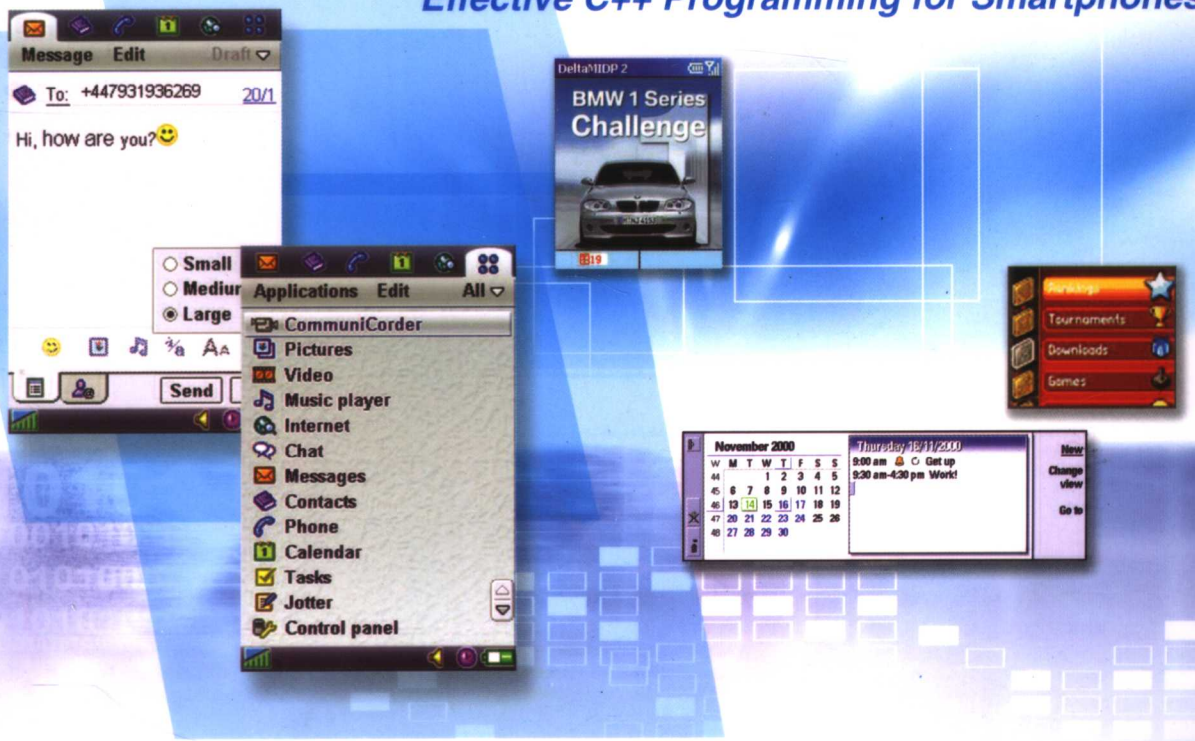
移动开发系列丛书

WILEY symbian

Symbian OS C++ 高效编程

〔美〕Jo Stichbury 著
谢 轩 译

Symbian OS Explained
Effective C++ Programming for Smartphones



人民邮电出版社
POSTS & TELECOM PRESS

移动开发系列丛书

Symbian OS C++高效编程

[美] Jo Stichbury 著

谢 轩 译

人 民 邮 电 出 版 社

图书在版编目（CIP）数据

Symbian OS C++高效编程 /（美）斯蒂克伯里（Stichbury, J.）著；谢轩译.
—北京：人民邮电出版社，2006.3

ISBN 7-115-14319-6

Ⅰ. S... Ⅱ. ①斯...②谢... Ⅲ. C语言—程序设计—应用—移动通信—携带电话机
Ⅳ. ①TN929.53②TP312

中国版本图书馆CIP数据核字（2006）第010077号

移动开发系列丛书

Symbian OS C++高效编程

◆ 著 [美] Jo Stichbury
译 谢 轩
责任编辑 陈 昇

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街14号
邮编 100061 电子函件 315@ptpress.com.cn
网址 http://www.ptpress.com.cn
北京顺义振华印刷厂印刷
新华书店总店北京发行所经销

◆ 开本：800×1000 1/16
印张：18
字数：403千字 2006年3月第1版
印数：1－4 000册 2006年3月北京第1次印刷

著作权合同登记号 图字：01-2005-1412号

ISBN 7-115-14319-6/TP・5177

定价：42.00元

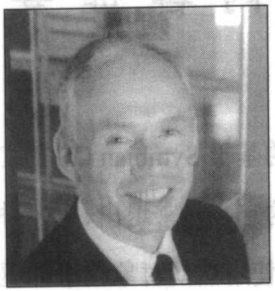
读者服务热线：(010)67132692 印装质量热线：(010)67129223

