Guide Utilisateur Wine

Huw Davies

Steven Elliott

Mike Hearn

James Juran

Ove Kaaven

Alex Korobka

Bruce Milner

Andreas Mohr

Dustin Navea

Eric Pouech

Adam Sacarny

John Sheets

Petr Tomasek

Guide Utilisateur Wine

par Huw Davies, Steven Elliott, Mike Hearn, James Juran, Ove Kaaven, Alex Korobka, Bruce Milner, Andreas Mohr, Dustin Navea, Eric Pouech, Adam Sacarny, John Sheets, et Petr Tomasek

Table des matières

Liste des tableaux

Chapitre 1. Introduction

1.1. Vue d’ensemble / A propos

1.1.1. But de ce document et public visé

Ce document, appelé le Guide de l’Utilisateur de Wine [Wine User Guide], est censé être à la fois un guide d’installation simple et un guide de référence étendu. Ainsi, tout en expliquant complètement comment installer et configurer Wine, il essaie aussi de documenter toutes les particularités de configuration et les domaines de support de l’environnement Wine dans son ensemble.

Il essaie de cibler à la fois l’utilisateur Wine débutant, en offrant une approche pas à pas, et l’utilisateur Wine expérimenté, en offrant le matériel de référence mentionné ci-dessus.

1.1.2. Questions brûlantes et commentaires

Si au cours de la lecture de ce document il y a quelque chose que vous n’arrivez pas à comprendre, qui selon vous pourrait être mieux expliqué, ou qui aurait dû être inclus, veuillez envoyer un courrier immédiatement au wine-devel (mailto:wine-devel@winehq.org), ou postez un rapport de bogue au Bugzilla Wine (http://bugs.winehq.org/) afin de nous faire savoir comment améliorer ce document. Souvenez-vous que le code source libre [open source] est "free as in free speech, not as in free beer" (libre comme la liberté de parole, et non gratuit comme dans "bière à volonté") : cela ne peut fonctionner que si ses utilisateurs sont très activement engagés!

Veuillez noter que je ne peux pas dire avoir été très impressionné par la quantité de réactions que nous avons reçues jusqu’à présent à propos de ce Guide depuis que j’ai ajouté ce paragraphe il y a plusieurs mois...

1.1.3. Vue d’ensemble du contenu / Etapes à suivre

Dans cette section nous essaierons de vous donner une vue d’ensemble complète sur la façon de procéder, du début à la fin, pour obtenir une installation de Wine fonctionnant complètement en suivant ce Guide. Nous *recommandons fortement* de suivre chaque étape de ce Guide à la lettre, car il pourrait sinon vous manquer des informations importantes.

Tout d’abord, nous commençons par expliquer ce qu’est Wine et par mentionner tout autre chose pouvant être utile à connaître (c’est couvert dans ce chapitre dont vous lisez une partie en ce moment même).

Pour être capable d'utiliser Wine, vous devez d'abord obtenir une copie de ses fichiers. C'est le but du prochain chapitre, Se Procurer Wine : on essaie de vous y montrer comment installer Wine sur votre système à vous (c'est-à-dire quelles méthodes d'installation sont disponibles dans votre cas), et d'expliquer ensuite les diverses méthodes : soit se procurer Wine via un fichier paquetage binaire convenant à votre système en particulier, soit le récupérer via un fichier d'archive *code source* Wine, ou encore se procurer le code source en développement le plus récent de Wine via *CVS* .

Une fois que vous avez votre copie de Wine, si vous avez opté pour le code source de Wine vous pouvez avoir besoin de suivre le chapitre suivant Compiler. Sinon, le chapitre suivant Installer Wine vous expliquera les méthodes à utiliser pour installer les fichiers binaires Wine à un emplacement dans votre système.

Une fois que Wine est installé sur votre système, le chapitre suivantConfigurer Wine va se concentrer sur les méthodes de configuration disponibles pour Wine pour installer un environnement Wine/Windows correct avec toutes les caractéristiques requises : il y a des applications d'aide à la configuration disponibles soit graphiques (par exemple WineSetuptTk) soit en mode texte (wineinstall) qui configureront complètement l'environnement Wine pour vous. Et pour ceux qui n'aiment pas une installation complètement automatisée (peut-être parce qu'ils veulent vraiment savoir ce qu'ils sont en train de faire), nous allons décrire comment mettre en place une configuration complète de l'environnement Wine.

Une fois que la configuration de l'environnement Wine est effectuée, le chapitre suivant Exécuter Wine vous montrera comment faire fonctionner des programmes Windows avec Wine et comment satisfaire aux besoins les plus spécifiques de certains programmes Windows.

En cas de problème, le chapitre Dépanner / Signaler des bogues donne une liste de méthodes courantes de dépannage et débuggage et les explique.

1.2. Qu'est-ce que Wine ?

1.2.1. Windows et Linux

Beaucoup de gens ont connu la frustration de posséder un logiciel qui ne fonctionne pas sur leur ordinateur. Avec la toute nouvelle popularité de Linux (http://www.tldp.org/FAQ/Linux-FAQ/index.html), cela se produit de plus en plus souvent à cause de systèmes d'exploitations différents. Votre logiciel Windows ne fonctionnera pas sous Linux, pas plus que votre logiciel Linux ne fonctionnera dans Windows.

Une solution souvent utilisée à ce problème est d'installer les deux systèmes d'exploitation sur le même ordintateur, en tant que système « double amorçage [dual boot] ». Si vous voulez écrire un document dans MS Word, vous pouvez démarrer dans Windows; Si vous voulez exécuter GnuCash, l'application

financière GNOME, vous pouvez fermer votre session Windows et redémarrer dans Linux. Le problème est que vous ne pouvez pas faire les deux en même temps. Chaque fois que vous alternez MS Word et GnuCash, vous devez redémarrer à nouveau. Cela peut vite devenir agaçant.

La vie serait tellement simple si vous pouviez exécuter toutes vos applications sur le même système, sans vous soucier de savoir s'ils sont écrits pour Windows ou pour Linux. Sous Windows, ceci n'est pas vraiment possible pour l'instant. [1] , Wine permet d'exécuter des applications natives Windows à côté d'applications natives Linux sur n'importe quel système de type Unix. Vous pouvez partager l'espace bureau entre MS Word et GnuCash, en faisant chevaucher leurs fenêtres, en les iconisant, et même en les exécutant depuis le même lanceur.

1.2.2. Qu'est-ce que Wine, et comment peut-il m'aider ?

Wine est une implémentation UNIX des bibliothèques win32 Windows, écrites depuis le départ par des centaines de développeurs bénévoles et publiées sous une license Open Source (pensez-y comme une couche de compatibilité Windows pour Linux et les autres systèmes d'exploitation similaires). N'importe qui peut télécharger et lire dans le code source, et réparer les bogues découverts. La communauté Wine est pleine de programmeurs de grand talent qui ont passé des milliers d'heures de leur temps libre à améliorer Wine afin qu'il fonctionne bien avec l' *Interface de Programmation* (API) win32, et se maintienne au même niveau que les nouveaux développements de Microsoft.

Wine peut exécuter des applications Windows de deux manières bien distinctes : comme binaires Windows précompilés (votre paquetage programme ordinaire, par exemple disponible sur CD), ou comme applications compilés nativement X11 (X-Window System) (http://www.xfree86.org/#whatis) (via la partie de Wine appelée Winelib). Si cela vous intéresse de compiler le code source d'un programme Windows que vous avez écrit, veuillez plutôt vous référer au Guide Utilisateur Winelib, qui explique ce sujet en particulier. Ce Guide Utilisateur Wine va par contre se concentrer sur l'exécution d'applications Windows standard en utilisant Wine.

1.2.3. Les capacités de Wine

Maintenant que nous en avons fini avec ce bla-bla ennuyeux qui servait d'introduction, nous allons vous dire ce dont Wine est capable / ce qu'il peut supporter :

- Support pour exécuter des programmes Win32 (Win 95/98, NT/2000/XP), Win16 (Win 3.1) et DOS
- Utilisation optionnelle des *DLL* externes avec lesquelles l'ordinateur a été vendu (c'est-à-dire les DLL originales de Windows)
- Affichage graphique basé sur X11 (affichage à distance possible sur n'importe quel terminal X), console en mode texte

- Fenêtres bureau-dans-une-boîte (Desktop-in-a-box) ou mélangeables
- Support DirectX plutôt avancé pour les jeux
- Bon support pour le son et les périphériques d’entrée supplémentaires
- Impression : pilote d’interface PostScript (psdrv) vers les services d’impression standard PostScript Unix
- Les modems et périphériques série sont supportés
- Mise en réseau Winsock TCP/IP
- Support d’interface ASPI (SCSI) pour les scanners, les graveurs CDs, ...
- Support Unicode, support de langue relativement avancé
- Débogueur Wine et messages des journaux de traçage configurables

1.3. Autres offres de Wine, souvent "Améliorées"

Il y a un nombre d’offres qui sont dérivées d’une manière ou d’une autre du code de base Wine standard .

Certaines sont des produits commerciaux de sociétés qui contribuent activement à Wine.

Ces produits essaient souvent de ressortir du lot ou de se distinguer de Wine, en offrant par exemple une plus grande compatibilité ou une configuration beaucoup plus facile et plus souple que votre version moyenne standard de Wine. En soi, c’est souvent une bonne idée de débourser quelques sous pour les versions commerciales, surtout parce que ces sociétés contribuent beaucoup au code de Wine, et de plus, je suis sûr qu’elles seront contentes de votre soutien...

Tableau 1-1. Diverses offres Wine

Produit	Description	Forme de distribution
ReWind (http://sourceforge.net/projects/rewind)	ReWind est une version de Wine dérivée de l’ancien arbre Wine sous lincence BSD (c’est la branche sous lincence BSD "complètement libre" de l’actuelle licence LGPL de Wine). A cause de sa license BSD il ne peut pas incorporer certains correctifs Wine qui ont été mis sous la licence plus restrictive (ou : protectrice) LGPL par leurs auteurs.	Libre, Open Source : licence BSD

Produit	Description	Forme de distribution
CodeWeavers CrossOver Office (http://www.codeweavers.com/products/office)	CrossOver Office vous permet d'installer vos applications de productivité Windows favorites dans Linux, sans avoir besoin d'une licence Système d'Exploitation Microsoft. CrossOver comprend une interface simple clique facile à utiliser, qui rend simple et rapide l'installation d'une application Windows.	Commerciale
CodeWeavers CrossOver Office Server Edition (http://www.codeweavers.com/products/office/server)	CrossOver Office Server Edition vous permet d'exécuter vos applications de productivité Windows dans un environnement client léger distribué sous Linux, sans avoir besoin d'une licence Système d'Exploitation Microsoft pour chaque machine client. CrossOver Office Server Edition vous permet de satisfaire les besoins litéralement de centaines d'utilisateurs simultanés, tous depuis un serveur unique.	Commerciale
CodeWeavers CrossOver Plugin (http://www.codeweavers.com/products/crossplugin)	CrossOver Plugin vous permet d'utiliser beaucoup de plugins Windows directement depuis votre navigateur Linux. En particulier CrossOver supporte complètement QuickTime, Shockwave Director, Windows Media Player 6.4, Word Viewer, Excel Viewer, PowerPoint Viewer, et plus...	Commerciale; Version démo disponible
CodeWeavers Wine preview (http://www.codeweavers.com/technology/wine)	La version avant-première Wine est légèrement une version de Wine légèrement plus ancienne qu'un test a permis de qualifier d'extra-stable. Elle inclut l'installeur graphique winsetuptk, permettant une configuration facile.	Libre, Open Source : Licence LGPL

Produit	Description	Forme de distribution
TransGaming Technologies WineX (http://www.transgaming.com)	WineX est une version de Wine dérivée de l'ancien arbre Wine sous licence BSD, avec actuellement un meilleur support des logiciels Direct3D et DirectX que le Wine standard, et avec un support de protection contre la copie ajouté pour les multiples protections contre la copie supplémentaire par exemple dans les jeux.	Commerciale ; téléchargement CVS libre (http://sourceforge.net/projects/winex) de la version réduite (pas de support de la protection contre la copie etc.)

1.4. Alternatives à Wine qui pourraient vous intéresser

Nous allons mentionner quelques alternatives (ou nous pourrions aussi dire : concurrents) à Wine qui pourraient être utiles si Wine n'est pas utilisable pour le programme ou travail que vous voulez lui faire faire, car ces alternatives fournissent habituellement une meilleure compatibilité Windows.

1.4.1. VMWare

VMWare (http://www.vmware.com) est un paquetage logiciel pour émuler une machine supplémentaire sur votre PC. En d'autres termes, il établit une machine virtuelle qui peut être utilisée pour exécuter n'importe quel type de système d'exploitation compatible Intel x86 en parallèle de votre système d'opération actuellement en cours. Vous pourriez donc utiliser Linux et Windows 98 en même temps dans une machine virtuelle sur le même écran.

Ca a l'air bien, hein? Eh bien, il y a quelques inconvénients, bien sûr... Tout d'abord, VMWare est plutôt cher, et deuxièmement, vous avez besoin d'une copie sous licence du système d'exploitation que vous voulez exécuter. Troisièmement, comme VMWare est une machine virtuelle, il est plutôt lent. Wine n'a aucune de ces limitations, mais cela veut malheureusement dire que vous n'aurez pas non plus la compatibilité relativement bonne d'un réel système Windows original si vous utilisez Wine.

1.4.2. Win4Lin

Win4Lin (http://www.win4lin.com) de NeTraverse vous permet d'exécuter une version spéciale de Win98 dans Linux. Comparé à VMWare, celui-ci a l'avantage d'être plus rapide, mais vous avez toujours les frais de license.

1.5. Caractéristiques requises de base de Wine

Cette section ne mentionne que les caractéristiques système les plus basiques de Wine, pour faciliter votre "décision d'achat" de Wine ;-) Pour une liste mise à jour beaucoup plus détaillée des caractéristiques pour compiler et/ou installer Wine, veuillez lire la section REQUIREMENTS (CARACTERISTIQUES) du fichier README (http://www.winehq.org/source/README), qui est également disponible dans le répertoire principal de l'arbre de code source Wine.

Dans le cas d'un paquetage binaire Wine, ces caractéristiques de Wine seront probablement satisfaites automatiquement par le processus d'installation ; si vous voulez néanmoins jeter un oeil sur les besoins détaillés (ce qui sans aucun doute ne peut pas faire de mal!), alors je voudrais signaler que l'on peut également souvent trouver le fichier README dans le répertoire des fichiers de documentation d'un paquetage Wine.

1.5.1. Caractéristiques requises

Pour exécuter Wine, vous avez généralement besoin de ce qui suit :

- Un ordinateur ;-)

  Wine : uniquement les PC >= i386 sont supportés pour le moment

  Winelib : d'autres éventuelles plateformes sont supportées, mais cela peut être difficile.

- Un système d'exploitation de type UNIX tel que Linux, *BSD, Solaris x86, ReactOS, Cygwin
- >= 32 Mo de RAM. Tout ce qui est en dessous est plutôt inutilisable. >= 96 Mo sont nécessaires à une "bonne" exécution.
- Un gestionnaire de fenêtre X11 (XFree86 etc.). Wine est prêt pour d'autres pilotes d'affichage graphique, mais le support pour l'écriture n'est pas très aisé. Le pilote d'affichage de console texte (ttydrv) est presque utilisable, donc vous n'avez pas forcément besoin d'installer X11 si vous n'en avez pas besoin pour les programmes que vous avez l'intention d'exécuter (en d'autres termes : principalement pour les programmes en mode texte).

Notes

1. Techniquement, si vous avez deux ordinateurs en réseau, l'un fonctionnant sous Windows et l'autre sous Linux, et que vous avez un quelconque logiciel serveur X fonctionnant sous Windows, vous pouvez exporter des applications Linux vers le système Windows. Un serveur X libre est disponible à http://xfree86.cygwin.com/. Cependant, ceci ne résoud pas le problème si vous ne possédez qu'un seul système informatique.

Chapitre 2. Se Procurer Wine

Si vous avez décidé que vous pouvez utiliser et voulez utiliser Wine (c'est-à-dire après avoir lu le chapitre d'introduction), alors vous devez tout d'abord trouver une bonne version de Wine compatible qui vous plaît et qui fonctionne sur votre système, et après en avoir trouvé une, la prochaine étape est le transfert des fichiers sur votre système d'une manière ou d'une autre. Ce chapitre est là pour vous dire ce à quoi vous devez faire attention pour accomplir ces deux étapes avec succès.

2.1. Comment télécharger Wine ?

Il y a trois méthodes différentes possibles pour amener (télécharger) les fichiers appartenant à Wine sur votre système :

- En récupérant un seul fichier *paquetage* Wine (spécifiquement adapté à votre système en particulier), qui contient divers fichiers *binaires* de Wine
- En récupérant un seul fichier archive compressé (habituellement .tar.gz), qui contient tous les fichiers *code source* d'une version standard de Wine
- En téléchargeant depuis un serveur *CVS* , qui contient les tout derniers fichiers code source de développement de Wine

2.1.1. Quelle type de Wine devrais-je choisir ?

Après vous avoir parlé des différentes méthodes de distribution de Wine disponibles, discutons des avantages et inconvénients des diverses méthodes.

Méthodes de distribution de Wine

Fichier paquetage Wine

Niveau de l'utilisateur visé : Débutant à Avancé

L'utilisation des fichiers paquetage Wine est facile pour trois raisons : Ils installent tout ce qui est nécessaire pour leur opération, ils préconfigurent habituellement beaucoup, et vous n'avez pas besoin de vous inquiéter de compiler quoi que ce soit. Vous pouvez obtenir les paquetages Wine officiels sur la page de téléchargement Wine sourceforge.net (http://sourceforge.net/project/showfiles.php?group_id=6241)

Code source Wine via un fichier archive

Niveau de l'utilisateur visé : Avancé à Expert

Un fichier d'archive code source Wine peut être utilisé si vous voulez compiler votre propre version standard de Wine. En utilisant les fichiers correctifs différentiels pour obtenir des versions de Wine plus récentes, vous pouvez facilement mettre à jour votre répertoire Wine périmé. Cependant, comme vous devez télécharger manuellement les fichiers correctifs et que vous pouvez seulement télécharger la version standard la plus récente de Wine, ceci n'est pas nécessairement la meilleure méthode à utiliser. Le seul avantage d'une archive source Wine est que c'est une version standard Wine avec moins de "bizarreries" de développement que le code CVS du moment. A part ça, on préfère de loin le code source CVS et il est presque aussi facile.

Code source Wine via extraction CVS

Niveau de l'utilisateur visé : Avancé à Expert/Développeur

L'extraction CVS Wine offre le meilleur moyen de s'éclater en profitant de ce qu'il y a de plus avancé au niveau développement et capacités de Wine, car vous pourrez télécharger chaque validation CVS [commit CVS] *au delà* même de la dernière version officielle de Wine. Comme la mise à niveau à la dernière version d'un arbre d'extraction CVS Wine est très simple, c'est une méthode recommandée pour installer Wine. De plus, en suivant méticuleusement les instructions de ce Guide, vous pourrez obtenir la meilleure compatibilité de l'environnement Wine possible (au lieu de devenir la victime des personnes s'occupant de la maintenance des paquetages incapables de suivre certaines instructions dans le Guide Paquetage Wine).

Pour résumer, la "meilleure" façon d'installer Wine est de télécharger le code source Wine via CVS pour obtenir le code le plus récent (qui pourrait s'avérer instable!). Vous pouvez ensuite facilement compiler et installer les fichiers Wine manuellement. La partie finale de la configuration (écrire le fichier de configuration et installer l'environnement pilote) pourra ensuite être gérée par WineSetupTk. Dans l'ensemble la meilleure manière d'agir, mis à part les 500 Mo d'espace disque dont vous aurez besoin.

Avec les fichiers d'archives du code source, vous exécutez des versions standard, ce qui est un avantage, et vous pouvez mettre à jour avec des versions plus récentes via les fichiers correctifs que nous sortons. Vous n'aurez cependant pas le code le plus récent et la souplesse offerte par CVS.

A propos des fichiers paquetages binaires : pas sûr. Il y a à peu près un milliard de bonnes raisons de ne pas les aimer autant que vous pourriez le penser : ils peuvent être périmés, ils peuvent ne pas "tout" inclure, ils ne sont *pas* optimisés pour votre environnement particulier (au contraire d'une compilation de source, qui devinerait et paramètrerait tout ce qui est basé sur votre système), ils sont souvent incapables de produire un environnement Wine complètement configuré. Du côté positif : ils sont plutôt faciles à installer et ils ne prennent pas autant d'espace qu'une compilation totale du code source. Mais c'est tout en ce qui concerne ses avantages. Je dirais donc que c'est OK si vous voulez quelque chose de *rapide* pour avoir une exécution test de Wine, mais il vaut probablement mieux configurer l'environnement vous-même pour une utilisation prolongée de Wine. Au final cela changera (nous ferons probablement des efforts de conditionnement de notre coté un jour), mais étant donné la vitesse fulgurante à laquelle Wine se développe actuellement, la meilleure façon de s'y prendre est de coller le plus possible au niveau de développement de Wine en cours.

Si vous exécutez une distribution de Linux ou un autre système qui utilise des paquetages pour garder des traces des logiciels installés, vous devriez avoir de la chance : Une version préconditionnée de Wine doit déjà exister pour votre système. Les sections suivantes vous diront comment trouver les derniers paquetages Wine et les installer. Restez pourtant prudent quant aux mélanges des paquetages système entre différentes distributions, et même sur des versions différentes de la même distribution. Un paquetage ne marchera souvent que pour la distribution pour laquelle il a été compilé. Nous allons traiter de Linux Debian, Linux Red Hat, Mandrake, SUSE et Slackware, FreeBSD, et d'autres distributions.

Si vous n'avez pas la chance d'avoir un paquetage disponible pour votre système d'exploitation, ou si vous préférez une version plus récente de Wine qu'une qui existe déjà en paquetage, il faudra télécharger le code source Wine et le compiler vous-même sur votre propre machine. Ne vous inquiétez pas, ce n'est pas si difficile à faire, surtout avec les nombreux outils utiles fournis avec Wine. Vous n'avez besoin d'aucune expérience de programmation pour compiler et installer Wine, même s'il serait bien d'avoir quelques petites connaissances sur l'administration UNIX. Le Guide de Développeur Wine traite du travail à partir de la source. Le problème principal avec les fichiers paquetages dont le suivi est assuré en externe est qu'ils n'ont pas de méthode de configuration standard, et sont en fait souvent incapables de configurer l'environnement Windows de Wine comme il faut (ce qui est exposé en gros dans le Guide des Paquetages Wine).

2.2. Se procurer un paquetage Wine

2.2.1. Linux Debian

Dans la plupart des cas sur un système Debian (ou n'importe quelle autre distribution utilisant des paquetages qui utilisent l'extension de fichier .deb, d'ailleurs), vous pouvez télécharger et installer Wine avec une seule commande, en tant que *root* :

```
# apt-get install wine
```

apt-get va se connecter à une archive Debian par Internet (vous devez donc être en ligne), puis télécharger le paquetage Wine et l'installer sur votre système. Fin de l'histoire. Vous pouvez cependant avoir d'abord besoin de mettre à jour correctement votre configuration paquetage en utilisant un *éditeur* en tant que *root* pour ajouter une entrée à `/etc/apt/sources.list` pour pointer sur un serveur de paquetages en activité et ensuite exécuter **apt-get update**.

Une fois cette étape franchie, vous pouvez sauter le chapitre d'installation Wine car apt-get a non seulement téléchargé, mais aussi déjà installé les fichiers Wine. Vous pouvez donc allez maintenant directement à la section Configuration.

Cependant, si vous ne voulez ou ne pouvez pas utiliser la méthode de téléchargement automatique pour les paquetages .deb fournie par la commande **apt-get**, alors lisez la suite.

Bien évidemment, la version préconditionnée Debian de Wine peut ne pas être la version la plus récente. Si vous exécutez la version stable de Debian, vous pouvez peut-être obtenir une version légèrement plus récente en chopant le paquetage de la distribution Debian dite "instable", bien que cela soit peut-être un peu risqué, selon l'importance de la divergence des distributions instable et stable. Vous pouvez trouver une liste des paquetages binaires Wine pour les différentes versions de Debian en utilisant le moteur de recherche de paquetage à www.debian.org (http://www.debian.org).

