第一部分 AUTOCAD对话框设计

第一章 DCL概论

1.1 引 言

在过去若干年中，Macintosh窗口画面曾经风靡了美洲大陆，而在中国，也有许多人喜爱在计算机上使用这种画面，这主要因为它操作简便，用户界面友好的缘故。不久前，Microsoft公司推出的Windows也采用了这种用户友好的窗口设计，以方便非专业人员操作。AutoCAD也是如此，第9版就开始提供所谓下拉式菜单及图标菜单的操作方式，以改善其界面，但这毕竟与窗口的含义有所不同。现在第12版又提出了对话框语言，以使用户自行设计对话框功能，这就是本书要介绍的DCL。

1.2 DCL的作用

1.2.1 设计对话框

对话框是用Dialog Control Language (DCL，对话控制语言)编写的ASCII文件(文本文件)定义的。对话框中的DCL描述将定义对话框以什么方式出现以及它包含的内容，例如按钮、列表、文本等。在第四章将说明有关DCL语法。

对话框的设计是有限制的，它的大小及各部分的布局(layout)是根据一个最小的位置信息来自动完成的。正如由字处理软件来格式化一段文本一样，布局也同样基于此一致性的原则。用户不必精确指定每个部分的大小或(X,Y)位置，所以，如同使用字处理软件一样，该自动布局可帮助用户处理设计上的不协调之处。

1.2.2 使应用程序支持对话框

在某些范围内，对话框的一部分可定义它的行为。例如，可以定义能"按下"的按钮，显示出一个列表框以便用户作一些选择等。但是，一个对话框的使用方式及其行为，实际上完全由它所属的应用程序所决定。AutoCAD Development System (ADS)则提供了处理对话框的功能(包括显示对话框、响应用户的选择等)。这个可编程的对话框(Programmable Dialog Box，PDB)功能或结构将在第六章说明。

注意 对话框严格地限制于"交互式"这种使用方式。一个脚本文件(Script)也可启动一个对话框，但无法对它加以控制，也无法在它被打开后提供输入。对于AutoLISP (com-

mand)函数及 ADS ads_command( )和 ads_cmd( )函数,情况亦如此。

AutoCAD 所支持的内部对话框以及用户定义的对话框,将视用户使用的计算机而有所不同。一个单独的 DCL 描述以及应用程序可在所有不同类型的计算机上定义一对话框和它的功能。从另一观点来看,对话框的实际外观完全依赖于每种计算机的图形用户接口(Graphical User Interface,GUI)。外观上的任何差异都是由于不同计算机所使用的不同显示驱动程序所导致的,所以,用户无需改变所使用的对话框设计或应用程序。例如,图 1-1-1 示出在两种不同类型的计算机上使用相同 DCL 语法的相同对话框。

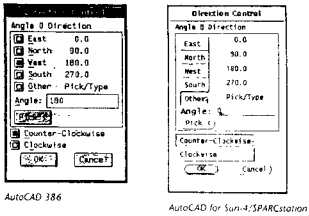


图 1-1-1 显示在不同类型计算机上的对话框

用户为什么希望在应用程序中使用对话框呢?原因之一就是为了维持对主机系统的观感,尤其是让应用程序运行于频繁使用窗口和对话框的系统中时(例如 Windows,the Macintosh 或 OPEN LOOK 等软件)。正是由于这些系统,使用户对拥有图形用户接口的应用软件的需求不断增加。另一个原因是,使用对话框比要响应连续提示文本的方式来得更容易、更自然,也更快一些。

设计一个对话框时,也需要考虑到用户在输入数据时顺序将会发生变化。这样会增加设计难度,而比起一般的设计似更不具"线性"性质,但可反映出用户的工作方式,所以在经过一些练习后,它也可变成较直观的操作方式。

若在开始设计之前,同时也对对话框和应用程序作详细的计划,则可节省时间并省去不少麻烦。若用户从未有过设计 GUI 应用程序或对话框的经验,请参考 1.4.6 节。

1.3 对话框简介

读者可能对 AutoCAD 对话框的使用方式已十分熟悉,它们的使用完全是自说明性的。图 1-1-2 示出了由标注部分所组成的一个标准 AutoCAD 对话框。在对话框的建立和设计过程中,对话框由控件(tiles)组成。

