计算机实用技术系列丛书(二)

X Window 编程实务

文　都　等编写
燕卫华　审校

学苑出版社

计算机实用技术系列丛书(二)

X Window 编程实务

文　都　编写
燕卫华　审校

学苑出版社

(京)新登字 151 号

内 容 提 要

大部分程序员用 Xlib、Xt Intrinsics 和 widget 包结合的方式来写基于 X 的应用程序。笔者称这样的程序员为应用程序员。Xt Intrinsics 和 widget 包定义一个结构模式，该 widget 允许通过建立新的 widget 类型来扩大工具包。称建立新 widget 的程序员为 widget 程序员。

本书介绍 Xt Intrinsics 包和 OLIT widget。Sun Microsystem 当前推出的 OLIT 是 3.0 版本，它是基于 Xt Xll R4 推出的 Xt Intrinsics。

计算机实用技术系列丛书(二)

X Window 编程实务

编　　写：文　都
审　　校：燕卫华
责任编辑：甄国宪
出版发行：学苑出版社　　**邮政编码**：100036
社　　址：北京市海淀区万寿路西街 11 号
印　　刷：施园印刷厂
开　　本：787×1092　1/16
印　　张：33.625　　**字　数**：788 千字
印　　数：1～1000 册
版　　次：1993 年 12 月北京第 1 版第 1 次
ISBN7－5077－0760－1/TP・7
本册定价：25.00 元

目　录

第一章　X 窗口系统简介 …… 1
1.1　客户服务器模型 …… 1
1.2　显示和屏幕 …… 2
1.3　资源 …… 3
1.4　请求 …… 3
1.5　窗口的基本概念 …… 3
1.5.1　窗口的层次结构 …… 3
1.5.2　X 的坐标系统 …… 5
1.5.3　映射与窗口可见性 …… 5
1.5.4　保存窗口的内容 …… 6
1.6　事件 …… 6
1.7　输入设备 …… 7
1.7.1　鼠标 …… 7
1.7.2　键盘 …… 8
1.8　窗口管理器 …… 8
1.9　应用程序与 X 窗口系统的界面 …… 8
1.10　总结 …… 10

第二章　Xt Intrinsics 程序设计 …… 11
2.1　命名协议 …… 11
2.2　X 工具包程序设计模型 …… 12
2.3　Xt Intrinsics 基本函数 …… 13
2.3.1　初始化 …… 13
2.3.2　建立专用工具包 …… 14
2.3.3　管理 widget …… 15
2.3.4　事件发送 …… 16
2.3.5　设置 widget 资源 …… 16
2.4　例子：memo …… 19
2.4.1　建立和使用 memo …… 21
2.4.2　建立实用程序库 …… 22
2.4.3　事件处理程序 …… 23
2.4.4　回调函数 …… 26
2.4.5　运用转换管理程序 …… 27
2.5　应用程序上下文 …… 33
2.6　小结 …… 35

第三章　资源管理程序 …… 36
3.1　什么是资源 …… 36
3.2　指定资源 …… 36
3.2.1　名字与类 …… 37
3.2.2　资源管理程序的匹配算法 …… 39
3.3　管理应用程序资源 …… 40
3.3.1　装入资源数据库 …… 40
3.3.2　检索应用程序资源 …… 42
3.3.3　从命令行中获得资源 …… 46
3.3.4　类型转换 …… 47
3.3.5　动态资源 …… 52
3.4　widget 资源协议 …… 55
3.5　总结 …… 56

第四章　OLIT widget 集 …… 57
4.1　widget 类 …… 57
4.2　Intrinsics widget 类 …… 58
4.2.1　Core widget 类 …… 58
4.2.2　Composite widget 类 …… 59
4.2.3　Constraint widget 类 …… 60
4.2.4　Shell widget 类 …… 60
4.3　OLIT widget 类 …… 61
4.3.1　OLIT widget 元类 …… 62
4.3.2　动作 widget …… 63
4.3.3　Manager widget …… 89
4.3.4　文本控制 widget …… 103
4.3.5　Container widget …… 114
4.3.6　Popup widget …… 138
4.3.7　注册帮助 …… 156
4.4　小结 …… 158

第五章　事件的处理 …… 159
5.1　事件的定义 …… 159
5.2　事件屏蔽(event masks) …… 160
5.3　事件类型 …… 160
5.3.1　键盘事件(Keyboard Events) …… 160
5.3.2.　指针事件 …… 162
5.3.3　交叉事件 …… 162
5.3.4　焦点事件(FocusEvents) …… 166

