

maemo 4 快速入门指南

2007年10月11日

Technology

NOKIA

版权© 2007 年，诺基亚公司，诺基亚公司保留全部权利。

“诺基亚”，“诺基亚论坛”与 maemo 是诺基亚公司的商标或注册商标。在此提到的其它产品和公司名称可能是其所有者的商标或商业名称。

声明

本文信息基于其现有状况，不存在任何保证，包括销售保证、适用某一特殊用途的保证，或从任何建议、规范或范例中衍生出来的保证。本文仅用作信息通报。

诺基亚公司不承担所有因实施本文档中所表述的信息而产生的相关责任，包括侵犯任何知识产权的责任。诺基亚公司并不保证或认为使用这些信息不会构成对这些权利的侵犯。

诺基亚公司保留不预先通知而随时修改本文档的权力。

授权许可

本作品以 CreativeCommons 之署名-相同方式共享 3.0 许可的方式授权。

目录

第 1 章 综述	6
1.1. 历史与理念.....	6
1.2. 读者对象	6
1.3. 阅读指南	6
1.3.1. 对于急于求成者	6
第 2 章 术语与定义	7
第 3 章 架构	9
3.1 内核.....	9
3.2 基本发行版.....	9
3.3 应用框架	9
3.4 maemo 与桌面 Linux 发行版的比较	10
3.4.1 CPU 架构差异.....	10
3.4.2 安全模型差异.....	10
3.4.3 基系统差异	11
第 4 章 用户界面开发	12
4.1. Hildon 桌面	12
4.2. 用 GTK+创建图形化用户界面元素	12
第 5 章 一些重要的系统服务	14
5.1 文件系统 - GnomeVFS.....	14
5.2 应用参数 - GConf.....	14
5.3 D-Bus	14
5.4 其他系统服务	14
5.4.1 警报框架	15
5.4.2 GPS API	15
第 6 章 多媒体.....	16
6.1 GStreamer - 多媒体框架.....	16
6.2 数据流编码与解码.....	16
6.3 音频.....	16
6.4 视频.....	16
6.5 数字信号处理器	17
6.6 图形与图像.....	17
6.7 游戏.....	17
第 7 章 通信	18
第 8 章 开发环境.....	20

第 9 章 移植软件.....	23
第 10 章 品质考虑	24
第 11 章 maemo.org	25
参考文献	26
请对本资源进行评价	30

修订记录

2007年11月20日	中文版本 1.0	中文版本首次发布

第 1 章 综述

本文力图为开发伙伴描述 maemo 平台的总体图像，以便他们能向诺基亚 Internet Tablets 提供应用。这些移动终端具有 ARM 架构，运行一个基于 Linux 的操作系统。Maemo 软件开发工具 (software development kit, SDK) 目前被提供给如 Debian 和 Ubuntu 这样的桌面 GNU/Linux 发行版。这个 SDK 也能通过一个虚拟环境用于其他一些操作系统。

本文主要讲解创建及移植 maemo 兼容软件所需要的一些基础知识。它易于入门，读者甚至无需熟悉如交叉编译或触摸屏用户界面设计这样的概念。熟悉 GNU/Linux, GTK+ 工具包和 C 编程语言的读者将感到十分得心应手。

本文也是为 GNOME Mobile 系列的任何平台或终端进行软件开发的良好起点。除了介绍 maemo 所提供的新框架、库和工具，也讲解了这个新一代互联网终端的总体设计概念。

1.1. 历史与理念

Maemo 平台于 2005 年 5 月 25 日在纽约举行的 LinuxWorld 峰会上当首款诺基亚 Internet Tablet 发布时推出。Maemo 的设计理念是成为移动硬件中真正的桌面电脑。其目标是：方便系统扩展，方便向平台创建新应用及引入现有的桌面应用。

Maemo 使用一些已知的开放式桌面框架以支持软件的易移植性和熟悉度。这一策略与许多基于 Linux 的媒体播放器和电话大不相同，后者是封闭的，或者需要特别的开发工具。Maemo 平台中约 90% 是开放的，其大部分来自上游的开源项目。那些不开放的部分包括如部分用户界面组件和设备驱动，由诺基亚或第三方供应商拥有。

一个重要的理念是：支持方便的开发和改写。开发伙伴可以对该平台作修改，例如引入自己的内核模块。诺基亚同时主持着针对该平台的一个开源 maemo 社区 (maemo.org)。

○ .

1.2. 读者对象

任何掌握 GNU/Linux 和 GNOME 技术的开发伙伴都会有兴趣阅读这一指南。由于文中提供了许多外部资源链接，本文甚至能作为有志于探索技术或成为开发者的新手的起点。

对那些希望充分利用这份指南的应用开发伙伴来说，他/她应该具有 GNU/Linux C 编程基础知识。我们假定他们了解那些最为普遍的工具，如 *gcc*。Maemo 用户界面开发的大部分都基于 GTK+。如果读者并不熟悉 GTK+，本文是学习该框架的起点。读者并不需要预先掌握 maemo 或移动终端编程知识。

1.3. 阅读指南

读者应先通读以下两章，了解一些术语和定义，及 maemo 架构。然后根据自己的兴趣依次阅读其他章节。

1.3.1. 对于急于求成者

尽管不建议如此，但你也可以根据 maemo 教程[36]用 maemo SDK 立即进行改写工作。

第 2 章 术语与定义

- **ABI** - **Application Binary Interface**, 应用二进制接口, 提供目标代码级兼容性。
- **API** - **Application Programming Interface**, 应用编程接口, 提供源代码级兼容性。
- **Applet** - 集成到 **Hildon** 桌面的一个小应用。
- **ARMEL** - 如 **Debian** 等用于 **ARM EABI** (面向 **ARM** 架构的 **ABI**) 的一个名字。
- **devkit** - **maemo SDK** 的一部分, 含有软件开发工具。**SDK** 中含有多个 **devkits**, 如 **doctools**。
- **Hildon** - 用于 **maemo** 平台的应用框架。由诺基亚开发, 基于 **GNOME/GTK+** 技术, 目前正在成为 **gnome.org** 中的一个上游项目。
- **Hildon Desktop** - **maemo** 发行版 **Chinook** 的主要用户界面组件, 是 **maemo desktop** 的重写本。
- **Internet Tablet** - 互联网优化的触摸屏移动终端产品线。该术语由诺基亚新创, 但得到了广泛使用, 包括其它一些产品。
- **initfs** - 初始文件系统, 在 **Linux** 内核启动阶段用作根文件系统, 启动后被安装到了 **/mnt/initfs**。
- **maemo** - 诺基亚所开发的面向移动终端的软件平台, 基于 **GNU/Linux** 和 **GNOME/GTK+** 技术。它包括一些专属的组件使之能运行于诺基亚的 **Internet Tablets**。
- **maemo.org** - 由诺基亚维护的开发伙伴社区网站, 总体上是面向开源和第三方开发伙伴的主要参考资源。
- **maemo desktop** - **maemo** 发行版 **Bora** 的主用户界面组件版本。
- **maemo-of-desktop** - 同 **maemo desktop**。
- **maemo SDK** - 软件开发工具包, 用于在 **PC** 上向 **maemo** 平台创建和移植应用。
- **诺基亚 Internet Tablet OS - maemo 平台** + 一些专属应用, 被打包到由诺基亚提供的正式终端映像中。
- **OSSO** - **Open Source Software Operations**, 开源软件操作, 诺基亚组织, 为 **Internet Tablets** 开发和集成软件。
- **rootfs** - 终端上的根文件系统。
- **rootstrap** - **SDK** 的一部分, 含有经过选择的来自 **rootfs** 的软件组件。**Rootstrap** 是 **Scrachbox** 内某个目标的根文件系统。
- **Sardine** - 一个实验性发行版, 基于 **Hildon for maemo**, 主要面向希望测试未来版本 **maemo** 中正在开发的“疯狂新功能”的开发伙伴们。
- **toolchain** - **SDK** 的一部分, 含有 **ARM** 交叉编译工具, 如编译器和连接器。

