Dependency Walker Help Contents

Overview of Dependency Walker
- [Why use Dependency Walker?](#)
- [Using Dependency Walker for Trouble Shooting Modules](#)
- [Using Dependency Walker for General Information about Modules](#)

Understanding the Module Session
- [The Module Session Window](#)
- [The Module Dependency Tree View](#)
- [The Module List View](#)
- [The Parent Import Function List View](#)
- [The Export Function List View](#)

Menus and Toolbar
- [The File Menu](#)
- [The Edit Menu](#)
- [The View Menu](#)
- [The Window Menu](#)
- [The Help Menu](#)
- [The Toolbar](#)

Why use Dependency Walker?

Have you ever...

- ...ran LINK /DUMP or DUMPBIN to determine the imports or exports of a module?
- ...wondered what minimum set of files are required to run a particular application or load a particular DLL?
- ...wondered why a certain module was being loaded with a particular application?
- ...wanted to remove all dependencies for a given module?
- ...wanted to know the complete path of all the modules being loaded for a particular application?
- ...wanted to know all the base addresses of each module being loaded for a particular application?   What about versions?   Or maybe machine types?
- ...received one of the following errors...
- The dynamic link library BAR.DLL could not be found in the specified path...
- The procedure entry point FOO could not be located in the dynamic link library BAR.DLL.
- The application or DLL BAR.DLL is not a valid Windows image.
- The application failed to initialize properly.
- The image file BAR.EXE is valid, but is for a machine type other than the current machine.
- Program too big to fit in memory.

Using Dependency Walker for Trouble Shooting Modules

Dependency Walker recursively scans all dependent modules required by a particular application.   During this scan it performs the following tasks:

- Detects missing files.   These are files that are required as a dependency to another module.   A symptom of this problem is the "The dynamic link library BAR.DLL could not be found in the specified path..." error.
- Detects invalid Files.   This includes files that are not Win32 compliant and files that are corrupt.   A symptom of this problem is the "The application or DLL BAR.EXE is not a valid Windows image" error.
- Detects import/export mismatches.   Verifies that all functions imported to a module are actually exported from the dependent modules.   All unresolved import functions are flagged with an error.   A symptom of this problem is the "The procedure entry point FOO could not be located in the dynamic link library BAR.DLL" error.
- Detects circular dependency errors.   This is a very rare error, but can occur with forwarded functions.
- Detects mismatched machine types of modules.   This occurs if a module built for one type of machine tries to load a module built for a different type of machine.

Using Dependency Walker for General Information about Modules

Dependency Walker is more than just a trouble shooting utility.   It also provides a great deal of valuable information about the module layout of a particular application and details on each module.   Dependency Walker provides the following information:

- A complete module dependency tree diagram of all the modules required by a particular application.
- A comprehensive list of all functions exported for each module.   These lists include functions exported by name, functions exported by ordinal, and functions that are actually forwarded to other modules.
- A list of functions that are actually called in each module by other modules.   These lists can help developers understand why a particular module is being linked with an application, and also provides information on how to remove unneeded modules from being dependencies.
- A complete list of the minimum set of files that are required in order for the application to run or DLL to load.   This list can be very useful when copying files to another machine or creating setup scripts.
- For each individual module found, the following information is provided...
- Full path to the module file.
- Time and Date of the module file.
- Size of the module file.
- Attributes of the module file.
- Type of machine that the module was built to run on.
- Type of subsystem that the module was built to run in.
- Whether or not the module contains debugging information.
- The preferred base load address of the module.
- The file version found in the module's version resource.
- The product version found in the module's version resource.
- The image version found in the module's file header.
- The version of the linker that was used to create the module file.
- The version of the OS that the module file was built to run on.
- The version of the subsystem that the module file was built to run in.
- A possible error message if any error occurred while processing the file.

File Menu Commands

The File menu offers the following commands:

Open...	Opens and processes a module file.
Close	Closes the active Module Session Window.
File 1, 2, 3, ...	Opens and processes the specified module file.
Exit	Exits Dependency Walker.

