The PCLinuxOS magazine

Volume 126 July, 2017

SUMMERTIME FUN

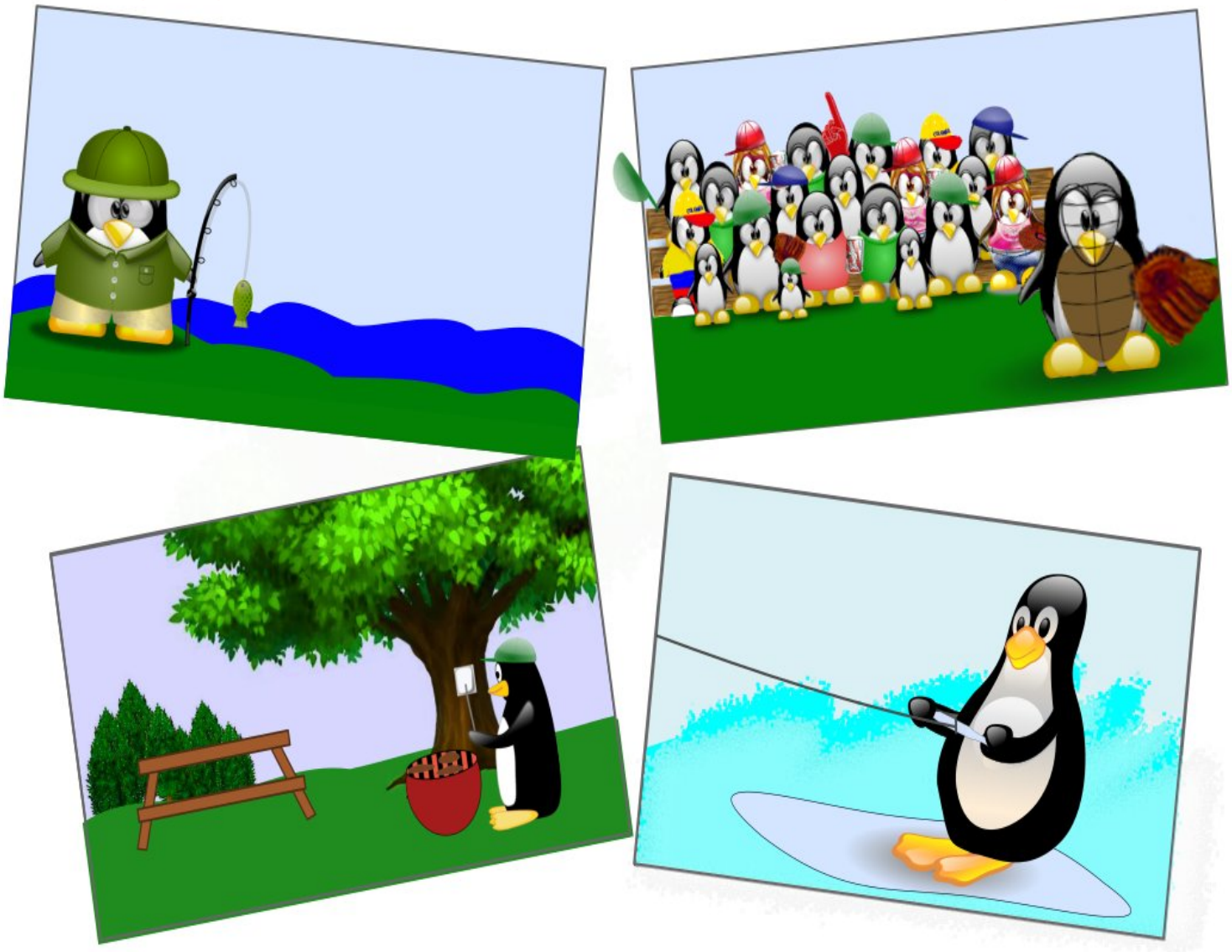


Table Of Contents

The PCLinuxOS magazine

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: ms_meme, Meemaw
Magazine Layout: Paul Arnote, Meemaw, ms_meme
HTML Layout: YouCanToo

Staff:
ms_meme
Meemaw
Gary L. Ratliff, Sr.
Daniel Meiß-Wilhelm
daiashi
loudog
YouCanToo
Pete Kelly
Antonis Komis
Smileeb

Contributors:
Khadis
Agent Smith
CgBoy
phorneker

From The Chief Editor's Desk ...

As hard as it is to believe, the July issue marks the start of my ninth (yes, 9th) year at the helm of the magazine! As I look back, it's a remarkable journey to reminisce about all the changes I've witnessed. With the magazine, we've experienced a LOT of growth. With PCLinuxOS, we've seen each release become more and more stable. With Linux in general, we've seen greater and greater hardware compatibility, coupled with a greater market penetration.

It had been said in the past that there just wasn't enough to write about to maintain a monthly magazine dedicated to PCLinuxOS. I think we've dispelled that errant thought a LONG time ago. We haven't missed our monthly publication schedule since I started, and even come out with several "special editions" of the magazine, as well.

We've added regular columns (our graphics tutorials, PCLinuxOS Recipe Corner, PCLinuxOS Friends & Family Members, PCLinuxOS Puzzled Partitions), and at the same time, witnessed some of our earlier monthly columns come to an end (DoubleTake & Mark's GIMP Tip, ms_meme's Forum Foibles). Some columns, such as Repo Review, have experienced a rebirth. Thankfully, ms_meme's Nook continues as it has ever since the first issue when I took over the magazine back in July 2009 (and we hope she NEVER stops!).

Above: *Mom, Ryan and Dad on opening night for Cars 3 – and Ryan's first ever trip to a theater.*

Right: *Dad and Lexi riding in a 1923 passenger rail car at Day Out With Thomas.*

June has been a pretty busy month at our house. Ryan and Lexi got to ride on a full size reproduction of Thomas The Tank Engine at A Day Out With Thomas. Then, they got to go to the Kansas City Zoo for a day. Then, my best friend and I spent two days putting together a wooden backyard playset (boxes and boxes of precut wood, screws, bolts, nuts, and accessories to be assembled). Ryan and Lexi absultely love the playset, and have spent a lot of time on it. Then, Ryan got to experience his first ever trip to a movie theater to see "Cars 3" on its opening night (good movie, by the way). Then, on Thursday nights during the summer, the theater on the town square shows free family movies outside in the parking lot, with the movie projected onto the side of the building. So the kids got to take in a movie (Despicable Me 2) in the outdoors, which finished just before a thunderstorm rolled through. In the mix, throw in some swimming in the small backyard pool (inflatable variety), visits with Grandpa, visits with Grandma, plus their regular daily routine, and you have a pretty eventful month.

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

Weak Password? Five Ways To Generate Strong Passwords

by Paul Arnote (parnote)

We've written about passwords and security several times within the pages of this magazine. We've shown you ways to generate secure passwords. We've talked about how a secure password can protect your data, both online and on your physical computer.

Yet, all too often, and despite all that they have heard and learned about password security, users still lean towards employing the weakest of passwords. Much of the time, users tend to use the same password on most (if not all) of the sites that they regularly sign into. Sure, it's easier to remember one password (complex or simple) than a dedicated password for each site.

Enigma Machine at the Imperial War Museum, London. (Public Domain)

We're not going to rehash our previous articles here on password security, and why it's important. If you didn't listen the first time, chances are good that you won't listen the second, third, tenth or hundredth time, either. At least, not until you become a victim of an insecure password. Users who take password security seriously are often viewed as paranoid, if nothing else. At least, until you become a victim of an insecure password.

Everything we've written previously about choosing and using a secure password, still applies. Everything we've written previously about using a "base password" that is modified for each site you log into, still applies. Everything we've written previously about the foolishness of reusing passwords, still applies. Everything we've written previously about using a password manager (such as KeePassX or LastPass), still applies.

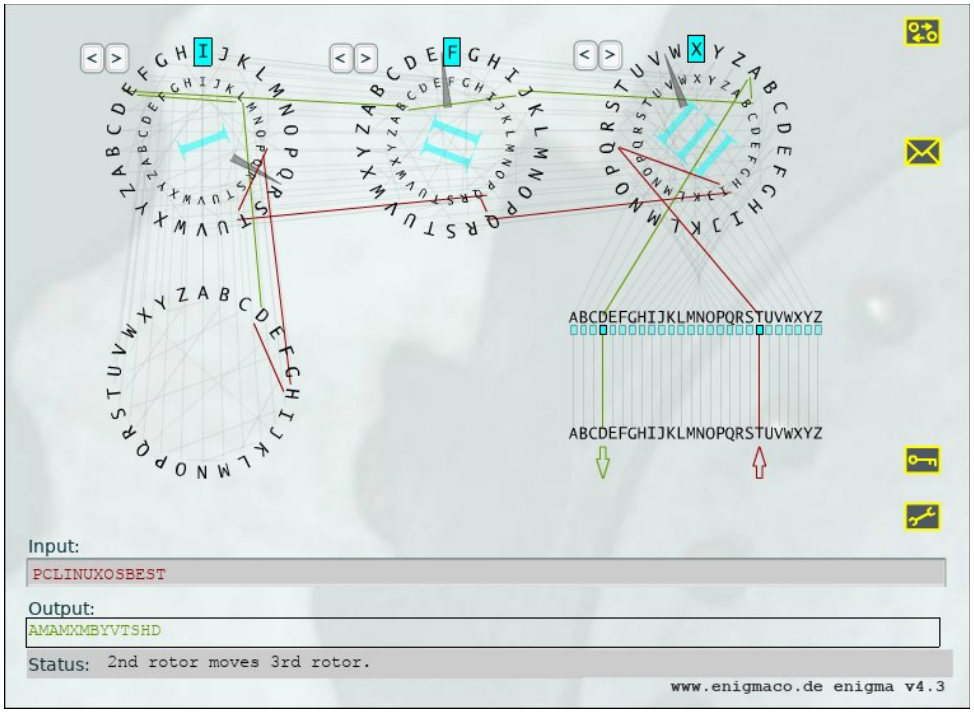
Rather, this article will showcase some tools you can use to help generate unique and secure passwords. How you use them is entirely up to you. Remember that longer, more complex passwords offer more security than passwords that are short and made up of everyday words. Use of numbers, symbols and punctuation increase security.

Lastly, you should take the time to test your password (or one similar to it with the same length and similar complexity, if you're truly paranoid) on a site like "How Secure Is My Password?". It will serve as confirmation that you have given due diligence to creating secure passwords, but keep in mind that it is anything but a guarantee of security. The best you can do is slow down a hacker to such a point that they will skip over your data and move on to less secure data that is easier pickings.

Enigma

If that small, six letter word conjures up visions of the losing side of World War II, you would be on the right path. Its place in the cryptologic Hall of Fame is guaranteed for all time. Given how much time, man hours and energy were placed into breaking the Enigma encryption, why not consider using it to create a secure password? The process is/was solid. What eventually led to the breaking of the Enigma code was its sloppy implementation by its users. Their sloppiness and laziness contributed to the codebreakers at Bletchley Park and elsewhere to more rapidly breaking the code. You can read more about the Enigma cipher machine here.

Today, a real, working Enigma machine is hard to come by. But thanks to an online simulator, we can all get a pretty accurate feel for how Enigma worked. In the image on the next page, we put in the phrase PCLINUXOSBEST, and get AMAMXMBYVTSHD back as the encoded output. Granted, the output isn't necessarily the easiest to remember, but if you're using a password manager, you won't necessarily have to remember it (but I wouldn't put all of my eggs in the password manager egg basket … I'd have a backup copy saved somewhere safe).



For what it's worth, because of the pseudo-randomness of Enigma and how it works, you will most likely get a completely different output than I got in the example above.

OpenSSL

This program is included in the base installation of most versions of PCLinuxOS. If you find it missing, install it from Synaptic. Don't shy away from it just because it is a command line program. It's quite easy to use.

openssl rand 14 -base64

```
[parnote-toshiba@localhost ~]$ openssl rand 14 -base64
ej/dWm7dM9i28mzvhlA=
```

The **rand** parameter evokes the pseudo-random generator, while **14** tells it how many bytes of data to create. The **-base64** option insures that the result can be typed from a keyboard.

IT professionals typically cite a minimum password length of 14 characters to be ideal, since a password of that length – utilizing uppercase and lowercase letters, numbers, symbols, and punctuation – will require enormous computing power to crack via brute force methods. This is why we use the 14 in the command above. Of course, you can use any number you want.

Looking at the output, you will realize that we have produced 19 characters. You can use the output as is, or you can truncate or edit the result to give you just 14 characters. The choice is entirely up to you. Just keep in mind that longer, more complex passwords are typically stronger, harder to crack, and thus more secure.

pwgen

This program is installable on PCLinuxOS, via Synaptic. It's job is to produce secure passwords that can be easily remembered by humans. Again, this is a command line program, but don't let that deter you or scare you off. It, too, is very easy to use.

pwgen 14 1

```
[parnote-toshiba@localhost ~]$ pwgen 14 1
xa6tooYo8Sae9Z
```

The **14** specifies how many characters long to make the password. The **1** tells pwgen to make only one password. Change the 1 to any other number and pwgen will create that many passwords.

```
[parnote-toshiba@localhost ~]$ pwgen 14
phaiFuf8ponaif tai1pia4XahRee reHieM9laicoob tu9Loosh3sheeh Nieveequoh0Eid
Que8oochaiy6xu Ohz4aeneimieZi Eeku3asoh1aoPh taim3Hain5aiW0 maese9nohwoPah
eT1zaunahNgeit Beix0jeacohta3 oDoo3eepahrie6 oos4aeRoohohDa sohyaiWaethee6
iwoDievei4Ohlu keecieGhoqu1bo yaej1xeiw5aaCi eoQuaise6weVon Eech2ISai1Quei
ahphei0iib2Uax Vuk8fao5eehohb Sahtho1ohzee7a Ai2ohqueum4ahb ohjahLui6uowou
Phae9neinien6s ooN7hee9lucu9p ahchohgode1ioR pieV4deeJ8ohpu ara6Iedaegish1
ahCahpahquee4v maefoo9Aith0ae aF1Aelee1choos yah6Ijooch5quo Ij7fooGhohghai
see9wukae6eiZa bohn5aiHooph5E keisohW4BaT9Ke ooJung9Ahwoeli xaeHu1Ao9aeWoo
oozuthei0Rieda Eevovu1UzahQuu eki5soY6ahngoh pooheenga4rahV queeN1of7eephi
quohLa1Ukeek9u oiyah2eerei4Ah Ziad2Teigau1nu vuaqu5Ahm8roof lahSh6nail2xeh
eiz0BaiXi3eazu iaZo1aebeel0ie eit5rohx8Ve9zo oaWaibe3iequai iew2Mohxaiheng
air6Oe2ahro3oo Acuk6au6ahZeil xagohjieQui7Am aet1chieQu3chi Hiengungoo3pho
sooCheiri6Mooj EejaeJ4veening chee4Aic5Iepoo niegich1Uo4rie TuideeC8AhchiX
aequoaheuv5Ne7 Liyieju6orie0a waNg1ohPhai0Xa kei9guph6euS3u oiVigohph2id4c
vai8aiquuThaih ohrei6Chud0quu uNie1phie8Eelo Kui4rah6ailai9 ohng9Er5phuudi
eecaJeik0veRai leidi0gik0ae1G phaiTheiv7zoh6 Hai5oa7afaer8i yooW0ofai5kaiz
yeoveeNge4si4B quaeghohSura4e faaph6eShoxexo hooThiis6heewo Ujah1Coh2kah5u
tohtoosae8Iepa ohcos7sirieluB KaibooFebai7th ohDeiwoviz5aix yeePhoosheeSa3
Oomees0xoh4sho Eek3shaigiungo Pheoxahtohzoe2 ohC9ahshiefae8 ooTh4thia8ohQu
Fah0kaing6keeb aoleePh5thi9eu mahr9ANgai2phu ooBucaipaet9zo oof9reiG6uaT2e
```

If you omit the last number telling pwgen how many passwords to generate, it will default to providing you with 100 passwords to choose from, all 14 characters long.

Of course, if you want longer or shorter passwords, simply replace the 14 in the command with the length of characters you want for your password. You will notice that each password generated by pwgen is exactly the number of characters you specify.

gpg, or Gnu PG, or Gnu Privacy Guard

Whichever name you prefer to use, they all relate back to the very same program. Gpg is typically thought of to secure the contents of files and email, something it excels at. But you can also leverage gpg to create secure passwords. Again, don't shy away from using this powerful command line program. Gpg is installed by default in most PCLinuxOS installations. If you find it missing, install it from Synaptic.

```
gpg --gen-random --armor 1 14
```

```
[parnote-toshiba@localhost Scripts]$ gpg --gen-random --armor 1 14
YUwpbs6buSh0WRjtnTU=
```

Similar to openssl, even though we specified 14 characters (or bytes) as the length of the password to generate, gpg provided us with a 20 character password.

Use Perl

Perl should be installed by default on your copy of PCLinuxOS. If it is not, install it from Synaptic. If you're not familiar with Perl, it is a programming language that is a lot like bash scripting, with a twist of C.

Enter the following into your favorite text editor, and save it somewhere in your /home directory as password.pl.

```
#!/usr/bin/perl

my @alphanumeric = ('a'..'z', 'A'..'Z', 0..9);
my $randpassword = join '', map $alphanumeric[rand
@alphanumeric], 0..13;
print "$randpassword\n"
```

Then, move to the directory where you saved the script, and enter the following:

```
perl password.pl
```

```
[parnote-toshiba@localhost Scripts]$ perl password.pl
hy0DvxYpcuuxt1
```

You should see something like what's depicted in the screen capture above, on your screen.

Notice the "0..13;" at the end of the third line of the script. This is where we set the length of the password that is generated. With this script as presented, it produces a 14 character random password. But, if you want a 20 character password to be generated, change the "13" at the end of the line to "19." Because Perl uses a zero based counting system (that is, it starts counting at zero, instead of one), always subtract one from the number of characters you want. You can change this number to any number you like, so you can also make shorter passwords (although I can't imagine why you'd want to do that, from a security standpoint).

Summary

So, there you have it. Five ways to generate unique, secure passwords under Linux. You owe it to yourself to follow the established recommendations of not only using a secure password, but to also avoid reuse of passwords between sites. Data, or rather access to data, has become the currency of criminals around the world. Once they have your data, life can get pretty difficult and troublesome. Keep your data – and yourself, ultimately – safe from those who would like to get access to it, but who have no rightful purpose to access your data.

ms_meme's Nook: We're Going To PCLOS

We're going to PCLOS PCLOS dot com
We're going to PCLOS PCLOS here we come
Tex has a crazy little system we're gonna get us one

We're gonna be standing in the forum waiting online
We're gonna be standing in the forum waiting online
Tex has a crazy little system what a great design

Tex what a brain let me explain
About his OS we never complain
We're going to PCLOS PCLOS dot com
He's got a crazy little system it's next to none

We're going to PCLOS PCLOS dot com
We're going to PCLOS PCLOS here we come
Tex has a crazy little system it gets the job done

Tex what a brain I'm telling you again
With his OS we will remain
We're going to PCLOS PCLOS dot com
Tex has a crazy little system we're gonna get us one

MP3 ***OGG***

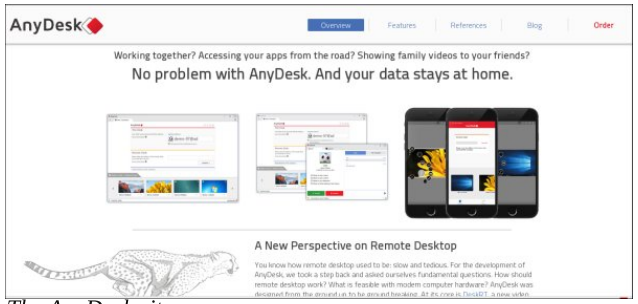
AnyDesk: Better Than Teamviewer?

by Alessandro Ebersol (Agent Smith)

AnyDesk is a remote desktop software developed by AnyDesk Software GmbH in Stuttgart, Germany. It provides two-way remote access between PCs and is available for all common operating systems. The software has been in active development since 2012 and surpassed 20 million downloads in 2017.

