

Wine FAQ

1. About this FAQ

1.1. Who maintains this FAQ ?

Dave Gardner maintained it from 1995-1998.

Douglas Ridgway (ridgway@winehq.org) took it over in 1999.

Andreas Mohr (amohr@codeweavers.com) converted it to FAQ-O-Matic in 2000.

Dimitrie O. Paun, Keith Matthews and Tom Wickline (in alphabetical order) reorganized it in 2002.

For suggestions/additions/complaints regarding this FAQ, please send an email to wine-faq@winehq.org (<mailto:wine-faq@winehq.org>)

1.2. What is the copyright on the FAQ? And how may I use it?

The original Wine FAQ, which this FAQ was based on, was copyright © 1995-1998 David Gardner.

It may be reproduced and modified under the same terms as Wine itself.

2. General Questions about Wine

2.1. What is Wine and what is it supposed to do?

Wine is a program which allows the operation of DOS and MS Windows programs (Windows 3.x and Win32 executables) on UNIX operating systems such as Linux. It consists of a program loader, which loads and executes a Windows binary, and a set of libraries that implements Windows API calls using their UNIX or X11 equivalents. The libraries may also be used for porting Win32 code into native UNIX executables, often without many changes in the source. Wine is free software, and its license (contained in the file LICENSE in each distribution) is the LGPL.

2.2. Does Wine emulate a full computer?

No, as the name says, Wine Is Not a (CPU) Emulator. Wine just provides the Windows API. This means that you will need an x86-compatible processor to run an x86 Windows application, for instance from Intel or AMD. The advantage is that, unlike solutions that rely on CPU emulation, Wine runs applications at full speed. Sometimes a program run under Wine will be slower than when run on a copy of Microsoft Windows, but this is more due to the fact that Microsoft have heavily optimized parts of their code, whereas mostly Wine is not well optimized (yet). Occasionally, an app may run faster under Wine than on Windows. Most apps run at roughly the same speed.

2.3. Are there any alternatives to Wine?

Yes, there are. You can use VMWare (<http://www.vmware.com>) to run a Windows installation inside a virtual machine, or use Win4Lin (<http://www.win4lin.com>) to run a specially adapted Windows version on Linux. Both solutions cost money for both the software itself and a Windows license.

Note that, like Wine, they can only use the hardware platform that the target programs were originally compiled for (see below).

2.4. What is the difference between Wine and x86 hardware emulators?

There are two free x86 hardware emulators: Bochs (<http://bochs.sourceforge.net>), and Plex86 (<http://savannah.nongnu.org/projects/plex86>).

Plex86 is the open-source free-software alternative for VMWare, VirtualPC, and other IA-32 on IA-32 "Virtual PC products." It can only run on the IA-32 architecture.

Bochs is a highly portable open source IA-32 (x86) PC emulator written in C++, that runs on most popular platforms. It includes emulation of the Intel x86 CPU, common I/O devices, and a custom BIOS. Currently, Bochs can be compiled to emulate a 386, 486 or Pentium CPU. Bochs is capable of running most Operating Systems inside the emulation including Linux, Windows® 95, DOS, and recently Windows® NT 4.

Both are licensed under the GPL. Bochs is older than Plex86, seems to be easier to install, but Plex86 will run faster because Plex86 uses a just in time binary compiler.

The drawback of all emulators is that you need a version of Windows in order to run Windows, and that they all have an impact on performance. Wine also gives much better desktop integration - for instance, programs use your standard window manager, system tray icons will appear in your tray area (if you have one), and you can run programs direct from the command line and the menus. The clipboard also works seamlessly.

2.5. When will Wine integrate an x86 CPU emulator so we can run Windows applications on non-x86 machines?

The short answer is 'probably never'. Remember, Wine Is Not a (CPU) Emulator. The long answer is that we probably don't want or need to integrate one in the traditional sense.

Integrating a CPU emulator in Wine would be extremely hard, due to the large number of Windows APIs and the complex data types they exchange. It is not uncommon for a Windows API to take three or more pointers to structures composed of many fields, including pointers to other complex structures. For each of these we would need a conversion routine to deal with the byte order and alignment issues. Furthermore, Windows also contains many callback mechanisms that constitute as many extra places where we would have to handle these conversion issues. Wine already has to deal with 16 vs. 32 bit APIs and Ansi vs. Unicode APIs which both introduce significant complexity. Adding support for a CPU emulator inside Wine would introduce at least double that complexity and only serve to slow down the development of Wine.

Fortunately another solution exists to run Windows applications on non-x86 platforms: run both Wine and the application inside the CPU emulator. As long as the emulator provides a standard Unix environment, Wine should only need minimal modifications. What performance you lose due to Wine running inside the emulator rather than natively, you gain in complexity inside of Wine. Furthermore, if the emulator is fast enough to run Windows applications, Photoshop for instance, then it should be fast enough to run that same Windows application plus Wine.

Two projects have started along those lines: QEMU (<http://fabrice.bellard.free.fr/qemu/>), an open-source project, and Dynamite (http://www.transitives.com/tech_overview.htm), a commercial CPU emulator environment from Transitives Technologies (<http://www.transitives.com/>) which has been paired with Wine (<http://www.transgaming.com/news.php?newsid=37>).