Edit Menu Commands

The Edit menu offers the following commands:

<u>Copy</u>	Copies the current selection to the clipboard as text.
<u>Select All</u>	Selects all items in the current view.

View Menu Commands

The View menu offers the following commands:

Toolbar	Shows or hides the toolbar.
Status Bar	Shows or hides the status bar.
View Full Path	Shows or hides full path strings in the Module Dependency Tree View and the Module List View.
Expand All	Expands all nodes in the Module Dependency Tree View.
Refresh	Updates all views for the active Module Session Window.
External Viewer	Launches the external module viewer for the selected modules.
Configure External Viewer...	Configures the external module viewer.
Properties	Display's the properties dialog for the selected modules.

Window Menu Commands

The Window menu offers the following commands:

Cascade	Arranges windows in an overlapped fashion.
Tile Horizontally	Arranges windows in non-overlapped horizontal tiles.
Tile Vertically	Arranges windows in non-overlapped vertical tiles.
Arrange Icons	Arranges the icons of all minimized windows.
Window 1, 2, 3, ...	Activates the specified window.

Help Menu Commands

The Help menu offers the following commands, which provide you assistance with this application:

Help Topics   Displays the table of contents for the online help documentation.

About Dependency Walker...   Displays program
                        information, version, and copyright.

Open Command (File Menu)

Use this command to open and process a module.   Dependency Walker uses a multiple document interface that allows more than one Module Session Window to be opened and visible at once.   Use the Window Menu to switch between the multiple open Module Session Window.   See Window 1, 2, 3, ... for more information.

Dependency Walker registers itself with the Shell's context menus for all known module extensions and for unknown modules that contain a PE signature.   You can right-click on a particular module anywhere in the Shell or in a Explorer window, and choose View Dependencies to launch Dependency Walker on that module.

Shortcuts

Keys: CTRL+O

Shell: Drag and drop modules on top of Dependency Walker to open them.

Shell   Right-click on a module file in the Shell and choose View Dependencies from the Shell's context menu.

Toolbar:

Close Command (File Menu)

Use this command to close the active Module Session Window.

Shortcuts

Keys: CTRL+F4.

Mouse: Single-click on the Close button in the Title Bar of the window you wish to close.

Mouse:Double-click on the System Menu icon in the Title Bar of the window you wish to close.

1, 2, 3, ... Command (File Menu)

Dependency Walker stores the eight most recently opened modules at the bottom of the File menu for your convenience.   To open one of the modules listed, select the module from the menu or type the number that corresponds with the module you want to open.

Exit Command (File Menu)

Use this command to close all Module Session Windows and exit Dependency Walker.

Shortcuts

Keys: ALT+F4

Mouse: Single-click on the main window's Close button in the Title Bar.

Mouse Double-click on the main window's System Menu icon in the Title Bar.

Copy Command (Edit Menu)

Use this command to copy the current selection to the clipboard as text.   This command is unavailable if there is nothing selected that can be copied.   Copying data to the clipboard replaces any contents previously stored on the clipboard.

For the Module Dependency Tree View and the Module List View, the module name is copied. If the View Full Path option is enabled, then complete path strings will be copied, otherwise just the module file names are copied.

For the Parent Import Function List View and the Export Function List View, the function names are copied.

Shortcuts

Keys: CTRL+C

Keys: CTRL+INSERT

Toolbar:

Select All Command (Edit Menu)

Use this command to select all the items in a particular view.   This command works only in the Module List View, the Parent Import Function List View, and the Export Function List View.   Select All is often useful before performing a Copy if the entire contents of a view wish to be copied.

Shortcuts

Keys: CTRL+A

Toolbar Command (<u>View Menu</u>)

Use this command to display and hide the Toolbar, which includes buttons for some of the most common commands in Dependency Walker, such as File Open.   A check mark appears next to the menu item when the Toolbar is displayed.