5.3.5 爆光事件(Exposure events) …… 166
5.3.6 结构控制(Structure Control) …… 166
5.3.7 状态通知 …… 167
5.3.8 颜色表通知 …… 168
5.3.9 通讯事件 …… 168
5.4 用 Xt Intrinsics 处理事件 …… 168
5.4.1 事件处理器的使用 …… 169
5.5 消费事件回调(consume event callback) …… 174
5.6 管理事件队列 …… 176
5.7 超时的处理 …… 177
5.7.1 超时在报警方面的使用 …… 177
5.7.2 循环超时 …… 181
5.8 工作过程的使用(using Work Proceduress) …… 183
5.9 其它输入源的处理 …… 187
5.9.1 输入回调的使用 …… 187
5.10 总结 …… 196

第六章 颜色的使用 …… 197
6.1 X 颜色模型 …… 197
6.1.1 颜色表 …… 197
6.1.2 颜色的分配 …… 201
6.2 示例:一个颜色表编辑器 …… 205
6.2.1 头文件: coloredit.h …… 206
6.2.2 源文件: coloredit.c …… 207
6.2.3 类资源文件 …… 213
6.3 总结 …… 215

第七章 光栅图象的操作 …… 216
7.1 象素图 …… 216
7.2 位图 …… 216
7.3 可画区域间的拷贝 …… 217
7.4 图象 …… 222
7.4.1 图象的创建 …… 222
7.4.2 图象的显示 …… 223
7.5 摘要 …… 227

第八章 图形环境 …… 228
8.1 图形环境的创建 …… 228
8.2 图形环境的操作 …… 230

8.2.1 显示功能(display function) …… 230
8.2.2 位级屏蔽(plane Mask) …… 231
8.2.3 前景与背景 …… 231
8.2.4 直线属性 …… 232
8.2.5 填充形式 …… 233
8.2.6 字体 …… 234
8.2.7 剪取屏蔽 …… 235
8.3 图形暴露(GRAPHICS EXPOSURES) …… 235
8.4 区域 …… 235
8.5 小结 …… 237

第九章 文本与字体 …… 238
9.1 字体 …… 238
9.2 文本操作 …… 239
9.3 示例: 一个文件浏览器 …… 240
9.3.1 加入平滑滚动 …… 248
9.4 小结 …… 251

第十章 X 图形原语的使用 …… 252
10.1 点的绘制 …… 252
10.2 线的绘制 …… 261
10.3 多边形和弧的绘制 …… 265
10.4 示例: 一个简单的绘画程序 …… 267
10.5 小结 …… 278

第十一章 客户之间的通信 …… 279
11.1 原子 …… 279
11.2 使用特性 …… 280
11.2.1 特性事件 …… 283
11.2.2 使用特性以共享数据 …… 283
11.3 与事件的通信 …… 291
11.3.1 客户消息事件 …… 291
11.3.2 一个实例 xtalk …… 292
11.4 X 的选择技术 …… 305
11.4.1 基本概念 …… 305
11.4.2 用 Xt Intrinsics 进行选择 …… 307
11.4.3 给 mcmo 程序增加选择功能 …… 309
11.4.4 一个简单的剪贴板 …… 312
11.5 OLIT 的拖曳和引入技术 …… 315

11.5.1 从源应用程序中拖出 …… 315
11.5.2 引入到目标应用程序中 …… 317
11.5.3 给 coloredit 程序增加拖曳功能 …… 318
11.5.4 给 draw 程序增加一个引入点 …… 323
11.5.5 DropTarget widget …… 328
11.5.6 用 DropTarget widget 实现拖曳操作 …… 330
11.5.7 用 DropTarget widget 注册一个引入点 …… 338
11.6 小结 …… 345

第十二章 新 widget 的创建 …… 347
12.1 widget 的内部结构 …… 347
12.1.1 widget 的类记录 …… 348
12.1.2 实例记录 …… 350
12.1.3 继承 …… 352
12.1.4 数据抽象 …… 354
12.2 一个简单的 widget: Dial widget …… 355
12.2.1 专用标题文件 DialP.h …… 356
12.2.2 公用标题文件 Dial.h …… 357
12.2.3 Dial widget 源文件 Dial.c …… 358
12.2.4 使用 Dial widget …… 370
12.2.5 编译 Dial widget 程序 …… 371
12.3 继承的使用与 SquareDial widget …… 371
12.3.1 专用标题文件 SquareDialP.h …… 372
12.3.2 公用标题文件 SquareDial.h …… 372
12.3.3 源文件 SquareDial.c …… 373
12.3.4 使用 SquareDial widget 类 …… 376
12.4 元类 …… 377
12.5 小结 …… 378