Si vous avez téléchargé une fichier paquetage .deb séparé (par exemple une version Wine plus récente comme mentionné ci-dessus) qui ne fait pas partie de votre distribution et ne peut donc pas être installé via la **apt-get**, vous devez utiliser **dpkg** à la place. Pour les instructions sur la façon de procéder, veuillez vous rendre à la section Installation.

2.2.2. Linux Red Hat, Mandrake, SUSE, et Slackware

Les utilisateurs de Red Hat, Mandrake, SUSE et Slackware peuvent télécharger le binaire wine depuis la page de téléchargement Wine sourceforge.net (http://sourceforge.net/project/showfiles.php?group_id=6241)

2.2.3. FreeBSD

Pour utiliser Wine vous avez besoin de construire et installer un nouveau noyau avec les options USER_LDT, SYSVSHM, SYSVSEM, et SYSVMSG.

Si vous voulez installer Wine en utilisant le système de portage FreeBSD, lancez dans un *terminal* :

```
$ su -
# cd /usr/ports/emulators/wine/
# make
# make install
# make clean
```

Ce procédé va récupérer la source wine sur Internet, puis télécharger le paquetage Wine et l'installer sur votre système.

Si vous voulez installer Wine à partir du CD-ROM FreeBSD, lancez dans un *terminal* :

```
$ su -
# mount /cdrom
# cd /cdrom/packages/All
# pkg_add wine_.X.X.X.tgz
```

Ces instructions d'installation FreeBSD installent complètement les fichiers Wine sur votre système ; vous pouvez ensuite aller à la section Configuration.

Vous pouvez aussi télécharger un paquetage FreeBSD de wine depuis la page de téléchargement Wine sourceforge.net (http://sourceforge.net/project/showfiles.php?group_id=6241)

2.2.4. Autres systèmes

Si votre système n'est pas spécifiquement mentionné ci-dessus, vous devez regarder d'abord la Page de Téléchargement WineHQ (http://www.winehq.org/download/). Cette page contient une liste de nombreuses archives variées de fichiers binaires (précompilés) Wine.

Vous pourriez aussi essayer d'utiliser Google (http://www.google.com/search?q=wine+package+download) pour retrouver divers paquetages de distribution.

2.3. Se procurer le code source de Wine

Si vous êtes sur le point de compiler Wine (au lieu d'installer des fichiers binaires Wine), soit pour utiliser le code le plus récent possible ou pour l'améliorer, il vous faut de prime abord vous procurer une copie du code source. Nous allons traiter de la façon de récupérer et compiler les distributions des sources depuis les archives officielles, ainsi que la façon de vous procurer le code source Wine dernier cri, encore tout chaud à partir de CVS (Système de Version Simultanées [Concurrent Versions System]).

Une fois que vous avez téléchargé le code source Wine conformément aux instructions ci-dessous, il existe deux manières de procéder : Si vous voulez installer et configurer manuellement Wine, allez à la section Compiler. Si vous voulez au contraire une installation automatique, allez directement à la section Configuration pour utiliser **wineinstall** et installer et configurer Wine automatiquement.

Vous pourriez aussi avoir besoin de savoir comment appliquer un correctif de code source à votre version de Wine. Peut-être avez-vous découvert un bogue dans Wine, et que vous l'avez signalé dans le Bugzilla Wine (http://bugs.winehq.org) ou dans la liste de diffusion Wine (mailto:wine-devel@winehq.org), et reçu un correctif d'un développeur pour, espérons-le, corriger le bogue. Nous allons vous montrer comment appliquer le correctif sans risque et l'annuler s'il ne fonctionne pas.

2.3.1. Se procurer le code source de Wine à partir des archives officielles

La manière la plus sûre de récupérer la source est à partir d'une des archives officielles. Une liste à jour

se trouve dans le fichier ANNONCE (http://www.winehq.org/source/ANNOUNCE) dans la distribution Wine (que vous devriez avoir si vous l'avez déjà téléchargée). Voici une liste des serveurs hébergeant Wine :

- ftp://ftp.ibiblio.org/pub/Linux/ALPHA/wine/development/ (ftp://ftp.ibiblio.org/pub/Linux/ALPHA/wine/development/)
- page de téléchargement sourceforge.net (http://sourceforge.net/project/showfiles.php?group_id=6241&package_id=77449)

Les versions officielles sont étiquetées par date selon le format "Wine-*AAAAMMJJ*.tar.gz". Le mieux à faire est de prendre la plus récente.

Je recommenderais de mettre dans le répertoire dans lequel vous comptez extraire Wine le fichier d'archive Wine choisi. Dans notre exemple, nous supposerons que c'est votre répertoire personnel.

Une fois le fichier d'archive Wine téléchargé, il nous faut extraire le fichier d'archive. Ce n'est pas très compliqué à faire. Placez vous d'abord dans le répertoire qui contient le fichier que vous venez de télécharger. Extrayez ensuite la source dans un *terminal* avec (par exemple) :

```
$ tar xvzf wine-20030115.tar.gz
```

Juste au cas où vous auriez récupéré une archive Wine qui utilise une extension `.tar.bz2` au lieu de `.tar.gz` : dans ce cas-là, vous n'avez juste qu'à utiliser **tar xvjf** à la place.

Maintenant qu'en ayant suivi les étapes ci-dessus vous avez un arbre Wine qui fonctionne complètement , vous voilà fin prêt pour passer aux étapes d'installation et de configuration qui suivent.

2.3.2. Se procurer le code source Wine à partir de CVS

On a voulu que cette partie soit rapide et facile, en montrant le strict mininum de ce qui est nécessaire pour télécharger le code source Wine via CVS. Si vous êtes intéressé par une explication très prolixe de CVS ou de sujets avancés sur CVS (réglages de configuration, serveurs mirroirs CVS, autres modules CVS sur WineHQ, CVSWeb, ...), veuillez lire le chapitre complet CVS dans le Guide de Développeur Wine.

2.3.2.1. vérification de l'installation CVS

Vous devez d'abord vous assurer que **cvs** est installé. Pour vérifier si c'est bien le cas, veuillez lancer dans un *terminal* :

```
$ cvs
```

Si cela a réussi, vous devriez avoir obtenu en sortie une belle aide d'"Utilisation" CVS. Si ce n'est pas le cas (par exemple une erreur "cvs: command not found") il vous reste encore à installer un paquetage CVS pour votre système d'exploitation, de la même manière que les instructions données dans les chapitres pour se procurer et installer un paquetage Wine sur divers systèmes.

2.3.2.2. Télécharger l'arbre Wine CVS

Une fois CVS installé, vous pouvez effectuer une ouverture de session sur notre serveur CVS et extraire (télécharger) le code source Wine. Effectuons d'abord l'ouverture de session, pour se connecter au serveur US :

```
$  export CVSROOT=:pserver:cvs@cvs.winehq.org:/home/wine
$  cvs login
```

Pour se connecter au serveur européen

```
$ export CVSROOT=:pserver:cvs@rhlx01.fht-esslingen.de:/home/wine
$ cvs login
```

Si **cvs** se connecte avec succès au serveur CVS, vous allez obtenir une invite de commande " CVS password:". Entrez simplement "cvs" comme mot de passe (le mot de passe est *sensible à la casse* : pas de lettre capitale!).

Après vous être connecté, vous êtes en mesure de télécharger l'arbre de code source Wine. Veuillez vous assurer que vous êtes bien dans le répertoire dans lequel vous voulez mettre le code source Wine (le code source Wine utilise le sous-répertoire `wine/` dans ce répertoire, car le sous-répertoire est nommé après le module CVS que vous voulez extraire). Nous supposons que votre répertoire courant est votre répertoire personnel d'utilisateur. Pour télécharger l'arbre Wine dans le sous-répertoire `wine/`, lancez :

```
$  cvs -z3 checkout wine
```

Il se pourrait que le téléchargement de l'arbre CVS soit assez long (de quelques minutes à quelques heures), selon votre vitesse de connexion. Une fois le téléchargement terminé, vous devriez noter dans quel répertoire se trouve le répertoire `wine/` fraîchement téléchargé, en lançant **pwd** (Afficher le Répertoire Courant [Print Working Directory]) :

```
$ pwd
```

Plus tard, vous pourrez vous remettre dans ce répertoire en lançant :

```
$ cd <un_répertoire>
```

où <un_répertoire> est le répertoire retourné par **pwd**. En lançant

```
$ cd wine
```

vous pouvez maintenant vous placer dans le répertoire de l'arbre CVS Wine que vous venez de télécharger. Maintenant qu'en ayant suivi les étapes ci-dessus vous avez un arbre Wine qui fonctionne complètement, vous voilà fin prêt pour passer aux étapes d'installation et de configuration qui suivent.

2.3.3. Mettre à jour l'arbre CVS Wine

Au bout d'un certain temps, vous pourriez avoir envie de mettre à jour l'arbre CVS Wine à la version du moment. Avant de mettre l'arbre Wine à jour, il pourrait aussi être fort utile de lancer **make uninstall** en tant que root pour désinstaller l'installation de la version précédente de Wine.

Pour poursuivre la mise à jour de Wine, faites un simple **cd** dans le répertoire de l'arbre CVS Wine, puis lancez, si vous utilisez le serveur US :

```
$ make distclean
$ cvs update -PAd
```

La partie **make distclean** est facultative, mais il est fort utile d'enlever les vieux fichiers de configuration de construction et compilation avant de mettre à jour vers une version plus récente de Wine. Une fois que la mise à jour CVS est terminée, vous pouvez passer à l'installation de Wine à nouveau comme avant.

2.3.4. Mettre Wine à jour avec un correctif

Si vous avez récupéré le code source Wine (par example via un fichier d'archive tar), vous avez la possibilité d'appliquer des correctifs sur l'arbre source pour mettre à jour à une version plus récente de Wine ou pour corriger les bogues et ajouter des fonctionnalités expérimentales. Peut-être avez-vous trouvé un bogue, que vous l'avez signalé dans la liste de diffusion Wine (mailto:wine-devel@winehq.org), et reçu un fichier correctif pour réparer le bogue. Vous pouvez appliquer le correctif avec la commande **patch**, qui prend un correctif en transmission continue depuis `stdin` :

```
$ cd wine
$ patch -p0 <../correctif_à_appliquer.diff
```

Pour enlever le correctif, utilisez l'option *-R* :

```
$ patch -p0 -R <../correctif_à_appliquer.diff
```

Si vous voulez effectuer un test pour voir si le correctif va s'appliquer correctement (par exemple, si le correctif a été créé à partir d'une version plus ancienne ou plus récente de l'arbre), vous pouvez utiliser le paramètre *--dry-run* pour lancer le correctif sans écrire dans aucun fichier :

```
$ patch -p0 --dry-run <../correctif_à_appliquer.diff
```

patch est plutôt doué pour extraire des correctifs du milieu d'un fichier, donc si vous sauvegardez un courriel contenant un correctif dans un fichier sur votre disque dur, vous pouvez invoquer le correctif présent dans le courrier sans ouvrir les en-têtes et autres textes du courriel. **patch** ignore tout ce qui ne ressemble pas à un correctif.

L'option *-p0* appliquée à **patch** lui dit de garder le nom du fichier complet venant du fichier correctif. Par exemple, si le nom du fichier dans le correctif est `wine/programs/clock/main.c`. Utiliser l'option *-p0* applique le correctif au fichier de même nom, c'est-à-dire `wine/programs/clock/main.c` . Utiliser l'option -p1 inhibe la première partie du nom du fichier et applique le correctif à `programs/clock/main.c`. L'option *-p1* est utile si vous avez donné un nom à votre répertoire principal wine différent de celui de la personne qui vous a envoyé le correctif. Pour l'option *-p1* **patch** doit être exécuté à partir du répertoire principal wine.

Chapitre 3. Compiler la Source Wine

Comment compiler wine, et les éventuels problèmes que cela soulève...

Au cas où vous auriez téléchargé les fichiers du code source de Wine, ce chapitre vous indiquera comment le compiler en fichiers binaires avant de les installer. Sinon, veuillez vous reporter directement au chapitre d'Installation pour installer les fichiers binaires Wine.

3.1. Compiler Wine

3.1.1. Ce dont vous avez besoin

Pour une liste à jour des requis au niveau logiciel pour compiler Wine et aussi les instructions pour la façon de faire, veuillez voir le fichier README (http://www.winehq.org/source/README), qui est également disponible dans le répertoire principal d'un arbre code source Wine.

3.1.2. Espace nécessaire

Vous avez aussi besoin d'environ 400 Mo d'espace disque disponible pour la compilation. Le binaire compilé libwine.so prend environ 5 Mo d'espace disque, qui peuvent être réduits à à peu près 1 Mo en élaguant ('strip wine'). L'élaguage n'est cependant pas recommandé car vous ne pouvez pas soumettre des rapports de panne corrects avec un binaire élagué.

3.1.3. Problèmes courants

Si vous avez eu un sig11 à répétition en compilant shellord.c, thunk.c ou tout autre fichier, essayez de ne compiler que ce fichier sans optimisation (en enlevant l'option -Ox de la commande GCC dans le Makefile correspondant).

Chapitre 4. Installer ou désinstaller Wine

Une forme de distribution classique de Wine (que vous avez probablement téléchargé en suivant le chapitre Se procurer Wine) comprend pas mal de programmes, bibliothèques et fichiers de configuration différents. Ils doivent tous être configurés correctement pour que Wine fonctionne bien. Pour arriver à cela, ce chapitre va vous guider tout au long des étapes nécessaires pour que les fichiers Wine soient installés sur votre système. Il ne sera *pas* question de la *façon de configurer* l'environnement Windows Wine; c'est ce que traitera le prochain chapitre.

Lorsque vous installerez Wine, assurez-vous de ne pas réécrire sur une installation précédente de Wine (car cela pourrait causer un nombre impressionnant de conflits agaçants et fatals); il est recommandé de désinstaller toute version antérieure de Wine (comme expliqué dans ce chapitre) afin d'éviter ce problème.

4.1. Installer ou désinstaller des paquetages Wine

Maintenant que vous avez téléchargé le fichier paquetage Wine Debian, RPM ou autre, probablement avec les instructions données dans le chapitre précédent, vous vous demandez peut-être "C'est bien beau, mais qu'est-ce que je fais avec ce truc maintenant?". Cette section, espérons-le, pourra mettre fin à votre perplexité, en donnant des instructions d'installation détaillées pour tous les types de paquetages bien connus.

4.1.1. Linux Debian

Au cas où vous n'auriez pas téléchargé et installé automatiquement le paquetage Wine via **apt-get** comme décrit dans la section Se procurer Wine, vous devez maintenant utiliser la **dpkg** pour l'installer. Mettez-vous dans le répertoire dans lequel vous avez téléchargé le fichier paquetage Debian .deb. Une fois là, tapez ces commandes, en adaptant le nom du fichier paquetage comme requis :

```
$ su -
Mot de passe :
# cd /home/user
# dpkg -i wine_0.0.20030115-1.deb
```

(Tapez le mot de passe root à l'invite de commande "Mot de passe")

Vous avez aussi peut-être envie d'installer le paquetage wine-doc, et si vous utilisez Wine avec la distribution 2.3 (Woody), également le paquetage wine-utils.

On peut désinstaller un paquetage Debian Wine en lançant :

```
# dpkg -l|grep wine
```

La deuxième colonne de la sortie (s'il y en a une) de cette commande indiquera les paquetages installés en rapport avec "wine". Les paquetages correspondants peuvent être désinstallés en lançant :

```
# dpkg -r <nom_du_paquetage>
```

où <nom_du_paquetage> est le nom du paquetage lié à Wine que vous voulez désinstaller.

4.1.2. Linux RedHat, Mandrake, SUSE et autres distributions utilisant RPM

La plupart des distributions fournissent un outil graphique pour installer les paquetages RPM, vous pouvez l'utiliser en cliquant simplement (Ou en double-cliquant, selon les paramètres de votre système) sur le RPM. Si vous n'avez pas de gestionnaire de RPM graphique, d'installer en utilisant un interpréteur de commande [shell], placez-vous dans le répertoire dans lequel vous avez téléchargé le paquetage RPM. Une fois que vous y êtes, tapez cette commande en tant que root, en adaptant le nom du fichier paquetage comme requis :

```
# rpm -ivh wine-20031212.i386.rpm
```

Vous avez peut-être aussi envie d'installer le paquetage wine-devel

Si vous avez installé wine graphiquement, vous pouvez le désinstaller en utilisant votre gestionnaire de RPM graphique (Gnorpm, Kpackage, Yast, Centre de Contrôle Mandrake etc.), autrement, on peut désinstaller un paquetage Wine RPM installé à partir d'un interpréteur de commande, en lançant :

```
# rpm -qa|grep -i wine
```

Cette commande indiquera les paquetages installés liés à "wine". On peut désinstaller les paquetages correspondants en lançant :

```
# rpm -e <nom_du_paquetage>
```

où <nom_du_paquetage> est le nom du paquetage lié à Wine que vous voulez désinstaller.

4.2. Installer ou désinstaller un arbre de code source Wine

Si vous êtes dans le répertoire de la version Wine que vous venez de compiler (par exemple en ayant exécuté **make depend && make**), alors vous pouvez maintenant installer cette version de Wine en lançant en tant que *root* :

```
# make install
```

Avec cela, les fichiers binaires Wine seront copiés à leur destination finale dans le système. Vous pouvez ensuite passer au chapitre Configuration pour configurer l'environnement Wine.

Si au lieu de cela vous voulez désinstaller la version de code source Wine actuellement installée, placez vous dans le répertoire principal de cette version et lancez en tant que *root* :

```
# make uninstall
```

Chapitre 5. Configurer Wine

Maintenant que vous avez réussi, espérons-le, à installer les fichiers programmes de Wine, ce chapitre vous énonce comment paramétrer correctement l'environnement Wine afin d'utiliser vos programmes Windows.

Nous vous donnerons d'abord une vue d'ensemble sur certains points relatifs à l'exécution des programmes et à la configuration qu'un environnement Windows complètement configuré doit satisfaire afin d'assurer qu'un grand nombre de programmes Windows fonctionne correctement sans rencontrer de points non paramétrés ou manquants. Ensuite, nous vous montrerons les programmes d'aide simples disponibles pour permettre aux utilisateurs, y compris débutants, de réaliser la configuration d'un environnement Wine de manière simple et rapide. La section suivante expliquera l'utilité du fichier de configuration de Wine, et nous proposerons une liste de l'ensemble de ces paramètres. Après cela, la section suivante détaillera la partie de la configuration la plus importante et malheureusement la plus difficile : comment configurer le système de fichiers et l'environnement de commandes DOS que les programmes Windows requièrent. Dans la dernière étape, nous vous énoncerons comment établir une base de registre de Windows qui fonctionne. Enfin, le reste de ce chapitre contient les descriptions de paramétrages spécifiques de Wine qui pourraient également vous intéresser.

5.1. Quelles sont les exigences d'un environnement Windows totalement opérationnel ?

Un système Windows est une structure très complexe. Il se compose de nombreuses parties différentes contenant des fonctionnalités très variées. Nous allons essayer de présenter sommairement ce qui est le plus important dans ce système.

- Le registre. De nombreuses clés sont censées exister et contenir des données significatives même sur un Windows récemment installé.
- Architecture des répertoires. Les applications sont paramétrées pour trouver ou installer des choses à des emplacements spécifiques prédéterminés. La plupart de ces répertoires sont supposés exister. Mais, contrairement aux structures de répertoires Unix, la plupart de ces emplacements ne sont pas codés en dur et peuvent être récupérés via l'API Windows et le registre. De ce fait, des caractéristiques supplémentaires sont requises par l'installation.
- Les DLL système. Sous Windows, celles-ci résident habituellement dans le répertoire `system` (ou `system32`). Certains programmes Windows vérifient qu'elles existent dans ces répertoires avant de tenter de les charger. Alors que Wine est capable de charger ses propres DLL internes (fichiers `.so`) quand le programme en demande une, Wine ne simule pas la présence des fichiers non-existants.

Bien que les utilisateurs soient, bien sûr, libres d'installer tout eux-même, l'équipe Wine fera faire tout ce qui semble nécessaire de faire au script d'installation automatique des sources Wine, `tool/wineinstall`; l'exécution du cycle traditionnel **`configure && make depend && make && make install`** n'est pas recommandée, à moins que vous sachiez ce que vous faites. Actuellement,

tools/wineinstall est capable de créer un fichier de configuration, installer le registre et créer l'architecture des répertoires lui-même.

5.2. Assistant permettant de simplifier la configuration

Gérer les paramètres du fichier de configuration de Wine peut être une tâche difficile, parfois insurmontable pour certaines personnes. C'est pourquoi il existe quelques utilitaires d'assistance pour mettre en place facilement un fichier de configuration de Wine initial avec des paramètres par défaut utiles.

5.2.1. WineSetupTk

WineSetupTk est un outil graphique de configuration de Wine, proposant des moyens de configuration incroyablement simples à manipuler, à utiliser pour configurer l'environnement Wine après avoir installé les fichiers Wine. Edité par CodeWeavers en 2000, c'est l'un des nombreux projets visant à rapprocher Wine des applications de bureautique, et mis à jour en 2003 par Vincent Béron, Alex Pasadyn et Ivan Leo Puoti.

Si vous utilisez Debian, installez simplement le package WineSetupTk (en tant que super utilisateur [root]) :

```
# apt-get install winesetuptk
```

Si vous utilisez une autre distribution, vous pouvez obtenir WineSetupTk depuis la page de téléchargement de sourceforge.net. (http://sourceforge.net/project/showfiles.php?group_id=6241)

5.2.2. wineinstall

wineinstall est un petit outil de configuration situé à l'adresse `tools/wineinstall` dans l'aborescence du code source de Wine. Il a été édité afin de permettre une compilation/installation simple et complète du code source de Wine par les personnes qui ne veulent pas s'embêter à lire des tonnes de documentations très bien faites et instructives. ;-)

Une fois que vous avez extrait l'arborescence du code source avec succès, allez à sa racine puis lancez (en tant qu'utilisateur) :

```
$ ./tools/wineinstall
```

Le fait d'exécuter cette ligne compilera puis installera Wine et configurera l'environnement (soit en fournissant l'accès à une partition Windows soit en créant un environnement basé sur des répertoires non-windows correctement configuré).

5.3. Vérification de la validité de la configuration

A faire : Après avoir terminé la configuration de Wine, vous pouvez la vérifier en exécutant winecfg. Cependant cette fonctionnalité n'est pas encore intégrée à winecfg, elle le sera dans la version future.

Veuillez prendre connaissance de la documentation sur la configuration exposée ci-dessous afin d'en apprendre d'avantage sur le paramétrage de Wine ou bien passez au chapitre Dépannage.

5.4. Le fichier de configuration de Wine

Cette section est censée contenir à la fois une introduction facile pas à pas au fichier de configuration de Wine (pour les nouveaux utilisateurs de Wine) ainsi qu'une référence complète pour tous les paramètres de ce fichier (pour les utilisateurs avancés).

5.4.1. Introduction au fichier de configuration de Wine

Le fichier de configuration de Wine est le fichier central de stockage des paramètres de configuration. Ce fichier (appelé `config`) se trouve dans le sous répertoire `.wine/` de votre répertoire utilisateur (répertoire `/home/user/`). En d'autres termes, le fichier de configuration de Wine est `~/.wine/config`. A noter que depuis que le fichier de configuration de Wine fait partie du système de fichiers du registre Wine, ce fichier *requiert* une en-tête "WINE REGISTRY Version 2" correcte, afin d'être reconnu convenablement, simplement comme tous les autres fichiers texte du registre de Wine (juste au cas où vous auriez décidé d'écrire votre propre fichier de registre vous-même depuis le début et que vous vous demandiez pourquoi Wine ne cesse de le rejeter).

