

第8章 变量和函数

看过下面的内容，Authorware的学习便开始进入新的天地，也可以说在多媒体编辑上与前面大相径庭，因为在这一章就要学习和使用变量和函数了。Authorware除了能够提供大量的系统函数和变量外，还允许使用自定义的变量和函数，这无疑又为Authorware多媒体创作开拓了广阔的空间。

本章主要内容：

- 变量基本概念。
- 变量的使用。
- 函数基本概念。
- 函数的使用。
- 关于变量和函数的补充知识。

8.1 变量简介

变量的加入，使Authorware的编程更加灵活多变，在本章的最后还对Authorware编程的特点及语法规则做了一些说明变量和语法，在前面列举实例的过程中，曾不时的提到过变量的使用，其中包括系统变量和自定义变量，当时只是简单介绍，在本节将系统地讲解Authorware变量的使用规范及应用技巧。

8.1.1 变量的类型

在Authorware中，变量可以分为系统变量、自定义变量和逻辑型变量三种，而且变量的使用有着特殊的约定，下面就其中的几个要点进行介绍。

首先，Authorware为用户提供了一套系统变量，这些变量各司其职，分别能完成判断、交互、框架、图形、视频播放以及网络等方面的任务。在变量的使用中，有时可以直接将变量放在文本框中使用，有时也可以以表达式的方式来使用，如经常使用的格式“Position:=CursorX”，在这里面的“:=”是作为变量的赋值运算符，这个等式表示将窗口中的光标的横坐标值赋给变量CursorX，而CursorX又将值赋给变量Position。对于某些特殊的变量，在使用中后面要有一个“@”字符再加上一个图标标题，如Animating@"IconTitle"的格式，这种变量称为引用变量，它表示如果IconTitle图标正在移动，此变量的值为真。使用引用变量可以调用流程线中的任何一个图标中的相关信息，例如，在前面使用过GoTo(IconID@"IconTitle")函数，它就是使程序流程返回IconTitle图标，再继续执行。

其次，自定义变量是用户根据程序的需要自己定义的一种变量，变量一般可以由字母来构成，但一定不要与系统变量相冲突。自定义变量最突出的作用是它可以用来存储数值或字符串。数值的赋值表示很简单，在前面已经多次提到，在字符串的赋值方面你一定要引起注意，因为在字符串的两侧要使用双引号" "。另外，在字符串的使用中还可以插入连接符号“^”，当字符串太长时，可以将它们分开来表示，如StringVar := "userName := "^Beijing

(34)^(John^Char(34)^)", 这样表示会使用户更容易阅读和理解其中的含义。如果要在一个字符串中使用双引号, 必须在双引号前加上反斜杠, 如以下形式: `stringVar := "userName := \"John\" "`。在一个字符串中使用反斜杠时, 必须在反斜杠前再加上一个反斜杠, 如以下形式: `"Name\\Class\\College\\Station "`。

再次, 逻辑型变量通常用于存储 True(真)或False(假)两种值。它们通常在某些判断语句或一些条件文本框中使用。

8.1.2 变量的应用场合

Authorware的变量主要应用在以下三种场合, 它们分别是:

1. 在对话框的文本框中使用变量。

在设置对话框中的某一选项时, 常常会使用到条件限制, 如图 8-1所示, 用户可以在 Properties: Sound Icon(声音图标属性)对话框中输入声音播放的条件, 如 VideoDone, 当前面的视频动画播放结束后, 该“声音”图标存储的声音文件就开始播放。同样的条件文本框还有 Properties: Movie Icon(数字电影图标属性)对话框中的 Until True 文本框, Properties: Video Icon(视频图标属性)对话框中的 Stop If 文本框, 交互响应属性对话框中的 Active If 文本框等, 都可以在其中使用变量和条件表达式。

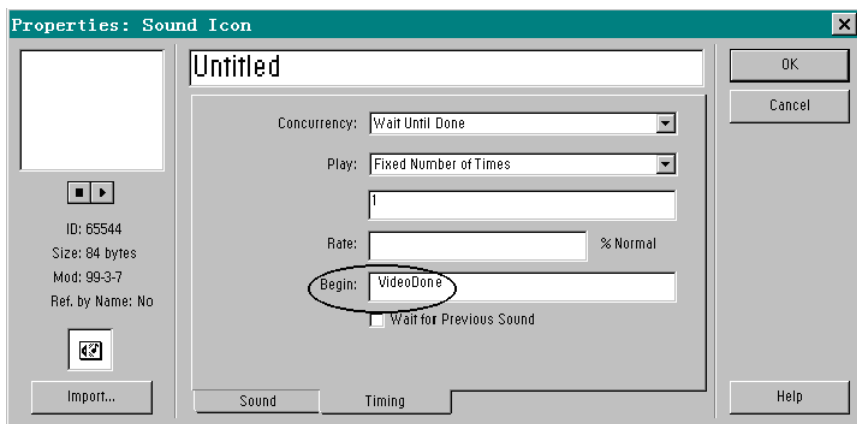


图8-1 设置条件变量

2. 在“计算”图标文本框中使用变量。

变量最常用的场合是在“计算”图标文本框中, 在其中不但可以输入简单的限制条件, 而且还可以输入大量的语句。如图 8-2所示, 在文本框中使用了循环语句, 循环条件是 $J \leq 10$, 若变量 J 和 SUM 的初值都为 0, 那么程序运行完该图标后, 变量 J 的值变为 11, 而变量 SUM 的值变为 55。

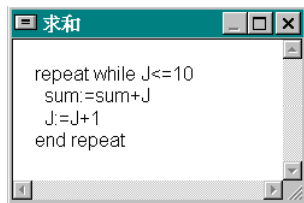


图8-2 计算图标文本框

3. 在“显示”图标或“交互”图标中使用变量。

在“显示”图标或“交互”图标中不但可以绘制图形和输入文字, 而且还能够进行变量计算。如图 8-3所示, 在“显示”图标的演示窗口直接输入系统变量 Time, 在程序运行时, 该变量将自动显示对应的数值。当程序运行后, 图中的 Time 就会变为当前的系统时间 18:06。另

外，如果单击图解工具箱的“移动”按钮，然后再选择该变量，变量也会自动运行出结果。

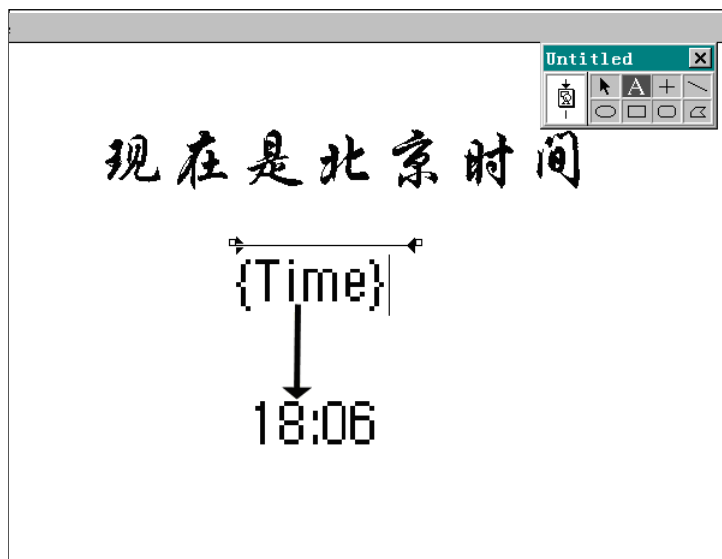



图8-3 在显示图标中使用变量

 **注意** 在演示窗口中输入的变量必须放在一对大括号内，这样Authorware才能将里面的单词认为是变量，当然，也可以在窗口中使用自定义变量。

8.2 变量的使用

在本节中，将开始系统地学习变量的应用。下面首先介绍“计算”图标的使用及其属性对话框的设置，然后，再讲解变量窗口的使用技巧。在最后，还为用户提供了大量的实例，它们都很简单、易懂，希望在学习的时候能够掌握一般变量的使用技巧，另外，也可以结合Authorware的帮助信息来尝试使用其他变量。

8.2.1 计算图标的使用

在“计算”图标中可以使用表达式来改变变量的值或者在程序中执行一次运算，用户可以在流程线的任意部位使用变量，也可以将它们连接到别的图标上。如图8-4所示，用户可以直接在文本框中输入表达式。单击窗口标题栏上的“关闭”按钮，可以关闭该窗口，Authorware会弹出提示对话框，要求用户保存修改。如果不想保存修改，可以按下Esc键关闭窗口。

