

The

Volume 76

PCLinuxOS

magazine

May, 2013



PCLinuxOS 64 Bit Blasts Off!

Windows Migration: The Terminal

***Windows Migration:
The Power Of The Terminal***

***Windows Migration: Accessing Linux
Partitions From Windows***

***Xfce Power User Tips, Tricks &
Tweaks: Multimedia Files***

Inkscape Tutorial: Fun With Shapes

PCLinuxOS Recipe Corner

PCLinuxOS Puzzled Partitions

Cloud Computing & The Third World

And more inside!

Table Of Contents

- 3 *Welcome From The Chief Editor*
- 4 *PCLinuxOS 64 Bit Blasts Off!*
- 6 *Windows Migration: Accessing Linux Partitions From Windows*
- 9 *Windows Migration: The Terminal – An Introduction*
- 11 *Screenshot Showcase*
- 12 *Windows Migration: Terminal – The Power Of The Terminal*
- 16 *Screenshot Showcase*
- 17 *Inkscape Tutorial: Fun With Shapes*
- 18 *Screenshot Showcase*
- 19 *PCLinuxOS Recipe Corner*
- 20 *PCLinuxOS Puzzled Partitions*
- 23 *Famous Forum Foibler: Neal*
- 24 *Xfce Power User Tips, Tricks & Tweaks: Multimedia Files*
- 33 *Screenshot Showcase*
- 34 *ms_meme's Nook: Top Of The Forum*
- 35 *Game Zone: Orcs Must Die! 2*
- 37 *Screenshot Showcase*
- 38 *Managing Running Processes From Terminal*
- 49 *Cloud Computing & The Third World*
- 51 *Screenshot Showcase*
- 52 *SLIM Desktop Manager: A Lightweight Alternative*
- 57 *More Screenshot Showcase*

The **PCLinuxOS** magazine

The PCLinuxOS name, logo and colors are the trademark of Texstar.

The PCLinuxOS Magazine is a monthly online publication containing PCLinuxOS-related materials. It is published primarily for members of the PCLinuxOS community. The magazine staff is comprised of volunteers from the PCLinuxOS community.

Visit us online at <http://www.pclosmag.com>

This release was made possible by the following volunteers:

Chief Editor: Paul Arnote (parnote)

Assistant Editor: Meemaw

Artwork: Sproggy, Timeth, ms_meme, Meemaw

Magazine Layout: Paul Arnote, Meemaw, ms_meme

HTML Layout: YouCanToo

Staff:

ms_meme
Patrick Horneker
Meemaw
Gary L. Ratliff, Sr.
Daniel Meiß-Wilhelm
AndrzejL
YouCanToo

Mark Szorady
Darrel Johnston
Andrew Huff
Pete Kelly
Antonis Komis
daiaishi

Contributors:

Pieter Louw

The PCLinuxOS Magazine is released under the Creative Commons Attribution-NonCommercial-Share-Alike 3.0 Unported license. Some rights are reserved.
Copyright © 2013.



Welcome From The Chief Editor

It's ready when it's ready. That has been the PCLinuxOS mantra for as long as I can recall. True to form, a project of over two years duration to produce a 64 bit version of PCLinuxOS bore fruit with the official release of that same 64 bit version. Started by Texstar before he took his "leave," the rest of the work was taken up and completed by the Bacon Brigade – the small band of developers and packagers who tirelessly work to maintain PCLinuxOS.

Of course, the Bacon Brigade is led by Neal Brooks, a.k.a. NealManBear. The month of April found Neal spending some time in the hospital for surgery. Texstar returned from his leave to keep things on track while Neal recuperates. Only Texstar knows if it's a temporary or permanent return, and he's not saying.

According to Old-Polack, who has been in telephone contact with Neal during his hospital stay, Neal has come through surgery in fine shape, and is in the process of healing up and getting stronger. Keep Neal in your prayers and thoughts as he recuperates. I suspect it won't be long before he's enjoying his favorite food (bacon, of course! DUH!) and a large cup of coffee.

Ms_meme, despite having "retired" her Forum Foibles and Forum Foibler columns, reprises the latter with a special Forum Foibler article this month, dedicated to Neal as he recovers. Georgetown, a.k.a. Mark Szorady, has found it necessary to end his monthly contributions to The PCLinuxOS Magazine so that he can better



focus his attentions to his business – which is producing his cartoons. I'd like to thank Mark for his continual support of The PCLinuxOS Magazine. Mark signed on during a tumultuous time when we were making efforts to revive the magazine, and has been a steady contributor since July 2009. Before coming to an end with the April 2013 issue, Mark contributed 46 consecutive "Double Take & Mark's Quick Gimp Tip" columns without ever missing an issue.

To replace the loss of these regular columns, we're going to try something a little new this month. The May 2013 issue will feature the premiere appearance of a set of puzzles created especially for PCLinuxOS users. The

puzzles include a Sudoku puzzle, a Scrabble™-like puzzle, a crossword puzzle, and a "Word Search" puzzle. You won't have to wait until the next month's issue to get the solutions to the puzzles. Instead, we're making the solutions a separate download from the magazine website. Check out the puzzle pages, called PCLinuxOS Puzzled Partitions, for more information.

Until next month, I bid you peace, happiness, prosperity and serenity.



PCLinuxOS 64 Bit Blasts Off!

by Paul Arnote (parnote)

On April 10, 2013, the L-O-N-G project of building and releasing a 64 bit version of PCLinuxOS became reality. Started by Texstar over two years ago, the much awaited and much anticipated 64 bit version of [PCLinuxOS](#) was released.

"I just wanted to say thank you to all the developers, testers and everyone who helped make this release a reality. Long live PCLinuxOS!" said Texstar.

Since Texstar has taken his leave, the push towards a 64 bit version of PCLinuxOS was taken up by the rest of the PCLinuxOS packagers and developers, sometimes referred to as the "Bacon Brigade," after their nearly unanimous favorite food.

The change required a slight modification to how PCLinuxOS RPM packages were produced, allowing both 64 bit and 32 bit versions of an application to be built from the same RPM package. Where 64 bit versions of applications weren't available, particular care was taken to insure that the 32 bit versions of the software were able to run on the 64 bit version of the operating system, via strict compatibility layers. As 64 bit versions of those 32-bit-only software packages become available, they will be repackaged to include separate 32 bit and 64 bit versions, replacing the 32 bit versions.

PCLinuxOS users who do not yet possess a computer with a 64 bit processor have nothing to fear, since the 32 bit versions of PCLinuxOS will continue to be made available for the foreseeable future.

"There are many people still running machines with perfectly good 32 bit processors, so for them the 64



bit version is not an option. Also there are still some applications that do not yet have a 64 bit counterpart. Some simply won't run without adding the 32 bit compatible libraries to the 64 bit system. This is the same for all 64 bit Linux systems, not just ours. Our goal is to provide the maximum amount of 64 bit applications possible, fill in with some 32 bit applications that will run with the compatibility

libraries, and replace the latter with 64 bit versions as they become available," said PCLinuxOS forum administrator Old-Polack.

"Just as 32 bit systems eventually replaced 16bit systems, entirely, so too will 64 bit systems eventually replace 32 bit systems, entirely, but that day is still a ways off," continued Old-Polack.

32 bit operating systems can typically address up to 4 GB of memory directly. Under a 32 bit operating system, the maximum amount of memory visible to the system is typically around 3.2 GB, since a portion of the memory is reserved for maintaining hardware registers and other items the computer needs to function properly. Up to 64 GB (62 GB of it available for use, after the computer reserves some of the space previously mentioned) of memory can be addressed on a 32 bit operating system with the use of a special PAE (physical address extension) kernel.

64 bit operating systems, on the other hand, can address up to 16 EB (exabytes) of RAM. PCs are (at this time) limited to 4 PB (petabytes) of physical memory, and this limit has not yet been reached, due to the physical size requirements of the memory chips to reach that limit.

AMD produced the first 64 bit processor for the x86 platform in 2003, with the release of the AMD64 line of processors. Since then, virtually all processor manufacturers have supplied a steady flow of 64 bit x86 processors. Intel started releasing 64 bit processors with their Intel Duo Core line of processors. If you have a reasonably new computer, chances are high that you will have a 64 bit processor nested inside.

So how do you tell if your computer is able to run the 64 bit version of PCLinuxOS? Actually, there are some surefire ways to find out (hey, this is Linux ... of course there is more than one way!). For the sake of brevity, we'll presume that you are already running PCLinuxOS (or some other Linux distro).

The first way, and the way that's available to any computer user running any operating system, is to perform an internet search for information about your CPU model.

The second way, is to type:

```
grep flags /proc/cpuinfo
```

at a command line prompt in terminal. Look in the "flags" field (one will be produced for every core of your multicore processor, if you have one) for the **lm** flag. All 64 bit processors have this flag (meaning long mode CPU), while 32 bit processors lack this flag.

Here is the output from my AMD FX-6100 six core CPU (shortened to show only the first core, since the exact same thing is reprinted for each core):

```
$ grep flags /proc/cpuinfo
flags       : fpu vme de pse tsc msr
pae mce cx8 apic sep mtrr pge mca cmov pat
pse36 clflush mmx fxsr sse sse2 ht syscall
nx mmxext fxsr_opt pdpe1gb rdtscp lm
constant_tsc nonstop_tsc extd_apicid
aperfmpperf pni pclmulqdq monitor ssse3 cx16
sse4_1 sse4_2 popcnt aes xsave avx lahf_lm
cmp_legacy svm extapic cr8_legacy abm sse4a
misalignsse 3dnowprefetch osvw ibs xop
skinit wdt lwp fma4 nodeid_msr topoext
perfctr_core arat cpb npt lbrv svm_lock
nrip_save tsc_scale vmcb_clean flushbyasid
decodeassists pausefilter pfthreshold
```

Compare that to the same command ran on another of my computers with a Pentium M single core processor:

```
$ grep flags /proc/cpuinfo
flags       : fpu vme de pse tsc msr
mce cx8 sep mtrr pge mca cmov clflush dts
acpi mmx fxsr sse sse2 ss tm pbe up bts est
tm2
```

Notice the absence of the **lm** flag, indicating that my Pentium M processor is a 32 bit processor. Another way of putting this would be that the 64 bit version of PCLinuxOS is **not** capable of running on my computer with the Pentium M processor. For this particular computer, I will have to remain with the 32

bit version of PCLinuxOS (which runs exceptionally well, I might add).

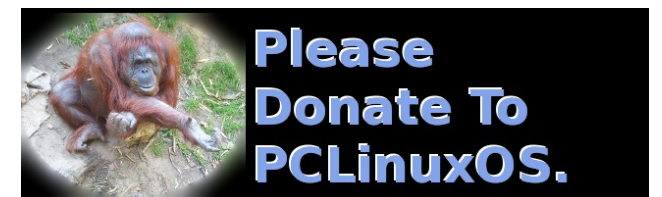
However, just because you recently installed PCLinuxOS, don't just assume that you are running a kernel that's capable of running 64 bit applications. Again, at a command prompt in a terminal session, type **uname -p**. If it returns i386, i486, i586 or i686, you are running a 32 bit kernel, and you are not able to run 64 bit applications under a 32 bit kernel. However, if it returns x86_64, you are running a 64 bit kernel, and you are able to run 64 bit applications.

Alternately, if you type **getconf LONG_BIT** at a command line prompt in a terminal session, it returns **32** if you are running a 32 bit kernel, and **64** if you are running a 64 bit kernel.

Caveats

If you have a Windows 8 Certified machine ... be sure to disable UEFI before attempting to install this (or any) version of PCLinuxOS on your machine. The bootloader does not yet support the UEFI specification at this time.

If you enjoy using WINE to run some of your "favorite" Windows applications, the WINE package in the 64 bit repository contains two RPMs. The first is the base package for WINE, and the other is a 32 bit compatibility package that allows 32 bit Windows applications to run on 64 bit PCLinuxOS.



Windows Migration: Accessing Linux Partitions From Windows

by agmg (Antonis Komis)

Dual booting is a very common scenario among Windows users who decide to try Linux. They dual boot because they don't yet feel comfortable enough to leave Windows and migrate completely to Linux, or simply because they need some programs or games that can't run under Linux. In both cases, when you find yourself between two worlds, you want these two worlds to be able to communicate with each other. Linux has native support for Windows file systems (NTFS, FAT), meaning that you'll be able to access your Windows partitions from your Linux installation. This is not the case, though, with Windows. To access your Linux partitions from Windows, you need third-party software, specifically designed for this purpose.

In this article, I will present a couple of applications that will allow you to access your Linux partitions from your Windows installation. Each one of these applications has support for the **ext4** filesystem, which most Linux distributions use by default. Of course they also support older versions of **ext** and even other filesystems, too.

All programs that I've decided to present in this article treat Linux partitions as read-only by default. This means that you'll be able to read from those partitions but not write to them. I believe this is the best option, because, **having complete access to your Linux partitions can be dangerous!** The Linux filesystem would be completely exposed, and the altering or deleting of a single file could render your Linux installation unusable. You must know exactly what you are doing before deciding to do it. This way, the risk of messing things up is kept to a minimum.

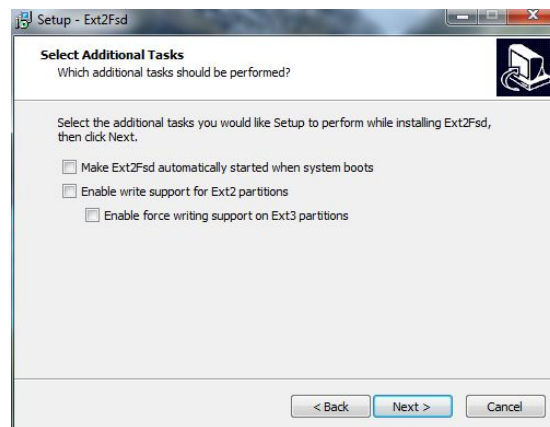
Ext2Fsd

Official page: <http://www.ext2fsd.com/>

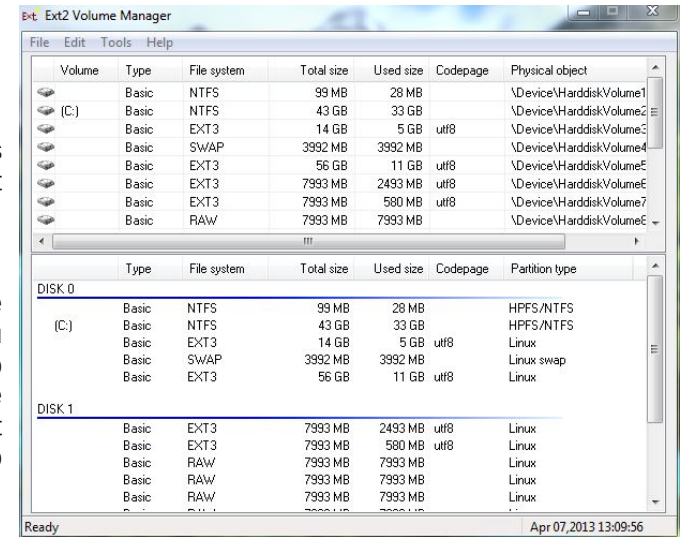
Ext2Fsd is an ext2 file system driver for Windows (2000, XP, Vista and Win7). When installed, it assigns a drive letter to each partition you select, allowing access to them from any Windows program.

During program installation, you will be given the option to enable **Ext2Fsd** service at startup. If you do, the service will run each time you boot into Windows, mounting the Linux partitions you have specified. If you don't, you'll have to manually start the service each time you want to have access to those partitions.

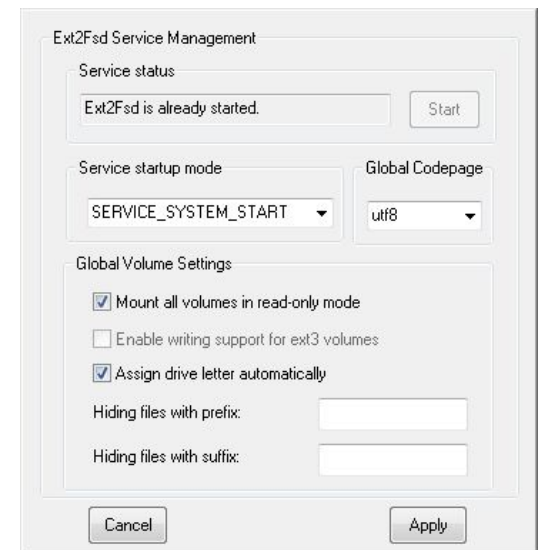
Additionally, Ext2Fsd gives you the option to have write access to Linux filesystems. For the reasons I've mentioned earlier, I suggest you leave this option disabled, unless you really need it and you really know what you're doing.



Let's have a look at the main program window (top, right):

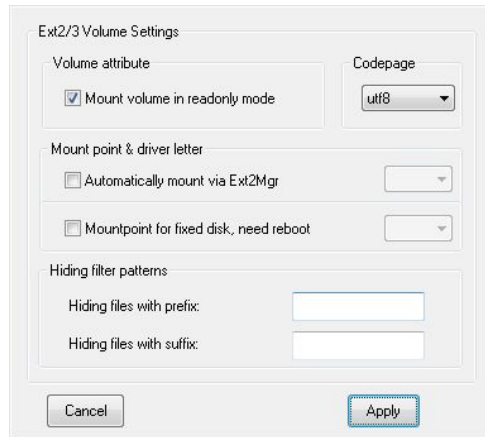


By pressing F7 (Tools -> Service Management), you can see the status of the service and some options on volume handling:



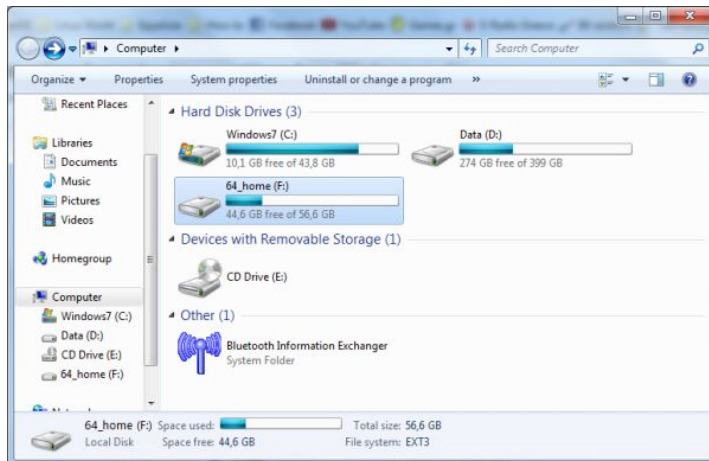
Windows Migration: Accessing Linux Partitions From Windows

Now, let's mount a partition. Right click on the partition that interests you and select **Ext2 Volume Management** (or simply press F3 with the desired partition selected). You will see the next window:

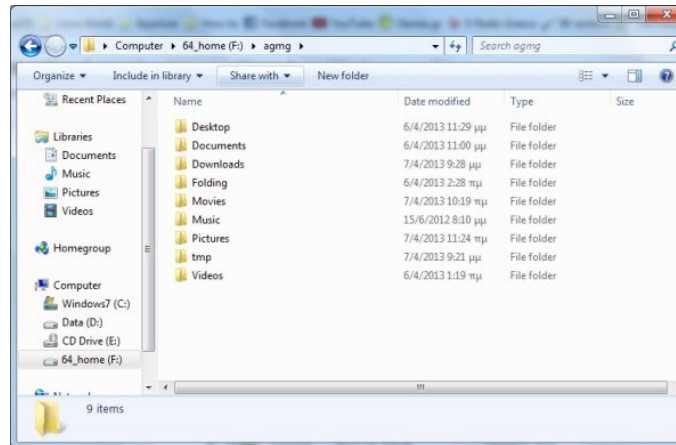


If the partition you want to have access to is located on a removable disk, select the option **Automatically mount via Ext2Mgr**. This will mount the partition each time you connect your disk. If it is a fixed disk, then the option **Mountpoint for fixed disk, need reboot** is the right choice in this case. Make your choice and perform a reboot, if needed.

Let's go to My Computer:



You can see that my Linux partition (64_home) is mounted and I can access it like a normal Windows partition. The program reports this partition as **ext3** but it's actually ext4 and the program manages it without any issues. I now have access to my data:



DiskInternals Linux Reader

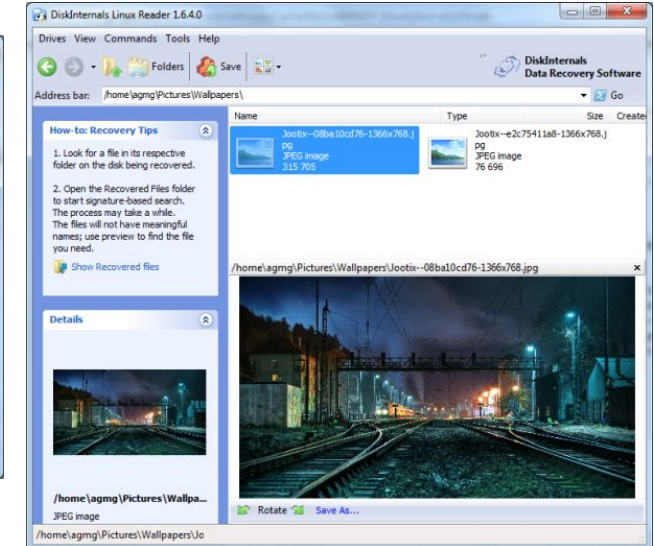
Official page:
<http://www.diskinternals.com/linux-reader/>

DiskInternals Linux Reader is a freeware program that supports Ext2/Ext3/Ext4, HFS and ReiserFS file systems.

Unlike **Ext2Fsd**, **Linux Reader** provides only read-only access to Linux partitions and does not allow you to make records on them. This guarantees that the interference in an alternative file system will not affect the work of Linux later.

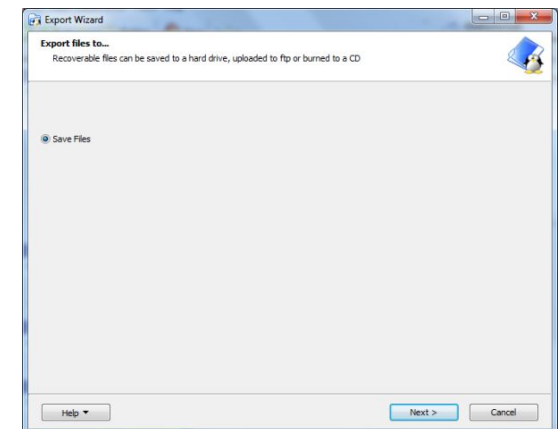
Linux Reader doesn't provide access via a drive letter – it's a separate application you launch to browse your Linux partitions. It has some nice features, such as the ability to use common Windows Explorer for extracting data and a Preview option. This option only supports specific file types (some image types, some

video formats, office documents, text files, HTML files, and a few others which I was unable to test). If you are looking for files that happen to be supported by Linux Reader's built-in file preview, you can preview the file(s) before saving them.



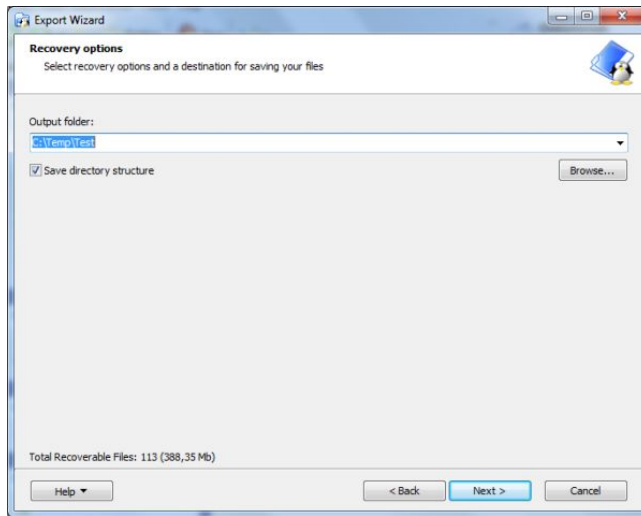
You can see the Preview option in action. By clicking on **Save as...**, the selected file can be saved to my Windows filesystem. This is necessary if I want to work on that file, because of the read-only access. The program can also save entire directories.

Right click on a folder and select **Save**



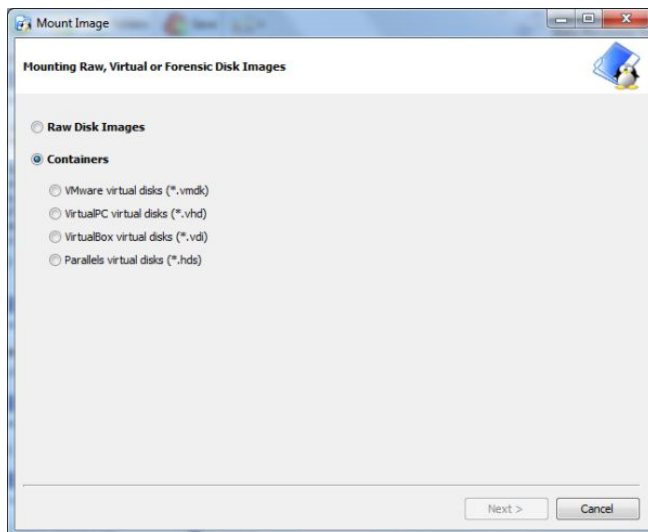
Windows Migration: Accessing Linux Partitions From Windows

Select **Save Files** and click **Next**



Select an output folder and you're done.

Another interesting feature is the ability to mount Raw disk images (such as ISO files) and Containers (such as VirtualBox virtual disks). I didn't test this feature thoroughly but seems like a nice addition.

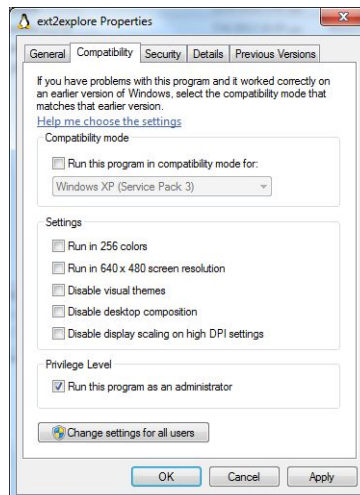
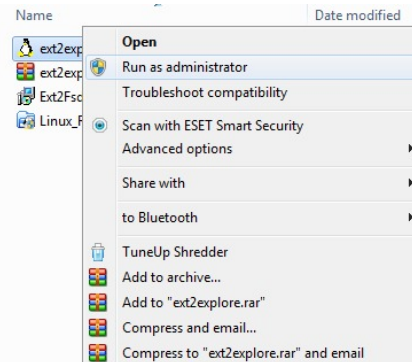


Ext2Explore

Official page: <http://ext2read.blogspot.com>

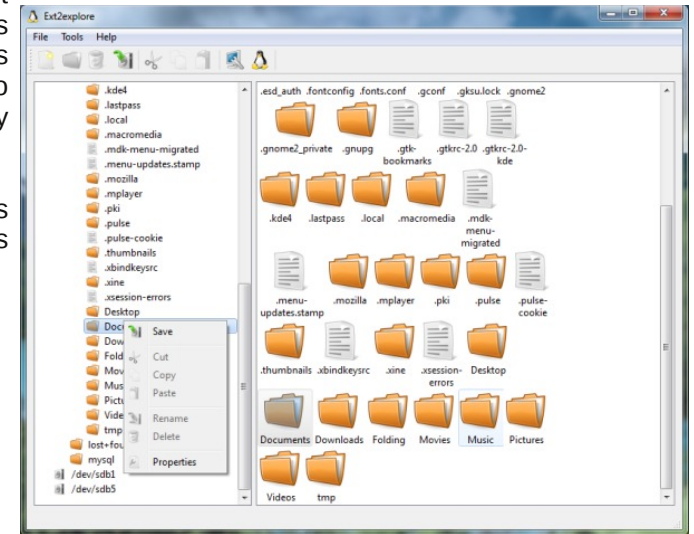
This program works similarly to Linux Reader but only supports Ext4, Ext3 and Ext2 filesystems. It has only one unique feature compared to the previous two: it is portable. This means that you don't have to install it on your system. It can be run from any location, including a removable drive.

