16 and 32 Installations

Run-Time Support
Setup Builder supports both 16-bit installations and 32-bit installations. It does this by running one of two versions of the script interpreter program (INSTxx.EXE) at install time. Likewise, there are 16 and 32 bit versions of the Setup Builder program (BLDR16.EXE and BLDR32.EXE).
When SETUP.EXE (a 16-bit program) is run to initiate an installation procedure, it determines which operating system is currently running and which interpreter file(s) are present on the diskette.

If Windows 3.x is running, SETUP.EXE will always look for and attempt to run INST16.EXE. If this is not found but INST32.EXE is, it is assumed that the installation is a 32-bit only installation and an error message will appear to advise of this. You cannot run the 32-bit version of the interpreter or Setup Builder under Windows 3.x.
An error will occur if neither interpreter is found.

If Windows 95/NT is running, SETUP.EXE firstly looks for INST32.EXE and attempts to run this. If it is not found, SETUP.EXE looks for INST16.EXE and attempts to run this. In this way, it is possible to build 16-bit only installs and run them under Windows 3.x and Windows 95/NT.
Likewise, it is also possible to create 32-bit only installations.
An installation is determined as 16 or 32 bit depending on which run-time support is specified. If both run-times are supplied, then an installation is a dual 16/32-bit installation.

To select the type of installation you require, select the <u>Project/Attributes</u> menu option and change to the 'Optional Diskette Files' section. You can then set the install run-time support required.

Script Language Compatibility
The script language is 100% compatible between the 16 and 32 bit versions of the interpreter, however, some functions may operate differently depending on the operating system running - they operate in a way appropriate to the operating system.

Shared Files
Shared file registration simply writes to REGISTRY.INI under Windows 3.x (INST16.EXE) but under Windows 95/NT (INST32.EXE) it writes to the proper 32-bit registry shared file section: (HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\SharedDlls).

Long File Names
The 32-bit version of Setup Builder and the 32-bit version of the interpreter support long file names. It is not possible to obtain this support in the 16 bit versions.
Note: If an installation is to be for dual 16 and 32-bit platforms (ie both INST16.EXE and INST32.EXE supplied), it must not contain any files or directories with long names because the 16-bit run time will not be able to handle the long names.

Registry
Setup Builder provides full support for the Windows registry from the script language. It is suggested that you refer to the Setup Script help file for technical information on the support provided.
Under Windows 3.x, only the HKEY_CLASSES_ROOT key of the registry is supported (because this is the only key that Windows 3.x has), however, under the 32-bit interpreter, all keys are supported.

It is proposed that for the time being, new versions of Setup Builder will continue to support both 16 and 32-bit installations, however, as the general move to Windows 95 and Windows NT takes place, 16-bit support will eventually be dropped.

About Dialog

Purpose
The **About Dialog** displays version and copyright information about this version of the Setup Builder application.

Available from
- Help/About Setup Builder menu option

Backdrop Display Dialog

Purpose
The **Backdrop Display Dialog** enables the user to select the backdrop objects within a project to be used for each screen resolution during an install procedure.

Available from
- Project/Backdrop Display menu option

Comments
The dialog contains a field for each screen resolution. Within each field is listed all the backdrop objects currently defined in the project. By creating or importing backdrop objects, you can extend the list in the fields of this dialog.
For each screen resolution you can assign a separate backdrop object. This is mainly of use where a backdrop has different sized bitmaps or more importantly, 16 and 256 colour bitmaps for appropriate resolutions.
It is possible to assign the same backdrop to all resolutions. The 'Default Backrop' field is the backdrop object to use if the screen resolution cannot be identified as any of the sizes listed in the dialog.
By selecting (Default) in each of the resolution fields, you are specifying that for that resolution you wish the default backdrop object to be displayed - the backdrop object in the last field. All Setup Builder projects default to this setting when first created. In most circumstances, this should be sufficient.

Press 'Ok' to save any changes, or 'Cancel' to exit without saving any changes.

Browse Dialog

Purpose
The **Browse Dialog** enables the user to search for a file name to be used on an underlying dialog.

Available from
- Icon File property in the <u>File Attributes</u> window
- Browse button on the <u>New Object</u> dialog

Comments
Select the drive/directory by clicking on the appropriate fields. Enter the name of the required file and press 'Ok' to select it, otherwise, press 'Cancel' to exit without selecting a file.

Build Project Dialog

Purpose
The **Build Project Dialog** enables you to select the disk drive and directory on which you require your setup disk suite to be built.

Available from
- Project/Build menu option
- 'Ok' button on the Optional Installation Files Dialog.

Comments
The default location is that which was used for the last build process. Enter a new location if required.

Press the 'Ok' button to select the drive and proceed to build the project, or 'Cancel' to exit without building the project

Notes
The name of the directory entered need not already exist since Setup Builder will automatically create it. The directory may also be several levels of subdirectory down - Setup Builder will perform a multi-level directory create to create a tree of directories.

If a single drive specification (eg a:) is entered, the directory used will be the root (and not the current directory on a:) since a '\' character is always appended to the directory entered by the user. If you want another directory, then it must be specified in full.

Warning: It is important to note that Setup Builder will delete all files in the specified directory before it copies the installation suite files to that directory. It will not delete any subdirectories of the build directory or any files in those subdirectories.

Cannot Create Project Error

An error has occured while attempting to create a new project file.

Possible causes:
An attempt to create a new project on a write-protected diskette

An attempt to create a new project in the same location and with the same name as the currently open project

An attempt overwrite an existing project file which is write-protected

An attempt to create a new project on a diskette which has not been formatted or has serious read/write failures - either reformat the diskette or discard it

Cannot delete the main directory Error

This error occurs when an attempt is made to delete the first directory name in the list of directories to create. The first directory name is the main application directory and it is a fixed name which cannot be deleted.

Cannot edit the main directory Error

This error occurs when an attempt is made to modify the first directory name in the list of directories to create. The first directory name is the main application directory and it is a fixed name which cannot be changed.

Cannot move the main directory Error

This error occurs when an attempt is made to move the first directory name in the list of directories to create. The first directory name is the main application directory and it is a fixed name which cannot be moved.

Cannot Open Project Error

This error occurs when Setup Builder has failed to successfully open a project file.

Possible causes
An attempt has been made to open a Project which does not exist. If the project appears in the most recently used project list at the bottom of the Project menu then it is likely that the project file has been deleted

A general failure to read a disk

An attempt has been made to open a project file on a disk which is write-protected. Setup Builder MUST have read-write access to project files in order to successfully open them

Color Dialog

Purpose
The **Color Dialog** is used to change the colour of text on a Backdrop or Cue Card object. It does not apply to text in a Dialog object, only Backdrop objects and Cue Card objects.

Available from
- 'Foreground Colour' property for a text control in the <u>Object Controls dialog</u>.

Comments
From the color dialog, select the required colour for the selected text control and then press 'Ok' to change the property.
Press 'Cancel' to exit without selecting a colour.

Commonly Reported Problems and Questions Asked

Following is a list of reported problems and commonly asked questions about Setup Builder together with the solutions.

'This application uses CTL3DV2.DLL, which has not been correctly installed'

Setup Builder uses CTL3DV2.DLL to provide a '3d' appearance to all controls in dialogs. CTL3DV2.DLL is a Microsoft product and expects to be installed in the \WINDOWS\SYSTEM directory. If it is not, then the above error results.
The Setup program for Setup Builder installs CTL3DV2.DLL in \WINDOWS\SYSTEM.
If you receive this error then another copy of the .DLL must have been found, probably in your \WINDOWS directory or a directory in your path.
It is suggested that you only have one copy of CTL3DV2.DLL on your machine to avoid this error.
Note that the application can be run perfectly successfully even if the above message does appear.

'This software has not been installed correctly. Please re-run the installation procedure'

Setup Builder uses a technique of coded serial numbers to determine the expiry date of the Shareware Edition.
It is possible that you have installed the application perfectly correctly and you still receive this error when you run the application. In this case, it is due to a bug in version 3.06. You will also receive this error if you tamper with certain entries in the .INI file.
The solution is to delete \WINDOWS\BUILDER.INI and re-install the software. Better still, obtain the latest version of the software since this bug has now been fixed.

C:\SETUP\DOSEXEC.BAT Cannot find file. Check to ensure the path and filename are correct

Please see File Compression Errors.

When I run my setup procedure and select 'de-install' I get a message 'The XYZ software has not been correctly installed. The De-installation procedure is unable to de-install the XYZ software.' and I am unable to de-install my software which I have only just installed

There are a number of reasons as to why a de-installation procedure fails with this message:

The most common reason is due to the name of the application .INI file being incorrect. When an item of software is installed, the setup procedure creates an entry in the application .INI file [Install] DeInstall= which stores the name of the de-install script file (only if de-installation is built into the setup procedure). The de-install script file may be found in the application installation directory and stores information on what files were installed and how to de-install them. It normally has the name of the application project file as a name, but with a different extension eg SETUP.SPJ -> SETUP.SCR. The .SCR file is used to de-install your software and is effectively pointed to by an entry in the application .INI file. Every application installed by Setup Builder must therefore have its own .INI file.
You can specify the name of the application .INI file in the Disk/Script File section of the Project/Attributes dialog. By default, the .INI file defaults to %Application% (which is the value set in the Application Name field in the Application section of the Project/Attributes dialog) which is the name of the application.
If you change the application name to a name longer than 8 characters or place spaces or invalid

file name characters in the application name, this will cause the .INI file to have an invalid file name which causes the installation directory of your software not to be saved in the .INI file (the .INI file can't be created with an invalid file name) which in turn causes the de-install to fail with the above message because the de-install script file cannot be found.
The solution is to change the name (in the Project/Attributes dialog) of the .INI file: eg to MYAPP.INI   This will solve the problem, but you will need to do a re-build of the project. Having done this, you can then set up the .INI file manually and place an entry in it:

[Install]
Path=C:\MYDIR\
DeInstall=C:\MYDIR\SETUP.SCR

You can then run your de-install.

If the de-install script file is deleted this will also result in the above message.

When I perform a Build, the build process starts but when it comes to compressing INSTxx.EXE the build 'locks'

This is to do with the Windows configuration in Windows 3.1 and Windows for Workgroups. Go to 'Control Panel' and select the '386 Enhanced' option. There is a checkbox 'Exclusive in Foreground'. This field should not be checked ie no cross. If it is, uncheck it otherwise all Windows applications will get 100% of the CPU time and all background processes such as the compression feature in Setup Builder will get no CPU time at all. This causes the compress to 'hang'.

See Also File Compression Errors.

I've built my 16-bit   (or dual 16/32 bit) install under Windows 95. When I run the SETUP.EXE under Windows 3.x or Windows NT I get a message which says 'This program requires a newer version of Windows'

This is because of LZEXPAND.DLL being included on the setup disk. When an installation is built and the LZEXPAND.DLL file is included (Project Attributes/Optional Files), the LZEXPAND.DLL file is copied from the WINDOWS directory of the currently running version of Windows. This means that if you are running Windows 95 or Windows NT, the Windows 95 or Windows NT version of LZEXPAND.DLL will be copied onto the diskette. Each version of the file is specific to its own operating system and will not run under other versions of Windows.
When SETUP.EXE is run, it uses LZEXPAND.DLL to uncompress the interpreter and backdrop bitmaps onto the hard disk in order to run the installation procedure. SETUP.EXE is a Windows 3.1 16-bit program. If LZEXPAND.DLL is placed on the diskette, that version will be used when SETUP.EXE is run. If the version of LZEXPAND.DLL is any version other than the Windows 3.x version, you will receive the above error message.

For this reason, it is recommended that you do not distribute the LZEXPAND.DLL file. It is supplied as a standard component of all versions of Windows and therefore does not need to be distributed. The Setup Builder default for new projects is not to include LZEXPAND.DLL.

File Compression Errors

One of the following file compression errors may occur:

Unable to find DOSEXEC.PIF
File compression requires the use of DOSEXEC.PIF. This error occurs when DOSEXEC.PIF cannot be found to perform file compression. DOSEXEC.PIF should be in the Setup Builder directory (Normally C:\SETUP).

Compression unable to find 'filename'
This error occurs when a file which does not exist is included in a project.
This would occur when a project is created and the source file is later deleted without removing it from the project.

Compression Failure on 'filename'
This error usually occurs when there is not enought space on the target disk, the disk is write protected, the file attempting to be created is write protected or the COMPRESS.EXE program fails.
It may also appear after other compression error messages to indicate that a general file compression error has occured.

Unable to find COMPRESS.EXE
This error occurs when Setup Builder is unable to find the COMPRESS.EXE program.
COMPRESS.EXE must be in the Setup Builder directory (Normally C:\SETUP).

Compression attempting to copy 'filename' onto itself.
This error occurs when the file specified in the project has the same name as a file on the setup disk being created.
This error would only occur if a setup disk was being created from the WINDOWS\TEMP directory since this is where temporary compressed files are stored prior to copying to the setup disk

DOSEXEC.BAT Cannot find file. Check to ensure the path and filename are correct
Setup Builder uses the Microsoft COMPRESS.EXE program to compress files. It creates a batch file (DOSEXEC.BAT) in the Setup Builder directory (normally C:\SETUP) which contains a call to C:\SETUP\COMPRESS.EXE and then uses a Windows .PIF file (DOSEXEC.PIF - also in the Setup Builder directory) to run the batch file.
The above error occurs whenever Setup Builder has not been installed in the C:\SETUP directory, for example on a network drive or simply another directory eg C:\BUILDER or if DOSEXEC.PIF has been changed in error.
To correct the problem you must use the Windows PifEdit program to modify DOSEXEC.PIF. You should modify the 'Program Filename' to point to the directory in which Setup Builder has been installed. A typical example modification might be:

G:\SETUP\DOSEXEC.BAT

You should then save the changed .PIF file.
COMPRESS.EXE is supplied with Setup Builder and must be in the same directory as Setup Builder. The Setup Builder installation procedure will ensure that this is the case.
The above should resolve the problem.

Import 3rd Party Components Dialog

Purpose
The **Import 3rd Party Components Dialog** enables the user to automatically import files into a project which are part of a third party product

Available from
- File/Import 3rd Party Components menu option

Comments
Select the components you require to be imported into your project by selecting one or more of the items listed and then press the 'Ok' button to add them to your project.
Press 'Cancel' to exit the dialog without adding any extra files to your project.

Notes
Third party components are those such as dynamic link libraries and support files supplied by other vendors, for example, the Visual Basic run-time file VBRUN300.DLL and the dynamic link libraries which make up Access to support a Visual Basic application are such third party components.
All the components supported by Setup Builder are listed in the IMPORT.INI file which may be found in the Setup Builder directory.
If you wish, you may add further components to this file but you must comply with the file format documented in IMPORT.INI.
IMPORT.INI will be upgraded in future versions of Setup Builder to support further third party products.

Delete Control Confirmation

This message appears when the user has selected to delete a control from an object.

Select 'Yes' to continue and delete the control.
Select 'No' to continue, but not delete the control.

Delete Directory Confirmation

This message appears when the user has selected to delete a directory from the list of directories to create.

Select 'Yes' to continue and delete the directory.
Select 'No' to continue, but not delete the directory.

Delete Install Confirmation

This message appears when the user has selected to delete an optional installation name.

Select 'Yes' to continue and delete the installation name.
Select 'No' to continue, but not delete the installation name.

Delete Object Confirmation

This message appears when the user has selected to remove an object from the current project.

Select 'Yes' to continue and remove the object from the project.
Select 'No' to continue, but not remove the object from the project.

Note: Removing an object from a project does not delete the object file.

Stop Build Confirmation

This message appears when the user has selected to stop a build process from continuing.

Select 'Yes' to stop the build process.
Select 'No' to continue the build process.

Contents for Setup Builder Help

Setup Builder is a utility program for creating and maintaining Windows-hosted application installation procedures.
To learn how to use Help, press the F1 key.

- What is Setup Builder ?
- What's New ?