内 容 提 要

本书旨在帮助读者编写良好的基于Symbian OS的C++程序。全书共分为21章，分别介绍了Symbian OS中的类命名约定、异常退出、清除栈、两段构造、描述符、良好的描述符风格、动态的数组与缓冲区、使用活动对象的事件驱动多任务、活动对象、Symbian OS的线程与进程、客户/服务器原理、客户/服务器框架实践、二进制类型、ECOM、严重错误、用断言发现bug、调试宏与测试类、兼容性、轻量级模板、API、良好的编码风格等内容。

本书适合于使用Symbian OS进行程序开发的人员。

序　言



软件工程师生活在软件变得极为普遍这样一个有趣的时代里。我们都越来越依赖于我们的个人电脑，并且使用个人电脑的软件，这些软件已经成为我们组织自己生活的基本工具了。但我们所“看到”的只是冰山的一角。大多数软件存在于表象之下，它们运行于各种各样的嵌入式系统中，诸如电子消费设备、汽车和飞机。Symbian OS 瞄准的是移动电话，这是一类数量极其庞大的嵌入式系统，它在全球各地广为使用。

移动电话中的软件数量增长迅速。在最近一段时间里，它已经超出了摩尔定律：在过去的 3 年里，高端电话中的嵌入式软件的大小已经从 2MB 跳增至 20MB。这在一定程度上是新 3G 网络的成熟所要求的，但主要还是由于手机集合了其他可接驳消费设备的功能，诸如：数码相机和便携式摄像机、数字音频播放器、视频播放器、电子记事本、移动游戏终端、可接驳无线广播、可接驳电视、电子邮件终端、无线电话，甚至还有电子支付卡。移动电话正在成为重要的可接驳的日常生活支持系统——一把电子的“瑞士军刀”。

Symbian OS 为我们所需要，是因为软件支持的功能爆炸式增长需要一个功能强大的操作系统，它必须能适合复杂的、持续运转的、电池供电的移动设备。

Symbian OS 的面向对象程序设计范型有助于管理系统复杂性，并且它深入到了整个 Symbian OS 的架构。这个架构使用了许多高级但很经典的部件，这些部件可以在其他的多任务操作系统中找到。典型的部件包括：抢占式多任务线程、进程、异步服务以及对共享资源访问进行串行化的内部服务器。Symbian OS 有一些特别的特性，如果想成为一个高效的 Symbian OS 程序员，对这些内容也要好好理解。这些特性被设计用于应对移动设备程序设计中的严格的要求，例如，在异步事件和错误的处理中，要避免内存泄漏和其他悬空资源[1]。

来自嵌入式领域的工程师可能需要从 C 迁移到 C++的面向对象世界中。而来自 PC 领域的工程师可能已经习惯于 C++，但也许并不习惯于移动电话程序设计中更严格的要求。在移动电话的程序设计中，健壮性、代码尺寸、内存的使用、性能和电池寿命都是很重要的。在这样的情况下，不可以定期重启以清除内存泄漏，而且无线信号大大增加了应用程序所需响应事件的数量。

无论你来自什么样的背景，经验水平如何，只要你阅读了本书并理解了 Symbian OS 的基本概念，那么你的效能就一定会大大提高的！

Symbian 公司首席技术官
Charles Davies

[1] 译注：“悬空资源”，意指丢失了所有权的资源，以及对无效资源的引用。

关于本书

编写良好的基于 Symbian OS 的 C++代码需要对基本概念以及操作系统的本质有清晰的理解。本书讲解了 Symbian OS 的关键特性，并且展示了如何最高效地使用这些知识。本书还关注了特别应用于 Symbian OS 的良好 C++风格的一些内容。通过理解和实践，提出编写基于 Symbian OS 的高质量 C++代码所需的专家意见应该会成为一种本能反应。

本书分为一系列独立的章节，每一章都讨论某些特定的 Symbian OS 重要特性。除了讲解基础知识，每章还会展示最佳的实践并阐明所有要避免的常见错误。这些章节都很简洁，既能够增长你的知识，却又不再需要获取其他额外信息。每章都给出了简单而直接的讲解，同时却没有遗漏任何重要内容。