一些 **Maemo SDK** 发行版

- **Mistral**: **maemo 2.0 release**, 对应于诺基亚 **Internet Tablet SE 2006** 版本 **2.01.2006.26-8**。

- **Scirocco**: maemo 2.1 release, 主要包括对 bug 的修正和其它一些增强。对应于诺基亚 Internet Tablet SE 2006 版本 2.2006.39-14。
- **Gregale**: maemo 2.2 release (对 bug 的修正和其它一些增强)。
- **Bora**: maemo 3.x release。对应于 Internet Tablet OS release “1.2006.47-20”, “2.2006.51-6” (maemo 3.0), “3.2007.10-7 (maemo 3.1)” 及 “4.2007.26-8”+“4.2007.38-2” (maemo 3.2)
- **Chinook**: maemo 4.0 release
- **Diablo**: 未来的 maemo release
- **Elephanta**: 未来的 maemo release

第3章 架构

本章从应用开发者的角度讲述 maemo 平台的高级架构。然后对 maemo 和广为流行的 Ubuntu 及 Debian 桌面 Linux 发行版作了比较。

3.1 内核

Maemo 使用 Linux2.6 操作系统内核。Linux 是开放源代码的操作系统，由成千上万的志愿者个人和公司开发，他们在 GNU GPL 授权下共享大家的成果。结构上 Linux 具有单一内核。所有内核代码都运行于监管模式下。可以在运行时通过可动态加载的内核模块对内核进行扩展。有许多 APIs 可用于终端驱动、文件系统和网络协议模块。开发伙伴可以添加新内核模块。

maemo 内核基于 ARM 内核分支，允许被开发者修改、重新编译，及刷新。《maemo 内核指南》[34]详细讲解了这些过程。如 WLAN 这类模块只有二进制形式，也就是说，当开发者修改内核时不得更改该模块 APIs。

3.2 基本发行版

Maemo 在很大程度上是基于 Debian[7]Linux 发行版中的一些开放源代码组件。Maemo 针对操作系统的核心部分构建于 GNU/Linux，而其针对用户界面架构的部分则基于 GNOME/GTK+。

Maemo 使用 Debian - *dpkg-tool* 二进制包的相同组件包系统。可以安装新包，删除旧包，整个系统可以用包管理框架升级。文件系统结构也来自 Debian。

为在 Internet Tablet 上运行该软件，已经进行了许多优化和增强工作。其中包括与电源管理、触摸屏输入、性能和大小优化等有关的一些问题。为降低空间占用 maemo 使用了来自 Busybox[4]的一个 shell 和命令行工具。

特别需要指出的是，maemo 平台紧密相关于 GOME Mobile Platform [16]。至本文撰写时，meamo 使用了这个平台中除服务发现之外的全部相同组件。

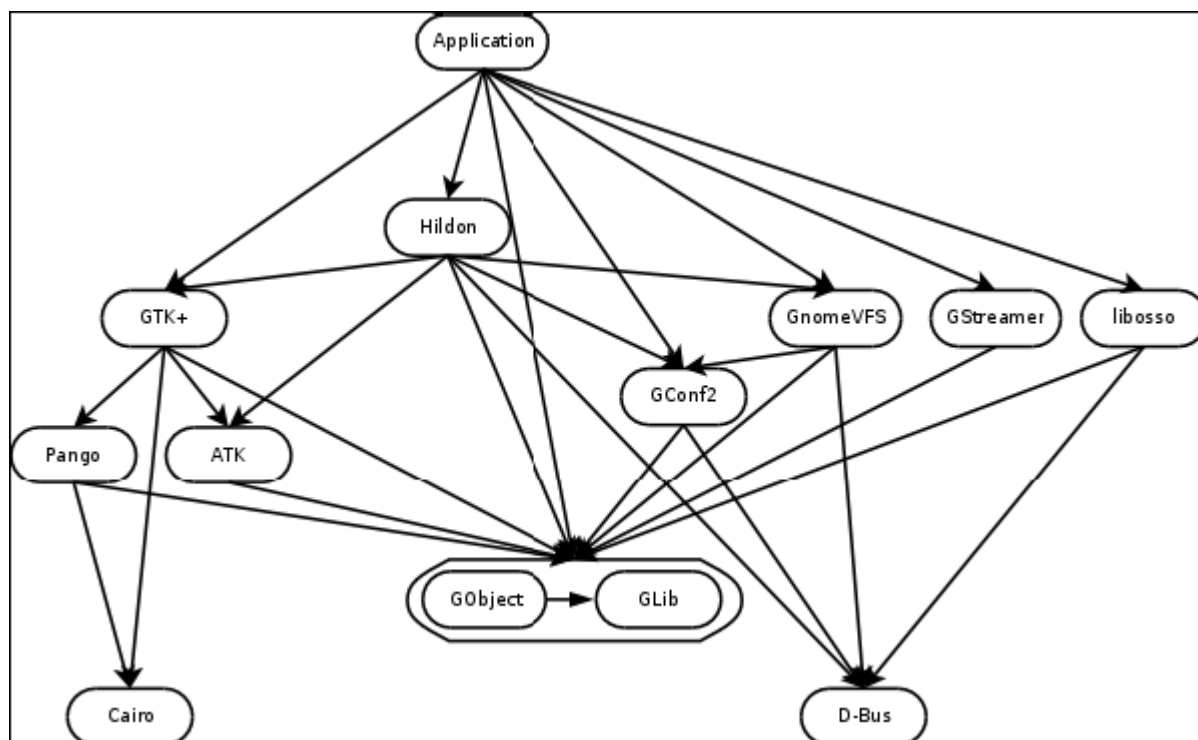
《maemo 架构指南》[39]详细讲解了 maemo 的架构。

3.3 应用框架

应用框架的目的是：为应用提供一个标准的结构，从而有助于应用开发。具有图形化用户界面的应用都倾向于具有类似的结构，例如事件驱动的运行模式。这些事件由用户触发，如，在触摸屏上按下一个按钮。事件也可以由应用引擎本身触发。这方面的例子是，当从网络上接收到了新数据。Maemo 的应用框架被称为 Hildon。它部分基于 GNOME 框架得以构建的同样技术，特别是 GTK+ 组件。

