WINE – INSTRUKCJA INSTALACJI I KONFIGURACJI
Installations- und Bedienungsanleitung für WINE

Peter Ganten
peter@ganten.org
7 lipca 2000

przekład z języka niemieckiego:
Wojciech Kapusta
w_kapusta@poczta.onet.pl
20 lutego 2003

Od tłumacza

Niniejsza instrukcja pochodzi z pakietu `wine-20021007.i386.rpm`*, zawarta jest w pliku* `installation-und-konfiguration.german`*. Przy tłumaczeniu nie dokonano żadnych zmian w treści, za wyjątkiem zmiany tytułu instrukcji (dosłownie:* ***Instrukcja instalacji i obsługi dla WINE)*** *oraz zmiany nazw kilku przykładowych osobistych katalogów i plików. Autor niniejszego przekładu nie jest zawodowym informatykiem i nie wnika w merytoryczną poprawność tego opracowania.*

1. Streszczenie

Tekst ten opisuje instalację, konfigurację i obsługę WINE. WINE jest środowiskiem czasu rzeczywistego do uruchamiania programów dla MS-Windows w systemach operacyjnych GNU/Linux i innych kompatybilnych z UNIX, na komputerach kompatybilnych z Intel-x86; program ten może być również wykorzystywany do przenoszenia na UNIX kodu źródłowego istniejących programów windowsowych. W opracowaniu tym chodzi jednak głównie o instalację i konfigurację środowiska czasu rzeczywistego dla programów windowsowych. Tekst ten był pierwotnie napisany jako materiał do wykładu o WINE i integracji aplikacji widowsowych pod GNU/Linux, na **Dni Linuksowe 2000** – 30 czerwca do 2 lipca 2000 w Stuttgarcie.

2. Wprowadzenie

WINE jest daleko idącym przyczynkiem do integracji aplikacji windowsowych pod GNU/Linux. WINE składa się m.in. programu ładującego (*loadera*), przy pomocy którego programy windowsowe i dosowe mogą być załadowane do pamięci i wykonane przez procesor komputera. Poza tym program ten udostępnia dużą część interfejsów (API) systemów operacyjnych opierających się na Windows. Interfejsy te wykorzystywane są przez programy widowsowe, uruchamiane przez WINE, tak że te programy znajdują to samo, oczekiwane środowisko, jak pod Windows. Ponieważ interfejsy te znajdują się w WINE i ich definicje istnieją w formie plików nagłówkowych (*header file*), WINE może być również wykorzystywany do przenoszenia kodu źródłowego programów windowsowych do GNU/ Linux czy innych systemów operacyjnych opartych na UNIX. Powstają wtedy prawdziwe programy uniksowe/linuksowe, mające tę samą funkcjonalność jak ich odpowiedniki pod Windows. Zastosowanie jednolitego interfejsu API pod Windows i Linuksem ma dla twórcy oprogramowania tę zaletę, że można projektować i rozwijać tylko jedną wersję kodu źródłowego, która może być używana w obu rodzinach systemów operacyjnych.

Projekt WINE został zapoczątkowany w 1993 roku, prowadzony jest głownie przez ochotników, komunikujących się między sobą przez listy dyskusyjne. W ostatnich czasach projekt WINE uzyskał dodatkowe wsparcie przez liczne przedsiębiorstwa komercyjne, stosujące WINE do przenoszenia swoich programów windowsowych na GNU/ Linux. Dziś WINE jest w stanie uruchomić pod Linuksem większość istniejących programów windowsowych (16 i 32 bitowych), w pewnym ograniczonym zakresie z WINE mogą być używane także programy dosowe. WINE uruchamia programy windowsowe bezpośrednio spod Linuksa, nie są do tego wymagane żadne specjalne rozszerzenia jądra, żadne szczególne uprawnienia, ani nie musi istnieć instalacja Windows. Projekt pozwala na uruchamianie programów pod Linuksem tak samo szybko jak pod Windows, ponieważ nie ma tu miejsca emulacja w rozumieniu

interpretacji poleceń dla procesora. W celu uruchomienia określonego programu pod GNU/Linux przy pomocy WINE wymagane są teoretycznie takie same zasoby systemowe, jak pod Windows. Opcjonalnie WINE może wykorzystywać istniejącą instalację Windows. Możliwe jest wtedy przejmowanie ustawień tej instalacji dla programów windowsowych i wykorzystywanie przez WINE oryginalnych części składowych Windows, które w WINE jeszcze nie są dostępne w wystarczającym stopniu.

Poniżej opisano, jak można zainstalować i skonfigurować WINE w systemie GNU/Linux. Punktem wyjścia była dystrybucja Debiana 2.2 (POTATO) i wersja WINE 20000614. Później treść została dopasowana do nowszych wersji WINE (200105xx). W przypadku stosowania innej dystrybucji Linuksa czy innej wersji WINE należy odpowiednio dopasować opisane postępowanie.

W międzyczasie ukazał się w Debianie perfekcyjnie dopasowany pakiet WINE „**wine**“ Ove Kaavena, który razem z „**winesetuptk**“ może być używany z dziecięcą łatwością.

3. Pakiet binarny czy kod źródłowy?

Większość dystrybucji Linuksa zawiera dzisiaj pakiety WINE. Są to binaria, zawierające WINE w formie umożliwiającej bezpośrednie uruchomienie. Aktualne wersje można również ściągnąć z różnych stron internetowych. Teoretycznie, po zainstalowaniu pakietu binarnego program powinien nadawać się do sensownej konfiguracji i natychmiastowego użycia. W rzeczywistości, w wielu przypadkach konieczna jest zmiana konfiguracji zainstalowanej z pakietem.

Z powodu szybkiego rozwoju WINE zaleca się obecnie stosowanie kodu źródłowego zamiast pakietów binarnych. Po zdobyciu kodu źródłowego należy go rozpakować, skonfigurować i skompilować. Powstaje wtedy plik binarny, dokładnie dopasowany do posiadanego systemu i oferujący dlatego wyższe prawdopodobieństwo optymalnych wyników, niż w przypadku pakietów binarnych, stworzonych dla inaczej skonfigurowanego systemu. Użycie kodu źródłowego daje poza tym możliwość stosunkowo łatwej aktualizacji programu, przy czym nie zawsze trzeba pobierać kompletnego pakietu. Ponadto, przez użycie aktualnego kodu źródłowego można stwierdzić, czy ewentualnie występujące błędy rzeczywiście istnieją w WINE, i czy nie zostały już usunięte. Przez to zwiększa się prawdopodobieństwo wysyłania sensownych raportów błędów do twórców WINE.

Instalacja pakietów binarnych zależy od systemu pakietów dystrybucji (najczęściej jest to **rpm** lub **deb**), jak również od samej dystrybucji. Informacje wymagane do instalacji powinny znajdować się w dokumentacji używanej dystrybucji. W niniejszym opracowaniu opisana jest instalacja WINE z kodu źródłowego.

4. Wymagania systemowe

Aby WINE mógł być skompilowany i uruchomiony, w systemie muszą być zainstalowane następujące programy i pliki:

Jądro Linuksa od wersji 2.2.x. (WINE można uruchomić także z jądrem w wersji 2.0.x, jednak te jądra nie obsługują pewnych właściwości, wymaganych przez WINE. Jest to zauważalne szczególnie wtedy, jeśli przy pomocy WINE mają być uruchamiane 32 bitowe programy windowsowe, w których jednocześnie wykonywanych jest kilka wątków (*threads*)). Numer wersji aktualnie używanego jądra wyświetlany jest po wpisaniu w konsoli polecenia:

```
uname -r
```

Zaleca się stosowanie biblioteki C GNU (**libc6**), w wersji od 2.1. Numer wersji aktualnie używanej biblioteki C może być wyświetlony po wpisaniu następującego polecenia:

```
ls -l /lib/libc.so.*
```

W niektórych systemach istnieją zarówno starsze biblioteki C **libc5**, jak i nowsze **libc6**. Decydująca jest z reguły nowsza wersja. Biblioteka C musi być współużytkowalna (*reentrable*), aby WINE mógł obsługiwać wielowątkowość (*multithreading*). Ma to miejsce w wszystkich nowszych dystrybucji Linuksa. Biblioteka C znajduje się w pakiecie **libc6**. Do kompilacji WINE wymagane są dodatkowo narzędzia programistyczne do biblioteki C. W Debianie znajdują się one w pakiecie **libc6-dev**.

Ponieważ WINE wykorzystuje X Window, wymagane są biblioteki **X**, i jeśli WINE ma być kompilowane, narzędzia programistyczne dla X. W Debianie biblioteki te zawiera pakiet **xlib6g** a narzędzia programistyczne pakiet **xlib6g-dev**.

Dla programów, które mają być uruchamiane w trybie tekstowym, WINE może wykorzystywać bibliotekę **libncurses**. Aby wkompilować w program obsługę dla nich, muszą być zainstalowane narzędzia programistyczne dla tej biblioteki (pakiet **libncurses5-dev**). Zastosowanie biblioteki **ncurses** jest jednak opcjonalne.

Również opcjonalna jest obsługa biblioteki kompatybilnej z **OpenGL**, jak np. **Mesa**. Jeśli w program ma być wkompilowana obsługa OpenGL, w systemie muszą być zainstalowane narzędzia programistyczne OpenGL, np. udostępniane przez pakiet **mesag-dev**. Dodatkowo są oczywiście wymagane same biblioteki OpenGL.

Do kompilacji WINE należy użyć kompilatora C GNU. Obecnie zaleca się wersję 2.95. Ponadto potrzebnych jest kilka standardowych narzędzi, jak **make**, **(f)lex**, **yacc** czy **bison**, które w większości systemów linuksowych powinny być już zainstalowane.

Zależnie od tego, czy w kodzie wykonywalnym, który ma być otrzymany, mają być zawarte informacje diagnostyczne (*debugging informations*), do kompilacji i instalacji WINE potrzeba między ok. 100 MB a 250 MB przestrzeni na dysku. Odnośnie procesora komputera nie stawia się żadnych specjalnych wymagań, wystarczający jest procesor klasy Pentium 133 MHz do uruchamiania np. edytorów tekstu. Jednak dla gier i innych aplikacji multimedialnych wymagany jest z reguły szybszy procesor. Ważne jest, aby komputer miał wystarczającą ilość pamięci operacyjnej (RAM). Do uruchamiania większych programów windowsowych komputer powinien być wyposażony w 64 MB RAM.

5. Zdobycie i instalacja kodu źródłowego

WINE można pobrać z Internetu z różnych serwerów, przez FTP albo HTTP. Aktualny kod źródłowy można znaleźć między innymi pod następującymi adresami:

* **ftp://metalab.unc.edu/pub/Linux/ALPHA/wine/development/**
* **ftp://ftp.infomagic.com/pub/mirrors/linux/sunsite/ALPHA/wine/development/**
* **ftp://orcus.progsoc.uts.edu.au/pub/wine/development/**
* **http://metalab.unc.edu/pub/Linux/ALPHA/wine/development/**

Przy rozwijaniu WINE nie stosuje się obecnie numerów wersji. Zamiast tego każde wydanie oznaczone jest liczbą, odpowiadającą dacie wydania danej wersji, wg schematu: rok, miesiąc, dzień.

Plik **Wine-20000614.tar.gz** z jednego z wyżej wymienionych adresów zawiera więc wersję WINE, wydaną 14 czerwca 2000. Zasadniczo zaleca się stosowanie najnowszej wersji. Po pobraniu kodu źródłowego, można go rozpakować do katalogu bieżącego przez wydanie następującego polecenia:

```
tar -xvzf Wine-20000614.tar.gz
```

Przy tym **Wine-20000614.tar.gz** należy naturalnie zastąpić aktualną nazwą ściągniętego pliku. Kod źródłowy rozpakowywany jest do podkatalogu katalogu bieżącego, którego nazwa składa się z słowa **wine** i, oddzielonej kreską, daty wydania danej wersji, a więc na przykład: **wine-20000614**. Zaleca się zmianę nazwy tego katalogu na **wine**, co można wykonać przy pomocy polecenia:

```
mv wine-20000614 wine
```

5.1 Aktualizacja kodu źródłowego za pomocą łatek

Obok skompresowanych archiwów **.tar**, zawierających kod źródłowy WINE, pod wymienionymi adresami znajdują się również tzw. pliki łatek, zawierające jedynie zmiany, które zostały opracowane pomiędzy poszczególnymi dwoma wydaniami WINE. Pliki te są normalnie o wiele mniejsze niż kompletny kod źródłowy, tak że poleca się ich stosowanie, przy aktualizacji programu z jednej wersji do następnej. Jeśli zainstalowany jest np. **Wine-20000526**, i ma być zaktualizowany do **Wine-20000614**, należałoby pobrać plik **WINE-20000614.diff.gz**. Zmiany opisane w tym pliku można zastosować do zainstalowanego kodu źródłowego; w tym celu najpierw należy przejść do katalogu zawierającego kod źródłowy (a więc do katalogu **wine**, założonego w wyżej opisany sposób) a następnie wywołać program aktualizujący **patch**:

```
gunzip -c ../Wine-20000614.diff.gz | patch -p1
```

Zakłada się przy tym, że plik łatki znajduje się w katalogu nadrzędnym do katalogu zawierającego WINE (**wine**), i że nosi on nazwę **Wine-20000614.diff.gz**. Nazwę pliku w podanym poleceniu wzgl. ścieżkę należy odpowiednio dopasować, jeśli używany jest plik o innej nazwie lub aktualizację przeprowadza się z innego katalogu.

