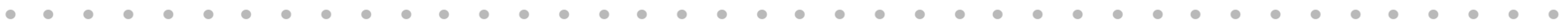


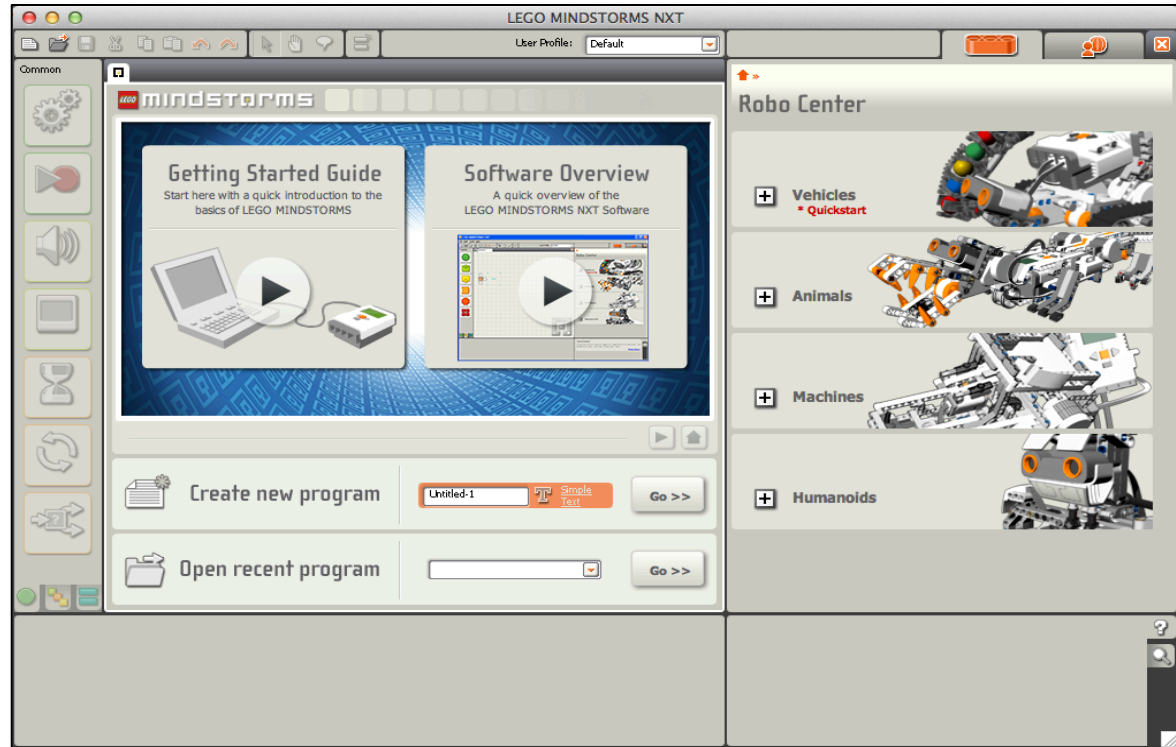
## Programming & Control Robots with DSLs

Eclipse DemoCamp – Zurich / 06-2013

Serano Colameo



# Visual Programming with Lego Mindstorms NXT



# A real Programming Language API would be nice...



lego mindstorms java

Web Images Maps Shopping Videos More Search tools

About 1,030,000 results (0.27 seconds)

[LeJOS, Java for Lego Mindstorms](#)  
[lejos.sourceforge.net/](http://lejos.sourceforge.net/)

Java based operating system for Lego Mindstorms RCX. Includes FAQs, downloads, robot files and utilities.