Hildon 向 GNOME/GTK+ 中添加了一些新功能，也对后者做了一些增强，包括 Hildon widget 集、Sapwood 图像服务器、任务导航器、Hildon 控制面板和状态条等。对标准 GNOME 的某些更改，如 *Sapwood*，是为了降低内存需求和提高小型手持设备的速度。此外，Hildon 框架主要用于支持移动性，如自动状态保存、各种触摸屏输入方法，及在物理小尺寸的设备上进行窗口管理等。

GNOME 和 GTK+ 开发伙伴们会对其编程 APIs 感到十分熟悉。该框架在其底层具有 *Glib* 和 *GObject* 对象管理系统。与 GTK+ widget 集一起提供的有 Hildon 扩展。进程间通信是用 *D-BUS* 消息完成的。用户文件通过 *GNOME-VFS* 访问，而多媒体应用则使用 *GStreamer* 来获取针对各种编码解码的加速支持。用户配置通过 *GConf* 保存，并可使用一个 XML parser API。



上图表示了一些最重要的组件及 maemo 应用开发伙伴必须面对的它们之间的依存关系。后面一些章节将详细讨论这些组件。

3.4 maemo 与桌面 Linux 发行版的比较

Ubuntu[71]是一个流行的基于 Debian 的 Linux 发行版，面向普通桌面。Ubuntu wiki[70]列出了 Ubuntu 与 Debian 之间的主要差别。鉴于 Maemo 的设计原则之一就是尽可能接近传统的桌面 Linux，我们在此将详细解释 Ubuntu/Debian 与 maemo 之间的差别。

3.4.1 CPU 架构差异

Debian 支持多 CPU 架构，而 Ubuntu 目前仅支持 x86。与此相比，maemo 却是一个嵌入式的 ARM EABI[2]发行版。Maemo 支持交叉编译，而不像 Ubuntu 是本地编译的。比起 Ubuntu 或 Debian，Maemo 还使用不同版本的工具链（GCC，glibc[19]等），用于支持 ARM 和各种架构间的差异。

3.4.2 安全模型差异

不同于如传统 Linux 桌面那样的多用户系统，maemo 被认为是一个单用户桌面系统。Maemo 中的安全模型注重保护用户免受来自远程或来自自身的攻击，而并不关注其他用户的数据。Maemo 还使用 *suid* 根二进制和 */etc/password*，然而 Ubuntu 那样强制用户使用 *sudo* 和 *shadow passwords*。

不同于 Ubuntu，maemo 像 Debian 那样使用一个根帐户，但却用一个默认的小密码。用户务必在以根用户登录而向终端安装如 *OpenSSH* 之前更改这个根密码。

3.4.3 基系统差异

基系统中的最大差异是：**maemo** 使用一个轻量级的 **BusyBox**[4]，替换了终端上的一些基本 GNU 工具[20]，如 **ls** 和 **sh**。在 **maemo** 中内核和 **initfs** 驻留在不同的分区，并不能如某个普通桌面 **Linux** 那样用一个包管理器来更新。**initfs** 中的程序使用 **uClibc**[72]，而不是 **glibc**。**Ubuntu** 具有 **Perl** 和 **Python** 语言，作为基本包，**Debian** 只有 **Perl**，而 **maemo** 两者都没有。**Maemo** 没有 **debconf**。**Ubuntu** 使用 **Upstart** 来启动终端，而不是如 **maemo** 般使用 **SYSV init scripts**。

第 4 章 用户界面开发

针对图形化用户界面编程的 maemo 平台的起点是 *Hildon* [27]，这是一个应用框架，包括一个轻量级的桌面，一套为手持设备而优化的 widgets，一套主题工具，和其他一些补充库和应用等。

*Hildon*在很大程度上基于GNOME [17]技术。从用户界面的角度来与GNOME桌面比较的话，*Hildon*被设计成能为移动嵌入式终端提供一个新的桌面。因而，它使用如一个轻量级的窗口管理器，名为 *Matchbox* [57]。

4.1. *Hildon* 桌面

诺基亚 Internet Tablets 上的最终用户体验由基于 *Hildon* 的 *Hildon* 桌面提供。*Hildon* 提供了几个库，用于与桌面环境 UI 元素和服务进行交互。

Hildon 框架的主 UI 元素被分成四个大类，它们全部可以用插件扩展：

- ***Hildon Home*** 是一个根桌面，可以被 *Hildon Home* applets 定制。
- ***Hildon* 状态条**提供信息区域和快速访问项目，主要用于交流终端状态的改变。
- ***Hildon* 任务导航器**插件实现最高层桌面菜单。
- ***Hildon* 控制面板**是针对应用配置的总接口，并用控制面板插件进行扩展。更多信息请参阅《为 maemo 编写 *hildon* 桌面插件》[77]一文。

maemo 中的 GUI 应用通常具有一个到多个 *HildonWindows*，这是最高级别的应用窗口。他们可以包括一个标准的菜单条和一个工具条。

Hildon 框架还包括其他一些辅助的 widgets 和服务，用于 *Hildon* 环境。这方面的例子有用于系统对话框和 widgets 的 *Hildon-FM*库、用于显示用户提示的 *HildonBanner* 和用于创建向导用户界面的 *HildonWizardDialog*。有关 *Hildon* widgets 信息的优秀资源是《Maemo 教程》[36]和《Maemo API 参考》[24]。

Hildon 框架的另一个重要元素是 *Hildon Input Method* API，这是在所包括的虚拟键盘和手写识别之外用于创建新用户输入系统的一个接口。《扩展 *Hildon* 的输入方法》[11]一文详细讲述了该 API 的使用。

4.2. 用 GTK+创建图形化用户界面元素

maemo 应用的用户界面是用 GTK+[26]用户界面工具包创建的。GTK+被广泛用于 UNIX GUI 应用，可以说，最著名的 GNOME 桌面环境就基于 GTK+。*Hildon desktop* 构建于一个被修订过的 GTK+ 2.10（名为 *maemo-GTK+2.10* [47]，非常相似的名字）之上，而所有的 *Hildon* widgets 也是 GTK+ widgets。

GTK+本身基于并使用几种低级库，分别具有特定作用。当用 GTK+开发应用时，开发伙伴通常也会明确使用这些库。

- **GObject** [21] 用 *GType* 系统提供面向对象的特性和信号处理功能。
- **Glib** [15] 中具有能用于一般非图形化编程任务的工具，如不同的数据结构、内存分配、文件操作、多线程及与底层操作系统交互等。
- **XLib** [79] 是用于与某个 X server 交互的库。
- **GdkPixPuf** [14] 用于简单的客户端图像操作。
- **GDK** [17] 是 Xlib 和 GTK 之间的一个库，基本上用于绘制 widgets 及处理窗口事件，并对拖放操作进行支持。

- **Pango**[62]用于文本渲染和排版，具有强大的国际化支持。
- **ATK**, Accessibility Toolkit, 可访问工具包，用于提供 GTK+ widgets 访问支持。

