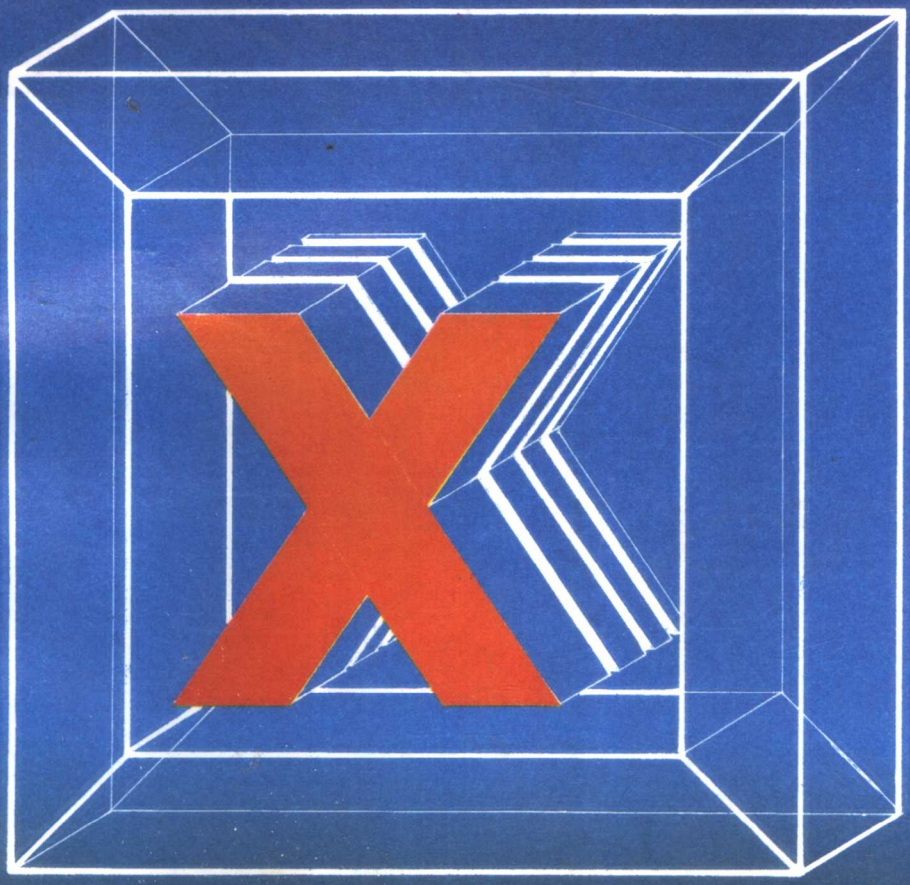
X-Window系統

C語言程序庫與協議

湯建平　張勇　編譯
宇　維　肖林



北京科海培訓中心

X 窗口系统
C 语言程序库和协议

汤建平　　张　勇
宇　维　　肖　林　　编

科海培训中心

编者序

X 窗口系统是美国麻省理工学院(MIT)与 DEC 和 IBM 公司合作开发的一种基于网络的、与设备无关的图形用户接口系统，它的问世得到世界计算机界的普遍重视和强烈反响．人们普遍把它当做一个计算机领域的里程碑．包括 IBM、SUN、APPOLLO、AT&T、DEC、HP、APPLE、XEROX、SONY 等等在内的大公司都加入了该行列，建立了 X 窗口系统的国际财团．几乎所有的台式计算机和工作站厂商都承认这一标准．X 窗口系统已正式提交 ANSI 作为一种工业标准．一些开放式软件环境组织也已接受 X 窗口系统作为一种透明的、多窗口系统的标准．它们包括 OSF(开放式软件基金会、X／OPEN、NIST 和 IEEE／0030 工作组．它们都在其开放系统环境的技术说明中包括了 X 窗口系统．