Application Main Window
- Menu Options
- Main Setup Builder Window
- Project Files window
- File Attributes window
- Project Objects window
- Build Output window

Errors and Troubleshooting/Technical Information
- Setup Builder Errors
- File Copying Errors
- File Compression Errors
- Commonly Reported Problems and Questions Asked
- Files not to distribute
- IMPORT.INI and Importing of 3rd Party Components
- 16 and 32 bit installations
- OLE Control registration
- De-Install Procedure

- Technical Notes
- Further Technical Information
- Setup Script Help

About Setup Builder Features
- What are User Defined Objects ?
- What are Optional Installations ?
- What Are Optional Include Files ?
- Multiple Directory Installations
- Multiple Program Installations
- Creating Program Manager Icons from icon files

Getting Started/Samples
- Quick Start to creating an Install Disk Suite
- Sample Files

Other Information
- Other products available
- What is Shareware ? - Description and Disclaimer
- Payment for and Registration of Software
- Software Price List

- Contacting the Author

- Application Limitations & Limitations on use
- Other Setup Builder Users

- Help File Copyright

File Copy Errors

One of the following file copying errors may occur:

The specified source file <filename> could not be found"
This error occurs when a file which does not exist is included in a project.
This would happen when a project is created and the source file is later deleted without removing it from the project.

An error occured while creating <filename>. Try removing write-protection from this file/disk.
This error usually occurs when there is not enought space on the target disk, the disk is write protected or the file attempting to be created is write protected.

There is not enough space on the target disk to copy <filename>. Try using another diskette.
This error occurs when there is not enough space on the target drive for the specified file.
Setup Builder will not normally report this error as it normally asks for another diskette when this error occurs.
However, this error may occur at the end of building a setup procedure when a temporary copy of the setup script file is copied from the project (.SPJ) file directory to the first diskette in the install suite.
Please see the Technical Notes **Setup Script diskette space** section for information on how to reserve more space for the script file.

You cannot copy <filename> onto itself - the source and target file names must not be the same.
This error occurs when the file specified in the project has the same name as a file on the setup disk being created.
This error would only occur if a setup disk was being created on the same disk as that containing all the files making up the project.
Since Setup Builder always creates setup procedures in the root directory of a disk, the source project files would have to be in the root of the same disk on which the setup procedure was being created.

An error occured while reading <filename>.
This error occurs when Setup Builder has failed to read a block from a file. It normally only occurs when there is disk corruption.

An error occured while writing <filename>.
This error occurs when Setup Builder has failed to write a block to a file. It normally only occurs when there is disk corruption.

■

This Windows Help file was written by Graham Plowman
using Help Builder Version 1.09.001 and refers to:

Setup Builder Version 5.01.000 / 02/12/96

Create Dependencies Warning

This warning occurs when SetupBuilder identifies that the project file (.SPJ) which is being opened has been moved to another directory which is different from that in which the project was originally created.

SetupBuilder assumes that if a project file has been moved, then it is likely that the component files have also been moved. Since the names of these files are held in the project file, they must be amended to reflect the new project directory.

If the user selects the Yes button, the dependencies will be rebuilt.

If the user selects the No button, dependencies will not be rebuilt but the project file will be amended to register the fact that it has been moved, but not the topic files.

Normally, if you copy all the component files of a project from one directory to another then you should answer 'Yes' to rebuilding dependencies.
If you just move the project (.SPJ) file then you should answer 'No'

Data Required error

This error occurs when data has not been entered in a screen field which must have data entered in it.
Enter the requested data and try again.

De-Install Procedure

When Setup Builder installs an item of software, if the project was built with the de-installation option selected, the install procedure also creates a de-installation procedure.
At install-time, the appropriate script interpreter (INST16.EXE or INST32.EXE) is copied into the WINDOWS\GPPSOFT directory (where WINDOWS is the directory where Windows is installed) on your hard disk. The DEINST.EXE program is also copied into this directory.
The install procedure creates a de-install script which has the name of the Setup Builder project file as a name and a numeric extension to prevent conflict with different software products (this numeric is incremented until the file is not found).
The de-install script file is also created in the WINDOWS\GPPSOFT directory.
You must not delete any of these files otherwise the de-install procedure will not work.

The install procedure recognises whether the software has been installed already or not. Typically, you will notice this at install time when it doesn't request the installation directory - because this is already known. This allows an installation to be repeated (eg extra optional components added) without creating another de-install script file - the exesting one is maintained instead. It also stops the same software being installed in several different directories which would otherwise confuse the de-install process because the de-installer wouldn't know which directory to de-install from.

The installation procedure also creates DEINST.INI which is in the Windows directory. This file is used by DEINST.EXE to record each item of software which is installed.

The install procedure will create a de-install icon. This icon runs DEINST.EXE with the name of the application as a parameter which is looked up in the DEINST.INI file. If no parameter is supplied, DEINST.EXE will display a list of products currently installed so that you can choose which one you want to de-install. You may find it useful to change the Setup Builder de-install icon to remove the parameter so that you can see a list of all the products installed by Setup Builder.
The DEINST.INI file registers where an item of software was installed, the name of the de-install script file and the name of the interpreter which installed it - so that the same interpreter can de-install it. DEINST.EXE runs the appropriate interpreter with the de-install script file name as a parameter.

The de-install procedure will remove all files and directories installed by the installation procedure. It will remove the application .INI file (16 bit installs only), any registry entries created (32 bit installs only) and any de-install script files and entries in DEINST.INI.
You should ensure that all installation procedures share the same .INI file or Registry section as that for which your software is written.
16-bit applications should have there own .INI files and NOT use WIN.INI. If you use WIN.INI as your application .INI file, the de-install procedure will remove it upon de-installation with obvious results.
By default, all .INI files should be placed in the Windows directory. Under 16-bit installations, Setup Builder will always store the .INI file in the Windows directory. Your application should use this same .INI file, but if you choose to place your application .INI in the application directory, you should ensure that it is properly removed by the de-install procedure - usually via Used-Defined Script.
32-bit applications should use there own registry section. The de-install procedure will ensure that the entire section for a software product is removed.

Note: The de-install procedure will not de-install any files created by your application which the installation procedure didn't install. To remove these, you must use 'User-Defined Script' to delete them (Delete function). Also, note that under Windows 95/NT, Windows Help creates '.GID' files for each help file opened. These are hidden files with the same name as the help file, but a .GID extension and these must be un-hidden by your de-installation user defined script, then deleted. Failure to do this will stop all directories being removed by the de-installation procedure.

Delete File

You have just selected to delete the currently selected file(s) from the list of files in the project.

The current dialog is asking you to confirm that you do in fact wish to remove the selected file(s) from the list.

Answer 'Yes' to remove the file(s) or 'No' to keep the file(s).

You can add the file(s) back in to the project at a later time via the Project/Edit dialog, but you will need to re-enter the File/Attributes for each file deleted.

Directory already exists in the list Error

This error occurs when an attempt is made to add a new directory and the directory has been found to already exist in the list. You cannot add the same directory to the list more than once.

See Also
Application Limitations

Files not to Distribute

The following files are often distributed by software vendors and do not need to be:

DDEML.DLL
This is the DDE Management library. It is a standard part of all versions of Windows and does not need to be distributed.

LZEXPAND.DLL
This is the file compression library. It is a standard part of all versions of Windows and does not need to be distributed.

COMMDLG.DLL
This is the Common Dialog library. This file is distributed very widely. It is a standard part of all versions of Windows and does not need to be distributed.

VER.DLL
This is the File Version checking library. It is a standard part of all versions of Windows and does not need to be distributed.

While Setup Builder does not stop you distributing the above files, it is recommended that you do not distribute the files since it is possible that you can cause corruption on a machine, particularly if you install a 16 bit version over a 32 bit version in say, Windows 95. Setup Builder installs can be made do perform version checking before copying a file by adjusting the file's 'overwrite confirmation' property.
If you must distribute the above files, make sure that the 'Confirmation' property in the File Attributes Window is set to 'Confirm Overwriting Newer File Only'. This will cause a version check to be performed and the file will not be upgraded if the one on the diskette is older than the one on the hard disk.
You could also use the 'Confirm Overwrite if file exists' option which will cause a prompt to appear every time an attempt is made to overwrite a file, regardless of its version.

Edit Project Dialog

Purpose
The **Edit Project Dialog** enables you to select all of the files which you require to be included on your setup distribution disk(s) and therefore in the project.

Available from
- Project/Edit menu option
- 'Ok' button on the New Project Dialog.

Comments
The top-left list box shows all of the files in the current directory - the current directory is displayed above the list box.
To include a file in the project, double-click on the required file and it will appear in the list box at the bottom left.
Note that Setup Builder will not allow you to include a file twice.

The top-right list box shows the current directories and below this, a drop down list shows the drives available. Click on these as required to change drives/directories.

The bottom-left list box shows all of the files which have been selected for inclusion in the project. To remove a file from the project, double-click on the file in this list.

The 'Add All' button will add all of the files shown in the top-left list to the bottom-left list, whereas 'Remove All' does the opposite.

Press the 'Ok' button to save the list, or 'Cancel' to exit without saving the list.

Notes
Any new files added to a project will have default file attributes so you should adjust these via the File/Attributes menu option.

See Also
Application Limitations

The selected script file is already encrypted Warning
The selected script file is already un-encrypted Warning

These warnings occur when an attempt is made to:

- Encrypt an already encrypted script file
- Un-encrypt a script file which is not encrypted

When these messages appear, no further action is performed on your script file and it will be left intact.

Encrypt/Unencrypt Script File Dialog

Purpose
The **Encrypt/Unencrypt Script File Dialog** enables you to encrypt or un-encrypt a script file created by Setup Builder or to encrypt scripts that you have written manually to be supplied on your distribution disks

Available from
- File/Encrypt / Un-encrypt Script menu option

Comments
Enter the name of the script file to be encrypted or un-encrypted in the **File Name** field or press the **Browse** button to search for the required file.
Select the appropriate operation by clicking the **Encrypt Script File** button or the **Un-encrypt Script File** button. Then press the **Ok** button to commence the conversion.

Notes
Setup Builder will not allow you to encrypt a script file which is already encrypted and similarly, it will not allow you to un-encrypt a script file which is not encrypted.

The script file interpreter automatically recognises encrypted script files and reads them accordingly. There is no need for any extra switches/parameters to tell it whether a script is encrypted or not.

The conversion process actually replaces your original script file with the newly converted version so you don't end up with two files: your original file and the newly converted version. The new file is created as a temporary file on your hard disk. It is then copied over your original file. The temporary file is deleted afterwards.

Setup Builder Errors

<Not Found>
An error occured while creating <filename>'. Try removing write-protection from this file/disk
An error occured while reading <filename>
An error occured while writing <filename>
An object with this name already exists
Cannot add any more directories to the project
Cannot add any more objects to the project
Cannot add any more optional installations to the project
Cannot delete the main directory
Cannot edit the main directory
Cannot move the main directory
Compression attempting to copy <filename> onto itself
Compression Errors
Compression Failure on <filename>
Compression unable to find <filename>
Create Dependencies
Data Required
Directory already exists in the list. It cannot be added again
DOSEXEC.BAT Cannot find file. Check to ensure the path and filename are correct
Failed to create the temporary preview script file
Failed to find object file
Failed to open the selected file
File already exists in the project
File copying errors
Install Already Exists
Invalid File Name
Invalid numeric value entered
Invalid object name
Number out of range
The maximum number of controls allowed in an object has been reached
There is no directory to be modified
There is no option to be modified
There is not enough space on the target disk to copy '..'. Try using another diskette
The selected script file is already encrypted
The selected script file is already un-encrypted
The specified source file <filename> could not be found
This object file already exists in the project
Too many files selected for project
Unable to create project file
Unable to find COMPRESS.EXE
Unable to find DOSEXEC.PIF
Unable to open selected project file
Unrecognised Object File
Unrecognised Object File version
Unrecognised project (.PRJ) file
Unrecognised project (.PRJ) file version
You cannot copy <filename> onto itself - the source and target file names must not be the same

Failed to execute program error

This error occurs when Setup Builder has failed to execute the script interpreter for previewing displays or testing scripts.

Possible causes
The is not enough memory available to tun the interpreter program. Close down some other currently running applications to free up memory.

File already exists in the project Warning

This warning occurs when a Visual Basic .MAK project file is being imported into a Setup Builder project or a file is being pasted from the Clipboard. Setup Builder has identified that an attempt is being made to add a file to the current project when it already exists in the project.

A file cannot be added to a Setup Builder project more than once.

File Attributes Window

Purpose
The **File Attributes Window** enables the user to make selections about how a file is to be installed during an install procedure. It is also possible to specify the same selections for a number of files at the same time: see the notes at the bottom of this page.

Available from
- File/Attributes menu option
- F/ATRIB toolbar button
- Double clicking on a file in the Project Files window

Comments
The window contains a field and a list which shows all the properties of the file being installed which can be changed.
To change a property, use the mouse to double click on the appropriate property. The mouse cursor will move to the field at the top of the window and the field will show the current setting of the property. Change the property as required and press the Enter key.
You can select another property at any time, simply by clicking on it with the mouse or using the cursor keys to move up and down the list. In doing so, the changes you have made to the current property will be automatically saved.

The list contains the following properties:

Add to Registry
This property enables a file to be registered in the Shared File Registry which is used to control the deletion of shared files during de-installation.
Please see the **Register** function in the script language help for a detailed description of how the Registry operates and how it should be used.
Please note that the file Registry facility is only activated if you have selected the 'de-install' option in the Project Attributes dialog.

Check if in use
Causes a check to see if the file (normally executables/DLLs etc) is in use by Windows. If it is, an error message will appear and the install procedure will be terminated.

Compress
Setup Builder enables you to place files on your setup disks in compressed form. The compression process uses the standard Microsoft LZExpand method.
It is suggested that you compress files which are bigger than 200K, but not those below about 2K since the latter will not actually save any diskette space because the smallest allocateable block on a diskette is usually 2K. This is however at the user's disgression.
Select Yes to compress the file or No if the file is not to be compressed.
If compression is selected, you must complete the Compress Extension property.

Compress Extension
This property contains the file extension to be used for the compressed file when it is placed on your setup diskette. By default, Setup Builder uses the Microsoft convension of placing a '_' as the last character (eg .EXE becomes .EX_ and .A becomes .A_), however, you may type a different extension if required, but it must not be the same as the extension of the original file.

Confirmation
This option enables you to specifiy the action to be taken during installation if a file already exists. The action may be:

'No overwrite confirmation' - the file will be installed anyway whether it already exists or not.
'Confirm overwrite if file exists' - displays the Overwrite Message property below as a message if the file already exists.

'Confirm overwriting newer file only' - checks to see if the file already exists and displays the Overwrite Message property below as a message to ask the user to confirm overwriting the file. The confirmation will only appear if the file being installed is older than the one to be overwritten. If the file being installed is newer than the existing file then the existing file will be overwritten automatically with no overwrite confirmation message. This enables files to be automatically upgraded where necessary. If the user answers 'no' to the confirmation, the file is not installed and copying proceeds with the next file.

If an overwrite confirmation has been configured and the file does not already exist on the user's machine, the file will be automatically installed with no message displayed.

If the new and old files are both exactly the same, the file is not copied, regardless of the overwrite confirmation option selected (except no confirmation which always overwrites). This is to reduce installation time.

File version comparison is performed using the version information library VER.DLL if the two files being compared contain version information. If not, comparison is performed on the basis of files dates and times. This allows files with version information and files with no version information to both be handled.

Copy Comment
This is the message which will appear at the top of the gauge dialog for the file when file copying is in progress in the installation procedure.

Cue Card
This property enables you to set the cue card which is displayed when the file is being installed. If (None) is selected, cue card display will be turned off and no cue card will be shown. If (Current) is selected, the cue card which is currently displayed on the screen will continue to be displayed. If any other item is selected, that cue card will be displayed.
By changing this property at regular intervals through the Project Files list, you can have a sequence of cue cards during file copying at install-time.

File Usage
This specifies how a file is to be copied and may be one of the folowing:

'Do Not Copy File' simply does not install the file at all. It is the programmers' responsibility to write some 'User defined' code to install these files as required. Often, this facility is used to supply files on install diskettes as extra components for the user to copy manually when required. 'Display in gauge' is the default. This causes the file to appear in the gauge and to be installed on the user's machine.

Icon File
Icon Index
You may also optionally specify the name of a .EXE or .DLL file from which to extract the icon to use in Program Manager. The index field enables the number of the icon to be specified. The first icon in a file is zero, the second one and so on.
If the icon file is left blank, the name of the file being installed will be used as a default. The default icon is zero. It is possible to specify the icon index and no icon file: the icon will be taken from the file being installed.