2.6. Why would anyone want Wine? Doesn't Windows suck?

First Wine is not about running Windows but about running Windows applications.

So if all your computing needs are fulfilled by native Unix applications, then you do not need Wine and should not be using it. However, if you depend on one or more of the tens of thousands of Windows applications, then Wine is the best way to use it without giving up on Unix. Let's look at the alternatives to see why:

The most obvious alternative is to dual-boot. This is the solution that provides the best compatibility. However it requires that you acquire a Windows license and then dedicate a good chunk of your hard-drive to Windows. But the worst is yet to come. Each time you will want to use that application you will have to reboot to Windows. This is especially significant if external factors dictate when you must use this application (e.g. credit card to process, email to retrieve from a Lotus Notes server). Then you will find yourself forced to close

all your Linux applications just to run that one Windows application. You may quickly get tired of this, or will find that such a situation is impossible to justify in a business environment.

The next solution is to install virtual machine emulation software such as VMWare, Win4Lin or Plex86. Then you can use windows applications without suffering such a big disruption. But it still requires that you acquire a Windows license and dedicate as much disk space to Windows. Furthermore you will pay for the added convenience: if using VMWare or Win4Lin you have to buy another license, and more importantly you now have to dedicate a good chunk of your computer's memory to the virtual machine. Performance will take a significant hit too.

Using Wine lets you avoid all of that overhead: Windows license, hard-drive space required by Windows, memory and performance hit taken by emulated virtual machines. Now you can start your Windows application straight from your regular desktop environment, place that application's window side by side with native applications, copy/paste from one to the other, and run it all at full speed.

It is also a pretty vital part of migrating a large organization, you can't change a 5000 desktop setup overnight without a lot of risk.

2.7. Can I use Wine to make the Windows driver for my network card / graphics card / scanner / etc. work on Unix?

The goal of Wine is to make it possible to run Windows applications on Unix, not Windows drivers or VxDs.

Drivers and Windows applications belong to different worlds. Applications run in user mode and use the APIs provided by the kernel and the other user mode dlls. In contrast, drivers are loaded in the Windows kernel, i.e. in ring 0 instead of ring 3, have to deal with specific memory management issues, and use instructions not available to regular applications. This means they would not be able to run in Wine since Wine runs entirely in user mode. Rather you would have to modify the Linux kernel. But in addition, drivers use a completely different API from regular Windows applications. So the work performed on Wine would not even be of any use for such a project. In other words, making it possible to use Windows drivers or VxDs on Unix would be a completely separate project.

However, if you want to reuse Windows drivers on a non-Microsoft operating system we recommend that you have a look at ReactOS (<http://www.reactos.com/>).

2.8. Which one of the different Wine packages out there is good for me?

Currently there is a broad selection of different Wine packages/versions:

Wine (<http://www.winehq.org>)

This is the "standard" source distribution of Wine. Its license is the LGPL, it can be downloaded for free.

ReWind (<http://rewind.sourceforge.net>)

This is a forked Wine tree that got created when Wine changed its license from X11 to the more restrictive LGPL, in order to let people continue to maintain an X11 licensed Wine version. Its license is X11, it can be downloaded for free.

TransGaming's WineX (<http://www.transgaming.com>)

This is TransGaming's Wine version specially suited for games. It includes more mature Direct3D support than WineHQ, although these days WineHQ has quite advanced D3D support as well. Most of the code is under the AFPL and can be downloaded for free.

However TransGaming also distributes binaries that contain improved copy protection support (needed for many games), support, and other enhancements. These packages are only available in binary form to subscribed customers (\$5/month).

CodeWeavers' Wine preview (<http://wine.codeweavers.com>)

This is a special packaged version of the standard Wine tree which has a nice setup for easy installation. License LGPL, free download. It's pretty old now, and not recommended for general use.

CodeWeavers' CrossOver Plugin (<http://www.codeweavers.com/products/crossover/>)

Special Wine installation to be used for running Windows Netscape browser plugins such as e.g. QuickTime in Linux browsers. Costs \$34.95. Well worth it (very stable and useful packaging).

CodeWeavers' CrossOver Office (<http://www.codeweavers.com/products/office/>)

Wine version with special packaging to make sure almost all important Office type programs work pretty well. Costs \$54.95. Seems to be well worth it so far according to some comments. (note: you're supporting a company actively contributing to Wine if you decide to buy either Plugin or Office.)

CodeWeavers' CrossOver Office Server Edition (<http://www.codeweavers.com/products/cxofficeserver/>)

Allows you to run your favorite Windows productivity applications in a distributed thin-client environment under Linux. Server Edition is also a great addition to Solaris environments, since there built-in support for Solaris desktops makes running Windows applications a possibility on Sun workstations as well. For pricing just follow this link: CrossOver Office Server Edition Pricing (<http://www.codeweavers.com/products/pricing.php>)

Other packaged versions of Wine (<http://www.winehq.org/download/>)

Various Wine packages can be downloaded for free from Wine HQ. They are not officially packaged by Wine HQ, and as such may have some configuration inconsistencies.

