Quick Recipes on Symbian OS

Mastering C++ Smartphone Development

Michael Aubert

With
Alexey Gusev, Tanzim Husain, Jenny Mulholland, Antony Pranata, Jukka Silvennoinen, Jo Stichbury

Reviewed by
Sandip Ahluwalia, Ashlee Godwin, Douglas Feather, Maximiliano R. Firtman, Matthew O'Donnell, Tong Ren, Attila Vamos, Jacek Wojciechowski, Colin Ward, Hamish Willee

Head of Symbian Press
Freddie Gjertsen

Managing Editor
Satu McNabb

John Wiley & Sons, Ltd

Quick Recipes on Symbian OS

Mastering C++ Smartphone Development

Quick Recipes on Symbian OS

Mastering C++ Smartphone Development

Michael Aubert

With
Alexey Gusev, Tanzim Husain, Jenny Mulholland, Antony Pranata, Jukka Silvennoinen, Jo Stichbury

Reviewed by
Sandip Ahluwalia, Ashlee Godwin, Douglas Feather, Maximiliano R. Firtman, Matthew O'Donnell, Tong Ren, Attila Vamos, Jacek Wojciechowski, Colin Ward, Hamish Willee

Head of Symbian Press
Freddie Gjertsen

Managing Editor
Satu McNabb

John Wiley & Sons, Ltd

John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester,
West Sussex PO19 8SQ, England

Telephone (+44) 1243 779777

Email (for orders and customer service enquiries): cs-books@wiley.co.uk
Visit our Home Page on www.wileyeurope.com or www.wiley.com

Other Wiley Editorial Offices

John Wiley & Sons Inc., 111 River Street, Hoboken, NJ 07030, USA

Jossey-Bass, 989 Market Street, San Francisco, CA 94103-1741, USA

Wiley-VCH Verlag GmbH, Boschstr. 12, D-69469 Weinheim, Germany

John Wiley & Sons Australia Ltd, 42 McDougall Street, Milton, Queensland 4064, Australia

John Wiley & Sons (Asia) Pte Ltd, 2 Clementi Loop #02-01, Jin Xing Distripark, Singapore 129809

John Wiley & Sons Canada Ltd, 6045 Freemont Blvd, Mississauga, Ontario, L5R 4J3, Canada

Wiley also publishes its books in a variety of electronic formats. Some content that appears
in print may not be available in electronic books.

Library of Congress Cataloging-in-Publication Data

Aubert, Michael.
Quick recipes on Symbian OS : mastering C++ smartphone development /
Michael Aubert, with Alexey Gusev . . . [et al.]
p. cm.
Includes bibliographical references and index.
ISBN 978-0-470-99783-3 (pbk. : alk. paper)
1. Smartphones – Programming. 2. Symbian OS (Computer file) 3. C++
(Computer program language) I. Gusev, Alexey. II. Title.
TK6570.M6A92 2008
005.26'8 – dc22

2008013156

British Library Cataloguing in Publication Data

A catalogue record for this book is available from the British Library

ISBN: 978-0-470-99783-3

Typeset in 10/12pt Optima by Laserwords Private Limited, Chennai, India
Printed and bound in Great Britain by Bell & Bain, Glasgow
This book is printed on acid-free paper responsibly manufactured from sustainable forestry
in which at least two trees are planted for each one used for paper production.

Pour Marcel et Paulette

Contents

List of Recipes

Foreword

David Wood, Executive Vice President Research, Symbian

This book has been designed for people who are in a hurry.

Perhaps you are a developer who has been asked to port some software, initially written for another operating system (such as may run on a desktop computer), to Symbian OS. Or perhaps you have to investigate whether Symbian OS could be suited to an idea from a designer friend of yours. But the trouble is, you don't have much time, and you have heard that Symbian OS is a sophisticated and rich software system with a not insignificant learning curve.

If you are like the majority of software engineers, you would like to take some time to investigate this kind of task. You might prefer to attend a training course, or work your way through some of the comprehensive reference material that already exists for Symbian OS. However, I guess that you don't have the luxury of doing that – because you are facing tight schedule pressures. There isn't sufficient slack in your schedule to research options as widely as you'd like. Your manager is expecting your report by the end of the week. So you need answers in a hurry.

