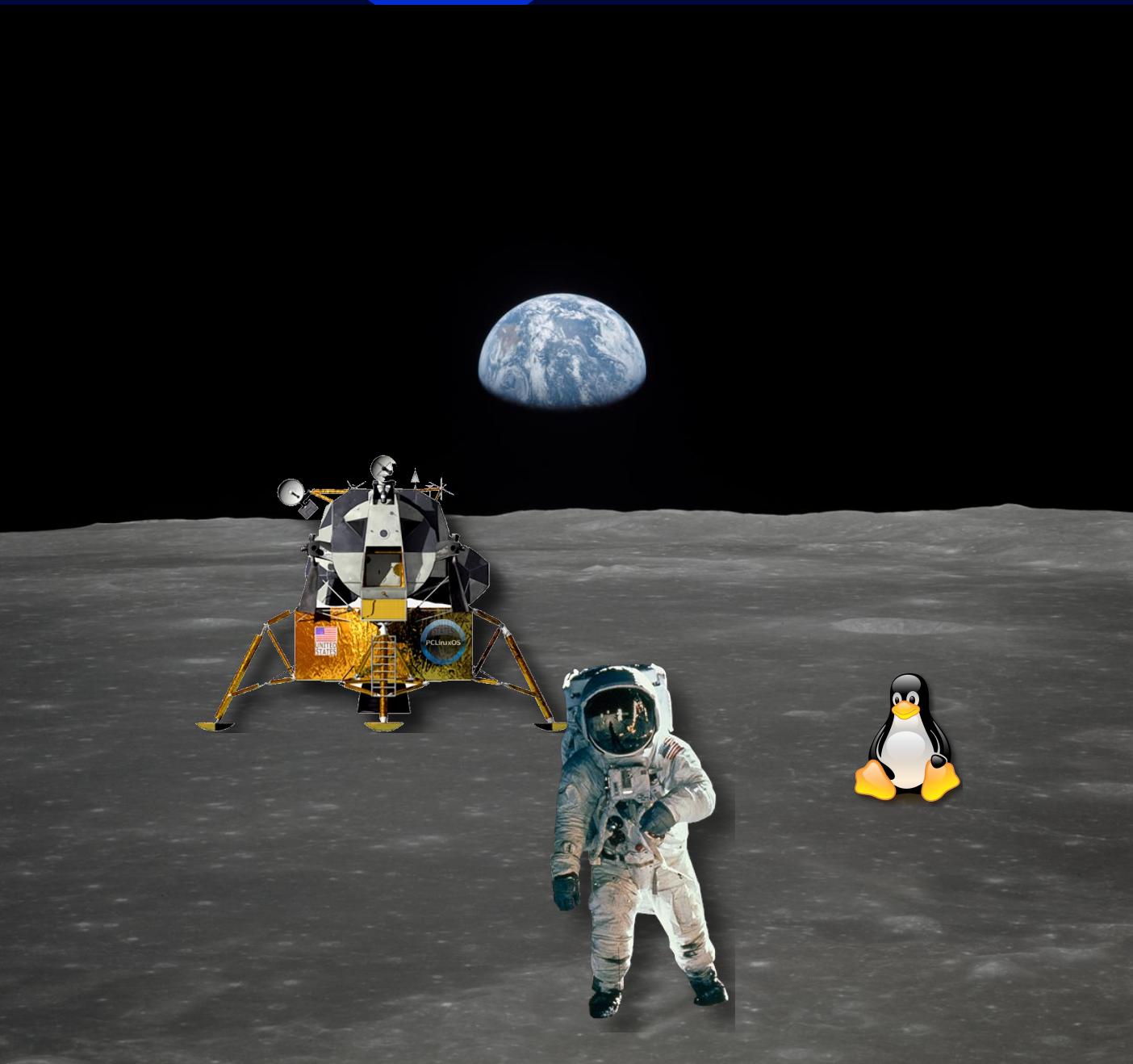


# The PCLinuxOS magazine

Volume 151

August, 2019



***Short Topix: Vulnerability Discovered Masquerading As A Gnome Extension***

***Casual Python, Part 7***

***Inkscape Tutorial: Rubber Stamp***

***De-Googling Yourself, Part 4***

***PCLinuxOS Family Member Spotlight: MarekTux***

***Repo Review: PhotoFlare***

***Cutting The Cord In 2019***

***The Ruby Programming Language: Blocks, Modules & Other Interesting Things***

***ms\_meme's Nook: Texstar Has High Hopes***

***PCLinuxOS Puzzled Partitions***

***And more inside ...***

# In This Issue ...

- 3 From The Chief Editor's Desk ...**
- 4 Screenshot Showcase**
- 5 Cutting The Cord In 2019**
- 10 Inkscape Tutorial: Rubber Stamp**
- 11 Screenshot Showcase**
- 12 ms\_meme's Nook: Texstar Has High Hopes**
- 13 PCLinuxOS Recipe Corner: Grilled Italian Chicken**
- 14 De-Googling Yourself, Part 4**
- 19 Screenshot Showcase**
- 20 Casual Python, Part 7**
- 29 Screenshot Showcase**
- 30 Repo Review: PhotoFlare**
- 31 PCLinuxOS Family Member Spotlight: MarekTux**
- 32 Screenshot Showcase**
- 33 Short Topix:  
Vulnerability Discovered Masquerading As Gnome Extension**
- 37 The Ruby Programming Language:  
Blocks, Modules And Other Interesting Things**
- 45 Screenshot Showcase**
- 46 ms\_meme's Nook: It's Linux Time**
- 47 PCLinuxOS Recipe Corner Bonus:  
Mediterranean Shrimp Linguine**
- 48 PCLinuxOS Puzzled Partitions**
- 52 More Screenshot Showcase**

The **PCLinuxOS** magazine

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

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

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

This release was made possible by the following volunteers:

**Chief Editor:** Paul Arnote (parnote)

**Assistant Editor:** Meemaw

**Artwork:** ms\_meme, Meemaw

**Magazine Layout:** Paul Arnote, Meemaw, ms\_meme

**HTML Layout:** YouCanToo

**Staff:**

ms\_meme

Meemaw

Gary L. Ratliff, Sr.

Daniel Mei -Wilhelm

daiashi

Cg\_Boy

YouCanToo

Pete Kelly

Smileeb

Alessandro Ebersol

**Contributors:**

MarekTux

The PCLinuxOS Magazine is released under the Creative Commons Attribution-NonCommercial-Share-Alike 3.0 Unported license. Some rights are reserved.  
Copyright © 2019.



# **From The Chief Editor's Desk ...**

Even though I was just shy of nine years old, I can remember it like it was yesterday. I'm sure many of you remember it, too.

I'm talking about the first landing of man on the moon.

July 20, 1969 at 20:17 UTC, the world saw the first two human beings land on a celestial body that wasn't Earth. They landed with about 50 seconds of fuel remaining.