本书不会教你如何编写 C++代码。本书假定读者已经熟悉了 C++语言最重要的概念。同时本书也不会从编写特定应用程序的角度带你了解 Symbian OS。取而代之的方法是，我将试图向读者传授对 Symbian OS 的中心概念和关键特性，以及良好的 C++技术的理解。引用 Scott Meyers[1]的一句话："本书中你会发现该做什么，为什么应该这么做；不该做什么，为什么不应该这么做的忠告"。正是他的书给我带来了编写本书的灵感。

适用对象

本书假设读者对 C++程序设计具有一定的理解，但并没有假定读者懂得深入的 Symbian OS 知识。本书除了包含诸如 Symbian OS 客户/服务器架构（第 11、12 章）和 ECOM（第 14 章）之类的 Symbian OS 复杂特性以外，还涵盖了 Symbian OS 的基础知识，诸如描述符（第 5、6 章）和活动对象（第 8、9 章）。

本书着重描述适用于所有 Symbian OS 版本的操作系统核心，以及用户接口，诸如 UIQ 和 60 系列。平台之间任何重要的差别都会被突出显示出来。在本书编写期间（2004 年春），Symbian 正准备推出新版本操作系统 Symbian OS 8.0。本书显式地标明了作者意识到新版本将引入变化的所有地方。

如果你是针对 Symbian OS 或者考虑要针对 Symbian OS 的开发人员，那么本书将为你展示如何编写最高效的 C++代码。对该操作系统特性和设计的深刻理解，以及对使用它们的自信都将使

[1] Scott Meyers，《*Effective C++：50 Special ways to improve your programs and designs*》，1997 年。查看参考书目可获取更多细节。

你从中获益。无论对于 Symbian OS 有多少经验，总会有新的技巧可以学习，这就是为什么本书适合所有层次开发人员的原因。本书反映出了很多与我合作过的颇具经验的 Symbian OS 开发人员的智慧。他们这些年教给了我很多东西，而在我详细考察 Symbian OS 各部分并撰写该书时，我甚至学到了更多东西。我希望你也能从本书中学到这些东西。

如何使用本书

前面曾提到，本书分为很多章，每一章都起到对 Symbian OS 中一个特性的导向作用。每章的标题、明细目录和索引、术语表和参考书目部分都被设计为能够帮助你找到需要的信息。

这些章不必按顺序阅读。本书甚至更适合为重读、引用或提供“每日一贴”而浏览。本书也无需从头读到尾。章与章之间如有重叠内容，都会互相交叉引用的，这样可以表明还有哪些地方与当前的讨论尤为相关。

出于简洁和讲解的目的，每章中也使用了示例代码用于展示良好代码风格和 Symbian OS 代码约定及布局。

本书使用的符号和代码约定

本书的文本布局无需解释，但对代码布局则需要作一些介绍。

这是示例代码：

```
this is example code
```

Symbian OS 的 C++代码使用建立好的命名约定，你应该遵守该约定，这样可以使代码最清晰地表达它的用意。除了表达，遵守约定的最主要好处就在于它可以反映对象的清除和所有权。

习惯这些约定的最好的方法就是阅读代码示例，诸如 SDK 中和本书中的示例。命名约定的主要特性会在这里有所描述。在第 1 章中，讨论了类名的前缀约定，第 2 章中阐述了函数名使用后缀 L 的原因，而第 3 章讨论了后缀 C 的使用。

大写

类名的首字母要大写：

```
class TClanger;
```

构成变量名、类名或函数名的单词应该紧密相连，并且在适当的地方每个单词的首字母都要大写（参数变量、自动变量、全局变量和成员变量以及函数参数应该具有小写的首字母）。而每个单词的其余部分都使用小写，包括那些首字母缩写：

```
void SortFunction();
TInt myLocalVariable;
CMemberVariable* iDevitsHaircut;
class CActiveScheduler;
class CBbc; // 首字母缩写通常也不作大写
```

全局变量是不提倡使用的，但是如果使用，则要么以大写字母开头，要么加以小写“g”作前缀。

前缀

成员变量都被加以小写“i”前缀，用于代表“instance（实例）”。

```
TInt iCount;
CPhilosopher* iThinker;
```

参数被加以小写“a”作前缀，代表“argument（参数）”。不要为元音开头的参数加上“an”前缀。

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

注意是 TBool aExampleBool，而非 TBool anExampleBool。

自动变量没有前缀，并且其首字母是小写。

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

类名应该以恰当的字母作前缀（“C”、“R”、“T”或者“M”，这在第 1 章有详细阐述）。

```
class CActive;
class TParse;
```

常量应以“K”为前缀。

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

枚举成员用“E”为前缀。而枚举则是类型，因此使用“T”为其前缀（可以在第 1 章中找到关于 T 类的更多信息）。

```
enum TChilliPeppers {EScotchBonnet, EJalapeno, ECayenne};
```

