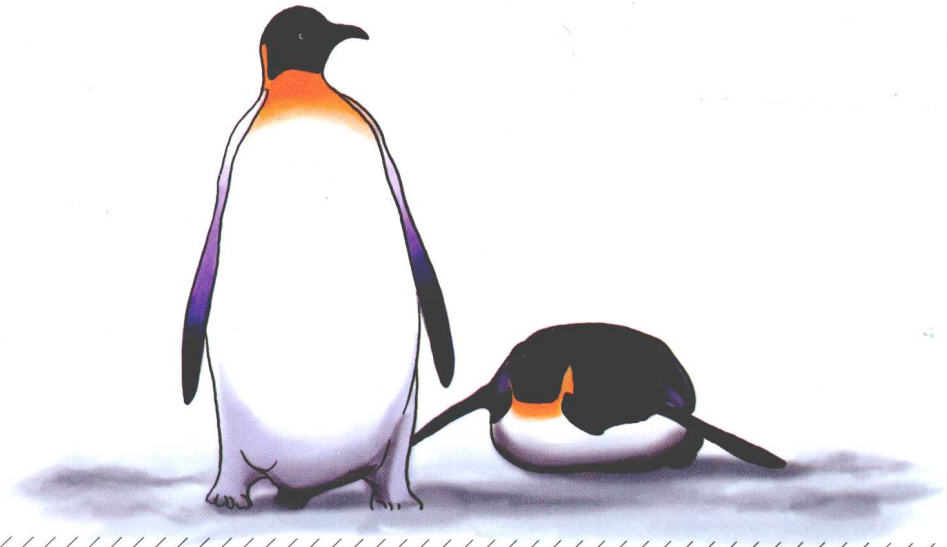

精通嵌入式 Linux编程

——构建自己的GUI环境

李玉东　李玉萍　编著

北京航空航天大学出版社

精通嵌入式 Linux 编程

——构建自己的 GUI 环境

李玉东　李玉萍　编著

北京航空航天大学出版社

内 容 简 介

本书针对使用 Linux 构建嵌入式系统的一个关键环节——图形用户界面(GUI),首先讲述了 Linux 编程的高级技巧,包括多进程、多线程等技术;然后通过实例重点讲述了窗口系统的基本知识与实现技巧,为读者开发自己的面向嵌入式 Linux 的 GUI 环境提供了一个参考实现范例。重点包括:LGUI 多窗口的设计与实现、LGUI 的消息管理、窗口与无效区的管理、设备上下文与图形设备接口的设计与实现等。

本书适用于使用 Linux 构建嵌入式系统的软件工程师以及希望深入了解窗口系统实现原理的读者。

图书在版编目(CIP)数据

精通嵌入式 Linux 编程:构建自己的 GUI 环境/李玉东,李玉萍编著. --北京:北京航空航天大学出版社,2010.5

ISBN 978-7-5124-0066-5

Ⅰ.①精… Ⅱ.①李… ②李… Ⅲ.①Linux 操作系统—程序设计 Ⅳ.①TP316.89

中国版本图书馆 CIP 数据核字(2010)第 065459 号

精通嵌入式 Linux 编程
——构建自己的 GUI 环境

李玉东 李玉萍 编著

责任编辑 刘彦宁 杨 昕

*

北京航空航天大学出版社出版发行

北京市海淀区学院路 37 号(邮编 100191) http://www.buaapress.com.cn

发行部电话:(010)82317024 传真:(010)82328026

读者信箱:bhpress@263.net 邮购电话:(010)82316936

北京市媛明印刷厂印装 各地书店经销

*

开本:787×960 1/16 印张:13.75 字数:308 千字

2010 年 5 月第 1 版 2010 年 5 月第 1 次印刷 印数:4 000 册

ISBN 978-7-5124-0066-5 定价:28.00 元

前 言

第一个问题，为什么要写这本书？

现在很多面向嵌入式 Linux 编程的书籍理论性很强，但并不针对某一领域，对于解决某一领域的特定问题指导性较弱。

利用嵌入式 Linux 来构建系统，应用于便携式终端产品的居多，就软件方面而言，这类产品主要有两个环节需要把握：一是 Linux 内核（包括驱动）的移植；二是 GUI（Graphical User Interface，即图形用户界面）层与应用层软件的设计。本书就是基于后一个环节展开的。