The AnyDesk Software GmbH was founded in 2014 by former employees of TeamViewer.
The interesting bit: for private use, AnyDesk is free.

AnyDesk uses DeskRT to provide a considerably better image quality and responsiveness than competing screensharing and remote desktop products. DeskRT is a new and innovative video codec specifically designed for the transmission of image material from graphical user interfaces.



The AnyDesk site

Features

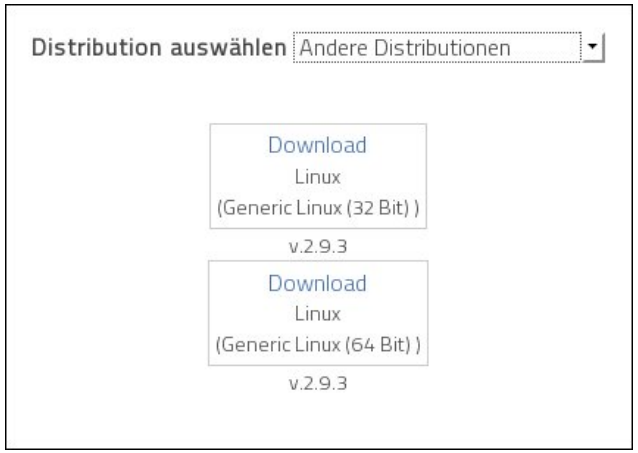
Features include, among others:

* Bidirectional remote control between Windows, macOS, Linux and FreeBSD
* Unidirectional access from the mobile platforms Android and iOS
* Secure TLS-1.2 Protocol
* File transfer
* Client to client chat
* Clipboard integration
* REST API
* Session Log
* Custom client alias

How to make it run in PCLinuxOS?

The program is provided in a small executable file, packaged in various formats (RPM, DEB and tar.gz).

To download for PCLinuxOS, go https://anydesk.com/download
and choose Generic Linux tar.gz, as shown below

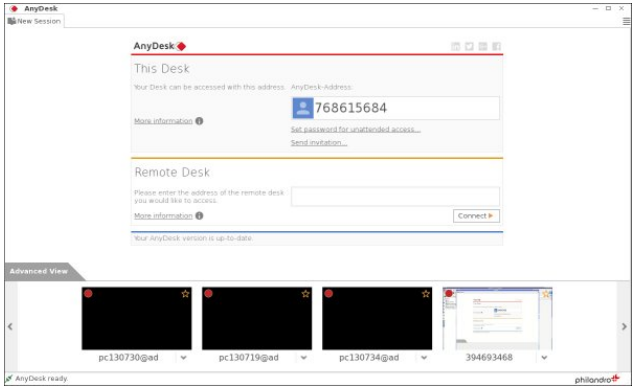


Then, it will download a package of 2.9 MB.

Extract the package, and an anydesk-2.9.3 folder will be created. Inside the folder are the files: anydesk, changelog, copyright, icons/ and README.

Now, give executable permission to the anydesk file and it is ready, the program will run without any problems. Note that it is a portable program that can run smoothly in your /home folder.

The AnyDesk client depends on the following library to run: **lib64gtkglext-1.0_0**, therefore it must be installed to allow it to run.



The program in action

The first time you run the anydesk program, it will create a unique number for the computer you are running. This number will be the machine ID for AnyDesk, and will be the means to connect to this machine.

AnyDesk can be configured for unattended access, configuring a password for it, which will be requested when there is an attempt to connect from

another computer (or mobile) running another copy of AnyDesk.

Limitations

Logon/XDM
At the moment the service is started only as per-user service. So it is not yet possible to connect to the login screen. You can however, if a user is already logged in, connect to the lock screen and log back in.

File Transfer/Clipboard
You can copy files using the clipboard. When transferring files to Linux/BSD, an icon will show up in the status bar. Doubleclick that icon to start the actual transfer. After the files will be copied, the destination folder will automatically open so you can move the files where you like.

Block Input
The "block users input" feature is not supported on Linux/BSD.

Hotkeys
Hotkeys are not supported.

Install AnyDesk in PCLinuxOS

As the program comes in a portable self executable file, you can place the folder in the /home folder of the computer(s) that you want to control remotely and make a ".desktop" file and place this file in the autostart folder, allowing the machine to run AnyDesk in the log on.

How to make it? Simple, follow the steps below:

1. Download the package AnyDesk 64 bits, as shown above.

2. Extract the package. The resulting file will create a folder anydesk-2.9.3.

3. Give execute permission to the executable anydesk.

4. Create the file (with a text editor of your choice) anydesk.desktop with the content listed below:

[Desktop Entry]
Name=AnyDesk
Comment=remote control application
Exec=/home/user/anydesk-2.9.3/anydesk
Icon=/home/user/anydesk-2.9.3/icons/48x48/apps/anydesk.png
Terminal=false
Type=application
Categories=Network;
Encoding=UTF-8

Place this file in ~/.config/autostart, give execute permission to it and it's ready: every time the computer goes on logon routine, the AnyDesk program will run. Remembering always that user is the user's home name where the folder is located.

The anydesk-2.9.3 folder can be protected against accidental deletion by changing its attribute to immutable (chattr + i), so that users will not be able to delete it.

So, is it better than Teamviewer?

Well, I did not see differences between AnyDesk and Teamviewer in terms of performance within a network, the AnyDesk company has more versions available for most platforms (BSD, for example) and can be used without the five minutes limitation of Teamviewer, which is very good. There is the limitation that AnyDesk can not be run as a daemon, such as TeamViewer can, but Teamviewer is a kludge to run in Linux (an EXE windows application that runs with Wine), while AnyDesk is a native application.

Apart from these small differences, and the limitations mentioned above, to remotely support non-profit organizations (work that I do often) without the annoyances of product purchase screens for Teamviewer, and Teamviewer's five minutes time limit, it's very good. Approved!

Just be careful how you use it. Some "tech support" scammers and other malicious individuals have used this (or similar software) to gain access to unsuspecting individuals' computers. Use some discretion before just handing over the keys to your computer's desktop. Be sure it is someone you trust and know well.

Game Zone: Day Of Infamy

by Stephen Morrish (PendragonUK)

Introduction

Day of Infamy is a World War 2 first person shooter game. Fast paced, hard core shooter where death comes quick. Play style is up to you, but most shots kill with a single hit. You may wish to keep your head down. Players that work together and communicate will achieve more than gung-ho lonewolf. The game is more realistic than an arcade shooter, but no so much as to make it hard to get into.

Features:

* Gritty. objective-based experience
* Squad-based role system
* Strategic leadership and artillery strikes
* Local and radio communications
* Workshop support

Who Makes It

New World Interactive is an independent developer and publisher. Many core team members started as hobbyists creating the games they wanted to play. Their first game Insurgency started as a mod and was brought to Steam as an independant title. Day of Infamy also started as a mod. Revisioning the modern combat of Insurgency to the WWII theatre and time.

About the Game

Day of Infamy takes you to the battlegrounds of Southern and Western Europe during World War II. There are ten maps, ranging from war-torn cities to farm villages, fortified beachheads and snow-covered forests. Whether you are fighting for the U.S. Army, Commonwealth Forces or the German Wehrmacht, there is a vast arsenal of over 70 historical weapons and attachments. Nine player classes and ten objective-based game modes. Cooperative gameplay features three distinct game modes against challenging AI enemy troops. The game comes with mod support and Workshop integration, providing players with a vast array of community created content. There are already over 500 mods which have been released for the game.

Pros

While the game is easy to get into and hard to master, Day of Infamy is a deeply rewarding team game. Get it right, and you are on top of the world. Get it wrong, and you'll be face down in the mud!

Cons

Some will complain that death comes too quick. That you are killed before you know where the shot has come from. (Basically, that the game is too hard.)

Conclusion:

The game play is just what some players of shooter games have been calling for. A serious shooter that does not hold your hand. No graphical hud letting you how many bullets in your weapon. No over powered bullet sponge super soldier common to many military games. You can run ‘n gun but your

aim must be on point and your head on a swivel. Team play is rewarded, the more cautious player will live long enough to win for the team.

System requirements

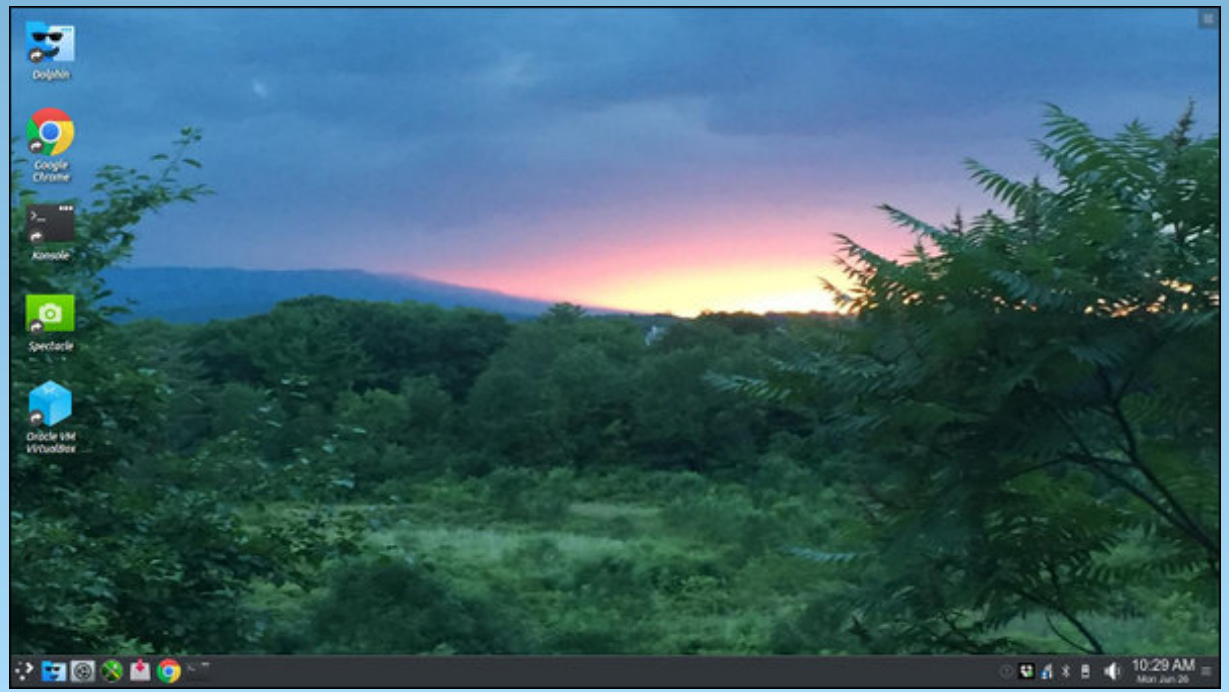
Minimum
Processor: Dual core from Intel® or AMD at 2.8 GHz
Memory: 4 GB RAM
Graphics: NVIDIA GeForce 8600/9600GT, ATI/AMD Radeon HD2600/3600 (Graphic Drivers: nVidia 310, AMD 12.11), OpenGL 2.1
Network: Broadband Internet connection
Storage: 12 GB available space
Sound Card: OpenAL Compatible Sound Card

Recommended
Processor: 4+ cores
Memory: 8 GB RAM
Graphics: 1536MB VRAM or more
Network: Broadband Internet connection
Storage: 15 GB available space
Additional Notes: Solid state hard drive

Where to get it

You can get this game on Steam. Currently, it will cost you $19.99 USD, £14.99 GBP, or €19.99 EU. The game is often discounted in sales, so if you time it right, you could have a real deal.

Screenshot Showcase



Posted by Texstar on June 25, 2017, running KDE.

PCLinuxOS Recipe Corner

Muffin-Tin Barbecue Bacon Meatloaves

Ingredients

1 lb extra-lean (at least 90%) ground beef
1/2 cup chopped cooked bacon
1/2 cup Original Bisquick™ mix *
6 green onions, thinly sliced, whites and greens separated
1 egg
1 teaspoon barbecue seasoning
1/4 cup barbecue sauce
3 slices (1 oz each) sharp Cheddar cheese, quartered

Directions

1. Heat oven to 450°F. Spray 12 regular-size muffin cups with cooking spray.

2. In large bowl, stir beef, bacon, Bisquick™ mix, green onion whites, egg and barbecue seasoning until well mixed. Divide mixture among muffin cups, pressing evenly into cups.

3. Bake 14 to 17 minutes or until meat thermometer inserted in center of loaves reads 160°F. Brush loaves with barbecue sauce. Top with cheese. Bake 1 to 2 minutes longer or until cheese just melts. Top with green onion greens.

Notes:

* Make your own homemade Bisquick mix using the recipe here: http://recipes.dm-enterprises.net/?p=recipe&recipe=289

Introduction To The Lumina Desktop

by phorneker

PCLinuxOS comes with MATE and KDE Plasma desktops on official releases. But, did you know there is a new desktop environment that can outperform both MATE and KDE Plasma?

The Lumina Desktop is a desktop that is fully featured and is focused on getting things done efficiently.

Lumina Desktop is the default desktop for TrueOS (http://www.trueos.org), a user friendly variant of the FreeBSD distribution. (One could argue that TrueOS is to FreeBSD what PCLinuxOS is to Linux itself.) It has been ported to various Linux distributions (including PCLinuxOS), and was built on the QT5 toolkit and the Fluxbox window manager.

Installing Lumina

Lumina is installable from Synaptic. But once installed, you should reboot PCLinuxOS to ensure that Lumina will launch as intended, and that Lumina is selectable from the Session menu in the login screen.

When you launch Lumina for the first time, Lumina will take time to locate applications already installed in PCLinuxOS.

Each time you launch Lumina, a famous quote will appear on the center of the screen, followed by a startup sound (a series of chimes). Then the desktop will appear.

This is the desktop you see the first time you launch a Lumina session. It looks like there is not much to this desktop, but there is a lot you can do with it. I did mention that Lumina was intended for getting things done efficiently. It starts with an uncluttered desktop.

The icon on the upper left hand of the screen was brought over from the KDE Plasma desktop. It is the Plasma desktop entry found in the **Desktop** folder in your home directory. Other icons will appear as Lumina searches the Desktop folder for new icons each time you launch Lumina.

The other element is the RSS feed from lumina-desktop.org (the official website for the Lumina Desktop).

Like KDE Plasma, just about everything here can be customized, including the layout of the desktop, not unlike the Full Monty desktop (recently discontinued), and the panel itself.

Here, the main menu is on the lower left hand corner of the screen. (This can be changed later.) To the right is the network applet (for Wi-Fi or wired connection), a clock and a battery status monitor.

At the top center of the screen is a pull down panel (not shown on the screen unless you look very closely at the top center of your physical display) where you can choose from favorite folders and applications. What appears here depends on what you use most often, or what folders and applications you declare to be your favorite.

As Lumina runs on top of Fluxbox, window decorations from Fluxbox are used for display of windows in this desktop.

Clicking on the menu button (the lower left corner of the screen by default) pulls up a translucent menu with everything we can expect from a main menu (and then some).

From the top down, we get a search box which works the same as the search box in KDE Plasma, underneath that is a battery status icon. Simply hover the mouse over that to get the battery status in a tooltip box.

Below that we get the standard list of applications most users will run under Lumina, namely Firefox, SMPlayer and VLC (provided these applications are installed in PCLinuxOS in the first place), followed by links to Downloads, Documents, Pictures and Videos. These four folders open using Insight file manager.

What you see here is only a portion of what is actually available in the menu. There are links that allow you to browse applications in the menu (for a more complete list), and browse files with the Insight file manager.

Below that is a Preferences link that brings up ways to customize the Lumina session.

Finally, there is a Leave button that allows you to exit the Lumina session. Clicking on the Lock button will activate a screensaver and lock the session. Simply move the mouse, then type your user password to resume the Lumina session.

The Leave button has three options, one simply logs you out of Lumina to the PCLinuxOS login screen, one is for rebooting of PCLinuxOS, and the third shuts down the system.

Click on **Browse Applications** to get the applications menu. Here, we still have a search box for locating applications on our PCLinuxOS installation.

Between the application list and the search box is a tri-state checkbox, a feature not found in KDE Plasma, MATE, or any other desktop downloadable from Synaptic.

Clicking on this checkbox cycles between three states, namely **off**, **partially on** (indicated by the left half triangle displayed inside the checkbox) and **on**.

Click on this checkbox to set it to **on**. Here, we get a list of categories which correspond to the menu categories found in the KDE Plasma and Mate desktops. Here the menu functionality corresponds to that of the KDE Plasma and MATE menus, that is, select the category, then select the application on the next menu that appears.

Click again on this checkbox to set it to **off**, we get a alphabetically sorted listing of all applications available in our PCLinuxOS installation. This is useful only if you do not have many applications installed on PCLinuxOS.

Now, click again to set this check box to **partially on**. Here, we get the list of applications, but each list is grouped according to the category the application is found in the KDE Plasma and MATE menus. This setting provides a unique view of your applications, as it includes an **unsorted** category for applications that are installed, but not found on the KDE Desktop and Mate menus.

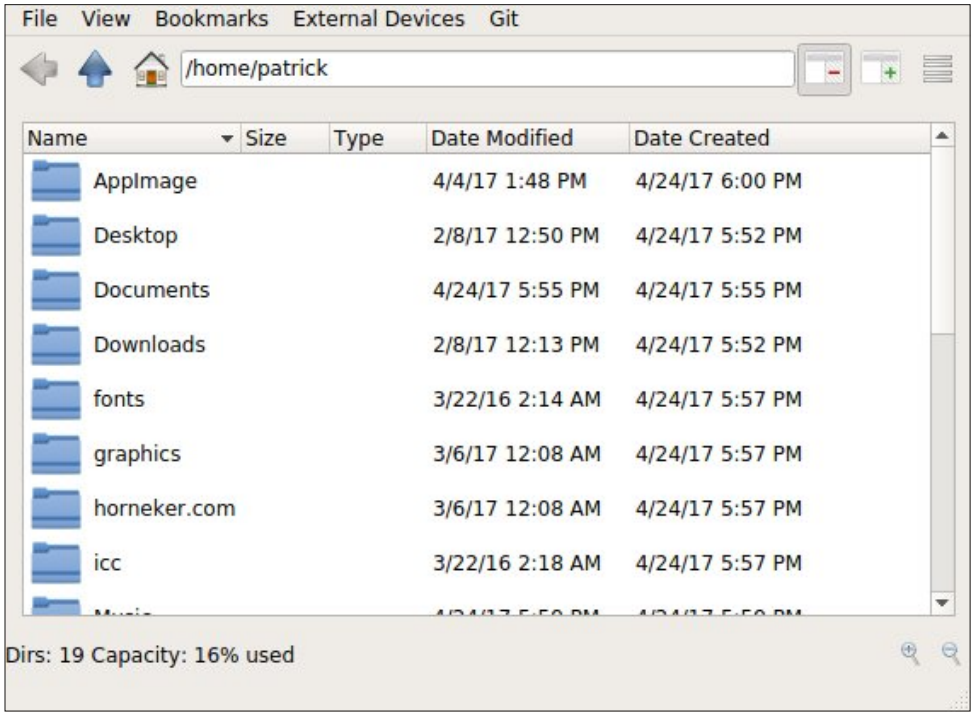
Finally, there is a **Back** button, that, as expected, takes you back to the main menu.

Clicking on **Preferences** gets you three options. The first option launches the **Configure Desktop** utility, which is where the customization of Lumina happens.

The second item in this menu is a volume control. This works similar to volume control widgets on KDE Plasma, Mate and XFCE in the sense that you can hover the mouse over the control and either drag the volume control or use the mouse wheel to control the volume. (On my laptop, there is a touch bar with a volume control slider where I simply slide on the touch bar in the volume control area. This also works with the volume control applet.)