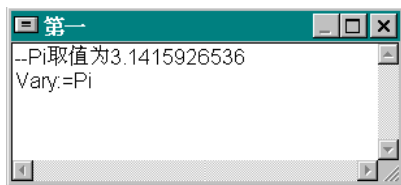



图8-4 计算图标文本框

 **提示** 也可以在“计算”图标的文本框中输入注释说明。如图8-4所示，可以在说明语句的前面加上两个字符“--”。注释符号后面的部分在程序执行的过程中不产生结果。说明语句可以在程序的任意部分，它的使用不会影响程序的运行。

8.2.2 系统变量的使用

打开Authorware的Window菜单，然后单击菜单中的 Variables命令，可以调出 Authorware 的Variables (变量)窗口，如图8-5所示。用户也可以直接单击工具栏上的“变量”按钮来调出变量窗口。在窗口的上方设有 Category(种类)、Interaction(交互)、Initial Value(初始值)、Current Value(当前值)四个标题。当Category列表框中为All选项时，在下面的变量列表框中将显示 Authorware的全部系统变量。选择列表框中的一个变量，如 AllCorrectMatched，在Category的后面便会出现该变量的种类，如 Interaction，在下面的Description文本框中将显示该变量的描述，用户可以直接参照里面的变量说明和示例来使用变量。

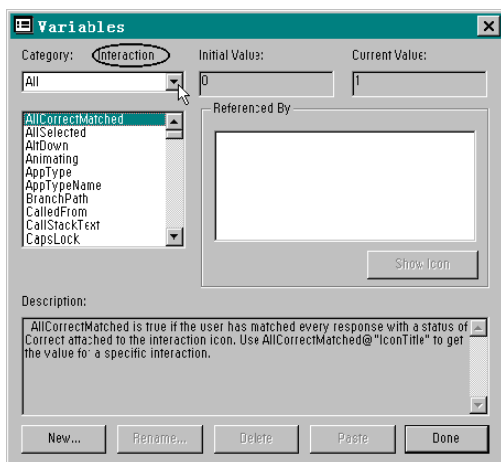


图8-5 变量窗口

打开Category下拉列表框，如图8-6所示，在列表框中显示 Authorware 5的十一种变量类型，它们分别是CMI(管理教学)、Decision(判断)、File(文件)、Framework(框架)、General(常用)、Graphics(图形)、Icons(图标)、Interaction(交互)、Network(网络)、Time(时间)和Video(视频)。另外，当用户打开一个 Authorware程序时，在列表框中也会显示该文件的标题，如图8-6中所示的Toolpal.a5p文件。

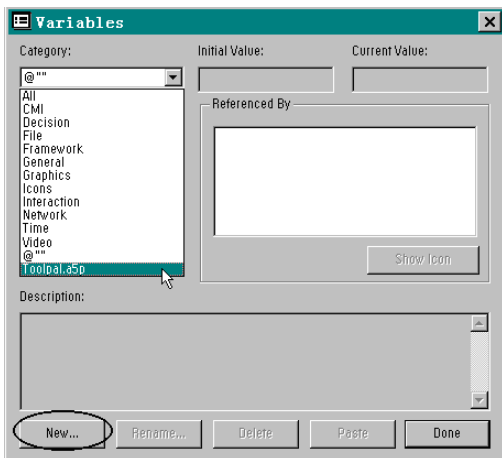


图8-6 选择变量类型

在系统变量的使用上，Authorware还提供了一种更为快捷的操作方式，这种方式会省却用户麻烦的键盘输入过程。如图 8-7 所示，打开“计算”图标文本框，在变量列表框中选择某一变量，如图中所示的 CMILoggedOut 变量，然后单击窗口中的 Paste 按钮，该变量就会被粘贴在“计算”图标窗口中。使用此操作固然简单，但不是所有的变量都可以采取这种方式，只有当 Paste 按钮变黑时才表示该变量可以进行粘贴操作。使用该操作的场合还可以在对话框或“显示”图标中进行。

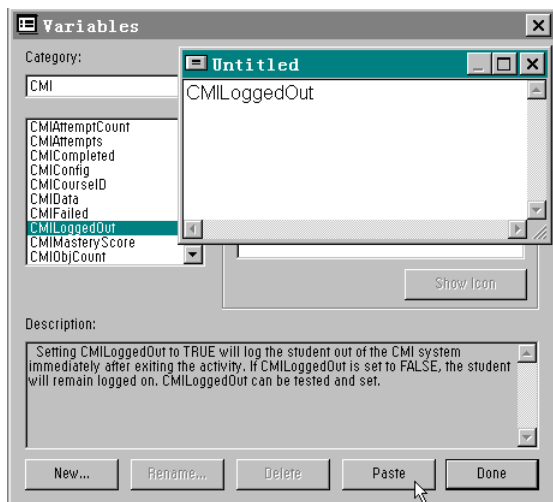


图8-7 粘贴变量

8.2.3 自定义变量的定义与赋值

对某一自定义的变量进行赋值时，可以单击变量窗口中的 New 按钮，随后将弹出如图 8-8 所示的 New Variable 对话框，在此可以对变量进行描述。

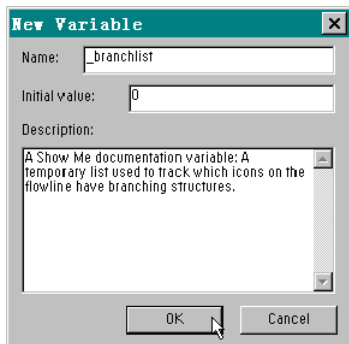


图8-8 变量赋初值

在 Name 文本框中可以输入自定义变量的名称，在其下面的 Initial Value 文本框可以输入变量的初始值，如 0。另外，为了方便以后的程序阅读和检查，还可以在 Description 文本框中输入变量的具体描述，描述完毕，单击 OK 按钮。

如果在Category列表框中选择程序文件时,如图8-9所示的Toolpal.a5p,在下面的变量列表框中将显示该程序中使用的所有变量,若用鼠标选择其中的一个变量,便可查看有关它的信息。如在Initial Value文本框中会显示该变量的初始值,在Current Value文本框中显示该变量的当前值,在此状态下用户也可以更改数值。另外,在Referenced By(参考)列表框中还会列出程序中使用该变量的图标名称。在图8-9中,选择列表框中的图标名称,然后再单击Show Icon(显示图标)按钮,在流程线上对应的图标就会显示出来,并且图标呈黑色,处于选中状态。

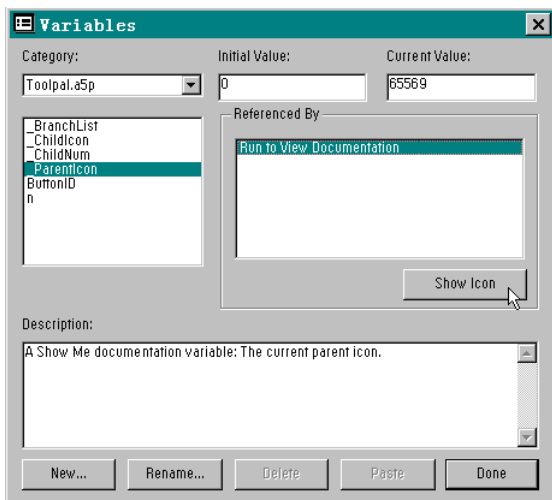



图8-9 观察变量

 **提示** 如果想将变量更改名称,可以单击窗口下面的 Rename按钮,然后在弹出的 Rename Variable文本框中输入新的变量名称即可。

8.2.4 计算图标与其他图标的连接

在大多数情况下,有时会考虑将“计算”图标放置在另一个图标的前面或者直接与该图标进行连接,比如,将“计算”图标与一个“判断”图标或“交互”图标相连接,如果该分支方式设为Try Again, Authorware将会重复执行该“计算”图标,如果将“计算”图标放置在它们的前面,则 Authorware仅能执行一次。

连接“计算”图标的过程可以这样进行:首先在流程线上选择某一图标,然后打开 Modify菜单,在Icon的子菜单中单击 Calculation命令就会弹出“计算”图标窗口,如图8-10所示,该窗口的标题与用户选择的图标标题一样,同样可以在“计算”图标文本框中输入各种语法、语句。在程序运行时, Authorware将先运行“计算”图标里的内容,然后再执行该图标。