The **Ext2explore.exe** program must be run as administrator or you'll get an error – you can do this from the right-click menu.



To save some time in the future, go into the file's properties window and enable the "Run this program as an administrator" option on the Compatibility tab.

As with **Linux Reader**, you'll have to save a file or directory to your Windows filesystem before you can open it in other programs.



Conclusion

Although the programs presented in this article can fulfill the purpose they were written for, there is another option for sharing files between your Windows and Linux partitions, much more convenient and easy to use. You can create an extra partition, formatted with a Windows file system that both systems will have access to. FAT32 has native support in both Windows and Linux, and Linux has no problems reading or writing to a NTFS-formatted partition. This way, you will be able to view and modify your files anytime, from any system, without the need of third party software. If you have some files that you want to be able to access from either operating system, simply copy them to the extra partition formatted in a file system that Windows can read.

Windows Migration: Terminal – An Introduction

by Pete Kelly (critter)

If you are reading this, then you are most probably a PCLinuxOS user who has previously only used Windows. That simple fact suggests that you are someone a little bit special, someone who doesn't accept just whatever is offered, but someone who wants to look at other options and make their own decisions. Someone who wants to explore and to see if there is more to something than it seems at first glance.

After you have been using Linux for a couple of weeks and you have become accustomed to where things are and what they are called, you may realize that, for the most part, you are working much as you used to in Windows: pointing, clicking, cutting and pasting, dragging and dropping and so forth. So what's the big deal about Linux, apart from being free?

I could mention that you don't need to use anti-virus programs, although they are available if you really want them, or I could talk about how reliable it is, but this is supposed to be about the terminal in Linux so let's talk about that.

What you have been using so far is a graphical interface to the system, but that is not the only way that you can interact with the system, and it is not the way that Linux was designed. Sure, the graphical applications with their buttons and menus make things easy to use and nice to look at too. That's fine, and it's the way that most people spend most of their time, myself included, but there are times when it is just not the best way to do things, and there are some things that just can't be done in the graphical environment. It is at these times that

we turn to the other way, the text way, usually known as the command line interface.

This command line interface is not unique to Linux. Windows and Mac machines also have one, although they are not used as much by ordinary users. The command line interface (I'm going to call it the CLI from here on out, since I tired of typing it out in full) is capable of unleashing the full power of Linux, but that is not the purpose of these articles. Here, we will discuss only the basic things that users should know about using the CLI.

Why bother?

Well there are quite few good reasons to learn to use this CLI.

One reason is that if you seek help from some of the more knowledgeable users on the forum or in the IRC support channel about a difficulty that you are experiencing, then they may ask you to type something into a terminal, and you should be familiar with that process.

Some things are done more easily with the CLI, and some things can be done only in the CLI.

There are lots of distributions of Linux, all with their own flavours and, unlike Windows, there are many graphical environments such as KDE, LXDE, Gnome and Xfce, that operate in a slightly different manner. But once you are familiar with the CLI, you can feel at home in any of these distributions.

In a graphical application, you are restricted to using only the tools and commands that the software developer decided to include. By using the terminal, this restriction is lifted, you decide which commands

to use, and how to use them. You can easily create your own tools and routines that do **exactly** what you want. You are in total command and you can talk to the system in a language that it understands.

Yet another reason to learn to use this CLI is to satisfy your curiosity, and to discover just what you can achieve in Linux, what with you being a little bit special n'all.

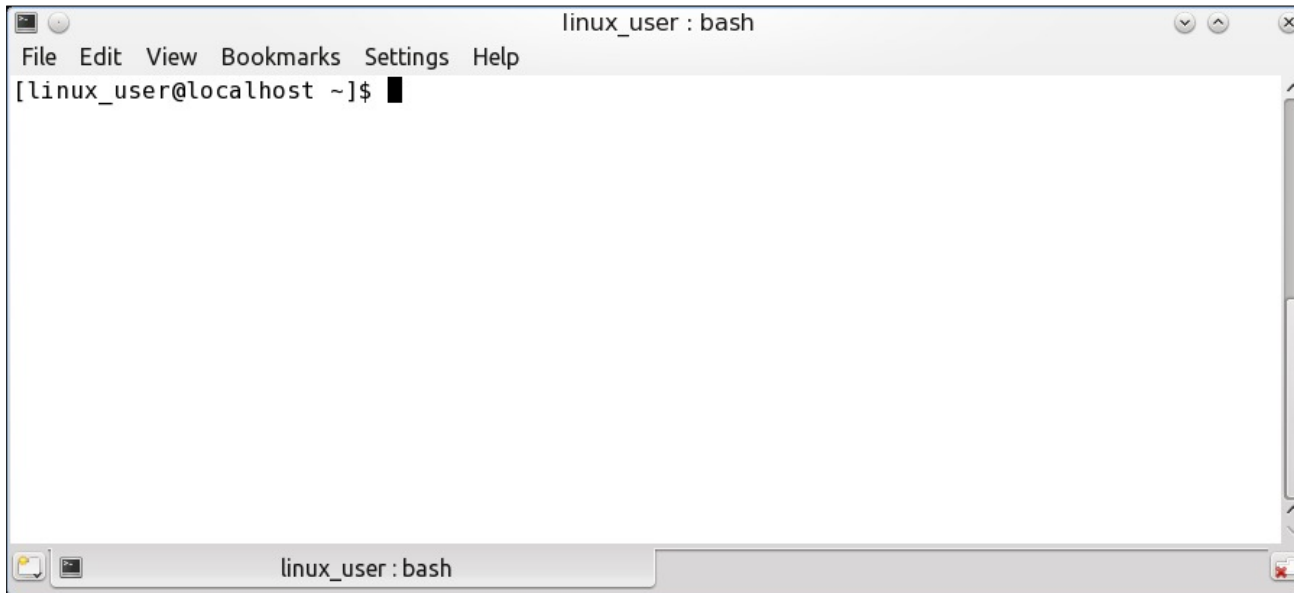
A terminal is how we use the CLI, and we usually use a terminal emulator application that runs in your familiar graphical environment, although this is not the only way. If you are using the KDE environment then you will have a terminal emulator called Konsole, but which one you use doesn't matter. If you use Xterm, Eterm or any of the many others, the results will be the same. If somebody tells you to open a terminal, then this is what they mean.

I'll use Konsole here, simply because I have a shortcut to it on my desktop.

Only ever open a terminal as root when absolutely necessary – the so called 'super user,' as you can damage the system with a wrongly constructed command when working as root. Sometimes you may be instructed to do so when seeking help from more experienced users, and even then, you should go carefully and reread what you have typed before pressing the Enter key. There are safer ways to do things when you need special privileges, which will limit the possibilities of causing accidental damage. We will cover these methods in a later article.

Getting started

Open a terminal. Yours may look slightly different.



Ignore the menu headings in the picture and look at the first line, the one that ends in \$. This is known as the prompt and it is *prompting* you to type in something. To demonstrate the use of the terminal I have created the unimaginatively named user `linux_user`. Your user name should appear in its place.

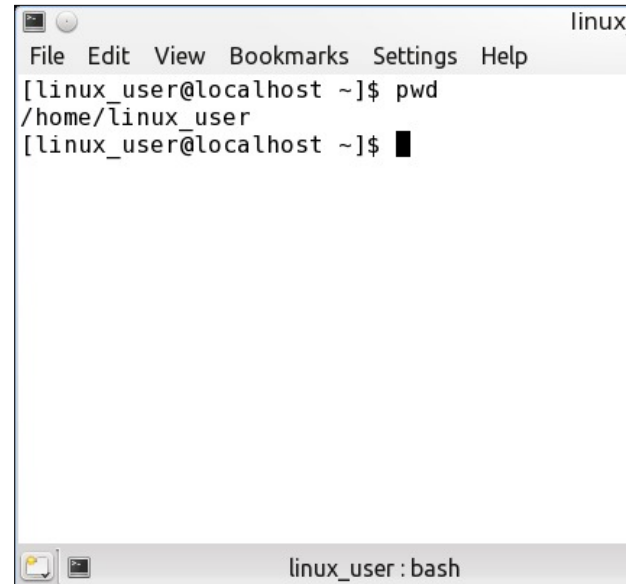
What appears here is fully customizable, but in a default PCLinuxOS installation it will show `user_name @ hostname` and the current directory. Here the squiggly character or tilde is used as a shorthand way of saying “my home directory.”

A directory is the same as a folder in Windows and, since Linux is a multi-user system, each user is allocated their own 'home' directory to store their personal files and settings. The hostname is the name of this computer as it will be seen on the local network which, if not set, will default to the word `localhost`.

To verify that we are actually working in our home directory we can type in the command **pwd** after the dollar sign (\$), and hit the enter key. The `pwd`

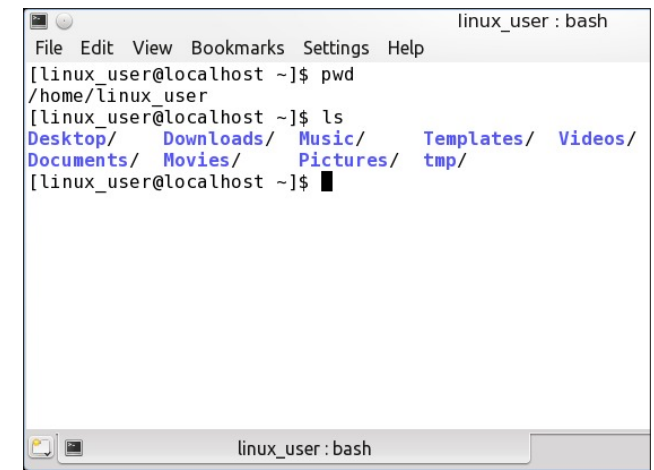
command means 'print working directory.' Make sure that you use lower case letters for the command, as Linux is case sensitive.

The command is executed and `/home/linux_user` is printed to the screen verifying that we are in fact in



our home directory, and the the prompt is repeated waiting for us to type in something else for it to work on. You may also have noticed that there is no drive letter, as you find under Windows. This is perfectly normal. Linux doesn't use drive letters, as the file system is structured differently. You may find this rather strange at first, but you will soon become familiar with it.

The next command I want to use is **ls** (small L, small S) which means 'list the contents of this directory' (This is like the **dir** command in MS-DOS or on the windows command line). Type **ls** at the prompt and press the Enter key.



The names of the files and sub-directories or folders in the directory are printed to the screen.

So that, essentially, is how the alternative, text based interface to the operating system is implemented. In the next few articles, we shall learn some more about the terminal and the CLI. Enough to get you on course to be able to follow directions when needed and to do the basic operations that will get you around in the terminal.

If you get through these basic instructions, then you will have a good understanding of what the CLI is all about. Mastering it will take a little longer. It is a skill

which is not necessary to be able to use Linux, but is still an extremely useful tool to have.

When you are ready to learn more about the CLI and dig a little deeper into the Linux command line, you can download the [Command Line Interface Intro Special Edition](#) of The PCLinuxOS Magazine. This edition of The PCLinuxOS Magazine presents all 12 articles from the pages of the magazine, arranged in an orderly way to teach you the command line basics from the very beginning.

With the graphical interfaces, such as KDE, you can **use** Linux. In the terminal you can **control** Linux. How much control you have depends upon your level of understanding of the Linux system, but a very basic knowledge can enable you to master some of the things are not so easily achieved using only the graphical interface.

Does your computer run slow?

Are you tired of all the "Blue Screens of Death" computer crashes?



Are viruses, adware, malware & spyware slowing you down?

Get your PC back to good health TODAY!

Get



Download your copy today! FREE!

Reach Us On The Web

PCLinuxOS Magazine Mailing List:
<http://groups.google.com/group/pclinuxos-magazine>

PCLinuxOS Magazine Web Site:
<http://pclosmag.com/>

PCLinuxOS Magazine Forums:
<http://www.pclinuxos.com/forum/index.php?board=34.0>

Visit Us On IRC

- Launch your favorite IRC Chat Client software (xchat, pidgin, kopete, etc.)
- Go to freenode.net
- Type "/join #pclosmag" (without the quotes)

Screenshot Showcase



Posted by ff103, April 7, 2013, running KDE.

Windows Migration: The Terminal – The Power Of The Terminal

by Pete Kelly (critter)

In the previous article, I showed you how to access a terminal, how to type in some very simple commands, and what results you could expect to see from them. Nothing very impressive or spectacular, but it worked. It showed that using the terminal in Linux is no black art. It is not something reserved only for computer geeks or the very knowledgeable, but it is something that can be used by anybody that can enter a few simple commands on the keyboard.

So far, you have used only two commands, **ls** & **pwd**, but I am sure that you can see that, although these are strange words, they are not mystical incantations from Hogwarts. Rather, these are but a simple way of instructing the system to perform a function on some file or data or to provide some information. There is no sorcery involved in using the terminal, but there is a little learning curve.

To get any significant benefit from using the terminal, you need to be comfortable in its use, and be able to perform actions at least as easily as you do in a graphical environment. You don't need to master it, just be comfortable with it. The X windowing system, the graphical system used by Linux, was not meant to replace the text based command line, but to complement it, and this is still true today. Although you can get along just fine using the graphical interface only, making use of both interfaces gives you access to the total power and flexibility of the Linux operating system.

Under the hood, Linux is different from Windows, and it will take a little while to get used to those differences. I can show you what is available and how to use it, but I shall leave you to decide just how

far you want to go and let you explore the possibilities. There are many sources of reference to get a deeper understanding of the topics presented here. The intention of these articles is to make you aware of what is possible, and of how 'Linux - the operating system' works behind the graphical desktop you usually see.

Don't be put off by this; it is by no means as difficult as is often described. A good understanding of just a few of the many hundreds of available commands will get you started on a new level of control of your Linux environment. The real power of the terminal is not knowing all of the commands, but knowing how to use them. In the terminal, you can have total control of all of the features of the commands, not just a sub-set as deemed adequate by the designer of a higher level graphical application. You can make your own commands by combining existing commands to achieve exactly the result you require. There is always more than one way to do something and there is little that is more satisfying than to do it 'your way.' That is the real power of the terminal.

Linux is different

When the Unix operating system, on which Gnu/Linux is based, was devised, a few very important, visionary ideas were introduced. Some equally important concepts were also established.

1. All of the tools and utilities should be designed to do just one thing, but to do it well.
2. There should be a way of combining the effects of applying multiple tools to a set of data. This led to the introduction of the 'pipe' to allow data to flow between tools and through filters to provide a

compound, transitional effect to the data. This is known as a data flow or stream.

3. The resultant modified data stream should be re-directed to one or more new destinations, and the input data should be available to be read from any selected source.

4. Errors and system information should have a distinct, concurrent output path that may be logged, reported or combined with the data output stream.

Add to this the fact that Linux/Unix treats everything as a file – even your mouse is treated as a file – and you can start to see that Linux is a different beast. The fact that it has been around for over four decades, (dating back to its Unix roots), and that it has been used by many millions of people, and by many thousands of major organisations during that period, is testament to its reliability and usefulness.

If you have used the Windows command line or MS-DOS, then you may well have used the wildcards ? & * to represent unknown characters in filenames, and you may even have written or used batch files. If you have used the new Windows "powershell," then you will have had access to regular expressions, to a command shell, and to an advanced scripting language. Linux has had these features since day one, and a whole lot more.

During the course of these introductory articles, I will show you some commands that are often used, and how to combine them. Occasionally, I shall break off to explain an important concept which you will not have encountered when using Windows.

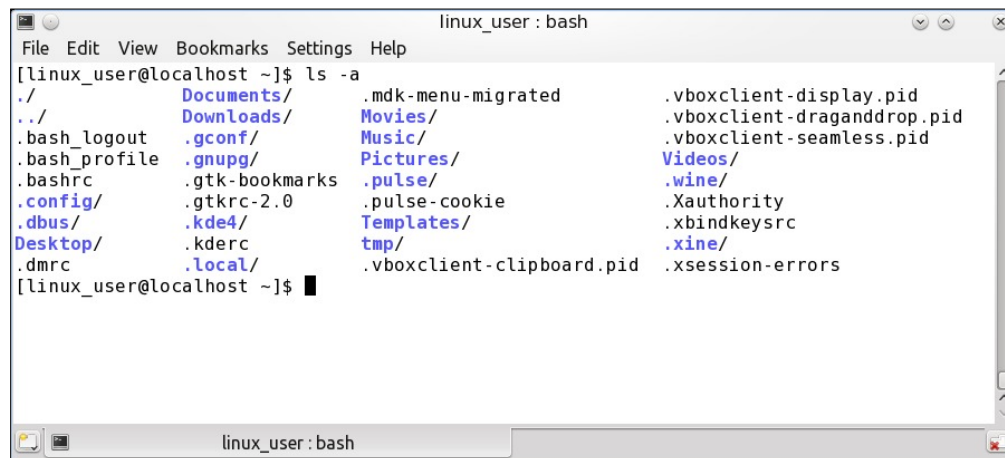
To start you off, I will show you what can be done in the terminal with the simple **ls** command and three or four other very simple commands.

Keep it simple, do it well

The **ls** command lists the contents of a directory (or directories). That's it. That is what the **ls** command was designed to do, and that is what it does. It does, however, do it extremely well. By applying certain options, we can adjust the output. We can expand or limit the amount of data output by the command. We can produce it in different formats. We can sort the output. We can include subdirectories, and we can include or exclude certain types of files. In short, we can demand to be shown exactly the data we want and only that data.

When you typed in the command **ls**, the terminal responded by printing a list of the contents of your home directory. Well actually, it only printed out some of the contents. Some files and directories are hidden. There is nothing sinister here. They are your files and you have a right to see them. They are hidden because you rarely need to see them. Hiding them simply reduces clutter under normal use.

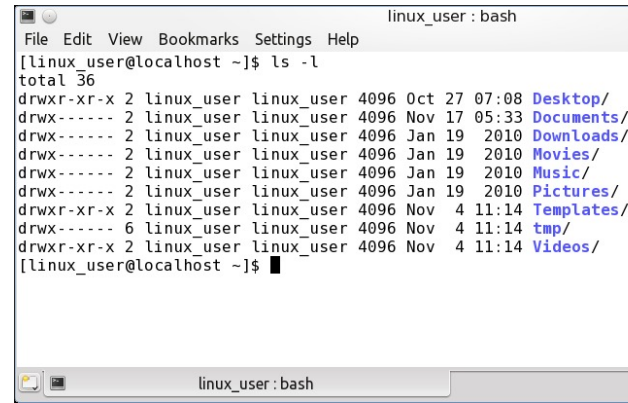
A hidden file name begins with a period. To show these, we need to change the **ls** (list) command, and we can do this by adding an **option** to the command. There are many options, and they usually begin with one or two hyphens. The one we are looking for is **-a**. The command **ls -a** translates as 'list all.'



```
linux_user : bash
[linux_user@localhost ~]$ ls -a
./          Documents/  .mdk-menu-migrated  .vboxclient-display.pid
../         Downloads/  Movies/             .vboxclient-draganddrop.pid
.bash_logout .gconf/    Music/              .vboxclient-seamless.pid
.bash_profile .gnupg/    Pictures/           .wine/
.bashrc      .gtk-bookmarks .pulse/            .Xauthority
.config/     .gtkrc-2.0 .pulse-cookie      .xbindkeysrc
.dbus/       .kde4/     Templates/         .xine/
.Desktop/    .kderc    tmp/              .xsession-errors
.dmenu       .local/   .vboxclient-clipboard.pid
[linux_user@localhost ~]$
```

Now you can see why they were hidden. Most of those hidden files will mean absolutely nothing to you – yet!

Another useful option is **-l**, and this tells the **ls** command to produce a long listing.



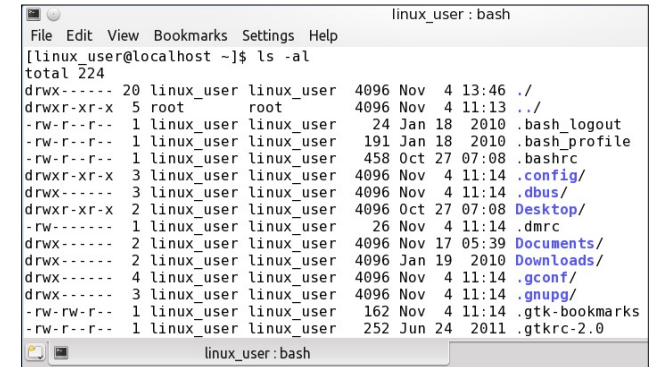
```
linux_user : bash
[linux_user@localhost ~]$ ls -l
total 36
drwxr-xr-x 2 linux_user linux_user 4096 Oct 27 07:08 Desktop/
drwx----- 2 linux_user linux_user 4096 Nov 17 05:33 Documents/
drwx----- 2 linux_user linux_user 4096 Jan 19 2010 Downloads/
drwx----- 2 linux_user linux_user 4096 Jan 19 2010 Movies/
drwx----- 2 linux_user linux_user 4096 Jan 19 2010 Music/
drwx----- 2 linux_user linux_user 4096 Jan 19 2010 Pictures/
drwxr-xr-x 2 linux_user linux_user 4096 Nov 4 11:14 Templates/
drwx----- 6 linux_user linux_user 4096 Nov 4 11:14 tmp/
drwxr-xr-x 2 linux_user linux_user 4096 Nov 4 11:14 Videos/
[linux_user@localhost ~]$
```

That gave a lot more information but omitted the hidden files. To include them, we can combine the options like this (image at top of next column).

Type **ls --help** and you will see that there are many more options that you can use. Some of the options may be meaningless to you. Just ignore them. If you don't understand them, then you probably don't need them. Experiment with those that interest you, it is perfectly safe to do so as the command only prints information to the screen. Most commands include this basic usage help.

Data flow & pipes

When you typed in **ls --help**, some of the information scrolled off the screen, as there was quite a lot



```
linux_user : bash
[linux_user@localhost ~]$ ls -la
total 224
drwx----- 20 linux_user linux_user 4096 Nov 4 13:46 ./
drwxr-xr-x 5 root root 4096 Nov 4 11:13 ../
-rw-r--r-- 1 linux_user linux_user 24 Jan 18 2010 .bash_logout
-rw-r--r-- 1 linux_user linux_user 191 Jan 18 2010 .bash_profile
-rw-r--r-- 1 linux_user linux_user 458 Oct 27 07:08 .bashrc
drwxr-xr-x 3 linux_user linux_user 4096 Nov 4 11:14 .config/
drwx----- 3 linux_user linux_user 4096 Nov 4 11:14 .dbus/
drwxr-xr-x 2 linux_user linux_user 4096 Oct 27 07:08 Desktop/
-rw----- 1 linux_user linux_user 26 Nov 4 11:14 .dmenu
drwx----- 2 linux_user linux_user 4096 Nov 17 05:39 Documents/
drwx----- 2 linux_user linux_user 4096 Jan 19 2010 Downloads/
drwx----- 4 linux_user linux_user 4096 Nov 4 11:14 .gconf/
drwx----- 3 linux_user linux_user 4096 Nov 4 11:14 .gnupg/
-rw-rw-r-- 1 linux_user linux_user 162 Nov 4 11:14 .gtk-bookmarks
-rw-r--r-- 1 linux_user linux_user 252 Jun 24 2011 .gtkrc-2.0
```

of it. The standard tool for displaying a text file on the terminal screen in Unix is a command named **more**. It reads the data from the file, and then displays one screenful of information at a time, waiting until you press the spacebar. When the spacebar is depressed, it will give you another page, or 'more.' Unfortunately there was no way of going backward to re-read the text, and so a new tool was introduced to include this functionality. What else could it be called but **less** (modern versions of **more** do allow bi-directional scrolling, but for many reasons, I would suggest that you stick to using **less**. After all, "less is more, more or less.") To use these commands, we introduce the vertical bar, or 'pipe' character "|." The data that is output from the first command is piped through to the next command as it is required.

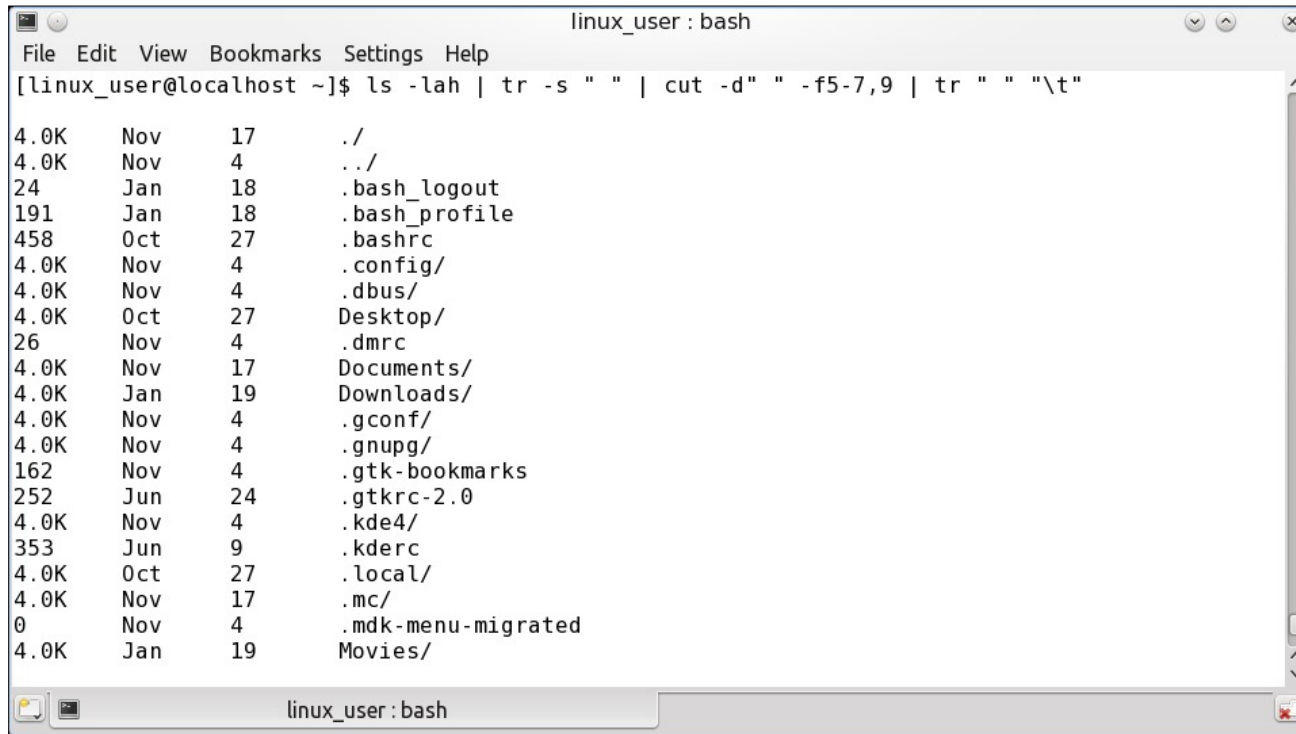
ls --help | less

Now we can use the arrow and page-up/page-down keys to read the information at our own pace. Press 'q' to exit the command.

In this manner complex commands can be constructed, adding commands and filters to produce exactly the results we require. Look at the following example (top of next page).