The third item in the menu is used to switch between virtual desktops (called Workspaces, not unlike what is found in the WindowMaker window manager, of which I have written a series of articles on that desktop.)

...and of course, we have the **Back** button, which takes up back to the main menu.



Clicking on **Browse Files** gets you into the Insight file manager. If you have an account with Git, you can connect a Git repository to this file manager and share files with other Git users.

Right clicking on the desktop gets you a menu where you can open a command line terminal, open the Insight file manager, run applications, set preferences, and finally exit from the Lumina session.

Among preferences you can set from this menu is the screensaver. Lumina uses the **xscreensaver** application, which is loaded with screensavers some of which were initially developed in the late 1980s and are still being maintained.

\If you have the **fortune-mod** package installed, you can configure the screensaver to display some great nostalgia from the early days of computing

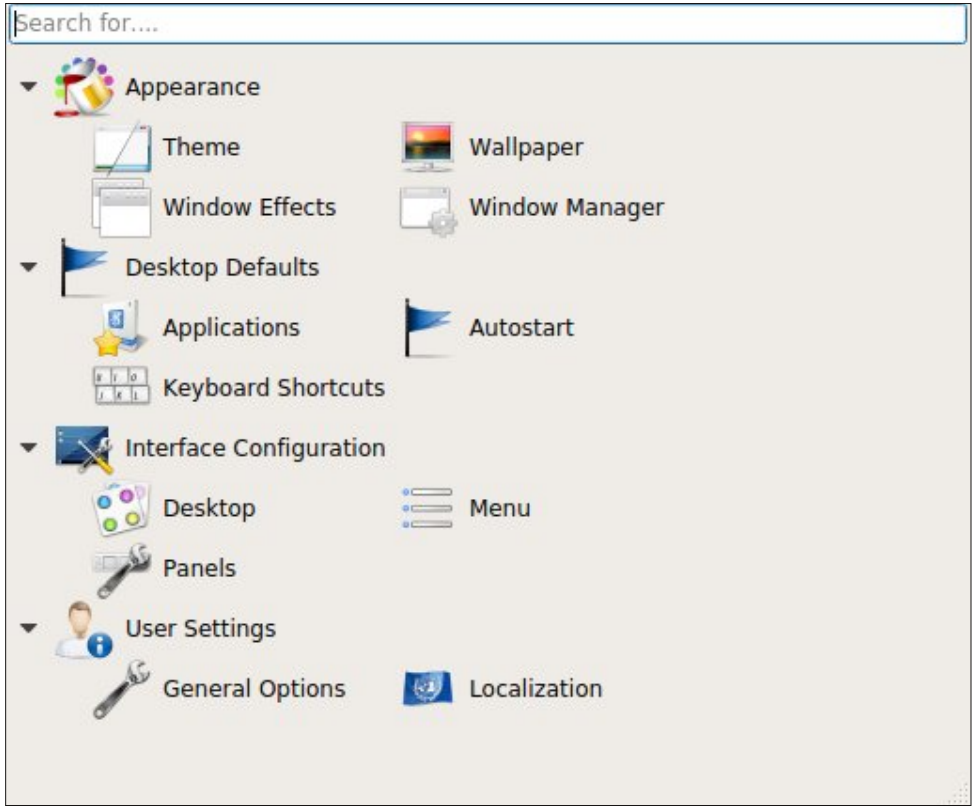
and the Internet, in addition to some famous quotes, song parodies, science fiction, movies, humor and other trivia. This is accomplished by selecting either the **noseguy** or **phosphor** screensavers.

If you scroll the list of available screensavers here, you will find a selection of screensavers that cannot be found in KDE Plasma (or Windows or MacOS-X for that matter). For Lumina, the default screensaver displays a maze and then traverses that maze.

If you have pictures you want to display in a screensaver, you can configure the screensaver to use those as well.

What we have seen is just the introduction to what should be a new desktop experience.

In the **Preferences** menu, there is a **Desktop Settings** utility we can launch to configure this desktop to do more.



The **Interface Configuration → Desktop** button allows you to set the screen resolution(s) and display arrangement (if you have multiple displays) not unlike the similar utility in KDE Plasma.

The **Interface Configuration → Panels** button allows you to select the overall layout for the Lumina display, or customize individual panels within the desktop. There are profiles containing presets that allow your Lumina display to emulate Windows (or KDE), Mac OS-X, MATE, or XFCE desktops. You can also start with a blank slate where you can create and configure panels.

There are a lot more options that can be configured through **Desktop Settings**.

I did say that Lumina was designed to get things done efficiently. You can select applications you use often to be placed on the desktop ready to run.

Applications in the menu can be placed on the desktop by right clicking on the application item and selecting **Pin to Desktop** from the popup menu.

That way, you simply double click on icons representing your often used applications to launch them. These icons can be rearranged on the desktop for efficiency.

We have covered the basics of the Lumina desktop. As we use this desktop, we will be customizing the desktop environment to fulfill the promise the developers of Lumina intended, that is, to make work more efficient.

Lumina Note: TrueOS comes with the AppCafe, a utility that is part of the official TrueOS release, and was designed for the Lumina desktop. AppCafe makes easy work of installing FreeBSD software by placing its repository (as well as the FreeBSD repository) in a AppStore like interface. Of course, the traditional methods of installing FreeBSD applications will still work on TrueOS.

The AppCafe is not available in **any** Linux port of Lumina.

PCLinuxOS

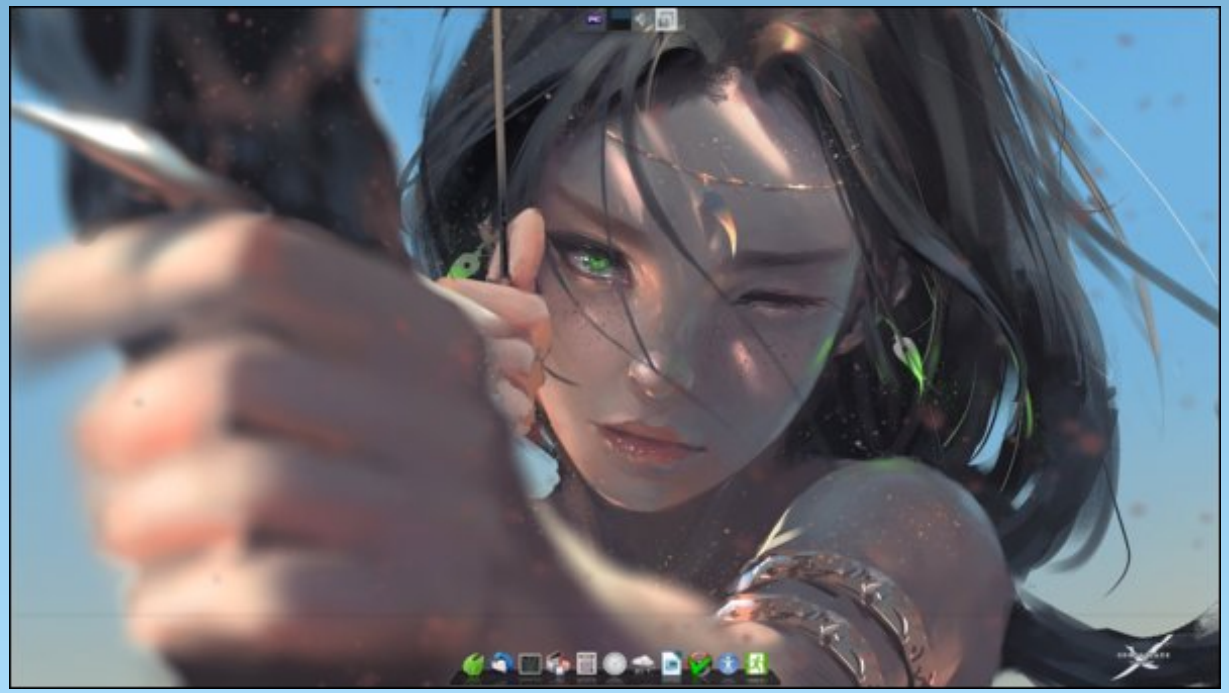
Users Don't

Text
Phone
Web Surf
Facebook
Tweet
Instagram
Video
Take Pictures
Email
Chat

While Driving.

Put Down Your Phone & Arrive Alive.

Screenshot Showcase



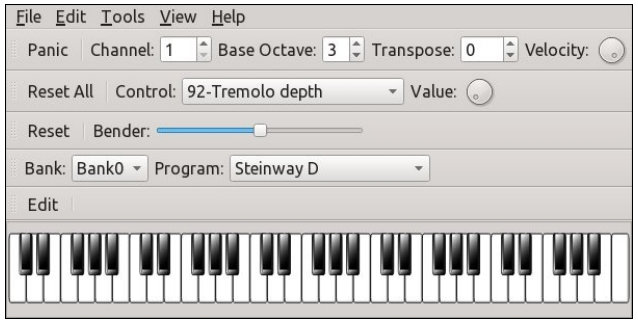
Posted by Robin on June 25, 2017, running Xfce.

Repo Review: Piano-Related Programs

by CgBoy

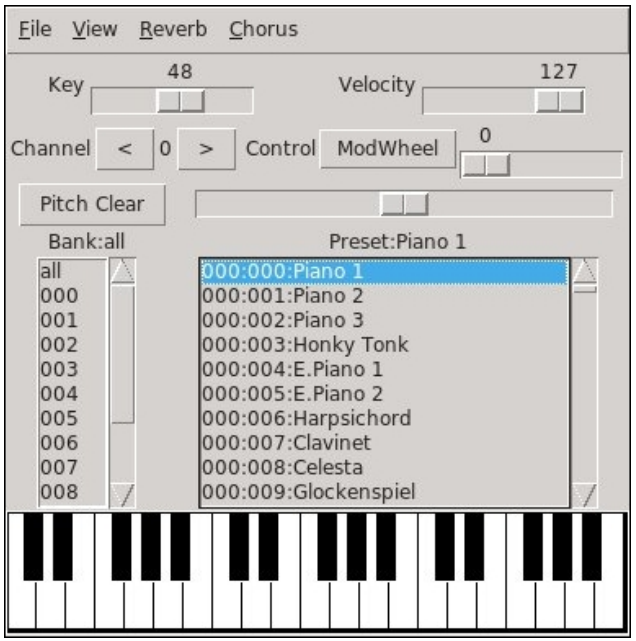
Pianos are some of my favorite musical instruments. They just sound so awesome. Last year I did an article reviewing guitar-related programs in the repository. So this month I'm reviewing some piano-related software. I have put these in the order in which I reviewed them.

Virtual MIDI Piano Keyboard. Virtual MIDI Piano Keyboard, or VMPK, is a virtual piano keyboard, which can be used with software and hardware MIDI synthesizers to generate audio. For this review I used a software synthesizer (FluidSynth). To play the virtual piano, you can use the mouse or computer keyboard. It's easy enough to change the keyboard mapping if you don't like the default one.



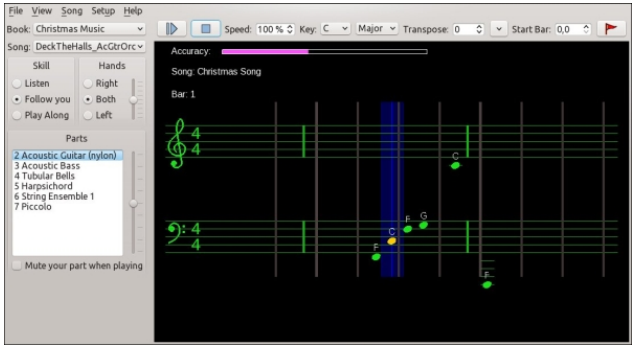
There are a variety of audio effects VMPK can use. Some of these include “Modulation”, “Tremolo depth”, and “Chorus depth”, along with many others. Virtual MIDI Piano Keyboard is pretty easy to use, and fun too. I enjoyed playing music using the computer keyboard.

Virtual Keyboard. Well, this is another virtual MIDI piano program. I used FluidSynth again for this review. Like VMPK, Virtual Keyboard uses the mouse and computer keyboard to control the piano. The keyboard mapping can be changed by editing one of the program's configuration files. Virtual Keyboard's interface isn't quite as easy to use as VMPK's, but it's still fairly simple.



The program also has some audio effects. Some of those include “Reverb”, “Chorus”, and “Sustain”, along with others. Although VMPK is slightly more advanced than Virtual Keyboard, I think I actually prefer this one.

Piano Booster. Piano Booster is meant to teach you how to play the piano. You load a MIDI file into it, and it shows you how to play it. There are three modes of learning. The first one, “Listen”, just plays the music while the notes scroll past. The second one, “Follow you”, plays the music and stops when it comes to a note, then continues when you press the right piano key. And the third one, “Play Along”, plays the music while the notes scroll past, and you can then play along to the music. You can select what instrument in the MIDI music file you want to play, along with the speed of the music. You can also set what hand you'll be using to play the piano.



Although Piano Booster is meant primarily for use with a MIDI keyboard, you can also use it with the computer's keyboard. For a synthesizer, I again used FluidSynth. Piano Booster is a really great program. It's easy enough to just load in a MIDI file and start practising playing the piano. I can see I'll most likely be using this program a lot, as I'm currently learning to play the piano myself. I really recommend this.

Summary

If you're looking for a virtual MIDI keyboard, I think I'd recommend Virtual MIDI Piano Keyboard. And if you're learning to play the piano, and you have a MIDI keyboard, Piano Booster would be a really helpful program.

Testimonial: A Technophobe On Linux!

by Robin

My home planet was first colonized by people who had rejected technology because it had been used to almost destroy our entire race! So my parents colonized a new world where we wouldn't be bothered by outsiders with technology to threaten us again. In the middle of a techno-hostile spot called "the briar patch," our new planet offers us a simple life. Unfortunately it offers something more, that some outsiders conspired to steal. If not for the efforts of the Federation starship Enterprise, all the Baku would have perished.

But in order to foil the plot against us, we had to allow some of their technology. The most amazing and frightening was an artificial life form named Data. I was scared to death of Data, but after a time I actually made friends with him.

If all this sounds too weird, read about the movie, Star Trek: Insurrection.

Anyway, the same kind of thing applies for me when it came to computers. At first a necessary evil, then a useful tool, and then a maddening, expensive appliance that required more from the user than it could offer in return (Windows). I thought about a Mac, but the price tag! So a little more fact-finding and there's this thing called Linux for desktop computers that costs nothing!

Okay, long story shortened: A "distro chooser" article led me to my first Linux, and from there to a few others. I was content on a favorite until this whole systemd thing started. It hadn't really been a problem for me in my limited use of Linux (and my lingering technophobia), but the more I read the more I got a little scared of the whole idea of some

"supervisory" software controlling and keeping a record of every single process on the computer! Not the Linux/Unix way. At all. It's intrusive and too far-reaching. Tentacles everywhere!

At first I experimented with a wonderful little Slackware-based distro to escape the tentacled monster. It was great! But it's really not for newbies, nor technophobes like we Baku from the briar patch. It's still awesome and very different from the Ubuntu-based (and systemd-encumbered) distros I had always used. Back to the web searching, and consulting my friend Commander Data, I found PCLinuxOS! All I can say is wwowwww. Super-duper-ultra-mega fast and simple enough for a technophobic Baku. I'm using the Xfce flavor on an older hand-me-down computer and it races along at something approaching Warp speed! Easy on the hardware, easy on the eyes, easy on the learning curve too.

I have "lurked" in the forums for weeks getting a few questions answered just by reading what others have dealt with. But yesterday I decided to join this wonderful community and maybe learn enough to contribute more than just gratitude and financial help (next payday).

Thanks to Texstar and Company for an awesome distro and a vibrant community that I look forward to getting to know.

10 (Potentially) Fatal Linux System Security Sins

by Paul Arnote (parnote)

Let's face it. Those of us that use Linux, do so because we like it. We like that it is free, as in beer. We like that it is robust. We like that we have many more choices under Linux. We like that it is secure. We like that it is virtually virus free. We like the community that has built up along side of Linux. There's a lot to like about Linux.

But, are YOU doing something that potentially destroys that sense of security you get from running Linux, maybe without even realizing it? Be careful, or that utopian Linux world may come crashing down around you.

Let's look at ten of those actions that can make using Linux a literal nightmare.

#1. Installing from outside the official repos. If you've spent any significant amount of time using Linux, you most likely have committed this cardinal sin. The Widgetery program that you need just isn't available in the official repos, and there isn't broad enough interest in it amongst the community for the maintainers to add it to the repos. However, you've located the source files. You take it upon yourself to compile and install the Widgetery program and some of its dependencies yourself, from the source files.

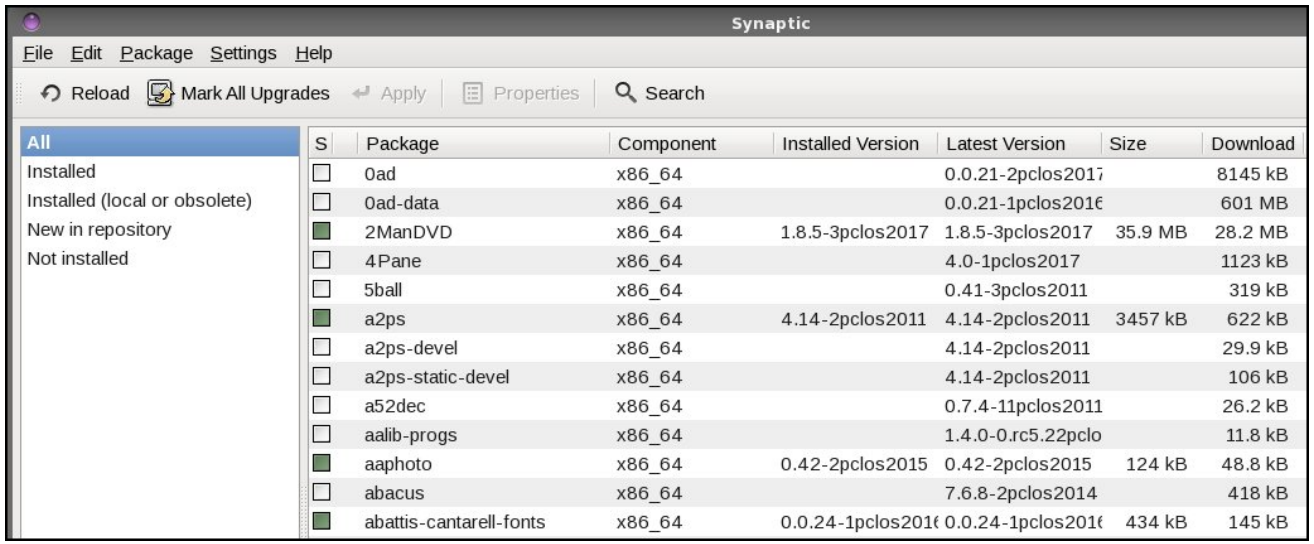
Initially, things are likely to run just fine. But let's say six months down the road, another program that has similar dependencies to Widgetery is added to the repos, and you install it. The newer program updates some of the lib files that Widgetery relies on, breaking Widgetery because it hasn't been updated to use the new lib files. Widgetery now refuses to run properly. Thinking that the problem is with Widgetery, you reinstall Widgetery and its dependencies from source. Now, the new program refuses to run, because the newer versions of the lib files that it needs are no longer there, having been replaced with the versions that Widgetery needs to run properly. This is known as "dependency hell," and represents an oversimplified example. Often times, this incompatibility goes on for multiple layers of dependencies, making it impossible for all but a well seasoned maintainer to resolve.

When a package is added to the official repos, great care has been undertaken by the repo maintainer (generally) to insure that when a new package is added and installed, it doesn't "break" other programs. Anyone can build and compile a program from source with a minimum of effort. But it takes some significant knowledge of the overall picture of the other programs that are in the repos to insure that everything runs well together. That's knowledge that most of us do not possess, and it's knowledge that takes quite a bit of time to acquire.

