第六篇　X Window系统的内部结构和使用

X Window 系统是在UNIX类的操作系统中应用最为广泛的基于窗口的用户图形界面。它使用方便，界面直观，并且和具体的计算机的硬件无关。同时它支持分布式的网络操作。所以，基于X Window 的应用程序一直在UNIX类的操作系统中占有主导的地位。 Linux出现以后， X Window 系统也有了在Linux系统上的实现。这就是XFree86系统。XFree86 比标准的MIT版的X Window 支持更多的硬件，这样，它的应用就更加的广泛。关于 XFree 86的安装和启动，请参考附录B。

麻省理工学院发布的 X Window系统以及其源代码叫做 X11系统。 Linux系统上使用的XFree86.3.1.2就是基于X11R6版本的。其他UNIX系统中的X Window应用程序基本上可以在Linux 的XFree 86系统上直接使用。而在Linux系统中编写基于XFree 86 的应用程序的需求也越来越多。所以在本篇，我们讨论一下Linux系统中X Window窗口系统的内部结构和具体的设置和使用。有了这些知识，你就可以尝试着在 XFree 86系统上编写一些使用的程序。等到积累了一定的经验以后，就可以编写一些大型的基于 X Window系统的程序了。下面，我们先从 X Window系统的最基本的概念讲起。

第27章　X Window系统的基本知识

27.1　X Window系统介绍

X Window 系统是一套在各种图形显示器上均可使用的窗口系统。它是由麻省理工学院(MIT)开发出来的。 X Windows系统(以下简称为X) 可在许多系统上执行。由于它和生产厂商无关、具可移植性、对彩色掌握的多样性及对网络之间的操作透明性，使得 X成为一个工业的标准。由于原始程序代码可自由使用，所以它也是一个优秀的研究媒介。

程序员可以利用X开发可移植性图形用户界面。 X最重要的特征之一是它独特的与设备无关结构。任何硬件只要提供X协议，便可以执行应用程序显示一系列包含图文的窗口，而不需要重新编译和链接。这种与设备无关的特性，使得只要是根据 X标准所开发的应用程序，均可在不同的环境下(如大型电脑、工作站、个人电脑上)执行，因而奠定了X成为工业标准的地位。

X可以在一些 UNIX 系统的电脑上执行，如 Alliant、Apollo、DEC、IBM、Hewlett-Packard、Sun等，也可在 DEC的 VAX/VMS、MS-DOS及一些其他的系统上执行。其他的一些厂商如 AT&T、Adobe、Control Data、Data General、Fujitsu、Prime、Siemens、Silicon Graphics、 Sony、Texas Instruments、Wang、Xerox 均曾表示支持X。

27.1.1　X的特点

X的特点及其受大众欢迎的原因如下：

• X具有网络透明性：通过网络，应用程序在其他计算机上输出显示，就和在自己计算机

上输出一样容易。此种通信结构和网络上另一端的计算机使用的语言完全无关，也和计算机硬件无关，甚至不需使用相同的操作系统。总而言之，程序可以在另一种不同类型显示器下执行，而不需要重新编译和重新链接。

- 可支持许多不同样式的用户界面。管理窗口的功能 (例如窗口的摆放、大小及显示顺序等等)并不包含于系统中，而是由应用程序来控制，因此可轻易地更换。不同样式的界面和不同的应用程序有关，例如滚动窗口中的文字和选择窗口中的一个物体，彼此间不会互相限制。
- X不是电脑操作系统的一部分：对操作系统而言， X只是一个应用程序而已，因此， X很容易在不同的系统上安装。
- 窗口是层次式的： 应用程序可以直接利用窗口系统已有的工具便可满足大部分的需求，而不需要借助于其他的输入或控制结构。 (例如：可利用一个分支的子窗口来产生菜单。)

27.1.2 什么是窗口系统

本节讨论一般窗口系统的一些基本概念， X可以当作其中一个特例，如果你已熟悉其他的窗口系统，本节内容仅需快速浏览即可。

X是一个在图形显示屏幕上建立和管理窗口的系统，它可以在拥有图形显示器和键盘的工作站或其他型号拥有图形显示器的终端机上执行。 X把指示位置的设备叫做指针，这个设备通常为鼠标。X支持现今电脑上常见的窗口用户界面。

使用窗口系统的情形与在普通办公桌上的工作相似。你的办公桌上通常放了一些纸、邮件和手边相关的工作、一些有用的工具 (如时钟，日历，计算器等)。当进行到工作的另一个部分时，你会重新安排桌面上的纸，你可能把工具集中放在一起，也可能不时参考桌上仍然看得到的纸的内容，过了一阵子，你可能把其中的一些资料暂时摆到一边，或者通通从桌面上移走。

上述是一个人的工作模式，如果电脑能提供这样的功能是很理想的。不幸的是，老式的终端机或显示器使得你一次只能在屏幕上做一件工作，而且只能看见一小部分的文字资料 (通常为24行)，图形的工作就更别提了。现在窗口系统正克服这点，通常它提供一个较大的屏幕，允许你同时看到几件工作项目，可以显示图形，甚至有彩色。

X便是依照上述窗口的工作模式开发出来的。 在X的环境下，一个窗口是屏幕上的一块长方形区域，且平行于屏幕的边，通常，每一个窗口被一个独立的应用程序所专用，数个应用程序可以“同时”在它自己所拥有的窗口上显示输出结果， X允许窗口重叠。

但即使窗口的一部分或全部被其他窗口遮盖，应用程序仍然可以对它自己所拥有的窗口输出信息。设备程序提供在屏幕上移动窗口、改变窗口大小、把窗口放在最上一层或最下一层等功能。即便是窗口可以重叠，但在同一屏幕开了许多窗口仍然非常费时。 因此和其他的窗口系统一样， X提供图标功能。我们在屏幕上用一个图标代表一个应用窗口，当我们对应用窗口图标化后，窗口以图标代之，从而空出了较多的屏幕空间；相反的动作为解除图标化，也就是以原先的窗口替换图标。

一些实用的功能，例如时钟或日历，并非系统内部提供的，而是由许多小的应用程序所提供的。

对于输出， X提供了许多在窗口写文字和画图形到的功能选择。许多种字体可供选择，并且提供许多图形的结构和绘图的基本方法，例如提供点、线、弧线、区域的画法。颜色的选择更是丰富。这些复杂的部分对用户而言是隐藏起来的，用户可以简单地使用它们，例如，在使

用时，你可以用“ *times-bold-i* ”表示要使用加倍粗的斜体字体；当你需要使用彩色时，你只要用日常的名称，例如“ yellow(黄色) ”或“ navy blue (天蓝色) ”。

X也提供多样化的输入功能。 X可以使用不同形式的键盘，如传统的 QWERTY键盘或Dvorak Style键盘，或者是不同国家的有特殊规定的键盘。处理用户界面是输入功能很重要的一个部分，键盘和鼠标发出的指令告诉系统如何构造一个窗口和处理窗口中的内容。

由于X的窗口处理功能并非是系统内部提供的，而是建立在用户层次上的，因此容易修改或更换。所以X能提供不同形态的用户界面。换个角度来说，用户界面所必需具有的灵活性几乎完全可由X独立提供。

27.1.3 X发展的历史

X于1984年在美国麻省理工学院 (MIT)电脑科学研究室开始开发。当时 Bob Scheifler 正在开发分布式系统，同一时间 DEC公司的 Jim Gettys 正在麻省理工学院做 Athena 计划的一部分，两个计划都需要一个相同的东西 —— 一套可以在 UNIX 机器上运行的窗口系统。因此他们从斯坦福大学得到了一套叫做W的实验性窗口系统。 因为是根据W窗口系统开始开发的，所以当开发到了足以和原先系统有明显区别时，他们把这个新系统叫做 X。

工作持续地进行，新的版本不断地产生 (当软件和前一版不相容时，新的版本便产生了)。在1985年中期决定了任何人只要付版权费便可使用 X。以下为X的大事记:

- 第10版: 1985年底。从此，在 MIT以外的人和组织，才开始对X有实质的贡献。
- 第一套商业化的X产品: DEC 于 1986 年 1月推出 VAXstation-II/GPX。
- 第10版第 3次发行: 1986年 2月。从此时起， X开始流传于世，人们把它移植到许多新的系统上。
- 第10版第 4次发行: 1986年11月。
- 第一次X技术会议: 1987年 1月于 MIT。
- 在1986年间，第10版X无法满足所有的需求已非常明显， MIT 和 DEC便从事于重新设计整个协议，这就是X第11版。
- 第11版第 1次发行: 1987年 9月。
- X协会成立: MIT X协会成立，目的是为了研究开发及控制标准，目前有 30个以上的机构加入。
- 第二次X技术会议: 1988年 1月。
- 第11版第 2次发行: 1988年 3月。
- 第11版第 3次发行: 1988年10月。

27.1.4 X的产品

严格地说， X窗口系统并不是一个软件，而是一个协议。这个协议定义一个系统产品所必需具备的功能 (就如同 TCP/IP、DECnet 或IBM的 SNA，这些也都是协议，定义软件所应具备的功能)。任何系统能满足此协议及符合X协会其他的规定，便可称为X的产品。

简单地说，本附录从现在起不再区分协议和软件。当我们提到 X，意指一个完整且适当的系统产品。

27.1.5 MIT 发行的X

MIT 所发行的X可以支持许多厂家的电脑，目前的版本 (第11版第 3次发行) 支持以下系统:

• Apple A/UX
• Apollo Domain/IX
• 4.3 + tahoe
• Digital Equipment Coporation Ultrix
• Hewlett Packard HP-UX
• IBM AOS
• Sun Mircosystems SunOS

此外还有更多的商业化产品。

此系统一直处于开发中，而且越来越多的人开始使用它，由第三方厂家开发的软件也逐渐增加，从而使系统版本一分为二：

• core版 —— 软件由MIT X协会提供。
• corelib 版 —— 软件由用户或第三方厂家提供。

为了实用，core和corelib 软件保存在不同的磁带上发行。

1. MIT 版包含了什么

这个版本包含了文件说明、源代码、配置文件、实用程序和其他一些建立完整操作系统所必需的东面，(没有提供任何目标文件或二进制文件，系统必需由源代码编译建立)，在此我们只从用户观点看这个系统，所以只描述那些窗口系统程序本身或一些用户所需的工具程序，省略设置实用程序、配置工具程序、本版需知等。

core版的程序可分为以下几类：

1) X窗口系统本身的程序。
2) 使用窗口系统必备的工具和设备程序：
• 日常的窗口相关功能的工具程序 (例如将窗口内容打印至打印机)。
• 一些你常常保持在桌面的实用程序 (例如时钟，日历)。
3) 可以利用窗口环境的一般应用程序。
4) 演示程序和游戏程序。
5) 信息和状态报告程序。
6) 定制你自己的环境的工具程序。

2. 系统程序

以下程序包含了所有和基本系统相关的程序。

1) X显示服务器 —— 这个软件控制了你的工作站的键盘、鼠标和屏幕。这是 X的心脏。此程序可以建立、删除窗口，同时可以应其他客户机程序的需求做写和画的动作。

这个服务器程序在各种硬件上有不同的实现，例如：

Xapollo —— 针对Apollo显示器。
Xhp —— 针对Hp 9000/300 的Topcat显示器。
Xibm —— 针对IBM 的APA16 和Megapel 显示器。
Xmacll —— 针对Apple 的Macintosh II。
Xplx —— 针对Parallax图形控制器。
Xqdss —— 针对DEC 的GPX 显示器 (VAXstation II/GPX)。
Xqvss —— 针对DEC 的QVSS显示器。
Xsun —— 针对Sun/2，Sun/3，Sun/4 和Sun/386i工作站。

