# Contents



Project Manager 3 for VB3 Win - Helpfile generated by VB HelpWriter.

# Installation

Project Manager 3 for VB 3 Win is written using VB 3.0 Pro Edition. It uses an Access database to store information on your projects. If you do not have Access 1.x or 2.0 (with the Compatability Layer) on your system, you must have VB installed with the Data Access option to ensure the Access Jet database engine files are in your Windows\System directory.

The database file **"VBPROJ2.MDB"** was created on a secure Access 2.0 system and requires its own "**SYSTEM.MDA** " file, which is supplied. If you are using the Access 1.1 version, the database file is **"VBPROJ1.MDB"** , and it requires its own **"SYSTEM.MDA"** file, also supplied. You will not be able to access the database without this file. To avoid conflicts with any other secure MDB's on your system, put all supplied files in the same directory on your hard drive. The name of the directory does not matter, but do not put any files other than the ones supplied for this program in it.

The program was written using a secure database as the structure of the database is critical to the operation of the program. If any changes are made to the structure, the program will fail. If you wish to create your own reports through Crystal Reports or another report writer, you may access the data in the tables by logging on with the UserName "guest" and no password. **Do not** attempt to make any changes to the database.

Install Project Manager as follows:

1./ Make a directory on your hard drive to store Project Manager in.

2./ Place the Project Manager executable ("**PMGR32.EXE**" for Access2/Compatability layer users, or "**PMGR31.EXE**" for the Access1.1 version), this helpfile, the file "**SYSTEM.MDA**" and the database file "**VBPROJ2.MDB**" or **"VBPROJ1.MDB"** in the directory you created for it.

**VBX's**

_____All VBX's the program was compiled with are supplied. Due to problems with incompatible VBX's with the same name, you may experience failure of the program if you attempt to run it with the VBX's you have on your system, unless they are identical to the supplied VBX's.

To avoid overwriting your own system's VBX's, you should install the VBX's, and their .LIC files, in the directory holding the program executable, not the Windows\System directory. You may try running the program with your own VBX's, and if successful you will not need the supplied VBX's.

VBX's known to cause problems if not the exact ones supplied are GRID.VBX, VSVIEW.VBX, VSVBX.VBX. It is possible that the other VBX's will cause program failure as well if they are not identical.

There is no install program supplied for the above reason. We have had too many occurrences of Visual Basic applications' install programs overwriting our system VBX's with VBX's of the same name but different date/size/version . Overwriting your system's VBX's with these the supplied VBX's could cause you serious problems with your distributed programs at some future point. While we are aware this uses disk space, this is the safest route.

**DISKINF.DLL**

The file DISKINF.DLL (all 1792 bytes of it) can be placed in either the Windows\System directory or the directory holding the project files. This .dll returns the size of a disk and the free space on a disk.

You may find DISKINF.DLL useful in your projects and you are free to do with it whatever you

wish. The syntax to call the two functions in it is as follows:

```
Declare Function GetDiskSize& Lib "Diskinf.dll" (ByVal Drive%)
Declare Function GetDiskFree& Lib "Diskinf.dll" (ByVal Drive%)
```

Drive 0 is the current drive, Drive 1 and 2 are floppies(even if the system has only 1 floppy), and 3 is the first hard disk. This .dll is handy when used in conjunction with the Windows API function GetDriveType% Lib "Kernel" (ByVal nDrive%) to determine the number, type, size and free space of the drives available on a system.

Be aware that if you call these functions on a CD-ROM drive with an audio CD loaded your system will lock up. We know of no way to detect this condition with API calls.

## Important Notes:

Ensure that your VB.INI file has a section as follows in it:

```
[Options]
SystemDB = (a path to your normal system.mda file would be here)
```

If you use Project Manager to load and run a project which uses an Access database and you do not have this line in your VB.INI file, VB will find the system.mda file which Project Manager uses. Because it is a secure database, an error message indicating an invalid Password or UserName will be generated and you will not be able to access your project's database. Ensuring this section is in VB.INI will point VB towards the proper system.mda file for it. This error will not occur if you run a project without using Project Manager to load it.

**Make sure all your project files have been saved as text and that you have current backups of your projects before running Project Manager.**

This is absolutely critical, failure to save your files as text may result in them being truncated when Project Manager tries to read them as text files.

# Project Manager Overview

Project Manager for VB 3.0 Windows is a multi purpose tool to aid in managing VB projects.

Before using Project Manager utilities on a project, make sure that you have a current backup of the project. Read and follow the instructions below.

To operate correctly, the program requires that you do the following three things:

**1./** Save your projects as Text, and let VB's editor format and write your files. This is done by selecting the Options|Environment menu items, and setting the default save to "Text". You should not use another editor to edit the files. The files must be formatted by Visual Basic's editor. Tabs, and other formatting characters left by other editors may not be stripped by certain VB functions, and will cause problems reading the files.

If you use another editor, then prior to adding any project to Project Manager you should load it into VB, run it under VB, then save the project before exiting. This should help ensure, but does not guarantee correct formatting of the files. Check your results after running the program on your project to ensure it can correctly read your files if they are written with editors other than VB's.

**2./** To allow the program to properly locate variables, you must explicity declare them in your projects. It is considered good practice to put Option Explicit in all your modules. This eliminates problems due to misspelling, etc., in code. You can ensure this is added to all your modues by setting it as a default in the Options|Environment menu item of VB.

**3./** Your forms must be saved with the default .FRM extension, and modules with the default .BAS extension. All project files (except the .VBX's your project uses) must be stored in the same directory as the .mak file. This is necessary to enable to program to operate faster, and without needless bloated code to search drives for various files.

Project Manager will refuse to load projects with .BAS or .FRM files stored in directories other than the one containing the .MAK file. However, If your projects do not have the default .FRM or .BAS extensions, or contain files stored in other directories, use "File|Consolidate Project Files". This function will attempt to copy the files to the directory holding the project's .MAK file, and will give you the opportunity to rename the extension with the correct one. See the topic File Menu for more information.

**Project Manager Database**

Project Manager 3 is available with either an Access 2 .MDB, or an Access 1.1 .MDB. If you already have **both** Access 2 and the Compatability Layer installed, the Access 2 program operates faster because of the Access 2 Jet database engines' advanced features. Both databases contain the same information and structure, will return the same information to the user, and offer the same utilities for working on projects.

Both databases were created on a secure Access system, and thus need their own System.MDA file which is supplied. You can open the database by logging on as "Guest" with no password. **Do not make any changes to the structure of the database**, as that will cause the program to fail or store erroneous information on your projects.

If you wish to run reports of your own design on the data in the database, use "guest" with no password for data access to the database. Do not change data in the database as that will cause program errors the next time you attempt to run the program. Fields which can safely be edited can be

edited using the reports form. See the Report Menu topic for further details.