You can experience similar catastrophes by installing packages intended for one Linux distro on a different distro, since the maintainer of the other distro might have packaged things just a little bit differently than the maintainer of your distro.

Another potential pitfall you can experience from installing programs from outside the official repos is that you might unwittingly and unknowingly install malicious software. That malicious software may put your data and sensitive information at risk. You have to do quite a bit of research to be certain that the programs you are installing aren't going to compromise your data or sensitive information. Once installed, that malicious software now potentially has access to your entire system, including your files, email contacts, emails, browser history … everything!

In short, it's rarely worth the risk to install programs from outside the repos. PCLinuxOS has a policy of NOT providing support for your installation if you have installed software from outside of the official PCLinuxOS repository. Quite simply, no one really



knows exactly what changes were made to your system. If you ever wondered why this seemingly onerous rule exists, you now know why.

#2. Not updating regularly. Distros, like PCLinuxOS, are constantly being updated. Failure to update your system regularly is a very unhealthy habit. You expose your system to security issues and vulnerabilities by not having the latest software installed. Updating your software allows you to stay one or two steps ahead of new exploits. Plus, updates often include new features and improved stability. Linux doesn't remain the stable, secure platform it is by remaining stagnant. As security vulnerabilities and exploits are discovered and closed, so should your operating system and programs be updated to protect against those vulnerabilities and exploits. As technology changes and moves forward, so should your operating system and programs. As programs evolve to better meet the needs of the user, so should your operating system and programs. The only way to keep pace is to keep your system updated with the latest versions available in the software repository.

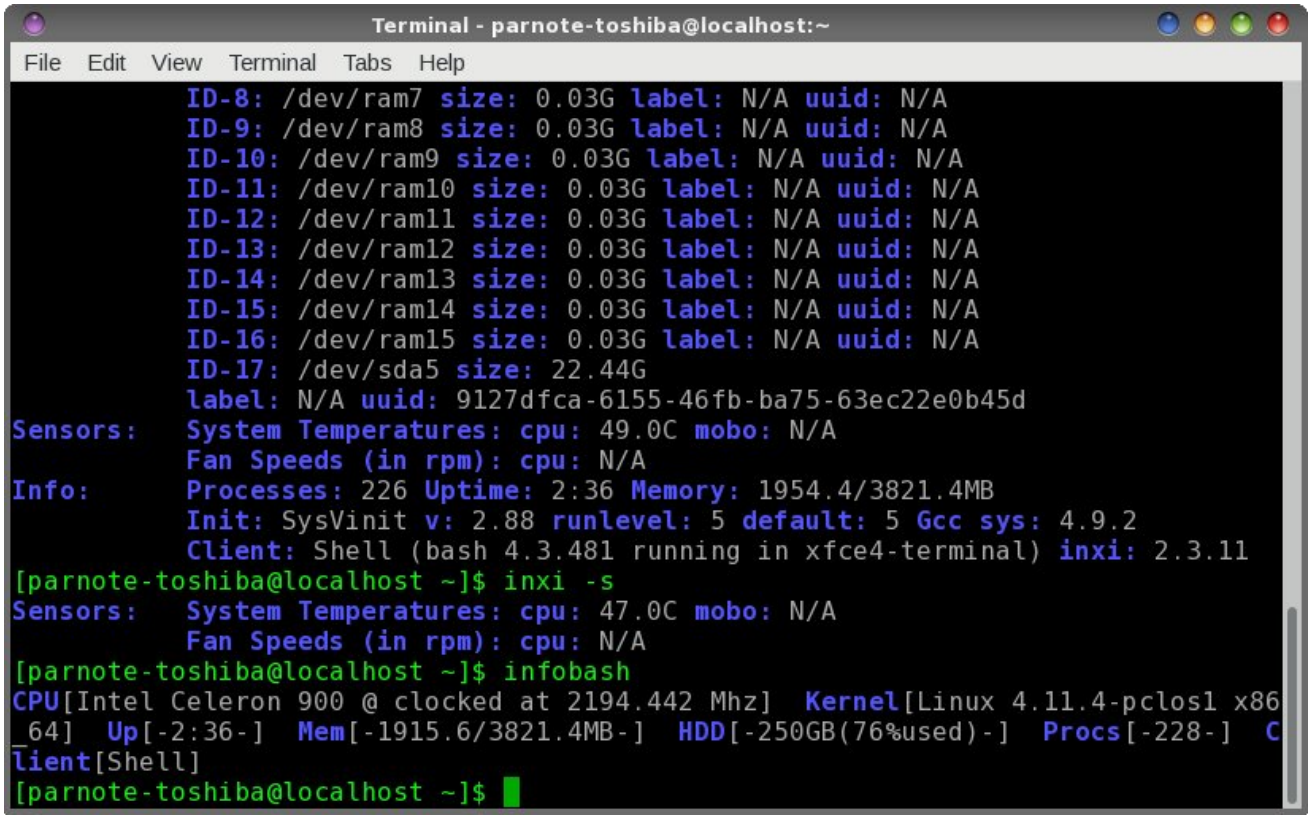
#3. Selectively updating. This is a sure fire way for failure. It won't strike you right this minute were you to selectively update, but it will eventually catch up with you. Updating only the few programs you regularly use, but none of the rest of your system, can cause broken dependencies and cause programs to no longer function properly. You should only ever update your entire system. In Synaptic, you do this by selecting "Mark All Updates," and then "Apply."

#4. Weak root password. You wouldn't leave the key to your house or car just sitting there in the lock, would you? Just like the keys to your house or car, the root password is the key to gain access to your Linux installation. Don't make your root password easy to crack by protecting your system with a weak root password. Yes, knowledgeable people can change your root password to anything they want, if they know how. But the vast majority of users don't know how to do it, or don't know that is even possible.

#5. Ignore your kernel. Just running updates isn't enough. Regular updates won't update your kernel. That is something you have to go in and do manually. Running on an outdated kernel can expose your computer to security vulnerabilities and exploits. Older kernels also will not offer the level of hardware support that newer kernels offer. Keep just one kernel that works on your computer, plus the latest kernel. You don't have to keep eight different kernels installed on your computer. There is no benefit, and it just takes up precious drive space. Let's say you have been using 4.11.4 and 4.11.6 comes out. You can remove version 4.11.3 of the kernel, keeping 4.11.4 as your "backup" kernel.

#6. Avoid the command line. It's easy to understand why some people shun the command line like they shun a major illness. Today's Linux desktops are stable and robust. They are easy to use. But what happens when your computer won't boot to a GUI desktop? You don't have to be a command line commando, but you should at least know a handful of command line commands, should you ever be forced to use the command line. Plus, who knows … you may discover that you actually LIKE the command line. Some things are much easier to accomplish there than through a GUI. You might also discover that things happen there much faster, are more reliable, and offer more security.

#7. Don't backup important system/configuration files. Face it. Until something else comes along to replace it, we're pretty much stuck with Xorg. Wayland has often been touted as Xorg's replacement, but even that seems to be way off on the horizon. So, until either Wayland or something else that hasn't gained prominence yet appears,



we'll be using Xorg. If you've used Linux long enough, you are familiar with the situation where Xorg is updated, but something in the update process didn't go exactly as intended. When you reboot, you find yourself unable to launch the X server. If you see that Xorg is going to be updated, make a copy of your working xorg.conf file (located at /etc/X11/xorg.conf) in the /root directory. That way, if something goes awry, you have your backup handy to be restored from the command line. The same thing goes with Apache, Samba, MySQL and any other critical system files you might need.

#8. Misunderstanding permissions. Part of the security of Linux is that some files are protected from standard users. That is, regular users don't have permission to edit certain files. While this can be somewhat inconvenient if you are the administrator of the computer, but logged in as a regular user (which you should be … more on that later), avoid changing the permissions on the file in question. It would be tempting to simply change the permissions by issuing the command chmod 777 at a command line prompt. But, that would be risky, since that gives read, write and execute properties to ALL users … even malicious users. So, here's a fairly easy way to understand user permissions:

777	read, write and execute permissions for all users
666	read and write permissions for all users
555	read and execute permissions for all users
444	read privileges for all users
333	write and execute permissions for all users
222	write permissions for all users
111	execute permissions for all users
000	no privileges to all users

Of course, it can get a bit more complicated than this, but this abbreviated table should give you some basic understanding of how permissions work. The last thing you want to do is give unfettered access to things that others should have no access to.

#9. Logging in as the root user. Never, ever, ever log in as the root user. If there are some tasks that you need to complete as the root user, su to root, complete your work, and then go back to a regular user. When you login as root, you have access to – literally – everything. It will also allow any malicious users or software the opportunity to carry out their malicious deeds. Plus, if you make a mistake (delete the wrong file, for example), it can be catastrophic for your system.

#10. Ignore log files. I suspect most Linux users are guilty of this. I know I don't pay as close

attention to my system's log files as I probably should. Fortunately, Linux stores most of its log files in one convenient location: /var/log. Your log files will help you figure out any problems you might be having with your system. Those problems could include hardware problems, hardware recognition, system services that are failing to load, etc. If you're suspecting security issues, check in /var/log/security. Otherwise, /var/log/messages should be your first place to look. This is a common log file where all generic errors and messages are logged. It should contain information about networking, media changes, etc.

The best and most complete way to view your log files is directly in a standard text editor. PCLinuxOS users have another choice, though. You can search through your system log files by opening the PCLinuxOS Control Center (PCC), and going to the "System" tab. There, under the "Administration tools" section, you can select "View and search system logs," highlighted with the red box in the image above. PCLinuxOS users can also install logcheck and logwatch via Synaptic, for two other options for monitoring log files.

Summary

One of the things we all love about Linux is its security. The last thing you will want to do is compromise that security with poor security habits. I'm not going to say that you will be exempt from any issues if you follow all of these practices, but I will say that you will have a much more secure system. In today's environment where there's a security risk to your computer and your data/information behind every blade of grass, you can't be too careful. You'd hate to open the doors wide to the kingdom, when keeping them closed and everything secure is so relatively easy.

Screenshot Showcase

Posted by phorneker on June 9, 2017, running Xfce.

PCLinuxOS Family Member Spotlight: Treedragon

As told to YouCanToo

What is your name/username?

Treedragon

How old are you?

Older than my teeth, a pensioner these days (over 65), and still blessed with a body that works reasonably well. Or so it felt until I picked up that last block of stone. And well except maybe for the hair, or lack thereof in some departments like on top, often hidden by the hat on many of the hotter days. But at least I did figure out why some of us go bald/er, while hair elsewhere sprouts with speed as we age. We need a viable vitamin D collection zone.

Are you married, single?

Been there, done that. It's all good.

How about Kids, Grandkids (names and ages)?

I have a daughter, Aethalia, named after my then yacht, which in turn I named after one of the wives of Zeus after a long rebuild and lengthen, we never did think of a boy's name came the day.

Aethalia was lucky really because she could have been named after the new motorbike I received on the same day she arrived, but then again FZR1000 doesn't quite roll off the tongue the same. Or, heaven forbid, the original name of the boat before I converted it into a gaff rigged cutter, Spratt II. However for many reasons I am the lucky one, especially as indicated by a long distance phone call from Aethalia some years later. She had checked on Google as to the meaning/source of her name, the first thing that came up was a giant fungus.... oops!

Zoe is my grand daughter, a six year old that knows exactly where she is going. With me being one that really appreciates the "quiet spaces" in life, I find myself of two minds about the long distance away from me they are.

Do you have pets, what is your favorite?

Not these days I decided that unless one could guarantee to be there for "a companion" to the end, it was better to get to know the locals, as in anything with legs or wings that haunted the garden and wouldn't run away when I spoke to it. To my mind, they have the advantage of being able to feed themselves, not reliant, while still being get to knowable, but … well I remember a dog from years ago. I would go there again I suspect.

Are you retired, still working and if working, what do you do?

The term retirement doesn't compute. I am known for raking leaves.

Where do you call home? What is it like? IE: weather, scenery

Home is wherever I am at all and any times.

Where do I live now? Touchy department, that … hehe, but it's green, very green, and the things called mountains up here are but hills compared with my more favoured parts of this country. We are tucked up in a small valley on the Coromandel Peninsular, of the North Island of NZ, a place of green and Tree Ferns 30 to 40 feet tall. Here there be pretensions of the sub-tropical with flourishing bananas as counterpoint to welcome hints of the the colder temperate moments, especially in winter when the banana palms turn to frizzled but photographable stumps. It rains, up to 100mm (4 inches) per hour occasionally and often it doesn't. Then all is well unless it really really doesn't, rain that is, the locals like to think that is a drought. So it is all a balance of shades of green and water, but very nice to visit. Even I say that, and then if there is the need to escape, the sea coast is just a few miles away.

You can get some hints of the place for yourself on the web at www.maharasculpturepark.kiwi

It's 10 acres of regenerating native forest with walks and other stuff, secluded but not too remote, not especially flash, more quirky and eclectic, but it does feel good. There is way too much grass that needs mowing, and a rich history of tribal battles, gold rushes, giant Kauri trees, epic storms, people that make you go hmmmmmm, and not very good broadband at all.

Where did you go to school and what is your education level?

I went to school in the other island, the South Island, the one with real mountains and real weather, (I like it cooool). I was destined, (brainwashed), and qualified for higher education, but direct experiences with some of those thus trained by the higher education system … very good in their field of expertise but … well, that did not feel right for me. As a result, I chose to go my own way. It was the absolute best decision of my life, one I recommend to all if appropriate.

What kind of things you like doing? hobbies, travel, fishing, camping?

I sometimes use a by line of "*I can't leave well enough alone*," so I pursue those things that grab my attention. They include all of the above, at least. These days I also choose creating stone sculpture, getting my head around modern digital photography, the creation of spaces that feel good, and am seeking to continue my love of fast, precision, motorcycle riding, no matter the bike type, style, or origin.

In alignment with the "can't leave well enough alone" department, the above mentioned FZR1000, with suitable "*not leaving well enough alone*" tweaking got me well over 300 kph/186.4 mph in the early nineties. Which was sort of good, as the then Police Radar Cameras here registered a blank photo after about 246 kph/152.8 mph. They did figure out who I was and were very polite about it, (*lack of evidence I think*).

The photographic aspect is capturing me quite a bit these days, particularly after it rains when all those weird fungi start sprouting. The subsequent fiddling with Darktable, trying to make it all look right, keeps me involved for lengthy amounts of time. The challenge here is to figure out how to photograph all this green and have it look like it feels. And no, I

don't mean that damp trickling down the back of one's neck. Visitors here routinely comment on how beautiful it is and how lovely it feels, but to capture that in an image, well I'm working at it in between leaf rakings.

Some say that being here must be like living in paradise. I hand them a leaf rake at that point.

Why and when did you start using Linux?

I bought a computer knowing nothing about computers. I still don't, but that doesn't stop the learning. It had Windows 98 installed, and I had been shown the Power On button … so I pressed it.

After much exploration, I came to a conclusion, but I learned to put up with it anyway by using it. Then I put together my own machine and also discovered Freeware. Hmmmmm. And then I read rumours of an OS, (Linux), for those of us that didn't know this from that and it didn't have the usual constraints. I waited a bit more, then heard about Xandros as an

easy step. Something resonated. I went there for a short while. Then I heard of PCLinuxOS .92 I think it was, I haven't looked back since.

What specific equipment do currently use with PCLOS?

Gigabyte GA-MA770-UD3 motherboard,
Ram DDR2 800 PC2-6400, 8GB in total,
CPU: 4 core AMD Phenom II 955,
2 x HP LP2475w screens,
Assorted hard drives with a HP SSD 40GB for / and /home,
Running KDE 5.xx very well

What would you like to see happen within PCLOS that would make it a better place. What are your feelings?

I arrived here years ago because it felt right and … well, that amounts to Texstar and his dedication feeling right in the first instance and onward over the years. And then, these days, there are so many others who help also. To my eye, there are very sound underpinnings to this distro and in the individuals and their decisions.

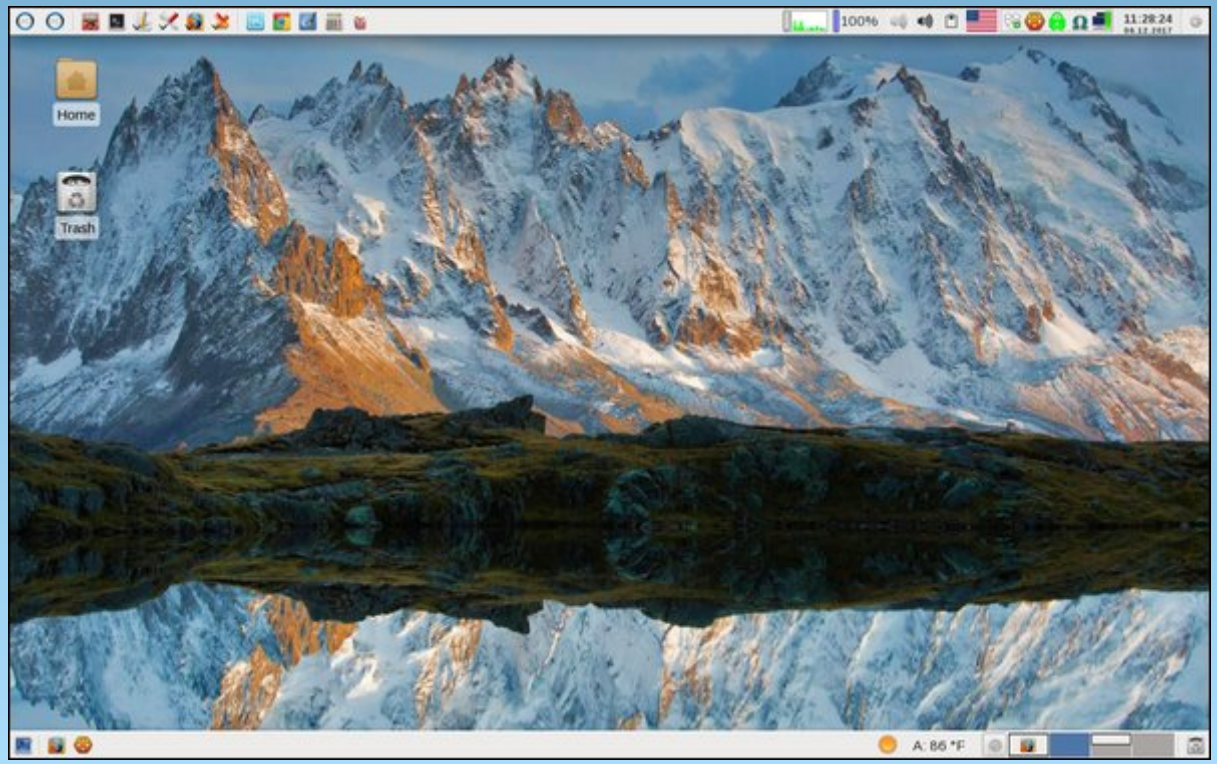
I like how occasionally I can help in some small ways within the community, as it grows with us all.

I am still here because it still feels right and it feels right because the family here cares … soooooooo, nothing I can say really.

PCLinuxOS Family Member Spotlight is an exclusive, monthly column by YouCanToo, featuring PCLinuxOS forum member. This column will allow "the rest of us" to get to know our forum family members better, and will give those featured an opportunity to share their PCLinuxOS story with the rest of the world.

If you would like to be featured in PCLinuxOS Family Member Spotlight, please send a private message to youcantoo, parnote or Meemaw in the PCLinuxOS forum expressing your interest.

