Project Analyzer 4.0
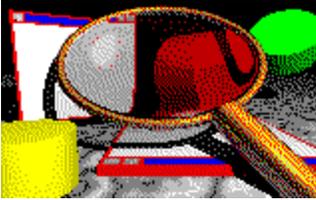

Overview of Project Analyzer
Introduction
How it works

Version history - What's new?

Registration, contact information and future versions
Pricing
Features in the registered version

Menus

File
Open project...
-
Generate constants module...
Archive project files...
-
Exit

View
Find procedure...
-
File details
Procedure details
Variables and constants
Hypertext
-
Metrics
Call tree

Report
File details
Procedure details
-
Module call tree
Procedure call tree
    All procedures
    Selected file only
    Selected procedure only
-
List files...
List procedures...
List variables and constants...
-

Name shadowing
Needless globals report
Library report
-
Problem report
Design quality report
Project summary

Options
 Report to
   Display
   Printer
   File...
Printer setup...
Printer font...
-
General options...
Hypertext options...

Tools
Save data for Super Project Analyzer...
Super Project Analyzer
-
Project Printer

Project Analyzer was written in Visual Basic 5.0 Pro
by Tuomas Salste, Finland
© 1994-1997

Visit the Visual Basic Shop WWW page
http://www.aivosto.com/vb.html
vbshop@aivosto.com
Shareware Visual Basic tools

Project Analyzer 4.0 - Helpfile generated by VB HelpWriter.

List files

Product Analyzer can produce list of files that belong to your project. To get a list, choose **Report|List files**. A dialog box will appear.

You can get lists sorted by:
- File name
- File type

If you want just a list of file names, leave the **Details** box unchecked. If you check it, the list will contain the following additional data:
- Comments
- Which files need procedures in this file
- Which files this file needs

Note: You get the commented lines in the (declarations) section of your .BAS, .FRM, .TXT or .GLB file. The program shows those comment lines that come before the first non-comment line, like this:

```
' This is a function module
' These comments will be reported

Dim Text As String
' This comment will not be reported
```

The file list goes to the <u>output</u> device defined with the **Output goes to** radio buttons.

Project Analyzer 4.0 - Helpfile generated by VB HelpWriter.

List procedures

Product Analyzer can produce list of procedures that belong to your project. To get a list, click the **List procedures** button or choose **Report|List procedures**. A dialog box will appear.

You can select if you want the following data included in the procedure list or not:
- Comments
- Which procedures use this procedure
- Which procedures this procedure uses

In the registered version there are also options to list procedures that are or are not needed by your program. This is useful for finding **dead code**.

Note on comments: The program will show those commented lines at the beginning of each sub/function that come before the first empty or non-comment line. Commented lines before and after the sub/function declaration line are shown, as well as procedure description (VB 4.0), if available. An example:

```
' This sub was created by N.N.
SUB MySub (Byval x As Integer)
Attribute VB_Description=Prints x + 5 on the form
' This sub does the following:
' It takes the parameter x and ...

' This line is not shown any more
Form1.Print x + 5

END SUB
```

The procedure list goes to the <u>output</u> device defined with the **Output goes to** radio buttons.

Project Analyzer 4.0 - Helpfile generated by VB HelpWriter.

Archive Project Files

Project Analyzer helps backing up your program files. Currently it supports two archiver programs, namely **ARJ** (current version: 2.41a) and **PKZIP** (version at this time: 2.04g). You can also easily build file lists of your project files in order to manually copy/archive them without effort.

Pack Project Files Dialog

Select files to be packed (or files whose file name are to be saved in the list file) in the list box. By default, all VB files belonging to your project are selected, but no DLLs. These are the files that you most probably want to save.

If you want to add files to the list you can click the Add button. Files ending in **.lst** are considered to be list files that contain a list of more files you'd want to add to the list box. You can create these .lst files with any text editor or with Project Analyzer itself.

The **ARJ Command** and **PKZIP Command** fields contain the commands and command switches used to create new archive files.

Sample ARJ Command: ARJ a MYARCH.ARJ !MYPROJ.LST
Sample PKZIP Command: PKZIP MYARCH.ZIP @MYPROJ.LST

You only have to supply the blue part of the command line; Project Analyzer does the rest. Now you can pack by pressing one of the **Pack** buttons.

By default, a temporary file is used to tell the archiver program what files to pack. If you want to save that list file for later use you can click the **Select** button.

Having named a list file, you can save the list file only if you want. In other words, a new button name **List files only** will appear.

Project Analyzer 4.0 - Helpfile generated by VB HelpWriter.

Introduction

Project Analyzer is a Finnish <u>shareware</u> tool designed to help creating, maintaining and reporting applications developed in Visual Basic 3.0, 4.0 and 5.0.

Project Analyzer makes a full, two-phase source code analysis. It collects detailed information about all the files, procedures, controls, variables, constants, Types, Enums, classes etc. in the VB project. The results can be viewed on the screen, in textual format in your favourite word processor, as a Help file, in a WWW browser, or on paper.

System requirements

- Windows 95 or NT (or later).
- Visual Basic 5.0 installed, or the run-time libraries. Visit **http://www.aivosto.com/vbcatalog.html** to download the needed DLLs.
- Basically, if you can run Visual Basic, you can run Project Analyzer.

Windows 3.x: If you are using Windows 3.x, get Project Analyzer version 3.1 from **http://www.aivosto.com/vbcatalog.html**. This will run in 16-bit environments.

You only need one version of Project Analyzer for analyzing VB 3.0, 4.0 and 5.0 projects.