以下分别说明图 1-1-2 中各不同成分的含义和使用方式。

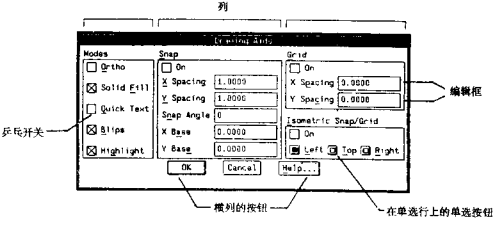


图 1-1-2　一个典型的标准对话框

有些计算机可让用户用键盘控制对话框，在此我们假定用户的计算机属于这种类型。

有些按钮对所有对话框来说是一条命令，例如，Tab 键(制表键)一般可从一个控件移到另一个控件；而按下空格键(Spacebar)则可连通或连断乒乓开关(toggle)。有两种用户熟悉的按钮，一个是接受键(通常是回车键)，用以接受对话框，另一个是取消键(在某些类型的计算机上是 Esc 键，其他的则是 Ctrl—C 键，也可能两者都是)，用以将它取消。一般来说，接受一个对话框即表示要保持用户的输入并以某种方式使用它们的值，取消一个对话框表示要放弃用户所输入的任何值。

当 AutoCAD 第一次显示一个对话框时，其中一个控件将含有初始的键盘输入光标，直到用户将光标移到另一个控件上时才会影响到这个初始控件。欲在控件之间移动，请按下 Tab 键，以移到另一个域上；也可按下按钮标号的首字符。例如，在上面的 DrawingAids 对话框范例中，可通过按下 L、T 或 R 等键，以在 Isometric Snap/Grid 控件中选择一个互锁按钮。有些计算机可能还要求同时按下 Alt 键，亦即必须按下 Alt—L、Alt＋T 或 Alt＋R 才可以移到互锁按钮上。除非选取一个按钮，否则可以改变光标，以在控件之间移动，但请勿作任何选择。

欲要选择一个所需的控件，必须按下接受键。对某些控件来说，双击鼠标左按钮相当于按下接受键。

在建立了一个自定义的对话框后，可以指定它的标号，控制哪个控件将含有初始光标，是否每个控件都可用 Tab 键到达，以及一个控件当前是否应被赋予功能(被解除功能的控件在显示时呈灰色)。

欲从一个 AutoCAD 画面中激活一个可编程的对话框以取代一个图形画面，仅需用一个 AutoLISP 函数或外部 ADS 函数来处理对话框即可。最简便的方法就是将该函数定义为一条 AutoCAD 命令(C:xxx)。例如，下述片段来自缺省的 acad.mnu 文件，显示文件下拉式画面的开头：

```
* * * POP1
[Pile]
```

```
[New...]^C^C_new
[Open...]^C^C_open
[Save...]^C^C_qsave
[Save As...]^C^C_saveas
[Recover...]^C^C_recover
[...]
```

在此范例中，每个选项都发出一条可处理对话框的命令（事实上，除了 New 选项外，所有选项都通过 getfiled 函数来激活标准的 AutoCAD 文件对话框）。

大部分支持 AutoCAD 的计算机对于它们所显示的对话框都有各自设置颜色的方式；对于那些不能设置对话框颜色的计算机（例如 DOS 386），若希望改变对话框的颜色，则可使用 AutoCAD 命令（DLGCOLOR）来改变它。

第二章　对话框的组成

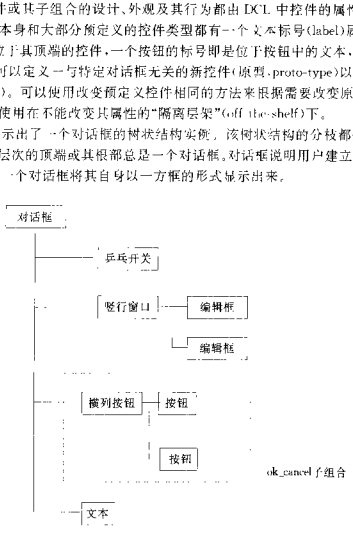
2.1　引　言

一个对话框是由位于其中的控件以及互锁按钮、乒乓开关或编辑框等部分所组成的。其基本控件类型(如按钮、编辑框、列表框、图象等)都是由 PDB 功能预先定义的。本章将说明它们的特点和使用方法。可以用包含(或不包含)的封闭对话框(或边界)将数个控件组合成更复杂的控件。这些组合本身也可被视为单一的控件。所以对话框可以被递归地组织成一个层次系统或树状结构,而位于树状结构中最顶端的就是对话框本身。

