Welcome to QNX Momentics

For Windows®, Linux®, and QNX® Neutrino® hosts
Contents

About This Guide ix
Welcome to QNX Momentics xi
What’s in this guide? xii
Typographical conventions xiii
   Note to Windows users xiv

1 What is QNX Momentics? 1
Choice, tools, source, and help 3
QNX Momentics at a glance 3
Getting started before you have your target 5
   x86 (PC) target 5
   x86 non-BIOS target 6
   Any supported CPU family 6
A word about licensing 7

2 Getting Started 9
Supported hosts 11
Installing and removing QNX Momentics 11
What do I do if my installation fails? 11
What did I just install? 12
What’s on my desktop? 15
Upgrading your software 15
Managing source code 16
Running QNX Neutrino self-hosted 16
Mixing a self-hosted machine with other hosts 17
Can different versions of QNX Momentics coexist? 17
Running QNX Neutrino on a target machine 18

3 How to Get Help 19
Overview of the documentation 21
QNX Momentics Development Suite 21
Integrated Development Environment 21
QNX Neutrino Realtime Operating System 22
Photon microGUI 23
BSPs, DDKs, and specialty tools 23
Dinkum C and C++ 24
Viewing the documentation 25
Within the IDE 25
The Photon Helpviewer 25
Keyword indexes 26
What should I read first? 27
During development (on your host) 27
At runtime (on your target) 28
Where key features are documented 28
Related reading 37
On QNX Neutrino 37
On POSIX 38
On TCP/IP 38
Getting quick help with the use command 39
Hover help in the IDE 39
Visit http://www.qnx.com 39
Support plans 40
Training 40
Custom engineering and consulting 40

Glossary 43
List of Figures

QNX Momentics is the development environment on your host for the QNX Neutrino RTOS running on your target. xi
The host-related directory structure. 12
The target-related directory structure. 13
Getting help in the IDE. 25
Photon Helpviewer. 26
Welcome to QNX Momentics

Thank you for choosing the QNX Momentics development suite. You have everything you need to build and maintain your QNX Neutrino-based embedded system: a comprehensive set of integrated development tools, lots of in-depth documentation, as well as powerful diagnostics and optimization tools for your system once it’s up and running on your target.

Board support packages (BSPs), driver development kits (DDKs), and technology development kits (TDKs) are available separately; you can download them from our website, http://www.qnx.com/.

Trusted and proven in countless embedded systems, QNX Neutrino has a growing reputation as the world’s most reliable RTOS. We now invite you to explore the advanced tools that QNX Momentics adds to QNX Neutrino.

QNX Momentics is the development environment on your host for the QNX Neutrino RTOS running on your target.
What’s in this guide?

The *Welcome to QNX Momentics* guide is intended to introduce you to the QNX Momentics development suite and help you start developing applications for QNX Neutrino. This guide is organized around these main topics:

- What is QNX Momentics?
- How do I install and set up my system?
- How do I get help?

The following table may help you quickly find information in this guide:

<table>
<thead>
<tr>
<th>To:</th>
<th>Go to:</th>
</tr>
</thead>
<tbody>
<tr>
<td>See a list of this product’s components</td>
<td>“QNX Momentics at a glance” in What is QNX Momentics?</td>
</tr>
<tr>
<td>Understand the license agreement</td>
<td>“A word about licensing” in What is QNX Momentics?</td>
</tr>
<tr>
<td>Use a virtual machine</td>
<td>“Getting started <em>before</em> you have your target” in What is QNX Momentics?</td>
</tr>
<tr>
<td>Install QNX Momentics on your host machine or remove it</td>
<td>“Installing and removing QNX Momentics” in Getting Started</td>
</tr>
<tr>
<td>Find out where components are installed on your host machine</td>
<td>“What did I just install?” in Getting Started</td>
</tr>
<tr>
<td>Upgrade your software</td>
<td>“Upgrading your software” in Getting Started</td>
</tr>
<tr>
<td>Run the QNX Neutrino RTOS on a target machine</td>
<td>“Running QNX Neutrino on a target machine” in Getting Started</td>
</tr>
<tr>
<td>Get started with the documentation</td>
<td>How to Get Help</td>
</tr>
</tbody>
</table>

*continued…*
Typographical conventions

To:  Go to:
Get technical support  How to Get Help
Find the meaning of terms used in QNX documentation  Glossary

If you’ve already developed applications with QNX Momentics 6.2.1, see the 6.3 and 6.2.1 Compatibility notes. You can find the latest version of this document on our website, http://www.qnx.com/.

Typographical conventions

Throughout this manual, we use certain typographical conventions to distinguish technical terms. In general, the conventions we use conform to those found in IEEE POSIX publications. The following table summarizes our conventions:

<table>
<thead>
<tr>
<th>Reference</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code examples</td>
<td>if( stream == NULL )</td>
</tr>
<tr>
<td>Command options</td>
<td>-lR</td>
</tr>
<tr>
<td>Commands</td>
<td>make</td>
</tr>
<tr>
<td>Environment variables</td>
<td>PATH</td>
</tr>
<tr>
<td>File and pathnames</td>
<td>/dev/null</td>
</tr>
<tr>
<td>Function names</td>
<td>exit()</td>
</tr>
<tr>
<td>Keyboard chords</td>
<td>Ctrl-Alt-Delete</td>
</tr>
<tr>
<td>Keyboard input</td>
<td>something you type</td>
</tr>
<tr>
<td>Keyboard keys</td>
<td>Enter</td>
</tr>
<tr>
<td>Program output</td>
<td>login:</td>
</tr>
</tbody>
</table>

continued…
Typographical conventions

<table>
<thead>
<tr>
<th>Reference</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Programming constants</td>
<td>NULL</td>
</tr>
<tr>
<td>Programming data types</td>
<td>unsigned short</td>
</tr>
<tr>
<td>Programming literals</td>
<td>0xFF, &quot;message string&quot;</td>
</tr>
<tr>
<td>Variable names</td>
<td>stdin</td>
</tr>
<tr>
<td>User-interface components</td>
<td>Cancel</td>
</tr>
</tbody>
</table>

We use an arrow (→) in directions for accessing menu items, like this:

You’ll find the Other... menu item under Perspective→Show View.

We use notes, cautions, and warnings to highlight important messages:

Notes point out something important or useful.

CAUTION: Cautions tell you about commands or procedures that may have unwanted or undesirable side effects.