有人说，嵌入式 Linux 的最大问题是其 GUI 没有统一标准。但不知道这是它的缺点还是优点，是否应该由如 Microsoft 或 Nokia 这种级别的公司在这个操作系统平台上构建一个全世界都一样的用户界面，然后大家都用它的 API 来开发应用程序呢？这个问题暂且不讨论。嵌入式产品对于界面的需要千差万别，MP3、MP4、导航仪、电视机顶盒、手机等五花八门。如果所有的界面都从“开始”菜单开始，操作起来不一定都很方便，另外，操作方式是用手指、遥控器、鼠标还是别的什么东西是可以选择的，因此，对于嵌入式产品，作者认为个性化用户界面才是合适的，任何一个 GUI 都不可能有如此好的适应性和可配置性，把一个 PDA 风格的 GUI 系统移植到机顶盒上，或把一个手机风格的 GUI 移植到工控机里都是没有意义的。解决问题的最好办法，就是自己构建一个小型的 GUI 环境，只针对具体应用，与其他系统无关。

那么，可能有人会说，量体裁衣，开发一个适合于自有项目的 GUI 环境固然很好，但这会不会很复杂，是不是会使项目周期拉长呢？本书可以告诉你，开发一个小型的嵌入式 GUI 系统其实很容易！何况网络上有如此之多的开源代码可供参考。当然，无偿复制开源软件用于商业目的是不允许的。但人们的思想是自由的，这一点谁也否认不了。

另外，现在已经开发完并开源的面向嵌入式 Linux 的 GUI 系统固然很多，而且还有一些人又在开发这个“柜”、那个“柜”的，但没有人仔细讨论到底一个嵌入式 Linux GUI 系统的体系结构如何能让使用者从全局把握系统，从而开发出自己的 GUI 环境，作者认为这是“授之以鱼”还是“授之以渔”的问题。

所以，作者写了这本书，通过嵌入式 Linux 特定环节的应用实例，即中间件层的 GUI 软件，来阐释 Linux 开发，同时使读者对于消息驱动的、轻量级窗口系统的实现有较为彻底的理解。

第二个问题，这本书有什么特点？

本书只针对GUI这个环节讨论技术问题，讨论其如何在嵌入式Linux上实现，并用到了Linux开发的技术细节，所以本书第一个特点是针对性强。

另外，作者不想把这本书搞成一个Linux编程的百科全书，讲清楚一个问题是最重要的，所以本书第二个特点是精炼。

本书出版之前，其早期版本作者一直放在网站上，有很多人下载并在网络上传播。这样做的目的不为赚钱，只希望对大家都有所帮助。

由于时间有限，书中可能还存在一些错误。另外，嵌入式Linux以及GUI技术的飞速发展，使得书中提到的一些概念有可能不再新颖，或者其中提到的GUI的实现方法不见得适用于任何项目。但通过本书，可以了解到作者对于一个小型窗口系统的实现思路。如果书中提到的概念或用于示例的LGUI实现代码有任何错误，欢迎读者批评指正。

本书能够顺利完成并出版，得到了很多老师与朋友的帮助，首先要感谢我的导师——北京大学人机交互与多媒体实验室的王衡副教授，她给了我大量的指导与帮助。另外，奚小君、秦艺丹、李夏、李文阳、王文翩、刘彦军、夏华、刘成功、王成、郑浩、张明、张小铃、李志华、赵处一、王成明、李自忠、秦仕军、范佳新、王小良等朋友也给予了大力支持，在此向他们表示感谢！