That's why Symbian Press commissioned the book you are now holding in your hands. We are assuming that you are a bright, savvy, experienced software developer, who is already familiar with C++ and with modern software programming methods and idioms. You are willing to work hard and can learn fast. You are ready to take things on trust for a while, provided you can quickly find out how to perform various tasks within Symbian OS. Over time, you would like to learn more about the background and deeper principles behind Symbian OS, but that will have to wait – since at the moment, you're looking for quick recipes.

Congratulations, you have found them!

In the pages ahead, you will find recipes covering topics such as Bluetooth, networking, location-based services, multimedia, telephony, file handling, personal information management – and much more. In most recipes we provide working code fragments that you should be able to copy and paste directly into your own programs, and we provide a full set of sample code for download from the book's website (***developer.symbian.com/quickrecipesbook***). We have also listed some common gotchas, so you can steer clear of these potential pitfalls.

Since you are in a hurry, I will stop writing now (even though there is lots more I would like to discuss with you), so that you can proceed at full pace into the material in the following pages. Good speed!

David Wood, Symbian, March 2008

About this Book

This book sets out to accomplish two goals:

- For readers who don't know Symbian OS C++ development, this book is a two-week crash-course in developing applications for mobile phones.
- For readers who know Symbian OS C++ development, this book explains how to use 10 different technologies in a condensed form. It can be used as a desk reference or to learn a new technology in a familiar environment.

The focus of the book is on the reader being able to manage the time it takes to understand and implement the new concepts in Symbian OS.

Chapter 1 will explain how to set up a development environment. It also contains useful information for you to read while you wait for your software development kit to download and install.

Chapter 2 is about your first HelloWorld application and basic use of the new toolchain.

Chapter 3 explains the essentials of the common Symbian C++ APIs and idioms. Without understanding these, you cannot make much progress.

Chapter 4 is the main course of this book. It consists of recipes for ten self-contained technologies, each explained as a time-bound development learning task. For each technology, the recipes are listed in order of increasing complexity.

Chapter 5 will help you reach the next level of Symbian OS development expertise.

Chapter 6 is dedicated to commercial-grade application development.

This book is by no means an introduction to software development in general or even to the C++ language. The authors expect the readers to be familiar with both.

We have included some links to help you find additional information, but to avoid repetition, we don't always refer you to the Symbian Developer Library API reference documentation whenever we introduce a new API. It's worth mentioning it here though. You'll find a huge amount of reference material in the Symbian Developer Library, which is available in every SDK, and can also be browsed online on the Symbian Developer Network (***developer.symbian.com/main/oslibrary/osdocs***). Chapter 6, Section 6.3 contains other links and suggestions for how to get more information.

About the Authors

Michael Aubert

Michael has worked on Symbian OS for 7 years, in the Java team at Symbian itself and the R&D team at iAnywhere. During that time, he has received in-depth exposure to a wide range of technologies including telephony, messaging, 3D graphics, networking, multimedia, PIM, cryptography, platform security and software deployment.

He holds an MSc in Software Engineering from E.S.I.A.L. and is probably the only person to have ever explained the crazy Java Team Event Server Framework to a French audience.

Alexey Gusev

Alexey started to play with mainframes at the end of the 1980s, using Pascal and REXX, but soon switched to C/C++ and Java on different platforms before moving into mobile technologies. After working for almost a decade as a team leader and architect on Windows Mobile, he decided to join the Symbian Core Development team, originally working on Security and later on USB.

He holds an MSc in Applied Mathematics and Physics from the Moscow Institute of Physics and Technology. He is also an Accredited Symbian Developer and regular author at ***www.developer.com***.

Tanzim Husain

Tanzim joined Symbian in 2004 as a member of the networking technology team and has worked there ever since, surviving two architectural

changes and three team re-organizations. Before joining Symbian, he worked extensively on Windows Mobile, delivering pioneering applications in the areas of mapping and GIS.

Tanzim holds a B.S. in Computer Science from NSU. Outside work, he likes to fiddle around with photography and enjoys escaping to the countryside. He tries to maintain an infrequently updated website/blog at ***www.tanzim.co.uk***.