2.9. What's the history of Wine?

The Wine project started in 1993 as a way to support running Windows 3.1 programs on Linux. Bob Amstadt was the original coordinator, but turned it over fairly early on to Alexandre Julliard, who has run it ever since. A newsgroup (news:comp.emulators.ms-windows.wine) was created in July 1994. Over the years, ports for other Unixes have been added, along with support for Win32 as Win32 applications became popular.

For more information, see <http://www.winehq.org/site/about> (<http://www.winehq.org/site/about>)

2.10. What is the current version of Wine?

A new version of Wine is distributed about every month. You will be able to keep up on all the latest releases by reading the newsgroup comp.emulators.ms-windows.wine (news:comp.emulators.ms-windows.wine), or by visiting the Wine HQ homepage (<http://www.winehq.org>). When downloading Wine from your FTP site of choice (see the Download page (<http://www.winehq.org/download.shtml>) for some of these choices), you can make sure that you are getting the latest version by watching the version numbers in the distribution file name. For instance, the distribution released on August 13, 2003 was called Wine-20030813.tar.gz. Patch files are also available. If you are current to the previous version, you can download and apply just the current patch file rather than the entire new distribution. The patch file names follow the same conventions as the monthly distribution. Read-only CVS (<http://www.winehq.org/site/cvs>) access is also available.

2.11. What is the current Status of Wine?

As of mid 2003, Wine consists of about 1.4 million lines of code, written by more than 550 developers from dozens of countries around the world. Wine is in active use by an estimated 100K people. Wine implements more than 90% of the calls in popular Windows specifications such as ECMA-234 and Open32.

You may also want to look at the Status page (<http://www.winehq.org/site/status>) for a global view on Wine's implementation progress.

2.12. When will Wine be finished?

Large software projects are never finished, only released. In any case Wine is chasing a moving target since every new release of Windows contains new API calls or variations on the existing ones.

Because Wine is being developed by volunteers, it is difficult to predict when it will be ready for general release. But due to the much increased interest by companies in porting apps via Wine, Wine development is constantly getting more and more active. Right now we are working on releasing Wine 0.9 in 2003.

2.13. Who is responsible for Wine?

Wine is available thanks to the work of many people. Please see the AUTHORS (<http://source.winehq.org/source/AUTHORS>) file in the distribution for the complete list. Some companies that are or have been involved with Wine development are CodeWeavers, TransGaming, Corel, and Macadamian.

2.14. Who are the folks and organizations who have contributed money or equipment to the Wine project?

People and organizations who have given generous contributions of money, equipment, or licenses, include:

- David L. Harper
- Bob Hepple
- Mark A. Horton
- Kevin P. Lawton
- The Syntropy Institute
- James Woulfe
- VMWare Inc. (<http://www.vmware.com> (<http://www.vmware.com>))
- Corel (<http://www.corel.com> (<http://www.corel.com>))
- CodeWeavers (<http://www.codeweavers.com> (<http://www.codeweavers.com>))

2.15. What undocumented APIs / interfaces are not understood? Would seeing Microsoft source help?

The best would be if the Windows API was fully documented, so Wine could be a perfect "clean-room" implementation. Seeing the source code might make it harder to prove that no copyright violations have taken place. That said, the documentation is often bad, nonexistent, and even misleading where it exists, so a fair amount of reverse engineering has been necessary, particularly in the shell (Explorer) interface. The biggest problem facing Wine though is simply lack of manpower. At one point, over 5000 people were working on Windows 2000. While Wine doesn't need to replicate all of Windows (we only cover the parts needed to make Windows programs work), that's still nearly 10 times more people working simply on one release than have *ever* worked on Wine, in the history of the project.

2.16. Is TransGaming's latest patch included in the standard Wine release?

No, it's not.

TransGaming makes money via a subscription service and the license of their WineX tree is incompatible with the Wine license. Thus WineX patches cannot be integrated into the Wine tree without express permission by TransGaming. They have submitted some of their work for integration into Wine, most notably DirectDraw and some DirectSound work, and such work has been integrated into the Wine tree. However it seems unlikely they will submit their Direct3D work.

2.17. Will there be a Windows version of Wine?

Some people are working on getting Wine code to compile on Windows using one of the following projects as a basis:

- Cygwin (<http://www.cygwin.com> (<http://www.cygwin.com/>))
- MinGW (<http://www.mingw.org> (<http://www.mingw.org/>))
- ReactOS (<http://www.reactos.com> (<http://www.reactos.com/>))

There's some progress, so a Wine version that's usable on Windows might be available at some time.

Part of the rationale for these projects is to find out areas where Wine portability is lacking. This is especially true of the ReactOS project which is a reimplementation of the Windows kernel and should thus be able to reuse most of Wine dlls.

Another reason for pursuing these projects is to be able to replace a single Windows dll with its Wine counterpart. Besides being a good test for the Wine dll, this lets us detect cases where we made incorrect assumptions about how the dlls interact.

3. What do I need in order to use Wine?

3.1. Under what hardware platform(s) and operating system(s) will Wine(Lib) run?

Wine is being developed specifically to run on the *Intel x86* class of CPUs under certain UNIXes that run on this platform. Winelib however is capable of porting the Windows applications *source code* to other platforms also, not only x86.