第十三章 Composite Widget 的创建 …… 379
13.1 Composite widget 的结构 …… 379
13.2 Composite Widget: Row Widget …… 380
13.2.1 私有头文件: RowP.h …… 380
13.2.2 公有头文件: Row.h …… 381
13.2.3 源文件: Row.c …… 381
13.2.4 Row widget 的使用 …… 393
13.3 小结 …… 395

第十四章 Constraint Widget 的创建 …… 397

14.1 Constraint Widget 结构 …… 397
14.2 一个约束 widget: Tree Widget …… 398
14.2.1 树的私有头文件: TreeP.h …… 398
14.2.2 树的公有头文件: Tree.h …… 400
14.2.3 树的资源文件: Tree.c …… 400
14.2.4 Tree widget 的使用 …… 416
14.3 小结 …… 420

附录 A OLIT 类树 …… 421
附录 B OLIT WIDGET 参考 …… 422
附录 C 键与按钮定义 …… 521
附录 D LibX.h …… 524
附录 E PIXMAPS …… 525

第一章　X窗口系统简介

X窗口系统是一种工业标准的软件系统，它使得程序员能开发出可移植的图形用户界面．X最重要的特点之一是它与设备无关的体系结构．只要是支持X协议的硬件，应用程序就可以无需修改、重新编译或重新链接，并可直接使用它显示包含文本和图形的窗口．这种与设备无关性，再加上X作为工业标准的地位，使得基于X的应用程序能在各种包含主机、工作站和个人机在内的环境中大显身手．

X是麻省理工学院(MIT)开发成功的，得到了来自数字设备公司(DEC)的支持．其名字X以及某些最早的设计思想，都来源于斯坦福大学研制的另一个更早的窗口系统，名为W．X是在MIT的试验室里为完成计算机科学的Athena计划而设计的，该计划需要一个分布式的与硬件无关的用户界面．早期的X版本基本在MIT和DEC两公司使用，但随着第十版的推出，许多生产厂都表示对把X作为一个商业产品感兴趣．早期的X版本主要由MIT公司的罗伯特·施卡福尔、罗恩·纽曼和DEC公司的吉姆·格蒂斯设计和实施的，同时许多人为X的第十一版做出了贡献．X窗口系统的第十一版得到了硬件国际联盟和软件生产厂家的支持．软件自动生产厂家许诺通过他们的每条生产线使X成为一个用户界面的标准库．X国际联盟支持和控制X窗口系统的标准说明．X可应用于UNIX操作系统和数字设备公司的VAX／VMS操作系统，以及许多个人计算机．许多公司现在开始生产支持X协议的硬件．

X系统和其它窗口系统之间最显著的不同是：X没有定义任何特殊的用户界面风格．X提供支持许多界面风格的机制而不是单推行任何一种规划．许多苹果公司和微软公司所用的窗口系统都支持特殊的用户界面风格．相反，X提供一套灵活的原语窗口操作，它小心地避免了产生任何不适的应用程序的界面外观和感觉．另外，X提供一个与设备无关的层次，其功能相当于一个多样界面风格的基础．因此基本的X窗口系统不提供经常出现在其它窗口系统中的卷滚条、菜单或对话框这类用户界面组件．大部分应用程序依靠建立在基本X协议顶层的高层库来提供这些组件．

1.1　客户服务器模型

X窗口系统的结构是基于客户服务器模型的．“服务器”是一个单进程，它负责管理所有的输入输出设备．服务器可以创建并操作屏幕上的窗口，产生文本和图形，并且管理诸如键盘和鼠标这样的输入设备．服务器还可以提供一个位于所有应用程序和显示硬件之间的可移植层．X服务器典型地运行于配有图形显示的工作站和个人计算机上．尽管一些生产厂家提供专用的X终端来实现X服务器的所有或部分硬件或固件．

使用X服务器提供的功能的应用程序称为“客户”．客户和X系统的通信是通过一个异步字节流协议的网络连接来进行的．X支持许多网络协议，包括TCP／IP、DECnet和Chaos等．多个用户可以并行地与一服务器连接，单客户又可与多个服务器连接．

