SETUP.EXE 4.8a Standard Edition Help

'*The Original*' Alternative Setup Bootstrap Program

Contents

SETUP.EXE **Standard Edition** is **free** - you may use this program without charge when distributing your own applications. If you wish to pass on SETUP.EXE except as the bootstrap for an application (which we encourage) then you must make sure that all the original files are included.

Although SETUP.EXE was originally created for use with Microsoft's Visual Basic (to provide an advanced alternative to the SETUP.EXE supplied with that product) it can easily be used with other Windows development systems such as Borland's Delphi.

SETUP.EXE 4.8a Standard Edition Help

'*The Original*' Alternative Setup Bootstrap Program

Contents

SETUP.EXE **Standard Edition** is **free** - you may use this program without charge when distributing your own applications. If you wish to pass on SETUP.EXE except as the bootstrap for an application (which we encourage) then you must make sure that all the original files are included.

Although SETUP.EXE was originally created for use with Microsoft's Visual Basic (to provide an advanced alternative to the SETUP.EXE supplied with that product) it can easily be used with other Windows development systems such as Borland's Delphi.

SETUP.EXE 4.8a Standard Edition Help

'*The Original*' Alternative Setup Bootstrap Program

Contents

SETUP.EXE **Standard Edition** is **free** - you may use this program without charge when distributing your own applications. If you wish to pass on SETUP.EXE except as the bootstrap for an application (which we encourage) then you must make sure that all the original files are included.

Although SETUP.EXE was originally created for use with Microsoft's Visual Basic (to provide an advanced alternative to the SETUP.EXE supplied with that product) it can easily be used with other Windows development systems such as Borland's Delphi.

SETUP.EXE Distribution Files

See Also

The following files should have been included in the distribution archive you received SETUP.EXE **Standard Edition** in. If you pass on SETUP.EXE **except** as a bootstrap for your application then you must include all of these files in their original forms. They should always be stored (where possible) in a ZIP archive called **ASETUP.ZIP**, although this may be altered to include the version number if necessary, for example **ASETUP48.ZIP**.

- **SETUP.EXE** The bootstrap program.
- **SETUPSTD.HLP** This help file.
- **SETUP.INF** An example INF file.
- **SETUP.ICO** The icon used in SETUP.EXE.
- **COMPRESS.EXE** Microsoft File Compression Utility.
- **EXPAND.EXE** Microsoft File Expansion Utility.
- **README.TXT** General product and installation notes.
- **SOURCE.TXT** Details on purchasing the SETUP.EXE Standard source code.
- **VENDINFO.DIZ** Standard product information file for SETUP.EXE.
- **FILE_ID.DIZ** Standard BBS description file for SETUP.EXE.

For details on how to obtain the latest version of SETUP.EXE see the Product Support topic.

AlsoDist

See Also

Product Support and Contact Information

What is SETUP.EXE used for ?

SETUP.EXE is the program which the user runs to install your application (as in A:\SETUP). It is designed to pre-install specific files (such as DLLs and VBXs) which are used by a main Setup which performs the actual application installation. Once these files are installed SETUP.EXE will run the main Setup program, passing it any user command line parameters. It is sometimes referred to as a bootstrap program.

Your main Setup program (for example, normally called SETUP1.EXE for Microsoft Visual Basic developers) which provides the user with a nice friendly interface when installing your application (asks where to install the application on the user's hard disk, progress meters, configuration options, and such like) may require other files in order to run (depending on the language it has been written in). For example, if the main Setup program is written in Visual Basic 3.0 then it will also require VBRUN300.DLL and more than likely SETUPKIT.DLL plus any VBXs it may use.

- If your main Setup program is run off the floppy disk all these files have to be present on it or somewhere in the user's Path (e.g. hard disk). If these files are not available from the Path then they must be on the floppy and then if Windows is still running when the installation is finished and the user removes the floppy disk the next application to run that needs these files will assume they are still on that floppy drive and Windows will put up an error message even though the files are in the System directory.
- The other issue is that all these files used by the Setup process waste a lot of disk space (for example, the base files for a standard Visual Basic Setup program amount to some 450k at least) when stored in a usable form on the floppy disk. If they were compressed you could fit that much more of your application on it instead. Of course, if they are compressed you can't run your main Setup program using them.
- The third point is that the installation process will run a lot slower from the floppy disk instead of the hard disk. However, this won't be such an issue if you are installing the application from a Network drive.

Hence a bootstrap program is used to copy (and optionally decompress) the files required for the installation process to the user's hard disk.

Product Support and Contact Information

See Also

If you have any queries about using SETUP.EXE, or encounter problems which you believe may be software bugs, please contact us for assistance. Product support is free and unlimited.

Feedback and suggestions are very welcome - your contributions to SETUP.EXE's evolution will help ensure it meets your needs in the future.

The latest version of SETUP.EXE can always be found on:

- **CompuServe** in the **MSBASIC** forum or **VBPJFO** forum with a file name of **ASETUP.ZIP**
- **Internet** at **ftp.coast.net**, directory **/SimTel/win3/visbasic/** with a file name of **ASETUP48.ZIP** (new versions will have a different version number in the file name)
- **World Wide Web** (WWW) at **http://www.webzone1.co.uk/www/chapter1** or **http://194.159.104.7/www/chapter1**

The source code to SETUP.EXE Standard Edition is also available, as is an advanced SETUP.EXE Professional Edition.

Mike Chapman,
Chapter One Developments Ltd.
27 Gorse Drive,
Smallfield,
Surrey, RH6 9GJ,
England.

Fax: (44) 0 1342-844507

CompuServe: 100030,351
Internet: 100030.351@compuserve.com
WWW: http://www.webzone1.co.uk/www/chapter1

Association of Shareware Professionals (ASP) Ombudsman Statement

See Also

SETUP.EXE Distribution Files
Professional Edition
Source Code

Licence Agreement and Warranty Disclaimer

License Agreement

You should carefully read the following terms and conditions before using this software. Use of this software indicates your understanding and acceptance of the following terms and conditions. If you do not agree with them, do not use the software.

This program and the related documentation are copyright and protected by international copyright treaties and all other applicable national laws. The sole owner is Mike Chapman of Chapter One Developments Ltd.

Warranty Disclaimer

Users of SETUP.EXE must accept this disclaimer of warranty:

"*SETUP.EXE is supplied as is. The author disclaims all warranties, expressed or implied, including, without limitation, the warranties of merchant ability and of fitness for any purpose. The author assumes no liability for damages, direct or consequential, which may result from the use of SETUP.EXE*."

SETUP.EXE Professional Edition

In addition to this **Standard Edition** of SETUP.EXE which is **free** there is also an advanced **Professional Edition** which is available as Shareware. It has a registration fee of $10 when ordered through the CompuServe Registration Database (SWREG 7632) or $15 by other means (PsL Part #: 14230).

It is only approx. 27k in size and has all the features of this **Standard Edition** including the following: (as at Version 5.5)

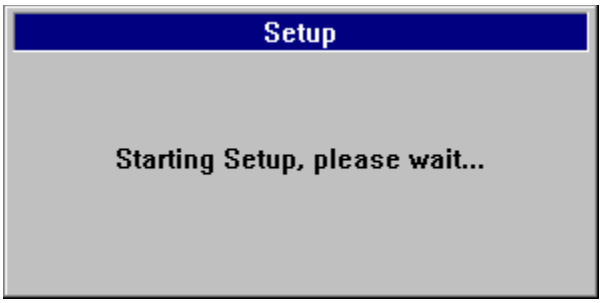
- It has optional 3d effects for the dialog box and message boxes,
- You can specify an icon or bitmap to be positioned anywhere in the dialog or a bitmap can replace the entire dialog in the style of a splash screen,
- It is **automatically** (and fully) compatible with both the Visual Basic 3.0 **and** 4.0 SetupWizard-generated SETUP.LST files - you do not have to manually specify which files SETUP.EXE copies allowing it to be completely compatible with the SetupWizards. Unless you wish to.
- You can specify where the Filen files will be installed on an individual basis (i.e. where the main Setup is installed, specifically the Windows System directory, or the default directory (the Temporary directory if used or the Windows System directory).
- It also supports multiple disks for the File*n* files, embedded version details (to speed the install), System Registry options, and (with the VB 4.0 SETUP.LST file) split files over disks.
- Files do not have to be compressed on the installation disk - if the file does not end in a _ then SETUP.EXE will look for a file of the same complete name and use that if it exists (useful for **CD-ROM setups** where the application can be run directly from the CD-ROM or installed using the same files).
- Any files can be checked to see if they already exist before the installation will proceed - for example, this is ideal for **Shareware authors** who might specify that their application requires VBRUN300.DLL but do not provide it to save space but still want a professional looking Setup. VBRUN300.DLL can be specified as required and a user would be notified if this does not exist so that the Visual Basic application will not fall over with a File Not Found error when run.
- The Setup can be specified to only run on certain versions of Windows by excluding the versions which are not supported by the product.
- All files except for SETUP.EXE and SETUP.INF can optionally reside in a different source directory, for example, in a CD-ROM setup.
- You can also specify a left position for dialog messages as well as a top position.
- There are more Setup Log details written for support purposes.
- The Professional Edition is distributed with it's own **Setup process** including a new-style Visual Basic main Setup program (SETUP1.EXE) to allow for easy installation of the product and to quickly demonstrate SETUP.EXE's use in a live situation.
-  Also included is a separate Editor application which is used to easily create and maintain the SETUP.INF files. Primarily provided to show a preview of the Setup dialog and error messages without having to keep running your full Setup and to allow easy inclusion of the embedded Windows version information from the bootstrap files for the Filen settings, it has evolved into a full-featured application supporting all 28 INF settings where you choose the options from lists instead of coding the numeric equivalents yourself.
- The Professional Edition SETUP.EXE is branded with your own licensing details and serial number in the version information.

The **source code** to the Professional Edition is also available, and for $20 when ordered through the **CompuServe Registration Database** (SWREG 7633) or $25 by other means (**PsL** Part #: 14230), you get the full C language source code to **both** the *Professional* and *Standard Editions* **and** a fully registered version of the Professional SETUP.EXE branded with your own licensing details and serial number. The Visual Basic source code to both the SETUP.INF Editor and the main Setup program (SETUP1.EXE) used to install the Professional Edition are also included free of charge.