后缀

一个函数名后跟“L”就说明它可能会发生异常退出。

```
void ConstructL();
```

一个函数名后跟“C”则说明它返回一个指针，而该指针已经被压入清除栈中。

```
CPhilosopher* NewLC();
```

一个函数名后跟“D”意味着函数将删除在其中引用的对象。

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

下划线

除了在宏（__ASSERT_DEBUG）或资源文件（MENU_ITEM）中，应该避免使用下划线。

代码布局

你会注意到，在本书中，Symbian OS 代码中大括号的布局是和紧接的语句采用相同缩进的。我不是这样一种用法的忠实拥护者，而且我也不觉得使用其他布局会有什么不同。但是，如果你想坚持以“Symbian 方式”编写代码，那么就应该遵守下面的约定：

```
void EatChilliL(CChilliPepper* aChilli)
    {
    if (!aChilli)
        {
        User:: Leave(KErrArgument);
        }
    TChilliType type = aChilli->Type();
    if (EScotchBonnet==type)
        {
        User::Leave(KErrNotSupported);
        }
    DoEatChilliL(aChilli);
    }
```

提示

在整本书中，提示和建议使用以下字体格式来表示，例如：

这些提示用于补充说明每一小节的内容，其中提供了对讨论中的关键点的提示。

Symbian OS 介绍

移动电话的一个关键特征就是它很小，尽量小，尽量轻。

移动电话总是伴随着你左右，随时准备发出或接收电话或短信，执行闹钟叫醒你，连接到电话网络或其他设备，组织你的个人信息或玩游戏。无论你想做什么，都会立刻想到用你的电话。它不应该具有和 PC 机一样的长启动时间。它应该在被拿起的时候就开始响应用户输入，而且应该是可靠的。作为一个移动电话持有者，你还能自己记住多少联系人的电话号码？你的手机为你保存个人数据，而且起码它不会丢失这些数据。

让我们从诸如 Symbian OS 之类的移动操作系统设计上检验这些特性所带来的结果。电话可能会很轻很小，但作为用户，我们还希望它具有相当长的电池使用寿命。这就使得高效的能源管理变得尤为重要。操作系统不能很快就耗光电池，并且必须允许处理器在可能的时候关闭系统的某些部分，但是它不能完全断电，因为它必须处理随时可能接入的呼叫和短信，以及信号警告。

用户期望电话是快速响应的，而不是对每次按键都反应迟缓。操作系统和硬件必须小心地平衡良好的性能速度的要求和耗能巨大的处理器对能量的需求。成本也是很重要的，它们限制了移

动设备上的处理器和内存容量。操作系统必须是高效的，要充分利用有限的空闲处理器和内存资源，同时还要使用最少的能量。

除了要高效，操作系统在有限的资源耗尽时还必须要健壮。它必须被设计成能够应付低内存、掉电或通信连接不可用等情况。内存管理是关键。操作系统必须精确跟踪宝贵的系统资源，并在不再需要时释放它们。内存缓慢地泄漏是不可接受的，这会导致性能和可用性的瓦解，直至用户被迫重启。操作系统应该使软件工程师编写无内存泄漏的代码更容易，并且还要在发生内存耗尽的情况时能够处理这个问题。

移动电话市场是很庞大的，每年生产的数量数以亿计。召回它们或要求用户通过服务包（service pack）来升级，即使可能，也会很困难。所以一旦出产了一部电话，它就是出产了，它不可以有任何严重的瑕疵。不仅是电话所基于的平台必须是构建良好的，它还必须为开发人员提供构建、调试和测试健壮代码的方法。

作为用户，我们也会要求我们的移动电话尽量便宜。我们想要最新最时髦的电话，它只是比前一个更小或多了一些特性，诸如蓝牙、集成的相机或视频播放器或者是 MP3 播放器。这就意味着一部电话在市场上的生命期是有限的。当制造商开发出一款移动电话时，它必须尽快准备好投放市场，而且理想的操作系统应该是灵活的，这样一来就可以通过扩展或升级基本设计，从而发布不同的机型。

Symbian OS 是为移动设备而设计的，从它诞生自 Psion 5 系列中的 EPOC32 时就是如此。当今移动电话的很多需求在那个时代同样是适用的，它的设计正体现了这一点。人们对移动操作系统的大量需求造就了 Symbian OS，从弹性能源管理和内存资源的小心使用，到 C++和面向对象程序设计技术的复杂使用。作为 Symbian OS 的一名开发人员，你可以获益于这个专门为移动设备创建并与市场一同演化的平台。