除了“计算”图标本身外,所有的图标都可以使用“计算”图标进行连接。进行“计算”图标连接后,在图标的左上方会出现一个“=”符号,如图中的 Palette图标或Run to View Documentation图标所示。要取消计算图标连接,可以打开计算图标窗口,然后将其中的所有内容删除,图标连接将自动取消。

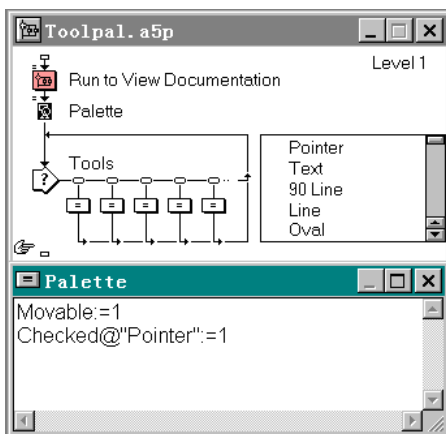


图8-10 连接“计算”图标

下面来提一下“计算”图标的属性对话框，该对话框的设置最为简单。选择“计算”图标，然后按Ctrl+I快捷键，就会打开该图标的属性对话框，如图8-11所示，打开了标题为Text的Properties: Calculation Icon(计算图标属性)对话框，在对话框的Function列表框中“显示”图标所使用的函数名称，在Variables列表框中显示所有的变量名称，如果使用鼠标选择其中的一个变量，在下面的Current Value框中就会显示该变量的当前值。

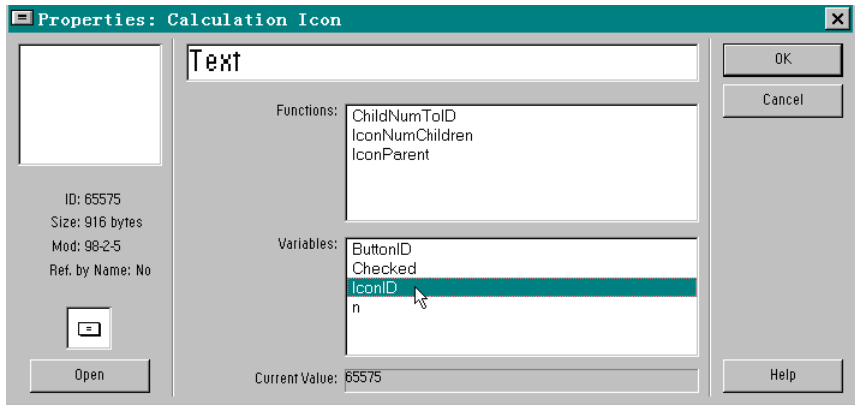


图8-11 计算图标属性对话框

8.2.5 变量在使用中常会遇到的问题

最后，再讲一下变量在使用中可能遇到的问题。通常，在变量的使用过程中系统变量不能进行赋值，也就是说在语句左边不可能是系统变量。

在自定义变量的命名中，变量的名称不能与系统变量或函数同名，但可以与图标的标题重名，但为了使程序便于阅读，一般情况下最好不要使用同一名称。

当程序中出现重名变量时，Authorware会弹出Variable Name Conflicts(变量名冲突)对话框，如图8-12所示。在左面的冲突变量列表框中显示冲突的变量名称，选择冲突的变量后单击Rename按钮，就可以在右面的文本框中输入新的名称了。当从别的程序中拷贝图标或流程

线时，如果两个程序中使用了同一变量，此时最容易出现变量冲突。

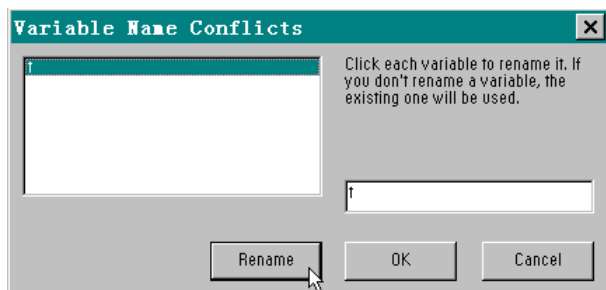


图8-12 变量冲突对话框

8.3 常用变量与应用实例

以上是变量的使用规范和应注意的一些问题，有了上面的基础，就可以尝试着使用某些简单的变量了。Authorware的系统变量有两百多个，逐个进行介绍显然不可能，下面列举几个使用变量的例子，每个变量都代表一种类型，通过这几个典型例子的介绍希望用户能掌握使用一般变量的技巧，如果觉得有了一定的收获，不妨参考Authorware帮助文件中的变量说明，逐个练习一下。

8.3.1 PathSelected变量示例，Decision类型

在前面曾经举过一个纸牌游戏的例子，程序开始运行后，弹出的纸牌是随机性的，当在屏幕上单击一下时，运行的画面就会停止。现在要知道最后弹出的是哪一张，如果使用前几章讲的知识来完成几乎是不可能的，而使用 Authorware提供的系统变量则轻而易举。下面就使用Decision 类变量中的PathSelected来实现此功能。

在讲解PathSelected变量时，顺便提一下连接“计算”图标的使用技巧。如图 8-13黑圈内所示，在流程线的最下方放置一个“显示”图标，将图标命名为“显示最后一张”，该图标的主要功能是显示最后结果，然后开始连接“计算”图标。首先打开 Modify菜单，单击Icon子菜单下的Calculation命令，在弹出的“计算”图标文本框中输入如图 8-15所示的语句，然后关闭该窗口，此时“显示”图标的左上角就会出现一个“=”符号。

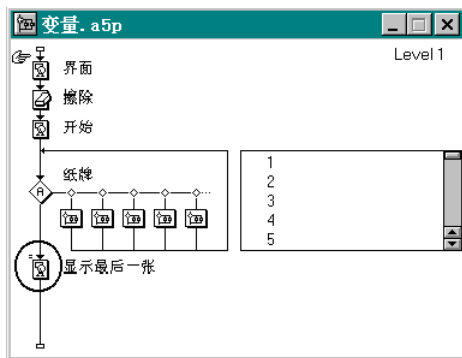


图8-13 连接计算图标

打开“显示”图标，单击图解工具箱中的“文本”按钮，在演示窗口中输入如图 8-14所示的文字。一定要注意，在变量 t 的两侧要加上一对大括号“{}”，当 Authorware 在运行该图标的内容时将把 t 认为是变量，如果 t 代表一定的数值，Authorware 将会把它显示出来。在关闭该窗口时，系统会弹出一个新变量窗口，提示用户将 t 变量赋初值，暂且输入 0。

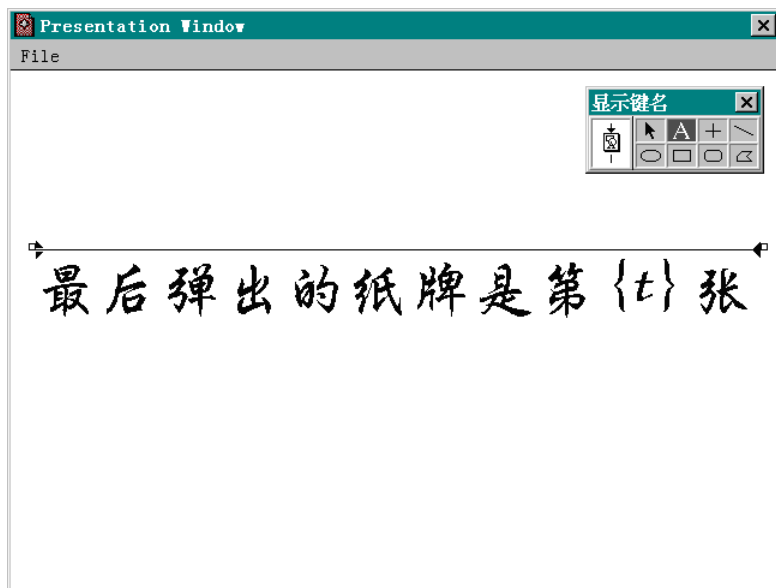


图8-14 输入变量

在“计算”图标的文本框中输入“ $t:=\text{PathSelected@}$ “纸牌””语句，如图8-15所示。该语句中的“ PathSelected@ “纸牌””的功能是计算“判断”图标最后一次执行的路径编号。在这里使用了变量 t 来接收此编号，然后在上图的“显示”图标中进行显示。

