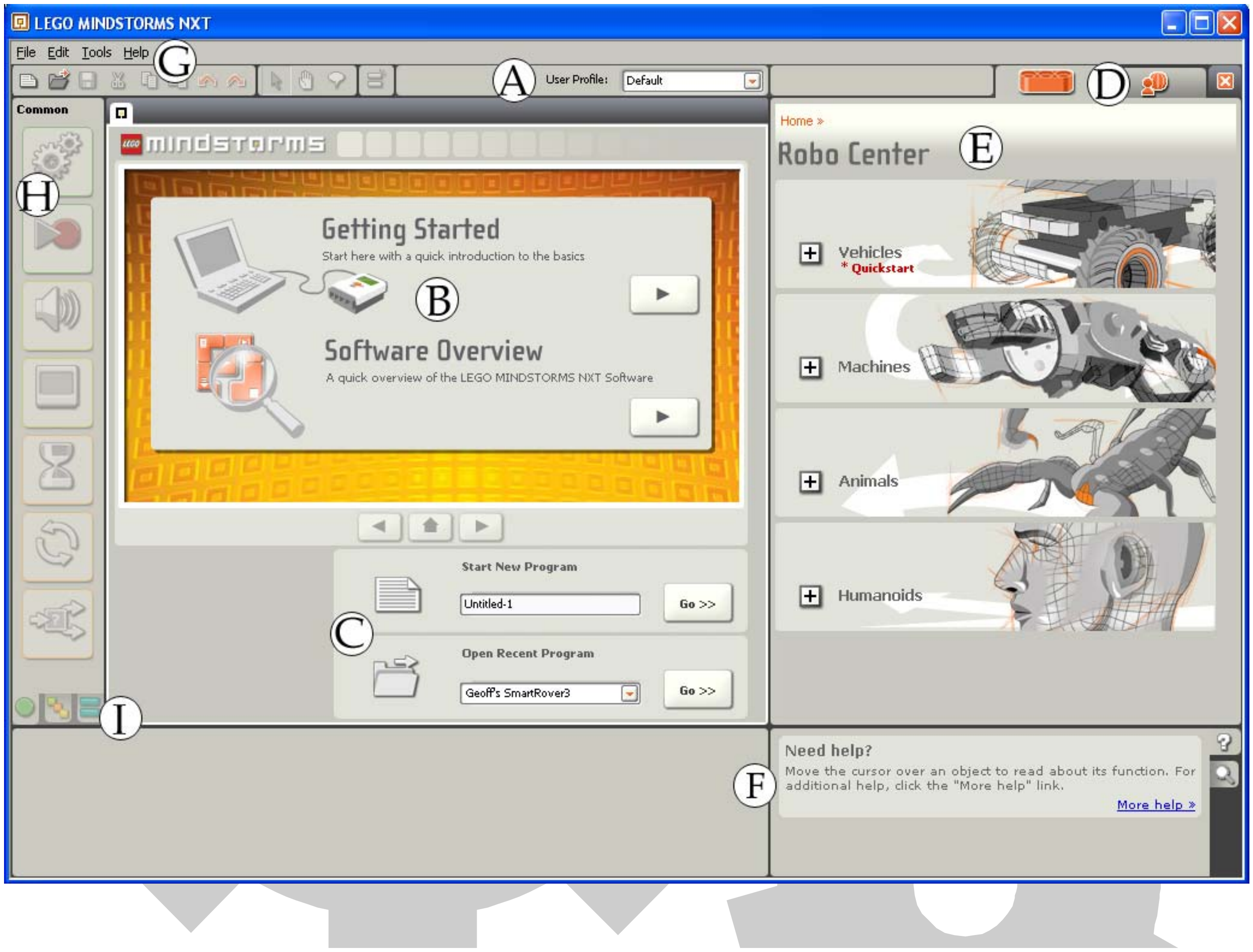
This feature and its selected subcomponents will require 0.00 Bytes of disk space.

Directory for LEGO MINDSTORMS NXT Software v1.0
C:\Program Files\LEGO Software\

On this screen, the install application will explain how much space is available on the computer's hard drive and how much space is necessary to install the Mindstorms NXT software. When you are comfortable, click next and the software will be installed to your PC.

[8] *Mindstorms – Home – Overview*. October 28 2006 <http://mindstorms.lego.com/Overview/NXT_Software.aspx>

An Overview of the Mindstorms NXT Software Interface



A – Allows the user to select his/her profile
(configurable in the Edit Menu under 'Manage Profiles')

B – Clicking the 'Play' button beneath either of these topics launches a Tutorial on getting to know the NXT for the first time.