**Project Manager Operation**

The first time you load Project Manager, you should set your program options. Select "Utilites| Set Options". Fill in the path to VB, the path to your default text editor for viewing files, and check whether you wish to use version and task descriptions.

Project Manager is an MDI application, and it allows multiple projects to be tracked, viewed and manipulated at the same time. VB only allows one instance to be running at a time, but you can still work on multiple projects concurrently in Project Manager.

Instead of loading VB, then loading a project, use Project Manager to do this. You can load Project Manager, then find your project in the program, then have it load VB with the project loaded and record the time.

Alternatively, you may pass a command string to the program when you set up the properties for its icon in Program Manager. The command string simply needs to be the fully qualifed path to a project .MAK file. Then, when Project Manager loads, it will load VB with that project loaded. If you use some other shell, you can put its command line as a parameter to Project Manager and Project Manager will start it. Note that you can only pass it one parameter, either a .MAK file or an .EXE file, and both must be fully qualified paths .

Project Manager is distributed as a fully functional program, however, unregistered versions will only allow you to load two projects into the database. In addition, you will see a register form when you load the program and will have to clear it. Read the "Register" topic for information on registering the program to eliminate this form and fully enabling the program.

When first loaded, Project Manager will load the MDI parent form, then a Project Window form which will be blank as you will have not added any projects to the database yet. To add a project, select "**File|Add Project**" or use "**Ctrl + A**" as a shortcut. Right clicking on a blank area of this form will pop up the "**File**" Menu.

When adding new projects, you will get a Common Dialog which will allow you to select the .MAK file of the project you wish to add. After confirming you wish to add this project, the program will read the project files and add the various pieces of information to the database.

You can either read only the .MAK file, which is very fast and stores information only on the project files, or fully read all the files, which parses each file and will take more time, depending on the size of the project you are adding. The information on the contents of the project, such as variables, controls, etc, is stored in the database after a full read. Normally, a full read is used only if you wish to run reports on it, or use some of the utilites which require a full update. Keeping only the .MAK file updated will minimize the size of the database.

You may select projects in the database by using the combo box on the command ribbon or the data control on the project window. On this form you may enter a title for your project, and a backup path in the text box provided. The backup path is used to backup all project files to your specified path in one action.

The various menu items are described under their appropriate forms' help file topics. Some frequently used menu items are operated via the buttons on the command ribbon. See the help topic "Command Ribbon" for information on the buttons and the combo box on the ribbon.

When selected, various reports, which are loaded by selecting the appropriate one under the **"Reports"** menu, will display information parsed from the program. There is virtually no limit (except

available memory) to the number of MDI child windows displaying projects and information about projects that can be open at a time.

Project Manager will write error handler code for your projects. The program will write either a simple message box error trapper, or it will write a more sophisticated Error Logging handler, which will write a log file of the errors that occur when your project is running. You can select which style of error handler your prefer by selecting the "**Utilities|Set Options**" menu items on the main form menu bar. See the Help topic "Error Handler Information" for more on this topic.

Project Manager will load VB with your project loaded, and record the time which you start work on it. It will also unload VB and record the time you stop working on your project, and keep a running total of time spent working on the project. This is accomplished by the "**Times**" menu items on the main form menu. Left clicking on a blank area of the Project Window form will bring up the "**Times**" menu as well.

The "**Utilities**" menu contains three menu items to locate dead code. These menu items will search your project for unused declared variables, unused constants, and unused procedures. The results of these searches will be displayed on a form which will allow you to check for them in your code using VB's search functions. In addition, enabling code move between projects or in a project between files is accomplished in this menu area. Recording faults reported on your projects is done under this menu item as well. See the Help topic on "Code Move", "Fault Recording" or "Unused Code" for more information on these items.

The **"Versions"** menu, as well as three buttons on the command ribbon will enable versions, and allow you handle versions. You can create a new version of a project, delete one, and add comments to the database about a version. See the Help topic on "Versions" for more information on this feature.

For more information on each form and menu item, view the help topic for these items.

# Project Manager MDI Parent Form

**---Menus---**

**File**

**New Project Window**

Selecting New Project Window will open a new Project Window MDI Child form which will display the first record in your Project database. If there are no records, the Project Window form will be blank.

**Close Project Window**

Closes the active Project Window MDI Child form.

**Exit**

Exits the program

**Database**

If you get an error message while attempting to compact, backup,or repair, exit Project Manager and re-load it, then try again.

**Compact Database**

Compacts the "**VBPROJx.MDB** " database. This makes the database take up the minimum amount of disk space and speeds access to project information. It should be done after any significant changes are made to the projects in the database and they have been updated. It's a good idea to select Backup Database before you Compact the database. While I have never experienced an error in a compact operation, anything is possible and you should not risk your data.

**Backup Database**

Will load an Input box to accept a path to backup the "**VBPROJx.MDB**" file to and copy it to that path. Ensure that the path you give it is valid and correctly formatted.

**Repair Database**

If your database is corrupt, or the program cannot read or run reports on the database, use this menu option to repair it. Corruption of the database can occur in many ways, particularly from exiting the program or Windows improperly. However, the Jet database engine is normally able to repair the database to a usable state.

**Help**

**Show Help**

Loads this help File.

**Search**

Loads the standard Windows Search Help dialog.

**Contents**

Loads the help file Contents screen.

**Help on Help**

Loads the standard Windows Help on Help helpfile.

**Register**

Loads the included registration form. Fill this in, print it and send it in to register your program

and fully enable it.  See the helpfile topic "Register"  for more information.

# Command Ribbon

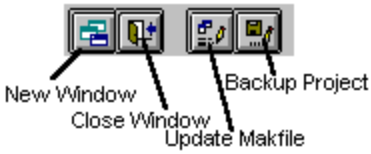
The command ribbon buttons duplicate the functionality of some commonly used menu items. The combo box on the command ribbon can be used to quickly locate a project by dropping it down, then selecting the desired project. On selecting it, the active form in the MDI Window will go to that project. If no Project form is loaded, one will load and go to the selected project.

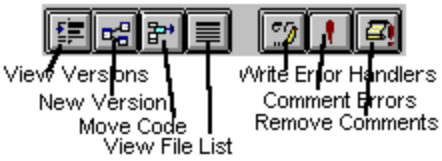
To view the function of any button, click and hold the left mouse button, then read the status bar. To release without executing the action, drag the mouse off the button without releasing the mouse button. Clicking the right mouse button will accomplish the same action without activating the button.

The list of button functions is as follows:

| Button # | Function |
|---|---|
| 1 | Open New Project Window |
| 2 | Close Project Window |
| 3 | Update .MAK File |
| 4 | Backup Project |
| 5 | View Versions |
| 6 | Create New Versions |
| 7 | Enable Code Move |
| 8 | File List View |
| 9 | Write Error Handlers |
| 10 | Comment Out Error Handlers |
| 11 | Uncomment Error Handlers |
| 12 | Show Help |

The buttons appear as below:



# Project Manager Project Window

The Project Manager Project Window displays information on each project added to your database. It displays information on the project files, backup path, etc. The menus for this form allow you to access the features of the program. Multiple project windows can be open at any time, each one can display a separate project.

