Controlarea vehiculelor cu roti utilizând kit-ul Lego Mindstorm NXT

Controlul Motoarelor

Introducere în clasa Motor

Această clasă este o idee abstractă a unui motor NXT. Pentru a fi util, un motor trebuie să fie conectat la unul din cele trei porturi pentru motoare ale NXT. Această clasă determina o instanţă pentru fiecare port. Acestea sunt: Motor.A, Motor.B şi Motor.C. Clasa ofera metode de control al motorului, precum şi pentru a afla ce se face un motor.

Va fi prezentat un set de cinci programme pentru a înţelege modul în care funcţionează un motor NXT.

Programul 1 – Miscari de baza

Acest program se foloseşte de metode de bază de control al miscarilor.

Metodele utilizate în acest program

Clasa	Numele metodei	Note
`Motor`	`forward()`	Porneste miscare de rotatie inainte
	`backward()`	Porneste miscare de rotatie inapoi
	`changeDirection()`	Inverseaza sensul de rotatie
	`stop()`	Oprire
`Button`	`waitForPress()`	Asteapta apasarea unui buton
`LCD`	`drawString(String`	Deseneaza un sir

	`str, int x, int y)`	

Motorul avanseaza si revine pana cand butonul este apasat cu afisarea pe ecran a mesajelor „Forward” si „Backword”.

```
        Motor.A.forward();
        LCD.drawString("FORWARD", 0, 0);
        Button.waitForPress();
        Motor.A.backward();
        LCD.drawString("BACKWARD", 0, 1);
        Button.waitForPress();
        Motor.A.reverseDirection();
        LCD.drawString("FORWARD", 0, 2);
        Button.waitForPress();
        Motor.A.stop();
    }
}
```

Programul 2 - Utilizarea Tahometrului.

Motorul NXT are un tahometrul inclus care monitorizează unghiul curent (în grade) al axului motorului. Scopul acestui program este de a utiliza tahometrul pentru a afla cât de repede se opreşte motorul.

Metode noi folosite in program

Clasa	Numele metodei	Note
`Motor`	`flt()`	Prescurtare pentru flost. Opreste alimentarea, dar nu aplica franarea

	`getTachoCount()`	Intoarce valoarea unghiului motorului in grade
	`resetTachocount()`	Initializeaza contorul cu 0
	`setSpeed(int speed)`	Viteza- grade pe secune .. Viteza maximă care poate fi menţinuta cu precizie este de aproximativ 110 de ori tensiunea bateriei.
	`int getActualSpeed()`	Intoarce viteza motorului (deg/sec)
`LCD`	`drawInt()`	

Ce ar trebui să facă programul

1. Roteste Motor.A înainte.
2. Aşteptaţi până la contorul tahometrului ajunge la 360.
3. Opreste motorul.
4. Aşteptaţi până când motorul sa oprit.
5. Afisaza valoarea citita la tahometru pe ecran.
6. Aşteptaţi pana un buton este apasat pentru a vă oferi timp pentru a citi pe ecran.
7. Setaţi viteza motorului la 720 (implicit este 360).
8. Reseteza contorul tahometrului la zero.
9. Repetaţi paşii de la 1 la 6.

Repetati întregul experiment prin modificarea codului prin utilizarea FLT () în loc de stop (); (FLT este abrevierea pentru float, cuvant rezervat Java). Această metodă seteaza puterea motorului la zero, dar nu se aplică frana.

Observaţi că, chiar utilizand frâna, motorul nu se opreste imediat, pentru ca are motor de inerţie.

```
LCD.drawString(" Test", 0, 0);  // a.i programul sa nu inceapa imediat
```

```
        Button.waitForPress();
        rotate720();
        Motor.A.resetTachoCount();
        Motor.A.setSpeed(800);
        rotate720();
```

- se roteste cu to 720, se opreste, afiseaza numaratorul tacho, asteapta;

```
  public static void rotate720()
  {
        Motor.A.forward();
        int count = 0;
        while( count < 720 )count = Motor.A.getTachoCount();
        Motor.A.stop();
//       Motor.A.flt();
        LCD.drawInt(count , 0, 1);
        while(Motor.A.getActualSpeed()>0);
        LCD.drawInt(Motor.A.getTachoCount(), 7,1 );
        Button.waitForPress();
        LCD.clear();
  }
}
```

Programul 3 – Controlul precis al rotatiei

Clasa Motor are o operatie de reglementare care ruleaza tot timpul. Acesta are sarcini principale, dintre care una este de a opri cu motor de la un anumit unghi. Acest program va testa acurateţea metodei rotate().