"Houston, Tranquility Base here. The Eagle has landed," said mission commander Neil Armstrong. The change of call signs from Eagle to Tranquility Base signified that the landing had been successfully completed.

On July 21, 1969 at 02:56 UTC, the world saw those same two human beings take their first steps on a celestial body that wasn't Earth.

As an eight-almost-nine year old, I remember being allowed to stay up late to watch this historic event. Estimates of between 500 million people and 650 million people from around the world watched live as Neil Armstrong descended the ladder from the lunar module. The actual number varies, depending on your reference source

With that first step on the lunar surface, Armstrong spoke his immortal [words](#). "That's one small step for [a] man, one giant leap for mankind." Buzz Aldrin joined Armstrong 19 minutes later, as the second human to walk on the moon. His words to describe his view on the surface of the moon were simple and succinct, but strong. He uttered two words: "Magnificent desolation." The two conducted and completed experiments on the surface of the moon during the 2.5 hour lunar EVA.

They gathered up 47.5 lbs (21.55 Kg) of samples of lunar rocks and soil during their activities on the lunar surface. After returning to the lunar module, they did some housekeeping and got seven hours of sleep. Upon awakening, they launched from the lunar surface to rendezvous with the command module, Columbia, piloted by Michael Collins and in lunar orbit.

The three successfully returned to Earth on July 24, 1969.

That was 50 years ago. It's not hard to find accurate accounts of the feat on the web. Wikipedia has a pretty thorough [entry](#) about it, as well.



What's especially sad is that we have had whole generations of people who have not experienced another human traveling to and walking on another world. We have generations of people who have never seen humans leave low-Earth orbit. They have not gotten the thrill of seeing humans expand their horizons and push the boundaries of what can be accomplished. For them, humans walking on the moon is about as real as the story in any Arthur C. Clarke science fiction novel.

Hopefully, this should all change soon. President Donald Trump (it doesn't matter if you like him or not, or what you think of him) and Vice-President Mike Pence have set an ambitious goal of returning humans to the moon by 2024. That's only five years away. This time, the goal is to return to the moon to stay.

Back in the days of Apollo 11, computers were still in their infancy. Your old flip style cell phone has/had more computing power than the onboard computers of the spacecraft of that era. Heck, the first IBM PC wouldn't be released for another 12 years.

More than likely, Linux will play a role in returning humans to manned spaceflight, and in our voyage to our celestial neighbors. It was reported a few years ago that a custom version of Debian has already replaced the Windows XP based computers on the International Space Station to provide better reliability. Many other systems on the ISS already run RedHat Linux and Scientific Linux. NASA has long embraced free and open source software. Elon Musk has tweeted that Dragon's code is written in C++ on Linux. Search for "linux and space travel" and see for yourself.

There is no doubt that Linux will play an important role in returning humans to space, blazing the way to historic endeavors, like putting humans on Mars. What's even better, my kids will experience the thrill that I experienced as a kid of seeing humans walking on our nearby celestial neighbors. Plus, all of those previous generations around the world will

also share in that thrill, the one that they missed out on when we switched to only going into low-Earth orbit.

Yes, it's exciting times, to be sure.

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



## Screenshot Showcase



Posted by dpascal, July 5, 2019, running Mate.

# **Cutting The Cord In 2019**

---

by Paul Arnote (parnote)

Over the past several years, “cord cutting” when it comes to cable or satellite TV has become quite popular. With cable TV, subscribers are getting fed up with a pretty much all-or-none subscription plan, forced to pay for channels they don’t watch and have no interest in. It’s a similar situation for satellite TV customers, too. In my neck of the woods, the typical cable bill runs somewhere between \$100 and \$200 per month (but that includes cable TV, high speed internet, and a VoIP landline phone connection).

The companies that run cable and satellite TV services make obscene profits, and the prices for the services keep going up and up, year after year. Nationwide, the average cable bill is \$107 per month (2018 data), and the cost for cable television has gone up 50 percent since 2010. It’s enough to make any reasonable person wonder if the prices keep going up each year (or is at least one of the reasons) as more and more customers cut the cord, and the companies keep raising the prices to meet projected earnings. In the end, only the stockholders of those companies are kept happy, while the customers keep fleeing the ever increasing prices and fleecing.



There are many paid options, and most people are very familiar with them. Streaming services, such as Hulu, Netflix, Amazon Prime, Sling and others offer popular alternatives to cable or satellite TV. But these cost money that many don’t want – or can’t afford – to pay.

In the end, paying for two or three of these streaming services is still cheaper than paying for the most basic cable or satellite plans out there. Plus, you get the option of viewing the programming you want, when you want, on YOUR schedule. You’ll likely see significant savings, as well. For example, subscribing to just Hulu and Netflix will cost you about \$200 a year. That sure beats the \$200 a month you’ll spend on some cable plans!

By all means, if you are an Amazon Prime member (I am not, but I know a lot of people are to take advantage of the free two-day shipping), take advantage of the free access to Amazon Prime Video. No one would blame you. It’s Amazon’s streaming service that comes as a “perk” for being an Amazon Prime member.

But, what’s a person to do when they want to NOT have to pay for television programming content? Fortunately, there are several options. This is what I plan to explore here. There’s another way to accomplish this, without having to install Kodi or MythTV. It will involve a little more work than sitting back in your recliner and changing the channel on your cable or satellite box, but it is doable. And, it isn’t that difficult.

Let me preface all of this by stating that the services are not listed in any particular order. You’ll also probably have more success if you are willing to be a bit more flexible with your browser choices. I’m a diehard Firefox user, and I really hate to fire up Google Chrome (but this article isn’t about all of the reasons I really dislike Google Chrome, so we won’t even go into any of that). However, some of these services won’t play in Firefox, but end up playing exceptionally well in Google Chrome. Many of these services play well in either browser. If you’re planning on “casting” your streaming media to a Google Chromecast, then Google Chrome will be pretty much a requirement.

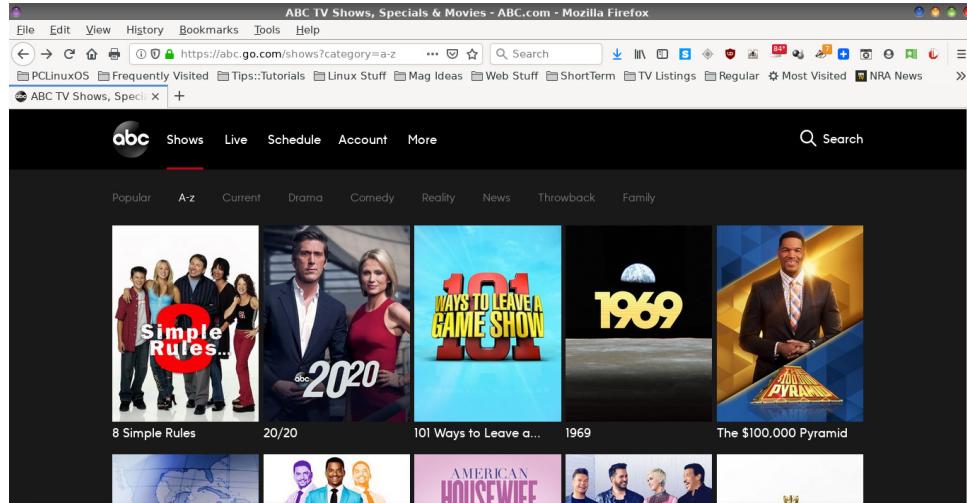
We’ll focus mainly on services that will play well on your desktop or laptop computer. Some services are designed primarily for viewing on mobile platforms (like Android or iOS), and desktop/laptop users find it a struggle to view them, if at all. Some services will require you to sign up for an account, while others will allow you to watch programming with or without an account.