X系统结构隐藏了大部分与设备相关的客户控制的服务器和硬件实现细节．假设客

户和服务器必须共同遵循 X 协议，则任何客户可以和任一服务器通信。

除了与设备无关外，X 的分布式体系结构又可使服务器和客户在网络上任意不同的机器上运行程序．根据这一特点，服务器有许多可能的应用。例如，假设想让．学校主机上执行的交互式的教学程序可在放在每个学生书桌上的廉价的个人计算机上显示信息。在这个方案中，教学程序是一个与服务器通信的客户．每个学生通过自己本地机器上的窗口并发地与程序相互作用．同一个程序同样与在教师桌上的另一个显示设备相连，教师可以检查每个学生的进展或把全班作为一个整体来检查．在每个学生机器上的窗口为远程的教学程序提供界面．同时学生机器上的其它窗口可为其它客户提供界面。例如：每个学生可用一个窗口与运行在中心邮件服务器上的邮件系统相互作用。同时其它窗口可为在学生机器上运行的编辑器提供界面．

服务器和客户可运行在不超出局域网的分离的机器上．X 还可透明地处理 no__solocate 结构．在去欧洲的旅途，有上述特性来读自己的电子邮件是一个很有趣的机会．基于 X 的主程序运行在加利福尼亚 palo Alto 工作站上，而 X 服务器在英格兰 Bristol 工作站上工作．虽然系统受卫星影响很小，X 还是能正确工作．邮件程序根本没有意识到是在几千米以外与它打交道．

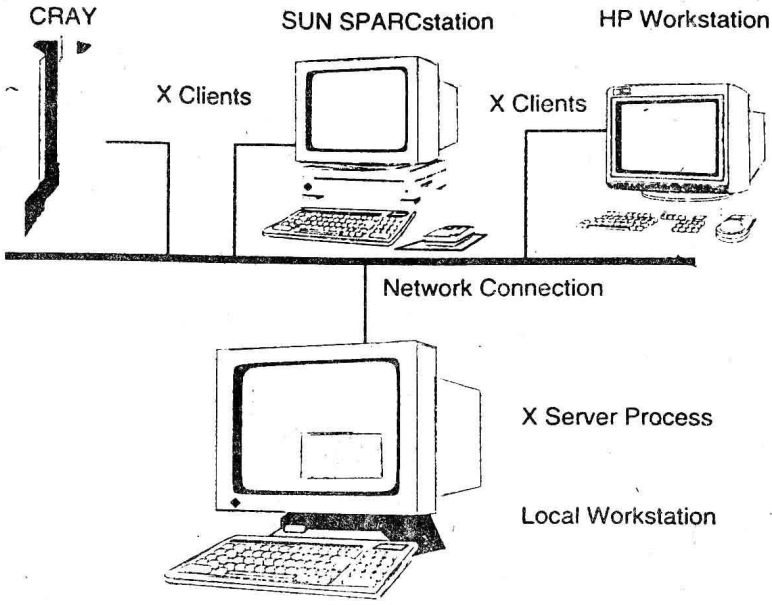

图 1.1　client server 模型

1.2　显示和屏幕

术语“显示”和“屏幕”通常指计算机用来显示正文和图形的阴极射线管(CRT)．两者可以互相替换．X 用术语“显示”意指单个 X 服务器进程，而“屏幕”是一个单硬件输出设备．单个显示屏可以支持许多屏幕。X 的术语“显示器”和“服务器”是可替换的．通常每

个 CPU 只有一个服务器.

客户与 X 服务器通信之前必须打开一个与服务器的连接. 一旦用户建立了这种联系, 就可使用服务器控制的任何屏幕. X 提供了一个安全的机制, 它否认执行在其它主机上的客户有权与显示器相连. 这种机制在前端主机 per__host 基础上工作.

1.3 资源

X 服务器可控制窗口系统所用的所有资源. 这些资源包括窗口、位图、字体、颜色和其它应用程序使用的数据结构. X 服务器说明这些资源归服务器私有, 使客户能透明地使用和共享这些数据结构. 用户程序可通过"资源标识符"(即所谓 ID)来获得每一个资源. 一个资源的 ID 是一 X 服务器分配的一个唯一的标识符.

X 服务器可以根据客户的需要来创建或清除资源. 同时当客户要求资源与服务器断开连接时, 服务器自动地清除这部分资源. X 允许客户说明资源的关闭模式, 关闭模式控制资源的生存期. 当客户切断了与服务器的连接后, 缺省模式清除分配给客户的所有资源.