Noi metode folosite în acest program

Clase	Numele metodei	Note
`Motor`	`rotate(angle)`	Roteşte prin anumite unghiuri de anumite grade
	`rotateTo(angle)`	Roteste la un anumit unghi

	`rotateTo(angle,true)`	Seteaza contorul la zero

Ce ar trebui să facă programul

1. Rotiţi motorul o rotatie completa.
2. Afisaza citirea de la tahometru pe LCD, rândul 0.
3. Rotiţi motorul la unghi 0.
4. Afisaza citirea de la tahometru pe LCD, rândul 1.
5. Aşteptaţi până când un buton este apăsat pentru a vă oferi timp pentru a citi LCD.
6. Stergeti LCD.
7. Setaţi viteza la 720.
8. Repetaţi paşii de la 1 la 6.

Notă: puteţi evita repetarea in cod prin scrierea unei metode care execute aceşti paşi.

Observati ca motorul de obicei se opreşte la 1 grad de unghiul specificat în cazul în care regulatorul isi face treaba. Acesta acţionează prin calcularea cât de departe va continua motorul sa se roteasca după ce a fost aplicată de frână. Aceasta se aplică de frână înainte de a ajunge la unghiul specificat. Face apoi ajustări minore la poziţia motorului pana este destul de aproape.

```
    LCD.drawString("Rotation", 0, 0);//afiseaza numele programului inainte sa ruleze
    Button.waitForPress();
    LCD.clear();
    RotationTest robot = new RotationTest();
    robot.go();// creaza o instanta a clasei si cheama “go”

}
```

- aceasta medota pune in miscare robotul

```
  public void go()
  {
     rotate720();
     Motor.A.setSpeed(800);
     rotate720();
  }
```

- opreste motorul cand butonul este apasat

```
  public void rotate720()
  {
     Motor.A.rotate(720);
     LCD.drawInt(Motor.A.getTachoCount(), 4, 0, 1);
     Motor.A.rotateTo(0);
     LCD.drawInt(Motor.A.getTachoCount(), 4, 0, 2);
     Button.waitForPress();
  }
}
```

Programul de 4.Intreruperea rotaţiei

Uneori veţi dori sa opriti motorul (sau sa faca altceva) înainte de a ajunge la unghiul specificat. Acest program va detecta cand un buton este apasat pentru a întrerupe sarcina de rotaţie, dacă apăsaţi pe buton, destul de curand. Metoda rotate()nu va returna până cand motorul se opreste la unghiul ţintă. Dar noile metode în acest program pot returna imediat. Motorul se va opri la unghiul specificat cu excepţia cazului în care o noua metodă motor este chemata între timp.

Noi metode folosite în acest program

Clase	**Numele metodelor**	**Note**

`Motor`	`rotate(angle,immediateReturn)`	Metoda returneaza imediat în cazul în care parametru boolean immediateReturn este **true**
	`rotateTo(angle,immediateReturn)`	
	`(boolean)isRotating()`	Returnează **false** atunci când motorul sa oprit la unghi specificat
`Button`	`int readButtons()`	Returnează ID-ul butonului apăsat.

Ce ar trebui să facă programul

1. Începeţi o rotatie de 720 grade.
2. În timp ce motorul se roteste, afişaza valoarea tahometurlui la poziţia rândului 1.
3. Atunci când este apăsat un buton, opreşte motorul.
4. Dupa ce motorul s-a oprit, afişaza valoarea tahometurlui în centrul rândului.
5. Aşteptaţi un buton sa fie apasat.
6. Începeţi o rotaţie la 0.
7. Repetaţi paşii de 2,3,5.

Observaţi: dacă apasati butonul înainte ca rotaţia sa fie completa, motorul se va opri fără sa execute o rotaţie completa. În caz contrar, metoda stop() nu are nici un efect.

- permite intreruperile intr-un ciclu de rotatie

public class RotInterrupt

```
{
   public void go()
   {
      System.out.println("\n rotatie");
      Sound.twoBeeps();
      Button.waitForPress();
      Motor.A.rotate(720,true);
      showRotation(1);
      Motor.A.rotateTo(0,true);
      showRotation(1);
   }
    public void showRotation(int row )
   {
      while(Motor.A.isRotating())
      {
         LCD.drawInt(Motor.A.getTachoCount(),4, 0, row);
         if(Button.readButtons()>0) Motor.A.stop();
      }
      while(Motor.A.getActualSpeed()>0);
      LCD.drawInt(Motor.A.getTachoCount(), 4,8,row);
      Button.waitForPress();
   }
   public static void main(String[] args)
   {
      new RotInterrupt().go();
   }

}
```

Hardware: I / O si Senzori

Această secţiune se referă la clasele pentru intrare şi ieşire, şi, de asemenea la senzorii standard NXT. Hardware-ul NXT are butoane de intrare, un afisaj cristale lichide (LCD) şi un mic difuzor pentru ieşire. leJOS NXJ dispune de reprezentari software pentru toti acesti biti de hardware.