Le paramètres disponibles dans le fichier de configuration comprennent :

- le paramétrage des répertoires
- le paramétrage des Ports
- l'apparence de Wine
- l'utilisation des DLL de Wine
- la configuration des pilotes multimédia et des DLL de Wine

5.4.2. Créer ou modifier le fichier de configuration

Si vous installez Wine pour la première fois et si vous voulez terminer l'installation de Wine en le configurant maintenant, vous avez alors la possibilité d'utiliser notre échantillon de fichier de configuration `config` (qui se trouve dans le répertoire `documentation/samples/` dans le répertoire du code source de Wine) comme base pour adapter le fichier de configuration de Wine aux paramètres que vous voulez. Je mentionnerais d'abord qu'il est conseillé de ne pas oublier de vous assurer que tout fichier de configuration préalable `~/.wine/config` a été mis à l'abri ailleurs au lieu de l'écraser au moment de remplacer l'échantillon du fichier de configuration.

Si vous n'avez pas de fichier de configuration pré-existant, et que vous avez ainsi besoin de remplacer votre échantillon de fichier de configuration à l'emplacement standard du fichier de configuration de Wine, tapez dans une *console*:

```
$ mkdir ~/.wine/
$ cp dir_to_wine_source_code/documentation/samples/config ~/.wine/config
```

Sinon, utilisez simplement le fichier de configuration existant à l'adresse `~/.wine/config`.

Maintenant, vous pouvez commencer à adapter les paramètres du fichier de configuration avec un *éditeur* en suivant la documentation qui suit. Notez que vous devriez changer les paramètres du fichier de configuration *uniquement* si winserver n'est pas actif (en d'autres termes, si votre utilisateur n'a pas de session Wine activée), sinon Wine ne les utilisera pas, et même pire, Wineserver les écrasera par les anciens paramètres une fois qu'il s'arrêtera !!

5.4.3. Que contient-il ?

Commençons par dresser un aperçu des sections que peut contenir un fichier de configuration, en indiquant si l'inclusion de chaque section est respectivement *indispensable* ou seulement *recommandée*.

Nom de la section	Requise ?	Utilité
[wine]	oui	Paramètres généraux de Wine
[DllOverrides]	recommandée	Prise de contrôle manuel par défaut pour le chargement des DLL
[x11drv]	recommandée	Paramètres de pilotes graphiques
[fonts]	oui	Apparence et reconnaissance des polices
[ppdev]	non	Emulation de port parallèle
[spooler]	non	File d'attente d'impression
[ports]	non	Accès direct aux ports

Nom de la section	Requise ?	Utilité
[Debug]	non	Que faire face à certains messages d'erreur
[Registry]	non	Spécifie l'emplacement des fichiers de Registre Windows
[programs]	non	Programmes à exécuter automatiquement
[Console]	non	Paramètres de la console
[Clipboard]	non	Interaction entre Wine et le bloc-notes X11
[afmdirs]	non	Paramètres de pilotes Postscript
[WinMM]	oui	Paramètres multimédia
[AppDefaults]	non	Remplacement des paramètres des sections précédentes pour des programmes particuliers

A présent, passons à la description des sections du fichier de configuration en détail.

5.4.3.1. La section [wine]

La section [wine] du fichier de configuration contient tous types de paramètres généraux de Wine.

```
"Windows" = "c:\\windows"
"System" = "c:\\windows\\system"
"Temp" = "c:\\temp"
"Path" = "c:\\windows;c:\\windows\\system;c:\\blanco"
"ShowDirSymlinks" = "1"
```

Pour une description détaillée du fichier de configuration de la couche lecteurs, et la signification de ces paramètres, veuillez consulter la section lecteurs de disques, ports parallèles et ports séries.

```
"GraphicsDriver" = "x11drv|ttydrv"
```

Règle le pilote graphique pour utiliser la sortie Wine : x11drv pour la sortie X11 et ttydrv pour la sortie console en mode texte. ATTENTION : Si vous utilisez ttydrv ici, vous ne pourrez pas exécuter beaucoup de programmes GUI Windows (ttydrv est toujours loin d'être parfait pour l'exécution d'applications graphiques). Ainsi, cette option est surtout intéressante pour, par exemple, une utilisation de Wine intégré dans des scripts sur serveurs Web. Notez que ttydrv reste très simpliste, donc s'il ne fonctionne pas, ayez recours à l'utilisation de "xvfb", un serveur X11 virtuel. Une autre façon d'exécuter Wine sans affichage serait de lancer X11 via Wvnc, puis de se connecter à cet afficheur VNC en utilisant xvncviewer (de cette manière, vous avez encore la possibilité de vous connecter à votre application et de la configurer si besoin est).

```
"Printer" = "off|on"
```

donne à Wine l'autorisation d'imprimer via le pilote d'imprimante. Cette option n'est en aucun cas requise pour notre pilote d'imprimante psdrv intégré. Ces outils ont été très peu testés, donc la plus grande prudence s'impose. Certains pourraient trouver cette option utile malgré tout. Si vous n'envisagez pas de continuer à imprimer via les pilotes d'imprimantes de Windows, il est carrément inutile d'ajouter cette option à votre fichier de configuration (il n'y est déjà probablement pas). Consultez également les sections [spooler] et [paralleleports].

```
"ShellLinker" = "wineshelllink"
```

Ce paramètre indique le script shell d'édition de liens à utiliser pour paramétrer les icônes Windows, que l'on affecte aux programmes en se servant de la fonctionnalité shell32.dll appropriée, par exemple sous KDE ou Gnome, pour créer des icônes sur le bureau, ou dans le menu démarrer, au cours de l'installation.

```
"SymbolTableFile" = "wine.sym"
```

installe le fichier de la table des symboles pour le débogueur Wine. Vous n'avez sûrement pas besoin de bricoler ce truc là. Peut être utile si votre Wine est allégé [stripped].

5.4.3.2. La Section [DllOverrides]

Pour cette section, le format est identique pour chaque ligne:

```
<DLL>{,<DLL>,<DLL>...} = <FORM>{,<FORM>,<FORM>...}
```

Pour charger la paire Kernel intégrée (cas sans importance ici) par exemple :

```
"kernel,kernel32" = "builtin"
```

Pour charger la paire COMMDLG native (essayez l'intégrée si cela ne fonctionne pas) :

```
"commdlg,comdlg32" = "native, builtin"
```

Pour charger le COMCTL32 natif :

```
"comctl32" = "native"
```

Voici un bon paramétrage générique (tel que défini dans la configuration incluse à l'origine dans le package Wine):

```
[DllOverrides]
"rpcrt4"       = "builtin, native"
"oleaut32"     = "builtin, native"
"ole32"        = "builtin, native"
"commdlg"      = "builtin, native"
"comdlg32"     = "builtin, native"
"ver"          = "builtin, native"
```

```
"version"      = "builtin, native"
"shell"        = "builtin, native"
"shell32"      = "builtin, native"
"shfolder"     = "builtin, native"
"shlwapi"      = "builtin, native"
"shdocvw"      = "builtin, native"
"lzexpand"     = "builtin, native"
"lz32"         = "builtin, native"
"comctl32"     = "builtin, native"
"commctrl"     = "builtin, native"
"advapi32"     = "builtin, native"
"crtdll"       = "builtin, native"
"mpr"          = "builtin, native"
"winspool.drv" = "builtin, native"
"ddraw"        = "builtin, native"
"dinput"       = "builtin, native"
"dsound"       = "builtin, native"
"opengl32"     = "builtin, native"
"msvcrt"       = "native, builtin"
"msvideo"      = "builtin, native"
"msvfw32"      = "builtin, native"
"mcicda.drv"   = "builtin, native"
"mciseq.drv"   = "builtin, native"
"mciwave.drv"  = "builtin, native"
"mciavi.drv"   = "native, builtin"
"mcianim.drv"  = "native, builtin"
"msacm.drv"    = "builtin, native"
"msacm"        = "builtin, native"
"msacm32"      = "builtin, native"
"midimap.drv"  = "builtin, native"
; you can specify programs too
"notepad.exe"  = "native, builtin"
; default for all other DLLs
"*" = "native, builtin"
```

Note : Si le chargement des bibliothèques en début de liste échoue; Wine passera immédiatement à l'utilisation de la deuxième ou troisième possibilité.

5.4.3.3. La Section [fonts]

Cette section sert à configurer la gestion des polices de Wine.

```
"Resolution" = "96"
```

Comme X gère les polices différemment de Windows, Wine utilise un mécanisme particulier pour ça. Il doit ajuster leur taille en utilisant la valeur définie dans le paramètre "Resolution". 60-120 sont des valeurs acceptables, 96 est une bonne valeur intermédiaire. Si vous disposez des véritables polices Windows, ce paramètre ne sera pas si important. Bien sûr, il est toujours bon d'avoir des polices X qui fonctionnent convenablement sous Wine.

```
"Default" = "-adobe-times-"
```

Les polices par défaut que Wine utilise. Jouez avec ce paramètre si ça vous amuse.

OPTIONNEL:

Le paramètre `Alias` vous permet de mapper une police X à une police utilisée sous Wine. C'est bien pour les applications qui requièrent une police spéciale que vous ne possédez pas, mais dont il existe un bon remplacement. La syntaxe est la suivante :

```
"AliasX" = "[Fake windows name],[Real X name]"<,optional "masking" section>
```

Plutôt simple. Remplacez "AliasX" par "Alias0", puis par "Alias1" et ainsi de suite. Le faux nom Windows (Fake Windows name) est le nom que la police aura sous une application Windows dans Wine. Le nom Real X (Real X name) est le nom de la police vu par X (exécutez "xfontsel"). L'option "masking" vous permet d'utiliser le faux nom Windows que vous définissez. S'il n'est pas utilisé, wine essaiera juste d'extraire lui-même le faux nom Windows et n'utilisera pas le valeur que vous entrez.

Voici un exemple d'alias sans "masking". La police ressortira dans les applications windows en tant que "Google".

```
"Alias0" = "Foo,--google-"
```

Voici un exemple avec "masking" activé. La police ressortira en tant que "Foo" dans les applications Windows.

```
"Alias1" = "Foo,--google-,subst"
```

Pour plus d'informations, parcourez le chapitre Polices.

5.4.3.4. Les Sections [spooler] et [ports]

La section [spooler] indiquera à wine où mettre en file d'attente les travaux d'impression. Utilisez cette section si vous voulez essayer d'imprimer. La documentation de Wine signale que la file d'attente d'impression est "plutôt primitive" actuellement, elle ne marchera donc pas parfaitement. *Cette section est optionnelle.* Le seul paramètre que vous utilisez dans cette section sert à faire correspondre un port (LPT1, par exemple) avec un fichier ou une commande. Voici un exemple, faisant correspondre LPT1 au fichier `out.ps`:

```
"LPT1:" = "out.ps"
```

La commande suivante relie/mappe une tâche d'impression sur LPT1 à la commande **lpr**. Notez le |:

```
"LPT1:" = "|lpr"
```

La section [port] n'est en général utile que pour les personnes qui ont besoin d'un accès direct aux ports pour les programmes qui requièrent des clés gigognes ou des scanners. *Si vous n'en avez pas besoin, ne l'utilisez pas*

```
"read" = "0x779,0x379,0x280-0x2a0"
```

Donne l'accès direct en lecture à ces entrées/sorties.

```
"write" = "0x779,0x379,0x280-0x2a0"
```

Donne l'accès direct en écriture à ces entrées/sorties. Sans doute une bonne chose à faire : donner les mêmes valeurs aux paramètres `read` et `write`. Ce truc ne fonctionnera que lorsque vous serez super-utilisateur (root).

5.4.3.5. Les Sections [Debug], [Registry], et [programs]

[Debug] est utilisé pour inclure ou exclure des messages d'erreurs, et pour les sortir dans un fichier. Ce dernier cas est rarement utilisé. *Tout cela est optionnel et vous n'aurez probablement pas besoin d'ajouter ou retirer quoi que ce soit dans cette section pour votre configuration.* (Dans les cas extrêmes, vous aurez peut-être besoin d'utiliser ces options pour gérer la quantité d'informations générées par *`WINEDEBUG=+relay`*)

```
"File" = "/blanco"
```

Configure le journal de bord (fichier de log) pour Wine. Paramétrez à CON pour enregistrer dans une sortie standard. *Ceci est rarement utilisé.*

```
"SpyExclude" = "WM_SIZE;WM_TIMER;"
```

écarte du journal de bord (fichier de log) les messages d'erreur concernant `WM_SIZE` et `WM_TIMER`.

```
"SpyInclude" = "WM_SIZE;WM_TIMER;"
```

inclut dans le journal de bord (fichier de log) les messages d'erreur concernant `WM_SIZE` et `WM_TIMER`.

```
"RelayInclude" = "user32.CreateWindowA;comctl32.*"
```

inclut uniquement les fonctions inscrites dans une trace *`WINEDEBUG=+relay`*. Cette entrée est ignorée s'il y a une entrée *`RelayExclude`*.

```
"RelayExclude" = "RtlEnterCriticalSection;RtlLeaveCriticalSection"
```

écarte les fonctions listées dans une trace *`WINEDEBUG=+relay`*. Cette entrée annule tous les paramètres de l'entrée *`RelayInclude`*. S'il n'existe aucune entrée, le traçage inclut tout.

Dans les deux entrées, les fonctions peuvent être spécifiées soit comme un nom de fonction, soit comme un module et une fonction. Dans ce dernier cas, spécifiez un astérisque en guise de nom de fonction pour inclure/écarter toutes les fonctions dans le module.

[Registry] peut être utilisé pour dire à wine où vos anciens fichiers registres windows se situent. Cette section est totalement facultative et inutile pour les personnes qui utilisent wine sans installation de windows existante.

```
"UserFileName" = "/dirs/to/user.reg"
```

L'emplacement de votre ancien fichier `user.reg`.

[programs] peut être utilisé pour dire quels programmes fonctionnent sous des conditions particulières.

```
"Default" = "/program/to/execute.exe"
```

Spécifie que le programme doit être lancé si wine est démarré sans spécification d'un programme.

```
"Startup" = "/program/to/execute.exe"
```

Spécifie que le programme doit automatiquement être lancé à chaque démarrage.

5.4.3.6. La section [WinMM]

[WinMM] est utilisé pour définir quels pilotes multimedia doivent être chargés. Puisqu'il se peut que ces pilotes dépendent des interfaces multimedia disponibles sur votre système (OSS, ALSA... pour n'en citer que quelques uns), il faut configurer quels pilotes doivent être chargés.

Le contenu de ces sections ressemble à ceci :

```
[WinMM]
"Drivers" = "wineoss.drv"
"WaveMapper" = "msacm.drv"
```

```
"MidiMapper" = "midimap.drv"
```

Toutes les clés doivent êtres définies :

- La clé "Drivers" est une liste de noms de modules séparés par ";", chacun d'entre eux contenant un pilote bas-niveau. Tous ces drivers seront chargés au démarrage de MMSYSTEM/WINMM et fourniront leurs propres caractéristiques.
- Le "WaveMapper" représente le nom du module contenant le pilote de l'adaptateur Wave. Un seul adpateur Wave peut être défini dans le système.
- Le "MidiMapper" représente le nom du module contenant le pilote de l'adaptateur MIDI. Un seul adapteur MIDI peut être défini dans le système.

5.4.3.7. La section [Network]

[Network] contient les paramètres liés au réseau. Actuellement, une seule valeur peut être fixée.

UseDnsComputerName

Un paramètre booléen (`Y` par défaut) qui fixe la manière dont Wine paramètre le nom de l'ordinateur. Dans le monde Windows, le nom qu'il donne à la machine est le *nom NetBIOS*. Il est contenu dans le `ComputerName` dans l'entrée de registre `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\ComputerName\ComputerName`.

Si cette option est fixée à "Y" ou manquante, Wine fixera le nom NetBIOS au nom hôte Unix de votre machine, si nécessaire tronqué à 31 caractères. Le nom hôte Unix est le résultat de la commande shell **hostname**, jusqu'au premier point ('.') non inclus. Il signifie, entre autres, que les programmes Windows fonctionnant sous Wine ne peuvent changer le nom de machine NetBIOS.

Si cette option est fixée à "N", Wine utilisera la valeur de registre ci-dessus pour fixer le nom NetBIOS. Seulement si l'entrée registre n'existe pas (normalement uniquement pendant le premier démarrage de Wine) il utilisera le nom hôte Unix comme d'habitude. Les programmes Windows peuvent changer le nom NetBIOS. Le changement sera effectif après un "redémarrage", c'est-à-dire après le redémarrage de Wine.

5.4.3.8. La section [AppDefaults]

La section est utilisée pour remplacer certains paramètres de ce fichier pour un programme particulier avec différents paramétrages. [AppDefaults] n'est pas le nom réel de cette section. Le nom réel se compose du mot principal AppDefaults suivi du nom de l'exécutable pour lequel cette section est

applicable. La fin du nom de la section est le nom de la section "standard" correspondant au fichier de configuration dont certains des paramètres doivent être écrasés par les paramètres spécifiques du programme qui sont définis par vous. Les trois parties du nom de la section sont séparées par deux barres obliques inverses.

Actuellement, wine permet l'annulation des paramètres sélectionnés seulement à l'intérieur des sections [DllOverrides], [x11drv], [version] et [dsound].

Voici un exemple qui annule les paramètres normaux d'un programme :

```
;; default settings
[x11drv]
"Managed" = "Y"
"Desktop" = "N"

;; run install in desktop mode
[AppDefaults\\install.exe\\x11drv]
"Managed" = "N"
"Desktop" = "800x600"
```

5.4.4. Et si ça ne marche pas ?

Il y a toujours un risque que les choses tournent mal. Si l'impensable se produit, signalez le problème à Wine Bugzilla (http://bugs.winehq.org/), essayez le newsgroup comp.emulators.ms-windows.wine, ou le canal IRC #WineHQ qui se trouve sur irc.freenode.net, ou les serveurs en rapport avec le sujet. Assurez-vous que vous avez parcouru la documentation minutieusement, et que vous avez également lu :

- `README`
- `http://www.winehq.org/trouble/`

S'il s'avère que vous avez bien fait vos recherches, vous pouvez vous attendre à recevoir des suggestions utiles. Sinon, préparez-vous, vous allez vous faire incendier.

5.5. Lecteurs de disques, ports parallèles et ports séries

5.5.1. Prérequis extrêmement importants

Si vous envisagez d'intégrer l'accès à un lecteur de CD-ROM dans votre configuration de Wine sous

Linux, *ne manquez pas* d'ajouter l'option de montage « unhide » à l'entrée du système de fichiers du CD-ROM dans `/etc/fstab`, par exemple :

```
/dev/cdrom /cdrom  iso9660 ro,noauto,users,unhide 0 0
```

Plusieurs CD-ROM d'installations de programmes Windows ou autres CD-ROM choisissent de marquer comme "cachés" des fichiers d'aide à l'installation très importants. Cela n'est pas très futé. Il n'y a pas de problème sous Windows, puisque les pilotes Windows de lecteur de CD-ROM par défaut affichent même les fichiers supposés « cachés ». Mais sous Linux, qui choisit par défaut de *cacher* les fichiers« cachés » sur CD, c'est *FATAL*! (les programmes abandonneront simplement par un message d'erreur « fichier d'installation non trouvé » ou similaire). C'est pourquoi vous ne devriez jamais oublier d'ajouter ce paramètre.

5.5.2. Courte introduction

Les applications Windows renvoient aux lecteurs de disques par des lettres tel ques `A:`, `B:` et `C:`, et aux ports parallèles et séries par des noms tels que `COM1:` et `LPT1:`.

Vous devez dire à Wine comment les interpréter. Cela se fait en spécifiant les noeuds du système de fichiers Unix et les périphériques vers lesquels Wine devrait pointer, comme décrit plus loin dans la section.

Vous pouvez mapper un lecteur de disques fixes Windows vers n'importe quel noeud dans votre système de fichiers Unix - il n'est pas nécessaire que ce noeud soit le noeud racine d'un lecteur. Par exemple, vous pourriez mapper votre lecteur Windows `C:` vers votre répertoire Unix `/usr/share/wine-C`. Alors, le fichier Windows `C:\Windows\Fonts` serait à `/usr/share/wine-C/Windows/Fonts` dans votre système de fichiers Unix.

Assurez-vous que vous avez attribué les lettres de lecteurs aux répertoires qui couvriront tous les éléments auxquels Wine doit accéder. Ceux-ci comprennent les programmes que vous exécutez, les fichiers de données dont ils ont besoin et le débogueur Wine (au cas où quelque chose tourne mal).

Le mieux est d'utiliser un certain nombre de lettres de lecteurs, et de les mapper vers les répertoires qui couvrent les petites sections des fichiers système contenant les fichiers auxquels Wine devra accéder. C'est plus sûr que d'attribuer simplement une unique lettre de lecteur au répertoire racine Unix /, ce qui permettrait aux applications Windows d'accéder à l'ensemble de votre système de fichiers Unix (à condition, bien sûr, de posséder les droits Unix). Si l'une de ses applications se comportait mal, ou si vous installiez accidentellement un virus, ceci pourrait vous rendre vulnérable.

Pour les supports de données amovibles, tels que les disquettes et les CD-ROM, vous devriez mapper une lettre de lecteur Windows au point de montage de ces lecteurs dans votre système de fichiers Unix - par exemple `/mnt/floppy` ou `/mnt/cdrom`.

Si vos applications accèdent directement aux ports séries et parallèles, vous devriez les mapper aux périphériques Unix correpondants - par exemple `/dev/ttyS0` et `/dev/lp0`.

5.5.3. Architecture de répertoires Windows

Voici l'organisation fondamentale que les programmes Windows et les installateurs attendent et que nous devons ainsi configurer convenablement dans Wine. Sans elle, ils fonctionneront rarement correctement. Si vous avez l'intention d'utiliser un environnement non-windows (n'utilisant pas une partition Windows existante), il est recommandé d'utiliser soit **WineSetupTk** soit **wineinstall** pour leur capacité à créer une arborescence de répertoires Windows initiale, puisque la création d'une architecture de répertoires manuellement est ennuyeuse et sujette à erreurs.

```
C:\                Répertoire racine du lecteur de disque primaire
  Windows\         Répertoire Windows, contenant les fichiers .INI, les accessoires, etc.
    System\        Répertoire Win3.x/95/98/ME pour les DLL communes
                     Répertoire WinNT/2000 pour les DLL 16-bits communes
    System32\      Répertoire WinNT/2000 pour les DLL 32-bits communes
    Start Menu\    Architecture des répertoires des lanceurs de programmes
      Programs\    Liens de lanceurs de programmes vers les programmes (fichiers .LNK)
  Program Files\   Binaires d'applications (fichiers .EXE et .DLL)
```

5.5.4. Le répertoire dosdevices

Le répertoire `dosdevices` contient les entrées qui spécifient à Wine comment mapper les lettres de lecteurs de disques Windows aux noeuds du système de fichiers Unix, et comment mapper les ports parallèle et série Windows aux périphériques Unix. Il est situé dans le sous-répertoire `.wine` de votre répertoire utilisateur (home), c'est-à-dire `~/.wine/dosdevices`.

Les entrées dans le répertoire `dosdevices` sont des liens symboliques vers les noeuds du système de fichiers Unix et les périphériques. Vous pouvez les créer en utilisant la commande **ln** dans une console Unix. Sinon, de nombreux gestionnaires de fichiers peuvent créer des liens symboliques.

Par exemple, si vous avez décidé de mapper votre lecteur Windows `C:` à `/usr/share/wine-c`, vous pourriez saisir la ligne suivante (après vous être déplacés dans votre répertoire `dosdevices`):

```
ln -s /usr/share/wine-c c:
```

Les supports de données amovibles sont un peu plus compliqués. En plus de créer un lien pour le système des fichiers présents sur le support, par exemple :

```
ln -s /mnt/floppy a:
```

vous devez aussi créer un lien pour le périphérique lui-même. Notez les deux ":" après la lettre du lecteur :

```
ln -s /dev/fd0 a::
```

Pour les ports séries et parallèles, vous créez simplement un lien vers le périphérique; notez qu'aucun ":" n'est requis après le nom de périphérique Windows :

```
ln -s /dev/ttyS0 com1
ln -s /dev/lp0 lpt1
```

Les partages Windows peuvent être mappés au répertoire `unc/`, ainsi tout ce qui essaie d'accéder `\\machinename\some\dir\and\file` ira chercher dans `~/.wine/dosdevices/unc/machinename/some/dir/and/file`. Par exemple, si vous utilisez Samba pour monter `\\machinename\some` sur `/mnt/smb/machinename/some` vous pouver faire

```
ln -s /mnt/smb/machinename/some unc/machinename/some
```

pour le rendre disponible dans Wine (n'oubliez pas de créer le répertoire unc s'il n'existe pas déjà).