**File List Box**

The file list box operates in three different modes:

1./File View mode: Displays the files the project references, sorted alphabetically and by file extension. When shown as a standard list of files, you may double click a file to load it into a text editor. You set the path to the text editor you wish to use by using the "**Utilities|Set Options"** Menu items, then clicking the "**Text Editor**" file tab.

2./Version Mode: Displays the versions of a project created by the program. See the "Versions" topic for more information on creating and using Project Versions.

3./Code Move Mode: Displays a list of the files shown as a graphical tree. Clicking a file will cause it to open and display the procedures contained in the file. See the "Code Move" topic for more information.

**Version**

_____This panel displays the version recorded as being currently in use for this project. If blank, the original directory version is current. It is not an editable field, the program will set and update it when you are working with versions.

**Code Lines**

_____Displays the number of valid lines (excludes comments and spaces) of code in your project. This is calculated when the project is read to extract information for the database. When you create a new version or change the current one, this field will be blanked as the information in the database may no longer be current.

**Last Updated**

_____

_____Displays information on when the project was last updated. Not editable, set and updated by the program. The date and time are displayed as formatted in your Windows short time setting.

**Full Update**

If checked, this box indicates that there is a full update currently stored in the database for the current project. This is not an editable field, it will be set and updated by the program. Certain program functions cannot run without a full update being done on the project immediately prior to using the functions on your project. These functions will prompt you to run a full update if necessary.

**Note:**

The functionality of the **"File"** and **"Times"** menus are duplicated by clicking on a blank part of the Project Window form with the mouse left and right buttons respectively. The mouse buttons

will pop up these two menus, which saves time moving to the normal menu bar area.

**---Menus---**

**File**

The items enable project handling, and project window handling, as well as making a clean exe file for a project. See the "File Menu" Help topic for information on this Menu.

**Reports**

All items will load a report containing information on what is in your project. Multiple instances of this form can be opened for the currently active project. You may print the contents by clicking the print button on the report window when the report is done loading. It will print to your default Windows printer. See the "Reports Menu" Help topic for information on this Menu. See the "Options" topic for information on setting report fonts and headings.

**Times**

Menu items under Times Load, Unload projects, record start, stop times, allowing changing task descriptions, and load the times report. See "Times" for further information.

**Utilities**

**Set Options**

Loads the Options form. See "Options" for help on functions that form provides.

**-**

**Write Error Handlers**

See the Help Topic "Error Handler Information" for more detailed information on this subject.

**-**

**'Check Unused' Menu Items**

These items check for unused code in your projects. See the help topic "Unused Code" for information on these features.

**Code Move**

Code move is enabled or disabled by the submenu items for this menu item. See the topic "Code Move" for more information on this subject.

**Window**

Standard window arrangements commands

**Cascade**
**Tile Horizontal**
**Tile Vertical**
**Arrange Icons**

**Help**

The help menu items are as described in the help topic.

To do maintenance on your database file, such as backing it up or compacting, close all open windows in the application and use the "Database" menu items on the main MDI Parent form.

# Options

Selecting "Utilities|Set Options" will display the options form, a tabbed dialog box which will give you access to the following code. This form is set system modal and must be closed with the OK or Cancel buttons on the form.

### Error Handlers

Allows the user to set the type of error handling code for the program to write, either Error Logging or Message box. The default is for Error Logging. When changed, the program will write the type of error handler desired to the "**PROJMGR.INI "** file in your Windows directory, and use that information for all error handler code until you change it again.

This area also allows you to select whether you wish to confirm each error code write as it occurs. Checking the box will enable this option.

See "Error Handler Information" for further details.

### Descriptions

Two check boxes will appear on selecting this item.

**Task descriptions** enable you to add comments when starting work on a project, to record what function is being done in that time period.

**Version descriptions** enable you to add comments to versions to track what changes are made in code in various projects.

### Path to VB

_____Set the fully qualifed path to VB.EXE in the text box to enable the program to load VB and make a clean .EXE file.

### Text Editor

_____Set the path to the default text editor you wish to use to load files into when they are double clicked in the main form list box.

### Path to PKZIP.EXE

Set the full path to PKZIP.EXE in this text box. This is used to run PKZIP.EXE if you wish to ZIP all the files in your project directory. Please set the full path including the executable name eg. "C:\ PKZIP\PKZIP.EXE".

### Quick Format Diskettes

This option tells PKZIP.EXE whether to quick format your backup diskettes or not. Please read the helpfile section "Zip Files in Project Directory" under the "File" topic for more information.

### Printing

These options allow you to choose the font and font size for the headings and body of your

reports. In addition, if you wish a custom heading on your reports, you can enter the desired text in the text box on the options form.

**OK Button**

Click the OK Button to save your changes and exit the Options form.

**Cancel Button**

Click the Cancel Button to discard your changes and exit the Options form. You will be prompted to confirm discarding your changes.

# File Menu

**New Project Window**
Loads a new Project Window.

**Close Project Window**
Closes the currently active Project Window. This does not exit the program, and if you have multiple Project Windows open, it only closes the active one.

**Add Project**

Opens a Windows Common Dialog box which allows you to select a project .MAK file to add to your project database. The function will ask you to confirm your selection, and offer you a choice of adding it in two ways.

If you answer 'Yes' to fully add the project, the program will parse the project files, and add the information from the project files to your database. While this code will not lock your system up, it is suggested you do not switch to other tasks while this function is running. The length of time it takes depends on the size of your project, what it contains, the speed of your system, etc. A DX2-66 clone with 8 MB of ram reads the files for this program in about 18 seconds, while a 386-40 clone with 8MB of ram takes about 49 seconds to do the same. A meter bar will keep you informed of the progress of the function.

If you answer 'No' to the dialog, it will read the .MAK file only and add the files contained in the project to the database, without reading them for contents. This provides a quick way to get a project into Project Manager if you do not want to immediately run reports or use some utilities on it. Using this function, and the associated Update.MAK File Only function will keep your database at the smallest possible size, as these functions do not store the contents of the project (other than files, times, fault records, and version comments) in the database.

**Update Project**

This function will ask confirmation that you wish to Update this project, and after confirmation, it will read your project to check for changes and add them to the database. The same cautions as above apply for this function. This function may take some time to run on large projects. Normally, you would only do this immediately before you use utilities on the project which require a full update, or wish to view/print contents of the project. It is important to run the Full Update only immediately before using a utility, and make no changes to the project until after you are done using the utility. If your utility changes the project files, such as moving code, then you should run another full update before using other utilities. The Error Handler functions, and Version functions do not require full updates, just updates of the .MAK file.