Jenny Mulholland

Since graduating with an MSc in Physics from the University of Cambridge in 2006, Jenny has worked in Symbian's Licensee Product Development team, as a member of the Comms Porting Group. Jenny recently renewed her Accredited Symbian Developer status.

Outside of work, when she is not in the pub with her colleagues, she enjoys performing concerts with the Chandos Chamber Choir and has recently taken up the flute.

Antony Pranata

Antony holds an MSc in Information Technology from the University of Stuttgart, Germany.

He has been involved in a number of Symbian OS projects in different technology areas, including security, tools, multimedia and location-based services. He currently works for Nokia in Canada.

Antony is a Forum Nokia Champion, Accredited Symbian Developer and Accredited S60 Developer. He has a personal website and blog at ***www.antonypranata.com***. He now lives in Vancouver with his wife, Emi.

Jukka Silvennoinen

Jukka holds a PhD in Computer Information Systems.

Before joining Forum Nokia recently, he spent several years developing many Symbian OS applications, mainly for the Asian markets. As a certified Nokia Trainer he was also a visiting lecturer in one of the best universities in Thailand.

Jukka can often be found haunting the Forum Nokia developer discussion boards and wiki. He recently renewed his Accredited Symbian Developer and Accredited S60 Developer status.

Jo Stichbury

Jo is Senior Technical Editor with Symbian Press. She has worked within the Symbian ecosystem since 1997; in the Base, Connectivity and Security teams of Symbian, as well as for Advansys, Sony Ericsson and Nokia.

Jo is the author of *Symbian OS Explained: Effective C++ Programming for Smartphones*, published by Symbian Press in 2004; she also co-authored *The Accredited Symbian Developer Primer: Fundamentals of Symbian OS* with Mark Jacobs, in 2006. Her most recent publication is *Games on Symbian OS: A Handbook for Mobile Development*, published in early 2008.

Jo became an Accredited Symbian Developer in 2005 and a Forum Nokia Champion in 2006 and 2007.

Acknowledgments

The authors would like to thank the Symbian Press team for allowing us this opportunity, for their patience and for making the process as smooth as possible.

The authors would also like to thank all the technical reviewers for their invaluable comments and for stopping us from making fools of ourselves.

Michael would like to thank: my fellow authors on this book for making me look cleverer than I am, the kind people at Symbian Press for entrusting this project to a new author, David Wood for initiating the project, my family and my current or former colleagues for their help and inspiration.

Tanzim would like to thank: my great colleagues, particularly Nadeem, Tom and Petr, for their insightful knowledge of the Symbian OS networking architecture, and my fellow authors for their helpful suggestions.

Jenny would like to thank: Aaron for all the weekends when his living room became my study, and the Quick Recipes team for all their help.

Antony would like to thank: Emi for her support and understanding during this project and my uncle, William Suryawijaya, for introducing me to the wonderful world of programming back in the 1990s.

Symbian Press would like to thank: each of the authors, who gave up so much of their time to contribute to this book, and our technical reviewers, for their generous feedback. In particular, we'd also like to thank Tanzim for dropping everything to work with us on the copy edits. We'd also like to thank Daniel Mattioli, for letting us 'borrow' Tanzim, and Emmanouil Papathanassiou and Neil Taylor, for help with our example code.

Symbian OS Code Conventions and Notations Used in the Book

For you to get the most out of this book, let's quickly run through the notation we use. The text is straightforward, and where we quote example code, resource files, or project definition files, they will be highlighted as follows:

```
This is example code;
```

Symbian C++ uses established naming conventions. We encourage you to follow them too, in order for your own code to be understood most easily by other Symbian OS developers, and because the conventions have been chosen carefully to reflect object cleanup and ownership, and make code more comprehensible. An additional benefit to using the conventions is that your code can then be tested with automatic code analysis tools, which can flag potential bugs or areas to review.

The best way to get used to the conventions is to look at code snippets in this book, and those provided with your chosen SDK.

