

Approfondire Java divertendosi  
con

**LEGO® MINDSTORMS™**

Daniela Ruggeri  
ruggeridany@yahoo.it

# Divertirsi con Java

- *Il modo migliore di apprendere concetti difficili è applicarli al gioco.*
- *Tramite la realizzazione pratica i concetti teorici diventano immediatamente chiari*
- *Favorisce la nascita di community e quindi l'aumento del grado di conoscenza sull'argomento.*
- *Crea una sana competizione tra gli individui che li stimola a migliorare*

# LEGO® MINDSTORMS™



# Componenti

- *2 motori adatti a far muovere corpo/ruote/arti del robot*
- *2 sensori di contatto utili per far compiere al robot determinate azioni al premere/rilasciare del sensore*
- *1 sensore ottico utile per determinare l'intensità di luce e effettuare determinate azioni in base a questa intensità*
- *Un trasmettitore a raggi infrarossi per la porta USB per trasferire i programmi da PC all'RCX (IR Torre)*
- *La Constructopedia che è il manuale di costruzione dei robot.*

# Componenti



# Altri sensori



# Il cuore del sistema

*componente programmabile RCX (Robotics Command Explorer) brik:*

- piccolo computer contenuto dentro un mattone LEGO giallo.*
- processore Hitachi 8-bit (16 MHz), 16 Kb di ROM, 32 Kb di RAM, 3 porte di input per i sensori, 3 porte di output per i motori, un display LCD a 5 caratteri e una porta di comunicazione a infrarossi.*



# Robotics Invention System 2.0



# LEGO Java Operating System

**lejos\_win32\_2\_1\_0.zip**

scaricabile dal sito <http://lejos.sourceforge.net/>

# DIRECTORY LEJOS

- **bin** contiene gli eseguibili
- **classes** sorgenti java
- **common** contiene i files che sono usati per generare il codice per il linker e la virtual machine.
- **docs** documentazione
- **examples** esempi
- **gameboy\_impl** file per la piattaforma Nintendo Game Boy
- **jtools** codice java base per il computer e codice del linker leJOS, usa i file di common/\*.db.

# DIRECTORY LEJOS

- **rcx\_impl** contiene codice C specifico per il firmware RCX.
- **regression** contiene tutta una serie di test di regressione per testare leJOS
- **tools** contiene codice C per creare tools LeJOS
- **unix\_impl** Questa directory contiene il codice necessario a creare un tool di emulazione Unix.
- **vmsrc** contiene codice C per la virtual machine.

## Librerie jar

- *rcxrcxcomm.jar* – gestione sensori, motori e comunicazione da RCX al pc
- *pcrcxcomm.jar* – gestione comunicazione da pc a RCX