LCD

Clasa LCD nu are nici o instanţa (acolo fiind doar un LCD pe NXT), astfel încât toate metodele sunt statice. Acesta poate fi utilizat în modul text si modul grafic.

Metodele de a scrie la LCD în mod text sunt: --

• nule drawString (String str, int x, int y)

Aceasta afisaza un text pe ecranul LCD începand de la coordona (x, y).

• nule drawInt (int i, int x, int y)

Aceasta afisaza un întreg incepand de la coordona (x, y). Valoarea intreaga este aliniata la stânga şi afisaza cât mai multe caractere care sunt necesare.

• nule drawInt (int i, int places, int x, int y)

Această variantă de drawInt aliniază la dreapta o vloare intreaga şi întotdeauna foloseşte numărul de caractere indicat de locuri. Acest lucru înseamnă că întotdeauna să scrie un număr fix de locuri de caractere şi, dacă este utilizat într-o buclă, valoarea anteriora va fi întotdeauna suprascrisa.

• nule clear ()

Goleşte ecran.

```
public class LCDTest {
public static void main(String[] args) throws Exception {
LCD.drawString("Free RAM:", 0, 0);
LCD.drawInt((int) System.getRuntime().freeMemory(), 6, 9, 0);
Thread.sleep(2000);
```

Reţineţi, de asemenea, că, în mod implicit, ecranul LCD este actualizat automat. Dacă doriţi să utilizati în cazul în care LCD-ul este reîmprospãtat puteţi apela LCD.setAutoRefresh (0) pentru a opri auto-actualizarea şi apelarea LCD-

ului. refresh(), atunci când doriţi o reîmprospătare a ecranului.

Butoane

Clasa Button are patru instante, accesate de campuri statice:

- Button.ENTER
- Button.ESCAPE
- Button.LEFT
- Button.RIGHT

Pentru a verifica daca un buton este apasat, folosesti:

- boolean isPressed()

Exemplu:

```
 public class ButtonPresses {
 public static void main(String[] args) throws Exception {
while (true) {
LCD.clear();
if (Button.ENTER.isPressed()) LCD.drawString("ENTER", 0, 0);
if (Button.ESCAPE.isPressed()) LCD.drawString("ESCAPE", 0, 0);
if (Button.LEFT.isPressed()) LCD.drawString("LEFT", 0, 0);
if (Button.RIGHT.isPressed()) LCD.drawString("RIGHT", 0, 0);
```

Bateria

Sunt doua metode statice pentru aflarea voltajului bateriei:

- int getVoltageMilliVolt()
- float getVoltage()

Examplu:

```
public class BatteryTest {
public static void main(String[] args) throws Exception {
LCD.drawString("Battery: " + Battery.getVoltage(), 0, 0);
Thread.sleep(2000);
```

Senzori

NXT vine cu patru senzori; senzori tactili, senzor de sunet, lumina si senzor ultrasonic . leJOS NXJ furnizeaza reprezentarile software pentru toate aceste tipuri de senzor, precum şi multe altele furnizate de terţi.

Un senzor trebuie să fie conectat la un port, şi senzorul trebuie să ştie care este acest port. Pentru a oferi aceste informaţie, veţi crea o instanţă a senzorului, şi veti furniza această informaţie constructorului. Posibilităţile sunt: SensorPort.S1, S2, S3 sau S4.

Sensor tactil

Pentru a utiliza senzorul tactil, veti crea o instanţă a acestuia, folosind constructorul:

- TouchSensor(SensorPort port)

Pentru a testa dacă senzorul tactil este apasat, este folosita metoda isPressed():

- boolean isPressed()

Exemplu:

```
public class TouchTest {
public static void main(String[] args) throws Exception {
TouchSensor touch = new TouchSensor(SensorPort.S1);
while (!touch.isPressed() ;
LCD.drawString("Finished", 3, 4);
```

Senzori de sunet

Senzorul de sunet suportă două moduri: DB şi DBA. Aceste moduri ofera frecvenţe diferite, astfel încât pentru a obţine o idee despre frecvenţa unui sunet sa comutati între moduri.

