

LabVIEW 中如何调用 Windows API

Lancker(原Simwe虚拟仪器技术版管理员, 创建人之一)

「LabVIEW 没有提供这样的功能, 必须呼叫 Windows API」, 有时候笔者常看到有些朋友会问许多问题, 实在是因为 LabVIEW 本身不提供这些功能, 或者实现很困难, 所以才会这样回答。虽然这样回答有点偷懒, 或者说不负责任, 但这的确是事实, LabVIEW 所提供的模块, 虽然也不在少数, 但是主要用于测控软件开发, 要想变点花样, 通常是行不通的, 这是笔者决定开始撰写本文的主要原因。

感觉上 LabVIEW 程式要呼叫 Windows API 是一件比较困难的事情, 或者说比较麻烦的事情, 但别忘了 Windows API 是大家的, 凡是在 Windows 工作环境底下执行的应用程式, 都有权利呼叫 Windows API。其实 LabVIEW 和 Visual C++/Visual Basic/Delphi 等开发软件一样, 可以呼叫 API, 而且实现比较方便, 与调用其他动态连接库文件 (.DLL) 几乎一样。

(笔者个人认为要做到将 API 函数灵活调用到 LV, 最好有 VC/VB 编程基础。我觉得如果花一两个月学习 VB, 对与 LV 的提高会起到意想不到的效果。VB 和 LabVIEW 都是电子工程师喜欢用于开发测试软件的工具, 其中有许多相似之处。它们具有入门简单, 方便地调用/移植其他代码。VB 调用 API 的参考资料很多, 但介绍如何在 LV 中调用 API 的资料却为数不多。本人写这篇文章, 虽然错误难免, 但还是希望对大家有所帮助。)

1、Windows API 简介:

1.1 简介:

Windows 作为多线程系统除了协调应用程式的执行、分配记忆体、管理系统资源...之外, 她同时也是一个很大的服务中心, 呼叫这个服务中心的各种服务(每一种服务就是一个函数), 可以帮应用程式达到开启视窗、描绘图形、使用周边设备...等目的, 由於这些函数服务的对象是应用程式(Application), 所以便称之为 Application Programming Interface, 简称 API 函数。

1.2 但 Windows API 与 C 语言最亲近

虽然说呼叫 Windows API(以下简称 API 或 API 函数)是每一个应用程式的权利, 但不可否认的 API 却与 C 语言最亲近, 因为 API 函数在参数的传递上就是以 C 语言为标准。

但这并不表示 LV 程式不能呼叫含有参数的 API 函数, 如果传递的参数是单纯的资料型别, 例如「整数」, 则 LV 与 C 语言还是相通的, 如果是特殊的资料型别(包含「字串」), 则必须遵循一定的规范, 否则不是无法得到正确的结果, 就是因为违反规定而被踢出系统。