# API Lejos

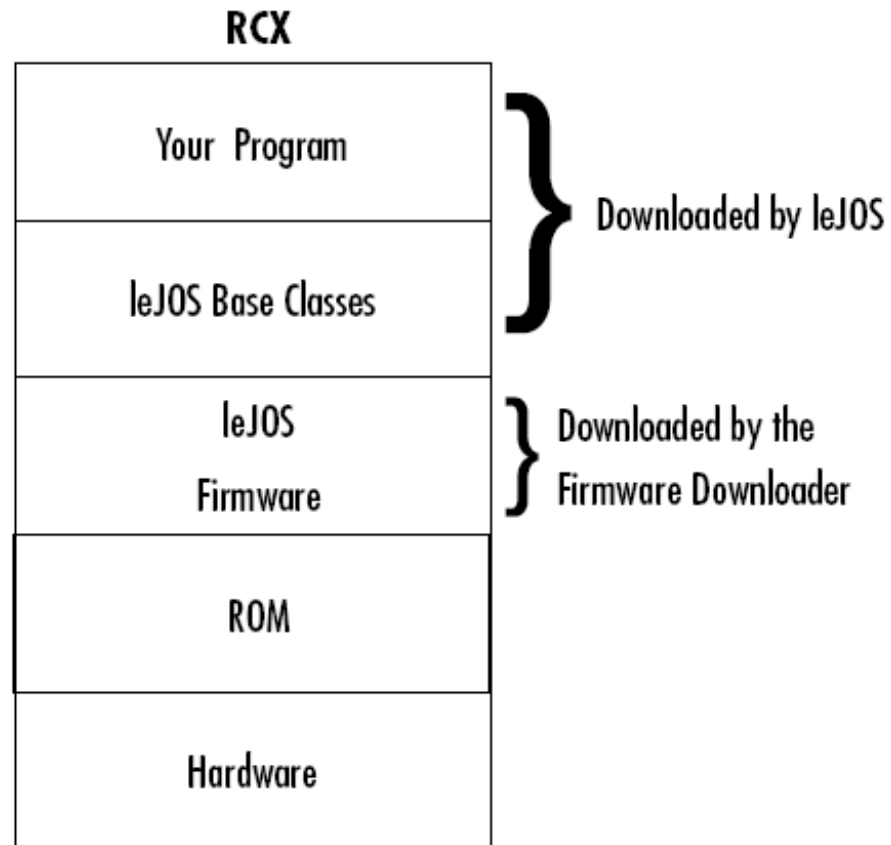
## Packages

<b>java.io</b>	Supporto per Input/Output.
<b>java.lang</b>	Classi del linguaggio base Java
<b>java.net</b>	Supporto per Networking/sockets
<b>java.util</b>	Utilità
<b>javax.servlet.http</b>	Classi per gestire un Web Server con <u>lejos</u>
<b>josx.platform.rcx</b>	Classi per gestire sensori RCX, motori RCX ecc.
<b>josx.rcxcomm</b>	Classi per gestire la comunicazione tra RCX e il PC
<b>josx.robotics</b>	Classi e interfacce per gestire azioni e comportamenti dei robot in maniera parametrizzata
<b>josx.util</b>	Altre classi di utilità

# Che cosa ha in meno?

- **Non supporta garbage collection**
- **Non supporta il comando switch**
- **Non supporta tipi primitivi long**
- **La lunghezza massima degli array è 511.**
- *instanceof* ritorna *true* per le interfacce. Non permesso sugli array.
- *java.lang.Class* non crea istanze (costruttore non applicabile) non si può sincronizzare su metodi statici.
- *Class.forName* restituisce un *ClassNotFoundException*, perché non supporta il caricamento dinamico delle classi.
- **Non supporta J2SDK.** Solo versione jdk 1.1.