Icon Parameters
This property specifies any parameters which the file may need if it is an executable when it is created as an icon. Typically, this would be any parameter options or file names.

Icon Text
This property   specifies the text which will appear below the icon in Program Manager if an icon is selcted to be created for a file.

Make Icon
This property specifies whether an icon is to be created in a Program Manager group for the file. If you select Yes, you should enter the Icon Text property.

OLE Registration
This property is only applicable to files which are OLE control files.
Self-registering executables should use the /REGSERVER option while other controls (eg .OCX) should use the DllSelfRegister option.
Please see OLE Control Registration for more information on Setup Builder's support for OLE Control registration.

Optional File Inclusion
This property marks a file to appear in the Optional Installation Files Dialog when a project is built, giving you the opportunity to select it for inclusion in the project at build-time.
See Also: What Are Optional Include Files ?

Optional Installs
This property enables you to select which Optional Installations a file should be included in. This property is a binary coded field. You should press the button to the right of the field at the top of the window to obtain the File Installations Dialog whicjh enables you to make your selections more easily.

Overwrite Message
If you require a specific message for an overwrite confirmation, complete this field.

The following strings may be placed in this message. These are substituted at run time:

$OLDFILE$ $NEWFILE$ The name of the old and new file
$OLDDATE$ $NEWDATE$ The date of the old and new file
$OLDTIME$ $NEWTIME$ The time of the old and new file
$OLDSIZE$ $NEWSIZE$ The size of the old and new file

The old file is the file already existing and the new file is the file which is to overwrite the existing file.
Note that the $ sign substitution in older versions of Setup Builder is now no longer supported.
If this field is left blank, a default message will be displayed which shows the names, dates, times and lengths of the two files being compared:

Setup is attempting to overwrite:

C:\TEMP\TEXT2.TXT
(24/6/95   11:24   47 bytes)

with:

A:\TEST\TEXT2.TXT
(24/6/95   11:10   47 bytes)

Are you sure you wish to overwrite this file ?

Protect After Copy
This property enables you to specify whether the setup procedure should write protect a file after it has been installed.

Target Location
This specifies the directory where the file is to be installed. It must NOT contain a file name and should end with a backslash if the directory name is a subdirectory such as %InstallPath %SAMPLES\

You can place a 'hard coded' directory name in this field, but it is adviseable to take advantage of the Setup system's ability to use variables in which case you may place the following in the field:

%InstallPath%   This will cause the file to be installed in the directory which the user enters when prompted to enter a path in which to install files during the installation process. This directory is always created automatically by the Setup Builder installer.

%WindowsDirectory%   This will cause the file to be installed in the Windows directory.

%SystemDirectory%   This will cause the file to be installed in the Windows system directory.

Examples:

        %InstallPath%   For C:\MYAPP\
        %InstallPath%SAMPLES\   For C:\MYAPP\SAMPLES\

If you   use the second example above (C:\MYAPP\SAMPLES\), the Setup Builder installer creates C:\MYAPP\ but you must manually create the SAMPLES directory using Setup Script in the Pre-File Copying User Script section in the <u>User Code dialog</u>. You need to enter the script:

        MkDir("%InstallPath%SAMPLES")

If you are implementing a de-install as well, you must also remove the directory manually in the de-install script section:

        RmDir("%InstallPath%SAMPLES")

Setup Builder will ensure that any files it installs in subdirectories are de-installed.

Unprotect Before Copy
This property enables you to specify whether the setup procedure should switch off write protection of the file before an attempt is made to copy over the existing file. It is a good idea to select this option if there is any risk of the file being write-protected as an attempt to install over a write-protected file will cause an install-time error.

Working Directory
The Working Directory field may be optionally left blank, but if supplied, should contain the name of the directory you wish to be made current for the icon when the icon is run. The directory of the file being installed is taken as the default.
It is recommended that any directory specifications are given in terms of the installation directory, for example: %InstallPath%
Please see the Setup Script file for further information on the script language and variables.

Notes
Using the **File Attributes** window it is possible to assign the same settings to a number of files in the project at the same time. To do this, click on the required files in the Project Files window (holding down the Ctrl key at the same time) which are to have the same attributes, then select 'File/Attributes' from the menu.
When you press 'Enter' in the **File Attributes** window or select another property, all of the selected files will be given the same setting for the property just changed.
If you select several files using this technique, the property shown in the **File Attributes** window is always the one which was selected last - the one with the dotted line around it in the Project Files window.

The **Project Files** window automatically disappears during a build to prevent you from changing properties during the build process. It also disappears when a project is closed or when a new project is opened.

The window saves its position and size so that next time it appears it is shown in the same place and the same size as when it was last displayed.

File Installations Dialog

Purpose
The **File Installations Dialog** enables the user to select which optional installations a file should be assigned to.

Available from
- 'Optional Installs' property in the <u>File Attributes</u> window

Comments
Select each installation to which the currently selected file(s) in the Project Files window are to be assigned by clicking on the required item.
Multiple installations can be selected by holding down the Ctrl key when clicking.

Press the 'Ok' button to save the changes or 'Cancel' to exit without saving any changes.

See Also
<u>Application Limitations</u>

The IMPORT.INI file

The IMPORT.INI file is supplied with Setup Builder and resides in the Setup Builder directory.

It is used by Setup Builder to list third party software vendor's products and the supporting files used by those products to provide a quick and easy way to import 3rd party products into a Setup Builder project for distribution with your application.
Typically, all Visual Basic applications must be supplied with VBRUN300.DLL and MSAFINX.DLL (The latter is necessary to stop the notorious 'File not found error'. It contains the iif() function and various date and financial functions). These files are listed in IMPORT.INI as Visual Basic 3.0 Runtime support files. By selecting 'Visual Basic 3.0 Runtime' files in the Import 3rd Party Components Dialog, the user can quickly and easily import the appropriate files into their project.

If your Windows installation is in a different location to C:\WINDOWS or any of the component files listed in IMPORT.INI are in different locations, you will need to edit IMPORT.INI to point to the appropriate locations of the files on your hard disk.

IMPORT.INI is a standard windows .INI file. Each section heading lists the name of the third party component and this is displayed in the Import 3rd Party Components Dialog. Under each section is a list of each of the files which make up that 3rd party component. The format of the individual file items must conform to the format documented in the IMPORT.INI file, otherwise Setup Builder will not read them correctly and may import unwanted files into your project.

Import Object Dialog

Purpose
The **Import Object Dialog** enables the user to import a previously saved object file into the current project. This enables a library of standard objects to be created and shared across multiple projects.

Available from
- Object/Import Object menu option

Comments
Select the drive/directory by clicking on the appropriate fields. Enter the name of the required file and press 'Ok' to select it, otherwise, press 'Cancel' to exit without selecting a file.

Installations Dialog

Purpose
The **Installations Dialog** enables the user to create up to 15 different optional installations with any name that is required.
Optional installations allow you to give the person installing your software the option to install only certain components of your software. Typically, this would be used to give the option to install (or not) sample and example files.
In the File Attributes you can modify the 'Optional Installs' property to select which installs a file in your installation belongs to. A file can belong to one or even all installations.
All of the install names which are created/listed in this dialog are displayed in the File Installations dialog.

Available from
▪   Project/Installations menu option

Comments
The dialog contains the following fields:

Defined Installations
This lists all the installation names that are currently defined in the project. When Setup Builder projects are first created, the 'Install the %Application software%' install is always created and all files default to this installation.
Press 'Delete' to remove an installation. Note that due to the way Setup Builder operates, when you remove an install, all files assigned to that install will automatically be adjusted and removed from that install.
Press 'Edit' to change the name of an installation. This does not affect files assigned to the install - they will automatically be assigned to the new name.

Install Name
Enter the name of an install in this field and press 'New' or 'Modify' to save it to the left hand list.

Install-Time Installation Options Dialog
This field controls the display of the dialog at install-time which gives the user the option of which parts of your software to install.
If 'Check Boxes' are selected, the user will be able to select any combination of the components of your software. This is the most common selection.
If 'Radio Buttons' are selected, the user will only be able to select one component. This is of use where the options are mutually exclusive - you don't want the user to install more than one option.

Press 'Close' to exit the dialog.

Install Already Exists error

This error occurs when an attempt is made to create a new optional installation name with the same name as an existing installation name.

You cannot have two optional installations with the same name.

Invalid File Name Error

This error occurs when a file name is required in a field and the file name has been incorrectly entered.

Possible causes

An invalid file name has been entered which may specify an invalid drive, a non-existant disk drive or path or even have invalid formatting characters within the name

The disk on which the file name specifies is write protected

The file already exists, but it is write protected

Invalid numeric value entered Error

This error occurs when a number is required in a field and it has either not been entered or the text entered does not constitute a valid number.

Possible causes
Non-numeric characters entered in a field expecting a number

Invalid object name Error

This error occurs when an object name is required in a field and no object name has been entered.

Possible causes
Leaving a field blank which requires an object name

Invalid directory Error

This error occurs when an invalid directory name has been entered.

Application Limitations

Setup Builder currently has the following limitations, which will be modified in future releases:

- 25 files per project (Shareware Edition), 2048 (Professional Edition)

- 25 bitmaps per project (Shareware Edition), 2048 (Professional Edition)

- 15 Optional Installations (Shareware and Professional Editions)

- 10 user defined objects per project (Shareware Edition), 64 (Professional Edition)

- 16 automatically created directories (both editions). Manual script is required for more.

- Setup Builder installs cannot install Windows NT Services.

Limitations on Use
Users of the Shareware Edition must **not** distribute the diskettes produced by Setup Builder, however users of the Professional Edition may do so freely.

There is no fee payable for distributing large numbers of installation procedures, however, the author does request that if you use Setup Builder for distributing other software for monetary gain that you mention Setup Builder and the author's <u>contact details</u> in your supporting documentation or help files.
The privilage of this lack of distribution fee is on condition that you do not sell on Setup Builder or any of its components for monetary gain. You may supply SETUP.EXE, INSTxx.EXE and a script file with your applications freely but you must not charge for them in your overall software charge and you must have registered your copy of Setup Builder.

Save Before Continue ?

The option you have just selected will cause loss of changes which have been made to the project since it was last saved.

The current dialog is asking you to confirm what you wish to do:

Yes Will save the project and continue with the option selected

No Will continue with the option selected without saving the changes and therfore loosing them

Cancel Will not save the changes, but it will not continue with the selected option either.

Main Setup Builder Window

The main Setup Builder window appears when the application is first started up and remains on the screen until the application is either minimised or closed.

Setup Builder uses the Windows 'Multiple Document Interface' (MDI) paradigm for its main window.
The main window may contain a number of other MDI 'Child' windows:

- The <u>Project Files window</u>
- The <u>File Attributes window</u>
- The <u>Project Objects window</u>
- The <u>Build Output window</u>

At the top of the Setup Builder main window is the menu and a toolbar. Toolbar options may be selected by simply pressing the appropriate button with the mouse. All of the toolbar options are duplicated on pulldown menu options.
At the bottom of the Setup Builder main window is the status bar. It displays messages about Setup Builder's current activity. It also displays a brief description of menu options when they are highlighted on a pulldown menu.
Both the toolbar and status bars may be removed/activated by selecting the appropriate option from the **View** menu.

To close the Setup Builder main window and therefore exit the application, either press Ctrl+F4 or use the mouse to double click on the system menu icon or select the **File/Exit** menu option.

Creating Program Manager Icons from icon files

Setup Builder installations can create Program Manager icons which use icons from files other than the file specified as the program to run for the icon.
Typically, this approach is used for help files. A .HLP file is placed as the program to run for an icon, but because the icon always defaults to the WINHELP.EXE icon, some developers want to change the icon. No matter what icon you place in a help file, Program Manager will never display it as an icon. The icon in the help file is only ever displayed when WinHelp is 'running' the file and it is minimised. In fact, in Windows 95 Microsoft have dropped the icon feature in a help file.

Making Setup Builder use your icon file at install time
Firstly, the file which contains the icon which you wish to use **must** be included in your project via the Edit Project Dialog. The file can be a .EXE, a .DLL or a .ICO file and it must be installed onto the target hard disk at install time, typically in the %InstallPath% directory.

To make Setup Builder use the icon file for a specific Program Manager icon, it must be specified in the **Icon File** field of the File Attributes Window, for example:

%InstallPath%MYICON.ICO
or
%InstallPath%MYEXE.EXE

The **Icon Index** field may optionally be set. The first icon in a file is always 0 (Setup Builder defaults to this), the second 1 and so on. A .ICO file only contains one icon, so the Icon Index must always be 0 for these files.
If the Icon Index specifies a value for an icon which does not exist, Program Manager will use one of it's own default icons. If Program Manager displays one of it's default icons and all indexes/file names are correctly specified, it can also indicate that Program Manager couldn't find the file. You may want to check that it has been installed correctly.

Before an icon is created, your project must have the option set in the Program Manager Group Dialog to create a Program Manager group. Setup Builder installs will not attempt to create icons if no group it created.

Once the above has been configured, a Setup Builder installation will correctly use your file to create the Program Manager icon.

See Also
Edit Project Dialog, Program Manager Group Dialog, File Attributes Window

Menu Options

The Setup Builder program has the following menu options:

- File   Options relating to files selected in the project
- Edit   Options relating to cutting, pasting and selection of files in the project
- View   Options relating to the Setup Builder main window appearance
- Project   Options relating to a Project
- Object   Options relating to user defined objects
- Help   Options relating to help

File Menu
New
Enables a new project file to be created.
See New Project Dialog

Open
Opens an existing project file on the disk.
See Open Project Dialog

Save
Saves the currently open project to disk.

Save As
Saves the currently open project under a new name.
See Save Project As Dialog

Close
Closes the currently open project file. You will be asked to save the project if it has been changed in any way since it was last saved.

Attributes
This option enables the user to specify how a particular file in the project is to be installed. This includes the directory, whether an icon is to be created and overwrite/usage checks.
See File Attributes Dialog

Delete
This option enables a file to be deleted from the list of files in the application main window.

Move Up
Move Down
Enables the user to move a file up and down the list on the project to change the order of the files.
This option only handles the currently selected file. It cannot handle multiple selected files.

Encrypt/Un-encrypt Script
Enables the user to encrypt or un-encrypt manually created script files.

Exit
Exits the Setup Builder application. You will be asked to save the project if it has been changed in any way since it was last saved.

Edit Menu
Cut
Cuts the selected files from the list of files in the main window to the application clipboard

Copy
Copies the selected files from the list of files in the main window to the application clipboard

Paste
Pastes files from the application clipboard into the list of files in the main window

Select All
Selects all files in the application main window

Unselect All
Unselects all files in the application main window

View Menu
Toolbar
Toggles on and off the display of the tool bar. This is the row of icons at the top of the Setup Builder main window.

Status Bar
Toggles on and of the display of the status bar. This is the bar at the bottom of the Setup Builder main window which displays various messages about progress of the currently selected facility within Setup Builder.

Full File Details
Toggles on and off displaying of file details in the Project Files window. File details are the file size, date, time and attributes.

Install Descriptions
Toggles on and of displaying of descriptions next to each file which show the settings about how a file is to be installed.

Project Menu
Edit
Enables the user to select files for inclusion in an installation disk suite.
See Edit Project Dialog

Bitmaps
Enables the user to select bitmap files for inclusion in the project to be used in user defined objects.
See Project Bitmaps Dialog

Import VB Project
Enables a Visual Basic project to be imported into the current Setup Builder project. The import process includes all files listed in the VB .MAK file into the Setup Builder project.
See Import VB Project Dialog

Import 3rd Party Components
Enables third party components to be imported into the current project. Third party components are those such as run-time dynamic link libraries and their support files.
See Import 3rd Party Components Dialog

Backdrop Display
Enables the user to configure/maintain the backdrop display for an installation
See Backdrop Display Dialog

Program Manager Group

Enables the user to specify how the setup procedure should make a Windows Program Manager group.
See Program Manager Group Dialog

Attributes
Enables the user to specify attributes about the project.
See Project Attributes Dialog

Dialog Text
Enables the user to specify the text in each of the dialogs which will appear during an installation procedure.
See Project Dialogs Dialog

User Code
Enables the user to enter user-defined code at various stages of a setup procedure.
See User Defined Code Dialog

Installations
Enables the user to configure and maintain user-defined optional installations.
See Installations Dialog

Build Setup Procedure
Builds the currently open project.