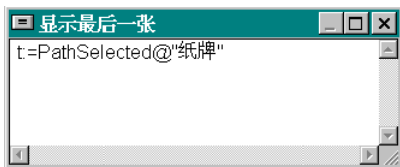


图8-15 运用变量

顺便提一下，由于表达式的右面代表一个数值，因此，也可以进行算术的四则运算，例如表达式可以写成

“ $t:=\text{PathSelected@}$ “纸牌”+1”，如果“ PathSelected@ “纸牌””的值是 6，那么程序运行的最后结果显示是 7。

另外当该变量单独使用时，它存储的是附属于一个“判断”图标所有路径中，用户最后一次所选路径的编号。

8.3.2 Key变量示例，General类型

下面的这个例子比较简单，只需要两步就能说明问题，在程序中的“计算”图标中使用 Key 变量、General 类型的变量一般都比较顺手，没有什么过多的设置。

在这个例子中，要达到如下的效果：当程序运行时，如果按下键盘上的某一键，屏幕上就会显示该键的名称。根据题目的要求，可以创建如图 8-16 所示的流程线，在程序的开始放置一个“等待”图标，在等待图标属性对话框中选择 Key Press 复选框。在程序的最后，要设


置跳转功能，此功能是为了能不断地进行键盘输入识别。GoTo(IconID@"按键")函数的功能在前面已经多次提到，它会使程序从后面跳到“按键”图标处继续执行。



图8-16 设置程序跳转

Key变量常用于存储用户最后一次按键的键名，键名包括 a、X等字母键，Enter、+、Esc、F2、F4、Home、TAB等功能键和UpArrow等方向键。

另外，在此类变量中还有一个与它功能相似的 KeyNum变量，不过该变量只能用于存储用户最后一次按键的数字键。

 **注意** Key变量对F1键不能进行识别，因为F1通常被设为提示帮助的快捷键。

在流程线上打开“显示键名”“显示”图标，在演示图标的文本框中输入如下文字，如图8-17所示。在Key变量的两侧同样要加上一对双括号“{}”。当程序运行时，按下键盘上的删除键后，屏幕上就会出现 Backspace字样，如图8-17中的箭头所指。

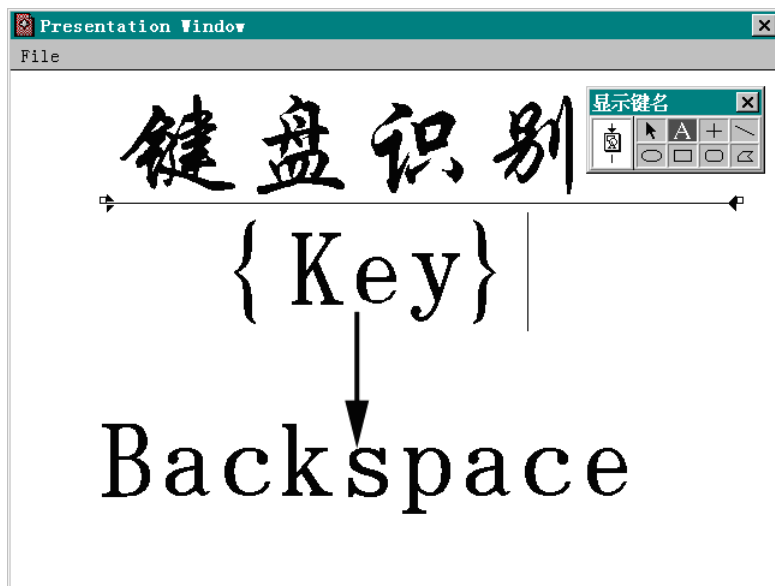



图8-17 键盘识别

当不断按下键盘时，窗口中显示的键名也不断更改，相信此时你也会明白程序中“等待”图标和“计算”图标的功能了。

 提示 1) “等待”图标有两个作用，首先它能使画面显示停止，让你看清楚显示的键名。其次，设置了Key Press 复选框，按下键盘后程序就会继续往下进行。2) 在程序中“计算”图标设置了跳转功能，使你可以不断地进行键盘输入。

8.3.3 MediaPlayer变量示例，General类型

在前面曾讲过一个做纪念光盘的例子，后来，又给这个例子添加了伴随声音，现在想给程序添加更为丰富的内容，比如说，想在每一首歌曲播放时显示各自相关的画面信息，用传统的方法是在每一个声音图标的下面放置“显示”图标，这种方法对一般的问题可能容易解决，但如果程序使用的是调用“声音”图标进行播放，显然就不会往下执行“显示”图标。因此在下面的例子中提供一种更为灵活的方法。

如图8-18所示，在“第一部分”“交互”图标的流程线上插入一个“计算”图标，在程序运行时，用它来调用相关的画面信息。在主流线的最后，放置一个“群组”图标，在里面放置各个“声音”图标要显示的画面信息，分别命名为“第二首歌画面”、“第三首歌画面”和“第四首歌画面”。

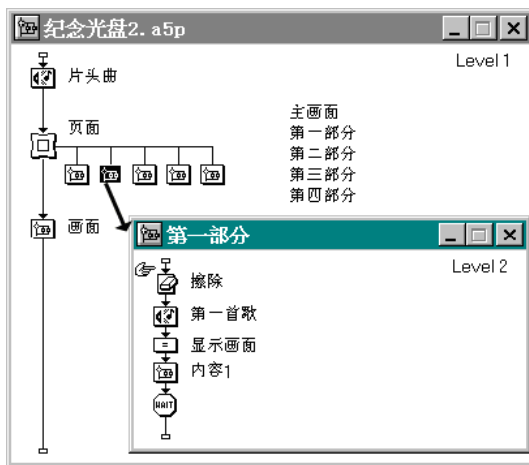


图8-18 设置流程线

打开“显示画面”计算图标，如图8-19所示，在文本框中输入一个条件语句，该语句的作用是当“第一首歌”声音文件正在播放时，就会显示“第一首歌画面”图标里的内容。根据这个道理，可以使用此方法来作出程序的其余部分。在这里，提示一下，首先，可以在其他的“群组”图标内放置与图8-19同样的“计算”图标，然后将每个图标内的歌名和画面的名称做相应的更改即可。



图8-19 设置画面



提示 当MediaPlayer变量单独使用时,如果当前的媒体正在播放,该变量存储的值为TRUE;当该变量作引用变量使用时,MediaPlayer @ "title"存储的是指定图标中的媒体文件是否播放的逻辑状态,如正在播放时,其值为TRUE。

当程序执行时,还可能遇到的另一个问题是画面重叠问题,当使用 DisplayIcon函数调用“显示”图标时,该画面将可能被覆盖在图 8-18中“内容1”画面的下方。因此,还要设置图标的层,如图8-20所示,在打开“第一首歌画面”显示图标,在 Layer文本框中输入图标的层为2,这样该图标中的内容将显示在下面图标画面的上方。

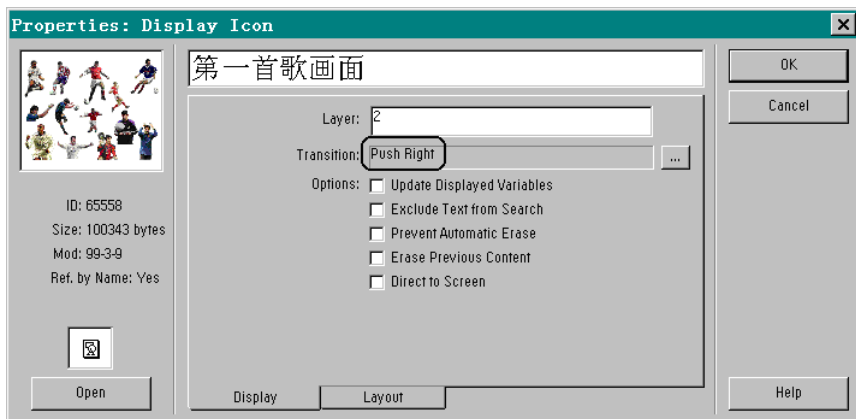



图8-20 设置过渡效果

另外,还可以设置一下画面的过渡效果,这样会使画面显示更为丰富。单击 Transition右边的  按钮,选择过渡类型为 Push Right,如图8-20的黑圈内所示。同样,也可以设置“内容1”图标中的图片过渡类型为 Push Left。

8.3.4 DayName、FullDate、FullTime变量示例,Time类型

在下面的例子中,将讲解三个变量,它们分别是 DayName、FullDate和FullTime,这三个变量能够调用系统的CMOS时间,在下面就使用它们来做一个怀表,如图 8-21所示。