当前计算机正朝着分布式处理、并行处理、网络化和软件生产工程化的方向发展，而 X 做为一个可移植的和网络透明的软件开发环境，显然同计算机领域的总体发展相一致．这是 X 在这两年内席卷整个计算机界的一个重要原因．X 做为一个工业标准，不仅是软件工作者的工作对象，而且也是一个功能强大的软件开发环境和工具．它一方面会不断地得到扩充，由此给用户提供越来越多和功能更强的软件开发工具，另一方面又使得软件开发更加工程化．可以预计，X 将会刺激和加快各种应用软件的开发．由此，X 在计算机界的影响已显露出来．从各方面看，X 都颇有可取之处．X 将在近期内起主导作用，人们称它为九十年代的软件开发环境．

基于上述因素，以及[redacted]计算机软件人员了解和使用 X 窗口系统，我们编辑了此书及《X 窗口系统应用编程》一书．本书全面地介绍了 X 窗口系统与 C 语言的接口程序库即 Xlib，使读者对其[redacted]全面了解．该书的第二部分给出 X 协议的说明，它独立于任何编程语言，是建立同其他语言接口的基础．书中还给出其他一些重要的附录．读者通过此书可以真正地理解 X[redacted]系统．

本书在编辑过程中，得到北京科海培训中心华根娣主任和夏非彼编辑的支持与帮助，在此表示衷心的感谢．姚振珍同志参加了本书第七章的编译工作．由于时间仓促和使用与理解 X 窗口系统的水平有限，对书中的缺点和错误，敬请读者批评指正．

编者

一九九一年四月

目 录

序言

X 窗口系统或称为 X，是一个网络透明窗口系统。用 X，可以在多个窗口中并行运行多个应用程序。在一个位图显示终端以单色或彩色生成正文和图形。网络的透明性意味着可以通过整个网络使用其它机器上运行的程序，就好象这些应用程序是在自己的机器上运行一样。因为 X 允许应用程序具有设备独立性，应用程序不必再重写、重编译或者重新链接以便在新的显示硬件上工作。X 是目前国际上最受推宠的环境。

X 提供了一些函数，用以在矩形窗口的一层中产生多种字体正文和两维图形(如点、线、弧和多边形)。每一个窗口可以认为是一个"虚拟屏幕"，并且其中可能包含子窗口，可以有任意深度。窗口可以象桌面上的一叠纸那样彼此叠搭，并且窗口可以移动、改变大小和动态地重组堆栈序。窗口是廉价资源，使用几百个子窗口的应用程序很常见。例如，窗口通常用来实现单个用户界面的成份，如滚动杆(scroll bar)、菜单、按钮等等。

尽管用户可能认为自己是系统的一个客户，从网络角度来讲，用户运行的应用程序称为客户程序，它们可以使用窗口系统的网络服务功能。一个在同用户显示终端连接的机器上运行的一个程序提供这些服务，所以称作 X 服务器。X 服务器是作为用户和应用程序间的中间媒介，处理客户程序向显示终端的输出并将用户的输入(用键盘或鼠标输入)送到对应的客户应用程序处理。客户和服务器用某种形式的进程通信进行信息交换，这种对话的语法和语义是由一个通信协议定义的。该协议是 X 窗口系统的基础，并且在本书的第二部分阐述了。客户用协议向 X 服务器发送请求，创建和操纵窗口，生成图形和正文，控制来自用户的输入以及同其它客户的通讯，作为对各种请求的回答。服务器使用协议向客户发回信息以响应不同的请求，向键盘发送信息以及将其它用户输入传给合适的客户。

因为网络往返同基本请求的执行相比是一种开销较大的操作，协议最初是异步的，数据可以同步双向(由客户到服务器和由服务器到客户)传送。在产生一个请求后，一个客户通常并不等待服务器执行了请求才产生新的请求。相反，客户产生了一个最终由服务器接收和执行的请求流。服务器并不确认是否收到请求，在多数情形中，也不确认请求的执行(这是可能的，因为正在使用的基础传输是可靠的)。

要显式地定义协议，以便使查询窗口的信息的需要减少到最小。客户不应依赖服务器去取得客户最初提供的信息。另外，客户并不通过向服务器发送请求来轮询输入，相反，客户利用各种事件中所关心的对寄存器的请求，并且服务器异步发送事件通知。异步操作可能是 X 和其它所熟悉的窗口系统之间的最重要的区别之一。