Stop Build
Enables the user to stop the build process once it has started.

Test Setup Procedure
Tests the currently open project by running it.

Object
View Objects
Opens the Project Objects window to display all the objects in the current project.

New Object
Enables the user to create a new object.

Object Properties
Enables the user to change the properties of an object.

Delete Object
Enables the user to remove an object from the project.

Object Controls
Enables the user to maintain the controls to be displayed in the object.

Import Object
Enables the user to import an existing object file into the current project.

Help Menu
Index
Shows the main index for help on Setup Builder.

Search for Help On
Enables the user to search for help on a specific topic.

Using Help
Provides help on how to use the help system.

Setup Script Help
Displays the Setup Script language help file.

About Setup Builder
Displays version and copyright information about this version of Setup Builder.

Multiple Directory Installations

Setup Builder supports Multiple Directory Installations (MDI's). MDI's are typically installations where there is a main application directory and one or more subdirectories beneath the main directory. Setup Builder's own installation procedure (which is supplied as the sample file SETUP.SPJ) is an example of such an installation.

Setting up a Multiple Directory Installation
Setup Builder supports MDI's automatically, although you do need to perform a few activities to set them up.

Firstly, you must configure the directories which are to be created. To do this, select the Project/Directories menu option to obtain the <u>Directories dialog</u>. This dialog enables you to define all the directories you require to be created by your installation procedure.
At install time, all of the directories in the dialog will be created in the order in which they appear in the list.

Getting files copied into the subdirectories
Having now adjusted the installation to create the appropriate directories, you will want files in your installation to be copied into them.
With Setup Builder this is very simple. In the <u>File Attributes Window</u> for each file being installed, you must specify the directory in which a file is to be installed. By default, all files are copied into %InstallPath% which is the main application directory. To have a file copied into a subdirectory, simply modify the **Target Location** field to include a subdirectory name, for example:

%InstallPath%SAMPLES\

Note that the trailing \ **must** be entered otherwise you will get run-time errors at install time which inform you that the file cannot be copied. Likewise, the %InstallPath% variable **always** ends in \ therefore you must not precede the subdirectory name with a \ character.
The **Target Location** drop-down list will show all the directories which you have previously created in the <u>Directories dialog</u> plus a few other valid directory names. Any of these directories may be selected.

Setting up De-Installation for an MDI
The above is all that is required to create an MDI, however, if your installation includes a de-installation procedure, there are a few considerations to be taken into account when creating the installation procedure.
Setup Builder de-installs will always ensure that any files which it installed are de-installed, regardless of what directory they were placed in (subject to whether a file is marked as shared in the shared file registry). Setup Builder deinstalls will not remove any files that your program(s) have created since the installation, so you must write Application De-Installation script code to delete these. Setup Builder will therefore automatically ensure that any files you install in a subdirectory are correctly deleted in the de-installation.
Setup Builder will also automatically remove any subdirectories set up in the <u>Directories dialog</u>. It removes the directories in the reverse order in which they were created. Therefore, when you build the list of directories to create you must ensure that root directories are created before subdirectories so that the de-install correctly removes subdirectories before attempting to remove higher level directories.

Notes
The number of directories you can created is described in <u>Application Limitations</u>.
If you require more directories to be created, then you need to use the technique used in older versions of Setup Builder whereby manual script is entered in the <u>User Code dialog</u> Pre-File Copying Code section to create directories and in the Application De-Installation section to remove directories.

Summary

To create an MDI, simply use the <u>Directories dialog</u> to set up all the directories in the order in which they are to be created and then set the appropriate directory in the **Target Location** field of each file in the <u>File Attributes Window</u>.

If your application includes a de-install procedure, make sure you write script code to remove any files which the application might create once it has been installed otherwise a de-install might not fully complete. There won't be a run-time error, but files and directories may not be properly removed.

See Also
<u>Sample Projects/Files</u>, <u>User Code Dialog</u>, <u>File Attributes Window</u>

Multiple Program Installations

Setup Builder supports installations which contain more than one executable and therefore, more than one Program Manager icon. Setup Builder's own installation procedure (which is supplied as the sample file SETUP.SPJ) is an example of such an installation.

Setup Builder installations require that you specify the name of your application's .INI file in the Project Attributes Dialog. Setup Builder does not actually have to use the .INI file used by your application, but it does require a .INI file in order that the de-installation procedure will work. When installing a single executable, it makes sense to use that program's .INI file as the file to store the de-install information for tidiness, however, where several executables are being installed, this does not necessarily make sense.
When installing several executables, you need to specify a .INI file that Setup Builder can use. It can be the .INI file of any of the executables your are installing or even a completely unrelated .INI file.
Note that whichever file you use, Setup Builder will automatically remove it during the de-installation, however, the de-install will not remove any .INI files which your executables create: you must manually do this using the   User Code Dialog, Application De-Installation section.
NOTE: Setup Builder installations automatically use the Windows 95/NT registry if an install is performed using the 32-bit version of the interpreter.

Creating multiple icons is very simple with Setup Builder: simply place 'Yes' in the **Make Icon** field of the File Attributes Window for each file which you require to have an icon.

See Also
Sample Projects/Files, User Code Dialog, File Attributes Window

New Object Dialog / Modify Object Dialog

Purpose
The **New Object Dialog** enables the user to create a new object in the project and the **Modify Object Dialog** enables the properties of an existing object to be maintained.

Available from
- Object/New Object menu option
- Object/Object Properties menu option

Comments
The dialog contains the following fields:

File Name
This is the name of the file to save the object definition in.

Object Name
This is the name of the object. Object names must be unique within a project and must be unique across all objects. You cannot have a Backdrop and a Cue Card with the same name for example.

Object Type
This is the type of the object. Once an object has been created you cannot change it's type as this could invalidate some of the controls you may have created in the object.
The valid object types are:

- Backdrop
- Dialog
- Cue Card

Border Style
Only applicable to Cue Card objects, this field specifies how the border of the Cue Card is to be displayed. Available options are:

- None
- Black Line
- 3d Sunken
- 3d Bevel

Caption
This field is only use by Dialog objects and specifies the text to be displayed in the caption of the dialog.

Positioning
Applicable to Cue Cards and Dialogs, this field specifies the positioning of the object. This may be:

- None
- Centre Horizontally
- Centre Vertically
- Centre Both

If the second option is specified, the Y coordinate of the object is still used to control vertical positioning. Likewise, if the third option is selected, the X coordinate still applies.

Coordinates
These specify the position and size of the object. For Backdrops and Cue Cards, these positions/sizes are in screen pixels whereas Dialogs they are in Dialog Units.

Press the 'Preview' button to obtain a preview of how the object will appear.
Press the 'Ok' button to save the changes or 'Cancel' to exit without saving any changes.

See Also
Application Limitations

There is no directory to be modified Error

This error occurs when an attempt is made to modify a directory name when no directory name has been selected for modification.

There is no option to be modified Error

This error occurs when an attempt is made to modify an optional installation name when no installation name has been selected for modification.

<Not Found>

This message appears in the <u>Project Files Window</u> to indicate that the file listed cannot be found.

Possible causes:
The file next to the message has been deleted since the project was created

The project .SPJ file or the listed file has been moved to another directory

Solutions
Reinstate the file or remove it from the project

Number out of range Error

A number has been entered which is out of the range which is expected. The error message indicates the expected range.

Object Controls Dialog

Purpose
The **Object Controls Dialog** enables the user to maintain the controls to be included on an object.

Available from
▪   Object/Object Controls menu option

Comments
The dialog contains the following fields:

Available Controls
This list displays all the controls valid for the object being maintained.
Double click on an item in this list or press the 'Add' button to add it to the Selected Controls list.
Valid control types are:

▪   Text   Static text control
▪   Edit   Edit field
▪   Groupbox   Group border
▪   Icon   Icon
▪   Bitmap   Bitmap
▪   CheckBox   Check Box
▪   RadioButton   Radio Button
▪   Button   Standard push button
▪   Colour Block   Coloured area (for lines etc)

Selected Controls
This list displays all the controls currently in the object being maintained.
Click on a control in this list to obtain the properties in the Control Properties fields.
Press 'Delete' to remove a control from the list of selected controls in the object.
Press 'Up' and 'Down' to move a control up or down the list.

Control Properties
These fields allow each property of a control to be changed.
Double click on a property to edit it in the top field otherwise place the mouse cursor in the the field to change the property.
Property values are automatically saved when another property or control is selected.

Available properties are (actual properties shown depend on the currently selected control type):

Bitmap
Specifies the name of the bitmap in the project to display on a bitmap control. Bitmaps are added to the project using the Project Bitmaps dialog.

Border
Specifies whether the control has a border.

Caption
Specifies the text to display on a text field, button etc. The caption may contain embedded variable names using the %..% notation or the delayed substitution #..# notation. Please see the Setup Script help file section 'Standards and Notations' for more information on embedded variables.

Case
Specifies the text case to force an edit field to.

Control ID
Required by input controls such as edits, push buttons, check and radio buttons, the property specifies a value which must be unique within a dialog object. The value should be greater than 20 as values below 20 are reserved for values such as IDOK, IDCANCEL and IDBACK which you may also place in this property. You can place a numeric value in this property or standard values such as IDOK, IDCANCEL and IDBACK.

Edit Length
Specifies the maximum number of characters allowed to be entered in an edit control.

Font Bold
Specifies whether a text control should be displayed using a bold fold

Font Name
Specifies the name of the font to use when displaying a text control

Font Size
Specifies the size of the font to use when displaying a text control.

Foreground Colour
This property specifies the colour to use to display text in.

Horizontal Position
Vertical position
Used by bitmaps, these properties specify the positioning of a bitmap within its parent object.

Icon Name
Specifies the icon to display for an icon control.

Shadow Offset
Text controls can be given a shadow which is simply the text itself drawn in a different colour but positioned offset from the normal text itself. This property specifies the offset in pixels. By default, the offset is zero (no shadow).

Text Alignment
Specifies the alignment of text within a control.

Shadow Colour
Specifies the colour to use for displaying the shadow of text.

Tile
Specifies whether a bitmap should be tiled within its parent object. If this property is set, it overrides all other settings which affect the positioning of a bitmap.

Variable
For input controls such as edits and check/radio buttons, this property specifies the variable name to save the control's selection/value to for the rest of your script to pick it up. This field should contain a variable name including its leading and trailing % characters. It must not contain a variable which uses the #..# delayed substitution notation.

X Coordinate
Y Coordinate
Width
Height
These properties specify the position and size of the control.

Press 'Preview' to obtain a preview of how the object will appear.

Press 'Ok' to save the changes to the object or press 'Cancel' to exit the dialog without saving any changes to it.

An object with this name already exists Error

This error occurs when a new object is created with or the name of an existing object is changed to the same name as an object which already exists in the project.
The error can also occur when importing an object file into the current project.

You cannot have two objects with the same name regardless of whether they are different types of object.

Failed to find object file Error

Setup Builder has attempted to open the object file listed in the error message and has failed to find or open it.

Possible causes
The object file has been manually deleted

The object file has been moved

The object file has been renamed

This object file already exists in the project Error

An attempt has been made to import an object file into the currently open project and Setup Builder has identified that the object file already exists in the project.
You cannot add an object file more than once to a project.

OLE Control Registration

Setup Builder supports OLE control registration in a number of ways.
All OLE controls and their supporting files must be added to the project in the normal way for all files to be installed.
Registration is performed after all files have been copied.

Self-Registering OLE Executables
Some OLE controls are written as executable (.EXE) files which can self-register themselves, normally by running the executable with the /REGSERVER parameter. To have a self-registering executable register itself, simply select the 'OLE Registration' property in the File Attributes window for the required file.
Controls registered with /REGSERVER can be unregistered using /UNREGSERVER.

Self-Registering OLE DLLs
Other OLE controls such as .OCX files cannot be run as executables and therefore must be registered using the DllSelfRegister option in the 'OLE Registration' property of the File Attributes window.
Self-registering DLLs are registered using the **DllSelfRegister** function and unregistered using the **DllSelfUnRegister** function.

.REG Files
It is also possible to register OLE controls using a .REG file. Setup Builder does not automate this process for you but you can write user-defined script to do this. Typically, you would create a .REG file (Documentation can be obtained from Microsoft books, the SDK help (See 'Registration Database') and many other sources) and write some script in the 'Post File Copying' section of a project:

WinExec("REGEDIT.EXE /s %InstallPath%MYFILE.REG", 0, 1)
IF %ERROR% < 32 MessageBox("Error: WinExec return error code %ERROR% and was unable to register the .REG file.", "%Caption%", MB_OK, MB_ICONEXCLAMATION)

Note that the .REG file must be included as one of your project files so that it is installed on the hard disk at install time.
.REG files will be supported in future versions of Setup Builder.

Other Methods
OLE DLL controls can also be registered using REGSVR.EXE and REGSVR32.EXE which are both supplied with Visual C++ and probably many other development tools.
Setup Builder does not provide automated support for this technique either, but it can be easily implemented using WinExec as for .REG files.

REGSVR takes the following parameters:

/u   Unregister
/s   Silent mode
Filename

REGSVR [/u][/s] filename.ocx

Failed to open the selected file Error

This error occurs when an attempt has been made to open a file, normally for encryption.
The error means that the file either does not exist or it cannot be opened for reading and writing

Optional Installation Files Dialog

Purpose
The **Optional Installation Files Dialog** enables the user to build a project and have the ability to include or omit files as required. This means that you don't need to maintain multiple projects if the only difference between them is that they install a few slighly different files.

Available from
- Project/Build menu option when some files in the project are flagged as being optionally included on install disks.

Comments
Note that this dialog does not appear when Project/Build Setup Procedure is selected if no files in the project have been marked for optional inclusion.

The dialog contains to list boxes:
The top list shows all files in the project which may be optionally included on your setup suite disks, but are currently not selected for inclusion.
The lower list shows all files in the project which may be optionally included on your setup suite disks and are currently selected for inclusion.

The 'Add All' button will move all files in the top list to the bottom list - ie select them all for inclusion on your setup disks.
The 'Remove All' button will move all files in the bottom list to the top list - ie omit them all from inclusion on your setup disks.

Press the 'Ok' button to save the changes or 'Cancel' to exit and stop the build process.

Pressing the 'Ok' button will then cause the <u>Build Project Dialog</u> to appear.

Other Setup Builder Users

Setup Builder is also used by the following organisations:

- Intel Corporation

- Texas State Auditors Office

- Cinema Magnetique Communication SA, Paris

- FSI GmbH, Kuessaberg, Germany

- Idaho State Library, USA

- Moore Data Management, Minneapolis, USA

- Dun & Bradstreet SA, Mexico

- British Royal Air Force

- Fujitsu Australia, Melbourne, Victoria

- American Airlines

- UK Nirex Limited, Harwell, England

- Canada Life, Co. Dublin, Ireland

- Philips Dictation Systems, Vienna, Austria

- University of Nottingham, England

- BP Oil Deutschland GmbH

- DataCard Canada Inc, Mississauga

- Airbus Industrie, Blagnac, France

- AEGON Inc, USA

- Pitney Bowes, Heppenheim, Germany

- British Steel PLC, South Humberside, England

- Reuter Nederland BV., Amsterdam

- Queensland Transport, Brisbane, Australia

Build Output Window

Purpose
The **Build Output Window** shows the progress and activity of building a setup procedure.

Available From
The window appears automatically when Setup Builder starts building a project.

Comments
The window may be closed at any time as Setup Builder will recreate it whenever it needs the window.

Payment for and Registration of Software

If you wish to continue to make use of the Setup Builder software you are expected to pay a registration fee.

The registration fee for the Setup Builder software may be found in the <u>Software Price List</u> section of this help file.

Compuserve also has a facility for payment for shareware software which is ideal for overseas users. Essentially, a shareware author registers their software in a database and then the facility transfers payment from one user's account to another.
You can pay for Setup Builder via this facility via (GO) SWREG at any time. Select the 'Register Shareware' option. The registration number of this software is 4153.

Your registration fee entitles you to use this software on a single computer for an unlimited period and to make as many copies as you wish for backup purposes.