当程序运行时,该怀表开始走时,就像使用的电子表一样。因此,使用上面的三个系统变量,分别要它们显示年月日、时间、星期。画面的背景是插入的图片,因此需要单击 Modify菜单下的Send To Back命令将它置后。

要完成图8-21所示的怀表功能,只要编辑很简单的程序就能达到要求。如图 8-22所示,在流程线的最上方放置一个“显示”图标,该图标有两个作用。首先,在该图标中插入图 8-21所示的怀表背景界面;然后,还要用它来连接“计算”图标,在“计算”图标的文本框中分别输入图8-22所示的变量和表达式,分别用自定义的变量 R来接收DayName时间,用O变量来接收FullDate年月日,最后用S变量来接收FullTime提供的星期时间。



技巧 连接完“计算”图标后,在“显示”图标的左上角显示一个“=”符号,如果你想更改图标里面的内容,只要双击这个“=”号就可以打开“计算”图标对话框了。

设置完时间变量后,还要在背景画面上显示时间。下面的工作对你可以说是太熟悉不过了,单击图解工具箱中的“文本”按钮,如图 8-23所示,在怀表的中心分别输入“{O}”、“{S}”。

和{R}”。当程序运行时，这三个变量就会把系统变量提供的时间显示出来了。

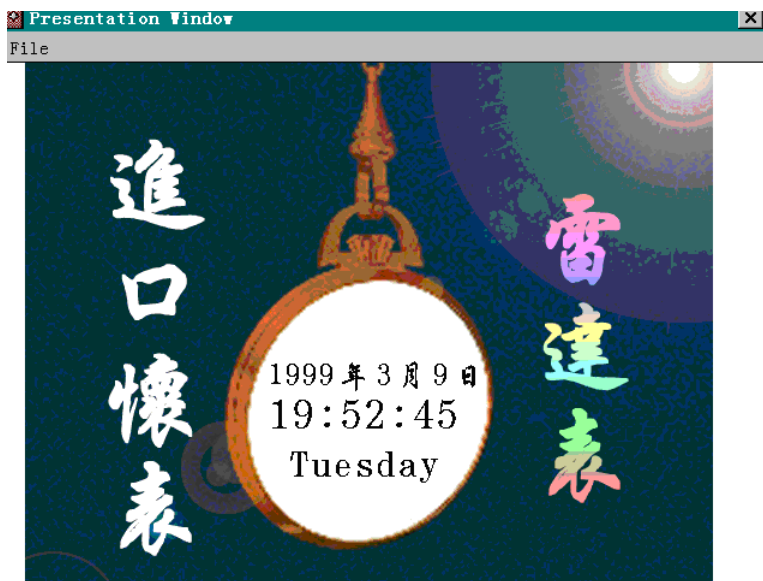


图8-21 怀表



图8-22 连接计算图标

注意 1) 输入变量后，还要双击图解工具箱中的“选择”按钮，然后在弹出的重叠模式对话框中选择透明模式。2) 在输入自定义变量的后，关闭窗口时，还要在弹出的新变量窗口中分别输入它们的初始值 0。

最后，再来谈一下流程线上的“返回”计算图标的作用。如图 8-24所示，在文本框中输入跳转函数的返回位置是“时钟”图标。如果没有下面的“返回”图标，程序只能将时间显示一次，然后就停止不动，只能每次单击“运行”按钮才能不断显示时间，这表当然也就不能称之为怀表了。设置了程序跳转，变量就能每次都得到更新，设计的怀表也就可以不断的走时了。

另外，在系统变量中还有一个变量 IconID，它在 Authorware 的编程中使用最为频繁，在前面已经多次使用过。该变量作引用变量使用时，IconID@"title"存储的是指定图标的唯一数字标识符，因此大多数的函数都要引用它，常常把它作为一个程序查找的标志来使用。同样，

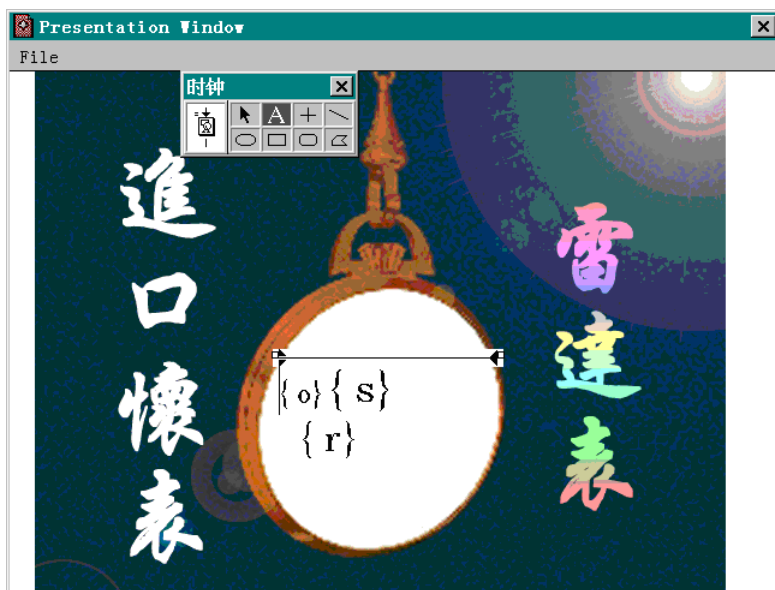


图8-23 输入变量

在Authorware的系统函数中也有一个 IconID 的函数，它的使用格式与此不同，常常将其写为 `number:=IconID("IconTitle")`，number 返回的是图标的 ID 数字号码，用户可以打开一个图标的属性对话框，在预览框的下面就能看到该图标的 ID 号码。

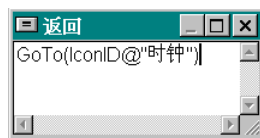


图8-24 设置返回图标

8.4 函数简介

函数通常指能够执行某种特殊任务的应用程序。Authorware的系统函数有300多个。例如，系统函数 `JumpPrintReturn("program", "document")` 可以打开由 Program 指定的应用程序，然后将指定的 document 文件在该应用程序中打印，打印完毕后继续演示。函数 `WriteExtFile` 的使用格式为 `result:=WriteExtFile("filename", String)`，该函数的作用是将字符串 String 写入由 filename 指定的文本文件中，如果写入成功，result 的返回值为 1，否则返回值为 0。

总之，Authorware 的函数功能有大有小，在编程中还可以将函数与变量结合使用，从而构成一种语法。在本章的最后，还要将 Authorware 的运算符、各种语法进行简要介绍。

Authorware 的函数包括两类，系统函数和自定义函数。系统函数是由 Authorware 本身自带的，自定义函数则主要包括 Xtras 和 DLL。Macromedia 公司为 Authorware 提供少量的自定义函数，如果想得到更多的函数，只能从别的开发者那里得到。

8.5 函数的使用

下面谈一下函数的使用问题，也就是函数窗口的具体使用。单击 Windows 菜单下的 Functions 命令，或直接单击工具栏上的“函数”按钮，都可以打开 Functions(函数)窗口。如图 8-25 所示，系统函数共分为 18 类：Character、File、CMI、Framework、General、Graphics、Icons、Jump、Math、OLE、Platform、Time、Video、Language、List、Network、Target、Xtras 等。

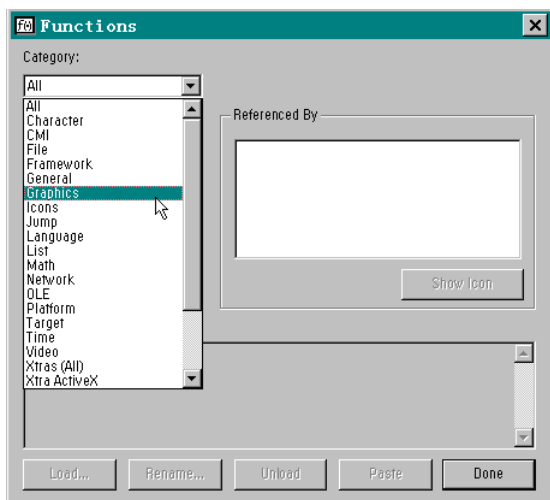


图8-25 函数窗口

在8.4节，提到过使用Paste按钮进行粘贴变量的操作，该操作可以大大减轻输入变量和函数的操作，在这里再给大家提供一种粘贴函数或变量的更为方便的办法。如图 8-26所示，首先，要在函数列表框中选择适当的函数，然后，在函数描述文本框中选择与函数相关的语法结构，单击鼠标右键，在快捷菜单中选择“复制”选项，最后，在函数输入的位置处再单击鼠标右键，选择“粘贴”选项，接下来只要将粘贴的函数进行相应的修改就可以使用了。

