Windows Meets Linux

The Wine User Experience

Owen Rudge

22nd January 2009

Hello

- I'm Owen Rudge
- I study computer science at the University of St Andrews, Scotland
- In 2008, I took part in Google Summer of Code, working on the Wine project

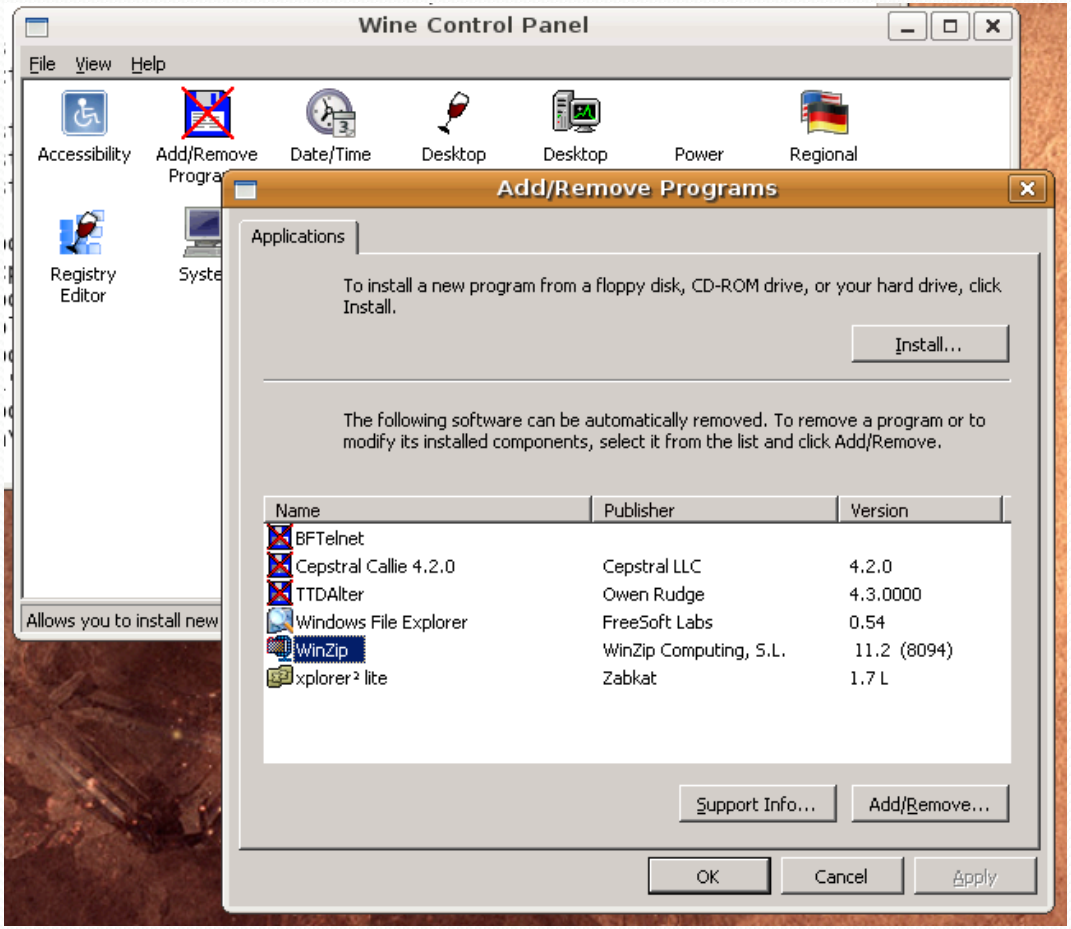
A brief overview of Wine

- Wine is a compatibility layer
- Provides an implementation of the Win32 API
- Allows the running of Windows applications on Unix-based platforms
- Project started in 1993, Wine 1.0 released in 2008

What I did last summer

- My goal was to improve Wine's configuration utilities
- Control panel improvements
- New control panel applets
- Work on improving winecfg

Wine control panel



What this talk is about

- Usability of Wine
- How Wine integrates with the native desktop environment
- The overall Wine user experience

The Wine user experience

What is the “user experience”?