2) Xinit —— 初始化程序，启动系统和设定服务器执行方式。

3) Xdm —— X显示管理器，它可以灵活地启动系统，使系统启动成符合个别需求的环境。它可以和 Xinit两者之间选择一个使用。

4) Uwm —— X窗口管理器。此程序决定如何管理你的桌面、移动窗口、重定窗口大小等等，你可以利用菜单，结合鼠标的按钮或键盘完成窗口操作。

这中间只有服务器程序是绝对必需的。不需其他的程序，你就可以在 X系统上运行其他的应用程序 。(Xinit 等程序可由其他相同功能程序替代。)

以上程序包含了窗口系统的基本元素。但除了在窗口上移动光标外，什么事也不能做。因此实际上，你需要更多的实用程序和应用程序。

3. 窗口系统中的实用程序

以下的工具程序并不是窗口系统的一部分，但它们是你使用窗口系统或利用窗口系统做更多的事所不可缺少的，它们分为以下两个部分：

(1) 窗口系统操作时常用的工具程序

只要你用窗口系统代替一般的电脑终端机，这些程序几乎是天天需要的：

xterm - X终端机模拟器。你的系统内大多数的程序并非特别为使用窗口系统所设计的。举例来说，一些最普通的系统程序 —— 列出文件目录，编辑器，编译器等，它们在普通的终端机可以正常的执行，可是它们不知道如何在X下操作运行。Xterm的作用就是建立一个X的窗口，且允许这些普通的 “ 哑终端机 ” 程序能够在这个窗口中执行。这些普通程序会认为它们是在 “ 真的 ” 终端机上执行。当然，你也可以用xterm 去启动其他的X程序而并非一定是那些普通程序。

xhost —— 让你控制网络上那些可以存取你的显示屏幕的其他主机。

xkill —— 一个可终止不希望运行的应用程序的工具程序。

xwd —— 将你窗口内目前的图像存储到一个文件中，使得你可以在稍后重建这个窗口、打印它或做一些其他你想做的事。

xpr —— 将先前 xwd所抓取的窗口图像转换成适合硬拷贝输出的格式。

xdpr —— 结合了 xwd和 xpr，允许你在一个步骤中就可以输出窗口的内容。

xmag —— 将屏幕上选中的一部分图像加以放大。

xwud —— 将先前 xwd所抓取的窗口图像重新显示于屏幕上。

x10tox11 —— 将能在第10版X执行的程序转换成可在第11版执行。

xrefresh —— 更新显示，将某些或全部的窗口重画一遍。

(2) 实用的程序

xclock —— 一个指针式或数字式的时钟。

xclac —— 一个计算器，可模拟科学工程型的计算器。

xload —— 用累计图来显示目前机器的负载分布。

xbiff —— X版的 biff。邮件到达告知程序， xbiff 会显示一个信箱的图标，当信箱上的旗子升起时，表示有你的信到达。

4. 一般的应用程序和工具程序

这些程序不直接和窗口系统相关，但他们可以利用窗口系统环境更加有效地运行。

xedit —— 一个文字编辑器，你可以用菜单或键盘发送命令，也可以用鼠标指针指定位置或一段文字。

xman —— 一个说明书或系统文件的浏览器。

xmh —— 一个邮件管理程序。

5. 示例和游戏程序

这些程序演示了X图形和彩色的能力。在你开始熟悉使用系统时，它们是一个良好的起点。

ico —— 显示一个20面体 (或其他多面体) 在窗口内进行碰撞运动的情形。

maze —— 以随机数建立一个迷宫并找出它的解法。

muncher —— 在窗口上描绘大量动态的图案。

plaid —— 在窗口上画一些持续变化的花格子图形。

xlogo —— 在窗口上印一个X的字形。

puzzle —— 智慧盘。在一个4×4方块盘上，移动编号1～15 的小方块，以排成特定型状的游戏。

6. 显示信息和状态的程序

以下的程序为你提供有关于窗口系统的信息和状态，你可以和自己的工具程序一起来使用他们。

xfd —— 在窗口内显示一个X中的字体，且可以选择提供更多有关此字体的信息。

xlsfonts —— X字体的目录程序，告诉你一个显示器上有哪些字体可供使用。

showsnf —— 显示服务器上原有的格式有关某一种字体的细节。

xwininf —— 显示某个特定窗口的信息，如大小、位置及其他特征。

xlswins —— 列出系统内所有的窗口，并可以选择地列出每个窗口的一些细节。

xprop —— 显示窗口的属性和字体。

xdpyinfo —— 提供你的显示器及控制它的服务器的细节。

xev —— 输出和窗口相关所有X“ 事件 ”的细节，用来侦错或给有经验的人使用的工具程序。

7. 可以用来定制适合你的系统的工具程序

一开始你可能不会使用这些程序，但过了一段时间，你可能发现你必需修改一下系统，例如想使用较大的缺省字体，窗口边框换成自己喜欢的颜色等。用以下的程序，可以使你的工作环境更加适合你。

xset —— 允许依照你的喜好设定显示特性。你可以设定一个键使它有效或无效，调整警告铃的音量，指定字体从何处获取等。

xsetroot —— 你可以选择显示屏幕背景的外观，当你的鼠标指针不在任何应用窗口内时，你可以改变屏幕背景的颜色或图案以及光标形状和颜色。

xmodmap —— 显示键盘的对应关系，也就是按什么键对应什么字符。利用此程序可以修改成适合你的对应关系，通常用来设定一些特殊键 (如 META，Shift-Lock等) 和功能键，但你可以视需要设定。

bitmap —— 让你建立和编辑图形的程序。例如用来改变光标的式样、编辑图标、改变窗口的背景图案等等。

xrdb —— 让你在数据库中显示或改变你喜爱的颜色或字体等等。以后应用程序可以使用的字体和颜色。也就是说，你可以设定一些缺省的特性，所有的应用程序或只有一些特定的应用程序使用这些特性作为缺省特性。

bdftosnf —— 将一种字体从BDF格式转成你的服务器原有格式。

27.2 X的基本结构

这里描述X的基本结构，并介绍许多基础的概念，其目的在于使你在稍后使用 X系统时能有一个了解。你将会知道系统程序做些什么和如何做，这样你将更快和更有效率地使用系统。

我们也会指出系统额外的作用，以及使用系统对你的影响。

27.2.1 X的基本元素

X不像早期的窗口系统，把一堆同类软件集中在一起，而是由三个相关的部分组合起来的：

1)“服务”程序：是控制实际显示设备和输入设备的程序。

2)“客户”程序：需凭借服务程序在指定的窗口中完成特定的操作。

3)“通信通道”：客户程序和服务程序用此来彼此交互信息。

1. 服务程序

服务程序用来控制实际的显示设备和输入设备 (键盘和鼠标或其他输入设备)的软件。服务程序可以建立窗口、在窗口中画图形、图像和文字；回应客户程序的需求。它不会自己执行动作，只有在客户程序提出请求后才完成动作。

每一个显示设备只有一个唯一的服务程序。服务程序一般由系统的供应厂商提供，用户通常无法修改。对操作系统而言，服务程序只是一个普通的用户程序而已，因此很容易更换一个新的版本，甚至可编译运行由第三方厂商提供的原始程序。

2. 客户程序

客户程序是指使用系统窗口功能的一些应用程序。把 X下的应用程序称作“客户”程序，原因是它们是服务程序的“顾客”：客户程序要求服务器应它的请求完成特定的动作。

客户程序无法直接影响窗口或显示，它们只能向服务程序发送请求，让服务程序来完成它们的需求。典型的请求通常是：“在 XYZ窗口中写一行‘ Hello，world ’的字符串”，或“在CDE窗口中用这种颜色从 A点到B点画一条直线”。

当然，针对窗口操作提出需求只是客户机程序的一部分，其他的部分是那些让用户执行的程序部分。例如：编辑文字、画一个系统的工程图、执行一个计算表格的计算等等。一般来说，客户程序和窗口系统是独立的，它们对于窗口几乎不需要知道什么。通常 (特别是指大型的标准绘图套装软件，统计套装软件等) 应用程序对许多的输出设备具有输出的能力。在 X窗口上的显示只是客户程序许多输出格式中的一种。所以，客户程序中和 X相关的部分在整个程序中只占了非常小的一部分。

用户可以使用不同来源的客户程序：一些是由系统提供的 (例如时钟) ，一些来自于第三方厂商，一些是用户为了特殊应用而编写的自己的客户程序。

3. 通信通道

X的第三个元素为通信通道，客户程序借助于它给服务程序发送请求。而服务程序借助于它向客户程序回送状态及一些其他的信息。

只要客户程序和服务程序都知道如何使用通道，通道的本身的结构并不是很重要。在系统或网络上支持通信类型的请求内建于系统基本的 X窗口函数库中，所有和通信类型有关的事件都从函数库独立出来，客户程序和服务程序之间的通信只要借助于使用这函数库（在标准 X版为xlib）即可。

总之，只要客户程序利用函数库，自然可以用到所有可用的通信方法。

客户程序和服务程序之间的通信大约可以分为两类，分别对应于两种 X系统的基本操作模式：

- 服务程序和客户程序在同一部电脑执行，则它们彼此均可使用机器上任何可用的方法做进程内通信(简称IPC)。在这种模式下， X可以像许多传统的窗口系统一样有效率地操作。

- 客户程序在一部机器上执行，而显示和服务程序在另一部机器上。则客户程序和服务程序的通信必需通过网络利用彼此同意的协议方可。目前，最常见的协议为 TCP/IP和DECnet，但其他任何的可信赖的协议亦可使用。

这种通过网络使得应用程序的操作如同在本地机器一样的能力称为网络透明性，这几乎是X独一无二的特性。这种特性使得它非常适合建立在灵活的多目标混合的计算机网络上。

因为客户程序和服务程序完全独立，这样开发了一种称为 X-terminal 的新型显示器。简单的说，X-terminal 是一种除了能直接在上面执行X服务程序外，什么也没有的工作站。它有键盘、鼠标和屏幕，以及一些和网络互相通信的方法（所以在其他主机上的客户机可在它上面显示），但并没有文件系统，也不提供一般目的的程序，一般目的的程序需要在网络上执行。

27.2.2 服务程序和客户程序如何交互通信

本节描述客户程序和服务程序互相通信时，双方各传输些什么信息。简单说，一个客户程序要求服务程序执行输出，输入则借助于“事件”来通知服务程序来处理（“事件”的含义：如按下键盘的键或鼠标的按钮等等）。

1. 客户程序送达服务程序的信息

当一个客户程序要求服务程序做一个动作，例如在一个指定的屏幕上建立一个有特殊特征的窗口，或者在一个窗口中写一行字符串，这时客户程序是借助于送请求到服务程序上来完成的。一个请求是一个被封装的简单区块，区块包含一个“操作码”来指示要执行何种操作，同时包括一些参数提供更多的请求细节。例如：清除一个窗口内的一个长方形区域，客户程序会送一个16位字节的请求区块，来指定是哪一个窗口，该长方形区域的左上角坐标及区域的高和宽。

这个格式有几个重要的特征：

- 请求区块的内容和客户程序与服务程序在何种类型上的机器上执行完全无关。一个客户程序可以输出请求给任何型号显示器上的任何 X窗口服务程序。请求和语言、机器及操作系统均无关。
- 每一个请求包含了窗口的细节和其他被使用的资源。对一个客户程序送至特定服务程序的请求可以提供超过一种以上的连接方法，所以在网络结构上提供的窗口数目没有限制。
- 请求区块通常大小为20个字节左右，算是相当小的，因为请求是设定为相当高级的，例如画一条线是指定两个端点而非记录一串屏幕上的点。通常屏幕上被影响到的像素的数目往往是区块本身大小的十到一百倍，这样不会使网络的负荷太重，网络的使用效率会非常高（一般认为X的服务程序和客户程序之间的传输是位图的观念是错误的）。

2. 服务程序送达客户程序的信息

服务程序也会利用通信通道送信息回到客户程序，这些信息包括回应客户程序请求是否成功和告诉客户程序有兴趣的特殊事件，这些事件包含的信息类似“窗口 XYZ 的鼠标左按钮被按”或“窗口ABC 已被重定大小等”。