C – Clicking Go next to one of these fields will either create a new Program or open an old one.

D – These two graphics toggle between the local software and Mindstorms Online.

E – The Robo Center contains sample robots and programs that one can construct using the NXT kit

F – Clicking 'More help' launches the Mindstorm NXT Help and Support Documentation from Lego

G – The Tools Menu is particularly helpful when calibrating the sensors of the NXT

H – These icons are used when Programming the robot and represent servos, sensors, display options and Programming Options

I – These 3 icons allow the user to browse through basic and advanced programming constructs.

An Introduction to Programming

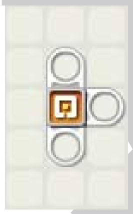
The Mindstorms programming environment is, at its core, a graphical user interface (GUI) that provides users the opportunity to write rudimentary as well as advanced programming constructs using a drag-and-drop technique. Benefiting the user, only a minor understanding of programming is required to get started with the NXT.

Below is an outline of a series of programs users can begin implementing which increase in complexity incrementally.

Linear Programming: Understanding the basics

Before making advanced robots that can tell time, drive around a room and interact with their environment, it is important to understand programming fundamentals. This introductory lesson will guide you in this process.

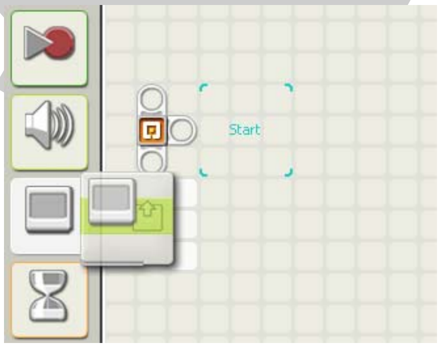
Create a New Program (Area C on the Interface Overview) and call it Basics1. To expand your screen, click the red X next to the My Portal button (Area D on the Interface Overview).



Hover your mouse over top of the orange area of the Lego piece on the screen and notice that Area F immediately changes to a description of the Starting Point. This Help section is ‘mouse aware’ and provides users with assistance on any programming aspect of the NXT and can be accessed whenever the software is launched.

Program A: Displaying and Talking

This program displays a message to the viewer and makes the NXT speak.

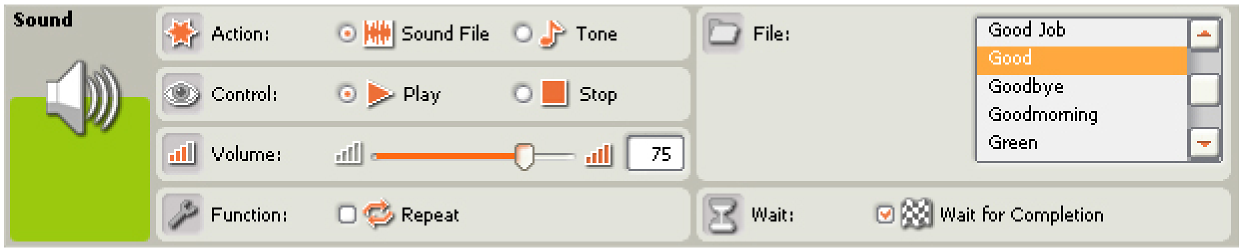


1. Click on the Display icon and drag it into the Start area as indicated by the blue text on the screen.

2. In the Display Properties area at the bottom of the screen, select an image from the File Scroll box (i.e. ‘Smile 01’).

3. Drag the Sound icon so that it is joined to the right of the Display icon on the Sequence Beam.

Program A: Displaying and Talking (continued)


Sound

Action: [✓ Sound File] Tone

Control: [✓ Play] Stop

Volume: 75

Function: Repeat

File: Good Job, Good, Goodbye, Goodmorning, Green

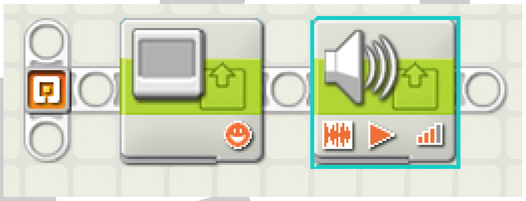
Wait: [✓ Wait for Completion]

4. Select whether you want the NXT to play a sound file or a tone (and choose an appropriate file or tone on the right hand side).