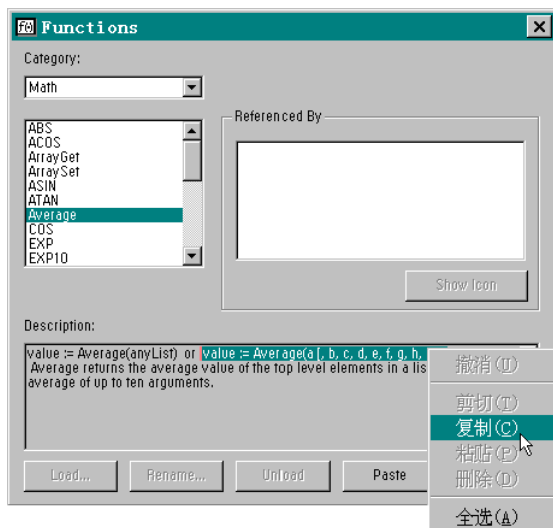


图8-26 粘贴函数

如果在Category下拉列表框中选择正在打开的程序文件，窗口中的 Load按钮就会被激活，此时单击该按钮就可以加载外部函数。

加载函数时，Authorware会弹出Load Function对话框，在列表框中选择适当的 UCD或DLL函数，然后单击“打开”按钮。如选择 A5dir函数，随后就会弹出如图 8-27所示的对话框。

在左面的函数列表框中选择函数名，然后单击 Load按钮，该函数就会被加载了。

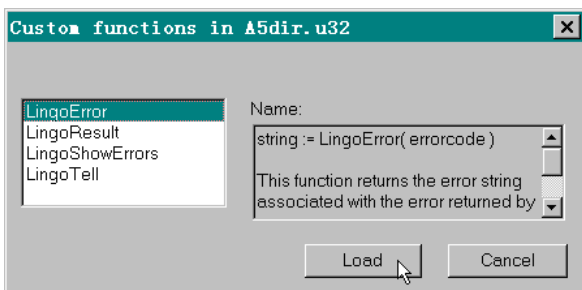


图8-27 加载UCD函数

加载后的函数同系统函数一样，用户可以先选择该函数，然后单击 Rename按钮来改名，单击 Unload按钮将其卸载。

创建新的函数与自定义函数不同，自定义函数需要很多的 Windows编程的知识和经验，涉及面也很广，所以不准备对创建自定义函数的细节做更多的介绍。下面给大家提供一点关于 DLL 和UCD的基本知识。

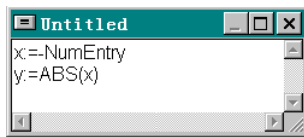


图8-28 使用函数

DLL，也就是 Dynamic Link Library 的缩写。对于 Windows来说，系统所需要的函数都是以动态链接的方式进行存储的，因此存储自定义函数的文件与当前的应用程序文件是分立的，当程序用到该函数，它会自动到指定的目录中去调用。在 Authorware软件下不能够进行 DLL 文件的编写。如果确实要创建 DLL文件，就必须熟悉 Windows的编程。而且，当 Authorware加载一个函数时，还要进行自定义函数文件的存储路径，以便在 Authorware调用函数时，能够知道函数的有关信息。

编写 DLL文件还有一个问题，那就是编写的 DLL常常不符合 Authorware的转换标准格式，这对不熟悉 Windows编程的用户来说更是雪上加霜。因此，为了帮助缺乏编程经验的用户使用 DLL文件，Authorware提供了对一个 DLL文件的透明方式(称为UCD)的加载。在图 8-27中所示的就是 UCD文件的加载。

另外，在“计算”图标窗口中也可直接使用函数，在窗口中的光标处直接进行输入。例如，如果想将输入的某一数值取绝对值，在使用绝对值函数之前首先要设置一个变量来存储这个绝对值，然后再将这个变量进行显示，如图 8-28所示。

8.6 常用函数与应用实例

在讲解函数的过程中，同样给你列举几个典型的实例，通过这几个例子的编辑过程向你讲述函数的使用方法。

在进入示例介绍之前，先来熟悉一下函数使用的格式。

1) RenameFile("filename", "newfilename")，这个函数的功能是更改文件名。在这个函数里有两个参数 filename和newfilename，编程语言中称它们为形参，在使用时，可以分别将这两个参数进行赋值，程序运行后就会分别将值代入对应的形参，也就实现了文件名。

2) Average(a, b, c, d, e, f, g, h, i, j)，这个函数的功能是求取数组的平均值，函数的形参有多个，用户可以直接在括号内输入对应的数值，如 numList := Average(1, 2, 3, 99)，也可以将

数组赋予一个数组变量，如，`numList := [1, 2, 3, 99]`，然后再将数组变量进行赋值，如`Value:= Average(numList)`，它们的效果是一样的。

3) `JumpOutReturn("program" [, "document"] [, "creator type"])`，函数的功能是使用一种应用程序来打开指定的文件。这个函数的参数格式比较特殊，`program`是应用程序名，如Winword, Notepad, Mspaint等；`document`是需要打开的文件名和路径；`creator type`是文件类型。如`JumpOutReturn("Winword ", "c:\readme.txt")`，该函数的功能是用Word打开readme文件。

上面谈了关于函数使用中的一些约定，如果已经大致了解了这些内容，下面就可以跟着进入具体函数的应用环境了。

8.6.1 Box() 函数示例，Graphics类型

在下面的例子中，使用`Box()`函数来绘制矩形。当程序开始执行时，首先在画面上显示图8-29中箭头所指的提示信息。用户可以在画面上的光标闪烁处输入矩形的宽度、长度和边框宽度值。当按下回车时，画面上又会提示使用鼠标在屏幕上单击，单击一下后，屏幕上就显示出你想要的矩形。

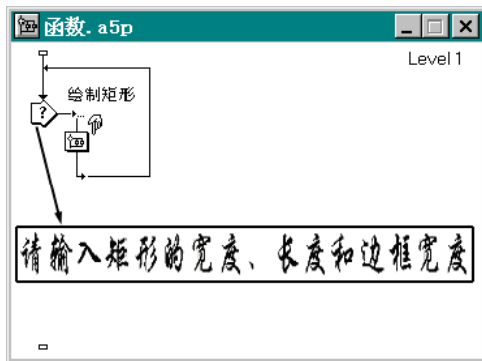


图8-29 流程图

由于程序中要求数值输入，因此要使用文本输入响应，同时还需将分支类型设为 Try Again，这样可以不断进行输入。

打开“群组”图标，组建如图8-29所示的流程线，在“提示信息”图标中输入提示信息“请在屏幕上单击绘制矩形的位置”。然后将下面的“等待”图标设置为 Mouse Click。即当在屏幕上单击后，程序开始往下执行，“擦除”图标将前面的提示信息擦掉，“绘图”计算图标开始绘图。

打开“绘图”计算图标，在文本框中输入图8-30所示的表达式，`NumEntry`、`NumEntry2`和`NumEntry3`三个变量分别能够获取键盘输入的宽度、长度和边框宽度值。然后使用变量 `M`、`N`和`T`将数值代入`Box`函数中。

`Box`函数的使用格式是 `Box(pensize, x1, y1, x2, y2)`，即用`Pensize`指定的线宽在屏幕上从 `(x1,y1)`点到`(x2,y2)`点画一个方框，其中 `(x1,y1)`是方框左上角的坐标值，`(x2,y2)`是方框右下角的坐标值。当 `Pensize < 0`时，该方框将以黑色填充；当 `Pensize=0`时，该方框将以白色填充；当 `Pensize > 0`时，将以`Pensize`指定值的像素点个数绘制框线。在这个例子中，`CickX`、`CickY`