The **ls** command is passed options to produce a long listing of all files with sizes in human-readable form. This is piped to the **tr** command, which

Windows Migration: The Terminal – The Power Of The Terminal



```
linux_user : bash
File Edit View Bookmarks Settings Help
[linux_user@localhost ~]$ ls -lah | tr -s " " | cut -d" " -f5-7,9 | tr " " "\t"
4.0K  Nov  17  ./
4.0K  Nov  4   ../
24   Jan  18  .bash_logout
191  Jan  18  .bash_profile
458  Oct  27  .bashrc
4.0K  Nov  4   .config/
4.0K  Nov  4   .dbus/
4.0K  Oct  27  Desktop/
26   Nov  4   .dmrc
4.0K  Nov  17  Documents/
4.0K  Jan  19  Downloads/
4.0K  Nov  4   .gconf/
4.0K  Nov  4   .gnupg/
162  Nov  4   .gtk-bookmarks
252  Jun  24  .gtkrc-2.0
4.0K  Nov  4   .kde4/
353  Jun  9   .kderc
4.0K  Oct  27  .local/
4.0K  Nov  17  .mc/
0    Nov  4   .mdk-menu-migrated
4.0K  Jan  19  Movies/
```

squeezes out extra spaces. This is then passed to the **cut** command, which removes all columns except columns 5, 6, 7 and 9. Finally, this is again given to the **tr** command which now converts spaces to tabs. This is a rather complicated example, but it demonstrates the principle of chaining together multiple commands to arrive at just the result you require. Ignore the details for now; just try to understand the basic idea.

Redirection

Commands in Linux have an input and an output even though, at first glance, some seem not to have. The **cd** command, which is used to change your working location to another directory, appears to have no input and no output. But it does have both, even though the input is not used. Where this input and output come from or go to is arbitrary. When

input or output is unspecified, a command will use what are known as 'standard input,' stdin, usually the keyboard, and 'standard output,' stdout, usually the terminal screen. We have seen an example of stdout with the **ls** command. When we used the pipe symbol '|' the commands output was redirected to another command.

We can also use the < and > characters to redirect stdin and stdout, respectively. It works like this. Suppose we want to save a listing of the files in our Documents directory to a file named docs-list, instead of just printing it to the screen. We would then issue the command **ls Documents > docs-list**.

If the file doesn't exist, then it will be created, and then the output from the **ls** command will be written to it. If the file exists, then the contents will be overwritten with the new data. To get around this, we

can use two greater than characters, thus **ls Documents >> docs-list**.

The new output is then appended to the file. If we want to sort the contents of the file, then we can feed the file to the input of the command, which will do our sorting. Unsurprisingly the name of the command we need is **sort**.

sort < docs-list

This sorts the contents, but prints them out to the screen, leaving the original file untouched. To produce a sorted file in one step we combine redirection (<,>) and pipes (|) like this:

ls Documents | sort > docs-list

The **ls** command produces output, which it pipes to the **sort** command, which then sorts the data and redirects its output to the destination file.

Error messages

In Linux, **stdin** and **stdout** are known as file descriptors, which are a bit like channels that carry data. There can be many active file descriptors, but these are reserved for the operating system, which refers to them as file descriptor 0 and 1. There is a third file descriptor reserved for system use, known as 'standard error' **stderr**, and this carries the number 2.

Commands used in Linux are programmed to output data to **stdout** and to output error messages and other information that is not part of the commands intended output to **stderr**. By default, **stderr** writes any output to the terminal screen, but that is often inconvenient. You probably don't want error messages mixed with valid data, and if you are running a long process in your absence, perhaps a backup, you may want to save any error messages to a log file that you can refer to when the process completes. This can easily be done by redirecting

stderr, or file descriptor 2. If you want to log the output, use a command like this:

command 2> logfile

command 2> logfile > datafile

The first would write any error messages from the command to the file logfile, but regular output would still go to the screen. The second would send errors to logfile and data to datafile. No output would appear on the screen.

To simply get rid of any error messages, Linux has a special device called **/dev/null**. Although a device is usually perceived as a physical device, such as a keyboard or a network card, Linux has some devices that have no physical presence. The device **/dev/null** is one of these. As Linux treats everything, including devices, as files, you can write to **/dev/null**, which sends all data it receives to irretrievable oblivion.

command 2> /dev/null

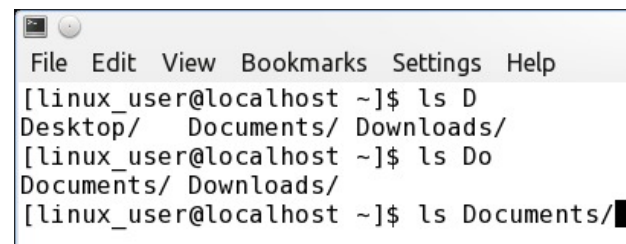
The Shell

All of this makes for an extremely flexible and powerful set of tools that you can use to work with your files and with the system configuration files. The text based environment in which you use these tools is controlled by a program known as a shell, which interprets what you type before deciding how to execute it. There are many different shells available but the most common one, and the default shell in PCLinuxOS, is named **bash**.

Bash is extremely powerful and very friendly. Over the many years of its use, lots of features have been added to assist and accelerate your work in the terminal. One of these helpful features is known as **command completion**.

Type **ls D**.

Press the tab key. Bash is intelligent enough to realise that you have entered the command **ls** and are now trying to enter the name of a file or directory so that you can list out the relevant information. The shell, bash, responds to the tab key by listing all of the files and directories, that it can see from here, which begin with an uppercase D. The directories Desktop, Downloads and Documents were listed on my system. Typing an additional **o** and hitting tab reduces the list to two, as **Desktop** is not a match. Enter a **c** and there is now only one possibility, **Documents**, and so, by pressing tab again, that is automatically filled in for me at the prompt ready for me to hit Enter, which produces my listing.

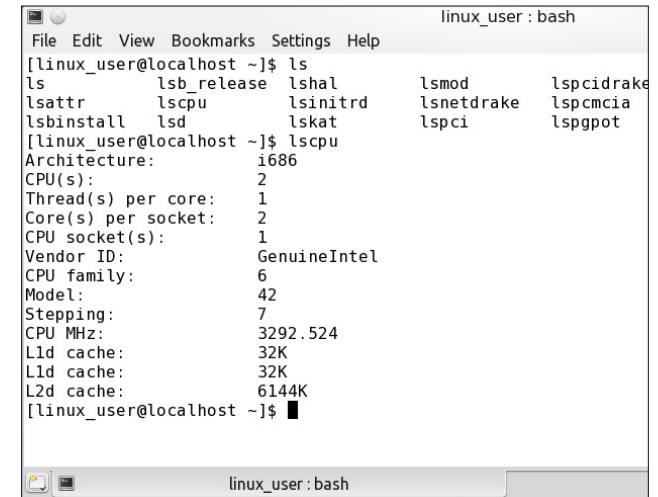


```
File Edit View Bookmarks Settings Help
[linux_user@localhost ~]$ ls D
Desktop/  Documents/ Downloads/
[linux_user@localhost ~]$ ls Do
Documents/ Downloads/
[linux_user@localhost ~]$ ls Documents/
```

This is a real time saver, and can also help when you are unsure of the name of a command. For example, I know that there is another command that begins with **ls**, and that will give me information about the machines processor.

I type **ls** and hit Tab. As this is the first thing after the command prompt bash realises that a command is being attempted, and I am shown all of the commands that the system knows and that begin with **ls**. The one I want is named **lscpu** (I should have guessed that!). I type **c**, hit the Tab key and the command name **lscpu** is completed for me at the prompt. I then press the Enter key, and there is my information (image top of next column).

When you have typed in a command and want to repeat it, you do not have to retype it. Press the up arrow key, and your previous command will appear ready for you to edit or use. Continue pressing the up and down arrow keys to scroll through your previously typed commands.



```
linux_user: bash
File Edit View Bookmarks Settings Help
[linux_user@localhost ~]$ ls
ls          lsb_release  lshal        lsmod        lspci-drake
lsattr      lscpu        lsinitrd     lsnetdrake   lspcmcia
lsbinstall  lsd          lskat        lspci         lspgpot
[linux_user@localhost ~]$ lscpu
Architecture: i686
CPU(s):       2
Thread(s) per core: 1
Core(s) per socket: 2
CPU socket(s): 1
Vendor ID:    GenuineIntel
CPU family:   6
Model:       42
Stepping:    7
CPU MHz:     3292.524
L1d cache:   32K
L1d cache:   32K
L2d cache:   6144K
[linux_user@localhost ~]$
```

There is also a feature called reverse search. To use it, press **Ctrl + R** and start typing what you can remember of a previous command. As you type, the most relevant previous command is placed at the prompt. When you have found what you want, press Enter to execute the command, or the left or right arrow key to edit it.

Finally there is command line history. Type **history** (**his** then Tab will finish the typing for you). You will be presented with a numbered list of all of your previous commands. Type an exclamation point (!) followed by the number of the command that you want to repeat, and then press the Enter key to immediately execute that command.

When you find that you often use a command in the same way, with the same options then you can create a shortcut, known as an alias, to save a little typing.

To generate a long listing with human-readable file sizes and sorted by file size the command would be **ls -lhS**.

Although this is not difficult, can you always remember that the S must be uppercase? The command **alias lh='ls -lhS'** makes this available by

simply typing the new command **lh**. Unfortunately, this has to be done every time that you open a terminal and start the bash shell. To make this happen automatically, you can add it to a hidden file in your home directory, named **.bashrc**.

There are many text editors available for the terminal in Linux, and PCLinuxOS has an excellent and easy one to use, named **nano**.

Type **nano ~/.bashrc** (there's that squiggle again) and your own copy of this file will open in the editor. You will most likely already have some aliases in here provided by the developers of PCLinuxOS, and a good place to put your new one would be with those, but it doesn't really matter. Type in your new alias on a blank line and then press **Ctrl + X**. Answer yes when prompted to save. Restart the terminal and your new command is available every time.

To see what other aliases the good folks over at PCLinuxOS development have added for you, type the command **alias**.



Screenshot Showcase

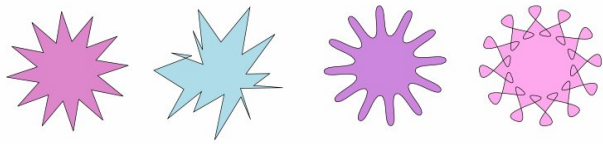


Posted by Texstar, April 21, 2013, running e17.

Inkscape Tutorial: Fun With Shapes

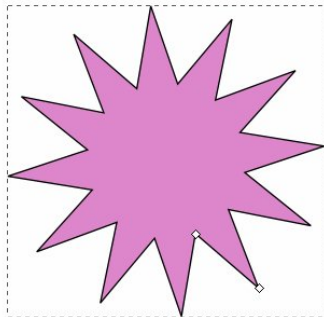
by Meemaw

Messing around with some of the tools in Inkscape can be fun! I started with a simple 12 point star (below, left) and made so many other designs.



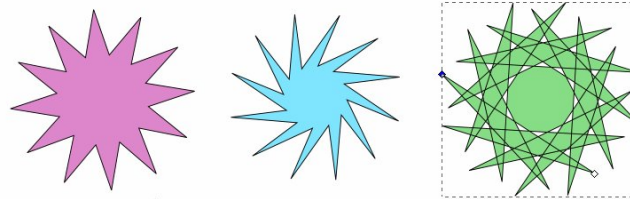
All of these started out with the Spoke Ratio 0.5, Rounded 0.0, Randomized 0.0 (left end), then I changed the settings for different effects. The second from the left is Spoke Ratio 0.5, Rounded 0.0 Randomized 0.25. The third one is Spoke Ratio 0.5, Rounded 0.25, Randomized 0.0. You can put negative readings on Rounded or Randomize and it will result in other effects. The fourth one has Rounded setting of -0.75.

Another thing you can do is grab one of the handles you see when you select your shape and click on the Nodes tool.

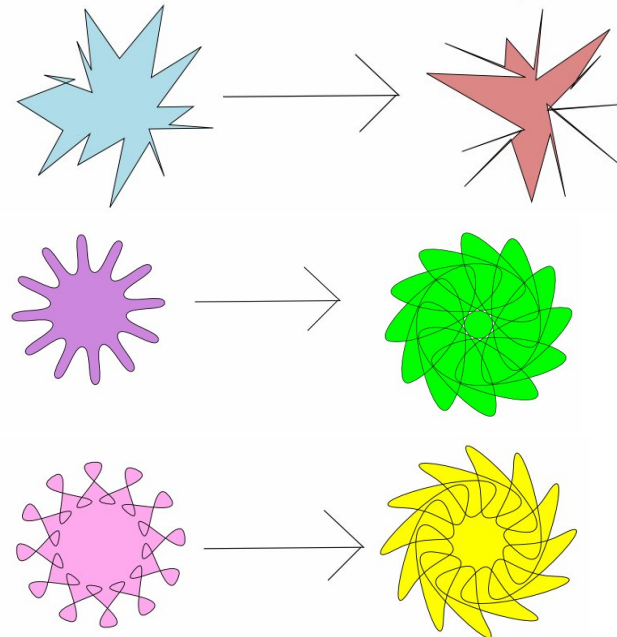


You can see in the example that there is a "handle" or "Node" on one of the outer points and another on one of the inner points. Grabbing the outer handle and moving it can change the direction of your spokes (center). The

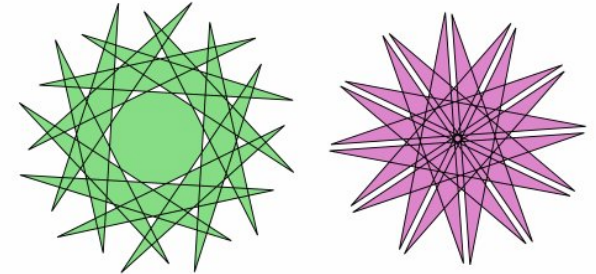
one at right is the result of grabbing the inner handle and moving it across the star until it crosses the other sides.



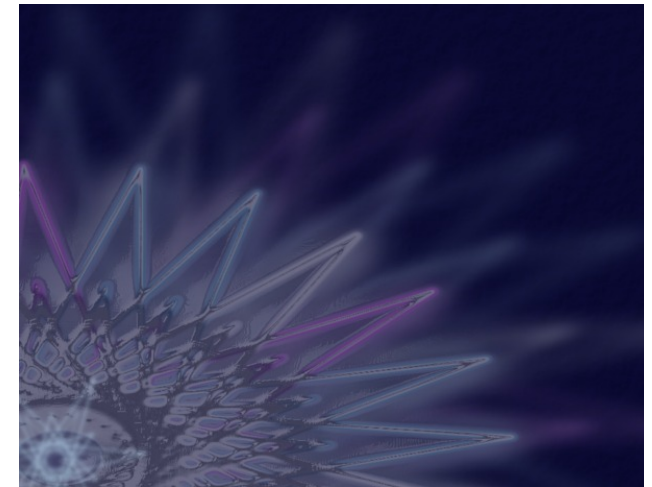
Here are more examples of moving handles/nodes:



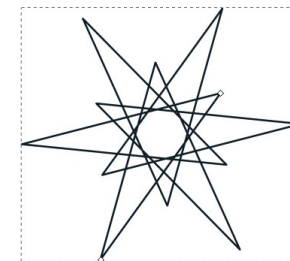
Believe it or not, both of these were made with the same 12 point star, just by moving the inner node (top right):



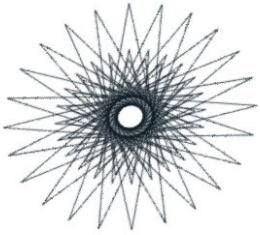
Where am I going with this?



Using just a few of the shapes I showed here, I made a wallpaper. On this one I started with a six-point star and then moved the nodes until I had this:



Duplicating and rotating can give you the starburst (next page, top).



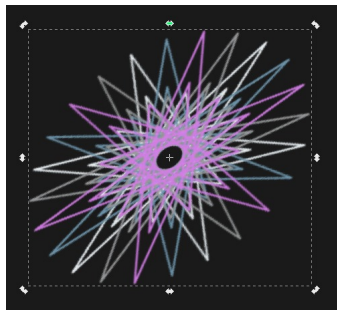
corner arrows will rotate the object around the plus sign, which is in the center, but can be moved.)

While I was duplicating, I also changed the colors to pink, blue & grey. I also used the rotate handles to change the starburst so it wasn't exactly round, but more elliptical. (The side arrows will skew the object, while the

outside the page border, you want to export the page instead of the drawing (choose the appropriate button from the top of the export window).

You can experiment all you want. Some projects are just for fun and even though you use very few items or even commands, you can still come up with

something you like. If you design something you think you will use a lot, duplicate it and set it to the side of your page. Also, you can do something interesting with any shape. One of the first Inkscape projects I did was to transcribe [Sproggy's Glass Panel Tutorial](#) from the video he made. In it, he used mostly rectangles to get his effect.



Then, I put a dark blue background behind everything so I could see the lighter colors. Actually, I put the dark blue background on it's own layer, so I wouldn't move the background while selecting something else.

After that, I duplicated the starburst at least twice and made them all huge, then moved them so the center of the starburst was in the bottom left corner. On those duplicates I set the blur up (like 4 or 5) and the opacity down (like 50%) so they are not as visible. I added a circle in the center of the top starburst to make it more the same color instead of having a dark center. To do that, I changed the blur and opacity of the circle until it was similar to the starburst, then grouped them.

On that starburst, I made sure the blur was less (like 1) and used **Filter > Bevels > Stained Glass** to make it shiny. Then I added one more of the basic stars on top, then put another dark blue rectangle at opacity 60% on top so everything looks a little darker. You should save each time you are happy with your work, and if you are closing your drawing. When you get it just right, save it again, then use the **File > Export Bitmap**. Since part of your drawing is

Screenshot Showcase



Posted by longtom, April 7, 2013, running KDE.

PCLinuxOS Recipe Corner



Macaroni and Cheese Ham Pie

Ingredients:

3 cups shredded Cheddar cheese (12 oz)
1 cup cubed cooked ham
1 cup uncooked elbow macaroni (3 1/2 oz)
2 1/4 cups milk
2 eggs
1/2 cup Original Bisquick® mix
1/4 teaspoon salt
Chopped fresh parsley, if desired

Create you own Bisquick mix here <http://recipes.dm-enterprises.net/?p=recipe&recipe=289>

Cooking Instructions:

1. Heat oven to 400°F. Spray 10-inch or 9 1/2-inch glass deep-dish pie plate with cooking spray.
2. In large bowl, mix 2 cups of the cheese, the ham and uncooked macaroni. Spread in pie plate.
3. In blender, place milk and eggs. Cover; blend on medium speed until smooth. Add Bisquick mix and salt; blend until smooth. Pour over mixture in pie plate.
4. Bake 35 to 40 minutes or until knife inserted in center comes out clean. Sprinkle with remaining cheese. Bake 1 to 2 minutes longer or until cheese is melted. Sprinkle with parsley. Let stand 10 minutes before serving.



**Looking for an old article?
Can't find what you want? Try the**

**PCLinuxOS Magazine's
searchable index!**

The **PCLinuxOS** magazine



**The
PCLinuxOS
Magazine**

**Created with
Scribus 1.4.1**

Want To Help?

Would you like to help with the PCLinuxOS Magazine? Opportunities abound. So get involved!

You can write articles, help edit articles, serve as a "technical advisor" to insure articles are correct, create artwork, or help with the magazine's layout.

Join us on our [Google Group mailing list](#).

PCLinuxOS Puzzled Partitions

	3	2		8			1	
		1		7		2	9	
			1				7	3
	9	3				1		
			4		9			
		5				9	6	
9	2				4			
	5	4		9		3		
	8			1		5	4	

Sudoku Rules: There is only one valid solution to each Sudoku puzzle. The only way the puzzle can be considered solved correctly is when all 81 boxes contain numbers and the other Sudoku rules have been followed.

When you start a game of Sudoku, some blocks will be pre-filled for you. You cannot change these numbers in the course of the game.

Each column must contain all of the numbers 1 through 9 and no two numbers in the same column of a Sudoku puzzle can be the same.

Each row must contain all of the numbers 1 through 9 and no two numbers in the same row of a Sudoku puzzle can be the same.

Each block must contain all of the numbers 1 through 9 and no two numbers in the same block of a Sudoku puzzle can be the same.



TUX SCRAPPLER RULES:

Follow the rules of Scrabble®. You can view them [here](#). You have seven (7) letter tiles with which to make as long of a word as you possibly can. Words are based on the English language. Non-English language words are NOT allowed.

Red letters are scored double points. Green letters are scored triple points.

Add up the score of all the letters that you used. Unused letters are not scored. For red or green letters, apply the multiplier when tallying up your score. Next, apply any additional scoring multipliers, such as double or triple word score.

An additional 50 points is added for using all seven (7) of your tiles in a set to make your word. You will not necessarily be able to use all seven (7) of the letters in your set to form a "legal" word.

In case you are having difficulty seeing the point value on the letter tiles, here is list of how they are scored:

- 0 points: 2 blank tiles
- 1 point: E, A, I, O, N, R, T, L, S, U
- 2 points: D, G
- 3 points: B, C, M, P
- 4 points: F, H, V, W, Y
- 5 points: K
- 8 points: J, X
- 10 points: Q, Z

Optionally, a time limit of 60 minutes should apply to the game, averaging to 12 minutes per letter tile set.

Have fun! It's only a game!

Tux Scrapppler

K₅

M₃

N₁

Y₄

E₁

G₂

A₁

—

—

—

—

—

—

—

Double Word

S₁

T₁

U₁

O₁

R₁

D₂

C₃

—

—

—

—

—

—

—

P₃

W₄

E₁

B₃

N₁

A₁

O₁

—

—

—

—

—

—

—

Triple Word

M₃

U₁

E₁

O₁

C₃

P₃

T₁

—

—

—

—

—

—

—

A₁

I₁

E₁

E₁

R₁

Z₁₀

L₁

—

—

—

—

—

—

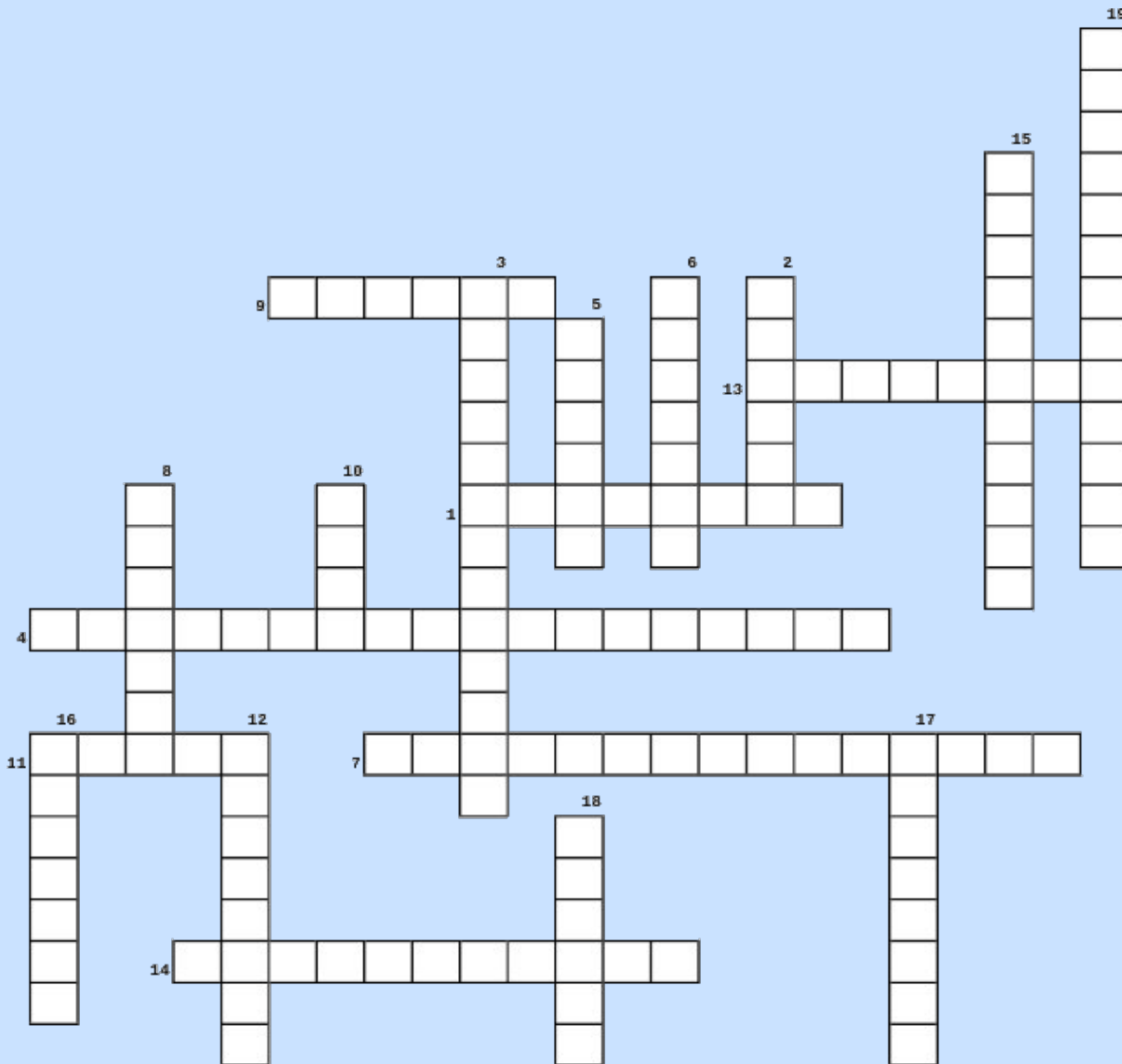
—

240 points possible. Average mark is 168 points.

Download Puzzle Solutions Here!