Finally, you may want to use a VPN. In fact, it may be necessary, like when you're trying to view BBC programming but you're not in Europe. At the very least, you may want to anonymize your viewing activity to some degree, and a VPN could very well help you do just that.

## OTA

If you cut the cord to your cable or satellite service, you will most likely want to receive the programming of your local channels. You will most likely want to at least receive the local news, weather and sports reports. If you're watching on your TV set, all of this can be achieved by receiving OTA (over the air) channels via an antenna setup. But if you're wanting to stream local programming, don't despair. For local news, weather and sports, it's as easy as connecting to your local broadcaster's website. Where I live, you can stream entire newscasts, as well as pick and choose which news stories you are interested in viewing, from no fewer than four different broadcasters.

Plus, you're not restricted to local news from just your area. Let's say you live in Bangor, Maine. You have family that lives in Atlanta, Georgia. You're planning a visit in three weeks to see them. You can pull up the local newscasts from the Atlanta broadcasters to stream on your computer.



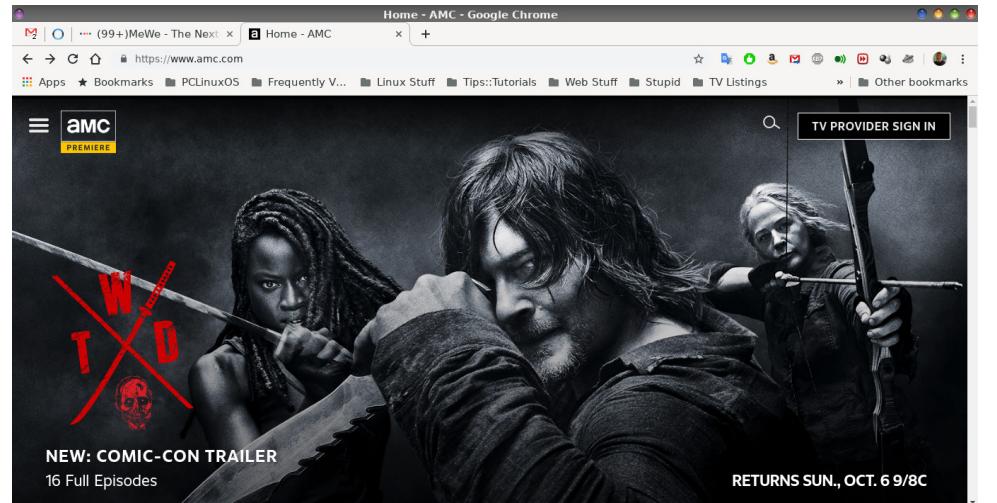
Being able to stream local independent stations may be a rather hit-or-miss proposition. But if the local station you want to stream is affiliated with a major network, you can also pull up the website for that major network to stream shows from their programming lineup. [ABC](#), [CBS](#), [NBC](#), [FOX](#) and [PBS](#) (you will have to provide your ZIP code for the PBS site) are the major over the air broadcast

networks in the U.S., and you can stream their programming on your computer with relative ease. It's pretty much the same with the minor broadcast networks, like [The CW](#) (and don't skip the sub-site [CW Seed](#), where you'll find some rather unique programming, like animated shows from DC Comics). The exception here is CBS. It isn't clear which items are free to stream, and which items require CBS All Access, CBS's paid streaming service that offers such programming as Star Trek: Discovery and the new, upcoming Star Trek series, Picard. They seem to spend an inordinate amount of time and effort to get you to sign up for a CBS All Access account.

## Cable Channels

Just because you give up your cable or satellite service provider, that doesn't mean that you also have to give up your favorite cable/satellite access only programming choices. Most of the cable network providers also have the ability to stream their programming directly on their website(s).

Some, like [AMC](#) and [History](#), have limited free offerings without signing in through a cable provider. This isn't unusual to find, so don't be too disappointed. They don't always make it easy for cord cutters to access their programming for free,



and I suspect it has something to do with deals made with cable and satellite service providers. Even though you may be locked out one time, it's worth going back again and again. Periodically, they unlock content that you might want to watch. Keep in mind that if you see a little padlock or key, or if you don't see the words "Free Episode" next to a description, you will most likely have to sign in

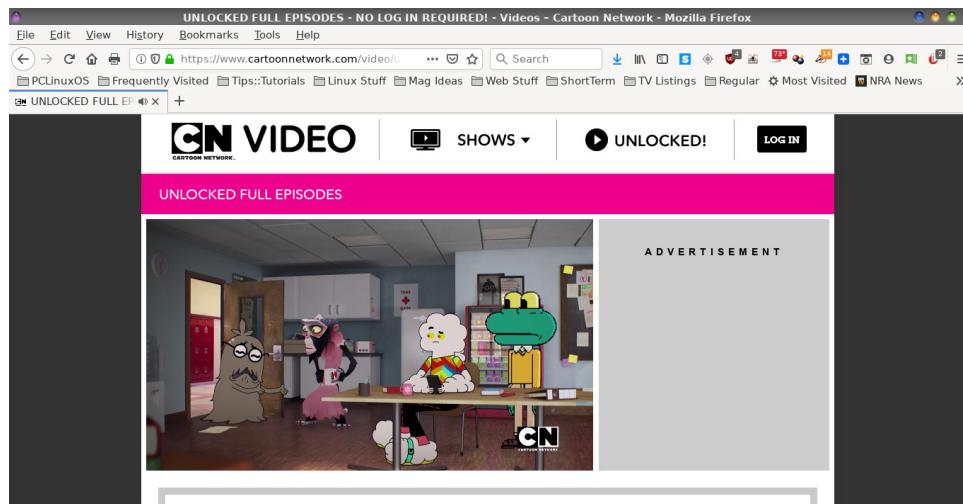
with the credentials for your cable or satellite service provider to view that content.

One thing to be on the lookout for is a button (usually somewhere at the top of the page) that says "Watch Live" or "Live Streaming." This will sometimes allow you to watch the programming that the networks are currently airing. Other times, it will lead to a page containing the live feed, asking you to log in with your cable or satellite service provider before allowing you to watch. It's another trip to the craps table at the casino.