为了获得最佳性能，当客户和服务器在同一机器上时，两者之间的通讯通常用共享内存实现。当客户和服务器处在不同的机器上时，通讯通过任意网络传送层进行，该传送层提供了可靠的、双向有序数据传送(通常称为可靠的双工字节流)。例如，TCP(在网间协议家族中)和 DECnet 流是两种最常用的传送层。为了在多机种环境中支持分布式计算，

设计了通讯协议以独立于操作系统、程序设计语言和处理器硬件，这样就可用一个显示终端同步地在多种硬件系统结构上，在多种操作系统下运行用多种语言编写的应用程序。

尽管 X 基本上是由一个网络协议定义的，多数应用程序设计者并不想考虑位、字节和消息格式，所以，X 有一个界面库。这个库提供了过程接口，该接口屏蔽了网络协议编码和交互传送的细节，并且自动地处理了向服务器有效地发送请求的缓冲区，很象 C 标准 I／O 库缓冲区输出，以便减少系统调用。库还提供了并不直接同协议相关的各种公用程序功能，但是在组建应用程序时很重要。这个库的确切接口对于不同程序设计语言是不同的，Xlib 是 C 程序设计语言的库，在本书第一部分介绍。

附图表示了一个完整的 X 环境的块图。每个服务器控制一个或多个屏幕、一个键盘和一个带有一个或多个按钮的鼠标器。可以有很多 X 服务器，通常网络上的每个工作站有一个。应用程序可以在任何机器上运行，甚至可在没有 X 服务器的那些机器上运行。一个应用程序可以同时与多个服务器进行通讯(例如，支持不同地区的个人的计算机会议)。多个应用程序可以同时在一个服务器上激活。

在 X 中，客户库提供了许多加进其它窗口系统的工具。用户将找不到诸如菜单、滚动杆和对话窗这样的内容的说明，而且在本书中也找不到有关特殊键和按钮序列的解释，协议和 Xlib 会尽可能地避免这样的模式判定。可以将协议和 Xlib 看作提供了丰富的能实现多种用户接口模式的机制的构造箱。工具箱(提供了菜单、滚动杆和对话窗等)、高级图形库(例如可能将抽象对象描述转换成图形请求)和用户界面管理系统(UIMS)均在 Xlib 的上面实现。尽管 Xlib 提供了基本原则，但例外情况是，应用程序将这些高级功能同 Xlib 的函数一起来编写，而不仅仅是用 Xlib 的基本函数。

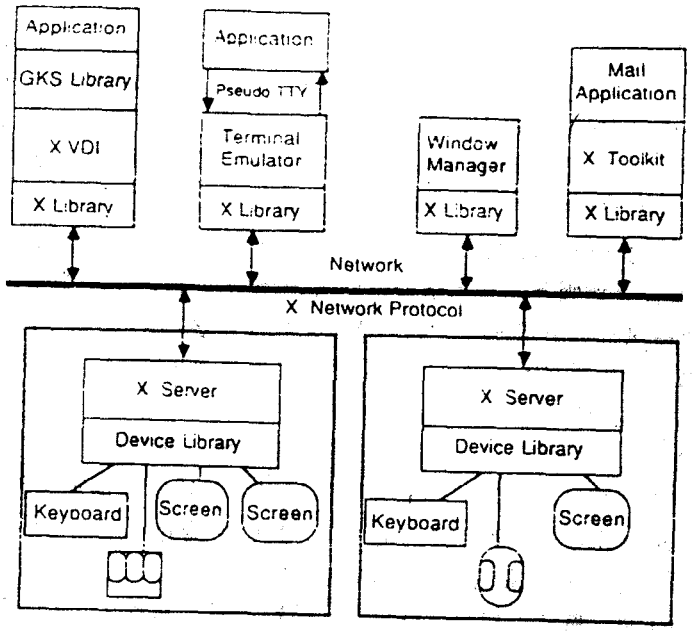


图 1　X 窗口系统块图

可以认为整个用户界面有两个主要成员:同用户的交互作用，它是应用程序的逻辑上

的内部(例如，将正文输入到一个正文编辑器中或改变一个表格中单元的内容)，以及做为应用程序的逻辑外部的交互作用(例如，对一个应用窗口进行移动和改变大小或将一个应用窗口变成一个图标)．外部用户界面在许多其它窗口系统中也是内部固有的，但同 X 并不一样．X 协议根本没有定义一个外部用户界面，相反，协议提供了一种可以用来建立各种外部用户界面的机制，设计了这些机制是为了一个称为窗口管理程序的单个客户可以提供独立于所有其它客户的外部用户界面．