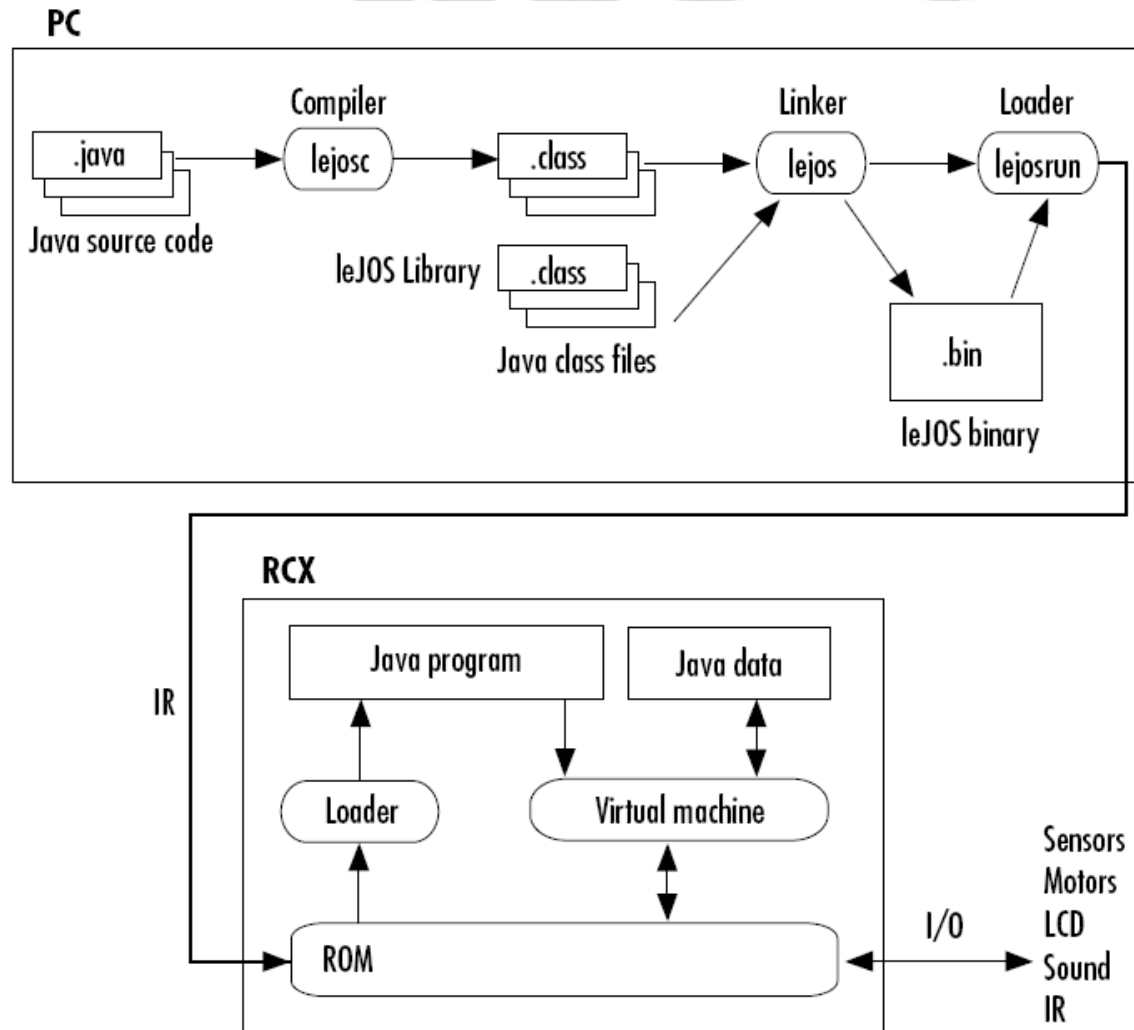
# Architettura leJOS



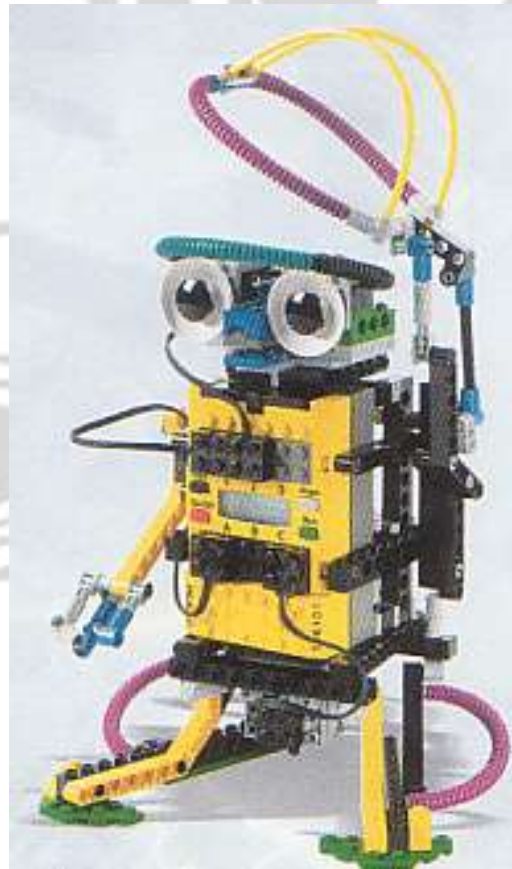
# Comandi per RCX

- *lejosfirm* fa caricare il firmware in RCX
- *lejos* compila i sorgenti java
- *lejosrun* trasferisce il programma binario in RCX tramite la torre
- *lejos* trasforma i programmi in file binari e li trasferisce in RCX tramite la torre IR Tower

# Trasferimento codice



# Inventorbot (roby)



# Specifiche di programma

1. Fare in modo che il robot emetta dei suoni e alzi il cappello quando viene toccata la mano
3. Fare in modo che il robot emetta suoni diversi e giri in sensi diversi a secondo la durata della pressione esercitata sulla mano
5. Fare in modo che il robot saluti se viene colpito da una luce in faccia

# Che serve sapere di Java?

- La base del linguaggio
- Gli eventi a delega
- I thread.