**Update .MAK File Only**

This function will run an update on the .mak file only, reading it for added or deleted files. This function deletes all information on items in the project from the database, and when run will reset the code lines value and last updated check boxes to zero values. This function is used to keep the database size down for quicker backups, and to quickly update changes in the .MAK file for project file backup purposes. You should update your project whenever you add or remove files from it.
Most project functions, such as code searches, moving code, would only be performed occasionally, so there is no need to keep data which is likely outdated in the database, or spend the time fully updating the project daily. This provides a quick way to update your project files when you are not going to run reports or use some utilites on the project. Functions which require a full update will prompt

for it if one is not currently stored.

### Delete Project

Removes a project and all its information, including start and stop times from the database. It will ask for confirmation before running, but after confirmation, the project is completely removed. The only way to recover this information is to add the project to the database again. The times, however, are permanently removed, so use this selection with caution. It does not, however, delete the actual project files or any versions of the project which you created from your hard disk.

Deletes of some parts of project contents (not times) occur automatically when certain functions run. Deletes with the Access 1.1 database take a bit longer than the Access 2 database.

### Backup Project

Backs up your project files to a path which you enter in the appropriate box on the main Project Window form. Type in a path with drive and directory to back up to. It is suggested that you backup to a directory even if you are using one floppy to hold each project separately. There are limits to the number of files which can be in the root directory of a drive, low density floppies can hold 112 entires, high density floppies 224, and normally other drives can have 512 entries. If there are more files than that, you will get error "Too Many Files". If the directory does not exist, the program will offer to create it for you if it is only 1 level below a directory that already exists, and will create it on subsequent disks if more than 1 is required to hold your files. If it is more than 1 level the program cannot create it and an error will result. Ensure you use valid DOS path characters.

This function will ask you if you wish to back up all the files in the project directory, or only the files listed in the .MAK file. If you keep all your project files, including help files, or database files, in one directory, this will copy them all. The program will back up all files in this directory if you click "Yes", or just the files in the .MAK file if you click "No." This is helpful when you wish to backup help files and all project files, or saves time when only the project files have changed. It does not copy VBX's.

The program will delete all files that are in the backup directory prior to backing up the current files. It does this to ensure the maximum number of files can be placed on each diskette. If you have a large number of files in the backup directory, this process can take some time so it is best to rotate your backups among several diskettes and quick format your floppies prior to backing up.

Project Manager will check the size of your project files and if one is too big for the target disk, will advise you that it cannot be copied, but will copy the remaining files. It will check the size of the files against the space left on the target disk and pack on as many files as it can fit. When full, it will prompt you to insert another diskette.

Note that if you have a version loaded, other than the project root one, it will backup the files in the current version directory, not the original project directory. This is true whether you select Backup all files in directory or Backup Project files only. When working with versions, remember the program is always working on the files in the current version directory.

If you wish to backup another version, simply enable Versions (see "Versions" for more info), then change to the version you wish to backup, then run Backup.

If you are backing up to floppies, it is suggested you alternate your backups among several sets of floppies. If you choose to backup only project files, and have previously backed up all files in the directory to the diskette you are using, you should use a different diskette as the program will delete all files in the backup path before copying the new files. This means that some files which were previously copied from the project directory may not be included in the current backup, and these files will be deleted

without being replaced.

**Zip Files in Project Directory**

Set the full path to PKZIP.EXE (ie: C:\PKZIP\PKZIP.EXE) in the PKZIP Path text box of the "Paths" tab in the Options form. Also set whether you want PKZIP to quick format your backup diskettes or not. This function is currently set up to pass a command string suitable for PKZIP version 2.04G, which seems to be the currently used version. Compatability with other versions cannot be guaranteed.

When you select this menu item, it will call PKZIP and zip up all the files in the project current. Remember it works on the current version directory. This function does a quick format on diskettes if you set that option, to get the maximum free space on the diskette, and uses the "span multiple disks" option if the ZIP file is larger than your target diskette. It gives the .ZIP file the same default name as the filename, less extension, of your project .MAK file with the new extension ".ZIP".

The function attempts to determine if you are backing up to a removable drive or not, and whether you have asked to format the drive. If it detects a drive other than a removable drive, it will not use the span multiple disks option, and will not format. Due to the difficulty in detecting drive types, if you are going to use the quick format option, ensure you check your backup path is to a floppy or other removable media that can be safely quick formatted.

**Make .EXE File**

This function will prompt for confirmation, and upon receiving it, will close VB if open, then reload the program with the project shown in the currently active project window loaded. It will then minimize VB and have it create an .EXE file with no extraneous code embedded in the .EXE. It will unload VB when done.

**Consolidate Files**

If you have a project with invalid file extensions (eg. files with extensions other than .FRM or .BAS) or files which are stored in directories other than the one holding the project .MAK file, the program will reject it. Run "File|Consolidate Files" to correct this. This function will read the .MAK file and copy the project files to the directory holding the .MAK file. It will prompt you for the correct extension if a file has a filename other than .FRM or .BAS. Note this code does not affect VBX's, which the program recognizes when stored elsewhere. The program will re-write the .MAK file as it goes to point to the correct new location of the project files. When finished, you can run Add Project to add the project to the database. Because of the myriad combinations of file locations, this function may not always be successful, though it has worked with numerous drive configurations and across a Windows For Workgroups network.

**Printer Select**

This function will display the standard Windows dialog box to select the default Windows printer. Use this function to set or change the default Windows printer prior to printing reports.

**Exit**

Closes all windows and exits the program entirely.

If you have used Project Manager to start a project and record the start time, then unloaded VB and Project Manager manually, there will be an incomplete time record in the database. The next time you load that project, it will advise you there is an incomplete record in the database for that project. You may not want the program running at certain times, but may wish to continue working on your project, this gives you the flexibility to do so if necessary.

If you manage to leave an open record in the database, you can edit it using by selecting "Times| View Times" and clicking on the line you wish to edit. The start time, stop time and description fields are editable in this report. Your changes will be written to the database as you make them, and the total time values will reflect the changes the next time you run the report.

# Report Menu

Items selected under the reports menu will load a form which features a grid displaying the information requested on the project.

Reports for Times, Faults, Version Comments are editable. Clicking on the line in the grid you wish to edit will cause text boxes to appear to enable editing.

When the reports form is loaded and is the active form, the command buttons and the combo box in the command ribbon of the parent window are inactive. You can reactivate them by clicking on a project window form, or closing the active reports form.

**Buttons**

**Close**

Unloads the current report.

**Print**

Prints the current report to the Windows default printer.

You can change the default printer by selecting "File|Printer Select". To add a customized heading to your reports, as well as set the head and body fonts and font sizes on the Options form, select "Utilites|Set Options", and click the "Printing" tab on the Options form. A limited range of font sizes is provided as selecting fonts which are too big will caused problems printing the reports. There is no preview before printing.