Features

Full source code analysis
- <u>File details</u>: procedure and control lists, file level cross-references, lines of code, comment to code ratio and other <u>metrics</u>
- <u>Procedure details</u>: cyclomatic complexity, lines of code, and other <u>metrics</u>, procedure level cross-references
- Find dead and live procedures, variables, constants, Types and Enums. Try the <u>Problem report</u>!
- Variable and constant declaration information, both dead and live
- <u>Call trees</u>
- <u>Project metrics</u> like cyclomatic complexity, comment to code ratio, ...

Reports for maintaining the project
- <u>Sub and Function lists</u> with comments for reporting purposes
- <u>File lists</u> with comments
- <u>DLL reports</u>
- <u>Problem report</u>
- <u>Project summary report</u>
- <u>Variable and constant lists</u> with dead or live variables & constants
- Design quality report
- Needless globals report
- Name shadowing report

Other powerful features
- Project file <u>archiving</u> with PKZIP or ARJ
- <u>Hypertext</u> style code viewer
- <u>Binary extension file viewer</u> (for FRX and other similar files)
- Removing unused constants from CONSTANT.TXT and DATACONS.TXT

See <u>version history</u> for new features.

Why? Why???

Maintain phase costs rule the software industry. When developing applications, especially those that someone else has to maintain, it's important to keep track on what each sub/function does, and what other procedures they need to work.

This is where Project Analyzer can help you. With it you can produce detailed cross-reference reports with supplied comments. You can view call trees. And you can find dead code, dead variables, and dead constants, and various other problems too. You can get the documents on the screen, on paper, or even on your company Intranet as HTML reports.

Maintaining programs is **hell**. But Project Analyzer will turn it into **heaven** :-)

How to use it?

Using Project Analyzer is simple. Open a **.mak** or **.vbp** file with the <u>Open Project</u> command in the File menu and wait. Project Analyzer will collect information about the **.bas**, **.frm**, **.frx, .cls, .dll** and other VB files in the background.

Note: If you are using VB 3.0, remember to save all your files in text format.

The analysis consist of two phases. The first phase collects the most data, and the second one will check for cross-references and find dead code etc. If a menu command or a button is greyed, don't panic, just wait for the analysis to end.

Hint: Test Project Analyzer by opening the included TEST30.MAK, TEST40.VBP, or TEST50.VBP files (named after VB30, VB40 and VB50). These are pure demo projects for you to see how Project Analyzer works. You can of course use your own projects too for this purpose. Project Analyzer does **not** write anything into your code, so your code will not get messed up.

When the analysis has ended, just double click a procedure or a file to see the results. Remember to push the **Hypertext** button. Don't forget to examine the View and Report menus either, many of the most powerful features reside there.

<u>Registration</u>
<u>Collecting information</u>
<u>Back to Contents</u>

Project Analyzer 4.0 - Helpfile generated by VB HelpWriter.

Collecting Project Information

When you start Project Analyzer, or when you choose **Open Project...** from the **File** menu, you get a dialog box asking you to choose a project file (a VB **.mak** or **.vbp** file).

Project Analyzer examines the project file to find out what files belong to your project. After determining the type of each file, the program further looks inside these files.

What information Project Analyzer searches for in the files:

Project Analyzer examines the **.bas, .frm, .frx, .cls, .dll** and other files belonging to your project. As an exception, Project Analyzer doesn't analyze .vbx, .ocx, .tlb nor .res files.

If you use VB 3.0, remember to save the files in text format. Project Analyzer can't read VB files saved in binary format, other than the .frx files. It will analyze .dll files based on the declarations in your basic code. - If the program finds a binary file it will simply ignore its contents. This will result in incomplete analysis, like showing dead variables that aren't dead in reality.

Project Analyzer builds a list of all files and procedures (i.e. subs and functions) that your project consists of. Code is analyzed at form and procedure level to list the controls and the variables, constants, Types and Enums. Furthermore, procedure and variable, constant, Type and Enum cross-references are examined.

Note: Information is collected in the background. So, when Project Analyzer is examining your files you can do other things like running other programs - or you can view the information the program has collected so far. There are some things you can't do when the collecting is in progress, like getting a list of all procedure calls - that's because the program hasn't yet finished collecting that piece of information!

The program reads your files twice. You'll notify when it's ready by looking for the word "Ready!" at the lower left corner of the program window. Should you want to stop the process, you can press the Stop button. To speed up the collecting process, the program checks if you have pressed the Stop button only every now and then, so be patient!

Use Option Explicit. Project Analyzer will get better results if you have declared all your variables. This is good practice anyway.

Project Analyzer 4.0 - Helpfile generated by VB HelpWriter.

Report Formats

Project Analyzer has very powerful, formatted report options. You can get the reports
a) On the screen
b) On paper
c) On the Clipboard
d) In a file for your favourite word processor
e) In a help file
f) On your company Intranet as HTML files

Use the **Options|Report to** menu command or one of the **Report to** radio buttons to select the report format.
Depending on what you select, you either get a formatted report, or a text-only version.

1. To the display
This is the default. The output goes to a special Display window where you can read the text. You have 2 options to take:
a) A formatted report
b) An unformatted, plain text report (from which you can copy parts to the clipboard)
You can select the type of the report in the **Options|General options** window. More about Options

There are 3 of these Display windows available. You can use them side-by-side.

2. To the printer
If you want, you can choose the output to go to the selected Windows default printer. This report is formatted.