Upon receipt of your registration fee you will receive a disk (mail orders only) containing the latest version of the software and advice on how to remove the 'shareware warning' dialog, stop the one month disabling feature and release the various shareware restrictions in the software.
You will also receive notice of updates to the software.
Free technical support is available to registered users.

See Also: <u>Contacting the Author</u>, <u>What is Shareware ? - Description and Disclaimer</u>, <u>Software Price List</u>

Program Manager Group Dialog

Purpose
The **Program Manager Group Dialog** enables the user to specify how a Windows Program Manager group is to be created after installation of all the files in the install suite.

Available from
- Project/Program Manager Group menu option
- 'PM GRP' toolbar button

Comments
The dialog contains the following fields:

Program Manager Group
This enables the user to specify how the program manager group will be created. It may be:

'No Group' in which case no Group file will be created
'Make Group' in which case a new group will be created.
'Use Existing Group File' in which case a new group will be created from an existing group file which you may have specified as a file to be installed.

Give User Option of building Group
Check this field if you wish the setup procedure to give the user the option of creating a group. If this is not checked and 'Make Group' above has be selected, the group will be created without giving the user any option.

Group Caption/File Name
If 'Make Group' is selected above, you should place the caption text of the new group in this field. If 'Use Existing Group File' is selected above, you should enter the file name of the group file to be used. It may contain variables (eg %InstallPath%)

Group File Protection
A group file may be write protected. This option enables you to protect the group file after installation so that noone can change it or to unprotect it before a re-installation.

Press 'Ok' to save any changes, or 'Cancel' to exit without saving any changes.

Failed to create the temporary preview script file Error

Setup Builder has attempted to create a temporary script file in order to perform an object preview but was unable to create the preview script file.

Possible causes
The directory in which Setup Builder creates its temporary files is specified in the following section in the BUILDER.INI file:

[Install]
TempPath=

The above error occurs when the TempPath points to a non-existant directory or the directory is full or write-protected.
Setup Builder must have write access to the temporary directory to be able to perform object previews.
It also uses the directory for several other temporary files.

Save Project As Dialog

Purpose
The **Save Project As Dialog** enables a project file to be saved under another name.

Available from
- File/Save As menu option

Comments
Select the drive/directory as required by clicking on the appropriate fields. Enter the new name of the project and press 'Ok' to save it.

Note: Saving a project under a new name will release the old file so that another user on a network can open it.

Project Attributes Dialog

Purpose
The **Project Attributes Dialog** enables the user to specify some basic attributes about the way in which the project is to operate during the installation process.

Available from
- Project/Attributes menu option
- Seventh toolbar button

Comments
The dialog contains the following fields:

Application Name
This field will be the contents of the %Application% variable during project installation. Enter the name of the application in this field. (eg Setup Builder) It is also used to form part of the key in 32-bit installations which is the section of the registry where items are stored relating to the application being installed.

Company Name
This value is only used when values are written to the Windows registry, normally only under 32-bit installations.
The value is used to specify the section of the registry in which to place entries relating to the application being installed.
Application entries are normally placed in the HKEY_CURRENT_USER section of the registry, Software\CompanyName\ApplicationName area. Various entries will be placed under this such as the directory of where the software is installed and the name of the de-install script (if appropriate).

Comments
These are comments you wish to place at the top of the Setup script file and each line should be preceded with //
Normally you would place a message here of what the Setup procedure is going to install - the name of the software/version number/date etc.

Display 'Welcome' Dialog
To be 'friendly' to the user, an installation procedure can display a welcome dialog. This field enables you to specify whether this is required.

Prompt for Installation Path
The default installation path (below) is the normal directory in which your application would be installed.
Using the prompt field, you may also specify whether you wish to give the user the choice of changing this at install time - it is best to do so.
The text typed in this field will form the %InstallPath% and %InstallDrive% variables.
Note that this dialog will not appear during an installation if Setup identifies that the application has already been installed. You must de-install the application first then install it again to be able to place the application in a different directory.
Setup will not allow you to install two versions of an application in different directories

Display 'Licensing' Dialog
If you wish to license your software to a specific user and/or company, you may use this option to display a dialog which will ask for this information.
The text entered will be written to a .INI file (16-bit interpreter) which is specified in the 'Licensing .INI file' field below or the registry (32-bit interpreter) section specified in the 'Company Name' field above. (HKEY_CURRENT_USER\Software\CompanyName\ApplicationName\Licensing section.)
Suggestion: Licensing can be implemented in your software by reading the .INI file to display the text in the About dialog of your application.

Licensing information is placed in the Licensing .INI file section:

[License]
UserName=Graham Plowman
Company=My Company Name Ltd etc

De-Installation Option
Setup Builder can give the user the ability to provide the user with the option to install or de-install a piece of software.
Check this field to give the user the ability to de-install your application.
Note that Setup Builder automatically creates the script code to de-install your application. This code deletes all files installed (subject to the Registry), the application .INI file, the application directory and any registry entries created (32-bit interpreter). You can add extra code in the 'Application De-Installation' section of the <u>User Code Dialog</u> to perform extra special de-installation such as removal of directories underneath the application directory which were created at install time.
The De-Installation option causes the installation procedure to create an extra icon in the Program Manager group which Un-Installs the application.

Diskette Label Prefix
This is the first 6 characters of the label to be placed on each diskette of the installation disk suite. The installation procedure uses this to ascertain whether the user has inserted the correct diskette in the drive.
The disks are numbered: LABEL01, LABEL02 etc where LABEL is the text you enter in this field.

Script File Name
The default is usually sufficient and must be used for installation diskette suites.
See the <u>Technical notes</u> for more information on this field and how a Setup procedure works.

Encript Script File
Script files may be encripted. Normally they are ASCII human-readable files, but if you require some security so that they are not readable, this field gives that option.

Default Installation Path
The default installation path is the directory in which your application will be installed. Even if you do not enable the user to change this using the 'prompt for installation path' above, you must specify this field.
The installation procedure will create this directory if it does not exist, but will proceed normally if it does exist.

Reserve Space
When an installation disk suite has been completed, Setup Builder writes the appropriate script file on the first diskette. This field enables you to specify how much space to reserve for this file.
See <u>Technical Notes</u> **Setup Script diskette space** for more information on this field.
Due to the increased functionality of Setup Builder version 5.00, you will need to increase this value for your existing projects. The default for new projects has also been increased.

Extra Space Required
This is the amount of extra disk space required for your software to be installed on a machine (eg for manually copied files or space used by User Defined script code). Setup Builder automatically calculates the amount required for your files, but this option enables you to allocate extra space if required.
Setup Builder calculates the space required by adding the size of all the files in your project together and then adding an extra margin for de-compression temporary files (normally the size of your largest compressed file).
Since Setup Builder allows multiple installations, it calculates the space required for each installation and then adds the extra space to the result.

Split Files Larger than
In order to make efficient use of diskettes in an install disk suite, Setup Builder will split large files across one or more diskettes. A large file is considered as any file that won't fit on a diskette either because the diskette is full or the file is larger than the diskette size.
In some situations, you may not want files to be split. In this case you might want a file to simply be placed on the next diskette.
This field enables the user to specify the 'threshold' file size: given a file which will not fit on a diskette, files smaller than the threshold will be placed on a completely new diskette. Those larger than the threshold will be split across diskettes.
It is recommended that the threshold is between 50 and 100K. It is not worth splitting files smaller than this.

Application .INI file
This is the name of the .INI file to which to write the licensing information entered in the 'Licensing' dialog. If the 'Licensing' dialog has been selected you should enter this field. If you do not, the name will default to the %Application% name .INI file. For example, if %Application% held 'test' then %IniFile% would hold 'test.ini'. However, this will fail if %Application% holds multiple words or is longer than 8 characters so it is best to set the Application .INI file field if you wish to implement licensing!
The Application .INI file is also used to store installation information of where an application was installed. This is so that the de-install process knows where to de-install the application from.

Copy setup.exe to diskette
Check this field to have the build process place setup.exe on the first diskette.
This is the program the user can run to initiate the installation suite.
You must check this field or the install.exe field or both. You cannot leave both unchecked.

Copy install.exe to diskette
Check this field to have the build process place install.exe on the first diskette
This is the program the user can run to initiate the installation suite.
You must check this field or the setup.exe field or both. You cannot leave both unchecked.

Copy lzexpand.exe to diskette
Check this field to have the build process place lzexpand.exe on the first diskette.
This file will automatically upgrade older versions found for the de-compression process.

Copy ctrl3dv2.dll to diskette
Check this field to have the build process place ctrl3dv2.dll on the first diskette.
If this file already exists on the target machine it will not be installed at all. It is only installed if it doesn't already exist on the target machine.

Copy 16-bit run-time
Copy 32-bit run-time
These two options enable you to specify which target platforms you wish to create your installation procedure for.
At least one of the two options must be checked. If non are checked, the 16-bit option will automatically be selected.
Please see the help section 16 and 32 bit installations for more information on how these options affect an installation.

Press 'Ok' to save any changes, or 'Cancel' to exit without saving any changes.

Project Bitmaps Dialog

Purpose
The **Project Bitmaps Dialog** enables you to select all of the files which you require to be included in your project for use by user defined objects such as backdrops, cue cards and dialogs.

Available from
- Project/Bitmaps menu option

Comments
The top-left list box shows all of the files in the current directory - the current directory is displayed above the list box.
To include a file in the project, double-click on the required file and it will appear in the list box at the bottom left.
Note that Setup Builder will not allow you to include a file twice.

The top-right list box shows the current drives and directories. Click on these as required to change drives/directories.

The bottom-left list box shows all of the files which have been selected for inclusion in the project. To remove a file from the project, double-click on the file in this list.

The 'Add All' button will add all of the files shown in the top-left list to the bottom-left list, whereas 'Remove All' does the opposite.

Press the 'Ok' button to save the list, or 'Cancel' to exit without saving the list.

See Also
Application Limitations

Directories Dialog

Purpose
The **Directories Dialog** enables you to maintain the names of directories to be created or removed by an installation/deinstallation procedure.

Available from
- Project/Directories menu option

Comments
The list at the bottom of the dialog shows all the directories which are currently configured. These directories will be created by an installation procedure and removed by a deinstallation procedure.
Note that a directory cannot be included in the list twice.

Directory names may contain variables but they should not include trailing \ characters. Some example valid directory names are:

%InstallPath%
%InstallPath%SAMPLES
%InstallPath%HELP

The first directory name is always required by an installation procedure. It cannot be moved, deleted or modified for that reason.
The directory names should appear in the order in which they are to be created. For example, a root directory should be created, followed by its subdirectories and then any further subdirectories to them. While the install procedure is not affected if you do not put the files in such an order because the underlying script function performs a 'multi-level directory create', the de-install procedure is affected because it removes one directory at a time starting at the end of the list and working backwards. The underlying script function for removing directories does not perform a 'multi-level remove' for safety reasons.

To add a new directory to the list, enter the name of the required directory in the edit field and press **New** to save it to the list.

To edit a directory, double click on it in the list so that it moves to the edit field. Adjust the text as required and press **Modify** to save it back to the list.

To delete a directory from the list, select it in the list and press **Delete**. You will be requested to confirm deleting the directory.

Press the 'Ok' button to save the directories, or 'Cancel' to exit without saving them.

See Also
Application Limitations

Project Dialog Text dialog

Purpose
The **Project Dialogs Text dialog** enables the user to specify the text in each of the dialogs which will appear during an installation procedure.

Available from
- Project/Dialogs menu option
- 'DLG TEXT' toolbar button

Comments
You should find the defaults sufficient.

You may place the following embeded variable names in any of the fields:

%Application% for the application name
%Caption% for the normal dialog caption (unlikely to be useful)
%InstallPath% for the directory name where the application is to be installed
%InstallDrive% for the drive letter where the appliaction is to be installed

The 'Language' field is used to control which language the installation procedure will be created in. An install procedure is currently built in one language: it does not change language at run time to suite the user. If you require installations for different languages, you need to build an installation for each language.
In future versions of Setup Builder, it is planned to allow an install to automatically select the language to display at run-time.
Press the 'Defaults' button to set the text for all the fields to their default values.
Changing the language and pressing 'Defaults' will cause the fields to be set to the defaults for the selected language.

Press 'Ok' to save any changes, or 'Cancel' to exit without saving any changes.

Project Files Window

The **Project Files Window** displays all of the files currently in the project. The window also provides the primary source for selecting files and setting their installation attributes.

Available From
 The window is automatically created when Setup Builder starts up and it cannot be closed.

Comments
The window always shows all of the file names in the current project. If there are no files, the list will be blank.
The View/Full File Details menu option affects whether file sizes, dates, times and attributes are shown in the list.
The View/Install Descriptions menu option affects whether a brief description of the properties set for each file is shown in the list..

By double clicking on a file, it may be selected into the File Attributes so that its properties may be set.
By holding down the Ctrl key, multiple files may be selected at the same time in order that the same property may be set for all the selected files.

The Up and Down toolbar buttons may be used to move a single file up and down the list so that you can change the order in which files appear in the list.

All of the Edit menu options perform their actions on the **Project Files Window**. You can cut out files from the list and paste them to different parts of the list.

Files may be deleted from the project either by using the Edit/Cut menu option, the File/Delete menu option or the Delete key.

Import VB Project Dialog

Purpose
The **Import VB Project Dialog** enables a Visual Basic .MAK project file to be imported into a Setup Builder project.

Available from
- Import VB button on <u>Project/Edit</u> dialog

Comments
Select the drive/directory by clicking on the appropriate fields. Enter the name of the required Visual Basic project and press 'Ok' to import it.

New Project Dialog

Purpose
The **New Project Dialog** enables a new project file to be created.

Available from
- File/New menu option
- First toolbar button

Comments
Select the drive/directory as required by clicking on the appropriate fields. Enter the name of the new project and press 'Ok' to create it.

Note: If you currently have a project open which has been changed since it was last saved, you will be asked if you wish to save it before a new project is created.

A newly created project is automatically locked such that another user cannot open the project or overwrite it in a network situation.

Project Objects Window

Purpose
The **Project Objects Window** displays a list of all the <u>User Defined Objects</u> included in the currently open project.

Available From
The window is opened using the Object/View Objects menu option.

Comments
The window shows three columns of information:

- The object type
- The object name
- The file containing the object

All of the Object menu options perform their actions on the **Project Objects Window**.
The Object/Object Properties menu option enables the user to maintain the properties of the currently selected object.
The Object/Controls menu option enables the user to maintain all the controls on a user defined object.
The Object/Delete Object menu option deletes the currently selected object from the project. It does not delete the physical object file. Similarly, an object can also be deleted by pressing the Delete key.

The **Project Objects Window** can be closed at any time by double clicking on it's system menu icon.

Open Project Dialog

Purpose
The **Open Project Dialog** enables an existing project file on the disk to be opened.

Available from
- File/Open menu option
- Second toolbar button

Comments
Select the drive/directory by clicking on the appropriate fields. Enter the name of the required project and press 'Ok' to open it.

Note: If you currently have a project open which has been changed since it was last saved, you will be asked if you wish to save it before another project is opened.

Setup Builder keeps .SPJ project files open for the duration the user has the project 'open'. This is to maintain network awareness and stops two people updating the same project at the same time.

Quick Start to creating an Install Disk Suite

This topic should give you a good understanding of the process of creating an installation procedure.

Before you can install any software or create any installation diskettes, you need to decide what files you want to install on your client's machine and where those files will be installed on that machine. It is also a good idea to have some idea of how you want your installation procedure to look.
You can then proceed to the first step:

Creating a Project
In order to use Setup Builder to create any installation diskettes, you must create a project as the first step.
A project contains all the information about an installation suite necessary to build diskettes for an application. It include the names of the files and all the attributes you set or select.
To create a project, select the File/New menu option to obtain the New Project dialog. Enter the name of the file you wish your project to be called and press the **Ok** button. The new project will then be created and the Edit Project dialog will appear. This dialog enables you to select all the files you wish to be placed on your install diskettes. You should select
all .COM, .EXE, .DLL, .VBX, .OCX and data files etc required by your application.
Note that you do not need to include CTL3DV2.DLL as Setup will automatically install this if required. Setup requires this file in order to operate itself.
The Edit Project dialog is also available from the Project/Edit menu option, so it is possible to add more files to your project later if required.
Press **Ok**. Your project has now been created.