Screenshot Showcase



Posted by parnote on June 12, 2017, running Xfce.

ms_meme's Nook: YouCanToo

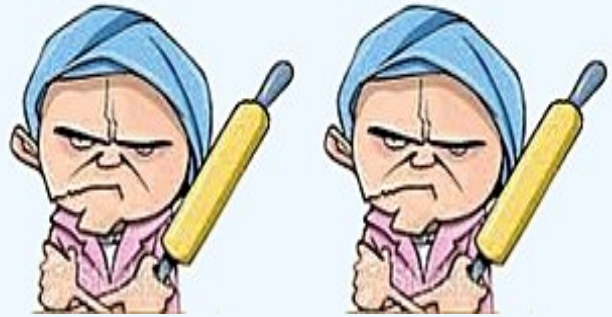
YouCanToo lives in Oregon
Starts his work before dawn
What he can do is a phenomenon
Never says no to anyone

YouCanToo YouCanToo
A one man crew
Lordy what he can do
YouCanToo YouCanToo
He knows how-to
Lordy what he can do

In the forum he is known as
Mr.Cranky Pants

A better name you would agree is
Mr. Smarty Pants

MP3 ***OGG***



YouCanToo lives in Oregon
His work is never done
What he can do is a phenomenon
His talent goes on and on

YouCanToo YouCanToo
A one man crew
Lordy what he can do
YouCanToo YouCanToo
He knows how-to
Lordy what he can do

Introduction To TeX/LaTeX (Using TeXLive)

by phorneker

So what is TeX and LaTeX? TeX is a typesetting language intended for publication of scientific and mathematical texts developed by Donald Knuth in 1978. It was used to publish books such as The Art of Computer Programming, which you can order from Amazon. The idea behind the typesetting language is for the author to focus on the actual content of the writing, and allow the computer to do the formatting and typesetting of pages.

TeX is actually the spelling of the greek letters Τεχ (Tao - Epsilon - Chi) and is properly pronounced as “tech”. Likewise, LaTeX is pronounced as “Lah-tech”, with “La” being the first two letters Leslie Lamport’s last name.

Documents are usually structured by chapters, pages, sections, subsections, headers, lists, tables, graphics, and other elements you would embed into a document in LibreOffice or Word Perfect. The same could be said for HTML and XML languages as their basic concepts are similar to TeX.

As with any programming language, TeX commands can be combined into stylesheets and defined as **macros**, similar to defining procedures and functions in programming languages such as C, Pascal, Python, and Java. Leslie Lamport developed a system of such files in 1985, and released that project as LaTeX. Current TeX distributions such as TeXLive (installable from the repository, or from a TeX User Group DVD if the **texlive-dummy** package is installed) integrate the LaTeX system of macros into the TeX distribution. Together, the system is useful for more than just typesetting mathematical publications. (I have used it to typeset a form of my resumé.)

Who is Donald Knuth?

Donald Knuth is a professor emeritus (retired) of computer science at Stanford University.

According to Wikipedia:

“He is the author of the multi-volume work The Art of Computer Programming. He contributed to the development of the rigorous analysis of the computational complexity of algorithms and systematized formal mathematical techniques for it. In the process he also popularized the asymptotic notation.

In addition to fundamental contributions in several branches of theoretical computer science, Knuth is the creator of the TeX computer typesetting system, the related METAFONT font definition language and rendering system, and the Computer Modern family of typefaces.

As a writer and scholar, Knuth created the WEB and CWEB computer programming systems designed to encourage and facilitate literate programming, and designed the MIX/MMIX instruction set architectures. Knuth strongly opposes granting software patents, having expressed his opinion to the United States Patent and Trademark Office and European Patent Organisation.”

This is only part of what he did in his lifetime.

Knuth is very particular about how people pronounce his last name. According to his website at Stanford University, the proper pronunciation of Knuth is **ka’-nooth**, and not **nooth**.

For more information on Donald Knuth, visit his website at http://cs.stanford.edu/~uno/.

IMO, Donald Knuth is a true legend and one of the founding fathers in the world of computer science.

The TeX typesetting system has several components, two of which I already mentioned, the **macro system** for combining TeX commands, and the TeX command language itself.

Another component of TeX is the **WEB programming language**, designed for literary programming (that is typesetting), originally written in Pascal (for teaching of good programming practices), then ported to C (for the **CWEB** derivative, sometimes called **WEBC** depending on the TeX distribution) to keep up with the programming trends of the day. The **WEAVE** and **TANGLE** programs are key components of **WEB** with **WEAVE** generating the TeX documentation from **WEB** sources, and **TANGLE** generating the Pascal source code from **WEB** sources. (In **CWEB**, **TANGLE** generates C source code instead of Pascal.)

In traditional programming languages, the documentation is embedded into the source code files. In **WEB**, the program source code is embedded into the program's documentation.

Much of the TeX sources are written in **WEB**, and **TANGLE** is used to create the source files, which are then compiled to produce the binaries that are run in PCLinuxOS. **WEB**, likewise, creates documentation from the same source files.

Electronic typesetting would not be complete without some way of generating the fonts (or typefaces) used to prepare the documentation for printing. **METAFONT** is the component that handles that task.

The capitalization and naming of these components is intentional, and should always be spelled this way. For example, **METAFONT** is spelled as **METAFONT**, and not **Metafont**, or **metafont**.

This is done in this article out of respect to Donald Knuth and his works.

METAFONT is used to create the actual fonts that are used to produce the DVI output files. Fonts used in the document are generated on the fly if they have not already been done. When you first use TeX, many of these fonts will need to be generated after the initial processing of the document. Once the fonts have been generated, they are stored on your system for future use.

When you installed the **texlive** and **texlive-dist** packages from the repository, many of these fonts, normally generated by **METAFONT**, have been installed for you saving significant time when compiling TeX and LaTeX documents.

DVI stands for **D**e**v**ice **I**ndependent and is a file format used for document preparation independent of any printer, display or typesetter used to produce the final documentation.

This type of code is similar to the Java pseudo machine code generated by the Java compiler, or p-code from UCSD Pascal in the sense that an interpreter is needed to execute the code, that is, the final printed product.

...which brings us to the final component of TeX, the **DVI utilities** (should really be called interpreters).

Much of what we do in TeX will not involve any of these DVI utilities. However, applications such as **Texmaker** and **LyX** require them to function.

For PCLinuxOS, the **dvips** utility interprets the DVI code, and produces PostScript code for output (usually spooled to CUPS or piped to **ps2pdf** for PDF generation).

Okular, included with KDE Plasma, can be used to read DVI files directly in PCLinuxOS.

As a result, the TeXLive distribution is a very large and complex system for typesetting of documents and expect installation from Synaptic to take about an hour to download on a DSL connection. Once installed, the possibilities of what you can do with TeX is as large as TeX itself.

To really understand the TeX typesetting system, you will need to obtain some actual reading material.

* The TeXBook by Donald Knuth, Addison-Wesley, ISBN 0-201-13448-9, 1984

* A Guide to LaTeX2e, by Helmut Kopka and Patrick Daly, Addison-Wesley, ISBN 0-321-17385-6, fourth edition, 2003.

* The LaTeX Companion, by Frank Mittelbach, Michel Goossens, Johannes Braams, David Carlisle, and Chris Rowley, Addison-Wesley, ISBN 0-201-36299-6, second edition, 2004

* Learning LaTeX, by David Griffiths and Desmond Higham, SIAM, ISBN 978-0-898713-83-1, 1997, 84pp

...and if you really want to see how large this system really is, just browse the CTAN repository (at http://ctan.org/tex-archive).

Important Note about CTAN: Donald Knuth's contributions to the repository are contained in a directory with his name on it (http://ctan.org/tex-archive/systems/knuth). Material contained in this part of CTAN **may not be modified without Donald Knuth's permission**. (As of this writing, Donald Knuth is still alive at the age of 80!)

More information on the WEB and CWEB programming language can be found at http://www-cs-faculty.stanford.edu/~uno/cweb.html.

Writing Documents in TeX/LaTeX

When writing documents in TeX and LaTeX, we typically use an application such as **Texmaker** or **LyX**, both of which are available in the PCLinuxOS repository. These applications make it easy to produce documents written in TeX. (LyX comes with its own format for storing documents, and saved with the **.lyx** extension.)

As TeX and LaTeX documents are plain text files, you may use any standard text editor to write TeX/LaTeX documents. When saving TeX/LaTeX documents, it is conventional to assign either the **.tex** extension (for TeX) or the **.ltx** extension (for LaTeX) to indicate that this is a TeX/LaTeX source document. I prefer to use **.tex** for both TeX and LaTeX for consistency as many documents I have come across that were written in TeX and LaTeX use this file extension.

Let us start with an easy (Hello World) example of a TeX/LaTeX document. Open up your favorite text editor and type in the following:

```
% This is an example document for TeXLive.
%
\documentclass{article}
\begin{document}
Hello World!
\end{document}
```

Now save the document as **hello.tex** and type the following:

```
latex hello.tex
```

...and look what happened:

```
This is pdfTeX, Version 3.14159265-2.6-1.40.17 (TeX Live 2016/PCLinuxOS)
(preloaded format=latex)
 restricted \write18 enabled.
entering extended mode
(./hello.tex
LaTeX2e <2016/03/31>
Babel <3.9r> and hyphenation patterns for 83 language(s) loaded.
(/usr/share/texmf-dist/tex/latex/base/article.cls
Document Class: article 2014/09/29 v1.4h Standard LaTeX document class
(/usr/share/texmf-dist/tex/latex/base/size10.clo))
No file hello.aux.
[1] (./hello.aux) )
Output written on hello.dvi (1 page, 232 bytes).
Transcript written on hello.log.
```

We have successfully compiled a TeX/LaTeX document. Let us look at what files are generated.

```
-rw-rw-r-- 1 patrick patrick       8 Jun  1 08:00 hello.aux
-rw-rw-r-- 1 patrick patrick  232 Jun  1 08:00 hello.dvi
-rw-rw-r-- 1 patrick patrick 2006 Jun  1 08:00 hello.log
-rw-rw-r-- 1 patrick patrick  108 Jun  1 08:00 hello.tex
```

Three files were generated from this one source file. (The **.log** file recorded everything that happened during the compile.) We got our **.dvi** file and **.aux** file. We need all files listed here when we generate the PostScript file for printing.

The **.aux** file contains one instruction for the next time the **latex** command is run on this file.

The logfile, however, requires some explanation. By default, TeXLive calls **pdfTeX** with the **latex** command.

```
This is pdfTeX, Version 3.14159265-2.6-1.40.17 (TeX Live 2016/PCLinuxOS)
(preloaded format=latex 2017.4.25)  1 JUN 2017 08:00
entering extended mode
 restricted \write18 enabled.
 %&-line parsing enabled.
**hello.tex
(./hello.tex
```

The LaTeX2e packages are loaded by default instead of the LaTeX 3 packages. LaTeX2e is an older version of LaTeX, which will do for simpler documents such as this example.

```
LaTeX2e <2016/03/31>
Babel <3.9r> and hyphenation patterns for 83 language(s) loaded.
```

Class files used by TeXLive are stored here. **.cls** files contain TeX coded definitions for document classes. **.clo** files provide options that are used when processing LaTeX documents.

```
(/usr/share/texmf-dist/tex/latex/base/article.cls
Document Class: article 2014/09/29 v1.4h Standard LaTeX document class
(/usr/share/texmf-dist/tex/latex/base/size10.clo
File: size10.clo 2014/09/29 v1.4h Standard LaTeX file (size option)
)
\c@part=\count79
\c@section=\count80
\c@subsection=\count81
\c@subsubsection=\count82
\c@paragraph=\count83
\c@subparagraph=\count84
\c@figure=\count85
\c@table=\count86
\abovecaptionskip=\skip41
```

```
\belowcaptionskip=\skip42
\bibindent=\dimen102
)
```

When you run **tex** or **latex** on a document file, there will not be a **.aux** file.

```
No file hello.aux.
\openout1 = `hello.aux'.
```

Now, the next time **latex** is run on this document, LaTeX will know what to do to process this file.

```
LaTeX Font Info:    Checking defaults for OML/cmm/m/it on input line 4.
LaTeX Font Info:    ... okay on input line 4.
LaTeX Font Info:    Checking defaults for T1/cmr/m/n on input line 4.
LaTeX Font Info:    ... okay on input line 4.
LaTeX Font Info:    Checking defaults for OT1/cmr/m/n on input line 4.
LaTeX Font Info:    ... okay on input line 4.
LaTeX Font Info:    Checking defaults for OMS/cmsy/m/n on input line 4.
LaTeX Font Info:    ... okay on input line 4.
LaTeX Font Info:    Checking defaults for OMX/cmex/m/n on input line 4.
LaTeX Font Info:    ... okay on input line 4.
LaTeX Font Info:    Checking defaults for U/cmr/m/n on input line 4.
LaTeX Font Info:    ... okay on input line 4.
[1

] (./hello.aux) )
Here is how much of TeX's memory you used:
 198 strings out of 493015
 2078 string characters out of 6139340
 53598 words of memory out of 5000000
 3830 multiletter control sequences out of 15000+600000
 3640 words of font info for 14 fonts, out of 8000000 for 9000
 1141 hyphenation exceptions out of 8191
        23i,4n,17p,129b,107s stack positions out of
5000i,500n,10000p,200000b,80000s

Output written on hello.dvi (1 page, 232 bytes).
```

This may not be important for this document, but as you write larger documents, this type of data will come into play when problems start creeping up.

To produce a PDF file, we will need to issue two commands. First:

```
dvips hello.dvi
```

Which generates the following:

```
This is dvips(k) 5.996 Copyright 2016 Radical Eye Software
(www.radicaleye.com)
' TeX output 2017.06.01:0800' -> hello.ps
</usr/share/texmf-dist/dvips/base/tex.pro>
</usr/share/texmf-dist/dvips/base/texps.pro>.
</usr/share/texmf-dist/fonts/type1/public/amsfonts/cm/cmr10.pfb>[1]
```

dvips used the 10 point Computer Modern font to typeset the Hello World example. As this font was already installed with the **texlive-dist** package, the font did not need to be generated.

And second:

```
ps2pdf hello.ps
```

Now we have our PDF file. We could use the **lpr** command on either the **.ps** or **.pdf** file to print to a CUPS printer, or we could simply use a PDF viewer to view the file.

As a result, our **hello.tex** file has expanded to six files!

```
-rw-rw-r-- 1 patrick patrick     8 Jun  1 08:00 hello.aux
-rw-rw-r-- 1 patrick patrick   232 Jun  1 08:00 hello.dvi
-rw-rw-r-- 1 patrick patrick  2006 Jun  1 08:00 hello.log
-rw-rw-r-- 1 patrick patrick  4614 Jun  1 08:32 hello.pdf
-rw-rw-r-- 1 patrick patrick 26194 Jun  1 08:28 hello.ps
-rw-rw-r-- 1 patrick patrick   108 Jun  1 08:00 hello.tex
```

...and this is just a simple “Hello World!” document.

Of course, the typical TeX/LaTeX document is much more complicated than this, but this gives you an idea of what electronic typesetting has been like.

To print this example, simply type:

```
lpr -P <name of printer> hello.ps
```

or

```
lpr -P <name of printer> hello.pdf
```

...where <name of printer> is the name of the printer queue used to configure the printer in CUPS.

Basic Structure of a TeX/LaTeX Document

Let us look again at the “Hello World” document.

```
% This is an example document for TeXLive.
%
\documentclass{article}
\begin{document}
Hello World!
\end{document}
```

This is the simplest form of a document written in LaTeX. Look at the first two lines. Any line that begins with the **%** character is ignored by TeX. These are comment lines and are for the benefit of anyone working with the document so he/she will know what is being done with the document in terms of what is being typeset, the content itself, or anything that may need doing in the future.

Just as in any programming language, anyone reading comments will understand what the code is supposed to do.

The **\documentclass{class}[options]** line tells LaTeX what type of document is being produced here. In the example, we are producing an **article**. Valid types for class in TeXLive are: **article, book, letter, ltnews, ltxdoc, ltxguide, report,** and **slides**.

You can supply options to these class types such as the font size and paper type. For example, **\documentclass{article}[12pt,letter]** tells LaTeX that this document is an article that is to be printed on letter paper using 12 point fonts (with the default being Computer Modern, a font similar to Times Roman).

The **ltnews** class allows for the 35 basic PostScript fonts to be available when used with the **lw35fonts** and/or the **type1fonts** options. Otherwise, LaTeX will allow **METAFONT** to create the necessary fonts for typesetting the document.

No matter what type of document you are typesetting, the **\documentclass{}** statement **must be included** for the document to be typeset, preferably at the beginning of the content.

The **\begin{}** and **\end{}** statements that appear in the example tell us that TeX and LaTeX documents are structured and organized into pages, chapters, sections, subsections, and the like.

All documents **must contain** a **\begin{document}** and a **\end{document}** statement to tell LaTeX that this is the actual content of the document. Once this has been established, we can then incorporate a hierarchy of the document itself.

Hence, all TeX and LaTeX documents must start with:

```
\documentclass{class}[options]
  \begin{document}

  \end{document}
```

Where class is a valid class type (which you can find by typing **ls /usr/share/texmf-dist/tex/latex/base/** in a terminal window), and **options** is the initial font size and paper type. Paper types can be **a4paper** (210 mm x 297 mm), **a5paper** (148 mm x 210 mm), **b5paper** (176 mm x 250 mm), **letterpaper** (the standard 8.5 in x 11 in), **executivepaper** (7.25 in x 10.5 in) or **legalpaper** (8.5 in x 14 in). The **landscape** orientation is an available option added to the paper type, and if not specified, then portrait orientation of the document is selected.

What we have so far is the basic skeletal structure of a TeX/LaTeX document. Everything placed after the **\documentclass{}** statement and before the **\begin{document}** statement is the preamble. The preamble is used to set any options needed to typeset the document. This includes any TeX/LaTeX extensions and packages (locally installed from Synaptic or installed from a CTAN archive). Everything placed between **\begin{document}** and **\end{document}** statements is what gets typeset. This includes titles, authors, title pages, page breaks, graphics, and other elements found in the TeX/LaTeX system (of which there are many).

This was only an introduction to the TeX/LaTeX typesetting system. I have presented the framework in which TeX and LaTeX documents are started. As we can see, this is a very complicated system for typesetting structured documents, and intended for typesetting of mathematical and scientific documents.

However, TeX and LaTeX can be used for most any kind of document (but not all) that you can do on products such as LibreOffice, AbiWord or WordPerfect.

Unlike those products, TeX and LaTeX allows you to focus on your content instead of on the formatting of the documents. For the next article, I plan to use Texmaker to build a document that would normally be created with LibreOffice.

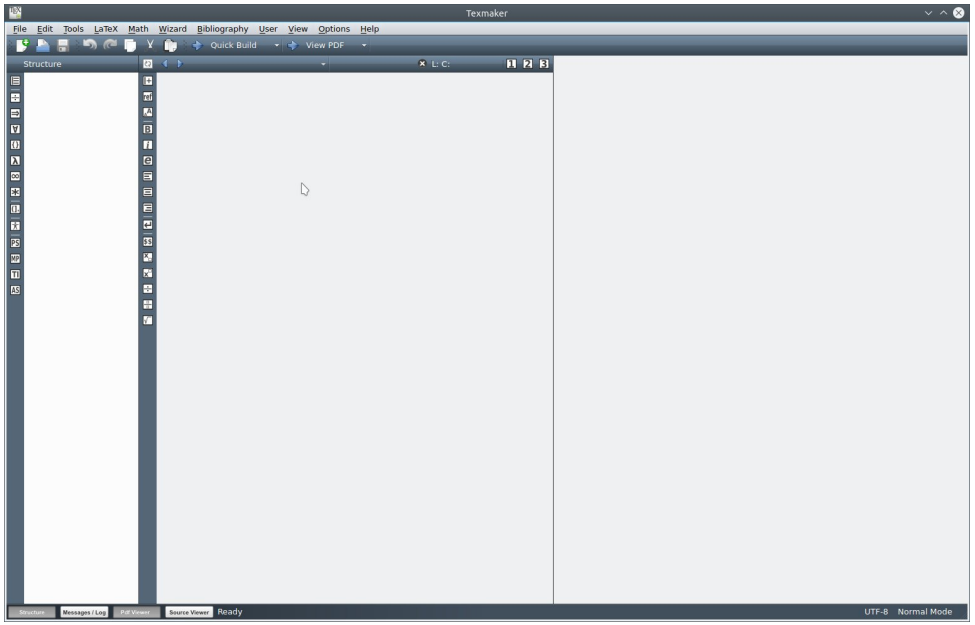
Texmaker: An Easier Way to Make TeX/LaTeX Documents

Just as there are integrated development enviroments such as Code:Blocks, Eclipse, Lazarus, and KDevelop for applications and system programming, there are integrated development environments for TeX and LaTeX.