To select a printer, click **Options|Printer setup**. To choose a printer font, click **Options|Printer font**.

3. To a file
Project Analyzer can direct its output to:
a) a Windows text file (plain text)
d) a Rich Text Format (RTF) file (formatted report)
e) an RTF file ready for Help Compiler to turn the report into a Help file (formatted report)
f) an Internet Hypertext (HTM) file (formatted report)

You can even tell Project Analyzer to run an editor, browser or the Help Compiler to open or process the report after it has been created. To do this, just check the **Run browser/editor after report** check box in the Save report dialog box.

Running a browser/editor/Help Compiler

Project Analyzer needs to know which program to run after creating a report. By default, the Windows default associations are used. Press the right arrow button in the Save report dialog box to see which program will be launched.

Possible choices for editors/browsers/Help Compiler:
a) TXT: Windows Notepad or any other text editor
b) RTF: Microsoft Word or Windows WordPad, or any other word processor supporting RTF
c) RTF for Help Compiler:
    1. Microsoft Help Compiler (HC.EXE, HC31.EXE, HCP.EXE, comes with VB 3.0 and 4.0, and is in your \VB\HC directory).
    2. Microsoft Help Workshop (HCW.EXE, comes with VB 5.0, and is in the \Tools\Hcw directory of the VB5 CD-ROM).
    Use of the Help Compiler is recommended instead of Help Workshop, because Project Analyzer writes

a .bat file script for the Help Compiler, which will automatically compile the help file.
d) HTML: Netscape or Explorer, or any other WWW browser

You can change the default by writing your own command line, or by associating the file name extension (**.txt**, **.rtf**, or **.htm**) with your browser/editor. In Win 3.x, you do this from **winfile.exe | File menu | Associate**, and in Win 95, from **Control Panel | View menu | Options | File types**.

Multiple reports in one file

You can add multiple reports in one file by selecting an existing file. If you have selected RTF for Help Compiler, Project Analyzer will also create a Table of Contents page that shows all the reports in that help file.

Note: When producing large reports (over 32kB in size) you can get an "Out of string space" error when the output is going to the unformatted display window. The report will be truncated to 32 kB. The solution is to use any other report type that allows reports of any length.

Project Analyzer 4.0 - Helpfile generated by VB HelpWriter.

About Project Analyzer

Contact information - Future versions - Registration

Contact me!

Project Analyzer was written by Tuomas Salste. It is distributed by a software company called **Aivosto Oy**.
Send email about Project Analyzer to me to **vbshop@aivosto.com.**

You may also want to visit the VBShop WWW page **http://www.aivosto.com/vb.html** Product information will be available through that page. You will also find information about other Visual Basic tools there.

New versions

Project Analyzer is still under further development. New versions with further enhancements can be expected in the future. New features will mainly be available in the registered version only.

A registration is valid for one major version number without any need to upgrade. So registered users of version 4.0 will get version 4.1 for free, for example, but not version 5.0. Special upgrade prices apply.

Distribution of the free updates is done solely on the Internet, by WWW or FTP. **Getting upgrade versions requires an Internet email address.**

More about pricing

_____

Shareware registration

Project Analyzer is **shareware.** What is shareware? In shareware, you can try before you buy. The unregistered version of Project Analyzer is freely distributable. You can use it to evaluate Project Analyzer, to see if (or why) you need it. The unregistered version is restricted in some ways. To really use Project Analyzer, you'll need to register. When you pay the registration fee, you get keywords for the full, registered version of Project Analyzer with more features.

Pricing for Project Analyzer

How do I register?

Be sure the read both of the following:
1. The general registration instructions in REGISTRA.HLP. They apply to all Visual Basic tools distributed by Aivosto Oy.
2. The special registration instructions below. They apply especially to Project Analyzer.

Payment method

The possible payment methods are described in REGISTRA.HLP. All major credit cards, U.S. checks, cash, and some other methods are accepted.

Registration keyword

After registering, you will get a registration keyword that will enable the registered features. Input this keyword in the About box when Project Analyzer starts, and you will be registered.

The same applies to Super Project Analyzer and Project Printer add-ins, but you input the keyword when the add-in starts.

Note: **There are no deliverables. You will not get the program physically in a box. You will get it via email and WWW/FTP, if you have access to the Internet.** In case you don't have an email address, you will receive a letter with the keywords.

Project Analyzer 4.0 - Helpfile generated by VB HelpWriter.

Hypertext

(version 1.1 onwards, <u>registered version</u> only)

You can see your code in hypertext form (like text in this help file) by using Project Analyzer's hypertext feature. The hypertext window is a read-only code window but with additional features. You can see called procedures easily by clicking at the underlined procedure names. If a constant or variable is <u>underlined</u>, you can jump to the declarations section where it was declared.

To open the hypertext window, select a file or a procedure in the **Files** or **Procedures** list box and either press the **Hypertext** button or select **View|Hypertext** hypertext in the menus. Another method is to press the **Control** key down and then click in the **Files** or **Procedures** list box. Still another method is to select Hypertext window as the default in <u>General Options</u>.

To see the procedures that call the shown procedure click the **Called by** button.

Hint: Try right-clicking highlighted variables, constants, procedures, Types and Enums. You will get a menu with links to their definitions, and in the case of global and module-level definitions you will also see where they are referenced and assigned to!

Options
You can change the colors of different types of hotlinks by selecting **View|Options**. If you want, Project Analyzer can ~~**overstrike**~~ dead variables too. Overstrike is available in the registered version.