Thus running Windows binaries on other platforms (e.g. Mac OS X on PowerPC) using just Wine is *not* possible. You would have to either run Wine in an emulated x86 environment or take the Windows application source code and recompile it using Winelib.

These are the platforms supported by Wine. Winelib support for other platforms keeps evolving, so it's not specifically listed here.

NetBSD, OpenBSD, UnixWare, and SCO OpenServer 5 worked at one time, but Wine now requires kernel-level threads which are not currently available (or understood by the Wine team) on those platforms.

The Wine development team hopes to attract the interest of other commercial UNIX and UNIX clone vendors as well.

BeOS: porting efforts (BeWine) used to be pretty strong, but BeOS has severe limitations in Unix call support. The demise of Be further hampered the project though it might come back one day on one of the open BeOS projects. In any case a functional port seems unlikely to ever happen at this stage.

Mac OS X / Darwin: The Darwine (<http://darwine.sourceforge.net/project.html>) is currently working on porting Wine to the Darwin/x86 platform. Their goal is to eventually make it possible to run x86 Windows applications on Darwin/PPC and then Mac OS X by using Bochs.

FreeBSD: This port is well maintained and should work with limitations in specific areas (mainly missing device/hardware support).

Linux/x86: Works, and as the most popular platform for both developers and users, it is the best supported platform of all.

3.2. What minimum CPU must I have in my computer to be able to run Wine and MS Windows applications smoothly?

We need to differentiate between Wine and Winelib here.

Wine won't run on any x86 CPU less than an 80386 due to address management limitations.

It is known to also work in the 80486 and upwards compatible CPUs. The basic test is, if you can run X11 now, you should be able to run Wine and MS Windows applications under it.

As always, the faster your CPU, the better. Having a math coprocessor is unimportant. However, having a graphics accelerated video card supported by X will help greatly.

Depending on your application you may find that faster speeds are required for sensible use. We can't give specific advice on that due to the vast range of applications out there. However the rule of thumb is that if your application runs fine on Windows, it should run fine on the same platform in Wine.

3.3. How much disk space will the Wine source code and binaries take on my hard drive?

You need approximately 250 megabytes of free hard drive space to store and compile the source code. Wine also needs about 18 megs in your /tmp directory. And about 50 MB are needed to do a make install.

Binary packages, especially those not containing debug information, have much lower disk space requirements, usually in the 20MB range.

3.4. What other software do I need to install, compile and run Wine?

Many development tools are needed in order to compile Wine. A list of required packages for several distributions is included in the README (<http://source.winehq.org/source/README> (<http://source.winehq.org/source/README>)).

To run Wine, you will need the following:

- The compiled Wine binary
- A properly configured wine.conf file (or ~/.winerc file)
- An installed and working X Window system
- Some Windows programs to test

3.5. How much RAM do I need to have on my UNIX system to be able to run Wine and MS Windows applications smoothly?

If you can run X smoothly on your UNIX system now, you should be able to run Wine and MS Windows applications just fine too, depending on how memory hungry the application is.

A Wine workstation will work with 16 megabytes of RAM and a 16 megabyte swap partition as long as you have a reasonable graphics card. Most applications will run reasonably with 64/64 Mb, interactive games are likely to need more. You can run Wine with 8/8, but it is going to be unusably slow and very constraining on the applications you can run. If you wish to be part of the development team and program Wine itself, be aware that the debugger is rather memory intensive. Some have suggested that 64 megabytes is the minimum RAM needed for Wine development, although some are able to work (albeit slowly) with 24 megabytes of physical RAM and lots of swap space.

3.6. How long does Wine take to build

Wine is getting to be quite large, and building from scratch takes a lot of processing. As of September 2002, compile times were around 20 minutes on an Athlon 1200 with 640 Mb and 45-50 minutes on a Cyrix 300 with 64 Mb. If you have a CVS copy, you may not need to rebuild the whole thing every time you update.

3.7. I have a Drivespaced, Doublespaced or Stackered DOS partition. Can Wine run MS Windows binaries located in such a partition?

Yes, but only if the operating system supports mounting those types of drives. There is a Linux file system driver called dmsdos that will allow read/write access to Doublespaced and Drivespace 1.0 drives. More specifically, it supports mounting DOS 6.0 and 6.2 Doublespaced, DOS 6.22 Drivespaced, and Windows 95 Doublespaced compressed partitions (read and write access works fine, but write access is slow). It can be found at <ftp://metalab.unc.edu/pub/Linux/system/filesystems/dosfs/> (<ftp://metalab.unc.edu/pub/Linux/system/filesystems/dosfs/>)

3.8. Do I need to have a DOS partition on my system to use Wine?

You do not need a licensed and installed copy of DOS or MS Windows to install, configure and run Wine. However, Wine has to be able to 'see' an MS Windows binary (i.e. application) if it is to run it.

3.9. Does MS Windows need to be loaded into that partition in order to run MS Windows programs under Wine?

Many folks have successfully installed and run programs in their UNIX file system without having a DOS partition or MS Windows. However, in many cases you need a directory and file infrastructure that is similar to an existing Windows installation. Some applications' installation programs want to distribute some of the package's files into the /windows and /windows/system directories in order to run, and unless these exist on your UNIX file system, those programs will not install correctly and probably will not run well, if at all. Most packages will set that up for you as part of the install process.