李玉东

2010年1月

目 录

第1章 概 论

没有必要从 Linux 的来源说起。首先，介绍 GUI 在嵌入式 Linux 系统中所处的位置以及目前有哪些系统可供借鉴。

1.1 嵌入式系统的基本概念

在目前日益信息化的社会中，计算机与网络已经渗透到人们日常生活的每一个方面，而嵌入式系统，正是这个过程的主要推动力量。与人们生活息息相关的家用电器、汽车电子、随身携带的手机、MP3、手表、PDA、数码相机、数码摄像机，这一切都与嵌入式系统密切相关；而在工业领域，使用嵌入式设备控制的生产流水线、数字机床、智能工具也扮演着极其重要的角色。

嵌入式系统的一般定义是：以应用为中心、以计算机技术为基础、软件硬件可裁减，适应应用系统对功能、可靠性、成本、体积、功耗严格要求的专用计算机系统。

广义上讲，凡是带有微处理器的专用软硬件系统都可以称为嵌入式系统。所以也有人说："嵌入式系统是将操作系统和功能软件集成于计算机硬件系统之中。"

狭义上讲，人们更强调的是那些使用嵌入式微处理器构成独立系统，具有自己的操作系统并且具有某些特定功能的系统。

1.2 嵌入式系统的特征

与通用计算机不同，嵌入式系统是针对具体应用的专用系统，一般具有成本敏感性，它的硬件和软件必须高效地设计，好的嵌入式系统是完成目标功能的最小系统。

嵌入式系统一般要求高可靠性，例如在高温、高压、电磁干扰严重的工业环境中，就对嵌入式系统有很高的要求。

嵌入式处理器的功耗、体积、处理能力在具体应用中也有很高的要求，这在消费类电子产品方面的表现非常明显。嵌入式处理器要针对用户的具体需求，对芯片配置进行裁减和添加，才能达到理想的效果。

嵌入式系统软件和嵌入式应用软件也与通用计算机软件有所不同。一般嵌入式软件要求

高质量的代码与高可靠性。另外，许多嵌入式应用系统，要求系统软件具有实时处理能力。在多任务嵌入式系统中，对重要性各不相同的任务进行统筹兼顾的合理调度是保证每个任务及时执行的关键。

1.3 选择 Linux 构建嵌入式系统

Linux 从 1991 年问世到现在，短短的十几年时间已经发展成为功能强大、设计完善的操作系统之一，不仅可以与各种传统的商业操作系统分庭抗争，在新兴的嵌入式操作系统领域内也获得了飞速发展。

嵌入式 Linux 的开发和研究是操作系统领域中的一个热点，目前已经开发成功的嵌入式系统中，大约有一半使用的是 Linux。Linux 之所以能在嵌入式系统市场上取得如此辉煌的成果，与其自身的优良特性是分不开的。

① 广泛的硬件支持。Linux 能够支持 x86、ARM、MIPS、ALPHA、PowerPC 等多种体系结构，目前已经成功移植到数十种硬件平台，几乎能够运行在所有流行的 CPU 上。Linux 有着异常丰富的驱动程序资源，支持各种主流硬件设备和最新硬件技术，甚至可以在没有存储管理单元(MMU)的处理器上运行，这些都进一步促进了 Linux 在嵌入式系统中的应用。

② 内核高效稳定。Linux 内核的高效和稳定已经在各个领域内得到了大量事实的验证。Linux 的内核设计非常精巧，分成进程调度、内存管理、进程间通信、虚拟文件系统和网络接口五大部分，其独特的模块机制可以根据用户的需要，实时地将某些模块插入到内核或从内核中移走。这些特性使得 Linux 系统内核可以裁减得非常小巧，很适合嵌入式系统的需要。

③ 开放源码，软件丰富。Linux 是开放源代码的自由操作系统，它为用户提供了最大限度的自由度；由于嵌入式系统千差万别，往往需要针对具体的应用进行修改和优化，因而获得源代码就变得至关重要了。Linux 的软件资源十分丰富，每一种通用程序在 Linux 上几乎都可以找到，并且数量还在不断增加。在 Linux 上开发嵌入式应用软件一般不用从头做起，而是可以选择一个类似的自由软件作为原型，在其上进行二次开发。

