

# **VBIT.DLL**

**ver. 1.40**

---

## **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 complete 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 extension 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 VBITTAB.BAS in your projects**

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

---

## 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 extension 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 unavailable)

**Usage:**

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



---

## 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\_