[Downloads](#) - [The leJOS NXJ Tutorial](#) - [leJOS NXJ API documentation](#) - [leJOS NXJ](#)

# LeJOS - Java for Lego Mindstorms Roboter



```
package ch.itemis.examples.xtend;
import lejos.nxt.Motor;

public class NXTJava {

    public static void main(String [] args) throws Exception {
        NXTConnector conn = new NXTConnector();

        conn.addLogListener(new NXTCommLogListener() {
            public void logEvent(String message) {
                System.out.println(message);
            }

            public void logEvent(Throwable throwable) {
                System.err.println(throwable.getMessage());
            }
        });

        conn.setDebug(true);

        if (!conn.connectTo("btspp://NXT", NXTComm.LCP)) {
            System.err.println("Failed to connect");
            System.exit(1);
        }

        NXTCommandConnector.setNXTCommand(new NXTCommand(conn.getNXTComm()));

        System.out.println("Tachometer A: " + Motor.A.getTachoCount());
        System.out.println("Tachometer C: " + Motor.C.getTachoCount());

        Motor.A.rotate(5000);
        Motor.C.rotate(-5000);

        Thread.sleep(10000);

        Sound.playTone(1000, 1000);

        System.out.println("Tachometer A: " + Motor.A.getTachoCount());
        System.out.println("Tachometer C: " + Motor.C.getTachoCount());

        conn.close();
    }
}
```



# Build with Xtend an internal DSL for LeJOS



```
package ch.itemis.examples.xtend;
import lejos.nxt.Motor;

public class NXTJava {

    public static void main(String [] args) throws Exception {
        NXTConnector conn = new NXTConnector();

        conn.addListener(new NXTCommListener() {
            public void logEvent(String message) {
                System.out.println(message);
            }
        });

        public void logEvent(Throwable throwable) {
            System.err.println(throwable.getMessage());
        }
    });

    conn.setDebug(true);

    if (!conn.connectTo("btspp://NXT", NXTComm.LCP)) {
        System.err.println("Failed to connect");
        System.exit(1);
    }

    NXTCommandConnector.setNXTCommand(new NXTCommand(conn.getNXTComm()));

    System.out.println("Tachometer A: " + Motor.A.getTachoCount());
    System.out.println("Tachometer C: " + Motor.C.getTachoCount());

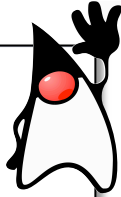
    Motor.A.rotate(5000);
    Motor.C.rotate(-5000);

    Thread.sleep(10000);

    Sound.playTone(1000, 1000);

    System.out.println("Tachometer A: " + Motor.A.getTachoCount());
    System.out.println("Tachometer C: " + Motor.C.getTachoCount());

    conn.close();
}
}
```



```
package ch.itemis.examples.xtend

import lejos.nxt.Motor

class NXTend1 {

    def static void main(String[] args) {
        val NXTConnector conn = new NXTConnector()
        if (!conn.connectTo("btspp://NXT", NXTComm.LCP)) {
            System.out.println("Failed to connect")
            System.exit(1)
        }
        NXTCommandConnector.setNXTCommand(new NXTCommand(conn.getNXTComm()))
        System.out.println("Tachometer A: " + Motor.A.getTachoCount())
        System.out.println("Tachometer C: " + Motor.C.getTachoCount())
        Motor.A.rotate(5000);
        Motor.C.rotate(-5000);
        Thread.sleep(10000);
        Sound.playTone(1000, 1000);
        System.out.println("Tachometer A: " + Motor.A.getTachoCount())
        System.out.println("Tachometer C: " + Motor.C.getTachoCount());
        conn.close();
    }
}
```