The latest shareware version of SETUP.EXE Professional Edition can always be found on:

- **CompuServe** in the **MSBASIC** forum or **VBPJFO** forum with a file name of **PSETUP.ZIP**
- **Internet** at **ftp.coast.net**, directory **/SimTel/win3/visbasic/** with a file name of **PSETUP55.ZIP** (new versions will have a different version number in the file name)
- **World Wide Web** (WWW) at **http://www.webzone1.co.uk/www/chapter1** or **http://194.159.104.7/www/chapter1**

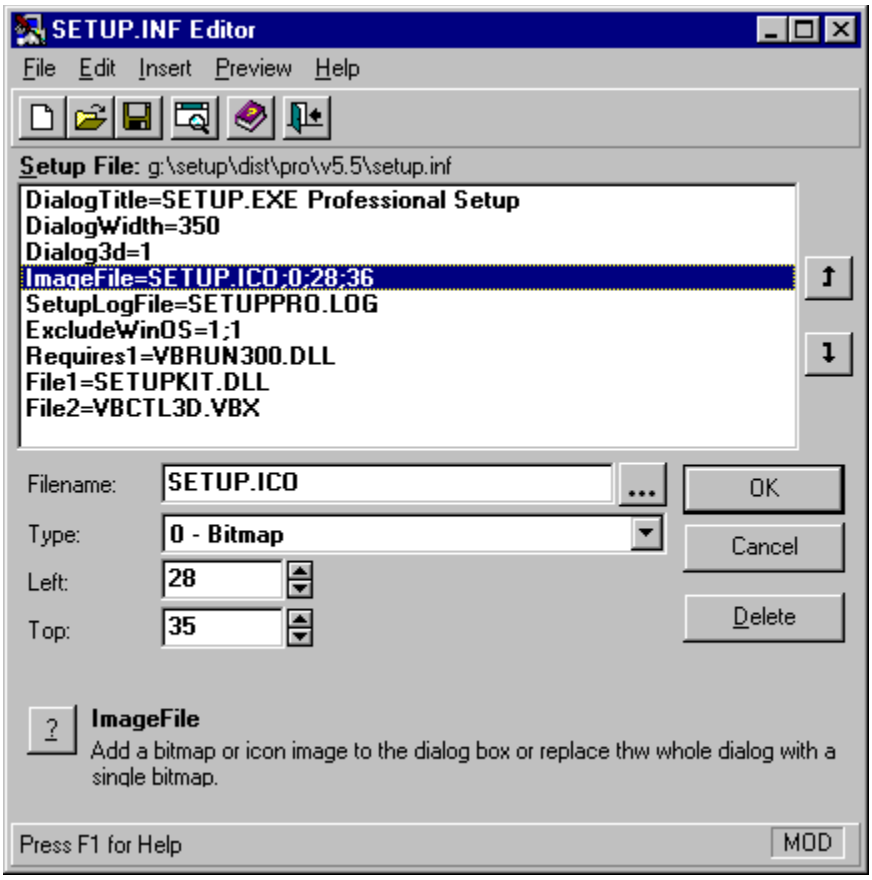
The only difference between the registered version of SETUP.EXE Professional and the shareware version is that the shareware version always displays an "(Unregistered Setup)" message in the dialog box.

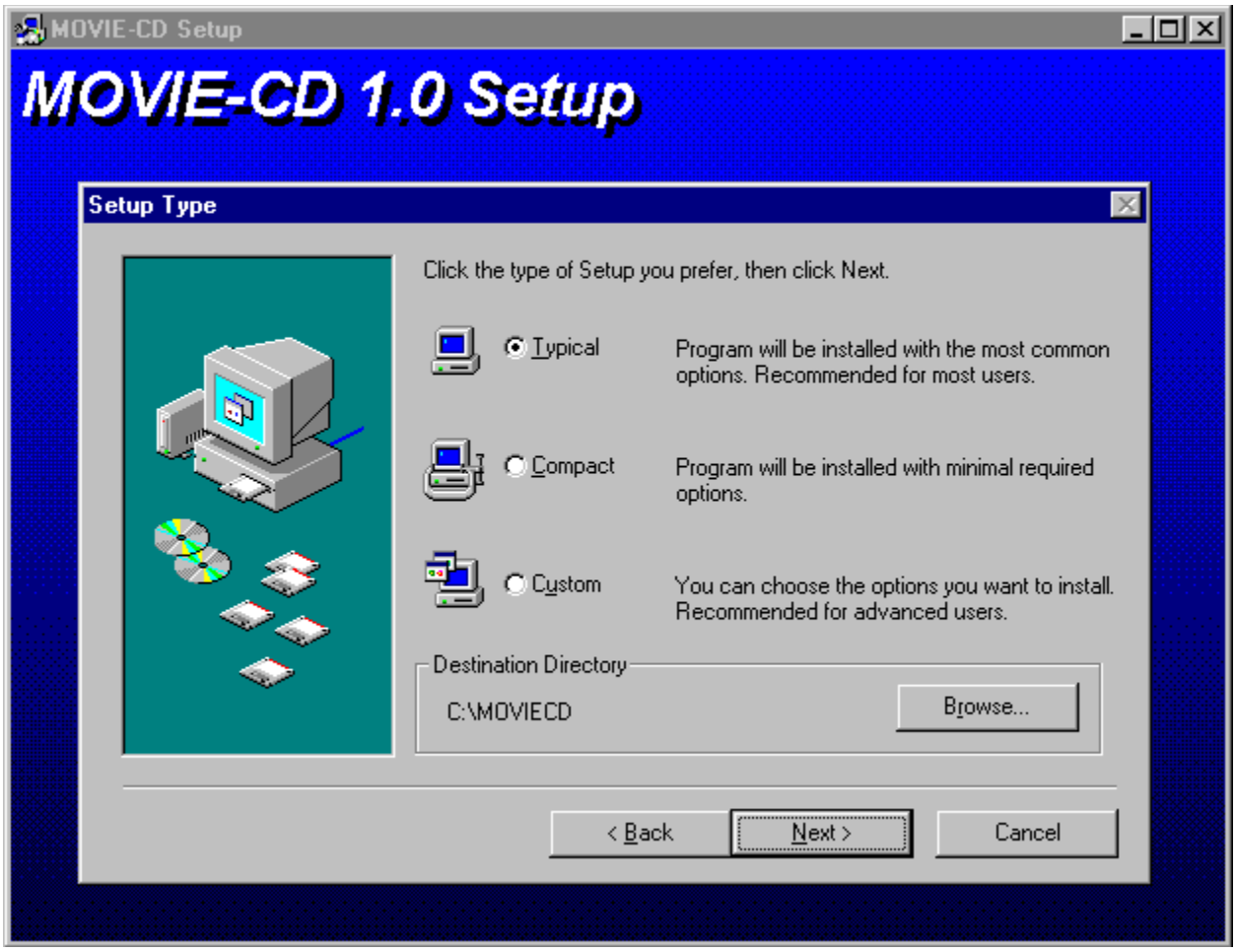

Setup
Starting Setup, please wait...


Setup Error
STOP
Unable to locate source file: A:\SETUP1.EX_
Please contact your distributer.
OK


Setup
Starting Setup, please wait...


MOVIE-CD
Version 1.0
Starting Setup,
please wait...
Copyright © 1995 Chapter One Developments Ltd.


SETUP.INF Editor
File Edit Insert Preview Help
Setup File: g:\setup\dist\pro\v5.5\setup.inf
DialogTitle=SETUP.EXE Professional Setup
DialogWidth=350
Dialog3d=1
ImageFile=SETUP.ICO,0,28,36
SetupLogFile=SETUPPRO.LOG
ExcludeWinOS=1,1
Requires1=VBRUN300.DLL
File1=SETUPKIT.DLL
File2=VBCTL3D.VBX
Filename: SETUP.ICO
Type: 0 - Bitmap
Left: 28
Top: 35
OK
Cancel
Delete
?
ImageFile
Add a bitmap or icon image to the dialog box or replace thw whole dialog with a single bitmap.
Press F1 for Help
MOD


MOVIE-CD Setup
MOVIE-CD 1.0 Setup
Setup Type
Click the type of Setup you prefer, then click Next.
Typical
Program will be installed with the most common options. Recommended for most users.
Compact
Program will be installed with minimal required options.
Custom
You can choose the options you want to install. Recommended for advanced users.
Destination Directory
C:\MOVIECD
Browse...
< Back
Next >
Cancel

Differences from the Microsoft Visual Basic 3.0-supplied version

See Also

A SETUP.EXE is provided with **Visual Basic 3.0** in the SETUPKIT\KITFILES subdirectory but it differs from this alternative SETUP.EXE Standard Edition in a number of significant areas.

There have been a number of updates to the Microsoft-supplied SETUP.EXE (the update can be obtained from Microsoft and downloaded from CompuServe) and the latest version that was used for this comparison was 1.00.004 and dated 23 Aug 1993.

Microsoft-supplied	SETUP.EXE 4.8a Standard
Requires that all Visual Basic programs running be closed before the Setup process can commence. Your users may not know which applications are Visual Basic or not and may not want or be able to close them down anyway - they may even be running a VB frontend ! This is also of no use if you wish to call the Setup process from a VB application.	No restriction imposed.
Does not pass command-line arguments to the Visual Basic Setup program.	SETUP.EXE will pass any command-line arguments to the Visual Basic Setup program SETUP1.EXE.
(No comparable features).	Developer-customisable Setup dialog box title, messages, size, background colour and border style.
(No comparable features).	Developer-customisable error messages so that, for example, you can inform the user of specific support procedures.
Will copy all files regardless of whether they are already on the user's hard disk although version checking is performed.	Provides *three* options for version checking. Default is Windows version information, another is by file dates, the last is to only install if it does not already exist.
Copies files to Windows and System directories only.	Can copy files to the Windows and System directories *or* can copy all the files (including the main Setup program) to a single temporary directory which is deleted after the main Setup program has closed. This directory can be developer-named or automatically generated.
(No comparable feature).	A Setup Log can be created of all the SETUP.EXE actions for support

	purposes. The log can have a developer-defined name.
Requires VER.DLL to be stored on the disk and in an uncompressed form or no installation will take place.	SETUP.EXE *does need* VER.DLL for it to install the files (in most cases) but all copies of Windows (except v3.0) provide it as standard so it is not mandatory for any version *other than* Windows 3.0. If you do require it then it can be compressed however.
Requires the more advanced version of LZEXPAND.DLL to be present on the user's hard disk. This file existed in an early form on Windows 3.0 machines and isn't compatible with this SETUP.EXE.	Works with the original Windows 3.0 LZEXPAND.DLL. This depends on the file version checking specified.
If a file cannot be installed (for example if it is in use) then the program halts for the user to abort the installation or retry.	Allows the user to make a *further* choice of skipping the file. The file can also be set to be skipped by the developer if it is in use.
Requires your Visual Basic Setup program SETUP1.EXE to be stored on the distribution disks with that name.	The Setup program can be stored with any name.
After installation the Visual Basic Setup program SETUP1.EXE is left in the user's Windows directory.	The main Setup program is automatically deleted by default after the installation.
Requires that any files that are compressed must be compressed with the **/r** switch to store the uncompressed file name.	Depends on the file version checking specified (but we recommend using **/r**).
The name of the INF file SETUP.LST is hard-coded.	The name of SETUP.EXE can be changed to something else (e.g. INSTALL.EXE) and the INF file will match this (e.g. INSTALL.INF).
The SETUP.LST file cannot contain anything other than a list of files to install.	The INF file uses standard INI file settings so that the main Visual Basic Setup program can use it for its own settings if required.
No icon.	Includes an icon within SETUP.EXE. This allows SETUP.EXE itself to be easily added to Program Manager without an icon having to be specified

	separately. (The icon can be modified using a resource compiler).
SETUP.EXE size: 19k	SETUP.EXE size: 17k

Differences from the Microsoft Visual Basic 4.0-supplied version

See Also

A SETUP.EXE is provided with **Visual Basic 4.0** in the SETUPKIT\KITFILES subdirectory but it differs from this alternative SETUP.EXE Standard Edition in a number of areas.