While searching out programming choices, you're likely to be surprised about whom owns who. AMC Networks, for example, include AMC, [BBC America](#), [IFC](#), and [Sundance](#), among others. NBC owns [USA](#), [Syfy](#), [Telemundo](#), [E!](#), [Universal Kids](#) and [Oxygen](#). The [Discovery](#) Networks include such programming choices as [Animal Planet](#), [American Heroes Channel](#), [TLC](#), [Science Channel](#), [DIY](#), and several others. I think you get the idea.

And just because, for example, AHC (American Heroes Channel) is owned by Discovery, don't expect that because a lot of stuff is "locked" content on Discovery that things will be the same on AHC. Another good example is on IFC. On the "mothership" site, AMC, a lot of content is "locked" and requires logging in with your cable or satellite service provider to view. However, on the IFC site, this is not necessarily so, as there is a significantly higher percentage of "unlocked" programs on the IFC site than on the AMC site.

On many of the "cable channel" sites, the extras and trailers are usually free to watch. Don't overlook them, as they can be quite informative and entertaining. While the Discovery Channel seems to be locked up tighter than Fort Knox, most



(if not all) of the "extras" are free to view without logging in through your cable or satellite service provider.

The cable news channels ([Fox News](#), [CNN](#), [MSNBC](#), etc.) almost always have plenty of news content. Usually, you can either watch live, or hand-pick which news stories you want to watch. The same goes for cable channels that present the weather, like [The Weather Channel](#).

You can keep the kids entertained with not only wholesome and educational shows from PBS and [PBS Kids](#), but you can also stream unlocked episodes of animated kid-friendly programming from the [Cartoon Network](#).

Here's something I found to be odd. WGN America is carried on just about every cable and satellite service provider's lineup available in the U.S. This is the broadcast that is sent out to the rest of the country. There is a WGN-9 in Chicago, and that broadcast **used to be** sent out to cable and satellite service providers everywhere. On it, you could watch Cubs games, watch the 9 p.m. news, and everything else, just like everyone else in Chicago. But WGN America doesn't broadcast Cubs games, or anything else that's "Chicago-centric." Instead, it's a lineup of mostly heavily recycled reruns. Oddly, there is no provision on the WGN America site to live stream that channel. But, you can [live stream](#) the local Chicago feed from WGN-9. The Cubs games are blacked out on the live stream of the local WGN-9, for those looking to catch a Cubs game ... or two ... or three ... or more.

Don't expect to find any of the programming from the premium cable channels lurking out there for free. HBO, Cinemax, Showtime, Starz, The Movie Channel, The Disney Channel, etc. all have their own streaming services that are available for a monthly subscription. You might be able to sign up for a free trial, but be certain that you cancel your subscription before the trial period is up. Otherwise, you'll end up with an unexpected charge.

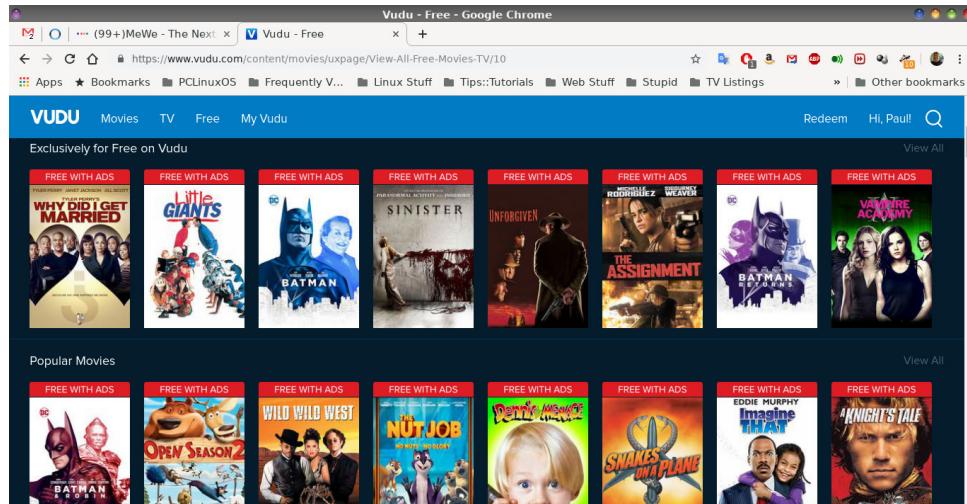
With everybody and their brother starting up new subscription model streaming services, subscribing to them all will cost you more than a monthly cable subscription. Most streaming services want between \$5 and \$10 a month just to stream their programming. This can add up very quickly, which is why I've spent most of my time focusing on free streaming services.

### Internet Only Streamers

Luckily for most cord cutters, you have some additional resources for streaming your television programming. These are services you'll probably rarely (if ever) see on your cable or satellite service provider's channel lineup.

This is also where things get most interesting for cord cutters. These internet only streaming services offer a wide selection of movies **and** television shows. Are you going to have access to the latest, most recent movies or television shows? Nope. But if you can't find something there to watch – or re-watch – then there is probably something wrong with you.

Before anyone says anything, I left one popular service off the list. That service is Sony Crackle. I cannot get it to play on PCLinuxOS, in any browser I've tried. So, if it won't play on PCLinuxOS, it gets omitted. There are a lot of people who praise Crackle online, so consider that to be yet another possibility if you are streaming on a mobile device.



**Vudu** not only has a healthy selection of movies, but it also has a healthy selection of television shows. Granted, most of the free offerings are older, having come out quite a few years ago. Still, there's a good chance you'll find something to watch. Maybe it was a movie that you always intended to see, but for some reason never did. Maybe it's a movie you want to watch again. The free movies are supported by periodic ads. Ad-free versions of the movies and newer movies are available, for a price. You will also need to sign up for an account. Vudu will not playback programming content in Firefox, so you'll have to use Google Chrome to view the content.

Vudu is also home to all the digital copies you see touted on DVDs and BluRays sold at Walmart and other retailers. Redeem the code for the digital copy that is enclosed with your qualifying DVD or BluRay disc (usually on some piece of paper inserted into the disc packaging), and you'll be able to access those movies from wherever you may happen to be, as long as you have an internet connection. My new Vizio Smart TV has connected access to Vudu, and many

other smart TVs do as well. It makes it extra convenient to pull up the list of movies and play them on the TV. Plus, those movies that you own a digital copy of are shown in high-definition, without any ads. We covered Vudu in the January 2019 issue of The PCLinuxOS Magazine.

**Tubi** is another streaming service that we have previously covered in The PCLinuxOS Magazine, back in the May, 2016 issue. With Tubi, you get access to thousands of titles, ranging from movies to television shows. You can search for titles by name or genre. Just like with Vudu, most of the available titles are a few years old, but there is a very wide variety of titles available. Tubi plays fine in either Firefox or Google Chrome.