See <u>Toolbar</u> for more help on using the toolbar.

Status Bar Command (View Menu)

Use this command to display and hide the Status Bar, which describes the action to be executed by the selected menu item or depressed toolbar button.   A check mark appears next to the menu item when the Status Bar is displayed.

See Status Bar for more help on using the status bar.

View Full Path Command (<u>View Menu</u>)

Use this command to toggle the View Full Paths option on or off.   When this option is on, a check mark appears next to View Full Paths menu item and the View Full Paths toolbar button is displayed as depressed.

When the View Full Paths option is on, both the <u>Module Dependency Tree View</u> and the <u>Module List View</u> will display the complete path to each module.   When this option is off, these views will display only the file names.

This option also effects the how the <u>Copy</u> command works.   <u>Copy</u> will copy full path strings when this option is on, and copy just file names when this option is off.

Shortcuts

Keys: F9

<u>Toolbar</u>:

Expand All Command (View Menu)

This command will expand all the module nodes in the Module Dependency Tree View, making the entire tree visible.

Refresh Command (<u>View Menu</u>)

This command will force the active <u>Module Session Window</u> to clear all of its views and reprocess the original module.   This can be useful during trouble shooting a module to determine is some action you performed, such as locating and copying a missing module, has alleviated the problem.

Shortcuts

Keys: F5

<u>Toolbar</u>:   c:\

External Viewer Command (<u>View Menu</u>)

The external viewer command is provided as a means to launch a secondary module viewer. The external viewer application is completely user configurable. See the <u>Configure External Viewer...</u> command for more information.

If the active view is the <u>Module Dependency Tree View</u>, the <u>Parent Import Function List View</u>, or <u>Export Function List View</u>, the External Viewer command will launch the external viewer application with the module that is currently selected in the <u>Module Dependency Tree View</u>. If the <u>Module List View</u> has the focus, then Dependency Walker will launch a separate instance of the external viewer application for every module that is selected in the list.

Shortcuts

Keys: ENTER (while one or more modules are highlighted in the active view)

Mouse: Double-click on a module.

<u>Toolbar</u>:

Configure External Viewer Command (View Menu)

This command will display a dialog to configure the external viewer application and arguments.   See the
Configure External Viewer Dialog topic for more information.

Properties Command (<u>View Menu</u>)

The properties command is provided as a means to launch the Shell's Properties dialog for selected modules.

If the active view is the <u>Module Dependency Tree View</u>, the <u>Parent Import Function List View</u>, or <u>Export Function List View</u>, then the Shell's Properties dialog will be displayed for the module that is currently selected in the <u>Module Dependency Tree View</u>.   If the <u>Module List View</u> has the focus, then Dependency Walker will display a separate Properties dialog for every module that is selected in the list.

Shortcuts

Keys: F10

<u>Toolbar</u>:

Cascade Command (Window Menu)

Use this command to arrange multiple opened Module Session Windows in an overlapped fashion.

Shortcuts

Toolbar:

Tile Horizontal Command (Window Menu)

Use this command to arrange multiple opened Module Session Windows as non-overlapping horizontal tiles.

Shortcuts

Toolbar:

Tile Vertical Command (Window Menu)

Use this command to arrange multiple opened Module Session Windows as non-overlapping vertical tiles.

Shortcuts

Toolbar:

Arrange Icons Command (Window Menu)

Use this command to arrange the icons for minimized windows at the bottom of the Dependency Walker's main window.

1, 2, 3, ... Command (Window Menu)

Dependency Walker displays a list of currently open Module Session Windows at the bottom of the Window menu.   A check mark appears in front of the Module Session Window name of the active Module Session Window.   Choose a module session from this list to make its window active.

Help Topics Command (Help Menu)

Use this command to display the opening screen of Help.   From the opening screen, you can jump to any area of Dependency Walker's online help documentation.

Once you open Help, you can click the Contents button whenever you want to return to the opening screen.