④ 优秀的开发工具。开发嵌入式系统的关键是需要有一套完善的开发和调试工具。传统的嵌入式开发调试工具是在线仿真器 ICE(In - Circuit Emulator)，它通过取代目标板的微处理器，给目标程序提供一个完整的仿真环境，从而使开发者能够掌握程序在目标板上的工作状态，便于监视和调试程序。但是，在线仿真器的价格非常昂贵，而且只适合做非常底层的调试。如果使用的是嵌入式 Linux，且软硬件能够支持正常的串口功能，则即使不用在线仿真器也可以很好地进行开发和调试工作，从而节省了一笔不小的开发费用。嵌入式 Linux 为开发者提供了一套完整的工具链(tool chain)，它利用 GNU 的 gcc 做编译器，用 gdb、kgdb、xgdb 做调试工具，能够很方便地实现从操作系统到应用软件各个级别的调试。

⑤ 完善的网络通信和文件管理机制。Linux 从诞生之日起就与 Internet 密不可分，支持

所有标准的 Internet 网络协议，并且很容易移植到嵌入式系统当中。此外，Linux 还支持 ext2、fat16、fat32、romfs 等文件系统，这些都为开发嵌入式系统应用打下了很好的基础。

1.4 GUI 在嵌入式 Linux 系统中的地位及要求

随着近年来手持式和家用型消费类电子产品的发展，人们对这些产品的用户界面产生了新的需求，例如：手机、PDA、便携式媒体播放器、家庭多媒体娱乐中心、数字机顶盒、DVD 播放器等产品。以前，这类产品的用户界面都比较简单，而现在可以看到，大部分产品都需要有漂亮的图形用户界面，甚至要求能够支持全功能的浏览器，使得用户能够随时随地进行网络信息的浏览。但是，由于消费类电子产品的成本敏感性，这些产品大多数希望建立在一个有限占用系统资源的轻量级 GUI 系统之上，这与 PC 中 GUI 系统有根本性的区别。

另外，一个轻量级 GUI 系统的需求存在于工业控制领域，由于工业控制领域对实时性的要求比较高，所以也不希望这些系统建立在庞大的、响应迟缓的 GUI 系统之上。尤其是在实时 Linux 系统出现以后，由于 Linux 系统的稳定性、可靠性、易移植性以及其广泛的软硬件支持，Linux 系统在工业领域也得到越来越多的应用，而一个轻量级的 GUI 系统也正是这类系统所需要的。

从用户的观点来看，GUI 是系统的一个至关重要的方面。用户通过 GUI 与系统进行交互，所以 GUI 应该易于使用并且非常可靠，而且它还需要有内存意识，可以在内存受限的微型嵌入式设备上运行。

从二次开发者的角度看，GUI 是一个友好的开发环境，开发者无需经过艰苦的学习就能适应开发过程。这样可以使得基于此平台的应用很快地丰富起来。对于二次开发商而言，也才有兴趣使用此产品，为终端产品制造商提供解决方案。

另外，必须清楚的是，嵌入式系统往往是一种定制设备，它们对 GUI 的需求也各不相同。有的系统只要求一些图形功能，而有些系统要求完备的 GUI 支持。因此，GUI 也必须是可定制的。

从系统的体系结构来看，GUI 系统属于应用层的软件系统，但通常而言，GUI 有别于一个简单的图形库，一个 GUI 系统通常会有自己的应用开发模式，从这个意义上讲，GUI 应该属于中间件的范畴。