**Retrovision Classic Movies** might be your destination if you really like older, classic movies and TV shows. Browsing through their offerings, I saw one movie from 1925 (Phantom of the Opera). They have a wide variety of television shows from the 1950s and 1960s, as well as a lot of movies mostly from the 1930s through the 1950s. I did see a smattering of movies from the 1970s and 1980s, though. There are the first five episodes of the Beverly Hillbillies, quite a few Ozzie & Harriet episodes, and even some old Popeye cartoons for the kids. Retrovision Classic Movies plays equally well in either Firefox or Google Chrome.

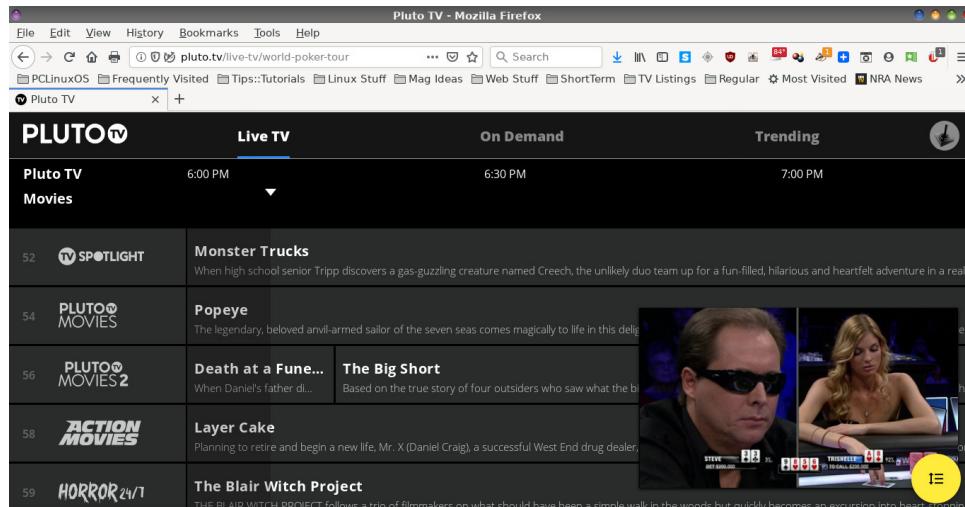
**Popcornflix** is another free streaming site that has tons of movies (including the first three Star Trek movies), and features about 100 different TV series. The movies are sorted by genre, including a "Staff Picks" section containing a collection of the highest rated movies and TV shows. You don't even have to sign up. Just visit and start viewing your selections! Popcornflix plays just fine in either Firefox or Google Chrome.

**Pluto TV** is a free streaming site with a guide that resembles the guide on many of cable and satellite services. They feature a collection of movies and television shows. You should be able to easily find something to watch amongst its offerings. Do expect the programming to experience slight interruptions for advertising, but it isn't extreme. Look at it this way: your programming on cable and satellite is also interrupted by advertising, and you pay for that! I first learned about Pluto TV when I was exploring my new smart TV, which had a built in app for it. The only downside is that you have to watch the programs when they are aired. There is no time shifting. Pluto TV plays well in either Firefox or Google Chrome.

**XUMO** is another free streaming site that is very similar to Pluto TV. It has a guide presentation of the programs that are currently playing. But what's different about XUMO is that it also has an On-Demand section. There is no need to sign up, sign in, or create an account. Just visit their website and start streaming! Here is a tip, though. You will be unable to view the XUMO guide unless you make the

## Cutting The Cord In 2019

web page full. On my Xfce desktop with Firefox, I do that by pressing F11. XUMO plays very well in either Firefox or Google Chrome.



**SnagFilms** is a streaming site that I can best describe as eclectic. Browsing through its offerings, I recognized few films in the catalog. Still, there were several that caught my eye as potentially interesting. SnagFilms plays well in either Firefox or Google Chrome.

**The Internet Archive** has a whole collection of movies and TV shows that you can stream. Some of the titles in the collection are reminiscent of the titles that they used to make fun of on MST3K, like "The Brain That Wouldn't Die." But there are also some real gems in there too, like Vincent Price's performance in "[The Last Man On Earth](#)," based on the SciFi novel "I Am Legend." The site runs equally well in either Firefox or Google Chrome.

It might not seem obvious at first, but [YouTube](#) also offers a ton of free movies. Pull up YouTube, go to the list on the left side of your browser window, scroll down to "More From YouTube," and select "Movies & Shows." The first category that is displayed should be "Free With Ads." There is a healthy selection of quality movies available for streaming. Similarly, you can find hundreds (if not thousands) of lesser known movies by typing "free movies" into the YouTube search bar.

### Use Your Card

No ... not your credit or debit card. Use your library card, you silly goose! Many libraries allow you to electronically check out movies and documentaries, as well

as ebooks, audiobooks, and music. If your library participates, you can typically "check out" programming using your library card, and you have three days to watch movies or documentaries. Here's the sweet part of the deal: in most locales, you can sign up for a public library card ... for FREE!

**Hoopla** is one such service that partners with participating libraries to allow you to do just that. I'm unable to locate a list of participating libraries on the Hoopla website. But, you can browse their list of movies [here](#). **OverDrive** is another service that does the same thing. It has functionality to see if your local library participates. Just go [here](#), and enter your zip code to see if your local library is listed. **Kanopy** is a third such service, with nearly 5,000 movie titles available to choose from. Just enter your local library (or university) name in the search box to see if your local library or university participates.

### Summary

If you are serious about "cutting the cord" to your cable or satellite service provider, it's becoming easier and easier to rid yourself of the overpriced, onerous cable bills. Even more, it's becoming quite easy to replace your cable or satellite service provider with FREE streaming options. While you might not be able to access the latest, greatest programming choices by using the free services, it's unlikely that you'll have much difficulty watching something that YOU want to watch, WHEN you want to watch it.

If you're willing to pay the much more modest fees for the subscription streaming services, then you'll also gain access to much more recent programming content.

These are choices that weren't generally available even five years ago. Linux users are accustomed to having choices. Isn't it nice to have choices?

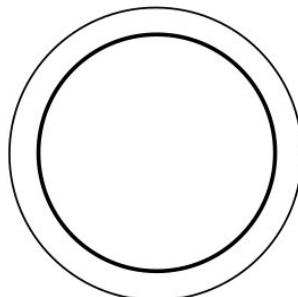


# Inkscape Tutorial: Rubber Stamp

by Meemaw

I saw this the other day! It's a quick and easy way to create text that looks like it was made with a rubber stamp.

Create your page, then draw a circle (remember, to keep your circle perfectly round, you should press the **<Ctrl>** key while you draw). Duplicate the circle and make the duplicate a bit smaller. Also, make the **Stroke** size a bit bigger for the inner circle.



Now create your text using the **Text tool**. Format it however you want, then place it in the center of the circles. You can use **Align and Distribute** to center everything.



Make sure all items are paths (for text, choose **Path > Object to Path** and for circles choose **Path > Stroke to Path**). Select all, then choose **Path > Union**.