当然，职责并不止于操作系统。要在移动电话上取得最大成功，你的代码也必须是高效的、健壮的、快速响应的并且是可扩展的。但一些移动计算方面的问题是出了名地难以解决，而且 C++程序设计即便在最有利的情况下也是很复杂的。这并不容易，但选择在 Symbian OS 上开发将给予你一个专门构建的平台所带来的很多好处。本书就将带给你最好的建议，这些建议都来自那些经验老道的 Symbian OS 开发人员。

关 于 作 者

Jo Stichbury 毕业于 Cambridge 的 Magdalene 学院，并在那里获得了 Stothert Bye 奖学金。她有一个自然科学的硕士学位，一个有机钼化合物化学的博士学位。在 Imperial 学院经过一段时间的博士后研究后，她于 1997 加入了 Psion Software。那时 Symbian OS 还只是被称为 EPOC32。从那时到现在，她一直在使用这个操作系统，无论是在 Sybmian 的“基础、连接和安全团队”，还是在 Advansys、Sony Ericsson 和 Nokia。

正如本书内容中所展示的，Jo 在一定程度上对 Clanger 和希腊神话有着很不一般的兴趣。她现在和她的搭档以及两只猫住在温哥华。

关 于 译 者

谢轩，毕业于南京大学软件学院，现任 Gameloft 上海公司软件工程师，从事手机游戏的开发，对 C++、泛型、模式有深入的研究和一定的实践经验，曾译、著若干篇技术文章发表于 CSDN 等技术论坛，目前兴趣方向是 C++、泛型和设计模式在手机及嵌入式软件开发中的应用。

作者致谢

从前，当我加入 Psion Software 的时候，市在上还没任何关于 Symbian OS（就是当时众所周知的 EPOC32）的书。所以我怀着感激之情向 Morgan Henry、Matthew Lewis、Peter Scobie 和 Graham Darnell 致谢。他们帮助我迈出了作为程序员的第一步。我要以此书记录他们以及所有与我合作过的 Symbian OS 开发者的洞察力和知识。

我要感谢为本书做出贡献的每一个人。Leon Clarke、Will Bamberg、Mark Jacobs 和 Paul Stevens 在早期给出了非常有用的建议；后来，Julian Lee、Keith Robertson 和 Dennis May 为我解答技术问题。Will Bamberg 向我提供了他自己的关于异常退出和内存耗尽检测的详尽文档，而且当我将这些文档拆散用于加入到第 12 章和第 17 章时，他也毫无怨言。

还要非常感谢来自我的所有评审者的意见。我要感谢这些官方评审员：Keith、David、John、Colin、Andrew、Morgan、Martin、Phil 和 Reem，他们为本书的技术精确性和我的平和思绪作出了巨大的贡献。我还接受了 Sony Ericsson 的 Peter van Sebille 和 John Blaiklock，Symbian 的 Mark Jacobs 和 William Roberts，以及 Will Bamberg 所给予的很多颇有益处的建议。

我要非常感谢 Sony Ericcson 的 Dave Morten、Steve Burke 和 Brigid Mullally。他们对我写作所花费的时间表示理解。没有他们的合作，我是无法完成本书的。

本书中的图表是由 Brigid 制作的。

如果没有我的搭档 Mark 的帮助和支持，本书也是不可能完成的。Mark 使一切都变得有可能。当开始写作时，我收养了两只暹罗猫，虽然它们花费大量的时间分散我的精力而不是帮助我，但也正因为此，在这里还是要包括它们。

我要感谢 Symbian Press，尤其是 Freddie Gjertsen 和 Phil Northam，感谢他们的耐心和坚韧。要感谢的还有 John Wiley 的 Gaynor Redvers-Mutton，以及推动整个计划开始的 Karen Mosman。

另外我也要感谢 Oliver Postgate 授权给我在书中包含了 Clanger 们[1]。

[1] 译者：参见书后附录 2：术语表。

Symbian Press 致谢

首先，我们要感谢 Jo，感谢她在本书的诞生过程中不知疲倦的努力。

还要感谢我们所有的技术审稿者，无论是署名的还是匿名的，尤其要感谢的是年轻的 Phil，他的易怒程度远远超过了他的年龄。如果没有“建设性”的批评，我们会是什么样子？

最后，但不是至少还要感谢 Boundary Row 可爱的 Gayle 和 Victoria。没有持续不断的饮品供给我们，Symbian Press 将不复存在。

封面设计源自 Jonathan Tastard 的概念。

Code checklist 是基于 Symbian 的系统管理组（System Management Group）的文档的。

目　录