# FoxPro for Windows 参数传递的技巧与应用

葛晓滨 （安徽省对外经济贸易计算中心 230061）

在 FoxPro for Windows（简称 FPW）的程序设计中，经常会使用参数传递功能在程序之间传 送数据，由于 FPW 应用程序间的参数传递机制与其他的 Windows 软件有所不同，虽不失大众化的软件参数传递的特点，但仍具有其独特的一面。研究 FPW 参数传递的过程并加以开发和利用，无疑对编写和设计 FPW 的应用系统会有很大的裨益。

## 一、参数传递的方式及规则

**1. 参数传递的两种方式**

FPW 规定应用程序及函数间参数的传递有两种方式：一种是值传递；一种是地址传递。如果用户不设定任何参数传递的条件，则默认参数传递是按照值传递方式进行的。

**2. 参数传递的两种应用格式**

一种是以程序方式调用，包括 EXE、APP、SPR、PRG 等类型文件间的参数传递，如命令格式：DO、ADD. SPR WITH VAR1, VAR2 便是在调用 ADD. SPR 程序的同时将变量 VAR1, VAR2 传递到 ADD. SPR 程序中；另一种是函数方式调用，包括 FPW 系统提供的函数以及用户自定义函数 UDF 等。如命令格式：RESULT = CALCVAR(VAR1, VAR2)是在调用 CALCVAR( )函数的同时将变量 VAR1, VAR2 传递到 CALCVAR( )函数中。

**3. SET UDFPARMS 命令在参数传递中的作用**

SET UDFPARMS 命令是 FPW 用于设定参数传递方式的语句，它有两种使用格式：

格式 1：SET UDFPARMS TO VALUE 将参数传递设置为传值方式

格式 2：SET UDFPARMS TO REFERENCE 将参数传递设置为传地址方式

这个语句一经使用，则在其后的程序（或函数）的参数传递过程均按此规则进行，但也有如下例外：

（1）将一对小括号“( )”置于调用参数上，可以强制将此参数传递方式调整为传值方式。

如 DO MYAPP. APP WITH (VAR1), (VAR2)或 = MYFUN((VAR1), (VAR2))

这两种形式的调用为强制传值方式的调用。

（2）将“@”符号置调用参数前，可以强制将此参数传递的方式置为传地址方式。

如 DO MYAPP. APP WITH @VAR1, @VAR2 或 = MYFUN(@VAR1, @VAR2)

这两种形式的参数传递强制为传地址方式。

以上两种例外，实际上为我们程序设计中随时改变参数传递的方式提供了方便。

**4. PARAMETERS( )及 TYPE( )函数的作用**

PARAMETERS( )函数可用于侦测 FPW 应用程序或函数实际传递参数的个数，通过这种侦测，可以避免程序或函数间参数的非法使用。FPW 设定调用和被调用程序间的参数传递是按照参数出现的先后次序对号入座的，参数间均以逗号分割。对于传递参数个数与接收参数个数不吻合的参数传递，FPW 有如下规则：

（1）如果传递数数量小于接收参数数量，则未被传递到值的参数内容被自动赋予逻辑值. F. 。

（2）如果传递参数数量大于接收参数数量，则 FPW 会给出“Wrong number of parameters”的出错信息。

为了避免这种参数传递过程中发生的错误，可以借助 PARAMETERS( )函数事先侦测出实际传送到被调用程序中的参数个数，并采取相应的容错措施避免参数传递数量错误的发生。

同时，FPW 提供的 TYPE( )函数可以有效地防止参数传递类型的错误，该函数可以返回所设定参数的类型，从而使用户可以及时甄别传送来的数据类型是否正确，并采取相应的容错措施。

**5. 参数的作用域**

（1）对于强制采用传值方式进行的程序调用，即使该变量已在调用程序中被设定为 PUBLIC 公用型变量，那么在被调用程序中对同名变量进行的任何修改也都不会影响到调用程序中该变量的值。否则，该变量的值会因被调用程序中该变量的变化而发生变化。