GUI 在整个系统中所处的位置如图 1-1 所示。

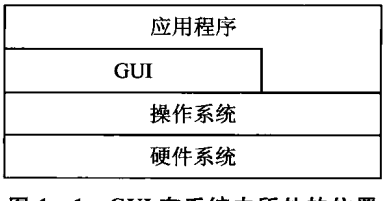

图 1-1 GUI 在系统中所处的位置

1.5 用户界面概况

1.5.1 用户界面的历史

计算机用户界面是指计算机与其使用者之间的对话窗口，是计算机系统的重要组成部分。计算机的发展史不仅是计算机本身处理速度和存储容量飞速提高的历史，而且是用户界面不断改进的历史。早期的计算机是通过面板上的指示灯来显示二进制数据和指令，人们则通过面板上的开关、扳键及穿孔纸带送入各种数据和命令。20世纪50年代中、后期，由于采用了作业控制语言(JCL)及控制台打字机等，使计算机可以批处理多个计算任务，从而代替了原来笨拙的手工扳键方式，提高了计算机的使用效率。

1963年，美国麻省理工学院在709/7090计算机上成功地开发出第一个分时系统CTSS，该系统连接了多个分时终端，并最早使用了文本编辑程序。从此，以命令行形式对话的多用户分时终端成为20世纪70年代乃至80年代用户界面的主流。

20世纪80年代初，由美国Xerox公司Alto计算机首先使用的Smalltalk-80程序设计开发环境，以及后来的Lisa、Macintosh等计算机，将用户界面推向图形用户界面的新阶段。随之而来的用户界面管理系统和智能界面的研究均推动了用户界面的发展。用户界面已经从过去的人去适应笨拙的计算机，发展到今天的计算机不断地适应人的需求。

用户界面的重要性在于它极大地影响了最终用户的使用，影响了计算机的推广应用，甚至影响了人们的工作和生活。由于开发用户界面的工作量极大，加上不同用户对界面的要求也不尽相同，因此，用户界面已成为计算机软件研制中最困难的部分之一。当前，Internet的发展异常迅猛，虚拟现实、科学计算可视化及多媒体技术等对用户界面提出了更高的要求。

1.5.2 图形用户界面的特征

图形用户界面(GUI)的广泛流行是当今计算机技术的重大成就之一，它极大地方便了非专业用户的使用，人们不再需要死记硬背大量的命令，而可以通过窗口、菜单，方便地进行操作。Visual已成为当前最流行的形容词，如Visual Basic、Visual C++等。为什么图形用户界面受到如此青睐？它的主要特征是什么？

1. WIMP

其中：

W(Windows)指窗口，是用户或系统的一个工作区域。一个屏幕上可以有多个窗口。

I(Icons)指图符，是形象化的图形标志。

M(Menu)指菜单，可供用户选择的功能提示。

P(Pointing Devices)指鼠标等，便于用户直接对屏幕对象进行操作。

2. 用户模型

GUI采用了不少桌面办公的隐喻，使应用者共享一个直观的界面框架。由于人们熟悉办公桌的情况，因而对计算机显示的图符含义容易理解，诸如：文件夹、收件箱、画笔、工作簿、钥匙及时钟等。

3. 直接操作

过去的界面不仅需要记忆大量命令，而且需要指定操作对象的位置，如行号、空格数、X及Y的坐标等。采用GUI后，用户可直接对屏幕上的对象进行操作，如拖动、删除、插入以至放大和旋转等。用户执行操作后，屏幕能立即给出反馈信息或结果，因而称为所见即所得(what you see is what you get)。用视、点(鼠标)代替了记、击(键盘)，给用户带来了方便。

1.5.3 图形用户界面系统的结构模型

一个图形用户界面系统通常由三个基本层次组成。它们是显示模型、窗口模型和用户模型。用户模型包含了显示和交互的主要特征，因此图形用户界面这一术语有时也仅指用户模型。图1-2给出了图形用户界面系统的层次结构。

图1-2中的最底层是计算机硬件平台。硬件平台的上面是计算机的操作系统。大多数图形用户界面系统都只能在一两种操作系统上运行，只有少数的产品例外。

操作系统之上是图形用户界面的显示模型。它决定了图形在屏幕上的基本显示方式。不同的图形用户界面系统所采用的显示模型各不相同。例如大多数在Unix之上运行的图形用户界面系统都采用X窗口做显示模型；MS Windows则采用Microsoft公司自己设计的图形设备接口(GDI)做显示模型。