2、使用 Windows API 的难处

当我们要开始使用 API 时, 必须知道叁件事情:(1) 要呼叫哪一个 API 函数;(2) 如何在 LV 中调用 API 函数;(3) 如何传递参数。

2.1 要呼叫哪一个 API 函数

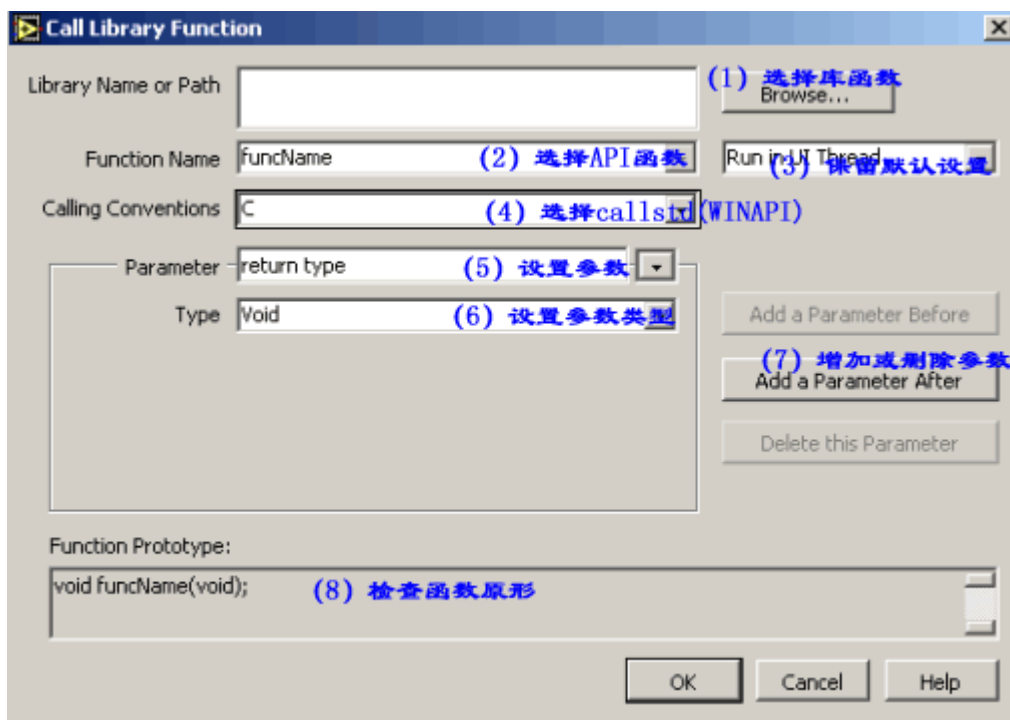
这是以上叁件事情当中最困难的一件，主要的原因是 Windows 的 API 实在太多了，大约有 1500 个，这还不包含 OLE、ODBC...等特殊的 API，此外，如果我们把 API 按不同性质加以分类，则使用每一类 API 函数所应具备的背景知识亦各有不同，以系统注册区相关的 API 函数为例，就必须先了解 Windows 如何安排系统注册区，以及存取系统注册区的方式。

不过也不必被 1500++ 个函数给打退堂鼓了，因为不是所有的程式设计都要仰赖 API，当我们面对一个问题时，首先还是寻求 LV 的解决方案，如果 LV 实在无法解决，才考虑使用 API，例如 LV6.0 以前对注册表操作需要调用 API，但现在新的版本有专门的模块（但实质上与调用 API 一样，只是操作起来方便了）。

要想了解哪些是常用 API，它们的功能，所属的动态连接库，可以查看一些手册，也可以查看一些 VB 中应用的例子（笔者就是从 VB 的代码中熟悉了一些 API，然后知道如何在 LV 中调用它们）。

2.2 如何在 LV 中调用 API 函数。

在 LV 中设置 API 其实与调用其他 .DLL 相同。选择模板中的 Functions->Advanced->Call Library Function Node，然后点击右键，从快捷菜单中选择 Configure。出现一对话框如下：



使用 Browse 到 Windows(或 WinNT)下面的 system32 中先选择 API 的库函数，如 User32.dll，然后在 FuncName 的下拉式菜单中选择你需要的函数，在 Calling Convention 中选择“stdcall(WINAPI)”。下面的工作是设置传递参数。

2.3 如何传递参数

LV 的参数类型中提供了几种在 LV 中常用的类型：

Numeric(数值): 整数 (8-, 16-, and 32-bit signed and unsigned integers), 单精度 (Single-precision) 和双精度 (double-precision);

Array(数组): 实现数组类型传递;

String(字符串): 实现字符串数据传递;

ActiveX: 处理 ActiveX 数据;

Waveform/Digital Waveform/ Digital Table: 主要为 LV 数据传递的类型, 一般 API 涉及不多。

Adapt to Type: 适合自定义参数。

由于 API 采用了 C 语言的参数传递方式, 而 C 语言的参数传递又与 LV 有着不小的差异, 以致不少呼叫 API 所造成的错误都发生在参数传递时, 而本期我们并不想花太多的篇幅放在如何传递参数上面, 以后有机会将放在如何调用 dll 方面的文章中作详细讲解。下面举些例子, 来说明如何调用 API, 我想让大家明白在 LV 中调用 API 其实不难。希望对大家入门有帮助。

3. LV 调用 API 示例:

(注: 由于在 VC/VB 中调用 API 函数的参数设置对 LV 中调用 API 有帮助, 特地将起代码一并附上, 以供参考)