5.5.5. Paramètres du système de fichiers dans la section [wine]

```
"Windows" = "c:\\windows"
```

Spécifie à Wine et aux programmes Windows où le répertoire `Windows` se situe. Il est recommandé d'avoir ce répertoire quelque part sur votre lecteur C configuré, et il est également recommandé d'appeler le répertoire simplement "windows" (c'est la configuration par défaut sous Windows, et certains programmes bêtes pourraient en dépendre). Ainsi, au cas où vous choisiriez un paramétrage "Windows" de "c:\\windows" et que vous choisiriez d'installer un lecteur C par exemple à `/usr/local/wine_c`, le répertoire correspondant serait `/usr/local/wine_c/windows`. Faites-en un si vous n'en avez pas déjà un . *Pas de slash qui traîne* (*pas de* `C:\\windows\`)! L'accès en écriture est fortement recommandé, puisque les programmes Windows considèrent qu'il y a toujours accès en écriture vers le répertoire Windows!

```
"System" = "c:\\windows\\system"
```

Indique où les fichiers du système Windows sont. Le répertoire du système Windows devrait résider sous le répertoire utilisé pour le paramétrage de `Windows`. Ainsi, quand on utilise l'exemple ci-dessus, le répertoire system serait `/usr/local/wine_c/windows/system`. Là encore, pas de slash qui traîne, et accès en écriture!

```
"Temp" = "c:\\temp"
```

Indique le répertoire dans lequel vous voulez que soient stockés vos fichiers temporaires, /usr/local/wine_c/temp dans notre exemple. Là encore, pas de slash qui traîne, et *accès en écriture*!!

```
"Path" = "c:\\windows;c:\\windows\\system;c:\\blanco"
```

se comporte comme le paramètre PATH sous les systèmes UNIX. Quand Wine est exécuté par **`wine sol.exe`**, si `sol.exe` réside dans le répertoire spécifié dans le paramètre `Path`, wine l'exécutera (bien sûr, si `sol.exe` réside dans le répertoire courant, c'est celui-là que wine lancera). Assurez-vous qu'il possède tout le temps votre répertoire `windows` et votre répertoire système (Pour cette configuration, il doit avoir `"c:\\windows;c:\\windows\\system"`).

```
"ShowDirSymlinks" = "1"
```

Par défaut, Wine ne relaie pas les liens symboliques de répertoires vers les programmes Windows, ceci pouvant planter certains programmes qui font des recherches récursives sur l'ensemble de l'arborescence des sous-répertoires, chaque fois qu'un lien symbolique de répertoire pointe vers lui-même ou vers un de ses répertoires parents. C'est pourquoi nous n'avons pas autorisé l'utilisation de liens symboliques de répertoires et avons ajouté ce paramètre pour réactiver ("1") cette fonctionnalité. Si vous avez *vraiment* besoin que Wine prenne en compte les liens symboliques de répertoires, alors activez-le, mais *attendez-vous à des plantages* dans certains programmes Windows en utilisant la méthode ci-dessus! (en d'autres termes, nous ne recommandons certainement pas de l'activer).

5.5.6. Explications détaillées supplémentaires sur les différences entre systèmes de fichiers.

Windows utilise une manière différente d'Unix (et moins bonne) pour décrire l'emplacement des fichiers sur l'ordinateur. Ainsi les programmes Windows s'attendent aussi à ce que le système prenne en charge cette méthode différente. Ayant l'intention de faire fonctionner des programmes Windows sous un système Unix, on rencontre des problèmes, puisque nous devons faire une conversion entre ces différentes techniques d'accès aux fichiers.

Windows utilise des lettres de lecteurs pour décrire les lecteurs ou tout autre forme de supports de stockage et pour accéder aux fichiers qu'ils contiennent. Par exemple, les noms courants de lecteurs sont `C:` pour la partition principale du système Windows sur le premier disque dur et `A:` pour le premier lecteur de disquettes. D'autre part, Windows utilise `\` (barre oblique inverse) comme symbole de

séparation de répertoires, alors qu'Unix utilise `/` (barre oblique). Ainsi, par exemple, on accéderait à un document sur la partition principale de données sous Windows par le nom de `D:\mywork\mydocument.txt`.

Voilà pour ce qui est de la façon de faire de Windows

Bon, le problème est que, sous Unix les « lettres de lecteurs » ça n'existe pas. A la place, Unix choisit un méthode qui est bien meilleure et qui consiste à avoir une seule arborescence uniforme de répertoires (commençant par le répertoire racine `/`), qui intègre différents supports de stockage tels que les partitions de disques durs ajoutées aux différents emplacements de répertoires à l'intérieur de l'arbre (un exemple serait `/data1/mywork`, qui est la partition principale de données montée/attachée au répertoire appelé data1 dans le répertoire racine `/`; mywork est un sous répertoire du système de fichiers de la partition de données qui est montée sous `/data1`). Sous Unix, on pourrait accéder au document Windows donné en exemple ci-dessus par le nom `/data1/mywork/mydocument.txt`, à condition que l'administrateur ait décidé de monter (attacher) la partition principale de données sur le répertoire /data1 à l'interieur de l'arborescence des répertoires Unix. Notez que sous Unix, l'administrateur peut *choisir* n'importe quelle emplacement de partition d'usage selon ce qu'il veut (ici, `/data1`), alors que sous Windows le système *sélectionne* n'importe quelle lettre de lecteur qu'il juge approprié pour la partition principale de données (ici, `D:`), et, même pire, s'il y a des modifications dans l'ordre des partitions, Windows *change* automatiquement la lettre de lecteur, et vous pourriez soudainement vous trouver avec une partition principale de données à la lettre de lecteur `E:`, avec toute la confusion sur le nommage et le référencement de fichiers que cela entraîne. Ainsi, la manière dont Windows utilise les lettres de lecteurs qui changent tout le temps est *clairement moins bonne* que celle d'Unix qui assigne un emplacement *fixe* dans l'arborescences des répertoires pour tous les supports de stockage de données. Comme nous le verrons bientôt, heureusement cet inconvénient de Windows causé par la modification des lettres de lecteurs ne nous affecte pas du tout dans Wine, puisque nous pouvons correctement faire correspondre des lettres de lecteurs *inchangeables* à des emplacements *fixes* à l'intérieur de l'arborescence des répertoires Unix (et même si l'emplacement du répertoire Unix respectif change, il vous est toujours possible de simplement mettre à jour le mappeur de lecteurs Wine pour répercuter la mise à jour de l'emplacement et en même temps conserver la lettre de lecteur d'origine).

Bon, maintenant que nous avons les bases sur correspondance entre les lecteurs et les noms de fichier sous Windows et Linux, il est sans doute temps de se demander comment Wine réussit le tour de passe-passe de mapper un emplacement de répertoire Unix à un lecteur Windows ...

Wine choisit de faire comme suit : dans Wine, vous n'assignez pas de support de stockage physique réel (tel qu'une partition de disque dur ou similaire) à chaque entrée de correpondance avec une lettre de lecteur. A la place, vous choisissez certaines arborescences de sous-répertoires à l'intérieur de l'arborescence des répertoires Unix (qui commence par `/`) auxquelles vous aimeriez assigner une lettre de lecteur.

Notez que pour toute arborescence de sous-répertoires Unix dans lesquelles vous avez l'intention de démarrer des programmes Windows, il est *absolument nécessaire* d'avoir une entrée de correspondance à un lecteur Wine :

Par exemple, si vous aviez un « espace de répertoires Windows » sous `/usr/mywine` pourvu des droits publics en écriture, pour pouvoir accéder à partir de Wine à cette arborescence de sous-répertoires, vous devriez avoir une entrée de correspondance de lecteur qui fait correspondre une certaine lettre de lecteur (par exemple, prenons la lettre de lecteur `P:`) soit à `/usr/mywine` ou à `/usr` (pour accéder aussi à tout répertoire appartenant au répertoire parent) ou encore à `/` (pour accéder aussi, par cette correspondance à la lettre de lecteur, à n'importe quel répertoire sur le système quel qu'il soit). L'emplacement du lecteur/répertoire DOS pour accéder aux fichiers dans `/usr/mywine` *sous Wine* dans ces cas de configuration serait alors respectivement `P:\` ou `P:\mywine` ou `P:\usr\mywine`

5.5.7. Installer Wine sans Windows

Un des objectifs principaux de Wine est de permettre aux utilisateurs d'exécuter des programmes Windows sans avoir à installer Windows sur leur machine. Wine implémente les fonctionnalités des DLL principales habituellement fournies avec Windows. Par conséquent, quand Wine sera terminé, il ne sera pas nécessaire que Windows soit installé pour utiliser Wine.

Wine a déjà fait assez de progrès pour qu'il puisse être possible de lancer vos programmes cibles sans que Windows soit installé. Si vous voulez l'essayer, suivez ces étapes :

1. Faites un lien symbolique dans `~/.wine/dosdevices` vers le répertoire où vous voulez que `C:` soit. Reportez-vous à la page du manuel de Wine pour plus d'informations. Le répertoire à utiliser pour émuler un lecteur `C:` sera le répertoire de base pour certains répertoires Windows spécifiques créés ci-dessous.
2. A l'intérieur du répertoire à utiliser pour C:, créez les répertoires vides suivants : `windows`, `windows/system`, `windows/Start Menu`, et `windows/Start Menu/Programs`. Ne pointez pas Wine vers un répertoire `Windows` plein de vieilles installations et d'un registre bancal. (Wine crée un registre spécial dans votre répertoire utilisateur `home`, dans `$HOME/.wine/*.reg`. Il vous faudra peut-être enlever ces fichiers). En une ligne : mkdir -p windows windows/system windows/Start\ Menu windows/Start\ Menu/Programs
3. Lancez et/ou installez vos programmes.

Parce que Wine n'est pas encore terminé, certains programmes fonctionneront mieux avec les DLL Windows natives qu'avec les remplaçants de Wine. Wine a été conçu pour que ce soit possible. Voici certains tuyaux de Juergen Schmied (et d'autres) sur la façon de procéder. On considère a priori que votre répertoire `C:\windows` dans le fichier de configuration ne pointe pas vers une installation native de Windows mais est dans un système de fichiers Unix séparé. (Par exemple, « C:\windows » est réellement un sous-répertoire « windows » situé dans « /home/ego/wine/drives/c »).

- Lancez le programme avec *`WINEDEBUG=+loaddll`* pour trouver quels fichiers sont requis. Copiez les DLL requises une par une vers le répertoire `C:\windows\system`. Ne copiez pas KERNEL/KERNEL32, GDI/GDI32, USER/USER32 ou NTDLL. Ces derniers implémentent la fonctionnalité du noyau de l'API Windows, et les versions internes de Wine doivent être utilisées.
- Editez la section « [DllOverrides] » de `~/.wine/config` pour spécifier « native » avant « builtin » pour les DLL Windows que vous voulez utiliser. Pour plus d'informations à ce sujet, voyez la page du manuel de Wine.

- Notez que certaines DLL de réseau ne sont pas requises même si Wine les recherche. Le fichier Windows `MPR.DLL` ne fonctionne pas actuellement; vous devez utiliser l'implémentation interne.
- Copiez SHELL.DLL/SHELL32.DLL, COMMDLG.DLL/COMDLG32.DLL et COMMCTRL.DLL/COMCTL32.DLL uniquement par paires dans votre répertoire Wine (ces DLL sont "propres" à utiliser). Assurez-vous qu'elles sont spécifiées dans la section « [DllPairs] » de `~/.wine/config`.
- Soyez cohérents : Utilisez ensemble seulement les DLL de la même version Windows.
- Mettez `regedit.exe` dans le répertoire `C:\windows`. (Office 95 importe un fichier `*.reg` quand il fonctionne avec un registre vide, nous ne savons rien sur ce point en ce qui concerne Office 97). A compter de ce jour, il pourrait ne plus être nécessaire d'utiliser regedit.exe, puisque Wine possède maintenant sa propre application Winelib regedit.
- Ajoutez également `winhelp.exe` et `winhlp32.exe` si vous voulez avoir la possibilité de parcourir la fonction d'aide de vos programmes (ou bien si vous trouvez que l'implémentation de la fonction winhelp de Wine dans programs/winhelp/ n'est pas assez bien, par exemple).

5.5.8. Installer Wine en utilisant comme base une partition Windows existante

Certains envisagent d'utiliser avec Wine les données d'une partition Windows existante afin de profiter de meilleurs compatibilités ou de faire fonctionner des programmes déjà installés sous une installation qui soit autant que possible d'origine. Notez que de nombreux programmes Windows considèrent a priori qu'ils ont l'accès intégral en écriture vers les répertoires Windows. Ceci signifie soit que vous devez configurer le point de montage de la partition windows pour les permissions en écriture en tant qu'utilisateur Wine (voir Traiter Les Partitions FAT/VFAT sur le façon de procéder), soit que vous devrez copier le contenu (ou certaines parties) de la partition Windows dans un répertoire d'une partition Unix et vous assurer que votre utilisateur peut écrire dans cette architecture de répertoires. Nous vous *INCITONS FORTEMENT* à ne pas utiliser directement une partition Windows avec un accès en écriture comme base pour Wine !! (certains programmes, notamment Explorer, corrompent de grandes parties de la partition Windows dans le cas d'une installation incorrecte; vous êtes prévenus). Inutile de préciser que la prise en charge du NTFS en écriture sous Linux reste très expérimentale et *dangereuse* (au cas où vous utiliseriez une version de Windows basée sur NT utilisant un système de fichiers NTFS). Nous vous conseillons donc d'adopter la méthode Unix de gestion des répertoires.

5.5.9. Les partitions FAT/VFAT

Ce document décrit comment les permissions sur les systèmes de fichiers FAT et VFAT fonctionnent sous Linux en portant l'accent sur leur configuration pour Wine.

5.5.9.1. Introduction

Linux est capable d'accéder aux systèmes des fichiers DOS et Windows en utilisant soit les modules FAT

(vieux systèmes de fichiers DOS 8.3) ou VFAT (systèmes de fichiers Windows 95 plus récents ou systèmes de fichiers à noms de fichiers longs ultérieurs). Les systèmes de fichiers FAT ou VFAT montés fournissent le principal moyen d'accès aux programmes existants et à leurs données depuis Wine pour les systèmes à double démarrage (Linux + Windows).

Wine mappe les systèmes de fichiers FAT montés, tels que `/c`, aux lettres de lecteurs, telles que « c: », au moyen des liens symboliques dans le répertoire `dosdevices`. Ainsi, dans votre répertoire dosdevices, vous pourriez taper la commande :

```
ln -s /c c:
```

Bien que les systèmes de fichiers VFAT soient préférables aux systèmes de fichiers FAT pour la prise en charge des noms de fichiers longs, le terme « FAT », utilisé dans toute la suite de ce document, correspond aux systèmes de fichiers FAT et leurs dérivés. En outre, « /c » sera utilisé, dans tout ce document, comme point de montage FAT dans les exemples donnés.

La plupart des distributions Linux récentes détectent les systèmes de fichiers FAT existants ou bien permettent qu'ils soient configurés de sorte qu'ils puissent être montés, dans un emplacement tel que `/c`, soit de manière persistante (au démarrage), soit ponctuellement, en cas de besoin. Dans un cas comme dans l'autre, par défaut, les permissions seront sûrement configurées à peu près comme ceci :

```
~>cd /c
/c>ls -l
-rwxr-xr-x   1 root     root           91 Oct 10 17:58 autoexec.bat
-rwxr-xr-x   1 root     root          245 Oct 10 17:58 config.sys
drwxr-xr-x  41 root     root        16384 Dec 30  1998 windows
```

Là où tous les fichiers appartenant au "super-utilisateur [root]", se trouve dans le groupe "super-utilisateur [root]" et sont seulement inscriptibles par le "super-utilisateur [root]" (permission `755`). Ceci est restrictif dans la mesure où il est indispensable que Wine soit lancé en super-utilisateur [root] afin que les programmes puissent écrire dans n'importe quelle partie du système de fichier.

Il y a trois approches principales pour surpasser les permissions restrictives mentionnées dans le paragraphe ci-dessus :

1. Exécuter Wine en tant que super-utilisateur [root]
2. Monter le système de fichiers FAT avec des permissions moins restrictives
3. Créer un système fantôme de fichiers FAT en le copiant complètement ou partiellement

Chaque approche sera traitée dans les sections suivantes.

5.5.9.2. Exécuter Wine en tant que super-utilisateur [root]

Lancer Wine en tant que super-utilisateur est la manière la plus simple et la plus complète de donner aux programmes que Wine exécute des accès libres aux systèmes de fichiers FAT. Le fait de lancer Wine en tant que super-utilisateur autorise également les programmes à faire d'autres choses, sans rapport avec les systèmes de fichiers FAT, telles qu'écouter les ports inférieurs à 1024. Exécuter Wine en tant que super-utilisateur est dangereux puisqu'il n'y a pas de limite à ce que le programme peut faire sur le système, ainsi c'est *FORTEMENT DECONSEILLE*.

5.5.9.3. Monter les systèmes de fichiers FAT

Les systèmes de fichiers FAT peuvent être montés avec des permissions moins restrictives que celles par défaut. Ceci peut être fait soit en changeant l'utilisateur qui monte le système de fichiers FAT, soit en modifiant explicitement les permissions avec lesquelles le système de fichiers est monté. Les permissions sont héritées du processus qui monte le système de fichiers FAT. Puisque le processus qui monte le système de fichiers FAT est habituellement un script de démarrage fonctionnant en tant que super-utilisateur, le système de fichiers FAT hérite des permissions du super-utilisateur. Ceci se répercute dans les fichiers du système de fichiers FAT par des permissions similaires à celles des fichiers créés par le super-utilisateur (root). Par exemple:

```
~>whoami
root
~>touch root_file
~>ls -l root_file
-rw-r--r--   1 root     root            0 Dec 10 00:20 root_file
```

ce qui correspond au propriétaire, au groupe et aux permissions des fichiers vus dans le système de fichiers FAT à l'exception des 'x' manquants. Les permissions sur le système de fichiers FAT peuvent être modifiées en changeant l'umask (désactiver les bits de permissions). Par exemple:

```
~>umount /c
~>umask
022
~>umask 073
~>mount /c
~>cd /c
/c>ls -l
-rwx---r--   1 root     root           91 Oct 10 17:58 autoexec.bat
-rwx---r--   1 root     root          245 Oct 10 17:58 config.sys
drwx---r--  41 root     root        16384 Dec 30  1998 windows
```

Le fait de monter le système de fichiers FAT avec un umask de `000` donne à tous les utilisateurs le contrôle complet sur le système de fichiers. Spécifier explicitement les permissions sur le système de fichiers FAT, une fois celui-ci monté, fournit un contrôle supplémentaire. Il y a trois options de montage

qui se rapportent aux permissions FAT : `uid`, `gid` et `umask`. Elles peuvent être chacune spécifiées une fois le système de fichier manuellement monté. Par exemple:

```
~>umount /c
~>mount -o uid=500 -o gid=500 -o umask=002 /c
~>cd /c
/c>ls -l
-rwxrwxr-x   1 sle      sle            91 Oct 10 17:58 autoexec.bat
-rwxrwxr-x   1 sle      sle           245 Oct 10 17:58 config.sys
drwxrwxr-x  41 sle      sle         16384 Dec 30  1998 windows
```

ce qui donne un contrôle complet à "sle" sur `/c`. Les options énumérées ci-dessus peuvent être rendues permanentes en les ajoutant au fichier `/etc/fstab`:

```
~>grep /c /etc/fstab
/dev/hda1  /c  vfat  uid=500,gid=500,umask=002,exec,dev,suid,rw 1 1
```

Notez que l'umask `002` est courrant dans le schéma des permissions sur les fichiers groupe privé utilisateur. Sur les systèmes de fichiers FAT, cet umask assure que tous les fichiers sont entièrement accessibles par tous les utilisateurs dans le groupe de l'utilisateur spécifié (`gid`).

5.5.9.4. Les systèmes fantômes de fichiers FAT

Avec les systèmes fantômes, on obtient une meilleure granularité de contrôle. Les parties du système de fichiers original peuvent êtres copiées de sorte que le programme puisse fonctionner en tout sécurité avec ces parties copiées pendant que le programme continue à lire directement les parties restantes. Ceci est réalisé au moyen de liens symboliques. Par exemple, considérons un système où un programme nommé AnApp doit pouvoir lire et écrire dans les répertoires `c:\windows` et `c:\AnApp` de même qu'avoir accès en lecture à la totalité du système de fichiers. Sur ce système, le système de fichiers FAT a des permissions par défaut qui ne doivent pas être changées pour des raisons de sécurité ou ne peuvent pas être changées faute d'accès super-utilisateur. Sur ce système un répertoire fantôme pourrait être mis en place de la manière suivante :

```
~>cd /
/>mkdir c_shadow
/>cd c_shadow
/c_shadow>ln -s /c_/* .
/c_shadow>rm windows AnApp
/c_shadow>cp -R /c_/{windows,AnApp} .
/c_shadow>chmod -R 777 windows AnApp
/c_shadow>perl -p -i -e 's|/c$|/c_shadow|g' ~/.wine/config
```

ce qui donne à tout le monde un accès complet en lecture et écriture aux répertoires `windows` et `AnApp` tandis que seul le super-utilisateur a accès en écriture à tous les autres répertoires.

5.5.10. Marques des lecteurs et numéros de séries

Wine peut lire les marques et les numéros de série des lecteurs directement depuis le périphérique. Ceci peut être utile pour de nombreux jeux Win 9x ou programmes d'installations distribués sur CD-ROM qui vérifient la marque du volume.

5.5.10.1. Qu'est-ce qui est pris en charge?

Système de fichiers	Types	Commentaires
Systèmes FAT	disques durs, disquettes	lit les marques et les numéros de séries
ISO9660	cdrom	lit les marques et les numéros de séries (pas encore les CD en mode mixte !)

5.5.10.2. Comment installer?

La lecture de marques et numéros de séries fonctionne automatiquement en spécifiant simplement le lien symbolique correct pour les périphériques (avec deux fois deux-points après les lettres de lecteurs) dans votre répertoire `dosdevices`. Notez, cependant, que si vous faites ça, ce matériel doit exister et doit être accessible par les utilisateurs lançant Wine.

Si vous ne voulez pas lire les marques et les numéros de séries directement depuis le périphérique, vous pouvez créer des fichiers à la racine des lecteurs nommés respectivement `.windows-label` et `.windows-serial`. Ce sont de simples fichiers ASCII que vous pouvez créer avec un éditeur de texte; la marque peut être définie par toute chaîne de caractères que vous souhaitez, le numéro de série doit être exprimé en nombre hexadécimal.

5.5.10.3. Exemples

Voici un exemple simple d'un CD-ROM et d'une disquette:

```
cd ~/.wine/dosdevices

ln -s /mnt/floppy a:
ln -s /dev/fd0 a::

ln -s /mnt/cdrom r:
ln -s /dev/hda1 r::
```

5.5.10.4. A faire / Questions Ouvertes

- La marque du CD-ROM peut être lue seulement si la piste de données du disque est située sur la première piste et si le cdrom est iso9660.
- de l'ECRITURE des marques et numéros de séries
- Qu'en est-il de la lecture des marques de volumes ext2? ...

5.6. Le Registre

Après Win3.x, le registre est devenu une partie fondamentale de Windows. C'est l'endroit où à la fois Windows lui-même et toutes les applications conformes Win95/98/NT/2000/XP/etc. stockent les données de configuration et d'état. Bien que la plupart des administrateurs systèmes sensés (et développeurs Wine) maudissent le registre Windows parce qu'il est vraiment tordu, il faut bien que Wine permette son fonctionnement d'une manière ou d'une autre.

