

Hints & Tips for 'Time To Win' Users

August 16th, 1995

by Michaël RENARD

CIS: 100042,3646



I would like to welcome all of you to the second issue of The Hints & Tips for TIME TO WIN Users.

I am writing these Hints & Tips initially to achieve the following results:

1. Disseminate my discoveries and ideas about 'Time To Win'.
2. Share some programming tips and techniques.
3. Act as a forum for 'Time To Win' interest.

Contents

What You Can Do To Help!

How to ...

About The Editor (for those of you who actually care...)

What You Can Do To Help!

Ideally, these Hints & Tips is for everyone to use. If you have something that you think others would benefit from, be it a neat idea, programming tip or trick, piece of elegant code, bug report (No! Never!), or whatever strikes your fancy, then I encourage you to forward it to me at 100042,3646. Because of its electronic nature, this Hints & Tips can be as large as necessary to accommodate any print-worthy information its readers may submit. Actually, the more assistance I get from all of you, the easier it will be for me!

Please also keep in mind that this is a completely volunteer effort on my part. The Hints & Tips will be free of charge to anyone who wants it. There will be no money changing hands in any form, so no advertising, no compensation for articles submitted, etc.

I am going to try a publishing schedule of every two weeks, time permitting. The style will be quite casual, so please excuse any lack of eloquence on my part.

[Contents](#)

About The Editor

For those of you who actually care, I thought I would tell a little bit about myself:

I have been a Basic 4.0/Basic 4.5/Basic PDS 7.0/ Basic PDS 7.1/Visual Basic 3.0 analyst and programmer (in that skill order) for the past 8 years or so. I've developed many applications about Time Management, Access Control and for a little part in Job Control. For all DOS language, I've created some interfaces and routines (also communication) in assembly language (MASM). For the Visual Basic language, I've created many routines in VC++.

Begin 1995, I started doing some add-ons for VB 3.0 and I've placed these add-ons on the Shareware policy.

These add-ons are :

Time To Win	+ 620 routines.
Time To Win Light	+ 360 routines.
VB/Error Handler	add/remove automatically error's management.
VB/Tracer-Profiler	trace/profile your VB application.
MC BUNDLE	bundle of 'Time To Win', 'VB/Error Handler', 'VB/Tracer-Profiler'

At this time and after the success of Time To Win, I write the 16/32 bits version for VB 4.0

[Contents](#)

How to ...

How to Extract the extension of a filename including '.' in the path

How to Parse some specials strings

How to Find duplicate files on drive(s)

Extract the extension of a filename including '.' in the path

When you've a fully qualified filename, it is possible that the path contains embedded period. If you want to extract the extension of the filename, it is not very easy to write it in VB.

The solution is to use to [cGetInPartR](#) routine.

Example 1 : Extraction of the extension

```
Dim FullFileName As String
Dim Extension      As String
Dim PathAndFile    As String

FullFileName = "C:\WINDOWS\CSERVE\..\SYSTEM\TIME2WIN.DLL"

Extension = cGetInPartR(FullFileName, ".", True)
PathAndFile = cGetInPartR(FullFileName, ".", False)

Debug.Print "FullFileName is "; FullFileName
Debug.Print "Extension is "; Extension
Debug.Print "PathAndFile is "; PathAndFile
```

Example 2 : Extraction of the extension, the filename and the pathname

```
Dim FullFileName As String
Dim Extension      As String
Dim PathAndFile    As String
Dim FileName       As String
Dim PathName       As String

FullFileName = "C:\WINDOWS\CSERVE\..\SYSTEM\TIME2WIN.DLL"

Extension = cGetInPartR(FullFileName, ".", True)
PathAndFile = cGetInPartR(FullFileName, ".", False)
FileName = cGetInPartR(PathAndFile, "\", True)
PathName = cGetInPartR(PathAndFile, "\", False)

Debug.Print "FullFileName is "; FullFileName
Debug.Print "Extension is "; Extension
Debug.Print "PathAndFile is "; PathAndFile
Debug.Print "FileName is "; FileName
Debug.Print "PathName is "; PathName
```

How to ...