About Dependency Walker Command (<u>Help Menu</u>)

Use this command to display program information, the version, and the copyright of your copy of Dependency Walker.

Context Help Command

Use the Context Help command to obtain help on a particular area of Dependency Walker. When you choose the Toolbar's Context Help button, the mouse pointer will change to an arrow and question mark.   Then click somewhere in the Dependency Walker window, such as another Toolbar button, menu item, or a view.   The Help topic will be shown for the item you clicked on.

Shortcuts

Keys: SHIFT+F1

Toolbar:

Restore Command (System Menu)

Use this command to return the active window to its size and position before you it was Maximized or Minimized.

Shortcuts

Mouse   Single-click the Restore button on the <u>Title Bar</u> of the maximized window or minimize icon bar.

Move Command (System Menu)

Use this command to display the four-headed arrow cursor which allows you to move the active window or dialog box with the arrow keys.

Note: This command is unavailable if you maximize the window.

Shortcuts

Keys: CTRL+F7

Mouse: Grab the Title Bar of the window and drag the window it to a new location.

Size Command (System Menu)

Use this command to display the four-headed arrow cursor which allows you to re-size the window with the arrow keys.

After the pointer changes to the four-headed arrow:

1. Press one of the DIRECTION keys (left, right, up, or down arrow key) to move the pointer to the border you want to move.   The cursor will change to one of the following images:
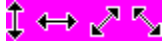
2. Press a DIRECTION key to move the border.

3. Press ENTER when the window is the size you want.

Note: This command is unavailable if you maximize or minimize the window.

Shortcuts

Mouse: Drag the size bars at the corners or edges of the window.

Minimize Command (System Menu)

Use this command to reduce the a window to an icon.

Shortcuts

Keys: ALT+F9

Mouse: Single-click the Minimize button on the <u>Title Bar</u>.

Maximize Command (System Menu)

Use this command to enlarge the active window to fill the available space.

Shortcuts

Keys: CTRL+F10.

Mouse: Single-click the Maximize button on the <u>Title Bar</u>.

Mouse:Double-click on the <u>Title Bar</u>.

Close Command (System Menu)

Use this command to close the window.

Shortcuts

Keys: CTRL+F4 to close the active <u>Module Session Window</u>.

Keys: ALT+F4 to close all <u>Module Session Windows</u> and Dependency Walker.

Mouse: Single-click on the Close button in the <u>Title Bar</u> of the window you wish to close.

Mouse:Double-click on the System Menu icon in the <u>Title Bar</u> of the window you wish to close.

Next Window Command (System Menu)

Use this command to switch to the next open Module Session Window.   Dependency Walker determines which window is next according to the order in which you opened the Module Session Windows.

See the Previous Window command also.

Shortcuts

Keys: CTRL+F6

Previous Window Command (System Menu)

Use this command to switch to the previous open Module Session Window.   Dependency Walker determines which window is previous according to the order in which you opened the Module Session Windows.

See the Next Window command also.

Shortcuts

Keys: SHIFT+CTRL+F6

Next Pane Command

This command allows you to use the keyboard to switch between the different views in a Module Session Window.   The Next Pane Command navigates forward through the views in the following order:

1. Module Dependency Tree View
2. Parent Import Function List View
3. Export Function List View
4. Module List View

See the Previous Pane command for navigating through the views in opposite order.

Shortcuts

Keys: F6

Previous Pane Command

This command allows you to use the keyboard to switch between the different views in a Module Session Window. The Previous Pane Command navigates backwards through the views in the following order:

1. Module List View
2. Export Function List View
3. Parent Import Function List View
4. Module Dependency Tree View

See the Next Pane command for navigating through the views in opposite order.

Shortcuts

Keys: SHIFT+F6

File Open Dialog

The following options allow you to specify which module file to open:

Look in

Lists the available folders and files.   To see how the current folder fits in the hierarchy on your computer, click the down arrow.   To see what's inside a folder, click it.