5.2 Pobieranie i aktualizacja WINE przy pomocy CVS

Alternatywnie można zainstalować kod źródłowy z serwera CVS projektu WINE. Zaletą tego sposobu jest, że w każdej chwili jest możliwe nieskomplikowane dopasowanie własnego kodu źródłowego do stanu rozwoju projektu, bez oczekiwania na nowe wydanie programu. Dla każdej osoby planującej współpracę przy projekcie zastosowanie CVS jest konieczne. Aby móc używać CVS należy oczywiście zainstalować program **cvs**. W Debianie jest on zawarty w pakiecie o tej samej nazwie. Po sprawdzeniu, że program jest zainstalowany, przy pomocy zmiennej środowiskowej **CVSROOT** można określić, skąd ma być pobierany wzgl. aktualizowany kod źródłowy. W przypadku stosowania powłoki **bash** można w tym celu wprowadzić następujące polecenie:

```
export CVSROOT=:pserver:cvs@cvs.winehq.com:/home/wine
```

Następnie można się zarejestrować na serwerze CVS. Do tego celu służy polecenie:

```
cvs login
```

Program pyta o hasło dostępu do serwera. Należy użyć hasło **cvs**. Potem można pobrać kod źródłowy z serwera, wydając następujące polecenie:

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

W katalogu bieżącym tworzony jest podkatalog o nazwie **wine**. Po zakończeniu wykonywania polecenia w katalogu tym znajdują się aktualny kod źródłowy projektu i kilka dodatkowych plików, wymaganych przez CVS.

W celu uaktualnienia stanu kodu źródłowego można użyć następującego polecenia:

```
cvs -z 3 update -PAd WINE
```

Informacje o używanych tu parametrach przy wywoływaniu CVS oraz o innych możliwościach tego programu znajdują się m.in. w podręczniku systemowym dla **cvs(1)** jak również na stronie internetowej CVS, pod adresem **http://www.sourcegear.com/CVS**. Dalsze ważne wskazówki odnośnie CVS i WINE dostępne są pod adresem **http://www.winehq.com/development/**.

6. Konfiguracja i kompilacja kodu źródłowego

Przy założeniu, że kod źródłowy znajduje się w podkatalogu **wine** katalogu bieżącego, należy najpierw przejść do tego podkatalogu, aby móc przeprowadzić dalsze działania:

```
cd wine
```

Następnie można wywołać skrypt **configure**. Skrypt ten przeprowadza szereg testów, sprawdzających m.in. czy system spełnia wszystkie konieczne warunki i czy zainstalowane są wymagane narzędzia programistyczne. Następnie tworzy on pliki, przez które sterowana jest kompilacja kodu źródłowego. Skrypt ten można uruchomić z różnymi parametrami, przy pomocy których można np. określić, że w tworzonych programach i bibliotekach nie mają być zawarte informacje diagnostyczne (*debug*). Pełna lista opcji dostępnych dla **configure** wyświetlana jest po wywołaniu skryptu z opcją **--help**. Normalnie wystarcza wywołanie tego skryptu w następujący sposób:

```
./configure
```

Jeśli **configure** nie może znaleźć ważnych danych czy właściwości systemu, może nastąpić komunikat błędu lub ostrzegawczy. Takie błędy należy usunąć przed kontynuacją kompilacji kodu źródłowego.

W następnym etapie odbywa się kompilacja kodu źródłowego. W tym celu należy użyć kolejno po sobie następujących poleceń:

```
make depend
make
```

Na partycji, na której znajduje się katalog z kodem źródłowym, wymagane jest tymczasowo około 230 MB przestrzeni dla kompletnej kompilacji. Większa część tej przestrzeni wymagana jest przez przy tym przez informacje

diagnostyczne (*debug*) w plikach wynikowych (*object files*), tworzonych podczas kompilacji. Jeśli nie zamierza się badać błędów w WINE, pliki binarne mogą być tworzone również bez informacji diagnostycznych; w tym celu drugie z wyżej wymienionych poleceń należy zastąpić poleceniem podanym niżej (wtedy do kompilacji wymagane będzie tylko około 80 MB pamięci dyskowej).

```
make CFLAGS="-O2"
```

Teraz WINE może zostać zainstalowany w systemie. W tym celu należy, z uprawnieniami administratora, wydać polecenie:

```
make install
```

W wyniku tego wykonywalne programy WINE zostają standardowo zainstalowane w katalogu **/usr/local/bin**, biblioteki programu w katalogu **/usr/local/lib**, strony podręcznika w katalogu **/usr/local/man**, a niektóre pliki nagłówkowe w katalogu **/usr/local/include/wine**.

Uwaga:
W niektórych dystrybucjach (np. w Debianie), system standardowo nie poszukuje bibliotek programów w katalogu **/usr/local/lib**. Jeśli przy uruchomieniu WINE otrzymuje się komunikat, że określone biblioteki nie mogą zostać załadowane, należy wpisać nazwę tego katalogu w pliku **/etc/ld.so.conf** (w oddzielnej linii) a potem wywołać program **ldconfig**.

7. Konfiguracja

Jak wiele programów uniksowych, WINE może być konfigurowany przez jeden plik konfiguracyjny, obowiązujący dla całego systemu, albo przez plik użytkownika w jego katalogu domowym. Plik konfiguracyjny użytkownika nosi nazwę **~/.wine/config**.

7.1 Budowa pliku konfiguracyjnego

Format pliku podobny jest do znanych z Windows plików ***.ini**, jednak jest trochę zmieniony, na format **Wine-Registry**. Plik składa się z pojedynczych bloków (sekcji), które rozpoczynają się od identyfikatorów, znajdujących się w nawiasach prostokątnych i oddzielnych liniach. Wewnątrz sekcji znajdują się pary zmiennych i wartości, połączone ze sobą znakiem równości. Pary te również znajdują się w oddzielnych liniach. Komentarze wprowadzane są do pliku przez znak średnika. Poza tym można używać pustych linii, dla podkreślenia struktury pliku. Przykładem takiej sekcji może być:

```
[Drive C]
"Path" = "/home/karol"
"Type" = "hd"
"Label" = "Dysk.C"
"Filesystem" = "win95"
```

Poza tym możliwe jest wykorzystywane w pliku konfiguracyjnym wartości zmiennych środowiskowych. W tym celu, w miejsce wartości należy wpisać nazwę zmiennej środowiskowej w nawiasie klamrowym i poprzedzić ją znakiem dolara. Jeśli zmiennej **Path** z powyższego przykładu miałaby być przyporządkowana wartość, którą w chwili wykonywania WINE ma zmienna środowiskowa **HOME**, odpowiednia linia powinna być zapisana następująco:

```
"Path" = "${HOME}"
```

W katalogu głównym kodu źródłowego WINE, w pliku **documentation/samples/config**, znajduje się przykład, mogący służyć jako szablon do stworzenia własnego pliku konfiguracyjnego. Plik ten zawiera wszystkie ważne sekcje i zmienne, musi być jednak dopasowany do własnej konfiguracji użytkownika, przed pierwszym użyciem WINE. Przy założeniu, że katalog kodu źródłowego WINE ma nazwę **wine** i jest podkatalogiem katalogu domowego, szablon ten może zostać skopiowany na właściwe miejsce przy pomocy następującego polecenia:

```
cp ~/wine/documentation/samples/config ~/.wine/config
```

Wartości, które zostają przyporządkowane zmiennym w pliku konfiguracyjnym, można podzielić na trzy typy: łańcuchy znaków, liczby i wartości logiczne. W przypadku wartości logicznych można wprowadzać **true** albo **false**, **1** albo **0** względnie **yes** albo **no**. W poniższych przykładach stosuje się zapis **true** / **false**.

7.2 Konfiguracja liter napędów

Między Uniksem / Linuksem z jednej strony a DOS i Windows z drugiej ma miejsce parę różnic w sposobie oznaczania nośników danych i plików. W Uniksie / Linuksie istnieje system plików z jednym katalogiem głównym – **root** (**/**), z którym związane są różne nośniki danych przez specjalne polecenie (**mount**). Wszystkie pliki na związanych nośnikach danych mogą być dlatego używane w obrębie tego systemu plików. DOS i Windows stosują jednak dla każdego rozpoznanego nośnika danych własny system plików. W celu jednoznacznego określenia danego pliku w tych systemach jest konieczne podawanie, obok ścieżki dostępu i nazwy pliku, tak zwanej litery napędu. Przeważnie litera napędu **A** odpowiada pierwszemu napędowi dyskietek a litera **C** pierwszej partycji dysku twardego systemu.

Ponieważ programy, które są napisane dla DOS czy Windows, używają liter napędów do określania plików, WINE musi odwzorować te litery w uniksowym systemie plików. Problem został rozwiązany w następujący sposób: w pliku konfiguracyjnym (**~/.wine/config**) każdej literze napędu przyporządkowuje się katalog w uniksowym systemie plików. Katalog ten przedstawia (z punktu widzenia programów windowsowych) katalog główny odpowiedniego napędu. Jeśli więc przykładowo katalog **/var/winroot** został przyporządkowany literze napędu **C** i program windowsowy pod WINE będzie próbował otworzyć plik **C:\Dokumenty\urz-skarbowy.doc**, w rzeczywistości zostanie otwarty plik **/var/winroot/Dokumenty/urz-skarbowy.doc**, pod warunkiem, że ten plik istnieje. W wyniku tego mechanizmu można również osiągnąć to, że z programów windowsowych, które uruchamiane są pod WINE, można mieć dostęp tylko do części uniksowego systemu plików.

Dalsza różnica między systemami plików DOS / Windows a Unix / Linux polega na uwzględnianiu dużych i małych liter w nazwach plików. Podczas gdy w Linuksie jest całkowicie możliwe, że w jednym katalogu jednocześnie znajdują się pliki z nazwami **list.txt**, **List.txt** i **list.TXT**, w Windows jest to wykluczone; ponadto może zostać otwarty plik np. **list.txt**, o ile istnieje, ale w rzeczywistości mógł być wywoływany **List.txt**. Ponadto, większość programów napisanych dla DOS czy Windows 16 bit, oczekuje, że nazwy plików nie będą dłuższe niż 8 znaków plus 3 znaki rozszerzenia. Z tego powodu WINE musi zadecydować, który plik powinien zostać w rzeczywistości otwarty, jeśli istnieje kilka możliwości w związku z istnieniem dużych i małych liter w nazwach, poza tym musi zmieniać nazwy plików na 8-znakowe, jeśli pliki wywoływane są przez programy 16-bitowe.

W przypadku, gdy WINE ma być używany z istniejącą instalacją Windows, należy zwracać uwagę na to, aby litery napędów zgadzały się pod Windows i WINE. Jeśli więc instalacja Windows znajduje się na przykład na partycji **/dev/hda1**, która pod Windows określana jest jako dysk **C**, pod GNU / Linux partycja ta powinna znajdować się w dowolnym katalogu a katalog ten powinien być w pliku konfiguracyjnym WINE przyporządkowany znowu literze napędu **C**, bo w przeciwnym przypadku oczywiście może dojść do konfliktów z programami już zainstalowanymi z istniejącą konfiguracją.