Setting File Attributes
Now that you have created your project and selected the files you want to install, you need to make selections about how you want the files to be installed, for example, the directory in which to install them and whether they will have icons on Program Manager.
To set the attributes of a file you can double click on it in the Setup Builder Project Files Window. Alternatively, you can also select the file in the Project Files Window and then select the File/Attributes menu option to obtain the File Attribute Window.
The File Attributes Window appears showing all of the property information which may be set for the selected file.
The Target Location is the directory in which the file is to be installed. By default, this appears as %InstallPath%. This represents a variable name used by the underlying Setup Script interpreter at run time. The variable holds the directory entered by the user in the 'Installation Path' dialog. It is not recommended that you 'hard code' the directory here because some users may not have the same drives that you have. You can however, place the file in a subdirectory of the application directory, so for example, if the user had entered C:\TEMP\ as the installation directory and you wanted the file to appear in C:\TEMP\WORK, then you would place %InstallPath%WORK\ in the Target Location field. The installation procedure automatically adds a trailing \ character to the path name entered by the user in the 'Installation Path' dialog ie the %InstallPath% variable.
The Copy Comment appears as a message on the screen when the file is being installed to tell the user what is being installed.
File Useage determines whether a file will be displayed in the progress gauge at install time. If it is not displayed, the file will not be copied and it becomes the users responsibility to copy the file from the diskette manually or the developers responsibility to manually write 'user code' in the Setup project to perform the action. This enables you to supply sample files on your install disks without actually installing them.
Confirmation determines what Setup should do at   install time if the file already exists on the target machine. This may be:

No overwrite confirmation - install the file regardless.
Confirm overwriting if file exists - If the file already exists, confirm overwriting it.
Confirm Overwriting newer file only - If the file exists, confirm overwriting it if the existing file is

newer than the one being installed. If the user says 'no', the file is not copied. If the new file is newer than the existing file, the file is copied and no confirmation occurs.

The overwrite message will appear when either of the 'Confirm Overwriting' options are selected and the file being installed has been found to already exist.
You can optionally compress files. By convention, compressed files have a different file extension. It is recommended that you use the default supplied, but you can change it if required. It must not be the same as the original file.
Optional File Inclusion enables you to maintain one project instead of several when you require more than one installation configuration - when each configuration contains different files. Making a file optional will cause a dialog to appear at 'Build' time for you to select the optional files to be included on the diskette.

The File Attributes dialog also enables you to specify whether a file should have an icon on Program Manager. Set the 'Make Icon' field to 'Yes' to   have an icon created.
The Icon Text will appear on the Program Manager group (discussed later on this page).
You can have parameters to the file, such as a file name if the icon is an executeable.
The icon file is normally left blank, but if supplied should refer to a .EXE file containing icons such as PROGMAN.EXE
If the icon file is specified, the the Icon Index should also be specified. Executeables contain multiple icons in a list - you see this in Program Manager when you select 'Change Icon'. The first is number zero, the second one and so on. Enter the number of the icon you require.
Icons may optionally have a Working Directory. The default as in Program Manager, is the directory of the executeable file.

It is also possible to set the read only attribute of files installed using the File Protection options. This feature is not often used, although an 'unprotect' before copy will ensure against an installation procedure failing due to read-only files.

Having set the above attributes for one file, you then need to repeat the exercise for all the other files in your project. If you have several files which have the same attributes (other than file name of course) you can select them all by single-clicking on them while holding down the Ctrl key and then select the File/Attributes menu option. In this situation, the files will all be set to have the same attributes as the last file you selected - the one with the dotted line around it in the list. Press **Ok** in the File Attributes Window to save the attributes to all the selected files.

Creating a Backdrop Display
This is a feature which is becoming very common and adds a very 'professional' looking touch to your installation - first impressions count.
The backdrop display appears on the screen from the start of an installation. When you installed Setup Builder, the backdrop was the blue shaded background.
Setup Builder always creates a default backdrop for you which caters for all screen resolutions including unrecognised resolutions, however, you can create your own backdrops and have different backdrops for different screen resolutions.
Backdrops in Setup Builder are implemented as User Defined Objects. You create a Backdrop Object which defines all the text, colours, lines and bitmaps to be displayed. Bitmaps must be added to the project using the Project Bitmaps Dialog before they can be assigned to controls within an object.
To set the backdrop objects for each screen resolution, select the Project/Backdrop Display menu option to obtain the Backdrop Display dialog.

Note that any bitmaps used in your backdrop will be automatically compressed and placed on the first diskette of your installation suite at 'Build' time, so you do not need to include them in your project.

Press **Ok** to save the backdrop information.

Creating a Program Manager Group
You can have your installation procedure create a Program Manager group when your software has been installed. Indeed you should do this if you have selected any of your files to have Program Manager icons.
To set up the group, select the Project/Program Manager Group to obtain the <u>Program Manager Group dialog</u>.
Click on 'Make Group' to have a group created. The title of the group will be that specified in the Group Caption field.
It is recommended that you give your customer the option of creating a group: check the 'Give user option' field. If they select 'No' at install time, no icons or Program Manager group will be created.
Please note that Setup Builder, unlike many other install utilities, will recognise if a program manager group has already been created for your application and it doesn't create another group with the same name if you install your software twice.
Similiarly, Setup Builder does not create duplicate icons in a group. The Group Caption and Icon Text are used as identifiers in order to determine this.
If you wish to supply a ready made group file in your installation procedure, check 'Use Existing Group File' and place the name of the file in the Group Caption/File Name field. In this situation, you can also use the Group File Protection options. They are not valid if you have selected 'Make Group' above. You will need to include the group file in your project and set its Target Location to %WindowsDirectory% in the <u>File Attributes dialog</u>.
Note that when icons are created they are created on the currently active Program manager Group. If you select 'No Group' the icons could appear on any group! Creating a group automatically makes the new group the current group so you do not need to worry about selecting the current group.
Group files and Group file protection is supported by the 32-bit interpreter, but they have no effect under Windows 95/NT because these operating systems do not use group files.

Press **Ok** to save the Program Manager Group information.

Setting the Application Name and Install Attributes
In order for an install suite to be created, a few settings must be set which describe the install script file.
Select the Project/Attributes menu option to obtain the <u>Project Attributes dialog</u>.
The application name is the name of your application, for example, Setup Builder. This text will be placed in the %Application% variable for use by the underlying script program.
You must enter your company name. This is used in 32-bit installations to form part of the registry entry name to which values for your application will be written.
Comments are script language comments you wish to appear at the top of the script file.
Normally, this will detail the application and version etc. To create new lines, press Ctrl+Enter. All non-blank lines must start with //

There are a number of default dialogs which the user can be prompted with during an installation. Normally, the 'Welcome' dialog and the 'Installation Path' dialog are used. The text shown in these dialogs will be discussed later on this page.

The default disk label prefix should be sufficient. The script file name must be SETUP.SCR for automated installations.
Set the Default Installation Path to the directory of where your software is to be installed. This will appear as the default in the 'Installation Path' dialog at install time.
The reserve space default and split files size should be sufficient. Refer to the help on the <u>Project Attributes dialog</u> for more information on these.
The Licensing .INI file is used if you have selected to display the 'Licensing Dialog' or the de-install option is seelcted. This dialog requests the user to enter their name and company name and stores them in the .INI file specified in this field. By default the field contains %Application%.INI so be sure to set this field or keep the Application Name to 8 or less characters.
Under 32-bit installations, licensing information is placed in the Windows Registry.

Press **Ok** to save the project attributes information.

Setting Dialog Text
An installation procedure contains a number of standard dialogs throughout its progress. It is possible to set the text displayed in these dialogs by selecting the Project/Dialog Text menu option to obtain the Dialog Text dialog.

You should find the defaults sufficient, but if not you can change them. Note that you can embed %Application% in the text to refer to the application name or %InstallPath% for the installation directory.
Multiple lines can be specified by inserting | (vertical bar) characters to denote carriage returns. Installations may be created for different languages by selecting the required language in the **Language** field and pressing the **Defaults** button to select the default translations for that language.
You need to create one install disk suite for each language you require. Setup Builder installs do not automatically change language at run-time.

Press **Ok** to save the project dialog text.

Creating the Disks
If you have followed this page from the top, you should now be ready to create you installation diskettes.
Insert a formatted diskette into your diskette drive and press the 'Build' toolbar button or select the Project/Build menu option.
If you have marked some of your files as optional, the Optional Files dialog will appear to enable you to select the files to be placed on your diskette. Press **Ok** to continue or **Cancel** to stop at this stage.
The Build Project dialog will then appear to ask you which drive/directory to create your installation procedure. Enter the required location and press **Ok** to continue.
Press **Ok** again to continue.
Setup Builder will then proceed to 'clean' the requested directory of files and create your installation suite within it. It will automatically ask you for more diskettes if a diskette becomes full. Large files will automatically be split across multiple diskettes if required and files will also be compressed if this option has been selected.
Once all files have been copied, Setup Builder automatically creates an install script and may ask you for the first disk again since this file is always placed on the first diskette. (If the script file will not fit, you need to increase the reserve space mentioned above).
Setup Builder automatically places all the component files it needs for its own use on your diskettes, so you don't need to worry about this.

Afterwards...
Once you have created your diskettes and install suite, it is strongly recommended that you test it on a variety of machines, ideally ones that don't have your software or any of its components already installed. You can test the installation on your machine by pressing the 'Run' toolbar button in Setup Builder, or alternatively, the 'Setup' or 'Install' icon in the Setup Builder Program Manager group.
It is suggested that you use Setup Builder to create a 'Master Disk set' and then use File Manager to diskcopy these diskettes in order to duplicate them.
It is possible to copy install diskettes into one directory on a hard disk, a network or a CD-ROM provided there are no split files present. The install will operate as normal from a hard disk and it recognises that it doesn't need to ask for extra diskettes! This is possible because diskettes are not labelled using DOS labels. Each diskette contains a label file instead.
It is suggested that CD-ROM installations are created in a hard disk directory before thay are copied to a CD-ROM.

This concludes the Quick Start to creating an Install Disk Suite. By now you should have a good understanding of how to create an installation procedure using Setup Builder.

If you have any further questions, please don't hesitate to contact the author.

Shared File Registry

Often during an installation of some software, it is common to place files in the \WINDOWS or \WINDOWS\SYSTEM directory. Sometimes, the files placed in these directories are used by more than one application, for example, .DLL files. At some point, it is likely that one or more of the applications sharing these common files will be de-installed, causing a potential problem with files that have been installed for shared use, especially if the shared files are deleted.

To resolve the problem of shared files being de-installed, Setup Builder supports a 'shared file registry' system. In simple terms, the registry is just a list of files with a count of the number of applications using them. As applications are installed, the counts are increased and when applications are de-installed, the counts are decreased. When counts reach zero, files are deleted.

The registry itself is simply the REGISTRY.INI file in your \WINDOWS directory. The file contains a list of all the files which have been registered as shared, together with a count of the number of applications which are currently installed which use the files.

When a file is registered, it may already exist in the registry (another application is already using it), in which case its count is simply increased by one. If it isn't already registered, the file is added to the registry with a count of 1. Before a file is registered, the Register function checks to see if the file exists. In this way, files which have already been installed by other applications which are not in the registry can be accounted for in the registry, so it is possible that if you Register a file, its count may start at 2 if the file already existed.

To integrate with this facility, Setup Builder creates script code which registers a file before it is actually installed/copied.

The Setup Builder de-install facility integrates with the shared file registry.

When a piece of software is de-installed, files are usually simply deleted, however, files which have been registered in the registry must only be deleted if their share count reaches zero.

Therefore, shared files must be removed with the **UnRegister** function and not the **Delete** function. UnRegister retrieves the share count for a file and decreases it, writing it back to the registry. The file is not deleted until the share count reaches zero (ie all applications using the file have been de-installed) and then the registry entry for the file is removed - the file is no longer registered as being shared.

The registry functions do not support wild card file names.

Example
Register("C:\WINDOWS\SYSTEM\CTL3DV2.DLL")
' Install the file/overwrite check etc
STOP

' De-Install
UnRegister("C:\WINDOWS\SYSTEM\CTL3DV2.DLL")

Sample Projects/Files

Setup Builder has been supplied with some sample files demonstrating many of the features supported by the product.
It is suggested that you view the files in order to gain a greater understanding of how Setup Builder operates.
All of the samples may be found in the SAMPLES and OBJECTS directories which are created when you install Setup Builder.
The samples will only have been installed if you selected to install them when you installed Setup Builder. You can re-run the Setup Builder install procedure again to install just the samples if required.

The following sample objects/components are supplied:

- ASKPATH.DLG
- DEFAULT.BKD
- DEFAULT.CUE
- DEINSTAL.DLG
- DIALOG1.DLG
- DIALOG2.DLG
- DIAMOND.BMP
- LICENSE.DLG
- OKBOX..DLG
- PANEL.BMP
- REGISTER.CUE
- ROUNDPT.BMP
- SQUAREPT.BMP
- WELCOME.DLG

The following sample projects are supplied:

- SAMPLE1.SPJ
- SAMPLE2.SPJ
- SETUP.SPJ

The following sample scripts are supplied:

- SAMPLE3.SCR/.INI

It is also suggested that you review the SETUP.SCR file (supplied in the Setup Builder distribution .ZIP file and on the Setup Builder distribution disk) which is the install script to install Setup Builder. It has purposely not been encrypted so that you can view it with any text editor and gain an appreciation as to how a Setup Builder installation works.

Sample Objects/components Descriptions

ASKPATH.DLG
A sample dialog to replace the in-built 'ASKPATH' dialog with a Windows 95-look dialog.

DEFAULT.BKD
This is the default backdrop used by Setup Builder when a project is created. You can change it so that you have a standard 'Corporate Style' backdrop for all installations you supply.

DEFAULT.CUE
A sample default Cue Card definition.

DEINSTAL.DLG
A sample dialog to replace the in-built 'DEINSTALL' dialog with a Windows 95-look dialog.

DIALOG1.DLG
A shell Dialog object.

DIALOG2.DLG
A shell Dialog object.

DIAMOND.BMP
A bitmap of a small diamond. Designed to be used as a bullet point.

LICENSE.DLG
A sample dialog to replace the in-built 'LICENSE' dialog with a Windows 95-look dialog.

OKBOX..DLG
A sample dialog to replace the in-built 'OKBOX' dialog with a Windows 95-look dialog.

PANEL.BMP
The bitmap used in the above Windows 95-look User Defined Dialogs.

REGISTER.CUE
A sample Cue Card object providing advertising during an installation to encourage your customers to register your software.

ROUNDPT.BMP
A bitmap of a small round point. Designed to be used as a bullet point.

SQUAREPT.BMP
A bitmap of a small square point. Designed to be used as a bullet point.

WELCOME.DLG
A sample dialog to replace the in-built 'WELCOME' dialog with a Windows 95-look dialog.

Note that if you override the in-built dialogs with dialog objects such as those above, you also override the multi-language functionality provide by Setup Builder such that it no longer operates

Sample Project Descriptions

SAMPLE1.SPJ
This project demonstrates a simple project which installs notepad and calculator in the C:\TEST directory. It demonstrates most of the features you might use for a simple installation.

SAMPLE2.SPJ
This project is the same as SAMPLE1.SPJ except that it has all of the dialogs overidden with User Defined Dialogs to give an example of using Setup Builder to create a 'Windows 95 / Microsoft Setup Wizard look-alike' installation.

SETUP.SPJ
This is the project file used by GPP Software to create the Setup Builder installation procedure - it is the project which created the install procedure that you used to install the software. It is supplied as an example of how various Setup Builder settings can be used however, since not all of the files in the project are supplied on the distribution disk, is it not a buildable project - it is supplied for your reference only.

Sample Script Descriptions

SAMPLE3.SCR/.INI
This script and .INI file is an example of how the Setup Interpreter can be used as a Windows batch file interpreter.
This script provides a simple example whereby you place the script as an icon in your Windows 'Startup' group. When Windows starts, you will be given the option to run all/none/some of the programs which have been configured in SAMPLE3.INI.
Please read SAMPLE3.SCR as it is fully commented with a full description as to how it should be used.

Select Script File Dialog

Purpose
The **Select Script File Dialog** enables the user to browse for a script file to be encrypted or un-encrypted.