This Microsoft SETUP.EXE has been improved over the Microsoft Visual Basic 3.0 one. The latest version that was used for this comparison was the 16 bit 4.00.2422 and dated 15 Aug 1995.

Visual Basic 4.0 has a Setup Toolkit for **16** and **32 bit** installations. The following comparison is with the 16 bit version. The 32 bit version is only used when installing 32 bit applications on Windows 95 and NT and includes support for registering applications and DLLs and uninstalling the application. In 1Q96 a 32 bit version of SETUP.EXE will be released which will include these features.

Note: SETUP.EXE Professional Edition can **automatically** read SETUP.LST files created by the VB 4.0 (and 3.0) SetupWizards without you having to manually specify file settings.

Microsoft-supplied	**SETUP.EXE 4.8a Standard**
Does not pass command-line arguments to the Visual Basic Setup program.	SETUP.EXE will pass any command-line arguments to the Visual Basic Setup program SETUP1.EXE.
(No comparable features).	Developer-customisable Setup dialog box title, messages, size, background colour and border style.
(No comparable features).	Developer-customisable error messages so that, for example, you can inform the user of specific support procedures.
Performs standard Windows file version checking. If the VB 4.0 SetupWizard is used, it uses SETUP.LST-embedded version details to avoid reading distribution disk. If versions do not match exactly, will read from distribution disk to determine anyway.	Provides *three* options for version checking. Default is Windows version information, another is by file dates, the last is to only install if it does not already exist. Can be slower than VB 4.0 one if it uses the SETUP.LST-embedded version details. The Professional Edition supports embedded version details.
You can hard code a specific directory to copy certain files to but	Can copy files to the Windows and System directories *or* can copy all

it will not be deleted after the installation.	the files (including the main Setup program) to a single temporary directory which is deleted after the main Setup program has closed. This directory can be developer-named or automatically generated.
No comparable feature with the 16 bit version. The 32 bit version creates a log which is used by the uninstalling application.	A Setup Log can be created of all the SETUP.EXE actions for support purposes. The log can have a developer-defined name.
Sometimes announces that "*Setup has run into a conflict with a running application. Please close all applications...*". Currently unsure which files cause this.	SETUP.EXE will continue to install the application without having to close applications.
Supports multiple disks and System Registry file options.	Available only in the Professional Edition.
SETUP.EXE size: 30k	SETUP.EXE size: 17k

AlsoDiff4

See Also

Differences from the Microsoft Visual Basic 3.0-supplied version
Compatibility with the Visual Basic 4.0 SetupWizard
SETUP.EXE Professional Edition features
SETUP.EXE Standard Edition features

Compatibility with the Visual Basic 4.0 SetupWizard

See Also

SETUP.EXE **Standard Edition** is not automatically compatible with the **Visual Basic 4.0** SetupWizard. To make use of it, each time you create distribution disks you will have to:

1. Run the SetupWizard as normal and create a distribution disk.
2. Examine the SETUP.LST file on the distribution disk.
3. The first file File1 listed in the [BootStrap] section is the name of the main Setup program to be run. Code this file in the SETUP.INF SetupFileName entry.
4. All the other files listed are the pre-installation files. For each one code a Filen entry in the SETUP.INF.
5. Do **not** delete the SETUP.LST file from the distribution disk - the supplied main Setup program requires this - although if you have written your own it may not. This is automatically copied by SETUP.EXE Standard if it exists on the distribution disk.
6. Copy the SETUP.INF file to the distribution disk.

To get this version of SETUP.EXE to be automatically copied to the distribution disks change the **SWDEPEND.INI** file found in the Windows directory:

[SetupWiz]

BootStrap=*setup.exe location*

Where ***setup.exe location*** is the fully-qualified path and file name of SETUP.EXE. This change is only necessary once.

SETUP.EXE Professional Edition can **automatically** read the **SETUP.LST** created by the SetupWizard so all you need to do is copy the SETUP.INF file as required without manually specifying file settings.

AlsoComp4

See Also

Compatibility with the Visual Basic 3.0 SetupWizard
SETUP.INF file

Resource Complier

A **Resource Compiler** is used to modify Windows resources within an EXE or DLL such as strings, icons and version information without having to recompile the files. It is also used to create the resource source files to be compiled . Examples include Microsoft's ***AppStudio*****,** Borland's ***Resource Workshop***, or Symantec's ***Resource Toolkit***.

SETUP.LST

SETUP.LST is Microsoft's version of the SETUP.INF file. It lists all the files that Microsoft's SETUP.EXE should install. For Visual Basic 3.0 each line of the file is expected to be a file to be installed and so it cannot be used to store other settings which your main Setup program might use. In this case you would need another separate file. For Visual Basic 4.0 the main Setup program also uses this file for its settings and the file information is held in a similar manner to SETUP.EXE Standard with File*n* settings.

SETUP.EXE

SETUP.EXE is the program which the user runs to install your application (as in A:\SETUP). It is designed to pre-install specific files (such as DLLs and VBXs) which are used by a main Setup which performs the actual application installation. Once these files are installed SETUP.EXE will run the main Setup program, passing it any user command line parameters. It is sometimes referred to as a bootstrap program.

Microsoft File Compression Utility (COMPRESS.EXE)

Compress (**COMPRESS.EXE**) is the DOS program which the developer uses to compress files in size to save space when distributing a software product. The resulting files are typically 25 to 45 percent smaller than the original files. These files can then be decompressed using APIs built into Windows in the LZEXPAND.DLL and VER.DLL files. The Microsoft File Expansion Utility (EXPAND.EXE) can be used to restore files previously compressed by the Compress utility from the DOS command line.

Command-line syntax for Compress is as follows:

compress [/?][/r] ***source destination***

Following are command-line options and parameters for Compress:

/?	Displays information about how to use Compress.
/r	Specifies that compressed files should be renamed.
source	Specifies the source filename. The name can include a drive letter, a directory path, or both; and it can contain wildcards.
destination	Specifies the destination. This parameter can consist of a directory (with optional drive letter), a filename, or any combination of the two.

If the source parameter contains wildcards and the destination parameter does not specify only a directory, the **/r** option must be used.

If the destination parameter does not contain a filename, Compress uses the filename or filenames specified by the source parameter when Compress copies the file or files to the location specified by the destination parameter.

COMPRESS.EXE is included with Microsoft's Visual Basic but does not come with some other Windows development systems such as Borland's Delphi, hence its inclusion in this product.

Microsoft File Expansion Utility (EXPAND.EXE)

Expand (**EXPAND.EXE**) is the DOS program that can be used to decompress files previously compressed by Compress (COMPRESS.EXE). Expand restores these files to their original sizes. It is generally included as part of a Setup distribution disk to allow a user to manually decompress files in the event of a Setup failure.

Command-line syntax for Expand is as follows:

expand [/?][/r] source destination

Following are command-line options and parameters for Expand:

/?	Displays information about how to use Expand.
/r	Specifies that compressed files should be renamed.
source	Specifies the source filename. The name can include a drive letter, a directory path, or both; and it can contain wildcards.
destination	Specifies the destination. This parameter can consist of a directory (with optional drive letter), a filename, or any combination of the two.

If the source parameter contains wildcards and the destination parameter does not specify only a directory, the **/r** option must be used.

If the destination parameter does not contain a filename, Expand uses the filename or filenames specified by the source parameter when Expand copies the file or files to the location specified by the destination parameter.

The following example shows how to create decompressed versions of all the files on drive A, writing them to a directory on drive C:

```
expand a:*.* c:\mydir
```

EXPAND.EXE is included with Microsoft's Visual Basic but does not come with some other Windows development systems such as Borland's Delphi, hence its inclusion in this product.

SETUP.EXE Standard Edition Features

See Also

Setup Dialog Box Customisation Features

- dialog title,
- up to four horizontally centralised messages at specific vertical positions,
- three different dialog box styles,
- dialog height and width,

File Installation Features

- Provides standard Windows file version checking and also file date and time comparisons. If a file cannot be installed then, depending on the reason, the user has a choice of abort, retry or ignore. The developer can control whether the file is automatically skipped if it is in use and whether the user has the option to ignore files which fail to install.
- Any number of pre-install files may be specified.
- All pre-install files may be compressed with Microsoft's COMPRESS.EXE.
- Files can be installed to the Windows and System directories or to a temporary installation directory which is automatically created and then deleted after the main Setup program has closed. This directory can be developer-named or automatically generated.
- The main Setup program (for example, SETUP1.EXE) can be deleted after it had been used.
- Optionally store the main Setup program on the distribution disk with a developer-specified file name and also copy it to the user's Windows directory with a different name.

Miscellaneous

- Various developer-defined error messages by INF file settings.
- Optional Setup Log for support purposes which details actions carried out during Setup.
- SETUP.EXE can easily be renamed and the INF file will match the new name (e.g. INSTALL.EXE).
- The INF file can be used by other programs like the main Setup program since it uses the standard INI file layout.
- Passes all command-line arguments to main Setup program.
- When used with a Visual Basic 3.0 main Setup program it allows the installation to proceed with other Visual Basic programs running.
- If a SETUP.LST file exists on the distribution disk it will be automatically copied to where the main Setup program is to support Visual Basic 4.0 main Setup programs.
- When installing to a PC running a networked copy of Windows it will copy the pre-install files to the user's Windows directory instead of the networked System directory.
- All text is stored in string tables for easy localisation through translation.
- It is only approx. 17k in size.
- It has been successfully used on Windows 3.x, Windows for Workgroups 3.1x, Windows 95, and Windows NT 3.x.