Up one Level

First button to the right of the **Look in** drop-down list box.   This button will move you up one folder in the hierarchy on your computer specified in the **Look in** field

Create New Folder

Second button to the right of the **Look in** drop-down list box.   This button will create a new folder in the current folder listed in the **Look in** field

List

Third button to the right of the **Look in** drop-down list box.   This button will cause all files and folders in the **File and Folder List** to be displayed in a multi-column list .

Details

Forth button to the right of the **Look in** drop-down list box.   This button will cause all files and folders in the **File and Folder List** to be displayed in a single column list along with size, type, date, time, and attribute details for each file.

File and Folder List

This list displays all the files and folders located in the folder specified by the **Look in** field that match the search specifications of the **File name** field and/or the **Files of type** field.   You may select any file in this list and press **Ok** to open the file.   You may also double-click any file in this list to open the file.

File name

This box allows you to type a full path to a file, a relative path to a file, a path to another folder to browse, a file name to open, or a partial filename with wildcards (* and ?) to search for.   Depending on what you choose to do, the **Look in** field and the **File and Folder List** will update to reflect the change.   If you type an exact match to a particular file, then that file will be opened.

Files of type

Select the types of files you want to open from the drop-down list.   The **File and Folder List** will update to show only the types of files specified by the **Files of type** field. Dependency Walker considers the following to be common file extensions for 32 bit Windows modules:

*.EXE, *.DLL, *.SYS, *.DRV, *.OCX, *.CPL, *.SCR, and *.COM

If you need to open a module file of a different type, select the "All Files (*.*)" entry in the **Files of type** field or type your own search specification in **File name** field.

Configure External Viewer Dialog

Command

This field specifies a path to the executable to be ran when the <u>External Viewer</u> command is invoked.

Arguments

This field specifies the command line arguments to be passed to the executable specified in the **Command** field when the <u>External Viewer</u> command is invoked.   You may use a **%1** anywhere in the argument string to represent the full path to the module file.   When the external viewer application is launched, all **%1** tokens will be replaced with the full path to the module file.   You should surround all **%1** arguments in quotes so that the external viewer can handle long filenames with spaces.   For example, "**%1**".

Browse

This button will display a <u>File Open Dialog</u>, which allows you to browse your file system for the executable file to be used as your external viewer.   If a file is chosen in this dialog, the **Command** field will be updated to show the new file.

About Dependency Walker Dialog

This dialog displays program information, the version, and the copyright of your copy of Dependency Walker.

Toolbar

The toolbar is displayed by default across the top of the application window, below the menu bar.   The toolbar provides quick mouse access to many tools used in Dependency Walker.

There are three ways you can learn what a particular toolbar button's action is.   You can float the mouse over the button and a tool tip will pop up with the command name.   You can press and hold the mouse down over a button and read the text displayed in the Status Bar for a more detailed description.   If you do not wish to execute the command, move the mouse off the toolbar button and release the mouse.   Last, you can use the Context Help utility to activate the online help documentation for the toolbar button.

The toolbar can be docked to the top, left, right, and bottom of Dependency Walker's main window, as well as free floated anywhere above Dependency Walker in its own mini window To change the docking location of the toolbar, simply grab the toolbar along its edge and drag it to where you would like it to go.

To hide or display the Toolbar, choose the Toolbar option from the View menu.

Click   To

◙   Opens and processes a module file.   See the Open… command for more information.
◙   Copies the current selection to the clipboard as text.   See the Copy command for more information.
◙   Shows or hides full path strings in the Module Dependency Tree View and the Module List View.   See the View Full Path option for more information.
◙   Launches the external module viewer for the selected modules.   See the External Viewer command for more information.
◙   Display's the properties dialog for the selected modules.   See the Properties command for more information.
◙   Arranges windows in an overlapped fashion.   See the Cascade command for more information.
◙   Arranges windows as non-overlapping horizontal tiles.   See the Tile Horizontally

command for more information.

Arranges windows as non-overlapping vertical tiles.   See the <u>Tile Vertically</u> command for more information.