1.4 请求

当客户的应用程序需要使用 X 服务器提供服务时, 客户向服务器提出"请求". 典型的请求有创建、硬件或重新构造窗口上或在窗口显示正文和图形. 客户同样可以知道窗口和其它资源当前状态的消息.

X 服务器对客户来说通常是异步运行的, 对所有客户, 它们相互之间也是异步的. 因为服务器按特殊应用程序具体到达次序来处理, 所以没有必要立即处理它们. 客户的请求将要排队, 直到服务器可以处理它们时, 客户不必等待服务器对这些请求的回答. 应用程序可要求服务器异步处理请求. 但这经常使性能较差, 因为服务器对每个请求都要周游网络连结.

1.5 窗口的基本概念

X 最基本的资源是窗口. 窗口仅代表了屏幕上的矩形部分. 不象其它窗口系统中的窗口, X 窗口没有标题、卷滚条或其它装饰性组件. X 窗口有背景色或图样的长方型显示. 每个窗口都有边界. 应用程序可以把两个或两个以上的窗口结合来建立卷滚条、标题条和其它高级用户界面组件.

X 服务器根据客户的请求来创建窗口. 服务器存放并维护代表窗口的数据结构. 客户用窗口的 ID 指定窗口. 客户可以向服务器发出更改窗口的尺寸、位置、颜色或其它特性的请求. 同样可以请求服务器把正文放到窗口之内或者在窗口中执行图形操作. 这些服务器可以根据客户的具体请求建立窗口. 假设服务器可以获得窗口的 ID, 任何客户可请求服务器操作任一窗口. 例如, X 窗口利用这一特性来控制屏幕上所有窗口的位置.

1.5.1 窗口的层次结构

X 分层来组织窗口, 称为"窗口树". 窗口树的顶层窗口称为根窗口. X 服务器为它控制的每个屏幕自动地建立一个根窗口. 根窗口占有整个物理屏幕, 不能被移动或改变尺

寸．除了根窗口外其余每个窗口都有父窗口（也称为祖先），同时也有子窗口（也称为后代）．相同父窗口的子窗口称为兄弟．

图 1.2 和图 1.3 说明了这种层次模型，并显示了几个窗口之间的关系．图 1.2 说明一组窗口如何显示在屏幕上．图 1.3 说明这些窗口形成的窗口树．窗口 A 和窗口 B 是根窗口的子窗口，窗口 C 和窗口 E 是窗口 A 的子窗口，窗口 D 和窗口 G 是窗口 E 的子窗口．同样，窗口 D 和窗口 F 是窗口 B 的子窗口，窗口 H 是窗口 F 的子窗口．

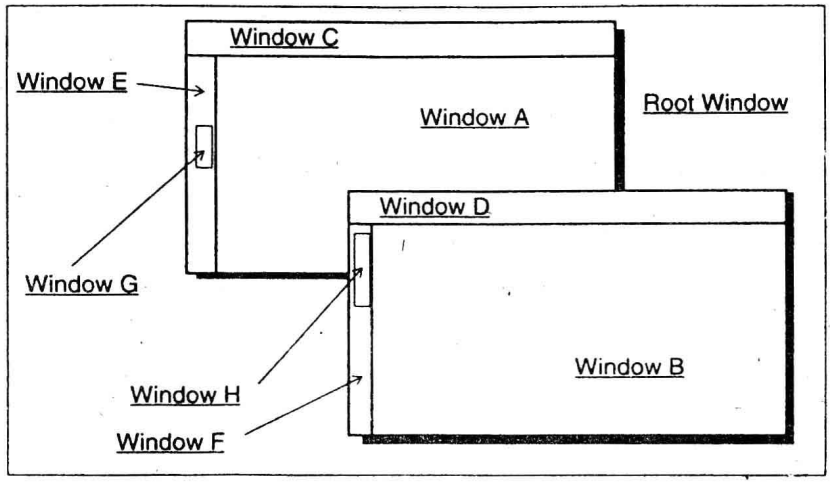

图 1.2　一个典型的窗口层次结构

X 对窗口的大小和位置有些限制，其中之一是只有位于父窗口边界范围内的部分是可见的．服务器裁剪父窗口边界的剩余部分，这就是说窗口 A、窗口 B 不完全可见，因为它们的子窗口在其上．例如窗口 C 覆盖住窗口 A 的顶端，换句话说窗口 C 的左上角和窗口 A 是同一点．