Parse some specials strings

[Problem :](#)

I'm reading about 10-20 barcodes a second through the comm port which is hooked to a scanner, i read it into a string.

```
timer1_timer
Input$=comm1.input 'read barcode reader
{strip chr$(13) and chr$(10) routine} - I need HELP here!
add barcodes to database
end sub
```

The input\$ contains some carriage returns and line feeds, it contains also a few barcode numbers which are separated by these random chr\$(13) and chr\$(10). Sometimes there is 2 chr\$(13) or there may be 2 chr\$(10), its unpredictable.

I need to strip all these chr\$(13) and chr\$(10) within the string and pull out just the barcode numbers. Im adding these barcode numbers to a database. If Time To Win provides a way to do this could you show me how or if not could you provide such a function?

I can code this in vb but it is not efficient for such a high speed barcode reader. I am in dire need for a function call on this one.

[Solution provided :](#)

I think that the solution is to use three functions of 'TIME TO WIN'.

- 1) Change any chr\$(13) into chr\$(10) with function 'cChangeChars'
- 2) Count the number of chr\$(10) with function 'cCount'
- 3) Extract all sub-string separated by a chr\$(10) with function 'cGetln'

Below there is the piece of code :

```
Dim Cr      As String
Dim Lf      As String
Dim Txt     As String
Dim Part    As String
Dim Tmp     As String
Dim i       As Integer
Dim j       As Integer
Dim n       As Integer

' some initializations
i = 0
j = 0
n = 0
Part = ""
Tmp = ""
Cr = Chr$(13)
Lf = Chr$(10)
Txt = "634CD" + Cr + Cr + Cr + Lf + "A125" + Lf + Lf + Cr + Lf + "K86L9200"

' change all Cr by Lf
Tmp = Txt
Call cChangeChars(Tmp, Cr, Lf)

' count the number of Lf
n = cCount(Tmp, Lf)
```

```
' add 1 to number of Lf for the number of sub-string  
n = n + 1
```

```
' extract all sub-string separated by a Lf  
For i = 1 To n  
    ' extract each sub-string  
    Part = cGetIn(Tmp, Lf, i)  
    ' check if sub-string is not empty  
    If (Len(Part) > 0) Then  
        ' process each sub-string  
        j = j + 1  
        Debug.Print "Sub-String " & j & " is " & Part  
    End If  
Next i
```

Below there is the result :

```
Sub-String 1 is 634CD  
Sub-String 2 is A125  
Sub-String 3 is K86L9200
```

How to ...

Find duplicate files on drive(s)

[Problem :](#)

I registered Time2Win a short time ago and have been toying with getting FilesInDirectory to search an entire hard drive for specified files and then dumps its findings into a list box to use in a series of VB utilities I am writing. No luck so far. Do you have any suggestions as to how I can extend the search parameters for FilesInDirectory?

[Solution provided :](#)

- 1) Read all directories for each drive selected with the function 'cAllSubDirectories'
- 2) Write and append all sub-directories to only one ascii file.
- 3) Read each directory in the ascii file and use the 'cFilesInDirectory' to find each file in each directory.
- 4) Write (and display for progress) the following information separated by a ',' for each file in each directory in an other file :
 - The filename;
 - The directory;
 - The date (using the 'cFileLastDateModified');
 - The time (using the 'cFileLastTimeModified');
 - The size (using the 'cFileSize');
 - A Cr/Lf for ending information.
- 5) Use the 'cFileSort' to sort the file (warning 16384 record max.).
- 6) Use program techniques to check the dump that you want :
 - Same name (no problem);
 - Same size (no problem);
 - Same contents (using the 'cCmpFileContents');
 - Same date and time (using the 'cCmpFileTime');

[Program :](#)