5.6.1. Le registre par défaut

Un registre Windows contient beaucoup de clés par défaut, et certaines d'entre elles sont nécessaires même pour les programmes d'installation afin qu'ils fonctionnent correctement. Les clés que les développeurs Wine ont trouvées nécessaires pour installer les applications sont réparties dans un fichier appelé `wine.inf`. Il est automatiquement installé à votre place si vous utilisez le script `tools/wineinstall` dans les sources de Wine, mais si vous voulez l'installer manuellement, vous pouvez le faire en utilisant l'outil **regedit** qui se trouve dans le répertoire `programs/regedit/` dans les sources de Wine. `wine.inf` est appliqué même si vous envisagez d'utiliser un registre Windows natif, puisque Wine requiert des paramètres de registre spécifiques dans son registre (pour des procédures spéciales de contournement pour certains programmes etc.). Ceci est fait automatiquement par Wine la première fois que vous le lancez.

5.6.2. Utiliser un registre Windows

Si vous pointez Wine vers une installation de Windows existante (en paramétrant les répertoires appropriés dans `~/.wine/config`), Wine est capable de charger les donner de registre depuis celle-ci. Toutefois, Wine ne sauvegardera rien dans le registre Windows réel, mais plutôt dans ses propres fichiers de registre (voir ci-dessous). Bien sûr, si une valeur particulière de registre existe à la fois dans le registre Windows et dans le registre Wine, Wine utilisera celle du registre de Wine. Dans la section [registry] (voir ci-dessous) du fichier config de Wine, il y a un certain nombre de paramètres de configuration spécifiques à l'utilisation par Wine du contenu du registre Windows.

5.6.3. Le Registre

Le contenu du registre initial par défaut à utiliser par les fichiers registre de Wine est dans le fichier `wine.inf`. Il contient les chemins de répertoires, les identifiants de classes, et autres; il doit être installé avant que la plupart des applications `INSTALL.EXE` ou `SETUP.EXE` ne fonctionne.

5.6.4. Architecture du Registre

Le registre Windows est une arborescence élaborée, à tel point que même les programmeurs Windows, pour la plupart d'entre eux, ne connaissent pas en entier l'organisation du registre, avec ces différentes "ruches" et les nombreux liens entre elles; on ne traitera pas cette question à fond car elle dépasse le cadre de ce document. Mais, les principales clés de registre que vous pourriez avoir besoin de connaître pour le moment sont :

HKEY_LOCAL_MACHINE

Cette clé d'admistration fondamentale (sous win9x elle est stockée dans le fichier caché `system.dat`) contient tout ce qui se rapporte à l'installation Windows actuelle.

HKEY_USERS

Cette clé d'administration fondamentale (sous win9x elle est stockée dans le fichier caché `user.dat`) contient les données de configuration pour tous les utilisateurs de l'installation.

HKEY_CLASSES_ROOT

lien vers HKEY_LOCAL_MACHINE\Software\Classes. Elle contient les données décrivant des choses telles que les associations de fichiers, les manipulateurs de documents OLE, et les classes COM.

HKEY_CURRENT_USER

lien vers HKEY_USERS\your_username, c'est-à-dire, votre configuration personnelle.

5.6.5. Fichiers de données du registre Wine

Dans le répertoire utilisateur, il y a un sous-répertoire appelé `.wine`, où Wine essaiera d'enregistrer son registre par défaut. Il enregistre dans quatre fichiers, qui sont :

`system.reg`

Ce fichier contient HKEY_LOCAL_MACHINE.

`user.reg`

Ce fichier contient HKEY_CURRENT_USER.

`userdef.reg`

Ce fichier contient HKEY_USERS\.Default (c'est-à-dire les paramètres de l'utilisateur par défaut).

`wine.userreg`

Wine enregistre HKEY_USERS dans ce fichier (à la fois l'utilisateur courant et l'utilisateur par défaut), mais ne le charge pas depuis celui-ci, sauf si `userdef.reg` est manquant.

Tous ces fichiers sont des fichiers textes lisibles en clair, ainsi, contrairement à Windows, vous pouvez en fait les exploiter par un éditeur de texte ordinaire si vous voulez (assurez-vous que vous n'avez pas lancé Wine quand vous les modifiez, sinon vos modifications seront ignorées).

CORRIGEZ-MOI: La configuration globale n'est actuellement pas implémentée. En plus de ces fichiers, Wine peut aussi facultativement charger, depuis le registre global, les fichiers situés dans le même répertoire que le `wine.conf` global (c'est-à-dire `/usr/local/etc` si vous compilez les sources). Il s'agit de:

`wine.systemreg`

Contient HKEY_LOCAL_MACHINE.

`wine.userreg`

Contient HKEY_USERS.

5.6.6. Administration système

Avec l'architecture de fichiers ci-dessus, il est possible pour un administrateur système de configurer le système de sorte qu'une installation d'un système Wine (et les applications) puisse être partagée par tous les utilisateurs, tout en laissant à tous les utilisateurs leur propre configuration personnalisée. Un administrateur peut, après avoir installé Wine et n'importe quel logiciel d'application Windows à laquelle il autorise l'accès aux utilisateurs, remplacer les fichiers de registres globaux par les fichiers `system.reg` et `user.reg` résultants (qui, nous le supposons, résideront dans `/usr/local/etc`, dans ce cas), ceci grâce à:

```
cd ~/.wine
cp system.reg /usr/local/etc/wine.systemreg
cp user.reg /usr/local/etc/wine.userreg
```

et peut-être même les lier symboliquement au compte de l'administrateur, pour rendre plus simple l'installation des applications dans le système entier, ultérieurement:

```
ln -sf /usr/local/etc/wine.systemreg system.reg
ln -sf /usr/local/etc/wine.userreg user.reg
```

Notez que le script `tools/wineinstall` fait déjà tout cela à votre place, si vous installez les sources de Wine en tant que super-utilisateur (root). Si vous installez alors les applications Windows au cours d'une session en tant que super-utilisateur, tous vos utilisateurs auront automatiquement la possibilité de les utiliser. Alors que la configuration des applications sera prise en charge depuis le registre global, les configurations personnalisées des utilisateurs seront enregistrées dans leurs propres répertoires utilisateurs.

Mais, prenez garde à ce que vous faites avec le compte admistrateur - si effectivement vous copiez ou reliez le registre administrateur vers le registre global, tout utilisateur pourrait avoir la possibilité de lire les préférences de l'administrateur, ce qui pourrait ne pas être bon si des informations sensibles (mots de passe, informations personnelles, etc) y sont stockées. Utilisez uniquement le compte administrateur pour installer des logiciels, et non pas pour du travail quotidien; utilisez un compte utilisateur ordinaire pour cela.

5.6.7. La section [registry]

Maintenant, jetons un coup d'oeil aux options du fichier de configuration de Wine pour l'utilisation du registre.

GlobalRegistryDir

: Facultatif. Paramètre le chemin pour rechercher le registre global.

LoadGlobalRegistryFiles

: Indique s'il faut essayer de charger les fichiers du registre global, s'ils existent.

LoadHomeRegistryFiles

: Indique s'il faut essayer de charger les fichiers du registre utilisateur, (dans le sous-répertoire `.wine` du répertoire utilisateur).

LoadWindowsRegistryFiles

: Indique si Wine tentera de charger les données du registre depuis un registre Windows réel dans une installation MS Windows existante.

WritetoHomeRegistryFiles

: Indique si les données du registre seront écrites dans les fichiers du registre de l'utilisateur. (Actuellement, il n'y a pas d'alternative, donc si vous désactivez ceci, Wine ne pourra pas du tout sauvegarder le registre sur le disque; après avoir quitté Wine, vos modifications seront perdues.)

SaveOnlyUpdatedKeys

: Indique si le registre complet est enregistré dans les fichiers de registre de l'utilisateur, ou seulement les sous-clés que l'utilisateur a réellement modifiées. Vu que le registre de l'utilisateur outrepassera tous les fichiers de registre global et les fichiers de registre Windows, c'est habituellement logique de ne sauvegarder que les sous-clés modifiées par l'utilisateur; de cette manière, les modifications sur le reste des registres global ou Windows affecterait tout de même l'utilisateur.

PeriodicSave

Cette option, si elle est paramétrée à une valeur non nulle, indique que vous voulez que le registre soit sauvegardé sur le disque à l'intervalle donné. S'il n'est pas paramétré, le registre sera enregistré sur le disque uniquement quand le wineserver s'arrêtera.

UseNewFormat

Cette option est obsolète. Wine utilise maintenant toujours le nouveau format; la prise en charge des anciens formats a été enlevée il y a quelque temps.

5.7. Configuration des DLL

5.7.1. Introduction

Si vos programmes ne fonctionnent pas comme prévu, c'est souvent parce que l'une des DLL est défectueuse. Ceci peut en général être résolu en remplaçant certains fichiers DLL intégrés à Wine par des fichiers DLL natifs Windows et vice versa.

Il peut être très utile, pour trouver les DLL chargées en tant que fichiers intégrés [built-in] et celles chargées en tant que fichiers natifs Windows [native], d'utiliser le circuit de débogage loaddll, activé via la variable d'environnement **WINEDEBUG=+loaddll**.

5.7.2. Introduction aux sections DLL

Il y a quelques points que vous devrez connaître avant de configurer les sections DLL dans votre fichier de configuration de Wine.

5.7.2.1. Les paires de DLL Windows

La plupart des DLL windows ont une forme win16 (Windows 3.x) et une forme win32 (Windows 9x/NT). Les combinaisons des versions de DLL win16 et win32 s'appellent "paires de DLL". Voici une liste des paires les plus communes:

Win16	**Win32**	**Native** [a]
KERNEL	KERNEL32	Non!
USER	USER32	Non!
SHELL	SHELL32	Oui
GDI	GDI32	Non!
COMMDLG	COMDLG32	Oui

Win16	**Win32**	**Native** [a]
VER	VERSION	Oui
Remarques : a. Est-il possible d'utiliser des DLL natives avec wine? (Voir section suivante)		

5.7.2.2. Les différents types de DLL

Il y a quelques différents types de DLL que wine peut charger:

native

Les DLL qui sont incluses à Windows. Beaucoup de DLL windows peuvent être chargées dans leur forme native. Parfois, ces versions natives fonctionnent mieux que leur équivalent "non-Microsoft" -- parfois, elles ne fonctionnent pas.

builtin

Forme la plus courante du chargement de DLL. C'est ce que vous utiliserez si la DLL est trop spécifique au système ou sujette à erreurs dans sa forme native (NOYAU par exemple), si vous ne possédez pas la DLL native, ou si vous voulez simplement être indépendant de Microsoft.

so

Bibliothèques ELF natives. Est devenu obsolète, ignoré.

elfdll

DLL windows encapsulées dans ELF. N'est plus utilisé, ignoré.

5.7.3. Les DLL en jeu

Les directives [DllDefaults] et [DllOverrides] du fichier de configuration de Wine provoquent certaines confusions. Les objectifs généraux de la plupart de ces directives sont assez clairs, pourtant - puisqu'on a le choix, Wine devrait-il utiliser ses propres DLL intégrées, ou devrait-il utiliser les fichiers `.DLL` qui se trouvent dans une installation Windows existante? Ce document explique comment cette fonctionnalité marche.

5.7.3.1. Les types de DLL

native

Une DLL "native" est un fichier `.DLL` écrit pour le vrai Microsoft Windows.

builtin

Une DLL "intégrée" (built-in) est une DLL Wine. Celles-ci peuvent être soit une partie de `libwine.so`, soit, plus récemment, dans un fichier `.so` particulier que Wine a la possibilité de charger à la demande.

5.7.3.2. La section [DllDefaults]

DefaultLoadOrder

Spécifie dans quel ordre Wine doit rechercher les types de DLL disponibles, si la DLL en question n'a pas été trouvée dans la section [DllOverrides].

5.7.3.3. La section [DllPairs]

A un moment donné, il y a eu une section appelée [DllPairs] dans le fichier de configuration par défaut, mais celle-ci est devenue obsolèle car les informations sur l'appariement sont maintenant intégrées dans Wine lui-même. (Le but de cette section était simplement d'être capable d'émettre un avertissement si l'utilisateur tentait de jumeler des DLL 16-bits/32-bits codépendantes de différents types). Si elle se trouve toujours dans votre `~/.wine/config` ou `wine.conf`, vous pouvez l'effacer en toute sécurité.

5.7.3.4. La section [DllOverrides]

Cette section spécifie comment vous voulez que les DLL spécifiques soient utilisées, en particulier si vous voulez utiliser des DLL "natives" ou non, au cas où vous en posséderiez certaines d'une configuration Windows réelle. Comme les DLL intégrées ne se mélangent pas encore d'une façon homogène aux DLL natives, certaines dépendances de DLL pourraient s'avérer problématiques, mais des solutions existent sous Wine pour les configurations de beaucoup de DLL répandues. Voir également la Page d'Etat[16] de WWN pour vous rendre compte comment vos DLL favorites sont si bien implémentées sous Wine.

Bien sûr, il est également possible de passer outre à ces paramètres en utilisant explicitement l'option ligne de commandes de Wine `--dll` (voir la page man pour les détails). Quelques indications pour choisir votre configuration optimale (énumérées par paires de DLL 16/32 bits) :

krnl386, kernel32

Leurs versions natives ne fonctionneront jamais, donc n'essayez pas. Laissez à `builtin`.

gdi, gdi32

Interface de Périphérique Graphique (Graphics Device Interface). Aucune tentative n'a été faite pour essayer de faire fonctionner GDI natif. Laissez à `builtin`.

user, user32

Contrôles des normes et de la gestion Windows. Il était possible d'utiliser les versions `native` de Win95 à un moment donné (si toutes les autres DLL qui en dépendaient, telles que comctl32 et comdlg32, fonctionnaient aussi comme `native`). Toutefois, ceci n'est plus possible depuis la séparation des espaces d'adressage, donc laissez à `builtin`.

ntdll

API du noyau NT. Bien que mal documentée, la version `native` de cette DLL ne fonctionnera jamais. Laissez à `builtin`.

w32skrnl

Win32s (pour Win3.x). La version `native` ne fonctionnera certainement jamais. Laissez à `builtin`.

wow32

Win16 prend en charge la bibliothèque pour NT. La version `native` ne fonctionnera certainement jamais. Laissez à `builtin`.

system

Fait partie du noyau Win16. Ne fonctionnera jamais en version `native`. Laissez à `builtin`.

display

Pilote d'affichage. Laissez à `builtin`, sans hésiter !

toolhelp

Sous-programmes d'assistance pour l'utilisation des outils. Rarement une source d'ennuis. Laissez à `builtin`.

ver, version

Versionnage. Ca ne sert presque jamais à rien de le bidouiller.

advapi32

Fonctions de registre et de sécurité. La version native peut marcher ou non, si on l'essaie.

commdlg, comdlg32

Boîtes de dialogue courantes, telles que l'outil pipette de couleur, les boîtes de dialogue de polices, les boîtes de dialogue d'impression, les boîtes de dialogue ouvrir/enregistrer, etc. Essayer la version `native` est sans danger.

commctrl, comctl32

Commandes ordinaires. Ce sont les barres d'outils, les barres d'état, les contrôles de listes, les logiciels intégrés. Essayer la version `native` est sans danger.

shell, shell32

Interface Shell (bureau, système de fichiers, etc). Etant un des éléments de Windows les plus dépourvus de commentaires, la chance peut vous sourire avec la version `native`, si toutefois vous en avez besoin.

winsock, wsock32

Sockets Windows. La version `native` ne fonctionnera pas sous Wine, donc laissez à `builtin`.

icmp

Sous-programmes ICMP pour wsock32. Comme avec wsock32, laissez à `builtin`.

mpr

La version `native` pourrait ne pas fonctionner du fait de problèmes liés à la conversion du format des instructions (16bits/32bits) [thunking]. Laissez à `builtin`.

lzexpand, lz32

Décompression Lempel-Ziv. La version `builtin` de Wine est censée bien fonctionner.

winaspi, wnaspi32

Interface avancée des périphériques SCSI. La version `native` ne fonctionnera certainement jamais. Laissez à `builtin`.

crtdll

Bibliothèque du moteur d'exécution C. Avec elle, la version `native` fonctionnera largement mieux que celle de Wine.

winspool.drv

Gestionnaire des fils d'attente d'impression. La version `native` risque fort de ne pas marcher.

ddraw

DirectDraw/Direct3D. Puisque Wine n'implémente pas le HAL DirectX, la version `native` ne fonctionnera pas en l'état actuel des choses.

dinput

DirectInput. Si vous exécutez cette version `native`, ça marchera peut-être

dsound

DirectSound. Il pourrait être possible de faire tourner cette version , mais n'y comptez pas trop.

dplay/dplayx

DirectPlay. La version `native` est censée mieux fonctionner ici, si tant est qu'elle fonctionne.

mmsystem, winmm

Système multimédia. Il y a peu de chance que la version `native` fonctionne. Laissez à `builtin`.

msacm, msacm32

Gestionnaire de compression audio. La version `builtin` fonctionne mieux, si vous réglez msacm.drv de la même façon.

msvideo, msvfw32

Video pour Windows. Il est sans danger (et recommandé) d'essayer la version `native`.

mcicda.drv

Pilote MCI pour CD Audio.

mciseq.drv

Pilote MCI pour séquenceur MIDI (lecture `.MID`).

mciwave.drv

Pilote MCI pour la lecture des fichiers audio Wave (lecture `.WAV`).

mciavi.drv

Pilote MCI pour la lecture de l'AVI (lecture video `.AVI`). Le mieux est d'utiliser la version `native`.

mcianim.drv

Pilote MCI pour les animations.

msacm.drv

Gestionnaire de compression audio. Réglez de la même manière que msacm32.

midimap.drv

Adaptateur MIDI.

wprocs

Pseudo-DLL utilisée par Wine pour les conversion des formats d'instructions (16bits/32bits) [thunking]. Il n'en existe pas de version `native`.

5.7.4. Les DLL système

L'équipe Wine a établi qu'il est nécessaire de créer de faux fichiers DLL pour ruser avec beaucoup de programmes qui vérifient l'existence de fichiers dans le but de déterminer si une fonction particulière (telle que Winsock et ses TCP/IP réseaux) est disponible. Si cela vous pose un problème, vous pouvez créer des fichiers vides dans le répertoire configuré `c:\windows\system` pour faire croire au programme qu'elle y est, et la DLL Wine intégrée sera chargée quand le programme le demandera réellement. (Malheureusement, `tools/wineinstall` ne crée pas de tels fichiers vides lui-même.)

Quelquefois, les applications essaient également d'examiner les ressources de la version qui se trouvent dans les fichiers physiques (par exemple, pour déterminer la version de DirectX). Les fichiers vides ne

conviendront pas dans ce cas, il est nécessaire d'installer plutôt les fichiers avec les ressources complètes sur la version. On s'occupe de ce problème actuellement. En attendant, vous pourriez encore avoir besoin de vous raccrocher à certains fichiers DLL réels afin de duper ces applications.

Et, il y a bien sûr les DLL que wine n'implémente pas vraiment correctement (ou pas du tout) actuellement. Si vous ne possédez pas de Windows réel, pour pouvoir y récupérer les DLL nécessaires, vous pouvez toujours en obtenir certaines depuis les sites d'archives des DLL Windows qu'on peut trouver avec un moteur de recherche Internet. Merci de vous assurer que vous respectez toutes les licences sur les DLL que vous allez chercher (certaines sont redistribuables, certaines ne le sont pas).

5.7.5. DLL manquantes

Au cas où Wine se plaindrait d'une DLL manquante, vous devez vérifier si ce fichier est une DLL disponible publiquement ou une DLL personnalisée appartenant à votre programme (en recherchant son nom sur Internet). Si vous réussissez à vous munir de la DLL, il est conseillé de vous assurer que Wine est capable de la trouver et de la charger. Les DLL se chargent habituellement suivant le mécanisme de la fonction SearchPath(). Cette fonction cherche les répertoires dans l'ordre suivant :

1. Le répertoire à partir duquel le programme a été démarré
2. Le répertoire courant
3. Le répertoire système Windows
4. Le répertoire Windows
5. Les répertoires spécifiés par la variable PATH.

En bref : mettez la DLL requise soit à l'intérieur de votre répertoire programme (il se pourrait que ceci ne soit pas vraiment propre), soit dans le répertoire système Windows. Retrouvez simplement ce répertoire en jetant un coup d'oeil à la variable "System" du fichier de configuration de Wine (qui indique l'emplacement du répertoire système Windows) et l'information lecteur associée. Notez que, très probablement, il ne faut pas utiliser les DLL natives basées sur NT, puisque la prise en charge par Wine de l'API NT est un peu plus faible que la prise en charge de l'API Win9x (conduisant ainsi à une compatibilité pire avec les DLL NT qu'avec une installation non-windows!), donc mieux vaut utiliser plutôt les DLL Win9x natives ou pas du tout de DLL natives.

5.7.6. Aller chercher les DLL natives sur un CD Windows

L'utilitaire Linux **cabextract** peut être utilisé pour extraire les fichiers .dll Windows natifs des fichiers .cab qui sont récupérables sur beaucoup de CD d'installation de Windows

5.8. Configurer les pilotes graphiques (x11drv, ttydrv etc.)

Actuellement, Wine prend en charge plusieurs sous-systèmes d'affichage différents (graphiques / textuels) aujourd'hui disponibles sur divers systèmes d'exploitation. Pour chacun d'entre eux, Wine implémente son propre pilote d'interfaçage. Cette section explique comment sélectionner l'un de ces pilotes et comment aller plus loin dans la configuration des pilotes respectifs. Une fois que vous en avez fini avec cela, vous pouvez considérer votre installation de Wine terminée.

Les pilotes d'affichage actuellement implémentés sous Wine sont : x11drv, qui est utilisé pour interfacer les graphiques X11 (celui que vous avez de fortes chances de vouloir installer) et ttydrv (généralement utilisé pour les applications en mode texte qui n'ont pas vraiment besoin de sortie). Une fois que vous avez décidé quel pilote d'affichage utiliser, vous le choisirez avec l'option `GraphicsDriver` dans la section [wine] de `~/.wine/config`.

5.8.1. Configurer le pilote graphique x11drv

5.8.1.1. Modes de fonctionnement de x11drv

Le pilote x11drv est constitué de deux parties distinctes au niveau conceptuel, le pilote graphique (la partie GDI), et le pilote de fenêtrage (la partie USER). Par contre, chacune d'entre elles est liée à l'intérieur du module `libx11drv.so` (que vous chargez avec l'option `GraphicsDriver`). Sous Wine, en fonctionnant avec X11, le pilote graphique doit dessiner dans des objets dessinables (à l'intérieur des fenêtres) fournis par le pilote de fenêtrage. Ceci diffère un peu du modèle Windows, où le système de fenêtrage crée et configure les contextes des périphériques contrôlés par le pilote graphique, et les programmes sont autorisés à se mettre dans cette relation n'importe où ils veulent. Ainsi, afin de fournir un compromis raisonnable entre la compatibilité et la convivialité, le x11drv possède trois modes de fonctionnement différents.

Mode géré : option Managed

> Mode par défaut. Spécifié en utilisant l'option `Managed` du fichier de configuration de Wine (voir ci-dessous). Les fenêtres ordinaires de haut niveau avec des bords épais, des barres de titres, et des menus systèmes seront gérées par votre gestionnaire de fenêtres. Il permet aux programmes de mieux s'intégrer au reste de votre bureau, mais peut ne pas toujours fonctionner parfaitement (cependant, une refonte de ce mode de fonctionnement est actuellement en cours, afin de le rendre plus robuste et moins inégal, et il est prévu qu'elle soit finie avant la version 1.0 de Wine).

Mode non-géré / normal

> Gestionnaire de fenêtre indépendant (tout gestionnaire de fenêtre en cours de fonctionnement est complétement ignoré). Les décorations de fenêtres (barres de titre, bordures, etc) sont dessinées par Wine pour respecter l'aspect et la convivialité d'un Windows réel. Ceci est compatible avec les programmes qui dépendent de leur capacité à calculer les dimensions exactes de n'importe laquelle de ces décorations, ou qui veulent dessiner leurs propres décorations. Le mode "non-géré" est

uniquement utilisé si le mode "Managed" ainsi que le mode "Desktop" sont désactivés (réglé à "disabled").