Există doi constructori:

- SoundSensor(SensorPort port)

Creaza un sensor de sunet folosind modul DB.

- SoundSensor(SensorPort port, dba)

Creaza un sensor de sunet folosind modul DBA daca al doilea parametru este true.

Puteti inversa modurile folosind:

- void setDBA(boolean dba)

Exemplu folosind modul DB:

Exemplul de mai sus oferă o afişare grafică a modului în care variază sunetul pe o perioadă de două secunde.

```
public class SoundScope {
public static void main(String[] args) throws Exception {
SoundSensor sound = new SoundSensor(SensorPort.S1);
 while (!Button.ESCAPE.isPressed()) {
 LCD.clear();
 for (int i = 0; i < 100; i++) {
 LCD.setPixel(1, i, 60 - (sound.readValue() / 2));
 Thread.sleep(20);
```

Senzori cu ultrasunete

Pentru a creea un exemplu, utilizati constructorul:

- UltrasonicSensor (Port aSensorPort)

Senzorul funcţionează în două moduri, continuu (defalt) şi ping. Când în mod continuu senzorul trimite ping cât de des se poate şi cel mai recent rezultat obţinut este disponibil prin intermediul apelarii

• int getDistance ()

Rezultatul este valoarea în centimetri. În cazul în care nu a fost detectat echo, valoarea returnata este 255. Nivelul maxim al senzorului este de aproximativ 170 cm.

Exemplu:

```
public class SonicTest {
public static void main(String[] args) throws Exception {
UltrasonicSensor sonic = new UltrasonicSensor(SensorPort.S1);
while (!Button.ESCAPE.isPressed()) {
LCD.clear();
LCD.drawString(sonic.getVersion(), 0, 0);
LCD.drawString(sonic.getProductID(), 0, 1);
LCD.drawString(sonic.getSensorType(), 0, 2);
LCD.drawInt(sonic.getDistance(), 0, 3);
```

Când în modul ping, un ping este trimis doar atunci când este făcut un apel

• void ping()

Aceasta stabileşte senzorul în modul ping şi trimite un singur ping şi până la 8 echoes sunt pimite. Acestea pot fi citite printr-un apel

• int readDistances(int [] distances)

Oferiti un sir de lungimi intregi care conţine date după ce metoda returneaza. O întârziere de aproximativ 20ms este necesară între ping şi getDistances. Această întârziere nu este inclusa în metodă. Apelurile la getDistances înainte de acest termen poate duce la o eroare sau la nici un

rezultat. Apelarea normala getDistance poate fi de asemenea folosita cu ping, returnand informaţii, pentru primul echo.

Apelarea ping () va dezactiva modul implicit continuu. Pentru a trece înapoi la modul continuu, apelati

• int continuous()

In urma realizarii pasilor de instalare a kit-urilor necesare comunicarii robotului Lego Mindstorms NXT cu PC-ul putem vedea in continuare interfata unei aplicatii ce consta in pornirea motorului, avansarea acestui robot atata timp cat doreste utilizatorul, oprirea lui sau dupa caz revenirea la pozitia initiala.

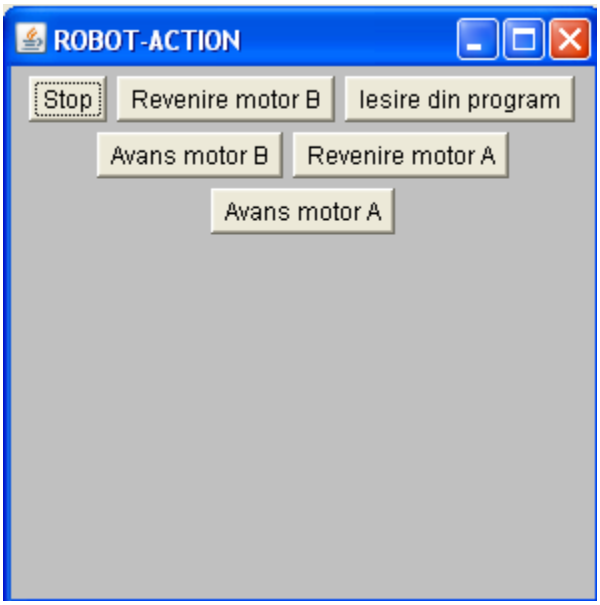

Fig 4.1 Interfata grafica ce permite manipularea robotului LEGO NXT, ce permite avansul si revenirea robotului pentru motorul A si motorul B.