Texmaker is one such application (downloadable from Synpatic) that makes document preparation easy. Other IDE packages available in the repository are Texworks, Texstudio, LyX, and Kile, as well as TeXmacs, which was at one time in of the repository, but now no longer is. The source package for TeXmacs, however, still remains available for building and installing on your PCLinuxOS installation.

Kile is a TeX development environment specific to the KDE 4 desktop. There is no KDE 5 version available, either in the repository, or on the Kile website. However, the Kile website has instructions on how to compile the Kile source code for KDE 5.

Even then, Kile is no match for Texmaker, which is the IDE that I recommend for TeX/LaTeX development. Texmaker has been optimized for KDE 5.

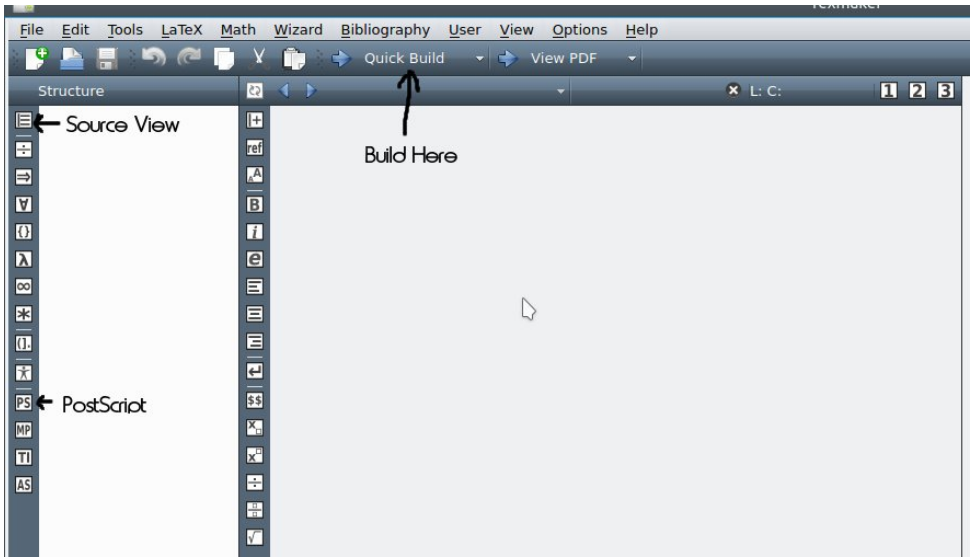


Texmaker is launched from the **Office** menu, and here is what you get when you launch Texmaker for the first time. There are three panels.

The first panel is a multi-use panel. There is a toolbar on the left side of the panel. Normally, this panel displays the structure of your TeX/LaTeX document. The top icon in the toolbar on the upper left side of this panel is used to display the document structure. The other buttons in the toolbar are used to select special characters for insertion into your document. Each button opens a palette of special characters in the panel.

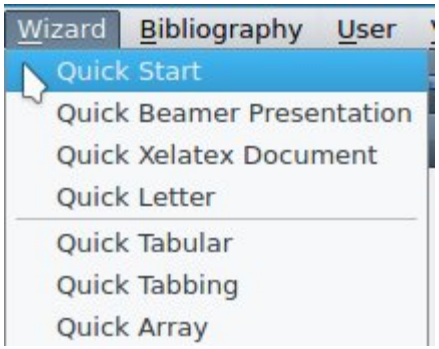
The second panel is where you edit TeX/LaTeX documents. The toolbar on the left hand side of this panel activates functions, not unlike the formatting toolbar found in LibreOffice. Some of these buttons activate popup menus. You can load multiple TeX/LaTeX documents into this editor, and switch between them with the toolbar at the top of this panel (as opposed to the top panel of the entire application window.

The third panel is a preview of your finished “typeset” document. To see anything in this window, you need to “Quick Build” the document and then select “View PDF” using the aptly labelled buttons located on the main toolbar. These buttons apply to the document currently being edited.

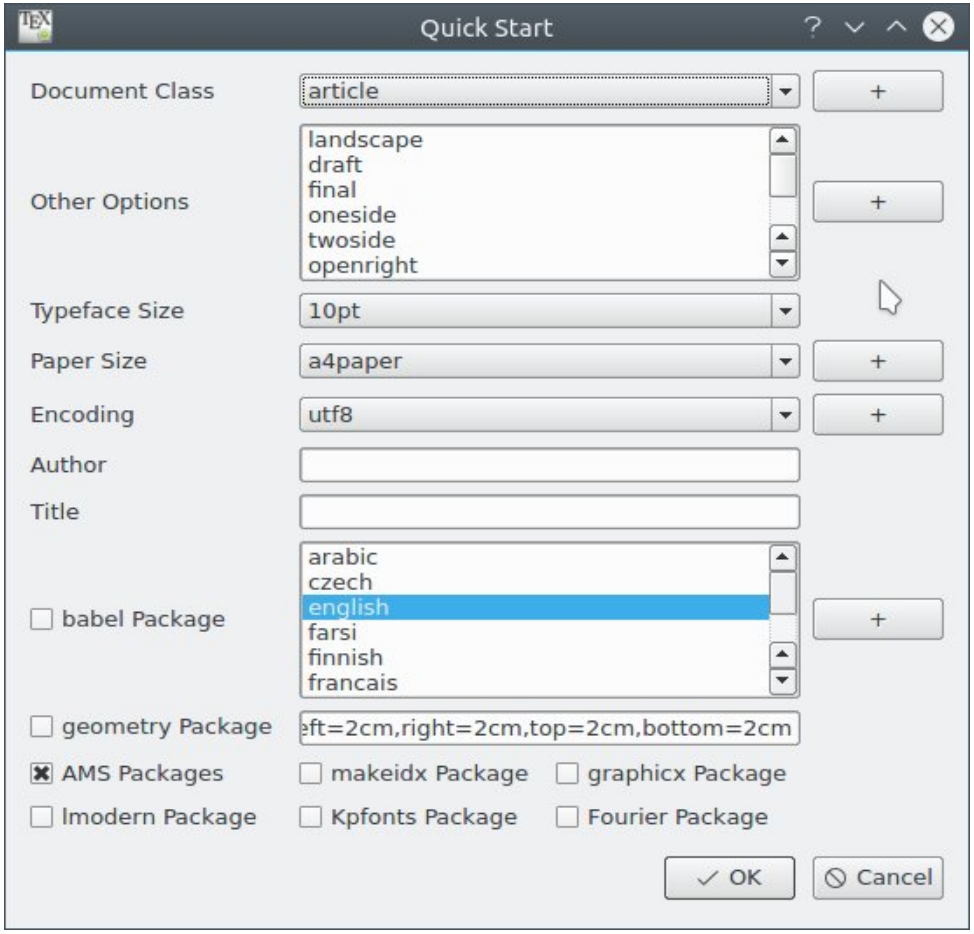


On the main toolbar, we have some familiar icons in their familiar locations, namely the **New Document, Load** and **Save**. Next, there are arrows where we can switch between documents. The next three buttons are for copying, cutting and pasting text from the clipboard, as with any word processing application. The arrow buttons next to that are for the compiling (building) of the TeX/LaTeX document, and for viewing the result in the right panel.

Ordinarily, we select **File → New** to create a new document. We can do that, but we get a blank document, the same as what we see when we launch Texmaker. To get a document we can really work with, we must use the **Wizard** menu.



Quick Start is what most of us would normally select to begin a new document. If you wish to write a letter, however, you would select **Quick Letter**.



As we can see, there are enough options available to be able to typeset most any kind document you wish. The buttons labelled with a plus sign allow you to add features such as classes, paper sizes, character encoding and languages, provided you have the support files for such options installed in your TeXLive installation. Texmaker loads the support files from **/usr/share/texmf-dist/tex/latex/base/**.

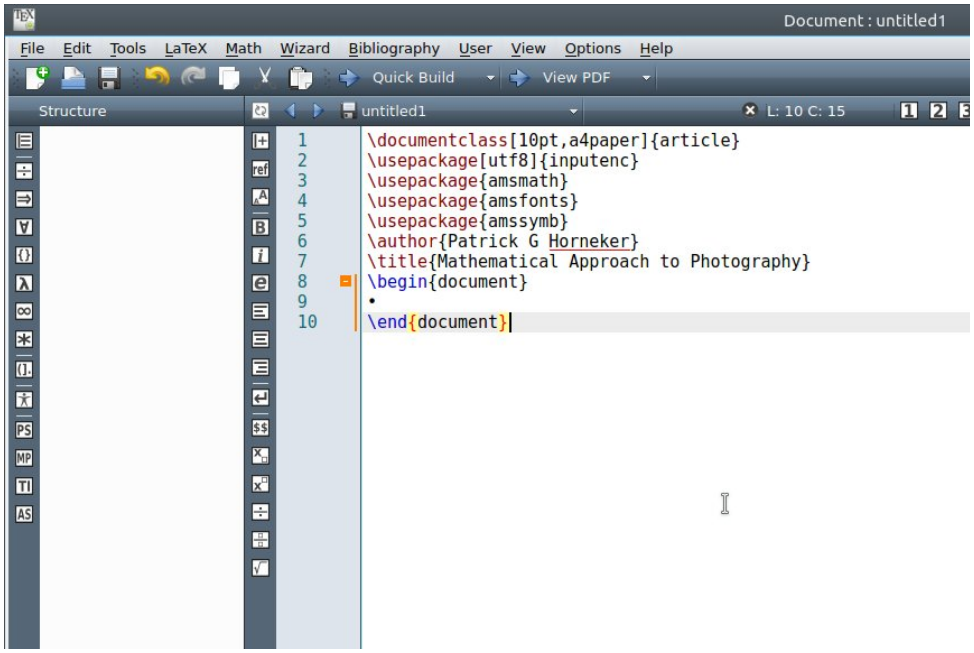
As shown, we would create a article, with portrait orientation on the page, using a 10 point Computer Modern font, on A4 size paper (about 8 x 10.5 inches).

We have the ability to change any of these parameters using this dialog box. Most of us will want to select **letterpaper** for paper size.

If we select the Geometry Package checkbox, we can change the margins of the overall document. In this examples, we get 2 centimeter margins on the printed page. The parameters here can be changed as well. By default, the class packages from the American Mathematical Society are selected by default. Selecting the **graphix** package allows graphics to be embedded into TeX/LaTeX documents.

Other than that, all you need to do is fill in the author and title fields.

Let us close this dialog and create a new example document. (Click on **OK** to do this.)



Look what happened. Our new example document has a preamble.

```
\documentclass[10pt,a4paper]{article}
\usepackage[utf8]{inputenc}
\usepackage{amsmath}
\usepackage{amsfonts}
\usepackage{amssymb}
\author{Patrick G Horneker}
\title{Mathematical Approach to Photography}
\begin{document}
•
\end{document}
```

We can make changes to this here the same as with any other text editor. For example, we can change the **\documentclass[]{}** statement. We can replace **a4paper** with **letterpaper** for output of the document to 8.5 x 11 inch paper. We now have:

```
\documentclass[10pt,letterpaper]{article}
\usepackage[utf8]{inputenc}
\usepackage{amsmath}
\usepackage{amsfonts}
\usepackage{amssymb}
\author{Patrick G Horneker}
\title{Mathematical Approach to Photography}
\begin{document}
•
\end{document}
```

Packages in TeX/LaTeX are archive files (with the **.ins** extension) containing fonts, classes, stylesheets, and other elements usually installed from a CTAN repository. Many of such packages have already been installed when the **texlive-dist** package was installed from Synaptic.

The **tlmgr** command is used to manage packages installed in TeXLive.

The **\usepackage{}** command is to TeX/LaTeX what **#include** is to C and C++. **\usepackage{}** adds commands and functionality to TeX that allow the document to correctly compile and achieve the desired result. For this example, the contents of the **inputenc** package are loaded, and with the **[utf8]** option, only support for UTF-8 encoded text is included and used by TeX. If a package is used without options, the entire package and all of its functionality is incorporated into TeX for use with the document being compiled.

For the **Quick Start** dialog, the **amsmath**, **amsfonts**, and **amssymb** packages are used by TeX/LaTeX. These packages were created by the American Mathematical Society. AMS is a major contributor to TeXLive and other TeX distributions. More information about the AMS and its use of TeX can be found at http://www.ams.org/publications/authors/authors.

The **\author{}** and **\title{}** statements define the author and title of the document. As these statements are part of the preamble, they only define the author and title, not type set it.

All of these statements make up the **preamble** of the document. The preamble of a TeX/LaTeX document defines variables and sets up the environment to be used when typesetting.

For this example, the title Mathematical Approach to Photography is a real academic project I did for a mathematical modelling class during the Spring 1989 semester at Valparaiso University (my alma mater).

In this project, I focused on the mathematics involved in conventional photography, be it calculating the focal length, aperture, shutter speed, or depth of field.

In the late 1980s (the time the academic project was developed), digital photography as we know it did not exist as a concept. Image scanners available at that time scanned in monochrome, and were connected either through SCSI interface, or through a proprietary board (that came with the scanner, and has long since been discontinued). Drivers for these scanners typically ran on DOS (and hence could potentially run on FreeDOS today).

I still have the original project and am working reviving it for this century.

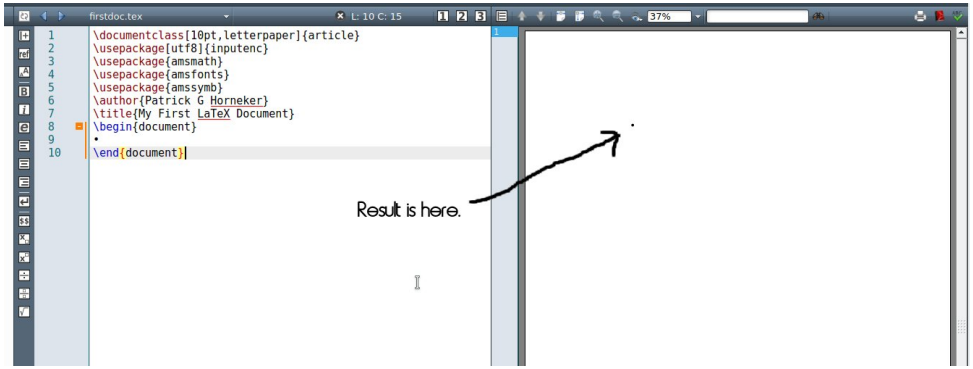
Now, we have the framework necessary to create our new document. You would want to change the title and author to whatever you want. For this example, I will change the title to “My First LaTeX Document”.

So now, we have:

```
\documentclass[10pt,letterpaper]{article}
\usepackage[utf8]{inputenc}
\usepackage{amsmath}
\usepackage{amsfonts}
\usepackage{amssymb}
\author{Patrick G Horneker}
\title{My First LaTeX Document}
```

\begin{document}
•
\end{document}

If we tried to **Quick Build** this, we will end up with a document with a dot located two inches from the left margin and two inches from the top margin. What gets typeset here is everything between **\begin{document}** and **\end{document}**.



By default, TeX documents are typeset with two inch margins on all sizes. The **geometry** package allows you to define your page margins. The margins were defined like this in order to accommodate for binding of TeX documents into books. Donald Knuth's The Art of Computer Programming was published this way.

Obviously, we want more than just a dot we can barely see. We need to place the cursor between the **\begin{document}** and **\end{document}** statements.

We could type some sample text to replace that dot. When typeset, the text will show up in the right panel in a 10 point Computer Modern font.

But typesetting is much more than that, and documents would be rather boring if we only used plain text.

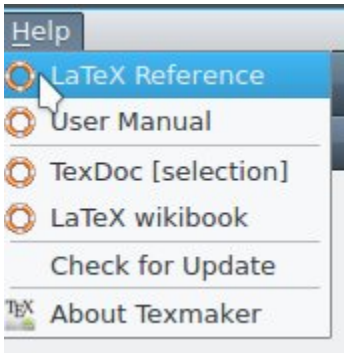
Unlike word processing, TeX and LaTeX have characters that have special meaning, namely:

&, $, #, %, _, {, }, ^, ~, and ****.

If you attempt to build TeX documents with these characters in plain text, you will get error messages instead of text output, unless you preface the character with a ****. For example, to typeset an ampersand, you would typeset **\&** to get a **&** on the printed output.

When it comes to spacing of characters, you only need to set one space to typeset one space, and one carriage return to typeset one carriage return (which happens to also mark the end of a typeset paragraph). If you attempt to typeset multiple spaces, only one space will actually be typeset.

TeX and LaTeX will automatically break lines where appropriate so you never have to worry about how a line of text will fit on a physical page when the line exceeds the length of the page.



Texmaker comes with some useful documentation. The **Help** menu has links for four useful documents, each of which opens up in a separate window.

The **LaTeX Reference** is a comprehensive guide to LaTeX commands, how they are used, and all available options that can be supplied to the commands.

The **User Manual** is just what we would expect of any user manual. It goes into detail on how Texmaker is used and configured.

TexDoc is a search engine for the TeX documentation.

And finally, **LaTeX wikibook** opens up the LaTeX section of Wikibook (in Firefox or whatever browser is the default for your PCLinuxOS installation).

Some Basic LaTeX Commands

Before we start typing our next document, there are some basic commands we need to know. TeX documents are structured into parts, chapters, sections, subsections, and subsubsections.

As explained earlier, there are characters that have special meanings in TeX. There are commands that allow you to enter such characters in your document.

There are LaTeX commands that perform basic text formatting such as bold, italic, and underline.

Let us go back to our TeX/LaTeX document and its basic structure.

All documents begin with this line: **\documentclass[options]{class}**

Where options and class are specified. The class is **required** for the command to be valid. For this example, we used the **article** class. Options are modifiers that tell TeX what to use. For our example, we used **10pt,letterpaper**. This tells TeX to typeset this document in 10 point Computer Modern on letter size paper.

Next, we have a **preamble**, which is used to setup the typesetting environment for the document.

\usepackage[utf8]{inputenc}
\usepackage{amsmath}
\usepackage{amsfonts}
\usepackage{amssymb}
\author{Patrick G Horneker}
\title{My First LaTeX Document}

...and finally the document itself.

\begin{document}

We enter the actual content and formatting commands here.

\end{document}

The first thing we need to do is to get the title to appear in the document. In our example, we need to get rid of the dot that is in the document. Simple highlight the dot and delete it. The cursor should now be where we need it to be to begin the document.

Commands in LaTeX typically begin with a forward slash, and tell TeX that the forward slash is the beginning of a command, or is to typeset one of the special characters (including another forward slash) in the document.

The **\maketitle** command is used to create the title of the document at the beginning of the document. You could type the statement in, or select **\maketitle** from the LaTeX menu to insert the **\maketitle** command.

If you ever programmed in C, C++, Java, or used **bash**, the forward slash character should be familiar to you. In bash, the forward slash is used to include spaces, and special characters in filenames with certain commands.