分别代表矩形的左上角位置。ClickX+M和ClickY+N则代表右下角位置。

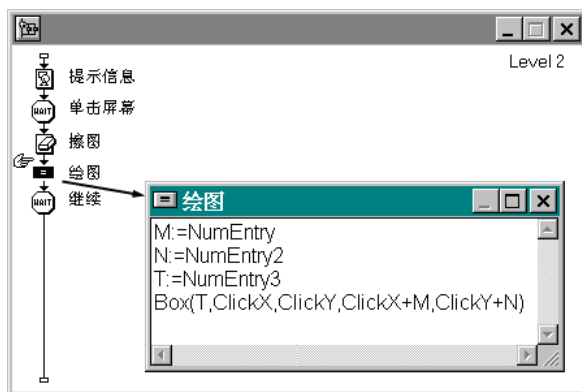




图8-30 设置计算图标

 **提示** 若用户在显示窗口中单击一次鼠标，则ClickX、ClickY的数值就等于从鼠标指针到窗口左边界或右边界的像素点数目，也相当于输入了矩形的绘制位置。

双击流程线上的“擦除”图标，打开 Properties：Erase Icon(擦除图标属性)对话框，如图 8-31所示，在该对话框的下面单击各个显示对象，将它们一一设为擦除对象，如图 8-31中的对象列表框中所示，被擦除的图标对象的标题出现在列表框中。



图8-31 设置擦除图标

 **技巧** 由于要擦除的对象是所有的图标，因此可以直接单击List下的Icon Preserve选项，设置为全部不保留。

在流程线的最后还有一个“单击屏幕”等待图标，如图 8-32所示，将它设为Mouse Click。当画完一个矩形后，单击屏幕，程序执行 Try Again分支，画面上又会提示用户重新输入。


 **提示** 如果选择了Show Button选项，屏幕上还会出现一个等待按钮，可以单击这个按钮来控制程序的执行。



图8-32 设置程序继续进行

8.6.2 Capitalize() 函数示例, Character类型

在下面的例子中, 使用 Capitalize() 函数来处理输入的字符, 它的使用格式通常是 Capitalize(String), 功能有两种: 该函数能将String指定的字符串中每个单词的首字母变成大写。另外, 它可以只将 String 指定的字符串中第一个单词的首字母变成大写, 格式为 Capitalize(String,1)。

由于程序要接收键盘上输入的字符, 所以要使用 EntryText 函数来接收字符串。如图 8-33 所示, 在“交互”图标下设置文本输入响应, 打开“群组”图标, 在流程线上分别放置“计算”图标、“显示”图标和“等待”图标, “等待”图标的功能与上一个程序中的功能相同, 它能将字符串更改的结果保留在屏幕上, 再单击一下等待按钮, 可以重新执行。

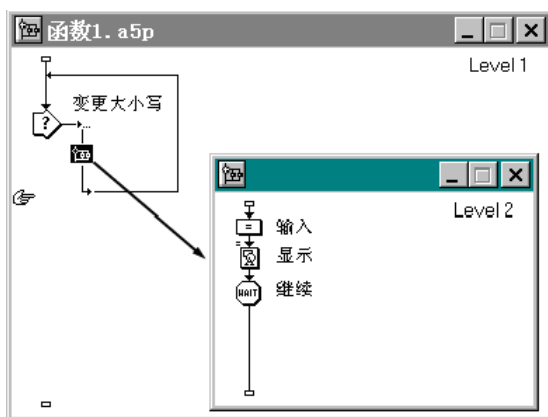


图8-33 流程图

如图8-34所示, 在“输入”计算图标的文本框中输入 EntryText 变量, 该变量能将键盘上输入的字符串赋给变量 aaa。在“显示”图标上连接“计算”图标, 如图 8-34 所示, 在文本框中输入 Name := Capitalize(aaa), 表达式的功能是将字符串变量 aaa 中的所有单词的第一个字母变为大写, 处理结果将显示在“显示”图标的演示窗口中。

在“交互”图标的文本框中, 通常只能输入很短的字符串, 根据程序的要求, 还要设置一下“交互”图标的文本框, 用户可以直接拖动文本框的调节方块将其拉至全屏。另外, 还要考虑到输入字符串的字体字号与输出字符串相同, 所以还要双击“交互”图标的文本输入

框，在弹出的 Properties：Interaction Text Field 对话框中设置字体字号，该设置一定要与“显示”图标中的文本设置相同，如图 8-35 中的黑圈内所示。

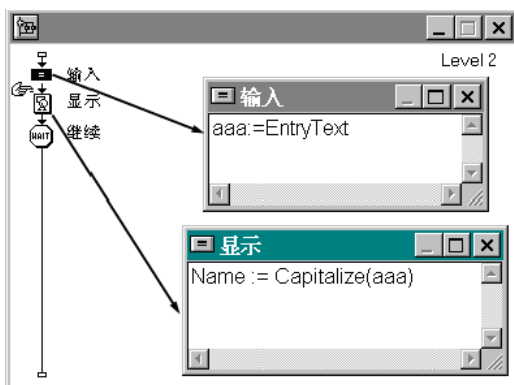


图8-34 输入函数表达式

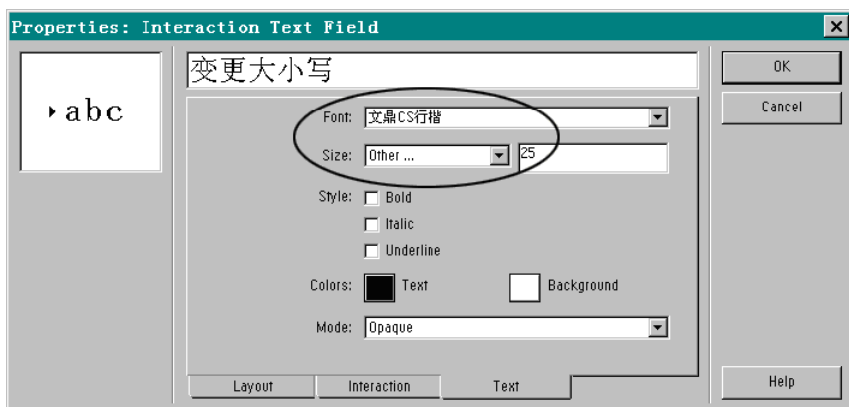


图8-35 设置字体

8.6.3 Beep()函数示例，General类型

Beep()函数可以控制PC喇叭，使系统响铃。在下面的例子中，使用 Circle函数来绘制一组圆，Circle函数的使用格式为 Circle(Pensize,x1,y1,x2,y2)，即在一个方框内绘制一个圆，该方框左上角的坐标是 (x1,y1)，右下角的坐标为 (x2,y2)。当 Pensize < 0 时，圆内以黑色填充；当 Pensize=0 时，整个圆以白色填充；当 Pensize > 0 时，圆周的线条宽度将等于 Pensize 指定的像素点值，中间不填充。

现在进行如下控制，即每当画完一个圆后，系统的 PC喇叭就会响一下。在程序中使用了 repeat while 函数来进行循环控制，用户可以在“计算”图标中的文本框中输入如下的语法结构：

```
repeat while x1<640
Circle(4, x1, y1, x2, y2)
  x1:=x1+10
  x2:=x2+10
```

```
y1:=y1+10  
y2:=y2+10  
Beep()  
end repeat
```


程序运行时，每当画完一个圆后，PC喇叭就响一下，每当循环一次，各参数就增加10，当X1的值大于640后，程序将跳出循环。

8.6.4 GoTo函数示例，Jump类型

在前面已经多次使用过该函数，它的使用格式为 `GoTo(IconID@"IconTitle")`，`IconID@"IconTitle"` 变量会获取 `IconTitle` (图标) 的ID号，因此 `GoTo` 函数就能使 Authorware 跳转到由此ID号指定的图标中去。

在跳转函数中还有一个函数 `JumpFile`，该函数的使用格式为：

`JumpFile("filename", ["variable1, variable2, ...", "folder"])`，其中 `filename` 为要打开的文件名称，`variable1, variable2, ...` 是该函数使用的各种参数，当函数跳转到指定文件后，它会将这些参数的值传递过去。`folder` 是用来指定 Authorware 记录文件的目录路径，通常情况下如果不进行设置，默认路径为 “C \ WINDOWS \ A5W_DATA \ ”。

 **注意** 在 Authorware 中，路径符号为 “\\”，而不是通常所使用的 “\”，例如，“c:\suo\yu\” 通常要写成 “c:\\suo\\yu”。

8.6.5 MediaPlayer、MediaPause、MediaSeek函数示例，General类型

`MediaPlayer` 和 `MediaPause` 函数用来控制数字电影、视频文件或声音文件的播放和停止。`MediaSeek` 函数的作用是设置指定图标中数字电影的帧数，使 Authorware 直接定位该帧的图像。

在下面的程序中，使用这三个函数来控制数字电影的播放。如图 8-36 所示，在程序的开始使用 “判断” 图标的一个分支来存放数字电影文件，插入数字电影文件的过程请查考本书的第7章。同时将 “判断” 图标的分支类型设置为 To Calculation Path，流程线将直接穿过 “判断” 图标。