1.1 Capitalization

The first letter of class names is capitalized:

```
Class TColor;
```

The words making up variable, class, or function names are adjoining, with the first letter of each word capitalized. Classes and functions have

their initial letter capitalized while, in contrast, function parameters, local, global, and member variables have a lower case first letter.

Apart from the first letter of each word, the rest of each word is given in lower case, including acronyms. For example:

```
void CalculateScore(TInt aCorrectAnswers, TInt aQuestionsAnswered);
class CActiveScheduler;
TInt localVariable;
CShape* iShape;
class CBbc;//Acronyms are not usually written in upper case
```

1.2 Prefixes

Member variables are prefixed with a lower case 'i', which stands for 'instance':

```
TInt iCount;
CBackground* iBitmap;
```

Parameters are prefixed with a lower case 'a', which stands for 'argument'. We do not use 'an' for arguments that start with a vowel.

```
void ExampleFunction(TBool aExampleBool, const TDesC& aName);
```

(*Note*: `TBool aExampleBool` rather than `TBool anExampleBool`).

Local variables have no prefix:

```
TInt localVariable;
CMyClass* ptr = NULL;
```

Class names should be prefixed with the letter appropriate to their Symbian OS type (usually 'C', 'R', 'T', or 'M'), as will be described further in Chapter 3:

```
class CActive;
class TParse;
class RFs;
class MCallback;
```

Constants are prefixed with 'K':

```
const TInt KMaxFilenameLength = 256;
#define KMaxFilenameLength 256
```

Enumerations are simple types, and so are prefixed with ‘T’. Enumeration members are prefixed with ‘E’:

```
enum TWeekdays {EMonday, ETuesday, ...};
```

1.3 Suffixes

A trailing ‘L’ on a function name indicates that the function may leave:

```
void AllocL();
```

A trailing ‘C’ on a function name indicates that the function returns a pointer that has been pushed onto the cleanup stack:

```
Ccylon* NewLC();
```

A trailing ‘D’ on a function name means that it will result in the deletion of the object referred to by the function:

```
TInt ExecuteLD(TInt aResourceId);
```

1

Introduction and Setup

The first part of this chapter will help you set up your development environment using free tools.

While your computer is busy downloading and installing software, you should read the second part of this chapter. It contains a host of critical information you need before you start using the development tools.

When all the tools are installed, there are a few more minor configuration steps to go through in the last part of this chapter.

1.1 Tools: What You Need and Where to Find It

1.1.1 System Requirements

System requirements for C++ development on Symbian OS v9.x are as follows:

- Microsoft Windows 2000 Professional with Service Pack 3 or MS Windows XP Professional with Service Pack 2, running on a laptop or desktop computer. At the time of writing this book, most Symbian OS SDKs do not support Windows Vista.
- At least 512 MB of RAM (1.5 GB recommended).
- 1-GHz or faster Pentium-class processor (2-GHz Pentium-class processor recommended).
- At least 1 GB of free disk space (5 GB recommended).
- 16-bit color display capable of a 1,024 × 768 pixels resolution.
- Java™ Runtime Environment (JRE) 1.4.1_02 or later (available from ***java.sun.com***).
- ActivePerl 5.6.1 build 568 or later (available from ***activestate.com***).

- Microsoft Core XML Services (MSXML) 4.0.
- ZIP decompression software to open the installation package.
- Local-administrator rights for installation and removal of software.
- The PC Suite software for your handset. You should find the latest version of this software on the website of the manufacturer of the handset.

ActivePerl may be bundled in some of the Software Development Kit packages you will be installing.

1.1.2 IDE

The recommended Integrated Development Environment for Symbian OS C++ development is called Carbide.c++. You can download the free Express Edition from: ***forum.nokia.com/carbide***. Section 1.2.1 describes Carbide.c++ in more detail.

1.1.3 SDKs

Symbian OS is split into two platforms: S60 3rd Edition and UIQ 3. (There is a third one, called MOAP, available only in Japan, which this book doesn't cover.)

Each platform is replaced as different versions (or Feature Pack numbers for S60).

Once you know which handsets you want your applications to run on, you can identify all your target platforms and their version using the following websites: ***www.s60.com/life/s60phone*** (for S60) and ***www.uiq.com/uiqphones*** (for UIQ).