If you have a DOS partition with MS Windows installed in it, make sure that your UNIX system can 'see' this partition (check your /etc/fstab file or mount the partition manually) so that Wine can run the MS Windows binaries located in the DOS partition. To run without a DOS partition, you need to set a UNIX path to be your drive C, and make sure that the /windows and /windows/system directories point to some place that actually exist.

Here's an example, copied from a machine which has no DOS partition but successfully runs Wine:

```
[Drive C]
    Path=/var/lib/wine
    Type=hd
    Label=MS-DOS
    Filesystem=win95
[wine]
    Windows=c:\windows
    System=c:\windows\system
    Temp=e:\
    Path=c:\windows;c:\windows\system;c:
```

In /var/lib/wine/windows, you will need to install a win.ini config file that you might find on a typical MS Windows 3.1 machine. The directory /var/lib/wine/windows/system should exist, but doesn't need to contain anything. However, to use MS DLLs, you can copy them into that directory. Note that this is a contravention of the Windows licence unless Windows is properly installed on the machine. If you have DOS/MS Windows installed on your system, you can mount that partition at bootup by modifying the file /etc/fstab in your UNIX partition (assuming that the UNIX kernel supports the DOS/MS Windows file system type).

If you edit this file by hand, it should contain something similar to the following:

```
/dev/hda1 /dosd msdos uid=0,gid=100,umask=007 0 0
```

This will allow you to read and write to the DOS partition without being root.

3.10. If Wine completely replaces MS Windows, will it duplicate all of the functions of MS Windows?

Wine's goal is to make it possible to run Windows applications on Unix. To this end it will provide replacements for just those DLLs and APIs that are needed by these Windows applications. This means that Wine will not provide replacements for DLLs that are not shipped with Windows or are always shipped with Windows application (e.g. the Visual Basic run time). This also means that implementing an API that no application ever uses is not a priority. Similarly, until there are applications out there that use the Win64 API, it will not be a priority. That being said, we will certainly try to keep our options open and to improve our API coverage as we can.

Also Wine is not an operating system, so that writing device drivers is not part of Wine's goals. However if you are interested in device drivers, the Linux (<http://www.kernel.org/>), FreeBSD (<http://www.freebsd.org/>) and ReactOS (<http://www.reactos.com/>) kernel developers would certainly appreciate your contribution.

Similarly Wine does not try to be a desktop environment so providing applets such as a calculator, a file manager or even window manager that look like Windows, are low priority or would even best be done as a separate project. Such projects would also to a large extent be redundant with other open-source projects. Again, there are projects that would certainly appreciate your contributions in this areas, such as the Gnome (<http://www.gnome.org/>) or KDE (<http://www.kde.org/>) desktop environments. You will get the added benefit that your contribution will then be usable by everyone, not just by Wine users.

3.11. Will I be able to install MS Windows applications in any flavor of a UNIX file system?

Wine is written to be file system independent, so MS Windows applications will install and run under virtually any file system supported by your brand of UNIX.

3.12. Will Wine run only under X, or can it run in character mode?

Most of Wine's development effort is geared towards MS Windows' GUI, but some limited support for character mode has appeared, by setting *GraphicsDriver=ttydrv* in wine.conf's *[wine]* section.

Wine's infrastructure is already somewhat prepared for supporting other graphics drivers than x11drv, but no real "alternative" graphics driver has been developed yet.

3.13. Will Wine run under any X window manager? Does it require a window manager at all?

Wine is window manager independent, so the X window manager you choose to run has (almost) no bearing on your ability to run MS Windows programs under Wine. Wine uses standard X libraries, so no additional ones are needed. Wine has its own window management, which acts like MS Windows. It can be turned off to use the native window manager by modifying Managed or Desktop settings as described in **man wine.conf**.

3.14. Will 32-bit Windows 95/98 applications run under Wine?

Yes, 32-bit programs are now about as well supported as 16-bit programs.

4. Getting Wine**4.1.** Where can I get Wine?

Because of lags created by using mirror, word of this newest release may reach you before the release is actually available at the ftp sites listed here. The sources are available from the following locations:

- <http://www.ibiblio.org/pub/Linux/ALPHA/wine/development/>
(<http://www.ibiblio.org/pub/Linux/ALPHA/wine/development/>)
- <ftp://ftp.infomagic.com/pub/mirrors/linux/sunsite/ALPHA/wine/development/>
(<ftp://ftp.infomagic.com/pub/mirrors/linux/sunsite/ALPHA/wine/development/>)

- <ftp://ftp.fu-berlin.de/unix/linux/mirrors/sunsite.unc.edu/ALPHA/wine/development/>
(<ftp://ftp.fu-berlin.de/unix/linux/mirrors/sunsite.unc.edu/ALPHA/wine/development/>)
- <ftp://orcus.progsoc.uts.edu.au/pub/Wine/development/>
(<ftp://orcus.progsoc.uts.edu.au/pub/Wine/development/>)

It should also be available from any other site that mirrors [ibiblio.org](http://www.ibiblio.org), see <http://www.ibiblio.org/pub/Linux/MIRRORS.html>. Some of these sites may archive previous versions of Wine as well as the current one. To determine which is the latest one, look at the distribution file name, which will take the form Wine-YYYYMMDD.tar.gz. Simply replace YYYYMMDD in the distribution file name with the numbers for year, month and date, respectively. The latest one is the one to get.