一个窗口管理程序可以自动地:

- 为每一应用提供标题区、边框和其它窗口装饰．
- 提供统一的方法，对窗口进行移动和改变大小．
- 若需要，强行实施一种严格的窗口布局策略(例如，填充屏幕图形，以便应用窗口不相互重叠．
- 为应用窗口提供统一的图标．
- 为在应用程序间切换键盘提供统一接口．

利用一个合适的约定集合，可以构造对由一个窗口管理程序提供的外部用户界面不敏感的应用程序，但应用程序可以不经修改就可在多种环境中运行而且仍能运行良好．

因为协议可以处理这么多用户界，所以不可能存单个的程序、工具箱、UIMS 或窗口管理程序使用协议和 Xlib 提供的所有功能．若不明白为什么存在某些功能，不要担心:它可能支持一种你不熟悉的用户界面风格．

原理

在早期的 X 开发中，我们讨论了在 X 中应该实现什么和不应该实现什么．例如，我们并不知道菜单或终端仿真程序是否在客户程序中实现得具有足够的性能，或者当在一个网络中执行时，"拖动画线"(在请求移动鼠标时动态地展开一幅简单的图)是否可接受．

总结出了以下原理，形成早期的 X 设计:

- 如果实现者离开了新功能仍能完成实际的应用，就不要增加它．
- 确定系统不是什么同确定系统是什么一样重要．不要满足所有需要，但一使系统可扩充，这样增加的需要可以在以后的兼容版本中得到满足．
- 不是从例子中总结的东西比从一个例子中总结的东西更糟．
- 在问题不能完全理解以前，最好先不要提出方案．
- 如果工作中的一小部分的效果有一点不足，就要使用更简单的方案．
- 尽可能地分解问题的复杂性．
- 提供机制而不是模式，尤其是窗口应用客户(Client)方面的用户接口模式．

X 窗口系统是成功的，到 1986 年人们纷纷索取 X 窗口系统．因此 X10.4 系统发行了．在 X11 版本明显与 X10 不兼容时，X10 的研究就不在进行了．1988 年 3 月，MIT 正式对 X11 第二次发行，早在 1987 年 9 月非完全发行 X11 是第一次．当前 X 窗口系统的开发是由 X 财团(于 1988 年 1 月成立)监督．

由于 X 财团成员已经推出 X 上开发的几个产品，所以 X 窗口系统在计算机新闻中已显露头角．另外，开放式软件基金会(OSF)接受 DEC、HP 和 Microsoft 的建议把 X 做为它的图形用户接口的主要成员，这个图形接口是为使 UNIX 更容易使用而设计的．

从很大的程度上讲，X 是一个应运而生的窗口系统。过去，小型机以下的硬件水平不能在可接受的价格内提供一个强有力的图形网络系统。由于工作站和微机的出现，X 窗口系统就成了最可行的产物。另外，还有其它原因非常需要 X 这样的工具。

适应性强的窗口系统

当用户越来越习惯窗口式界面时，X 窗口系统提供了灵巧和适应环境的用户界面。由于 Apple Macintosh 和 Microsoft Windows 在微机上的成功，以及 Sun Microsystems 的 News 和 DEC 的 DECWindows 的成功都证明窗口界面大受欢迎。过去几年的研究表明窗口界面易于学习和使用。但 X 不仅仅是一个窗口系统，它还有比前面提到的系统有更大的潜力。在很大程度上，这些系统仅是一个单一的硬件环境和单一的产品。