AlsoFeatures

See Also

Differences from the Microsoft Visual Basic 3.0-supplied version
Compatibility with the Visual Basic 3.0 SetupWizard
Differences from the Microsoft Visual Basic 4.0-supplied version
Compatibility with the Visual Basic 4.0 SetupWizard
SETUP.EXE Professional Edition

Change History

Version 4.8a Standard 27 Jan 1996

Updated documentation with latest contact and ordering information, and reduced SETUP.EXE size.

Version 4.8 Standard 8 Jan 1996

Removed limit on the number of Filen files when using the Temporary directory.

Removed default text for the ErrorMessageInstallOther error message which specifically referred to which buttons to choose.

Fix: If SETUP.EXE is renamed to a full 8.3 file name like SETUPABC.EXE is would be unable to read the corresponding SETUPABC.INF file.

Fix: Retry button on the ErrorMessageInstallOther error message now works when using Setup Log.

Help file renamed to SETUPSTD.HLP to differentiate from the Professional Edition.

Version 4.7.1 Standard 11 Oct 1995

Fix: Resolved sharing problem introduced with 4.7 VB4 compatibility.

Version 4.7 Standard 20 Sep 1995

SETUP.EXE split into *Standard* and Professional editions.

Support for Visual Basic 4.0 - if a SETUP.LST file exists on the distribution disk it will be automatically copied to where the main Setup program is to support Visual Basic 4.0 main Setup programs.

Help file updated with Visual Basic 4.0 information.

Removed unnecessary FreeSpace setting reducing SETUP.EXE size.

Fix: if VER.DLL did not exist (on any version of Windows) a GPF would most likely occur. Now checks for VER.DLL regardless of which version of Windows and attempts install. If it fails it does not use version but date and time checking when installing files.

Fix: if source file did not exist but the destination file did then it would not note that there was an error and would skip the file.

Fix: resolved problem in some situations when a Temporary directory not used the main Setup program would not be able to read multiple disks. Thanks Scott Kalb.

Fix: fixed obscure problem starting main Setup program when running on Windows 3.10 without SHARE.EXE running. Thanks Michael Garlick.

Fix: can now delete the Temporary directory on NT - this was resolved by Michael Garlick's fix too.

Fix: if a Filen file name was 12 chars long and it was to be skipped if in use this would not happen.

Version 4.6 12 Aug 1995

Additionally detects if running on Windows NT or Windows 3.11 for Setup Log information and notes SETUP.EXE version number too.

ErrorMessageTempDir now has built-in default message.

When running on Windows 95 the background automatically uses the 3d colour.

Will run on 286 PCs now too.

Fix: Properly detects Windows 3.0 and does not require VER.DLL present in order to run.

Fix: Will now successfully delete the Temporary directory when running a multiple-disk setup.

Fix: If Windows dir is on any drive but C: now successfully deletes the Temporary directory.

Fix: If Windows dir is in the root directory of a drive the files will now be successfully installed. This fix also refers to all other paths used.

Updated documentation including help file.

World Wide Web and anonymous FTP sites available for support.

Version 4.5.1 21 Jun 1995

Dialog background colour defaults to system window backcolour instead of always white.

Writes DialogTitle to Setup Log.

Writes Windows & DOS versions to Setup Log.

Fix: Converts Windows dir to upper case to prevent problem with check for network copy when Windows was launched like "c:\windows\win".

Support for Windows 3.0 with Windows VER.DLL file version checking.

Version 4.5 12 Jun 1995

Option for a Setup Log with developer-defined file name.

Option to delete the main Setup program after it closes.

New style Setup icon.

Enhanced and updated help file.

Inclusion of COMPRESS.EXE and EXPAND.EXE for non-Visual Basic developers.

Source code can be purchased by methods other than through CompuServe.

Association of Shareware Professionals (ASP) accreditation.

Version 4.2.1 26 May 1995

Fix: when unable to launch main Setup it now closes SETUP.EXE.

Fix: when using a temp dir SETUP.EXE changes to that dir so that the files there are used before any in the path.

Version 4.2 9 Apr 1995

Temporary directory option for all files to be copied to.

Option not to provide a Skip button for file installation or free space problems.

Updated product so that it is does not appear to be Visual Basic-only.

Enhanced help file.

Version 4.1 30 Mar 1995

Fix: Resolved problem launching main Setup program on Windows NT.

Removed trailing space at end of installation path passed to main Setup program.

Enhanced launch error message.

Version 4.0 22 Mar 1995

Standard Windows file version checking implemented with fall-back to file dates (this is also a separate option). If the file is in use the developer can set it to be automatically skipped.

Standard fixed double border style with title bar implemented.

Command-line arguments now passed to main Visual Basic Setup program.

Default message, height and width modernised.

All text within the EXE (including references to the INI settings) have been moved to string resources which allow them to be easily translated into other languages for localised international versions. To modify these you can use a resource compiler.

The name of SETUP.EXE can be changed (for example to INSTALL.EXE) by just renaming it and the INF file name becomes the same (without the extension - e.g. INSTALL.INF).

New detailed custom error message for installation failures with user option to abort, retry or ignore.

If there is not enough space on the drive the user now has the option of abort, retry or ignore.

Overwrite setting for individual files removed (since proper version checking now provided).

Updated help file which no longer needs support DLL.

Source code made available.

Version 3.5 26 Apr 1994

INF file settings to specify up to 4 message lines which are automatically centred but have user-defined vertical positioning.

Settings to specify border style and back colour of the dialog.

Developer-defined error message text for different errors.

Setting to specify different file name for SETUP1.EXE on the distribution disk.

You can now force a file to be installed but it still doesn't perform any real version checking yet.

Version information for SETUP.EXE added along with an icon.

Improved help file with contents outline.

Minor changes to some message text.

- **Version 3.0.1 27 Oct 1993**
  Fix to allow it to run on Windows 3.0 machines.

  Fix to allow it to correctly calculate the free disk space when Windows is launched from DOS in the style of "c:\windows\win".
- **Version 3.0 28 Aug 1993**
  Use INF file for customisable Setup dialog box title, message and size. Pre-installs any number of files and checks for required disk space. Is thus VB version independent.
- **Version 2.0 5 Feb 1993**
  Centralise Setup dialog box.

  Change to copy VBRUN200.DLL instead of VBRUN100.DLL and also copy DDEML.DLL.
- **Version 1.01 1 Aug 1992**
  Pass install path to VB Setup program.
- **Version 1.0 15 Jul 1992**
  First public release.

Portions of this product were provided by Bob Feller and Microsoft.

How to use SETUP.EXE

1. Determine which DLLs and files your main Setup program (for example, SETUP1.EXE) requires to run.

   For main Setup programs written in **Visual Basic** prior to 4.0 this is likely to include VBRUNx00.DLL and SETUPKIT.DLL. It may also include some VBXs like THREED.VBX and if you have created one, your Setup help file. Visual Basic 4.0 Setups generally require substantially more files.

   For main Setup programs written in **Delphi** this will be the main Setup program, CTL3DV2.DLL (if it uses 3d effects) and any DLL or VBX files used (plus BIVBX11.DLL if VBX files are used).

2. Create your distribution disk. For Visual Basic developers this is as specified in the Visual Basic 3.0 and 4.0 on-line help (search on Setup Toolkit). Other versions of VB describe the Setup Toolkit elsewhere. Note its compatibility with the VB 3.0 and 4.0 SetupWizards. For Delphi developers it will depend on the main installation program you have written - what compression technique you have used. Check the DEPLOY.TXT file in the \DELPHI directory for details on what files are required for Delphi applications.

3. Code the required settings in the SETUP.INF text file.

4. Copy SETUP.EXE and SETUP.INF to your distribution disk (if the VB SetupWizard was used and modified then this will already have been carried out).

If your main Setup is **Windows 3.0** compatible and might be installed on a Windows 3.0 machine then you may need to check out the Windows 3.0 Special Requirements topic.

Note: The VB 4.0 main Setup can only be run on Windows 3.1 or higher.

Windows 3.0 Special Requirements

If your main Setup program (for example, SETUP1.EXE) and your application are **Windows 3.0** compatible (most applications are by default but quite a number of third party add-ons will only run on Windows 3.1 and above) and if there is the possibility that it might be installed onto a Windows 3.0 machine then you must include **VER.DLL** on your distribution disk. VER.DLL was not part of that Windows version and is required for proper Windows file version checking. This file can be compressed but regardless has to be named VER.DL_.

You do not need to have VER.DLL specified in a Filen setting since if SETUP.EXE detects a Windows 3.0 machine then it will force an install of this file in order to be able to install the other files.

DDEML.DLL will probably also be required since this and were not part of that version of Windows. This is required to perform DDE to the Windows Shell (such as Program Manager) in order to add icons. This should be specified in a **Filen** setting.

Compatibility with the Visual Basic 3.0 SetupWizard

See Also

SETUP.EXE **Standard Edition** is not automatically compatible with the **Visual Basic 3.0** SetupWizard. To make use of it, each time you create distribution disks you will have to:

1. Run the SetupWizard as normal and create a distribution disk.
2. Examine the SETUP.LST file on the distribution disk.
3. The first file listed is the name of the main Setup program to be run. Code this file in the SETUP.INF SetupFileName entry.
4. All the other files listed are the pre-installation files. For each one code a Filen entry in the SETUP.INF.
5. Delete the SETUP.LST file from the distribution disk.
6. Copy the SETUP.INF file to the distribution disk.

To get this version of SETUP.EXE to be automatically copied to the distribution disks change the **SETUPWIZ.INI** file found in the Windows directory:

[SETUPWIZ]

BOOTSTRAP=*setup.exe location*

Where ***setup.exe location*** is the fully-qualified path and file name of SETUP.EXE. This change is only necessary once.

SETUP.EXE Professional Edition can automatically read the **SETUP.LST** created by the SetupWizard so all you need to do is copy the SETUP.INF file as required without creating File*n* settings.

See Also

Compatibility with the Visual Basic 3.0 SetupWizard
Differences from the Microsoft Visual Basic 4.0-supplied version
SETUP.EXE Professional Edition features
SETUP.EXE Standard Edition features

See Also

Compatibility with the Visual Basic 4.0 SetupWizard
SETUP.INF file

SETUP.INF file

Example

SETUP.INF is a text file that lists all the files which SETUP.EXE pre-installs on the user's machine. It also contains various optional configuration settings for the initial Setup dialog box and installation features. *This file must appear in the same distribution directory/disk as SETUP.EXE*.