With the **Calligraphic tool**, draw randomly over the figure, covering nearly all of it. Select all and choose **Path > Intersection**. As you can see, the part covered with the Calligraphic tool is the part that's visible afterwards.



Change the color to whatever you want, then apply a slight Gradient to it. For a more natural look, you can add a bit of **Gaussian Blur**.



PCLinuxOS

## Users Don't

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

**While Driving.**

**Put Down Your  
Phone & Arrive Alive.**

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

#### Material Submitted by Users

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

#### Entire Agreement

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



The logo for PCLOS-Talk, featuring three overlapping speech bubbles in red, orange, and yellow. To the right, there is a red speech bubble containing the text "Instant" and an orange speech bubble containing the text "Messages".

**PCLOS-Talk**  
Instant Messaging Server

*Sign up TODAY! <http://pclostalk.pclosusers.com>*

# Screenshot Showcase



Posted by mutse, July 3, 2019, running Mate.

# ***ms\_meme's Nook: Texstar Has High Hopes***

When you want something  
Well designed  
Don't get behind  
Get PCLinuxOS

Once there was a silly young man  
On his 'puter Windows ran  
Then he had a better plan  
Told Windows to scram

Texstar had high hopes  
He had high hopes  
You cannot deny  
He knew the ropes

So anytime  
You want the very best  
Join with all of the rest  
Oops that's how it all began  
Oops that's how it all began  
Oops that's how it all began now

When you want something  
Well designed  
Don't get behind  
Get PCLinuxOS

What makes our OS great  
Listen while I elaborate  
It never is out of date  
Always first rate

Texstar has high hopes  
He's got high hopes  
You cannot deny  
He knows the ropes

So anytime  
You want the very best  
Join with all of the rest  
Oops don't forget to donate  
Oops don't forget to donate  
Oops don't forget to donate now



**MP3**

**OGG**

# PCLinuxOS Recipe Corner



*from the kitchen of  
youcantoo*

## Grilled Italian Chicken

### Ingredients:

4 boneless skinless chicken breasts (about 1 1/4 lb)  
1 medium yellow bell pepper, cut into 4 wedges  
4 plum (Roma) tomatoes, cut in half  
1 small red onion, cut into 8 wedges  
1/2 cup reduced-fat Italian dressing

### Directions:

1. Heat gas or charcoal grill. Cut 4 (18x12-inch) sheets of heavy-duty foil. Place 1 chicken breast, 1 bell pepper wedge, 2 tomato halves and 2 onion wedges on center of each sheet. Pour 2 tablespoons of the Italian dressing over chicken and vegetable mixture on each packet.

2. For each packet, bring up 2 sides of foil over chicken and vegetables so edges meet. Seal edges, making tight 1/2-inch fold; fold again, allowing space for heat circulation and expansion. Fold other sides to seal.

3. Place packets on grill over medium heat. Cover grill; cook 18 to 22 minutes, rotating packets 1/2 turn after 10 minutes, until juice of chicken is clear when center of thickest part is cut (170F). Place packets on plates. Cut large X across tops of packets; carefully fold back foil. Serves 4.

### Options:

Individual grill packets are super quick to assemble and very easy to pop on the grill for cooking. For a "no-dishwashing-required" meal, serve the packets on sturdy disposable plates.

Serve these easy packets with a crisp green salad and cantaloupe wedges.



ELECTRONIC FRONTIER FOUNDATION  
DEFENDING YOUR RIGHTS IN THE DIGITAL WORLD



**PCLOS-Talk**  
Instant Messaging Server

**Sign up TODAY!** <http://pclostalk.pclosusers.com>



# *De-Googling Yourself, Part 4*



by Alessandro Ebersol (Agent Smith)

In the previous articles, I discussed how Google was born, how it watches over its users, how it harms both individuals and businesses, and now we will begin to see how to protect ourselves from Google.

## Alternatives to Google

To effectively reduce Google's meddling in our lives, we must reduce its use.

There is one catch, however. Google, which was coincidentally born when the commercial Internet began to establish itself as a popular mass media and file exchange, unfortunately, is so ingrained in

the internet itself that to have it completely removed from the users' lives can be an impossible task.

But we can diminish its presence and gradually undermine its influence on our lives to the point where Google will be irrelevant.

So let's see what Google's alternative services and products exist, and how we can slowly go down that road.

And, as Google started with a search service, there's nothing better than starting with alternatives to Google Search.



**ELECTRONIC FRONTIER FOUNDATION**  
DEFENDING YOUR RIGHTS IN THE DIGITAL WORLD



## Google Search

Google not only tracks your searches, but where you go online, what you write in your emails, who you send them to, and more. Its business model consists of profiling its users and profiting from it. It's much worse than being followed by a private detective.

You cannot predict who will have this information in 20 years and how it might be used. We already know that intelligence agencies are asking companies like Google and Yahoo to give this information to them. This applies to people who use Google, Yahoo, Bing, even if they don't live in the U.S. Thus, using Google search is a cumulatively dangerous act.

## Alternatives to Google Search



## DuckDuckGo

DuckDuckGo (DDG) is an Internet search engine that emphasizes the protection of the privacy of its users and avoiding the filter bubble of custom search results. DuckDuckGo distinguishes itself from other search engines by not ranking its users, and showing all users the same search results for a given search term. It prioritizes the return of the best results, rather than more results by generating these search results from more than 400 sources, including crowd sourced sites like Wikipedia, and other search engines like Bing, Yahoo! and Yandex. As of June 2019, it had an average of 39,107,576 daily direct searches.

The company is headquartered in Paoli, Pennsylvania, Philadelphia, and has 67 employees

as of July 2019. The company name is a reference to the duck, duck, goose children's game.

Some of DuckDuckGo's source code is open source hosted on GitHub under the Apache 2.0 license, but the core is proprietary. The company registered the domain name ddg.gg on February 22, 2011, and acquired duck.com on December 12, 2018, which is used as shortened URL alias that redirects to duckduckgo.com.

DuckDuckGo positions itself as a search engine that puts privacy first, and as such, does not store IP addresses, record user information, and uses cookies only when necessary. DuckDuckGo creator Gabriel Weinberg states: "By default, DuckDuckGo does not collect or share personal information. That's our privacy policy in a nutshell." However, they keep records of all search terms used.



### Qwant

Qwant is a French web search engine, launched in July 2013 and operated in Paris. It claims not to employ user tracking and does not customize search results to avoid capturing users in a filter bubble. It is available in 13 languages.

The site handles more than 10 million search requests per day and more than 50 million users per month worldwide, spread across its three main entry points: the normal homepage, the lightweight version, and the "Qwant Junior" for kids, with filtered results.

In the development phase, Qwant searches were powered by Bing, in addition to its own indexing capabilities. Qwant also confirmed the use of Bing's advertising network. As of March 2019, Qwant is the 41st most visited website in France, and the 879th most visited website in the World.