就像从客户程序来的请求一样，服务程序的回应也是一些和语言、机器、操作系统无关的简单区块。

事件是X的基本功能。所有的键盘输入，鼠标按钮输入和鼠标移动都是由事件来控制。更进一步说，客户程序完全依赖事件才能获得那些在系统中发生的，而它必需知道的信息。下面我们将从一些普通的输入和移动功能着手，实际了解 事件是如何工作的。

3. 键盘输入

当你从键盘按下一个键，服务程序将会查觉到这个动作。服务程序便送出一个 <Key Press> 的事件通知那些已经登记的对这种情况有兴趣的应用程序。这种通知有一些限制：不是通知目前被鼠标指针指到的窗口，便是通知目前被指定接受所有键盘输入的窗口。这种限制称为设定键盘焦点。

当键被释放时，另外一个<Key Release> 的事件产生了（通常几乎是立刻），一般除了那些修饰键（例如Shift或 Control），应用程序很少会对释放键这个 事件有兴趣。

送到客户程序的信息区块告诉客户程序它们是键盘 事件，只是“ 编号第几的键已被按下（或释放）”，不包含是不是ASCII 或EBCDIC字符及如何解释等内容，而把这些留给客户程序去处理，这种做法使得客户程序看起来似乎复杂，但是标准的 xlib函数库中有非常简单的子程序可供控制解释键盘事件，而且通常缺省成你所希望的键盘型号。换个角度来看，这种“ 软件 ”的键盘字符相关方式提供了很大的灵活性。在服务程序这方面，对不同型号的键盘均可以完全重新对应；在客户程序这方面，每一个单独的键都可以“ 程序化 ”，例如按一个键即可以输入一串用户特定的字符串，或者完成一个特殊的功能等。

稍后我们会再详细讨论，到目前为止，这些将不会影响你使用系统。事实上，对于 X系统如何处理你按下一个“ A ”键，并将它转换成一个ASCII 的“ A ”字符送到你的应用程序的这类事情，你不需要太关心。

4. 关于鼠标指针位置的 事件

当屏幕上的鼠标指针进入或离开客户程序所控制的窗口时，客户程序可以要求了解这些事件。这些事件(<EnterWindow>和<LeaveWindow>) 告诉客户程序是进入或离开窗口以及是哪一个窗口。

当鼠标指针进入窗口时通常用类似“ 高亮度显示 ”窗口这一类的方式表达，有些应用程序可改变窗口的边框（例如从灰到黑），有些则会改变颜色，用以强调你目前正在处理这个应用程序（窗口）。

5. 当一个窗口未被覆盖时

X和大多数其他的窗口系统有一个很大的不同点，那就是客户程序必需负责保持它的窗口最近的内容，服务程序只是维持窗口在任何时刻均在屏幕上显示，但它不负责保持窗口的内容。

当原先被其他的窗口遮住的窗口（或窗口的一部分）变成可见时，服务程序并不知道应该显示这个窗口的哪个部分。服务程序送一个 exposure事件给拥有这个窗口的客户程序，告诉它窗口的哪一个部分刚刚已变成可见，客户程序便会决定该怎么作，在大多数的情况（一般为简单的应用程序或小窗口），客户程序只是重画整个窗口，因为只画窗口未被遮盖的部分往往要多花额外计算，并不值得。在更复杂的应用程序，客户程序才会只重画窗口必需要出现的部分，这是由应用程序的编写者决定，他必需在效率（窗口更新的速度）和只重画部分窗口程序码的复杂程度之间作取舍。

依赖客户程序来重画窗口内容的方式对效率特别重视，尤其是下拉式菜单，你总不希望选下菜单之后，菜单消失一段事件之后才让下面的窗口显示出来吧，为了克服这点，有些 X的产品包含了被称为save-under（存下层）的实用程序。

你可以告诉服务程序，如果可能的话，尽量在一个窗口被遮盖前将其被遮盖的内容存下，当遮盖的窗口被移走时便可立即重现而不需要送重现 事件给客户程序。

一个类似而更常用的，被称为backing store 的方式也被开发出来，你可以告诉服务程序尽可能在一个窗口被遮盖前将其全部内容存下，同样的，这种方式可以改进客户程序重画窗口的

效率，backing store 和save-under两者的不同处是前者保存整个窗口的内容，而后者只存被遮盖的部分。

虽然有了save-under和back store这两种产品，但你不能指望此种结构，客户机程序仍然随时保持准备接受重现事件。即使服务程序真的维护了一段时间的窗口内容，也可能因为内存不足而被迫停止，转而开始重新送出重现事件。

27.2.3 X的网络概况

我们曾经提过，客户程序和服务程序只需通过网络便可在不同的机器上执行，下面几节我们将看看如何利用这种实用的方法节省了计算的资源，从而增进了网络的成长。

1. 如何实际使用X网络

当服务程序在一个连接了显示器的机器上执行，而客户程序在另一部机器上执行时，鼠标和键盘的输入由服务程序所在的机器搜集，可是客户程序却可以在别的地方使用到这些输入，这是如何办到的？我们以下面的例子解释。

假如你在使用一个由X服务程序控制显示器的工作站，如果它是独立的，很明显，客户程序也在此工作站上执行。即使连接了网络，大部分的时候你还是在你自己的工作站执行客户程序。可是因为有一些特殊的实用程序，你的机器上并没有，而你却希望在你的机器上显示程序的输出，这时你便需要网络上的机器了。利用你的操作系统提供的一些普通的网络设备程序，你便可以让客户程序在远程的机器上执行，而指定输出显示在你自己工作站的显示器上。

假设客户程序的名称为xgraph，在UNIX系统上，你所发出的命令类似下面：

```
rsh neptune xgraph -display venus:0
```

则xgraph程序在远程名为neptune 的机器上执行，且xgraph的输出会送到你自己名为venus的机器上的 0号显示器上。从现在起，我们将参照这种远程显示的模式操作，当客户程序在一部机器上执行时，服务程序在另一部机器上执行。

现在总结一下：你使用远程显示的设备程序使得客户程序在远程的机器上执行，而且告诉它将输出显示在执行X服务器程序的本地机器上。

2. X的网络设备有何用途

在一部机器上执行客户程序而把输出显示到另一部机器有何用途？这些用途和实用是极常见的，以下是一小部分的用途：

- 远程的机器速度比你的计算机快很多（可能是因为加了浮点运算器或它根本就是一部超级电脑）。
- 在你的局域网上，远程的机器是一部文件服务器，它提供了大量磁盘资源，为了降低网络的负担，你可以把一些类似搜索的大量操作，需要用到的大量磁盘动作的程序放在远程机器上执行，这样一来，只有执行结果而不是大量操作磁盘的动作会通过网络传送。
- 远程机器有特殊的结构适合特别的工作，可能是专门的数据库机器，或者是为一个单独的应用特别设计的特殊目的机器。
- 远程的机器有只能在其上执行的特殊软件。在现代的工作站，在网络上有些软件许可只有少数的机器拥有已是愈来愈多的趋势，因为软件许可只发给那些付费的工作站。在这种情况下，可以实际地在远程的机器上执行这些有许可的软件，而将显示传回你自己机器上，这相当地实用。
- 你需要同时存取好几部机器，通常系统的管理员有此需求。
- 你需要同时输出到数部显示器。

3. X网络结构产生的简易性

就像前面所提过的，所有从客户程序向服务程序发出的请求，由于它们的格式和内容是和设备无关的。而所有和设备相关的事完全集中在服务程序，对于任何显示器的硬件，只有对应于此种显示器的服务程序才需要去关心。只要提供针对一个显示器的服务程序，所有可执行 X 客户程序的其他机器立即可使用这个显示器，不需要重新编译或重新链接，甚至连显示器是什么型号都不需知道。

这种把设备的相关性独立出来给服务程序的方式，对许多工作站网络的供应商变得可行且轻松，这种灵活性在两方面特别有用：

当一部执行X客户程序的新机器加入网络，它立即可以使用任何执行 X 的显示器。

相反，当一个新的显示器加入时，它立即可被任何机器上现存的所有 X客户程序使用。

4. 在网络上使用非X的应用程序 —— 终端机模拟器

如果在远程机器上的程序并不是 X客户程序或甚至连 X是什么都不知道，你仍然可以像远程的机器一样使用它们。这就需要用到 X窗口“ 终端机模拟器 ”，一个假装它是终端机的程序。这样一来，你便可以让任何程序在这个假的终端机执行。这个终端机模拟器利用 X显示输出（和得到键盘输入），当然输出也可以送到本地或远程的显示器。

27.3 从用户界面的角度概观X

本节将观察重点放到系统控制的用户界面，例如，系统如何显示有人使用，以及包含那些结构等。

X设计的目标之一就是能支持许多不同型号的用户界面。一般其他的窗口系统提供特殊的交互方法，而X则提供一般性的结构，让系统建立者据以建造所需的交互的样式。例如，在一个X系统中可从菜单中选一个动作来构建窗口，但其他对窗口的操作则全靠鼠标来做，这种灵活性允许系统开发者完全在 X的基础上产生全新的界面，也因为界面并未内建于窗口系统，因此用户在任何时刻根据他们特别的需求可选用适当的界面。例如，对于完成一些相同的工作 —— 建立、移动、重定大小屏幕上的窗口等，初学者较老手喜欢简单的系统，而 X可分别提供最适合他们的用户界面。

用户界面分为两个部分：

- 管理界面：命令最高层的窗口如何在屏幕上配置或重配置，也就是说，如何管理桌面。
- 应用界面：决定你和应用程序间交互的“ 样式 ”，即你如何利用窗口系统的设备程序来控制应用程序及输入资料给它。

27.3.1 管理界面：窗口管理器

管理界面是系统的一部分，用以控制屏幕上最上层的窗口（换句话说：如何重新配置你的桌面），这个部分在系统中称之为窗口管理器，它的功能有改变窗口的大小或位置、将窗口在堆栈中重新安排位置或将窗口改变成图标 等等。

在X中，窗口管理器只是另一个客户程序程序而已。它以及系统界面的开发和服务程序是完全分开的，因此你可以更换它们，这类似于 UNIX系统中的外壳命令行解释器。外壳只是一个用户处理程序，如果你改变它，你也改变了系统的用户界面。

1. 手动的和自动的窗口管理器

有两类的窗口管理器：手动的和自动的。手动的窗口管理器，窗口在屏幕上的位置和大小完全由用户控制，手动的窗口管理器只是用户用来完成工作的工具，大部分的手动窗口管理器

允许应用窗口重叠。

而自动的窗口管理器尽可能由它自己来控制桌面，对于屏幕的布置尽可能让用户少插手。它在新建立一个窗口时自动决定窗口的大小和位置，和当窗口移动时如何重新安排其余的窗口，通常自动的窗口管理器将屏幕分成一块块像磁砖一样的区域，也就是说安排应用窗口彼此不会重叠，而且尽量占用最多的屏幕空间。

通常当你希望告诉手动的窗口管理器你要完成什么动作时，是借助于使用菜单或者结合了按鼠标的按钮和移动鼠标指针的方法。例如，重新摆放一个窗口的位置，你可以移动鼠标指针进入窗口，按住左边的按钮，移动鼠标指针然后在新位置释放按钮，窗口管理器是如何知道这些鼠标事件的意图的？或是换个角度，服务程序是如何知道事件是来自应用窗口还是窗口管理器？

答案是由窗口管理器告知服务程序有哪些特定的 事件（按按钮等等）需要被送达，这和哪一个窗口发生的无关。这种处理称之为抓取。窗口管理器可以指定希望抓取哪一个鼠标按钮，而这抓取发生在鼠标的按钮被按下且键盘上一些特定的键（一般称为修饰键)也被按住，当按钮被按下时，抓取开始操作，服务程序送出所有鼠标的事件（包括鼠标的移动事件）到窗口管理器直到按钮再度释放，窗口管理器把这些事件的资料解释成来自用户的指令来工作。以移动窗口为例，窗口管理器在按钮按下时被告知鼠标指针的位置，而当按钮释放时再度被告知，对鼠标指针的位移做一些简单计算便可据以移动窗口。