WARNING: Warnings tell you about commands or procedures that could be dangerous to your files, your hardware, or even yourself.

Note to Windows users

In our documentation, we use a forward slash (/) as a delimiter in all pathnames, including those pointing to Windows files.

We also generally follow POSIX/UNIX filesystem conventions.
Chapter 1

What is QNX Momentics?

In this chapter...

Choice, tools, source, and help  3
QNX Momentics at a glance  3
Getting started before you have your target  5
A word about licensing  7
Choice, tools, source, and help

As a complete package designed for embedded systems developers, the QNX Momentics development suite gives you everything you need at every stage of your product-development cycle:

Choice  Choice of host (Windows, Linux, QNX Neutrino), choice of target (ARM, MIPS, PowerPC, SH-4, XScale, x86), and choice of development language (C, C++, Embedded C++, Java).

Tools  Code development, editors, source control, compilers, libraries, profilers, analyzers, optimizers, etc.

Help  Documentation, forums, technical support programs, etc.

Full source code for numerous startup programs, IPLs, device drivers, etc. is available from our website.

QNX Momentics at a glance

Here are the main parts of the QNX Momentics development suite:

QNX Neutrino RTOS

The whole point of it all. If Neutrino is the “engine” that will empower the embedded system you’re developing, then QNX Momentics is the “factory” where you modify your engine as well as build, test, and finish your vehicles.

Integrated Development Environment

This is your toolbox. The IDE’s task-oriented interface helps you quickly set up your project, choose your programming language, choose a target processor, compile your code, connect to your target, transfer your application to your target, run it, debug it, profile it, and fine-tune it.
QNX Momentics at a glance

Command-line tools

If you aren’t using the IDE, you can use command-line tools to develop applications. For example, you can use `qcc` to compile and link, and `mkifs` to create an OS image.

Libraries

ANSI C, POSIX, Dinkum C++ (full and embedded), GNU C++ (x86 only), graphics, widgets, compression, etc.

Documentation

How-to guides, references, context-sensitive help, and technotes. See the chapter How to Get Help to help you find your way through the documentation.

If you’ve installed QNX Momentics, you can download these components from our website after logging into your myQNX account:

Board Support Packages

Software and step-by-step instructions to help you get Neutrino and your applications running on specific evaluation boards.

Driver Development Kits

Full source and detailed documentation to help you write your own drivers for various devices: audio, graphics, input (mice, keyboards, etc.), network, and USB.

Note that the QNX Momentics development suite includes the DDK documentation.

You can also get Technology Development Kits (TDKs), kits that augment the base Neutrino OS platform in QNX Momentics with specialized, value-added technologies. The TDKs currently include:

- Acoustic Processing
- Advanced Graphics (formerly 3D Graphics)
Getting started before you have your target

- Multimedia
- Web Browser

For more information about these components, contact your sales representative.

Getting started before you have your target

Assuming you plan to develop a Neutrino-based embedded system of some kind (e.g. a vehicle telematics system, a router, a medical imaging device), you can start developing your application, even before you have your target hardware.

And if you haven’t yet decided on the CPU family for your target (e.g. PowerPC, XScale), you can still begin developing your application. As a rule, with QNX Momentics you write your application code once, then compile it however many times you need for whatever supported targets you’re using. As you’ll see if you plan to use the IDE, you can build the very same project for any of our supported processors with just a couple of mouse clicks.

If you’re developing on a self-hosted Neutrino system, you can run your application directly on your development machine.

Consider the following scenarios:

x86 (PC) target

If your final target hardware will be a PC in some shape or form (e.g. a PC/104 SBC module), you can simply connect your host to any spare PC and use that as your temporary target. You should expect to make very few changes (if any) in your application when the time comes to move to your final target.

And if you don’t happen to have a spare PC, you could use a virtual machine — products such as Workstation by VMware (http://www.vmware.com) can emulate a separate, complete hardware environment right on your desktop.
Getting started before you have your target

Once you’ve set up a virtual machine (VM) on your host, you can then use it as your target — you can run, test, and debug your Neutrino applications on the VM as if it were an actual machine connected to your host.

Virtual machines don’t necessarily support hard realtime.

x86 non-BIOS target

In this case, the target computer (e.g. an LX800) isn’t a PC. It doesn’t have a BIOS, so it won’t boot and load Neutrino in the same way as a PC. In order to run Neutrino on such a target, you need an appropriate BSP for your particular hardware, including an IPL that could set up the processor and load Neutrino properly.

We’re continually adding new BSPs to our extensive selection; for a list, see the Products area of our website, \texttt{http://www.qnx.com}. If you don’t see a BSP for your target hardware, contact us.

You can develop your application without target hardware, because much of your code will be hardware-independent. For device drivers and other hardware-specific areas of your system, you may be able to use a virtual platform tool that accurately emulates your specific embedded system hardware.

For instance, Virtio (\texttt{http://www.virtio.com}) makes virtual platforms that emulate various processors and peripherals, right down to the CPU’s instruction set. You can download a free time-limited evaluation copy from the vendor’s website.

Any supported CPU family

Whatever your target hardware, you can still go a long way down your development path using QNX Momentics right now. Develop all your application-level components first, leaving device drivers and other hardware-specific details until you have your target hardware.
A word about licensing

As mentioned earlier, you might want to consider Virtio if you need a functionally accurate virtual prototype of your embedded platform before you have the actual hardware itself.

Platform-related issues

Whatever the specific CPU family, your application code will, in most cases, run unchanged when you go to your final target. But you may need to be aware of a few platform-related issues (e.g., endian differences, CPU speeds, I/O addressing, alignment of data structures, memory limitations, synchronization on multicore vs uniprocessor systems, etc.).

Fortunately, QNX Momentics includes several header files and convenience functions that will help you anticipate and easily handle such problems.

For more information on these and other platform-related issues, see Freedom from Hardware and Platform Dependencies in the Neutrino Programmer's Guide.

A word about licensing

Here are some general questions about licensing. For more information, visit the Licensing area of http://www.qnx.com, where you’ll find details on our licensing model, types of licenses, etc., or email licensing@qnx.com.

Where can I find my EULA?
You’ll find your Momentics End User License Agreement and the full license guides on your CD and also in the Licensing area of our website, http://www.qnx.com

I’ve installed QNX Momentics on my workstation. Can I also make a copy for my laptop?
Certainly. Our licensing policy is per seat (i.e., per person), not per machine. You may use QNX Momentics on your primary
A word about licensing

workstation as well as on a laptop or on a PC at your home, provided that you use only one system at any given time.