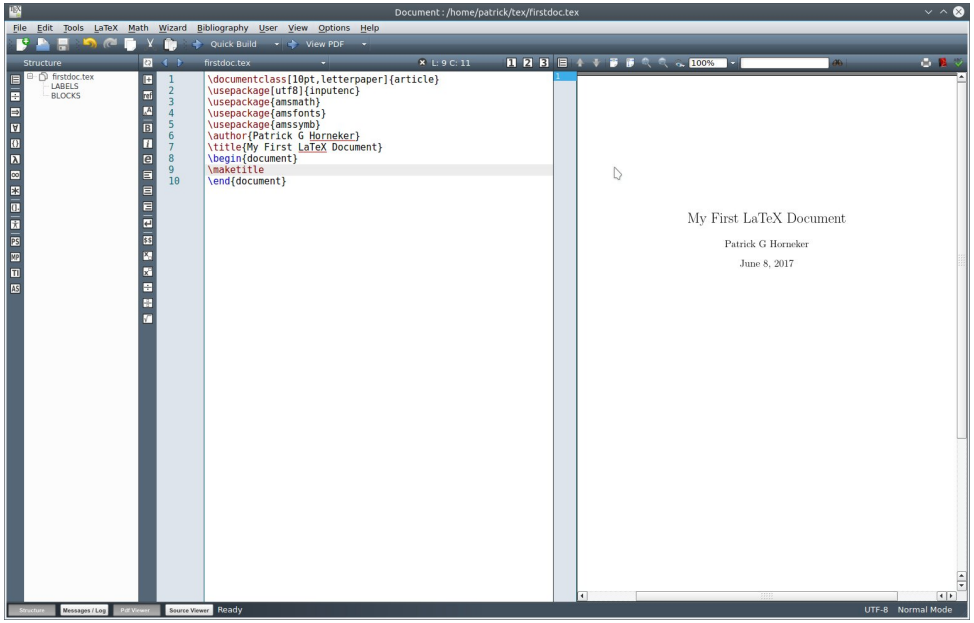
For example, **chdir ~/.wine/drive_c/Program\ Files\ \(x86\)** changes the directory to a directory called "**~/.wine/drive_c/Program Files (x86)**", which is the applications directory where 32-bit Windows applications are typically installed in WINE.

In C and C++, The **** is used in commands such as **printf()** to print special characters.

For example, **printf('\n');** simply prints a carriage return.

Now, let us save this file, then compile this document with the **Quick Build** command. (The right arrow button next to "Quick Build" does the job.) Look what just happened.

The preview panel automatically updated to reflect the state of the document we are editing. Here we have the title of the document and author, both of which were defined in the preamble. The date shown is the date this document was last built with TeX.



Typically, the **\maketitle** command is placed at the beginning of the document, just underneath the **\begin{document}** command. In fact, this is the only place where the **\maketitle** command should be in any TeX document.

In a **article** class document, we have **sections**, **subsections** and **subsubsections**. **Pages** and **chapters** only make sense for documents of the **book** class.

The **\section{}** command is used to label sections in your TeX/LaTeX document. These are automatically numbered when the document is compiled in TeX. The only requirement here is the actual title to be typed within the **{}**. If you do not want sections to be numbered, use the **\section*{}** command instead of **\section{}**.

For our example, we could type in:

\section{This is the First Section}

Followed by

\section{This is the Second Section}

So our document reads as follows (without the preamble for readability):

```
\begin{document}
\maketitle
   \section{This is the First Section}
   \section{This is the Second Section}
\end{document}
```

The indentation is not required here and is included for readability of the source document.

Texmaker gives you a bit of help here in the form of tooltips. As you type commands in, Texmaker gives you hints in the form of a popup menu. Selecting anything from the popup menu here will complete the command, making it easier to create your document.

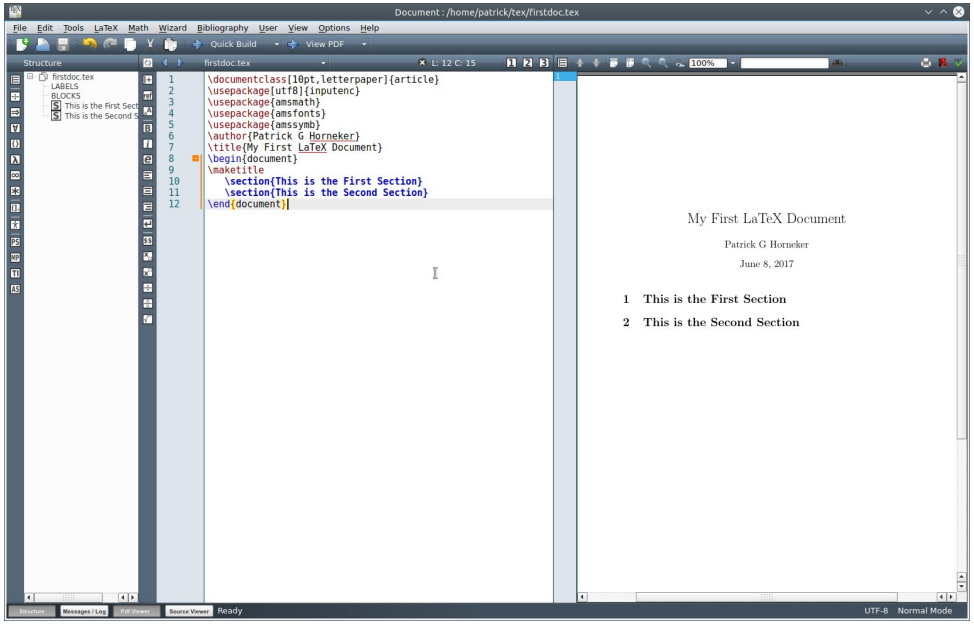
Also, if you indent, Texmaker will automatically sense that and will indent the next command you enter.

After you select **Section** from the popup menu, the dot inside the parameter is highlighted, and you simply type in the parameter in place of the popup.

Now, let us save and build the document.

On the preview panel, we can see the immediate changes to the document. But look what else happened. The left panel of Texmaker now has two new entries in the document structure, namely the sections we have created.

Texmaker shows us what the finished product will look like and how it is structured as we create the document. (This is why I prefer to use this product as opposed to Kile, which has a preview, but only when it is requested.)



Had we used **\section*{}** instead of **\section{}**, the numbers of the sections would not have appeared in the document.

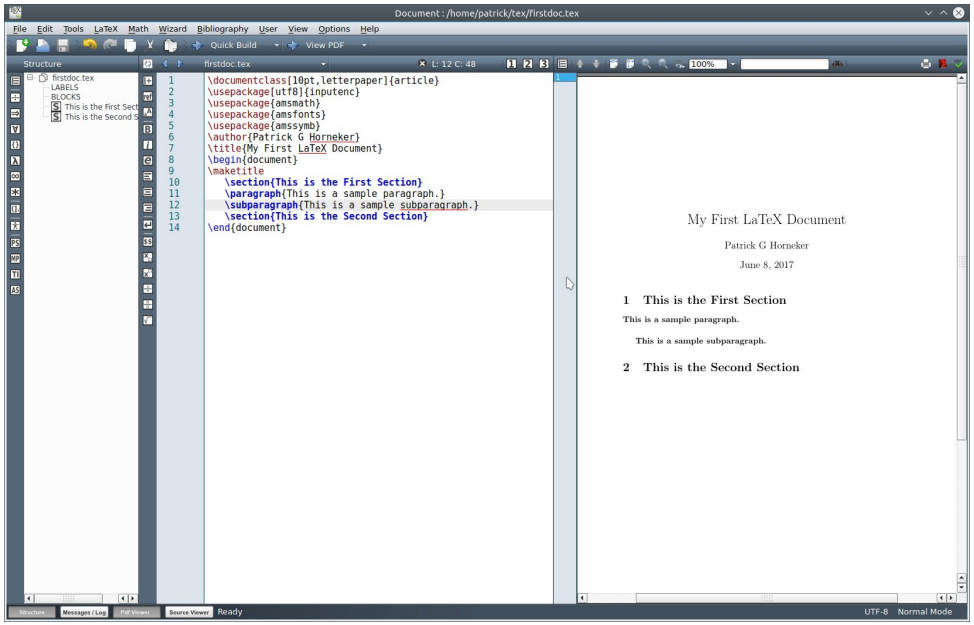
Entering plain text here will simply typeset the text from the left margin to the right margin. This will happen no matter what sections, and subsections are defined. The result may not be what you want the document to look like.

When typing paragraphs in LaTeX, it is a good practice to always begin with the **\paragraph{}** command. This allows you to organize each paragraph, and ensures the paragraph will fit into the structure of the document making the writing look as intended.

The **\subparagraph{}** command functions similar to the **\paragraph{}** command, only the text is indented a bit more than ordinary paragraphs. Let us insert a \paragraph{} and a \subparagraph{} command in our document and see what happens.

```
\begin{document}
\maketitle
   \section{This is the First Section}
   \paragraph{This is a sample paragraph}
   \subparagraph{This is a sample subparagraph}
   \section{This is the Second Section}
\end{document}
```

The difference between **\paragraph{}** and **\subparagraph{}** can be seen in the preview. Note the spacing and indentation between the elements in the document. TeX determines the layout of the document, so all you have to do is concentrate on the actual content.



If we were to print this out with **File → Print** selected from the menu (or simply type Control-P), what appears in the preview will print on your printer.

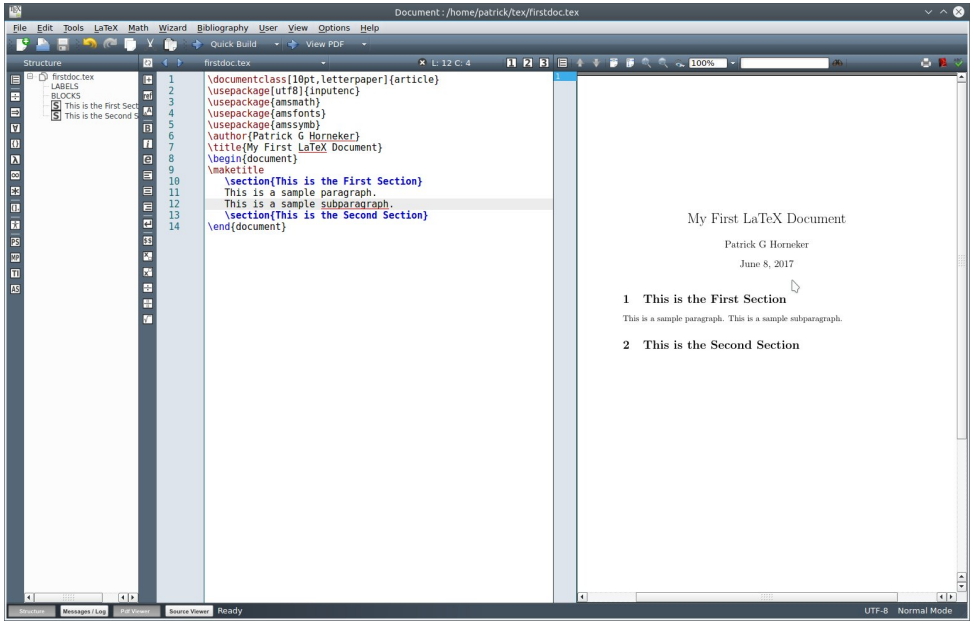
Selecting the LaTeX menu allows you to select commands you can insert into your document. The **\maketitle** command is one such command found in this menu.

The commands we have discussed can be inserted by selecting the command from the **LaTeX → Sectioning** menu.

Now, let us take out the **\paragraph{}** and **\subparagraph{}** commands leaving only the plain text. Then rebuild the document.

As we can see, the plain text blocks are simply typeset in linear fashion and there is no structure to speak of. This is why sectioning is so important in TeX.

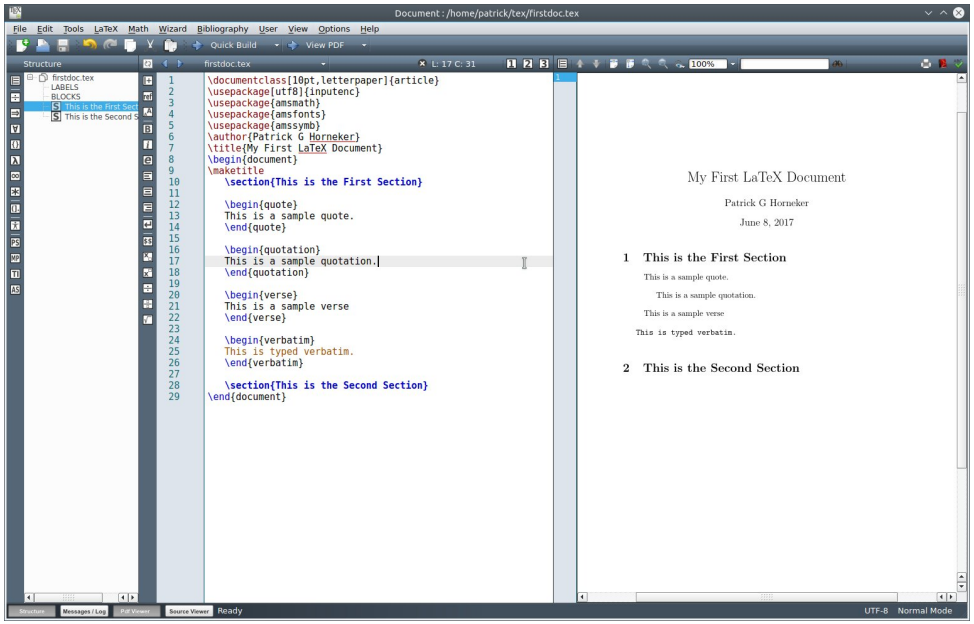
Suppose we want to show something as it is, or wanted to insert a quote from another writing. We would create an environment and place that block of text into the environment. Environments are to TeX what a function or procedure is to a computer program. TeX processes text in that environment according to the styles and formatting commands contained in that environment instead of the normal formatting TeX uses.



Environments begin with a **\begin{}** statement and end with a **\end{}** statement. The entire document is an environment itself as indicated by the **\begin{document}** and **\end{document}** statements.

I have included samples from four types of environments selectable from the **LaTeX → Environments** menu. You can also left flush, right flush, justify and center text in environments as well. As we can see TeX environments are quite flexible.

Note that the **verbatim** environment highlights the text in the editor in red to show what is being typeset verbatim. Also, the verbatim text in the preview is in Courier rather than Computer Modern font.



So far, we have seen the basics of how TeX documents are formed and formatted.

Typesetting Symbols

TeX and LaTeX have commands to typeset common symbols, characters that have special meaning in TeX, accents, arrows, greek letters and mathematical symbols. At the top of the next column is a list of commonly used symbols and the commands required to typeset them.

One would think that typesetting **** would give you a ****. Instead, **** will break up the current line at the point where the **** has been placed, and a new line will begin directly underneath the current line. If the **** happens at the bottom of the current page, the text will continue at the beginning of the first line on the next page. To keep this from happening, a ***** is substituted for the ****.

The **\pagebreak** command forces a new page to begin where this command is placed.

\today will insert the current day the document was compiled, and will change every time the document is recompiled.

Command	What is Typeset
\copyright, \textcircled{c}	©
\TeX, \LaTeX	The logos for TeX and LaTeX
\textasciicircum	^
\textasciitilde	~
\backslash	\
\textasteriskcentered	*
\textbar	|
\textexclamdown	¡
\textquestiondown	¿
\textdollar, \$	$
\textemdash	--
\textendash	---
\%	%
\textgreater	>
\textless	<
\leftarrow	←
\rightarrow	→
\textregistered, \textcircled{r}	®
\textbraceleft, \{	{
\textbraceright, \}	}
\&	&
\#	#
_	_
\ldots	…
\textbullet	Bullet normally used with list items

~ by itself prints a space, and **will not allow a line break** for the rest of the line of text being processed.

\textbf{} typesets the text contained in {} in boldface.
\textit{} italicizes the text contained in {}.
\textsl{} is similar to \textit{} only a slanted version of the normal typeface is used instead of the italic font.
\textsc{} typesets lowercase lettering in {} with shortened capital letters.
\textnormal{} typesets the text contained in {} normally.
\underline{} underlines text contained in {}.

TeX provides three basic fonts that work with any version of TeX, namely Roman, Sans Serif, and Typewriter.

By default, the Roman font is used. This is the Computer Modern font that comes with every TeX distribution.

\textrm{} typesets text in **{}** with Computer Modern explicitly.

The Sans Serif depends on the distribution. For TeXLive, the font generated is similar to Cantarell or Gill Sans, which should be familiar to most of us as many companies use this font in their graphics (be it advertising or in store graphics).

\textsf{} typesets text in **{}** with Sans Serif explicitly.

\texttt{} typesets text in **{}** with Typewriter (Courier) explicitly. The font here is very similar to the font found in typewriters of years past, mainly with the IBM Selectric and Wheelwriter series.

The IBM Wheelwriter 5 had a serial port that allowed the typewriter to be used as a daisy-wheel printer. To configure this under PCLinuxOS, you would need to have an actual serial port and install the **cups-serial** package from Synaptic, or connect a serial cable to a USB to Serial adapter (configured as **/dev/ttyUSB0**) and the print queue in CUPS configured with a Raw Queue as the Wheelwriter acts as a standard **teletype** printer (as only plain text is supported by the Wheelwriter).

This configuration is useful with Word Perfect running on FreeDOS in VirtualBox with the serial port configured to use the actual serial port.

This should be enough to get you started on typesetting (simple) documents. There is much more to cover for the TeX language.

For samples of what can be done, feel free to look at two documents I created:

* Online Dating
* Importance of Creative Commons

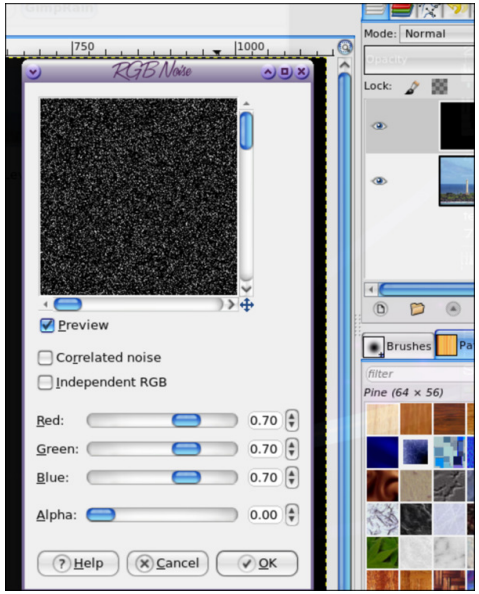
Better yet, search Google for Donald Knuth's The Art of Programming and The TeX Book (this is rather extensive reading) for more information. Also search Google for **TeX command reference** to get helpful documents on the TeX typesetting language.

GIMP Tutorial: Add Rain To A Photo

by Meemaw

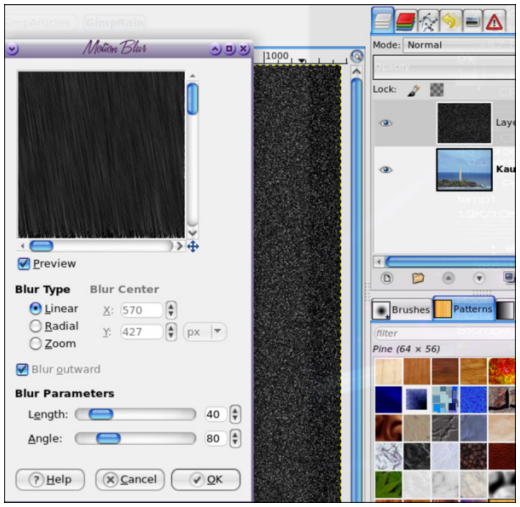
GIMP does all sorts of special effects! You can take an ordinary photo and add or subtract almost anything using GIMP. This month we're going to make it rain in an otherwise clear photo.