All the following SETUP.EXE settings appear under the **[BootSetup]** section heading:

DialogTitle=	SetupLogFile=
DialogMessagen=	DeleteSetup=
DialogWidth=	UseTempDir=
DialogHeight=	TempDir=
DialogColor=	ErrorMessageTempDir=
DialogStyle=	VersionCheck=
SetupFileName=	NoIgnore=
SetupDistName=	ErrorMessageInstall=
ErrorMessageLaunch=	ErrorMessageInstallOther=
SetupLog=	Filen=

The SETUP.INF file name was chosen because most commercial Setup programs seem to use this name as a standard. Since the settings in SETUP.INF are coded in a standard INI format the file can also be used by the main Setup program (for example, SETUP1.EXE) for its own configuration options if necessary.

In some countries **INSTALL.EXE** is the more usual name for the bootstrapper (and companies like Lotus and Borland tend to use this too). You can simply rename SETUP.EXE to INSTALL.EXE (or anything you like) and the name that SETUP.EXE expects the INF file to be will correspondingly be changed - e.g. INSTALL.INF. If you wish to change the INF file extension to something else see the topic on Localisation.

Example SETUP.INF file

See Also

```
[BootSetup]
DialogTitle=Movie Application Setup
DialogMessage1=Please wait...;20
DialogMessage2=Initialising Setup;40
DialogWidth=300
DialogHeight=150
DialogColor=0
DialogStyle=0
SetupFileName=MOVIESET.EXE
SetupDistName=MOVIESET.EX_
ErrorMessageLaunch=
SetupLog=1
SetupLogFile=MOVIESET.LOG
DeleteSetup=1
UseTempDir=1
TempDir=~MOVIESET
ErrorMessageTempDir=
VersionCheck=0
NoIgnore=1
ErrorMessageInstall=
ErrorMessageInstallOther=
File1=VBRUN300.DLL
File2=SETUPKIT.DLL
File3=THREED.VBX;1
```

See Also

SETUP.INF file

ErrorMessageInstallOther setting

See Also

If SETUP.EXE is unable to install a file using Windows version checking then the following style message will be displayed and the installation will pause for user intervention:

The reason given after the file can be one of the following (see the File Version Checking topic for more details):

- **Out of disk space on destination drive.**
- **File is in use.**
- **Drive is write-protected.**
- **General copying error.**

The user then has the option of aborting the Setup, skipping this file or re-attempting the copy after taking rectifying action.

By default no additional text such as the **Choose ABORT to cancel the Setup or IGNORE to skip this file** text shown in the above example is displayed but you can specify any text you wish as follows (for example, to include your own specific product support / instructions):

[BootSetup]

ErrorMessageInstallOther=*error message*

The ***error message*** can be any characters you like and up to 160 characters in length.

You can change the available buttons to **RETRY** and **CANCEL** by specifying the NoIgnore setting if you do not want the user to be able to skip the file.

See Also

SETUP.INF file
ErrorMessageLaunch setting
SetupLog setting
NoIgnore setting
ErrorMessageInstall setting
File Version Checking
Localisation

ErrorMessageLaunch setting

See Also

If SETUP.EXE is unable to run the main Setup program (for example, SETUP1.EXE) that is has just installed then the following message will be displayed and the installation pauses for user intervention:

The **Installation Aborted** text can be customised to include your own specific product support / instruction details as follows:

[BootSetup]

ErrorMessageLaunch=*error message*

The ***error message*** can be any characters you like and up to 160 characters in length. It will completely replace the Installation Aborted line.

Also provided for your support use is the DOS error number reason for the failure and the full path and file name of the file which was being launched. Theoretically, the DOS error number could be any of the following:

- **0** System was out of memory, executable file was corrupt, or relocations were invalid.
- **2** File was not found.
- **3** Path was not found.
- **5** Attempt was made to dynamically link to a task, or there was a sharing or network-protection error.
- **6** Library required separate data segments for each task.
- **8** There was insufficient memory to start the application.
- **10** Windows version was incorrect.
- **11** Executable file was invalid. Either it was not a Windows application or there was an error in the .EXE image.
- **12** Application was designed for a different operating system.
- **13** Application was designed for MS-DOS 4.0.
- **14** Type of executable file was unknown.
- **15** Attempt was made to load a real-mode application (developed for an earlier version of Windows).
- **16** Attempt was made to load a second instance of an executable file containing multiple data segments that were not marked read-only.
- **19** Attempt was made to load a compressed executable file. The file must be decompressed before it can be loaded.
- **20** Dynamic-link library (DLL) file was invalid. One of the DLLs required to run this application was corrupt.
- **21** Application requires Microsoft Windows 32-bit extensions.

AlsoErrorLaunch

See Also

- SETUP.INF file
- SetupFileName setting
- SetupDistName setting
- SetupLog setting
- ErrorMessageInstall setting
- ErrorMessageInstallOther setting
- Localisation

ErrorMessageInstall setting

See Also

If a file cannot be found on the distribution disk or SETUP.EXE itself cannot be installed then the following message will be displayed and the installation process will stop:

The **Installation Aborted** text can be customised to include your own specific product support / instruction details as follows:

[BootSetup]

ErrorMessageInstall=*error message*

The ***error message*** can be any characters you like and up to 160 characters in length. It will completely replace the Installation Aborted line.

AlsoErrorInstall

See Also

SETUP.INF file
SetupFileName setting
ErrorMessageLaunch setting
SetupLog setting
VersionCheck setting
Filen setting
ErrorMessageInstallOther setting
File Version Checking
Localisation

SetupDistName setting

See Also

The SetupDistName setting specifies the name of the main Setup program (for example, SETUP1.EXE) as it is stored on the distribution disk. It is an optional setting and is coded in the SETUP.INF file as follows:

[BootSetup]
SetupDistName=*name*

The ***name*** has to be exactly as the file is stored on the disk but with no path, e.g. MOVIESET.EX_. If this setting is not specified then SETUP1.EX_ will be used instead.

The file does not have to be compressed. If you do compress it then you must use the COMPRESS.EXE program supplied with Visual Basic (and included with this product for users of other Windows development systems like Borland's Delphi).

You do not have to specify an underscore (_) at the end of the extension but your file can still be compressed if you wish. SETUP.EXE will not care.

Note: If SETUP.EXE cannot find the above file on the distribution disk or it has a problem copying it then the following message will be displayed and the installation process will stop:

The **Installation Aborted** text can be customised using the ErrorMessageInstall setting.
When not using a temporary installation directory this file will be copied into the Windows directory even if another file of that name exists. In that case the existing one would be overwritten.

AlsoSetupDist

See Also

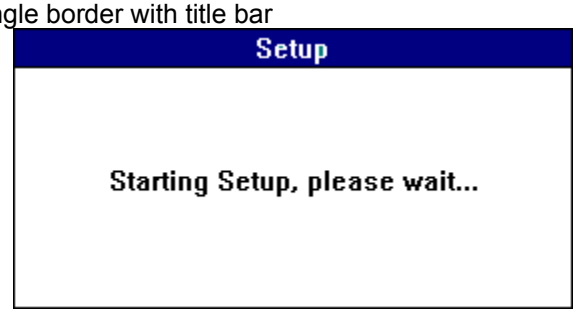
SETUP.INF file
SetupFileName setting
SetupLog setting
ErrorMessageInstall setting
Localisation

DialogStyle setting

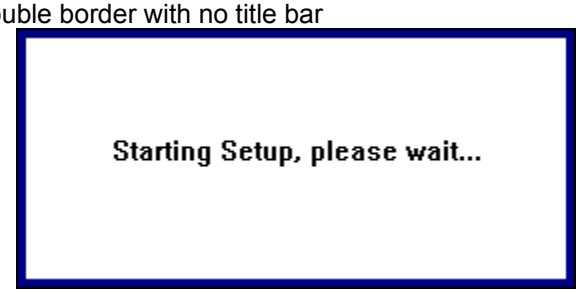
See Also

The Setup dialog box can have three different border styles:

- Double border with title bar
- Single border with title bar



- Double border with no title bar



[BootSetup]
DialogStyle=*value*

Where ***value*** is **0** for double border with title bar (which is the default), **1** for single border with title bar, or **2** for double border and no title bar.

You can customise the message displayed using the DialogMessagen setting.

AlsoDialogStyle

See Also

SETUP.INF file
DialogTitle setting
DialogColor setting
DialogMessagen setting

DialogColor setting

See Also

The Setup dialog box background colour can be partially customised.

It can be set to either use the windows default or forced to the 3d colour which is generally grey . [BootSetup]

DialogColor=*value*

Where ***value*** is **0** for the windows default (which is the default) or **1** for 3d.

Except when running on **Windows 95** the borders (DialogStyle setting) will not change to the 3d colour. To determine the 3d colour SETUP.EXE uses the current ButtonFace colour. When running on Windows 95 the DialogColor setting is ignored and the background is automatically set to the 3d colour.

AlsoDialogColor

See Also

SETUP.INF file
DialogStyle setting

DialogTitle setting

See Also

The title bar text of the Setup dialog box can be customised.

To specify the title bar text code the following setting in the SETUP.INF file:

[BootSetup]

DialogTitle=*text*

Text can be any characters you like and up to 80 characters in length. If no DialogTitle setting is specified then the default text used will be **Setup**.

You may find you need to increase the width of the dialog box using the DialogWidth setting from its default if you want a particularly long title. You can skip display of a title by changing the DialogStyle setting.

See Also

SETUP.INF file
DialogWidth setting
DialogStyle setting

DialogMessagen setting

See Also

You can specify up to four different lines of text to be displayed in the Setup dialog box.

To specify the message text code the following settings in the SETUP.INF file:
[BootSetup]

DialogMessage*n*=*text* ; *position*

DialogMessage*n*=*text* ; *position*

etc...

Where ***n*** is the number of the **DialogMessage*n*** setting, e.g. DialogMessage1, then DialogMessage2, etc. up to a maximum of 4. ***Text*** can be any characters you like and up to 60 characters in length. ***Position*** is the vertical position that Text will start at within the dialog box and is measured in pixels.

Text and ***Position*** must be separated by a semi-colon (;) and that character cannot then be used in the actual message.

If no DialogMessage*n* settings are specified then the default text used will be **Starting Setup, please wait...** at a vertical position of **45** pixels.

The messages are horizontally centralised and you may find you need to increase the width of the dialog box using the DialogWidth setting from its default if you want a particularly long message.

DialogWidth setting

See Also

The width of the Setup dialog box can be customised.

