VBIT Fileroutines for Visual Basic

FileExist	Check if a file exist. FileExist returns (True/False)
FileFindPath	Searches for a given filename, and returns the filename compelet with path
FileGetAttr	Returns the attributes for a file as string. Format : "ADHRS"
FileGetDate	Returns date for a file. Format : "YYYYMMDD"
FileGetExt	Returns extension of a complete filename.
FileGetFileName	Returns filename without extention of a complete filename.(D:\path\file.ext)
FileGetPath	Returns PATH-part of a complete filename (D:\PATH\FILE.EXT)
FileGetSize	Returns filesize as Long
FileGetTime	Returns the time for a file as string. Format : "HHMMSS"
FileGetVersion	Get the current version number of VBIT
IniFileGetString	Read INI file
IniFilePutString	Write to INI file

Include VBIT.BAS and VBITITAB.BAS in your projects

See also VBITITAB.WRI,VBITVTSS.WRI, VBIT.WRI and VBIT.HLP for description of other VBIT functions.

1996 © InfoTech AS, Bergen, Norway

Function   FileExist

Function to check if a file exist. FileExist returns (True/False)

Usage:
```
Function FileExist%(fileName$)
```

Example:
```
AutoTab& = ITabRead("C:\AUTOEXEC.BAT",IT_TEXTFILE+IT_ASCII)
' edit file......
' Write back, but keep original file as ".BAK":
ITabWrite(autoTab&,"C:\AUTOEXEC.$$$",IT_TEXTFILE+IT_ASCII)
If FileExist("C:\AUTOEXEC.BAK") Then
Kill "C:\AUTOEXEC.BAK"
End If
Name "C:\AUTOEXEC.BAT" As "C:\AUTOEXEC.BAK"
Name "C:\AUTOEXEC.$$$" As "C:\AUTOEXEC.BAT"
ITabDelete autoTab&
```

Function   FileFindPath

This function searches for a given filename, and returns the filename complete with path.

The search is first performed in the current path, if the file is not found the search continues in the \WINDOWS\SYSTEM\ directory, and finally in the PATH settings from environment.

Usage :
```
f_file$ =  FileFindPath$(file$)
```

Example:
```
file$ =  FileFindPath$("VBIT.DLL")

'Result : file$: "C:\WINDOWS\SYSTEM\VBIT.DLL"
```

Function   FileGetAttr

Returns the attributes for a file as string. Format : "ADHRS"

>   A: Archive (set when file is changed - used by back-up systems)
>   D: Directory name
>   H: Hidden file
>   R: Read-Only file
>   S: System file

Usage:
```
file$ = "C:\WINDOWS\SYSTEM\VBIT.DLL"
Attr$ = FileGetAttr(file$)
```

Function FileGetDate

Returns date for a file. Format : "YYYYMMDD"

Usage:
```
file$ = "C:\WINDOWS\SYSTEM\VBIT.DLL"
date$ = FileGetDate(file$)
```

Function FileGetExt

Returns extension of a complete filename.

Usage:
```
file$ = FileGetExt(file$)
```

Example:
```
fil$ = "C:\WINDOWS\SYSTEM\VBIT.DLL"
f_ext$  = FileGetExt(fil$)           ' f_ext:  "DLL"
f_fil$  = FileGetFilename(fil$)      ' f_fil:  "VBIT"
f_name$ = FileGetName(fil$)          ' f_name: "VBIT.DLL"
f_path$ = FileGetPath(fil$)          ' f_path: "C:\WINDOWS\SYSTEM\"
```

Function FileGetFileName

Returns filename without extention of a complete filename.(D:\path\file.ext)

Usage:
```
f_fil$  = FileGetFilename(fil$)
```

Example:
```
fil$ = "C:\WINDOWS\SYSTEM\VBIT.DLL"
f_fil$  = FileGetFilename(fil$)      ' f_fil:  "VBIT"
```

Function FileGetPath

Returns PATH-part of a complete filename (D:\PATH\FILE.EXT)