# Calcolo del tempo

- **javax.util.Timer**

Oggetto Timer per gestire un intervallo di tempo

- **javax.util.TimerListener**

Interfaccia ascoltatrice usata con la classe Timer

In particolare il metodo **timedOut()** dell'interfaccia viene chiamato allo scadere del tempo fissato.

# Cambiamento stato sensore

- **osx.platform.rcx.Sensor**

Astrazione di un sensore. In particolare **Sensor.S1** rappresenta il sensore collegato alla porta 1

- **josx.util.SensorListener**

Interfaccia ascoltatrice usata con la classe **Sensor**

Il metodo **stateChanged(Sensor aSource, int aOldValue, int aNewValue)** per cambiamento stato del sensore.

- **osx.platform.rcx.SensorConstant**

Interfaccia usata con la classe **Sensor** di tutte le costanti riguardanti un sensore

## Altre classi usate

- **osx.platform.rcx.Motor**  
Astrazione di un motore. In particolare **Motor.A** e **Motor.C** rappresentano i motori collegati alle porte A e C
- **osx.platform.rcx.Button**  
Astrazione di un bottone RCX. In particolare nell'esempio è utilizzato **Button.RUN** ma esistono anche **Button.VIEW** e **Button.PRGM**
- **osx.platform.rcx.Sound**  
Classe usata per emettere suoni

# Realizzazione calcolo del tempo

// questa variabile vale true se il tempo trascorso è minore di 3 secondi

```
boolean timeLess3 = true;
```

*/\* Creazione istanza della classe Timer associandola alla lista ascoltatrice che conterrà il tempo che passa \*/*

```
Timer timer = new Timer(3000,new TimerListener() {
```

// trascorsi 3000 millisecondi viene chiamato questo metodo

```
public void timedOut() {
```

```
    timeLess3 = false;
```

```
    timer.stop();
```

```
}
```

```
});
```

# Sensori

<i><b>SENSOR_TYPE_LIGHT</b></i>	<b>Sensore ottico</b>
<i><b>SENSOR_TYPE_TOUCH</b></i>	<b>Sensore di contatto</b>
<i><b>SENSOR_TYPE_TEMP</b></i>	<b>Sensore di temperatura</b>
<i><b>SENSOR_TYPE_ROT</b></i>	<b>Sensore di rotazione</b>
<i><b>SENSOR_TYPE_RAW</b></i>	<b>Tutti</b>
<i><b>SENSOR_MODE_ANGLE</b></i>	<b>Misura dell'angolo (sensore di rotazione)</b>
<i><b>SENSOR_MODE_BOOL</b></i>	<b>Boleano true/false</b>
<i><b>SENSOR_MODE_DEGC</b></i>	<b>Temperatura in gradi Celsius</b>
<i><b>SENSOR_MODE_DEGF</b></i>	<b>Temperature in Fahrenheit</b>
<i><b>SENSOR_MODE_EDGE</b></i>	<b>Fa la media delle transazioni e aggiunge 1 per ciascun booleano da falso a vero e viceversa.</b>
<i><b>SENSOR_MODE_PCT</b></i>	<b>Percentuale</b>
<i><b>SENSOR_MODE_PULSE</b></i>	<b>Conta le transazioni e aggiunge 1 per ciascun booleano da false a vero e viceversa.</b>
<i><b>SENSOR_TYPE_RAW</b></i>	<b>Raw valore tra 0 (0V) – 1023 (5V)</b>