Enters context help mode.   See the <u>Context Help</u> command for more information.

Status Bar

For Help, press F1

The status bar is displayed at the bottom of Dependency Walker's main window.   To display or hide the status bar, use the Status Bar option from the View menu.

The status bar describes actions of menu items as you use the arrow keys or mouse to navigate through menus.   This area similarly shows messages that describe the actions of Toolbar buttons as you depress them and before releasing them.   If after viewing the description of the toolbar button command you wish not to execute the command, then move the mouse pointer off the toolbar button and release the mouse button.

Module Dependency Tree View

The Module Dependency Tree View displays a hierarchical view of all the modules' dependencies.   There are three ways a module can be a dependent of another module:

1. Implicit Module Dependency: Module A is implicitly linked with a .LIB file for Module B at compile/link time, and Module A actually calls one or more functions in Module B. Module B is a load time dependency of Module A. and will be listed in Module A's import table.

2. Dynamic Module Dependency: Module A is not linked with Module B.   At runtime, Module A dynamically loads Module B via a LoadLibrary() type technique.   Module B becomes a run time dependency of Module A, but will not be listed in Module A's import table.

3. Forward Module Dependency: Module A is implicitly linked with a .LIB file for Module B at compile/link time, and Module A actually calls one or more functions in Module B. One of the functions called in Module B is actually a forwarded function to Module C. Module B and Module C are both load time dependencies of Module A, but only Module B will be listed in Module A's import table.

Dependency Walker fully handles cases 1 and 3.   Case 2 is out of the scope of Dependency Walker since there is no way to detect a dynamic module dependency without actually loading the application as a running process and monitoring it.   Even this approach is not guaranteed to find all dynamic module dependencies since certain modules may only be dynamically loaded when the application enters a particular state and requires the functionality of that particular module.

Dependency Walker starts with the root module you chose to open and scans its import table to build a list of required dependent modules.   Dependency Walker then scans each of these dependent modules for their dependent modules.   This recursion continues until all modules and their dependent modules have been processed.

To prevent a bloated tree and possible infinite circular loops with dependent modules, Dependency Walker stops processing a given branch of the tree when it reaches a module that it has already processed somewhere else in the tree.   Duplicate modules are marked with a small arrow in the middle of their accompanying image (see below).   To determine what the branch would have looked like if Dependency Walker had processed it, simply look at the branch under the original instance of that module that was processed elsewhere in the tree.

Dependency Walker also scans each dependent module looking for forwarded function calls to other modules.   If a forwarded function is found and actually called by the parent module, then the module that the function is forwarded to is also pulled in and added to the dependency tree.   These forwarded modules are specially mark in the dependency tree with a small state image next to their accompanying image (see below).

While processing the dependency tree, Dependency Walker performs several validity checks along the way.   It checks to make sure each module is a valid Windows 32 bit module.   It checks for mismatched binaries, such as an Intel x86 module with a DEC Alpha module.   It scans import and export function tables looking for unresolved external functions.   It checks for circular dependencies, which are allowed, and for circular forwarded dependencies, which are not allowed.   Any errors that are encountered while processing the tree will be displayed using a special image (see below) for the particular modules in error and/or by a message box.

Modules are initially shown with just the file name of the module.   You can display the full path to each module by using the <u>View Full Path</u> option.   You may also copy the selected module's file name or path to the clipboard by selecting the <u>Copy</u> command.   The actual text copied will differ depending on how the <u>View Full Path</u> option is set.

The following is a table of the possible images that can accompany each module in the dependency tree:

Normal module with no errors.

Forwarded module with no errors.

Duplicate module with no errors.

Forwarded duplicate module with no errors.

Module with one or more missing export functions that are required by the parent module.   The <u>Parent Import Function List View</u> will list the actual unresolved functions that are causing the problem.

Forwarded module with one or more missing export functions that are required by the parent module.   The <u>Parent Import Function List View</u> will list the actual unresolved functions that are causing the problem.