To specify a width code the following setting in the SETUP.INF file:

[BootSetup]
DialogWidth=*width*

The ***width*** is measured in pixels. If no DialogWidth setting is specified then the default width is **300** pixels.

See Also

SETUP.INF file

DialogHeight setting

DialogStyle setting

See Also

SETUP.INF file
DialogWidth setting
DialogStyle setting

DialogHeight setting

See Also

The height of the Setup dialog box can be customised.

To specify a height code the following setting in the SETUP.INF file:

[BootSetup]

DialogHeight=*height*

The ***height*** is measured in pixels. If no DialogHeight setting is specified then the default height is **150** pixels.

SetupFileName setting

See Also

The SetupFileName setting specifies the name to rename the main Setup program (for example, SETUP1.EXE) to when it is copied to either the user's Windows directory or the temporary installation directory. It is an optional setting and is coded in the SETUP.INF file as follows:

[BootSetup]

SetupFileName=*name*

The ***name*** has to have the file extension specified but no path, e.g. MOVIESET.EXE. If this setting is not specified then SETUP1.EXE will be used instead.

SETUP1.EXE must be stored on the distribution disk as SETUP1.EX_ (an underscore as the last character of the extension) **unless** you specify a different name in the SetupDistName setting.

Note: If SETUP.EXE cannot find SETUP1.EX_ (or the **SetupDistName** specified one) on the distribution disk or it has a problem copying it then the following message will be displayed and the installation process will stop:

The **Installation Aborted** text can be customised using the ErrorMessageInstall setting.
When not using a temporary installation directory this file will be copied into the Windows directory even if another file of that name exists. In that case the existing one would be overwritten.

See Also

- SETUP.INF file
- SetupDistName setting
- SetupLog setting
- ErrorMessageInstall setting
- Filen setting
- File Version Checking
- Localisation

SETUP1.EXE

For Visual Basic developers **SETUP1.EXE** is the default name of the Visual Basic Setup program which actually performs the task of installing your application onto the user's machine. Other development environments may refer to this differently. This file is copied to the user's machine by SETUP.EXE. It does not have to be copied as SETUP1.EXE or stored on the distribution disk with that name either; specify the SetupFileName and SetupDistName settings as required.

Filen setting

See Also

The File*n* setting specifies the name of a pre-install file to copy to the user's hard disk before the main Setup program (for example, SETUP1.EXE) is run. Any DLLs or VBXs which your Setup program uses must be specified (you may also have a help file you've created for the Setup too). You can have any number of these settings and they are coded in the SETUP.INF file as follows:

[BootSetup]
File*n*=*name*
File*n*=*name*
etc...

Where ***n*** is the number of the **File*n*** setting, e.g. File1, then File2, File3, etc. The ***name*** has to have the file extension specified but no path, e.g. VBRUN300.DLL.

You can specify a further argument on each **File*n*** setting to indicate that if this file is in use then the attempt to copy should be skipped. See the file version checking topic for more details.

[BootSetup]
Filen=name ; *skipinuse*

Where **skipinuse** is **0** if the user should be notified if a file that should be installed (because it is newer) cannot because it is in use (this is the default) or **1** if the copy and notification of this file should be skipped.

The actual file must be stored on the distribution disk with the last character of the extension as an underscore (_), e.g. VBRUN300.DL_. Remember that the **File*n*** setting has to have the full name (without the underscore).

Note: The file does not have to be compressed. If you do not compress the file it must still have the last character of the extension as an underscore (_).

If you do compress the files then you must use the COMPRESS.EXE program (supplied with Visual Basic and included with this product for users of other Windows development systems such as Borland's Delphi).

Note: If your app could be installed on a Windows 3.0 PC then you will need to distribute **VER.DLL** but you **do not** specify a **File*n*** setting for it - if SETUP.EXE detects it is running on Windows 3.0 then it will install it automatically. See the Windows 3.0 Special Requirements topic for more details.

If a file is specified which SETUP.EXE cannot find it on the distribution disk or has a problem copying it then the following message will be displayed noting the problem file and the installation process will stop:

The **Installation Aborted** text can be customised using the ErrorMessageInstall setting.

See Also

- SETUP.INF file
- SetupFileName setting
- SetupLog setting
- VersionCheck setting
- ErrorMessageInstallOther setting
- File Version Checking
- Localisation

Localisation

What do you do when you want to distribute a non-English language version of your product?

SETUP.EXE allows you to specify your own error message details but you also need to translate all the other text references too.

To allow you to modify these all of the text references are stored within the SETUP.EXE program in string tables. You can easily translate (or just change) them to anything you like using a resource compiler or editor.

Also stored in string tables are all the INF file setting names so these can also be modified.

In some countries **INSTALL.EXE** is the more usual name for the bootstrapper (and companies like Lotus and Borland tend to use this too). You can simply rename SETUP.EXE to INSTALL.EXE (or anything you like) and the name that SETUP.EXE expects the INF file to be will correspondingly be changed - e.g. INSTALL.INF. If you wish to change the INF file extension to something else then you will need to change the string table resource **INI_INF**.

String Table Details:

Resource Id	String Details	Max Length	Description
DEF_TITLE	"Setup"	60	*The default title of the Setup dialog box. See the* DialogTitle *setting.*
DEF_MSG	"Starting Setup, please wait...;45"	60	*The default message displayed in the Setup dialog box. See the* DialogMessagen *setting.*
DEF_DIST	"SETUP1.EX_"	12	*The default distribution name for the main Setup program. See the* SetupDistName *setting.*
DEF_FILE	"SETUP1.EXE"	12	*The default name for the main Setup program to be installed as. See the* SetupFileName *setting.*
DEF_LOGFILE	"SETUPEXE.INI"	12	*The default name for the Setup Log file. See the* SetupLogFile *setting.*
DEF_LST	"SETUP.LST"	35	*The default name for the VB 4 SETUP.LST file which is automatically copied to where the main Setup is if it exists.*
ERR_SETUP	"Setup Error"	50	*The title used in error message dialog boxes.*
ERR_ABORT	"Installation Aborted."	60	*Used in the* ErrorMessageInstall,ErrorMessageLaunch *and* ErrorMessageFreeSpace *messages.*
ERR_LOCATE	, "Unable to locate source file: "	60	*Used in the* ErrorMessageInstall *message.*
ERR_LAUNCH	"Error launching Setup file: "	60	*Used in the* ErrorMessageLaunch *message.*
ERR_INSTALL	"Unable to install file: "	60	*Used in the* ErrorMessageInstallOther *message.*
ERR_INUSE	"This file is in use. Please close down the application	200	*Used in the* ErrorMessageInstallOther *message.*

	which is using this file and choose RETRY."		
ERR_OUTOFDISK	"Out of disk space on the destination drive. Please free up some space and choose RETRY."	200	*As above.*
ERR_TO	" to"	15	*As above.*
ERR_WRITEPROT	"The drive is write-protected."	100	*Used in the* ErrorMessageInstall *and* ErrorMessageInstallOther *messages.*
ERR_GENERAL	"General copy error: "	60	*Used in the* ErrorMessageInstallOther *message.*
ERR_ERROR	"Error: "	15	*Used in the* ErrorMessageLaunch *message.*
ERR_DELTEMP	"Unable to delete the temporary directory:"	100	*Used in the* ErrorMessageTempDir *setting message.*
ERR_DELMAN	"Please delete this directory manually."	100	*Used in the* ErrorMessageTempDir *setting message.*
INI_APPTITLE	"BootSetup"	35	*The INI section heading used for all the settings in the* INF file*.*
INI_DLGTITLE	"DialogTitle"	35	*The name of the* DialogTitle *setting.*
INI_DLGMSG	"DialogMessage"	35	*The main part of the* DialogMessagen *setting.*
INI_DLGWIDTH	"DialogWidth"	35	*The name of the* DialogWidth *setting.*
INI_DLGHEIGHT	"DialogHeight"	35	*The name of the* DialogHeight *setting.*
INI_DLGCOLOR	"DialogColor"	35	*The name of the* DialogColor *setting.*
INI_DLGSTYLE	"DialogStyle"	35	*The name of the* DialogStyle *setting.*
INI_FILENAME	"SetupFileName"	35	*The name of the* SetupFileName *setting.*
INI_DISTNAME	"SetupDistName"	35	*The name of the* SetupDistName *setting.*
INI_ERRLAUNCH	"ErrorMessageLaunch"	35	*The name of the* ErrorMessageLaunch *setting.*
INI_SETUPLOG	"SetupLog"	35	*The name of the* SetupLog *setting.*
INI_LOGFILE	"SetupLogFile"	35	*The name of the* SetupLogFile *setting.*
INI_DELSETUP	"DeleteSetup"	35	*The name of the* DeleteSetup *setting.*
INI_USETEMPDIR	"UseTempDir"	35	*The name of the* UseTempDir *setting.*
INI_TEMPDIR	"TempDir"	35	*The name of the* TempDir *setting.*
INI_ERRTEMPDIR	"ErrorMessageTempDir"	35	*The name of the* ErrorMessageTempDir *setting.*
INI_VERSIONCHK	"VersionCheck"	35	*The name of the* VersionCheck *setting.*
INI_NOIGNORE	"NoIgnore"	35	*The name of the* NoIgnore *setting.*
INI_ERRINSTALL	"ErrorMessageInstall"	35	*The name of the*

			ErrorMessageInstall *setting.*
INI_ERRINSTALL2	"ErrorMessageInstallOther"	35	*The name of the* ErrorMessageInstallOther *setting.*
INI_FILE	"File"	35	*The main part of the* Filen *setting.*
INI_INF	".INF"	4	*The file extension used for the* INF file*.*

You can code **\t** to imbed a tab character and **\n** to perform a new line.

File Version Checking

SETUP.EXE uses file version checking to ensure that only newer files are installed on the user's machine, thereby avoiding file conflicts if an older version were installed.

Three methods are available (which are described in the VersionCheck setting topic) but the main one (and the default and recommended method) is using Windows built-in version checking.

The vast majority of Windows files (EXE, DLL, VBX) have standard version information imbedded in them. This includes a file version number in the form **n.n.n.n**.

SETUP.EXE formats each element of this version number to 4 hex digits which prevents a standard comparison failure when comparing files with versions like **3.10.4** and **3.8.3**. In this case the second file would be treated as newer although we can see this isn't the case. With SETUP.EXE's formatting these would become **0003.0010.0004.0000** and **0003.0008.0003.0000** and the comparison would correctly show the first as the newest.