Przykład przyporządkowania katalogów uniksowych i liter napędów w pliku konfiguracyjnym przytoczono już wyżej. Taka definicja składa się z sekcji, której nazwę tworzy się z słowa kluczowego **Drive** i litery napędu, dla którego ma obowiązywać ta definicja. Przykładem byłoby więc **[Drive C]**. Potem następują różne zmienne, przez które określane są właściwości napędu. Najważniejszą z tych zmiennych jest **Path**. Przy jej pomocy określa się, któremu katalogowi uniksowemu ma odpowiadać ten napęd (przykłady: **"Path" = "/home/karol"**, **"Path" = "${HOME}"**). Dalsze zmienne mają następujące znaczenia:

Type
Windows może informować aplikacje, jakiego typu jest określony nośnik danych (dysk twardy, CD-ROM, itd.). Przy pomocy tej zmiennej informuje się WINE, jakiego typu powinien być dany napęd. Możliwymi wartościami są: **floppy** (napęd dyskietki), **hd** (partycja dysku twardego), **cdrom** (napęd DC-ROM) i **network** (sieć). Ogólnie zaleca się podawanie typu, który odpowiada katalogowi uniksowemu, przyporządkowanemu danemu napędowi.
Przykład: **"Type" = "floppy"**.

Label
Pod DOS i Windows napędy mogą mieć tak zwane etykiety. Nazwa ta może być wymagana przez aplikacje windowsowe. Przy pomocy tej zmiennej można określić, jaką nazwę nośnika ma zwrócić WINE, w przypadku żądania nazwy napędu przez aplikację. Nazwa nośnika (etykieta) nie może być dłuższa niż 11 znaków.
Przykład: **"Label" = "Dysk1"**.

Serial
Każdy napęd ma pod Windows tak zwany numer seryjny, który również jest żądany przez programy windowsowe. Zmienna ta pozwala określić w formie ośmiopozycyjnej liczby szesnastkowej, jaki numer seryjny ma być zwracany w takim przypadku.
Przykład: **"Serial" = "23f78a6b"**.

Filesystem
Ta zmienna określa, jakie właściwości ma mieć emulowany system plików na danym napędzie. Możliwe są następujące wartości:

> **msdos**
> Na takim napędzie dopuszczalne są tylko nazwy plików o długości 8 znaków i z rozszerzeniem składającym się z 3 znaków. Nie są uwzględniane różnice między dużymi a małymi literami. Alternatywnie dla **msdos** mogą być używane określenia **dos** lub **fat** dla tego typu systemu plików.
>
> **win95**
> Napęd ten dopuszcza długie nazwy plików. DOS i 16-bitowe programy windowsowe mogą jednak nadal używać krótkich nazw plików. Nie są uwzględniane różnice między dużymi a małymi literami. Jest to ustawienie zalecane dla prawie wszystkich aplikacji. Alternatywnie dla **win95** typ ten może być określany również jako **vfat**.
>
> **unix**
> Ten system plików na napędzie zachowuje się podobnie jak typowy uniksowy system plików, tzn. nazwy plików mogą mieć normalnie dozwoloną długość i rozróżniane są duże i małe litery. Z ustawieniem tym nie radzi sobie większość programów windowsowych. Problemy występują na przykład wtedy, gdy program windowsowy najpierw zapisze plik pod nazwą **Dane** a potem będzie chciał go otworzyć pod nazwą **DANE**.

Uwaga:
Należy uwzględnić, że przez to ustawienie nie określa się, jakie właściwości ma będący podstawą uniksowy system plików, lecz jakie właściwości mają być emulowane przez WINE dla odpowiedniego napędu. A więc jest całkiem możliwe (a najczęściej wymagane) stosowanie ustawienia **win95** dla napędu, który znajduje się w uniksowym systemie plików. Jeśli jednak na nośniku danych, na którym znajduje się katalog przyporządkowany temu napędowi, istnieje system plików FAT, obsługiwany przez linuksowy sterownik FAT (a nie, jak to jest powszechne, przez sterownik VFAT), musi być używany system plików **msdos**, ponieważ w przeciwnym przypadku może zdarzyć się, że WINE będzie próbował na tym nośniku zapisywać o długich nazwach, co doprowadziłoby do błędu.
Przykład: **"Filesystem" = "win95"**.

Device
W szczególnych przypadkach konieczne jest, aby programy windowsowe bezpośrednio, a więc z obejściem systemu plików, zapisywały czy odczytywały z nośnika danych. Żeby było to możliwe także przy pomocy WINE, można tu podać nazwę pliku urządzenia, która reprezentuje ten nośnik pod Linuksem. Jest to sensowne tylko wtedy, jeśli katalog przyporządkowany danemu napędowi odpowiada punktowi montowania podanego tu nośnika danych. Bezpośredni dostęp do urządzenia powinien być normalnie dozwolony tylko dla takich nośników danych, których zawartość nie musi być specjalnie chroniona (ew. dyskietki), albo z których i tak możliwy jest tylko odczyt (np. CD-ROMy). Aby mógł być możliwy zapis na tym nośniku, jest oczywiście dodatkowo konieczne posiadanie wystarczających praw do danego pliku urządzenia.
Przykład: **"Device" = "/dev/fd0"**.

FailReadOnly
Szereg programów windowsowych otwiera z zasady pliki do odczytu i zapisu, także jeśli ma nastąpić jedynie odczyt z danego pliku. Zachowanie to prowadzi zazwyczaj do tego, że pliki, w których nie wolno zapisywać z WINE, albo które znajdują się na nośnikach, na których nie można zapisywać (jak CD-ROM), nie mogą być otwierane. Z tego powodu WINE otwiera pliki standardowo do odczytu, jeśli plik nie został otwarty do odczytu i do zapisu. Jeśli zmienna **FailReadOnly** ustawiona jest na wartość **true**, WINE zachowuje się tak, jak to jest powszechne w Uniksie i wysyła do programu windowsowego komunikat błędu, jeśli plik nie może zostać otwarty do zapisu. Z reguły zaleca się przejęcie ustawienia standardowego.
Przykład: **"FailReadOnly" = "true"**.

ReadVolInfo
Jeśli wartość tej zmiennej ustawiona jest na **true**, WINE próbuje odczytać numer seryjny i nazwę (etykietę) nośnika danych danego napędu, bezpośrednio z tego nośnika. W tym celu musi być przyporządkowany napędowi plik urządzenia (**Variable device**). Ustawienie to jest sensowne przede wszystkim dla takich programów, które działają tylko wtedy, gdy przykładowo w napędzie znajduje się właściwy CD-ROM i stwierdzają to w oparciu o nr seryjny lub nazwę (etykietę) nośnika.
Przykład: "**ReadVolInfo" = "true**"

7.3 Ustawienia ogólne

W sekcji **[wine]** pliku konfiguracyjnego przeprowadza się najważniejsze ustawienia ogólne. W istocie chodzi tu o określenia katalogów. Należy uwzględnić, że te określenia katalogów muszą nastąpić w sposób normalny dla DOS i Windows. Oznacza to, że przed każdym katalogiem należy podać literę napędu. Litera napędu i katalog oddzielone są dwukropkiem, poza tym poszczególne katalogi nie są oddzielane od siebie normalnym ukośnikiem lecz ukośnikiem odwrotnym (*backslash*). Dane te są tłumaczone na uniksowe nazwy plików przez przyporządkowanie liter napędów, opisane w poprzednim rozdziale.

Windows
W systemie Windows katalog **Windows** odgrywa szczególną rolę. Programy zapisują w nim często pliki inicjujące a programy instalacyjne kopiują niekiedy różne dane do tego katalogu. Przy pomocy zmiennej **Windows** określa się, który katalog ma być traktowany przez programy uruchamiane pod WINE jako katalog **Windows**. Podany tu katalog musi istnieć, zanim WINE zostanie uruchomiony po raz pierwszy. Jeśli WINE ma używać istniejącej instalacji Windows, należy podać tu katalog, w którym znajduje się ta instalacja.

Jeśli WINE ma być używany z istniejącą instalacją Windows i instalacja ta znajduje się na partycji dysku twardego, która pod Windows oznaczona jest literą napędu **C:** a pod Linuksem reprezentowana jest przez plik urządzenia **/dev/hda1**, partycja ta może być pod Linuksem połączona z katalogiem **/Windows**. Wtedy katalogowi temu, w sekcji dla konfiguracji liter napędów, należałoby przyporządkować literę napędu **C:**, jak w przykładzie:

```
[Drive C]
"Path" = "/Windows"
"Type" = "hd"
"Label" = "windows"
"Filesystem" = "win95"
```

Dalej, jeśli nazwa katalogu Windows tej instalacji brzmi **windows** (a więc w Windows **C:\windows** i w Linuksie **/Windows/windows**), w sekcji **[wine]** pliku konfiguracyjnego należałoby przeprowadzić następujące ustawienie:

```
"Windows" = "C:\\Windows"
```

W przypadku, gdy WINE miałby być jednak używany bez istniejącej instalacji Windows, jako katalog główny dla tego napędu może służyć dowolny katalog, zawierający katalog Windows, na przykład **/Windows**. W tym katalogu należy założyć katalog **windows**, który, jak opisano wyżej, w sekcji **[wine]**, musi być zadeklarowany jako katalog systemowy Windows.

System
Katalog **System** ma podobne znaczenie, jak katalog **Windows**. W Windows znajdują się w tym katalogu w zasadzie biblioteki programów; standardowo jest to podkatalog katalogu Windows. Katalog ten musi również istnieć, zanim WINE zostanie uruchomiony po raz pierwszy. Także tu musi być podany katalog **System** istniejącej instalacji Windows, jeśli taka instalacja ma być używana. W Windows 95 / 98 katalog ten ma standardowo nazwę **system**, a w Windows NT nazwę **system32**.
Przykład: **"System" = "C:\\Windows\\System"**.

Temp
Katalog **Temp** jest używany przez wiele programów windowsowych do zapisu danych tymczasowych. Aby mogło się to odbyć, należy podać tu katalog, znajdujący się w napędzie, odpowiadającym katalogowi uniksowemu z prawem do zapisu.
Przykład: **"Temp" = "D:\\tmp"**.

Path
Zmienna ta ma to samo znaczenie co zmienna środowiskowa **PATH** w Uniksie. Jej wartość składa się z łańcucha poszczególnych nazw katalogów, przeszukiwanych pod kątem uruchamianych programów, jeśli nazwa takiego programu zostanie podana bez nazwy katalogu (ścieżki dostępu). Należy zwrócić uwagę, że poszczególne elementy tej zmiennej pod Windows nie są oddzielane dwukropkiem tylko średnikiem; poza tym wiele programów windowsowych oczekuje, że w tej zmiennej zawarte są katalogi **Windows** i **System**.
Przykład: **"Path" = "C:\\Windows;C:\\Windows\\System;D:\\Winstuff"**.

Profile
Zmienna ta jest używana przez WINE do załadowania części rejestru systemowego dotyczącej użytkownika, z istniejącej instalacji Windows. Jeśli chodzi tu o istniejącą instalację Windows 95/98, która nie została skonfigurowana dla wielu użytkowników, albo w przypadku, gdy WINE ma pracować bez istniejącej instalacji Windows, zmienna

Profile nie musi być określana. Jeśli jednak ma być używana instalacja Windows NT albo Windows 95/98 z wieloma użytkownikami, należy tu określić, z którego katalogu WINE ma załadować dane z rejestru, dotyczące użytkownika. Katalogi te znajdują się zazwyczaj w podkatalogu **Profiles** katalogu **Windows** i noszą nazwy użytkowników, których konfiguracje zawierają.
Przykład: **"Profile" = "C:\\Windows\\Profiles\\Anna"**.

GraphicsDriver
WINE może stosować różne sterowniki do wyświetlania graficznego. Przy pomocy tej zmiennej określa się sterownik, który ma być używany. Chwilowo do dyspozycji są dwa sterowniki, a mianowicie **x11drv** dla systemu **X Window** i **ttydrv** do stosowania WINE na konsoli. Sterownik **ttydrv** obecnie nie jest w pełni funkcjonalny, dlatego zaleca się stosowanie tylko sterownika **x11drv**; jest to również ustawienie standardowe, gdy ta zmienna nie jest określona.
Przykład: **"GraphicsDriver" = "x11drv"**.

7.4 Konfiguracja bibliotek

Tak jak programy uniksowe czy linuksowe, programy windowsowe składają się z reguły z właściwego pliku programu i szeregu bibliotek programowych, które łączone są z programem przy jego ładowaniu lub później. Wiele bibliotek programowych pod Windows stanowi jednocześnie interfejs dla systemu operacyjnego. Obok właściwych programów windowsowych wymagane są więc te biblioteki dla umożliwienia pracy programu.

WINE udostępnia dużą ilość bibliotek normalnie dostępnych w Windows. Biblioteki dostępne są albo w postaci oddzielnych plików, standardowo w katalogu **/usr/local/lib**, albo są zawarte bezpośrednio w pliku programu. WINE jest również w stanie używać normalnych bibliotek Windows; jest to konieczne np. w sytuacji, gdy przez program wymagana jest biblioteka, nie będąca standardową biblioteką Windows, lecz dodawana do systemu podczas instalacji danego programu. Takie biblioteki nie są normalnie dostarczane przez WINE.