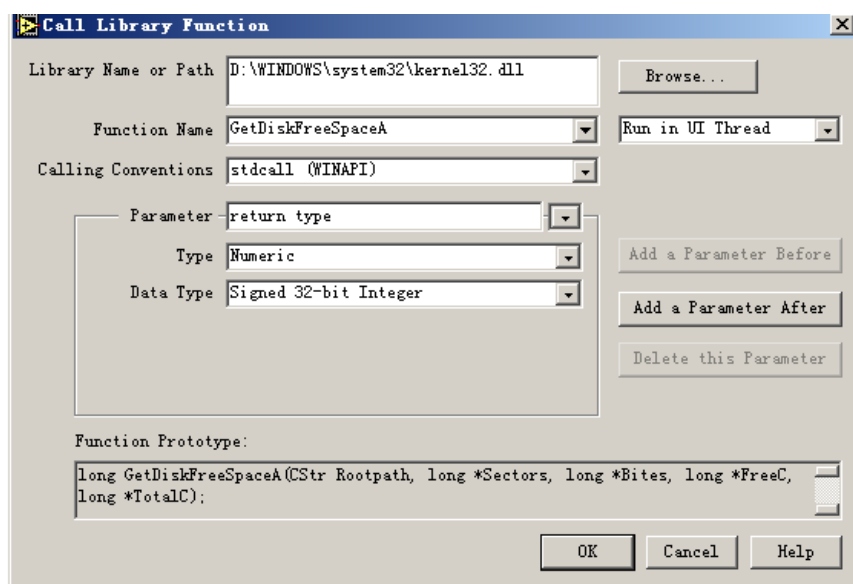
3.1 LV 获得硬盘的可用空间 (数字指针和字符串传递):

API 函数: GetCursorPos

```
VC中定义: BOOL GetDiskFreeSpace(  
LPCTSTR lpRootPathName,    // pointer to root path  
LPDWORD lpSectorsPerCluster, // pointer to sectors per cluster  
LPDWORD lpBytesPerSector,   // pointer to bytes per sector  
LPDWORD lpNumberOfFreeClusters,  
// pointer to number of free clusters  
LPDWORD lpTotalNumberOfClusters  
// pointer to total number of clusters
```

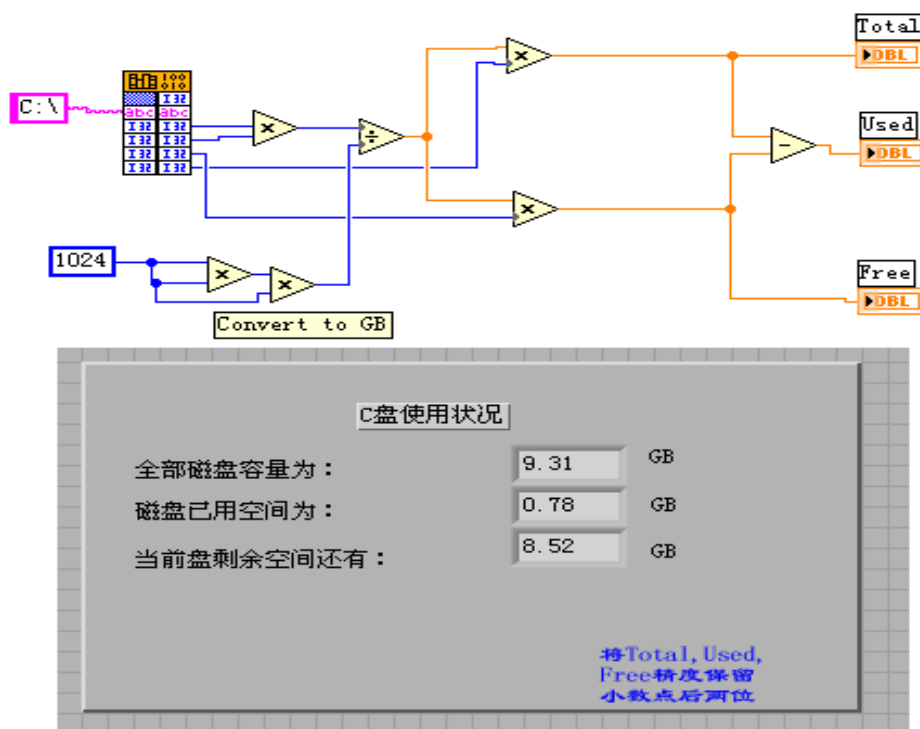
```
VB 中宣告为: Declare Function GetDiskFreeSpace Lib "kernel32" Alias  
"GetDiskFreeSpaceA"  
(ByVal lpRootPathName As String,  
lpSectorsPerCluster As Long,  
lpBytesPerSector As Long,  
lpNumberOfFreeClusters As Long,
```