Can I get a Neutrino runtime system?

You can purchase the QNX Neutrino x86 Runtime Kit, which includes runtime binaries and data, but no development tools or source; for more information, please contact your QNX sales representative.

As an OEM, how do I get a runtime license?

To obtain a runtime license (which governs how you may distribute certain Neutrino runtime files as part of your product), please contact your QNX sales representative.
In this chapter...

Supported hosts 11
Installing and removing QNX Momentics 11
What do I do if my installation fails? 11
What did I just install? 12
What’s on my desktop? 15
Upgrading your software 15
Managing source code 16
Running QNX Neutrino self-hosted 16
Mixing a self-hosted machine with other hosts 17
Can different versions of QNX Momentics coexist? 17
Running QNX Neutrino on a target machine 18
Supported hosts

To install and use QNX Momentics, you’ll need a desktop machine running one of the following for your host development environment:

- Windows Vista, XP, or 2000
- Linux Red Hat Enterprise WS 4 or 5, Red Hat Fedora Core 6 or 7, SUSE 10, or Ubuntu 6.0.6 LTS or 7.0.4
- QNX Neutrino 6.3

For particular system requirements (e.g. amount of disk space you’ll need to install), see the installation note that came with your software.

Installing and removing QNX Momentics

QNX Momentics uses InstallShield on all hosts except Neutrino (which uses tarballs and shell scripts) to help you install and remove software. For specific instructions, see the installation note that was shipped with your software.

What do I do if my installation fails?

If for some reason (e.g. hardware incompatibility) your QNX Momentics system doesn’t install properly, you should:

1. Review the requirements and instructions given in the installation note that was shipped with your software.
2. If you can access the Internet, check the lists of supported hardware posted on our website (http://www.qnx.com).
3. Check the troubleshooting sections in the Neutrino User’s Guide.
4. Contact us for help, or post questions in one of our forums. For more information, see the How to Get Help chapter.
What did I just install?

QNX Momentics software is organized around these two main areas:

- **Host-related**
  All your libraries, executables, etc., designed to run on your host system (e.g. Windows).

- **Target-related**
  All CPU-specific components, as well as certain common things that you can use on any target system.

The `QNX_HOST` environment variable identifies the directory that holds the host-related components:

```
$QNX_HOST
```

You choose the path

```
usr
```

```
bin
```

```
lib
```

```
share
```

```
qu江湖
```

```
system
```

```
ntoarm, ntonmips, ntoppc, ntonsh, ntox86
```

```
bin
```

- **Host-specific development tools for target platforms**

---

*The host-related directory structure.*
The QNX_TARGET environment variable identifies the directory that holds the target-related components:

```
The target-related directory structure.
```

Neutrino also uses these environment variables to locate files on the host machine:

**QNX_CONFIGURATION**

The location of the configuration files and licenses for QNX Momentics.

**MAKEFLAGS**

The location of included *.mk files.
What did I just install?

Here’s where some of the key components are installed:

<table>
<thead>
<tr>
<th>Component</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Buildfiles</td>
<td>$QNX_TARGET/platform/boot/build/boardname.build</td>
</tr>
<tr>
<td>Command-line utilities</td>
<td>For the host: $QNX_HOST/usr/bin and $QNX_HOST/platform/bin</td>
</tr>
<tr>
<td></td>
<td>For the target: $QNX_TARGET/bin and $QNX_TARGET/platform/bin and $QNX_TARGET/platform/sbin</td>
</tr>
<tr>
<td>Device drivers (binaries)</td>
<td>$QNX_TARGET/platform/sbin</td>
</tr>
<tr>
<td>Device drivers (DLLs)</td>
<td>$QNX_TARGET/platform/lib/dll</td>
</tr>
<tr>
<td>Filesystems</td>
<td>$QNX_TARGET/platform/sbin</td>
</tr>
<tr>
<td>GUI-related</td>
<td>$QNX_TARGET/usr/photon</td>
</tr>
<tr>
<td>Shared libraries</td>
<td>$QNX_TARGET/platform/lib</td>
</tr>
<tr>
<td>System header files</td>
<td>$QNX_TARGET/usr/include</td>
</tr>
<tr>
<td>Documentation</td>
<td>Eclipse plugin directory, also in $QNX_TARGET/usr/help/product on self-hosted Neutrino systems</td>
</tr>
</tbody>
</table>

If you install any BSPs or DDKs, they’re installed under $QNX_TARGET, or in a directory of your choosing. The IDE creates a workspace, a directory that holds your development projects. By default, this workspace is in your home directory on Linux and Neutrino, and in C:\QNX632 on Windows.

For information about the directory structure on a Neutrino runtime system, see “Where everything is stored” in the Working with Files chapter of the Neutrino User’s Guide.
What’s on my desktop?

How you access the components of QNX Momentics depends on your host machine:

Windows  You can start the IDE by clicking its icon on the desktop:

or by choosing Start→QNX Momentics IDE. The Start→All Programs→QNX Momentics 6.3.2 menu lets you add or activate licenses, configure your machine to build for a specific version of QNX Momentics, run Phindows, and start the IDE.

Linux  The Start→Programming menu lets you add or activate licenses, and start the IDE.

Neutrino  The Launch→QNX Momentics 6.3.2 menu lets you start the IDE; the Launch→Configure menu lets you add and activate licenses. The Help item in the Launch menu and on the shelf starts the Helpviewer, where you’ll find the documentation.

Upgrading your software

Here’s the general procedure for updating a version of QNX Momentics or other components you’ve purchased:

1  Go to the QNX Software Systems website (http://www.qnx.com) and log into your myQNX account. If you don’t already have a myQNX account, please register now.

2  Follow the instructions for registering your product. You’ll need the Product Registration serial number and password from your License Certificate.
Running QNX Neutrino self-hosted

3 Go to the Download area.
4 Select the product (e.g. “QNX Momentics 6.3.2”) or search by keywords.

In the next step, you’ll download a file. Don’t download it into a directory whose path contains spaces. For example, don’t download the file into

C:\Documents and Settings\my_userid\Desktop.

5 Download the appropriate file and follow the instructions.

If you installed an evaluation copy of QNX Momentics, you can upgrade to a permanent copy without reinstalling. For more information, see our website.

Managing source code

You’ll probably want to use some sort of version-control system to manage and track changes to the software that you develop.