有关 GTK+及相关库的更多信息请参阅该项目网站[26]，其中有 GTK+ API 文档[37]。

第 5 章 一些重要的系统服务

Maemo 平台中的底层系统服务与某些桌面 Linux 发行版本中所使用的稍有不同。本章概要介绍一些最为重要的服务。

5.1 文件系统 - GnomeVFS

Maemo 包含一个强大的文件系统框架，GnomeVFS。这个框架使应用能使用大量不同的文件访问协议，而无需了解任何有关的底层细节。获支持的协议方面的例子是：本地文件系统、HTTP、FTP 和通过蓝牙的 OBEX。

在实践中，这意味着：所有的 GnomeVFS 文件访问方法对开发伙伴和最终用户都是透明的，只要通过该框架进行文件操作。用于文件操作的 API 比起标准平台所提供的更为灵活。例如，它支持同步读写、MIME 类型支持和文件监控。

我们强烈建议在 maemo 应用中使用 GnomeVFS，事实上随 maemo OS 一起发行的很多应用就是这么做的。一个好的亲手实践起点就是研究一下 maemo 范例包中的 GnomeVFS 范例。API 方面的详细信息可以参考 GnomeVFS API 参考文献[18]。

5.2 应用参数 - GConf

GNOME desktop 环境使用 GConf 为桌面和应用储存配置设定。守护进程 GConfd 控件跟踪数据库中的变化，当其变化时，它向使用它的应用部署新的设置。控制面板应用就使用了 GConf。

5.3 D-Bus

对于进程间通信（interprocess communications, IPC），maemo 在很大程度上依赖于 D-Bus。D-Bus 能让程序导出其编程接口，以便被其他进程以一致的方式调用，而并不需要定义一个定制的 IPC 协议。使用这些导出的 API 无需知道使用哪种语言，所以只要编程语言支持 D-Bus，它也能访问这些接口。

名为 *libosso* 的 maemo 特定库为 D-Bus 通信提供了一些有用的包装程序（wrapper）。它还含有每个 maemo 应有所需要的某些功能。应用必须使用这个库进行初始化。应用可以用它来进行连接以侦听系统的硬件状态消息，如“电池电量低”。这个库也用于应用状态保存和自动保存功能。《maemo 教程》[36]很好地讲解了这个库。

5.4 其他系统服务

Maemo 平台为不同的公共任务提供许多库和 APIs。这里将讲解警报框架和 GPS API，其他一些相关服务则在本文稍后部分讲解。

5.4.1 警报框架

Maemo 中所包含的警报框架向开发伙伴们提供了一种方便使用的接口，便于处理各种定时事件。它提供一种一致的用户界面、音频播放、启动终端、发送 D-Bus 信号或执行一个文件等。文档《使用警报接口》[73]详细讲解了这个框架。

5.4.2 GPS API

Maemo 中的 *GPS* 框架由一个 *GPS daemon* 和控制它的一个库组成。《*maemo* 连接向导》[41]一文给出了一些例子，说明如何在各种应用中使用这个库。

第 6 章 多媒体

Maemo 为各种多媒体应用提供了众多的可能性。maemo 针对 Internet Tablet 终端构建，这种终端硬件提供了一个大型高解析度触摸屏、音频输入和输出、通过照相机实现的视频和图像捕获、面向数据流的快速网络连接，及一个数字信号处理器，用于高效音频和视频操作。

对开发伙伴而言，除了直接编写 DSP 之外还有一些开放的编程接口可用于利用这些功能。Maemo 中视频和音频编程的推荐方法是使用 GStreamer 框架，而对静态图像进行操作则另有一些库。针对如游戏这类应用的交互式多媒体则可用 SDL 框架来完成。文档《多媒体架构》[59]更为详尽地讲解了这些组件在物理设备和 maemo SDK 上是如何组织的。

6.1 GStreamer – 多媒体框架

Maemo 中主要的多媒体框架就是 GStreamer。它基于由众多元素组成的 pipelines 概念。元素本身实际上可以是对某个数据流进行某种操作的任何东西。

Maemo 包括许多这样的元素，用以支持音频和视频效果、编码和解码，以及与终端硬件进行交流。使用这种方案，开发伙伴们无需关心如底层硬件管理、音频和视频压缩方案等细节。如果 OS 中所包含的元素并不足够，可以为 maemo 编译更多的元素，也可以开发一些新元素。

可以在本项目网站上找到有关 GStreamer 的开发伙伴文档[23]。

6.2 数据流编码与解码

Maemo 终端中包括许多视频和音频编解码器，使各种 maemo 应用能够读写几乎所有的通用视频和音频格式。这些编解码器也得到了 GStreamer 框架的支持。许多情况下，开发伙伴甚至无需明确指定所使用的编解码器，因为 GStreamer 会自动探测格式及使用哪种编解码器。GStreamer 基础插件 *Playbin*[22]为所有音频和视频内容提供了方便的抽象层。

由于非技术原因，大部分的编解码器都没有随 SDK 提供，当开发那些有赖于这种功能的应用时最好记住这一点。

6.3 音频

对于音频编程，maemo 有两个主要的 APIs，即 GStreamer 和 ESound。通常，如向用户提示某个事件（如低电量）的系统提示音都通过 ESound 播放。更高级的操作，如播放音乐文件或录制音频，则通常应该用 GStreamer 来完成，后者提供了更好的性能和更灵活的 API。大部分 maemo 中计算密集型的 GStreamer 元素都使用终端 DSP 实现，从而大大提高了他们的效率。

Linux 内核还有两个较低级别的音频接口：ALSA 和 OSS。其中 ALSA 是通过一个插件包（为 SDK 的一部分）获得支持。而传统的 API OSS 并没有获得该内核的支持，但是 ALSA 有一个 OSS 仿真系统，它能在大多数情况下工作。

有关音频 API 方面的文档请访问 GStreamer 网站[23]、ESound 白皮书[2]和 ALSA 项目网站[1]。更详尽资料请参阅《多媒体架构》指南[59]。

6.4 视频

尽管该框架隐藏掉了许多实现和硬件方面的细节，开发者还是有必要了解接口下面所发生的事情。视频捕获是通过一个 Linux 内核的 Video4Linux API 实现的，而图像则用 X Window System 显示。

实际上所有 GNU/Linux 应用依赖于这些组件完成视频任务，因而移植现有应用轻而易举，寻求支持也很容易。

有关使用捕获功能和输出功能方面的详细指导请参阅《如何使用 Maemo 照相机》一文[30]。

6.5 数字信号处理器

在 Internet Tablet 内部有一个专门的数字信号处理器 (digital signal processor, DSP)。其设计针对如流编码和音效而作了优化。Maemo 有针对 DSP 的高级编程支持，形式为一些 GStreamer 元素，它们能对大部分获支持的文件格式进行解码。通过使用 DSP，计算载荷也被从主处理器中分离出来，从而大大提高了系统性能、响应性能和电池性能。