一个控件或其子组合的设计、外观及其行为都由 DCL 中控件的属性(attributes)定义。例如,对话框本身和大部分预定义的控件类型都有一个文本标号(label)属性,而一个对话框的标号即是位于其顶端的控件,一个按钮的标号即是位于按钮中的文本,依此类推。

DCL 也可以定义一与特定对话框无关的新控件(原型,proto-type)以及控件组(子组合,subassemblies)。可以使用改变预定义控件相同的方法来根据需要改变原型的属性。另一方面,子组合仅使用在不能改变其属性的"隔离层架"(off the-shelf)下。

图 2.1 示出了一个对话框的树状结构实例,该树状结构的分枝都是已预定义好的控件,而该树状层次的顶端或其根部总是一个对话框。对话框说明用户建立在反映此树状结构的 DCL,同时一个对话框将其自身以一方框的形式显示出来。



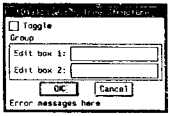


图 1-2 1　对话框的树状结构

2.2　对话框的组成

AutoCAD PDB 功能可直接支持预定义的控件，它们的定义注释可在文件 base. dcl 中找到。

当用户按下一个按钮选定了一个可活动的控件之后，对话框就会通知处理该对话框的应用程序作出响应。这就是一般所谓的动作(action)或响应(callback)。任何预定义的活动式控件都可以拥有一个相关联的动作。而项动作的效果是用户看得见的(例如，更新一个状态)。动作是伴随一个可引发此动作的原因码(reason code)而产生的，该原因的含义完全由可引发该控件的种类而定。

以下介绍对话框的各组成部分：

名称　**按钮(Button)**
图示： OK
在 DCL 中的名称　button

说明：

button 是一类似按钮的控件。此按钮的标号可指定出现在按钮之内的文本。此按钮功能对于那些可被用户立刻看见的动作来说是很方便的。例如：退出对话框，再进入到另一个对话框等等。

当用户完成使用(或读取)对话框时，对话框至少应包含一个 OK 按钮，以便用户单击。有许多对话框也会包含一取消(cancel)按钮，可使用户在不保存任何改变的情况下，退出对话框。

名称　**编辑框(Edit Box)**
图示： Arrow size　0.1800

在 DCL 中的名称	edit_box

说明：

编辑框是提供用户可在其中输入或编辑一行文本的场所。一个可选择的标号会出现在编辑框的左边。如果所输入的文本长度超出了编辑框的范围，它还可以水平滚动。

在某些 GUI 中，类似的控件被称为文本框、文本域或编辑域。

名称 图象按钮（Image Button）

图示：

在 DCL 中的名称	image_button

说明：

此控件是一个可显示图案图象的按钮。

在某些 GUI 中，类似的控件将被称为图标、图片、或雕刻文本。当用户选择了一个图象按钮后，程序就可以取得该选定点的坐标。这是相当有用的。例如，用户显示一个缩影图并赋予不同的含义，从而在其中选择不同的区域。

名称 列表框（List Box）

图示：

在 DCL 中的名称	list_box

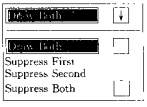
说明：

一个列表框就是一个由若干行字符串组成的列表。其目的是要显示一个列表以便用户可从中选出一项。通常列表会有不同的长度，但是，当一个会在对话框中占用很多空间的不同种类控件出现时（例如一组互锁按钮），列表框就会使用固定长度的列表。当用户选定一行后，它以高亮度显示。一个列表框可以包含比它所显示的行数更多的行。因此，通常有一个滚动条出现在列表框的右边（仅当列表不能一次显示全部内容时，滚动条才能滚动）。通过推动滚动条图标或连击它的箭头符号，可以在整个列表框