X 允许兄弟窗口之间可以覆盖，其方法好象桌上收集的纸张那样．"栈次序"决定了哪个窗口或窗口的哪一部分显示在屏幕上面（因此是可见的）．如果两个窗口在屏幕上占有重叠区域，则在栈顺序中较高的完全或部分地遮蔽较低的窗口．例如在图 1.2 中，窗口 B 的栈顺序比窗口 A 的高．客户可以要求 X 服务器改变一个窗口在栈次序中的位置（例如，升高一个窗口使其高于其它窗口）．A 窗口的栈次序随着兄弟次序改变而改变，因此从用户的观点来看，一个窗口的后代随着窗口在栈次序中上升或下降而上升或下降．

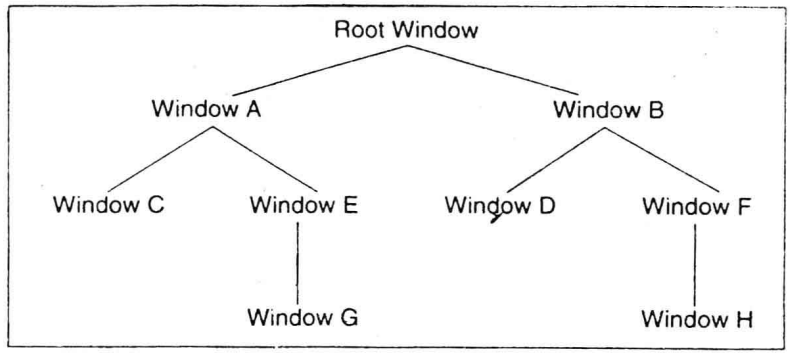

图 1.3　图 1.2 的窗口树

1.5.2　X 的坐标系统

包括根窗口在内的每个窗口都有其自己的一套坐标系统．每个窗口的左上角的坐标是(0，0)，X 坐标向左为正，Y 坐标向下为正坐标。应用程序相对某一窗口说明在屏幕上点的坐标．窗口 A 的位置（窗口的左上角）用与其父窗口坐标系统的相对位置来表达．在图 1.4 中，窗口 A 的位置相对于根窗口坐标是(50，100)，但在 A 窗口的坐标是(0，0)，每个窗口的坐标系随着窗口移动而移动，允许应用程序可以不考虑窗口的位置而在窗口中布置正文、图形或子窗口．

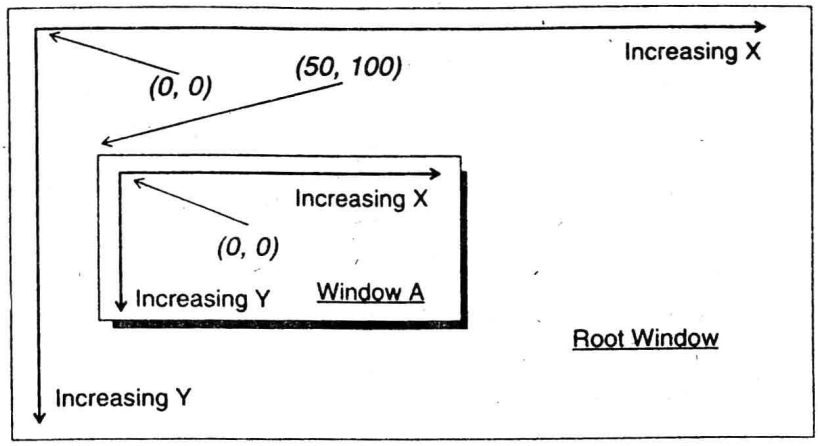

图 1.4　X 坐标系统

1.5.3　映射与窗口可见性

虽然每一个 X 窗口用屏幕上一个长方形区域来表示，但窗口对用户来说不必所有的时候都是可见的．当服务器创建窗口时，服务器在服务器内分配和初始化代表窗口的数据

结构．服务器不激活依赖于硬件的用来在屏幕上显示窗口的例程．客户可向服务器提出映射请求，以显示窗口，如客户为窗口提出映射要求，则窗口被认为映射了，但它还可能是不可见的，有以下原因：

• 在屏幕上窗口被另一个窗口覆盖，只有当覆盖它的窗口被移走，窗口不再是被掩盖的，覆盖窗口被删除，或者两窗口的栈次序发生变化、从而使覆盖窗口在栈次序中低于被覆盖的窗口时，它才是可见的．覆盖窗口从屏幕上移走或交换两个窗口的栈次序．这使覆盖窗口的栈次序低于其它窗口的栈次序．