LV 中这样设置：



检测硬盘剩余空间程序和控制面板如下图：

(源代码见附件 (LV6.1 版本) : DiskFreeSpace.vi)



3.2 LV 中获的鼠标的坐标 (自定义型数据传递)：

API 函数：GetCursorPos

注:任何转载或摘抄请注明文章出处(中国仿真互动 <http://www.Simwe.com/>)

```

VC中定义: BOOL GetCursorPos(
    LPPPOINT lpPoint // address of structure for cursor position
);
typedef struct tagPOINT {
    LONG x;
    LONG y;
} POINT;
  
```

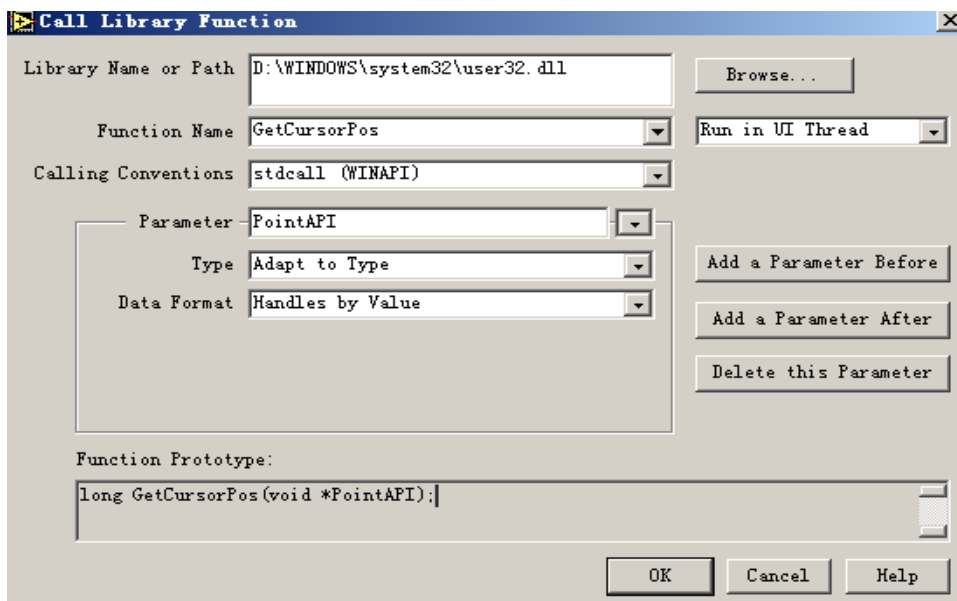
VB 中宣告为: Declare Function GetCursorPos Lib "user32" (lpPoint As POINTAPI) As Long