中来回的滚动。根据应用程序的设计要求，它也可以让用户选取多行。在某些 GUI 中，类似的控件被称为滚动框或滚动列表。

名称	下拉式列表（popup list）

图示：



在 DCL 中的名称	popup_list

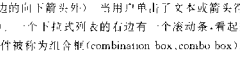
说明：

下拉式列表或简单的下拉功能与列表框类似。当一个对话框首次出现时，其下拉列表处于一个"折叠"状态，在屏幕上看起来很像一个按钮（除了右边的向下箭头外）。当用户单击了文本或箭头符号后，该列表会弹出并出现更多的选择（但这并非必要）。一个下拉式列表的右边有一个滚动条，看起来很像一个列表框的滚动条。在某些 GUI 中，类似的控件被称为组合框（combination box、combo box）或下拉式列表框（drop-down list box）。

当一个下拉式列表被折叠起来时，当前的选择会出现在它的显示控件中。下拉式列表不允许作多重选择。

名称	互锁按钮（Radio Button）

图示：



在 DCL 中的名称	radio_button

说明：

互锁按钮的功能类似汽车收音机上的按钮。用户一次只能选择一个，而且当按下一个按钮后，任何以前被选中的按钮都会被关闭。一个可选择的标号将出现在互锁按钮的右边（某些计算机中位于其上方）。当用户企图将一个互锁按钮放在互锁按钮框以外的地方时，DCB 功能会发出一个错误警告。

名称	滑动条（Slider）

图示：

在 DCL 中的名称	slider

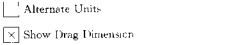
说明：

滑动条是用来取得一个数值的。用户可将滑动条的指示器向左或向右(向上或向下)拖动，以取得一个由应用程序决定的值。这个值被当作一个字符串返回，而字符串将包含特定范围内一个被作上记号的整数(该整数是16位的值，其范围为－32 768～32 767)。应用程序可根据需要来测试和增减这个值。在某些GUI中，类似的控件被称为dial。

在一些计算机上，额外的控制(例如加和减按钮)可使用户以固定的增量来移动滑动条

名称	乒乓开关(Toggle)

图示：



在DCL中的名称	toggle

说明：

乒乓开关控制一布尔值("0"或"1")。它以带选择性标对的小方格形式显示在对话框的右边(某些计算机中位于上方)。当用户单击它时，一个核对记号或X记号将出现(或消失)在对话框中。乒乓开关可使用户查看或改变打开/关闭选择的状态。

如上所述，控件可组成横向或纵向的复合形式，称之为簇(cluster)。一行或一列可视为一个单一的控件。可将控件组合到行或列中，以表示内部组织对话框，这是对话框的一种设计方法。

一行或一列可用边界线围起来，以便将可选标号内容包含在其中(一个没有框的簇将无法被标记)。

用户不能选取一个只包含该簇独立(可选择的，活动的)控件的簇。不能分配动作给控件簇，但纵向或横向互锁按钮例外。

用户可以在对话框中定义横向或纵向互锁按钮。由于以这种方式使用的控件簇包含了其他控件子集(children)，故称为子组合。当用户在对话框中引用一个子组合时，不能改变它的属性。在base.dcl文件中定义了许多标准的子组合。下面向用户介绍具有上述功能的对话框元件。

名称	列(Column)

图示：

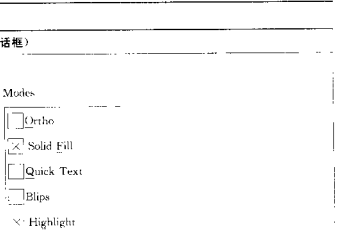
在DCL中的名称	column

说明：

一列中的控件将按出现在 DCL 文件中的顺序纵向排列。一列可以包含任何种类的控件（除了单个的互锁按钮外），也可以包含行和其他的列。

名称 | **加框列(Boxed Column)（纵向对话框）**

图示：



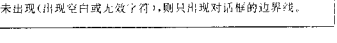
在 DCL 中的名称 | boxed_column

说明：

纵向对话框是一由边界线围起来的纵向排列的选项。一个对话框是由诸如纵向对话框这样的元件表现出来的。假如一个纵向对话框拥有一标号，则该标号可能出现在边界线之上或是嵌入其中，这视 GUI 如何显示对话框而定。如果该标号未出现（出现空白或无效字符），则只出现对话框的边界线。