Xtend

```
package ch.itemis.examples.xtend

import static extension ch.itemis.examples.xtend.NXTendBuilder.*

class NXTend3 {

    def static void main(String[] args) {
        NXT.connect
        sayHello
        MotorA.showTachoCounter
        MotorC.showTachoCounter
        MotorA.forward = 500
        MotorB.forward = 500
        MotorC.backward = 500
        Thread.sleep(10000);
        Sound.playTone(1000, 1000);
        System.out.println("Tachometer A: " + Motor.A.getTachoCount())
        System.out.println("Tachometer C: " + Motor.C.getTachoCount());
        conn.close();
    }
}
```

```
package ch.itemis.examples.xtend

import static extension ch.itemis.examples.xtend.NXTendBuilder.*

class NXTend3 {

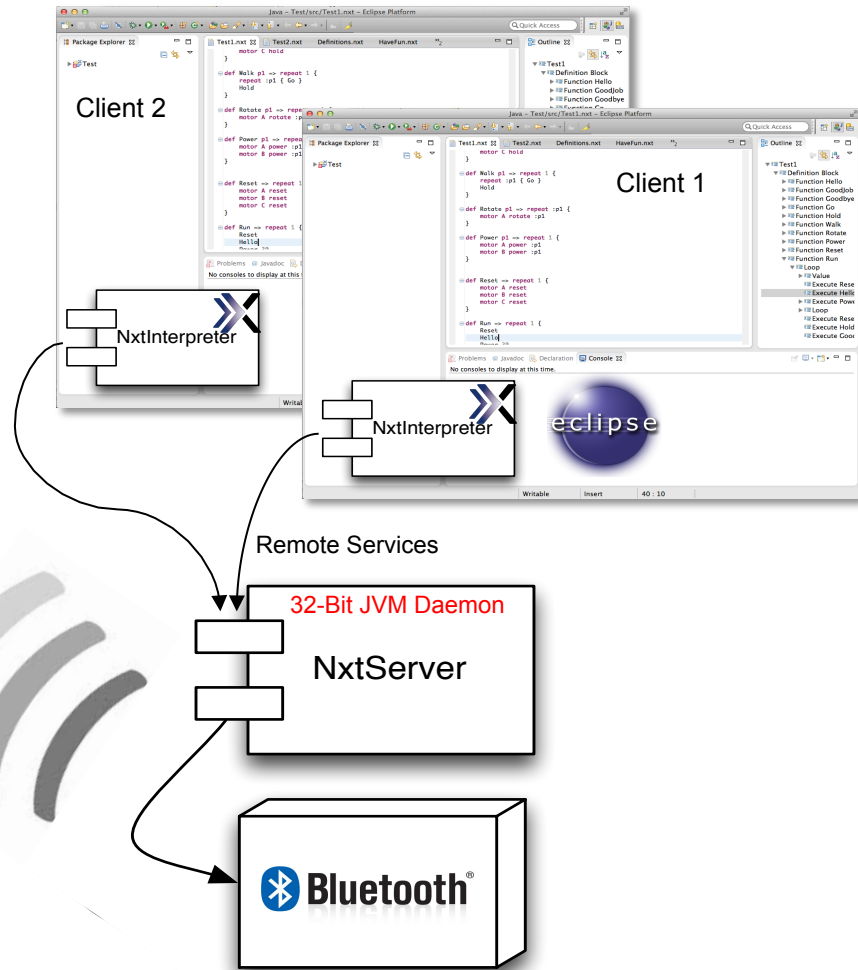
    def static void main(String[] args) {
        NXT => [
            val motors = #{MotorA, MotorC}
            connect
            sayHello
            motors.forEach[ showTachoCounter ]
            motors.forEach[ forward = 500 ]
            playTone = 1000 -> 2000
            motors.forEach[ showTachoCounter ]
            disconnect
        ]
    }
}
```

“Exactly the same behaviour but with less language noise!”

# Why not develop a DSL for the Lego Roboter?



# Client / Server Architecture



# Language Concept



## „Functions & Function Blocks like Lego Bricks“



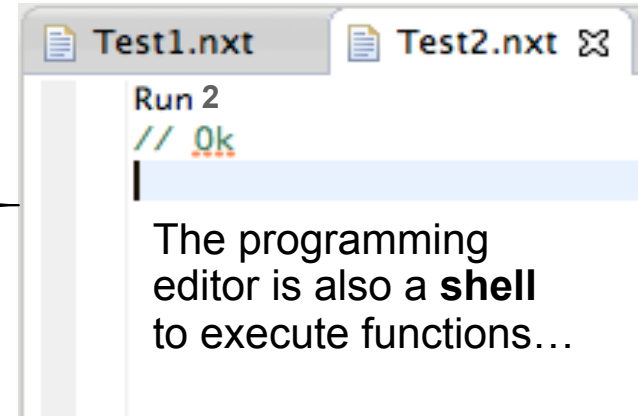
- Single Statement Functions

```
def Hello => playSound 'Hello.rso'  
def Go => motor A forward  
def Connect => device NXT
```