PCLinuxOS Crossword Puzzle: May 2013



[Download Puzzle Solutions Here!](#)

1. Open this to directly access the Linux command line
2. Popular maker of graphic chips
3. Linux Founder
4. KDE is one of these
5. 1 GB is the minimum recommended level of this for KDE
6. The main file manager application under KDE
7. PCLinuxOS slogan
8. PCLinuxOS Founder
9. _____ is your friend
10. The powerful raster graphics editor in PCLinuxOS
11. The primary place to go for PCLinuxOS support
12. The PCLinuxOS _____: monthly periodical
13. The powerful vector graphics editor in PCLinuxOS
14. Primary free office suite for PCLinuxOS
15. Current repository maintainer & lead packager
16. A popular web browser
17. Popular Chinese solitaire game played with special tiles
18. Main application to use to watch TV on your tuner card equipped computer
19. PCLinuxOS _____: where you go to configure your computer

PCLinuxOS Forum Member Word Search

P V L T P C U K P C P I E S S I B E C D E U H B X Z G U P F
 X P J R J B M I T T X X Y M V K N B P K X X S W U L H Q K Q
 X O L D G U Y D J C S O A M T L V I T G T G C L S U T E T E
 E K C E F Y O M C S T A D I B E S S G H A U M L G R I Z H M
 M V H B P E F B O J Q W W T N E O X S P L X K O D T R V E F
 K T P H F F X G N I R D M A L G U V W R P I N L Z T M I C B
 N E A L M A N B E A R S E X J X H E R J N S O T N H J D H R
 P C L I N U X O S W I U L Y C X A R U E W U N E J A U X I S
 W N T P A O N V N D L B P O P W Q O W B U E X Y F Q T J E W
 C I D N N I O L D J I M M J N A M A E S G C O A R U O D F U
 I A Y O D F M I Y K Y K I B P I D I T O T P L T N R N X F D
 F F W K R V C B M N C F S Z K M U O V I E Z F E I L E J L O
 K R N Y Z X J I C I P A Y T R C E P F C R O W E V G M T Y U
 P Z Q T E B Y V R Q L I L C A B E Y L D R I J F P G J W H L
 M V W X J A O B I V V G L O D Z B L R N Y R K C K E B G E F
 R X V A L I D C H J K X A S P O M R B C H A I G O P W Z H S
 V W G V S L M F Z X W Z C A J D U L K P S Z V O P N K R H R
 A J U Y A C Y W C B W F I A U D L H L Q C H S R J U I B V T
 V G B B X U D U I O C E D B G Z S O U W G E C F U W N P P O
 Y R E U N O S O J L F O A E P L E R D N U O C S N E K L G V
 X U P N I D Y F V F K C R D P T E A D L N A O H G J S U E F
 E E F Z T O K A J R E M E M S M W G B N G C U G Q B C L O I
 N K F S G S O U N D M B T R U D I A U M W A O C A A H D R X
 A E J T S B M T M K T W Q D V A B H G H O S T B U N N Y G P
 F E G U A J E I N E E L O M H B Y F C C Q N J Y K E D C E A
 L A P N X X C Z T A E E C J U J M N I E V L U F C T C Y T X
 U L A O S X P X Y H C M Z B F E Y G G L O Q K G T F A Z O W
 X B G T Y N S J C R S U A L Z R Y C N P E J X R C F A P O Q
 W P A G M F E B B C T U O W W M D R U Q R T Y G T K V F N U
 P R Q G F R Q Q K D I F F Y N G F T S F H Y E P K N B M P C

RADICALLYSIMPLE
 PCLINUXOS
 TEXSTAR
 GSEAMAN
 NEALMANBEAR
 PINOC
 GHOSTBUNNY
 MEEMAW
 MENOTU
 JIMWILK
 RUDGE
 BUBBABLUES
 JOECHIMP
 NOK
 SCOUNDREL
 OLDPOLACK
 GANDY
 TERRYH
 TERRYN
 YOUCANTOO
 PAGES
 AGMG
 BALDBRICK
 CROW
 ANDRZEJL
 DEBAAS
 AGENTSMITH
 MSMEME
 GEORGETOON
 XENAFLUX
 GLAMDRING
 ELCUERVO
 OLDJIM
 THECHIEF
 OLDGUY
 SLONIUPL
 YANKEE

Download Puzzle Solutions Here!

Famous Forum Foibler: Neal



How Neal sees himself



How Archie sees Neal



How MeeMaw sees Neal



**Texstar sees Neal bringing
the packin' and the sackin'**



**Neal's idea of
the packin' and the sackin'**



How the Forum sees Neal

Xfce Power User Tips, Tricks & Tweaks: Multimedia Files

by Paul Arnote (parnote)

If you want to clear a room or kill a party, start talking about working with multimedia files. Many folks will head for the exits. Multimedia files are often viewed as being temperamental and hard to work with. However, it doesn't have to be this way. Sure, they are big, and they eat up a lot of CPU cycles when you work with them. There's nothing we can do about that, but we can streamline the process.

In the previous **Xfce Power User Tips, Tricks & Tweaks** articles, we talked about streamlining the way Xfce users could work with graphic files (March 2013) and an easier, more reliable way to work with archive files (April 2013). Guess what. If you're an Xfce user, we can also streamline the way(s) you work with multimedia files, doing tasks such as previewing multimedia files, playing multimedia files, and even converting multimedia files.

If you haven't already done so, you will need to make sure that both Zenity and Yad are installed from the PCLinuxOS repository. Most of the items that follow use one or both of the dialog creation libraries. Some additional programs you may want to install include ffmpeg, pacpl, mplayer, sox, aplay, mediainfo and mencoder. All of the additional applications are command line tools for working with multimedia files.

Playing a sound when Xfce starts

Most of us Linux users are Windows refugees. Exceptionally few of us cut our computer baby teeth on Linux. Most of us came to Linux after suffering through the inadequacies of one Windows version or another. One thing that some Linux users miss is

some kind of sound playing when our desktop loads. While some may view such a thing as trivial and even sophomoric, the playing of the sound indicates, for many, that the desktop is loaded and ready for business.

Under Xfce, there's no clear-cut way to play a sound when your desktop loads. That feature simply does not exist. Fortunately, there's a relatively simple way to restore the startup sound that signifies that your desktop has loaded and is ready for use.

The first thing to do is to install either **aplay** or **sox** command line utilities from Synaptic. I usually have both installed, allowing me to switch back and forth between the two. Of the two, aplay is a little more restrictive, since it will only work with *.wav, *.voc, *.raw or *.au files. Sox, however, will work with just about any sound file you might have on your system, including *.flac, *.mp3 and *.ogg files. There may be other command line programs that also allow you to control the playback of audio files, but to be perfectly honest, I've found these two to perfectly fulfill my needs and I haven't looked any further.

Now that you have either aplay or sox installed, it's time to set up your startup sound. Open the **Xfce Settings Manager**, then find and launch the **Session and Startup** manager. Select the **Application Autostart** tab, then select the **Add** button near the bottom left of the window.



Fill in the "Name" and "Description" fields in the dialog box. For the "Command" field, I recommend first trying out the "bare" command at a command line prompt in a terminal session, to insure that you have everything correct. Here, I use the **play** command (from sox) to play the sound file specified at the end of the command, at a volume of 30% (-v 0.3), and to use "magic" file-type detection (the information stored in the header of the sound file) to determine how to play the sound file. Using the "magic" file-type detection makes the play command much more universal, rather than specifying a separate command and Thunar Custom Action for each individual sound file type (e.g., -t mp3 would tell the play command to read the file as a *.mp3 file). Thus, my entire command is as follows (all on one line):

```
play -v 0.3 --magic  
~/Sounds/Borealis/Startup1_3.wav
```

Be sure to provide a **full path** to the sound file you are wanting to use as your startup sound. Also, be careful with the volume switch. If you want to play your sound file back at 50 percent volume, enter 0.50. If you enter 50, the play command will instead play your sound file back at a volume that's 50x louder than normal. If you want to play your sound file back at full volume, use 1.0 as your volume level.

Once you are sure that you have the command correct and as you like it, copy the command from your terminal session and paste it into the Command field of the dialog box. Select the OK button. Now, the next time you log out and back into your Xfce desktop – or the next time you start your computer – your startup sound will be played.

Advertisement



Playing a sound when shutting down

Just as users have grown accustomed to a sound playing to signify that their computer is nearly ready for them to work with, users have also grown accustomed to hearing a sound play when they exit, logout or shutdown their computer.

Unlike the startup sound, which is relatively easy to configure and use, setting up a logout sound is a bit more challenging. Also, despite Xfce having a setting to "Enable Event Sounds" (Xfce Settings Manager > Appearance > Settings), I've not yet been able to find a place within the Xfce settings to otherwise enable or define "event" sounds. The lack of such an ability makes defining certain events more difficult, but not impossible.

I've tried – unsuccessfully – executing the necessary commands as a "one-liner" panel launcher. However, I did find that I could execute the commands from a bash script, and it works perfectly!

First, copy/enter the following text into your favorite text editor:

```
#!/bin/bash

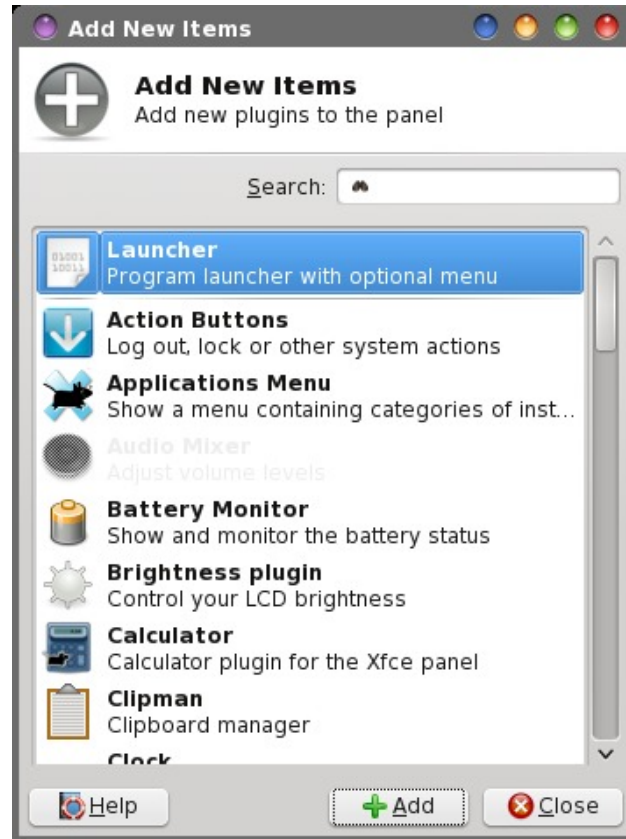
# sound-shutdown.sh
# Paul Arnote
# Play our sound when we logout
# The '&' is important
# Without the '&', the logout command isn't
# executed until the sound has finished
# playing, causing a delay.

play -v 0.40 ~/Sounds/Borealis/Exit1_2.wav &

# Bring up the logout choice window
xfce4-session-logout
```

I named the bash file sound-shutdown.sh. Be sure to either place bash file somewhere in your path, or (if you don't place it somewhere in your path) provide the full path and filename to your bash file. Be sure

to make the file executable (right click on the file, go to the permissions tab, and check the "Allow this file to run as a program" check box).

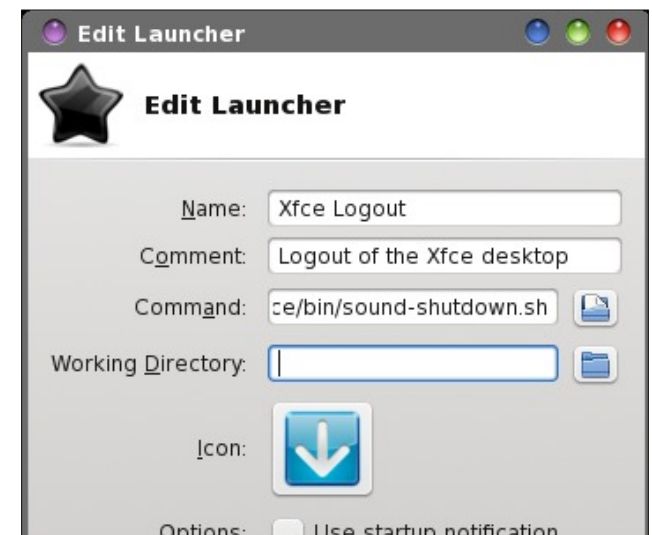


Next, remove the Xfce Panel Plugin named "Action Buttons." You won't need them any longer, with this alternative in place. In its place, create a new launcher on the panel. Select the green "+" button to select the application to run with your new launcher. Pick anything. What we are looking for isn't in the list, so we'll be changing it anyway. In the image below, I've selected the "About Xfce" application.

Now highlight the launcher application, then select the bottom button on the right side (the one with the pencil on it) to edit our launcher. I've named my new launcher "Xfce Logout," which is what you enter on



the first line of the launcher dialog box. On the second line, type a description for your launcher. This is the text that is displayed when you hover your mouse over your new launcher. I simply entered "Logout of the Xfce desktop."



On the third line, enter the name of your bash file that you saved (in my case, it's sound-shutdown.sh). Remember, if the bash file is located somewhere within your path, only the filename is necessary. Otherwise, you will need to specify the full path and filename to your bash file.

Next, select an icon for your new "Log Out" launcher. Because of what we're asking it to do, I chose a "logout" icon to represent my new launcher.



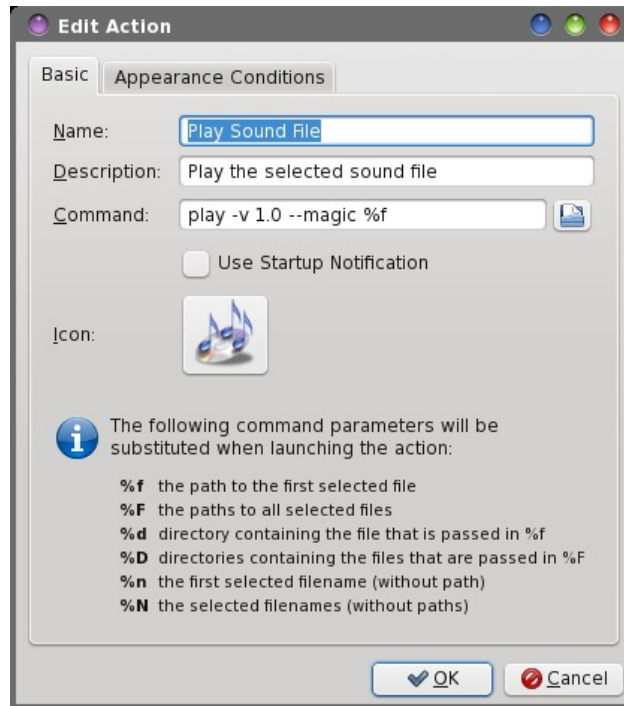
Now, when you select your new launcher, your chosen sound file will play and the logout chooser dialog box will be shown, giving you a choice of logging out, restarting or shutting down your computer. As an added bonus, instead of having three, four or more buttons on your panel (one for each option you choose with the Action Buttons), you now have just one button, saving precious space on your panel.

Should you ever wish to change the sound that is played when you log out of your Xfce desktop, simply change the sound file that the play command points to in the sound-shutdown.sh bash file.

Play a sound file

If you have a collection of sound effect files hanging around on your system, it can sometimes get tiring to open each one in a sound player program just find the one you are looking for. Or, sometimes you have a collection of your favorite music files that you want to listen to as you work. We can handle either situation very easily, with this Thunar Custom Action.

From Thunar, select the Edit > Configure Custom Actions... menu item. From the dialog box that appears, select the first button (Add a new custom action) in the column of buttons on the right side of the window.



On the first line of the new dialog box that opens, type "Play Sound File" as the name of our new Thunar Custom Action. On the second line, enter a description. I entered "Play the selected sound file." On the third line, enter the following:

play -v 1.0 --magic %f

This will invoke the "play" command (installed as a part of the sox package) at a volume (-v) of 1.0. In play and sox speak, this is 100% ... if you want it played at half volume (50%), enter 0.50. Similarly, if you want the sound to be played at only 30% volume, enter 0.30 instead of 1.0 for the volume. The --magic flag tells the play command to use the information in the file header to figure out what codec to use for the playback of the file. The %f represents the single sound file you selected. Finally, select an icon to display next to your new custom action (click on the button that says "No Icon" on it).

Next, on the "Appearance Conditions" tab of the dialog box, select the checkbox next to "Audio files" and make sure no other check boxes are marked. For file pattern, simply leave it as * (the default).

Select the OK button at the bottom of the dialog box. You will be returned to the first dialog box. Select the "Close" button.

Now, using Thunar, go to a directory on your computer where you have some sound files saved, right click on it, and select "Play Sound File" from the right-click context menu. This should work perfectly on every sound file that has a proper header (yes, there are headerless sound files out there) that your system recognizes as a sound/audio file (dependent on what audio codecs you have installed). Your system should already be set up (minimally) to know how to handle MP3, OGG and WAV files. As an added bonus, MP3 playlists (*.m3u files) and general playlists (*.pls) files will also be played with this Thunar Custom Action.

So what happens if you click on a sound file without a proper header? The command simply will not play it. But you can easily "fix" the sound file by giving it a proper header. Open the sound file in Audacity, and then export (File > Export) the sound, re-saving it as the exact same file name. Audacity will apply a proper header to the sound file when you re-save it.

Granted, when played this way, you have no extra control over the playback of the file (or files, in the event that you selected a playlist file). You can't adjust the volume after you start playing (other than adjusting your speaker volume). You can't pause the file playback. You can't jump ahead or back in a playlist. You can't fast forward or rewind through a song. But, there is a way to stop a song (or playlist) from continued playback. In a terminal, simply type **pskill play** at a command prompt. You should have instant silence, just in case your playlist of Led Zeppelin or Deep Purple was rocking the house when the telephone rang. Of course, there's nothing preventing you from also just turning down the volume on your speakers -- or muting your sound card -- when you answer the telephone.

If you want to have a little more fun with this, try adding "reverse" to your play command. Your command (in the second Thunar Custom Action dialog box) should now look like this:

play -v 1.0 --magic %f reverse

Now, whatever sound file you select to play, it will be played **backwards**. Although probably not very useful, it is amusing to play with (especially if you are wanting to revisit the "Paul is dead" Beatles conspiracy).

Similarly, if you want to play with some of the other special effects available with the play command, keep in mind that the gain in, gain out, decay, delay and speed settings are all based on the premise that 100% = 1.0, 50% = 0.50, 30% = 0.30, etc (just like the volume setting). You can see an entire listing of all of the effects by typing **play --help** at a command prompt. To see the help for each individual effect, type **play --help-effect "name-of-effect"** (without the quotes). Thus, the echo command can be employed like this (all on one line):

**play -v 1.0 --magic %f echo
1.0 1.0 1.0 1.0 0.75 0.75**

Another effect you may want to try is the "reverb" effect. You can easily kill/waste a couple of hours just playing with the effects.

Loosen up and have a little fun playing with the play command. After all, it just reads your sound files, so any errors you make in the execution of this command aren't going to permanently damage your sound files. If you are unsure about how to format a command, try it from a command prompt first, replacing the %f used by Thunar with a full path and filename to the sound file you want to use.

Preview a sound or video file

If you're anything like me, it's not uncommon to draw a blank regarding what file is what when you're browsing through your collection of music and sound files stored on your computer. Plus, if you've ever downloaded videos from video sites like YouTube,

the filenames are sometimes quite cryptic and provide literally NO clue to the video contents.

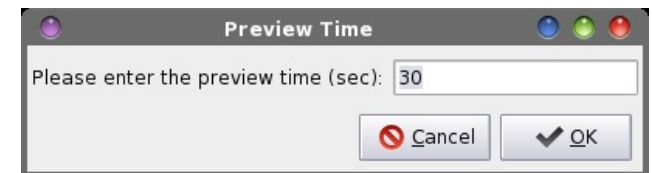
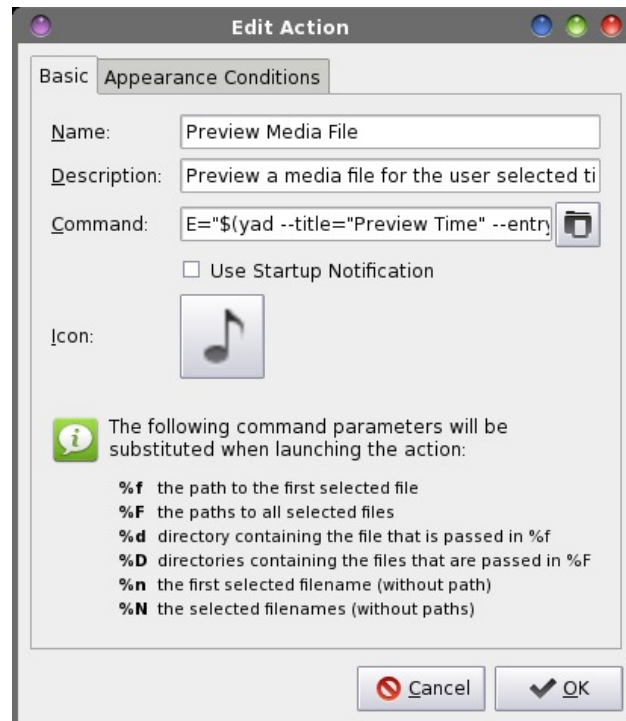
Thanks to this Thunar Custom Action, you can easily preview just about any sound or video file that your PCLinuxOS installation is configured to handle. Because you can only watch one video at a time or listen to one sound file at a time, this Thunar Custom Action is setup to work on only one file at a time, too.

Create a new Thunar Custom Action (Thunar > Edit > Configure custom actions...). On the first and second lines, enter the name and description for your new custom action. I entered "Preview Media File" and "Preview a media file for the user selected title" for these fields.

Now, enter the following on the third line as your command:

E="\$({yad --title="Preview Time" --entry --entry-label="Please enter the preview time (sec):" --entry-text="30"}"); mplayer %n -vo x11 -really-quiet -endpos \$E

Pick an icon for your new Thunar Custom Action. Under the "Appearance Conditions" tab, leave the "File Pattern" entry set to the default value of "*" and place a checkmark in front of both Audio files and Video files.



This command uses yad to display a dialog box that allows you to specify how long you want the preview to last. The default time is 30 seconds, but you can enter whatever value you want. Thus, if you want the preview to last for three minutes, you would enter 180 seconds. Then, mplayer is called to play the file (%n), and the video output is directed to output any video using the x11 video driver (optional ... ignored

when playing audio files ... done so that you can capture the video in screenshots, if you like.). The error and warning messages from mplayer are set to bare minimum (-really-quiet) and the end position (-endpos \$E) is set to end playback after the number of seconds input in the yad dialog box. Mplayer usually does an outstanding job of picking the proper codec to use for playback based on the file extension of the multimedia file being played back. Thus, there is no real need to tell mplayer how to playback a file.



Above is the playback of a country music video (Jalepenos by The Bellamy Brothers ... look it up) I downloaded from YouTube. As you can see, this Thunar Custom Action does an excellent job of playing back your video files. Since it's impossible for me to "show" audio playback, you'll have to take my word that audio playback is equally as impressive – or even better yet, you can try it for yourself.

Simply right click on a multimedia file, then select "Preview Media File" from Thunar's right-click context menu. Enter the length of time you want the preview to play for, then sit back and enjoy!

Converting sound files

Remembering the commands to convert from one audio file format to another is often difficult. Thanks to Thunar Custom Actions, we can define a two-click shortcut to convert from WAV to MP3, MP3 to WAV, OGG to WAV and WAV to OGG.

Actually, we have four different custom actions -- one for each conversion. Each custom action is dependent on having ffmpeg installed from the repos. Ffmpeg is the command line utility that converts between various multimedia formats.

To convert from WAV to MP3: `ffmpeg -i %f -acodec libmp3lame -ab 128k `basename %f .wav`.mp3 | yad --progress --title="File Conversion" --progress-text="Please wait ... File(s) being processed." --percentage=40 --auto-close --auto-kill`

First line: Convert WAV to MP3

Second line: Convert WAV audio file to MP3 file

Appearance Conditions: *.wav, check mark Audio files

To convert from MP3 to WAV: `ffmpeg -i %f `basename %f .mp3`.wav | yad --progress --title="File Conversion" --progress-text="Please wait ... File(s) being processed." --percentage=40 --auto-close --auto-kill`

First line: Convert MP3 to WAV

Second line: Convert MP3 audio files to WAV file

Appearance Conditions: *.mp3, checkmark Audio files

To convert from WAV to OGG: `ffmpeg -i %f -acodec libvorbis -aq 60 `basename %f .wav`.ogg | yad --progress --title="File Conversion" --progress-text="Please wait ... File(s) being processed." --percentage=40 --auto-close --auto-kill`

First line: Convert WAV to OGG

Second line: Convert WAV audio file to MP3 file

Appearance Conditions: *.wav, check mark Audio files

To convert from OGG to WAV: `ffmpeg -i %f `basename %f .ogg`.wav | yad --progress --title="File Conversion" --progress-text="Please wait ... File(s) being processed." --percentage=40 --auto-close --auto-kill`

First line: Convert OGG to WAV

Second line: Convert OGG audio file to WAV file

Appearance Conditions: *.ogg, check mark Audio files

Besides ffmpeg, you will need to insure that you have the libmp3lame and libvorbis sound libraries installed. Chances are high that they are already installed on your computer.

The ``basename %f .ext`.ext` line strips the old extension off before attaching the new extension. Use extra care -- those are backticks, NOT single apostrophes. (Hint: the backtick is on the key (on the U.S. keyboard) next to the number "1" key, in the row of number keys at the top of the keyboard).



The commands also display a yad progress bar dialog box, to let you know that the file conversion is being carried out. Depending on the speed and number of cores of your computer's processor, the file conversions can take a little time (up to a minute or so). The progress bar dialog box lets you know that "something" is still being done. Because we're calling the progress bar dialog box from the Thunar Custom Action command line itself, the percentage is set to a static 40 percent. I've had very limited success at getting the progress bar to pulsate, unless it is called from a bash script. You can set the static percentage of the progress bar dialog box to whatever value you choose. I chose 40 percent because it gives the appearance of actually being active – at least to my eyes.

Also, because we have not created a loop to process multiple files (easy enough to do, if you create and call a bash script), this custom action can only be used on single files, converting one at a time.

The command can easily be expanded to convert other audio **and** video files. You can find good examples of the uses of ffmpeg [here](#) and [here](#), providing you additional Thunar Custom Actions. Just use the examples I gave here as templates.

Converting Sound Files, Part 2

A quick search in Synaptic nets us another real diamond for converting sound files from one format to another. Below is the description:

Perl Audio Converter is a tool for converting multiple audio types from one format to another.

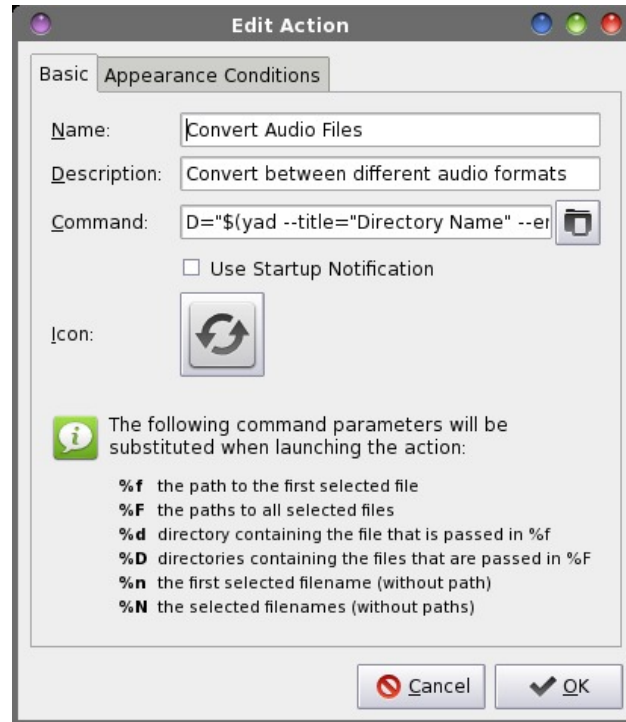
Perl Audio Converter is a tool for converting multiple audio types from one format to another.

It supports: AAC, AC3, AIFF, APE, AU, AVR, BONK, CDR, FLA, FLAC, LA, LPAC, M4A, MP2, MP3, MP4, MPC, MPB, OFR, OFS, OGG, PAC, RA, RAM, RAW, SHN, SMP, SND, SPX, TTA, VOC, WAV, WMA, and WV. It can also convert audio from the following video extensions: RM, RV, ASF, DivX, MPG, MKV, MPEG, AVI, MOV, OGM, QT, VCD, SVCD, M4V, NSV, NUV, PSP, SMK, VOB, FLV, and WMV.

A CD ripping function with CDDb support, batch conversion, tag preservation for most supported formats, independent tag reading/writing, and extensions for Konqueror, Dolphin and Amarok are also provided.

When you read that description, the last part of the last paragraph might make you think that pacpl is a KDE-centric application. Nope. It's just that the **extensions** included in the package are centered around KDE applications. Pacpl is a command line utility. You can run it from a command line prompt in

a terminal session. In our case, we're going to adapt it for use as a Thunar Custom Action.



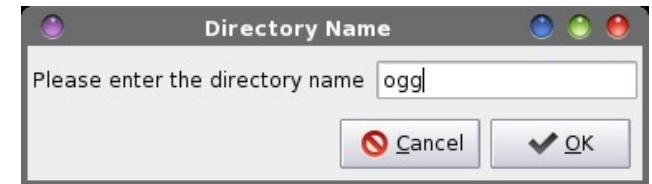
Create a new Thunar Custom Action (Thunar > Edit > Configure custom action...). Fill in the name and description fields. I used "Convert Audio Files" for the name, and "Convert between different audio formats" for the description.