5. Select the Volume and whether or not this sound is to repeat.

6. The 'Wait for Completion' option dictates whether or not the program will wait until the sound has finished playing before moving onto the next part of the program.

7. When you are ready to test your program follow the steps below to download to the NXT. Notice in this example, the program will first display an image then play a sound.



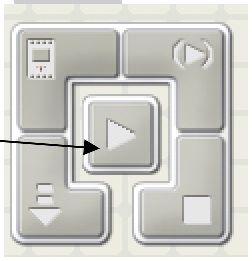
Testing a Program: Downloading to the NXT

1. First, open the rear cover of the NXT to ensure it has 6 working AA batteries.

2. Next, using a USB cable, connect one end to your computer and the other to the NXT.

3. Press the orange Power button to turn on the NXT.

4. On your computer, in the Mindstorms software, click the download and run button.



5. The NXT will execute your program. To run the program again, simply depress the orange button on the front of the NXT.

Connecting Servos and Motors to the NXT

Servos and Motors necessitate the use of specific ports on the NXT. When connecting servos and motors, ensure that they are being connected appropriately (i.e. the Ultrasonic Sensor must be connected to the Ultrasonic Sensor port and not the Touch Sensor port on the NXT).

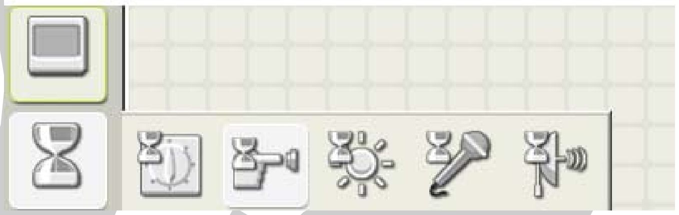
Program B: Working with Servos and Sensors

This program will cause a Servo to spin 103° when a Sensor button is depressed.

1. Using the network cables supplied, connect a servo to Port A and the Touch Sensor to Port 1 on the NXT.

2. Click File -> New to begin a new program.

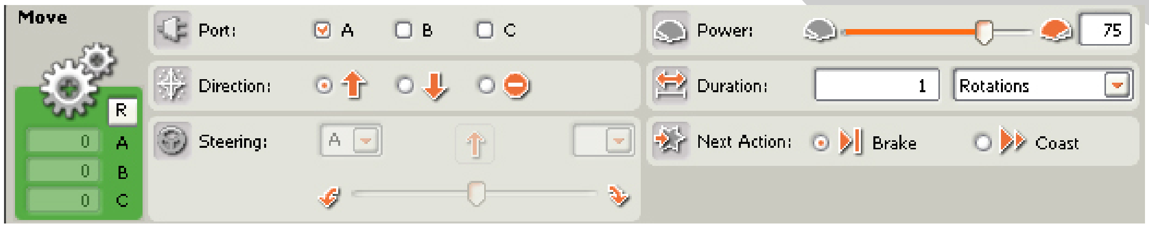
3. On the PC, click the 'Wait' Sensor icon and scroll to the right to select the Touch Sensor icon.



4. Drag the touch Sensor icon to the Start area on the Sequence Beam.

5. In the Touch Sensor dialog box, be sure to specify Port 1 and change the Action of the button to 'Bumped'.

6. Drag the 'Move' icon to the right of the Touch Sensor icon on the sequence beam. Be sure to select Port A only, the up Direction arrow and change the Rotations option to Degrees. Specify 103° and force the Next Action to 'Brake'.



Move

R
0 A
0 B
0 C

Port: [✓ A] B C

Direction: [✓ ↑] ↓ ⊖

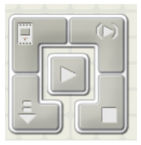
Steering: A

Power: 75

Duration: 1 Rotations

Next Action: [✓ Brake] Coast

7. Click the 'Download and Play' button to test your program.



Intermediate Programming Concepts

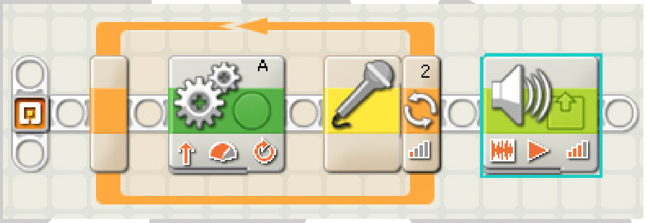
Loops and Conditions are fundamental to programming robots that can adapt to their environment. For example, a robotic car might need to make a 90° turn if it comes close to a wall or a robotic arm might need to distinguish between colored objects. The following programs provide a basic understanding of how Loops and Conditions can be used to effectively order functions in a given program.