- Function Blocks

```
def Go => block {  
  motor A forward  
  motor B forward  
  motor C forward }  
  
def Reset => block {  
  motor A reset  
  motor B reset  
  motor C reset }  
  
def Hold => block {  
  motor A hold  
  motor B hold  
  motor C hold }  
  
def Walk p1 => repeat 1 {  
  repeat :p1 { Go }  
  Hold  
}
```

```
def Run p1 => repeat :p1 {  
  Connect  
  Hello  
  repeat 10 {  
    Reset  
    Walk 1  
  }  
  Hold  
  Goodbye  
}
```





# How to write an Interpreter



“Navigate through the AST tree and dispatch/evaluate elements (nodes)”

The image shows a screenshot of an IDE with two panels. The left panel displays the source code for `NxtDslInterpreter.xtend`. The right panel shows the AST tree for the file `platform:/resource/ch.itemis.nxt/src-gen/ch/itemis/nxt/NxtDsl.ecore`.

```

NxtDslInterpreter.xtend
/**
 * Copyright (c) by itemis Schweiz GmbH
 *
 * @author Serano Colameo
 */
package ch.itemis.nxt.ui.interpreter

import ch.itemis.nxt.nxtDsl.Command

class NxtDslInterpreter extends NxtClient {

  def interpret(Resource resource) {
    resource?.main?.run
  }

  def run(Main it) {
    switch (block) {
      ExecutionBlock : block.evaluate
      default : println('Nothing to do!')
    }
  }

  def dispatch void evaluate(ExecutionBlock it) {
    commands.evaluate
  }

  def dispatch void evaluate(Iterable<CommandExecution> commands) {
    commands.forEach(clic.evaluate)
  }

  def dispatch void evaluate(CommandExecution it) {
    function?.command?.evaluate(values)
  }

  def dispatch void evaluate(CommandExecution it, List<Value> values) {
    evaluate
  }

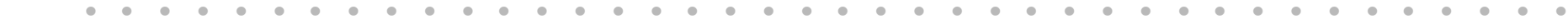
  def dispatch void evaluate(PlayToneCommand it, List<Value> values) {
    setValues(duration, values)
    setValues(freq, values)
    playTone(freq.value, duration.value)
  }
}

```

The AST tree on the right shows the following structure:

- platform:/resource/ch.itemis.nxt/src-gen/ch/itemis/nxt/NxtDsl.ecore
  - nxtDsl
    - Main
    - ExecutionBlock
    - DefinitionBlock
    - Command
    - CommandExecution -> Command
      - function : CommandDefinition
      - values : Value
    - CommandDefinition
    - Parameter
    - RepeatCommand -> Command
    - MotorCommand -> Command
    - Motor
      - port : Port
      - ref : Parameter
    - Port
    - MotorDirective
    - MotionDirective
    - SimpleDirective
    - DeviceCommand -> Command
    - SoundCommand -> DeviceCommand
    - SayHelloCommand -> SoundCommand
    - SayGoodbyeCommand -> SoundCommand
    - SayGoodJobCommand -> SoundCommand
    - DeviceNameCommand -> DeviceCommand
    - PlayToneCommand -> SoundCommand
    - ValueHolder
      - value : Value
      - ref : Parameter
    - Value
      - num : ElInt

# Demo – „Roboter in Action“





Serano Colameo

Phone: +41 (0)56 442 68 63

Email: [serano.colameo@itemis.ch](mailto:serano.colameo@itemis.ch)

itemis Schweiz GmbH | Dorfstrasse 69 | CH-5210 Windisch | [www.itemis-schweiz.ch](http://www.itemis-schweiz.ch)