# Unità di misura Sensori

<b>Volt</b>	<b>Raw</b>	<b>Sensor Ohms</b>	<b>Ottico</b>	<b>Temp. C</b>	<b>Contatto</b>
0.0	0	0	-	-	1
1.1	225	2816	-	70.0	1
1.6	322	4587	100	57.9	1
2.2	450	7840	82	41.9	1
2.8	565	12309	65	27.5	0
3.8	785	32845	34	0.0	0
4.6	945	119620	11	-20.0	0
5.0	1023	Inf	0	-	0

# Stati Sensore di contatto

```
Sensor.S1.activate(); // attiva il sensore collegato alla porta 1
Sensor.S1.setPreviousValue (0); // imposta valore iniziale del sensore a 0 (rilasciato)
// imposta tipo di sensore (sensore di contatto) e valori in output (booleani)
Sensor.S1.setTypeAndMode (SENSOR_TYPE_TOUCH,
                          SENSOR_MODE_BOOL);

Sensor.S1.addSensorListener(new SensorListener() {
// questo metodo viene chiamato quando il sensore cambia dallo stato premuto a quello
// rilasciato.
    public void stateChanged(Sensor aSource,
                             int aOldValue,
                             int aNewValue) {
        if (aOldValue == 1 || aNewValue == 0) {
// sensore in stato di premuto. Start del timer
            timeLess3=true;
            timer.start();
            return;
        }
    }
}
```

# Durata meno di 3 secondi

```
timer.stop();
try {
    if (timeLess3) {
        Sound.playTone(2100,300);
        Motor.C.forward();
        Thread.sleep(1000);
        Motor.C.stop();
        Sound.beep();
        Motor.A.forward();
        Thread.sleep(2000);
        Motor.A.stop();
        Motor.C.backward();
        Thread.sleep(1000);
        Motor.C.stop();
    }
}
```

# Durata più di 3 secondi

```
else {  
    Sound.systemSound(true,3);  
    Motor.C.backward();  
    Thread.sleep(1000);  
    Motor.C.stop();  
    Sound.systemSound(true,4);  
    Motor.A.backward();  
    Thread.sleep(2000);  
    Motor.A.stop();  
    Motor.C.forward();  
    Thread.sleep(1000);  
    Motor.C.stop();  
    Sound.systemSound(true,5);  
    timeLess3=true;  
}
```

# Stati Sensore ottico

```
Sensor.S3.activate(); // attiva il sensore collegato alla porta 1
Sensor.S3.setPreviousValue (0); // imposta valore iniziale del sensore a 0
(rilasciato)
// imposta tipo di sensore (sensore ottico) e valori in output (booleani)
Sensor.S3.setTypeAndMode (SENSOR_TYPE_LIGHT,
                           SENSOR_MODE_PCT);

Sensor.S3.addSensorListener(new SensorListener() {
// questo metodo viene chiamato quando il sensore cambia dallo stato
// premuto a quello rilasciato.
public void stateChanged(Sensor aSource,
                        int aOldValue,
                        int aNewValue) {
if (aNewValue > 80) {
Motor.A.forward();
Thread.sleep(2000);
Motor.A.stop();
}
}
```

# Comandi Linux per RCX

```
#!/bin/bash
```

```
CLASSPATH=.:
```

```
~/lejos_1_0_4alpha/lejos/lib/pcrcxcomm.jar:
```

```
~/lejos_1_0_4alpha/lejos/lib/rcxrcxcomm.jar
```

```
path=c:\jdk1.4.0\bin;~/lejos_1_0_4alpha/lejos/bin:$path
```

```
# driver usb accessibile al link http://legousb.sourceforge.net)
```

```
RCXTTY=USB
```

```
export PATH RCXTTY CLASSPATH
```