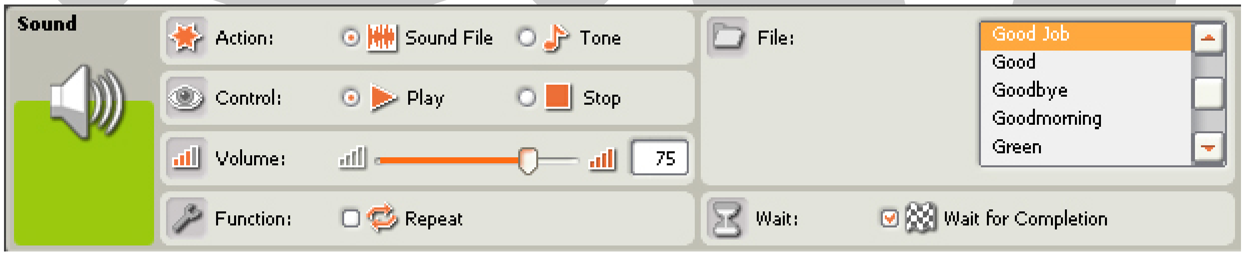
Program C: Working in Loops

This program will cause the NXT to continually turn a Servo until a noise above 30 decibels occurs. When

First, connect the Sound Sensor to Port 2 and a Servo to Port A.

1. Drag a Loop into the Start Position on the Sequence Beam.
2. In the Properties box, change the Control from Forever to Sensor. Next, change the Touch Sensor to Sound Sensor and drag the 'Until' bar till the number reads 30 (decibels).
3. Drag a 'Move' icon into the Loop. In the Properties box, change the Port to A.
4. Drag a sound icon to the Sequence Beam after the Loop. Set the Action to Sound File, Control to Play and set the File to 'Good Job'.

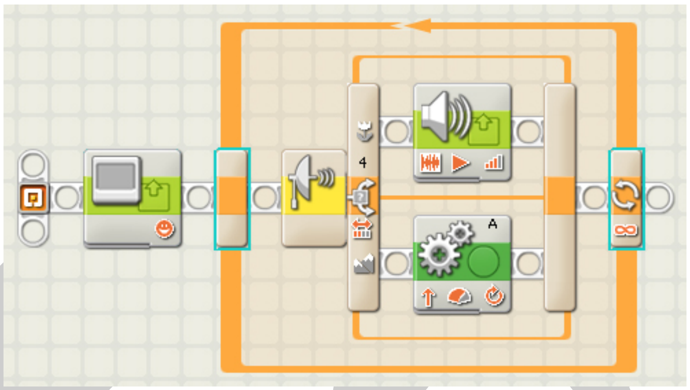



Sound
Action: [✓ Sound File] Tone
Control: [✓ Play] Stop
Volume: 75
Function: Repeat
File: Good Job, Good, Goodbye, Goodmorning, Green
Wait: [✓ Wait for Completion]

5. Click the Download and Play' button to test your program.

Program D: Working with Conditions

This program causes the NXT to Display a message and then, if the NXT is closer than 12” to a solid surface, play a sound. Otherwise turn a servo. This program Loops forever. Before you begin, examine this programming structure.



1. Before programming the NXT, connect a Servo to Port A and the Ultrasonic sensor to Port 4.

2. Drag a Display icon to the Start area on the Sequence Beam and select an image in the Properties box.

3. Drag a Loop icon to the right of the Display icon and leave the Properties set to ‘Forever’.

4. Drag a Switch icon (commonly referred to by Programmers as a Conditional statement) inside the Loop so it appears as above. Change the control to Sensor, the Sensor to Ultrasonic, select Port 4 and set the distance to 12”.

5. In the top Sequence Beam inside the Switch area, drag a Sound icon and select the Sound File ‘Laughing 02’.

6. In the lower area of the Sequence Beam, drag a Motor icon. Change the Port to A and the Duration to 1 Rotations, Next Action ‘Coast’.

7. Download and run the program.

Note: When the Ultrasonic sensor is pointed toward a solid surface more than 12” away, the Servo spins. When the sensor is closer than 12”, the NXT plays ‘Laugh 02’.