Wine binary packages are available for several OS'es and distributions. See the download page (<http://www.winehq.org/site/download>) for the most recent list.

4.2. Is there a CVS tree?

Current Wine sources are also available via anonymous client/server CVS. You will need CVS 1.9 or above. If you are coming from behind a firewall, you will either need a hole in the firewall for the CVS port (2401) or use SOCKS.

To login to the CVS tree, do

```
export CVSR00T=:pserver:cv@cv.winehq.org/home/wine
cvs login
```

Use "cvs" as the password (without the quotes). Note that /home/wine is a path on the server, not on your machine. To check out the entire Wine source tree (which may be slow), use

```
cvs -z 3 checkout wine
```

or if you just want a subtree, or individual file, you can do that too with

```
cvs -z 3 checkout wine/ANNOUNCE
```

Be aware, though, that getting the entire Wine source tree via CVS is pretty slow, especially compared to getting Wine from an FTP mirror near you. For a CVS mirror list, see

<http://www.winehq.org/site/cvs#cvsservers> (<http://www.winehq.org/site/cvs#cvsservers>)

Patch files are also available, so that you don't have to download, install, and configure the entire distribution each week if you are current to the previous release. Patch file release names follow the same numbering convention as do the general releases, and take the form

Wine-YYYYMMDD.diff.gz

Patch files are available from the same sites that distribute the full release. To upgrade to a new release by using a patch file, first cd to the top-level directory of the release (the one containing the README file), then do a "make clean", and patch the release with

```
gunzip -c patch-file | patch -p1
```

where patch-file is the name of the patch file something like Wine-YYYYMMDD.diff.gz. You can then re-run ./configure, and then run make depend && make

If you are mirroring the Wine distribution from the tsx-11 site and wish to be listed here in this FAQ, please add it to the "things to go into the documentation" area.

4.3. Can I get Wine using cvsup?

The CVS mirrors don't offer cvsup support yet, but the main server does. Use a wine . sup file of:

```
*default host=cvs.winehq.org
*default base=/cvs
*default prefix=/cvs/wine
*default release=wine
*default delete

# If your network link is a T1 or faster, comment out the following line.
#*default compress

*default use-rel-suffix
wine
```

5. Installing and Configuring Wine

5.1. How do I compile the Wine distribution source code?

See the README (<http://source.winehq.org/source/README>) for instructions. Additionally, you may want to set the `TMPDIR` environment variable `TMPDIR=~/.tmp` or `TMPDIR=/tmp` (if you are root).

5.2. How do I install Windows in Wine under Linux?

Simple answer: you CAN'T. Windows demands direct access to the hardware and cannot get it with Wine and UNIX in the way

Wine is supposed to be primarily used WITHOUT Windows. If you want to use a Windows installation, then use an existing installation alongside the UNIX installation (see the dual-boot HOWTO for your OS for more details). Or alternatively use the cabextract utility to extract Windows install archives to a directory that you want to use as Wine's Windows tree.

5.3. How do I configure Wine to run on my system?

Wine requires that you have a config file as `~/.wine/config`. The format of this file is explained in the `wine.conf` man page. The file `documentation/samples/config` (<http://source.winehq.org/source/documentation/samples/config>) (<http://source.winehq.org/source/documentation/samples/config>) contains a config file example. More explicit directions can be found in the README file (<http://source.winehq.org/source/README>) (<http://source.winehq.org/source/README>) that will be located in the base Wine directory after you gunzip and untar the distribution file.

5.4. How do I upgrade Wine without losing my working configuration?

Upgrading the wine installation does not affect the existing wine configuration. So after upgrading wine you still have the old (working) wine configuration.

5.5. If I want to use a Windows install, which versions are OK?

Either use a classic no-windows install (Wine is getting better all the time) or use a Win9x install (Win95, 98, 98SE, ME). DON'T configure Wine to use an NT-based Windows install (NT, Win2K, WinXP, Win2K3).

In general, most Windows installations contain vast quantities of garbage that can confuse Wine and make it less reliable. If you can, it's best to install the programs you want into Wines fake windows drive.

5.6. If I use a Windows install with Wine, which one works best?

As of 02/2002:

I'd say Win98SE is the best version to use with Wine, as it's fairly widespread amongst developers and relatively old. Using Win2K files is *definitely* worse than a plain no-windows Wine install, and Win ME is said to be problematic, too (as probably no developer uses it). In short: all Win9x <= W98SE are good.

5.7. Installing applications generated by Visual Basic won't run. What should I do?

Make sure you have all the VB run time libraries installed. You may need to use the native DLL vbrun60.dll. You can get some from dll-files.com.