Available from
- **Browse** button on the <u>Encrypt/Unencrypt Script File Dialog</u>

Comments
Select the drive/directory by clicking on the appropriate fields. Enter the name of the required script file and press 'Ok' to select it.

Failed to create directory Error

This error occurs when Setup Builder has failed to create the directory which was specified for the location to create the new installation procedure.

Possible causes
An invalid directory name has been specified

An invalid drive letter was specified

An attempt has been made to create an installation procedure on a disk which is write-protected. Setup Builder MUST have read-write access to the target diskette

Technical Notes

File locking/sharing
It is advisable to install SHARE or VSHARE to maintain file locking on project files on your local drive(s). Network drives are normally configured for this anyway.

Windows Version
The Setup Builder software has been developed and tested under Windows 3.1, Windows 95 and Windows NT.

How the Setup Procedure operates
When a Setup procedure is run, the user executes SETUP.EXE, usually from a diskette. This program copies the files INST16.EXE or INST32.EXE (depending on which version of Windows is running) into the \windows\GPPSOFT directory. It also copies SETUP.SCR and any user defined object bitmaps to your \windows\TEMP directory (or wherever Windows is installed. The GPPSOFT and TEMP directories are created if they do not exist) which must have write access.
The INSTxx.EXE program is then run on your hard drive by the SETUP.EXE program and passed the SETUP.SCR file name as a parameter script file to be interpreted (run). This approach is used because diskettes may be changed during the installation process.
SETUP.EXE only recognises SETUP.SCR which is why your script file should always be named SETUP.SCR for an installation diskette suite.
However, if you create script files to be run from your hard disk via the File Manager Association with .SCR files, the script file may have any name since INSTxx.EXE will accept any file name as a parameter script file to be run.
At the end of running a script, the CLEANUP.EXE program is run from the last disk. This removes the setup script and backdrop bitmaps from your hard disk, leaving it tidy. If an installation is cancelled half way through a multiple-disk installation, you may be asked for the last disk before the installation terminates. This is so that CLEANUP.EXE can be found and run.
CLEANUP.EXE is also placed on the first disk of an installation so that if an installation is cancelled before it is even started, you will not be asked for the last disk. This enables you to quit an installation before file copying commences.

Setup Script diskette space
When Setup Builder creates the first diskette of an installation suite, it reserves a portion of the disk by creating a dummy script file. As the build proceeds, files are copied on to subsequent diskettes as required. At the end, Setup Builder creates a temporary copy of the Setup script file in the same directory as the project (.SPJ) file and then requests the first diskette again in order to copy the Script file onto it, overwriting the dummy file. This process is used because the script cannot be created until Setup Builder knows which disk each file has been copied to. This will not be until all files have been copied.
The amount of space reserved by default is 32K. Copying of the script file to the diskette may cause a diskette space error if there has not been enough diskette space reserved for the script file.
In this situation, you need to change the reserved space. To do this, check the size of the temporary copy of the script .SCR file which is created in the same directory as the .SPJ project file (using File Manager or similar product) and adjust the 'Reserve Space' field in the Project Attributes Dialog to show a number which represents the number of K Bytes you wish to reserve (default is 32, reserving 32 x 1024 bytes)

Program Manager Group file protection
Setup Builder offers the ability to set the write protection flag of a program manager group file to stop a user from changing the group or deleting items from it.
The default for group creation is to create a new group with same name as that of the application.
Protection of the group file uses the 'Group Caption/File Name' field in the Program Manager Group Dialog to obtain the name of the file to adjust. If this is the default (%Application%) and 'Make Group' is selected, then the name of the file obtained will be the value in the 'Application Name' field of the Project Attributes Dialog. In this case, the application name must be one word

and less than 8 characters long otherwise group file protection may not work.
Similarly, when Program Manager creates groups given a caption, it often appends a numeric to the caption to obtain a name which is different from an existing name. This means that Setup cannot guarantee successful implementation of group file protection when the group is not created from an existing file.
For best results where group file protection is required it is best to create the group from a file using 'Use existing group file' in the <u>Program Manager Group Dialog</u> and placing the actual file name of the group file in the 'Group Caption/File Name' field. The group file itself may be installed as part of your installation process either into your own directory (best) or the Windows directory (risk of overwriting something with the same same).
NOTE: Program Manager group files are only supported under Windows 3.x as Windows 95 and Windows NT no longer use them.

Backdrop Bitmaps
Backdrop bitmap files are copied by the script into the \windows\TEMP directory (or wherever Windows is installed) when the script first runs so that they may be used at any time during an install script. The script automatically deletes them just before it finishes.

Test Project Dialog

Purpose
The **Test Project Dialog** enables you to select the disk drive and directory from which you wish to run a setup disk suite in order to test it.

Available from
- Project/Test menu option
- Toolbar button

Comments
The default location is that which was used for the last build process. Enter a new location if required.

Press the 'Ok' button to select the drive/directory and proceed to run the setup process, or 'Cancel' to exit without running the process

Notes
If a single drive specification (eg a:) is entered, the directory used will be the root (and not the current directory on a:) since a '\' character is always appended to the directory entered by the user. If you want another directory, then it must be specified in full.

The maximum number of controls allowed in an object has been reached Error

This error occurs when an attempt is made to add a new control to an object and the maximum number of controls within an object has been reached.

See Also
Application Limitations

Cannot add any more directories to the project

This error occurs when an attempt is made to add a new directory and the maximum number of directories allowed has been reached.

See Also
Application Limitations

Too Many Files Error

Setup Builder has a limit of 25 files per project for the Shareware Edition and 1024 files per project for the Professional Edition.

This error occurs when the user has attempted to exceed this limit by adding more files.

Cannot add any more optional installations to the project Error

Setup Builder is unable to add another optional installation to the project..

Possible causes:
An attempt has been made to add too many optional installations to the project.

See Also
Application Limitations

Cannot add any more objects to the project Error

This error occurs when an attempt is made to create a new object and the maximum number of objects within a project has been reached.

See Also
Application Limitations

Unrecognised Project ID Error

Setup Builder project files have a file identifier encoded in them.

An attempt has been made to open a project file which has an unrecognised file identifier. This means that Setup Builder does not recognise the selected file as a Setup Builder project file. You should select another file or if the file really was a project file, it has probably become corrupted, in which case you should recreate the file.

Unrecognised Object File Error

Setup Builder object files have a file identifier encoded in them.

An attempt has been made to open an object file which has an unrecognised file identifier. This means that Setup Builder does not recognise the selected file as a Setup Builder object file. You should select another object file or if the file really was an object file, it has probably become corrupted, in which case you should recreate the object.

Unrecognised Object File version Error

This error occurs when Setup Builder has failed to recognise the file version number of an object file.

Possible causes
The object file has been corrupted - recreate it again

You are using an older version of Setup Builder to read a newer version of an object file which the version of Setup Builder you are using does not recognise

Unrecognised Project Version Error

Setup Builder project files have a file version number encoded in them in order that future versions of Setup Builder can recognise files for upgrade/conversion purposes.

An attempt has been made to open a project file which has an unrecognised version number. Setup Builder should only give this error when an attempt is made to open a file which is not a project file, but just happens to have the correct file identifier - a rare situation.
Setup Builder recognises all previous file versions and automatically converts them to the latest format for the program release when they are saved. This will make files non-backward compatible.

User Code Dialog

Purpose
The **User Code Dialog** enables the user to manually enter specific code at various stages of the installation process.

Available from
- Project/User Code menu option
- 'SCRIPT' toolbar button

Comments
You should refer to the Setup Script Help to see the commands and standards used.

The User Code dialog enables user-defined code to be entered at several stages of the installation procedure.
Select the required code from the drop down list at the top of the dialog:

Initialisation code
This code is executed immediately at the start of a script, just after the default variables have been set up. You might want to use this to determine whether your application has been previously installed and exit so that it is not installed again. You may even require a different installation procedure to be executed in this event.

Pre file copying code
This code is executed before any files are copied (in the gauge). You might wish to use this to delete any unwanted files for example.

Post file copying code
This code is executed after files have been copied (in the gauge). You may wish to use this to ask the user quaestions (via the MessageBox function) and to manually copy any files previously specified as 'Do not copy file'.

Post PM group creation
This code is executed after Program Manager group creation. This would normally be used if you wanted to create icons for which no files have been copied (eg NOTEPAD.EXE). In this instance it is likely that none of the project files would have 'Make Icon' selected.

Installation Completion
This code is executed right at the end of a Setup script after all copying and installation and after the 'Installation Successful' message box.
This section can be used to provide features such as starting NOTEPAD.EXE to view a README.TXT file after installation or even starting the application just installed.
No code is executed after this code in the script file.

Application de-installation
This code is only included in a setup script if the 'de-installation' option in the Project Attributes dialog is checked.
The de-installation code created by Setup Builder asks the user to confirm de-installation. It then deletes all files which were installed by the install procedure. The user-defined code for de-installation is then executed, followed by the automatic deletion of the application directory, .INI file and Program Manager group. A message then appears to advise the user that de-installation has been completed.

Under most situations, you should not require any user code to de-install an application, however if you create sub-directories of the application directory or any extra .INI files, then these must be deleted by manual code. Setup Builder will not remove directories other than the main application directory and even this won't be removed if it is not empty or there are further sub-directories. Setup Builder will ensure that files installed in sub-directories are deleted provided it

installed them in the first place. If not, you must delete them with manual script code.

:USER1-6
This code is placed towards the end of the setup script and may be branched to by any of the
other user code sections using a GOTO script statement, for example:

GOTO :USER1

These user code sections are ideal for placing script for different types of installations.
At the end of one of these user sections, you should place a GOTO :END statement, otherwise the
running of your script will pass into the next user code section - which you may actually want it to
do.

It is recommended that you place some text in all the user code sections, build your setup disk
suite and then view the .SCR script file to understand the positioning of the user code sections.

Select some text and press 'Cut' to cut the text to the Windows Clipboard.
Select some text and press 'Copy' to copy the text to the Windows Clipboard.
Press 'Paste' to paste text from the clipboard into the user code.

Press 'Ok' to save any changes, or 'Cancel' to exit without saving any changes to any of the user
code sections.

Disk Required

Setup Builder requires a new disk to be inserted in your disk drive. This may be because it requires the first disk of an installation suite or because it has copied as many files onto the current installation disk as possible and there is now no more space on it, so another diskette is required.

Insert the diskette with the number requested into the drive and press 'Ok' to continue building the installation disk suite or 'Cancel' to stop.

NOTES:
Upon copying all files onto your installation diskettes, Setup Builder may ask for the first diskette (disk #1) again. This is because the script file is always placed on the first diskette of an installation suite. Please also see Technical Notes **Setup Script Diskette Space** for more information.

As part of its disk 'tidy' procedure, Setup Builder deletes all files in the specified directory of new diskettes inserted during the project building procedure.

Setup Builder does not format diskettes for you so you should ensure that all installation suite diskettes are pre-formatted.

Setup Builder will **not** split large files across multiple diskettes. Therefore, if a file is larger than your largest diskette, Setup Builder may keep on asking for new disks and never place anything on them. This is a limitation of Setup Builder.

What are Optional Installations ?

When building an installation procedure, you often want to be able to give your customer the opportunity to select which components of your software are to be installed.

Setup Builder provides this ability via the **Optional Installations** feature.
With Setup Builder optional installations, you can create up to 15 optional installations with any name you wish to use. Typically, the name describes the files to be installed as it is used at install-time as a prompt option for the user. At install-time, a dialog appears which displays all of the optional installations. It does not appear if there is only one installation - it is assumed that all files will belong to this anyway as there is no point in giving the user the choice of one optional installation.

Setup Builder gives you the option to present the install-time dialog to the user as a dialog of checkboxes whereby the user can pick and choose the components they want, or a dialog of radio buttons whereby the user can only select one option.

Optional Installations are maintained by the Installations Dialog available from the Project menu.

Having created your optional installations, for each file in the Project Files Window, you can select which installations a file belongs to. You can have a file belonging to one or more or even all installations if you require.
To set which installations a file belongs to, double click on a file in the Project Files Window to obtain the File Attributes Window. Within this window, select the **Optional Installs** property and press the **...** button at the top of the window to obtain the File Installations Dialog. You can also double click on the property to obtain this dialog. Within the dialog, you can select which optional installations a file belongs to. To select multiple installations, hold down the Ctrl key while clicking with the mouse. Press **Ok** to save the selections.

What is Setup Builder ?

Setup Builder is a utility program for creating Windows Hosted setup procedures for installing software under Microsoft Windows 3.1, Windows 95 and Windows NT.

What are User Defined Objects ?

Setup Builder Version 5.00 provides a number of features which are completely defineable by the user. The features are:

- Backdrops
- User Defined Dialogs
- Cue Cards

All of these features have been implemented as 'User Defined Objects'. Each backdrop, dialog or cue card is considered as an object.
Each object has properties such as the object name, size and screen position and each object has member controls such as edit fields, text fields, bitmaps and buttons.

All of the User Defined Object facilities are available from the Object menu.
To create a new object, select the Object/New menu option. The New Object Dialog will appear which requests the properties for the new object such as its type, file name and size. You can get back to this dialog at any time by selecting the Object/Object Properties menu option so that you can change the object properties.
To maintain all of the controls on an object, select the Object/Object Controls menu option to obtain the Object Controls Dialog. This dialog enables you to add and maintain the controls which are appropriate for the type of object being maintained.
Control properties are maintained in the same manner as used in the File Attributes Window.
You can preview your object from the New Object, Modify Object and Object Controls dialogs at any time by pressing the **Preview** button.
Objects can be shared between setup projects such that you can build up a library of standard objects. You can import an object into the project using the Object/Import Object menu option. You cannot have two objects in a project with the same name - Setup Builder will prevent this occuring with an error message.

Backdrops
Backdrops usually appear as a blue backdrop display during an installation. Typically, they display information about the software being installed and often a bitmap such as a company logo.
Setup Builder allows you to create multiple backdrops in your project and select the for different screen resolutions. It also offers the ability to use one standard backdrop for all resolutions. You select which backdrops to use in the Backdrop Display Dialog.

Dialogs
Dialogs request information to be entered by the user. Typically, they consist of edit fields, text and buttons.

Cue Cards
These are also known as 'bill boards' and appear while files are being copied at install-time. Typically, they are used to display advertising material or information about your product while it is being installed with the idea of keeping the user's attention.
Cue Cards normally only contain text and graphics.

Bitmaps
Bitmap controls may be added to all object types. Before a bitmap can be selected for use into a control, it must be included in the project using the Project Bitmaps Dialog.
At project build-time, all bitmaps included in the project are copied to the first (and subsequent if required) diskette and they are always compressed to conserve space. At install-time, they are copied to the user's hard disk for the duration of the installation and deleted again afterwards.

See Also
Application Limitations

What are Optional Include Files ?

Setup Builder allows you to maintain one project file which can be used to create several different installation suites which may have different files within them, but several common files.
This is achieved by the **Optional Include Files** feature which enables you to select which files are to be placed on the install disk(s) at build time.

When setting <u>File Attributes</u>, change the **Optional File Inclusion** property to 'Yes' (the default is always 'No'). At build time you will then be given the option to select whether to include all the files marked as optional - this is done with the <u>Optional Installation Files Dialog</u>. If no files are marked as optional, the dialog does not appear.

The **Optional Include Files** feature means that you don't necessarily need to create a second project (with its associated maintenence issues) just to install slightly different files to another project.

Note that if optional files have their **Make Icon** property set, if the files are optionally left out, Setup Builder correctly removes the script code which creates the corresponding icons. Install-time disk space requirements are also correctly adjusted.

What's New ?

All new Setup Builder features are documented in the README.TXT file supplied with the Setup Builder installation.

Contacting the Author

Graham Plowman is the author of GPP Software products and he can be contacted by post at the following addresses:

PO Box 1124, Manly 2095, NSW, Australia

or

'Fiddlers Rest', Gaveston Hall Drive, Nuthurst, Horsham, West Sussex, RH13 6RG, England

All post sent to the British address will be forwarded to the Australian address, so please allow for a slight delay.

