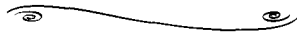


Contents

	Foreword	xix
RCX: The Robot's Brain	Chapter 1 Introducing LEGO MINDSTORMS	1
<ul style="list-style-type: none">■ The RCX is a microcomputer that interfaces with input and output devices. Programs can be written on a PC and then downloaded to the unit through the IR tower.■ The RCX uses two types of memory: read-only memory (ROM) and modifiable random access memory (RAM).■ The RCX can be expanded in two ways: using a different programming software like NQC or the Java APIs, or replacing the default firmware with a new one.	Introduction	2
	The LEGO MINDSTORMS RIS Kit	2
	A Brief History of the LEGO MINDSTORMS RIS	3
	What's Included with the Robot Kit	4
	RCX: The Robot's Brain	7
	How It Works	7
	The Physical Structure	7
	The Logical Structure	9
	Expanding the RCX Brain	11
	Replacing the RIS Software	11
	Replacing the RCX Firmware	14
	The RIS Software Environment	16
	Installing the Firmware into the RCX	16
	A Visual Programming Interface: RCX Code	18
	RCX Bytecodes	20
	The LEGO Assembly Code for the RCX	20
	LEGO Expansion Kits	21
	Alternative Processing Units	22
	Add-on Building Elements	24
	Summary	26
	Solutions Fast Track	26
	Frequently Asked Questions	28

Serial Port Control and Status Signals		Chapter 2 The Java Communications API	31
<hr/>		Introduction	32
Abbreviation	Definition	Overview of the Java Communications	
RTS	Request To Send	Extension API (JCE API)	32
CTS	Clear To Send	Understanding the JCE Framework	34
DTR	Data Terminal Ready	Port Discovery and Enumeration	34
DSR	Data Set Ready	Port Ownership Management	37
RI	Ring Indicate	Asynchronous event-based I/O	42
CD	Carrier Detect	Encapsulation of Underlying	
OE	Overrun Error	Native Ports	43
FE	Framing Error	Java Communication API's Event Based	
PE	Parity Error	Architecture	43
BI	Break Indicator	Installing and Configuring the	
DA	Data Available	Java Communications API	46
BE	Output Buffer Empty	Installing the Native Library	46
<hr/>		Installing the Java comm.jar Library	47
		The javax.comm.properties	
		Configuration File	47
		Configuring your Development	
		Environment	48
		Reading and Writing to Serial Ports	50
		Simple Read Example	50
		Simple Write Example	54
		Debugging with Serial Ports:	
		The "Black Box" Example	57
		Selected Code	62
		Extending the Java Communications API	64
		Using More than Serial or Parallel Ports	65
		USB Port Access	68
		Summary	76
		Solutions Fast Track	77
		Frequently Asked Questions	78
		Chapter 3 Communicating with the RCXPort API	81
		Introduction	82
		Overview of the RCXPort Java API	82

**Troubleshooting
Problems with RCXPort**



There are several things that could potentially go wrong when trying to run RCXPort. The more common mistakes are listed below:

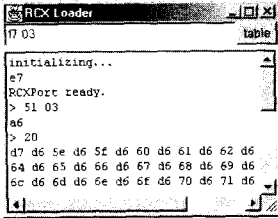
- Make sure your RCX is turned on and within range of the IR tower.
- Make sure the IR tower is properly connected to the correct serial port.
- Make sure that another program (possibly running in the background) isn't using the serial port.
- Make sure you have named the correct serial port (COM1, for instance) in the command line.
- If you are downloading byte code from a file, make sure that file is in the same directory as RCXPort.
- Make sure you have downloaded the Java Communications API, and that it is correctly installed.
- Make sure that your classpath references both *rcxport.jar* and

How RCXPort Works	82
Formatting RCX Commands	82
RCXPort Object Model	84
Limitations of RCXPort	85
Compiling Java into Machine Code	85
Restrictions of Using Direct Mode	85
Reliance on Java Communications API	86
Programming the RCX Using RCXPort	86
Downloading Programs with RCXPort	91
Interfacing External Software with RCXPort	93
An Advanced Example Using RCXPort	97
Summary	106
Solutions Fast Track	106
Frequently Asked Questions	108

**Chapter 4 Communicating with
the RCXJava API 111**

Introduction	112
Designing an RCX Java Communications Architecture	112
The Basic Components of an RCX API	122
Port Configuration and Error Handling	122
Protocol Management and Message Parsing	122
Tower Communications	123
RCX Communications	123
Reusability: Protocols and Ports	123
Supporting Similar Protocols	123
Using Java Interfaces to Support Ports Other than Serial Ports	124
Overview of the RCXJava API	124
The RCX Package	125
Classes	125
Interfaces	126
Exceptions	127

RCXLoader



Using the RCXLoader Application 129

 The User Interface 131

 Handling and Parsing Response
 and Error Messages 131

Beyond Serial Port Communications:

 The RCXApplet Example 131

 Communicating over the Network 132

 Using Sockets 135

 Building and Extending the Simple Applet 139

Direct Control Programming

 for the RCX Using Java 147

 Basic Remote Control Application 147

 Creating a Direct Control
 Framework for Java Programs 150

 Direct Control Using AI 151

Summary 164

Solutions Fast Track 165

Frequently Asked Questions 167

Create Custom Components

- Q:** What happens if more than ten programs are downloaded to the RCX at once?
- A:** The 11th (and further) programs will appear in the program list, represented using blanks.
- Q:** What if I have a LEGO tower connected to the USB port?
- A:** Set the RCXTTY environment variable to the value USB, instead of to a serial port value.