- How the user interacts with the application
- Including obvious visual aspects such as windows, menus and controls
- But Wine is just a compatibility layer! What does Wine have to do with the user experience? Surely it’s up to the applications themselves?
- The user experience can be very different between different desktop environments and operating systems
- Wine must bridge the Windows user experience with that of the native environment

The ideal Wine experience

- User Jane wants to install her new piece of software. Unfortunately, it's only available in a Windows version, and she's just installed Ubuntu Linux.
- Hearing that the latest version of Ubuntu can run many Windows applications, she pops the CD in her drive.
- The application installer pops up. A few clicks later, Jane's got a new item in her Applications menu.

The ideal Wine experience (2)

- She starts her application. It looks and feels much the same as all her other applications.
- She can double-click the application's data files in the file browser; she can print from her program; she can copy and paste the program's output into her native applications like OpenOffice just fine.
- Jane is a happy user ☺

The reality of it…

- At the moment, if Jane were to pop her disc in, it's likely nothing would happen
- Double-clicking the SETUP.EXE file would hopefully start the installer
  - Assuming she chose to install Wine when she installed her OS
  - If it crashes though, the only sign of that would be in a console window
- If the application did install, then there would be a new icon in the Applications menu.

The reality of it… (2)

- She starts her application. It has elements of her Ubuntu theme, but otherwise looks rather grey and Windows-like.
- She can double-click the application's data files in the file browser, but she'll be asked what she wants to do with them.
- She should be able to print from her program. Copying and pasting data other than plain text might prove problematic though.
- Jane is a bit confused. She's lucky to have got this far!

What we aim to do

- We want Jane to be smiling, not frowning
- We need the process of using Wine to be as simple as possible
- If it works well enough, the user shouldn't be aware of Wine
- This is a challenging prospect

The Windows user experience

- Most Windows applications adhere to the basic standards Microsoft have set
- Common layout for menus, toolbars, dialog boxes
- Standardised keyboard shortcuts
- Applications can generally be expected to behave in a consistent manner

Bridging the gap

- The typical Linux user interface is similar, but not identical, to a Windows user interface
- The interface of systems such as Mac OS X is even more different (e.g., common menu bar for all apps, Dock instead of taskbar, no MDI windows)
- Wine should offer a way of making Windows applications interface with these systems as if they were native applications

What do users expect?

- Many new users to Linux would have no idea what Wine is
- Those that have heard of Wine may not know how it works
- Many users expect a “virtual desktop”, as one might get with a VM solution like VMWare or Virtual Box
- It often comes as a surprise to those users when they learn that Wine’s integration goes deeper than that
- Wine is still quite scary for a lot of users

The virtual desktop option

- It would be easiest for Wine to just give the user a virtual desktop that behaved exactly like a Windows system
- No need to worry about user interface differences
- User is then aware that their applications are specifically Windows applications, operating in their own fashion
- Wine does support this
- But is it the best way of doing things?

A fully integrated experience

- In an ideal world, a Windows application running under Wine should be near-indistinguishable from a native application
- The user would be unaware of the existence of Wine, and their application would Just Work™
- Applications would look and behave like native applications
- File associations, drag and drop, copy and paste, etc, would work seamlessly with native applications
- This is what Wine ultimately hopes to achieve

A fully integrated experience (2)

- Integration isn't just about Wine applications looking and acting like native applications
- The desktop environment needs to integrate with Wine applications
- If you install a Windows application, it should appear in your operating system's uninstallation facility
- File associations for Windows applications should appear alongside native applications
- Your Windows-based screensavers should appear alongside your native screensavers

Particular challenges

Visual styles

- One of the most obvious issues with native integration is the look of an application
- Linux window managers have many varied styles and themes. Mac OS X has its own distinctive theme.
- Windows applications with the traditional “Windows Classic” theme can look very out of place
- Ideal aim is to have Wine adopt the visual theme of the host environment

Visual styles (2)

- Some challenges:
  - Native window style may not provide all functionality expected by Windows applications
  - Some Windows applications try to take control and modify the window decorations or theming themselves
  - Different window managers have different theming methods – need separate implementations for each
  - Keeping up-to-date with style changes while applications are running – constant polling?