Duplicate module with one or more missing export functions that are required by the parent module.   The <u>Parent Import Function List View</u> will list the actual unresolved functions that are causing the problem.

Forwarded duplicate module with one or more missing export functions that are required by the parent module.   The <u>Parent Import Function List View</u> will list the actual unresolved functions that are causing the problem.

Missing module.   This module could not be found in the local directory or search path.

Missing forwarded module.   This module could not be found in the local directory or search path.

Invalid module.   See the <u>Module List View</u> for an error message describing the module error.

Invalid forwarded module.   See the <u>Module List View</u> for an error message describing the module error.

Module List View

The Module List View displays a list of all unique modules that are required dependencies for the module you opened.   This list defines the minimum set of files needed for the module to execute or load as a running process.

Modules are initially shown with just the file name of the module.   You can display the full path to each module by using the <u>View Full Path</u> option.   You may also copy the selected modules' file names or paths to the clipboard by selecting the <u>Copy</u> command.   The actual text copied will differ depending on how the <u>View Full Path</u> option is set.   If more than one module is selected, a list will be copied to the clipboard with carriage returns after each module.

Along side each module are several columns that contain useful information about each module.   The columns include the following:

- Full path or file name for the module file.   See the <u>View Full Path</u> option for toggling between the two modes.
- Time and Date of the module file.
- Size of the module file.
- Attributes of the module file.
- Type of machine that the module was built to run on.   Possible values are Intel x86, MIPS R3000, MIPS R4000, MIPS R10000, DEC Alpha AXP, PowerPC, and Hitachi SH-3.
- Type of subsystem that the module was built to run in.   Possible values are Native, Win32 GUI, Win32 console, OS/2 console, Posix console, and 8 subsystem.
- Whether or not the module contains debugging information.
- The preferred base load address of the module.
- The file version found in the module's version resource.
- The product version found in the module's version resource.
- The image version found in the module's file header.
- The version of the linker that was used to create the module file.
- The version of the OS that the module file was built to run on.
- The version of the subsystem that the module file was built to run in.

The modules can be sorted on the data in any column in the list.   Simply click on the column header button for the column you wish to sort by.   A arrow (^) is displayed in the column header for the column that the list is currently sorted by.   You can also size a column to its "best fit" width by double-clicking the divider line between two columns in the column header.

If a module was not found or was not a valid Windows 32 bit binary, then an error message will be displayed in place of the normal column information for that module.

Parent Import Function List View

The Parent Import Function List View displays the list of parent import functions for the currently selected module in the Module Dependency Tree View.   Parent import functions are functions that are actually called in the given module by the parent module.

The selected module needs to export every function that the parent is importing from it.   If the selected module does not export one of the functions that the parent module expects to call, then an unresolved external error would occur if the module was attempted to be loaded.   See the Export Function List View for viewing the selected module's export functions.

Dependency Walker searches the exported function list for every parent import function to ensure there is a match.   If any function is unresolved, then the function is marked with an error image (see below) and the module is mark with an error image as well in the Module Dependency TreeView.

The Parent Import Function List View can also help you locate unnecessary modules in an application.   The fact that the parent module is calling functions in the selected module is what makes the selected module a dependency of the parent.   If you can safely stop the parent module from calling all the functions listed in the parent import function list for a given module, then that module will no longer be a dependent of the parent module.

The following are the possible images that can accompany each function in the parent import list:

        Resolved parent import.
        Unresolved parent import.

The Parent Import Function View is comprised of four columns:

Ordinal	The ordinal value of the imported function, if the function is imported by ordinal. This value can be "N/A" if the function is imported by name.
Hint	The hint value for the imported function. The hint value is used internally by the operating system's loader to quickly match imports with exports. It is used as an index into the array of exported functions in the selected module.
Function	The name of the imported function, if the function is imported by name. This can be "N/A" if the function is imported by ordinal.
Entry Point	The entry point memory address for the function. This is usually "Not Bound", meaning that the entry point address will not be known until load time. If an address is given, then the parent module has bound by the BIND program. BIND is a program that runs through a module's import table and stores the most probable entry point address for each function. It does this for each function by opening the import module, looking up the function, and added that function's offset address with that module's preferred base load address. This results in a faster load time if a bounded module's dependent modules actually load at their preferred base load addresses.