Type POINTAPI ' POINTAPI 为一自定型别

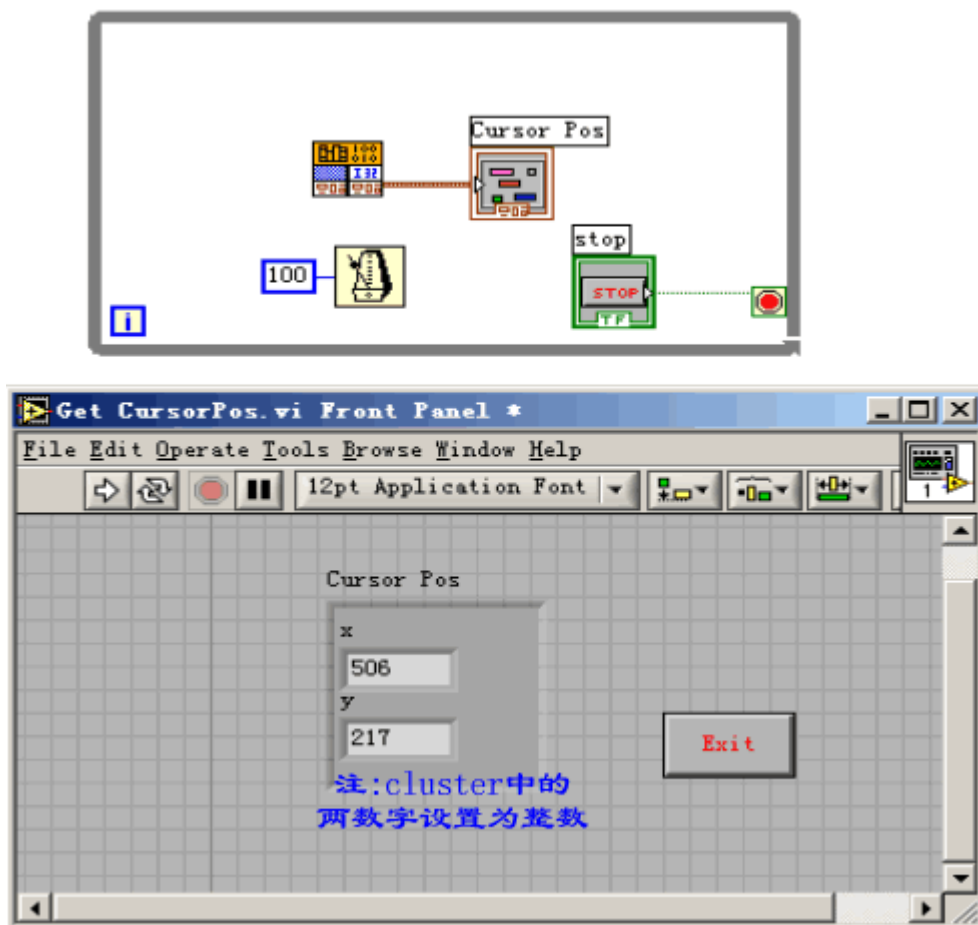
x As Long

y As Long

LV 中我们使用 Adapt to Type 这样设置：



下面我们制作一个小程序，每隔 100ms 显示一下鼠标的坐标，Block Diagram 和 Front Panel 如下：（源代码见附件（LV6.1 版本）：[Get_CursorPos.vi](#)）



3.3 LV中获得磁盘类别 (其他略去, 代码见附件Get_Disk_Type.vi)

3.4 LV中开关光驱功能 (其他略去, 代码见附件CDDoor_Tools.vi)

4. 了解更多 API 函数：

了解和查阅 API 函数的功能和参数设置，主要途径有：a) 查阅介绍 Windows API 的图书；b) 查阅微软 MSDN Library 有 VC 定义的原型；c) 一些编程网站的在线查询；d) 通过学习 VB 中如何调用了解 API 函数的例子。

附：如何在 VB 中宣告 API 函数

了解 API 函数的宣告并不困难，因为我们可以请 VB 的「API 检视员」来帮忙，以下是使用「API 检视员」的方法：

1. 首先选取 VB 功能表的「增益集/增益功能管理员」，然后在「增益功能管理员」交谈窗中核取「VB API Viewer」，按下「确定」钮后，VB 的「增益集」功能表栏底下就会出现「API 检视员」，选取此一命令，即可启动「API 检视员」。

2. 第一次执行 API 检视员时，须利用功能表的「档案/载入文字档」载入 VB Winapi 目录底下的 Win32api.txt，接着在「可选项的项目」底下即会列出所有的 API 函数，若我们双按其中的函数，则该函数的宣告即会出现在「选取的项目」底下，此时再按下「复制」钮，可将选取的函数宣告复制到剪

贴簿，过程如图-2，接着回到 VB 的程式视窗，再选取功能表的「编辑/复制」，即可将函数的宣告从剪贴簿中复制过来。

图中利用「API 检视员」将 API 的宣告复制到剪贴簿接下来请注意 API 宣告式复制到 VB 程式的位置，此时您有两种选择：(1) 先利用 VB 功能表的「专案/新增模组」新增一个一般模组(.bas 档)，然后将 API 宣告式复制到此一模组的程式视窗中，(2) 将 API 宣告式复制到表单程式视窗的“(一般)”区块底下，但复制过来之后，必须在 Declare 前面加上 Private 保留字。