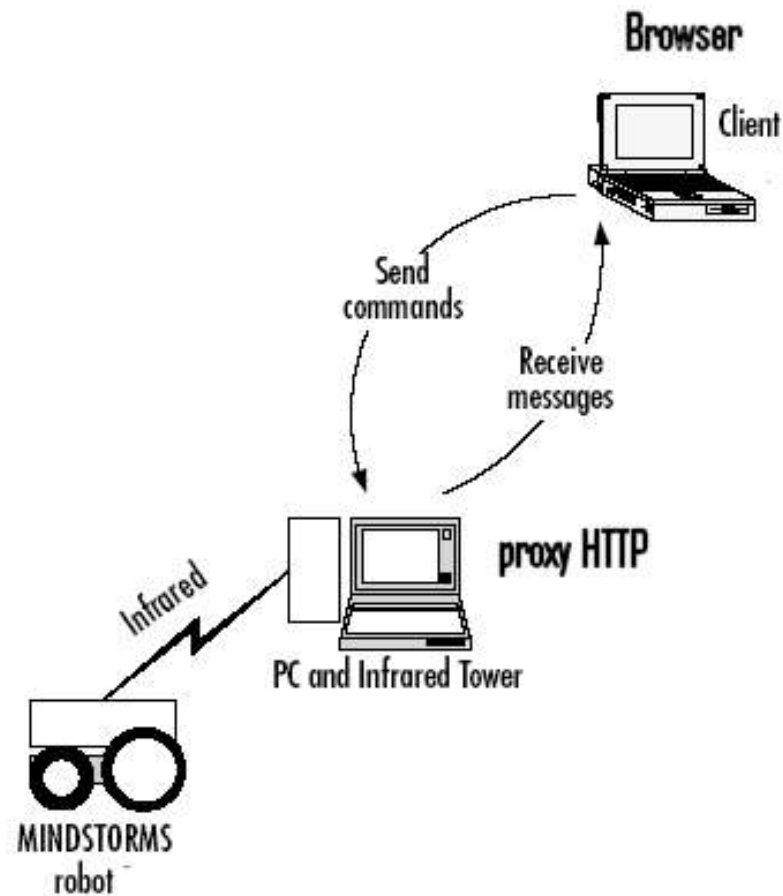
```
lejos Inventorbot.java
```

```
lejos Inventorbot
```

# Comandi DOS per RCX

```
set classpath=.;c:\lejos\lib\pcrcxcomm.jar;  
c:\lejos\lib\rcxrcxcomm.jar  
set path=c:\jdk1.4.0\bin;c:\lejos\bin;%path%  
set RCXTTY=USB  
lejos Inventorbot.java  
lejos Inventorbot
```

# Controllo via HTTP



# Specifiche di programma

1. Fare in modo che il robot emetta dei suoni, alzi il cappello e giri a sinistra quando riceve il comando 'sinistra'
3. Fare in modo che il robot emetta altri suoni, alzi il cappello e giri a destra quando riceve il comando 'destra'

Alla fine il robot deve rispondere di aver eseguito il comando in linguaggio in html.

# Che serve sapere di Java?

- servlet
- linguaggio HTML

# josx.rcxcomm.HttpProxy

Da shell:

```
java josx.rcxcomm.HttpProxy -port <portaproxy>
```

Dal browser:

```
http://<ipserver>:<portaproxy><transazione>?var1=v  
alore1&var2=valore2....
```

# Comandi

Da shell:

```
java josx.rcxcomm.HttpProxy -port 9000
```

Dal browser:

```
http://localhost:9000/comando?saluto=sinistra
```

o

```
http://localhost:9000/comando?saluto=destra
```

# Impostazioni iniziali su RCX

```
public class HTTPInventorbot extends HttpServlet implements
    SensorConstants {
    private OutputStream out;
    private static String html1 = "<HTML><BODY><P>";
    private static String html2 = "<FONT color=#ff0000
    size=7>ESEGUITO&nbsp;SALUTO A
    DESTRA!!!</FONT>";
    private static String html3 = "</P></BODY></HTML>";
    private static String html4 = "<FONT color=#ff0000
    size=7>ESEGUITO&nbsp;SALUTO A
    SINISTRA!!!</FONT>";
    private static String type = "text/html";
```