Pentru a comanda robotul LEGO NXT folosim urmatorul cod:

```
import lejos.nxt.*;
import lejos.nxt.comm.*;

import lejos.nxt.LCD;
import lejos.nxt.remote.*;
import lejos.nxt.addon.*;
import java.awt.event.*;
import java.awt.*;
import java.awt.Button;

public class Communication extends  Frame implements ActionListener{
	/**
	 *
	 */
	private static final long serialVersionUID = 1L;
	private	Panel	topPanel;
	Button stop,out;
	Button forward,forwardB;
	Button backward,backwardB;
	final int FORWARD_A = 1;
	final int BACKWARD_A = 2;
	final int FORWARD_B = 5;
	final int BACKWARD_B = 6;
	final int STOP = 3;
	final int OUT = 4;

	public Communication()
	{

		setTitle( "ROBOT-ACTION" );
		setSize( 300, 300);
		setBackground(Color.lightGray);

		// panel
		topPanel = new Panel();//se creeaza un obiect de tip JPanel
		topPanel.setLayout( new FlowLayout() );//setam layout-ul de
tip flowLayout

		add(topPanel);


		forward = new Button("Avans motor A");
		topPanel.add(forward,FlowLayout.LEFT);
		forward.addActionListener(this);

		backward = new Button( "Revenire motor A" );
		topPanel.add(backward,FlowLayout.LEFT);
		backward.addActionListener(this);

		forwardB = new Button("Avans motor B");
```

```
            topPanel.add(forwardB,FlowLayout.LEFT);
            forwardB.addActionListener(this);

            backwardB = new Button( "Revenire motor B" );
            topPanel.add(backwardB,FlowLayout.LEFT);
            backwardB.addActionListener(this);

            stop = new Button("Stop");
            topPanel.add(stop,FlowLayout.LEFT);
            stop.addActionListener(this);

            out = new Button( "Iesire din program" );
            topPanel.add(out,FlowLayout.RIGHT);
            out.addActionListener(this);
      }


      public void actionPerformed(ActionEvent e){
      Object event = e.getSource();//(e.getActionCommand());



      int nr = 0 ;
      if (event == forward ) { nr = 1;}
      if (event == backward ) { nr = 2;}
      if (event == forwardB ) { nr = 5;}
      if (event == backwardB ) { nr = 6;}
      if (event == stop ) { nr = 3;}
      if (event == out ) { nr = 4;}

      switch(nr) {
    case FORWARD_A:
      setMotorActionForward();
       break;
    case BACKWARD_A:
      setMotorActionBackward();
       break;
    case STOP:
       stopMotor();
       break;
    case BACKWARD_B:
      setMotorActionBackward_B();
      break;
    case FORWARD_B:
      setMotorActionForward_B();
       break;
    case OUT:
      OutProgram();

       break;
}

      /*    if(e.getSource() == forward){

                  setMotorActionForward();
```

```
            }
            else
                  if (e.getSource() == stop)
                  {
                        stopMotor();
                  }
                  else  if (e.getSource() == backward)
                  {
                        setMotorActionBackward();
                  }*/
      }
  public static void setMotorActionBackward()
  {

  Motor.A.backward();//robotul se intoarce
  //LCD.drawString("BACKWARD", 0, 0); // se afiseaza pe display-ul LEGO
in coltul din dreapta sus mesajul "BACKWARD"

  }

  public static void setMotorActionForward()
  {

  Motor.A.forward();//robotul avanseaza
  //LCD.drawString("FORWARD", 0, 0); // se afiseaza pe display-ul LEGO
in coltul din stanga sus  mesajul "FORWARD"

  }

  public static void setMotorActionBackward_B()
  {

  Motor.B.backward();//robotul se intoarce
  //LCD.drawString("BACKWARD", 0, 0); // se afiseaza pe display-ul LEGO
in coltul din dreapta sus mesajul "BACKWARD"

  }

  public static void setMotorActionForward_B()
  {

  Motor.B.forward();//robotul avanseaza
  //LCD.drawString("FORWARD", 0, 0); // se afiseaza pe display-ul LEGO
in coltul din stanga sus  mesajul "FORWARD"

  }
  public static void stopMotor()
  {

    Motor.A.stop();
    Motor.B.stop();

  }
  public static void OutProgram()
  {
        System.out.println("Good Bye");
        System.exit(0);
```

```
  }
  public static void main( String args[] )
      {
            new Communication();
        Communication mainFrame= new Communication();
            mainFrame.setVisible( true );

            System.out .println("Welcome to LEGO NXT");
            UltrasonicSensor ultrasonic = new
UltrasonicSensor(SensorPort.S2);
            System.out.println("distance:" + ultrasonic.getDistance());
      }
}
```