Hint: <u>Project Printer</u> can save your source code in hypertext format into a **.hlp** file. The result is quite similar to the Hypertext window, but you no longer need Project Analyzer to view the file.

List variables and constants

Product Analyzer can produce list of the global and module-level variables and constants that belong to your project. To get a list, choose **Report|List variables and constants** in the main window, or press the **Report** button in the <u>Variables and constants window</u>. A dialog box will appear.

You can get two kinds of lists:
1. List all **global** variables and constants in alphabetical order
2. List all **global** and **module level** variables and constants ordered by file they appear in

The file list goes to the <u>output</u> device defined with the Output goes to radio buttons.

Note 1: Variables and constants information is available when analysis phase 1 is complete.

Note 2: You can get more versatile variable and constant lists from the <u>Variables and constants window</u>.

Version History

This version history lists the major changes.
See a more detailed version history in file VERSION.TXT.

Version 4.0 (Autumn 1997):
- Added support for Visual Basic 5.0
- User defined Types and Enums included in the analysis
- Call tree enhanced

Version 3.1 (Spring 1997):
- Super Project Analyzer
- Project Printer
- Project metrics
- Name shadowing report
- HTML reports
- Help file reports
- Lots of enhancements and bug fixes

Version 3.0 (Summer-Autumn 1996):
- Added support for Visual Basic 4.0
- Project Analyzer now needs Visual Basic 4.0 DLLs to run. If you only use VB 3.0, please visit VBShop
(**http://www.netgate.net/~vbshop/vb.html**) to download the required DLLs.
- Class modules supported
- Hypertext window enhanced
- Call tree enhanced
- Needless globals report
- Variables with no type in Problem report
- A lot of enhancements to various features

Version 2.1 (September 1995):
- Project summary report
- Generate constants module command
- Modifications to look better with Windows 95
- Enhancements in the hypertext window
- Average cyclomatic complexity (version 2.1.05 onwards, registered version only)
- About 25% faster analysis (appears in the second phase) due to improving PROJECT.DLL
- Formatted reports on the screen too (version 2.1.07 onwards)
- DLL Report (version 2.1.07 onwards, registered version only)
- Can handle larger projects than the previous versions

Version 2.0 (August 1995):
- The Display window is no more the only way of seeing the results of an analysis. Separate detail windows have been implemented.
- File details and Procedure details can now be viewed on the screen in a separate window in addition to the reports
- File details show the form's icon and controls too
- New Variables and constants window with information about references to them (dead variables & constants, referencing files)
- New Call tree window
- File level call trees added in addition to procedure level call trees
- FRX files can be examined too (just double click the FRX file in the main window)
- Shows declared functions in external DLLs
- Problem report (registered version only)
- Formatted reports using RTF files
- New statistics, like cyclomatic complexity (registered version only) and lines of code for each procedure
- Find command now in the Hypertext and Procedure details window too

Version 1.1 (March 1995):
- Find dead procedures (registered version only)
- Global and module level <u>Variable and constant lists</u>
- Procedural <u>call trees</u> (registered version only)
- <u>Hypertext</u> style code viewer (registered version only)
- Saving the results of the analysis for future use (registered version only)
- A **Find procedure** command to quickly locate a procedure

Version 1.0 (February 1995):
The initial release.

Call tree

(version 1.1 onwards, <u>registered version</u> only)

You can see the procedure and file dependencies in the form of a call tree. The tree is available in two formats:

1. Call tree in a separate window

Use the **View|Call tree** menu command to see an expandable view of the dependencies. The call tree types in this window are:
1.1. Module calls. This tree shows those other modules that are needed for a module to run. A module may use procedures, variables, constants, and also Type and Enum declarations coded in other modules. It may also call another Form's controls and properties.
1.2. Procedure calls. This tree shows the procedure calls inside your project.

The buttons

Pictures   Toggles pictures on/off.
Expand   Expands the call tree from the cursor onwards.
Hypertext   Shows the selected piece of code in the <u>Hypertext window</u>.
Report   Turns the tree into a textual report.

Max call depth at the bottom of the window shows how deep the call tree has been at a maximum. This number will increase as you expand the tree.

If a procedure calls itself or another procedure that then eventually calls the first procedure back, it will be marked **<recursive call>** in the tree.

2. Textual report

The reports are available using the **Report|Procedure call tree** and **Report|Module call tree** menu commands. These call trees are the same as on the Call tree window. Three subtypes of a procedure call tree report are available:
2.1. All procedures
2.2. Procedures in selected file only
2.3. The selected procedure only

Note 1: This call tree features are available in the registered version only. In the unregistered version you can see the call tree only for small projects.

Note 2: The call trees are available after analysis phase 2 has completed.

Note 3: You can see dependencies in <u>hypertext</u> or <u>list</u> form too.

Note 4: The program may hang if your call trees are extra large. In this case, see smaller call trees at a time. It is recommended not to take the All procedures call tree on anything but small projects.

File details window

(version 2.0 onwards)

The **File details** window shows information about a specified file in your project. The easiest way to see File details is to double click the **Files** list box in the main window.

Project Analyzer can show information about:
- Form files
- Basic modules
- Class modules
- DLL's
- FRX/CTX/PGX/DOX/DSX files (see the end of this topic)

Basic files

- Code lines
- Size in kilobytes
- File icon (if exists)
- File version: An internal version number of the file. This version is not necessarily the same as the version of your VB. Not all file types have a version number.
- Subs and functions: Name, type (functions only), lines of code, and 'Dead' status if the procedure is not used by your project.   - Double click a procedure to see procedure details.
- Variables, constants, Types and Enums defined in this file: Global and module-level definitions.
- Controls (Form files only)
- List of files that call this file
- List of files that this file calls (after phase 2)
- Various metrics (after phase 2)

Some of the information will not be available before the analysis phase 2 has been completed. Question marks '?' indicate that the analysis is not complete yet. To see the results after completion of analysis use the View| Refresh menu command, or press Ctrl-R.