把各种类型的计算机联在一起是次要的，更主要的是为软件设计与开发人员考虑诸如不同厂家计算机对他们的妨碍。而 X 窗口在这方面很成功。X 窗口系统是与操作系统无关的和网络透明的系统。通过把窗口管理系统和窗口服务器分开，这样就可以在不需仿真卡和外来的网络系统情况下，把完全不同的硬件构造联在一起。由于用户接口仅用来产生 X 调用，所以也就与操作系统无关了。

另外还由于系统设计者的信条是提供机制而不是模式，软件设计者以及在某种程度用户都可以对界面具体风格拥有最后的发言权。通过编程，应用系统本身定义窗口界面和其最终的效果。还可以建立自己所想要的界面类型。既有 Macintosh 的图标(icon)和窗口也有与 NEWS 类似的功能。至于 X 本身的界面风格并不是很重要的，在 X 的机制上可以建立各种风格的用户界面。

客户(Client)／服务器(Server)

X 窗口系统是基于简单的客户／服务器关系。客户是执行任务的应用程序，而显示服务器是控制和在显示器上显示所有输出、跟踪客户输入和刷新窗口的程序。由于 X 是一个网络环境，客户和服务器没必要出现在同一系统中(虽然在一些情况下，它们可以这样，而且也能这样)。X 窗口系统允许分布式处理。例如，Apollo 工作站上可以运行服务器并可调用联网内 Cray 超级计算机的处理功能，而在 Apollo 监视器上显示 Cray 巨型机处理结果。

X 术语与现有计算机科学术语有些不同。在微机和小型机网络中，服务器是一个硬件设备。它可以在网络中心内运行分布数据和处理网络内的工作站和终端。而 X 中却完全不是这样。基于设计者的目地，X 的服务器是控制显示设备的局部软件程序。由于网络中其它系统可以使用你的显示设备，X 服务器与局域网的文件服务器不同。

在 X 中，显示设备是键盘、鼠标和一个或多个显示屏幕，它通常与工作站有关。一个显示设备可运行用键盘和鼠标联结的多个显示屏。但单一用户则限制到单一的工作站，多个显示屏授权到一个单一显示设备。

服务器象在远程系统和局部系统上运行的程序之间(或称客户之间)的交通警察。服务器的工作是：

- 允许其它客户使用显示设备。
- 传送网络信息。

- 截取从其他客户传送的网络信息。
- 进行二维绘图，把客户程序从处理图形的工作中解放出来。
- 跟踪客户间共享的资源(如：窗口、光标、字体和图形环境)。
- 允许分布式处理。
- 当X在多任务操作系统下时，允许多任务处理(如在UNIX中，X允许调用UNIX的多任务处理功能)。

X 的组成

在许多方法上，X 是其各部分的综合。首先把 X 分成几部分，随后可看到这些部分是怎样在一起工作。

MIT 发表的通用 X 窗口系统是由 Xlib 图形子程序库，X 网络协议、X 工具箱和几个窗口管理程序构成的。应用程序编程人员通过 Xlib 图形窗口函数联接客户程序。

Xlib

Xlib 含有 300 多个匹配 X 协议请求或提供使用的函数。Xlib 实际上所做的是把 C 语言函数调用转换成实现指定函数的 X 协议请求，如 XDrawLine 是画一条线。这些函数包括窗口的创建、删除、移动和控制大小；画线和多边形；设置背景模式；跟踪鼠标等。Xlib 还允许以不同的方式存取窗口，这包括覆盖窗口和同时向多个窗口输出。它支持多种字体、图形模式、画线以及彩色和单色应用程序。

X 工具箱

X 工具箱是为编程方便而提供的程序例程库。各公司所提供的这些工具箱稍有不同，而且版本固定，但各个工具箱之间有许多相似之处。多数工具箱包括滚动窗口、按键、弹出式菜单、窗口边界和对话框等。有关 X 工具箱的讨论见科海培训中心发行的《X 窗口系统应用编程》一书。

X 网络协议

X 网络协议定义了在客户和服务器之间传送请求所使用的数据结构。从技术上讲，X 网络是异步的基于流处理通信而不是基于过程调用或核心调用接口。应用程序在这里并不做此工作，协议是 Xlib 的函数。这种结构加快了信息交换。