The following table shows the installation action SETUP.EXE would automatically make given the circumstances when files do and do not have imbedded version information:

Source	Destination	Install ?
Version info	Version info	If source file is newer
Version info	No version info	YES
No version info (Use file date/time)	No version info (Use file date/time)	If source file is newer
No version info	Version info	NO
(Anything)	File does not exist	YES

The only downside to using Windows file version checking is it does slow down the installation from floppy disks on some machines. This occurs because Windows scans the file for version information and it is especially slow when large files are compressed. One way around this would be to use the temporary install directory option. Since no files already exist there SETUP.EXE does not attempt to determine version information from the distribution files and so avoids the speed hit.

If SETUP.EXE is unable to install a file using Windows version checking then the following style message will be displayed and the installation will pause for user intervention:

The reason given after the file can be one of the following:

- **Out of disk space on destination drive.**
- **File is in use.** For example, a DLL that *needs* to be installed because it is newer cannot because an application that uses that DLL is currently running. To allow the file to be copied that application must be closed. *See below for more information.*
- **Drive is write-protected.** If the user does not have write access to the installation directory / drive then this message will be displayed. The administrator needs to give the user the appropriate access rights.
- **General copying error.** This could happen for a number of reasons and the following **VIF_** errors detail what the problem is:

  VIF_WRITEPROT, VIF_SHARINGVIOLATION, VIF_OUTOFMEMORY, VIF_CANNOTREADDST.

The user then has the option of aborting the Setup, skipping this file or re-attempting the copy after taking rectifying action.

File In Use Error

By default if a file is in use which needs to be installed because it is newer then the above error message will displayed. For some files this might be unnecessary, for example VER.DLL. This file exists as part of Windows 3.1 and 3.11 (and later versions of Windows). Your installation might be installing the latest copy 3.11 and when attempting to install this on a 3.1 machine would notify the user that it cannot because it's in use. Since this is in use by Windows itself all the user can really do is abort the Setup or ignore this file. Possibly somewhat confusing for them.

We could safely ignore this for this particular file so using the Filen setting we can specify that in the above situation the VER.DLL file should automatically be skipped.

Distribution File Does Not Exist Error

If a file does not exist on the distribution disk then the following style message is displayed and the installation will stop.

VersionCheck setting

See Also

There are three different types of file version checking available:

- **Windows file version checking**
  The vast majority of Windows files (EXE, DLL, VBX, OCX) have standard version information imbedded in them. This includes a file version number in the form **n.n.n.n**.

  In this check SETUP.EXE compares the file version of the file to be installed with the one that may already exist and if the one to be installed is newer the existing one is overwritten.

  It handles this intelligently and for more information see the File Version Checking topic. This is the default method and is highly recommended.
- **Based on the file date and time stamps**
  This check simply compares the file dates of the source and destination and will only install if the file to be installed is newer.

  (This is also the fall-back method for the above method if no version information is present in both files).
- **Install only if the file doesn't currently exist**
  This was the original method the SETUP.EXE used. If the destination files does not exist then the file will be installed. No version checking of any kind is carried out.

[BootSetup]

VersionCheck=*value*

Where ***value*** is **0** for Windows version checking (which is the default), **1** for date and time stamps, or **2** for only if the file doesn't exist.

See Also

SETUP.INF file
SetupLog setting
File Version Checking

Source Code

The source code to SETUP.EXE Standard Edition may be purchased for personal and in-house use only. You may not distribute a competitive Setup bootstrapper. SETUP.EXE is written in C and has been used with both Microsoft and Borland compilers.

You can order the source code through the **CompuServe Registration Database**, by Credit Card (phone, fax, mail, CompuServe email, Internet WWW) or by Cheque (mail).

NOTE: When upgrades of SETUP.EXE are released you will be emailed the source to these too at no extra charge.

To register the source code through the **CompuServe Registration Database** only costs **$10** (compared to $15 by other methods).

```
GO SWREG
```

Select "**Register**", then select "**Registration ID**" and quote Program Id **5171.**

As soon as CompuServe notifies us of your registration (via email) we will email you the latest version of SETUP.EXE along with all the source files.

You can also register by *Credit Card* through the **Internet World Wide Web** at Chapter One Development's SETUP.EXE Standard page:

`http://www.webzone1.co.uk/www/chapter1/asetup.htm`

UseTempDir setting

See Also

By default SETUP.EXE will copy the main Setup program (for example, SETUP1.EXE) to the user's Windows directory and all the files specified by the Filen settings to the user's System directory (unless the System directory is a shared Network directory in which case all the files are copied to the Windows directory).

The UseTempDir setting provides the option to copy all the files (including the main Setup program) to an automatically created temporary installation directory instead. Once the main Setup program has closed this directory (and all the files which were copied to it) is deleted. It is coded in the SETUP.INF file as follows:

[BootSetup]
UseTempDir=*value*

Where ***value*** is **0** for no temporary directory (the default) or **1** to use a temporary directory.

The directory is created off the user's **C drive** and by default begins with a **~S** followed by six numerics although this name can be developer-specified by coding the TempDir setting.

Using a temporary installation directory avoids the speed hit when using Windows file version checking which occurs when installing from floppy disks on some machines, especially when large files are compressed. Since no files already exist in the directory SETUP.EXE does not attempt to determine version information from the distribution files and so avoids the speed hit.

See Also

SETUP.INF file
SetupLog setting
TempDir setting
ErrorMessageTempDir setting

TempDir setting

See Also

When copying files to a temporary installation directory (specified with the UseTempDir setting) the default directory created off the user's **C Drive** begins with a **~S** followed by six numerics. This name can be changed by coding the following setting in the SETUP.INF file:

[BootSetup]
TempDir=*dir*

Where ***dir*** is the name of the directory to be created for copying all the files to. This can only be a single level and without any drive specification, e.g. **MOVIESET**. It should follow the usual DOS directory naming convention.

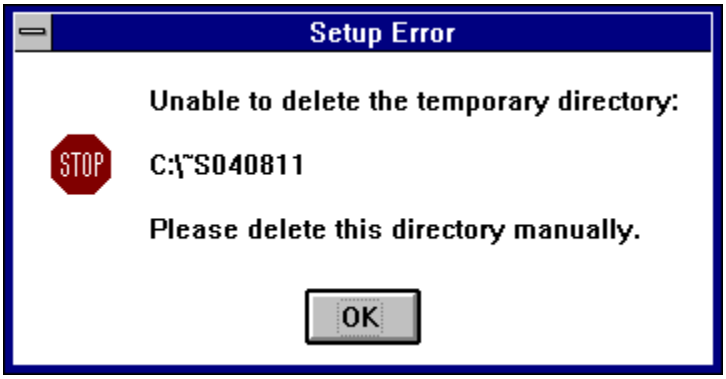
See Also

SETUP.INF file
SetupLog setting
UseTempDir setting
ErrorMessageTempDir setting

ErrorMessageTempDir setting

See Also

If you have specified that you want to use a temporary directory for all the files (using the UseTempDir setting) and the temporary directory cannot be deleted after the main Setup program (for example, SETUP1.EXE) has finished then the following message will be displayed:



The **Please delete this directory manually** text can be customised to include your own specific product support / instruction details as follows:

[BootSetup]

ErrorMessageTempDir=*error message*

The ***error message*** can be any characters you like and up to 100 characters in length. It will be added after the directory line.

See Also

SETUP.INF file
SetupLog setting
UseTempDir setting
TempDir setting
Localisation

NoIgnore setting

See Also

The NoIgnore setting is used to determine whether or not the **Ignore** button is available on the message box when unable to install a file. It is an optional setting and is coded in the SETUP.INF file as follows:

[BootSetup]

NoIgnore=*value*

Where ***value*** is **0** (which is the default) for an **Ignore** button:

and **1** for no **Ignore** button:

When no Ignore button is provided the **Cancel** button performs the same function as the **Abort** button by halting the installation at this point.

See Also

SETUP.INF file
SetupLog setting
ErrorMessageInstallOther setting

SetupLog setting

See Also Example

A Setup Log can be created for support purposes in order to help solve obscure problems. All the actions SETUP.EXE makes during an installation are recorded in it. By default this is always created (and this is recommended).

[BootSetup]

SetupLog=*value*

Where ***value*** is **0** for no Setup Log or **1** (which is the default) to create the log.

The log is always created in the Windows directory. The default name is **SETUPEXE.INI** although this can be changed using the SetupLogFile setting.

The contents of the created log are structured like an INI file. Each time SETUP.EXE is run a new section is created (the last one in the file being the most recent). See the Setup Log Contents topic for complete details.

See Also

SETUP.INF file
SetupLogFile setting
Setup Log Contents

Setup Log Contents

See Also Example

A Setup Log can be created by SETUP.EXE using the SetupLog INF file setting. The log can be used for support purposes in order to help solve obscure problems. All the actions SETUP.EXE makes during an installation are recorded in it.

The contents of the created log are structured like an INI file. Each time SETUP.EXE is run a new section is created (the last one in the file being the most recent).

An example section heading follows:

```
[1995/06/10 16:36:09]
```

The section heading contains the date and time that the Setup was carried out in the format **yyyy/mm/dd hh:mm:ss**.

The next line displays the DialogTitle text.

```
A=Movie Application Setup
```

The next line displays the Windows version number and the DOS version number in that sequence following by the version number of SETUP.EXE in brackets and whether it is the Standard or Professional version.

```
V=3.11 6.22 (4.8.1.0 Std)
```

If Setup is running on Windows NT then the Windows and DOS versions are replaced with `NT`.

The following four lines specify various directory names:

```
1=C:\WINDOWS
2=C:\WINDOWS\SYSTEM\
3=C:\~S061434\
4=A:\
```

Where the numbers indicate the following directories:

1. Windows directory.
2. Windows System directory.
3. The directory where the files defined by the Filen settings are installed.
4. The distribution directory from where SETUP.EXE is run from. This is passed to the main Setup program.

The next set of lines detail the files that were installed and the results. The main Setup program is given first with the fully qualified path (unless this was an installation to a temporary directory this should be the Windows directory).

```
C:\~S061434\SETUP1.EXE=101
```

The value is the installation result and can be one of the following:

101 File copied regardless (with no version checking carried out).

400 File not installed. Unable to locate the distribution file.

401 File not installed. Destination is probably write-protected or does not exist.

Then next lines detail the Filen setting files and their results.

```
SETUPKIT.DLL=100
VBRUN300.DLL=100
```

These files are being installed in the Windows System directory (detailed by the **2=** line) and the installation method depends on the VersionCheck setting. The result value can be one of the following:

100 File copied. File version checking was used.

101 File copied regardless (no version checking carried out).

102 File copied. File did not exist already (when VersionCheck setting is 2).

200 File not installed. File versions match or file to be installed is an older version.

201 File not installed. Destination file contains version info but file to be installed does not so it isn't installed.

202 File not installed. No version info exists in either destination or file to be installed so file date comparisons were made and the dates are the same or the file to be installed is older.