<table>
<thead>
<tr>
<th>For information about:</th>
<th>See:</th>
</tr>
</thead>
</table>

QNX Momentics includes clients for both CVS and Subversion.

Running QNX Neutrino self-hosted

You can develop software on a self-hosted QNX Neutrino system. For more information on working with Neutrino, see the Neutrino User’s
Mixing a self-hosted machine with other hosts

If you have a Neutrino host, you can communicate with other hosts in various ways:

- You can access resources — such as files, directories, and processes — on other Neutrino machines as if the resources were on your own computer; see Using Qnet for Transparent Distributed Processing in the Neutrino User’s Guide.

- You can use TCP/IP; see TCP/IP Networking in the Neutrino User’s Guide.

- You can mount DOS and Linux filesystems right on your Neutrino box, or use CIFS or NFS to mount filesystems across a network; see Working with Filesystems in the Neutrino User’s Guide.

Can different versions of QNX Momentics coexist?

You can install both QNX Momentics 6.3.x and 6.2.1 on your machine. You can use the IDE or the qconfig utility to set up your environment to build for the correct version. On Windows hosts, use QWinCfg, a graphical front end for qconfig that you can launch from the Start menu.

Coexistence of 6.3.x and 6.2.1 is supported only on Windows hosts.

For more information, see the Compiling and Debugging chapter of the Neutrino Programmer’s Guide and the IDE Concepts chapter of the IDE User’s Guide.
Running QNX Neutrino on a target machine

Running QNX Neutrino on a target machine

Neutrino is well suited to embedded systems. For information about creating OS images, downloading them to your target hardware, and running your software, see:

- the BSP documentation for your particular target (in the IDE’s help system, or in the Photon Helpviewer on Neutrino)
- *Building Embedded Systems*
- *IDE User’s Guide*

You don’t always need to have the hardware to run your software; for more information, see “Getting started *before* you have your target” in the What is QNX Momentics? chapter in this guide.
Chapter 3
How to Get Help

In this chapter…

Overview of the documentation 21
Viewing the documentation 25
What should I read first? 27
Where key features are documented 28
Related reading 37
Getting quick help with the use command 39
Hover help in the IDE 39
Visit http://www.qnx.com 39
Support plans 40
Training 40
Custom engineering and consulting 40
The first place to look for help is in the QNX Momentics documentation, but if you still have problems, there are several other avenues of help.

Overview of the documentation

In QNX Momentics, the online documents are in HTML, which you can access in the IDE’s help system. On self-hosted QNX Neutrino systems, you can also look at the documentation in the Photon helpviewer.

For the latest documentation, or to download PDF versions, visit our website, http://www.qnx.com. Printed books are also available.

To help you find your way around the QNX Momentics documentation set, we’ve provided a documentation roadmap.

While the roadmap works best online because of all its links, here’s a hardcopy version that can still help you know where to look for information.

The complete QNX Momentics documentation set contains the following books, arranged here under each main component:

**QNX Momentics Development Suite**

*Quickstart Guide: 10 Steps to Your First QNX Program*

A tutorial that helps you install QNX Momentics on a host machine, install the QNX Neutrino RTOS on a target machine, set up communications between the two systems, and then use the IDE to develop a program on the host machine and run it on the target.

**Integrated Development Environment**

*IDE User’s Guide*

Describes the QNX Momentics Integrated Development Environment, how to set up and start using the tools to build Neutrino-based target systems, etc.
Overview of the documentation

QNX Neutrino Realtime Operating System

System Architecture
Describes the concepts and architecture of the QNX Neutrino microkernel, resource managers, processes, threads, message-passing services, and more.

QNX Neutrino User’s Guide
Explains how to interact with a running Neutrino system. Covers both Photon and text-mode interfaces, as well as various system-administration topics.

QNX Neutrino Programmer’s Guide
Tells you how to get started writing programs, including interrupt handlers, resource managers, etc.

Building Embedded Systems
Tells you how to get the OS running on your target embedded system, write an IPL, customize a startup program, etc.

Utilities Reference
Describes the Neutrino configuration files, utilities, and manager processes you’ll use during development and at runtime. See the Utilities Summary chapter in the reference for a listing of all the entries by category.

QNX Neutrino Library Reference
Describes the C library data types and functions, including POSIX threads, kernel calls, resource manager functions, etc. See the Summary of Functions chapter in the reference for a listing of all the functions by category.

Technical Notes
Deals with a series of topics (e.g. IP tunneling) that aren’t covered in the basic documentation set.
Overview of the documentation

Photon microGUI

Photon Programmer’s Guide

Gives you a hands-on tour of the Photon Application Builder (PhAB). You’ll learn how to quickly assemble a GUI from predefined widgets, link the GUI to an application, and generate C source to bring the GUI to life.

Multimedia Developer’s Guide

Describes the multimedia architecture for developers who wish to write multimedia plugins.

Photon Library Reference

Provides concise descriptions of Photon’s and PhAB’s global data structures and functions.

Widget Reference

Contains guidelines for programming widgets, along with concise descriptions of all global data structures, resources, and convenience functions associated with widgets. It also gives you practical examples of how to use Photon widgets and widget functions.

Building Custom Widgets

Explains how to create a custom widget and how to bind it into PhAB. If you need a widget whose features extend the standard capabilities of the Photon widget library, this guide is for you.

BSPs, DDKs, and specialty tools

BSP guides

Describe how to get Neutrino running on your target board. You’ll find a separate guide for each BSP that you’ve installed. We support boards in these processor families: ARM/XScale, MIPS, PowerPC, SH-4, and x86. You can download BSPs from our website, http://www.qnx.com/. The Download area arranges the BSP by platform.
Overview of the documentation

DDK guides  
Describe how to write drivers for QNX Neutrino. You’ll find a separate DDK guide for audio, character, graphics, input, network, and Universal Serial Bus (USB) devices. You can download the DDKs from our website, but QNX Momentics includes the documentation.

System Analysis Toolkit User’s Guide
Describes how to use the SAT with our instrumented microkernel. You can log every communication and state change within the microkernel, including interrupts, all parameters/return values from kernel calls, and scheduling decisions, resulting in a deeper and more detailed analysis of system elements. You can even perform kernel-level diagnostics remotely.

Photon Multilingual Input
Tells you how to input Chinese, Japanese, and Korean characters in Photon.

Phindows Connectivity
Tells you how to access Photon from a Windows machine.

Dinkum C and C++

Dinkum C++ Library
A conforming implementation of the Standard C++ library.