5.8. When I click on *.exe file in my file Manager, nothing happens.

The normal Wine releases don't have .exe extensions registered for Wine in KDE/Gnome yet. You have to open a terminal window instead (often an icon showing a "black screen") and type something like:

```
cd /my/windows/program/directory
wine myprogram.exe
```

Or alternatively you could download the CodeWeavers Wine preview which includes .exe extension registration for KDE/Gnome and a nice setup program.

5.9. bash says "wine: Command not found" What can I do?

Try to re login into bash. That might fix it.

If it doesn't, then make sure the wine binary is in your *PATH*.

Run as root:

```
find / -name "wine" -type f -perm +111
```

to find the path where the wine binary is in. Then check whether *PATH* includes it:

```
echo $PATH
```

If not, add that e.g. to /etc/profile by doing:

```
export PATH=$PATH:/path/to/wine/binary
```

That should help.

If you used a package manager (**rpm** or **apt**) - Verify your packages. The package `winesetuptk.rpm` is only a front-end for making a meaningful config file, it DOES NOT install the wine package...

For complete packages, use <http://rpmseek.com/> (<http://rpmseek.com/rpm-pl/wine.html?hl=com&cx=0::>) or the Download (<http://www.winehq.org/site/download>) section.

5.10. How do I remove Wine from my Computer?

It depends on how you installed. If you used an RPM, the right command is this: **rpm -e wine (as root)**

If you installed from source (the .tar.gz file), the right way to do it is to change to the root of the source tree (the directory with the configure script, readme etc) then run as root: **make uninstall**

6. About running Wine

6.1. How do I run an MS Windows program under Wine?

When invoking Wine, you must specify the entire path to the executable, or by file name only. For example to run Windows' solitaire, type any of the following:

- **wine sol** or **wine sol.exe** (using the search path to locate the file).
- **wine c:\\windows\\sol.exe** (using a DOS file name).
- **wine /usr/windows/sol.exe** (using a UNIX file name).
- **wine "c:\\windows\\sol.exe"** (using quoted DOS file name).

The path of the file will also be added to the path when a full name is supplied on the command line.

6.2. I have installed and configured Wine, but Wine cannot find MS Windows on my drive. Where did I go wrong?

If you have a DOS partition, first make sure that you have mounted it, either by putting the entry into `/etc/fstab`, or by manually mounting it.

Remember too that unless your version of UNIX can see through it, or you are running a utility that can see through it, your DOS partition must not be located on a Drivespaced, Doublespaced or Stacker partition, as neither Linux, FreeBSD, NetBSD or Wine can natively 'see' files located in these compressed DOS partitions.

Check your path statements in the `wine.conf` file. No capital letters may be used in paths, as they are automatically converted to lowercase.

6.3. I was able to get various MS Windows programs to run, but parts of them do not work. What is wrong?

Wine is not complete at this time, so some of each programs' features may not work. They will in time as more of the MS Windows API calls are included in Wine.

6.4. I have run various MS Windows programs, but since the program menus do not work, how can I exit these programs?

Kill the xterm shell window that you called up to run your MS Windows program, and the X window that appeared with the program will be killed too.

6.5. My program doesn't work, what can I do?

If you are a programmer and know C, then start debugging Wine and help us make it better! If you can't, then you will have to either convince a Wine developer to try and make your program work (there must be a downloadable version or demo for that), or hire somebody to do it for you. If this application is an internal corporate application, you may be able to hire a Wine developer to do consulting work for you on the matter.

Alternatively, you may be able to get the app working by taking native DLLs from a Microsoft Windows install, and using them (set the dlls to native in the config file). Not all DLLs can be replaced that way - in particular DirectX cannot be, nor can some core system DLLs like `user`, `ntdll`, `kernel32` etc

6.6. Can I use Wine with SuSE, Peanut or other Linux Distro's?

You can use Wine on any sufficiently recent Linux installation. The amount of work getting Wine up and running depends on whether there are proper packages available or a source compile has to be done.

6.7. Does Wine work with AMD Processors?

Yes, it does. Wine should work on any processor compatible with the Pentium or greater.

6.8. Can I launch a Unix program from a Windows program?

Sure, Wine supports that. Just enter the unix program name wherever a program has something that it's supposed to execute, and it should just work.

6.9. I get "Error installing iKernel.exe: (0x1400)" when running an InstallShield 6 installer.

If you get the error "Error installing iKernel.exe: (0x1400)" at any point, it's probably because there are leftover processes from a previous try. You can verify this with the command

```
$ ps auxxw | grep wine
```

If that command shows old copies of wine running your setup, you need to kill them before you can run the setup program. If there are no other Wine programs running, you can kill them all with the command

```
$ killall wine
```

If you're also running Wine programs you care about, you'll have to kill off the old Setup instances one by one using kill and the individual PIDs (or perhaps Wine's spiffy Task Manager, which doesn't exist yet).

You should repeat the **ps** to make sure the old Wine processes are gone.

7. Getting help**7.1. Is there any documentation for Wine?**

Yes, see <http://www.winehq.org/site/documentation>. (<http://www.winehq.org/site/documentation>)

7.2. I couldn't find the answer to my question in the documentation, but I've written a document explaining how to solve it. What should I do?