210 File not installed. Destination file was in use.

211 File not installed. Out of disk space.

212 File not installed. Drive/Path write-protected or file being overwritten read-only.

300 File not installed. Destination file was in use and the **SkipInUse** flag on the Filen setting indicated it could be skipped.

400 File not installed. Unable to locate the distribution file. The full installation file path is provided.

401 File not installed. Destination is probably write-protected or does not exist. The full installation file path is provided.

501 File not installed. File date comparisons used and the dates are the same or the file to be installed is older (when VersionCheck setting is 1).

502 File not installed. File already exists and it is not to be overwritten (when VersionCheck setting is 2).

Theoretically the following **VIF_** errors could also be specified as the result value (the words not their actual hex values):

VIF_WRITEPROT, VIF_SHARINGVIOLATION, VIF_OUTOFMEMORY, VIF_CANNOTREADDST

If the main Setup program is unable to be launched then the following two lines are created:

```
L=2
CL=C:\~S061434\SETUP1.EXE A:\ -r
```

The first line details the DOS error code for the launch (see the ErrorMessageLaunch setting for the available values). The second line provides the full command line as it was used for the launch - the full path to the main Setup program, the installation path, followed by an user-entered arguments to SETUP.EXE.

When required to delete the main Setup program after the installation (the DeleteSetup setting) or when using a temporary directory (the UseTempDir setting) which must be deleted after the installation, a Windows timer is initiated. If this cannot be created then the following line is added:

```
T=0
```

The following line is created if the main Setup program is deleted:

```
DS=1
```

The next line determines the result of deleting the temporary directory if one was created.

```
DT=1
```

The value is **1** if the directory was deleted or **0** if the delete failed..

If the installation was successful then the following line is added at the very end:

```
C=1
```

See Also

SETUP.INF file
SetupLog setting
SetupLogFile setting

Example

```
[1995/06/11 11:06:02]
A=Movie Application Setup
V=3.11 6.22 (4.8.1.0 Std)
1=C:\WINDOWS
2=C:\WINDOWS\SYSTEM\
3=C:\~S061506\
4=A:\
C:\~S061506\SETUP1.EXE=101
THREED.VBX=100
SETUPKIT.DLL=100
VBRUN300.DLL=100
DS=1
DT=1
C=1
```

SetupLogFile setting

See Also

When creating a Setup Log to record the actions of SETUP.EXE during an installation (specified with the SetupLog setting) the default log file name is **SETUPEXE.INI**. This name can be changed by coding the following setting in the SETUP.INF file:

[BootSetup]

SetupLogFile=*name*

Where ***name*** is the name of the log to be created.

The log is always created in the user's Windows directory.

See Also

SETUP.INF file
SetupLog setting
Setup Log Contents

DeleteSetup setting

See Also

By default SETUP.EXE will copy the main Setup program (for example, SETUP1.EXE) to the user's Windows directory. After the installation has been completed this file is left there.

The DeleteSetup setting is used to specify whether the main Setup program should be automatically deleted after the installation which keeps the user's PC free of an unnecessary file.

[BootSetup]
DeleteSetup=*value*

Where ***value*** is **0** to leave the main Setup program or **1** to delete it after the installation (the default).

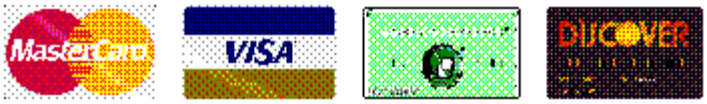
If a temporary directory is being used (with the UseTempDir setting) then the main Setup will be deleted regardless of the DeleteSetup setting.

See Also

SETUP.INF file
SetupFileName setting
UseTempDir setting

Public (software) Library Registration Service

You can order the source code to SETUP.EXE Standard Edition from the **Public (software) Library** by payment with one of the following credit cards: MasterCard, Visa, American Express or Discover.



PsL only provide the registration service - any questions about the status of the shipment of the order, refunds, registration options, product details, technical support, volume discounts, dealer pricing, site licences, etc, must be directed to Chapter One Developments Ltd.

To insure that you get the latest version of the SETUP.EXE source code, PsL will notify us the day of your order and we will ship the product directly to you. When ordering, please tell the operator you are ordering part number **14067**.

Telephone:	**800-2424 PsL** or **713-524-6394**
Fax:	**713-524-6398**
Mail:	PsL P.O. Box 35705 Houston, Texas 77235-5705 USA
CompuServe Email:	**71355,470**

The above numbers are for credit card orders only. Chapter One Developments Ltd cannot be reached at these numbers.

Please use the following form when ordering by Fax, Mail or Email.

Click on this line to print an order form on your printer.

(If you don't have a printer, please write or type the information below on a piece of paper.)

'The Original' Alternative SETUP.EXE Standard Edition Source Code 4.8a
(or latest version) - Part #: **14067**

NAME_____

COMPANY_____

STREET_____

STREET_____

CITY_____

STATE_____ ZIP_____

COUNTRY_____

TELEPHONE NUMBER.._____

FAX NUMBER........_____

EMAIL ID ...._____

CREDIT CARD TYPE _____

CREDIT CARD NUMBER_____

EXPIRATION DATE _____

WHERE DID YOU OBTAIN THE EVALUATION COPY OF SETUP.EXE FROM?

_____

Source Code single copy quantity: ___ @ $ 15.00 ea _____

Shipping/Handling - Outside Europe $ 2.00 _____

Total payment: _____

'The Original' Alternative SETUP.EXE Standard Edition Source Code 4.8a (or latest version) - Part #: **14067**

NAME_____

COMPANY_____

STREET_____

STREET_____

CITY_____

STATE_____ ZIP_____

COUNTRY_____

TELEPHONE NUMBER..._____

FAX NUMBER....._____

EMAIL ID ....._____

CREDIT CARD TYPE _____

CREDIT CARD NUMBER_____

EXPIRATION DATE _____

WHERE DID YOU OBTAIN THE EVALUATION COPY OF SETUP.EXE FROM?

_____

Source Code single copy quantity: ___ @ $ 15.00 ea _____

Shipping/Handling - Outside Europe $ 2.00 _____

Total payment: _____

Cheque Registration

You can order the source code to SETUP.EXE Standard Edition direct from Chapter One Developments Ltd.

Local currency cheques are acceptable. For non-UK pound currency please make the amount equivalent to **8 GBP** (UK pounds). Shipping outside of Europe is equivalent to **1 GBP**.

Please make your cheques payable to: **Chapter One Developments Ltd** and mail them to the following address:

27 Gorse Drive,
Smallfield,
Surrey,
England,
RH6 9GJ.

Use the following form when ordering by mail.

Click on this line to print an order form on your printer.

(If you don't have a printer, please write or type the information below on a piece of paper.)

'The Original' Alternative SETUP.EXE Standard Edition Source Code 4.8a
(or latest version)

NAME_____

COMPANY_____

STREET_____

STREET_____

CITY_____

STATE_____ ZIP_____

COUNTRY_____

TELEPHONE NUMBER.._____

FAX NUMBER........_____

EMAIL ID ...._____

WHERE DID YOU OBTAIN THE EVALUATION COPY OF SETUP.EXE FROM?

_____

Source Code single copy quantity: ___ @ (equiv.£ 8.00) ea _____

Shipping/Handling - Outside Europe (equiv.£ 1.00) _____

Total payment: _____

'The Original' Alternative SETUP.EXE Standard Edition Source Code 4.8a
(or latest version)

NAME_____

COMPANY_____

STREET_____

STREET_____

CITY_____

STATE_____ ZIP_____

COUNTRY_____

TELEPHONE NUMBER..._____

FAX NUMBER....._____

EMAIL ID .....:_____

WHERE DID YOU OBTAIN THE EVALUATION COPY OF SETUP.EXE FROM?

_____

Source Code single copy quantity: ___ @ (equiv.£ 8.00) ea _____

Shipping/Handling - Outside Europe (equiv.£ 1.00) _____

Total payment: _____

ASP Ombudsman Statement

Mike Chapman of Chapter One Developments Ltd is a member of the **Association of Shareware Professionals** (ASP). ASP wants to make sure that the shareware principle works for you. If you are unable to resolve a shareware-related problem with an ASP member by contacting the member directly, ASP may be able to help.

The ASP Ombudsman (OMB) can help you resolve a dispute or problem with an ASP member, but does not provide technical support for members' products. Please write to the ASP Ombudsman at 545 Grover Road, Muskegon, MI 49442-9427 or send a CompuServe message via CompuServe Mail to ASP Ombudsman 70007,3536. The OMB may be contacted by FAX by sending to the ASP FAX number: (616) 788-2765. In communication with the OMB please include a telephone and/or FAX number if available.

What is Shareware?

Known Limitations

- You cannot have bootstrap files on more than one disk.
- Various but specific to VB4.

The Professional Edition does not have the above limitations.

Command Line Parameters

The user can enter command line parameters for SETUP.EXE which your main Setup program (for example, SETUP1.EXE) can act on.

These parameters are passed to the main Setup program without modification along with the **full path** from which SETUP.EXE was run (for example, `A:\` or `N:\NETWORK\APPS\`)

If the user entered `A:\SETUP.EXE /d to` run your application Setup then the parameters passed to your main Setup program (which would be stored in the **Command$** variable for Visual Basic developers) would be:

```
A:\ /d
```

What is Shareware ?

Shareware distribution gives users a chance to try software before buying it. If you try a Shareware program and continue using it, you are expected to **register**. Individual programs differ on details -- some request registration while others require it, some specify a maximum trial period. With registration, you get anything from the simple right to continue using the software to an updated program with printed manual.

Copyright laws apply to both Shareware and commercial software, and the copyright holder retains all rights, with a few specific exceptions as stated below. Shareware authors are accomplished programmers, just like commercial authors, and the programs are of comparable quality. (In both cases, there are good programs and bad ones!) The main difference is in the method of distribution. The author specifically grants the right to copy and distribute the software, either to all and sundry or to a specific group. For example, some authors require written permission before a commercial disk vendor may copy their Shareware.

Shareware is a *distribution* method, not a *type* of software. You should find software that suits your needs and pocketbook, whether it's commercial or Shareware. The Shareware system makes fitting your needs easier, because you can try before you buy. And because the overhead is low, prices are low also. Shareware has the ultimate money-back guarantee - if you don't use the product, you don't pay for it.

Please show your support for the Shareware concept by registering and paying for the Shareware you use. It's the registration fees you pay which allow us to support and continue to develop our products.