X 的运行环境

任何类的虚拟计算机微机、小型机、大型机等都可以运行 X。市场上几乎所有机器都配有 X。

与机器无关是它的最大优点。如果不能在 SUN 工作站或 DEC VAX 上使用 X，还可以在找到在 MS-DOS 下运行的 X 窗口系统版本。在微机中，只要是 AT 型机并有图型卡并加上 X 就可。微机上的 X 产品也在不断涌现。

第一章　Xlib 简介

X Window System(X 窗口系统)是由 MIT 设计的网络透明窗口系统,运行于 4.3BSD UNIX、ULTRIX-32、许多其它 UNIX 变种、VAX/VMS、MS/DOS,以及其它几种操作系统.

X 显示服务器可以在带有单色或彩色位象显示硬设备的计算机上运行,服务器将用户的输入分布到各客户程序中或接收各客户程序输出的要求,这些客户程序或者在同一机器上或者在网络的其它地方. Xlib 是应用程序(客户程序)用来以流连接方法同窗口系统接口的 C 子程序库. 尽管用户程序通常同 X 服务器在同一机器上运行,但不一定非要这样.

《Xlib—C 语言 X 接口》是低级 C 语言同 X 窗口系统协议接口的参考指南,它既不是 X 窗口系统程序设计的教材,也不是用户指南,相反,它不仅详细描述了库中的每个函数,而且讨论了有关的背景信息.《Xlib—C 语言 X 接口》假设了一个图形窗口系统和 C 程序设计语言的一个基本协议,其它高级抽象(例如,X 的工具库提供的)都建在 Xlib 库之上,要了解这些高级库的进一步信息,可参考有关工具库文档.《X 窗口系统协议》提供了 X 行为的最后的描述,虽然这里出现了更多的信息,但协议文档仍是主要文档.

为了介绍 X 程序设计,本章要讨论:

- X 窗口系统的概貌
- 错误
- 命名和参数约定
- 程序设计考虑
- 本文中用的约定

1.1 X 窗口系统概貌

X 窗口系统支持包含重叠窗口或子窗口的一个或多个屏幕,一个屏幕是一个物理监视器和硬件,可以为彩色的或黑白的. 对于每个显示服务器和工作站可以有多个屏幕,一个 X 服务器可以为任意个屏幕提供显示服务,带有一个键盘和一个指点器(通常为鼠标器)的单用户的一套屏幕称为一个显示终端.

一个 X 服务器中的所有窗口都安排了严格的层次. 在每一层的顶端是一个根窗口,它覆盖了每个显示屏幕,每个根窗口部分或全部由其子窗口覆盖. 除了根窗口外,所有窗口都有父辈窗口. 通常每一应用程序至少有一个窗口. 子窗口还可以有自己的子窗口,这样,应用程序就可以建一棵任意深度的树,X 提供了窗口的图形、正文和光栅操作.

子窗口可以比父辈窗口大,也就是,子窗口的部分或全部可以伸出到父辈窗口的外边,但一个窗口的所有输出都由其父窗口截取. 若几个子窗口有重叠部分,则其中一个在其余的上面并盖住其余的子窗口. 如果其它窗口没有后备存贮,则该窗口的输出区由窗口系统压