Graham can also be contacted on:

Compuserve ID: 100105,536
Internet: 100105.536@compuserve.com
Internet: gplowman@ozemail.com.au

Internet Home Page/Information/Download Service:
http://www.ozemail.com.au/~gplowman

Integrity Checking

This application has an automatic self-checking feature to advise of corruption to the executeable file.
The main purpose of this is to enable the program to identify whether it has been interfered with, either by a binary file editor or a computer virus.
Note that attempts to change icons and messages etc using products such as AppStudio or Resource Workshop will render this application unuseable since they will cause a failure of the check.
If the application executeable is corrupted in any way, the following message will appear:

Integrity Check Violation Error: Integrity has been violated!

If this message does appear, you should re-install this application after checking the reason for it occuring.

Note that all of the author's software is Virus Checked before it is distributed.

Other integrity-related messages are possible. If you need assistance, please <u>contact the author</u>.

PO Box 1124, Manly 2095, NSW, AUSTRALIA or
'Fiddlers Rest', Gaveston Hall Drive, Nuthurst, Horsham, West Sussex, RH13 6RG, ENGLAND

Price List for Software

```
Prod                                            CIS      CIS     CIS    CIS
No      Product                 Price   Price   Price    Shipping Total  SWREG
                                AU$     UK£     US$      US$      US$    ID
#00     HelpBuilder             $80.00  £40.00  $64.00   $10.00   $74.00 #4152
        HelpBuilder Upgrade     $40.00  £20.00  $32.00   $5.00    $37.00 #5193
        HelpBuilder User Manual $15.00  £7.00   $10.00   $8.00    $18.00 #5401

#01     Setup Builder           $120.00 £60.00  $96.00   $15.00   $111.00 #4153
        Setup Builder Upgrade   $60.00  £30.00  $48.00   $7.00    $55.00 #5194

#03     HelpBuilder SpellChecker $50.00 £25.00  $40.00   $6.00    $46.00 #10705

#04     SpellChecker (Developer) $80.00 £40.00  $64.00   $10.00   $74.00 #10521
        Distribution License (per copy) $30.00 £15.00 $24.00 $4.00 $28.00 #11064
        Unlimited Royalty Free
        Distibution License     $500.00 £250.00 $400.00  $60.00   $460.00

        Postage, packing & handling (orders received by post)
        Per disk (in Australia)         $5.00
        Per disk (outside Australia)    $10.00   £5.00
        Per manual                      $10.00   £5.00
        Any number of user license numbers $5.00 £2.50
```

When ordering, payment can be made by cheque, bankers draft, credit card or via the Compuserve SWREG facility.

Cheques
Please print off the appropriate order form 'Australian Order Form' or 'UK Order Form' and complete the details.
Cheques must be made out to GPP Software either in Australian Dollars (please post to Australian address) and drawn on an Australian bank or in UK Pounds Sterling (please post to UK address) made out to G.Plowman and drawn on a British bank.
Please do not send cheques drawn on your own local bank if you are not in Australia or England: we are charged currency conversions by our bank. To cover this cost we have to contact you for this extra charge which of course, delays the processing of your order.
European customers can use Eurocheques. Please make these out in UK Pounds Sterling and post them to our UK address. Eurocheques are not valid for non-European currencies.

Bankers Drafts
Please print off the appropriate order form 'Australian Order Form' or 'UK Order Form' and complete the details.
Bankers drafts must be made out to GPP Software either in Australian Dollars (please post to Australian address) and drawn on an Australian bank or in UK Pounds Sterling (please post to UK address) made out to G.Plowman and drawn on a British bank.

Credit Cards
Please print off the 'Australian Order Form' (ORDERAUS.TXT) and complete it including the credit card details.
The completed form must be sent to our Australian address. This can be done by post or by fax

(please contact us first for our fax number).
NOTE: Because our bank makes a 4% charge on credit card transactions, we have to also make a 4% charge on top of the total value of a credit card order.

Compuserve SWREG
SWREG provides an extremely convenient method of payment which means that you don't have to worry about currency conversions etc. SWREG will deduct a payment from your Compuserve account and credit it to ours. This is controlled by the Compuserve accounts system.
To use this method of payment, GO SWREG and select the 'Register Shareware' option. Complete the requested details.
This method of payment is the quickest method we can offer: it normally takes about 24 hours to register.

Postage Charges
These are applicable to all orders made by post or fax. The charge is applicable per disk ordered. We do not normally send out disks to customers who register 'online' as we send them the software and a license number via E-Mail, however, all orders by post will receive a diskette containing the latest version of the software.

Disks
Unless you request otherwise, 3 1/2 inch disks will be sent.
Disks are normally only sent to customers who order by post. Customers who register 'online' can download the latest version.

General Policy
If you register via Compuserve, you will receive an acknowledgement E-Mail advising of your license number and information on the latest versions and the support available.
All registrations receive unlimited time usage of the software, free technical support and notifications of upgrades.
Payment must be made in full before goods are delivered. Registrations cannot be made over the phone with payment to follow.

Upgrades
Upgrades are normally free of charge except where a major upgrade occurs in which case a nominal upgrade charge may apply (normally half the full price).
If you register by post and you have an E-Mail address, please advise us of that address to enable easier and quicker advice of upgrades and new versions.

Conditions
It is a condition of registration that you accept that you have fully tried and tested this software and found that it meets your requirements to your satisfaction. Please note that once a license number has been given, no refund will be possible.

Prices are correct as at 16/11/96. This price list is valid for a maximum of 3 months from that date. Please contact GPP Software for the current prices before you mail order your software. Compuserve SWREG prices are always up to date. All prices and specifications are subject to change without notice. E.& O.E.

Products Available from the Same Author

The following products are all available from the same author:

Help Builder

HelpBuilder is a Microsoft Windows 3.1 application for creating and maintaining Windows Help files and is designed for developers wishing to create Windows Help files to distribute with their products and for anyone wishing to create stand-alone Windows Help files.
This help file was created using Help Builder.
The application is not an 'add-on' to a commercially available word processor, rather, a Multiple Document Interface (MDI) application integrated with the Microsoft HC31.EXE help compiler (or equivalent compatible eg HCP.EXE)(HelpBuilder is configureable). Since it is likely that anyone using this product is likely to be an applications developer using Visual Basic, Microsoft C/C++ or Borland C++ they will already have HC31.EXE as it is supplied with these products.
HelpBuilder does not 'shell' out to DOS to compile files but provides a complete integrated Windows hosted compilation process.

Windows Help and Help Compiler Features Supported

- Topic pages
- Links and 'popup' windows to other topics, including optional 'picture hotspots', macros and 'windows'
- Browse sequence support
- Ability to place bitmaps in help files
- Fonts, colours, bold and italic text
- Header files
- Context identifiers
- Locking of and changing colour of top of Windows Help page
- Help file start up macros
- Topic start up macros
- Changing of 'locked' and client 'area' colours in Windows Help
- Compiler configuration - warning levels, CD ROM, compression etc
- Configuration of 'windows'
- Build tags
- Optional add-on spelling checker

Product Features

- Full keyboard support
- Projects to group individual topic files for a specific application
- Topics held in separate files in order that they may be shared between projects
- Toolbar selection of many facilities and status bars
- Error checking on project building
- Self-checking executeable - to detect corruption/tampering eg Virus
- Project Wizard to create a basic project
- Application Installation and De-installation procedure
- Printing of topics and projects
- Fully integrated and context sensitive Windows Help file including 'How to..' section
- Can create Help files for VB, C/C++, PowerBuilder etc - any language supporting Windows Help and passing of context numbers
- Ability to insert RTF commands
- Network locking on project and topic files
- Automatic creation of .HPJ and .RTF files

- Sample demonstration help file project
- Multi-tasking during compilation - no 'DOS' windows!
- Helpfiles 'testable' from within HelpBuilder
- Configurable compiler - via application .INI file
- Topic locator
- Importing of Norton Guides help projects
- In-built text macros

HelpBuilder SpellChecker

HelpBuilder SpellChecker is an addon Dynamic Link Library for HelpBuilder which provides spell checking of help topics. It requires HelpBuilder version 1.09.001 or greater.

HelpBuilder SpellChecker Features

- Small, self contained .DLL, requiring no third party products
- Supports dictionaries in different languages (ie different dictionary files)
- Words can be added to dictionary (extendable dictionary)
- Ignore word option during spell checking
- Ignore all occurences of a word option during spell checking
- Prompts with alternative spellings and suggestions
- Online help

Setup Builder

Setup Builder is a Windows 3.1 application for creating and maintaining Windows Hosted installation procedures and is aimed at developers who wish to provide that 'professional' touch to the installation of their software.
The installation procedure for this piece of software was created using Setup Builder.
The interpreter part of the product can also be used to run Windows Hosted DOS-like batch files using the script language supported.
Forget about distributing Microsoft's 0.5 Mb offering just to install a 100-200K executable! This product uses less than 300K of diskette space and is highly tailorable and doesn't require the user to get involved with C/C++, DLL's or Visual Basic in order to create install disks.
Setup Builder is not 'tied' into any one development tool: it can be used with any development tool you may choose.

Setup Builder Application Features

- Setup projects to which files can be added/removed
- Automatic creation of installation disks, requesting extra disks
as required
- Automatic splitting of large files across diskettes (including compressed files)
- Automatic creation of install script code
- Automatic creation of de-install script code
- Ability for user to place 'user-defined' code in install procedure if required
- Windows hosted with toolbar and status bar
- All files can have attributes set: target location, overwrite checks, in use checks, file compression, create program manager icons etc
- Ability to create program manager groups and icons
- Up to 1024 files in an installation (Professional Edition)

- Optional encription of install scripts (so that users can't change them)
- Context sensitive, comprehensive Windows Help file
- Network locking on project files
- Visual Basic .MAK project import facility
- 3rd party product import facility eg VB run-time, Access etc (Configureable)
- Ability to change text on all dialogs used in an installation (also allows different languages - defaults to English, French, German, Italian, Spanish or Dutch)
- Optional installation of components at install-time

Interpreter Supported Features (Interpreter is placed on install disks to 'run' install procedures)

- Script language (similar to DOS batch files) supporting if and goto statements, string and numeric variables
- Windows extensions - predefined dialogs (user definedable text), message boxes, creation of program manager icons/groups and ability to delete them
- User defined objects - Backdrops, dialogs and cue cards (bill boards)
- Comprehensive Windows Help file
- String and numeric manipulation (left, right, mid, +, -, *, /)
- Copying of files
- Optional progress gauge during copying
- Date and time functions
- DOS file management - create/delete/renaming of directories, disk space and file presence/overwrite/version checks
- Windows .INI file management
- File management - create/delete/renaming/copying/date & time setting, reading and writing of ASCII files
- Nested script files - call a script from another script
- External calls to run other Windows applications
- Shared file registry (stops un-installing of files shared by multiple applications)
- English, French, German, Italian, Spanish and Dutch language support on dialogs

SpellChecker

The Developer Edition of SpellChecker is a Windows 3.1 addon Dynamic Link Library which can be built into your applications to provide a spell checking facility which would otherwise normally require the use of DDE/OLE communications with products such as Microsoft Word. SpellChecker is an integrated .DLL, which is compact in size and is easy to build into your applications quickly. It is supplied complete with a help file which documents how to use and program it. SpellChecker may be built into any application constructed using a development tool which supports .DLL API calls such as C/C++, Visual Basic etc

SpellChecker Features

- Small, self contained .DLL, requiring no third party products
- Simple to build in to applications via .DLL API calls (documented).
- Supports dictionaries in different languages (ie different dictionary files)
- Words can be added to dictionary (extendable dictionary)
- Ignore word option during spell checking
- Ignore all occurences of a word option during spell checking
- Prompts with alternative spellings and suggestions
- API call to spell check a Windows Edit field
- API call to spell check a string buffer

- API call to spell check an ASCII file
- API call to spell check an individual word
- Online help - runtime (separate file)
- Online help - development/programming information (separate file)

Assistance/Further Information

If you require any assistance or further information on any of the above products, please contact the author at the address in the <u>Contacting the Author</u> section of this help file.

What is Shareware ?

Definition of 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. 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 or it doesn't do what you want it to do or you don't like it, then you don't pay for it.

Reasons for Registering

- All the 'unregistered copy' messages will disappear
- All the 'limiting' features of the Shareware version will be removed
- You will receive notifications when the next major versions become available
- You will receive free technical support
- You will automatically become a registered user of future versions with no more fees to pay (subject to major versions)

Grant

GPP Software grants you a non-exclusive license to use this software free of charge for a one month evaluation period after which you must register (pay) for a copy of this software if you wish to continue to use it. Upon registration you are granted a license to use this software for an unlimited time period.

You may not modify, translate, reverse engineer, decompile, disassemble or create derivative works based on this software except where this documentation specifically states otherwise.

Title

Title, ownership rights and intellectual property rights in and to this software shall remain in GPP Software. This software is protected by international copyright treaties. Your license does not give you any rights to title, ownership or copyright.

Disclaimer of Warranty

Users of this software must accept this disclaimer of warranty:
"This software is supplied 'as is' without warranty of any kind.   The author disclaims all warranties, expressed or implied, including, without limitation, the warranties of merchantability and of fitness for any purpose. The entire risk as to the quality and performance of this software is borne by you.
The author assumes no liability for damages, direct or consequential, which may result from the use of this software."

Limitation of Liability

UNDER NO CIRCUMSTANCES AND UNDER NO LEGAL THEORY, TORT, CONTRACT OR OTHERWISE, SHALL GPP SOFTWARE BE LIABLE TO YOU OR ANY OTHER PERSON FOR ANY INDIRECT, SPECIAL OR INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY CHARACTER INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF GOODWILL, WORK STOPPAGE, COMPUTER FAILURE OR MALFUNCTION, OR ANY AND ALL OTHER COMMERCIAL DAMAGES OR LOSSES. IN NO EVENT WILL GPP SOFTWARE BE LIABLE FOR DAMAGES IN EXCESS OF THE LIST PRICE OF A LICENSE FOR THIS SOFTWARE, EVEN IF GPP SOFTWARE SHALL HAVE BEEN INFORMED OF THE POSSIBILITY OF SUCH DAMAGES, OR FOR ANY CLAIM BY ANY OTHER PARTY.

Termination

Your license will terminate automatically if you fail to comply with the limitations described. On termination you must destroy all copies of the software.

Miscellaneous

This software is a "shareware program" and is provided at no charge to the user for evaluation. Feel free to share it with your friends, but please do not give it away altered or as part of another system.   The idea of "user-supported" software is to provide personal computer users with quality software without high prices, and yet to provide incentive for programmers to continue to develop new products.   If you find this program useful and you wish to continue to use it after a reasonable trial period, you must make a registration payment. The registration fee will license one copy for use on any one computer at any one time.   You must treat this software just like a book.   An example is that this software may be used by any number of people and may be freely moved from one computer location to another, so long as there is no possibility of it being used at one location while it's being used at another.
Just as a book cannot be read by two different people at the same time.

Commercial Users should register as soon as possible and not let the one month expiry feature stop the program and cause problems with meeting project deadlines: 'urgent' registrations are still subject to the speed of E-Mail systems and the postal system - please don't leave registration to the last minute. You are not allowed to distribute the output of the Shareware Edition of this software for financial gain. Various limiting features have been built into the Shareware Edition. Site-License arrangements can be made by contacting the author.
Anyone distributing this software for any kind of remuneration must first contact the author. In this way the distributor can be kept up-to-date with the latest version of this software.

You are encouraged to pass a copy of this software to your friends for evaluation.   Please encourage them to register their copy if they find that they can use it.   All registered users will receive a copy of the latest version of this program, a license number, free technical support and notifications of upgrades.

It is a condition of registration that you accept that you have fully tried and tested this software and found that it meets your requirements to your satisfaction. Please note that once a license

number has been given, no refund will be possible.

GPP Software

Further Technical Information

Further technical information about GPP Software products may be obtained from the GPP Software Technical Support help file which may be found in the Compuserve WinShare forum. Simply GO WINSHARE and select 'Library Search'. Enter 100105,536 as the 'Contributor' and then press 'Search'. Click on the required item(s) and press 'Retrieve' to download them.

The GPP Software Technical Support help file is a file containing technical information (Technical Reports - TR's) about GPP Software products which is not supplied in the help files supplied with the products.
It supplies information on technical issues and how to resolve them as well as reported faults, their current state of resolution and work-arounds etc.
There is also technical information about how various parts of the applications work.
This file will be continually updated as the products are developed and as information becomes available.
Please note that as faults are identified and fixed, the fixes will automatically appear in future versions of the software, normally the next release.