**Constants**

Loads the information form which contains information on your project constants. It will tell you the file the constant is in, the name of the constant, the scope, value, and procedure (if any) it is declared in.

**Controls**

Loads the information form which contains information on controls in your projects. You will be advised what file the control is in, the name of the control, and the type of control it is.

**Declarations**

Loads the information form which contains information on external functions declared in your project. You will be advised the name of the Function and the origin of it, and the file it is declared in.

**Menus**

Loads the information form with information on menu items contained in the project. It will tell you the file the menu is contained in, the menu name and caption, submenu name and caption, and the shortcut key for that menu item.

**Procedures**

Loads the information form displaying Procedures in your project. It will advise you the file name the procedure is in, the Procedure name, the arguments the procedure takes, its return value, whether it is a Sub or a Function, and if it is a Private or Static procedure.

**Variables**

Loads the information form with Variables declared in your project. It will advise you the name of the variable, the procedure it is declared in, the file name it is declared in, the scope and the data type of the variable, and whether it is a static variable or not.

If you are using DefType statements, and do not explicitly state the data type of a variable, the program will record the data type as "not declared".

**Version Comments Report**

Loads a report with the comments you have entered for Versions for the current project. The description field is editable in this report.

**Faults Report**

Loads a report with a summary of the faults report for the current project. The two fields for recording the description of the fault itself and the method corrected are editable in this report, to give you a editor/browser form.

**Projects Report**

Loads a report with a summary of all projects currently in the database.

**Times Report**

The times report is accessed from the Times menu, or by left clicking on the project window form, which will pop up the times menu. In this report, the start time, stop time, and task description fields are editable. If changes are made to the times, they will not be reflected in the totals until the next time you close the report as the totals are calculated and updated whenever the report is run.

**Close All Reports**

This function will close all open report forms.

# Versions

**Versions Overview**

_____Creating a new version of the current project is done using the Versions menu items, or the first group of command buttons to the immediate right of the combo box. During the process, you will be asked to confirm the procedure on several occasions. Selecting "No" or "Cancel" in any dialog will terminate the process without making a new version. The rationale behind creating versions in the program is as follows:

A project directory must contain all the project files for Project Manager to work. Read "Project Manager Overview" and search Help for "Consolidate Project Files" for more details on this requirement. When you wish to create a new version (for example, when you are going to make some major change in the project and may wish to revert to an earlier version if it doesn't work, or add error handling code before compiling), you select "Create New Version From Current", or the second command button to the right of the Combo Box.

You will be asked to confirm that you wish to create a new version from the current one. If you have not created a version before, the directory which holds the project will be the current one. The panel under the heading "Version" on the project window form tells you what version is current. If blank, the current version is the original version. When you confirm the action, you will be prompted with a form to enter a version name.

The Version name you supply will be the name for a subdirectory to be created under the project directory. Thus, the name has to be a valid DOS directory name, 8 letters or less and only containing valid DOS characters. It is suggested you use only letters and/or numbers for version names, unless you are sure the characters you intend to use are valid. Using invalid characters will prevent the program, DOS and Windows from being able to find the directory. For example, DOS will accept a space in a directory name, but will not be able to find that directory, remove it, or do anything with it except delete it using the deltree command.

Click the cancel button on the dialog will cancel the versions process.

The program will now ask you to confirm reading the .MAK file. It does this to ensure that the files shown as being in that version in the database are accurate and up to date. If you chose no at this stage, the process terminates. Chose 'Yes' to allow the program to continue. Updating the .MAK file also removes the information on the project from the database, as it is likely no longer accurate.

You will now be asked to confirm one of two options in creating a new version. If you keep all your support files, such as databases, helpfiles, etc., with your project in its directory, you can optionally create a new version and have all the files in the project directory copied to the new version directory. Selecting 'Yes' from this dialog will copy all the files in the project directory to the new version directory. Selecting 'No' will copy only the project files that are contained in the .mak file to the new version directory. In this way, you can create entirely new versions, including new database versions, helpfiles, etc., without altering your previous ones at all.

Having supplied a valid version name, the program creates the new subdirectory, then copies all the project files from the current version to the newly created directory. As it does so, it adds a commented out text stamp to the beginning of each project file which says 'Project Manager Version' & the version name you supplied. It then will copy the .MAK file to that directory, and add the version stamp to the end of the .MAK file.

If you have enabled "Use Version Descriptions" in the Options form, you will be prompted to enter comments on this version of the program. You may enter any comments you wish to enable you to

create an audit trail on the project. When you view a report on version comments, the comments will be dated and related to the various versions. You can add comments to a version at any time by selecting "Add Version Comments" from the Versions menu.

The program will then ask if you wish to make the version you just created the current one, and if you agree, will now show that this version is current in the Version panel on the project window form. If not, the new directory and new version files are created, but the former version is still current. The Version text box always holds the current version, if blank the original project version is current.

The program will now ask if you wish to load this version. If you select yes, it will load VB with the newly created version loaded. When you have created multiple versions of a program, Project Manager will always load into VB the files in the subdirectory which are shown as being the Current Version.

You now have an exact copy of your project, and can try your new changes. You can create a new version from the original version at any time, or from any other version of that project that you have already created. The program always version stamps the newly created files, so if you create a version from a previously created version, you will have an audit trail in the files of where they came from.

Remember that the actual files you are working on when you load a project into VB using Project Manager are the ones in the subdirectory shown as holding the current project.

An example of this behaviour is as follows:

Original Project directory: "c:\vb\projects\test"
New Version Name: "10000"
New Version Subdir: "c:\vb\projects\test\10000"
File Stamp: "Project Manager Version 10000"
Make Version 10000 Current?: User responds 'Yes'
Load Version 10000?:User responds 'Yes'

VB is loaded, the .mak file used is the one stored in the c:\vb\projects\test\10000 directory, and all project files used are from that directory. Any code added will be to this version, and the version from which it was created will not be affected.

**Handling Versions**

To view, change or delete the versions of a project, select "Versions|View Versions" , the first button to the right of the combo box, or the hotkeys "Ctrl + R". The list box on the project window form will now turn into a directory representation of the current project directory and any version subdirectories created under it.

**Change Version**

To change to a different version, after enabling versions as above, simply double click the version you wish to make current. You will be prompted to confirm the change, and on confirmation, will be asked if you wish to load the new current version into VB. The Current Version text panel will reflect the new current version.

**Delete Version**

To delete a version of a project, enable Versions as described under View Versions above. Then, hold down the shift key, and left click on the version you wish to delete. The mouse icon will change to a drag icon. Now drag off the project window form onto the MDI Parent form. As soon as the

mouse icon is over the MDI Parent form, a dialog will pop up asking you to confirm the delete. On confirming it, the program will delete all the files in that subdirectory and remove the subdirectory. Note that if it is the current version, the program will not allow you to delete it until you make another version current. You also cannot use Versions to delete the original project directory and files.