Mode Bureau-dans-une-Boîte [Desktop-in-a-Box] : option Desktop

Spécifié en utilisant l'option `Desktop` du fichier de configuration de Wine (voir ci-dessous). (en ajoutant une géométrie, par exemple `800x600`, pour un bureau dimensionné ainsi, ou même `800x600+0+0` pour positionner automatiquement dans le coin en haut à gauche de l'écran). Il s'agit du mode le plus compatible avec le modèle Windows. Toutes les fenêtres des programmes seront simplement des fenêtre dessinées par Wine à l'intérieur de la fenêtre bureau fourni par Wine (qui sera elle-même gérée par votre gestionnaire de fenêtres), et les programmes Windows peuvent évoluer librement à l'intérieur de cet espace de travail virtuel et croire qu'il est tout entier à eux, sans perturber vos autres applications X. Note : Actuellement, il y a une fenêtre bureau pour chaque programme; on arrangera cela ultérieurement.

5.8.1.2. La section [x11drv]

Managed

Wine peut laisser le gestionnaire de fenêtres gérer les fenêtres. Cette option spécifie si c'est ce que vous voulez par défaut.

Desktop

Crée une fenêtre principale de bureau à une dimension donnée afin d'afficher tous les programmes Windows à l'intérieur. L'argument dimension peut par exemple être "800x600".

DXGrab

Si vous n'utilisez pas le DGA, vous souhaiterez peut-être avoir un autre moyen pour convaincre le curseur de souris de rester à l'intérieur de la fenêtre de jeux. C'est le but de cette option. Bien sûr, comme avec le DGA, si Wine plante, vous êtes mal (bien que pas autant que dans le cas de la DGA, puisque vous pouvez encore utiliser le clavier pour sortir du X).

UseDGA

Spécifie si vous voulez que DirectDraw utilise l'*Architecture Graphique Directe* (DGA = Direct Graphics Architecture) de XFree86, qui peut prendre le contrôle de tout l'écran et exécuter le jeu en mode plein écran à vitesse maximum. (Avec DGA1 (XFree86 3.x), il vous reste toujours à configurer le serveur X pour la profondeur d'image (nombre de bits par pixel) requise pour le jeu dans un premier temps, mais avec DGA2 (XFree86 4.x), le changement de profondeur en cours d'exécution peut être possible, selon les capacités de votre pilote.) Mais il faut savoir que si Wine plante en mode DGA, il se peut que vous ne puissiez pas reprendre le contrôle de votre ordinateur sans redémarrer. DGA exige normalement soit les droits de super-utilisateur (root) soit un accès en lecture/écriture à `/dev/mem`.

DesktopDoubleBuffered

S'applique seulement si vous utilisez l'option ligne de commande *`--desktop`* pour entrer dans une fenêtre de bureau. Indique si la création de la fenêtre de bureau doit être faite avec une image en double zone tampon, dont la plupart des jeux OpenGL ont besoin pour s'exécuter correctement.

AllocSystemColors

S’applique uniquement si vous avez un affichage basé sur une palette, c’est-à-dire si votre serveur X est réglé à une profondeur d’image de 8 bpp et si vous n’avez pas démandé une table de couleurs privée. Elle spécifie le nombre maximum de cellules de la table de couleurs partagée (entrées de le palette) que Wine devrait occuper. Plus cette valeur est haute, moins les couleurs seront disponibles pour les autres programmes.

PrivateColorMap

S’applique seulement si vous avez un affichage basé sur une palette, c’est-à-dire si votre serveur X est réglé à une profondeur d’image de 8 bpp. Elle spécifie que vous ne voulez pas utiliser la table de couleurs partagée, mais une table de couleurs privée, où les 256 couleurs sont toutes disponibles. L’inconvénient est que la table de couleurs privée de Wine n’est visible que quand le pointeur de souris est à l’intérieur de la fenêtre Wine, ainsi les flashs psychédéliques et les couleurs branchées deviendront habituels si vous utilisez beaucoup la souris.

Synchronous

A utiliser pour les opérations X11 de débogage. Si Wine plante avec une erreur X11, vous devez activer le mode Synchronous pour rendre indisponibles les requêtes en mémoire cache afin de s’assurer que l’erreur X11 se produit directement après que l’appel à X11 correspondant apparaîsse dans le journal de bord (fichier de log). Ralentira la sortie X11 !

ScreenDepth

S’applique uniquement aux affichages à profondeurs d’images multiples. Spécifie laquelle des profondeurs d’images disponibles Wine doit utiliser (et en informer les applications Windows).

Display

Spécifie quel affichage X11 utiliser, et si spécifié, remplacera la variable d’environnement DISPLAY

PerfectGraphics

Cette option détermine seulement si les sous-programmes rapides X11 ou les sous-programmes Wine exactes seront utilisés pour certains codes ROP dans les opérations de calcul de rendu [blit]. La plupart des utilisateurs ne remarqueront aucune différence.

5.8.2. Configurer le pilote graphique ttydrv

Actuellement, le ttydrv n’a pas d’option de configuration spéciale à paramétrer dans le fichier de configuration.

5.9. Indiquer le numéro de version Windows / DOS

Le numéro de la version Windows / DOS, qu’un programme peut récupérer, via la fonction GetVersion()

de Windows par exemple, est primordiale : si, pour quelque raison que ce soit, votre installation de Wine n'indique pas au programme le numéro de version adéquat, ce dernier peut faire de très mauvaises interprétations et planter (dans le pire des cas sans aucun message d'erreur !). Heureusement Wine utilise des algorithmes, plus ou moins intelligents, pour la détection de version de Windows; ils tenteront de retrouver la version demandée par un programme et la lui communiqueront. C'est pourquoi il est recommandé de *ne pas* configurer à la légère cette version, ou d'en "forcer" la valeur : ceci s'avèrerait assez préjudiciable à un bon fonctionnement. En d'autres termes : ne fixez explicitement le numéro de version de Windows que si la détection de version par Wine n'a pas pu aboutir et que le programme plante.

5.9.1. Renseigner la valeur du numéro de version Windows / DOS qui doit être retournée par Wine

Les numéros de version peuvent être configurées dans la partie [Version] du fichier de configuration.

"Windows" = "<version sous forme de chaîne de caractères>"

Par défaut : aucune; la valeur est donnée par le mécanisme de détection semi-intelligent basé sur l'inspection des DLL. Elle est utilisée pour spécifier quelle version de Windows transmettre aux programmes (forcer cette valeur outrepasse le mécanisme standard de détection !). Les valeurs valides peuvent être "win31", "win95", "win98", "win2k", "winxp" par exemple. Option également possible comme valeur par défaut pour une application (utilisation recommandée/préférée).

"DOS"="<version sous forme de chaîne de caractères>"

Cette option est utilisée pour spécifier la valeur de la version de DOS qui doit être transmise aux programmes. Elle n'est utilisée que si Wine émule une version "win31" de windows ! Les valeurs habituelles pour DOS sont 6.22, 6.20, 6.00, 5.00, 4.00, 3.30, 3.10. Option également possible comme valeur par défaut pour une application (utilisation recommandée/préférée).

5.10. Gestion des polices

5.10.1. Les polices

Note : L'utilitaire **fnt2bdf** est fourni avec Wine. Il se trouve dans le répertoire `tools`. Les liens vers les autres outils mentionnés dans ce document sont répertoriés sur le site principal de Wine : http://www.winehq.org/development/

5.10.1.1. Convertir les polices de Windows

Si vous avez accès à une installation Windows, vous pouvez utiliser le programme **fnt2bdf**(dans le répertoire `tools`) pour convertir les polices bitmap (`VGASYS.FON`, `SSERIFE.FON`, et `SERIFE.FON`) dans le format reconnu par l'interface graphique X.

1. Extraire les polices bitmap avec **fnt2bdf**.
2. Convertir les fichiers `.bdf` sortis à l'étape 1 en fichiers `.pcf` avec **bdftopcf**.
3. Copier les fichiers `.pcf` dans le répertoire du serveur de polices qui est habituellement `/usr/lib/X11/fonts/misc` (vous aurez probablement besoin des privilèges du super-utilisateur). Si vous désirez créer un nouveau répertoire de polices, il est nécessaire de l'ajouter au chemin d'accès des polices.
4. Exécuter **mkfontdir** dans le répertoire où les polices ont été copiées. Si l'interface graphique est déjà lancée, exécuter **xset fp rehash** pour avertir le serveur graphique X de la présence de nouvelles polices. Il se peut qu'il faille aussi ou au lieu de cela redémarrer le serveur de polices (exécuter par exemple **/etc/init.d/xfs restart** sur une Red Hat 7.1)
5. Editer le fichier `~/.wine/config` pour enlever les alias des polices que vous venez d'installer.

Wine peut fonctionner sans ces polices mais l'aspect de l'interface peut alors être assez différent. Par ailleurs certaines applications tentent de charger leurs propres polices au vol (WinWord 6.0) et comme Wine n'implémente pas encore cette fonction, un message comme celui-ci s'affiche :

```
STUB: AddFontResource( FONTE.FON )
```

Vous pouvez également convertir ce fichier. Attention, un fichier `.FON` peut ne pas contenir de polices bitmap et dans ce cas **fnt2bdf** génèrera une erreur. Ensuite, bien que le message ci-dessus ne disparaîsse pas, Wine va corriger le problème en utilisant la police que vous avez extraite de `FONTE.FON`. **fnt2bdf** fonctionnera seulement avec les polices de Windows 3.1. Il ne fonctionnera pas avec des polices TrueType.

Que faire pour les polices TrueType ? Il existe plusieurs outils commerciaux permettant de les convertir au format Type1 mais la qualité des polices obtenues est loin d'être éblouissante. L'autre solution est d'utiliser un serveur de polices qui gère le TrueType (Caldera en possède un, et il existe aussi **xfstt** qui est une solution libre, que vous pourrez trouver dans `Linux/X11/fonts` sur le site de Sun ou sur un miroir; si vous utiliser FreeBSD vous pouvez utiliser le port dans `/usr/ports/x11-servers/Xfstt`. Enfin il existe **xfsft** qui utilise la bibliothèque TrueType : voir la description de freetype).

Cependant des rumeurs circulent sur une gestion native du TrueType via FreeType dans le futur (confidences, confidences... :-)

5.10.1.2. Ajouter des alias pour les polices dans `~/.wine/config`

De nombreuses applications supposent que les polices de la version originale de Windows 3.1 sont toujours présentes. Par défaut, Wine crée un certain nombre d'alias qui les associe aux polices de l'interface graphique X existantes :

Polices Windows	**...associées à...**	**polices de l'interface graphique X**
"MS Sans Serif"	->	"-adobe-helvetica-"
"MS Serif"	->	"-bitstream-charter-"
"Times New Roman"	->	"-adobe-times-"
"Arial"	->	"-adobe-helvetica-"

Il n'existe pas d'alias pour la police "System". De plus, aucun alias n'est créé pour les polices que les applications installent lors de l'exécution. Il est recommandé de convertir les polices manquantes (voir ci-dessus) pour régler ce problème. Si cela s'avère impossible, comme pour les polices TrueType, vous pouvez forcer l'association de cette police à une police ressemblante de l'interface graphique X en ajoutant un alias dans la partie [fonts]. Veillez à bien utiliser une police de l'interface graphique X qui soit présente (avec **xfontsel**).

```
AliasN = [Police Windows], [police X] <, flag optionnel "mask X font" >
```

Exemple:

```
Alias0 = System, --international-, subst
Alias1 = ...
...
```

Précisions :

- Il ne doit pas y avoir de trou dans la suite `{0, ...,N}` car sinon tous les alias qui suivent le premier trou ne seraient pas lus.
- Habituellement les programmes de conversion de polices transforment les noms de polices de l'interface graphique X en noms de polices lisibles par les programmes Windows de la manière suivante :

Police de l'interface graphique X	**...sera transformée en...**	**Nom de l'extraction**
--international-...	->	"International"
-adobe-helvetica-...	->	"Helvetica"
-adobe-utopia-...	->	"Utopia"
-misc-fixed-...	->	"Fixed"
-...	->	

Police de l'interface graphique X	**...sera transformée en...**	**Nom de l'extraction**
-sony-fixed-...	->	"Sony Fixed"
-...	->	

Etant donné que `-misc-fixed-` et `-sony-fixed-` sont différentes, Wine a modifié le nom de l'extraction de la deuxième pour être sûr que les programmes Windows puissent les distinguer car seul le nom de l'extraction apparaît dans les boîtes de dialogue de sélection des polices.

- L'alias avec l'option de masque ["masking"] remplace le nom original de l'extraction de telle façon que l'on obtient la correspondance suivante dans l'exemple :

Police de l'interface graphique X	**...est renommée en...**	**Nom de l'extraction**
--international-...	->	"System"

Les alias "sans masque" ["nonmasking"] sont transparents pour l'utilisateur et ils ne remplacent pas les noms d'extractions.

Wine annule un alias lorsque la police de l'interface graphique X native est disponible.

- Si vous n'avez pas accès aux polices mentionnées dans le premier paragraphe, essayez de remplacer la police "System" par un alias sans masque. L'application **xfontsel** vous permettra de voir les polices disponibles pour l'interface graphique X.

  ```
  Alias.. = System, ...police, en gras,sans serif
  ```

D'autre part certaines applications Windows font appel à des polices sans spécifier le type de caractères de la police. La liste des polices débute avec Arial dans la plupart des logiciels Windows, cependant la liste des polices de l'interface graphique X débute avec la police de la première ligne du fichier `fonts.dir` quelle qu'elle soit. C'est pourquoi Wine utilise l'entrée suivante pour déterminer quelle police est à utiliser par défaut.

Exemple:

```
Default = -adobe-times-
```

Précisions :

Il est préférable de spécifier, comme police par défaut, une famille de polices modulable (incluant le gras et l'italique) car le programme d'association des polices passe en revue toutes les police disponibles jusqu'à ce que la taille, ainsi que les autres attributs demandés, correspondent parfaitement ou que la fin

de la liste soit atteinte. Généralement les installations de l'interface graphique X possèdent des polices modulables dans les répertoires `../fonts/Type1` et `../fonts/Speedo`.

5.10.1.3. Gérer les informations en cache sur les fontes

Wine stocke des informations détaillées à propos des polices disponibles dans le fichier `~/.wine/cachedmetrics.[display]`. Vous pouvez copier ce fichier ailleurs et ajouter le ligne suivante à la partie [fonts] `~/.wine/config`:

```
FontMetrics = <fichier contenant les informations>
```

Si Wine détecte une modification de la configuration des polices de l'interface graphique X, il regénèrera complètement les informations sur les polices et il écrasera ensuite le fichier `~/.wine/cachedmetrics.[display]` pour le remplacer par le fichier contenant les nouvelles informations. Ce processus peut prendre un certain temps.

5.10.1.4. Polices trop petites ou trop grandes

Des programmes Windows peuvent demander à Wine de produire une police avec une taille indiquée en points. Mais la proportion point-pixel dépend de la taille réelle de votre écran (15 pouces, 17 pouces, etc...). L'interface graphique X essaie de fournir une estimation de cette taille mais elle est parfois assez différente de la taille véritable. Vous pouvez changer cette proportion en ajoutant la ligne suivante dans la partie [fonts] :

```
Resolution = <valeur entière>
```

Généralement, les valeurs élevées donnent des polices plus grandes. Faites le test avec des valeurs comprises entre 60 et 120. 96 est un bon point de départ.

5.10.1.5. Message "FONT_Init: failed to load ..." au démarrage

La cause la plus probable vient du fichier `fonts.dir` qui est corrompu, dans votre répertoire de polices. Vous devez relancer **mkfontdir** pour regénérer le fichier. Lisez le manuel de ce programme pour plus de détails. Si vous ne pouvez pas exécuter **mkfontdir** sur votre machine parce que vous n'êtes pas super-utilisateur, utilisez **xset -fp xxx** pour enlever le chemin d'accès corrompu jusqu'aux polices.

5.10.2. Configurer un serveur de polices TrueType

Suivez les instructions ci-dessous pour configurer un serveur de polices TrueType sur votre machine.

1. Récupérer l'archive des sources de freetype (`freetype-X.Y.tar.gz` ?).
2. Lire la documentation, décompresser, configurer et installer
3. Tester la bibliothèque, par exemple **ftview 20 /dosc/win95/fonts/times**
4. Récupérer `xfsft-beta1e.linux-i586`
5. L'installer et le configurer pour qu'il se lance au démarrage, dans un script rc par exemple. Recourir au manuel de **xfs**.
6. Suivre les conseils donnés par <`williamc@dai.ed.ac.uk`>
7. J'ai récupéré **xfsft** sur http://www.dcs.ed.ac.uk/home/jec/progindex.html. Il tourne en tâche de fond sur ma machine. Voici la configuration pour mon fichier `/usr/X11R6/lib/X11/fs/config`:

```
clone-self = on
use-syslog = off
catalogue = /c/windows/fonts
error-file = /usr/X11R6/lib/X11/fs/fs-errors
default-point-size = 120
default-resolutions = 75,75,100,100
```

`/c/windows/fonts` contient, comme l'indique son nom, les polices du Windows 95 présentes sur ma partition *C:*; sur Windows 3.1, il doit s'agir de `/mnt/dosC/windows/system`.

Dans le fichier `/c/windows/fonts/fonts.scale` j'ai ce qui suit :

```
14
arial.ttf -monotype-arial-medium-r-normal--0-0-0-0-p-0-iso8859-1
arialbd.ttf -monotype-arial-bold-r-normal--0-0-0-0-p-0-iso8859-1
arialbi.ttf -monotype-arial-bold-o-normal--0-0-0-0-p-0-iso8859-1
ariali.ttf -monotype-arial-medium-o-normal--0-0-0-0-p-0-iso8859-1
cour.ttf -monotype-courier-medium-r-normal--0-0-0-0-p-0-iso8859-1
courbd.ttf -monotype-courier-bold-r-normal--0-0-0-0-p-0-iso8859-1
courbi.ttf -monotype-courier-bold-o-normal--0-0-0-0-p-0-iso8859-1
couri.ttf -monotype-courier-medium-o-normal--0-0-0-0-p-0-iso8859-1
times.ttf -monotype-times-medium-r-normal--0-0-0-0-p-0-iso8859-1
timesbd.ttf -monotype-times-bold-r-normal--0-0-0-0-p-0-iso8859-1
timesbi.ttf -monotype-times-bold-i-normal--0-0-0-0-p-0-iso8859-1
timesi.ttf -monotype-times-medium-i-normal--0-0-0-0-p-0-iso8859-1
symbol.ttf -monotype-symbol-medium-r-normal--0-0-0-0-p-0-microsoft-symbol
wingding.ttf -microsoft-wingdings-medium-r-normal--0-0-0-0-p-0-microsoft-symbol
```

Dans le fichier `/c/windows/fonts/fonts.dir` j'ai exactement la même chose.

Dans le fichier `/usr/X11R6/lib/X11/XF86Config` j'ai

```
 FontPath "tcp/localhost:7100"
```

avant les autres lignes `FontPath`. C'est tout ! Un effet secondaire intéressant bien sûr : toutes les pages internet qui spécifient Arial comme police sont affichées en Arial dans Netscape...

8. Eteindre le serveur graphique X et le redémarrer (déboguer les erreurs commises durant la configuration).
9. Tester sur un exemple **xlsfont | grep arial**

5.11. Imprimer avec Wine

Imprimer des documents avec Wine

5.11.1. Imprimer

On peut imprimer avec Wine en utilisant le pilote PostScript intégré (+ ghostscript afin de générer une sortie pour les imprimantes qui ne gèrent pas le PostScript).

Notez que, pour l'instant, certaines imprimantes "WinPrinters" [NdT : uniquement compatibles avec Windows], (imprimantes bon marché et peu évoluées, sollicitant le processeur pour contrôler directement les têtes) ne fonctionneront pas avec leurs pilotes d'impression pour Windows. A l'heure actuelle, nous ne savons pas si elles seront un jour prises en compte.

5.11.1.1. Le pilote PostScript intégré à Wine

Il permet d'imprimer dans un fichier PostScript via le pilote intégré de Wine. Reportez-vous aux instructions ci-dessous pour l'installation. Le fichier contenant le code du pilote postScript se trouve dans le répertoire `dlls/wineps/`.

Le pilote se comporte comme s'il s'agissait d'un fichier DRV [NdT : Fichier contenant le code d'un pilote spécifique à un périphérique.] nommé `wineps.drv` qui est actuellment intégré à Wine. Bien que le pilote émule un pilote 16 bits, il fonctionnera à la fois avec des applications 16 bits et 32 bits, tout comme les pilotes win9x.

5.11.1.2. La gestion des files d'attentes [Spooling]

La gestion des files d'attentes est relativement primaire. La partie [spooler] du fichier de configuration de wine associe un port (par exemple LPT1:) à un fichier ou une commande grâce à une redirection [pipe]. Voir l'exemple ci-dessous.

```
"LPT1:" = "foo.ps"
```

```
"LPT2:" = "|lpr"
```

associe LPT1: à un fichier `foo.ps` et LPT2: à la commande **lpr**. Si une tâche d'impression est envoyée sur un port ne se trouvant pas dans la liste, un fichier portant le nom du port est créé; par exemple pour le port LPT3:, un fichier nommé LPT3: serait alors créé.

Il existe à présent des files d'attentes virtuelles appelées LPR:nomImprimante, qui envoient les données à la commande **lpr -Pprintername**. Il n'est pas nécessaire de les spécifier dans le fichier de configuration, elles sont automatiquement gérées par `dlls/gdi/printdrv.c`.

5.11.2. Le pilote PostScript de Wine

Ce pilote permet de générer des fichier PostScript sans recourir à un pilote d'impression externe. Dans ce cas, Wine utilise les filtres PostScript d'impression fournis par le système, qui utilisent quasiment tous GhostScript si nécessaire. Ils doivent être configurés lors de l'installation du système ou par votre administrateur système.

5.11.2.1. Installation

5.11.2.1.1. Installer les imprimantes CUPS

Avec CUPS, il n'est pas nécessaire de configurer un .ini ou les entrées du registre, tout est détecté automatiquement.

5.11.2.1.2. Installer les imprimantes LPR basées sur le fichier /etc/printcap

Si votre machine n'utilise pas encore CUPS, elle utilise probablement un systéme LPRng ou assimilable à LPR dont la configuration est basée sur le fichier `/etc/printcap`.

Si c'est le cas, les imprimantes inscrites dans le fichier `/etc/printcap` sont contrôlées pour vérifier si elles sont compatibles PostScript et pour la plupart automatiquement configurées.

Etant donné que Wine ne peut pas détecter le type de l'imprimante, il faut indiquer un fichier PPD dans la partie [ppd] du fichier `~/.wine/config`. Utilisez le raccourci et procédez comme ci-dessous :

```
[ppd]
"ps1" = "/usr/lib/wine/ps1.ppd"
```

Ou bien spécifiez un fichier PPD générique qui est adapté pour toutes les autres imprimantes. Un fichier PPD générique se trouve dans le fichier `documentation/samples/generic.ppd`.

5.11.2.1.3. Installer d'autres imprimantes

Cette section n'est pas nécessaire si l'une des deux sections ci-dessus convient, elle est utile seulement si vous possédez une imprimante spéciale. Indiquez

```
Wine PostScript Driver=WINEPS,LPT1:
```

dans la partie [devices] et

```
Wine PostScript Driver=WINEPS,LPT1:,15,45
```

dans la partie [PrinterPorts] du fichier `win.ini`. Pour configurer l'imprimante comme imprimante par défaut, ajoutez aussi

```
device = Wine PostScript Driver,WINEPS,LPT1:
```

dans la partie [windows] du fichier `win.ini`.

Vous devrez aussi ajouter des informations dans le registre. Le moyen le plus simple pour le faire est de personnaliser le contenu du pilote PostScript du fichier `wine.inf` (voir ci-dessous) et d'utiliser le programme **programs/regedit/regedit** de Winelib. Par exemple, si vous avez installé les sources de Wine dans `/usr/src/wine`, vous pouvez utiliser la suite de commandes suivante :

- **`#vi /usr/share/wine/wine.inf`**
- Editez la copie du fichier `wine.inf` selon les besoins de vos imprimantes PostScript. Vous devez spécifier au minimum un fichier PPD par imprimante.
- **`$wineprefixcreate`**

5.11.2.1.4. Configuration nécessaire pour tous les types d'imprimantes

Vous n'aurez plus besoin des fichiers Adobe Font Metric (AFM) (PostScript type 1) pour les polices que vous souhaitez utiliser. Wine l'intègre désormais.