You need to download all the SDKs for all your target platforms. At the time of writing, this could mean up to five different SDKs for development on Symbian OS v9.x: S60 3rd Edition Maintenance Release, Feature Pack 1 and Feature Pack 2 Beta, along with UIQ 3.0 and UIQ 3.1.

The SDKs can be downloaded from: ***developer.symbian.com/main/tools***.

1.1.4 Compilers

There are two free compilers used for Symbian OS C++ development.

The Nokia x86 compiler targets the Symbian OS emulator for Microsoft Windows included in the SDK. You will see this referenced as WINSCW.

The open source GCC-E compiler targets the actual handsets, where Symbian OS runs on an ARM processor.

Both compilers should be bundled in the SDK installation package, but GCC-E must be installed to a folder whose path contains no 'space' characters or your builds will fail with 'missing separator' errors.

1.2 While You are Waiting

1.2.1 Carbide.c++

There are four different editions of Carbide.c++:

- Express Edition
- Developer Edition
- Professional Edition
- OEM Edition.

The free Express Edition is the only one that doesn't allow debugging an application running on the actual phone (on-target debugging). All versions allow emulator debugging.

Carbide.c++ is based on Eclipse and is highly customizable. You can even create your own plug-ins to extend its functionality.

When you first launch Carbide.c++ it allows you to define a 'workspace'. We suggest you create a different workspace for each SDK.

There is a very nice Flash tutorial to introduce Carbide.c++ at ***developer.symbian.com/main/learning/flash***, and a booklet about getting started with Carbide.c++ available from ***developer.symbian.com/carbide_booklet_wikipage***.

1.2.2 Development Communities

In order to get more information, meet your fellow developers, ask questions and find out the solutions for common problems, you should visit the following links:

- Symbian Developer Network: ***developer.symbian.com***.
- Forum Nokia: ***forum.nokia.com***.
- UIQ Developer Community: ***developer.uiq.com***.
- Sony Ericsson Developer Forum: ***developer.sonyericsson.com***.
- MOTODEV Developer Forum: ***developer.motorola.com***.

While you are waiting, you should register on, at the very least, the following website:

- Symbian Signed: ***www.symbiansigned.com***.

There are also many independent sites, which contain a lot of helpful and educational information; for instance, ***www.newlc.com***.

1.2.3 Concepts of Mobile Development

On an open operating system such as Symbian OS, third-party developers like you can install their own applications.

Programming for smartphones puts you in a position where you have quite limited resources, such as CPU capabilities, battery power, input methods, available memory amount and so on. Software developers targeting such a challenging environment need to focus on code efficiency and robustness more than they may be used to for desktop platforms.

We are working in an environment where any heap memory allocation can fail and where the cost of forcing the CPU to switch between processes (or even between threads) is non-negligible.

Several Symbian OS development efforts are worth being aware of:

- Open Source projects at ***www.symbianos.org***.
- P.I.P.S. and Open C are there to help you port applications using the Posix libraries to Symbian OS (see ***www.forum.nokia.com/openc***).
- The Standard Template Library is being ported to Symbian OS.
- The Net60 framework from Red Five Labs allows you to use Microsoft .NET Compact Framework (see ***www.redfivelabs.com***).
- There are runtime environments for Java, Ruby and Python available for Symbian OS (see ***developer.symbian.com/main/getstarted/hub/runtimes_hub.jsp***).

1.2.4 ARM Hardware

When targeting the actual handset, Symbian OS binaries are based on the Application Binary Interface (ABI) for the ARM Architecture. ABI is a standard for the interfaces of binary code running in ARM environments. The specification is published by ARM at: ***www.arm.com/products/DevTools/ABI.html***.

We will not go into the details of various ARM instruction sets and ABI versions in this book.

1.2.5 Emulator

In addition to debugging using a stack trace, breakpoints and variable monitoring, the emulator allows you to test your deployment packages to ensure your application will be installed properly on the real smartphone.