Dinkum C99 Library
A conforming implementation of the Standard C library, as revised in 1999.

Dinkum EC++ Library
A conforming implementation of the Embedded C++ library as specified by the Embedded C++ Technical Committee.
Viewing the documentation

Within the IDE

Click Help→Help Contents. There you’ll find several booksets listed, including *A Roadmap to the QNX Momentics Development Suite*. The other documents listed, such as the Workbench User Guide and *JDT Plug-in Developer Guide*, pertain to the Eclipse platform and its various plugins.

Getting help in the IDE.

Note that the IDE’s internal help system has a builtin search facility. For details, see “Using the QNX Help system” in the IDE Concepts chapter of the IDE User’s Guide.

The Photon Helpviewer

If you’re using Neutrino self-hosted, you can also access the documentation via our native Helpviewer.
Viewing the documentation

Photon Helpviewer.

To open the Helpviewer, click the Help button in the Applications group on the shelf or select Help from the right-click menu on the desktop.

For more information, see “Getting help with the Helpviewer” in the Using the Photon microGUI chapter of the Neutrino User’s Guide, as well as the entry for helpviewer in the Utilities Reference.

Keyword indexes

Nearly every book in the QNX Momentics documentation set has its own keyword index. At the top and bottom of the online documents, you’ll find a link to the keyword index file (keywords-all.html).
What should I read first?

Many people simply don’t read manuals cover to cover. They often browse or search the documentation for a specific topic, usually for information on how to do a certain task. But if you want to approach your tasks with enough knowledge to work effectively, it’s a good idea to start with the System Architecture guide; it will help you understand Neutrino’s unique features, particularly its message-based interprocess communication (IPC) and microkernel architecture.

Once you know how Neutrino works, you’ll then want to know how to work with it; our Quickstart Guide: 10 Steps to Your First QNX Program is a short tutorial that will get you started in a matter of minutes. Which document you’ll need next depends on when you need it.

During development (on your host)

Most of the documents in the bookset are geared towards developing your Neutrino-based applications. Of these development books, some are how-to guides, and some are reference works.

Some books are useful during development as well as at runtime. For instance, the Utilities Reference is a comprehensive document that includes descriptions of both development utilities (e.g. make), which you wouldn’t normally use on your target, as well as runtime programs (e.g. devc-sersci), which you would run only on your target.

Here are the main how-to guides and their corresponding reference books:
Where key features are documented

<table>
<thead>
<tr>
<th>How-to guide:</th>
<th>Companion reference:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDE User's Guide</td>
<td>Utilities Reference (for underlying command-line utilities, such as <code>make</code>); QNX Neutrino Library Reference for API</td>
</tr>
<tr>
<td>QNX Neutrino User's Guide</td>
<td>Utilities Reference</td>
</tr>
<tr>
<td>QNX Neutrino Programmer's Guide</td>
<td>QNX Neutrino Library Reference</td>
</tr>
<tr>
<td>Photon Programmer's Guide</td>
<td>Photon Library Reference; Photon Widget Reference</td>
</tr>
<tr>
<td>Building Embedded Systems</td>
<td>Utilities Reference; QNX Neutrino Library Reference</td>
</tr>
</tbody>
</table>

If you’ve already developed applications with QNX Momentics 6.2.1, see the 6.3 and 6.2.1 Compatibility notes. You can find the latest version of this document on our website, [http://www.qnx.com/](http://www.qnx.com/).

At runtime (on your target)

The most runtime-oriented document in your bookset is the QNX Neutrino User's Guide, which describes how to use and interact with a running Neutrino system. The book covers both Photon and text-mode interfaces, as well as various system-administration topics.

Other runtime-oriented documents include the Utilities Reference and the IDE User's Guide (for information on diagnostic tools you’d use on a running system).

Where key features are documented

The following list may help you learn which document to look in when you want information on certain key features or components of the OS and tools.
See the *System Architecture* guide for information on almost every topic in this list.

Adaptive partitioning

Adaptive Partitioning *User’s Guide*

Asymmetric multiprocessing (AMP)

Multicore Processing in the *System Architecture* guide

Audio support

*Audio Developer’s Guide* (for writing audio applications); *Audio DDK* (for writing audio device drivers); `io-audio` manager and the `deva-*` drivers in the *Utilities Reference*.

Bound multiprocessing (BMP)

Multicore Processing in the *System Architecture* guide; the Multicore Processing *User’s Guide*; `procnto*` in the *Utilities Reference*.

BSPs

Each BSP that you install includes documentation; see the BSP’s installation and release notes.

Buildfiles

Making an OS Image in *Building Embedded Systems*; `mkifs` in the *Utilities Reference*.

CD-ROM

Working with Filesystems in the QNX Neutrino *User’s Guide*; `cam-cdrom.so` in the *Utilities Reference*.

Channels

`Channel*`, `Msg*`, and `Connect*` functions in the QNX Neutrino *Library Reference*; Writing a Resource Manager in the QNX Neutrino *Programmer’s Guide*.

Compiling

Developing C/C++ Programs in the IDE *User’s Guide*; Compiling and Debugging in the QNX
Where key features are documented

Neutrino *Programmer’s Guide*; **make** and **qcc** in the *Utilities Reference*.

Compression  Making an OS Image in *Building Embedded Systems*; **deflate** in the *Utilities Reference*.

CPU time, sharing among competing processes  Adaptive Partitioning *User’s Guide*

Data server  Setting Up an Embedded Web Server in the QNX Neutrino *User’s Guide*; **ds** in the *Utilities Reference*.

DDKs  Driver Development Kits (DDKs). QNX Momentics includes the documentation, but if you want the DDKs, you must download them from our website.

Debugger  Debugging Programs in the IDE *User’s Guide*; Compiling and Debugging in the QNX Neutrino *Programmer’s Guide*; **gdb** and other debugging utilities in the *Utilities Reference*.

Device management

  **devctl()** in the QNX Neutrino *Library Reference*; Writing a Resource Manager in the QNX Neutrino *Programmer’s Guide*; **dev* drivers in the *Utilities Reference*.


Embedded systems

  *Building Embedded Systems*.

Endian issues  Freedom from Hardware and Platform Dependencies in the QNX Neutrino *Programmer’s Guide*.
Where key features are documented

Environment, configuring


Environment variables

Commonly Used Environment Variables appendix in the Utilities Reference; environ() in the QNX Neutrino Library Reference; Using the Photon microGUI and Configuring Your Environment chapters in the QNX Neutrino User’s Guide.