**This function actually does remove the version files and the subdirectory** containing them from your hard disk, so **be sure you are removing the correct one**.

**Add Version Comments**

You can add comments to the database at any time by selecting this item from the Versions menu. You will be prompted with a form to enter your comments in, and the date and version number will be added to the database. This assists in tracking changes to a project.

**Return to File List**

_____You can return to the list file view by using the fourth button to the right of the combo box, selecting "Versions|Disable Versions", or using the hotkeys "Ctrl + O".

# Code Move

## Overview

The code move functions allow you to keep a library project with commonly used functions in it, then copy those functions to a new project quicker than using copy and paste. It is more efficient than adding complete files, as you select only the procedures you wish to add to the new project. You can also use the move code within a project feature to balance the size of your code modules, or move code into logical groups of functions. Ensure that your library project has been edited, loaded and formatted by VB to enable Project Manager to correctly read it.

Code move is enabled by using the third button to the right of the command ribbon combo box, or selecting "Utilities|Code|Enable Code Move" or using the hotkey combination "Ctrl + E". You may move code in three ways:

**1./** Copy an entire .BAS or .FRM file between projects

**2./** Copy a single procedure between projects

**3./** Move a procedure from one file to another in the same project

To move code accurately, you should should run full updates on the projects you wish to move/copy code between. Select "File|Update Project" or use hotkeys "Ctrl + U". If the "Full Update" check box is not checked, the program will not run the code move functions. Full updates are necessary immediately prior to moving code to ensure the procedures actually still do exist in the files they are shown as being in. After you have moved or copied code, load the project into VB, run it in design mode, then save it. This allows the files to be correctly formatted for Project Manager to read them, as Project Manager embeds CRLF pairs when performing these operation.

After fully updating the project or projects involved, open two project windows. If copying code between projects, set each window to display a separate project, and if moving code in a project set each window to display the same project. Activate the source project window by clicking its title bar, and click the third button to the right of the combo box to enable code move in that project. The list box will change to display a file list with folder icons. Do the same with the project window form for the target project. If the file list of the second project window does not change to a file list with folder icons on one click of the command button, click the button again.

### Copying Files Between Projects

If you are copying a file between two projects, click on the .mak file name in the list of files in the target project window. Now activate the source window, and left double click and hold on the file you wish to copy to the target project. The icon will change to a drag folder. Now drag the file over the file list on the target project and release the mouse button. A dialog will ask you to confirm that you wish to add the file to the target project. Choose Yes to do so, No to cancel.

### Copying Procedures Between Projects

To copy a single procedure between projects, activate two windows as above. Highlight the file

in the target project window file list that you wish to add the procedure to. Activate the source project window and click on the file containing the procedure you wish to copy. The procedures in the file will now be displayed. Left double click and hold on the procedure you wish to copy, then drag over the target file list. On releasing the mouse button, you will be prompted to confirm the addition of the code to the new project. Choose Yes to do so, No to cancel.

**Moving Code Within a Project**

To move code in a project, again enable two windows as above, but this time with the same project. In one window, click on the file you wish to add the code to. In the other window, click on the file which contains the code you wish to move. Now left double click and hold on the procedure you wish to move, and drag it over the file list on the target project window. Release the mouse button, and answer Yes or No to the prompt to move the code. This procedure removes the selected code from the source file and adds it to the target file.

Turn off Code Move by selecting the fourth button to the right of the combo box, or selecting "Utilities|Code|Disable Code Move".

Remember to load the project into VB, run it and save it after using the code move functions.

# Error Handler Information

Many developers prefer to add error handling code last in a project. If error code is added during the developement phase of running the project under VB, errors will be trapped and you will not be able to discover the exact line in a procedure that is causing the error. Thus, error handlers are often added last. This can be very tedious work, and is the reason why these functions were built into Project Manager.

Select "Utilites|Errors" to access the following three menu items:

**Write Error Handlers**

_____Select this function to write error handlers in the current project.

When running the Write Error Handler function, you will see a dialog box which tells you what your chosen options are regarding type of error handlers and confirmation of writes. Selecting Cancel at this dialog will abort the process and allow you to change your options if desired.

**Comment Out Error Code**

This function will read through your project for lines that begin with "On Error" and add comments to them. This will effectively disable your error handlers. When you load and run the project under VB design mode, any errors will jump to the offending line in code, enabling you to continue work on projects with error handlers already in place.

**Remove Commented Out Error Handlers**

This function simply removes the comments from "On Error" lines, so you can compile your project with the error handling code intact.

Whenever using these two options, unload the project from VB before running them, then reload it for the changes to take place.

**Confirming Error Writes**

Checking this box in the Options form, Error tab, will cause the program to ask you to confirm each error handler write for each file and each procedure it finds in your project which does not contain error code. This enables you to selectively add error trapping. If not checked, the program will simply write the code as it checks the files with no input from the user.

**Information on Error Handling Code**

The two types of error code the program can write are described below. Remember that these are basic error handlers, and cannot anticipate all the potential errors that could be generated by a program, such as files left open, database objects not closed, etc. Provision for these types of errors will have to be added manually, if desired.

To set the type of error code you wish to enable, select "Utilites|Set Options" and click the Errors tab.

**Use Message Box**

Select this option to enable the MessageBox error handler. The program will write an error handler in each procedure which will trap errors, display a message box of the description and number of the error, and then return control to the user without crashing the program. This is the most basic error handler which will keep your program from crashing unexpectedly on the user, but cannot anticipate types of errors which may occur and take appropriate action to recover from them.

The project title must be entered in the project window when running this function as this information is used for the title bar of the message box in the error handlers.

**Use Log Error**

Select this option to enable the Log Error error handler. This will add a function in your project to write an error log text file in your project directory. The text file will record the error that occurred, the error number, the function that caused it, the form containing the function if a form module, and the time it occurred. It will provide the user a message box advising what the error was, then exit and return control to the user without the application crashing. Again, this error code will keep your app from crashing, but is not able to anticipate specific errors and take appropriate action to recover from them.

A sub called "LogError" does the actual writing of the error log text file. When the error handler writing code is running it will look for a Sub with the name of "LogError". If you have such a Sub or Function in your project it will prevent the code from writing the log error function. If it does not find the "LogError" Sub, it will add a file to your project called "ERROR.BAS". This file will contain a declaration of a Global Variable "Global Procname as string". This variable is added to each of your functions, and is passed as an argument to LogError. It will add this module to the .MAK file for your project.

You can open the project, decide whether you wish to keep the ERROR.BAS file, or add its contents to another .BAS file in your project. If you have only one form in the project, you can add the contents of ERROR.BAS to the form module, and declare the variable "Dim Procname as string" in the declarations section of the form. If you do wish to remove "ERROR.BAS" from your project, make sure you add the Global variable to your project in a .BAS file as "Global Procname as string", and copy and paste the code for Sub LogError into a module. You can then remove the "ERROR.BAS" file from your project.