6.6 图形与图像

Maemo 包括一些用于图像和图形的库：

- **Cairo**，用于二维矢量图形的库。其功能如提供高质量的矢量图、支持导入及导出众多格式，如 SVG、PNG、PDF 和 Postscript，并绑定了许多编程语言。
- **GDK 及 GdkPixbuf**，GTK+ 赖以构建的位图图形库。这两个库共同提供打开及保存 JPEG、PGN、TIFF、ICO 和 BMP 格式的位图的功能，紧密集成于 GTK+，并提供通过图形原语进行绘制的工具。也支持如加载 SVG 图像等。
- 还有许多特殊的库用于对各种图像格式进行更为精细的控制，如 libpng、libjpeg，及 libwmf。

请访问 Cairo 网站[5]获取各种指南、教程，及参考资料。而在 GTK+ 项目网站[25]上则可找到有关 GDK 和 GdkPixbuf 方面的文档。

6.7 游戏

Maemo 向游戏开发伙伴们提供了一个公共的启动屏幕和设置屏幕，也提供了名为 *osso-games-startup* 的框架。游戏开发和移植者用这个库可以减少公共启动屏幕的编码量，而将精力集中于真正的游戏编程方面。*osso-games-startup* 的操作系统通信功能也提供一个接口，以便游戏的正确表现。例如，它方便了全屏幕模式和异常情形的管理工作，如以统一且用户友好的方式提示低电量。这方面的深入描述及 API 使用介绍请参阅《使用游戏启动屏幕》一文[74]。

第 7 章 通信

对于与外部世界的通信，**maemo** 平台提供了一些框架，范围从蓝牙文件传输到拨叫视频电话。本章讲述对 **maemo** 应用开发者至关重要的一些组件。

7.1 获取互联网接入

LibConIC 是所有应用希望使用互联网连接时必须使用的库。它关心如搜索可用的 **WLAN** 网络和用户在用户选择某个连接后设置 **IP** 网络。**LibConIC** API 与处理 **WLAN** 和蓝牙连接的 **maemo connectivity daemon (ICd)** 一起工作。**Maemo 连接指南**[41] 较深入地讲解了 **LibConIC** 和 **ICd**。

7.2 VoIP、即时消息和 Presence

Telepathy[69] 通信框架提供一个统一的 API，用于 presence、消息和语音/视频通话。所支持的通信协议有长长一列：IRC、ICQ、XMPP (Jabber)、SIP、MSN，等。**Telepathy** 的连接管理程序是可扩展的，其通信功能使用 **D-Bus** 实现。关于在 **Maemo** 中如何使用 **Telepathy** 请参阅《实现客户联接管理程序》[32] 中的深入讲解。

7.3 地址簿与名片夹

Maemo 平台具有用于地址簿数据的集中储存，这是使用 **Evolution Data Server (EDS)** 实现的。这使得各不相同的应用能毫不费力地共享具有标准格式的联系信息，用户无需在每个程序中对它们进行个别定义。

要从某个应用数据中访问某个 **EDS** 联系人，**maemo** 有 **libosso-abook** 库 - 它具有强大的 widgets，支持以一致的用户界面对联系人信息进行访问和管理。

如想快速掌握如何使用 **libosso-abook**，请阅读《使用 **Maemo** 地址簿 APIs》文档[75]。更多深入信息请参阅 **Maemo API 参考资料**[41] 及 **Evolution** 项目网站[10]。**Maemo-examples** 包中也包括了一些经过注解的代码，其中涉及对各种地址簿 API 的使用。

7.4 蓝牙

针对蓝牙的高级 API 作为 **maemo** 连接子系统的一部分提供。用其 **D-BUS API** 程序就可以找到远程蓝牙设备，如手机，也可以通过 **OBEX** 对象推送发送文件，及与远程设备建立配对。对于这些任务我们建议应用使用这一框架，因为它不但具有大量简单的 API，而且使应用的外观和性能具有一致性。

对于那些不获 **maemo** 连接框架支持的蓝牙操作，**maemo** 中包括了一个低级的 **BlueZ D-BUS API**，这也是面向所有 **Linux** 系统的主要的蓝牙接口。**BlueZ API** 实际上具有能用于蓝牙系统的各个方面的功能，结果是，它比更高级别的 **Maemo Connectivity** 子系统所能提供的更为复杂。

《**Maemo 连接指南**》[41] 讲述了高级 **D-BUS API** 及其使用方法。有关 **BlueZ API** 方面的更多信息可以在 **BlueZ** 网站[3]上找到。**maemo-example** 包中也包括有关这两个库的范例代码。

7.4.1 OBEX

蓝牙终端使用 **OBEX** 协议交换各种数据对象。**Maemo** 中包括一些库，以实现这个协议。对于 **OBEX FTP**，最简便的方法是使用 **GnomeVFS** 的 **OBEX** 后端。更为精细的则是 **libgwobex**，**GnomeVFS** 就将其用于自己的实现。更低级的接口就是 **OpenOBEX** 库。

有关 GnomeVFS 的更多信息请参阅本文的 GnomeVFS 一节。《Maemo 连接指南》[41]中有有关 libgwobex 和 *OpenOBEX* 的介绍章节。有关 libgOBEX API 请参阅《Maemo API 参考资料》[37]。有关 OpenOBEX 方面的更多信息可于该项目网站[60]上找到。

第 8 章 开发环境

运行于桌面的 **maemo** 开发环境被称为 **maemo SDK** [54]。它只能被安装和运行于某个 **Linux** 操作系统。针对 **maemo SDK** 的获支持的 **Linux** 发行版目前有 **Debian** 和 **Ubuntu**，但也可以在其他发行版上安装 **maemo SDK**。对于如 **Windows** 这样的其他操作系统，可以使用一个 **VMWare** 镜像[80]以提供一个可工作的 **Linux** 环境。

8.1 Maemo 软件开发工具包

Maemo SDK 在某个 **GNU/Linux** 桌面系统上主要以名为 **Scratchbox**[66]的工具创立一个沙箱开发环境。在很多方面这个环境表现为终端上的操作系统，但是加上了一些开发工具。这意味着，开发过程非常类似于一般的桌面 **GNU/Linux**，而如交叉编译这样的嵌入式开发都由 **Scratchbox** 全透明处理。

8.1.1 硬件架构

Maemo SDK 具有两种开发环境，它们针对两种架构：针对本机工作的 **x86** 和用于实际终端架构的 **ARMEL**。两者各有所长，对 **maemo** 开发都有作用。重要的是要理解：**maemo SDK** 实际上以预先配置的 **targets** 在某个有效的 **Scratchbox** 安装内部提供这两种环境。后面会详细解释这一点。

一般说来，在实际开发中会使用 **x86** 环境，因为它实际提供了与正常 **GNU/Linux** 应用相同的表现。同时，尽管底层架构与实际终端不同，程序表现通常与编译后在 **ARMEL** 上运行完全一致。

当某个应用在 **x86** 环境中能良好运行时，下一步就是为 **ARMEL** 环境对其进行编译。编译和打包的过程与在 **x86** 的完全相同，然而稍微慢一点，因为某些必需软件需要仿真。开发伙伴无需关心交叉编译。这也是 **maemo SDK** 使用 **Scratchbox** 的最主要理由。