名称 | **行(Row)**

图示：



在 DCL 中的名称 | row

说明：

一行好比一列，但它的控件是按出现在 DCL 文件中的顺序水平排列。

名称 | **对话框行(Boxed Row)（横向对话框）**

图示：

在 DCL 中的名称 | boxed_row

说明：

横向对话框是一由边界线围起来的行。如果一个横向对话框拥有一标号，则该标号可能出现在边界线之上或是嵌入其中，这视 GUI 如何显示对话框而定。如果该标号未出现（空白或无效字符），则只出现对话框边界线。

名称	互锁列(Radio Column)(纵向互锁按钮)

图示：

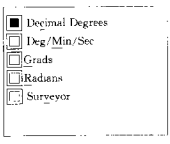
在 DCL 中的名称	radio_column

说明：

互锁列是一次只能选择一个互锁按钮控件的列。互锁列以一组固定的相互独立的选项出现在用户面前；这与不能相互独立的摇摆开关以及无法拥有一固定长度行列表始好相反。互锁列可被分配一个动作(action)。

名称	对话框互锁列(Boxed Radio Column)(纵向互锁对话框)

图示：



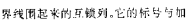
在 DCL 中的名称	boxed_radio_column

说明：

对话框互锁列是一由边界线围起来的互锁列。它的标号与加框列相同。

名称	互锁行(Radio Row)(横向互锁按钮)

图示：



在 DCL 中的名称	radio_row

说明： 互锁行像互锁列一样，是一次只能选择一个的互锁按钮控件的行。互锁行可被分配一个动作。

建议： 互锁行（横向互锁对话框）比互锁列（纵向互锁对话框）难用，且不引人注目，所以用户仅在选项数目较小（2～4个）或其标号较短时才使用它们。

名称 对话框互锁行（Boxed Radio Row）（横向互锁对话框）

图示：

在DCL中的名称 boxed_radio_row

说明：

对话框互锁行是一由虚界线围起来的互锁行，它的标号为对话框的标号。

下面说明的预定义控件不会导致任何动作且不能被选择。它们只是用来提供显示信息或强调视觉效果，或用来支持对话框设计。这些装饰性的和信息性的控件如下：

名称 图象（Image）

图示：

在DCL中的名称 image

说明：

图象是一个可以在其中显示向量图案的矩形，例如，图象可用来显示AutoCAD中的图形、线型、文本以及彩色光栅等。

名称 文本（Text）

图示：

Directories: Files:

在DCL中的名称 text

说明：

文本控件用来显示一个文本字符串以表示控件或信息。例如，大部分惯用的警告对话框（alert box）仅包含一些文本或一个OK按钮（在某些计算机上，AutoCAD警告对话框是由主机系统显示的，而且并不是AutoCAD对话框）。

大部分控件除了能作为控件显示以外都有自己的标号属性，故用户并不需要一直使用文本控件。但最好有一个经常保持空白的文本控件，以便于显示用户动作、错误信息或是警告（特别是在不需要用警告对话框的情况下）。

名称	衬垫（Spacer）
在 DCL 中的名称	spacer

说明：

衬垫是一个不显示任何内容的空白控件，它能调整控件的大小和设计，PDB 功能可自动处理衬垫问题，但要确认它与其他对话框的协调性，用户仅在特定情况下使用衬垫控件。

受限控件及其属性

用户的 DCL 文件不应该使用表1-2-1所示的供内部使用的控件类型或控件属性。

表1-2-1　被限制的控件类型及属性

被限制使用的类型	被限制使用的属性
cluster（簇）	horizontal_margin
tile（托件）	vertical_margin

注意： 一般来说，除非用户正在重新定义标准的退出按钮的子组合，否则不应使用基本的退出按钮类型（cancel_button，help_button 与 ok_button）。