• 窗口祖先没被映射．只有其祖先窗口都被映射，该窗口才能显示在屏幕上．映射过的窗口，但有一没被映射的祖先，它还是不可见的．当一个窗口的所有祖先都被映射，它自动地变为可见的．

• 一个窗口被祖先彻底裁剪．如果一个窗口位于任何祖先可见边界之外，它在屏幕上就是不可见的．如果祖先窗口改变尺寸直到包含了该窗口的所占区域，或窗口移到所有祖先可见范围内，则该窗口是可见的．

1.5.4 保存窗口的内容

在窗口覆盖系统中，当一个窗口被另一个窗口覆盖时，窗口的内容必须保存，以便以后恢复．许多系统以应用程序不考虑过程的方式来保存或恢复一个窗口的内容．因为窗口系统经常以“位图”或“光栅”来保存窗口的内容，这样的窗口有时被称为保存光栅窗口．

在 X 系统中由使用窗口的客户保存一个窗口的内容．X 的一些工具支持保存光栅或后援存贮的技术，X 承认这种技术，但应用程序不能依靠这一特性，因为不能保证服务器工具为所有的窗口都提供这种服务．为在屏幕上的每一个窗口存贮完整的光栅图象，这就要在服务器计算机系统的内存资源中设置一个巨大的请求，从而使窗口的数目增加．当窗口暴露时服务器能有效地通知客户，并依靠用户重新显示窗口的内容．在任何时候，每一个 X 客户必须准备重建窗口的内容，虽然这是极少有的问题，但这增加了应用程序员的负担．大部分应用程序保留了他们窗口的内部表示．后援存储特性最好用于支持计算极多的应用程序，这些程序很难迅速重建他们的输出．对不支持后援存储的服务器来说，这样的应用程序必须经常排序，以关闭屏幕图象来存储窗口的当前内容．

许多 X 服务器支持存储隐匿内容．存储隐匿内容是一种在特殊窗口下保存屏幕上图象的技术．当窗口移到新位置或从屏幕上移开，图象还可以恢复．这是通过在屏幕上区域被窗口覆盖之前取该区域的瞬像来完成的．对存储隐匿内容工作来说，所有在屏幕上的窗口及屏幕的状态在采样瞬象和图象恢复之间保持稳定．存储隐匿内容主要用于建立弹出式菜单和建立小的瞬间窗口来获得平稳的视觉效果．

1.6 事件

X 服务器和客户应用程序之间的通信是通过向客户应用程序发送“事件”来进行的．服务器产生事件作为用户行为的直接或间接结果（例如在键盘上接一键移动鼠标或按鼠标按钮）．服务器也可产生事件来通知客户窗口的状态的改变．例如当窗口的内容需要恢复时，服务器向客户发送 Expose 事件．X 支持 33 种事件并提供允许客户定义其它事件类型的机制．

服务器通过把事件放在客户可读的先进先出队列中来把事件发送给客户，每个事件由报告事件类型包、事件发生的窗口和特殊事件类型的其它具体数据组成.

大部分 X 应用程序是事件驱动式的. 设计为直到事件发生时才给事件响应，并等待下一事件，事件驱动式为交互式应用程序提供一个自然模式. 在第五章将详细说明事件和串驱动模式.

1.7 输入设备

X 支持各种输入设备. 根据实现不同，一个服务器可以支持图表、轨迹、扫描仪及其它数据输入设备和点设备. 最普通的输入设备是键盘(用于正文的输入)和鼠标(用于定位设备和选择设备).

1.7.1 鼠标

鼠标是用于用户指向在屏幕上的位置，并通过按按钮发出命令的设备. 用户通过控制图形在屏幕上的位置来指向屏幕位置，称为"指针(鼠标)".