在 **ARMEL** 环境中编译的应用可以直接放到终端上运行。而且，还有可能在 **maemo SDK** 的 **ARMEL** 环境内运行这些应用。这是因为 **maemo SDK** 提供了自动仿真功能。这种仿真还不完备，所以实际测试还是必须在终端上完成。

由于性能表现的原因，在整个开发过程中使用仿真还不十分理想。这解释了用 **Scratchbox** 在可能情况下无仿真地使用主机电脑中的工具，从而在 **ARMEL** 环境中获得良好的本机性能。例如，用一个 **x86-ARMEL** 交叉编译器在 **ARMEL** 环境中编译，但是 **Scratchbox** 隐藏了一些细节，所以开发伙伴们能如在任何 **GNU/Linux** 系统上一样执行 **GCC**。

8.1.2 在 maemo SDK 上开发

Maemo SDK 在 **Scratchbox** 内提供了全部开发工具。而且，**Hildon Desktop** 也是以 **Scratchbox** 内部的一条单一命令 **af-sb-init start** 启动的。然而，**Hildon Desktop** 需要显示在一定大小的第二个 **X server** 上。

这个规则的例外是：必须在主机 **Linux** 环境中，而不是在 **Scratchbox** 中，启动诸如 **Xephyr** [78]这样的 **X server**。**Xephyr** 的使用方法在《**maemo SDK** 教程》[36]中介绍。

8.1.3 Scratchbox 中的开发工具

由于 **Scratchbox** 环境实际上是一个完整的 **GNU/Linux** 系统，它包含有标准的 **GNU/Linux** 开发工具。调试用 **GDB**[13]、**valgrind**[76]、**ltrace** 和 **strace**[68]这类工具实现。性能分析可以用诸如 **htop**[31]、**oprofile**[61]及 **time** 这类工具实现。而编译则使用 **GCC** 工具链。这其中的某些工具所提供的图形化用户界面也可以得到利用。当然这并非完整的工具列表，如果 **SDK** 随附的工具不能满足个人偏爱，那么其大部分的 **utilities** 无需改变就可以方便地用在 **Scratchbox** 中。

《Maemo 教程》[36]很好地介绍了如何使用 *Scratchbox*，而《Memo 调试指南》[42]则深入讲解了各种调试工具。

8.1.4 终端上测试与调试

尽管 *Scratchbox* 能非常精确地仿真终端上的某个完整目标环境，它们却并非 100%相同。特别对于那些使用了终端特定硬件的应用，其在终端上的表现会不同于其在 *Scratchbox* 上的。幸运的是，可以使用 *SSH* 或名为 *sbrsh* 的工具从 *Scratchbox* 在终端上透明地运行目标二进制文件，因而在终端上测试软件非常直观。

《CPU 透明性指南》[6]讲述了 *sbrsh* 的入门。可以从 *maemo* 网站[48]下载 *SSH* 服务器和客户端。

8.1.5 Scratchbox 深入

Scratchbox 是 *maemo SDK* 的交叉编译环境。*Scratchbox* 的默认安装在大多数条件下都能如常工作，但对于特定使用情形就有必要了解某些细节。

Scratchbox 内部的 *target* 含有一个根文件系统，当在 *Scratchbox* 内创建了一个新 *target* 时，必须为之定义一个 *toolchain*。应用就是使用这个 *toolchain* 为 *target* 构建的。某个 *target* 的实例就是 *X86* 和 *ARMEL*，它们都是由 *maemo SDK* 在 *Scratchbox* 之上提供的。

Host Tools（主机工具）对于主机是本地，所以方便和快速，它们总是优先于 *target tools*（目标工具），且对于目标架构（如交叉编译应用）透明。主机工具由开发包和 *toolchains*（工具链）组成。

某个工具链提供最小集合的工具用于为目标终端编译二进制文件。必须针对每个 *Scratchbox* 目标选择一个且只能选择一个工具链。

*CPU 透明性方法*以对用户透明的方式，使应用在某个模拟器、目标终端，或直接在主机上运行。一些可用的 *CPU 透明性方法*来自某个特殊的 *devkit*，名为 *cputransp*。对于每个 *maemo SDK* 的预定义目标选择并定义一个 *CPU 透明性方法*。

一个 *工具链*是用于为目标环境生成二进制文件的工具集合。除了一个编译器（*gcc*）之外它还含有一个连接器（*ld*）和其他一些 *binutils*，如 *strip*、*objdump* 和 *strings*。

devkit 是相对主机为本地的工具的集合。可以针对某个目标选择或禁用某个工具包。*devkit* 的例子是 *doctools devkit*，它提供了构建文档的工具（如 *doxygen*）。

rootstrap 是针对目标终端的一个根文件系统。*Maemo SDK* 在 *Scratchbox* 内部为两种目标（*X86* 和 *ARMEL*）提供根文件系统。注意：用户的主目录被所有目标共享。

从 *Scratchbox* 的视角看，*Maemo SDK* 是一套预配置的 *目标*和 *根文件系统*，它既面向 *X86* 也面向 *ARMEL* 架构，位于一个有效的 *Scratchbox* 安装之上。

有关 *Scratchbox* 的更多信息请参阅 *Scratchbox* 网站[66]。

8.2 打包、部署及分发

对于安装和管理应用包，*maemo* 使用了流行的 *Debian* 包管理系统。从用户的角度看，这一系统是透明的，因为包管理是使用 *Application Manager*（应用管理器）完成的。在其内部有一个灵活的包框架，它使开发伙伴们能方便地创建可安装可管理的包，而无需关注管理的实际细节。

8.2.1 Deb 安装包

Debian 包管理系统使用 **deb-packages**，除了将被安装的文件之外，它还具有对包进行描述的 **meta data**、与其他包的依存关系，及可选的安装和删除脚本。使用各种各样的包开发工具，这些包的创建过程非常简单。

有关包创建过程在文档《制作安装包》[56]中讲述。

8.2.2 包仓库

包管理系统使用包仓库，它们基本上是 **web** 或 **FTP** 站点，含有各种包及其信息。并不一定要求某个包出现在任何仓库中，但这样做却有明显的好处：用户可以方便地对包进行更新，其他开发者能以自动依存关系使用这些包，而包仓库中的包则可以使用应用管理器找到。

有关包仓库方面的信息请参阅《Debian 仓库手册》[8]。

8.2.3 一键安装

Maemo 还有一个选项，用于创建安装指令文件，它使用户能在某个网站上仅仅通过点击一个链接来安装应用。这些文件的格式非常简单，具体讲解请参阅《Hildon 应用管理器文档》[28]。

8.3 编程语言

目前，**C** 是 **maemo** 的官方编程语言。由于社区的努力，还存在对其他一些语言的非官方支持。这方面的例子是：**SDK** 本身可以编译 **C++**，且通过添加 **hildonmm bindings** [29]就可以以 **C++**方式创建 **Hildon** 应用。**Python** 脚本语言也以 **pymaemo**[63]的形式得到了很好的支持，而 **Ruby bindings** 正在开发中 [65]：不要忘记 **Mono**[58]。对于 **Java** 支持，**JaliMo**[33] 就是一个值得关注的有趣项目。