On the third line, we need to insert the command to be executed for our new Thunar Custom Action. This is where the real magic occurs. Here is the command (below):

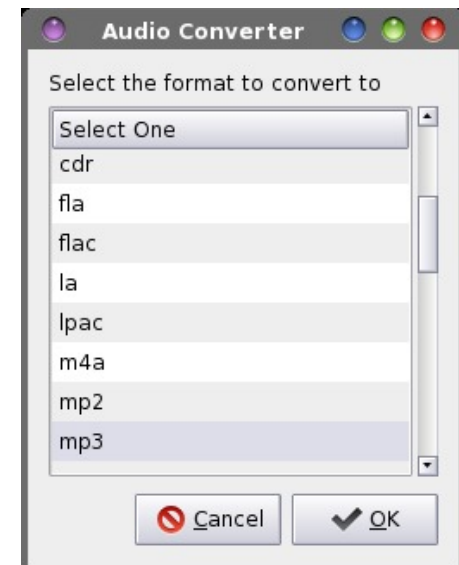
```
D="$(yad --title="Directory Name" --entry
--entry-label="Please enter the directory
name"); if [ ! -d $D ]; then mkdir $D; fi;
F="$(zenity --list --column="Select One" --
title="Audio Converter" --height=300 --
width=250 --text="Select the format to
convert to" aac ac3 aiff ape au avr bonk
cdr fla flac la lpac m4a mp2 mp3 mp4 mpc
```

```
mpp ofr ofs ogg pac ra ram raw shn smp snd
spx tta voc wav wma wv)"; pacpl --outdir $D
--normalize -t $F %N | yad --progress --
title="File Conversion" --progress-
text="Please wait ... File(s) being
processed." --pulsate --auto-close --auto-
kill
```

Yes, it's a long one. While I am unable to find the reference again, I seem to remember that the length of the command for a Thunar Custom Action cannot exceed 4 KB. Even with a command this long, we're still a LONG ways away from that length limit.

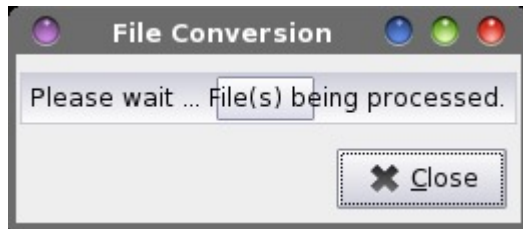


Here's a rundown of what's going on in the command. First, it gives you a chance to enter a specific directory for your converted files, via a yad dialog box. You type in the directory name, and if the directory doesn't already exist, it is created. If you don't want to save your converted files to a special directory, simply enter ./ as the directory to use.



Second, a zenity dialog box is displayed, allowing you to select which audio format you want to use to convert your audio files into. All of the audio formats supported by pacpl are included in the list dialog box.

Selecting OK in this dialog box causes pacpl to run. Pacpl will output the converted file into the specified directory (\$D), normalize the output file, convert the audio file(s) to the specified audio format (\$F), and work on multiple files (%N). Thus, you can select several files at once – even files of different audio formats – and convert them ALL to one common audio format.



While the conversion is taking place, a pulsating yad progress bar dialog box is displayed. This way, you know that the conversion process is still working. Depending on the speed and number of cores of your computer's processor, this conversion process can take some time. Remember that audio files tend to be rather large, and it might take some time to work through multiple files.

Now, when you select multiple files, then select Convert Audio Files from Thunar's right-click context menu, your selected files will be converted into one common audio file format. As an added bonus, pacpl can extract the audio from many different video formats.

Convert video files

Another task you might want to perform on your collection of video files (from whatever source) is to convert them all into one common format. Of course,

as you add to your collection, you may also want to convert the new additions to your video library to the same common format. For the sake of clarity, by format, I mean videos that all use the same video and audio codecs, and the same video and audio sample bitrates. This Thunar Custom Action will enable you to do just that.

To get started, copy or type in the following bash script into your favorite text editor.

```
#!/bin/bash
```

```
ConvertTo=$(zenity --list --column="Select
One" --title="Video Converter" --height=300
--width=250 --text="Select the format to
convert to" avi 3gp flv mov mp4 asf wmv mpg
mov)
if [ $? == 1 ]; then
    exit
fi
```

```
Video=$(zenity --entry --entry-text="1000"
--title="Video Converter" --text="Enter
video bitrate (Kb/sec)")
if [ $? == 1 ]; then
    exit
fi
```

```
Audio=$(zenity --entry --entry-text="128"
--title="Video Converter" --text="Enter
audio bitrate (Kb/sec)")
if [ $? == 1 ]; then
    exit
fi
```

```
tail -f ~/Scripts/convert-video.sh | zenity
--progress --pulsate --auto-close --auto-
kill --title="Converting" --
text="Converting video to $ConvertTo ...
Please wait." &
```

```
name=$(echo $1 | cut -f1 -d.)
```

```
ffmpeg -i "$1" -ab "$Audio"K -vb "$Video"K
"$name.$ConvertTo"
killall -KILL tail
```

```
zenity --info --title="Video Converter" --
text="Conversion to $ConvertTo
finished.\n\nPlease check file to
insure\nno errors on conversion."
```

```
exit 0
```

The script allows you to choose a video format to convert your selected video file(s) into, and then input the video bitrate, then input the audio bitrate for the converted video. The script then uses the **tail** command to track if the script (itself) is still running and displays a pulsating zenity progress bar dialog box. The "old" file extension is stripped off of the filename, and **ffmpeg** is run to convert the video, assigning it a new file extension. It then "kills" the execution of the **tail** command, and displays a zenity dialog box to let you know that the file conversion is complete.



Create your new Thunar Custom Action. Provide a name and description for it. I entered "Convert Video

File" for the name, and "Convert from one video format to another" for the description.

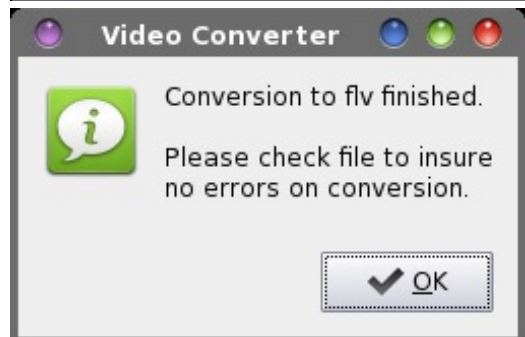
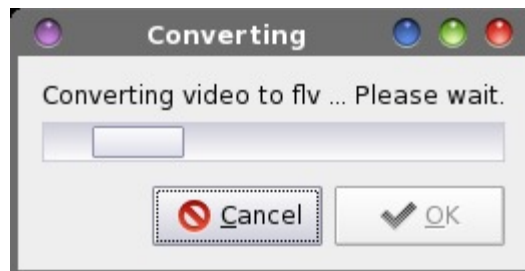
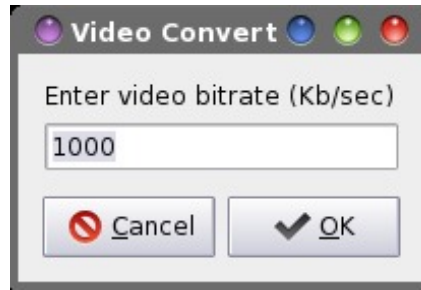
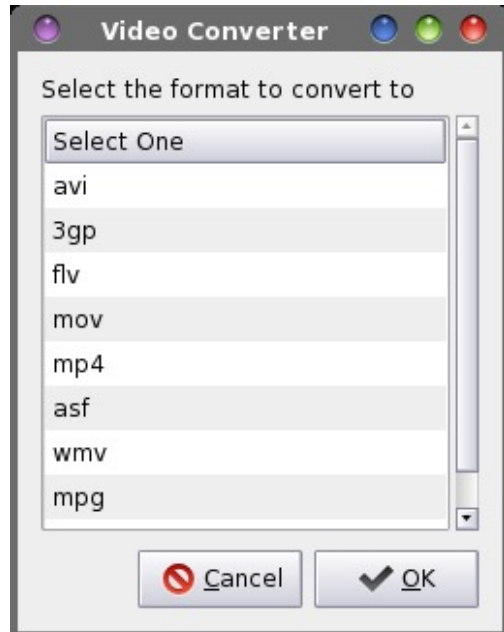
Next, enter the following into the "Command" field:

\$HOME/Scripts/convert-video.sh

Since I store all of my custom bash scripts in my /home directory in a subdirectory called /Scripts, I've pointed the command to that location. Either save the bash script to a location that is listed in your PATH statement (type **echo \$PATH** at a command line prompt in a terminal session), or provide the full path and filename in the command field of the Edit Action dialog box.

Select an icon for your new custom action. I chose one that relates to videos, naturally. Under the "Appearance Conditions" tab, leave the "File pattern" set to "*" and make sure ONLY "Video files" is checked.

Below is what you will see on your screen when you use this Thunar Custom Action.



Now, converting from one video file format to another is a very simple task using Xfce and Thunar.

Create a playlist

In today's digital world, many of us have a rather large collection of music files stored on our computer. With that many files, there's no way to sit down and listen to them all at one time. What's best is to create custom playlists that list the music you want to listen to. This next Thunar Custom Action does exactly that.

First, copy or type in the following bash file. Be sure to save it either somewhere in your path, or be prepared to provide the full path and filename of the file when you call the file from your Thunar Custom Action. I called mine (creatively enough) make-playlist.sh.

```
#!/bin/bash
```

```
# Create a playlist from a group of selected songs
```

```
List=$(zenity --entry --title="Make Playlist: Enter a Name" --text="The m3u extension will be added for you automatically.")
```

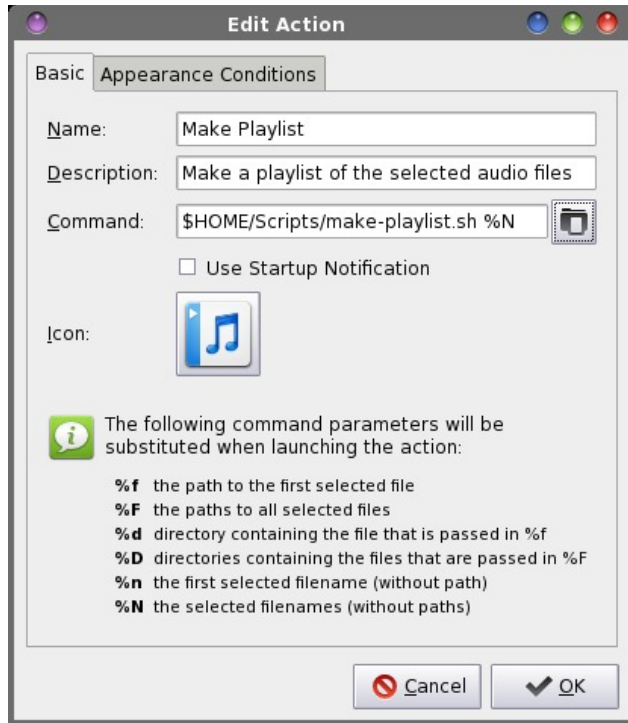
```
ret="$?"
```

```
if [ "$ret" = "252" ]; then
    exit
elif [ "$ret" = "3" ]; then
    exit
elif [ "$List" = "" ]; then
    List="playlist"
fi
```

```
n=1
echo "#EXTM3U" >>./"$List".m3u
```

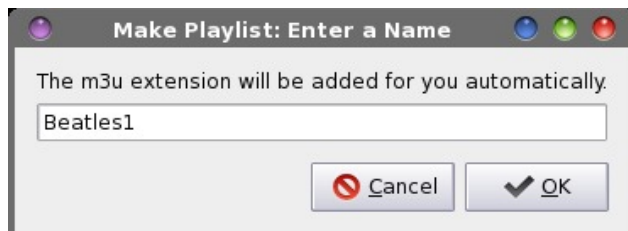
```
for i in "$@"; do
    echo "$i" >>./"$List".m3u
done
```

```
exit
```



Next, create a new Thunar Custom Action (if you've gotten this far, I don't need to tell you how to do that any longer). Enter a name and description for your new custom action. I entered "Make Playlist" for the name, and "Make a playlist of the selected audio files" as the description.

Enter the name of the bash script in the "Command" field. I entered it as **\$HOME/Scripts/make-playlist.sh %N**, since I keep all of my custom scripts in that particular directory. Next, choose an icon for your new custom action. I chose a generic audio file icon to use.



The script will display a zenity dialog box that asks you to provide a name for the new playlist. Enter only the name, since the *.m3u file extension will be automatically added for you. If you fail to enter a name, the filename will default to playlist.m3u. Next, the name of each file is appended to the playlist file, and the file created.

Here are the first 10 entries in a playlist I created from my digital files that I ripped from my CD of the Beatles One album:

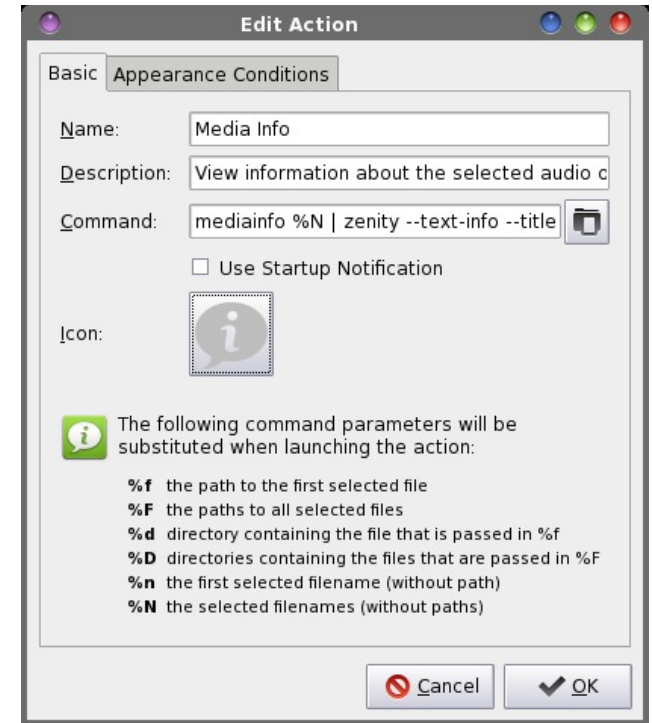
- 01 - Love Me Do.mp3
- 02 - From Me To You.mp3
- 03 - She Loves You.mp3
- 04 - I Want To Hold Your Hand.mp3
- 05 - Can't Buy Me Love.mp3
- 06 - A Hard Day's Night.mp3
- 07 - I Feel Fine.mp3
- 08 - Eight Days A Week.mp3
- 09 - Ticket To Ride.mp3
- 10 - Help!.mp3

Now, whenever you select multiple music files, then select "Make Playlist" from Thunar's right click context menu, you can create custom playlists for those music selections that can be played with virtually any MP3 player on your desktop, or via the "Play Sound File" Thunar Custom Action that we previously created.

View info about media files

Sometimes, you run across a multimedia file (audio or video) that is ... shall we say ... problematic. Looking inside the file can sometimes give us clues as to why the file is giving us problems. The next Thunar Custom Action does exactly that.

For this custom action, you will need to install **mediainfo** from the PCLinuxOS repository. Mediainfo is a command line utility that is specifically written to extract all the data flags that are stored inside a multimedia file.



Create your new Thunar Custom Action. Enter a name and a description for it. I entered "Media Info" as the name, and "View information about the selected audio or video files" as the description.

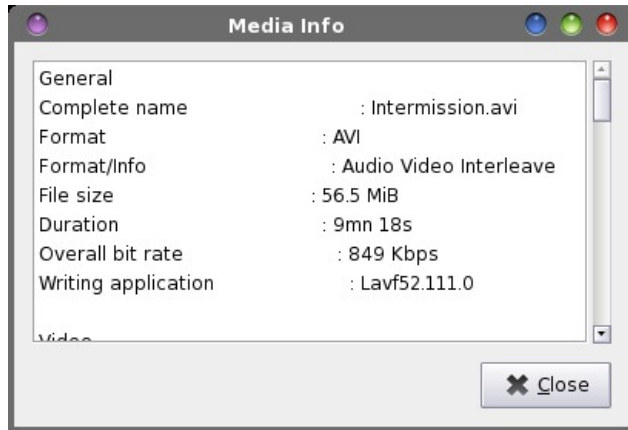
In the "Command" field, enter the following:

mediainfo %N | zenity --text-info --title="Media Info" --width=400 --height=250

This will run mediainfo on the selected files, and display the information in a zenity dialog box on your screen. Select an icon for your new custom action. Under the "Appearance Conditions" tab, leave the "File pattern" selection at the default "*" setting, and make sure both "Audio files" and "Video files" are checked.

Now, whenever you select "Media Info" from Thunar's right click context menu, you will see

something similar to the following displayed on your computer screen.

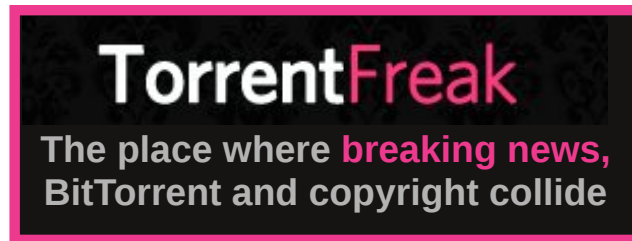


All of the available information for *all* of the selected media files is displayed in the window, one file after another.

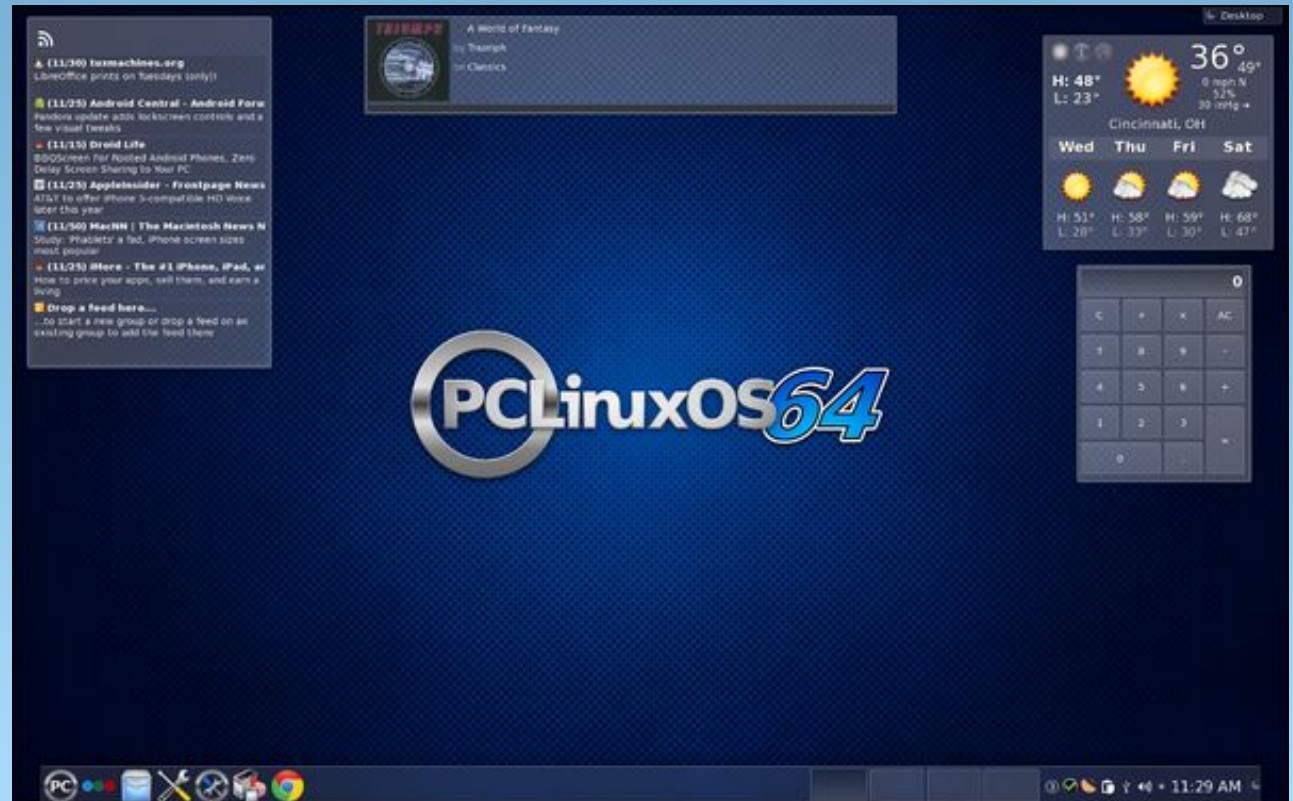
Summary

As you can see, there are many things you can do in Xfce to make it easier to work with multimedia files. Basically, if you can play it on your computer in a media playback program, you can also play it from a Thunar Custom Action.

There are more custom actions you can define, and the needs of each person is going to differ, depending on what tasks they are trying or wanting to perform. The possibilities are limited only by your imagination and your command line prowess.



Screenshot Showcase



Posted by ruel24, April 2, 2013, running KDE.



**The PCLinuxOS
Magazine**

**Created with
Scribus 1.4.1**

ms_meme's Nook: Top Of The Forum

FORUM

Such a feeling's coming over me
I downloaded PCLOS you see
What Windows never brought
PCLOS has got
And I know I'm going to like it a lot

Everything an OS should be
Is now waiting on my desktop for me
And the reason why
Is Texstar did apply
Everything that Windows has not

I'm on the top of the Forum
Looking down on the Sandbox
And the only explanation I can find
Are the songs that I've played
And the friends that I've made
They put me on the top of the forum

SANDBOX

MP3

OGG

Game Zone: Orcs Must Die! 2

by daiashi

Orcs Must Die! 2 Key Features:



About The Game

Orcs Must Die! 2 thrusts players and their friends back into battle against a terrifying mob of orcs and other monsters. With a redesigned spellbook, players will have a broader range of options to build a deadly arsenal of defenses. Investing hard-earned skulls in an enormous new upgrade system will allow players to cater to their favorite methods of burning, grinding, tossing, or dismembering orcs.

Orcs Must Die! 2 begins days after the ending of the original game. With the rifts closed and the magical Order dead, the War Mage finds himself suddenly thrust into a new battle against the mindless orc mob. He has a powerful, if questionable, new companion in the Sorceress, an ex-War Mage responsible for the destruction of the Order.

- **Co-Op!** - Play as the War Mage, the headstrong hero who charges into danger, or play as the more strategic Sorceress who keeps the mob at bay with mind-control and magic.

- **Story-based Campaign** – Pick up where the original game left off with a brand new story-based campaign that you can play in Single-Player or Co-Op!

- **Endless Mode** - Play alone or join a friend to put your skills to the test against endless waves of increasingly difficult enemies.

- **Classic Mode** - Steam players who own the original Orcs Must Die! will automatically unlock co-op versions of 10 fan-favorite levels from the original game featuring new enemies!

- **Over 20 Deadly Enemies** - Face an army of vile new creatures like Earth Elementals, Trolls, and Bile Bats. And they've brought all of your favorite trap-fodder from the original Orcs Must Die! along with them!

- **More than 50 Traps, Weapons, and Guardians** – Choose from an enormous armory of new and classic defenses, including an all new assortment of magical trinkets.

- **Massive Upgrade System** – With more than 225 persistent trap and weapon upgrades to unlock, you can build an arsenal perfectly suited to your slaying style.

- **Extensive Replayability** – Multiple game modes, “Nightmare” difficulty, and an enormous skull-ranking system provide hours of replayability.

- **Scoring System and Leaderboard** – Compete with your friends for supremacy on single-player and co-op leaderboards!

System requirements

Software: Wine and steam.

Hardware:

- **OS:** Windows XP, Windows Vista or Windows 7
- **Processor:** 2GHz Dual Core
- **Memory:** 2 GB RAM
- **Graphics:** NVIDIA GeForce 6800 or ATI Radeon x1950 or better with 256MB VRAM
- **DirectX®:** dx90c
- **Hard Drive:** 5 GB HD space
- **Additional:** Broadband Internet Connection recommended for co-op play

About The Company

Based in Plano, Texas, Robot Entertainment is a world-class independent game development studio owned and operated by many of the founders of Ensemble Studios. With a team that has proven experience and expertise in creating games that appeal to massive audiences, Robot Entertainment is focused on titles that set new standards for their respective genres, as well as groundbreaking original Intellectual Properties. In addition to their

Game Zone: Orcs Must Die! 2

previous work on the [Age of Empires](#) series and [Halo Wars](#), their released titles include [Orcs Must Die™!](#) for PC and Xbox360, [Hero Academy™](#) for iOS and PC, and [Orcs Must Die 2™!](#) for PC.