Jeśli WINE używany jest z istniejącą instalacją Windows, można ewentualnie w niektórych przypadkach używać bibliotek z instalacji Windows, zamiast dostarczanych przez WINE. Są one w wielu przypadkach bardziej kompletne i dlatego mogą uruchamianemu programowi lepiej zapewnić oczekiwaną funkcjonalność. Należy przy tym jednak zwrócić uwagę na to, że możliwe jest to tylko w przypadku takich bibliotek, które nie zawierają funkcji systemowych. Natomiast biblioteki zawierające jedynie zwykły kod programu, jak np. często wymagane dialogi, mogą być bezproblemowo używane z istniejącej instalacji Windows.

W Windows 95/98 większość bibliotek jest do dyspozycji w różnych wersjach: wersjach 32-bitowych, które mogą być ładowane przez programy 32-bitowe, i w wersjach 16-bitowych, które mogą być używane przez programy 16-bitowe. Obie wersje używają z reguły kodu programu z przynależnej drugiej wersji (pod Windows 95/98 właściwe funkcje znajdują się przeważnie w bibliotekach 16-bitowych, które są ładowane i wywoływane przez wersje 32-bitwe). Dlatego konieczne jest ładowanie zawsze obu wersji biblioteki jako biblioteki Windows albo WINE, inne ustawienia prowadzą z reguły do błędów natychmiast po wywołaniu WINE. Poniższe zestawienie wskazuje, które najważniejsze biblioteki 16- i 32-bitowe są wspólne, i daje informacje, czy te biblioteki mogą być ładowane z istniejącej instalacji Windows, czy też muszą być koniecznie dostarczane przez WINE. W zestawieniu najpierw wymieniana jest biblioteka 16-bitowa a następnie 32-bitowa.

krnl386
kernel32
Ta biblioteka udostępnia interfejs dla podstawowych funkcji systemów operacyjnych Windows, jak dostęp do plików, wejście i wyjście czy synchronizacja procesów. Dlatego tu nie mogą być używane biblioteki z instalacji Windows.

-
ntdll
Ta biblioteka zawiera interfejs dla systemu operacyjnego Windows NT. Dlatego musi być używana wersja WINE.

-
advapi32
Tu znajdują się m.in. funkcje dla dostępu do rejestru Windows jak również funkcje zabezpieczające i kryptograficzne. Z reguły zaleca się stosowanie wersji WINE tej biblioteki.

winsock
wsock32
Tu znajduje się interfejs Protokołu Internetowego (*Internet Protokoll*) IP z Windows. W czasie pracy z WINE wykorzystywana jest funkcjonalność IP systemu linuksowego, tak więc tu muszą być używane wersje WINE tych bibliotek. Wywołania IP programów windowsowych zostają przez nie przekazane dalej do Linuksa.

gdi
gdi32
GDI odpowiada za *Graphics Device Interface*. Biblioteka ta stwarza jednolity interfejs do wyświetlania na ekranie i dla drukarek. Także tu muszą być używane wersje WINE..

user
user32
User udostępnia m.in. funkcje dla zarządzania oknami, dla menu albo do obsługi schowka. Wersje tych bibliotek z Windows 95/98 mogły być wcześniej używane z WINE, pod pewnymi warunkami. Biblioteki **USER** z Windows NT wywołują z reguły funkcje w jądrze (*kernel*) NT i dlatego nie mogą być używane z WINE. Zaleca się stosowanie bibliotek **user** z WINE.

lzexpand
lz32
Obie te biblioteki udostępniają funkcje do dekompresji archiwów **LZ**. Takie funkcje są wymagane głownie przez programy instalacyjne. Windowsowe wersje tych bibliotek używają niektórych funkcji z biblioteki jądra (*kernel*), które obecnie nie są zaimplementowane w WINE. Dlatego muszą być używane wersje dostarczane przez WINE.

commctrl
comctl32
Ta biblioteka (*common controls*) udostępnia funkcje do tworzenie często używanych elementów okien, jak listwy narzędziowe czy wskazania (paski) stanu. Może być używana zarówno wersja z WINE jak i z Windows.

commdlg
comdlg32
Tu znajdują się kompletne dialogi, często używane przez programy windowsowe. (wybór koloru, czcionki, wyszukiwanie i zastępowanie itd.). Także tu można używać do wyboru – wersję z Windows i z WINE.

shell
shell32
Ta biblioteka powłoki zawiera większą część windowsowych interfejsów użytkownika. Jest ona wykorzystywana szczególnie przez Explorera (managera plików) i wiele innych aplikacji, które obsługują funkcje jak np. „przeciągnij i upuść“ (*Drag-and-Drop*). W zasadzie można używać zarówno wersję windowsową jak i z WINE.

-
crtdll
To jest standardowa biblioteka C czasu rzeczywistego z Windows. Chwilowo wersja windowsowa jest bardziej kompletna i dlatego niektóre programy nie działają prawidłowo z wersją z WINE.

Obok wymienionych tu najważniejszych bibliotek systemowych WINE udostępnia szereg dalszych bibliotek, na przykład do obsługi aplikacji multimedialnych. Jeśli do dyspozycji jest instalacja Windows, w wielu przypadkach zaleca się wypróbowanie, czy program działa lepiej z wersją WINE, czy z windowsową tych bibliotek.

W pliku **~/.wine/config** znajdują się dwie sekcje, przy pomocy których określa się, które biblioteki mają być ładowane z instalacji Windows. Poza tym, ustawienia te mogą być nadpisane przy wywołaniu WINE z konsoli. Ogólnie zaleca się przejęcie ustawień z przykładowej wersji pliku konfiguracyjnego (**documentation/samples/config**), i zmianę ich tylko w przypadku, gdy określone programy nie działają prawidłowo z tymi ustawieniami wstępnymi. Jeśli mają być używane biblioteki z instalacji Windows, należy zwrócić uwagę na to, żeby mogły być znalezione przez WINE. Wymaga to z reguły wskazania na katalog **windows** względnie **system** istniejącej instalacji Windows przy pomocy zmiennych **Windows** i **System** w ogólnej części konfiguracji oraz wymienienia obu tych katalogów w wartości zmiennej **Path**.

W sekcji **[DllDefaults]** można przeprowadzić następujące dwa ustawienia:

DefaultLoadOrder
Przy pomocy tego ustawienia określa się, jaka ma być standardowa kolejność prób załadowania biblioteki przez WINE. Kolejność ta jest definiowana przez następujące słowa kluczowe:

> **native**
> WINE ma próbować załadowania windowsowej wersji danej biblioteki.
>
> **builtin**
> WINE ma próbować załadowania własnej wersji danej biblioteki.

elfdll
biblioteki **lfdll** są szczególną formą bibliotek windowsowych, udostępnianych przez WINE. Pierwotnie były planowane do zastąpienia wbudowanych bibliotek, jednak z biegiem czasu wbudowane biblioteki otrzymały wiele funkcji planowanych dla e**lfdll**, tak że ta forma bibliotek obecnie nie ma specjalnego znaczenia.

so
W niektórych przypadkach istnieją linuksowe i windowsowe wersje bibliotek, nie różniące się zarówno pod względem działania jak i możliwych do wywołania funkcji w bibliotece. Jeśli program windowsowy chce załadować windowsową wersję takiej biblioteki, WINE może próbować zamiast niej załadować wersję linuksową i wywoływać jej funkcje zamiast funkcji w wersji windowsowej. Przy pomocy słowa kluczowego **so** można ustawić takie zachowanie.

Przez kolejność, z jaką te słowa kluczowe zostały przyporządkowane zmiennej **DefaultLoadOrder** określa się, w jakiej kolejności WINE ma używać poszczególnych strategii. Jeśli, na przykład, wprowadzono **"DefaultLoadOrder" = "native, builtin, so"**, gdy dana biblioteka musi być załadowana WINE próbuje najpierw załadować bibliotekę windowsową. Jeśli się to nie uda (z reguły gdy tej biblioteki nie ma lub nie można jej znaleźć), WINE próbuje użyć swojej biblioteki, i jeśli również takiej nie ma, próbuje bezpośrednio załadować jej wersję uniksową.

W sekcji **[DllOverrides]** można z powrotem zmienić określone w poprzedniej sekcji zachowanie dla poszczególnych bibliotek. Podczas gdy ogólnie strategia próby załadowania w pierwszej kolejności windowsowej wersji biblioteki jest dobra, dla określonych bibliotek nie może to mieć miejsca w żadnym wypadku (jak opisano wyżej) i muszą być używane wersje z WINE. Jako zmienne wymienione są w tej sekcji te biblioteki, dla których musi obowiązywać specjalna kolejność. Za znakiem równości podawana jest kolejność wymagana dla danych bibliotek, tak jak to opisano w poprzedniej części.
Przykład:

```
[DllOverrides]
"krnl386, kernel32" = "builtin"
"gdi, gdi32"        = "builtin"
"user, user32"      = "builtin"
"shell,shell32"     = "native, builtin"
```

7.5 Konfiguracja sterownika graficznego X11

Sterownik graficzny **X11** jest interfejsem między funkcjami graficznymi systemów windowsowych a systemem X Window. Umożliwia używanie przez programy windowsowe pod WINE systemu X Window podobnie, jak używają one w Windows normalnej karty graficznej. Zachowanie się tego sterownika ustawiane jest w sekcji **x11drv** pliku konfiguracyjnego. Do dyspozycji są następujące zmienne:

PrivateColorMap
Jeśli zmienna ta ustawiona jest na **true**, WINE używa własnej palety kolorów. W przypadku serwerów X, które obsługują głębię 256 kolorów (8 bit) lub mniejszą, konsekwencją tego jest, że inne okna mogą być ewentualnie wyświetlane w złych kolorach, gdy okna WINE są na pierwszym planie. Jednak z tym ustawieniem WINE może lepiej oddawać kolory, niż bez niego. W przypadku serwerów X obsługujących głębię kolorów większą niż 256, ustawienie to nie ma znaczenia.

AllocSystemColors
Jeśli WINE nie używa własnej palety kolorów, można tu ustawić, ile kolorów z palety systemowej może być używane przez WINE. Najwyższą możliwą wartością jest tu 256, gdyż przy lepszej głębi kolorów nie używa się palety.
Przykład: **"AllocSystemColors" = "100"**.

PerfectGraphics
W niektórych miejscach WINE ma możliwość wykonywania operacji graficznych albo w sposób zapewniający dużą szybkość, albo z wysoką dokładnością. Jeśli zmienna ta ustawiona jest na **true**, pierwszeństwo ma dokładność przedstawienia.
Przykład: **"PerfectGraphics" = "false"**.

UseDGA
DGA (*Direct Graphics Access*) – bezpośredni dostęp do grafiki – jest rozszerzeniem **Xfree86**, umożliwiającym bezpośredni dostęp do pamięci karty graficznej. W wyniku tego operacje graficzne mogą być przeprowadzane o wiele szybciej niż normalnie. Biblioteka **DirectDraw** (Direct X) z WINE może obsługiwać to rozszerzenie, przez co szczególnie gry mogą działać z podobną prędkością jak pod Windows. Jeśli DGA wymaga bezpośredniego dostępu

do sprzętu, z reguły konieczne są do tego uprawnienia administratora. Użycie DGA włączane jest przez ustawienie **"UseDGA" = "true"** a wyłączane – przez **"UseDGA" = "false"**.

Uwaga:
Aplikacje używające **DirectDraw** standardowo próbują przełączyć ekran na określoną rozdzielczość i głębię kolorów. WINE może wprawdzie zmieniać rozdzielczość, jeśli w pliku **XF86Config** znajduje się definicja wymaganego trybu; ponieważ jednak system X Window nie obsługuje zmiany głębi kolorów, często konieczne jest uruchomienie serwera X w wymaganej głębi kolorów przed uruchomieniem WINE.

UseXShm
Tu chodzi o inne rozszerzenie systemu X Window, które umożliwia szybsze operacje graficzne.
Przykład: **"UseXShm" = "true"**.

DXGrab
Opcja ta powoduje, że wskaźnik myszy – przy użyciu **DirectDraw** (DirectX) – nie może opuścić okna sterowanego przez **DirectDraw**. Jest to konieczne do prawidłowej obsługi niektórych programów, równocześnie zapobiega się uaktywnianiu innych okien myszą, np. aby zakończyć pracę WINE.
Przykład: **"DXGrab" = "false"**.

ScreenDepth
Niektóre serwery X obsługują różne głębie kolorów na tym samym ekranie. Przy pomocy tej zmiennej można wybrać, jaka głębia kolorów ma być używana w takim przypadku.