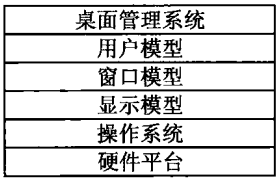

图1-2 图形用户界面系统的层次结构

显示模型之上是图形用户界面系统的窗口模型。窗口模型确定窗口如何在屏幕上显示、如何改变大小、如何移动及窗口的层次关系等。它通常包括两个部分：一是编程工具；二是对如何移动、输出和读取屏幕显示信息的说明。因为X窗口不但规定了如何显示基本图形对象，也规定了如何显示窗口，所以它不但可以充当图形用户界面的显示模型，也可以充当它的窗口模型。

窗口模型之上是用户模型。图形用户界面的用户模型又称为图形用户界面的视感。它也包括两个部分：一是构建用户界面的工具；二是对于如何在屏幕上组织各种图形对象，以及这些对象之间如何交互的说明。比如，每个图形用户界面模型都会说明它支持什么样的菜单和什么样的显示方式。

图形用户界面系统的应用程序接口由其显示模型、窗口模型和用户模型的应用程序接口共同组成。例如OSF/Motif的应用程序接口就是由它的显示模型和窗口模型的应用程序接

口 Xlib，以及用户模型的应用程序接口 Xt Intrinsics 及 Motif Toolkit 共同组成的。

1.5.4 用户界面的发展：GUI＋新人机交互技术

人机交互是研究人、计算机以及它们相互影响的技术。随着计算机处理、存储能力的飞速提高，以及体积、成本的降低，人们已将注意力逐渐转移到改善人机交互的手段和界面方面，越来越期望计算机来适应人的习惯和要求。人们对于无所不在的计算要求，使手持移动计算逐渐成为当今主流的计算模式之一，人机交互的效率和自然性已经成为了移动计算普及和应用中最为核心的问题之一。能够将听、说、写等日常生活中的基本技能用于计算机操作，是提高计算机的可用性、友好性和自然性的重要方面。

随着虚拟现实、科学计算可视化及多媒体技术的飞速发展，新的人机交互技术不断出现，更加自然的交互方式将逐渐为人们所重视，这尤其体现在手持设备上。手持设备自身的物理特征决定了纯粹基于 WIMP 界面的交互方式很难在移动计算环境中取得良好的效果。交互设备和交互手段的限制降低了信息输入效率，而过小的屏幕又给输出信息的呈现造成了困难。与交互效率降低并存的还有交互自然性的下降，用户使用指点工具在过小的屏幕上进行点击、选择等精确操作，增加了交互过程中的认知负担，从而容易导致用户的厌烦心理。因此，提高移动系统人机交互的效率和自然性，是一个值得关注和研究的问题。

多通道人机交互(multi－modal human－computer interaction)是一种使用多种通道与计算机通信的人机交互方式。多通道人机交互是提高交互效率和自然性的有效途径。通道(modality，也有译为模态、模式)涵盖了用户表达意图、执行动作或感知反馈信息的各种通信方法，如言语、眼神、脸部表情、唇动、手动、手势、头动、肢体姿势、触觉、嗅觉或味觉等。多通道人机交互是提高交互效率和自然性的有效途径。单一的交互方式是导致交互效率低下的一个重要原因，多通道用户界面允许用户通过各种不同的交互通道以及它们之间的相互组合、协作来完成交互任务，这正好弥补了单一通道交互给用户带来的限制和负担。

多通道人机交互比传统 WIMP 界面适用于更多领域的应用程序以及更广泛的用户群。笔和按键是目前手持设备上使用最广泛的输入手段，它们都是单一通道的输入模式，而单纯的图形文本显示也是一种单通道的输出方式。单一的交互方式是导致交互效率低下的一个重要原因。多通道用户界面允许用户通过各种不同的交互通道以及它们之间的相互组合、协作来完成交互任务，这正好弥补了单一通道交互给用户带来的限制和负担。同时，移动设备所访问和处理的信息由各种不同的媒体承载，在这样一个混合媒体的环境中，系统需要多个通道来对应各种不同的信息类型。