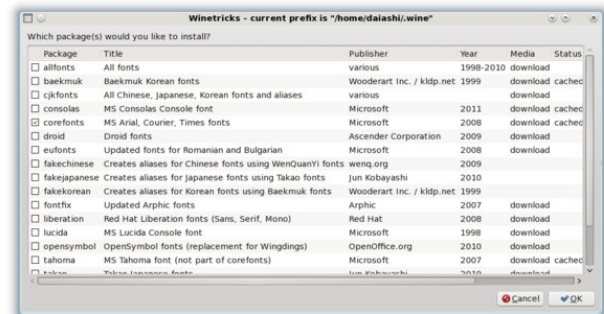
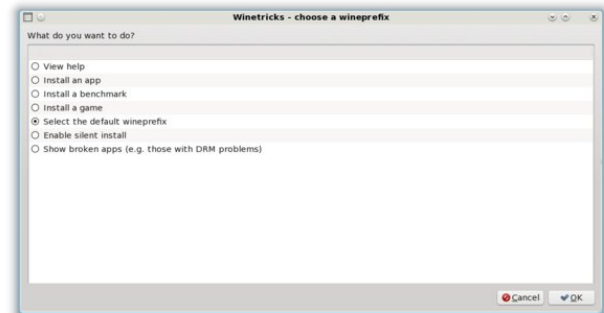
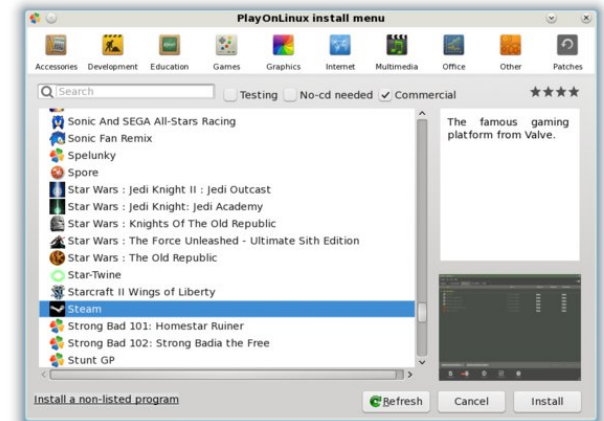
Some Gameplay Screenshots



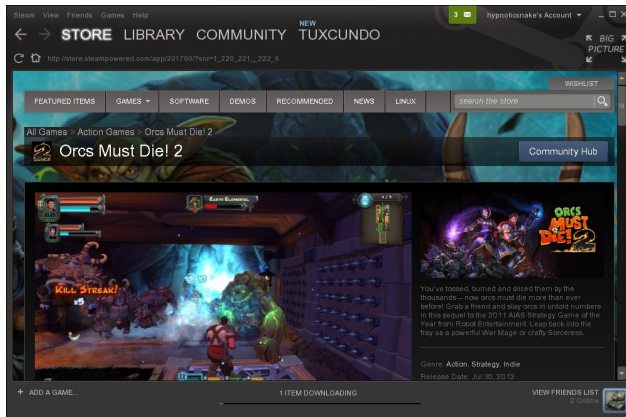
Getting It To Run

Ok, first things first. Head on over and grab [Steam](#). WINE should install it with no problems. You will need Winetricks from Synaptic to get some needed files for steam to run properly. These are corefonts and vcrun6, both from winetricks. Alternately, you

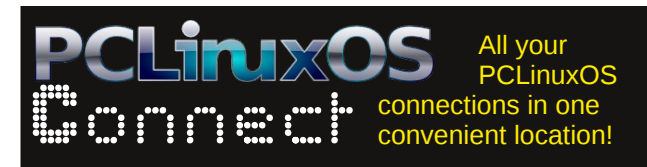
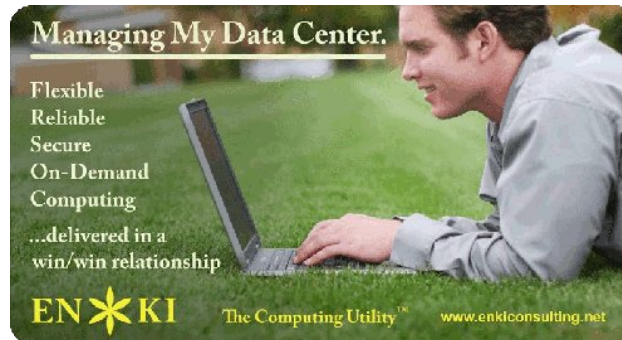
Playonlinux is installed, search for Steam under the game list or type it in the search bar.



can get steam through Playonlinux, which can be found in Synaptic package manager. Then, after



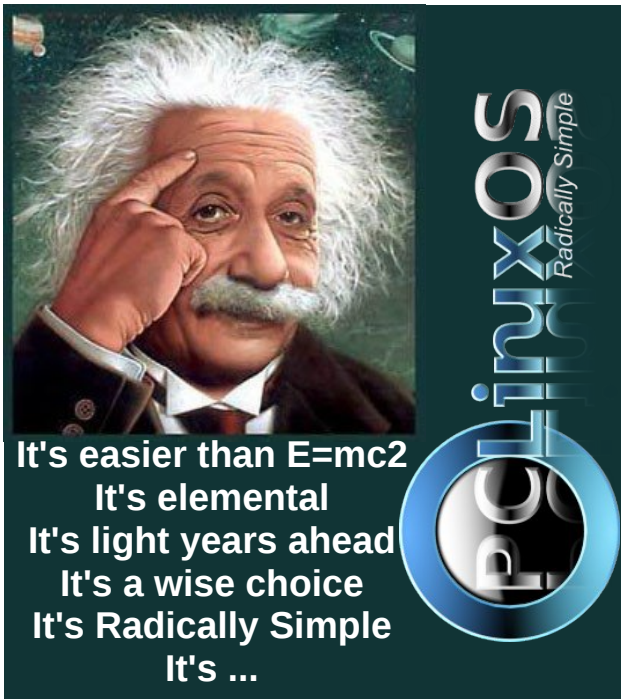
After steam is installed, search for Orcs Must Die 2. The demo will be there. The download is not that large. The game played great, except for one exception. You may have to enter a `chmod o-r /dev/input/js*` as root if you have no mouse control, as the game is looking for a game pad. Unless, of course, that's what you're using. It's a pretty fun game, and coop as well.



Screenshot Showcase



Posted by nymira, April 1, 2013, running KDE.



Managing Running Processes From Terminal

by Antonis Komis (agmg)

The Linux terminal has several useful programs and commands for managing the processes running on your system. In this article, we'll discover what a process is and list the most well-known commands that will allow you to view, kill or change the priority of a running process.

What is a process?

Every computer program is a collection of instructions, written to perform a specific task. When you run a program, the operating system loads it on RAM (Random Access Memory), in order for the CPU (Central Processing Unit) to access it and execute it. A process is an instance of a computer program that is currently being executed. It contains the program code and its current activity. Depending on the operating system, a process may be made up of multiple threads that execute instructions concurrently.

In multitasking operating systems, multiple processes can share processors (CPUs) and other system resources. Multitasking allows each processor to switch between tasks that are being executed without having to wait for each task to finish. Depending on the operating system implementation, switches could be performed when tasks perform input/output operations, when a task indicates that it can be switched, or on hardware interrupts.

```
top - 22:50:10 up 1:31, 3 users, load average: 2.70, 2.61, 2.56
Tasks: 170 total, 1 running, 169 sleeping, 0 stopped, 0 zombie
Cpu(s): 27.4%us, 5.0%sy, 67.4%ni, 0.0%id, 0.0%wa, 0.0%hi, 0.2%si, 0.0%st
Mem: 4019552k total, 3851232k used, 168320k free, 1081484k buffers
Swap: 4088504k total, 8k used, 4088496k free, 1262452k cached
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
3606	root	1	19	280m	196m	3032	S	136	5.0	134:46.38	FahCore_a3.exe
1144	root	1	0	380m	128m	86m	S	20	3.3	10:47.86	X
13376	agmg	1	0	315m	29m	21m	S	18	0.8	0:00.55	ksnapshot
3608	agmg	1	0	2964m	92m	37m	S	10	2.4	1:38.33	kwin
3625	agmg	1	0	2971m	108m	39m	S	3	2.8	1:15.63	plasma-desktop
7127	agmg	1	0	347m	34m	19m	S	2	0.9	0:24.34	plugin-containe
9860	agmg	1	0	541m	35m	22m	S	2	0.9	0:07.69	konsole
3896	agmg	1	0	567m	45m	24m	S	1	1.2	2:50.88	transmission-qt
11974	agmg	6	0	11144	1200	852	R	1	0.0	0:04.38	top
2606	agmg	1	0	16828	2036	736	S	0	0.1	0:04.23	dbus-daemon
3027	agmg	1	0	355m	11m	5860	S	0	0.3	0:00.40	klauncher
3031	agmg	1	0	822m	33m	20m	S	0	0.9	0:05.15	kded4
3134	agmg	1	0	443m	21m	11m	S	0	0.6	0:01.43	qglobalaccel
3381	agmg	1	0	521m	22m	13m	S	0	0.6	0:00.92	kssmserver
3636	agmg	1	0	152m	6388	4928	S	0	0.2	0:00.57	akonadi_control
3640	agmg	1	0	1114m	33m	5404	S	0	0.9	0:06.48	mysqld
3915	agmg	1	0	232m	49m	12m	S	0	1.3	0:44.80	net_applet
3948	agmg	1	0	302m	18m	13m	S	0	0.5	0:00.48	kwalletmanager
6645	agmg	1	0	1347m	478m	41m	S	0	12.2	16:38.00	firefox
9025	root	1	0	0	0	0	S	0	0.0	0:11.56	kworker/1:1
12489	root	1	0	0	0	0	S	0	0.0	0:01.46	kworker/0:1
1	root	1	0	4124	704	604	S	0	0.0	0:00.67	init
2	root	1	0	0	0	0	S	0	0.0	0:00.00	kthreadd
3	root	1	0	0	0	0	S	0	0.0	0:00.22	ksoftirqd/0
5	root	1	0	0	0	0	S	0	0.0	0:00.00	kworker/u:0
6	root	RT	0	0	0	0	S	0	0.0	0:00.00	migration/0
7	root	RT	0	0	0	0	S	0	0.0	0:00.00	migration/1
9	root	1	0	0	0	0	S	0	0.0	0:00.24	ksoftirqd/1
11	root	1	-20	0	0	0	S	0	0.0	0:00.00	khelper
12	root	1	0	0	0	0	S	0	0.0	0:00.00	kdevtmpfs
13	root	1	-20	0	0	0	S	0	0.0	0:00.00	netns

top

The **top** program provides a dynamic, real-time view of a running system. It displays system summary information (like uptime, cpu, memory and swap usage etc) and a list of the most CPU-intensive processes running on your system. The types of system summary information shown and the types, order and size of information displayed for tasks are all user configurable and that configuration can be made persistent across restarts. The top program is installed by default in PCLinuxOS and can be run by issuing the command **top** in your favorite terminal emulator (below, left).

Let's have a more detailed look at this window.

First row: top

```
top - 22:50:10 up 1:31, 3 users, load average: 2.70, 2.61, 2.56
```

Information displayed:

- Current system time (22:50:10)
- System uptime - time the system is up and running (1:31)
- Users logged in (3 users)
- Average load on the system (The 3 values refer to last 1, 5 and 15 minutes, respectively)

Second row: tasks

```
Tasks: 170 total, 1 running, 169 sleeping, 0 stopped, 0 zombie
```

Information displayed:

- Total number of processes (170 total)
- Number of running processes (1 running)
- Number of sleeping processes (169 sleeping)
- Number of stopped processes (0 stopped)
- Number of zombie processes (0 zombie)

Third row: cpu(s)

```
Cpu(s): 27.4%us, 5.0%sy, 67.4%ni, 0.0%id, 0.0%wa, 0.0%hi, 0.2%si,
```

Information displayed:

- Percentage of the CPU used for user processes (27.4%**us**)
- Percentage of the CPU used for system processes (5.0%**sy**)
- Percentage of the CPU used for priority upgrade - nice (67.4%**ni**)
- Percentage of the CPU not used (0.0%**id**)

- Percentage of the CPU used for processes that wait an I/O operation (0.0%**wa**)
- Percentage of the CPU used for hardware interrupts (0.0%**hi**)
- Percentage of the CPU used for software interrupts (0.2%**si**)
- Percentage of the CPU 'stolen' from a virtual machine by the hypervisor for other tasks (such as running another virtual machine). This will be 0 on desktop and server without Virtual machine. (0.0%**st** — Steal Time)

Fourth row: memory usage

```
Mem: 4019552k total, 3851232k used, 168320k free, 1081484k buffer
```

Information displayed:

- Total available memory (4019552k total)
- Used memory (3851232k used)
- Free memory (168320k free)
- Cached memory (1081484k buffers)

Similar information is displayed in the fifth row but for swap (virtual memory)

```
Swap: 4088504k total, 8k used, 4088496k free, 1262452k cached
```

The following rows constitute the list of the processes running on your system. Let's have a look at the information that different columns provide:

- **PID**: Process Identifier
- **User**: The user that owns the process
- **PR**: The priority of the process
- **NI**: The nice value of the process (we'll talk about nice later)
- **VIRT**: virtual memory used by the process
- **RES**: physical memory used from the process
- **SHR**: shared memory of the process
- **S**: indicates the status of the process: S=sleep R=running Z=zombie
- **%CPU**: This is the percentage of CPU used by this process
- **%MEM**: This is the percentage of RAM used by the process
- **TIME+**: This is the total time of activity of this process
- **COMMAND**: this is the name of the process

The top program can be run with several command-line arguments:

- d** Specifies the delay between screen updates. You can change this with the **s** interactive command.
- p** Monitor only processes with given process id. This flag can be given up to twenty times. This option is neither available interactively nor can it be put into the configuration file.
- q** This causes top to refresh without any delay. If the caller has superuser

privileges, top runs with the highest possible priority.

-**S** Specifies cumulative mode, where each process is listed with the CPU time that it, as well as its dead children, has spent.

-**s** Tells top to run in secure mode. This disables any interactive commands.

-**i** Start top ignoring any idle or zombie processes.

-**C** display total CPU states instead of individual CPUs. This option only affects SMP systems.

-**c** display command line instead of the command name only.

-**H** Show all threads.

-**n** Number of iterations. Update the display this number of times and then exit.

-**b** Batch mode. Useful for sending output from top to other programs or to a file. In this mode, top will not accept command line input. It runs until it produces the number of iterations requested with the **n** option or until killed. Output is plain text suitable for display on a dumb terminal.

Several single-key commands are recognized while top is running (interactive commands). Some are disabled if the **s** option has been given in the command line.

space Immediately updates the display.

^L Erases and redraws the screen.

h or **?** Displays a help screen giving a brief summary of commands, and the status of secure and cumulative modes.

k Kill a process. You will be prompted for the PID of the task, and the signal to send to it. For a normal kill, send signal 15. For a sure, but rather abrupt, kill, send signal 9. The default signal is 15, SIGTERM. This command is not available in secure mode.

i Ignore idle and zombie processes. This is a toggle switch.

I Toggle between Solaris (CPU percentage divided by total number of CPUs) and Irix (CPU percentage calculated solely by amount of time) views. This is a toggle switch that affects only SMP systems.

Managing Running Processes From Terminal

n or **#** Change the number of processes to show. You will be prompted to enter the number. This overrides automatic determination of the number of processes to show, which is based on window size measurement. If 0 is specified, then top will show as many processes as will fit on the screen; this is the default.

q Quit.

r Re-nice a process. You will be prompted for the PID of the task, and the value to nice it to. Entering a positive value will cause a process to be niced to negative values, and lose priority. If root is running top, a negative value can be entered, causing a process to get a higher than normal priority. The default re-nice value is 10. This command is not available in secure mode.

S This toggles cumulative mode, the equivalent of **ps -S**, i.e., that CPU times will include a process's defunct children. For some programs, such as compilers, which work by forking into many separate tasks, normal mode will make them appear less demanding than they actually are. For others, however, such as shells and init, this behavior is correct. In any case, try cumulative mode for an alternative view of CPU use.

s Change the delay between updates. You will be prompted to enter the delay time, in seconds, between updates. Fractional values are recognized down to microseconds. Entering 0 causes continuous updates. The default value is 5 seconds. Note that low values cause nearly unreadably fast displays, and greatly raise the load. This command is not available in secure mode.

f or **F** Add fields to display or remove fields from the display.

o or **O** Change order of displayed fields. See below for more information.

l toggle display of load average and uptime information.

m toggle display of memory information.

t toggle display of processes and CPU states information.

c toggle display of command name or full command line.

N sort tasks by pid (numerically).

A sort tasks by age (newest first).

P sort tasks by CPU usage (default).

M sort tasks by resident memory usage.

T sort tasks by time / cumulative time.

W Write current setup to `~/toprc`. This is the recommended way to write a top configuration file. Using this command, next time top is started, it will use all configuration edits you've made (except those which -by design- can't be included in the configuration file).

You can find a lot more information about **top** in its man (manual) page:

<http://unixhelp.ed.ac.uk/CGI/man-cgi?top>

htop

```
1 [|||||] 100.0% Tasks: 115, 190 thr; 4 running
2 [|||||] 100.0% Load average: 2.63 2.67 2.69
Mem [|||||] 1566/3925MB Uptime: 02:26:10
Swp [|||||] 0/3992MB

PID USER PRI NI VIRT RES SHR S CPU% MEM% TIME+ Command
3686 root 1 19 280M 195M 194M S 156.5 5.0 3811.49 ./FahCore a3.exe -dir work/ -nice 10 -suffix 08
3688 root 39 18 280M 195M 194M R 78.5 5.0 1845.57 ./FahCore a3.exe -dir work/ -nice 19 -suffix 08
1144 root 6 0 390M 132M 80916 S 11.6 3.4 17:02.08 /etc/X11/X :0 vt8 -deferglyphs 16 -nolisten tcp
3608 agmg 4 0 2963M 92988 36860 S 4.5 2.3 2:34.12 kwin
3625 agmg 6 0 2971M 101M 34116 R 3.2 2.6 1:51.29 kdeinit4: plasma-desktop [kdeinit]
19054 agmg 6 0 2260M 1944 1268 R 3.2 0.0 0:00.75 htop
16617 agmg 1 0 535M 29176 17884 S 3.2 0.7 0:06.53 kdeinit4: konsole [kdeinit]
19096 agmg 1 0 315M 38656 22528 S 2.6 0.8 0:00.57 /usr/bin/ksnapshot -caption KSnapshot
17127 agmg 1 0 355M 40120 20212 S 2.6 1.0 1:28.01 /usr/lib64/firefox-20.0/plugin-container /usr/l
14028 agmg 1 0 620M 65012 37248 S 1.9 1.6 1:28.96 /usr/bin/qlview /home/agmg/Pictures/processes
6645 agmg 1 0 1400M 532M 41960 S 1.9 13.6 33:56.74 /usr/bin/firefox
3896 agmg 1 0 567M 46852 25328 S 1.3 1.2 3:29.53 /usr/bin/transmission-qt
3900 agmg 1 0 567M 46852 25328 S 0.6 1.2 3:07.25 /usr/bin/transmission-qt
3623 agmg 1 0 1008M 44592 18064 S 0.6 1.1 0:02.62 /usr/bin/Anatifa4
3628 agmg 1 0 864M 56136 27368 S 0.6 1.4 0:13.00 /usr/bin/lancelot
3729 agmg 1 0 387M 23408 19168 S 0.6 0.6 0:00.49 /usr/bin/akonadi agent launcher akonadi_maildir
3775 agmg 1 0 608M 38124 20240 S 0.6 0.9 0:05.67 kdeinit4: krunner [kdeinit]
6645 agmg 1 0 446M 28064 11660 S 0.6 0.5 0:00.66 kdeinit4: klipper [kdeinit]
3805 agmg 1 0 599M 26912 15792 S 0.6 0.7 0:00.95 kdeinit4: kmix [kdeinit]
3815 agmg 1 0 232M 40204 10992 S 0.0 1.2 1:11.40 /usr/bin/perl /usr/bin/net applet
3287 agmg 1 0 849M 21876 13812 S 0.0 0.5 0:01.24 /usr/bin/kactivitymanagerd
3472 root 1 0 4324 4164 1428 S 0.0 0.1 0:06.92 /usr/sbin/snmpd -sd -f /dev/null -p /var/run/
3650 agmg 1 0 1114M 35272 2524 S 0.0 0.9 0:04.02 /usr/sbin/mysqld --defaults-file=/home/agmg/.lo
1 root 1 0 4124 704 604 S 0.0 0.0 0:00.72 init [5]
109 root 1 0 11268 1384 484 S 0.0 0.0 0:00.05 /sbin/udev -d
888 root 1 0 4136 780 560 S 0.0 0.0 0:00.00 /usr/sbin/acpid
891 root 1 0 6228 668 524 S 0.0 0.0 0:00.26 syslogd -m 0
910 root 1 0 4120 484 384 S 0.0 0.0 0:00.01 klogd -x
933 messagebu 1 0 15888 1864 900 S 0.0 0.0 0:11.58 dbus-daemon --system

F1Help F2Setup F3Search F4Filter F5Tree F6SortBy F7Nice F8Lice F9Kill F10Quit
```

htop is an improved version of **top**. It gives you the ability to scroll the list, both vertically and horizontally, to see all processes and their command lines. Additionally, you can perform process-related tasks (kill, nice) without entering the PID of the process.

Let's have a look at the top section of this window:

```
1 [|||||] 100.0% Tasks: 115, 190 thr; 4 running
2 [|||||] 100.0% Load average: 2.63 2.67 2.69
Mem [|||||] 1566/3925MB Uptime: 02:26:10
Swp [|||||] 0/3992MB
```

This section provides information about CPU Usage (separate graph for each core), memory and swap usage as long as information about running tasks and system load and uptime. You can fully customize this section to display more or less information by pressing **F2** (Setup).

Below this section is the list of running processes which is similar to the one we saw previously on **top**. As mentioned earlier, you can use the arrow keys of your

Managing Running Processes From Terminal

keyboard to navigate inside this list, both vertically and horizontally. **htop** also supports mouse interaction.

The footer of this window displays the **htop** menu commands.



By pressing **F1** you get a complete list of **htop** commands:

```
htop 1.0.2 - (C) 2004-2011 Hisham Muhammad
Released under the GNU GPL. See 'man' page for more info.

CPU usage bar: [low-priority/normal/kernel/virtualiz used%]
Memory bar:   [used/buffers/cache used/total]
Swap bar:     [used used/total]
Type and layout of header meters are configurable in the setup screen.

Status: R: running; S: sleeping; T: traced/stopped; Z: zombie; D: disk sleep
Arrows: scroll process list          F5 t: tree view
Digits: incremental PID search      u: show processes of a single user
F3 /: incremental name search       H: hide/show user threads
F4 \: incremental name filtering     K: hide/show kernel threads
Space: tag processes               F: cursor follows process
U: untag all processes             + -: expand/collapse tree
F9 k: kill process/tagged processes P M T: sort by CPU%, MEM% or TIME
] F7: higher priority (root only)   i: set IO priority
[ F8: lower priority (+ nice)       I: invert sort order
a: set CPU affinity                F6 >: select sort column
F2 S: setup                        l: list open files with lsof
? F1 h: show this help screen      s: trace syscalls with strace
F10 q: quit

Press any key to return.
```

htop lets you, among other things, search for a specific process by its name, sort the process list by any field (CPU usage, memory usage etc) and select multiple processes to kill or increase/decrease their priority (nice).

Like top, htop can also be run using command line arguments:

- d --delay=DELAY** Delay between updates, in tenths of seconds
- C --no-color --no-colour** Start htop in monochrome mode
- h --help** Display a help message and exit
- p --pid=PID,PID...** Show only the given PIDs
- s --sort-key COLUMN** Sort by this column (use --sort-key help for a column list)

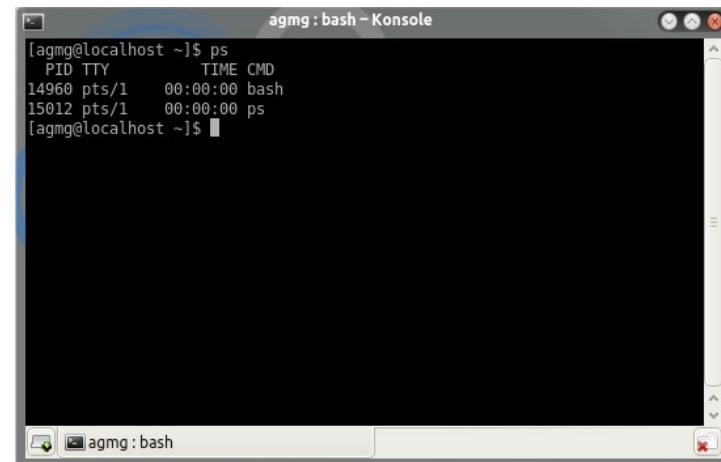
-u --user=USERNAME Show only the processes of a given user

-v --version Output version information and exit

If you want to get more information on htop, simply type **man htop** in terminal or visit the following page: <http://linux.die.net/man/1/htop>

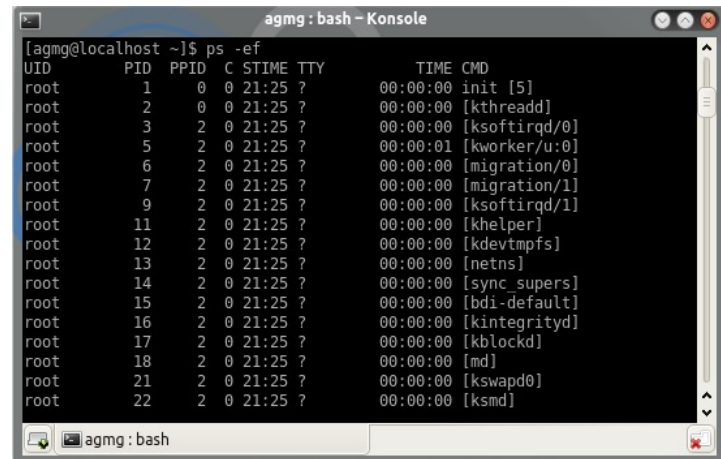
ps

The **ps** command is another way of displaying running processes from terminal. Typical syntax for the **ps** command is **ps [options]**



Running **ps** without options, displays information about current terminal:

To display full information about each process currently running, type **ps -ef**

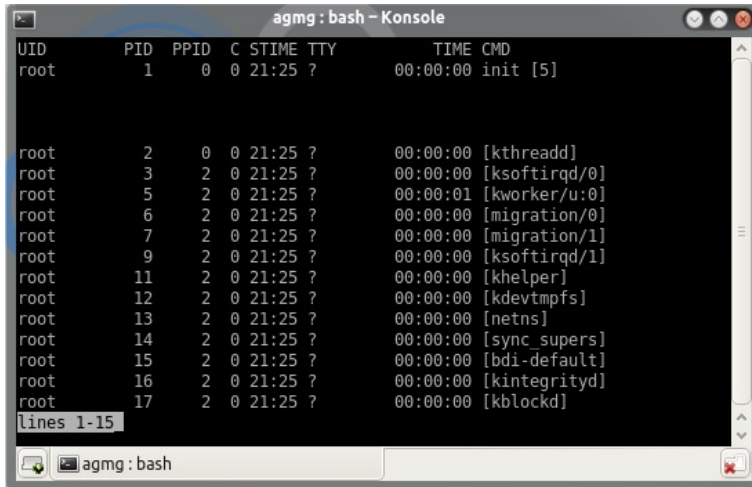


-e lists information about every process that is currently running

-f generates a full listing.

Managing Running Processes From Terminal

To make the list easier to read, you can pipe the output through the `less` command to scroll it at your own pace (**`ps -ef | less`**):

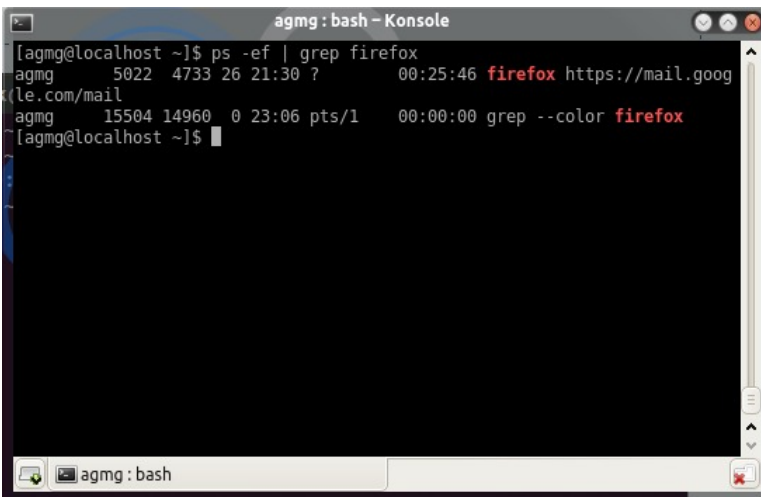


```
agmg : bash - Konsole
UID      PID  PPID  C  STIME TTY      TIME  CMD
root      1    0    0  21:25 ?        00:00:00 init [5]

root      2    0    0  21:25 ?        00:00:00 [kthreadd]
root      3    2    0  21:25 ?        00:00:00 [ksoftirqd/0]
root      5    2    0  21:25 ?        00:00:00 [kworker/u:0]
root      6    2    0  21:25 ?        00:00:00 [migration/0]
root      7    2    0  21:25 ?        00:00:00 [migration/1]
root      9    2    0  21:25 ?        00:00:00 [ksoftirqd/1]
root     11    2    0  21:25 ?        00:00:00 [khelper]
root     12    2    0  21:25 ?        00:00:00 [kdevtmpfs]
root     13    2    0  21:25 ?        00:00:00 [netns]
root     14    2    0  21:25 ?        00:00:00 [sync_supers]
root     15    2    0  21:25 ?        00:00:00 [bdi-default]
root     16    2    0  21:25 ?        00:00:00 [kintegrityd]
root     17    2    0  21:25 ?        00:00:00 [kblockd]

lines 1-15
agmg : bash
```

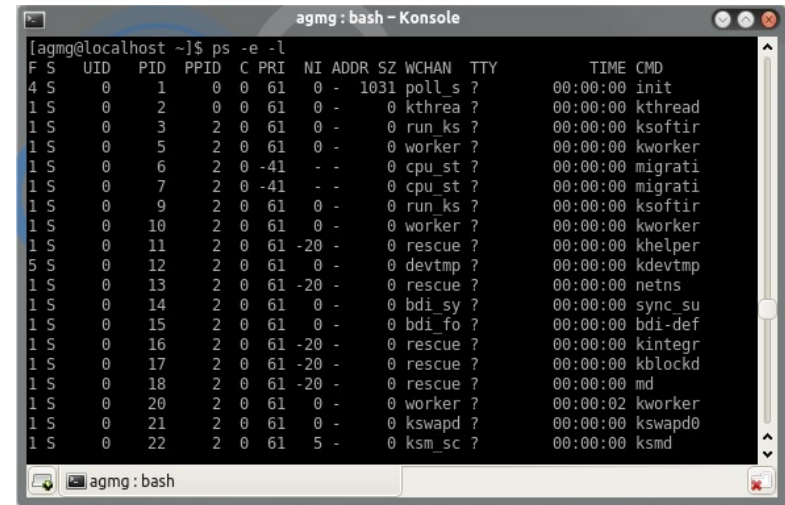
Press **q** to exit when you're done.