Dim rdChan1	As Integer	' channel for reading file 1
Dim rdChan2	As Integer	' channel for reading file 2
Dim wrChan	As Integer	' channel for writing file
Dim fDirs	As String	' name of the temporary file holding all directories
Dim fFiles	As String	' name of the temporary file holding all files
Dim fFilesSorted1	As String	' name of the temporary file holding all files sorted by name,
directory, size		
Dim fFilesSorted2	As String	' name of the temporary file holding all files sorted by name,
directory, size		
Dim nDir	As Integer	' number of directories founden
Dim nFile	As Integer	' number of files founden
Dim sDir	As String	' directory in progress
Dim sFile	As String	' file in progress
Dim sName	As String	' filename in progress
Dim sDate	As String	' date of the file in progress
Dim sTime	As String	' time of the file in progress
Dim sSize	As Long	' size of the file in progress
Dim sDir2	As String	' directory 2 in progress
Dim sFile2	As String	' file 2 in progress
Dim sSize2	As Long	' size of the file 2 in progress
Dim fSeek	As Long	' file seeking
'some initializations		
rdChan1 = 1		
rdChan2 = 2		
wrChan = 3		


```
fDirs = "C:\TMP\~ALLDIR.TMP"
fFiles = "C:\TMP\~ALLFILE.TMP"
fFilesSorted1 = "C:\TMP\~ALLFLS1.TMP"
fFilesSorted2 = "C:\TMP\~ALLFLS2.TMP"
```

```
nDir = -1
```

```
' Point 1 and 2 :
```

```
' read all directories on drive C: in one call in one string
sDir = cAllSubDirectories("C:", nDir)
debug.print "Directories founden on drive C are " & nDir
```

```
' write the directory string into a file
Close #wrChan
Open fDirs For Output Shared As #wrChan
Print #wrChan, sDir;
Close #wrChan
```

```
' read all directories on drive D: in one call in one string
sDir = cAllSubDirectories("D:", nDir)
debug.print "Directories founden on drive D are " & nDir
```

```
' write the directory string into a file
Close #wrChan
Open fDirs For Append Shared As #wrChan
Print #wrChan, sDir;
Close #wrChan
```

```
' Point 3 and 4 :
```

```
' open the file for reading all directories
Close #rdChan1
Open fDirs For Input Shared As #rdChan1
' open the file for writing all files
Close #wrChan
Open fFiles For Output Shared As #wrChan
' process each directory one by one
While Not EOF(rdChan1)
    ' read each directory name
    Line Input #rdChan1, sDir
    ' read the first file in the directory in progress
    sFile = cFilesInDirectory(sDir + "\*.*", True)
    ' check if the file is correct
    Do While (Len(sFile) > 0)
        ' yes, check the filesize
        sSize = cFileSize(sDir + "\" + sFile)
        If (sSize >= 0) Then
            ' file size is correct : save filename, directory, date, time and size (all items separated by a comma)
            ' sDate = cFileLastDateModified(sDir + "\" + sFile)
            ' sTime = cFileLastTimeModified(sDir + "\" + sFile)
            Print #wrChan, sFile; ",";
            Print #wrChan, sDir; ",";
            ' Print #wrChan, sDate; ",";
            ' Print #wrChan, sTime; ",";
            Print #wrChan, Trim(sSize)
            ' read the next file in the directory in progress
            sFile = cFilesInDirectory(sDir + "\*.*", False)
        End If
    Loop
Wend
```

' close opened files

Close #rdChan1

Close #wrChan

' Point 5 :

' sort the file holding all files (ascii and record sort)

Debug.Print "File Length : "; cFileSort(fFiles, fFilesSorted1, SORT_ASCENDING Or SORT_CASE_INSENSITIVE,
-1, -1, -1, nFile)

Debug.Print "Total Files : "; nFile

Debug.Print "File Copy : "; cFileCopy(fFilesSorted1, fFilesSorted2)

' Point 6 :