Robotics Challenge #1: Thinking Outside the Box

Programmer: ____________________________ Date: _________________

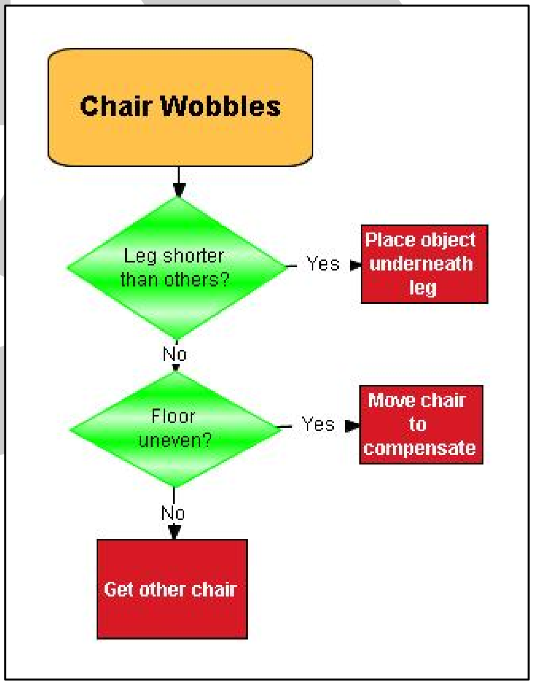
Topic: An introduction to programming

The truth is that robots are only as smart as the people who program them. Despite our best efforts to date, robots cannot think for themselves; they are not cognoscente of their surroundings the way we are and they cannot adapt to their environments unless they are programmed to do so. This means, then, that when programming a robot, the user must anticipate everything the robot will need to complete a task.

To prepare you for the kind of instructions you will need to provide your robot, complete activity 1 to learn more about algorithmic writing.

Activity #1: Algorithmic Writing

According to Wikipedia, "an **algorithm** is a procedure (a finite set of well-defined instructions)[9]. Flow charts are strong examples of algorithms which define very clearly what is to happen in any given scenario.



Consider for a moment the chair you're sitting in. Your mind actually made rather complex logic decisions when you made the decision to try and sit down. Before sitting down, presumably, your eyes scanned the chair to examine its structural strength and stability. If you found that one leg was short, you would likely have lowered yourself to examine the situation more intensely. If you found that there was something on top of the seat, you would likely have examined it and, perhaps, wiped it aside. If all went well, and your senses okayed all aspects of the chair, odds are you sat down.

Activity:
Using SmartIdeas (or a sheet of blank paper), construct a flowchart to display the kinds of decisions you would need to make if you were to shoot a basketball at a hoop.

Robotics Challenge:

Design a robot capable of sensing its proximity to a floor. It should snore constantly if it is closer than 2 feet.

[9]*Wikipedia*. October 13 2006 <http://en.wikipedia.org/wiki/Algorithm>

Robotics Challenge #2: Thinking Outside the Box 2

Programmer: ___________________________ Date: ________________

Topic: An introduction to programming

Often, programmers will design a robot only to realize later that with slight modifications, this robot can respond in very different ways. Typically this additional programming involves only minor tweaks, but even slight changes can lead to very different outcomes.

The activity below is designed to give you the opportunity to modify the robot you constructed in the first Activity. How will it respond differently? What kind of responses could this robot have with only slight changes to its structure or programming? Think outside the box.

Activity #2: Algorithmic Writing

Brainstorming is often a great way of generating interesting and innovative ideas. In fact, programmers will often work together to solve complex problems using ideas generated from each others' musings.

Activity:
In small groups of 4 persons, based on the robot you created in the previous Activity, generate a list of 10 possible modifications for your robot that, in your mind, can be considered minor (30 minute modifications).

Finally, place an asterisk next to those modifications you feel are most useful and most easily implemented to your current robotic framework and program.

Robotics Challenge:

Design a robot capable of sensing its proximity to a floor. It should snore constantly if it is closer than 2 feet.

Robotics Challenge #3: Working in Conditions

Programmer: ____________________________ Date: _________________

Topic: Working in Conditions

The most common kind programming statement is the "If" statement, frequently referred to as a conditional statement. An "If" statement requires the robot to determine whether or not *something* is going on, and if it is, "Then" do *this*.

For example, let's say you wanted to build a robot that spoke when spoken to. In other words, your robot program could be summarized this way:

IF you hear something, **THEN** speak.