Visual styles (3)

- Currently, Wine can create windows in the native style
- Controls remain Windows-themed
  - Support for Microsoft theme formats being worked on
- Some work on retrieving live colour scheme data and updating appropriately
  - Results aren't always pretty!
- User can customise colour scheme within Wine itself

Desktop environment integration

- Desire is to have Windows applications behave just like Linux applications
- In an ideal world:
  - Windows applications would appear in Programs menu
  - Wine control panel applets would appear in native control panel
  - CDs with Windows “autorun” files would be recognised
  - Windows file associations, etc, could be configured from the native environment
  - … and so on

Desktop environment integration (2)

- Biggest challenge:
  - Window managers do not have a unified interface
  - Separate implementations may be required for GNOME, KDE, Mac OS X...
- Solution:
  - Wine should provide appropriate interfaces
  - Individual maintainers can then write applets that fit into their desktop of choice
- Still involves some duplication
  - Perhaps standards such as FreeDesktop could help?

Desktop environment integration (3)

- This is, in my opinion, the main integration challenge for Wine
- End-users are familiar with their desktop environment setup – their Windows applications should just fit
- Wine's current integration is limited:
  - New applications do appear in a Wine group in the Applications menu
    - But they don't get removed when the apps are uninstalled!
  - Not much integration beyond this
    - It's being worked on!

Application interaction

- Windows applications should be able to interact with native applications
  - Drag-and-drop
  - Copy and paste
- The same issues present themselves as before
- Plus issues with data format
  - Windows has its own standards for data transfer
  - Linux and OS X have theirs
  - Are they compatible? Will we need to convert?

Application interaction (2)

- Wine currently has some basic drag-and-drop support for files
  - Limited to an older Windows programming interface
- Likewise, basic copy and paste support is implemented
  - Plain text is fine
  - Some issues with character encoding (Unicode, etc)
  - More complex data formats can pose problems

Filesystem differences

- Not the actual physical filesystem
  - Although Unix permissions can pose some issues for Windows applications, which may only think in terms of “read-only”, “archive”, “system” and “hidden”
- The “shell” filesystem
  - My Computer, My Documents, etc
- Emulation of drive letters
  - Unix has no concept of drive letters, but most Windows applications rely on them

Filesystem differences (2)

- Converting between Unix and Windows paths
  - In terms of converting between the emulated filesystem and the native filesystem
  - Also in terms of dealing with paths passed to programs
- This last issue is a fairly big problem for Wine
  - How do we know that a command line argument should be interpreted as a path? Should we transparently convert it?
    - /home could mean a directory called “home” in the root directory
    - It could also be a parameter, “/HOME”, rather than a path

Filesystem differences (3)

- Case-sensitivity
  - Most Unix filesystems are case-sensitive
  - Most Windows filesystems are not
- Finding the right file can be more of a pain than might be expected
  - Need to perform case-insensitive check on all files in a directory
- How to deal with multiple files with the same name in a directory, separated by case?

Filesystem differences (4)

- Wine currently emulates a virtual drive C:
  - Stored in a user's home directory by default
- CD-ROM drives, etc, also redirected to appropriate paths in the Unix filesystem
- Entire Unix filesystem exposed as its own drive
  - Meaning applications that aren't in the virtual drive C: can still be used
- No ideal solution found to the argument conversion issue yet
  - Some kind of wrapper?

Input devices

- Ensuring full compatibility with input devices can be an issue in certain cases
- Keyboard layouts and character sets
  - Possibly differences between Windows keyboard layout and native layout (e.g., Mac keyboards)
  - Ctrl versus Apple
- Mouse input
  - Number of buttons?
  - Mouse/touchpad gestures?