(2)对于传地址方式的程序调用,若用户在被调用程序或函数中修改了同参数同名变量的内容,则对该变量的修改会被反馈到调用程序中,使调用程序中的变量也发生相同的变化。

## 二、与参数传递相关的几则技巧

### 1.在 SPR 文件中接收参数的技巧

FPW 为用户提供了功能相当强大的屏幕设计工具——屏幕生成器,通过屏幕生成器功能,用户可以将屏幕设计转化为直观的 WYSWYG(所见即所得)的设计方式,而且可以自动将 SCX 屏幕文件编译为 SPR 或 EXE 等类型的可执行文件,用户还可以通过屏幕上的控件实现屏幕间的相互调用,这种屏幕间的相互调用必然会带来参数传递上的需求。屏幕程序间带参数传递调用的命令格式通常为:

DO〈屏幕文件名〉.spr WITH〈参数列表〉

其中,SPR 程序是由 FPW 在 SCX 屏幕文件基础上自动生成的可执行屏幕程序,而用户所直接使用的屏幕工作文件是 SCX 文件。SCX 文件为用户划分了若干个不同的段,如 SETUP 段、CLEANUP 段等。其中 SETUP 段是 SCX 中最早执行的段,但在 FPW 编译时通常被放置在 SPR 程序的第五个层次中,而非 SPR 程序的起始处。由于 FPW 规定子程序接收上一级程序参数的 PARAMETERS 语句必须是子程序的第一条可执行语句,这样置于 SETUP 段的 PARAMETERS 语句便无法正确接收到上一级屏幕传递过来的参数。解决此问题的方法是在被调用屏幕的 SETUP 段的第一行加上 #SECTION 1 预处理命令,然后再使用 PARAMETERS 语句。这样就可将 PARAMETERS 语句在 FPW 编译时,提前到 SPR 程序的首部,使其成为 SPR 程序的第一条可执行语句,从而实现 SPR 程序间参数的正常传递。

### 2.实现程序间数组的传递

数组是程序设计人员所喜爱使用的一类程序变量,它可以通过一个变量名(数组名)及数组坐标表述一组相关联的数据。在 FPW 的程序设计中,数组功能更是为程序设计人员提供了极大的方便,一方面它可以通过 FPW 的 SCATTER 及 GATHER 命令实现与 DBF 数据库间的数据交换;另一方面,它还拥有一系列功能极为强大的数组操作运算函数,实现了多种功能的运算。如数组元素的排序功能、查询功能等多种函数。FPW 定义数组的维数最大可至两维,相对于其他的编程语言不同的地方在于 FPW 数组可以存储任意类型的数据,而这正是 FPW 为程序员提供的方便之处。例如,编程人员可以将各种程序变量集成在一个数组中,通过数组实现大批量数据库程序间的传递。请注意:这正是采用数组传递程序间数据的优势所在,因为 FPW 设定程序间一次传递的变量数不能超过 24 个,由于这个规定,使得大批量的数据的传递受到限制,而采用数组,就使得这个问题迎刃而解。FPW 数组传递的规则如下:

(1)若需要传递整个数组给被调用程序或函数,则需要采用使用传地址方式的传递参数。使用 SET PARMS TO REFERENCE 命令或采用"@"强制功能。如命令:

DO SUB.APP WITH @ARRAY

(2)若只需要传递数组中的某个元素给被调用程序(或函数),则需要采用传值方式的调用,使用 SET PARMS TO VALUE 命令或采用"()"强制功能使用传值方式。如命令:DO SUB.APP WITH(ARRAY(2))