According to its founder, Qwant doesn't want to compete with Google, but prefers to "show something different." Users can create a free account, which allows them to post to "boards", a feature with functions similar to those of a social bookmarking platform.

Previous prominent features, such as a Wikipedia-based Knowledge Chart (called the Qnowledge Graph), appear to have been discontinued.

In July 2016, Mozilla signed a contract with Qwant to allow them to distribute an officially sanctioned version of the Firefox browser with Qwant as the default search engine. Qwant currently has a Firefox browser based on the Apple App Store available for iOS.

When it was launched in 2017, the Brave web browser introduced Qwant as one of its default search engines.

The French government in 2018 decreed that all government searches would be done using Qwant.

In June 2019, Qwant launched Qwant Maps, an open source mapping service that uses the OpenStreetMap database to deliver privacy while respecting maps and routing. He also unveiled Masq by Qwant, an open source technology that enables online services to deliver personalized results from data stored securely on a user's device.

The namesake company behind the search engine was co-founded in February 2011 by Jean-Manuel Rozan, a financier; Éric Leandri, a computer security expert; and Patrick Constant, a search engine expert. It employs more than 160 people, spread across five French cities (Paris, Nice, Ajaccio, Rouen, Epinal) and has offices in Germany and Italy. The company claims it makes money through commissions it receives when users visit advertised sites like eBay and Tripadvisor from their search results.



### Startpage

Startpage is a web search engine that highlights privacy as a distinguishing feature. It was formerly known as the Ixquick meta search engine, with Startpage being a variant service. Both sites were merged in 2016.

Ixquick was founded by David Bodnick in 1998 and belongs to the Netherlands-based Startpage BV since it acquired the Internet company in 2000. Ixquick and its sister project Startpage.com reached their last daily direct query record (28 days average) of 5.7 million as of February 2, 2015.

The company also provides the standalone proxy service, Startpage.com Proxy, which is incorporated into the Startpage search engine, allowing users the option to open all search results through proxy. The company has also developed a privacy protection service called StartMail. This service was launched to the public in 2014.

### How Startpage Search Works

You can't beat Google when it comes to online searches. Then Startpage pays to use the brilliant search results from Google, and removes all trackers and records. The result: the best and most private search engine in the world.

### No personal data store

Startpage does not collect or share your personal information. Ever. There is literally no user data on the company's servers. The company does not profile the user, and cannot be forced to surrender its data to the authorities simply because it has no data to deliver.

### Incognito

Clicking search results means leaving the protection of Startpage.com. This can lead to a cookie truck being installed on your device. For this reason, the

company developed the “Anonymous Viewing” feature. With “Incognito”, users can visit search results in complete privacy and keep browsing. They’ll never know they were there. The resource can be found next to all search results.

### No filter bubble

Other search engines use search habits to deliver results that they think the user wants, basically trapping users in a result chamber. With Startpage.com, the filter bubble is broken for a wider range of results.



### Ecosia

Ecosia is a Berlin-based Internet search engine that plants trees by donating 80% or more of their surplus income to non-profit organizations that focus on reforestation and conservationism. Ecosia considers itself a social business, is CO<sub>2</sub> negative, claims to support full financial transparency, protects the privacy of its users and is certified by B-Lab as a charitable corporation.

The site maintains a running total of the number of trees planted. According to its website, as of July 16, 2019, the search engine was responsible for planting more than 62 million trees. Its online shopping and search advertising revenue is donated to a reforestation program, currently in Burkina Faso. In November 2015, Ecosia celebrated 3 million planted trees and US\$ 5 million donated to the environment. Ecosia is also available on mobile devices, one can find the app on the Google Play Store or iOS App Store.

The search engine at launch originally provided a combination of Yahoo! and technologies from Bing and Wikipedia. Ads were delivered by Yahoo! as part of a revenue sharing agreement with Ecosia.

Ecosia search results are now provided by Microsoft Bing, enhanced by the company’s own algorithms. It is currently available as a web browser or mobile app on Android and iOS devices.

By 2018, Ecosia pledged to become a privacy-friendly search engine. Searches are encrypted, not stored permanently, and data is not sold to third party advertisers. The company states in its privacy policy that it does not create personal profiles based on search history, or use external tracking tools such as Google Analytics.

Ecosia displays ads alongside its search results and is paid by partners whenever a user is directed to an advertiser via a sponsored link. A single survey in Ecosia generates on average about half a cent (0.005 EUR). Ecosia takes Ecosia 0.22 euro (€) and 1.1 seconds to plant a tree.

Ecosia was first launched on December 7, 2009 to coincide with the UN climate negotiations in Copenhagen. Over time, Ecosia has supported various tree planting programs. Until December 2010, Ecosia funds went to a WWF Germany program that protected the Juruena National Park in the Amazon basin. To protect this area, organizers drafted and funded plans with logging companies and local communities.

By 2011, the search engine had raised over €250,000. In 2013, about 200,000 people were using Ecosia and 116,000 seedlings were funded by donations from Ecosia.



### YaCy

YaCy (pronounced “ya see”) is a free distributed search engine based on peer-to-peer (P2P) networking principles. Its core is a computer program written in Java distributed on several hundred computers, as of September 2006, called YaCy-peers. Each YaCy-peer independently crawls over the Internet, parses and indexes web

pages, and stores indexing results in a common database (called an index) that is shared with other YaCy-peers using P2P networking principles. It’s a free search engine that everyone can use to create a search portal for their intranet and to help them search the public Internet clearly.

Compared to semi-distributed search engines, the YaCy network has a decentralized architecture. All YaCy-peers are the same and no central server exists. It can run in a crawl mode or as a local proxy server, indexing web pages visited by the person running YaCy on your computer. Various mechanisms are provided to protect user privacy. Access to search functions is through a locally running web server, which provides a search box for entering search terms and returns search results in a format similar to other popular search engines.

YaCy is available on Windows, Mac and Linux. YaCy was created in 2003 by Michael Christen.

The YaCy search engine is based on four elements:

#### Crawler

A search engine that scans web pages and analyzes their content.

#### Indexer

Creates a reverse word index (RWI), that is, each RWI word has its list of relevant URLs and ranking information. Words are saved in the form of word hashes.

#### Search and Administration Interface

Made as a web interface provided by a local HTTP servlet with the servlet engine.

#### Data storage

Used to store the reverse word index database using a distributed hash table.



### Search Engine Technology

\* YaCy is a complete appliance with user interface, index, administration and monitoring.

\* YaCy collects web pages with a web crawler. The documents are then parsed, indexed and the search index is stored locally. If your peer is part of a peer network, the local search index will also be merged into the shared index of that network.

\* A search is started and the local index contributes along with a global peer search index on the YaCy search network.



#### Gigablast

Gigablast is an open source directory and search engine. Founded in 2000, it is an independent Albuquerque, New Mexico-based web crawler engine developed and maintained by Matt Wells, a former Infoseek employee and graduate of New Mexico Tech.