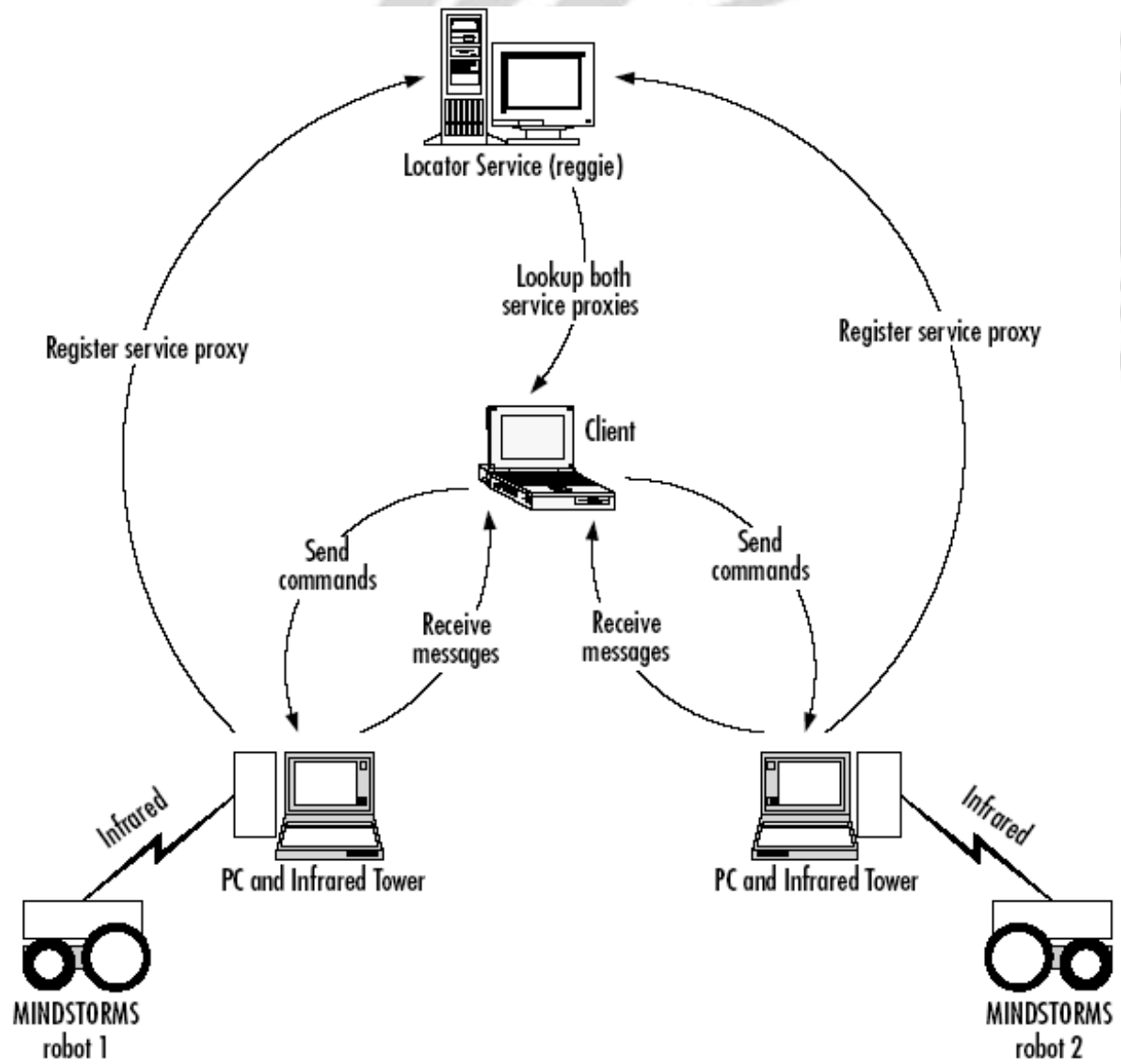
# public void doGet() (ricezione)

```
out = response.getOutputStream();
String saluto = request.getParameter("saluto");
if (saluto != null) {
    try {
        if (saluto.charAt(0)=='s') { // parametro saluto='sinistra'
            tournLeft(); // gira a sinistra
            println(html1);
            println(html4);
            println(html3);
        } else {
            tournWrite(); // gira a destra
            println(html1);
            println(html2);
            println(html3);
        }
    }
}
```

