NAME

wine − run Windows programs on Unix

SYNOPSIS

wine *program* [*arguments*]
wine --help
wine --version

For instructions on passing arguments to Windows programs, please see the **PROGRAM/ARGUMENTS** section of the man page.

DESCRIPTION

wine loads and runs the given program, which can be a DOS, Windows 3.x, Win32 or Win64 executable (on 64-bit systems).

For debugging wine, use **winedbg** instead.

For running CUI executables (Windows console programs), use **wineconsole** instead of **wine**. This will display the output in a separate window. Not using **wineconsole** for CUI programs will only provide very limited console support, and your program might not function properly.

When invoked with **--help** or **--version** as the only argument, **wine** will simply print a small help message or its version respectively and exit.

PROGRAM/ARGUMENTS

The program name may be specified in DOS format (*C:\\WINDOWS\\SOL.EXE*) or in Unix format (*/ms-dos/windows/sol.exe*). You may pass arguments to the program being executed by adding them to the end of the command line invoking **wine** (such as: *wine notepad C:\\TEMP\\README.TXT*). Note that you need to '\' escape special characters (and spaces) when invoking Wine via a shell, e.g.

wine C:\\Program\ Files\\MyPrg\\test.exe

It can also be one of the Windows executables shipped with Wine, in which case specifying the full path is not mandatory, e.g. *wine explorer* or *wine notepad*.

ENVIRONMENT

wine makes the environment variables of the shell from which it is started accessible to the Windows/DOS processes started. So use the appropriate syntax for your shell to enter environment variables you need.

WINEPREFIX

If set, the contents of this variable is taken as the name of the directory where Wine stores its data (the default is *$HOME/.wine*). This directory is also used to identify the socket which is used to communicate with the **wineserver**. All**wine** processes using the same **wineser ver** (i.e.: same user) share certain things like registry, shared memory, and config file. By setting **WINEPREFIX** to different values for different **wine** processes, it is possible to run a number of truly independent **wine** processes.

WINESERVER

Specifies the path and name of the **wineserver** binary. If not set, Wine will try to load */usr/lib/wine/wineserver*, and if this doesn't exist it will then look for a file named "wineserver" in the path and in a few other likely locations.

WINELOADER

Specifies the path and name of the **wine** binary to use to launch new Windows processes. If not set, Wine will try to load */usr/lib/wine/wine*, and if this doesn't exist it will then look for a file named "wine" in the path and in a few other likely locations.

WINEDEBUG

Turns debugging messages on or off. The syntax of the variable is of the form [*class*][+|-]*channel*[,[*class2*][+|-]*channel2*]

class is optional and can be one of the following: **err**, **warn**, **fixme**, or **trace**. If*class* is not specified, all debugging messages for the specified channel are turned on. Each channel will print

messages about a particular component of Wine. The following character can be either **+** or **-** to switch the specified channel on or off respectively. If there is no*class* part before it, a leading **+** can be omitted. Note that spaces are not allowed anywhere in the string.

Examples:

WINEDEBUG=warn+all
: will turn on all warning messages (recommended for debugging).

WINEDEBUG=warn+dll,+heap
: will turn on DLL warning messages and all heap messages.

WINEDEBUG=fixme-all,warn+cursor,+relay
: will turn off all FIXME messages, turn on cursor warning messages, and turn on all relay messages (API calls).

WINEDEBUG=relay
: will turn on all relay messages. For more control on including or excluding functions and dlls from the relay trace, look into the **HKEY_CURRENT_USER\Software\Wine\Debug** registry key.

For more information on debugging messages, see the *Running Wine* chapter of the Wine User Guide.

WINEDLLPATH
: Specifies the path(s) in which to search for builtin dlls and Winelib applications. This is a list of directories separated by ":". In addition to any directory specified in **WINEDLLPATH**, Wine will also look in */usr/lib/x86_64-linux-gnu/wine*.

WINEDLLOVERRIDES
: Defines the override type and load order of dlls used in the loading process for any dll. There are currently two types of libraries that can be loaded into a process address space: native windows dlls (*native*) and Wine internal dlls (*builtin*). The type may be abbreviated with the first letter of the type (*n* or *b*). The library may also be disabled (''). Each sequence of orders must be separated by commas.