第三章　对话框的属性

3.1　控 件 属 性

一个控件的属性可以定义它的设计和功能。一个属性就像一个程序语言变量，它包括了一个名称和一个值，且值必须为下述特定类型之一：

3.1.1　整数

代表距离的数值(整数和实数)。例如：一控件的宽度和高度都是以字符宽度或字符高度单位来表示的。

3.1.2　实数

一个实数必须以数字开头。例如：0.1 是正确的，而.1 是不正确的。

3.1.3　由引号括起来的字符串

一个将文本括在双引号("")内的字符串。如果该字符串必须包含一双引号，则应在双引号字符之前加上反斜杠"\"。字符串亦可包含其他转义序列，表 1-3-1 列出了可被 DCL 接受的反义字符(Escape character)。

表 1-3-1　在 DCL 字符串中所允许的转义序列

转义序列	特性
\"	引号
\\	反斜杠
\n	换行
\t	水平制表

3.1.4　保留字

保留字是由字母组成的标识符。例如，许多属性要求一个或者是真(true)或者是假(false)的值。保留字对大小写是敏感的，True 并不等于 true。

属性名称也是对大小写敏感的。要调用 Width 就不能键入 width。

要点：应用程序总是将属性恢复为字符串。如果用户的应用程序使用的是数值，则必须根据实际情况将字符串值转换为数值。从应用程序的观点来看，一个保留字和一个字符串之间的唯一区别是：保留字总是以字母开头，而字符串是可以数字或特殊字符开头并且可包含空白。

有些属性，例如宽度和高度，对所有控件来说都是命令，可以通过选择来指定属性；许多

属性在未定义时都有其缺省值，而另一些属性对某些控件来说具有特定的含义。例如，一个图象的背景颜色。如果用户企图将此属性用到一个不同种类的控件上，AutoCAD 将会报告一个错误发生，或甚至忽略该属性。

3.2 用户自定义的属性

用户可以使用自己定义的属性。属性名称可以是任何不与预定义的标准属性(参见下面的章节)相冲突的有效名称。一个属性名称就像一个关键字一般，可以包含字母、数字或下划线，而且必须以字母开头。

警告：如果用户企图定义一个与预定义属性相冲突的属性名称，则 PDB 功能不会将该属性当作一个新的属性，并且将使用用户用标准属性分配的值。这将导致一个很奇怪的对话框，并会出现一个和干的错误信息，使用户难以排错。

用户在属性中所分配的值以及它们的含义都是由用户的应用程序定义的。用户所定义的属性值必须符合上一节中所描述的类型。(get attr)、ads get attr()以及 ads get attr string()函数会像它们预定义的属性一样恢复用户定义的属性。用户定义的属性的使用及其含义完全由用户的应用程序决定，定义用户自己的属性与定义特定应用程序的属性类似。这两种方法都可使 PDB 功能能管理用户所提供的数据。用户定义的数据都是只读的，当对话框正在作用时它们是静止的。如果用户在运行时必须动态改变该值，则必须使用客户数据(client data)。用户可以检查自己在应用程序的 DCL 文件中定义的属性值，但客户数据对用户来说是不可见的。

AutoCAD 支持对话框本身所定义的属性，errmsg 对每一控件都拥有唯一的字符串。一个普通的错误处理程序在它显示警告时将使用 errmsg 的值。例如，假设控件将下列的值赋给 errmsg：

```
errmsg = "Grid Y Spacing";
```

如果用户输入一个无效值(例如一个负数)，则 AutoCAD 将显示下列错误信息：

```
Invalid Grid Spacing.
```

上述 Invalid 这个字以及随后的句点是由错误处理程序提供的。

用户可在 acad.dcl 文件中找到可支持对话框的 DCL。其他用户定义属性的情况还有：在一个控件上限制其值以及可使其活动的控件的子对话框的名称。

3.3 预定义属性

本节叙述由 PDB 定义的属性，首先给出任何类型的控件属性，然后说明那些使用特殊类型控件属性的含义。表 1-3-2 是按字母顺序的预定义属性，在稍后的章节中将讨论到有关属性的细节。