Activity #3: Writing Conditions

Often, Conditional Statements are invoked as a series of decision-making steps. For example, let's say we want the robot to turn around if someone claps or if the lights turn on. A program like this can be summarized this way:

```
{
Go Forward
        If you hear a clap
                Then ->         Turn Around
        ElseIf you sense light
                Then ->         Turn Around
        Else
Keep Going Forward
}
```

The curly brackets indicate the beginning and end of the program. The ElseIf statement follows-up the introductory "If" statement and serves the same purpose. The Else statement indicates the end of the series of "If" statements. Notice that for each "If" or "ElseIf" there is a follow-up "Then" statement to clearly identify the subsequent action that is to take place.

Activity:
Using the above framework as a model, write a program that causes a robotic vehicle to stop when it encounters an obstacle nearer than 2 feet or when it sees something Red in its path.

Robotics Challenge:
Create a robot that 'wakes up' when the lights are turned on and drives across the floor.

Robotics Challenge #4: Working in Loops

Programmer: ______________________ Date: ____________

Topic: Working in Loops

A loop requires a robot to continually perform a certain task until a condition is met. Loops are commonly found in instances where programs execute a loop and then move on to a new series of instructions. These loops can run several times before they are satisfied.

For example, let's say you want an oven element to heat up until it reaches 350°. In other words, you want the oven to sense its temperature while it is heating and stop when it reaches 350°. This kind of loop is often referred to as a Condition-Controlled Loop.

Activity #3: Writing Conditions

There are many instances where we want robots to repeatedly execute a series of instructions. For example, police use radar guns to constantly measure the speed on oncoming traffic. The radar gun will constantly measure the velocity of oncoming objects until a button is pushed causing it to stop. Similarly, a thermostat constantly measures the temperature of your home. When the temperature reaches a certain low, the furnace turns on and stays on until it reaches a set temperature at which point it shuts down again. Like the radar gun, this program runs endlessly while there is electricity running through its circuits.

Activity:
Brainstorm a list of 5 different machines that execute loops on a daily basis.

Robotics Challenge:
Create a robot on wheels and place it in an enclosed arena (i.e. a rectangular area on the floor walled with textbooks). Be sure that there is an opening at one end of the arena and program the robot to continue to search for the exit until it finds its way out.

Robotics Challenges: Thinking beyond the Program

Programmer: ______________________________ Date: __________________

Topic: Thinking Outside the Box

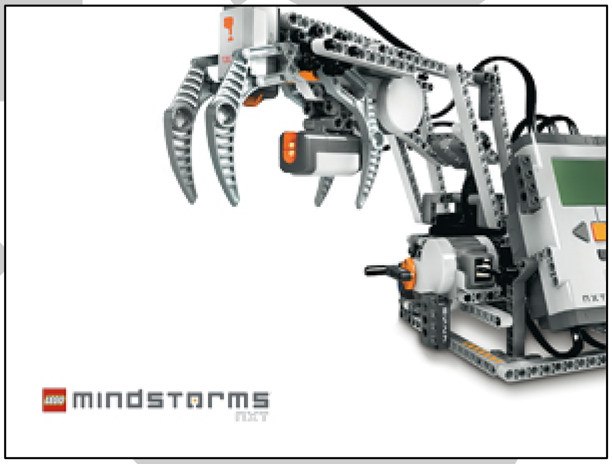
When a good, creative program developer sits down to write a program, he/she doesn't think about the confines of the machinery he/she has to work with. Instead, a programmer thinks beyond the boundaries of the machinery to what a robot might do given limitless possibilities.

What kind of robot do you want to make? Do you want to create a robot that wakes you up in the morning? Or, a robot that can tie your shoes? What about a robot who can make you breakfast or shoot a basketball?

This list is meant by no means to be an exhaustive repository of ideas but instead a place of inspiration that might spur ideas of your own. Some of the ideas below may require extra materials such as an elastic band or perhaps a marker. Remember, there's nothing wrong with thinking outside the box!

What will your robot do?

Many of the following ideas were inherited from users on the Lego Mindstorms Forum.



1. Can your robot scale a wall 1ft high?
2. Can you make an elevator?
3. Can your robot find its way around a maze?
4. Can you make a crane?
5. Can your robot go up and down stairs?
6. Can you make a robot pet?
7. Can you make a robot that can sense and cross large gaps?
8. Can you make a catapult?
9. Can you make an automatic toothpaste squeezer?
10. Can you make a robot that can sweep a room clean?

Resources
There are many terrific ideas posted on the Internet. Follow these links to grow your brain.

http://www.firstlegoleague.org/

http://mindstorms.lego.com/buildinginstructions/

http://mindstorms.lego.com/NXTLOG/default.aspx