  Each dll may have its own specific load order. The load order determines which version of the dll is attempted to be loaded into the address space. If the first fails, then the next is tried and so on. Multiple libraries with the same load order can be separated with commas. It is also possible to use specify different loadorders for different libraries by separating the entries by ";".

  The load order for a 16-bit dll is always defined by the load order of the 32-bit dll that contains it (which can be identified by looking at the symbolic link of the 16-bit .dll.so file). For instance if *ole32.dll* is configured as builtin, *storage.dll* will be loaded as builtin too, since the 32-bit *ole32.dll* contains the 16-bit *storage.dll*.

  Examples:

  WINEDLLOVERRIDES="comdlg32,shell32=n,b"
  : Try to load comdlg32 and shell32 as native windows dll first and try the builtin version if the native load fails.

  WINEDLLOVERRIDES="comdlg32,shell32=n;c:\\foo\\bar\\baz=b"
  : Try to load the libraries comdlg32 and shell32 as native windows dlls. Furthermore, if an application request to load *c:\foo\bar\baz.dll* load the builtin library *baz*.

  WINEDLLOVERRIDES="comdlg32=b,n;shell32=b;comctl32=n;oleaut32="
  : Try to load comdlg32 as builtin first and try the native version if the builtin load fails; load shell32 always as builtin and comctl32 always as native; oleaut32 will be disabled.

WINEPATH
: Specifies additional path(s) to be prepended to the default Windows **PATH** environment variable. This is a list of Windows-style directories separated by ";".

For a permanent alternative, edit (create if needed) the **PATH** value under the **HKEY_CURRENT_USER\Environment** registry key.

WINEARCH
Specifies the Windows architecture to support. It can be set either to **win32** (support only 32-bit applications), or to **win64** (support both 64-bit applications and 32-bit ones in WoW64 mode).
The architecture supported by a given Wine prefix is set at prefix creation time and cannot be changed afterwards. When running with an existing prefix, Wine will refuse to start if **WINEARCH** doesn't match the prefix architecture.

DISPLAY
Specifies the X11 display to use.

OSS sound driver configuration variables:

AUDIODEV
Set the device for audio input / output. Default */dev/dsp*.

MIXERDEV
Set the device for mixer controls. Default */dev/mixer*.

MIDIDEV
Set the MIDI (sequencer) device. Default */dev/sequencer*.

FILES

/usr/lib/wine/wine
The Wine program loader.

/usr/lib/wine/wineconsole
The Wine program loader for CUI (console) applications.

/usr/lib/wine/wineserver
The Wine server

/usr/lib/wine/winedbg
The Wine debugger

/usr/lib/x86_64-linux-gnu/wine
Directory containing Wine shared libraries

$WINEPREFIX/dosdevices
Directory containing the DOS device mappings. Each file in that directory is a symlink to the Unix device file implementing a given device. For instance, if COM1 is mapped to */dev/ttyS0* you'd have a symlink of the form *$WINEPREFIX/dosdevices/com1 -> /dev/ttyS0*.
DOS drives are also specified with symlinks; for instance if drive D: corresponds to the CDROM mounted at */mnt/cdrom*, you'd have a symlink *$WINEPREFIX/dosdevices/d: -> /mnt/cdrom*. The Unix device corresponding to a DOS drive can be specified the same way, except with '::' instead of ':'. So for the previous example, if the CDROM device is mounted from */dev/hdc*, the corresponding symlink would be *$WINEPREFIX/dosdevices/d:: -> /dev/hdc*.

AUTHORS

Wine is available thanks to the work of many developers. For a listing of the authors, please see the file *AUTHORS* in the top-level directory of the source distribution.

COPYRIGHT

BUGS

A status report on many applications is available from the **Wine Application Database** ⟨https://appdb.winehq.org⟩. Please add entries to this list for applications you currently run, if necessary.

Bugs can be reported on the **Wine bug tracker** ⟨https://bugs.winehq.org⟩.

AVAILABILITY

The most recent public version of **wine** is available through WineHQ, the **Wine development headquarters** ⟨https://www.winehq.org/⟩.

SEE ALSO

wineserver(1), **winedbg**(1),
Wine documentation and support ⟨https://www.winehq.org/help⟩.