表 1-3-2 预定义属性一览

属性名称	相关事项	含义(被指定时为 true)
action	所有活动的控件	AutoLISP 动作程序
alignment	全部的项	在一个组上的水平或垂直位置
allow_accept	编辑框、图象按钮以及列表框	当选择此控件时,is_default 按钮即变为活动
aspect_ratio	图象、图象按钮	一个图象的外观比例
big_increment	滑动条	要移动的距离增量(计算机本身必须支持增量控制)
children_alignment	行、列、互锁行、互锁列、加框行、加框列、加框互锁行、加框互锁列	一个排列成行的子组合
children_fixed_height	行、列、互锁行、互锁列、加框行、加框列、加框互锁行、加框互锁列	在配置过程中,一个子组合的高度不会增加
children_fixed_width	行、列、互锁行、互锁列、加框行、加框列、加框互锁行、加框互锁列	在配置过程中,一个子组合的宽度是固定的
color	图象、图象按钮	一个图象的背景(填充)颜色
edit_limit	编辑框	用户可以输入的最大字符数
edit_width	编辑框、自动退出列表	控件编辑(输入)部分的宽度
fixed_height	全部的项	分区增长时高度固定
fixed_width	全部的项	分区增长时宽度固定
height	全部的项	字段高度
initial_focus	对话(dialog)	初始化光标在字段上的位置
is_bold	文本	以黑体字显示(计算机本身必须支持黑体字)
is_cancel	按钮	当删除键(通常是 Ctrl+C)被按下时,按钮将发挥作用(计算机本身必须支持热键)
is_default	按钮	当确认键(通常是 Enter 键)被按下时,按钮将发挥作用(计算机本身必须支持热键)
is_enable	所有活动的字段	控件初始化时是有效的
is_tab_stop	所有活动的字段	控件是通制表键停止(计算机必须支持热键)
key	所有活动的字段	应用程序所使用的控件名称

（续）

属性名称	相关事项	含义（被指定时为 true）
label	加框行、加框列、加框互锁行、加框互锁列、按钮、对话框、编辑框、列表框、自动退出列表、互锁按钮、文本以及乒乓开关	显示控件的标签
layout	滑动条	滑动条是水平的或是垂直的
list	列表框、自动退出列表	在列表中所显示的初始值
max_value	滑动条	一个滑动条的最大值
min_value	滑动条	一个滑动条的最小值
mnemonic	所有活动的控件	控件的内存字符（计算机本身必须支持热键）
multiple_select	列表框	允许选取多个项的列表框
small_increment	滑动条	要移走的距离增量（计算机本身必须支持增量控制）
tabs	列表框、自动退出列表	现表项是显示的新表键停止
value	文本、活动的控件（除了按钮及图象按钮以外）	控件的初始值
width	全部的项	控件的宽度

以下对 key 和 value 属性做一些说明。

key 和 value 是两个最主要的属性。key 是应用程序在引用控件时所使用的名称。每个有效的控件都必须有一个对对话框唯一的 key 值。分配给控件的值在运行时可以改变，value 属性可指定控件的初始值。

key 是一个包含在引号内的字符串（不是缺省值）。用户指定一个程序引用此特定控件的 ASCII 码。在一特定的对话框中，每个 key 值都必须是唯一的。

既然 key 值对用户来说是不可见的，所以它的名称就可以由程序设计者任选（只要它对对话框是唯一的）。同理，key 属性用在其他语言编写的应用程序中时不必转换。

value 一个包含在引号内的字符串（非缺省值），是一控件的初始值。一个控件的值随控件种类而变。3.3 节中已说明了每个预定义控件类型如何使用它的值。通过用户输入或（set_tile）调用可在运行时改变一个控件的值。

设计对话框（先前调用（new_dialog）函数创建的）时并未考虑到一控件的 value 属性。设计完成后且已显示对话框时，（new_dialog）将使用 value 属性，以便将位于对话框中的每个控件初始化（就好像使用 set_tile 调用一样）。这意味一个控件的 value 属性对对话框中的控件大小及空间没有影响。