1.6 Linux 图形环境及桌面平台简介

在介绍嵌入式 Linux 上的 GUI 系统之前，作为背景知识，先介绍一下 Unix/Linux 上的图

形环境与桌面平台。

虽然 Linux 桌面应用近年来取得了长足进展，但就所占市场份额而言，桌面平台依然是 Windows 的天下。对大多数习惯于使用 Windows 的用户来说，Unix/Linux 的图形环境可能与他们对于图形界面系统的认识有比较大的出入。Linux 实际上是以 Unix 为模板的，无论系统结构还是操作方式也都与 Unix 无异。可以说：Linux 就是 Unix 类系统中的一个特殊版本。众所周知，微软 Windows 在早期只是一个基于 DOS 的应用程序，用户必须首先进入 DOS，然后再启动 Windows 进程；而从 Windows 95 开始，微软把图形界面作为默认，命令行界面只有在需要的情况下才开启，后来的 Windows 98/Me 实际上也都隶属于该体系；到 Windows 2000 之后，DOS 被彻底清除，Windows 成为一个完全图形化的操作系统。但 Unix/Linux 与 Windows 完全不同，强大的命令行界面始终是 Unix/Linux 的基础。

在 20 世纪 80 年代中期，图形界面风潮席卷操作系统业界，麻省理工学院（MIT）也在 1984 年与当时的 DEC 公司合作，致力于在 Unix 系统上开发一个分散式的视窗环境，这便是大名鼎鼎的“X Window System”项目。不过，X Window 并不是一个直接的图形操作环境，而是作为图形环境与 Unix 系统内核沟通的中间桥梁，任何厂商都可以在 X Window 基础上开发出不同的 GUI 图形环境。MIT 和 DEC 的目的只在于为 Unix 系统设计一套简单的图形框架，以使 Unix 工作站的屏幕上可显示更多的命令，对于 GUI 的精美程度和易用程度则并不讲究。1986 年，MIT 正式发行 X Window，此后它便成为 Unix 的标准视窗环境。紧接着，全力负责发展该项目的 X 协会成立，X Window 进入了新阶段。与此同步，许多 Unix 厂商也在 X Window 原型上开发适合自己的 Unix GUI 视窗环境，其中比较著名的有 SUN 与 AT&T 联手开发的 Open Look、IBM 主导下的 OSF（Open Software Foundation，开放软件基金会）开发出的 Motif。而一些爱好者则成立了非营利的 XFree86 组织，致力于在 X86 系统上开发 X Window，这套免费且功能完整的 X Window 很快就进入了商用 Unix 系统中，且被移植到多种硬件平台上，后来的 Linux 也直接从该项目中获益。

基于 X 的应用软件是通过调用 X 的一系列 C 语言函数实现其各种功能的。这些函数称为 Xlib（X 库），它提供了建立窗口、画图、处理用户操作事件等基本功能。Xlib 是一种底层库，用它来编写图形和交互界面程序虽然非常灵活，但却比较复杂甚至繁琐。为此又发展出了一些比 Xlib 更高层的库函数，称做工具包（Toolkits），它们将一些常用的界面图形（如窗口、菜单、按键等，通常称做工具包中的组件 Widgets）按面向对象编程的方式组织到一起供应用软件使用，而工具包的 Intrinsics（内在、本质）允许在它们之上建立新的 Widget。

Xlib、X Intrinsic 以及 Toolkits 之间的关系是：Xlib 控制 X 协议以及网络问题，对开发者而言只是一些非常原始的接口函数，只提供基本的 C 语言 API。在 Xlib 上是 X Intrinsic，X Intrinsic 为高层的 Toolkits 提供面向对象的框架，如果需要自己开发一套 Toolkits，可以从这里开始。

再向上就是 Toolkits 库。Toolkits 则提供完整的用户界面开发包，里面有了“菜单”、“按