有一件事需要用户配合，那就是鼠标和修饰键组合而成的抓取不应该为应用程序所知道，所以必需确定窗口管理器这种抓取键的组合不会和应用程序冲突，大多数的窗口管理器可以很容易地定义这些抓取的组合键，而保留给它自己使用。

2. 窗口管理器额外提供的功能

窗口管理器除了具有重新配置窗口的基本功能外，也提供额外的功能改进界面的品质。通常，加入额外功能的目的是为了降低键盘输入的需要，而改成尽量多用鼠标。

一个常见的功能是提供自己可以配置的一般性菜单，这样你只要选取一个菜单选项便可启动窗口应用程序。这个启动的命令通常包含了指示应用窗口在何处出现，大小多少，文本用什么颜色等等。所以应用程序不需要太多的用户输入便能启动。一个常见的菜单用法为，当你在网络上工作时，你可以定义一个菜单列出所有你在网络上可用的主机，这样在菜单上只要选择主机名称便能和任一主机建立连接。

3. 窗口管理器和图标

当一个窗口转换成一个图标时，图标是如何来的？窗口又发生了哪些事？

图标的结构非常简单，它只是窗口的代表图案，当系统图标化一个应用窗口时，窗口管理器只是不映射这个窗口（也就是说，告诉服务程序不再显示这个窗口）而把图标窗口映射出来。解除图标化则把上述的处理反过来。

窗口管理器通常提供缺省的图标，但是客户程序可以提供它自己的图标并建议使用它，有些窗口管理器接受这个要求，有些则忽略此请求仍用自己的图标，只把这个请求当作给窗口管理器的暗示。

当应用程序被图标化以后，它的主窗口便不再被对应出来。如果窗口管理器因任何理由中断了，则这个窗口永远也无法再对应出来了。如果希望避免这样的事情发生，当窗口管理器图标化一个窗口时，它把这个窗口加入一个名为 save set的名单，这个名单由服务程序负责维护，这样当窗口管理器被中断时，此窗口也可重新对应出来。

4. 应用程序给窗口管理器传递配置信息

就如同要求显示一个特定的图标一样，应用程序也能给窗口管理器传递其他的暗示或配置信息，这包括：

- 应用程序和图标窗口的名称。
- 当应用程序和图标窗口建立时，它们在屏幕上位置的信息。
- 对窗口大小的限制（例如，客户程序可以宣告“我所占用的窗口大小绝不可小于宽度若干x 长度若干”）。
- 对窗口重定大小的特别要求（例如，一个显示文本的窗口，可以要求在重定大小时按特定的间隔放大或缩小，以使得窗口内的字符永远是完整的一个，不致窗口边框的那一行(列) 有半个字的情况出现）。

我们可以注意到大部分重定大小或图标化的事是由窗口管理器做的，这是因为它是一个公共的客户程序。任何客户程序均可随意重定大小，但如果所有客户程序这样做，便会造成混乱，因此要这些应用程序和平共存的原则是：不要自行重定大小，把它交给窗口管理器，也就是让用户去决定。

27.3.2 应用程序界面和工具箱

应用程序界面决定了用户和应用程序间交互的样式。例如如何用鼠标选一个选项等。 X不提供标准的应用程序界面，只提供基本的结构以便建造它们。

把那些具有通用性的应用程序界面放在一起，便形成了一个工具箱，它是基础窗口系统软件中最高最有效率的层次。较低层次的细节，将被隐藏起来，因此简化了程序，同时维持界面的一贯样式变得容易。当用户控制应用程序时好像有一套“虚拟语法”一般。需要注意很重要的一点是，工具箱在编译程序的时候才能决定，所以一个客户机的应用程序界面在编译的时候就被决定了，如果不重新编译便无法改变。

MIT 版的X大多数的应用程序均使用标准的工具箱和一套来自 MIT 的工具箱软件构成要素，这使得你可以得到一个一致性的界面。除此之外，有些结构提供了定制的应用程序操作方法和设定它们的缺省值。

27.3.3 其他系统角度

本节将讨论应用程序之间传递信息所用的属性结构，窗口的树状层次组织和 X不包含在操作系统中的优点。

1. 客户程序之间的通信 —— “属性”

客户程序和服务程序之间的通信是借助于送出请求和接收事件。但有时客户程序需要和其他的客户程序传递信息，例如，正常的应用程序需要告诉窗口管理器它的位置和大小，这就需要X的属性结构了。

“属性”是一小段数据的名称，这一小段数据存在服务程序中且关联到一个特定的窗口，任何客户程序均可向服务程序查询某一特定窗口“属性”的值。

让我们看一个客户程序如何把它所喜欢的图标名称传递给窗口管理器的例子：客户程序把图标名称存到这个窗口的WIM_ICON_NAME “属性”去。当窗口管理器执行图标化这个应用窗口时，它会去找这个应用窗口的WIM_ICON_NAME的“属性”，而后显示“属性”中的图标名称。

应用程序也可以和不是窗口管理器的其他的应用程序通信，一个常见的例子是在分属不同应用程序的窗口之间做剪贴操作，一段文本从一个应用程序中“切下”稍后再“贴”到另一个

应用窗口，在此用到了“属性”，“属性”依序编成“CUT_BUFFER0”、“CUT_BUFFER1”…等等，所有的应用窗口便可借此交换信息。

最后一个例子是称为资源的属性，它用来定义应用程序的缺省值设定。在根窗口中有一个名为RESOURSE_MANAGER的属性存放着所有设定的名单，所有的应用程序都会存取它，用来做是否要执行任何设定的依据。

2. X中窗口的层次式

本节描述窗口在系统中的组织及如何建立和窗口对应用程序的影响。

所有在X中的窗口都可视为一个树状结构层次的一部分，树的根部便是根窗口，涵盖了整个屏幕，应用窗口都是根窗口的子代，上层的窗口可以拥有它自己的子窗口。

在X的设计理念下，制造一个窗口非常容易，你可以利用窗口来控制选项，像菜单、滚动条、控制按钮等等，即使是大量的选项也无妨。这种观点更有利于程序员而不是用户，但的确对用户“定制”特定的程序时有影响，在本节以后会再度提到。

为了允许应用程序有子窗口，X提供了大量的设备程序供客户程序使用，这样不但能完成一致性，也避免了相同的需求造成了重复的工作。

子窗口的位置和大小并不受父窗口的限制，子窗口可大可小，可以大过父窗口或只占父窗口的一部分，但是它会被父窗口剪裁，也就是说，子窗口所有超出父窗口的部分将会消失不见。

在实际的应用上，你可以将上层的窗口定义成几乎占住整个屏幕，就不必担心子窗口有些部分会看不到了。

另外一种方式就是把下拉式菜单定义成为根窗口的子窗口，这样菜单便可以比应用窗口还大。

3. X不是嵌入于操作系统中的

不像其他大多数的系统，X并非嵌入于操作系统中，而只是比用户层次稍高而已。更精确地说，X不需要嵌入于系统。虽然有些制造厂商可能是为了效率的原因将服务程序和操作系统结合在一起，但不嵌入于操作系统的结构有下列的好处：

- 易于安装和改版，甚至去除。这种工作不需要重新启动系统，也不会对其他应用程序造成干扰。
- 第三方厂商很容易加强它的功能。例如你的制造厂商提供的系统不够好，你可以向别人买更好或更快的版本。
- X不会指定操作系统，因此成为一种标准，这也是第三方厂商开发软件的原动力。
- 为了开发者利益。在服务程序上开发工作时，当程序中断只会中断窗口系统，不会造成机器的损坏或操作系统核心的破坏。没有操作系统核心码的程序也较易排错。

27.4 术语和符号

27.4.1 术语

在X中，一个窗口是指屏幕上的一块长方形区域，它的边平行于屏幕的边。大多数的窗口以一种颜色作为背景色，而以另一种颜色作为前景色。例如一个典型的文字窗口，背景色为白色，前景色 (也就是文字本身) 则为黑色。窗口可以有一个边框，通常边框的颜色和背景色不同。有些窗口在窗口上方可能有一个标题栏或控制栏，在某些情况下用以显示有关这个窗口的信息，你可以对控制栏作某些固定的动作来管理窗口。系统会显示在屏幕上显示一个鼠标指

针，当你移动鼠标时，整个屏幕只有一个鼠标指针在对应移动。屏幕上许多文字窗口拥有自己专用的文字光标，这些光标通常指示你输入文字的位置。

1. 几何意义：位置和大小

X用到一些有几何意义的术语来说明一个窗口的位置和大小。 大部分的X程序接受一个含有几何意义的命令行来启动它们，这个命令行说明了这个程序的窗口有多大，以及在屏幕的哪一个位置显示。通常格式如下:

宽度×高度 + x偏移量 + y偏移量

宽度和高度的单位为像素(屏幕上的一点) 或字符，视应用的状况而定。程序的说明通常会告诉你用什么单位。上述的式子是说明建立一个大小为宽×高的窗口，窗口的位置为左边框距屏幕左边界 x偏移量个像素，上边框距屏幕上边界 y偏移量个像素。例如假设一个程序以字符为窗口大小单位，则格式

80×24 + 600 + 400

其意义为: 建立一个 80 字符宽 24 字符高的窗口，并且窗口的左边框距屏幕左边界 600个像素，上边框距屏幕上边界 400个像素。

如果需要的话，也可以只指定大小或只指定位置，程序对未指定的部分会使用缺省值 ，或给你一些提示，视实际在系统中执行的状况而定。

2. 鼠标和鼠标指针的术语

有一些输入设备会在执行 X时在显示器上指出屏幕上你有兴趣的项目或区域，通常为一个有数个按钮的鼠标 (一般为三个按钮，分别称为左按钮，中按钮，右按钮)。当你移动鼠标，系统会对应地移动屏幕上的鼠标指针。接下来，我们对鼠标上的三种操作术语定义如下 :

- 单击 : 按下鼠标的按钮随即释放，按钮被按下的时间，仅有一瞬间而已。
- 按住按钮: 将鼠标的按钮按下，且一直保持按住按钮的状态。
- 释放按钮: 将先前按住的按钮释放。

通常单击用来指定屏幕上的一个对象，按住按钮再释放按钮 (一般在这期间会移动鼠标) 往往用来移动或描绘一块区域。

拖动对象: 利用鼠标指针指定一个对象，按住按钮，保持按住状态移动鼠标指针直到某处再释放按钮。做这种操作时，系统通常有一些方式来表示对象被移动，例如在拖动一个对象的期间，系统会将对象周围加上一个细线的方框。

我们常常利用拖动方式来改变一个对象的大小，通常系统显示方框，根据你的拖动动作改变大小，此种方法叫作橡皮筋法。(因为方框好像用橡皮筋做的一样)

3. 键盘的术语

标准的终端机键: Shift、Delete、Backspace、Esc 或 Escape、Return、CapsLock。

光标控制键: 采有上下左右箭头的键，如 Up、Down、Left、Right。

特殊键: 按住Ctrl或Control键，再按其他的键 (例如 A键)，用Ctrl-A表示，有些终端机有Meta键，也同样的用Meta-A表示。

27.4.2 符号

在一些情况下，你输入的命令行或系统输出的文字，因为太长而无法在同一列而必需分为数列，如果它是shell 命令，或是一段 C语言程序码，我们在第一行的最后加上一个倒斜线(\)后，在下一行继续，例如:

```
mkfontdir/usr/lib/X11/fonts/misc\
```

```
/usr/lib/X11/fonts/15dpi\
/usr/lib/X11/fonts/100dpi
```

然而极少数的情况下，我们用符号“ (contd.) ”表示本行因排版限制的缘故在下列继续，如:

```
PID TT STAT  TIME COMMAD
1901 c0 S 0:01 x :0
1902 c0 S 0:01 xterm -geometry +1+1 (contd.)
      -n login -display unix:0 -c
1903 p1 S 0:00 -sh (csh)
```