DLL files (DLL, EXE, ...)

DLL's are analyzed like BAS files, except:
- Variables and constants are not listed
- Files that are called by this DLL are not analyzed
- The lines of code in a DLL cannot be analyzed
- DLL file version is not analyzed
Only those procedures that were declared with a Declare statement are shown.

Report and Hypertext

To see the file details as a textual listing, press the **Report** button. To see the selected procedure as hypertext, select the **View|Hypertext** menu command.

FRX/CTX/PGX/DOX/DSX files

Project Analyzer can examine the contents of a binary extension file that is there if you saved your Form file in text format (VB 3.0 has this option, later versions do it automatically) and the form contained pictures or any other binary data. In Visual Basic 4.0 and earlier, there are only FRX files available. In VB 5.0 and later, there are lots of other files that are essentially of the same type, for example, a CTL file is often accompanied by a CTX file.

These files include the binary properties of your controls. You can see what's inside a binary extension file by double-clicking it in the main window of Project Analyzer, or by selecting File Details.

Project Analyzer cannot show all kinds of binary data, just the most usual types of them. Some supported properties are:
- PictureBox.Picture
- Form.Icon
- Form.MouseIcon
- TextBox.Text (if it's a multiline box)
- ListBox.List and ListBox.ItemData

If Project Analyzer cannot show a picture, it will try to show the data as text.

You can extract a picture from your VB project by saving it to a file. Just click the **Save As** button.

Note: VBX, OCX and RES files are not analyzed by Project Analyzer.

Project Analyzer 4.0 - Helpfile generated by VB HelpWriter.

Procedure details window

(version 2.0 onwards)

The **Procedure details** window shows information about a specified sub or function. The procedure can be a basic procedure, an event or a library procedure.

The following information is shown:
- Procedure name and type
- Procedure declaration
- Procedure file
- The 'dead' state of the procedure (after phase 2)
- Lines of code (doesn't apply to library procedures)
- Procedures that call this procedure
- Procedures that are called by this procedure (doesn't apply to library procedures)
- Various metrics (after phase 2)

Some information is available only after analysis phase 2 has been completed. Question marks '?' indicate that the analysis is not complete yet. To see the results after completion of analysis use the View|Refresh menu command, or press Ctrl-R.

To see the information as a textual listing, press the **Report** button. To see the entire procedure code, press the **Hypertext** button.

Project Analyzer 4.0 - Helpfile generated by VB HelpWriter.

Variables and constants window

Project Analyzer can show a list of all global and module-level definitions either as a <u>textual report</u> or in a list box. To see the list box, either press the **Vars & consts** button in the main window, or select the **View|Variable and constant list** menu command.

You can choose from the following options:
- Show constants
- Show variables
- Show global definitions defined in the (declarations) section of a module
- Show module-level definitions defined in the (declarations) section of a form or module
- Sort by file
- Sort by name
- Show live definitions (those that are referenced somewhere in the program)
- Show dead definitions. Those ones are dead that are not referenced even once in the program, even though a dead variable may be assigned to.

To see the actual definition of a variable or a constant, press the **View declaration** button. That will bring up the <u>Hypertext window</u> with the corresponding (declarations) section. To view a <u>textual listing</u>, press the **Report** button.

To see which procedures reference or assign to the selected variable or constant, select the **View references** command. A list will show up with all assignments and references to that variable or constant.

Note 1: Variables and constants information is available when analysis phase 1 is complete. The reference statistics are available when phase 2 is complete. To display the information, close the window and reopen it after the analysis.

Note 2: To see procedure-level variables or constants, use the <u>Hypertext window</u>.

Project Analyzer 4.0 - Helpfile generated by VB HelpWriter.

Problem report

(version 2.0 onwards, <u>registered version</u> only)

Project Analyzer can provide you with a **Problem report** about your project. It lists some minor but sometimes annoying things.
- Basic files saved as binary - save as text to work with Project Analyzer **and** for better reliability with VB.
- Files without Option Explicit. It is good programming practice to always declare your variables.
- Dead procedures, i.e. subs and functions that are not in use
- Variables without a specified type (implicit variants)
- Too complex procedures
- Minimizable forms without icon
- Forms with ControlBox but without icon (needed in Win95)
- EXE file without title and icon (this happens if you haven't made an EXE file yet)

It is recommended that you take the Problem report only after the analysis has been completed.

Project Analyzer 4.0 - Helpfile generated by VB HelpWriter.

Project metrics

(version 3.1 onwards, <u>registered version</u> only)

To monitor your programming performance, programmers often use some simple metric, like lines of code or EXE file size. However, this is not always enough. Project Analyzer helps you to monitor the understandability, complexity, and reusability of your code.

You can see various metrics in many different ways
- **View|Module metrics** and **View|Procedure metrics** windows are good to find long and complex procedures and modules
- <u>Project summary report</u> is good to see the overall performance
- Design quality analysis report is also good to see the overall performance
- <u>File details window</u> shows metrics by file
- <u>Procedure details window</u> shows metrics by procedure

Below you can read a brief discussion on understandability, complexity and reliability, and the various metrics used to estimate them.