# Spedizione risposta

```
private void println(String s) throws IOException {  
    for(int i=0;i<s.length();i++)  
        out.write((byte) s.charAt(i));  
}
```

# Esempio scenario JINI



# LEGO® remote control

MINDSTORMS  
ULTIMATE ACCESSORY SET



package

`josx.platform.rcx.remotecontrol`

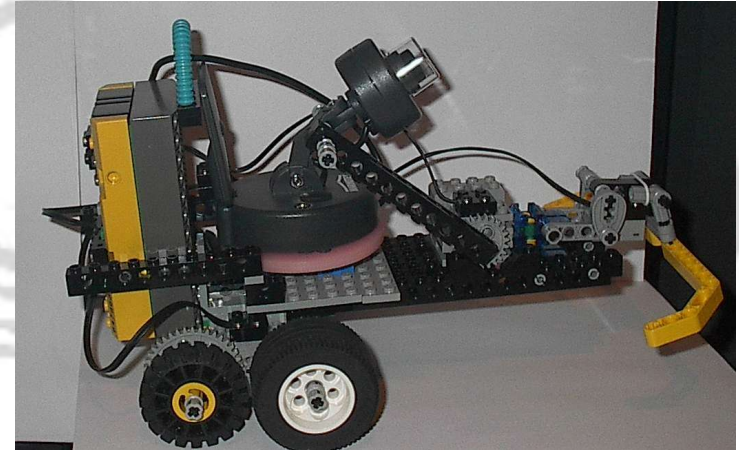
1. ascoltare i messaggi che arrivano dal remote control
2. identificare i comandi associati con il messaggio
3. eseguire il codice stabilito dall'utente su RCX associato a quel comando

# Classi

- **RemoteControlSensor** componente responsabile dell'intercettazione dei messaggi
- **RemoteControlListener** interfaccia ascoltatrice eventi associati ai bottoni del Remote Control
- **RemoteControlAdapter** classe adattatore degli eventi

# Package josx.vision

- Intercettare movimenti
- Intercettare colori
- Intercettare l'intensità di luce
- Scattare istantanee
- Registrare video compresi suoni
- Emettere suoni
- Riconoscere parole (tramite Java Speech API e IBM ViaVoice)



**LEGO Mindstorms  
Vision Command**

## Telecamere che può utilizzare:

- Lego Cam di Lego Mindstorms Vision Command
- X10 XCam2 wireless camera

# RemoteControl.java



# LEGO® MINDSTORMS® NXT



- Brick con microprocessore a 32-bit
- più memoria e FLASH
- 3 motori con sensori di rotazione
- Nuovi sensori del suono
- Nuovi sensori visuali ultrasonici
- Nuovi sensori di contatto

# LEGO® MINDSTORMS® NXT

- 519 nuovi elementi LEGO TECHNIC per costruzioni resistenti e migliori.
- 4 porte input, 3 porte output, 7 cavetti
- Display a matrice
- Piezo speakers
- supporto USB 2.0 e Bluetooth
- Interfaccia per PC e Mac
- Programmi di costruzione robot

# Articoli

- *Robotics: Using Lego Mindstorms and Java*  
<http://today.java.net/pub/a/today/2005/02/21/robotics.html>
- *Futurama: Using Java Technology to Build Robots That Can See, Hear Speak, and Move*  
<http://java.sun.com/developer/technicalArticles/Programming/robotics/>
- *Programmare Lego Mindstorms con java*  
<http://www.jia.it/modules.php?name=Sections&op=viewarticle&artid=25>

# Link utili

<http://mindstorms.lego.com>

<http://lejos.sourceforge.net/>

<http://digilander.libero.it/lego.mindstorms/>

# Bibliografia

- ♦ **Programming Lego Mindstorms with java**  
di *Dario Laverde*
- ♦ **Core Lego Mindstorms Programming**  
di *Brian Bagnall*

# Buon Divertimento!



[ruggeridany@yahoo.it](mailto:ruggeridany@yahoo.it)