Event handling (OS)

MsgDeliverEvent() in the QNX Neutrino Library Reference; Writing an Interrupt Handler in the QNX Neutrino Programmer’s Guide.

Event handling (Photon)


File handling

creat() and related functions via its “See also” section in the QNX Neutrino Library Reference; basename, cat, etc. in the Utilities Reference (see “File and directory manipulation” in the Utilities Summary).

Filesystems

Working with Filesystems in the QNX Neutrino User’s Guide.

Fonts


Graphics

Graphics DDK; devg-* drivers in the Utilities Reference.

Hardware

Connecting Hardware in the QNX Neutrino User’s Guide.
Where key features are documented

HID (human input devices)

hidview and devh-* drivers in the Utilities Reference.

Helpviewer (Photon)

Using the Photon microGUI in the QNX Neutrino User’s Guide; helpviewer in the Utilities Reference.

I/O management iofunc*, Interrupt*, and open() and related functions via open()’s “See also” section in the QNX Neutrino Library Reference; Writing a Resource Manager in the QNX Neutrino Programmer’s Guide.

IPC (interprocess communication)

pthread_mutex*, SyncMutex*, sem_*, SyncSem*, pthread_cond*, and SyncCondvar* in the QNX Neutrino Library Reference. See also “Message Passing,” below.

IPL (Initial Program Loader)

Writing an IPL Program in Building Embedded Systems.

Images (OS)

Making an OS Image in Building Embedded Systems; mkifs in the Utilities Reference.

Images (graphical)


Input

Input DDK; devh-* and devi-* drivers in the Utilities Reference.

Instrumented kernel

System Analysis Toolkit User’s Guide.
Where key features are documented

Interrupt handling

*InterruptAttach() and related functions in its “See also” section in the QNX Neutrino Library Reference.*

Kernel calls

*Channel*, *Clock*, *Connect*, *Debug*, *Interrupt*, *Msg*, *Sched*, *Signal*, *Sync*, *Thread*, *Timer*, and *TraceEvent() in the QNX Neutrino Library Reference.*

Keyboard support

Using the Command Line in the QNX Neutrino User’s Guide; *mkkbd* and related utilities in its “See also” section in the Utilities Reference.

Libraries

Compiling and Debugging in the QNX Neutrino Programmer’s Guide; Building OS and Flash Images in the IDE User’s Guide.

Linker

Compiling and Debugging in the QNX Neutrino Programmer’s Guide; *qcc* and related utilities in its “See also” section in the Utilities Reference.

Makefile structure

Developing C/C++ Programs in the IDE User’s Guide; Conventions for Makefiles and Directories in the QNX Neutrino Programmer’s Guide.

Memory management


Message passing

*Msg*, *Connect*, and *Channel* in QNX Neutrino Library Reference. See also “IPC,” above.
Where key features are documented

Message queues  *mq_* functions in the QNX Neutrino *Library Reference*; *mqueue* and *mq* in the *Utilities Reference*.

Multicore systems

Multicore Processing in the *System Architecture* guide; the Multicore Processing *User’s Guide*; *procnto* in the *Utilities Reference*.

Native networking (Qnet)

See “Transparent distributed processing,” below.

Network drivers  *Network DDK*; *devn-* drivers in the *Utilities Reference*.

Package filesystem

Working with Filesystems in the QNX Neutrino *User’s Guide*; *fs-pkg* in the *Utilities Reference*.

Permissions (on files, directories)

Managing User Accounts and Working with Files chapters in the QNX Neutrino *User’s Guide*; *chmod()**, stat()**, and *umask()* in the QNX Neutrino *Library Reference*; *chmod* in the *Utilities Reference*.

Process manager  *procnto* in *Utilities Reference*; *procmgr_* in the QNX Neutrino *Library Reference*.


Profiler (application)

Profiling an Application in the IDE *User’s Guide*.

Profiler (system)  Analyzing Your System with Kernel Tracing in the IDE *User’s Guide*.
Where key features are documented

**Pulses**  

**QoS (Quality of Service)**  
“Quality of Service and multiple paths” in the Native Networking (Qnet) chapter in *System Architecture*; `netmgr_ndtostr()` in the QNX Neutrino *Library Reference*.

**RAM disk**  
`io-blk.so` (using the `ramdisk=size` option), `devb-ram`, and `devf-ram` in the *Utilities Reference*.

**Realtime scheduling**  
`Sched*` functions and `sched_param` structure in the QNX Neutrino *Library Reference*.

**Resource management**  
Writing a Resource Manager in the QNX Neutrino *Programmer’s Guide*; `resmgr_*` and `iofunc_*` in the QNX Neutrino *Library Reference*.

**Resources, sharing among competing processes**  
Adaptive Partitioning *User’s Guide*.

**Symmetric multiprocessor (SMP)**  
Multicore Processing in the *System Architecture* guide; the Multicore Processing *User’s Guide*; `procnto*` in the *Utilities Reference*.

**Self-hosted development**  
Compiling and Debugging in the QNX Neutrino *Programmer’s Guide*.

**Signal handling**  
`sigaction()` and related functions via its “See also” section in the QNX Neutrino *Library Reference*. 
Where key features are documented

Startup programs

Customizing Image Startup Programs in Building Embedded Systems; startup-* in the Utilities Reference.

Synchronization

Sync* functions in the QNX Neutrino Library Reference.

System information

pidin and sin in the Utilities Reference;
Analyzing Your System with Kernel Tracing in the IDE User’s Guide; Fine-Tuning Your System in the QNX Neutrino User’s Guide.

Target agent (qconn)

qconn in the Utilities Reference; IDE Concepts in the IDE User’s Guide.

Threads

pthread_ * and Thread* in the QNX Neutrino Library Reference; Programming Overview in the QNX Neutrino Programmer’s Guide.

Time

Configuring Your Environment in the QNX Neutrino User’s Guide; Clock*, time(), and related functions via their “See also” sections in the QNX Neutrino Library Reference.

Timestamps (for files)

touch in the Utilities Reference; utime() and related functions via its “See also” section in the QNX Neutrino Library Reference.

Timers

alarm(), Timer*, and timer_ * in the QNX Neutrino Library Reference.

Transparent distributed processing

Using Qnet for Transparent Distributed Processing in the QNX Neutrino User’s Guide; Transparent Distributed Processing via Qnet in the QNX
Related reading

Neutrino *Programmer’s Guide*; *npm-qnet.so* in the *Utilities Reference*; *netmgr_* *in* the QNX Neutrino *Library Reference*.