当X设置时，需要设定一些目录树。我们把目录树的顶端定为 $TOP，在我们的系统中，$TOP对应的目录为 /usr/local/src/X11，相同地，主目录参考自 $HOME。

27.5 启动和关闭X

在此假设系统管理员已经在系统上设置好了X，事实上即使不曾用过或不熟悉X，设置X也不会很困难。因此如果有必要，需自己设置X。

在还未开始前，我们需要先知道已设置好的 X的执行程序在哪里。 MIT 版缺省的目录为 /usr/bin/X11，但很多地方是用 /usr/local/bin 或/usr/local/bin/X11。 当你知道了之后，把它加到你的搜索路径里，如果你使用 C-Shell，可以在你的.login文件 (或者可能是.cshrc文件) 设定路径，如果你使用Bourne Shell，则在 .profile文件中设定。例如，在 .login文件中使用C-Shell 的命令行设定路径:

```
set path = (. /usr/local/bin/X11 /usr/ucb /usr/bin /bin)
```

如果不设定路径，X将无法正常启动。 当你设好之后，为了确定起见，先登录再退出系统一次，以便检查路径是否设定正确 (用 echo $PATH 指令)。

27.5.1 启动X

在你的显示器启动X，键入命令：

```
xinit
```

则会发生：

1）你的整个屏幕会被设定成灰色。

2）一个巨大的“ X ”光标出现。你可以用鼠标在屏幕上移动它，但按鼠标按钮或键盘都对它无影响。

3）一个xterm 终端机模拟器的窗口出现在屏幕左上角，当光标移到这个窗口时，会改变成文本光标，xterm 准备接受你的命令。

X现在已启动，你可以把xterm 这个窗口当成一个普通的终端机来使用，执行一些普通的指令。不过它最大的价值在让你可以开始执行其他的 X程序，这一点我们将于稍后告诉你。现在先来了解一下X的启动动作做了些什么。

首先，xinit在你的显示器上启动执行X服务器程序。服务器程序建立一个它自己的根窗口，并把窗口的背景色设定成灰色，把光标设定成一个大“ X ”。

在服务程序执行的期间，服务程序一直控制着键盘及鼠标，这就是你能在屏幕上移动光标的原因。但是因为目前没有任何客户程序要求了解键盘和鼠标 “ 事件 ”，所以服务程序只是追踪鼠标和光标的移动，而所有其他的键盘或鼠标输入虽然都经过服务程序处理但均被放弃， (因为没有客户程序有兴趣)，这就是按键盘或鼠标按钮没有反应的原因。

接下来，xinit 启动执行xterm 程序。xterm 对服务程序而言是一个客户程序，xterm 要求服

务程序建立一个窗口，而且保持了解在这个窗口中的鼠标和键盘事件， xterm 设定在窗口中执行一个外壳程序，当鼠标指针移至窗口之内便准备接受输入。

键盘输入被送至外壳就如同在一部真的终端机上输入一样，从外壳（及其子程序）的输出借助于xterm 显示在窗口上。 xterm 也接受鼠标输入，使得你能设定不同的程序操作参数和进行文本的剪贴。

你可以观察到系统执行这些动作的步骤。例如当在系统启动后，在 xterm窗口内执行 ps a 命令:

```
PID TT STAT TIME COMMAND
1900 C0 S 0:00 xinit
1901 C0 S 0:01 X:0
1902 C0 S 0:01 xterm -geometry +1+1 -n login -display unix:0 -c
1903 p1 S 0:00 -sh (csh)
1904 p1 R 0:00 ps
```

以上的显示说明 xinit 在主控台显示器上运行。它初始化服务程序， X显示为零。接着 xterm 在一个虚拟的终端机上执行。 xterm 启动一个外壳程序，使得它能处理你在xterm 窗口所下的命令。最后，我们执行ps命令产生上述的列表。

我们将在以后讨论更多的xterm 细节。 从现在起，我们假设xterm 被视为一个DEC VT102 的终端机，我们把重点转移到系统启动之后，我们能做些什么。

27.5.2 执行X程序的方式

你目前有一个X服务程序控制的显示器，一个叫 xterm 的客户程序允许你输入命令。本节介绍如何执行其他的X程序。

因为X的客户程序和X服务程序完全独立，所以不需要特别的动作启动它们。你可以像执行一般的程序一样执行它们。但是这些客户程序需要确实知道它们用的是哪个显示器。实际上因为xterm 一开始设定了DISPLAY 环境变量，给定了它使用的显示器名称，而其他的客户程序用此当作缺省显示器，因此你不需多做其他的事。

1. 如何执行X的时钟，xclock

我们用X的时钟当作一个简单的例子。先确定鼠标指针停在xterm 窗口中，然后输入命令：

```
xclock
```

一个小的时钟影像出现在屏幕左上角，覆盖了第一个窗口一部分。

现在有三个问题要解决：

- 第一个问题：由于xterm 这个“终端机”已经有一个程序（xclock）在执行，所以我们无法再输入其他的命令，该怎么办？

唯一的办法就是停掉xclock，但当你按下Ctrl-c或DEL 键时，xclock便会消失。要克服这种状况，你需要非同步执行xclock，用命令:

```
xclock &
```

则目前xterm 至少能接受你输入其他的命令。

- 第二个问题：如何中止xclock？

X服务器程序本身没有提供直接的界面中止应用程序，但是有一个叫 xkill的客户程序可让你终止应用程序。在xterm 窗口内输入xkill 命令便可启动这个程序。xkill 会显示一个覆盖性的方形光标，移动这个光标到任何你想终止的应用程序的窗口中，按左按钮，应用程序的窗口会消失，且应用程序和xkill 会一起结束。你也会得到如下的信息：

xkill:killing creator of resource 0x40004d

XIO:fatal IO error 32 (Broken pipe) on X server “ unix:0.0" after 207 requests (178 known processed) with 0 events remaining。

The connection was probably broken by a server shutdown or kill-client。

如果为了某些缘故无法进到应用程序的窗口内用 xkill 中止它，你通常可以用UNIX的办法：找出进程的ID，然后终止它。例如：

```
$ps a | grep xclock
1907 p2 I 0:00 xclock
1909 p2 S 0:00 grep xclock
$kill 1907
[1] Terminated xclock
$
```

• 第三个问题：如何避免时钟和 xterm 窗口重叠？这个问题换个问法是：你如何安排应用程序窗口的位置？

你可以用前面说明过的几何意义的参数来解决。例如输入命令：

```
xclock -geometry 200x300+400+500 &
```

这个命令告诉xclock建一个宽200 高300 个像素的窗口，位于屏幕左上角右边400 个像素，下边500 个像素。

如果你拥有彩色显示器，那么不妨以 xclock进行你指定和使用彩色的实验。 xclock有数种选项做彩色识别：

-bg color 设定背景颜色

-fg color 设定前景颜色

-hd color 设定时钟指针的颜色

-hl color 设定时钟指针边线的颜色

输入指令：

```
xclock -bg turquoise -fg red -hd magenta
```

你可以看到一个彩色的钟，稍后我们会再说明颜色的正确使用名称。

xclock启动之后，便不再需要和用户交互了。后面我们将介绍另一个需要从键盘和鼠标输入的小程序。

2. xcalc —— 桌上型计算器

xcalc 是一个X的计算器，移动鼠标指针到xterm 窗口，输入命令：

```
xcalc - geometry +700+500 &
```

一个像TI-30 型计算器的窗口出现了，你可以用鼠标或键盘来操作它。

使用鼠标时，你可以移动鼠标指针到你需要的计算器按钮，按鼠标左按钮表示按下按钮。如果是用键盘，键盘上的一些键对应计算器按钮，例如依序按键盘键 1、+、2、+、3 和 =键，代表了算1、2、3 的总和。由于至少目前你可以用鼠标指针指到计算器的任一按钮，因此那些键盘和计算器比较不明显的对应关系，在此不作进一步说明。

xcalc 比 xclock 有一个优点，那就是容易中止它。在计算器AC按钮上按鼠标右按钮即可中止，大部分的X应用程序均有类似的中止设备。

27.5.3 关闭X

要关闭X窗口，只要移动鼠标指针到最初xterm 的窗口，输入：

```
logout
```

则窗口消失，服务器程序终止，X也被关闭。

详细来说，xterm 查觉到外壳程序终止时，也将终止自己。而 xinit 一查觉xterm已经结束，便终止服务程序后离开。

27.6 窗口管理器基础——uwm

系统应该需要更多、更实用及容易使用的功能，在X中，这些功能由窗口管理器提供。

27.6.1 什么是窗口管理器

X系统最基本的部分——服务程序，只提供最基本的窗口功能，如建立窗口、在窗口中写入文字或画图形、控制键盘和鼠标的输入和删除窗口等。服务程序不提供用户界面，它只提供建立界面的基本结构。

我们把用户界面分为两个部分——管理界面和应用界面，下面先讨论讨论管理界面。管理界面由窗口管理器控制，提供管理桌面的功能，例如建立应用窗口，在屏幕上移动它们，重定大小等等。

你也需要能够：

- 使一个原来被遮住的窗口重新显现。
- 实用地启动或中止应用程序。
- 刷新屏幕。
- 图标化和解除图标化。

27.6.2 启动 uwm

当X启动后，你可以在屏幕上的任何外壳窗口启动uwm，因为窗口管理器也只是一个普通程序而已。你可以在执行X的任何期间内启动uwm，但通常是在一开始的时候。

现在你可以先启动X，接着在xterm 窗口内输入下列命令：

```
uwm &
```

uwm 执行后会让终端机的喇叭发出声音表示它已初始化且准备工作，但你在屏幕上看不到有任何改变。执行一个ps a命令，你可以看到现在有一个uwm 程序如下：

```
PID TT STAT TIME COMMAND
1900 co S 0:00 xinit
1901 co S 0:01 x:0
1902 co S 0:01 xterm -geometry +1+1 -n login -display unix:0 -c
1903 p1 S 0:00 -sh (csh)
1904 p1 I 0:00 uwm
1905 p1 R 0:00 ps
```

现在我们有一个窗口管理器了，接下来我们将利用它完成一些基本的操作。

27.6.3 基本窗口操作——uwm 的菜单

uwm 有一个菜单的功能，可用来管理菜单，其存取的方法如下：

1）将鼠标指针移到灰色屏幕背景的任何地方。

2）按住鼠标的中间按钮不放，一个标头为“WindowOps”的下拉式菜单将会出现。

3）继续按住按钮，上下移动鼠标指针，被鼠标指针指到的选项会以高亮度显示或以反白表示，当你释放按钮，表示此高亮度显示的选项被选择。

如果你不想选择，那就按一下其他鼠标键，或者将鼠标指针移到菜单的边框外面，则菜单将会消失。

现在选择Refresh Screen (刷新屏幕)，并且释放按钮，则屏幕闪动一下并完全重画。

27.6.4 移动窗口

在屏幕上移动窗口的步骤如下：

1）将鼠标指针移至背景，按住鼠标中间按钮，调出 uwm 的下拉式菜单。

2）选择“ Move ”选项并释放按钮，此时光标改变成“ 手指 ”形状。

3）将“ 手指 ”移动到你打算移动的窗口中，按下任何按钮，同时按住不放，窗口上出现了九字格，且光标变成十字箭头形状。

4）继续保持按住按钮，移动光标，将九字格拖动至你想摆放窗口的新位置。

5）释放按钮，窗口会跳到新的位置，同时九字格消失。

27.6.5 重定窗口大小

你可以在一维空间或两维空间中重定窗口大小。例如：你可以只把窗口加宽，或同时将窗口变高及变窄。重定窗口大小步骤如下：

1) 调出uwm 的下拉式菜单，选择Resize选项，如同移动窗口，你的光标变成“ 手指 ”形。

2) 移动光标到欲重定大小之窗口的右下角。

3) 按住鼠标按钮，保持按住状态，有三种变化发生。

- 光标变成“ 十字箭头 ”形。
- 九字格出现，但不像前节和窗口一样大，它比较小。
- 出现一个长方框，显示目前窗口的大小。

4) 移动光标，延展或挤压九字格直到大小合乎需求。

5) 释放鼠标按钮，窗口改变大小将和九字格一致，同时九字格消失。

1. 九字格的目的

在重定大小的操作中，九字格具有让你预先看到重定窗口的大小，而当你在步骤 3按下按钮时，当时光标在九字格的位置决定了你的动作：

- 当你在九字格的四个角的格子或最中间那一格按下按钮，你可以任意水平或垂直改变窗口的大小。
- 当你在九字格四边中间那一格按下按钮，你就只能在一度空间改变大小，你只能移动窗口最接近你按下按钮的格子的那一边。