第 9 章 移植软件

Maemo 平台的设计充分考虑到了向移动 **maemo** 环境方便移植普通的 **GNU/Linux** 桌面软件。本文前面部分已经讲到了一些基本的工具，它们能方便交叉编译且有助于与某些 **GNU** 自动工具打交道。本章就有关应用编程和用户界面方面的差异性给出了一些指导性意见。

9.1 命令程序

移植无用户界面的软件在很多情况下是微不足道且十分直观的。首先，将某个程序的源代码解包到一个 **Scratchbox** 用户的主目录中。接着，在 **ARMEL** 目标中运行 **configure** 和 **make**。然后，可以在终端上测试已被编译的程序。最后，软件需要被打包。

9.2 从 GTK+到 Hildon

对于使用 **GTK+**的图形化用户界面应用，其移植过程同上面所说的一样，但是还不够。最容易注意到的第一件事情就是虚拟键盘不见了。应用用户不能与应用交互了。

简单地说，用户界面部分需要用 **Hildon** 而不是用 **GTK+**重新规划。对于 **GNOME** 组件的依存，如果有的话，需要被删除或用相应的 **maemo SDK** 组件替换。如果该应用使用了在 **maemo SDK** 中所没有的任何组件，这部分也需要由开发伙伴进行移植。

《**Maemo 4.0** 移植指南》[35] 介绍了向 **maemo** 环境移植某个现有的 **GTK+**应用时必须考虑的详细信息。

9.3 本地化

Maemo 应用的本地化是用通用 **gettext** 包完成的。将一个应用翻译到不同的语言在《本地化手册》[50] 中介绍。

第 10 章 品质考虑

Maemo 开发伙伴在开发一个高质量应用时需要关注许多细节性问题。比较桌面 **Linux**，开发者更需要关心性能表现、资源使用、电源消耗、安全和使用性等问题。安全认知项目[64]向开发伙伴们提供了一般性的测试计划，帮助他们创建优质软件。

10.1 性能与响应

用户希望处于可控状态：终端必须在任何时候都能响应，给出适当的反馈。可以通过如下三种方式达到 UI 响应性：第一，使用进程事件在操作中显示一个标志并阻塞 UI。第二个选项是将操作分成几部分，同时更新比如一个进度条。第三，在 Glib 的帮助下达到线程化，但这将越来越难于实现和调试。对于某些耗时任务我们推荐第二个选项。如果操作只需一两分钟，那么可以使用第一个选项。

作为其他方面的一般性指引：避免 CPU 过度繁忙也是很自然的。这也适用于那些完全冗余的操作，如不必要的屏幕刷新。

10.2 资源使用

必须非常小心地考虑 maemo 应用的资源使用问题，因为一个应用如果占用太多的资源，就不实用。紧张的资源有 RAM、flash、网络带宽、文件描述符，及 IO 带宽等。此外，除了基本内存占用之外，防止内存泄漏也是很自然的事情。有些工具，如 Valgrind[76]有助于探测可能的内存泄漏。

10.3 电源消耗

如果用户始终需要对终端充电，或者终端待机时电源耗尽，该终端将不再能使用。因此，在任何地方都不能使用轮询或忙等待。相反，基于事件的方案，如 GTK 的主循环则是较好的替代方案。此外，必须避免过于频繁的定时器或超时。

10.4 安全的软件设计

品质与强壮性一起构成安全的软件。具体讲述在 maemo 平台上进行安全软件设计的文档[67]给出了有关在移动平台上创建安全软件及一些特殊问题的详细信息。

第 11 章 maemo.org

面向 maemo 开源社区的开发者之家网站是 maemo.org。它向开发伙伴提供文档、社区支持、bug 报告和完整的项目托管。本章介绍这些服务。

11.1. 文档

maemo.org 网站为每一个 maemo 主发行版提供官方文档[43], 其形式有教程、指南和 API 文档。本篇快速指南也是官方文档的一部分。

所有人都可以阅读社区文档, 而在 maemo wiki [55] 中注册过的用户则可以对这些文档进行编辑。我们感谢对社区 wiki 的贡献及对有关官方文档的反馈意见!

11.2. 支持

从社区获得支持的场所是面向 maemo 开发者的邮件列表[51]。在向邮件列表提出可能的常见问题前, 最好先到邮件列表存档[44]中查阅一下。面向 maemo 用户的列表也非常活跃, 比起应用开发者, 它更以最终用户为目标。

至本文撰写为止, 最为活跃的 maemo 即时消息渠道是位于 *freenode* 的 maemo IRC 渠道 [49]。

对 maemo 平台的商业支持由 *诺基亚论坛*[12] 提供。

11.3. 报告 Bug

对 maemo 平台提供反馈意见的合适方式是使用公共 maemo Bugzilla [40]。它可以被匿名浏览, 但需要注册以便提交新 bug 报告或功能需求。同样的问题管理系统用于反馈终端上有关软件运行、SDK, 及网站 (包括其文档) 等方面的问题。

11.4. 项目托管

[maemo garage](http://maemo-garage.org) [46] 是面向 maemo 社区的完整项目托管网站。这是 maemo 应用实际开发非常适宜的地方, 名至实归。Garage 向开发伙伴们提供针对每个项目的版本控制管理系统、wiki、问题追踪和论坛。

Maemo 网站的 *应用目录*[45] 或 *下载专区* 专门让最终用户方便地下载并在其 Internet Tablet 上安装开源软件产品。Maemo 应用目录菜单[38] 中有关于维护者如何向应用目录列表中添加一个最新测试应用的指示, 希望能将其放在 garage 的第一个位置。

11.5. 路线图与新闻

Maemo 社区成员能通过阅读新闻[52] 与当前进展保持同步。有关未来计划的路线图[53] 永远向大家开放, 目的是避免重复工作, 也让大家能了解 maemo 本身的开发方向。