**USB**
Connecting Hardware in the QNX Neutrino *User’s Guide*; *devu-* *, devh-usb.so, devi-hid, and io-usb* in the *Utilities Reference*; USB DDK.

**Unicode**

**Version control**

**Web server (slinger)**
Setting Up an Embedded Web Server in the QNX Neutrino *User’s Guide*; *slinger* in the *Utilities Reference*.

**XIP (execute in place)**
Building OS and Flash Images in the IDE *User’s Guide*; Writing an IPL Program in *Building Embedded Systems*.

**Related reading**

**On QNX Neutrino**


  You can get this book from QNX Software Systems; the online documentation for QNX Momentics includes some sample chapters.
Related reading


You can get this book from the publisher; for more information, see http://www.parse.com.

On POSIX

The latest POSIX standards documents are available online here:

http://www.opengroup.org/onlinepubs/007904975/nframe.html

For an up-to-date status of the many POSIX drafts/standards documents, see the PASC (Portable Applications Standards Committee of the IEEE Computer Society) report at http://pasc.opengroup.org/standing/sd11.html.

In addition to the POSIX standards themselves, you might find the following books useful:


On TCP/IP

Getting quick help with the use command


Some of the advanced API features mentioned in these TCP/IP books might not be supported.

Getting quick help with the use command

Similar to the UNIX man command, Neutrino presents a simple usage message for each command-line utility. At the OS system prompt, type:

```
use utility_name
```

and you’ll see a brief description as well as the command-line options for that utility.

Hover help in the IDE

When working on your projects in the IDE’s editor, whenever you hover your mouse pointer over a function name in a line of code, you’ll see a popup text box containing that function’s purpose and synopsis (provided that it’s a documented Neutrino function).

For details, see the chapter on C/C++ development in the IDE User’s Guide.

Visit [http://www.qnx.com](http://www.qnx.com)

The QNX Software Systems website ([http://www.qnx.com](http://www.qnx.com)) offers help on using QNX Momentics through such facilities as:

- detailed lists of supported hardware
- Community area, which has many technical articles and other resources
Support plans

- Licensing area, which gives details on product licensing
- Partners area, which includes lists of distributors and resellers around the world
- technical forums — members of the Neutrino user community at large as well as our own developers and staff frequent these forums daily
- and more

For general help with Eclipse, the open platform for our IDE, visit the Eclipse consortium website (http://www.eclipse.org). There you’ll find valuable support in the form of newsgroups, mailing lists, articles, and more.

Support plans

You can access a wide range of support resources, depending on the particular support plan that you purchase.

For more information about our technical support offerings, including email addresses and telephone numbers, please see the Support + Services section of our website (http://www.qnx.com).

Training

QNX training services offers many hands-on courses at your choice of location (QNX headquarters, various training centers around the world, or your site). For details, including the current training schedule, see the Training area in the Support + Services section of our website (http://www.qnx.com).

Custom engineering and consulting

Depending on the nature of your particular project, you may choose to engage us to help in areas such as:

- custom BSPs
Custom engineering and consulting

- device driver development
- hardware troubleshooting and integration
- application development
- migration/porting services

For more information, see contact your local sales representative.
Aviage

Product name for middleware developed by QNX Software Systems.

BSP

Board Support Package — a set of software components (IPL, startup code, drivers, buildfiles, as well as the QNX Neutrino RTOS) and documentation intended to help you get Neutrino up and running on a particular board.

buildfile

A “control” file that specifies the particular startup program, environment variables, drivers, host-target communications, etc. that will be used to generate an OS image. The file provides instructions to a utility program such as mkifs, which generates OS images. In the context of the IDE, the buildfile is sometimes called a target file.

Development License

Development using QNX Software Systems products is governed by the terms of the applicable developer license agreement to which you must agree to install the software product. For more information, see the Licensing area of our website, http://www.qnx.com/.

Distribution License

Distribution of QNX Software Systems products is governed by the terms of the QNX Runtime License Agreement or the QNX OEM License Agreement. For more information, see the Licensing area of our website, http://www.qnx.com/.

DDK

Driver Development Kit — a set of source code and documentation intended to help you write your own drivers for various devices: audio, graphics, input (mice, keyboards, etc.), network, and USB.
**Eclipse**

Name of a tools project and platform developed by an open consortium of vendors (Eclipse.org), including QNX Software Systems.

The IDE within the QNX Momentics suite consists of a set of special plugins integrated into the standard Eclipse framework.

**EULA**

End User License Agreement.

**IDE**

Integrated Development Environment — in QNX Momentics, the IDE is based on the Eclipse framework, and includes many highly integrated tools for code development and system analysis.

IDE also stands for Integrated Drive Electronics, a standard interface used between a computer’s bus and disk storage devices.

**IPL**

Initial Program Loader — the software component responsible for setting up the machine into a usable state, such that the startup code can then perform further initializations.

**Momentics**

Product name for the development suite created by QNX Software Systems for its Neutrino RTOS.

**Neutrino**

Product name of the RTOS created by QNX Software Systems.

**OCL**

Open Community License.
QCL

QNX Community License.

QNX

Name of an earlier-generation RTOS created by QNX Software Systems. Also, short form of the company’s name.

QoS

Quality of Service — a policy (e.g. loadbalance) used to connect nodes in a network in order to ensure highly dependable transmission. QoS is an issue that often arises in high-availability (HA) networks as well as realtime control systems.

QSS

QNX Software Systems.

RTOS

Realtime operating system.

Runtime License

See the Licensing area of our website, http://www.qnx.com/.

startup code

The software component that gains control after the IPL code has performed the minimum necessary amount of initialization. After gathering information about the system, the startup code transfers control to the OS.