2. 大小限制

那个显示目前窗口大小的长方框，其大小的单位视情况有所不同。文字窗口，其意义为若干行乘若干行字符 (例如xterm 通常为80×24 字符大小)，图形窗口，其单位则为像素 (例如xclock缺省的大小为150×150 像素)。

有些窗口的外形或大小会有所限制，例如xcalc 有最小尺寸的限制：它不允许你把窗口缩小到连计算器上按钮都无法显示的地步。xterm 虽然可以任意重定大小，但它以字符为单位，它不会允许窗口最下一行字符只出现一半的情况发生。而xclock几乎对任意大小或外形均不受限制。

27.6.6 建立新窗口

利用窗口管理器uwm的NewWindow选项，我们可以很容易地建立一个新窗口。我们在下

面描述如何启动一个新的xterm ，uwm 如何帮助你启动其他的应用程序，以及你如何控制应用窗口的起始位置和大小。

1. 建立新的xterm 窗口

建立新的xterm 窗口步骤如下：

1) 移动光标到背景窗口，调出uwm 的下拉式菜单，选择“ New Window ”选项，在释放按钮的一瞬间，有三种变化发生：

- 光标改变成“ 左上角 ”形 。
- 一个闪动的新窗口边框出现了，光标在左上角。
- 一个类似我们前面讲过表示窗口大小的长方框出现和以前不同的是，它比以前多了窗口的名称 。

2) 移动光标使得新窗口的左上角移到你所需要的位置。

3) 按一下左按钮，一个新的窗口便产生了，显示窗口大小的长方框和闪动的边框同时消失。

你可以像使用原始xterm 窗口一样使用这个新窗口来执行普通的应用程序或X的应用程序。

2. 建立可以供任何应用程序使用的窗口

我们仍然可用以前的方法 —— 在xterm 窗口的外壳程序中输入一行命令来启动应用程序。但是现在你有窗口管理器程序在执行，所以你可以用交互的方式来控制窗口的起始位置，而不需在命令行中设定几何意义的参数。 (事实上， uwm也可控制窗口起始的大小，我们会在下面描述。)

举一个例子，假设我们要在屏幕的右上角启动 xclock：

1) 在xterm 窗口中，输入命令行：

```
xclock &
```

就如同NewWindow选项一般，你可以看到一个描述窗口大小的长方框，一个“ 左上角 ”形光标，一个和时钟同样大小的闪动边框。

2) 不要按任何钮，只要把边框拖动到任何你想要摆放的位置。

3) 按左按钮，一个时钟替换了闪动边框出现。

3. 指定新窗口的大小

前面提到当你建立新窗口时，若你按的不是左按钮，会有一些奇怪的情况发生，事实上三个按钮各有不同的意义，你可以依需要做适当的选择：

1) 左按钮：按下左按钮会使得：

- 位置 —— 将窗口左上角的位置依目前光标的位置决定。
- 大小 —— 应用程序本身原先缺省的大小。

2) 中间按钮：你不应该按中间按钮。但如果你按住不放的话，你可以借助于改变窗口的右下角来改变窗口的大小，然后释放按钮：

- 位置：窗口左上角的位置依你按下中间按钮时光标的位置决定，右下角则根据你释放按钮时决定，按住按钮的期间，窗口的边框就像橡皮筋般可延展或压缩。
- 大小：根据释放按钮时的右下角决定。

如果应用程序指定了窗口最小的尺寸限制，则橡皮筋边框被压缩到比最小窗口还小时会自动消失，确保你无法建立一个比最小窗口限制还小的窗口。

3）右按钮：按右按钮会使得：

- 位置：窗口左上角依目前光标的位置决定。

• 大小：窗口的宽度为缺省的宽度，窗口的高度由光标的位置直到屏幕的底边，如果大小低于应用程序缺省之最小窗口限制的话，则用缺省的高度来代替。当然，这也意味着会有一部分的窗口超出屏幕，所以无法看到。

4. 更多有关于几何意义的参数的设定

关于几何意义的参数的设定，过去我们都是用窗口左上角的位置相对于屏幕左上角位置的方式设定，其实，我们可以用窗口的任何一个角来决定窗口位置，先复习一下几何意义的参数设定方式：

width × height <xpos> <ypos>

宽度×高度 <x位置><y位置>

<xpos> 决定了窗口水平的坐标，可用下列方式表示：

+offset ：表示窗口的左边位于距离屏幕左边offset个像素的位置。

-offset ：表示窗口的右边位于距离屏幕右边offset个像素的位置。

<ypos> 决定了窗口垂直的坐标，同样地也可用下列方式表示：

+offset ：表示窗口的上边位于距离屏幕上边offset个像素的位置。

-offset ：表示窗口的下边位于距离屏幕下边offset个像素的位置。

以下有几个例子：

100×100+50+60：这是我们过去用的方式，窗口的左上角位于距离屏幕左边 50个像素，上边60个像素。

100×100-0-0：窗口的右下角位于屏幕的右下角。

100×100-80+160：窗口的右上角位于距离屏幕右边80个像素，屏幕上边160 个像素。

100×100+20-40：窗口的左下角位于距离屏幕左边20个像素，屏幕下边40个像素。

上述例子的正负号代表了窗口的边和屏幕的边的关系，而不是偏移量的正负号，事实上偏移量有它自己的正负号，例如：

100×100+600+-50：窗口位于屏幕的中上方，且窗口的上半部超出屏幕。

100×100--50-+20：窗口位于屏幕的右下角，且窗口的下边距屏幕 20个像素，窗口的右半部超出屏幕。

27.6.7 管理屏幕空间

现在可以启动许多的应用程序，建立许多的窗口，这些窗口很可能会互相重叠。但是你有三种方法可以用来管理你的窗口，使你可以加更方便地存取它们：

• 把窗口缩小，利用前述的Resize选项。

• 把窗口“堆栈”式地排列起来，你现在需要的窗口摆到堆栈最上层，其他的放在较下层，你可以用菜单上的 Raise Lower CircUp 和CircDown来改变堆栈次序。

• 把窗口换成非常小的窗口，称为“图标”，因此所占的屏幕空间极小，但只要需要你随时可还原它们，你可以利用菜单上的NewIconify和AutoIconify选项来办到。

1. 变动堆栈中窗口的次序

窗口在屏幕上，就如同文件在你桌面上，可以互相重叠。

为了让你容易获得你想要的窗口， uwm 允许你：

• 将一个窗口移到堆栈最上层，不管它现在在堆栈的哪个位置。

• 将一个窗口移到堆栈最下层，不管它现在在堆栈的哪个位置。

• 循环堆栈，将所有在堆栈中的窗口移动一层，将最后一层的窗口移到堆栈另一端开头，

你可以向上或向下循环。

(1) 将窗口移到堆栈最上层 —— Raise

Raise 选项将一个窗口移到堆栈最上层，所以这个窗口应该变成全部可见。你可以移动任何窗口而不管它目前在堆栈何处。把一个窗口移动到堆栈的最上层的步骤是：

1) 从菜单中选取 “ Raise ” 选项，光标变成手指状。

2) 将光标移到你想要移动的窗口上。

3) 按任意一个鼠标按钮，窗口保持在原来的位置，但那些原来被其他的窗口遮住的部分均会重现，其他的窗口则被盖在下面。

(2) 将窗口移到堆栈最下层 —— Lower。

Lower 选项可将一个窗口移到堆栈的最下层，你可以移动任何窗口而不管它目前在堆栈何处。把一个窗口移动到堆栈的最下层的步骤是：

1)从菜单中选取 “ Lower ” 选项，光标变成手指状。

2)将光标移到你想要移动的窗口上。

3)按任意一个鼠标按钮，窗口保持在原来的位置，其他原来被它遮住的窗口会显现出来，而它本身的部分则被这些窗口遮住。

(3) 循环堆栈 —— CircUp和CircDown

CircUp和CircDown选项用来旋转堆栈内的窗口，所差别的只是它的 “ 方向 ” 而已。循环向下的步骤为：

从菜单中选取CircDown选项，所有在屏幕上的窗口位置均不变，但原来在最上层的窗口被移至最下层，所有原来被它遮住的窗口现在变成遮住它。

CircUp和上述成对比，它把原来最下层的窗口移至最上层，遮住那些原来遮住它的窗口。

2. 图标化窗口

虽然你可以靠着Raise 或Lower 变动窗口的顺序。但有时窗口实在太多了，为了给你自己更多的屏幕空间，你可以将那些目前不需要的窗口图标化。图标化的意义是把应用窗口换成一个非常小的窗口后摆在一边，直到再次需要用它们为止。有些应用程序拥有它们特别的图标，但是大部分都是让窗口管理器去建一个。 uwm 的缺省图标是一个把应用程序名称摆在中间的灰色长方形。

共有两种方法可以图标化一个窗口，第一种特别适合尚未图标化的窗口，第二种适合曾经图标化的窗口。

(1) 图标化一个新窗口 —— NewIconify

1) 从菜单上选取NewIconify选项，出现 “ 手指状 ” 光标。

2) 将光标移到需要图标化的窗口。

3) 按下鼠标任意钮，保持按住状态，光标变成 “ 十字箭头 ” 形，且出现一个小九字格，这个九字格代表未来的图标。

4) 保持按住按钮，将九字格拖动至你想要的位置。

5) 释放按钮，九字格会被图标替换，原来的窗口消失。

因为NewIconify让你选择图标的位置，所以它适合新的窗口；当然对任何应用窗口均可使用，特别是你想改变图标位置的时候。

(2) 图标化一个 “ 旧 ” 的窗口 —— AutoIconify

AutoIconify 会将图标放在上一次出现的位置，如果这个窗口未曾图标化过，则放在光标所在的位置。

1) 从菜单上选取AutoIconify选项，出现“手指状”光标。

2) 将光标移到需要图标化的窗口。

3) 按任何按钮，原来的窗口消失，图标出现在上一次出现的位置，若这个窗口是第一次图标化，则图标出现在目前光标所在的位置。

(3) 移动图标

一个图标就像一个窗口，因此你可以利用Move选项，像移动窗口一样移动图标。

3. 解除图标化 —— 将图标还原成一个窗口

将图标还原成一个正常的窗口，它的步骤和图标化类似。甚至在菜单上，使用相同的选项，换句话说，AutoIconify和NewIconify这两个选项，如果是在窗口的状况下选择，会变成图标，反之如果是图标的情况下，则会变成窗口。

对于位置的处理也是同样。使用AutoIconify时，当你在图标上按按钮，原来的窗口会在原来的位置出现。如果用NewIconify选项，按住按钮则会出现和原窗口大小相同的九字格，你可以拖动九字格至你要摆放窗口的位置，释放按钮则在选定的位置上出现原来的窗口。