Updates and additions to the Wine documentation directory should be sent to the wine-patches mailing list at <http://www.winehq.org/site/forums> (<http://www.winehq.org/site/forums>). Website and FAQ additions should be added to the appropriate Wine Knowledge base directory.

7.3. Is there a Usenet newsgroup for Wine?

Yes, and it's called comp.emulators.ms-windows.wine (news:comp.emulators.ms-windows.wine). The newsgroup serves as a place for users and developers to discuss Wine, and for minor announcements for the general public. Major announcements will be cross posted to other appropriate newsgroups, such as the following:

- comp.os.linux.announce (news:comp.os.linux.announce)
- comp.windows.x.announce (news:comp.windows.x.announce)
- comp.emulators.announce (news:comp.emulators.announce)

If your Usenet site does not carry these newsgroups, please urge your ISP's sysadmin to add and/or uplink them.

7.4. Is there a World Wide Web site for Wine?

Wine HQ (<http://www.winehq.org>) is the official site.

7.5. Is there an IRC channel for Wine?

Sure. It's channel #WineHQ on irc.freenode.net see (<http://freenode.net>). Usually several Wine developers hang out there just to help YOU ;-)

7.6. I think I've found a bug. How do I report this bug to the Wine programming team?

Bug reports should be submitted to our online Bugzilla system (<http://bugs.winehq.org/>). You should include at least the following:

- The Wine version tested
- The Windows application name, including the version, and, if applicable, a URL the application can be downloaded from
- A brief description of the bug
- The relevant part(s) of the output of the Wine debugger
- A screenshot of the visual problem, if applicable

For more information about reporting bugs please see the [How to report a bug](http://www.winehq.org/Docs/wine-user/bug-reporting.shtml) (<http://www.winehq.org/Docs/wine-user/bug-reporting.shtml>) section of the Wine Users Guide.

8. Helping Wine or becoming a Wine developer**8.1. How do I become a Wine developer? What do I need to know?**

If you can program C, that's a good start. Download the sources via (CVS, (<http://www.winehq.org/site/cvs>)) subscribe to the mailing lists, look around the source, and pay attention to the `comp.emulators.ms-windows.wine` newsgroup and the mailing lists (<http://www.winehq.org/site/forums>). See if there's anything that you think you can fix or work on. You won't have much trouble finding areas that need work in Wine (grep for `FIXMEs` in the source).

8.2. How can I help contribute to the Wine project, and in what way(s)?

You can contribute programming or documentation skills, or monetary or equipment donations, to aid the Wine developers in reaching their goals.

For a list of ideas of how you can help, please consult the [Wine contrib page](http://www.winehq.org/site/contributing) (<http://www.winehq.org/site/contributing>).

8.3. I want to help beta test Wine. How can I do this?

Wine still consists of some Alpha code at this time. However, anyone is welcome to download the latest version, and try it out at any time.

8.4. I have written some code that I would like to submit to the Wine project. How do I go about doing this?

Submitting a patch for inclusion in Wine is pretty simple. Basically all you have to do is send the patch to the `wine-patches` mailing list (<http://www.winehq.org/mailman/listinfo/wine-patches>). Still there are a couple of recommendations about the patch format and all so it's best to read our page describing how to submit patches (http://www.winehq.org/site/sending_patches). This will also give you more details about the whole process and in particular to what will happen to your patch once submitted.

9. Developing programs using Wine/WineLib**9.1. Can I use Wine to port my Win32 sources to Unix?**

That is the idea of Winelib. Right now you may still have some difficulties, but this is changing all the time. Read the Winelib User's Guide (<http://www.winehq.org/Docs/winelib-user/>) for info.

9.2. Will MFC work with Wine? What do I need to do?

Wine is not implementing an MFC replacement nor does it intend to. However it is possible (with a lot of work) to compile the MFC from source and thus produce an `mfc42.dll` library.

Please refer to the Winelib User's Guide (<http://www.winehq.org/Docs/winelib-user/>) for how to do this.

9.3. Are there any commercial applications which have been ported using Wine?

Here are few examples of applications ported using Wine or Winelib:

- Corel's WordPerfect Office Suite 2000 was ported to Linux using Wine.
- Kylix, the Linux version of Delphi, was ported to Linux using Winelib. The IDE actually uses a combination of QT and Winelib which would not have been possible to achieve using only Wine. The generated applications however do not depend on Wine in any way.
- MusicMatch Jukebox 5 has also been ported (http://www.itworld.com/nl/lnx_desktop/01042001/) to Linux using Winelib. However more recent versions have not, and version 5 is no longer available.
- Ability Office (<http://www.ability.com/linux/abilitylinux.php>)
- IBM's Websphere (<http://www7b.boulder.ibm.com/dl/swws/wwsgddb-p>)

Many other important applications have already been ported. (we are speaking of several top 500 applications here)

9.4. How can I detect Wine?

You really shouldn't want to do this. If there's a quirk in Wine you need to work around, it's much better to fix it in Wine.

10. Wine HQ issues**10.1. Why are the mailing lists set to reply to author, not to mailing list?**

There are very valid reasons for doing so.

10.2. How to unsubscribe from the mailing lists?

Please see: <http://www.winehq.org/site/forums> And select [(Un-)Subscribe]