The presence of a Function or Sub in your project called LogError will cause Project Manager to not write the function LogError function and add ERROR.BAS to your project. In this event, select "Edit| Copy" from this Helpfile menu to copy the following code and paste it into your project. Don't forget to add the variable "Procname as string" as a Global variable as well.

```
Global Procname as string

Sub LogError (aNum as integer, Procname as string)

    Dim FileNum as Integer
    FileNum = FreeFile

    MsgBox Error(aNum) &" "& aNum, 16,"Test"
    Open App.Path & "\errlog.txt" For Append as FileNum
    Write #FileNum, Now, Error(aNum), aNum, Procname
    Close #FileNum

End Sub
```

If you have standard items through your code, such as meter bars or status bars, you may wish to include code in this error handling sub to zero, hide, close or take appropriate action to reset these items

and other interface items, such as the Mousepointer.

### Error Report Categories

_____After writing error code, the program will offer to display information on the error code added to the project in a printable report. The following information is contained in the report.

**Error Handler Commented Out**
This list box shows procedures in which the error handler exists but has been commented out.

**Error Handler Not Added**
This list box shows procedures in which error handling code was not added at the request of the user.

**Error Handler Added**
This list box shows the procedures to which error handling code was added at the request of the user.

# Unused Code

It is critical that these functions have the latest updated information on the project, and will offer to update the project before running this code. Accurate results can only be obtained if the project is updated before running any of these functions. If you run the three functions in order, printing the reports as you progress, and make no changes to the project between searches, you will only have to update the project once prior to running the functions.

The report generated after running these functions displays information on unused code in your project. It will show the appropriate information, depending on whether the user wanted to check for unused Constants, Variables or Procedures. These functions may take some time to run, depending on the size of your project. It is usually only necessary to run these when cleaning up code prior to a final test & compile of the project.

**Check Unused Procedures**

This function will iterate through all the files in your project. It will check each procedure or declared external function to see that it is used in the project. At the end of the procedure, it will advise you via a report what procedures do not appear to be used in your project. Before deleting them, use VB's search function to ensure that the procedure is not in fact called. This function may take some time while running, and a meter bar will inform you of its progress as it occurs.

**Check Unused Variables**

This function will check all files for declared variables that are not used. It will advise you in a form when finished what variables do not appear to be used. Use VB's search function to ensure that the variables are not in fact used before deleting them.

**Check Unused Constants**

This function checks all files for constants and determines which are not used. The cautions above regarding deleting apply here as well.

# Register

Project Manager 3 for VB 3 Win is written by:


Bud Street,
A.F. Street Consulting,
R.R.#1,
Peterborough, ON, Canada
K9J 6X2

Phone 705-292-6621
CIS 72152,333

Michael Ruane, Lower Hutt, New Zealand CIS 100236, 3711 provided much input into the interface design, version and faults features, and testing of the program. His assistance was invaluable in making the program interface and features more professional and functional.

Register the program by sending $40.00 U.S. funds, or $55.00 Canadian funds to the author at the above address, cheque or money order made payable to Bud Street.

Registration of the program fully enables it as described below, and entitles you to support via e-mail, or telephone. E-Mail support is the preferred method.

To register, select "Help|Register" to show the Registration form. Fill in the blanks on the form, select the "Print Form" button to print the form, then send it to the above address, along with a money order in the above amount.

You may also register online in CompuServe SWReg Forum, GO SWREG and enter Registration ID # 3649 when prompted for the registration number of the program. Provide the information required as prompted. As soon as notice of registration is received from CompuServe, the registration number will be e-mailed to you, along with the details of your name, address, etc., which you provided to CompuServe.

Site licences are available at a discount, please contact the author direct for information on pricing.

## Enabling Registered Version

After registration is received, you will be given a registration number to enter in the appropriate place on the registration form. Enter it exactly as given, and ensure the details in all other fields match those which are provided to you with the registration number. The program will now be fully enabled. Ensure that the information in the registration form is identical to that sent in when registering. The registration number is generated based on the information you provide when registering. If you alter anything in this form after registering, the program checksum will not be correct. If this process fails, carefully check all fields for matches to the supplied information before contacting the author for assistance.

The unregistered version is fully functional, however, there are limits on the usage of some functions. In addition to seeing the registration form whenever the program loads, the following restrictions apply. Unregistered programs will only allow you to work with two projects at a time. Printing of reports is disabled for all reports except the times report. You will only be able to create two versions of a project. Code move is limited to 1 file copy and 1 procedure copy/move. These restrictions allow you to sample the program's features enough to get a feel for what the program can do, but provide an incentive for you to register the program and fully enable it. The "try before you buy" concept is great, so is the concept of getting paid for one's product when people use it regularly.

If you evaluate the program and do not like it, or have problems with it, please take a moment to send an E-Mail advising why you did not like it, or what problems you had with it. If it generated an error log file, I would appreciate a copy of the text of that file as well. If there is something you would like to see in the program, please advise. This information will assist in further developing the program and making it more useful to VB programmers.

## Liability Disclaimer

This program has been tested on various systems and has worked without problem. However the author assumes no liability for whatever this program may do on your system, and you use it at your own risk.

Always ensure you have current backups of your projects on hand. The program has been tested thoroughly, but unforseen errors can occur when dealing with diverse environments, and it is best to maintain frequent backups as insurance.

Due to the myriad of possible system configurations, programming styles, third party tools, etc., it is suggested that you verify any information given to you by the program using VB's search functions before you delete any code. Verify that the program functions properly with your projects on backed up projects first. While it has been accurate on the test projects, including the VB sample projects, in picking out unused code and reporting information, verify its' results on your code before you act on it.

The program has been tested on various stand alone machines with several different drive configurations, as well as on a small Windows For Workgroups network. It has also been tested under VB Standard (with and without Access 2 and the Compatability Layer) and VB Pro (also with and without Access2 and the Compatability Layer), and functions under each configuration with the appropriate .exe file and .mdb file.

## Future Enhancements

If there are any additions to the program which you would feel would make it more useable, please advise. Any constructive suggestions you have regarding current program functions, operation or interface are welcome also.

# Times Menu

The Times Menu items allow you to open and close VB with a project loaded, and change the description of the task you are performing on the current project.

**Open Project(Record Start Time)**

This menu item will load VB with the project shown in the currently active window loaded. It will record the start time and, if you have checked Use Task Descriptions on the Options form, will pop up a dialog which will enable you to record comments on the task which you are working on currently. You may also select a Fault Record, if any exist for this project, for the current task. This enables you to record time spent on various fault corrections during program development.