Managed
WINE może wyświetlać okna programów windowsowych niezależnie od stosowanego menedżera okien albo poddać je kontroli tego menedżera. Ta druga możliwość oferuje lepszą integrację programów windowsowych z środowiskiem roboczym Linuksa, ponieważ okna wyświetlane i obsługiwane są tak samo jak okna programów linuksowych. Który z dostępnych trybów wyświetlana ma być używany, podaje się zazwyczaj w poleceniu przy wywoływaniu WINE. Przy pomocy tej zmiennej można określić w pliku konfiguracyjnym, czy okna sterowane prze WINE mają standardowo używać menedżera okien (**"Managed" = "true"**).

DesktopDoubleBuffered
Opcja ta powinna być ustawiona na **true**, jeśli z WINE używane są programy wykorzystujące OpenGL. Przez to polepsza się obraz tych aplikacji.

7.6 Konfiguracja czcionek

Sterownik graficzny **X11 x11drv** do przedstawiania napisów używa czcionek, dostępnych dla serwera X albo bezpośrednio albo poprzez serwer czcionek. Szereg aplikacji windowsowych oczekuje jednak dokładnie określonych czcionek, z reguły dostępnych pod Windows, a w wielu aplikacjach okna i inne elementy wyświetlane są inaczej niż pod Windows, jeśli muszą być użyte inne czcionki.

Windows używa normalnie dwóch różnych typów czcionek: mianowicie tak zwanych czcionek TrueType i zwykłych czcionek bitmapowych. Oba typy czcionek standardowo nie są dostępne w **Xfree86** w wersjach 3.x. Jednak możliwe jest przekształcenie windowsowych czcionek bitmapowych do formatu, który może być połączony z **Xfree86**. Czcionki TrueType mogą być połączone poprzez specjalne serwery czcionek TrueType, które w międzyczasie zostały dołączone do większości dystrybucji. Jeśli jest do dyspozycji instalacja Windows, zaleca się ogólnie użycie znajdujących się tam czcionek również pod Linuksem, aby programy uruchamiane pod Windows mogły znaleźć wymagane przez siebie czcionki. Jeśli WINE jest używany bez czcionek windowsowych, próbuje zastąpić wymagane czcionki przez czcionki systemu X Window, przez co mogą wyniknąć zmiany w obrazie graficznym uruchamianych programów.

7.6.1 Dowiązanie czcionek bitmapowych

Windowsowe czcionki bitmapowe znajdują się normalnie w podkatalogu **fonts** katalogu **Windows** a ich nazwy mają rozszerzenie **.fon**. Możliwe jest jednak, że zawarte są również w plikach wykonywalnych albo w bibliotekach. W podkatalogu **tools** katalogu głównego z kodem źródłowym WINE znajduje się program **fnt2bdf**, przy pomocy którego czcionki można wyekstrahować z tych plików i przekształcić w format *Bitmap Distribution Format* (**.bdf**). W tym celu program ten należy wywołać następująco:

```
fnt2bdf -o nazwa_podstawowa plik_czcionki
```

Przy tym należy podać: dla **pliku_czcionki** – nazwę pliku, w którym znajdują się czcionki do wyekstrahowania, a dla **nazwy_podstawowej** – początkową część nazwy plików, które mają być ekstrahowane. Należy uwzględnić, że w windowsowym pliku czcionki z reguły znajduje się więcej czcionek, które są następnie ekstrahowane do różnych plików. W przypadku niektórych czcionek konieczne jest poinformowanie programu, jak kodowane są te czcionki. W tym celu należy użyć opcji **-c**, a następnie podać oznaczenie kodowania. Przegląd wszystkich obsługiwanych opcji programu zostanie wyświetlony, jeśli uruchomi się go bez żadnych parametrów. Aby, przykładowo, wyekstrahować czcionki bitmapowe z pliku **SERIFF.FON** do plików, których nazwy zaczynają się określeniem **seriff**, należy wywołać ten program następująco:

```
fnt2bdf -o seriff SERIFF.FON
```

Z pliku oryginalnego tworzony jest szereg plików z różnymi czcionkami. W następnym etapie pliki te muszą być przekształcone w tak zwany *Portable Compiled Format* przy pomocy programu **bdftopcf(1)**. Program ten powinien być częścią składową każdej instalacji X11 i mieć dokumentację w podręczniku systemowym. Aby przekształcić na przykład plik **seriff_r400-23.bdf** i zapisać wynik w pliku **seriff_r400-23.pcf**, program może zostać wywołany w taki sposób:

```
bdftopcf -o seriff_r400-23.pcf seriff_r400-23.bdf
```

Pliki tak utworzone mogą być następnie skopiowane do jednego z katalogów czcionek, używanych przez serwer X albo serwer czcionek. W tym katalogu należy następnie wykonać polecenie **mkfontdir**, aby na nowo został wykonany znajdujący się tam indeks czcionek. Zanim czcionki te będą rzeczywiście dostępne pod X i będą mogły być używane przez WINE, należy zrestartować X albo w konsoli wpisać następujące polecenie:

```
xset fp +rehash
```

W podkatalogu **tools** katalogu z kodem źródłowym WINE znajduje się skrypt powłoki o nazwie **font_convert.sh**, który automatycznie przeprowadza wymienione wyżej kroki i automatycznie konwertuje i instaluje wszystkie czcionki bitmapowe, znajdujące się w katalogu.

7.6.2 Dowiązanie czcionek TrueType

Jak już wspomniano, serwery X z **XFree86** w wersjach 3.x nie obsługują czcionek TrueType. Od **XFree86** 4.0 już się to zmieniło, tak że te czcionki w przyszłości będą mogły bezproblemowo być dostępne pod X – i w związku z tym także pod WINE. Niestety, protokół czcionek X (*X font protocol*) nie może dostarczyć wszystkich informacji odnośnie czcionek TrueType, wymaganych przez niektóre programy windowsowe, tak więc te programy nie pracują poprawnie także gdy dostępne są wymagane czcionki. W przypadku **XFree86** 3.x proponuje się zastosowanie serwera czcionek TrueType. Obecnie są do dyspozycji trzy różne takie programy, mianowicie: **xfs-xtt** (Debian: **xfs-xtt**), **xfstt** (Debian: **xfstt**) i **xfsft** (programy te są dostępne w Internecie pod adresem: **http://www.dcs.ed.ac.uk/home/jec/programs/xfsft/**).

Autor osiągnął najlepsze wyniki z programem **xfsft**. Serwer ten jest łatwy w konfiguracji i jest kompatybilny z powszechnie stosowanymi serwerami czcionek, tak że nie trzeba uruchamiać dwóch różnych serwerów czcionek na jednym komputerze. Program ten służył również jako podstawa dla integracji wsparcia dla czcionek TrueType w **XFree86** w wersji 4.0.

Aby używać czcionki udostępnione przez serwer czcionek, należy podać serwerowi X adres serwera czcionek. Można to określić w pliku **/etc/X11/XF86Config** albo przez wydanie poniższego polecenia:

```
xset fp+ tcp/localhost:7100
```

Przy tym trzeba ewentualnie zamienić **localhost** na nazwę komputera, na którym uruchomiony jest serwer czcionek, a **7100** na port, który używany jest przez serwer czcionek. Jeśli serwer czcionek uruchomiony jest na tym samym komputerze, co serwer X, do komunikacji między serwerem X a serwerem czcionek mogą być wykorzystane również uniksowe gniazda domeny (*UNIX Domain Sockets*). Odpowiednie polecenie może wyglądać następująco:

```
xset fp+ unix/:7100
```

7.6.3 Ustawienia czcionek w pliku konfiguracyjnym WINE

W pliku konfiguracyjnym **`~/.wine/config`** wzgl. **`wine.conf`** do dyspozycji są następujące zmienne, przy pomocy których można mieć wpływ na pracę WINE z czcionkami:

`Resolution`
Pod Windows programy mogą podawać wielkość czcionek, które mają być użyte, w punktach (zamiast w pikselach). Żeby czcionka mogła być wyświetlona w właściwej wielkości, musi być znana rzeczywista wielkość ekranu, co pod X niekoniecznie się zdarza. Przy pomocy zmiennej **`Resolution`** można dlatego ustawić, jakie wielkości czcionek mają być wybierane przez WINE w takich przypadkach. Wartością standardową jest 96, wartości sensowne leżą w zakresie między 60 a 120.
Przykład: **`"Resolution" = "100"`**.

`Default`
Przy pomocy tej zmiennej określa się, jaką czcionkę WINE powinien stosować jako standardową. Łańcuch znaków, który należy przy tym podać, składa się z nazwy producenta czcionki i jej nazwy; określenia te połączone są ze sobą znakiem minus (-), poza tym minus musi znajdować się na początku i na końcu. Czcionki dostępne pod X11 mogą być wybrane np. przy pomocy programów **`xfontsel(1)`** lub **`xlsfonts(1)`**.
Przykład: **`"Default" = "-adobe-times-"`**.

`DefaultFixed`
Ta zmienna określa, jaką czcionkę WINE powinien stosować jako standardową czcionkę o stałej szerokości. Nazwę żądanej czcionki należy podać tak jak w przypadku zmiennej **`Default`**.
Przykład: **`"DefaultFixed" = "-sony-fixed-"`**.

`DefaultSerif`
Ta zmienna określa, jaką czcionkę WINE powinien stosować jako standardową czcionkę szeryfową.
Przykład: **`"DefaultSerif" = "-adobe-times-"`**.

`DefaultSansSerif`
Ta zmienna określa, jaką czcionkę WINE powinien stosować jako standardową czcionkę bezszeryfową.
Przykład: **`"DefaultSansSerif" = "-adobe-helvetica-"`**.

`Fontmetric`
Gdy WINE uruchamiany jest po raz pierwszy, odczytuje z serwera X właściwości dostępnych czcionek. Ponieważ ta operacja jest długotrwała, wynik jej zapisywany jest w katalogu domowym danego użytkownika. Przy następnych uruchomieniach programu, odczyt tych właściwości musi być wykonany tylko w przypadku zmiany dostępnych czcionek. Przy pomocy tej zmiennej można określić, w jakim pliku mają być zapisane dane czcionek. Przez to unika się np. potrzeby przeprowadzania tego odczytu ponownie dla każdego użytkownika.
Przykład: **`"Fontmetric" = "/var/lib/WINE/font.cache"`**.

`Alias`
Jak już wyżej wspomniano, może się zdarzyć, że aplikacje windowsowe chcą użyć czcionek nieudostępnionych przez serwer X. Przy pomocy zmiennej **`Alias`** można przyporządkować nazwom takich czcionek inne czcionki, dostępne pod X. Wartości przyporządkowane tej zmiennej składają się z trzech elementów, oddzielonych od siebie przecinkami. Pierwszym elementem jest nazwa czcionki, która ma być zastąpiona. Drugim jest nazwa czcionki X, w takiej formie, jaką należy podać również w definicji zmiennej **`Default`**. Jako trzeci element można podać opcjonalnie słowo kluczowe **`subst`**. To słowo kluczowe powoduje, że istniejąca już definicja (tworzona automatycznie przez WINE) zostaje zastąpiona przez definicję aliasu. Ponieważ możliwe jest utworzenie większej ilości definicji aliasu, po nazwie zmiennej **`Alias`** należy zawsze podać liczbę, przy pomocy której określa się, o którą definicję aliasu chodzi. Rozpoczyna się z **`0`**; nie może być pominięta żadna kolejna liczba.
Przykład: **`"Alias0" = "System, --international-, subst"`**.

7.7 Konfiguracja interfejsów i dostępu do sprzętu

WINE może zezwolić aplikacjom bezpośredni dostęp do portów szeregowych i równoległych, jak również do dalszych, dowolnych adresów wejść i wyjść. Zasadniczo należy przy tym pamiętać, że dany proces WINE musi mieć wystarczające uprawnienia, aby taki dostęp był rzeczywiście możliwy. Jeśli WINE uruchomiony jest z zwykłymi uprawnieniami użytkownika (a nie z uprawnieniami administratora), oznacza to z reguły, że należy zrewidować prawa dostępu do plików urządzeń reprezentujących te urządzenia, do których trzeba zezwolić dostęp programom windowsowym. Bezpośredni dostęp do adresów wejść i wyjść jest pod Linuksem dozwolony tylko dla administratora.