```
agmg : bash - Konsole
[agmg@localhost ~]$ ps -ef | grep firefox
agmg    5022  4733  26  21:30 ?        00:25:46 firefox https://mail.goog
le.com/mail
agmg    15504 14960   0  23:06 pts/1    00:00:00 grep --color firefox
[agmg@localhost ~]$
```

You could also pipe the output through **`grep`** to search for a specific process without using any other commands. For example, the command **`ps -ef | grep firefox`**, will list all Firefox related processes:

You can also view the process in long format using the **`-l`** option (top, right):

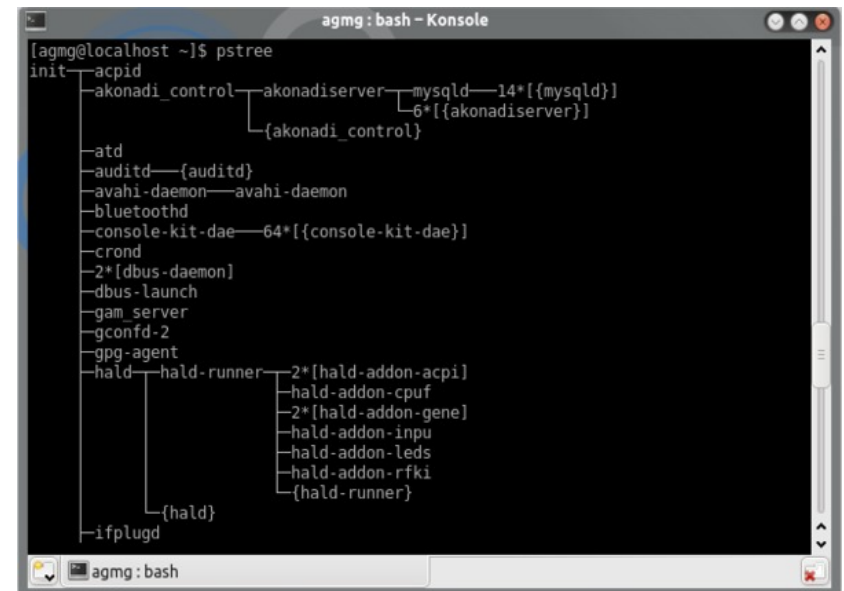


```
agmg : bash - Konsole
[agmg@localhost ~]$ ps -e -l
F S      UID      PID  PPID  C  PRI  NI ADDR SZ WCHAN  TTY      TIME  CMD
4 S      0         1    0    0  61   0  -  1031 poll_s ?        00:00:00 init
1 S      0         2    0    0  61   0  -    0 kthrea ?        00:00:00 kthread
1 S      0         3    2    0  61   0  -    0 run_ks ?        00:00:00 ksoftir
1 S      0         5    2    0  61   0  -    0 worker ?        00:00:00 kworker
1 S      0         6    2    0  41   -  -    0 cpu_st ?        00:00:00 migrati
1 S      0         7    2    0  41   -  -    0 cpu_st ?        00:00:00 migrati
1 S      0         9    2    0  61   0  -    0 run_ks ?        00:00:00 ksoftir
1 S      0        10    2    0  61   0  -    0 worker ?        00:00:00 kworker
1 S      0        11    2    0  61  -20  -    0 rescue ?        00:00:00 khelper
5 S      0        12    2    0  61   0  -    0 devtmp ?        00:00:00 kdevtmp
1 S      0        13    2    0  61  -20  -    0 rescue ?        00:00:00 netns
1 S      0        14    2    0  61   0  -    0 bdi_sy ?        00:00:00 sync_su
1 S      0        15    2    0  61   0  -    0 bdi_fo ?        00:00:00 bdi-def
1 S      0        16    2    0  61  -20  -    0 rescue ?        00:00:00 kintegr
1 S      0        17    2    0  61  -20  -    0 rescue ?        00:00:00 kblockd
1 S      0        18    2    0  61  -20  -    0 rescue ?        00:00:00 md
1 S      0        20    2    0  61   0  -    0 worker ?        00:00:02 kworker
1 S      0        21    2    0  61   0  -    0 kswapd ?        00:00:00 kswapd0
1 S      0        22    2    0  61   5  -    0 ksm_sc ?        00:00:00 ksmd
```

For a complete listing of the **`ps`** command options, you can consult its manual page: <http://linux.die.net/man/1/ps>

pstree

`pstree` is a command similar to **`ps`** but displays the process list using a tree view:



```
agmg : bash - Konsole
[agmg@localhost ~]$ pstree
init--acpid
    --akonadi_control--akonadiserver--mysqlqd--14*[{mysqlqd}]
                        --6*[{akonadiserver}]
                        --{akonadi_control}
    --atd
    --auditd--{auditd}
    --avahi-daemon--avahi-daemon
    --bluetoothd
    --console-kit-dae--64*[{console-kit-dae}]
    --crond
    --2*[{dbus-daemon}]
    --dbus-launch
    --gam_server
    --gconfd-2
    --gpg-agent
    --hald--hald-addon-acpi--2*[{hald-addon-acpi}]
            --hald-addon-cpu--2*[{hald-addon-cpu}]
            --hald-addon-gene--2*[{hald-addon-gene}]
            --hald-addon-inpu--2*[{hald-addon-inpu}]
            --hald-addon-leds--2*[{hald-addon-leds}]
            --hald-addon-rfki--2*[{hald-addon-rfki}]
            --{hald-addon-rfki}
            --{hald}
    --ifplugd
```

Managing Running Processes From Terminal

ps-tree also has several command line arguments. Let's have a look at some of them:

-a Show command line arguments. If the command line of a process is swapped out, that process is shown in parentheses. **-a** implicitly disables compaction.

```

[agmg@localhost ~]$ pstree -a
init
├─acpid
├─akonadi_control
│   └─akonadiserver
│       ├──{akonadiserver}
│       ├──{akonadiserver}
│       ├──{akonadiserver}
│       ├──{akonadiserver}
│       ├──{akonadiserver}
│       └─{akonadiserver}
│   └─{akonadi_control}
├─atd -l 1.8
├─auditd
│   └─{auditd}
├─avahi-daemon
│   └─avahi-daemon
├─bluetoothd --udev
└─console-kit-dae --no-daemon
    ├──{console-kit-dae}
    ├──{console-kit-dae}
    ├──{console-kit-dae}
    ├──{console-kit-dae}
    ├──{console-kit-dae}
    └─{console-kit-dae}

```

-A Use ASCII characters to draw the tree.

```

[agmg@localhost ~]$ pstree -A
init--acpid
|
|-akonadi_control--akonadiserver--6*[{akonadiserver}]
|
|   |--akonadi_control
|   |
|   |--atd
|   |--auditd--[{auditd}]
|   |--avahi-daemon--avahi-daemon
|   |--bluetoothd
|   |--console-kit-dae--64*[{console-kit-dae}]
|   |--crond
|   |--2*[dbus-daemon]
|   |--dbus-launch
|   |--gam_server
|   |--gconfd-2
|   |--gpg-agent
|   |--hald--hald-runner--2*[hald-addon-acpi]
|   |
|   |   |--hald-addon-cpu
|   |   |--2*[hald-addon-gene]
|   |   |--hald-addon-inpu
|   |   |--hald-addon-leds
|   |   |--hald-addon-rfki
|   |   |--hald-runner
|   |
|   |--hald
|
|--ifplugd
|--kactivitymanage--5*[{kactivitymanage}]

```

-c Disable compaction of identical subtrees. By default, subtrees are compacted whenever possible (top, right).

[illegible]

-l Display long lines. By default, lines are truncated to the display width or 132 if output is sent to a non-tty or if the display width is unknown.

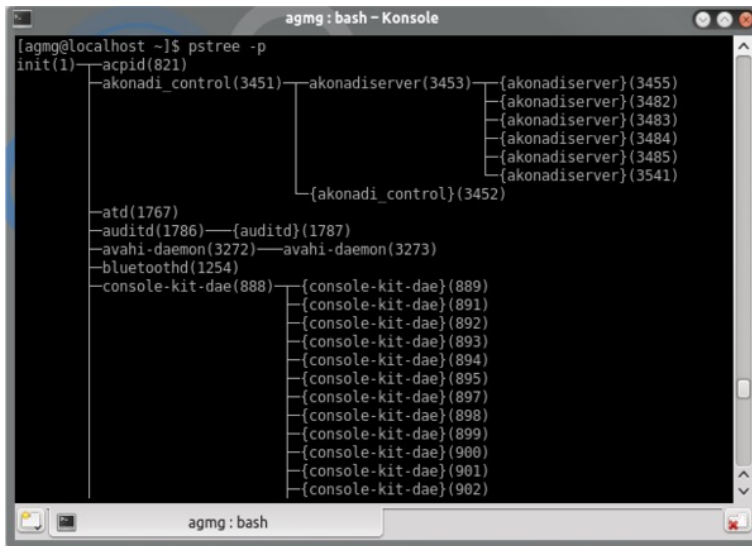
```

[agmg@localhost ~]$ pstree -l
init--acpid
      --akonadi_control--akonadiserver--6*[{akonadiserver}]
                        --{akonadi_control}
      --atd
      --auditd--{auditd}
      --avahi-daemon--avahi-daemon
      --bluetoothd
      --console-kit-dae--64*[{console-kit-dae}]
      --crond
      --2*[{dbus-daemon}]
      --dbus-launch
      --gam_server
      --gconfd-2
      --gpg-agent
      --hald--hald-runner--2*[{hald-addon-acpi}]
                        --hald-addon-cpu{f}
                        --2*[{hald-addon-gene}]
                        --hald-addon-inpu
                        --hald-addon-leds
                        --hald-addon-rfki
                        --{hald-addon-rfki}
                        --{hald}
      --ifplugd
      --kactivitymanage--5*[{kactivitymanage}]

```

-p Show PIDs. PIDs are shown as decimal numbers in parentheses after each process name. **-p** implicitly disables compaction (next page, top left).

Managing Running Processes From Terminal

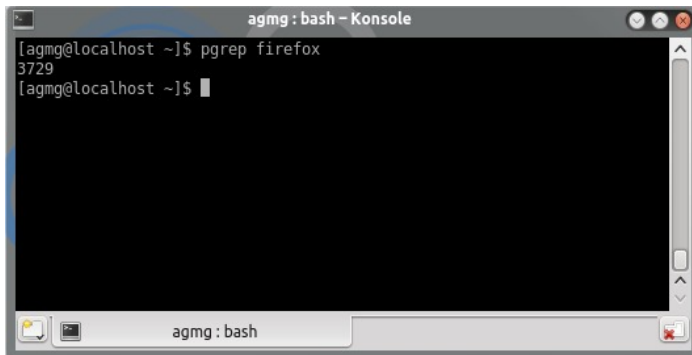


```
[agmg@localhost ~]$ pstree -p
init(1)
├── acpid(821)
├── akonadi_control(3451)
│   ├── akonadiserver(3453)
│   │   ├── {akonadiserver}(3455)
│   │   ├── {akonadiserver}(3482)
│   │   ├── {akonadiserver}(3483)
│   │   ├── {akonadiserver}(3484)
│   │   ├── {akonadiserver}(3485)
│   │   └── {akonadiserver}(3541)
│   └── {akonadi_control}(3452)
├── atd(1767)
├── auditd(1786)──{auditd}(1787)
├── avahi-daemon(3272)──avahi-daemon(3273)
├── bluetoothd(1254)
└── console-kit-dae(888)
    ├── {console-kit-dae}(889)
    ├── {console-kit-dae}(891)
    ├── {console-kit-dae}(892)
    ├── {console-kit-dae}(893)
    ├── {console-kit-dae}(894)
    ├── {console-kit-dae}(895)
    ├── {console-kit-dae}(897)
    ├── {console-kit-dae}(898)
    ├── {console-kit-dae}(899)
    ├── {console-kit-dae}(900)
    ├── {console-kit-dae}(901)
    └── {console-kit-dae}(902)
```

If you would like to get more information on **ps** and its options, you should consult its manual page: <http://unixhelp.ed.ac.uk/CGI/man-cgi?ps+1>

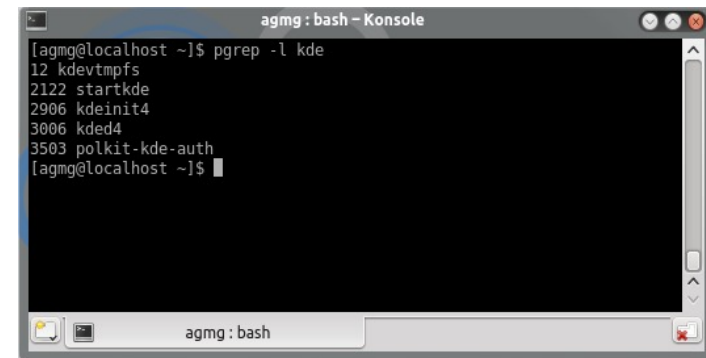
pgrep

The **pgrep** command searches the process table on the running system and prints the process IDs of all processes that match the criteria given on the command line. For example, if you want to find out the PID of Firefox, you must type **pgrep firefox**



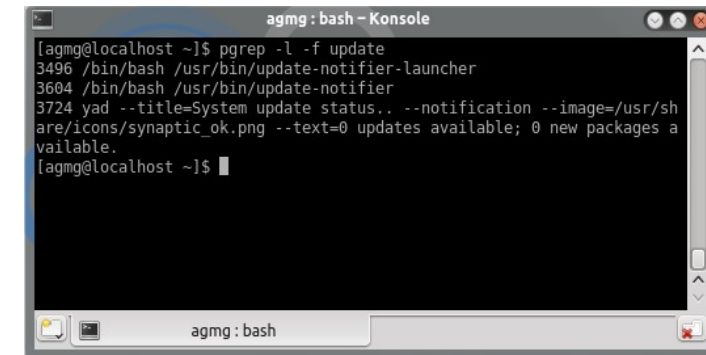
```
[agmg@localhost ~]$ pgrep firefox
3729
[agmg@localhost ~]$
```

If you want to view the process name in addition to the process ID for each matching process, you can use the **-l** argument (long output): (top, right)



```
[agmg@localhost ~]$ pgrep -l kde
12 kdevmf
2122 startkde
2906 kdeinit4
3006 kded4
3503 polkit-kde-auth
[agmg@localhost ~]$
```

Another useful argument which can be used in conjunction with **-l**, is **-f** which will display the full argument list of each matching process, as long with its PID:



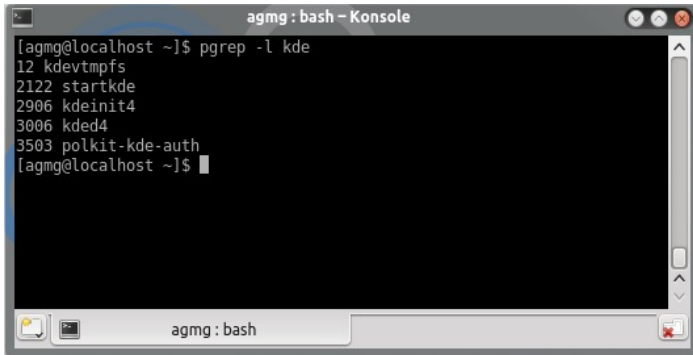
```
[agmg@localhost ~]$ pgrep -l -f update
3496 /bin/bash /usr/bin/update-notifier-launcher
3604 /bin/bash /usr/bin/update-notifier
3724 yad --title=System update status... --notification --image=/usr/share/icons/synaptic_ok.png --text=0 updates available; 0 new packages available.
[agmg@localhost ~]$
```

The **-n** argument will select only the newest (most recently started) of the matching processes (in the example is used in conjunction with the **-l** argument):



```
[agmg@localhost ~]$ pgrep -n kde
3503
[agmg@localhost ~]$ pgrep -n -l kde
3503 polkit-kde-auth
[agmg@localhost ~]$
```

The **-o** argument will select only the oldest (least recently started) of the matching processes (again, in the example, it is used in conjunction with the **-l** argument): next page, top left



```
agmg : bash - Konsole
[agmg@localhost ~]$ pgrep -l kde
12 kdevtmpfs
2122 startkde
2906 kdeinit4
3006 kded4
3503 polkit-kde-auth
[agmg@localhost ~]$
```

More information on **pgrep** and its options can be found in its manual page: <http://www.lehman.cuny.edu/cgi-bin/man-cgi?pgrep+1>

pskill, killall, kill

All of the above commands can be used to send any signal to a process, each one in a different way, but are mostly used to terminate (send TERM signal - SIGTERM - Value 15) or forcibly kill a stubborn process (send KILL signal - SIGKILL - Value 9). The TERM signal is the default signal sent by these commands to the specified process.

For more information about Signals in Linux, please visit this page: <http://man7.org/linux/man-pages/man7/signal.7.html>

Using pskill

pskill can kill a process using its name. For example if you want to kill the Firefox process, simply type **pskill firefox**

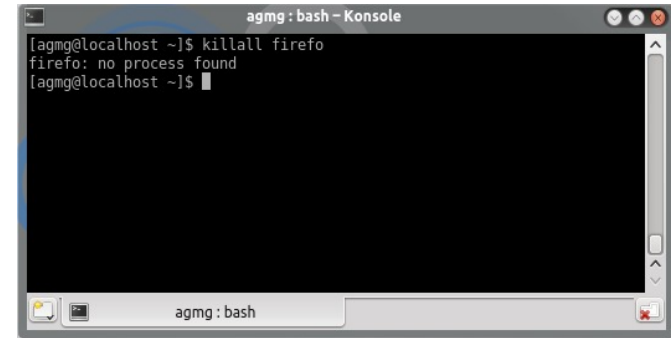
Note that pskill will kill every process that includes “firefox” in its name. You can view which processes they are, by using the **pgrep -l firefox** command described previously.

The default signal that pskill sends to a process is the TERM (terminate) signal. You can change that by issuing the command with the **-signal** argument. The signal may be specified either as a name (with or without a leading SIG), or numerically. For example, if you want to forcibly kill Firefox, you can type

pskill -9 firefox or
pskill -SIGKILL firefox or
pskill -KILL firefox

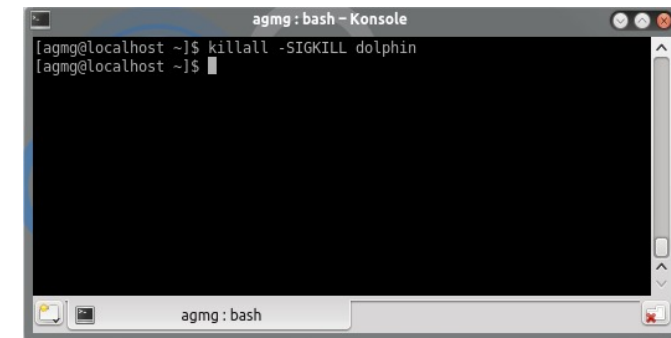
Using killall

killall works in similar way with pskill but by default it requires the precise process name. So, if you type **killall firefo**, you’ll see the following error:



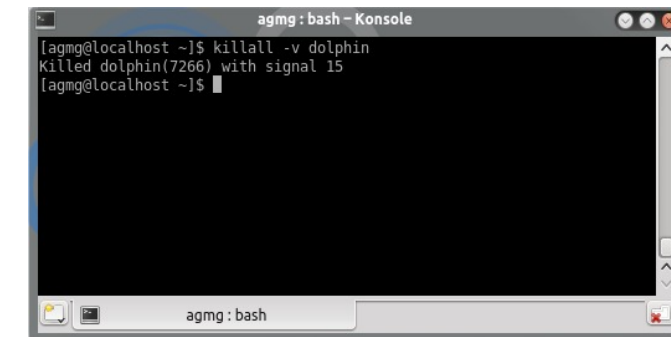
```
agmg : bash - Konsole
[agmg@localhost ~]$ killall firefo
firefo: no process found
[agmg@localhost ~]$
```

killall also sends the TERM signal by default. You can change that by using the **-s** argument, which (like with pskill) can be either the name of the signal (with or without a leading SIG) or its value:



```
agmg : bash - Konsole
[agmg@localhost ~]$ killall -SIGKILL dolphin
[agmg@localhost ~]$
```

Using the **-v** argument will instruct **killall** to report if the signal has been successfully sent:

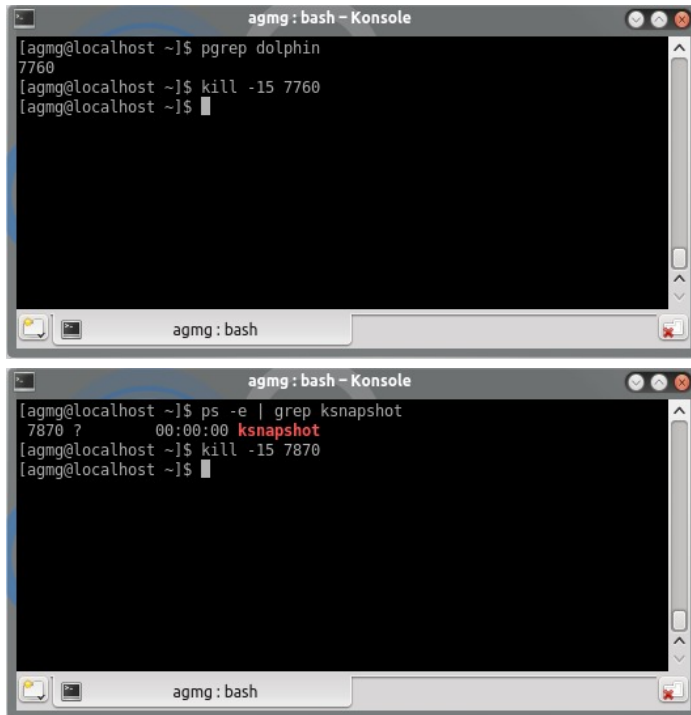


```
agmg : bash - Konsole
[agmg@localhost ~]$ killall -v dolphin
Killed dolphin(7266) with signal 15
[agmg@localhost ~]$
```

More information on **killall** and its options can be found in its manual page:
<http://linux.die.net/man/1/killall>

Using kill

kill sends a signal to a process using its PID. Again, the default signal sent is TERM but you can change it like with **pkill** or **kill** (using the **-s** argument). It's the least convenient way because you have to know the process ID before issuing the command. You can use any of the **top**, **htop**, **ps** or **pgrep** commands described above to find out the PID of the process you want.



```
agmg : bash - Konsole
[agmg@localhost ~]$ pgrep dolphin
7760
[agmg@localhost ~]$ kill -15 7760
[agmg@localhost ~]$

agmg : bash - Konsole
[agmg@localhost ~]$ ps -e | grep ksnapshot
7870 ?        00:00:00 ksnapshot
[agmg@localhost ~]$ kill -15 7870
[agmg@localhost ~]$
```

Again, if you want more knowledge about **kill**, don't hesitate to read its manual page:
<http://unixhelp.ed.ac.uk/CGI/man-cgi?kill>

nice & renice

The above commands are both used to change the priority of a process. But what is process priority?

When talking about processes, priority is all about managing processor time. Instead of technical stuff, let me give you an example. Let's say you're surfing the internet while listening to your favorite mp3 files. What a great time to combine all the videos you took on your last vacation to a larger one, using your favorite video editor. At some point, while surfing, listening to your music and encoding your videos, you are starting to experience slowdowns in your browser or small pauses on the mp3 playback. This happens because the third task you have assigned to your computer, the video encoding, is taking too much of your system resources, putting your other tasks in the back burner. If you don't mind your video to finish a couple of hours later, you can tell your processor to give less attention to this task and more to your surfing and listening to music (this is just a fictional example but I hope you get the picture).

In Linux we can set guidelines for the CPU to follow when it is looking at all the tasks it has to do. These guidelines are called **niceness** or **nice** value. The Linux niceness scale goes from -20 to 19. The lower the number the more priority that task gets. If the niceness value is a high number like 19 the task will be set to the lowest priority and the CPU will process it whenever it gets a chance. The default nice value is zero.

By using this scale we can allocate our CPU resources more appropriately. Lower priority programs that are not important can be set to a higher nice value, while high priority programs like daemons and services can be set to receive more of the CPU's focus. You can even give a specific user a lower nice value for all of his/her processes so you can limit their ability to slow down the computer's core services.

The **nice** command is used to run a command with an adjusted scheduling priority. On the other hand, the **renice** command is used to change the priority of an already running process. The process will use the modified priority for as long as it runs. It will return to its default priority once it's shut down and started again.

The syntax of the nice command is: **nice [OPTION] [COMMAND [ARG]...]**

-n, --adjustment=ADJUST increment priority by ADJUST first. ADJUST is 10 by default. Range goes from -20 (highest priority) to 19 (lowest).

--help display this help and exit

--version output version information and exit

Example:

Running Firefox with lower priority: **nice -n 5 firefox**

Running Firefox with higher priority: **nice -n -6 firefox**

The syntax of the renice command is:

renice *priority* **[[*-p*] pid ...] [[*-g*] pgrp ...] [[*-u*] user ...]**

renice -n increment **[[*-p*] pid ...] [[*-g*] pgrp ...] [[*-u*] user ...]**

The **renice** command alters the scheduling priority of one or more running processes. The following *who* parameters are interpreted as process ID's, process group ID's, user ID's or user names. The **renice**'ing of a process group causes all processes in the process group to have their scheduling priority altered. The **renice**'ing of a user causes all processes owned by the user to have their scheduling priority altered. By default, the processes to be affected are specified by their process ID's.

The following options are available:

-g Force *who* parameters to be interpreted as process group ID's.

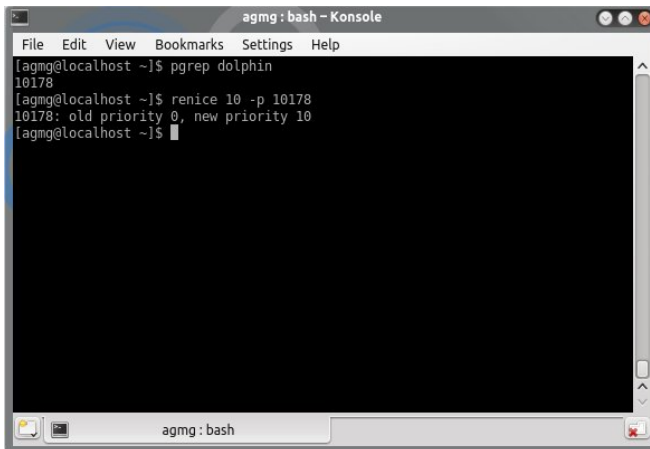
-n Instead of changing the specified processes to the given priority, interpret the following argument as an increment to be applied to the current priority of each process.

-u Force the *who* parameters to be interpreted as user names or user ID's.

-p Reset the *who* interpretation to be (the default) process ID's.

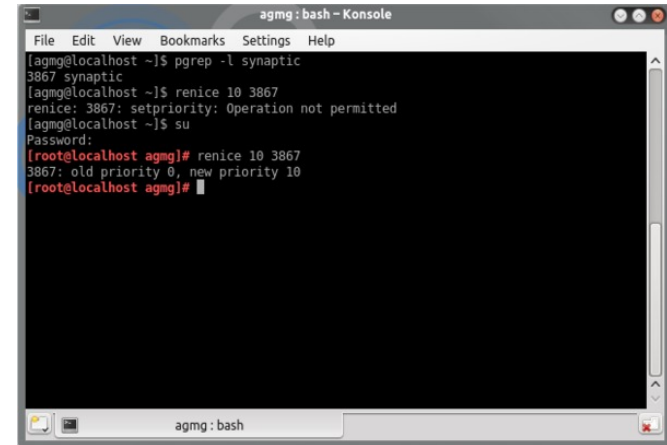
Some examples:

Let's say we want to lower the priority of Dolphin file manager. First we have to find out its process ID (I've used **pgrep** in my examples):



```
agmg: bash - Konsole
File Edit View Bookmarks Settings Help
[agmg@localhost ~]$ pgrep dolphin
10178
[agmg@localhost ~]$ renice 10 -p 10178
10178: old priority 0, new priority 10
[agmg@localhost ~]$
```

If you try to **raise** the priority of a certain process that is not owned by you, you will get a “operation not permitted” message. Only a superuser can raise any process priority. Users other than the super-user may only alter the priority of processes they own, and can only increase their “nice value” within the range 0 to PRIO_MAX (20).