Usability of Wine itself

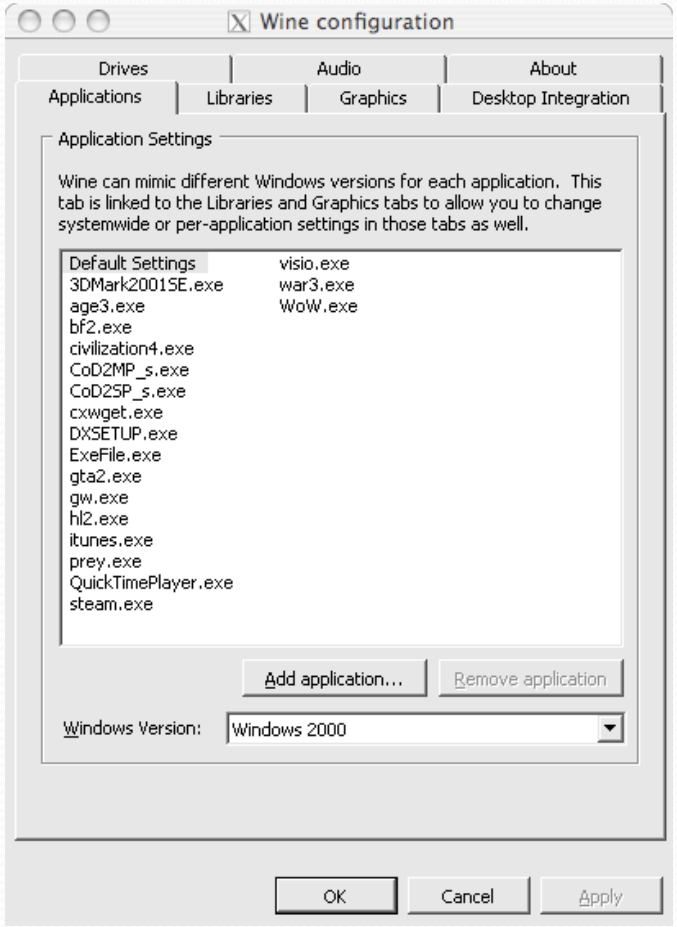
When Wine itself is the focus

- As much as possible, the user should not need to know about Wine
  - Some explanation of issues such as virtual directories, compatibility settings, and security issues is important, however
- When Wine does require attention, it should be straightforward and clear what is going on

Configuration of Wine

- Wine has a number of configurable settings
  - Graphics, sound, integration, compatibility settings and more
- Most are currently accessible through *winecfg*
- *winecfg* is comprehensive, but cluttered
  - Intimidating for the new user

winecfg interface



Configuration of Wine (2)

- Simplify *winecfg*
- Provide more user-friendly control panel applets instead
  - Resemble the configuration utilities in Windows itself
- Make configuration applets available from within the normal desktop environment
  - Back to integration again

Other interaction with Wine

- Errors and crashes are probably the other main reason users may be interacting with Wine
- Currently, call traces and similar crash data is dumped to the console
  - If not running an application from the console, this data is missed
- Pop up an error message instead
  - Possibly incorporate “check for solution on AppDB” feature
  - Ability to submit bug report

Conclusion

Conclusion

- These are just some of the issues that are faced with making Wine suitable for widespread adoption
- The other major issue, of course, is Wine's ability to run Windows software
  - With every release, we can run more software
  - Some programs run better on Wine than on Windows!

Conclusion (2)

- For Linux to become a serious competitor on the desktops of home users, the ability to run Windows applications may be considered essential
  - Particularly games
- Wine developers are working on improving integration and the general user experience
- Keep an eye out on Wine
  - Good things are happening!

Thank you for your time!

Questions?

- You can contact me by e-mail: owen@owenrudge.net
- Or check out my site at www.owenrudge.net

- More on Wine: www.winehq.org
- Desktop integration projects: http://tinyurl.com/d9be72