缩．若一个窗口由第二个窗口覆盖,则第二个窗口只遮住第二个窗口的父辈窗口,这些父辈窗口也是第一个窗口的父辈窗口．

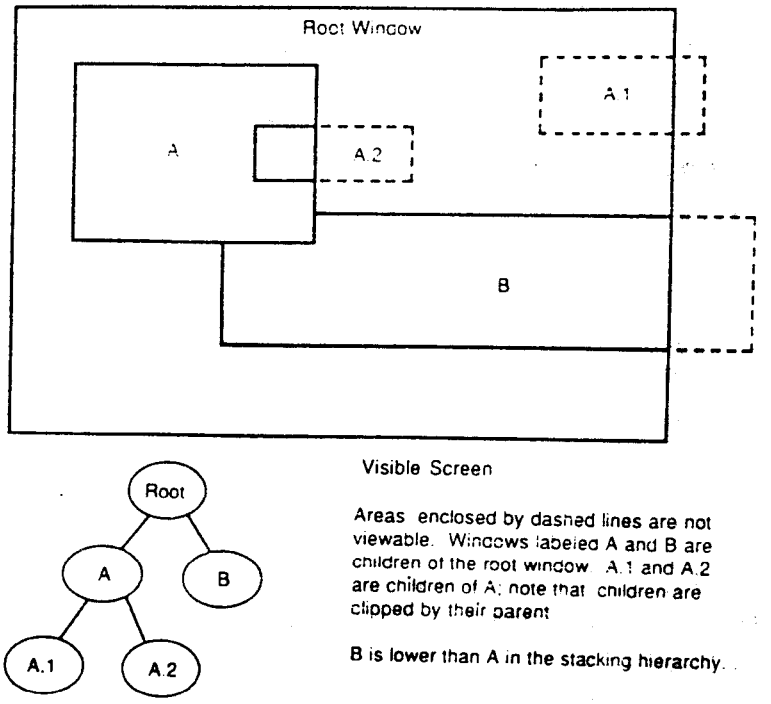


图 1.1 窗口层次

一个窗口的边可以有 0 个或多个象素宽度．象素可以为任何模式(象素映象)和颜色．一个窗口通常但并非总是有一个背景映象模式．当窗口未被覆盖时,该模式要由窗口系统重画．每一窗口都有自己的座标系统,若子窗口没有景映象,则它们遮住其父窗,父窗口中的图形操作通常由其子窗口截取．

X 并不保证保留窗口的内容,当一个窗口的部分或全部都隐藏起来并再显示到屏幕上时,其内容可能会丢失,显示服务器接着就发送给用户程序一个 Expose 事件,以提示它窗口的一部分或全部需要重画,程序必须根据要求准备重新生成口内容．

X 还提供了对图形对象的屏外存贮,称作象素映象(pixmap)．平面象素映象(深度为 1)通常指的是位象．象素映象可以在多数图形函数中使用，还可以在各种图形操作中用于定义模式和图块(tile)，窗口和象素映象一起称为可画域．

Xlib 中的多数函数仅仅是将请求加到一个输出缓冲区中,这些请求以后在 X 服务器上异步执行,返回存贮在服务器中的信息值的函数要直到收到一个显示的回答或出现错误才返回(即,将这些函数锁住)．可以提供一个错误处理器,当报告错误时调用它．

若客户程序并不希望异步执行一个请求,则可以调用 XSync 满足这个请求,XSync 一直锁住,直到以前缓冲区中的异步事件均已发生并起作用．有一个很严重的副作用:Xlib 中的输出缓冲区总是在调用一个函数时刷新,该函数由服务器返回一个值或等待输入．

许多 Xlib 函数将返回一个整数资源 ID,它允许用户引用存贮于 X 服务器中的对象,这

些资源可以为任何类型的 Window、Font、Pixmap、Colormap、Cursor 和 GContext,它们都定义在文件<X11/X.h>中. 这些资源根据请求创建,并根据请求或当断开联系时破坏(或释放),其中多数资源都可以在各应用程序中共享. 事实上,窗口都是由窗口管理程序显式地操纵. 字体和光标自动在多个屏幕间共享. 字体根据需要装入或卸载并由多个用户程序共享. 在 X 服务器中,字体通常放在超高速缓存内. Xlib 没有提供对应用程序间图形正文的共享.

要给客户程序发送事件,事件可以是一个请求的副作用(例如,重建窗口的堆栈时要产生 Expose 事件),还可以完全异步(例如,从键盘上). 客户程序要求发送事件,因为其它应用程序可以向你的应用程序发送事件,程序必须准备处理(或放弃)所有类型的事件.

输入事件(例如,按下一个键或移动鼠标)由服务器异步传到,并且要排队,直到一个显式的调用(例如,XNextEvent 或 XWindowEvent)请求它们为止. 另外,一些库函数(例如,XRaiseWindow)产生 Expose 和 ConfigureRequest 事件,这些事件也是异步到达,但客户程序在调用一个能引起服务器产生事件的函数后,通过调用 XSync 可以显式地等待这些事件.