The S60 emulator preferences allow you to set up several useful parameters, like Bluetooth and IrDA (Infrared) ports and platform security

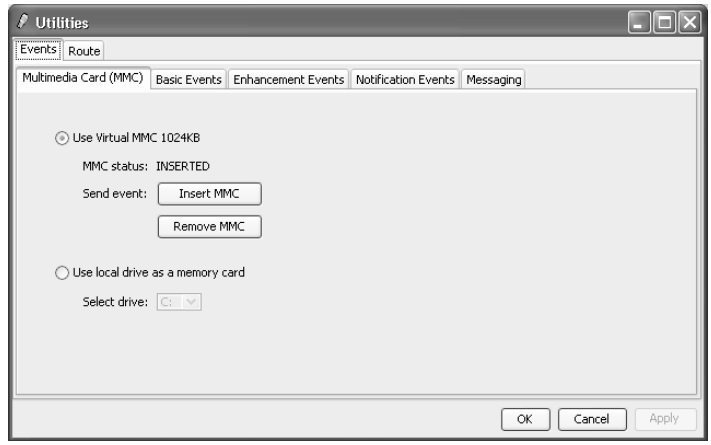

Figure 1.1 S60 Utilities

settings. In addition, the emulator has a powerful utilities application which helps you test your software by triggering various events, like memory card insertion (Figure 1.1).

The UIQ emulator also offers some configuration capabilities via a separate utility – *SDKConfig* (Figure 1.2) – a graphical configuration tool located in the `...\epoc32\tools\distrib` folder of your SDK.

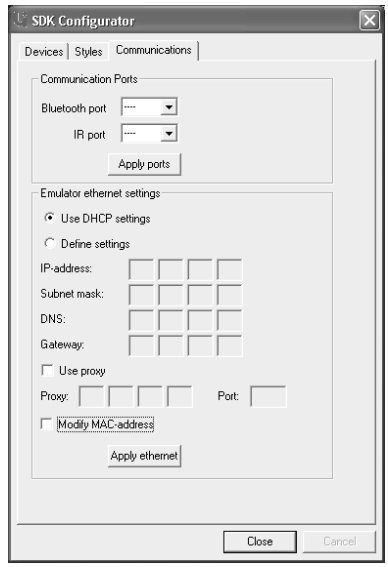

Figure 1.2 UIQ SDK Configurator

SDKConfig launches a number of scripts from `...\epoc32\tools`. This utility can be used to select the default device, change the UI style of the emulator and set up communication ports and Ethernet connection. The easiest way to define Ethernet setting is to use DHCP and uncheck the 'Modify MAC-address' checkbox.

1.3 Post-Installation

1.3.1 Command Line Tools

Without arguments, the `devices` command lists all the installed SDKs from a DOS prompt:

```
S60_3rd:com.nokia.s60
S60_3rd_FP2_Beta:com.nokia.s60 - default
S60_3rd_FP1:com.nokia.s60
UIQ3.1:com.symbian.UIQ
```

It also allows you to switch between SDKs as follows:

```
devices -setdefault @S60_3rd_FP1:com.nokia.s60
```

1.3.2 SDK Directories Structure

Each SDK installs its own emulator binaries, handset libraries, tools code examples and documentation on your hard drive.

`...\epoc32\winscw\` is where you will find the emulator drives.
`...\epoc32\release\winscw\udeb\` is where you will find the binaries for the emulator, particularly `epoc.exe` which launches the emulator.
`...\epoc32\release\gcce\urel` is where you will find the ARM libraries your code will compile against.

The code examples are split between the generic Symbian OS examples and the examples using APIs specific to the S60 or UIQ platform.

SDK documentation is mainly in the form of CHM or PDF files. It contains information related to all the APIs publicly available in Symbian OS, along with information about the tools included in the SDK.

1.3.3 Emulators

The DOS command line to start the emulator is always:

```
...\epoc32\release\winscw\udeb\epoc.exe
```

On a UIQ 3.1 emulator, it is recommended to lower the color depth by replacing `WINDOWMODE COLOR16MU` by `WINDOWMODE COLOR64K` in the file

```
...\epoc32\release\winscw\udeb\z\system\data\wsini.ini
```

In the Symbian OS file system, the Z: drive represents the phone's ROM.