Do konfiguracji portów szeregowych służy sekcja **[serialports]** w pliku konfiguracyjnym. Poszczególne zmienne w tej sekcji opisują porty szeregowe, tak jak to jest powszechne pod DOS i Windows (**Com1**, **Com2**, itd.). Jako wartość nadawana jest tym zmiennym nazwa pliku urządzenia, który reprezentuje odpowiedni port pod Linuksem. Jeśli więc aplikacja windowsowa pod WINE powinna mieć dostęp przez port **Com1** do urządzenia, które pod Linuksem reprezentowane jest przez plik **/dev/ttyS0**, w pliku konfiguracyjnym, w tej sekcji musi znaleźć się następujący wpis:

```
[serialports]
"Com1" = "/dev/ttyS0"
```

Konfiguracja portów równoległych odbywa się analogicznie do konfiguracji portów szeregowych. Nazwa odpowiedniej sekcji brzmi **[parallelports]** a nazwy portów równoległych pod DOS i Windows brzmią **Lpt1**, **Lpt2** itd. Jeśli więc programy windowsowe pod WINE mają mieć dostęp do portu równoległego przez nazwę **Lpt1**, a port ten pod Linuksem reprezentowany jest przez plik urządzenia **/dev/lp0**, w pliku konfiguracyjnym należy ująć tę sekcję z następującym wpisem:

```
[parallelports]
"Lpt1" = "/dev/lp0"
```

Aby umożliwić dostęp do określonych adresów wejść i wyjść, żądane adresy należy podać w sekcji **[ports]** w zapisie szesnastkowym. Jako adresy można podać albo pojedyncze adresy (przykład: **0x779**) albo zakresy adresów. W przypadku zakresów podaje się dolny i górny adres zakresu, połączone kreską (przykład: **0x280-0x2a0**). Jeśli trzeba skonfigurować więcej adresów lub ich zakresów, należy podać je kolejno po sobie, oddzielone przecinkami. Adresy, z których ma następować odczyt, należy przyporządkować zmiennej **read**, a te, do których ma nastąpić zapis – zmiennej **write**. Jeśli określony adres ma być dostępny zarówno do odczytu jak i zapisu, należy go przyporządkować obu zmiennym. Przykład:

```
[ports]
"read" = "0x378,0x379,0x220-0x2a0"
"write" = "0x379,0x220-0x2a0"
```

7.7.1 Dostęp do urządzeń SCSI

WINE umożliwia również bezpośredni dostęp do urządzeń SCSI (poprzez interfejs ASPI). Do tego celu nie są konieczne specjalne dane konfiguracyjne, jednak jądro systemu musi być tak skonfigurowane, aby zawierało obsługę standardowego dostępu SCSI (*generic SCSI*). Poza tym pliki urządzeń, reprezentujących standardowe urządzenia SCSI (standardowo **/dev/sg0**, **/dev/sg1** itd.) muszą być odpowiednio udostępnione.

7.8 Konfiguracja rejestru Windows

Windowsowe systemy operacyjne udostępniają bazę danych, w której programy zapisują m.in. dane konfiguracyjne, i z której mogą je później odczytywać. Ten tak zwany rejestr tworzony jest również przez WINE. Poza tym WINE jest w stanie zaimportować rejestr z istniejącej instalacji Windows, aby programy uruchamiane pod WINE miały do dyspozycji wszystkie dane, dostępne również pod Windows. Ma to znaczenie szczególnie wtedy, gdy programy te zostały zainstalowane pod Windows i są uruchamiane pod WINE. Należy zwrócić uwagę na to, że WINE nigdy nie zmienia rejestru istniejącej instalacji Windows. Jeśli więc programy pracujące pod WINE zapisują dane do rejestru, nie są one dostępne pod Windows. WINE zapisuje dane rejestru w własnych plikach, znajdujących się zazwyczaj w katalogu domowym danego użytkownika. Obok danych rejestru poszczególnych użytkowników, przez administratora mogą być stworzone pliki obowiązujące w całym systemie. Te pliki znajdują się normalnie w tym samym katalogu, co pliki konfiguracyjne obowiązujące dla całego systemu, a więc w **/etc** albo w **/usr/local/etc**, jeśli WINE został skompilowany z ustawieniami standardowymi. W sekcji **[registry]** pliku konfiguracyjnego dostępne są niżej wymienione zmienne, przy pomocy których można określać między innymi, czy ma zostać zaimportowany istniejący rejestr, i gdzie WINE ma zapisać dane z własnego rejestru.

LoadGlobalRegistryFiles
Jeśli wartość tej zmiennej ustawiona jest na **true**, WINE wczytuje dane rejestru obowiązujące dla całego systemu. Jest to ustawienie standardowe.
Przykład: **"LoadGlobalRegistryFiles" = "true"**.

LoadHomeRegistryFiles
Jeśli wartość tej zmiennej ustawiona jest na **true**, WINE wczytuje dane rejestru z podkatalogu **.wine** katalogu domowego użytkownika wywołującego program. Dane rejestru tego użytkownika ładowane są po danych obowiązu-

jących dla całego systemu, tak że użytkownicy mogą nadpisywać dane z rejestru globalnego swoimi własnymi wartościami.
Przykład: **"LoadHomeRegistryFiles" = "true"**.

LoadWindowsRegistryFiles
Przy pomocy tej zmiennej określa się, czy ma być ładowany rejestr istniejącej instalacji Windows. Jeśli wartość tej zmiennej ustawiona jest na **true**, WINE sprawdza samodzielnie, jaka wersja Windows jest zainstalowana, i wczytuje dane rejestru tej instalacji. Należy zwrócić uwagę na to, że działa to tylko wtedy, gdy katalogi **Windows** oraz **System** są poprawnie określone w części ogólnej pliku konfiguracyjnego i odpowiadają istniejącej instalacji. Poza tym, ewentualnie jest konieczne właściwe określenie zmiennej **Profile** w części ogólnej.
Przykład: **"LoadWindowsRegistryFiles" = "true"**.

WriteToHomeRegistryFiles
Jeśli wartość tej zmiennej ustawiona jest na **true**, WINE próbuje zapisywać wszystkie zmiany, powstałe w rejestrze podczas pracy programu, w plikach rejestru w katalogu **.wine** użytkownika wywołującego WINE. Jest to z reguły konieczne, aby programy windowsowe (w szczególności programy instalacyjne) mogły zapisywać zmiany w konfiguracji.
Przykład: **"WriteToHomeRegistryFiles" = "true"**.

PeriodicSave
Jeśli zmiany w rejestrze mają być zapisywane przez WINE, normalnie odbywa się to automatycznie, przy zakończeniu pracy WINE. Jednakże zmiany nie są zapisywane, jeśli WINE nie może być poprawnie zakończony z powodu błędu. Dlatego możliwe jest określenie przy pomocy tej zmiennej, w jakim odstępie czasu program ma automatycznie zapisywać rejestr. Liczba będąca wartością tej zmiennej interpretowana jest jako odstęp czasu w sekundach. Aby rejestr był zapisywany automatycznie, np. co 10 minut, należy ustawić tę zmienną następująco: **"PeriodicSave" = "600"**.

SaveOnlyUpdatedKeys
Przy pomocy tej zmiennej można określić, czy WINE ma zapisywać jedynie te części rejestru, które zmieniły się podczas pracy programu, czy też ma być zapisywany zawsze cały rejestr. Podany przykład służy do określenia zapisu kompletnego rejestru: **"SaveOnlyUpdatedKeys" = "false"**.

Wskazówka:
Ponieważ import dużego rejestru Windows jest procesem stosunkowo czasochłonnym, zaleca się import rejestru **("LoadWindowsRegistryFiles" = "true")** i zapis rejestru kompletnego **("SaveOnlyUpdated Keys" = "false")** tylko przy pierwszym uruchomieniu WINE. Po tej procedurze istnieje pełny rejestr, zapisany w własnym formacie WINE, tak że przy późniejszych uruchomieniach WINE można zrezygnować z importu **("LoadWindowsRegistryFiles" = "false")**.

7.9 Ustawienie wyglądu

W sekcji **[Tweak.Layout]** pliku konfiguracyjnego można poprzez zmienną **WINELook** określić, jaki wygląd Windows ma oddać WINE. Wartość **Win31** powoduje, że WINE wyświetla wszystkie elementy okien w sposób znany z Windows 3.11. Analogicznie, wartości **Win95** i **Win98** powodują wyświetlenie bardziej nowoczesnej szaty graficznej. Przykład:

```
[Tweak.Layout]
"WINELook" = "Win98"
```

7.10 Konfiguracja konsoli windowsowej

W przeciwieństwie do 16-bitowej wersji Windows, przy pomocy API Win32 można, tak jak pod UNIX, tworzyć programy dla trybu tekstowego, które pod Windows uruchamiane są normalnie w tak zwanym „Trybie MS-DOS“, jakkolwiek programy te nie mają wiele wspólnego z MS-DOS. Pod WINE programy te używają zwykle standardowego wejścia i wyjścia terminala, z którego został uruchomiony WINE. W sekcji **[Console]** pliku konfiguracyjnego można dokładniej określić właściwości konsoli dla programów windowsowych.

Drivers
W tym miejscu określa się, w jaki sposób ma być udostępniona konsola. WINE może wykorzystywać do tego terminal związany z procesem WINE, albo dla każdego programu windowsowego, wymagającego nowej konsoli, uruchamiać nowe okno terminala (jak np. **xterm**). Ogólnie zaleca się otwieranie dla każdej konsoli własnego okna terminala, ponieważ może dochodzić do niepożądanych efektów ubocznych, gdy różne procesy windowsowe i sam

WINE będą używać tego samego terminala. Dalej, możliwe jest używanie biblioteki **ncurses**, aby wykorzystywać określone właściwości, jak wyświetlanie różnych kolorów. Zmiennej **Drivers** można obecnie nadawać połączenia następujących wartości: **tty**, **xterm** i **ncurses**. Jeśli używa się kilka z nich, należy je oddzielać od siebie znakiem **+**. Wartość **tty** powoduje, że WINE używa dla konsoli ten terminal, z którym połączony jest proces WINE. Przez podanie **xterm** otwarte zostaje nowe okno terminala, jeśli program windowsowy zażąda nowej konsoli. Wprowadzenie wartości **ncurses** powoduje korzystanie z biblioteki **ncurses**. Działa to tylko wtedy, gdy przy kompilacji programu została uaktywniona jej obsługa.
Przykład: **"Drivers" = "ncurses+xterm"**.

XtermProg
Przy pomocy tej zmiennej można określić, który program ma być wywołany, w celu wyświetlenia konsoli w własnym oknie (np. przy ustawieniu **"Drivers" = "xterm"**). Mogą być używane wszystkie programy emulujące terminal, które rozumieją parametry trybu tekstowego używane przez **xterm**.
Przykład: **"XtermProg" = "wterm**".

InitialRows
Przy pomocy tej zmiennej określa się, ile wierszy ma mieć konsola po uruchomieniu.
Przykład: **"InitialRows" = "24"**.

InitialColumns
Przy pomocy tej zmiennej określa się, ile kolumn ma mieć konsola po uruchomieniu.
Przykład: **"InitialColumns" = "80"**.

TerminalType
Tu określa się, z jakiego typu terminala ma pochodzić biblioteka **ncurses**. Typowymi wartościami są **xterm** albo **linux**.

7.11 Schowek

Koncepcja schowka różni się między Windows a systemem X Window. Aby skopiować do schowka na przykład tekst, pod Windows należy go normalnie najpierw zaznaczyć, a potem przez polecenie menu skopiować lub wyciąć do schowka. Pod X są do dyspozycji co najmniej dwa typy schowka. Po zaznaczeniu tekstu, jest on do dyspozycji jako tak zwany wybór pierwotny; może on być następnie natychmiast wklejany do innych aplikacji (mniej więcej jak przy naciśnięciu środkowego przycisku myszy). Dalszym wyborem jest właściwy schowek, do którego teksty czy inne dane z wielu aplikacji X mogą być kopiowane, a stamtąd z powrotem wklejane. W sekcji **[clipboard]** pliku konfiguracyjnego można określić, jak ma współdziałać schowek Windows z schowkiem systemu X Window.