Understandability

Bad understandability will most probably result in more errors and maintaining problems. You can estimate the understandability of your program with the following metrics:

Lines of code / procedure (View|Metrics, Design quality report)
Procedures longer than 50 lines are often too long.

Comment to code ratio and **whitespace to code ratio** (Design quality report)
The more comments in your code, the easier it is to read - and understand. Whitespace is also important for legibility.

Length of identifiers (Design quality report)
The longer your variable, procedure etc. names are, the more likely they are to be descriptive.

Cyclomatic complexity (View|Metrics)
Target for less than 10. See below.

Depth of conditional nesting (see View|Metrics)
Target for less than 5. See below.

Complexity

There are many kinds of program complexity: structural, informational, and logical, for example. High complexity may result in bad understandability and more errors. Complex procedures also need more time to develop and more testing.

The best measures for estimating the complexity of a procedure are:
- Lines of code
- Cyclomatic complexity
- Informational complexity
- Structural fan-out
See below for more information.

Reusability

Reusability is a magic word in programming. Measures for reusability are
- Reuse ratios reported by <u>Super Project Analyzer</u> (reuse in a group of projects)

- Structural fan-in. If it's high, the procedure/module is called (reused!) many times.
- Number of class modules

What do the different metrics mean?

Note: You can see most of the metrics only after the whole analysis has been completed.

Lines of code

As simple as it may seem, lines of code is quite a good measure of how complex a program is. Project Analyzer calculates lines of code as follows:

Lines of code = Total lines - Commented lines - Empty lines

Lines of code does not include control declarations in a .frm file.

Cyclomatic complexity
(McCabe)

Cyclomatic complexity is a measure of the structural complexity of a procedure. The higher the number, the more complex the procedure, and the harder it is to maintain it. Cyclomatic complexity for VB procedures is calculated as follows:

Cyclomatic complexity = Number of Branches + 1

Branches are caused by IF, SELECT CASE, DO...LOOP and WHILE...WEND statements.

"Normal" values for cyclomatic complexity range from 1 (very simple) to 9 (moderately complex). If C is more than 10, you may want to split the procedure.

Cyclomatic complexity is the minimum number of test cases you must have to execute every statement in your procedure. This is important in testing. Carefully test procedures with the highest cyclomatic complexity values.

An average cyclomatic complexity, as well as the distribution of complexity, is reported in the Project Summary Report.

Note: Cyclomatic complexity often gives quite high values for procedures with long SELECT CASE statements.

Nested conditionals

Nested conditionals metric is related to cyclomatic complexity. Whereas cyclomatic complexity deals with the absolute number of branches, nested conditionals is only interested in how deeply nested these branches are.

If you have a procedure with deeply nested conditionals, you should consider splitting it up. Those procedures can be very hard to understand and error-prone.

Nested loops

Nested loops is a very rough estimate of the mathematical complexity of a procedure. The more nested loops there are in a procedure, the more likely it is that those loops take up a significant amout of time to execute.

Structural fan-in/fan-out
(Constantine & Yourdon)

For procedures:

Structural fan-in = the number of procedures that use this procedure

Structural fan-out = the number of procedures this procedure calls

For modules:

Structural fan-in = the number of modules that use variables, constants, or procedures in this module

Structural fan-out = the number of modules whose variables, constants, or procedures this module needs

A high structural fan-in denotes reusable code.

The higher the structural fan-out, the more dependent that procedure is on other procedures, and more complex too.

Informational fan-in/fan-out and informational complexity
(Henry & Kafura)

Lines of code, cyclomatic complexity, or structural fan-out are not perfect in predicting the "real" complexity of a procedure. For example, a procedure may access a number of global variables and be very complex without having to call many other procedures.

Informational fan-in = Procedures called + parameters referenced + global variables referenced
Informational fan-in estimates the information a procedure reads

Informational fan-out = Procedures that call this procedure + [ByRef] parameters assigned to + global variables assigned to
Informational fan-out estimates the information a procedure returns

Combined, these give a new metric: *informational fan-in x fan-out*. This is reportedly good in predicting the effort needed for implementing a procedure, but not so good in predicting complexity. To predict complexity, we need a new metric: informational complexity. It is calculated as follows:

Informational complexity = lines of code x (informational fan-in x informational fan-out)

Project Analyzer 4.0 - Helpfile generated by VB HelpWriter.

Rich Text Format

(version 2.0 onwards)

Project Analyzer can produce formatted <u>reports</u> and save them in a Rich Text Format (RTF) file. A number of popular word processors read RTF files, including Microsoft Word and Windows 95 WordPad. Project Analyzer also produces RTF for the Help Compiler to turn reports into **.hlp** files.

<u>More information on report formats</u>

MS Word

A special option for Microsoft Word users is the **Create Microsoft Word style RTF** check box on the **Save Report** dialog box. This option may produce ugly reports for other word processors (for example, for WordPad).

If this option is set, Project Analyzer optimizes the RTF reports for Word. It uses styles like Normal, Heading 1, Heading 2, ... You can easily create a Table of Contents for your report with Word.

RTF for Help Compiler

You can also create an RTF file for the Microsoft Help Compiler. This way, you can turn the reports into Help files. Project Analyzer will create the necessary .prj file and launch Help Compiler automatically.

The **Create Microsoft Word style RTF** checkbox has no effect on RTF for Help Compiler.

Project Analyzer 4.0 - Helpfile generated by VB HelpWriter.