Choose the photo you would like to use. Create a new layer filled with black. This is the layer we'll use to make the rain. Go to **Filters > Noise > RGB Noise** and uncheck **Independent RGB** so that the three color sliders are linked. Click on any one of the sliders and drag it to the right so that the values of all the colors show a reading of about 0.70. The **Alpha** slider should be clear to the left. When you've selected your setting, click OK. The layer will now look kind of like it's snowing, but we have more to do.

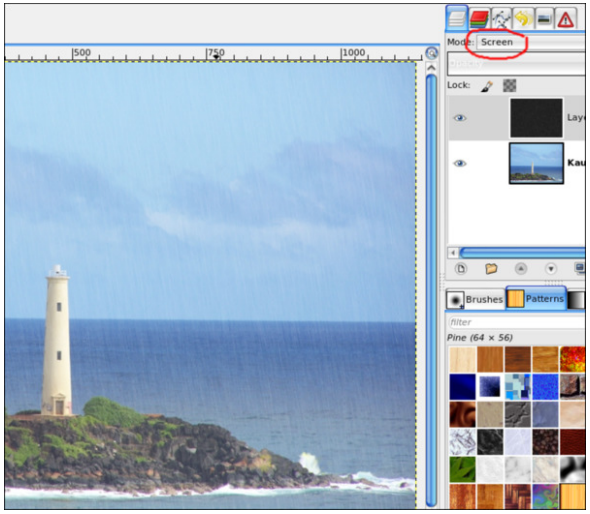


If you desire, you can use different settings for this step. Generally moving the sliders further to the right will make the rain appear heavier.

Ensuring that the speckled layer is selected, go to **Filters > Blur > Motion Blur** to open the **Motion Blur** dialog. Ensure that the **Blur Type** is set to **Linear**. Then you can adjust the **Length** and **Angle** parameters. I set the Length to forty and the Angle to eighty, but you can experiment with these settings to produce the result that you think looks best. Higher Length values will make it look like harder rain and different Angle settings will make it look like the rain is coming from the other direction or driven harder by the wind. Click OK when you're happy with the effect.

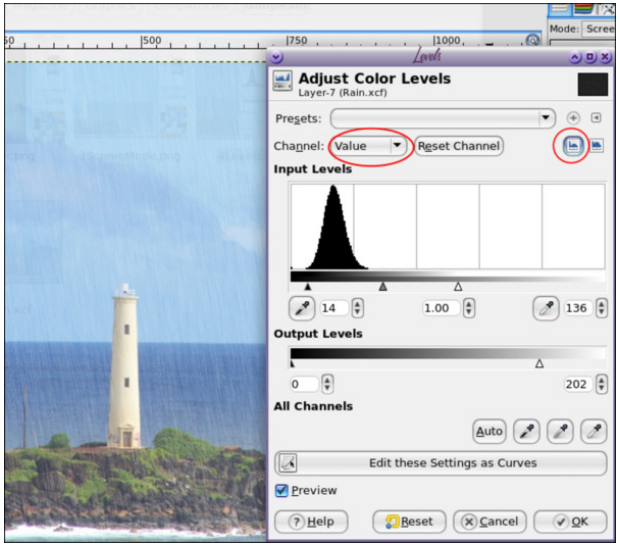


With the rain layer still selected, click on the **Mode** dropdown menu in the **Layers** palette (at the top) and change the Mode to **Screen**.



I know the rain is hard to see so far, but the next step will help considerably. Go to **Colors > Levels** and check that the **Linear Histogram** button is set (red circle at right) and that the **Channel dropdown** is set to **Value** (red oval at left).

In the **Input Levels** section you will see that there is a black peak in the histogram and three triangular drag handles beneath. First, drag the white handle across to the left until it is aligned with the right hand edge of the black peak. Now, drag the black handle to the right and check the effect on the image as you're doing this (make sure that the Preview checkbox is activated). You can move these handles back & forth until you get what you want. I actually had to move the white handle back to the right a lot to get the effect I wanted.



When you're happy with the effect, you can drag the white handle on the Output Levels slider a little to the left. This reduces the intensity of the fake rain and softens the effect. Click OK when you're happy with your result.

It may look the way you want it, but if not, there are a couple more things you can try. You might also go to **Filters > Blur > Gaussian Blur** and experiment with the Horizontal and Vertical values, and see if that makes any difference. It didn't seem to on mine.

One last thing you can do is select the **Eraser Tool** from the Toolbox and select a large soft brush in the Tool Options, and reduce the Opacity to 30%-40%. Using the large brush, stroke a few areas of the rain layer make it look more varied and natural (right).

Also, if you feel like you have messed up thoroughly and just want to start over, you can always just delete that layer in the Layers dialog. You will be left with your original picture, free to have another try. If you like it, be sure to Export it with a new file name so you still have your original photo.

This is the process to use if you want a lot of control over your creation. However, if you want rain quickly, you can use the GIMP filter Render Rain written by our friend dinasset over at GIMP Scripts. It is a one-step script that does the rain quickly (next page).

The PCLinuxOS Magazine

Created with Scribus

Screenshot Showcase

Posted by Meemaw on June 2, 2017, running Xfce.

Tip Top Tips: Changing The Mouse Cursor Color

Editor's Note: *Tip Top Tips is a monthly column in The PCLinuxOS Magazine. Each month, we will feature – and possibly even expand upon – one tip from the PCLinuxOS forum. The magazine will not accept independent tip submissions specifically intended for inclusion in the Tip Top Tips column. Rather, if you have a tip, share it in the PCLinuxOS forum's "Tips & Tricks" section. Your tip just may be selected for publication in The PCLinuxOS Magazine.*

This month's tip comes from PCLinuxOS forum member **Ramchu**.

I have my mouse cursor set to Oxygen Blue in System Settings, but when I would open FireFox the mouse cursor color would change white, just one of those little things, that really bugged me.

This will keep the mouse cursor the same over all applications.

First go into **System Settings > Appearance > Workspace Theme > Cursor Theme**, then pick a color theme and click **Apply**.

Open your file manager. I use the Dolphin File Manager, if you're using a different file manager, getting to the needed file may be a little different.

Now, click on **Control > Hidden Files > .kde > share > config > kcminputrc**, and open it in a simple text editor. Edit the line **cursorTheme=**. To change the size, edit the line **cursorSize=**. Then click **File > Save** to save the settings. **Logout and back in** for the settings to take effect.

I wanted the mouse cursor theme to be Oxygen Blue across all applications.

This is what the file looks like on my system:

[Mouse]
Acceleration=2.5000000000000004
MouseButtonMapping=RightHanded
ReverseScrollPolarity=false
Threshold=4
cursorSize=0
cursorTheme=Oxygen_Blue

Rudge added a thread here with a similar tip.

Firefox on KDE all but REFUSES to show a black mouse pointer within the application.

On KDE, Firefox's default mouse pointer is white and within the application, it won't look at the mouse pointer icons that you pick in "System settings." It just stays white even if every other app shows a black pointer.

Here's what you have to do to get a black pointer in Firefox.

Open **System settings, Application Style, GNOME Application Style**.

To the right of "Cursor themes," pick the theme that includes the pointer you want and click "Apply" at the bottom right.

You might have to reboot the desktop for it to take affect but, presto, you will now have a black pointer in Firefox.

This works for Thunderbird, too.

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, rot 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.

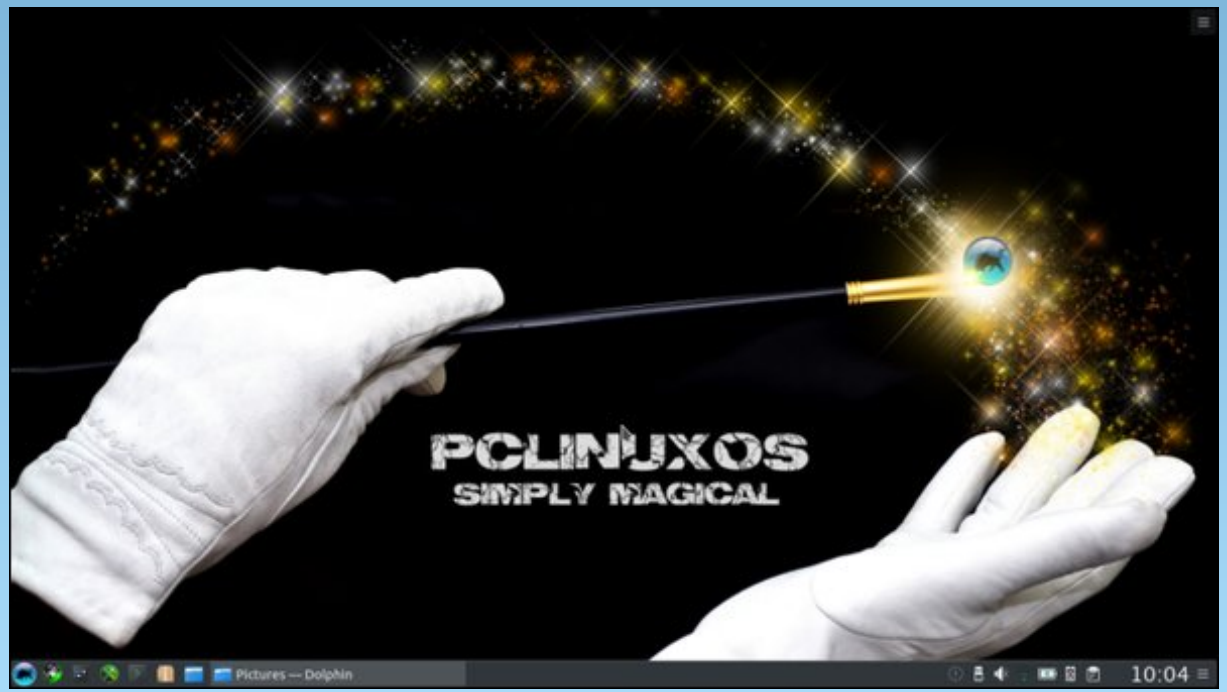
4. No representations, warranties or guarantees whatsoever are made as to the accuracy, adequacy, reliability, completeness, suitability, or applicability of the information to a particular situation. All trademarks are the property of their respective owners.

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.

Screenshot Showcase



Posted by luikki on June 13, 2017, running KDE.

PCLinuxOS Puzzled Partitions

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

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 prefilled for you. You cannot change these numbers in the course of the game.

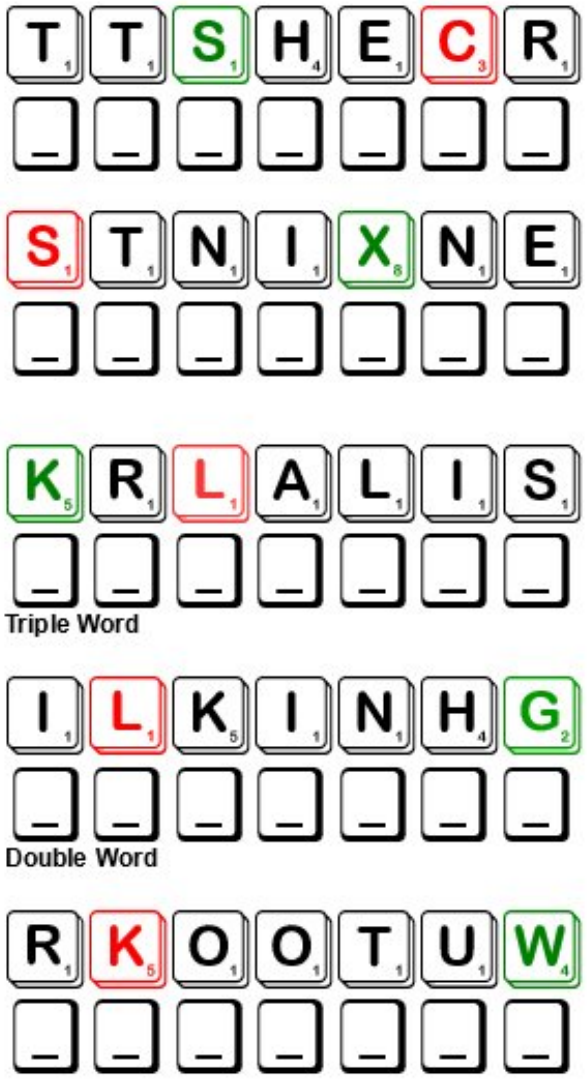
Each column must contain all of the numbers 1 through 9 and no two numbers in the same column of a Sudoku puzzle can be the same. Each row must contain all of the numbers 1 through 9 and no two numbers in the same row of a Sudoku puzzle can be the same.

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.

SCRAPPLER RULES:

1. 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.
2. Red letters are scored double points. Green letters are scored triple points.
3. 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.
4. 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.
5. In case you are having difficulty seeing the point value on the letter tiles, here is a 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
6. Optionally, a time limit of 60 minutes should apply to the game, averaging to 12 minutes per letter tile set.
7. Have fun! It's only a game!

Download Puzzle Solutions Here



Possible score 222 average score 155.

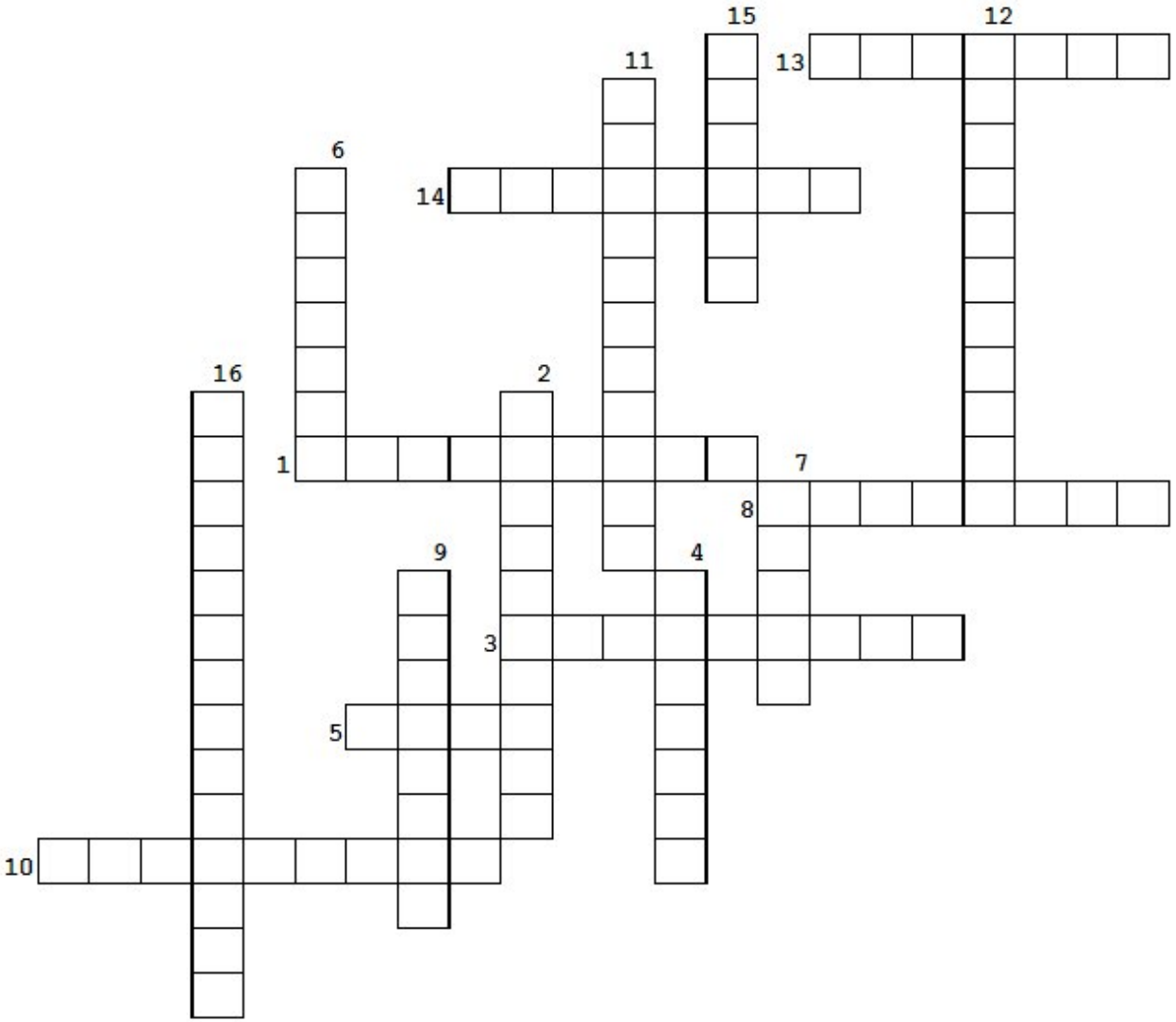
PCLinuxOS Word Find: July 2017 Exercise

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

Aerobics
Athletics
Balance
Basketball
Biking
Boxing
Breathe
Cardiovascular
Climb
Coach
Competition
Conditioning
Crunches
Curling
Cycling
Dancing
Dumbbells
Endurance
Energy
Exercise
Gymnasium
Hiking
Ice-skating
Jogging
Judo
Massage
Physiotherapy
Punch bag
Punt
Resistance
Rowing
Running
Shape up
Skating
Skiing
Skill
Sportsmanship
Stretching
Swimming
Tennis
Therapeutic
Treadmill
Weights
Workout
Yoga

Download Puzzle Solutions Here

Exercise Crossword

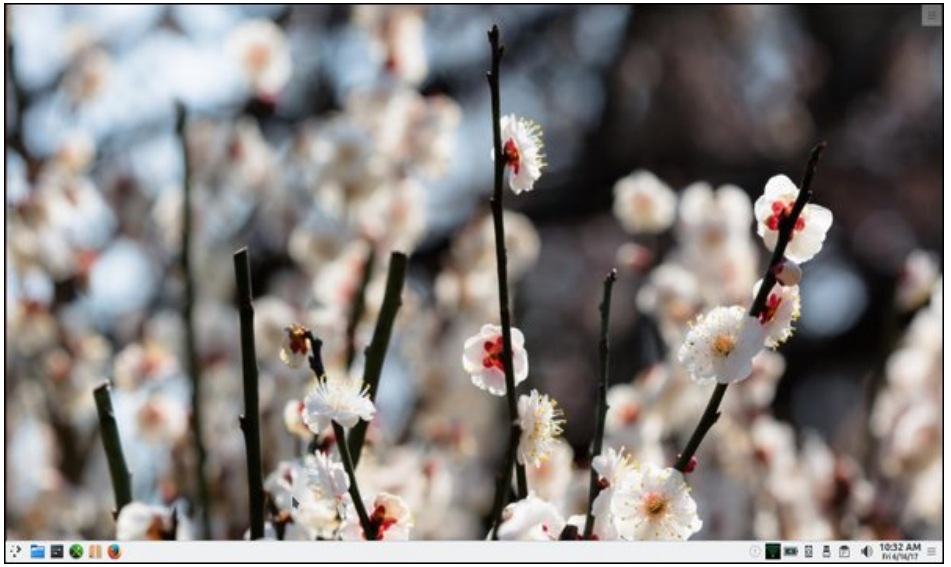


1. Usually where the school basketball games are held.
2. This sport is great for tall players
3. You can walk on this machine inside
4. One of the simplest & most beneficial exercises
5. A discipline which includes breath control, meditation, and specific bodily postures
6. Put the music on and have fun _____
7. Use a rock wall or a mountain
8. An exercise to strengthen your abs
9. Exercise designed to strengthen heart & lungs
10. An event demanding great physical stamina
11. Contest between two teams
12. Related to the healing of disease
13. You can do this on ice or on the sidewalk
14. Hope the water is warm for this
15. Fist-fighting with padded gloves
16. Relating to the heart & blood vessels

Download Puzzle Solutions Here

More Screenshot Showcase

Posted by francesco_bat on June 4, 2017, running Openbox.



Posted by exploder on June 16, 2017, running KDE.

Posted by Crow on June 21, 2017, running KDE.

Posted by Analogue Man on June 4, 2017, running KDE.