27.6.8 中止应用程序窗口

uwm 菜单有一个选项让你删除一个应用程序窗口，当你决定不再需要或是想要删除一个窗口时，步骤如下：

1）从菜单上选取KillWindow选项，光标变成“手指状”。

2）将光标移到你想要删除的窗口上。

3）按一下鼠标任何按钮，窗口消失，内含的应用程序随之中止执行。

当窗口消失后，你可以在原来下命令的xterm 窗口看到和前面使用xkill后类似的信息。

27.6.9 激活uwm 菜单的其他方式

截至目前为止，我们激活 uwm 菜单唯一的方法就是将光标移到屏幕的背景上且按住鼠标的中按钮，但是如果一个应用窗口占用了整个屏幕，那该怎么办？你会因为找不到屏幕背景而无法激活菜单，那么什么事都不能做了吗？

答案很简单，有另外的办法激活菜单：

1) 同时按下Meta和Shift 键，按住不放。

2) 按住鼠标的中按钮，uwm 菜单即可出现 (你可以现在或稍后放开Meta和Shift键)。

3) 像前几节的方法一样选择选项。

菜单的操作方法和以前一样，只有一点不同：如果你把光标移出菜单的边，菜单不仅是消失而已，一个题为Preferences(首选项) 的菜单出现了，你可以利用这个菜单来设定一些参数。例如键盘被按时会不会有声音，喇叭的音量等等。如果你并不需要设定，将光标移出菜单，或者按鼠标的任一按钮即可离开菜单。

27.7 使用X的网络设备

X的网络特点在于让你可以在网络上的任何机器执行应用程序，而将其输出显示在你自己机器的显示器上。这是X最重要的功能之一，而且很容易使用。

以下将描述你如何指定一个远程终端机，如何实际使用这些功能。最后，我们再描述如何在网络上从其他的机器上控制或限制存取你的显示器。

27.7.1 指定远程终端机——display选项

几乎所有的X程序都接受以一个命令行的选项来指定使用哪一个显示器 (换个说法，连接到哪一个X服务器)，这个选项的格式为：

-display displayname

让我们更进一步讨论显示器名称(displayname)的格式。

你会告诉程序它的输出是哪一个显示器 (网络上任何你可以选择的显示器)。明显地，网络上指定机器的名称一定包含在内。但不止于此，因为一些 (大型)机器可以有好几个I/O 工作站，每一个工作站又拥有自己的键盘，鼠标等等；更进一步，一个工作站还可能控制了好几部终端机。综上所述，显示器名称需要包含三个元素， hostname（主机名），display number（显示器号）和screen number（屏幕号），我们将详细解释并举例说明。

1. 主机名

主机名是在网络上与显示器直接连接的机器名称，主机名也决定了应用程序和服务程序是如何连接的。简单地说：

假使服务程序在你自己本地的机器上执行，你有两种选择：

1) 省略掉主机名，系统会选择最有效率的方式和服务程序交互。

2) 定主机名为“ unix ”，系统将用UNIX域套接字作通信。(“ Unix域 ”意指套接字用传统Unix文件名称 (例如/dev/urgent)来命名。)在命名之后需加一个冒号 (:)， 即使你省略主机名，你仍需要加冒号。

假使服务程序在远程的机器上执行，你一样有两种选择，依你网络上用的通信系统而定：

1) TCP/IP ：大多数的UNIX系统使用此种通信方式。 简单的方法是用在你局域网上已知的普通名称(例如venus或saturn)。你也可以用完全的 Internet名称(例如expo.lcs.mit.edu或它的Internet地址129.89.12.73)。在名称后，需要加一个冒号。

2) DECnet ：用你连接到的机器上的DECnet节点名，在主机名加两个冒号(::)。

2. 显示器号

显示器是一组监视器，屏幕，连接一个键盘和鼠标的逻辑屏幕的组合。换句话说，即是用户工作的地方，在一个给定的CPU 上，显示器从 0开始编号，显示器号即是指哪个编号的显示器被使用，即使显示器号为 0，也不可省略。

3. 屏幕号

对于连接到显示器上数个屏幕也被从 0开始编号，屏幕号为你使用屏幕的编号和显示器号以一个句点 (.)隔开，屏幕号 为 0时可省略，若省略时，其前面的句点一并省略。

4. 例子

以下为一些显示器格式的例子：

• 假设为本地的机器，缺省屏幕为 0，以下二者均可：

```
unix:0
:0
```

• 假设你指定你自己的机器 (通常是venus)，但你需要检验TCP/IP网络的操作和明显地指定屏幕：

```
venus:0.0
```

• TCP/IP网络上，远程的机器名为pluto ， 仅有一个显示器，指定屏幕号为 0：

```
pluto:0.1
```

• DECnet网络上，显示器号为 1，缺省屏幕号 为 0：

vomvx2::1

27.7.2 实际使用远程的显示器

我们已经知道了如何指定远程的显示器，现在来练习一下：假设你是在 venus 工作，想要在saturn上执行一个例如是xterm 的应用程序。你必须在saturn执行xterm 且指定venus 的显示器，则命令如下：（为了清楚起见，下面我们的命令行包含了命令行前外壳程序对机器名称的提示。）

```
venus% xterm -display venus:0.0（注意：不完整！）
```

以上的指令是在本地的机器启动xterm ，并非在远程的机器启动，不符合需求。

如果在你的操作系统上，并未支持远程机器的操作，你可以借助于连接到 saturn的终端机输入下面的命令：

```
saturn% xterm -display venus:0.0 （注意：不完整！）
```

则xterm 会在saturn启动，在venus 上建立窗口，窗口会向 venus 的鼠标和键盘取得输入，这的确是你想要的，现在你可以回到venus 机器开始工作。

但由于你的操作系统事实上支持远程机器的功能，所以你不需要离开你的机器便可完成上述的指定，命令如下：

```
venus% rsh saturn xterm -display venus:0.0
```

以上是利用普通的远程外壳的设备程序 —— rsh。

1. 容易发生的错误

如果你搞混了，一开始发出这样的命令：

```
venus% xterm -display saturn:0.0 (不正确)
```

会发生什么事？假如这命令被接受， xterm 在你本地的机器上执行，而在远程的机器saturn上建立窗口，你在你的屏幕上只能看到外壳读到的命令行，其他什么也没有，系统是正确的工作，但不是你想要的。

如果你很幸运，你可能因没有足够的权限或 saturn上并没有服务器程序在执行，以致无法和saturn上的服务器程序连接上， xterm 会传回一个类似下列的信息而结束：

```
X Toolkit Error: Can ’t open display。
```

现在你就知道有错了。

2. 设定缺省显示器

如果你不明确地指定显示器名称，程序会以UNIX环境变量DISPLAY 来决定使用哪一个显示器，在启动xterm 时，系统会设定这个变量的内容，所以大部分情况下，你什么都不必担心。

如果你远程登录其他的机器，在其间你执行 X的应用程序，并希望回到你自己的机器上显示，那你必需明确地设定DISPLAY 变量，类似下面：

```
venus% rlogin saturn
Last login: Mon Nov 28 20:01:02 on console
  ...(在远程机器上的登录标题)
saturn%  (远程机器上的外壳提示)
saturn%  setenv  DISPLAY  venus:0.0
saturn%  xcalc  &
```

换句话说，如果不设定 DISPLAY 变量，则在saturn上执行的每一个 X程序都必须包含 -display venus:0.0 选项。

27.7.3 控制存取显示器 —— xhost

我们前面提到过有时你无法连接到特定的显示器，通常的原因是你没有相应的权限，所以

X否认你的存取。

X用很简单的结构控制存取：你指定一份可以存取你的显示器的主机名单，在这些主机上执行的应用程序均可存取你的显示器，其他不在名单上的主机则不被允许。你可以用 xhost 程序来控制存取：

允许一或多个机器存取：

```
xhost + host1 [+host2...]
```

去掉允许一或多个机器存取：

```
xhost - host1 [-host2...]
```

所有的机器均被允许存取：

```
xhost +
```

换言之，所有的存取控制均被解除。

恢复存取控制：（通常因为曾经下了xhost + 的命令）

```
xhost -
```

再次取得对存取的控制，只有先前明确地被允许的机器可供存取。

27.8 终端机模拟器——详细介绍xterm

xterm 是终端机模拟器——它是一个可以使X应用程序窗口看起来像普通终端机一样的程序，而这些应用程序无需知道有关窗口系统的功能。我们已经使用过 xterm的一小部分，在下面将更深入地探讨它所提供的特殊额外功能。并且说明许多 X程序共用的一些应用程序界面的角度。

xterm 模拟一个“哑终端机”，但它也提供许多一般终端机没有的功能：

- 设定终端机模式与特性的弹出菜单
- 可以上下移动屏幕图像的滚动条。当文字行因屏幕滚动而消失时，可以将它拉回。
- 模拟 Tektronix 4014 终端机。
- 可选择性地记录屏幕行到一个使用记录文件中。
- 剪贴文字区块。
- 可选择文字颜色，窗口背景等。
- 可选择 VT100 与 Tek 窗口字体。
- 可设定键盘。

我们首先描述选择功能的菜单结构，接着描述如何使用选择功能。

27.8.1 选择 xterm 功能——菜单与命令行选项

xterm 有它自己的内建菜单结构，可在使用期间改变设定。有三个菜单可供利用。

- xterm X11：这里的大多数选择项目为程序控制功能，例如 ：continue program (程序继续或终止程序)。欲下拉此菜单，必需同时按住 Control键与鼠标左按钮。
- modes：设定大多数终端机的特性与选择 Tektronix 模拟功能。欲下拉此菜单，须同时按住 Control键与鼠标中间按钮 (当处于VT102 窗口时)。
- Tektronix：控制 Tektronix 窗口的外表。当处于 Tektronix 窗口时，须同时按住Control键与鼠标中间按钮即可下拉此菜单。

菜单的操作类似 uwm，借助于按鼠标按钮可下拉菜单，不释放按钮移动鼠标指针至想选的项目上；释放按钮后即选定该项。然而，有一点不同的地方是，不能被选择的菜单项目 (因

为此时选择将无意义) 是以较淡的型号显示。例如：因为尚未打开一个 Tektronix 窗口，所以 Hide VT Window 项目的颜色较淡。

许多菜单的功能也能以启动 xterm 的命令行选项来设定。(事实上有某些功能仅能以命令行选项的型号去选择)。下面几节我们将告诉你可以设定不同功能的菜单选择与命令行选项的选择方式。

27.8.2 滚动 xterm屏幕

下拉 xterm X11 菜单并选择 Scrollbar 项目。高亮度显示的部分告诉你两件事：

1) 屏幕上的行数与保存在滚动条缓冲区的行数之比率。

2) 缓冲区的哪个部分目前显示在屏幕上。

你可以利用鼠标按钮移动滚动区的高亮度显示部分，以改变显示在屏幕上的文字。为简化说明，我们假设滚动缓冲区包含 100行文字。

1. 移动滚动条到指定点

假如你想移动文本到某一指定位置，例如，想看第 50 行之后的内容：

1) 移动鼠标指针到滚动条。光标变成垂直双箭头。

2) 按鼠标中间按钮，光标变成水平箭头，且高亮度显示的顶端跳至光标处。例如，假如你想看的部分从 50 行开始，你应该将光标移到滚动区的中央。

3) 假如窗口显示你所要的部分，则可以放开按钮。

4) 保持按住按钮，移动鼠标指针，高亮度显示部分跟随着鼠标指针移动 (而窗口内的文本也随着高亮度显示区而滚动)，直到释放按钮。

2. 向前滚动文本

滚动窗口内的文本使文字行往上移出屏幕顶端 ，高亮度显示区向滚动条底部移动，窗口内并显示最近打入的文本。上卷的步骤如下 ：

1）移动鼠标指针到滚动条。和前面一样光标变成垂直双箭头。

2）按下鼠标左按钮，光标变成向上箭头。

3）放开按钮，与箭头在同一行的文字移到屏幕顶端，且高亮度显示区也随着调整。注意到移动量的多少与你放开按钮时的位置有关 (若接近顶端，你可以获得的移动量小，接近底部则当然可以获得较大的滚动量)。