使用 FPW 传递数组时需要注意,对于指定数组元素的参数传递,这种传递始终是以传值方式进行的。

### 3.促进屏幕的集成化、优化系统结构

由于 FPW 提供了强大屏幕生成器功能,从而使得许多程序员完全摒弃了以往在 xBASE 系列软件中以手工书写 FMT 以及 PRG 程序的繁琐方式,代之以所见即所得的屏幕生成器自动产生的方式,而且可以通过 READ 语句的控制,实现屏幕的分级设计、逐层调用,从而实现 FPW 应用软件在总体布局上的层次化、模块化。但由于 READ 语句的调控层最深只能到第 5 层,使得程序设计人员不得不在系统设计中要充分考虑屏幕调用层次的限制,采取优化程序设计、集成化程序模块的方法。例如,可以将多个内容上相似、功能上不同的屏幕模块集成到一起,形成一个通用模块,这样不仅解决上述问题,提高系统的集成度,而且还可以大大提高程序设计效率。采用这种设计方式,参数传递的功能和作用就要扮演一个重要的角色,我们以一个具体的程序设计实例说明这种方法和技巧。

数据编辑和查询模块是任何一个管理型应用软件所必不可少的功能。在 FPW 中,我们可以提取它们在屏幕设计上的共性,将这两者集成到一起,形成一个通用模块。数据编辑和查询模块的共性在于它们都需要提供给用户一个数据窗口,该窗口需要列示出系统所涉及到的所有数据项。不同之处在于前者需要一个可对

数据项进行编辑的控件，通过它的控制，可以让用户修改屏幕上的数据项，而后者需要一个可进行数据查询的控件使当前数据窗口上的数据项定位于用户所设定的条件上。根据这些共性及不同，我们在集成这两个屏幕时，可以先设计出两者公用的数据屏幕部分，然后将编辑控件和查询控件也同时放置到屏幕上，同时在定义编辑控件和查询控件时，在其 WHEN 语句的 FUNCTION 功能上加一个用于调控激活或屏蔽该控件的控制参数，设计好屏幕上的有关参数后，再打开屏幕的 SETUP 段，用 PARAMETERS 语句接收调用程序传递的控制参数，并且对这个参数进行判定。若参数为编辑标志，则自动将编辑控件的调控参数置为激活状态，同时将查询控件的调控参数置为屏蔽状态。反之，若参数为查询标志，这自动将编辑控件的调控参数置为屏蔽状态，同时将查询控件的调控参数置为激活状态。由此通过 PARAMETERS 的应用，使得 FPW 应用系统的数据编辑与查询模块二合一。而且一旦系统所涉及到的数据项发生变化，编程人员只需要修改这一个共用屏幕中的数据项即可，从而优化了程序设计结构，减少了程序员的工作量，为程序的后续修改及调整也提供了方便。

**4. 应用程序的保护**

在 FPW 应用软件的开发过程中，如果将所有的过程都集中到一个可执行模块中，就会使得应用系统过于庞大，影响系统的加载及运行速度。为此，需要我们按照程序的功能划分出若干个子模块，通过这些子模块，完成相应的子功能，再通过一个主控模块来管理和调配这些子模块。按照这种设计模式，系统在提交给用户时就会有若干个 EXE 或 APP 类型的可执行文件，而能供用户使用的只有主控模块，其他的模块不能被用户所直接使用，由于在 WINDOWS 中，用户可以调用执行任何的 EXE 文件。那么如何限制用户对这些子模块的使用呢？参数传递的控制是实现这种使用限制的一种行之有效的途径。具体的动作方法是在 FPW 应用系统的各个子模块的程序前加上一个用户无法猜测到的调用参数，这个参数由主程序在调用这个子模块时传递而来，然后在子模块中加上对所传递参数的判定。如果传递过来的参数正确，就继续执行下面的程序，否则就退出这个子模块程序的执行。具体实例如下：

例如有一个 FPW 应用系统的主程序 MAIN.EXE 及其分支程序 SR.EXE，在主程序中使用 DO SR.EXE WITH "GXB"语句调用分支模块 SR.EXE，在 SR.EXE 的起始部分加入如下的程序语句：

```
PARAMETERS CALLVAR
IF PARAMETERS() #1.OR.CALLVAR#"GXB"
= MSGBOX("SR.EXE 不能直接运行，请执行 MAIN.EXE","揭示信息",0)
ENDIF
.....
```

这样，通过参数调用的控制实现了对应用程序使用的限制，从而有效地保障了 FPW 应用系统的正常运行。同时这个问题的解决，将提高 FPW 应用系统的封闭性和安全性。