The functions can be sorted on the data in any column in the list.   Simply click on the column header button for the column you wish to sort by.   A arrow (^) is displayed in the column header for the column that the list is currently sorted by.   You can also size a column

to its "best fit" width by double-clicking the divider line between two columns in the column header.

Export Function List View

The Export Function List View displays the list of export functions for the currently selected module in the <u>Module Dependency Tree View</u>.   Export functions are functions that a module exposes to other modules.   They can be thought of as the module's interface.

Dependency Walker uses the exported list to check for unresolved external errors in the selected module.   For more information, read the topic on the <u>Parent Import Function List View</u>.

While Dependency Walker scans the export list for a module, it checks each function to see if it is really a forwarded function.   A forwarded function is a function that appears to be exported from a particular module, but in fact the code for the function actually lives in another module.   The operating system's loader recognizes this and loads the forwarded module if necessary to resolve any imports from the parent module.   Dependency Walker, like the operating system's loader, also loads the forwarded module if necessary.

The following are the possible images that can accompany each function in the export list:

   Export function that resides in the selected module.
   Forwarded export function that resides in a different module.

The Export Function View is comprised of four columns:

Ordinal
: The ordinal value of the exported function, if the function is exported by ordinal.   This value can be "N/A" if the function is exported by name.

Hint
: The hint value for the exported function.   The hint value is used internally by the operating system's loader to quickly match imports with exports. It is used as an index into the array of exported functions in the selected module.

Function
: The name of the exported function, if the function is exported by name. This can be "N/A" if the function is exported by ordinal.

Entry Point
: The entry point memory address for the function.   This is usually a relative offset from the base address at which the module will load at by the operating system's loader.   This base address is usually the base address listed in the <u>Module List View</u> for the particular module.   If the function is forwarded to another module, then a forward string will be displayed instead of an address.   The forward string is in the form of ModuleName.FunctionName.

The functions can be sorted on the data in any column in the list.   Simply click on the column header button for the column you wish to sort by.   A arrow (^) is displayed in the column header for the column that the list is currently sorted by.   You can also size a column to its "best fit" width by double-clicking the divider line between two columns in the column header.

Title Bar

The title bar is located along the top of a window.   For Dependency Walker's main window (shown above), it contains the name of the application and the active-module session name if a module has been loaded.   For a <u>Module Session Window</u>, it will contain the name of the session module.

To move a window, drag the title bar.   To resize a window, drag the size bars at the corners or edges of the window.

Dependency Walker's main window's title bar contains the following elements:

- System Menu button.   This is actually displayed as a small Dependency Walker icon on left size of the Title Bar
- Name of the application, "Dependency Walker"
- Name of the active module session; for example, "notepad.exe"
- Minimize button
- Restore/Maximize button
- Close button

Scroll Bars

Scroll bars are displayed at the right and bottom edges of each view.   The scroll boxes inside the scroll bars indicate your vertical and horizontal location in the document.   You can use the mouse to scroll to other parts of the view.

Module Session Window

A module session window is created for every module that is opened and processed.   The window is split into the following four views:

- Module Dependency Tree View
- Module List View
- Parent Import Function List View
- Export Function List View

All views support right-click context menus to commonly used commands for that view.   All views support context help.   You may press F1 anywhere in Dependency Walker to get help on the item that currently has the focus.   You may also use the Context Help tool to allow you to simply click on the item you wish to get help on.

For navigating through the views, see the Previous Pane command and the Next Pane command.   For navigating through the open Module Session Windows, see the Previous Window command, the Next Window command, and the Window 1, 2, 3, ... command.

No Help Available

Sorry, there is no help available for this area or topic.