Usage:
```
path$ =  FileGetPath$(file$)
```

Example:
```
wordpath$ =  FileGetPath$(FileFindPath$("WINWORD.EXE"))
'wordpath$: "D:\WINWORD\"
```

Function FileGetSize

Returns filesize as Long. If the file does not exist, the return value is 0.
(Visual Basic's FileLen causes a run-time error if the file is uavaliable)

Usage:
```
size& = FileGetSize(file$)
```

Function FileGetTime

Returns the time for a file as string. Format : "HHMMSS"

Usage:
```
Time$ = FileGetTime(file$)
```

Example:
```
file$ = "WINWORD.EXE"
time$ = FileGetTime(FileFindPath(file$))
```

Function FileGetVersion

Get version number of given file.
If the given file is not found in the given path (or current directory), the routine will search for it in \WINDOWS\
SYSTEM\ and eventually the path given by the current environment.

Usage:
```
Result$ = FileGetVersion(FileName$)
```

The version number is returned as String, but it can be converted to a Double.

The version information is obtained from the standard Windows version information found in most files made for
Windows, e.g. DLL-, EXE-, DRV- files. If this information is not present in the given file, this routine will make up a
version number based on the file's date and time.

The Windows file version information is internally given by two 32-bit numbers, but is usually presented as four 16-bit
numbers seperated by periods, e.g. **"3.10.002.5"**. There are also similar information for product version, but this is
ignored here.

This routine will return a string which is possible to convert to a number (Double), and will use the following rules:
The version is given by: **"aa.bb.cc.dd"**, where **"aa.bb"** normally is the official version number and **"cc.dd"** is the
revision number, if given at all. If the second part is zero, it is ignored by this routine. If it is not zero, the second
number will be divided by 1000 and appended to the first number after stripping off the decimal point of the second
number. A version number like this,"3.10.002.5", will look like this when returned from this routine: **"3.100025"**
("3.10" & ".0025").

If the file does not contain any version information, the version number returned from this routine will be made up
using the file's date and time information: **"0.0000YYYYMMDDhhmmss".**

Example:
```
Result$ = FileGetVersion("VBIT.DLL")     ' Result will be "1.20" for this
                                         ' version
Result$ = FileGetVersion("C:\VB\VB.EXE")  ' Result: "3.000537"
Result$ = FileGetVersion("C:\CONFIG.SYS") ' Result: "0.000019950310120748"
                                         '(date/time info)
```

Function   IniFileGetString

Read data from an INI-file. Filename, section and a profile name is given and the function returns a string containing the profile string. If the profile name do not exist, the return value is an empty string. If the filename is given without any path, the system will start looking for the file in the Windows directory.

Usage:
```
Result$ = IniFileGetString(FileName$, Section$, Name$)
```

The section name must be given without brackets,

> **Wrong** => "[SectionName]"
> **Correct** => "SectionName"

Example:
```
StartProg$ = IniFileGetString("SYSTEM.INI", "boot", "shell")
' Returns perhaps "progman.exe"

String$ = IniFileGetString("WIN.INI", "MS user info", "DefName")
' Return information about the user from "WIN.INI"
```

See also: IniFilePutString

Function   IniFilePutString

Write data to an INI-file. Given the filename, section, name and the data to be written.
If the filename is given without any path, the system will start looking for the file in the Windows directory . The section name must be given without brackets. The function returns True(-1) if the call was successful, else False(0).

Usage:
```
Result% = IniFilePutString(FileName$, Section$, Name$, Data$)
```

Example:
```
Result%=IniFilePutString("MYPROG.INI", "Licence", "Name", "John Doe")
```

' Will write within the file "\WINDOWS\MYPROG.INI":
[Licence]
Name = John Doe

```
OK%=IniFilePutString("WIN.INI", "Desktop", "Wallpaper", "c:\pic\my.bmp")
```
' This statement will change the wallpaper, taking effect from the next startup of Windows.

See also : IniFileGetString