TDK

Technology Development Kit — a kit that augments the base Neutrino OS platform in QNX Momentics with specialized, value-added technology.
Index

A

account center (myQNX) 15, 21
Acoustic Processing TDK 4
Advanced Graphics TDK 4
AMP (asymmetric multiprocessing) 29
architecture, system 22
audio 29

B

BMP (bound multiprocessing) 29
BSPs (Board Support Packages) 4, 6, 14, 23
buildfiles 14, 29
Building Custom Widgets 23
Building Embedded Systems 22
Butenhof, David R. 38

C

C functions 22, 24
C++ 24
CD-ROMs 29
channels 29
Chinese characters 24
CIFS (Common Interface File System) 17
coexistence of multiple versions of QNX Momentics 17
Collins-Sussman, Ben 16
custom engineering 40
documentation 22
usage messages for 39
compiling 30
customization 30
configuration files 13
documentation 22
conventions
typographical xiii
custom engineering 40
CVS (Concurrent Versions System) 16
Index

D

data servers 30
data types 22
DDKs (Driver Development Kits) 4, 14, 24
debugger 30
desktop 15
device drivers 14, 30
Dinkum 24
distributed processing 17, 37
distributors 40
documentation
  HTML 21
  knowing which to read first 27
  overview of 21
  PDF 21
  printed 21
  roadmap 21
  third-party 37
  via the Photon Helpviewer 26
  which to read for key features of the OS 28
  within the IDE 25
DOS filesystems 17
Driver Development Kits (DDKs) 4, 14, 24
environment variables 31
  MAKEFLAGS 13
  QNX_CONFIGURATION 13
  QNX_HOST 12
  QNX_TARGET 13
environment, configuring 31
evaluation copy, upgrading 16
event handling 31
execute in place (XIP) 37

F

Fenner, B. 38
files
  configuration 13, 22
  handling 31
  timestamps 36
filesystems 14, 31
  mounting DOS and Linux 17
Fitzpatrick, Brian W. 16
fonts 31
forums 40
functions, C 22

G

Gallmeister, Bill O. 38
Getting Started with QNX Neutrino 2 37
graphics 31, 32
GUI 14, See also Photon developing 23
H

hardware
  connecting 31
  supported 39
header files 14
help
  via the Photon Helpviewer 26
  within the IDE 25
Helpviewer (Photon) 26, 32
HID (human input devices) 32
hosts
  choice of 3
  development on xi
  host-related components 12
  supported 11
hover help 39
how-to guides 27
Hunt, Craig 38

I

I/O management 32
IDE (Integrated Development Environment) 3
  Help system 25
  starting 15
  User’s Guide 21
  workspace directory 14
images (graphical) 32
images (OS) 32
index in online documentation 26
installation, general information
  about 11
InstallShield 11
instrumented kernel 24, 32
international characters 24, 37
interrupt handling 33
IPC (interprocess communication) 32
IPL (initial program loader) 32

J

Japanese characters 24

K

kernel
  calls 33
  instrumented 24, 32
keyboards 33
Korean characters 24
Krten, Robert 37

L

libraries 4, 14, 33
Library Reference
  Photon 23
  QNX Neutrino 22
licenses 7, 13
linker 33
Linux 11
  filesystems 17
Index

M

Makefile structure 33
MAKEFLAGS 13
man command 39
memory management 33
menus 15
message passing 33
message queues 34
mkifs 4
multicore systems 34
multilingual input 24
Multimedia Developer’s Guide 23
Multimedia TDK 5
myQNX account center 15, 21

N

network drivers 34
networking, transparent (native) 17, 37
NFS (Network File System) 17

P

packages 34
pathname delimiter in QNX
   Momentics documentation xiv
PDF documents 21
permissions 34
Phindows 24
Photon
   Building Custom Widgets 23
   Helpviewer 26
Library Reference 23
multilingual input 24
Multimedia Developer’s Guide 23
Programmer’s Guide 23
Widget Reference 23
Windows, accessing from 24
Pilato, C. Michael 16
platform-specific issues 7
POSIX
documents available online 38
link to site for up-to-date
status 38
POSIX.4: Programming for the Real
World 38
printed books 21
process manager 34
processes 34
professional services 40
profiler 34
Programmer’s Guide
   Photon 23
   QNX Neutrino 22
Programming with POSIX
   Threads 38
pulses 35

Q

qcc 4
qconfig 17
qconn 36
Qnet 17, 37
QNX_CONFIGURATION 13
Index

QNX_HOST 12
QNX_TARGET 13
QNX Cookbook: Recipes for Programmers 38
QNX Momentics
about xi
coeexistence of multiple versions 17
End User License Agreement (EULA) 7
evaluation copy, upgrading 16
features of 3
parts of 3
system requirements 11
upgrading 15
QNX Neutrino
Building Embedded Systems 22
getting started with 21
Library Reference 22
Programmer’s Guide 22
self-hosted 16
System Architecture 22
Technical Notes 22
User’s Guide 22, 28
Utilities Reference 22
QNX Software Systems website 39
QoS (Quality of Service) 35
Quickstart Guide: 10 Steps to Your First QNX Program 21
QWinCfg 17

R

RAM disk 35

realtime scheduling 35
reference books 27
resellers 40
resource management 35
Rudoff, A. 38
runtime
directory structure 14
documentation 28
kit 8
license 8

S

scheduling 35
self-hosted development 16, 35
documentation 25
signal handling 35
slinger 37
SMP (symmetric multiprocessing) 35
software
developing 22
managing 16, 37
updates 15
source code 3
startup programs 36
Stevens, W. R. 38
support plans 40
synchronization 36
system
administration 28
header files 14
information 36
requirements 11
System Analysis Toolkit 24
Index

System Architecture 22

T

targets
  agent (qconn) 36
  developing for 3, 18, 22
  running on xi, 5
  target-related components 12
TCP/IP 17
TCP/IP Illustrated, Volume 1: The Protocols 39
TCP/IP Illustrated, Volume 2: The Implementation 39
TCP/IP Network Administration 38
TDKs (Technology Development Kits) 4
Technical Notes 22
technical support 40
  public forums 40
Technology Development Kits (TDKs) 4
threads 36
time 36	
timers 36
timestamps 36
tools 3
training 40
transparent distributed processing 17, 37
tutorials 21
typographical conventions xiii

U

Unicode 37
UNIX Network Programming, Volume 1: The Sockets Networking API 38
UNIX Network Programming, Volume 2: Interprocess Communications 38
USB (Universal Serial Bus) 37
use command 39
User’s Guide
  IDE 21
  QNX Neutrino 22, 28
  utilities 14, 27
Utilities Reference 22

V

version control 37
Version Control with Subversion 16
Virtio 6
virtual machine 5
Virtual PC 5
VMware 5

W

Web Browser TDK 5
web server (slinger) 37
websites
  Eclipse consortium 40
  QNX Software Systems 39
Widget Reference 23
Index

Windows  11
    accessing Photon from  24
workspace directory  14
Workstation  5

X

x86 Runtime Kit  8
XIP (execute in place)  37