```
agmg: bash - Konsole
File Edit View Bookmarks Settings Help
[agmg@localhost ~]$ pgrep -l synaptic
3867 synaptic
[agmg@localhost ~]$ renice 10 3867
renice: 3867: setpriority: Operation not permitted
[agmg@localhost ~]$ su
Password:
[roote@localhost agmg]# renice 10 3867
3867: old priority 0, new priority 10
[roote@localhost agmg]#
```

For more information on these two commands you can consult the manual pages.

Manual page for nice: type **man nice** in terminal or go to: <http://www.manpagez.com/man/1/nice/>

Manual page for renice can be found here: <http://www.manpagez.com/man/8/renice/>

xkill

xkill is a command for forcing the X server to close connections to clients. This program is very dangerous, but is useful for aborting programs that have displayed undesired windows on a user's screen. **xkill** is not installed by default in PCLinuxOS. To install it, open Synaptic Package Manager and search for **xkill**.

If the **xkill** is issued without a command line argument, it will display a special cursor (a skull) with which you should select a window to be killed. Click on the application which you want to abort forcefully or select the X window/application whose client you wish to kill with the left mouse button. If a pointer button is pressed over a non-root window, the server will close its connection to the client that created the window. If you change your mind, you can click the right mouse button to abort.

Managing Running Processes From Terminal

You can use **xkill** with the following command line arguments:

-display *displayname* This option specifies the name of the X server to contact.

-id *resource* This option specifies the X identifier for the resource whose creator is to be aborted. If no resource is specified, xkill will display a special cursor with which you should select a window to be kill.

-button *number* This option specifies the number of pointer button that should be used in selecting a window to kill. If the word "any" is specified, any button on the pointer may be used. By default, the first button in the pointer map (which is usually the leftmost button) is used.

-all This option indicates that all clients with top-level windows on the screen should be killed. Xkill will ask you to select the root window with each of the currently defined buttons to give you several chances to abort. Use of this option is highly discouraged.

-frame This option indicates that xkill should ignore the standard conventions for finding top-level client windows (which are typically nested inside a window manager window), and simply believe that you want to kill direct children of the root.

```
agmg: bash - Konsole
[agmg@localhost ~]$ xwininfo
xwininfo: Please select the window about which you
would like information by clicking the
mouse in that window.
xwininfo: Window id: 0x340001d "agmg: xwininfo - Konsole"

Absolute upper-left X: 393
Absolute upper-left Y: 181
Relative upper-left X: 0
Relative upper-left Y: 0
Width: 561
Height: 538
Depth: 32
Visual: 0xa5
Visual Class: TrueColor
Border width: 0
Class: InputOutput
Colormap: 0x3400001 (not installed)
Bit Gravity State: NorthWestGravity
Window Gravity State: NorthWestGravity
Backing Store State: NotUseful
Save Under State: no
Map State: IsViewable
Override Redirect State: no
Corners: +393+181 -412+181 -412-49 +393-49
-geometry 561x538+385-41

[agmg@localhost ~]$
```

An easy way to find out the id of a window is by running the **xwininfo** command. The **xwininfo** utility is not installed by default, but you can install it easily through Synaptic Package Manager.

When you type **xwininfo** in terminal, the mouse cursor will change to a cross, allowing you to select the window that you want to get information on.

<-----

The **Window ID** is the information we need. The command to kill the desired window, would therefore be:

xkill -id 0x340001d

You can find the **xkill** manual page either by typing **man xkill** in terminal or by visiting the following page: <http://www.manpagez.com/man/1/xkill/>



Cloud Computing & The Third World

by Pieter Louw

We live in a wonderful world today, where connectivity aided by advanced technology makes life so much easier than just the day before. Cape Town is less than half a day away from London by air, compared to the same voyage three centuries ago that may have taken three to four months! It is exciting to see the development in communications and how it turns the world into a better place.

Information technology lifted the veil that had clouded much of the world beyond the horizon in mystery. The technological advancement of gadgets sees an incremental growth and, unlike a short decade ago, mobile phones are being used for a host of different functions. Only about ten per cent of its use nowadays involves making phone calls!



signal coverage by any specific supplier in a given area, users sometimes may need more than one operator. For instance, a user may need to lug around a Vodacom device when visiting a rural area,

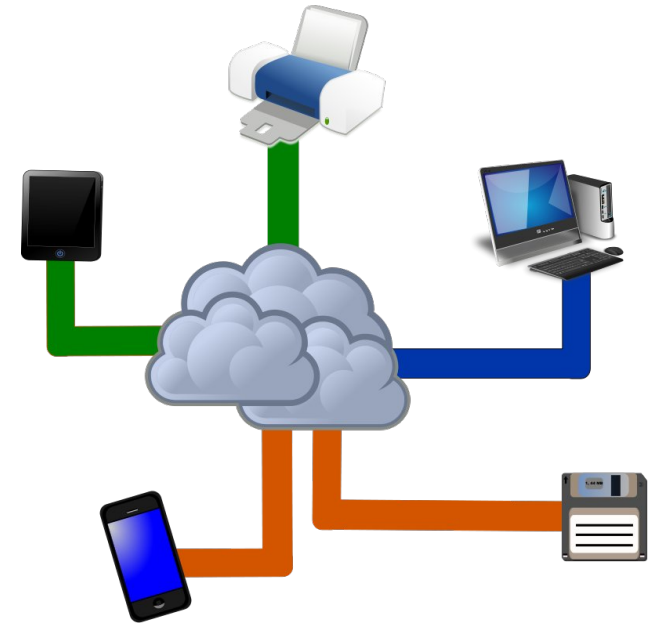
Africa is the only continent where ownership of cellular phones outstrips that of fixed lines. It is not uncommon to see even very poor individuals carrying two or more mobile phones. This is so because some need to have access to GSM networks all the time but, due to random availability of

MTN for the CBD and 8ta for fast HSDPA coverage in between. Dual-SIM devices such as the excellent Samsung Duos sees its sales being suppressed by mobile operators in South Africa, which is sad. If it is not approved by ICASA, it may even be a criminal offence to have it in your possession!

Smartphones and computers are being designed and built to connect via the cloud nowadays, by people living in developed countries who never may have known a world sans wireless broadband altogether. They assume that everybody on earth has no need for USB connectivity. Bluetooth, to them, may have no reason to exist, etc. They are used to steering their course through life with the automated ease of appliances and devices that just never require someone to stop for opening a gate, or having to draw a pile of water, or having to snare something for dinner. They harvest at the mall and remain connected wherever they are.

They then proceed to create devices and systems that will also be used in developing countries, where billions of people still have no potable water or flushing toilets. In my country, millions live in corrugated iron shacks and ride bicycles or travel on horseback. We are the most advanced country on the African continent. Yet, even in the legislative capital with some four million residents, it is quite easy to find a spot where not even Mxit or Whatsapp will function.

Apple and others seemingly try to force us through the cloud, something that does not exist in these countries, mostly. Android devices also have their difficulties connecting to a PC to sync, and all and sundry are lobbying for Nokia PC Suite-alike functionality for all mobile phones. That should be part of all computer operating systems by default. That is the ideal.



Applications such as Mobisynapse and Airdroid assist users of MS Windows in connecting directly via USB or Bluetooth, but exclude Linux users. The name Airdroid is descriptive and I am not blind to that. It is a free app which we do appreciate for what it is. I am also not ungrateful for a free app that works well.

Finding devices that can use it independently in developing countries is a real challenge. Because of the insane cost of MS Windows, honest folks like myself use Linux, but **38% of Windows software installed in my country are pirated copies.** Windows XP SP3 as well as Windows 7 Ultimate are even sold with used computers on Gumtree, much of it illegitimately. Just look at the many ads on Gumtree.co.za that offer used laptops sporting Windows 7 Ultimate – and I will be surprised if even ONE of these is legitimate!

Simply because mobile phone manufacturers disrespect Linux users, they leave cash-strapped users with no option but to use MS Windows. As this operating system is prohibitively expensive, this gives rise to the software being pirated en masse.

Finding an integrated system on a limited budget in a developing country with erratic and insanely expensive internet connectivity becomes a quest at the very least. Phone and PC can't link to sync, there is little or no cloud and, even where present, most people simply cannot afford it.

First world countries make laws that effectively exclude their third world trade partners from transacting with them. Third world countries simply cannot afford the infrastructure to provide for its impoverished users. It means that the first world countries are fencing themselves in and also alienating themselves from markets that must buy their products.

The first world seems to be ignorant of how their decisions affect others.

There is a real need out there for an Airdroid-like application that will also be a "cable-droid" via USB, or at the very least, via Bluetooth. It should ideally support at least Linux, as the latter is being used widely by those not having access to higher specification computer systems.

It could be offered as a feature-rich paid application with a thinware free equivalent, with limited functionality. In meeting the ICT needs of developing countries, the first world will be exposing itself to a few billion more humans patronising them economically.

People here still spend hours in internet cafés every day, as there is neither land-line nor cellular connectivity where they live. They travel relatively long distances to even get to such outlets. They do

so, as they need to use the internet for educational purposes as well as for commercial ends.

If we get these people to connect with others in a meaningful way, the economy can grow and new markets will be opening up. Manufacturers in developed countries will have better access to the raw materials they need and will also enjoy better exposure for their products to new consumer markets.



USB or Bluetooth connectivity is an absolute must, and I sincerely hope that someone will act upon this information and get going. I call upon Linux developers to consider this when developing the next distro or release – make a combination of USB, Bluetooth and WLAN connectivity part of your systems architecture, especially now that we see Linux on computers, mobile phones, tablets and TVs. Third world users really require this functionality to become economically active, something that will also affect the economies of first world countries positively.

The following factors impact heavily upon technology users and, in forcing them through the cloud, will alienate them from both the rest of the world and their own bouquet of devices. I will

paraphrase to explain the above-mentioned problem hopefully with more clarity down below.

SIGNAL AVAILABILITY AND "CLOUD COVERAGE."

- Large areas have no telephony either because of sparse population or the **lack of infrastructure**.
- As **copper theft is a major problem**, fixed lines to farms, rural villages and mining or constructions sites have all but disappeared.
- Demographics means that the **cost** of setting up GSM infrastructure can never be recouped.
- **Electricity supply** or the lack thereof does not allow for erecting more towers.

Thousands of square kilometres of farmland and mining sites are being left with no communications infrastructure, except for bad dirt roads. Mobile phone users sometimes need to climb or drive up a hill or mountain to send and receive email. If a diamond mine has an email address, it doesn't automatically mean that its owners and staff can browse the web! Personally, I know a number of diamond miners who have to undertake a fortnightly trip back to civilisation to execute email and internet tasks as well as to sync their phones with their laptops! The situation becomes insane where there is a real need for using desktop computers.

Perhaps vendors should be listening more to their clients, instead of forcing people into the cloud and out of options by doing so. Cellular phone makers, such as Nokia, are missing out hugely upon sales of new devices in a burgeoning market simply because they don't port Nokia Suite to Linux, for instance. Instead, they recently had to retrench ten thousand staff, people who may still have had an income, had Nokia listened to my voice and that of others!

It is disappointing to see how end users become voiceless because of mobile phone manufacturers, systems developers and other role players who a) disregard requests from clients and b) become inaccessible by virtue of automated, or at least impersonal, help desks.

If we want to see the world progress in a meaningful way, we need to start using the technology at hand to truly communicate. XYZ's support people send an automated response every time I lodge a request, but they never follow up on this with me. XYZ Forum is an unfriendly place where one has to fit into a rigid, frigid matrix or be reprimanded. Is this how they want to attract users?

It is time for technology providers to mature into being effective communicators. Bullying end users into submission while forcing irrelevant technology upon them is rather unwise.



Screenshot Showcase



Posted by agmg, April 1, 2013, running KDE.



A magazine just isn't a magazine without articles to fill the pages.

If you have article ideas, or if you would like to contribute articles to the PCLinuxOS Magazine, send an email to:
pclinuxos.mag@gmail.com

We are interested in general articles about Linux, and (of course), articles specific to PCLinuxOS.

SLIM Desktop Manager: A Lightweight Alternative

by Peter Kelly (critter)

What is it?

The simple login manager is a display manager designed to be both light on resources and simple to use and configure, and it accomplishes both of these goals admirably. This makes it ideal for lighter window managers and desktop environments. But, since it is desktop independent, it may be used with almost any installation.

Why on earth would you bother to change a login application?

Well, the login screen is the first screen that you see after booting up. Therefore, it should look the way you want it to. Although all of the login managers provided as default with the PCLinuxOS releases are configurable, it is sometimes not an easy matter to get them to behave as you would like. The SLIM login manager can help change that.

What can it do?

On first installation, it looks rather underwhelming. There is a box to type in a user name and then a password and nothing else. This is unlike the more robust login managers such as KDM, the KDE specific login manager, which has buttons to perform various functions, a clock, a logo and possibly other ornaments. However, don't be put off by the simple, uncluttered appearance, as slim has a few hidden tricks that make it a real competitor. Add to this the simplicity of configuration and we have a real winner for those who like to have things set up just how **they** like it.

Does it have any drawbacks? Well, maybe one small one that some power users might find irksome. It apparently does not support remote login capabilities. For most users, though, this is unlikely to present a problem.



This is a snapshot of the default login screen. As you can see, it is pretty basic and it would seem that the only option you have here is to type in a user name.

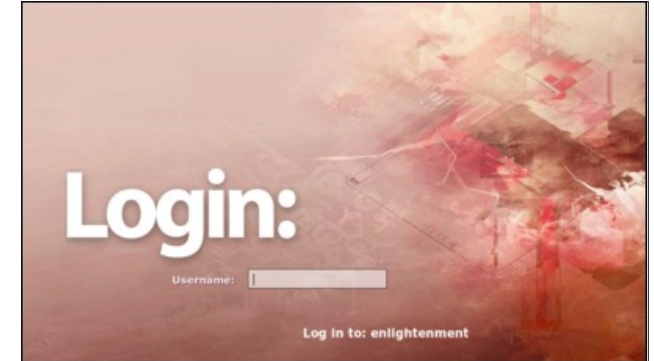
Other graphical login managers offer the option to perform actions such as shutting down or restarting the system. This login manager offers similar options, but in a slightly different manner. There are some special user names that you can use to login that perform these actions for you. Login as "halt", enter the root password and the system will shut down.

Other 'special user' names are:

1. reboot
2. suspend
3. console – to launch a terminal

4. exit – to the shell

No buttons, no clutter. Just the ability to control the system.



Press the F1 key repeatedly and you will be prompted to login to a series of different desktop environments or window managers. Enter a valid username and password, and if that desktop is available, you will be logged into that desktop. If not available, you will be returned to the login screen.

No buttons, no clutter. Just the ability to access the desktop that you **want**.

Installation

To install SLIM, open synaptic, press Reload, Mark All Upgrades, (this is necessary to make sure that everything is compatible), then click Apply (to install the selected packages). Afterwards, click Search. In the search box type slim and in the drop-down "look in" box select name. Enter slim and perform the search. You should get two results, slim and slim-themes. Install them both.

Note: At the time of this writing, SLIM is not available in the 64 bit version, which has only just been officially released. For the impatient, more info about SLIM, including some documentation and themes, can be found [here](#).

Configuration

Configuring SLIM is a matter of editing a text file `/etc/slim.conf`.

Since this file affects **all** users of the system, you will need root privileges to perform the edits. You may have your own way of opening an editor as root, perhaps Dolphin's right click "Edit as root" option.

I do it this way. Open a terminal and type `su -`. Press Enter. Enter the root password at the prompt, then type:

```
cp /etc/slim.conf /etc/slim.conf.bak.
```

This will create a backup of the file in case things don't go to plan. Next, type `nano /etc/slim.conf` to open the configuration file in nano editor.

This file has a lot of comments to help you set things up. Although this is only a text file, it can look daunting. So I have reproduced it below without those comments, and have added line numbers so that I can take you through it line by line. It may look like a lot, but few changes are necessary for most users.

```
1. default_path
   ./:/bin:/usr/bin:/usr/sbin:/sbin

2. default_xserver      /usr/bin/X

3. xserver_arguments    -dpi 75 -br
   -nolisten tcp

4. halt_cmd             /sbin/shutdown -h now

5. reboot_cmd          /sbin/shutdown -r now
```

```
6. console_cmd          /usr/bin/xterm -C
   -fg white -bg black +sb -T "Console
login" -e /bin/sh -c "/bin/cat /etc/issue;
exec          /bin/login"

7. #suspend_cmd         /usr/sbin/suspend

8. #hibernate_cmd       /usr/sbin/hibernate

9. xauth_path           /usr/bin/xauth

10. authfile            /var/run/slim.auth

11. numlock             on

12. hidecursor         false

13. login_cmd           if [ -r ~/.xinitrc
   ]; then exec /bin/bash -login
   ~/.xinitrc %session; else
   PREFFERED=%session exec /bin/bash -login
   /etc/X11/xinit/xinitrc; fi

14. daemon              yes

15. sessions
   lxde,openbox,fluxbox,enlightenment,afterste
p,xfce4,icewm,wmaker,blackbox,kde,gnome

16. screenshot_cmd      import -window root
   /slim.png

17. welcome_msg         Welcome to %host

18. session_msg         Log in to:

19. shutdown_msg       The system is
   shutting down...

20. reboot_msg          The system is
   rebooting...

21. default_user        jane

22. focus_password      yes

23. #auto_login         no

24. current_theme       default

25. lockfile            /var/run/slim.lock
```

```
26. logfile            /var/log/slim.log
```

Most of these lines you can leave as they are, but this is what they mean.

1. This is where SLIM will look for the utilities it needs to perform its functions. Most people can ignore this.

2. Where to find the application that actually does the graphical interface. As different distributions put things in different places, this may occasionally need to be changed. Fortunately, we don't, so we can ignore it.

3. All but senior power users can ignore this one.

4 to 8. These are the commands that you want executed when you want to, for example, shutdown the system. Those with a "#" are inactive. You may want to put the system into a sleep state and find that the suspend command does not work, but pm-suspend does. This is where to change it.

9 & 10. Leave as they are unless you **really** know what you are doing.

11. This may be useful. If you find that your password is not being accepted (and you can't see what you typed because only asterisks are echoed to the screen) then it could be that the numlock key is active. Change to off.

12. Hide the mouse cursor. This may or may not work.

13. This is the command that actually gets you to your selected desktop. Leave it as it is. If you find a good reason to change it then please let me know.

14. You want to say yes here.

15. This is a list of the available desktops to which you can log in. We shall be changing this.

16. If you have the ImageMagick utility installed, and I recommend that you do, then pressing the F11 key while at the slim screen will put a screenshot of that screen in the root (/) directory for you to include when sharing your breathtaking new theme online.

17. The basic message that you want to see while being prompted to log in.

18. The message displayed when pressing the F1 key to select a desktop to which you want to log in.

19 & 20. Should be self explanatory.

21. The default user name to display. Leave blank to start with an empty user name field.

22. Yes means to put the cursor in the user box, ready for typing.

23. If you want to be automatically logged in with no password, remove the # and change to yes. This is a security risk, so beware.

24. The name of the current theme to display.

25 & 26. Leave these as they are.

I think that only lines 15 & 24 need to be edited by most users.

As an example: if you installed the lxde version of PCLinuxOS, then added the xfce and enlightenment desktop environments, (use synaptic to install task-enlightenment and task-xfce), you would want to edit this to read:

lxde, enlightenment, xfce

Pressing the F1 key would then cycle through these three options.

Themes, by default, are stored in /usr/share/slim/themes, and each has their own

directory. Once you are happy with your new theme, that is where to place it.

The second part of the configuration is only required if you do want to switch between desktop environments.

Create a file named

.xinitrc

in your home directory.

Add the following (for our example)

```
DEFAULT_SESSION=startlxde
case $1 in
lxde)          exec startlxde      ;;
xfce4)         exec startxfce4     ;;
enlightenment) exec enlightenment_start ;;
*)             exec $DEFAULT_SESSION ;;
esac
```

The first line tells SLIM which desktop to launch if you haven't pressed F1 and selected a particular desktop, otherwise it launches the chosen desktop. Lines 3, 4 & 5 execute the relevant command to launch the chosen desktop, which must be one of those listed in line 15 of the configuration file. That's it.

Usage

After installing slim you need to tell the system to use it. Open the PCLinuxOS Control Center, select Boot, Set up Display Manager and you should see a screen like this.

Select SLiM and say yes when prompted to restart the desktop manager.

You will be logged out and then presented with a screen like the first screenshot.



You may now log in as usual, press F11 to create a screenshot, press F1 to cycle through the available desktop choices, or type in one of the previously mentioned 'special' user names to shutdown, reboot etc.

Themes

Creating your own themes and previewing them is a breeze and one of the best reasons for choosing this login manager.

A theme consists of a directory with the name of the theme. In the directory are three files.

1. A background image named background.jpg or background.png.
2. A panel image named panel.jpg or panel.png.
3. A text file named slim.theme.

As an example we shall say that our default desktop is LXDE and that we want a login screen that provides a consistent look through the boot process. To make life easier I am going to "borrow" some graphics and a theme script and modify them.

Create a directory in your home directory named `pclos-lxde` (for convenience).

Open a file manager and navigate to `/usr/share/slim/themes/rear-window`.

Copy the files `panel.png` and `slim.theme` to your `pclos-lxde` directory.

Navigate to `/usr/share/plymouth/themes/` and copy the file `background.png` to our theme directory.

We have a new theme! To view it type:

```
slim -p /home/user/pclos-lxde
```

Replace user with your login name.

The `-p` option allows you to preview the theme. The preview is fully functional but will not re-log you in. If you press the F1 key you will be shown the available options. You can even go through the motions of typing in one of the special user names such as `halt` followed by the root password to see what would be shown on the actual login screen. This is a nice touch that makes creating themes so much easier.



Well, it worked, but it's not quite what I wanted.

The header is the hostname of this PC, which has not been set.

The panel should be central and higher.

The background color of the input boxes looks out of place.

Open `panel.png` in your favourite graphics editor and change the color of the input fields to white.

Open the file `slim.theme` in an editor.

The first two lines credit the author of this file. We should leave those in place and add some of our own so that it looks like this.

```
# pclos-lxde theme for SLiM
# based on
# rear-window theme for SLiM
# slackhack@linuxmail.org
```

Make the following additional changes:

change `input_panel_x` from: 12% to: 50% this will centralise the panel.

change `input_panel_y` from: 88% to: 75% to raise the panel a little.

change `%host login:` to: `Login:`

change `username_color` from: `#e3d6aa` to: `#7b9ed2`

change `username_font` from: `Verdana:size=14:bold` to: `Verdana:size=18:bold`

change `username_shadow_color` from: `#333333` to: `#0d1e3a`

Save the file.

Of course, these changes are only my preferences and you may have to adjust them to suit your taste and your screen resolution. But this should demonstrate that the values are quite intuitive. I load the background into a graphics editor and use the color picker tool to extract the color values.

Preview the theme again.

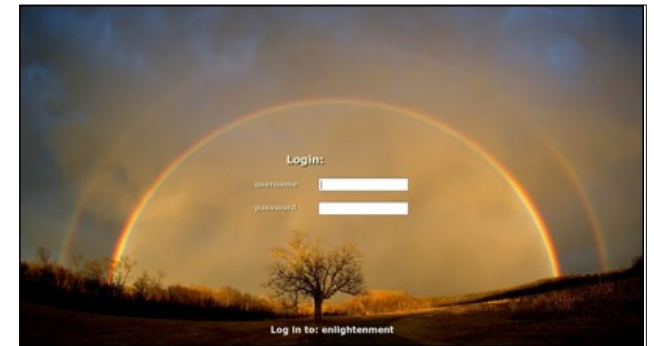


That looks better. Now all we have to do is copy the entire theme folder to where SLIM will look for it and to tell SLIM to use this theme.

Edit the file `/etc/slim.conf` (as root).

Change the line that reads current theme default (line 24 in my listing above) to current theme `pclos-lxde`.

Copy the folder and contents to `/usr/share/slim/themes`, and we are done.



You can now have your own personalised login screen limited only by your imagination.



Support PCLinuxOS! Get Your Official

PCLinuxOS
Merchandise Today!

PCLinuxOS

International Community PCLinuxOS Sites



Disclaimer

1. All the contents of The PCLinuxOS Magazine are only for general information and/or use. Such contents do not constitute advice and should not be relied upon in making (or refraining from making) any decision. Any specific advice or replies to queries in any part of the magazine is/are the person opinion of such experts/consultants/persons and are not subscribed to by The PCLinuxOS Magazine.
2. The information in The PCLinuxOS Magazine is provided on an "AS IS" basis, and all warranties, expressed or implied of any kind, regarding any matter pertaining to any information, advice or replies are disclaimed and excluded.
3. The PCLinuxOS Magazine and its associates shall not be liable, at any time, for damages (including, but not limited to, without limitation, damages of any kind) arising in contract, tort or otherwise, from the use of or inability to use the magazine, or any of its contents, or from any action taken (or refrained from being taken) as a result of using the magazine or any such contents or for any failure of performance, error, omission, interruption, deletion, defect, delay in operation or transmission, computer virus, communications line failure, theft or destruction or unauthorized access to, alteration of, or use of information contained on the magazine.
4. No representations, warranties or guarantees whatsoever are made as to the accuracy, adequacy, reliability, completeness, suitability, or applicability of the information to a particular situation. All trademarks are the property of their respective owners.
5. Certain links on the magazine lead to resources located on servers maintained by third parties over whom The PCLinuxOS Magazine has no control or connection, business or otherwise. These sites are external to The PCLinuxOS Magazine and by visiting these, you are doing so of your own accord and assume all responsibility and liability for such action.

Material Submitted by Users

A majority of sections in the magazine contain materials submitted by users. The PCLinuxOS Magazine accepts no responsibility for the content, accuracy, conformity to applicable laws of such material.

Entire Agreement

These terms constitute the entire agreement between the parties with respect to the subject matter hereof and supersedes and replaces all prior or contemporaneous understandings or agreements, written or oral, regarding such subject matter.



More Screenshot Showcase



TOP LEFT: Posted by Crow, April 8, 2013, running LXDE. **TOP RIGHT:** Posted by jogurtmen, April 2, 2013, running KDE. **BOTTOM LEFT:** Posted by RobNJ, April 4, 2013, running KDE. **BOTTOM RIGHT:** Posted by Roc4fun, April 11, 2013, running KDE.