1.2 错误

某些函数返回 Status,一个出错指示的整数. 若函数执行未成功,返回 0. 若函数返回一个 0 的状态,则它还没有修改返回参数. 因为 C 不提供多个返回值,所以,许多函数必须通过将函数的结果写入客户程序传递的存贮器返回. 缺省情况下,错误或者由标准库函数或者由自己提供的函数处理,返回指向串的指针,若串不存在,则返回 NULL.

X 服务器在运行时报告它查明的协议错误. 若对于一个给定的请求,产生了不止一个错误,则服务器可以报告其中任一个错误.

因为 Xlib 通常并不立即将请求送到服务器中(即,放到缓冲区中),所以通常错误的报告要比其出现的时侯晚. 为了调试,Xlib 为强行执行同步操作(见 8.12.1)提供了一种机制. 当可以同步时,错误产生时就报告它.

当 Xlib 探查到一个错误时,它调用一个错误处理程序,用户程序可以提供这个程序,若来提供错误处理程序,则打印出这个结果并终止程序.

1.3 Xlib 中的命名和参数约定

Xlib 遵循如下命名约定和函数语法. 若记住了函数要求的信息,则这些约定可以使函数的语法变得更可预测.

主要命名约定有:

- 为了区别X符号和其它符号,库为外部符号使用了混和大小写. 将小写留给变量,所有大写留给用户定义的宏,这同已有约定相同.
- 所有的Xlib函数以大写的X开始.
- 所有函数名和符号的开始字母均大写.
- 所有用户可见的数据结构名均以大写X开始,更一般地,任何用户可能间接引用的一律以大写 X 形始.
- 宏和其它符号不以大写X开始,为了将它们同用户符号区分开,宏中的每个词均要

大写。

- 数据结构中的所有成员或变量要小写。若需要,复合词用下划线(_)连接。
- 显示终端参数,若需要,总是参数表中的第一个。
- 在使用时,所有资源对象要紧跟在显示终端参数后面
- 当一图形正文同另一类型的资源(最常用的为一可画域)一起存在,图形正文要出现在其它资源的后面,可画域比所有其它资源级别高。
- 在参数表中,源参数总是在目的参数的前面。
- 在同时使用x,y,宽度,高度参数的地方,x和y参数总是在宽度和高度参数前面。
- 当屏蔽同结构一起用时,在参数表中,屏蔽总是在指向结构的指针前面。

1.4 程序设计考虑

主要程序设计考虑有:

- 不同厂家的工作站的键盘有很大区别。若想使自己的程序可移植,应该在这里特别留心。
- 许多显示系统限制了屏外存贮器的数量,若可以,应使用最小的象素映象和后备存贮。
- 用户应能控制其屏幕的级别。所以,应编写自己的应用程序同窗口管理打交道,而不是假设能控制整个屏幕。然而,在最上面的窗口中所做的一切应由应用程序完成。进一步的讨论参见第九章。
- X中的座标和规格(size)是16位的量,它们在接口中通常声明为int(在某些机器上int为16位),比16位数大的值被截断,规格(宽和高)是无符号量。采用这种策略可以使给定的执行层次要求的带宽最小。

1.5 《Xlib—C 语言 X 接口》中使用的约定

本文使用如下约定:

- 每一函数通过一个一般性的讨论介绍,这种讨论将它同其它函数区别开。接着介绍函数声明,并专门解释每一参数。若需要,接着在参数后面对函数进行一般性讨论。在应用的地方,解释说明的最后一段列出了函数可能产生的所有 Xlib 错误码。要看对 Xlib 错误的完整讨论,可参阅 8.12.2 节。
- 为了消除传给函数和函数返回的参数间的二义性,对于传给函数的参数的说明以词"指定"开始,对于所有返回的参数用词"返回"开始,对于既是传给函数的又是返回来的用词"指定并返回"开始。
- 指向用于返回一个值的结构的任何指针以其名中的后缀__return指示,所有传给这些函数的其它指针用于只读。一些参数使用指向既用于输入又用于输出的结构的指针,这些结构用后缀__in__out 指示。
- Xlib定义了布尔值True和False。