Vous aurez besoin d'un fichier PPD pour votre imprimante. Celui-ci décrit certaines caractéristiques de votre imprimante telles que les polices installées, comment sélectionner l'alimentation papier manuelle etc. Plusieurs fichiers PPD sont proposés sur le site d'Adobe, jetez un oeil sur

ftp://ftp.adobe.com/pub/adobe/printerdrivers/win/all/ (ftp://ftp.adobe.com/pub/adobe/printerdrivers/win/all/). Voir ci-dessous pour plus d'informations sur la configuration du pilote afin d'utiliser ce fichier.

Pour permettre l'impression en couleur, l'option `*ColorDevice` doit être mise à `true` dans le PPD, sinon l'imprimante imprimera en niveaux de gris.

Pensez à mettre l'option `printer=on` dans la partie [wine] du fichier de configuration de Wine, ceci permet d'imprimer à partir de pilotes externes et n'a aucune répercussion sur le pilote PostScript intégré.

Si vous êtes chanceux, vous devez à présent pouvoir générer des fichiers PS à partir de Wine !

J'ai effectué le test avec notepad/write, Winword6 et Origin4.0 ainsi que des applications 32 bits comme win98 wordpad, winWord97, PowerPoint 2000 avec plus ou moins de succès; vous devriez obtenir un fichier de sortie mais il se peut qu'il ne se trouve pas à l'endroit demandé.

5.12. Prise en compte du SCSI

Cette partie décrit la configuration de l'interface ASPI de Windows. ASPI constitue un lien direct entre les périphériques SCSI et les programmes windows. ASPI transmet simplement au bus SCSI les commandes SCSI que les programmes lui envoient.

Si vous indiquez le mauvais périphérique dans votre fichier de configuration, vous pouvez envoyer de mauvaises commandes au mauvais périphérique : par exemple formatter votre disque dur (en supposant que le périphérique vous y autorise; si vous êtes super-utilisateur, tous les paris sont annulés).

Alors veuillez vous asurer que *tous* les périphériques SCSI qui ne sont pas utilisés par le programme possèdent les permissions les plus restreintes !

5.12.1. Configuration requise pour Windows

1. Le logiciel doit utiliser des pilotes compatibles "Adaptec" (ASPI). Avec du matériel Mustek, vous pouvez au moins choisir soit la carte intégrée soit les pilotes compatibles "Adaptec" (ASPI). Ça ne marchera pas autrement. Les logiciels accédant au scanner via un pilote DOS ASPI (par exemple ASPI2DOS) sont également supportés.
2. Vous aurez probablement besoin d'une véritable installation du logiciel sous windows pour configurer correctement les identifiants de LUN/SCSI. Je ne suis pas tout à fait sûr de ça.

5.12.2. Configuration requise pour Linux

1. Votre carte SCSI doit être prise en compte sous Linux. Rien ne fonctionnera avec une carte SCSI non reconnue. Même pour les contrôleurs "crades et pas chers" spécifiques aux scanners, il existe des pilotes pour Linux disponibles sur internet. Si vous voulez utiliser votre périphérique IDE, vous devez utiliser un émulateur ide-scsi. Jettez un oeil sur http://www.linuxdoc.org/HOWTO/CD-Writing-HOWTO.html (http://www.linuxdoc.org/HOWTO/CD-Writing-HOWTO.html) pour les instructions pour la configuration de ide-scsi.
2. Compilez les pilotes SCSI génériques dans votre noyau.
3. Cette étape ne semble plus être nécessaire avec les nouveaux noyaux (2.2.x) : Linux utilise par défaut une mémoire tampon pour SCSI plus petite que windows. L'instruction "define" `SG_BIG_BUFF`, utilisée lors de la compilation du noyau (dans un fichier `sg.h`), possède une valeur trop faible par défaut. Le projet SANE recommende par défaut une valeur de `130560` ce qui semble fonctionner plutôt bien. Ceci nécessite cependant une recompilation du noyau.
4. Effectuez le make pour les périphériques du scanner (périphériques SCSI génériques); jettez un oeil au manuel pratique concernant la programmation SCSI sur http://www.linuxdoc.org/HOWTO/SCSI-Programming-HOWTO.html (http://www.linuxdoc.org/HOWTO/SCSI-Programming-HOWTO.html) pour la numérotation des périphériques.
5. Je recommande de donner les droits en écriture sur le périphérique du scanner à un groupe. J'ai créé un groupe appelé `scanner` et je m'y suis ajouté. Faire l'exécution en tant que super-utilisateur augmente le risque d'envoyer de mauvaises instructions au mauvais périphérique. Avec un utilisateur normal vous êtes mieux protégé.
6. Pour les applications Win32 (WNASPI32), l'auto-détection par Wine fonctionne. Pour les applications Win16 (WINASPI) vous devez ajouter une ligne particulière concernant votre scanner dans le fichier ~/.wine/config. Le format est le suivant : `[scsi cCtTdD]` où `"C" = "controller"`, `"T" = "target"`, `"D"="LUN"`.

   Par exemple, j'ai réglé le mien à `0` pour "controller", `6` pour target,`0` pour LUN.

   ```
   [scsi c0t6d0]
   "Device" = "/dev/sgi"
   ```

   Tout cela varie en fonction de votre configuration SCSI.

5.12.3. Notes

L'inconvénient majeur est que ça fonctionne uniquement sous Linux pour le moment. Le code ASPI a uniquement été testé avec :

- un Mustek 800SP avec un contrôleur BusLogic pour Linux [BM]

- Un Siemens Nixdorf 9036 avec Adaptec AVA-1505 sous Linux via DOSASPI. Notez que j'ai quand même rencontré des problèmes de couleur (résultat à peine lisible) [AM]
- un lecteur CD Fujitsu M2513A (640MB) utilisant un pilote générique SCSI. Le formatage et l'ejection fonctionnait parfaitement. Merci à Uwe Bonnes pour le matériel ! [AM]

5.13. Utiliser ODBC

Cette partie décrit la façon dont ODBC fonctionnne dans Wine et comment le configurer.

Le système de gestion d'ODBC dans Wine, tout comme le système d'impression, a été réalisé pour laisser autant que possible la main au système Unix à un niveau élevé. Plutôt que de faire en sorte que tout le code windows fonctionne bien avec Wine, il utilise un pilote ODBC adéquat pour UNIX, comme UnixODBC. Ainsi si vous configurez Wine pour utiliser la DLL intégrée odbc32.dll, cette DLL de wine va s'interfacer avec le paquetage ODBC pour Unix et le laisser faire le travail à la place alors que si vous configurez Wine pour qu'il utilise la DLL odbc32.dll native, il utilisera le pilote ODBC natif etc.

5.13.1. Utiliser un système ODBC pour Unix avec Wine

La première étape pour utiliser un système ODBC Unix avec Wine est bien sûr de faire marcher le système ODBC Unix lui-même. Commencez par télécharger le code source ou une RPM etc. Plusieurs systèmes ODBC Unix existent; pour sa part, l'auteur est habitué à unixODBC (avec le pilote pour la base de données DB2 d'IBM). Généralement ce type de système inclue un outil, comme **isql**, qui vous permet d'accéder aux données en ligne de commande afin de pouvoir vérifier que le système fonctionne bien.

L'étape suivante est de lier la bibliothèque ODBCUnix à la DLL odbc32 intégrée à Wine. La dll odbc32 intégrée consulte (actuellement) la variable d'environnement *LIB_ODBC_DRIVER_MANAGER* pour récupérer le nom de la bibliothèque ODBC. Par exemple le fichier .bashrc de l'auteur contient la ligne :

```
export LIB_ODBC_DRIVER_MANAGER=/usr/lib/libodbc.so.1.0.0
```

Si cette variable d'environnement n'est pas intialisée, la DLL recherche une bibliothèque appelée libodbc.so et vous pouvez donc ajouter un lien symbolique vers votre propre bibliothèque. Par exemple exécutez en tant que super-utilisateur les commandes :

```
# ln -s libodbc.so.1.0.0 /usr/lib/libodbc.so
# /sbin/ldconfig
```

La dernière étape de la configuration est de vérifier que Wine est configuré pour utiliser la version intégrée de odbc32.dll, en modifiant la configuration de la DLL. Cette DLL intégrée se comporte comme

une interface d'accès [stub] entre le code appelant et la bibliothèque ODBC Unix .

Si vous rencontrez des problèmes, vous pouvez utiliser la commande WINEDEBUG=+odbc32 avant d'exécuter wine pour afficher une trace de ce qui se passe. Une mise en garde : en fait, certains programmes trichent un peu et passent outre la bibliothèque ODBC. Par exemple le moteur de Crystal Reports consulte le registre pour vérifier le DSN. Une solution à ce problème est décrite sur le site d'unixODBC où se trouve une section consacrée à l'utilisation d'unixODBC avec Wine.

5.13.2. Utiliser les pilotes ODBC de Windows

Les pilotes ODBC natifs semblent, à ce qu'on dit, fonctionner avec beaucoup de bases de données dont MSSQL et Oracle. En fait certaines comme MSSQL sont uniquement accessibles sous Linux via une application de Winelib. Au lieu de simplement recopier les fichiers DLL, la plupart des pilotes ODBC nécessitent de lancer un installeur Windows pour configurer correctement certaines choses comme la mise à jour des clés de registre.

Afin de configurer MSSQL, il vous faut d'abord télécharger et exécuter l'installeur mdac_typ.exe de microsoft.com. Afin de configurer vos connections ODBC, vous devez ensuite exécuter CLICONFG.EXE et ADBCAD32.EXE sous Wine. Vous les trouverez dans le répertoire windows\system après avoir exécuté mdac_typ. Comparez les messages de sortie avec ceux des programmes exécutés sur une machine Windows native. Certains points, comme les protocoles, peuvent manquer car ils doivent être installés en même temps que le système d'exploitation. Si c'est le cas, vous pourrez toujours recopier les fonctionnalités manquantes à partir d'une installation Windows existante ainsi que toutes les clés de registre nécessaires. Une installation de Windows native configurée pour être utilisée par Wine devrait pouvoir fonctionner de la même manière.

Types de bases de données testées avec succès sous Wine :

Type de Base de Données	Fonctionnalités utilisables
MS SQL	100%

Informez-nous de tout autre avancée sur la liste de diffusion wine-devel (mailto:wine-devel@winehq.org).

Chapitre 6. Exécuter Wine

Ce chapitre décrit tous les aspects de l'exécution de Wine, comme par exemple la simple invocation de Wine, les paramètres en lignes de commandes des différents programmes d'exploitation de Wine etc.

6.1. Utilisation de base : les "programmes" du menu démarrer et les mini-applications [applets] du panneau de configuration

Si vous utilisez une installation factice de Windows, vous pouvez installer les applications avec Wine, de la même manière que sous Windows : en exécutant un installeur. Il suffit simplement d'accepter les options d'installation par défaut, la plupart des installeurs effectueront par défaut l'installation dans "C:\Program Files", ce qui est une bonne chose. Si l'application vous le propose, vous pouvez créer des icônes sur votre bureau ainsi que dans votre menu "Programmes". Dans ce cas, vous pourrez démarrer l'application en cliquant simplement dessus.

Pour désinstaller un programme, on utilise d'habitude le désinstalleur fourni avec celui-ci et généralement situé dans l'application "ajout/Suppression de programmes" ["Add/Remove Programs"] du panneau de configuration. Pour utiliser l'équivalent avec Wine, exécutez le programme **uninstaller** (il est situé dans le répertoire `programs/uninstaller/` des sources de Wine) dans une *console* :

```
$ uninstaller
```

Certains programmes installent de mini-applications associées dans le panneau de configuration, comme Internet Explorer ou QuickTime par exemples. Vous pouvez accéder au panneau de configuration de Wine en tapant dans une *console* :

```
$ wine control
```

ce qui ouvrira une fenêtre contenant le panneau de configuration, comme sous Windows.

Si l'application n'installe pas de raccourci dans le menu "Programmes" ou sur le bureau, vous devrez lancer l'application en lignes de commandes. En se souvenant de l'endroit où vous l'avez installée, quelque chose de ce goût là :

```
$ wine "c:\program files\nom_application\application.exe"
```

devrait probablement faire l'affaire. Le chemin d'accès n'est pas sensible à la casse, mais pensez par contre à ajouter les guillemets. Certains programmes n'utilisent pas toujours des noms explicites pour les répertoires ou les fichiers EXE, donc vous devrez regarder dans le répertoire "program files" pour voir où se trouve tel ou tel élément.

6.2. Comment exécuter Wine

Il suffit simplement d'invoquer la commande **wine** pour obtenir un petit message d'aide :

```
Wine 20040405
Usage: wine PROGRAM [ARGUMENTS...]   Run the specified program
       wine --help                   Display this help and exit
       wine --version                Output version information and exit
```

Le premier argument doit être le nom du fichier que vous voulez faire exécuter par **wine**. Si l'exécutable est listé dans la variable d'environnement *`Path`*, vous pouvez indiquer uniquement le nom du fichier. Cependant, si l'exécutable ne se trouve pas dans le *`Path`*, vous devez donner le chemin d'accès entier jusqu'à l'exécutable (au format Windows, et non UNIX !). Par exemple pour le *`Path`* suivant :

```
[wine]
"Path"="c:\\windows;c:\\windows\\system;e:\\;e:\\test;f:\\"
```

Vous pourriez exécuter le fichier `c:\windows\system\foo.exe` ainsi :

```
$ wine foo.exe
```

Cependant il faudrait lancer le fichier `c:\myapps\foo.exe` avec cette commande :

```
$ wine c:\\myapps\\foo.exe
```

(notez la présence de l'anti-slash "\" !)

Pour plus d'informations sur l'exécution de programmmes en mode texte (CUI), consultez la section ci-dessous.

6.3. Les environnements graphiques du style Explorer pour Wine

Si vous préférez utiliser une interface graphique pour gérer vos fichiers, vous devriez vous intéresser à Winefile. Cette application de Winelib est incluse avec Wine et se trouve avec les autres programmes Wine. C'est un moyen simple de voir la configuration de vos disques et de localiser vos fichiers, en plus vous pouvez lancer des programmes directement à partir de Winefile. Veuillez noter que beaucoup de fonctions ne sont pas encore implémentées.

6.4. Les options de Wine en lignes de commandes

6.4.1. --help

Affiche une mini-page d'aide sur les lignes de commandes

6.4.2. --version

Affiche des informations sur le numéro de version de Wine. Utile pour vérifier votre installation.

6.5. Variables d'environnement

6.5.1. WINEDEBUG=[canaux]

Wine n'est pas parfait et beaucoup d'applications Windows ne fonctionnent toujours pas sans bogue sous Wine (mais d'un autre côté de nombreux programmes ne fonctionnent pas sans bogue sous Windows non plus !). Pour simplifier la tâche des personnes qui cherchent à expliquer ce qui provoque chaque bogue, Wine fournit un certain nombre de *canaux de débogage* dans lesquels vous pouvez puiser.

Chaque canal de débogage, lorsqu'il est activé, va déclencher l'affichage de messages du journal de bord [log] dans la console où vous avez invoqué **wine**. A partir de là vous pouvez rediriger les messages vers un fichier et les examiner à loisir. Mais attention, soyez prévenus ! Certains canaux de débogage génèrent un volume énorme de messages du journal de bord. Parmi les coupables les plus prolifiques, on compte le canal `relay` qui balance un message chaque fois qu'une fonction win32 est appelée, `win` qui traque le passage de messages à windows, et bien sûr `all` qui est un alias pour chacun des canaux de débogage existant. Pour une application complexe, votre journal de débogage peut facilement dépasser 1Mo. Une trace de `relay` peut souvent atteindre voire dépasser les 10 Mo de messages de journal de bord selon le temps d'utilisation de l'application. (Comme décrit dans la partie Débogage de "configuration de Wine",

vous pouvez modifier ce que la trace de *`relay`* contient). Le fait d'utiliser le journal de bord ralentit sensiblement Wine, aussi n'utilisez pas *`WINEDEBUG`* à moins de vraiment vouloir un fichier de journal de bord.

Dans chaque canal, vous pouvez spécifier une *classe de message* pour filtrer les différentes erreurs en fonction de leur gravité. Les quatre classes de messages sont : *`trace`*, *`fixme`*, *`warn`*, *`err`*.

Pour activer un canal de débogage, utilisez la forme *`class+channel`*. Pour le désactiver, utilisez *`class-channel`*. Pour utiliser plusieurs canaux dans la même option *`WINEDEBUG`*, séparez les par des virgules. Par exemple, pour demander les messages de classe *`warn`* du canal de débogage *`heap`*, vous pouvez invoquer **wine** comme ceci :

```
$ WINEDEBUG=warn+heap wine nom_du_programme
```

Si vous n'indiquez pas la classe de message, **wine** affichera les messages des quatre classes pour ce canal :

```
$ WINEDEBUG=heap wine nom_du_programme
```

Si vous souhaitez voir les messages de bord pour tous les canaux, hormis le canal relay, vous pouvez faire quelque chose comme :

```
$ WINEDEBUG=+all,-relay wine nom_du_programme
```

Voici une liste des canaux de débogage et des classes de Wine. Des canaux supplémentaires seront ajoutés (ou enlevés) dans les versions futures.

Tableau 6-1. Canaux de débogage

accel	adpcm	advapi	animate	aspi
atom	avicap	avifile	bidi	bitblt
bitmap	cabinet	capi	caret	cdrom
cfgmgr32	class	clipboard	clipping	combo
comboex	comm	commctrl	commdlg	computername
console	crtdll	crypt	curses	cursor
d3d	d3d_shader	d3d_surface	datetime	dc
ddeml	ddraw	ddraw_fps	ddraw_geom	ddraw_tex
debugstr	devenum	dialog	dinput	dll
dma	dmband	dmcompos	dmfile	dmfiledat
dmime	dmloader	dmscript	dmstyle	dmsynth
dmusic	dosfs	dosmem	dplay	dplayx
dpnhpast	driver	dsound	dsound3d	edit

enhmetafile	environ	event	eventlog	exec
file	fixup	font	fps	g711
gdi	global	glu	graphics	header
heap	hook	hotkey	icmp	icon
imagehlp	imagelist	imm	int	int21
int31	io	ipaddress	iphlpapi	jack
joystick	key	keyboard	listbox	listview
loaddll	local	mapi	mci	mcianim
mciavi	mcicda	mcimidi	mciwave	mdi
menu	menubuilder	message	metafile	midi
mmaux	mmio	mmsys	mmtime	module
monthcal	mpeg3	mpr	msacm	msdmo
msg	mshtml	msi	msimg32	msisys
msrle32	msvcrt	msvideo	mswsock	nativefont
netapi32	netbios	nls	nonclient	ntdll
odbc	ole	oledlg	olerelay	opengl
pager	palette	pidl	powermgnt	print
process	profile	progress	propsheet	psapi
psdrv	qcap	quartz	ras	rebar
reg	region	relay	resource	richedit
rundll32	sblaster	scroll	seh	selector
server	setupapi	shdocvw	shell	shlctrl
snmpapi	snoop	sound	static	statusbar
storage	stress	string	syscolor	system
tab	tape	tapi	task	text
thread	thunk	tid	timer	toolbar
toolhelp	tooltips	trackbar	treeview	ttydrv
twain	typelib	uninstaller	updown	urlmon
uxtheme	ver	virtual	vxd	wave
wc_font	win	win32	wineboot	winecfg
wineconsole	wine_d3d	winevdm	wing	winhelp
wininet	winmm	winsock	winspool	wintab
wintab32	wnet	x11drv	x11settings	xdnd
xrandr	xrender	xvidmode		

Pour plus de détails concernant les canaux de débogage, consultez Le guide Wine du le développeur [The Wine Developers Guide] (http://wine.codeweavers.com/docs/wine-devel/).

6.6. Options de lignes de commandes pour wineserver

En principe, Wine démarre automatiquement wineserver après le premier processus Wine. Cependant, wineserver possède des options utiles en lignes de commandes que vous pouvez utiliser si vous le démarrez à la main, par exemple via un script d'authentification utilisateur ou autre.

6.6.1. -d<n>

Fixe le niveau de débogage à la valeur <n> pour l'affichage dans la console où wineserver a été invoqué. En d'autres termes, tout ce qui est supérieur à 0 indique à wineserver d'afficher une sortie de débogage particulière.

6.6.2. -h

Affiche le message d'aide des options de lignes de commandes de wineserver.

6.6.3. -k[n]

Tue le wineserver courant, et ce avec le signal n, éventuellement donné en paramètre.

6.6.4. -p[n]

Ce paramètre rend wineserver persistant, durant les n secondes, éventuellement indiquées en paramètres. Ceci empêche wineserver de s'eteindre immédiatement.

En général, wineserver s'éteint quasiment immédiatement après que le dernier processus de wine utilisant wineserver se soit éteint. Cependant, étant donné que wineserver charge beaucoup de choses au démarrage (comme la base de registre entière de Windows), son démarrage peut être lent et il peut être utile de l'empêcher de s'éteindre après la fin de toute session de Wine, en le rendant persistant.

6.6.5. -w

Ce paramètre met en attente la nouvelle instance de wineserver jusqu'à ce que l'ancienne instance active se termine.

6.7. Configuration des variables d'environnement Windows/DOS

Votre programme peut nécessiter la présence de variables d'environnement correctement initialisées pour fonctionner convenablement. Dans ce cas, vous devez initialiser cette variable d'environnement dans l'interpréteur de commandes de Linux, puisque Wine transmet l'ensemble de la configuration des variables d'environnement de l'interpréteur de commandes à l'espace des variables d'environnement de Windows. Voici un exemple pour un interpréteur de commandes bash (les autres interpréteurs peuvent avoir une syntaxe différente !) :

```
export MAVARIABLE=mavaleurdevariable
```

Ainsi votre programme Windows pourra accéder de manière sûre à la variable d'environnement MAVARIABLE dès que vous démarrerez ce dernier en utilisant Wine. Si vous souhaitez initialiser MAVARIABLE de façon permanente, vous pouvez écrire la configuration dans /etc/profile ou alors dans ~/.bashrc dans le cas où vous utilisez bash.

Notez qu'il existe cependant une exception à la règle : si vous désirez changer votre variable d'environnement PATH, vous ne pouvez bien sûr pas le faire de cette manière étant donné que cela modifie la valeur de la variable d'environnement PATH d'UNIX. Vous devez utiliser la variable d'environnement WINEPATH à la place. Une autre manière d'indiquer le contenu de la variable d'environnement PATH de DOS est de changer le paramètre "path" dans le fichier de configuration de Wine dans la partie [wine].

6.8. Programmes en mode texte (CUI: Interface utilisateur en mode console [Console User Interface])

Les programmes en mode texte sont des programmes dont l'affichage en sortie se fait uniquement en texte (étonnant, non ?). Dans la terminologie Windows, on les appelle exécutables CUI (Interface Utilisateur en mode Console [Console User Interface]), par opposition aux exécutables GUI (Interface Graphique Utilisateur). L'interface de programmation Win32 [API Win32] fournit un ensemble complet de fonctions pour gérer cette situation, ce qui va des fonctionnalités basiques comme l'affichage de texte à l'écran, aux fonctionnalités de haut niveau (comme la gestion du plein écran, la gestion de la couleur, les déplacements du curseur, la gestion de la souris), ou encore des fonctionnalités comme l'édition des lignes ou la gestion des flux d'entrés bruts ou mis en forme.

Etant donné le large panel de fonctionnalités présentées ci-dessus et leur usage courant dans le monde Un*x, Wine propose trois différentes manières d'exécuter un programme en mode console (c'est-à-dire un exécutable en mode console) :

- en flux bruts
- avec wineconsole en console utilisateur

- avec wineconsole en console de type curses (console semi-graphique)

Les noms ci-dessus sont un peu compliqués. "flux bruts" signifie qu'aucune fonctionnalité supplémentaire de wine n'est fournie pour effectuer le lien entre l'accès à la console unix et l'accès à la console windows. Les deux autres solutions nécessitent l'utilisation d'un programme spécifique de Wine (wineconsole) qui fournit des fonctionnalités avancées. Le tableau suivant décrit ce que vous pouvez faire (et ne pas faire) avec ces trois solutions.