3. 向后滚动文本

向后滚动窗口文本，文字行由屏幕底部移出，使你可以看见先前打入的文字行。操作程序类似向前滚动，但方向相反，此时使用鼠标右按钮，出现向下箭头 。

4. 其他滚动选项

只要你已经启动滚动功能，有两个方式的菜单的选项可供利用。

- Scroll to bottom on tty output (若有 TTY 输出将输出自动卷到底) 若你目前不在滚动区的底部，稍后某些终端机的输出到达窗口时会自动地移动到滚动区的尾端。此功能为缺省的。若此功能被关闭，你要看最新的输出必须自行滚动窗口。
- Scroll to bottom on key press (按键才卷到底) 若你不在滚动区的底部，稍后你按一个键，窗口会自动移动至滚动区的尾端。此功能不缺省，但通常你的终端机设定成当你键入时回应一个字符，这些字符为TTY 输出，且将引起窗口被卷到底部。

5. 以命令行选项控制滚动

-sb：允许使用滚动条(缺省：禁用disable)。

-sl num : 保存被滚离屏幕的若干行文本(缺省为 64)。

-sk : 启用当按键才滚到底(缺省：禁用)。

-si : 启用当终端机输出时滚到底(缺省：启用)。

27.8.3 记录与终端机的交互过程——写记录

下拉 xterm X11 菜单，并选择 logging 选项(假如你现在是第二次选择同样的菜单，在 logging 选项旁边，你会看到一个沙漏标志，表示它是启动的)。从此以后，所有终端机输出除了被送到屏幕以外，也会被送至一个文件。你可以获得一个使用过程的永久记录。缺省的状况是将输出写到 xtermlog.pid 文件中。其中 pid 为xterm 处理识别码。此文件保存在启动 xterm 时的目录 (你也可以利用下面介绍的命令行选项去改变日志文件名)。

你可以借助于 xterm X11菜单停止或再度记录，反覆的停止和开始记录，你可以作选择性的记录，记录的输出永远附加在日志文件之后，每一次都不会覆写日志文件。

1. 以命令行选项控制登录使用过程

-l : 启用登录使用过程。

-lf file : 将日志文件写入指定文件，以替换缺省文件 (指定日志文件仅设定日志文件名而不启用登录功能; 必需另外使用 -l 来启用登录)使用一个管道作为登录文件。

-lf 选择项有一个特殊功能 ：假如 file 参数以管道记号 (|) 开头，则其余部分视为登录输出的一个管路。例如，假设你的系统外壳提示是 venus%，使用下列命令去启动 xterm 并记录于 cmdlog 文件，只需键入：

```
xterm -l -lf '| grep "^venus% " > cmdlog'
```

27.8.4 剪贴文本

你可以从 xterm 窗口“剪” 部分文本，即拷贝文本到一个“剪切缓冲区”，稍后可将文本“贴”回，即取回。你可以将文本贴回同一个窗口，或任何提供相同结构的窗口。你可以现在或稍后“贴” 回。 但你只有一个缓冲区，所以后来所“剪” 的资料将覆盖掉先前的资料。

1. 剪切

“剪” 一段文字的操作：

1) 移动光标至你想要 “剪” 的那一段文字的一端。

2) 按下鼠标左按钮，并保持按住。

3) 拖动光标至该段文字的另一端 ，在你移动的时候介于开始位置与光标位置间的文字会以高亮显示。

4) 放开按钮，被选到的文字维持高亮度显示，任何先前所选择的高亮度显示文本 (甚至在别的窗口中) 变为非高亮度显示。

2. 粘贴

“贴” 一段文本的操作：

1）移动光标至你想要插入一段文字的位置。

2）按鼠标中间按钮，先前被选定的文字被插入(目前的选择仍保持高亮度显示)。

当你将文本“贴” 入一个窗口，它真的就像你用键盘打入的一样，你可以使用正常的行编辑键去消除字符、单字、或整行 (当然假如“贴”了许多行，你只能编辑最后一行，就像你只能编辑最后敲入的一行一样)。

3. 剪一个字或一行

假如你想剪一个字或一行，你可以直接选择它而不需拖动过它。

- “ 剪 ” 一个字 : 鼠标指针移到一个字的任何位置，并按两次鼠标左按钮，该字即被选择。
- “ 剪 ” 一行 : 鼠标指针移到一行的任何位置，并按三次鼠标左按钮，该行即被选择。

按鼠标按钮两次与三次是所有基于鼠标的系统的共同用法，但在这个例子有特殊的功能。连续而独立的几次按与一个多次按是不同的，其差异取决于按下按钮与释放按钮的期间内有没有其他的事发生。所以下列操作算作按三次 :

按下 ... 暂停几秒 ... 释放 ... 另一个按钮

暂停 ... 释放 ... 暂停

这是很有用的，因为在按后只要你保持按下，只要更进一步借助于使用 up/down，你可以改变选择模式 (字符，字，行)。

4. 扩大选定区块或 “ 剪 ”

只要你有一个选定区块，你可以扩大它 (或缩减它)，如下 :

1) 移动鼠标指针至你想选择的新端点，它可以是在已存在的一个区域里面 (当你想缩减它时) 或外面 (当你想扩大它时)。

2) 按鼠标左按钮 : 选择区的端点调整为目前鼠标指针位置。或若以按下、拖动和放开按钮来替换按。在这个情况下，选择区跟随光标变动，且为高亮度显示。

有一个实用的技巧可以选定文字区块。首先标明你想选定的文本的一端，按鼠标左按钮，然后移至另一端按鼠标右按钮，中间的文字即被选定。 (这是扩大选定区的变相方法。开始时选择区是空的，即你已按鼠标左按钮，但没拖动通过任何文字。然后你按鼠标右按钮来扩大这个空选定区。)

5. 字或文字行边界的选择

假如你想选择一些字或文字行，你可以借助于在按按钮之前小心地定位鼠标指针来完成。但这里有一个简捷的方式 —— 再次利用多次按按钮，选择文字或文字行 :

1）将光标移到你想 “ 剪 ” 的文字的一端。

2）按下鼠标左按钮，并保持按着。

3）拖动光标至你想要 “ 剪 ” 的文字的另一端，开始点至光标间的文本为高亮度显示。

4）放开并迅速连续地再按下按钮，高亮度显示区扩展至最接近单字的边界。

5）放开并迅速连续地再按下按钮，高亮度显示区扩展至已选定行的尾端。

6）放开并迅速连续地再按下按钮，高亮度显示区回复至原来的大小，亦即选定区回到字符边界。

27.8.5 使用 Tektronix 模拟功能

xterm 可以模拟一个 Tektronix 4014 终端机和一个 VT102 终端机，使你可以用它来显示图形。当你在一个远程机器执行非-X应用程序而想在你的显示器上看图时，此功能特别有用。

xterm 为每一个 “ 终端机 ” 使用一个不同的窗口，所以你可以将所有文字显示于一个窗口，而另一个窗口显示图形。但在某个时间只有一个窗口活动着，所以所有的键盘输入或希望 “ 贴 ” 入的文本将被导引至活动窗口，甚至当鼠标指针在别的窗口时也一样。你可以使用终端机的换码序列或使用方式菜单选择你需要的窗口。你可以使用你的窗口管理器完全分开地处理两个窗口。例如，你可以图标化 VT 窗口，然而保留一个打开的 Tek窗口等。你也可以使用适当的 xterm菜单选项 (Tek Window Showing、Hide VT Window等) 去隐藏或显现一个窗口。

1. Tektronix 的特殊功能

Tektronix 菜单 (同时按下 Control键与鼠标中间按钮可得到)。它提供一些类似方式控制 xterm 窗口的功能。但它仅提供用于 Tektronix 窗口的项目。

- 改变字符的大小 ：你可以从四个不同的大小选择，范围从大字符 (缺省值) 到小字符。你可以在任何时刻改变它，甚至在一行中间。在改变之前已出现在屏幕上的字符不受影响。
- 清除屏幕 ：Tektronix 的一个特性是它的屏幕不滚动。在屏幕上有两列 (左与右) 为文本，当其中之一已写满，输出切换到另一端。然而，已显示的字符不清除，因此屏幕不久会变混乱，除非你下命令清除它。想这样做的话需于 Tektronix 菜单选择 PAGE : 屏幕会被清除，且光标被移至左上角。
- 重置“ 终端机 ”：在Tektronix 菜单选择 RESET。字符的大小与线条的型号 (可能因一个程序输出至窗口而改变) 被设回缺省值，同时清除屏幕。
- 将窗口内容复制至一个文件 ：在Tektronix 菜单选择 COPY，自从最近一次的PAGE功能后任何写到屏幕的内容都会被复制到名为 COPYyy-mm-dd-hh.mm.ss 的文件。yy-mm-dd-hh.mm.ss为当时时刻。该文件被保存到启动 xterm 时的目录下。

注意 重绘 Tek 窗口会花一些时间，当它重绘时，Tek 窗口内的光标变成一个闹钟。

27.8.6 使用不同的字体

xterm 可以让你从正常的文字与粗体文字选择不同的字体。字体选择必须有固定的宽度且彼此大小相同。你目前尚不知道如何找到可以利用的字体，但它的应用范围很广。 下面的例子我们将只用其中的两种字体，这两种字体是 core 版本提供的字体的一部分：

- 8x13 (一个字符大小为 8 像素宽，13 像素高)。
- 8x13b (一个粗体变体)。

欲指定特殊字体必须使用命令行选项 :

- -fn font : 使用 font 的正常字体，替换缺省的字体。
- -fb font : 使用 font 的粗体字体，替换缺省的字体; 缺省状态下，xterm 不区分粗体字的文本。

27.8.7 使用颜色

假如你有彩色显示器，你可以用命令行选项设定一组 (些) 窗口元素去指定颜色：

- -fg colour : 以 colour 颜色输出前景，亦即文字。
- -bg colour : 以 colour 颜色作窗口背景。
- -bd colour : 以 colour 颜色画窗口边界。
- -ms colour : 以 colour 颜色为鼠标指针颜色。
- -cr colour : 以 colour 颜色为光标颜色。

参照连接在网络上的机器对窗口设定的颜色码，你可以发现非常有用。设定鼠标与光标为显眼的颜色也是有帮助的，使你在纷杂的窗口中较容易看得到它们。

27.8.8 其他 xterm 选项

Xterm可以接受许多其他的选项。有些是设定终端机的特性，例如 :-display与 -geometry，这在前面已经讨论过。所有的这些都在xterm 联机帮助中有具体的描述，但下面是一些有用的杂项:

- -iconic : xterm 应该以图标启动的方式替换由正常方式打开窗口。 (当使用uwm 为你的窗

口管理器，图标的初始位置将决定于图标被产生当时的光标位置。我们将在“定义应用程序的缺省选项 —— Resources”里教你如何明确地指定一个图标位置。)

- -title string：使用 string 为窗口标题头，且某些窗口管理器可能将它包含在窗口标题条中。
- -C：这个窗口应该将接收的输出送到系统控制台 (例如，磁盘已满信息，设备错误等)。若你没有一个窗口具有这个选项指定，控制台信息可能直接出现在你的屏幕 (亦即不在一个固定窗口中) 并扰乱显示。若这种情况发生时，只要使用 uwm 的菜单选择 RefreshScreen 来恢复正常显示即可。
- -e prog [args]：在窗口中执行具有选择性参数的 prog 程序，替换启动一个外壳程序(此选项必须在命令行的最后，所有在它后面的部分都将视为 args 的部分)。你经常需要使用 -e 去远程登录到一个不支持X的远程系统，例如：

```
xterm -title saturn -e rlogin saturn -l root
```

27.8.9 设定终端机键盘

X 本身可让你改变键盘对照表，所以你可以针对不同的情况改变它以适合一个国家的使用习惯。但这个对照表仅决定那个“字符码”联结到那个给定的键。客户程序则可指定任意的字符串给任何键或键组 (组合键)。使用这个结构你可以设定一个 xterm。这个功能特别适用于邮件程序，或排错程序。你可以指定一般命令给功能键，或控制字符，甚至单一字符。