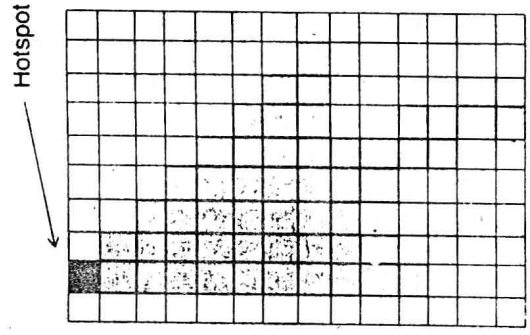

图 1.5 一个典型的鼠标光标

另外 X 经常用术语"指针"，来指"鼠标"或"子图形". 用户通过移动桌上的鼠标来控制鼠标的位置服务器保存鼠标并跟踪鼠标器的位置. 当鼠标进入或离开窗口、改变位置或用户按下、松开鼠标按钮时，客户可以请求服务器报告事件. 客户也可以查询服务器来判定鼠标当前的位置，同时客户也可改变鼠标的外型和大小，这种特性经常用于指示当前任务或应用程序的状态. 例如 (见图 1.5)，当用户定位在屏幕上的位置时，鼠标可以假设为对话箭头的形状，当用户用卷滚条时，鼠标为一个垂直的箭头. 当应用程序忙时鼠标变为一个沙漏.

每个鼠标都有一个热点，热点在鼠标范围之内定义屏幕上鼠标的精确位置. 如果鼠标的热点在窗口或其子窗口可见的部分之内称为鼠标被窗口"包含". 鼠标是包含热点的最小窗口.

1.7.2 键盘

当键改变状态时，服务器产生一个事件．事件结构的信息中包括定义按下、松开键的代码．客户如果需要可把这种代码转换成 ASCII 字符码．键代码与任何特殊键盘安排是相互独立的，任何特殊键盘安排允许应用程序处理由不同生产厂家生产的各种键盘．

1.8 窗口管理器

窗口管理器允许用户控制窗口在屏幕上的大小、位置．在大多数窗口系统中，窗口管理器和窗口的其余组成部分是不可分的．在 X 中窗口管理器是一个普通的客户程序．X 提供某些特征允许窗口管理器控制窗口的大小和位置．例如窗口管理器可以要求 X 服务器间接要求窗口服务器处理窗口结构，而不是直接响应请求．如果应用程序为窗口提出映射要求，而窗口管理器要求这事件是可更改的．X 服务器向窗口管理器发送 MapRequest 事件．窗口管理器就有机会在窗口映射之前，采取某些措施或者拒绝映射窗口．一些窗口管理器利用这一特性并根据已定出的一套规则来设置窗口或更改窗口的尺寸．例如 tiling 窗口管理器可以先重新安排或改变在屏幕上已有的其它窗口来确保没有窗口被覆盖．许多窗口管理器根据这一特性在映射前给窗口加框或标题．

窗口管理是一个复杂的课题，不仅影响用户如何与系统接口，也影响应用程序之间以及与 X 服务器如何接口．ICCCM 定义所有窗口管理器和应用程序必须遵循的协议，来使它们正确地交互作用．实际上这些规则大部分与设计窗口管理器以及用户界面包或直接从 Xlib 库中选择程序的程序员有关系．

1.9 应用程序与 X 窗口系统的界面

虽然 X 服务器协议是它定义在网络包和字节流的层次上的，程序员可以提供其窗口系统界面的库为应用程序的基础．最广泛的与 X 的低层接口是标准 C 语言库，称为 Xlib．Xlib 是一组广泛的函数，可以直接存取、控制显示、窗口、输入设备．LISP 和 ADA 也有同样的库．

虽然应用程序可用 Xlib 建立应用程序，但这种相对来说低层的库是枯燥的，且很难正确运用．正确处理一个窗口管理器的协议需要几百行的代码，许多程序员喜欢用 X 设计的高层工具箱．这本书除了讨论 X 工具箱外，还讨论 InterViews（斯坦福大学建的），Andrew（Carnegie Mellon 开发的），XView1(Sun Microsystems 开发的)等．这些工具箱的是基于 Xlib 而且易于运用的，它们对程序员隐藏了许多实现的细节．

本书将讨论标准 X 工具箱．X 工具箱由两部分组成：一个称为 Xt Intrinsics 的层和一组称为 widgets 用户界面的组件集．Xt Intrinscs 支持许多不同的 widget 组．本书中的例子用到 OLIT 的 widget，其它流行的 widget 包括 Motif（来自开放软件基金会(OSF)）、Athena（MIT X 软件免费提供）．从应用程序员观点来看大部分 widget 集提供相同的功能．熟悉一个 widget 集的程序员将很快学会运用其它专用 widget 集．Xt Intrinsics 和 OLIT widget 工具包都是用 C 写的并建立在 Xlib 的顶部．OLIT widget 说明包括：卷滚条、菜单和按钮等用户界面组成．Xt 通用工具箱提供一个允许程序员结合以上组件来产生一个完全的用户界面的框架．图 1.6 基于 widget 集和 Xt Intrinsics 基础上