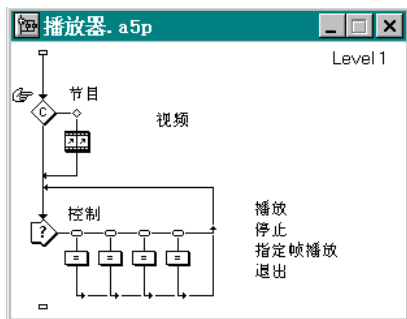


图8-36 设置流程线

在 “交互” 图标的分支中，设置了四个按钮响应分支，这四个按钮响应分别完成播放、停止、退出和指定帧播放。

打开交互分支中的“播放”计算图标，在文本框中输入如图 8-37所示的函数。@“视频”代表流程图上面“视频”图标的 ID号。当该分支开始响应后，MediaPlay(@“视频”)函数就可以从视频文件的第一帧开始播放。

如图 8-38所示，在“停止”计算图标的文本框中输入“MediaPause(@“视频”,TRUE)”，该函数共有两个参数，后面的参数可设为TRUE和FALSE两个值，当设为TRUE时，该函数执行终止播放命令，当设为FALSE时，该函数将不会控制视频文件的播放。

打开“指定帧播放”计算图标的文本框，在其中进行如图 8-39所示的语句控制。Random是一个随机函数，它的使用格式是Random(min, max, units)，其中min和max分别是函数变化的上下限，units用来指定函数的变化程度，如图 8-39所示的Random(1, 62,1)就表示函数值在1~62范围之间变化，变化的幅度是1。

随机变量的值赋予一个自定义的变量Fram，在MediaSeek(@“视频”, fram)函数中，fram是该函数指定的视频文件的具体的帧。因此，当该函数执行时，“数字电影”图标将直接跳至指定的帧。

当程序执行时，由于“判断”图标的流程线直接穿过流程线，所以在程序一开始执行时，将直接到达“交互”图标进行交互响应控制。如图 8-40所示，当播放视频文件时，如果要单击“指定帧播放”按钮，Random函数将随机控制播放的帧，因此，每次单击都可能播放不同的画面。



图8-37 播放函数



图8-38 停止播放函数



图8-39 控制指定帧



图8-40 播放视频信息

最后，还设置了一个退出按钮，在“退出”计算图标的文本框中你可以使用Quit(0)函数或者Quit(1)。



提示 Quit(x)函数的作用是使 Authorware直接退出演示过程。当 x=0时，返回到 Authorware窗口；当x=1时，返回到Windows的程序管理器，如果在演示过程中是从一

个文件跳转到另一个文件，将返回到先前的文件；x=2，返回到DOS环境。

8.6.6 FileType()函数示例，File类型

最后，再来给大家讲解一个函数，FileType(“filename”)函数的运算结果将返回一个数值，这个数值用于表示文件或目录的不同类型，其中0表示无此文件或错误，1~15数值分别表示不同的文件类型。下面使用这个函数来完成一种任务，即用它来查找指定的文件。程序如下：

```
file:="d:\\ttt\\myfile.txt"
If FileType(file)= 0 then
    msg:="指定的文件找不到"
else
    msg:="找到指定的文件"
end if
```

在这个程序中，要查找的文件是D盘ttt目录下的myfile.txt文件，当程序执行时，Authorware便开始查找，如果找不到，函数返回值是0，变量msg就会显示“指定的文件找不到”字符串，当找到后，函数的返回值就不为0，变量msg就会显示“找到指定的文件”字符串信息。

8.7 一点补充

在本章的最后，再补充一点有关变量和表达式的知识，另外还要讲解一下运算符的概念。如果用户对编程感兴趣，可以阅读本节，如果用户对计算机刚刚入门，则可以省略下面的内容。Authorware的编程同一些高级语言也有相似的地方，例如，它同样可以使用一些函数和变量，使用一些条件语句、循环语句，也可以将变量和函数结合在一起，组成某种语句。

语句与表达式有所区别，例如 $3 + 2$ 是一个表达式，但不是一个有效的语句。语句是Authorware的一个有效结构，它能够通过执行某种计算或执行某种操作来输出一个结果。例如，UserName := “John Smith” 是一个赋值语句，它的结果是将一个字符串定义给UserName变量。在Authorware 5中，由变量、函数和表达式构成了很多的语法结构，并和运算符一起组成了Authorware的Language类函数，你可以打开Authorware的函数窗口，在Category下拉列表框中找到这类函数，下面将分别进行介绍。

8.7.1 If.....Then结构

这是一种条件结构，它通常用在某些事件或结果的判断上。在日常生活中，常常说“如果今天下雪，那我将去滑雪，否则的话，只有去打扫卫生了”。你可以使用类似的语言，在Authorware中还允许使用一个或多个条件。

条件语句的使用格式如下：

```
If 条件 1 then      执行语句1
else if 条件 2 then  执行语句2
else                执行语句3
end if
```

在这个格式里，它的含义是：如果满足条件1，Authorware将执行语句1；而如果满足条件2，将执行语句2；否则的话，Authorware只能执行语句3；执行完这个条件结构后，Authorware将由End If来结束整个判断。

8.7.2 Repeat With In 结构

这是一种循环结构，通常它被应用在数组的计算上，有时也可以用来重复执行某个语句或操作，基本语法如下：

```
repeat with 变量 in 列表
  执行语句
end repeat
```

上面语法的含义是：如果变量元素在指定的列表中，该结构将重复执行下面的语句，执行完一个变量后，该函数就会指定下一个变量元素，直到该变量元素超出列表范围，才由 end repeat来结束循环。你可以阅读下面的函数示例：

```
List := [20, 30, 50]
Total := 0
repeat with X in List
  Total := Total + X
end repeat
```

在上面的例子中，指定一个数组 List，它里面有三个元素 20、30和50，下面的循环结构用来累加数组元素的和。例如，使用 X来指定列表中的元素，当循环开始后，X指定了列表中的第一个值 20，然后进行累加 $Total=0+20=20$ 。在进行第二次循环后，X被指定为 30，累加后 $Total=20+30=50$ 。当计算完最后一个值 50后，将退出循环，Total的值就变为 100。

8.7.3 Repeat With结构

这也是一种循环结构，但与上面的那种循环结构略有不同，在这种循环结构中直接限制了循环的范围，而且它不必再对其中的变量进行自加，循环时，该函数自动进行累加，这种循环结构的格式如下：

```
repeat with counter := start [down] to finish
  执行语句
end repeat
```

在上面的结构中，start和finish分别是函数循环的上下限，当函数执行到 start超出循环范围时，将自动退出循环。

下面使用这种函数来做一个例子，这个例子演示了系统如何重复响铃五次。如下面的程序所示，程序每次通过循环结构时，beep()函数总会使PC喇叭响一次，然后 i值增加 1，当 i增加到5后，Authorware将退出循环。

```
repeat with i := 1 to 5
  beep()
end repeat
```

Language函数还有一种循环结构 Repeat While，这种结构在前面已经讲过，可以参考前面的函数示例掌握它的用法。

8.7.4 Authorware中的运算符

下面再来了解一下 Authorware对于运算符的规定。

Authorware可以进行四则运算，在数值之间要靠各种运算符来连接，Authorware共有5种

运算符号，它们分别是：

- 1) 赋值运算符：“=” 能将运算符右边的值赋给运算符左边的变量，例如 `Name:="Su Guo-jun"`。
- 2) 关系运算符：“=” 等于、“<” 小于、“>” 大于、“<=” 小于等于、“>=” 大于等于、“<>” 不等于。
- 3) 逻辑运算符：“~” 逻辑非、“&” 逻辑与、“|” 逻辑或。
- 4) 算术运算符：“+” 加、“-” 减、“*” 乘、“/” 除、“**” 指数，例如 `X**2=X2`。
- 5) 连接运算符：“^”，可将两个字符串连接成一个字符串。

关于逻辑运算符再做一点提示：

- `Result = Variable1 & Variable2` 这个表达式的作用是：只有当 `Variable1` 和 `Variable2` 的值同时为真，`Result` 的值才为真。
- `Result = ~ Variable`，如果变量 `Variable` 的值为真时，`Result` 的值为假，反之，若 `Variable` 为假，`Result` 的值为真。
- `Result = Variable1 | Variable2`，只要变量 `Variable` 和 `Variable2` 中有一个为真，`Result` 的值就为真；只有 `Variable1` 和 `Variable2` 同为假时，`Result` 才为假。