Options

On double click show

These options let you specify what happens when you double click the **file list** or the **procedure list** on the main screen.

1) Report produces the standard reports for a file or a procedure
2) Detail window shows file details or procedure details in a separate window
3) Hypertext window opens the hypertext window with the selected procedure or (declarations)

Show reports to Display as

1) Formatted text Default. Reports to Display are formatted.

2) Unformatted text. Plain text reports in a text box. Choose this if you want to select part of the report for Copy & Paste.

More about report formats

Refer to files by

1) File name. Use the file name (without path) to refer to any files in your project.

2) Module name. If a file has a module name, use it to refer to the file. Otherwise, use the file name.

Gray windows instead of colorful ones

By default, Project Analyzer uses bright background colors in several windows. Check this option if you prefer ordinary gray backgrounds (in fact, the color used is vbButtonFace, that is, the Windows default button face color).

Hypertext options

Project Analyzer 4.0 - Helpfile generated by VB HelpWriter.

Features in the registered version

The following features are available in the registered version only. Some of them will work in the shareware version too, provided your project has no more than 10 source files.
- Dead code information
- Dead variable, constant, Type and Enum information
- Project metrics for professional programming
- Call trees
- Problem report
- DLL report
- Name shadowing report
- Needless globals report
- Hypertext window
- Generate constants module command (for VB 3.0)

Registration information

Add-ins

Project Analyzer has a couple of add-ins that need a separate registration. Add-ins need a registered version of Project Analyzer to work.

- Super Project Analyzer is an add-in tool for analyzing groups of applications.
- Project Printer prints source code and generates hypertext source documents.

Project Analyzer 4.0 - Helpfile generated by VB HelpWriter.

Project summary

(version 2.1 onwards)

To get a summary of your project, use the **Report|Project summary** command. This report includes some statistics about your project. It also shows some Visual Basic limitations, like the global symbol table size of your project, and its maximum size, 64 kB.

Not all of the measures are exact, these approximated measures are marked as *(approx)*.

Project Analyzer 4.0 - Helpfile generated by VB HelpWriter.

Generate constants module

(version 2.1 onwards, VB 3.0 only, registered version only)

The use of the CONSTANT.TXT and DATACONS.TXT files in a project tends to take up a large amount of memory, because these files include so many **global const** declarations, and only a few of them are needed in one project. Including a large number of unused constants in a project will fill up the global symbol table, as well as decrease the performance of the program.

If you use Project Analyzer, you can solve the problem easily. When you are developing your application, you can include the CONSTANT.TXT and DATACONS.TXT files in your project; you don't need to cut and paste any declarations to another global module. When your project is about to be ready, just analyze it with Project Analyzer, and select **File|Generate constants module**. This command will generate a new module including only those constant declarations that are needed by your application.

After generating a new module, remove the old one (CONSTANT or DATACONS) from your project and add the new one. You're done.

Note 1: This command applies mainly to VB 3.0 projects. VB 4.0 doesn't require you to declare a vast amount of constants, but has predefined constants instead.
Note 2: This command is available only after analysis phase 2 has completed.

Project Analyzer 4.0 - Helpfile generated by VB HelpWriter.

Library report

(version 2.1.07 onwards, <u>registered version</u> only)

This report lists all DLL, OCX and VBX files used by the project.

All declared DLL procedures are listed, and if they are not used, they are marked as "dead".

VBX and OCX files included in your project are listed at the end of the report, as well as any libraries that you have checked in the References dialog box (VB 4.0 and later). If you project file (.vbp) includes information on the version of the library, that is shown too.

Project Analyzer 4.0 - Helpfile generated by VB HelpWriter.

Super Project Analyzer

(version 3.1 onwards, needs registered Project Analyzer and a separate registration)

Super Project Analyzer is an add-in tool for Project Analyzer. It is registered separately. It also needs the registered Project Analyzer to work.

A Super Project is a group of ordinary Visual Basic projects that share some files. Super Project Analyzer analyzes these project groups and reports which procedures, variables and constants etc. are shared or dead.

Instructions

To use Super Project Analyzer, save a couple of normal .vbp/.mak analyses using the **Tools|Save data for Super Project Analyzer** menu command. This will write a .sud (Super Project Data) file for each VB project. This file includes information on the modules, procedures, variables and constants used by that project.

After saving a few (at least 2) .sud files, start Super Project Analyzer (**Tools|Super Project Analyzer**). Then open the .sud files with the **Project|Add project** command in Super Project Analyzer. You can now see the modules, procedures, variables and constants used by all of these projects.

Note: In the unregistered Project Analyzer, Super Project Analyzer can only handle projects with less than 10 source files. If you have registered Project Analyzer, Super Project Analyzer can handle up to 25 source files per project without separate registration. The registered Super Project Analyzer, of course, has no such limitations.

Features

Select **Shared only** to limit the display to those elements that are included in at least 2 projects. **Shared Only** may also show dead elements that are included in at least 2 project but are not used by them.

Select **Dead only** to see which elements are not used by any project. You may consider removing them.

Note: Super Project Analyzer does not list VBX, OCX, or FRX files. It does list DLLs and DLL procedures.

Reports

To get a report on what you see on the screen, press the **Report** button, or select the **Report|Report what you see** menu command**.**

Select the **Report|Reuse report** menu command to get a report on reuse ratios. The report includes reuse ratios for modules, procedures and variables & constants by project and for the whole super project as well.