ClearAllSelections
Jeśli zmienna ta ustawiona jest na **true**, zawartość schowka windowsowego jest kasowana i zastępowana przez zawartość schowka systemu X Window, jeśli w innej aplikacji X coś zostało zaznaczone (wybór pierwotny). A więc, jeśli przy korzystaniu z tego ustawienia coś zostanie umieszczone w schowku przy pomocy programu windowsowego, a potem w aplikacji X zostanie zaznaczony np. tekst, pierwotna zawartość schowka Windows znika i do dyspozycji jest w nim zaznaczenie z aplikacji X. Jeśli wartość tej zmiennej ustawiona jest na **false**, zawartość schowka pozostaje nienaruszona w przypadku wyboru pierwotnego. Aby więc np. skopiować tekst z **XEmacs** do **Worda**, nie wystarcza zaznaczenie tekstu w **XEmacs**, lecz tekst ten musi być wyraźnie skopiowany do schowka, jeśli uprzednio znajdowały się tam dane skopiowane z innej aplikacji Windows.
Przykład: **"ClearAllSelections" = "true"**

PersistentSelection
Po zakończeniu WINE, zawartość schowka normalnie nie jest dostępna dla innych programów X. Jeśli zmienna ta ustawiona jest na **true**, WINE uruchamia mały program działający w tle (**wineclipsrv**), który przechowuje zawartość schowka tak długo, aż zostanie ona zastąpiona przez inne dane. Dopiero wtedy program ten kończy działanie.
Przykład: **"PersistentSelection" = "true"**

7.12 Konfiguracja sterownika drukarki PostScriptowej

WINE może używać dwóch typów sterowników drukarki, mianowicie prawdziwych, 16-bitowych sterowników windowsowych, jakie używane są przez Windows 3.11 czy 95/98, albo własnego sterownika PostScriptowego, przy pomocy którego można drukować z większości aplikacji windowsowych w formacie PostScript. Te dane PostScrip-

towe można przesłać następnie poprzez oprogramowanie buforujące systemu (normalnie **lpr/lpd**) na drukarkę lokalną lub sieciową.

Wskazówki odnośnie zastosowania 16-bitowych sterowników drukarki znajdują się m.in. w pliku **printing** w podkatalogu **documentation** katalogu z kodem źródłowym WINE. Poniżej omówiona zostanie jedynie konfiguracja wbudowanego sterownika PostScriptowego (co i tak ogólnie jest preferowane).

Najpierw należy zgłosić istnienie sterownika drukarki. W tym celu w pliku **win.ini** w katalogu Windows należy wprowadzić następujące zmiany:

```
[windows]
device=WINE PostScript Driver,WINEPS,LPT1:

[devices]
WINE PostScript Driver=WINEPS,LPT1:
```

Jeśli używa się WINE bez istniejącej instalacji Windows, może mieć miejsce sytuacja, że pliku **win.ini** nie będzie. W takim przypadku należy utworzyć ten plik na nowo i umieścić w nim wymienione wyżej linie. Jeśli plik ten już istnieje, trzeba zlokalizować w nim sekcje **[devices]** oraz **[windows]**, i uzupełnić o powyższe wpisy. W żadnym przypadku sekcje te nie mogą znajdować się kilka razy w tym pliku.

Aby można było drukować również z programów 32-bitowych, konieczne jest wprowadzenie kilku wpisów w rejestrze. Wpisy te znajdują się w pliku **psdrv.reg** w podkatalogu **documentation** katalogu z kodem źródłowym WINE i można je zaimportować przy pomocy programu **regapi**, znajdującego się w podkatalogu **programs/regapi** katalogu z kodem źródłowym. Program należy najpierw skompilować, o ile już tego nie wykonano. W tym celu należy przejść do katalogu **programs/regapi** i wydać w nim polecenie **make** (warunkiem tego jest udana, wcześniejsza kompilacja WINE). Potem wymagane klucze rejestru można zaimportować przez wydanie następującego polecenia:

```
./regapi setValue < ../../documentation/psdrv.reg
```

Uwaga:
W przypadku programu **regapi** chodzi o tak zwany program **WineLib**. Takie programy wykorzystują WINE do realizacji funkcji typowych dla Windows. Aby mogły pracować, w systemie musi istnieć działający i skonfigurowany program WINE. Ten krok nie może być więc wykonany, zanim nie zostanie ukończone tworzenie pliku konfiguracyjnego i testowanie działania WINE.

Dalej, wymagany jest tak zwany plik PPD. Plik ten opisuje różne właściwości drukarki i jest konieczny, aby WINE mógł np. decydować, czy może drukować w kolorze, czy w odcieniach szarości. Jeśli nie używa się drukarki PostScriptowej, konieczny jest plik PPD, opisujący właściwości sterownika GhostScriptowego stosowanej drukarki, ponieważ program ten używany jest z reguły do przekształcania danych z formatu PostScript do formatu drukarki. W Debianie pliki te są dostępne w pakiecie **ppd-gs**. Jeśli pakiet ten ma być używany, należy wcześniej przeczytać plik **/usr/doc/ppd-gs/README**.

Szereg plików PPD dostępnych jest na **ftp://ftp.adobe.com/pub/adobe/printerdrivers/win/all/ppdfiles/**. W katalogu tym znajdują się samorozpakowujące się archiwa **.zip**, zawierające pliki PPD, dla drukarek różnych producentów. Archiwa te mogą być rozpakowane przy pomocy programu **unzip(1)**. Aby więc rozpakować plik **hp.exe**, należy wydać polecenie:

```
unzip -L hp.exe
```

Następnie należy wybrać żądany plik PPD, przy tym trzeba ewentualnie trochę poeksperymentować, w celu osiągnięcia optymalnych wyników. Plik można skopiować przykładowo do katalogu **/usr/local/etc/** a następnie należy poinformować o nim WINE przez wpis w pliku konfiguracyjnym:

```
[psdrv]
"ppdfile" = "/usr/local/etc/HP4M3_V1.PPD"
```

Jeśli plik ten znajduje się w innym katalogu albo ma inną nazwę, trzeba oczywiście odpowiednio dopasować wartość zmiennej **ppdfile**.

Dalej, WINE wymaga plików fontmetrycznych (*fontmetric file*) czcionek, dostępnych na drukarce (albo poprzez **ghostscript**). Szereg takich plików fontmetrycznych można zainstalować w Debianie przy pomocy pakietu **tetex-extra**. Po instalacji znajdują się one w katalogu **/usr/share/texmf/fonts/afm**. Normalnie mają rozszerzenie nazwy pliku **.afm** (*Adobe FontMetric*). Pliki, których należy użyć, muszą być określone w sekcji **[afmfiles]** pliku konfiguracyjnego. Nazwę każdego pliku należy przyporządkować zmiennej, której nazwa składa się z łańcucha znaków **file** i kolejnego numeru. Przy wyborze plików AMF należy uwzględnić, że trzeba poda-

wać tylko te pliki, dla których odpowiednia czcionka została rzeczywiście wymieniona w wcześniej zdefiniowanym pliku PPD. Nazwy czcionek znajdują się normalnie w plikach fontmetrycznych, które mogą być przeglądane przy pomocy edytora tekstowego. Początek odpowiedniej sekcji w pliku konfiguracyjnym może wyglądać następująco:

```
[afmfiles]
"file1" = "/usr/share/texmf/fonts/afm/adobe/times/ptmb8a.afm"
"file2" = "/usr/share/texmf/fonts/afm/adobe/times/ptmbi8a.afm"
"file3" = "/usr/share/texmf/fonts/afm/adobe/times/ptmr8a.afm"
"file4" = "/usr/share/texmf/fonts/afm/adobe/times/ptmri8a.afm"
"file5" = "/usr/share/texmf/fonts/afm/adobe/helvetic/phvbo8an.afm"
"file6" = "/usr/share/texmf/fonts/afm/adobe/helvetic/phvb8a.afm"
"file7" = "/usr/share/texmf/fonts/afm/adobe/helvetic/phvb8an.afm"
"file8" = "/usr/share/texmf/fonts/afm/adobe/helvetic/phvbo8a.afm"
"file9" = "/usr/share/texmf/fonts/afm/adobe/helvetic/phvro8an.afm"
"file10" = "/usr/share/texmf/fonts/afm/adobe/helvetic/phvr8a.afm"
"file11" = "/usr/share/texmf/fonts/afm/adobe/helvetic/phvr8an.afm"
"file12" = "/usr/share/texmf/fonts/afm/adobe/helvetic/phvro8a.afm"
```

Także tu należy oczywiście dopasować nazwy plików do używanych w rzeczywistości plików fontmetrycznych. Sterownik drukarki powinien być po tym do dyspozycji jako **WINE PostScriptDriver** w dialogach dotyczących drukowania aplikacji windowsowych.

7.13 Konfiguracja bufora

Standardowo dane do drukowania wysyłane są do pliku w aktualnym katalogu bieżącym, którego nazwa odpowiada określeniu przyłączenia drukarki używanej pod Windows. W sekcji **[spooler]** pliku konfiguracyjnego można przekierować te dane do innego pliku, albo przekazać dalej, do innego programu. W tym celu należy w sekcji **[spooler]** podać nazwę przyłączenia jako zmienną (na przykład: **LPT1:**) i przyporządkować jej nazwę pliku, do którego mają być skierowane dane do drukowania. Aby dane skierować do wejścia standardowego programu, należy podać nazwę żądanego programu za znakiem potoku **|** (*pipe*).

Jeśli więc dane wysyłane do portu **LPT1:** mają być przekazane do programu **lpr**, aby doprowadzić je do bufora, w pliku konfiguracyjnym należałoby dodać następującą sekcję:

```
[spooler]
"LPT1:" = "|lpr"
```

7.14 Konfiguracja multimediów

Architektura multimediów pod Windows składa się z różnych typów sterowników i interfejsów, które mogą być używane przez programy windowsowe. Architektura ta jest odtwarzana przez WINE, przy czym różne części składowe są mniej lub bardziej kompletne. Wyczerpujący opis architektury multimediów znajduje się w pliku **multimedia** w podkatalogu **documentation/status** katalogu z kodem źródłowym WINE.

WINE udostępnia własny sterownik do sterowania sprzętem dźwiękowym. Sterowanie odbywa się poprzez pliki urządzeń **/dev/dsp**, **/dev/audio**, **/dev/mixer** itp. Dlatego należy zwracać uwagę na to, aby istniały odpowiednie prawa do odczytu i zapisu dla tych plików, jeśli WINE ma wykorzystywać obsługę dźwięku. Sterownik ten opiera się na sterowniku OSS (*Open Sound System*), który standardowo jest częścią składową jądra Linuksa.

Oprócz bibliotek **winmm** i **mmsystem** można stosować wszystkie inne składniki systemu multimediów również z istniejącej instalacji Windows. Jest to pomocne przede wszystkim w przypadku niektórych sterowników MCI, które nie są jeszcze w pełni zaimplementowane w WINE. Sterowniki MCI są ładowane, jeśli w sekcji **[mci]** pliku **system.ini** w katalogu Windows znajdują się polecenia w następującej postaci:

```
[mci]
cdaudio=mcicda.drv
sequencer=mciseq.drv
```

Przy stosowaniu istniejącej instalacji Windows odpowiednie polecenia powinny się już tam znajdować. Jeśli WINE używany jest bez instalacji Windows, jako szablon może służyć plik **system.ini** z podkatalogu **documentation/samples** katalogu z kodem źródłowym WINE. Przy pomocy zmiennej **mci** w sekcji **[options]** pliku konfiguracyjnego WINE można przepisać definicje z tego pliku **system.ini**. Ma to sens, gdyż nie należy ładować pewnych sterowników, podanych w pliku **system.ini**, ponieważ nie pracują one

poprawnie z WINE. Do nich należy obecnie sterownik MCI **videodisk**. Aby załadować wszystkie sterowniki MCI (oprócz **videodisk**), w pliku konfiguracyjnym można wprowadzić następującą sekcję:

```
[options]
"mci" = "CDAUDIO:SEQUENCER:WAVEAUDIO:AVIVIDEO:MPEGVIDEO"
```

W sekcji **DllOverrides** pliku konfiguracyjnego, jak opisano wyżej, można określić – jak w przypadku bibliotek – czy ma być ładowany sterownik z istniejącej instalacji Windows, czy też ma być używany sterownik dostarczany przez WINE. Należy przy tym uwzględnić, że musi być podawane rozszerzenie nazwy pliku **.drv** dla sterowników, w przeciwieństwie do bibliotek. Aby, na przykład, załadować z istniejącej instalacji sterowniki MCI **mciavi** i **mcianim**, a pozostałe sterowniki wziąć z WINE, w sekcji **[DllOverrides]** należałoby dodać następujące linie:

```
"mciavi.drv, mcianim.drv" = "native, builtin"
"mcicda.drv, mciseq.drv"  = "builtin, native"
"msacm.drv, midimap.drv"  = "builtin, native"
"mciwave.drv"    = "builtin, native"
```