Chapter 5 The leJOS System 169

Introduction 170

Basic leJOS Usage Guidelines 170

 Using the lejos Compiler 172

The LEGO Java Operating System 173

 The TinyVM 176

Overview of the leJOS Architecture 177

 Exploring the josx.platform.rcx Package 178

 Using the Button and ButtonListener
 Classes 179

 Using the MinLCD, LCD, Segment,
 LCDConstants, and TextLCD Classes 180

Using leJOS: A Simple Example 195

 Controlling Motors 195

 Reading Sensors 196

Summary 200

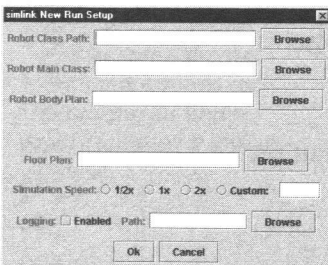
Solutions Fast Track 200

Frequently Asked Questions 202

Debugging leJOS Programs

- The best way to debug a leJOS program is to use the Sound and LCD classes in unison to provide you with feedback of the robot's state.
- Normal Java exception handling applies to leJOS, allowing you to separate code for normal operation and code for error situations.

Dialog to Create a New Simulator Run



Chapter 6 Programming for the leJOS Environment 203

Introduction	204
Designing Java Programs to Run in leJOS	204
Using Memory Wisely	205
Using the Right Java Classes (and Using Them Correctly)	206
An Advanced Programming	
Example Using leJOS	210
Controlling the Steering	222
Restricted Steering	223
Getting Back to the Line	225
Debugging leJOS Programs	236
Using Sounds and the LCD	236
Exception Handling with leJOS	237
Testing leJOS Programs	238
Using the leJOS Simulator	238
Summary	241
Solutions Fast Track	241
Frequently Asked Questions	243

Chapter 7 leJOS Tools 245

Introduction	246
Programming Environments for leJOS	246
The Command-Line Tools	
that Interact with the RCX	247
Using the lejos Compiler	247
Using the lejos Linker	248
Using the lejosfirmddl Downloader	250
The Command-line leJOS Emulator	251
Using the emu-lejos Emulator	251
Using the emu-lejosrun Linker	251
Using Existing IDEs	251
Configuring Forte	252
Using the leJOS Visual Interface	253

**Encoding of Java Types
in Signatures**

Java Type	Encoding
void	V
boolean	Z
char	C
byte	B
short	S
int	I
long	J
float	F
double	D
package.....	Lpackage/
package.Class .../package/	Class;
array	[(type of elements follows)

The leJOS Visual Interface	253
Installing IVI	254
Setting Up IVI	255
Basic Usage	257
Using a leJOS Simulator: Simlink	258
Getting Started with Simlink	258
Installing and Configuring Simlink	260
Running Your First Simulation	261
Designing a Floor Plan for Simlink	262
Non-visual Declarations	264
Visual Declarations	264
Navigational Declarations	268
Creating a New Simlink Robot Body	269
Creating a Body: Passive Components	270
Active Body Classes: Sensors and Wheels	272
Creating a Simple Robot Design	277
Future Tools for Designing Robots	280
Additional Tips and Tools for leJOS	281
RCXDownload	282
RCXDirectMode	283
Summary	285
Solutions Fast Track	285
Frequently Asked Questions	287

Chapter 8 leJOS Internals 289

Introduction	290
Advanced Usage of leJOS	290
Multiprogram Downloading	291
Storing Persistent Data	292
Examining leJOS Internals	297
From Source Code to Execution	297
Inside the leJOS Linker	299
The C Wrapper	299
The Java Main Program	300
Building the Binary	302
The leJOS Binary Format	303

Inside the leJOS Firmware	304
The Structure of the leJOS Virtual Machine	305
Real-Time Behavior	306
RCX Memory Layout	308
The Emulator	310
The leJOS Source Code	311
Extending leJOS with Native Methods	314
Native Methods in leJOS	314
Adding a Native Method	315
Additional Tips and Tricks with leJOS	323
Changing Stack Sizes	324
Determining the Amount of Free Memory	324
Measuring Latency	324
Summary	327
Solutions Fast Track	328
Frequently Asked Questions	330

Overview of Jini

- Jini is a Java technology built on top of RMI, enabling clients and services to interact in a network with little administrative overhead.
- One feature of Jini is that it is vary applicable to embedded devices, including devices like the RCX.
- The Jini Technology Starter Kit (TSK) includes all of the required jar files as well as some service implementations such as reggie, an implementation of a lookup service.

Chapter 9 Programming LEGO MINDSTORMS with Jini	333
Introduction	334
Overview of the Jini Architecture	334
Jini as a Network Protocol	336
A Simple Jini Service Example	337
What's Required for Installing and Running Services	337
A Simple Service and Client	341
Proxies and Service Architectures	355
Selecting the Right Architecture	356
Using Proxies	356
A RCX Jini Proxy Service	356
Why a Proxy?	356
Interfacing with the RCX Java API	358
Using the RCX Jini Service:	
Example Server and Client	358
A RCX Jini Server	359
A RCX Jini Client	378

Summary	394
Solutions Fast Track	394
Frequently Asked Questions	396
Appendix A Resources	399
Appendix B Programming	
LEGO MINDSTORMS with Java Fast Track	407
Index	421