> **Reuse ratio for modules = Shared modules / All modules in project**
>
> **Reuse ratio for procedures = Shared procedures / All procedures used in project**
> (thus excluding dead ones)
>
> **Reuse ratio for vars&consts = Shared vars&consts / All vars&consts references in project**
> (thus excluding ones that are completely dead or that are only assigned to)

Saving a Super Project definition

To save a Super Project definition to a .sup file, use the **File|Save Super Project** menu command. You can later open this file to skip opening each project separately.

Note: In addition to the .sup file, you will need to have the saved .sud files in order for the analysis to work. The .sup file only includes the file names of those .sud files.

Registration

To use Super Project Analyzer, you will need to register both Project Analyzer and Super Project Analyzer.

Prices
More about registration

Project Analyzer 4.0 - Helpfile generated by VB HelpWriter.

Project Printer

(version 3.1 onwards, needs registered Project Analyzer and a separate registration)

Project Printer is an add-in tool for Project Analyzer. It is registered separately. It also needs the registered Project Analyzer to work.

Project Printer is designed to document VB source code. It can
- Print your source code
- Document references
- Generate hyperlinked source documents in a HTML or HLP file

Project Printer is not just an ordinary code printer. It
- highlights variables, constants and procedure calls
- overstrikes dead variables and constants
- can list calls to and from a procedure
- can report various metrics along with the code
- can generate Table of Contents
- even generates hyperlinked documents (like in the Hypertext window)

You can start Project Printer by selecting the **Tools|Project Printer** menu command.

Note 1: Project Printer gives best results after the analysis has been fully completed.

Note 2: The unregistered Project Printer can only handle projects with up to 10 source files. The registered Project Printer has no such limitations.

Output types

Project Printer is versatile. In step 2/3, it asks you to select the output type. Do this in the main window in the ordinary way, as would do when printing any other report.

Paper. Ordinary formatted source code printout with Table of Contents. Optionally with metrics and 'called by' list.

Help file. The easy way to browse a project. Project Printer generates a help file that is very much similar to the Hypertext window of Project Analyzer. Calls are hyperlinked. The Help file is always there for you to take a quick look at your project, even on computers without Project Analyzer.

HTML file for Intranet documentation. Calls are hyperlinked. Useful for a group of programmers. Even store your code bank in HTML.

RTF file. Format the report in any way you want with your favourite word processor.

Other types. You can get the report in any other report type Project Analyzer supports, for example, on the screen, but the results are not as usable as in the above types.

Options

Project Printer has a number of options in step 3/3. Default values are set based upon which report type has been selected when you continue from step 2/3.

Source style. **Simple** is a fast, plain text report. **Enhanced** parses through the code and highlights identifiers, overstrikes dead ones. In HTML and HLP, it also sets hyperlinks for references.

Show contents. On paper, generates a Table of Contents with page numbers at the end of the report. In a Help file, it's on a separate page. In other report types, it is at the start of the report.

List global/module-level vars and consts. Check this if you want a list of vars/consts declared in (declarations). The list includes information on where a variable or constant has been referenced or assigned to, or if it's dead.

Show incoming calls. Lists the procedures that call this procedure.

Show outgoing calls. Lists the procedures that this procedure calls.

Show metrics. Shows various programming metrics, like complexity etc.

Registration

To use Project Printer, you will need to register both Project Analyzer and Project Printer.

Prices
More about registration

Project Analyzer 4.0 - Helpfile generated by VB HelpWriter.

Pricing

All prices below are in U.S. dollars.
You can also register in your own currency, but please email **vbshop@aivosto.com** for the price.
The prices are subject to change in future versions.

Project Analyzer

Licence	Single user	Additional users	Site licence
Project Analyzer 4.x	$90	$45 / user	$495
Upgrade from Project Analyzer 3.x to 4.x	$30	$15 / user	$165

The price includes all Project Analyzer versions with major version number 4. You can expect a major version number change after the next major Visual Basic version has come out. Minor version upgrades can be expected several times a year. There will be a special low upgrade price for registered users when new major versions come out.

An upgrade licence is valid if you have registered a earlier version of Project Analyzer that the upgrade is for. You can upgrade to Project Analyzer 4.x from version 3.x.

A *single user licence* is valid for one user only. If more than one user uses Project Analyzer, you will need to register each one separately (*multiuser licence*). If lots of programmers in your organization need Project Analyzer, you can consider a site licence. A site licence covers a single organization for an area of up to one hundred miles (160 km) in radius. A site licence price is equal to the price of a multiuser licence for 10 users.

Add-ins	Single user	Additional users	Site licence
Super Project Analyzer	$30	$15 / user	$165
Project Printer	$40	$20 / user	$220

The above add-ins require a separate registration. In addition, they need the registered Project Analyzer to work. An add-in licence is valid for all future Project Analyzer versions (the possible versions 5, 6, etc).

Source code

Project Analyzer 4.x source code for VB 5.0 Pro	$150
Source upgrade from 3.x to 4.x	$50

The source code is also available. This price includes all updates with the major version number 4. You don't have to purchase multiuser or site licences for the source, but you need to have registered Project Analyzer. The source has been developed in Visual Basic 5.0 Professional.

How do I pay?

The possible payment methods are described in REGISTRA.HLP. All major credit cards, U.S. checks, cash, and some other methods are accepted.

Note: There are no deliverables. You will not get the program physically in a box. You will get it via email and WWW/FTP, if you have access to the Internet. In case you don't have an email address, you will receive a letter with the keywords.

More about registration
Features in the registered version

Project Analyzer 4.0 - Helpfile generated by VB HelpWriter.