Tableau 6-2. Différences générales entre les consoles

Fonctionnalité	**flux bruts**	**Wineconsole & console utilisateur**	**Wineconsole & console type curse**
Comment exécuter (supposons un exécutable appelé foo.exe)	`$ wine foo.exe`	`$wineconsole --backend = user foo.exe`	`$ wineconsole foo.exe` Vous pouvez aussi utiliser l'option --backend=curses
Bonne prise en compte des applications CUI en mode texte (qui affichent les informations ligne par ligne)	Oui	Oui	Oui
Bonne prise en compte des applications CUI en plein écran (gestion des couleurs, gestion de la souris entre autres...)	Non	Oui	Oui
Fonctionne même si X11 n'est pas lancé	Oui	Non	Oui
Implémentation	Fait correspondre les flux de données standards de Windows aux flux de données standards d'Unix stdin/stdout/stderr)	Wine console crée une nouvelle fenêtre (ceci requiert la présence de la DLL USER32) où toute l'information est affichée.	Wineconsole utilise la console unix active (à partir de laquelle le programme tourne) et utilise l'espace de la console avec l'aide de la bibliothèque (n)curses pour intéragir avec l'utilisateur.
Problèmes connus			Comportement étrange dans le cas où deux consoles Windows ou plus sont lancées à partir de la même console Un*x.

6.8.1. Configuration des exécutables de type CUI

Lorsque l'on utilise wineconsole, il existe plusieurs options de configuration disponibles. Wine (comme sous Windows) stocke plusieurs options par application dans un registre. Cela permet à un utilisateur, par exemple, de définir la taille par défaut de la mémoire tampon de l'écran pour une application donnée.

A ce jour, seul le mode console utilisateur permet de régler ces options (nous vous recommandons de ne pas éditer à la main le contenu du registre). L'interface de paramétrage s'ouvre lorsqu'un utilisateur fait un clic-droit dans la console (un menu apparaît) et vous pouvez y choisir l'une des alternatives suivantes :

- Par défaut [Default] : cela permet de déterminer les paramètres partagés par toutes les applications qui n'ont pas encore été configurées. Ainsi, lorsqu'une application est exécutée pour la première fois (sur votre machine et sous votre compte utilisateur) dans wineconsole, ce dernier appliquera ces paramètres par défaut à l'application. Après, l'application possèdera ses propres configurations, et vous pourrez les modifier à votre guise.

  Propriétés [Properties] : cela permet de déterminer les paramètres de l'application. Lorsque vous avez terminé vos réglages, une invite de commande vous demandera si vous souhaitez :

  1. Conserver ces modifications pour cette session uniquement (la prochaine fois que vous relancerez l'application, les modifications que vous venez de faire ne seront pas prises en compte).
  2. Utiliser ces modifications pour cette session et les sauvegarder également, de telle manière que la prochaine fois que vous relancerez l'application, vous utliserez encore ces nouveaux paramètres.

Voici une liste des éléments que vous pouvez configurer ainsi que leur signification :

Tableau 6-3. Options de configuration de Wineconsole

Option de configuration	Signification
Taille du curseur	Définit la taille du curseur. Trois options sont possibles : petit (33% de la hauteur des caractères), moyen (66%) et grand (100%)

Option de configuration	Signification
Menu popup	On a vu précédemment que la fenêtre popup de configuration de wineconsole s'affichait grâce à un clic-droit dans la console. Néanmoins, ça peut être un problème si l'application que vous exécutez à l'intérieur de wineconsole a besoin de recevoir les évènements de clic-droit. En cochant control ou shift, vous selectionnez un bouton additionnel au clic-droit pour l'ouverture de la fenêtre popup. Par exemple, le fait de cocher shift enverra un évènement à l'application lorsque vous ferez un clic-droit dans la fenêtre sans tenir la touche shift appuyée, et ouvrira la fenêtre lorsque ferez un clic-droit en tenant la touche shift appuyée.
Edition rapide	Cette boîte à cocher vous permet de choisir si le clic-gauche doit être interprété comme un évènement à envoyer à l'application exécutée (case décochée) ou bien comme une selection de zone rectangulaire de l'écran servant à être copiée plus tard dans le presse-papier (case cochée).
Historique	Cela vous permet de configurer combien de commandes vous souhaitez pouvoir ré-effectuer. Vous pouvez également décider de stocker toutes les occurences (décocher) ou seulement la dernière (cocher) d'une commande que vous auriez tapée plusieurs fois à la suite - éventuellement entrecoupée par d'autres commandes.
Police	La propriété de police vous permet de choisir la police par défaut pour la console (type de fonte, taille, couleur d'arrière-plan et d'avant-plan.
Mémoire tampon de l'écran & taille de la fenêtre	La console, telle que vous la voyez, est faite de deux parties différentes. D'un côté, il y a la mémoire tampon de l'écran qui contient toutes les informations que votre application affiche à l'écran, puis la fenêtre qui affiche une zone donnée de la mémoire tampon de l'écran. Notez que la fenêtre est toujours plus petite ou de même taille que la mémoire tampon de l'écran. Si la taille de la fenêtre est strictement plus petite que la taille de la mémoire tampon, des barres de défilement apparaîtront sur la fenêtre, de telle manière que vous puissiez voir tout le contenu de la mémoire tampon de l'écran.

Option de configuration	**Signification**
Fermer en quittant	Si l'option est cochée, alors la wineconsole se fermera lorsque l'application se terminera. Dans le cas contraire elle restera ouverte jusqu'à ce que l'utilisateur la ferme manuellement : cela permet de voir les dernières informations affichées par un programmme après qu'il se soit éteint.
Mode d'édition	Lorsque l'utilisateur entre des commandes, il ou elle peut choisir entre plusieurs modes d'édition : • Emacs : les mêmes raccourcis clavier que sous emacs sont disponibles. Par exemple, Ctrl-A ramènera le curseur en début de ligne. Consultez votre manuel emacs pour plus de détails concernant les commandes. • Win32 : il s'agit des raccourcis clavier standards de la console Windows (principalement basés sur les touches flêche).

Chapitre 7. Dépannage / Signaler des bogues

7.1. Que faire si un programme ne marche toujours pas?

Il y a des fois comme ça, où vous avez tout essayé, vous avez même sacrifié un chat, par un soir de pleine lune, puis vous l'avez mangé, accompagné d'ail pourri et de poisson pas frais, tout en entamant la danse du malin; mais rien à faire, pas moyen de faire fonctionner ce foutu programme sur quelque version de Wine que ce soit. Ne désespérez pas, nous sommes là pour vous aider... (en d'autres termes : combien êtes-vous prêt à payer?)

7.1.1. Vérifiez votre configuration de Wine

Jettez donc un oeil à la Section vérification de la configuration

7.1.2. Utilisez les paramètres de différentes versions de Windows

Dans de nombreux cas, utiliser des paramètres de différentes version de Windows peut aider.

7.1.3. Utilisez différents chemins d'accès à l'exécutable

Ca peut marcher aussi : Utilisez **wine prg.exe** et **wine x:\\chemin\\complet\\vers\\prg.exe**

7.1.4. Jouez sur la configuration des DLL

Exécutez avec l'option WINEDEBUG=+loaddll pour voir quelles DLL sont utilisées et si elles sont chargées en étant natives ou prédéfinies. Ensuite assurez-vous que les bons fichiers DLL natifs se trouvent dans le répertoire C:\windows\system configuré et jouez avec les paramètres concernant l'ordre de chargement des DLL, en ligne de commande ou dans le fichier de configuration.

7.1.5. Vérifiez l'environnement de votre machine !

Juste une idée en l'air : se pourrait-il que votre environnement de compilation ou d'exécution de Wine soit défaillant ? Assurez vous qu'il n'existe absolument aucun problème avec les paquetages dont Wine dépend (gcc, glibc, les bibliothèques de votre interface graphique X, OpenGL (!), ...) C'est bizarre parfois il y en a qui n'arrivent pas à trouver ce qui plante lorsqu'ils utilisent les "mauvais" fichiers

d’entête avec les "bonnes" bibliothèques !!! (Ce qui se traduit par des journées de débogage pour essayer en vain de trouver pourquoi cette fonction de bas niveau produit une erreur d’une façon complètement incompréhensible... ARGH !)

7.1.6. Utilisez différentes GUI [Interfaces utilisateur graphiques] (Gestionnaire de fenêtre)

Dites à Wine via le fichier de configuration d’utiliser soit le mode Poste de travail, soit le mode de gestion via le serveur, soit le mode texte "normal" et moche. Ca peut faire une sacrée différence ça aussi.

7.1.7. Vérifiez votre application !

Votre application comporte peut-être une sorte de protection contre la copie ? De nombreuses protections contre la copie ne fonctionnent pas avec Wine actuellement. Néanmoins certaines seront prises en compte à l’avenir. (la couche CD-ROM ne possèdent pas encore toutes les fonctionnalités).

Rendez vous sur GameCopyWorld (http://www.gamecopyworld.com) et essayez de trouver un crac pour votre jeu, qui puisse se débarasser de cette vilaine protection contre la copie. J’espère que vous possédez bien une copie légale du logiciel... :-)

7.1.8. Vérifiez votre environnement Wine

Fonctionner avec ou sans partition Windows peut avoir d’énormes conséquences. Configurez Wine pour faire qu’il fasse le contraire de ce que vous aviez avant. Installez aussi DCOM95 ou DCOM98. Ça peut être très utile.

7.1.9. Reconfigurez Wine

Parfois le processus d’installation de Wine change et les nouvelles versions de Wine se basent sur ces changements. C’est notamment le cas si vous votre installation date un peu. Renommez votre répertoire `~/.wine` actuel afin de le sauvegarder. Utilisez le processus d’installation recommandé pour votre distribution de Wine afin de créer une nouvelle configuration. Utilisez les informations présentes dans l’ancien `~/.wine` comme référence. Pour les distributions de Wine en code source, configurez Wine en exécutant le script tools/wineinstall en tant qu’utilisateur pour lequel vous souhaitez faire la configuration. C’est une opération relativement sûre. Vous pouvez ensuite supprimer le nouveau répertoire `~/.wine` et renommer l’ancien.

7.1.10. Consultez les informations suivantes

Il y a de fortes chances pour que quelqu'un ait déjà essayé de faire la même chose que vous. Vous trouverez sûrement de l'aide sur les pages suivantes :

- Parcourez dans la base de données des applications de WineHQ (http://appdb.winehq.org) pour trouver des astuces à propos du logiciel. Si votre version du logiciel n'est pas dans la liste, vous pouvez trouver de l'aide en consultant la rubrique concernant une version différente.
- Frank's Corner (http://www.frankscorner.org) contient une liste d'applications ainsi que les informations détaillées pour les configurer. Vous trouverez plus d'aide dans le forum des utilisateurs.
- Google (http://www.google.com) peut être utile selon la façon dont il est utilisé. Il peut être utile de chercher dans les groupes de Google (http://groups.google.com), et notamment dans le groupe comp.emulators.ms-windows.wine (http://groups.google.com/groups?hl=en&lr=&ie=UTF-8&group=comp.emulators.ms-windows.wine).
- Freenode.net (http://www.freenode.net) héberge un canal IRC pour Wine. Vous pouvez y accéder en utilisant n'importe quel client IRC comme Xchat. Les paramètres dont vous avez besoin sont les suivants : serveur = irc.freenode.net, port = 6667, et canal = #winehq
- Si vous avez un logiciel qui doit utiliser le moteur d'exécution de Visual Basic [Visual Basic Runtime Environment] vous pouvez le télécharger à partir du site de Microsoft suivant (http://www.microsoft.com/downloads/details.aspx?FamilyID=bf9a24f9-b5c5-48f4-8edd-cdf2d29a79d5&DisplayLang=en/)
- Si vous savez qu'il vous manque une DLL, comme mfc42 par exemple, vous pouvez peut être la trouver sur www.dll-files.com (http://www.dll-files.com/).
- La liste de diffusion (http://www.winehq.org/site/forums#ml) de Wine peut aussi se révéler utile, notamment wine-users [utilisateurs de Wine]. La liste wine-devel [développeurs Wine] peut aussi convenir selon le type de problème rencontré. Si vous laissez un message sur wine-devel, vous devez être prêt à faire un minimum de travail pour aider à diagnostiquer le problème. Lisez la section ci-dessous pour savoir comment déboguer ce qui cause problème.
- En désespoir de cause, vous aurez peut-être envie tester une version commerciale de Wine pour voir si votre application est prise en compte.

7.1.11. Déboguez !

L'étape suivante consiste à trouver la cause de votre problème. Il existe un large éventail de problèmes possibles allant de simples problèmes de configuration à des fonctions pas du tout implémentées dans Wine. Dans la partie suivante, nous décrivons comment rédiger un rapport de bogue et comment commencer à déboguer un plantage. Pour plus d'informations sur l'utilisation des fonctionnalités de débogage de Wine, n'hésitez pas à consulter le guide de Wine pour les développeurs [The Wine Developers Guide].

7.2. Comment signaler un bogue

Veuillez signaler tous les bogues ainsi que toute information utile, sur l'interface Bugzilla pour Wine (http://bugs.winehq.org/). Parcourez dans la base de données de Bugzilla pour vérifier si votre problème n'a pas déjà été signalé. Si c'est la cas ajoutez au rapport de bogue original toute information utile.

7.2.1. Tous les rapports de bogue

Voici quelques conseils simples pour rendre plus efficace votre rapport de bogue (et ainsi être certain d'être plus vite renseigné et dépanné) :

1. Laissez le plus possible d'informations utiles dans votre message.

   Ceci signifie que nous avons besoin d'informations plus détaillées qu'un simple "MS Word plante chaque fois que je le lance. Quelqu'un sait pourquoi ?" Ajoutez au moins les informations suivantes :

   - Quelle version de Wine vous utilisez (tapez **wine -v**)
   - Le nom du système d'exploitation que vous utilisez, quelle distribution (si il y en a une), ainsi que sa version (par exemple Linux Red Hat 7.2)
   - Quel compilateur vous utilisez ainsi que sa version, (tapez **gcc -v**). Si vous n'avez pas compilé Wine indiquez alors le nom du packetage et où vous l'avez obtenu.
   - La version de votre Windows si vous l'utilisez avec Wine. Mentionnez aussi le fait que vous n'utilisez pas Windows le cas échéant.
   - Le nom du programme que vous tentez de faire fonctionner, son numéro de version et l'URL à laquelle on peut obtenir le logiciel (si il est disponible sur le web).
   - La commande exacte que vous avez tapé pour démarrer Wine. (par exemple, **wine "C:\Program Files\Test\program.exe"**).
   - Les étapes exactes à suivre pour reproduire le bogue.
   - Toute information supplémentaire que vous jugez pertinente ou utile, comme la version de votre serveur graphique X en cas de problème avec votre X, la version de libc etc.

2. Relancez le programme avec la variable d'environnement WINEDEBUG `WINEDEBUG=+relay` (par exemple, **WINEDEBUG=+relay wine sol.exe**).

   Cela affichera dans la console des informations supplémentaires qui peuvent se révéler utiles pour déboguer un programme. Cela ralentit aussi l'exécution du programme. Il existe des cas où le bogue semble disparaître avec le paramètre `+relay` activé. Veuillez le mentionner dans le rapport de bogue.

7.2.2. Plantages

Si Wine plante en exécutant votre programme, il est important d'avoir cette information pour avoir une chance de savoir ce qui cause le plantage. Ceci peut générer pas mal d'informations (plusieurs Mo), donc il est préférable de les écrire dans un fichier. Lorsque l'invite de commande `Wine-dbg>` apparaît, tapez **`quit`**.

Vous souhaitez peut-être essayer `+relay,+snoop` plutôt que `+relay`, mais veuillez noter que `+snoop` est relativement instable et plantera souvent avant un simple `+relay`! Si c'est le cas alors veuillez utiliser *uniquement* `+relay`!! Un rapport de bogue avec un plantage en mode `+snoop` est inutile dans la plupart des cas ! Vous pouvez aussi activer d'autres paramètres, selon la nature du problème que vous voulez essayer de comprendre. Consultez le manuel de Wine pour une liste complète des paramètres.

Pour obtenir une trace en sortie, utilisez une des méthodes ci-dessous :

7.2.2.1. La méthode simple

1. Cette méthode est faite pour permettre à quelqu'un de totalement novice de fournir un journal de bord approprié dans l'eventualité d'un plantage.

   Perl *doit* être installé sur votre machine pour que cette méthode fonctionne. Pour savoir si perl est installé, tapez **which perl**. Si cela retourne quelque chose du style `/usr/bin/perl`, vous êtes bon. Dans le cas contraire, passez directement à "La méthode difficile". Si vous n'êtes pas tout à fait sûr, continuez. Lorsque vous essayerez d'exécuter le script, vous vous apercevrez *très* facilement que vous n'avez pas perl.

2. Placez vous dans le repertoire <`chemin jusqu'à wine>/tools`
3. Tapez la commande **./bug_report.pl** et suivez les instructions.
4. Postez le bogue sur Wine Bugzilla (http://bugs.winehq.org/). Avant de faire ça, veuillez rechercher dans la base de données Bugzilla pour savoir si votre problème a déjà été repertorié dans un rapport de bogue. Ajoutez votre propre description détaillée du problème, accompagnée des informations utiles. Joignez votre "joli rapport en bonne et dûe forme" à votre bogue. Ne faites pas de copier-coller du rapport dans la description de votre bogue - il est relativement volumineux. Conservez la sortie complète de débogage au cas où il soit nécessaire aux développeurs Wine.

7.2.2.2. La méthode difficile

Il est quasiment certain que seules les 100 dernières lignes de la trace environ soient nécessaires pour trouver l'endroit où le programme plante. Pour obtenir ces 100 dernières lignes, il faut effectuer les opérations suivantes :

1. Redirigez toute la sortie de `WINEDEBUG` vers un fichier.

2. Séparez les 100 dernières lignes dans un autre fichier en utilisant la commande **tail**.

Il suffit pour cela de suivre l'une des méthodes suivantes.

Dans n'importe quel interpréteur de commande:

```
    $ echo quit | WINEDEBUG=+relay wine [autres_options]
nom_du_programme >& nomfichier.out;
$ tail -n 100 nomfichier.out > fichier_rapport
```

(Le résultat est que tous les messages de débogage vont être écrits dans le fichier uniquement et puis quitter automatiquement. C'est assurément une bonne idée que d'utiliser cette commande, étant donné que Wine affiche tellement de messages de débogage qu'ils inondent la console et consomment des cycles processeur.)

dans les interpréteurs de commandes tcsh et du type csh :

```
    $ WINEDEBUG=+relay wine [autres_options]
nom_du_programme |& tee nomfichier.out;
$ tail -n 100 nom_fichier.out > fichier_rapport
```

dans les interpréteurs de commandes bash et du type sh :

```
    $ WINEDEBUG=+relay wine [autres_options]
nom_du_programme 2>&1 | tee nom_fichier.out;
$ tail -n 100 nom_fichier.out > fichier_rapport
```

`fichier_rapport` contiendra maintenant les cent dernières lignes, avec la pile mémoire et la pile des appels de fonctions, qui sont les informations les plus importantes. Veuillez ne pas supprimer cette partie même si vous ne comprenez pas ce que ça signifie.

Postez le bogue sur Wine Bugzilla (http://bugs.winehq.org/). Vous devez joindre le fichier de sortie `fichier_rapport` de la partie 2), accompagné d'informations utiles sur la méthode utilisée pour le créer. Ne faites pas de copier-coller du rapport dans la description du bogue - c'est relativement volumineux et cela va rendre difficile la lecture de votre rapport de bogue. Si vous procédez comme expliqué ci dessus, vous aurez de fortes chances de recevoir de l'aide.

Veuillez faire une recherche dans la base de données de Bugzilla pour vérifier si votre problème a déjà été répertorié. Si c'est le cas, joignez votre fichier de sortie `fichier_rapport` au rapport de bogue original et ajoutez toutes informations complémentaires utiles.

Glossaire

Fichier exécutable

Un fichier en code machine exécutable, sous forme compilée : données héxadécimales (par opposition à un fichier de code source).

CVS

Système de contrôle de versions : progiciel pour la gestion du développement de logiciels par plusieurs personnes. Reportez-vous au chapitre concernant le CVS dans le guide Wine pour les développeurs [The Wine Developers Guide] pour de plus amples informations.

Distribution

Les revendeurs proposent habituellement les CD de systèmes d'exploitation sous la forme de distributions (terme généralement rencontré dans le contexte Linux). Un environnement Linux peut être proposé sous de nombreuses configurations différentes : par exemple, les distributions peuvent être réalisées pour convenir aux jeux, aux applications scientifiques aux serveurs, aux postes de travail, etc.

DLL

Une DLL (bibliothèque de liens dynamiques) est un fichier qui peut être chargé et exécuté par des programmes de manière dynamique. En gros, il s'agit d'un dépôt externe de code pour les programmes. Etant donné qu'habituellement différents programmes réutilisent la même DLL au lieu d'avoir ce code dans leur propre fichier, l'espace de stockage requis s'en trouve fortement réduit. Un synonyme de DLL pourrait être "bibliothèque".

Editeur

Un éditeur est généralement un programme permettant de créer ou de modifier des fichiers texte. Il existe divers éditeurs en mode texte ou en mode graphique disponibles sur Linux.

Voici quelques exemples d'éditeurs graphiques : nedit, gedit, kedit, xemacs, gxedit.

Voici quelques exemples d'éditeurs en mode texte : joe, ae, emacs, vim, vi. Dans une *console*, vous pouvez les exécuter simplement comme ceci :

```
$ nomdelediteur
fichier
```

Variable d'environnement

Les variables d'environnement sont des définitions textuelles utilisées dans un *interpréteur de commandes [Shell]* pour stocker les configurations importantes du système. Dans un interpréteur de commandes **bash** (le plus couramment utilisé sous Linux), vous pouvez visualiser toutes les variables d'environnement en exécutant :

```
set
```

Si vous souhaitez changer la valeur d'une variable d'environnement, vous pouvez exécuter ceci :

```
export MAVARIABLE=mavaleur
```

Pour supprimer une variable d'environnement, exécutez :

```
unset MAVARIABLE
```

Paquetage [Package]

Un paquetage est un fichier compressé dans un format spécifique à une *distribution*. Il contient les fichiers de tel ou tel programme particulier que vous souhaitez installer. Les paquetages sont généralement installés grâce aux gestionnaires de paquetages **dpkg** ou **rpm**.

root

root est le nom donné au compte administrateur (ou super-utilisateur) de la machine. Pour exécuter un programme en tant que root, ouvrez simplement une *console* et tapez :

```
$ su -
```

Une invite de commande vous demandera le mot de passe de root sur votre machine et ensuite vous serez autorisé à effectuer les tâches d'administration de la machine qui requièrent les privilèges spéciaux de root. Le compte root est signalé par la présence de l'invite de commande

```
#
```

alors qu'un '$' indique un compte utilisateur normal.

Interpréteur de commandes [Shell]

Un interpréteur de commandes est un outil permettant aux utilisateurs d'interagir avec la machine. En général, les interpréteurs sont des outils en mode texte et utilisables en ligne de commandes. Les exemples d'interpréteurs de commandes connus sont **bash**,**tcsh** et **ksh**. Wine se base sur le fait que vous utilisez **bash** pour les tâches d'installation de ce dernier, étant donné qu'il s'agit de l'interpréteur le plus répandu sous Linux. Les interpréteurs sont généralement exécutés dans une *console*.

Code source

Le code source est le code qui compose un logiciel avant que le programme soit compilé, c'est à dire qu'il s'agit des instructions originales de création du programme qui disent au compilateur à quoi le programme doit ressembler une fois qu'il sera compilé en *Fichier exécutable*.

Console

Une console est en général une fenêtre graphique que l'on utilise pour exécuter un **Interpréteur de commandes**. Si Wine vous demande d'ouvrir une console, vous devez en fait cliquer sur l'icône de votre bureau qui représente une grande fenêtre noire (ou, dans d'autres cas, une icône représentant un coquillage marin [Note du traducteur : shell signifie coquillage en anglais].) Wine se base sur le fait que vous utilisez l'interpréteur de commandes **bash** dans une console, donc si votre console utilise un autre interpréteur, tapez simplement :

```
bash
```

dans la console.