参考文献

- [1] Alsa 项目主页。 <http://www.alsa-project.org/>。
- [2] 面向arm架构的应用二进制接口 (abi) 。
<http://www.arm.com/products/DevTools/ABI.html>
- [3] Bluez项目主页。 <http://www.bluez.org/>。
- [4] Busybox。 <http://www.busybox.net/>。
- [5] Cario项目主页。 <http://cairographics.org/>。
- [6] Cpu透明性指南。 http://maemo.org/development/documentation/how-tos/3-x/howto_cpu_trans_bora.html。
- [7] Debian-统一的操作系统。 <http://www.debian.org/>。
- [8] Debian仓库指南。 <http://www.debian.org/doc/manuals/repository-howto/repository-howto.en.html>。
- [9] Esound白皮书。 <http://developer.gnome.org/doc/whitepapers/esd/>。
- [10] Evolution项目主页。 <http://www.gnome.org/projects/evolution/>
- [11] 扩展hildon的输入方法。 http://maemo.org/development/documentation/how-tos/4-x/extending_hildon_input_methods.html。
- [12] 诺基亚论坛。 <http://www.forum.nokia.com/>。
- [13] Gdb: gnu项目的debugger。 <http://sourceware.org/gdb/>。
- [14] Gdkpixbuf库。 <http://library.gnome.org/devel/gdk-pixbuf/>。
- [15] Glib 参考手册。 <http://developer.gnome.org/doc/API/glib/>。
- [16] Gnome mobile。 <http://www.gnome.org/mobile/>。
- [17] Gnome; 免费软件桌面项目。 <http://www.gnome.org/>。
- [18] Gnomevfs api 参考。 <http://library.gnome.org/devel/gnome-vfs-2.0/stable/>。
- [19] Gnu c 库, <http://www.gnu.org/software/libc/>。
- [20] Gnu核心工具。 <http://www.gnu.org/software/coreutils/>。
- [21] GObject 参考手册。 <http://library.gnome.org/devel/gobject/>。
- [22] Gstreamer playbin base plugin。
<http://gstreamer.freedesktop.org/data/doc/gstreamer/head/gst-plugins-base-plugins/html/gst-plugins-base-plugins-playbin.html>。
- [23] Gstreamer 项目主页。 <http://gstreamer.freedesktop.org/>。
- [24] Gtk+ api 文档。 <http://www.gtk.org/api/>。
- [25] Gtk+ 项目主页。 <http://www.gtk.org/>。

- [26] Gtk+ 网站。 <http://www.gtk.org/>。
- [27] GHildon。 <http://live.gnome.org/Hildon/>。
- [28] Hildon 应用管理器文档。 <http://hildon-app-mgr.garage.maemo.org/doc.html>。
- [29] Hildonmm 项目网页。 <https://garage.maemo.org/projects/maemomm/>。
- [30] 如何使用camera api。 <http://maemo.org/development/documentation/how-tos/4-x/how-to-use-camera-api.html>。
- [31] htop - linux交互进程查阅器。 <http://htop.sourceforge.net/>。
- [32] 实现客户联接管理程序。 <http://maemo.org/development/documentation/how-tos/4-x/implementing-custom-connection-managers.html>。
- [33] Jalimo。 <http://www.jalimo.org/>。
- [34] maemo内核指南。 <http://maemo.org/development/documentation/how-tos/4-x/kernel-guide-for-maemo.html>。
- [35] maemo 4.0 移植指南。 <http://maemo.org/development/documentation/how-tos/4-x/maemo-4-0-porting-guide.html>。
- [36] Maemo 4.0 教程。 <http://maemo.org/development/documentation/tutorials/maemo-4-0-tutorial.html>。
- [37] Maemo api 参考。 <http://maemo.org/development/documentation/apis/>。
- [38] maemo 应用目录手册。 <http://maemo.org/community/application-catalog/user-manual.html>。
- [39] maemo 架构。 <http://maemo.org/development/documentation/how-tos/4-x/maemo-architecture.html>。
- [40] maemo bugzilla。 <http://bugs.maemo.org>。
- [41] Maemo 连结性指南。 <http://maemo.org/development/documentation/how-tos/4-x/maemo-connectivity-guide.html>。
- [42] Maemo 调试指南。 <http://maemo.org/development/documentation/how-tos/4-x/maemo-debugging-guide.html>。
- [43] maemo 开发伙伴文档。 <http://maemo.org/development/documentation/>。
- [44] maemo开发者邮件列表归集。 <http://lists.maemo.org/pipermail/maemo-developers/>。
- [45] maemo 下载。 <http://maemo.org/downloads/>。
- [46] maemo garage。 <http://garage.maemo.org>。
- [47] maemo-gtk+ 2.10。 <http://live.gnome.org/Maemo/Gtk210Changes>。
- [48] maemo 主页。 <http://maemo.org/>。
- [49] maemo irc 渠道。 <http://maemo.org/community/irc.html>。
- [50] maemo 本地化指南。 <http://maemo.org/development/documentation/how-tos/4-x/maemo-localization-how-to.html>。

- [51] maemo 邮件列表。 <http://maemo.org/community/maemo-mailing-lists.html>。
- [52] maemo 新闻。 <http://maemo.org/news/>。
- [53] maemo 路线图。 <http://maemo.org/intro/roadmap.html>。
- [54] maemo sdk 发行版。 <http://maemo.org/development/sdks/>。
- [55] maemo wiki。 <http://maemo.org/community/wiki>。
- [56] 制作应用包。 <http://maemo.org/development/documentation/how-tos/4-x/making-application-packages.html>。
- [57] Matchbox 主页。 <http://matchbox-project.org/>。
- [58] Mono。 <http://www.mono-project.com/Maemo>。
- [59] 多媒体架构。 <http://maemo.org/development/documentation/how-tos/3-x/multimedia-architecture.html>。
- [60] Openobex 主页。 <http://dev.zuckschwerdt.org/openobex/>。
- [61] Oprofile - linux系统性能分析器。 <http://oprofile.sourceforge.net/>。
- [62] Pango 参考手册。 <http://library.gnome.org/devel/pango/>。
- [63] Pymaemo 主页。 <http://pymaemo.garage.maemo.org/>。
- [64] 品质认知。 <http://maemo.org/development/documentation/how-tos/4-x/quality-awareness.html>。
- [65] Ruby maemo bindings 项目网页。 <https://garage.maemo.org/projects/ruby185/>。
- [66] Scratchbox 主页。 <http://scratchbox.org/>。
- [67] 安全软件设计。 TODOnotpublishedwithmaemoChinookBETA。
- [68] Strace。 <http://sourceforge.net/projects/strace/>。
- [69] Telepathy。 <http://telepathy.freedesktop.org/>。
- [70] 面向debian 开发者的Ubuntu。 <https://wiki.ubuntu.com/UbuntuForDebianDevelopers>。
- [71] Ubuntu 主页。 <http://www.ubuntu.com/>。
- [72] Uclibc。 <http://www.uclibc.org/>。
- [73] 使用警报接口。 <http://maemo.org/development/documentation/how-tos/4-x/using-alarm-interface.html>。
- [74] 使用游戏启动屏幕。 <http://maemo.org/development/documentation/how-tos/4-x/using-games-start-up-screen.html>。
- [75] 使用maemo 地址簿apis。 <http://maemo.org/development/documentation/how-tos/3-x/howto-using-a-book-bora.html>。
- [76] Valgrind。 <http://valgrind.org/>。

- [77] 为maemo 编写hildon 桌面插件。 http://maemo.org/development/documentation/how-tos/4-x/writing_hildon_desktop_plug-ins_for_maemo.html。
- [78] Xephyr。 <http://www.freedesktop.org/wiki/Software/Xephyr>。
- [79] Xlib 手册。 <http://www.the-labs.com/X11/XLib-Manual/>。
- [80] 安装maemo sdk 的Xubuntu镜像。 https://garage.maemo.org/frs/?group_id=277。

请对本资源进行评价

请花一点时间[对本资源评分](#)，帮助我们改进文档质量，也让我们了解哪些资源对你最为有价值。