**Close Project(Record Stop Time)**

_____This menu item will close VB and unload the project shown in the currently active project window. When you use Project Manager to load a project, it records which project you loaded. If the project you are telling it to close does not match the one recorded internally as last opened (ie you closed VB manually and loaded another project or you have move Project Manager's active window to a different project), a message box will tell you what project you loaded, and let you record the stop time for the correct project.

**Change Task(Record Stop, Start, Description)**

_____If you wish to keep a log of all activity on your project by the task, use this menu item to start a new record of the task being performed. This will not close VB or unload your project, it will simply close off the open time record, start a new one, allow you to add comments to the database on what task you are now working on, and select a Fault Record to relate the task to.

**View Times Report**

_____This menu item loads the Times Report for the project shown in the currently active Project manager window. This is an editable report. Date and Times are displayed as "General Date", which formats it according to your date/time settings in Windows Control Panel, International.

**Task Description Dialog**

_____This dialog box allows you to enter comments describing the current task which you are working on. If you have enabled "Use Task Descriptions" in Options|Descriptions, this dialog will appear whenever you record the start time. You can also select a fault to relate this time record to, if fault records exist for this project.

Click OK to accept and close the dialog. If you click Cancel, it will cancel the changes and abort the times process.

# Fault Recording

Fault Recording allows you to track reported faults on your projects, and to track the time spent on fault correction. You can access the Fault Recording data entry form from the "Utilities|Fault Recording" menu items. Under this menu item are the following two items:

**Add New Fault Record**

Selecting this item will load the fault recording data entry form with a new record started. The Project ID will be entered, as will the current date and time. These fields are editable on this form in case you wish to change this information. Be particularly careful to enter the correct Project ID number if you choose to enter it manually. Improperly recording this item will cause reports run on this table and the times tables to display incorrect information.

**Edit Fault Records**

Selecting this item will display the fault recording data entry form with the records loaded for the currently displayed project. You may browse the records and edit the data in this view. When you run the Faults Report under the Reports Menu, the description and method corrected fields are editable. Click on the grid to enable two edit windows for these items.

## Fault Recording Data Entry Form

This form contains buttons to Add New Records, or Undo the information when you have activated the Add New Record function, either by clicking Add New Record or by loading the form from the menu item Add New Fault Record. The OK button closes and unloads the form.

If you attempt to switch away from this form, you will be prompted to either unload the form or reload it. You will also be prompted to check to see that your Project ID number is correct for the fault records you wish to add or enter.

### Fields

**FaultID**

The Fault ID field is not editable and is provided for internal program use only.

**Project ID**

This field records the ID number of the project the fault record currently displayed relates to. You may edit this field manually, but do so with extreme caution. Incorrect values here will cause problems with reports later.

**Date/Time**

This value is set when the form is displayed from the Add New menu item. It is editable manually. It is meant to record the date/time the fault was reported.

**Reference**

This field allows you to assign an identifier to the fault which will be meaningful to you in later tracking the fault. It could be a code, or a specific part of the project such as a form or control, or whatever you wish to identify the fault easily to you.

**User Name**

Is meant to hold the name of the person reporting the fault.

**Version**

The program version number the fault was corrected in.

**Description**

A description of what the fault was, how it was generated, etc.

**Corrected**

A description of how the fault was corrected, by whom, etc.

---

# About Form

Project Manager includes an about form with some information built into it to return the amount of free memory, free resources, and the size and free space on each non-removable drive on your system.

You can use this form as a resource checker for your applications. After you have compiled an application, run it while Project Manager is running. Then as you use your application, you can switch to Project Manager and click the "Refresh" button on the About form which will re-check the above information and display additional text boxes on the form. The text boxes on the left of the form under the label "Current" give the current sytem information, while the text boxes on the right under the label "Previous" give the system information that existed prior to clicking the "Refresh" button.

Do not load this form if you have an Audio CD in a CD-ROM drive accessible to your computer, as this will cause your computer to lock up.

# Glossary


"VBPROJ1.MDB"
"VBPROJ2.MDB"

**C**

CRW.EXE
CRWACC20.EXE

**D**

DISKINF.DLL
Do not

**E**

error handler code
explicit declaration

**F**

fully qualified paths

**H**

Hotspot_1

**I**

Important Note

**M**

MDI

**O**

Option Explicit

**P**

PKZIP.EXE
PMGVBX.ZIP
PROJMGR.INI

**R**

Report Files

**S**

secure database
SYSTEM.MDA

## SYSTEM.MDA

File used by MS Access Jet Database engine to establish security.  Project Manager requires its own system.mda file stored with the .exe file.

## Report Files

Supplied with Project Manager, with file extension "RPT". Crystal Reports report files for viewing and printing contents of projects, Times worked on projects, Versions comments and Projects stored in database.

**CRW.EXE**
Crystal Reports executable file, normally found in directory VB\Report

## CRWACC20.EXE

File supplied with Compatability Layer to modify Crystal DLL's and CRW.EXE to enable Crystal Reports to read Access 2 databases

**PMGVBX.ZIP**
Zip file containing VBX's and Crystal Reports runtime files for VB Standard users

## Important Note

A note that you ignore at your own risk

## unregistered versions

Version of Project Manager for which the author has not been paid.  Send in your $45 US today, or Register online through CompuServe GO SWREG

**VBPROJx.MDB**
Generic name for the Project Manager databases, either the Access 2 or Access 1.x version

## PROJMGR.INI

File stored in Windows directory, records information on user preferences for Project Managers options

## secure database

Database secured using Access security features, requires a password and UserID to access the database.  Enabled in this case to prevent program failure due to someone putting their fingers where they shouldn't.

## explicit declaration

Declaring variables before using them in code.  Done with Dim or Global declaration statement in procedures or modules.  If you need to know this, you are over your head and should read your VB documentation right now!

## MDI

Multiple Document Interface - has a parent background window and can have numerous child windows open at any time

**fully qualified paths**
a full path to an executable, eg c:\vb\reports\crw.exe

## error handler code

Code to trap run time error, handle them and allow the program to continue execution without making an ugly exit and possibly taking Windows with it

## versions

Duplicate copies of a project created by Project Manager in response to user request, see the Versions help topic for more information

## Do not

This means don't do it, or you will be sorry.

**Hotspot_1**
Opens a new Project Window

**"VBPROJ2.MDB"**
Access V2 database for use with Pmgr32.exe file for VB users with Access 2 and the Compatability Layer installed.

**"VBPROJ1.MDB"**
Access V1.1 database supplied with Pmgr31.exe file for VB users WITHOUT Access 2 and the Compatability Layer installed.

## Option Explicit

Forces declaration of variables.

## PKZIP.EXE

File compression program, you need version 2.04G for the function "Zip up Project Directory" to work.

## DISKINF.DLL

Tiny .dll which returns the size and free space on a disk.  See the helpfile topic "Installation" for information on using this .dll in your programs.