7.15 Ustawienia rejestru

Szereg programów windowsowych oraz sam WINE wymagają dla poprawnego działania określonych wpisów do rejestru Windows. Jeśli WINE jest używany bez istniejącej instalacji Windows albo rejestr Windows nie jest importowany, wpisów tych nie ma i muszą być zaimportowane przy pomocy wspomnianego już wyżej programu **regapi**. Jako szablon dla nich może służyć plik **winedefault.reg** z katalogu z kodem źródłowym WINE. Plik ten należy jednak sprawdzić, zanim zostanie zaimportowany, czy zgadzają się podane tam litery napędów i ścieżki. Poza tym należy oczywiście określić w pliku konfiguracyjnym, że rejestr ma być zapisany przy zakończeniu pracy programu, aby zaimportowane dane były dostępne także przy następnym uruchomieniu WINE. Następnie można przejść do podkatalogu **programs/regapi** katalogu z kodem źródłowym WINE. Tam należy skompilować ten program poleceniem **make**, o ile nie zostało to jeszcze wykonane. Potem można zaimportować wymagane dane przy pomocy następującego polecenia:

```
./regapi setValue < ../../winedefault.reg
```

8 Wywołanie WINE i opcje wiersza poleceń

WINE, jak każdy inny program, może być wywołany z wiersza poleceń. Nazwę programu windowsowego, który ma być uruchomiony, należy podać w poleceniu, obok WINE. Programy, które znajdują się w katalogach, wymienionych w zmiennej **Path** w sekcji WINE pliku konfiguracyjnego, mogą być wywoływane bez podawania pełnej ścieżki dostępu. Podawanie rozszerzenia nazwy pliku **.exe** jest opcjonalne. Jeśli więc ma być uruchomiony WINE i program windowsowy **winmine** (Saper), należałoby wpisać następujące polecenie, zakładając, że plik **winmine.exe** znajduje się w katalogu wymienionym w zmiennej **Path**.

```
wine winmine.exe
```

Jeśli mają być uruchamiane programy, znajdujące się w katalogach nie wymienionych w zmiennej **Path**, konieczne jest podanie ścieżki dostępu. Tu można używać albo ścieżki DOS-/Windows albo UNIX. Oba poniższe polecenia są więc równoważne, jeśli napęd **C:** byłby przyporządkowany uniksowemu katalogowi **/var /winroot**.

```
wine c:\\windows\\winmine.exe
wine /var/winroot/windows/winmine.exe
```

Lewy ukośnik (*backslash*) ma dla powłoki **Bash** specjalne znaczenie, i dlatego normalnie nie jest przekazywany do WINE. Dwukrotne wprowadzenie tego znaku powoduje, że przekazywany jest ukośnik zwykły. Nazwy plików i katalogów w Windows mogą zawierać spacje. Także tu konieczne jest zastosowanie pewnego tricku, aby spacje nie prowadziły do tego, że powłoka potraktuje poszczególne części nazw jako różne argumenty. Poniższe, przykładowe polecenie nie doprowadziłoby do żądanego efektu:

```
wine /var/winroot/Programme/Microsoft Games/RoA Trial Version/PACDEMO.EXE
```

W tym przypadku powłoka przekazałaby do WINE cztery argumenty, a mianowicie **/var/winroot/Programme/Microsoft**, **Games/RoA**, **Trial** i **Version/PACDEMO.EXE**, po czym program nie zostałby znaleziony. Aby powłoka nie rozdzielała nazw przy napotkaniu spacji, przed spacjami należy również umieścić lewy ukośnik:

```
wine /var/winroot/Programme/Microsoft\ Games/RoA\ Trial\ Version/PACDEMO.EXE
```

Jeśli uruchamianemu programowi windowsowemu należy przekazać argumenty, przed nazwą programu należy umieścić dwie kreski **--**, aby wskazać WINE koniec opcji (po to, żeby opcje programu, które mogą ewentualnie mieć postać **--XXX**, nie zmyliły WINE). Żeby, na przykład, uruchomić program **notepad.exe** i nadać temu programowi jako argument **readme.1st**, należy wywołać WINE następująco:

```
wine -- notepad readme.1st
```

W przypadku, gdy kilka programów windowsowych ma być uruchomionych kolejno po sobie, WINE musi być wywołany kilka razy, z odpowiednimi nazwami programów, jako argumentami.

8.1 Opcje wiersza poleceń

Obok nazw programów, które mają być uruchomione, WINE rozumie szereg opcji, przy pomocy których można globalnie wpływać na operacje programu. Opcje te interpretowane są bezpośrednio przez WINE i nie są przekazywane do wywoływanego programu windowsowego.

--help
Opcja ta powoduje wyświetlenie dostępnych opcji.

--version
Opcja służąca do wyświetlenia numeru wersji WINE.

--dll biblioteka[,biblioteka ...]=b|n[:biblioteka[,biblioteka,...]=b|n]
Przy pomocy tej opcji można nadpisać ustawienia z sekcji **[DllOverrides]** pliku konfiguracyjnego, a więc można podać, które biblioteki mają być załadowane z istniejącej instalacji Windows, a które mają być udostępnione bezpośrednio przez WINE. Opcji tej należy nadać postać, składającą się nazw bibliotek, oddzielonych od siebie przecinkami (bez spacji). Następnie należy wpisać znak równości i za nim albo literę **b**, w celu określenia, że podane biblioteki mają być dostarczone przez WINE, albo literę **n**, co oznacza, że te biblioteki mają być załadowane z instalacji Windows. Wyrażenia te można powtarzać, wtedy należy je oddzielić dwukropkiem (bez spacji). Na przykład, jeśli mają być użyte biblioteki **shell**, **commdlg** i **commctrl** z ich przynależnymi bibliotekami 32 bitowymi z instalacji Windows oraz **advapi32** z WINE, mimo, że w pliku konfiguracyjnym podano inaczej, opcja ta może wyglądać następująco:

```
--dll commdlg,comdlg32,commctrl,comctl32,shell,shell32=n:advapi32=b
```

--debugmsg +|-foo,+|-bar
Ta opcja służy do kontroli, jaki rodzaj informacji ma być wydawany przez WINE w celu śledzenia błędów i przebiegu procesów. Dostępnych jest szereg tak zwanych kanałów, których nazwy są wskazywane, jeśli ta opcja użyta bez dodatkowych danych. Poszczególne kanały odpowiadają normalnie różnym zakresom działania WINE, których zachowanie może być sprawdzane przez wydawanie określonych informacji. Na przykład, żeby włączyć komunikaty dla kanału **file**, należy użyć opcji **--debugmsg +file**. Jeśli mają być podawane komunikaty kanałów **file** i **dosfs**, trzeba podać **--debugmsg +file,+dosfs**. Dalsze informacje na ten temat znajdują się w pliku **debug-msg** w podkatalogu **documentation** katalogu z kodem źródłowym WINE. Dwoma szczególnie ważnymi kanałami są **relay** i **snoop**. Jeśli włączony jest kanał **relay**, otrzymuje się komunikaty, jakie funkcje WINE wywoływane są przez programy windowsowe, i z jakimi parametrami, oraz jakie wartości zwrotne dają te funkcje. Kanał **snoop** wskazuje, jakie funkcje wywoływane są z prawdziwych bibliotek Windows. Komunikaty kanału **relay** służą często programistom WINE do stwierdzenia, gdzie wystąpił błąd. Dlatego zaleca się, aby w raportach błędów przesyłać do **WINE-Newsgroup** ostatnie 200 linii komunikatu WINE, który powstał przy wywoływaniu tego programu z opcją **--debugmsg +relay** przed wystąpieniem błędu.

9 Źródła błędów

WINE jest jeszcze w stadium rozwojowym, dlatego wiele programów windowsowych nie działa z nim, lub działa tylko częściowo. Jest bardzo prawdopodobne, że niektóre programy nigdy nie będą działały z WINE, np. programy, wymagające własnych sterowników windowsowych, które nie będą mogły być załadowane pod Linuksem. Pomimo to wiele programów windowsowych działa z WNE bardzo dobrze a ilość tych działających programów zwiększa się stosunkowo szybko. Jeśli jakiś program nie działa zgodnie z oczekiwaniem, pomocne mogą być odpowiedzi na poniższe pytania:

WINE w ogóle nie działa

W pewnych wersjach pakietów biblioteki C (wersja 2.1.3) jest błąd, prowadzący do tego, że WINE przestaje działać z komunikatami błędów bezpośrednio po uruchomieniu. Problem ten można usunąć instalując nowszą wersję biblioteki C albo nadając wartość zmiennej **`LANG`**. Przy użyciu powłoki **`Bash`** można to wykonać wydając następujące polecenie: **`export LANG=pl_PL`**.

WINE dalej nie działa

Jeśli WINE został zainstalowany w opisany sposób, należy sprawdzić, czy w komputerze nie znajduje się jednocześnie inna, starsza wersja, zainstalowana na przykład z dystrybucją. Taka sytuacja może prowadzić do prób załadowania niekompatybilnych bibliotek.

Katalogi `Windows` i `Windows/system` nie są podane prawidłowo

Komunikat **`Invalid path 'c:\windows' for windows directory`** mówi, że w sekcji **`WINE`** pliku konfiguracyjnego w zmiennej **`Windows`** podano nieistniejący katalog Windows. Należy więc sprawdzić, czy istnieje katalog **`Windows`**, i czy napęd, na którym się on znajduje, przyporządkowany jest właściwemu katalogowi uniksowemu.

Nie zgadza się przyporządkowanie liter napędów

Jeśli przy uruchomieniu WINE otrzymuje się komunikaty jak: **`Warning: /var/winroot/windows/sol.exe not accessible from a DOS drive`**, WINE informuje, że program, który ma być uruchomiony, znajduje się w katalogu, który w wyniku przyporządkowania w pliku konfiguracyjnym nie jest skojarzony z żadnym napędem. W takim przypadku należy sprawdzić przyporządkowanie napędów.

Programy windowsowe nie znajdują ustawień i bibliotek

Jeśli WINE używany jest z istniejącą instalacją Windows, konieczna jest zgodność liter napędów w Windows i WINE. Jeśli jakiś program zapisał w rejestrze, że określone składniki znajdują się w katalogu **`C:\Windows\System`**, wtedy program ten musi znaleźć te składniki również pod WINE w tym samym katalogu. Możliwe jest jednak stosowanie z WINE dodatkowych napędów (np. przyporządkowanie własnego katalogu domowego do jakiejś litery napędu), których nie ma w Windows.

Gry uruchamiają się w oknie, mimo że wybrano pełny ekran, i działają za wolno

Większość gier używa określonej głębi kolorów, na którą przełącza się Windows, gdy dana gra zostaje uruchomiona. Niestety, w systemie X Window nie jest możliwa zmiana głębi kolorów podczas pracy, dlatego WINE musi emulować w oknie żądaną głębię kolorów, jeśli serwer X nie pracuje z tą głębią. Wymaga to bardzo dużo mocy obliczeniowej komputera a poza tym nie można przełączyć się do trybu pełnoekranowego. Środkiem zaradczym jest tylko uruchamianie serwera X w właściwej głębi kolorów, zanim uruchomi się WINE. Do tego celu służą opcje **`-bpp`**, w serwerach X projektu **XFree86** wersji 3.3.x wzgl. **`-depth`** w wersji 4.0. Można również polecić uruchomienie drugiego serwera X, co najłatwiej można wykonać poleceniem **`xinit(1)`**. Aby uruchomić na przykład program **`q2test.exe`** z WINE na drugim serwerze X z głębią kolorów 8 bit na piksel z bieżącego katalogu roboczego, można wykorzystać następujące polecenie:

```
xinit /usr/local/bin/WINE q2test.exe --display :1 -- -bpp 8 :1
```

Jeśli WINE znajduje się w innym katalogu niż **`/usr/local/bin`**, polecenie to należy oczywiście odpowiednio dopasować.

Tryb DGA nie działa, mimo że X jest uruchomiony w właściwej głębi kolorów; gry są dalej za wolne

Zastosowanie DGA dozwolone jest normalnie tylko dla administratora, a więc program ten musi być uruchomiony z jego prawami. Alternatywnie, może wystarczyć udzielenie użytkownikowi praw zapisu i odczytu dla pliku urządzenia **`/dev/kmem`**.

Nie słychać dźwięków
Najpierw należy sprawdzić, czy poprawnie działa obsługa dźwięku jądra Linuksa, a więc, czy możliwe jest używanie karty dźwiękowej z innymi programami linuksowymi. W następnym etapie można sprawdzić, czy karta dźwiękowa nie jest używana przez inny program i w związku z tym WINE może nie mieć do niej dostępu.

Dalsze informacje

Centrum Internetowe dla informacji o WINE znajduje się pod adresem **`http://www.winehq.com`**. Tam podane są odnośniki do wielu dalszych dokumentów, jak WINE-FAQ, WINE-HOWTO, instrukcji sporządzania raportów o błędach i innych.