Search engine source code is written in the C and C ++ programming languages. It was released as open source software under the Apache version 2 license in July 2013. In 2015 Gigablast claimed to have indexed over 12 billion web pages and received billions of queries per month.

Gigablast has provided and provides search results for other companies such as Ixquick, Clusty, Zulua, Snap, Blingo and Internet Archive.

#### History

Matt Wells worked for Infoseek search engine until leaving in 1999 to start work on what would become Gigablast, coding everything from scratch in C ++. It was originally designed to index up to 200 billion web pages. Gigablast went live on July 21, 2002.

### Features

Gigablast supports a number of specialized searches and Boolean algebra operators. It also supports a related concepts feature called Giga Bits and a blog search feature.

A feature called Gigabits provides relevant information beyond what the user is searching for.

Gigablast also claims to be, as of 2010, the "leader" clean energy search engine with 90 percent of its power coming from wind power.



#### Dogpile

Dogpile is a World Wide Web search engine that searches results from Google, Yahoo!, Yandex, Bing and results from other popular search engines, including audio and video content providers like Yahoo!

Dogpile went live in November 1996. The site was created and developed by Aaron Flin, who was frustrated with the varying results of existing indexes and intended to make Dogpile query multiple indexes for the best search results. Originally, it provided Yahoo! (directory), Lycos (directory A2Z), Excite (directory Excite Guide), WebCrawler, Infoseek, AltaVista, HotBot, WhatUseek (directory) and World Wide Web Worm. He naturally made comparisons with MetaCrawler, a multi-search engine threaded entity that existed before, but Dogpile was more advanced, and could also search Usenet (from sources including DejaNews) and FTP (via Filez and other indexes).

In August 1999, Dogpile was acquired by Go2net, which was already operating MetaCrawler. Go2net was then acquired by InfoSpace in July 2000 for \$4 billion. Dogpile received a redesign of its interface for the first time in December 2000.

The Dogpile search engine won the JD Power and Associates Award for Best Residential Online Search Service in 2006 and 2007.

In August 2008, Dogpile and Petfinder agreed to a research partnership.

In November 2008, Dogpile launched the Search and Rescue program, which donates money to animal-related charities. The program also helps people find help for animals in need. By early December 2008, people using the Dogpile search engine had raised \$ 100,000 for Dogpile's Search and Rescue program.

In July 2016, Blucora announced the sale of its InfoSpace business to OpenMail for \$45 million in cash, placing Dogpile under the ownership of OpenMail. OpenMail was later renamed System1.

### Features

\* Category Links: Links to help users focus their search on specific categories like news, audio, etc.

\* Yellow Pages: Allows users to search using the Yellow Pages.

\* Web search box: The area where users enter the search term. Enter keywords and press the Search button to retrieve the results.

\* Search button: The button pressed to search for results.

\* Preferences: Links to a page where users can set a variety of custom search preferences.

\* Spell Correction: Provides spelling suggestions for misspelled words and automatically corrects commonly typed keywords.

\* Search filter: Block potentially explicit content for multimedia searches in the Moderate setting and for all searches when in the heavy setting.

- \* Statistics bar: Shows how many results were returned for the search term.
- \* About Results: Discover Dogpile's policies regarding sponsored and unsponsored search results.
- \* IntelliFind: Recommends additional content based on the original search term.
- \* Are you looking for?: Offers spelling suggestions for words that may be incorrect and other search keywords that appear to be related to the original search term.
- \* Recent Searches: Track the 15 most recent searches. The list is reset when the browser is closed.
- \* Favorite Searches: Shows recent popular searches from other users

### SearX



SearX (/ sɜːrks /) is a free metadata engine, available under the GNU Affero General Public License version 3 license, to protect the privacy of its users. To this end, SearX does not share users' IP addresses or search history with the search engines from which it collects results. Tracking cookies served by search engines are blocked, preventing modification of results based on user profiles. By default, SearX queries are sent via HTTP POST to prevent user query keywords from appearing in Web server logs. SearX was inspired by the SeekS project, although it does not implement peer user result ranking. -to-peer from SeekS.

Each search result is provided as a direct link to their site, rather than a crawled redirect link used by Google. In addition, when available, these deep links are accompanied by "cached" and / or "proxy" links that allow you to view result pages without actually visiting the sites in question. "Cached" links point to

saved versions of a page on archive.org, while "proxy" links allow you to view the current active page through a SearX-based web proxy. In addition to general search, the engine also features guides for searching on specific domains: files, images, IT, maps, music, news, science, social media, and videos.

Along with the best known instance on searx.me, SearX also features dozens of user-run instances under their own URLs, some of which are available as hidden Tor services. "Meta-searX" sites refer to a different random instance on each search. A public API is available for SearX, as well as for Firefox search provider plugins, and for an Android app.

### Why use SearX?

- \* SearX may not offer personalized results like Google, but it does not generate a profile about the user.
- \* SearX doesn't care what the user is looking for, never shares anything with third parties and cannot be used to compromise the user.
- \* SearX is free software, the code is 100% open and anyone can help make it better. Your code is available on Github.
- \* If you care about privacy, whether you are a conscious user or believe in digital freedom, make SearX your default search engine or run it on your own server.

### Technical Details - How does it work?

SearX is a meta search engine, inspired by the SeekS project. It provides basic privacy by blending its queries with searches on other platforms without storing survey data. Queries are made using a POST request in all browsers (except chrome \*). Therefore, they do not appear in their logs or in your URL history. For Chrome \* users, there is one

exception, SearX uses the search bar to make GET requests.

SearX can be added to your browser's search bar; Additionally, it can be set as the default search engine.

SearX was created by Adam Tauber.



### Gibiru

Following the DuckDuckGo search engines and Startpage (the former ixquick), Gibiru is the latest addition to the slowly growing group of anonymous search engines, and all promise to put privacy before their profits. Like Startpage, one of Gibiru's advantages is that you can use it to privately search Google's own databases, offering all the benefits of Google search, but without any privacy risks.

Gibiru founder and CEO Steve Marshall announced in a press release that his service is exactly what Google was 10 years ago. This comment exactly mimics DuckDuckGo CEO Weinberg's statement when he launched his website. Like DuckDuckGo and Startpage, Gibiru claims to allow its users to search privately, claiming that the site does not use your IP address or cookie data when returning search results.

### Escape the Google filter bubble

The search engine used by Gibiru apparently follows a modified Google algorithm that allows you to get the search results you are used to from Google while still hiding your identity from Google's servers. If true, your searches will not be modified according to who you are and you will be free from the so-called "Google filter bubble". Gibiru was founded in 2009 by Steve Marshall.



Well, here we end the first part of recommendations for alternatives to Google services and products. I scanned the search engines one by one and was impressed by Gigablast, Dogpile and Gibiru, very fast and very relevant results, which is important now that Google is censoring results.

So continue with us next month, when I will continue this series, and, let's go, De-Gooling ourselves, a little bit every day. I seldom use Google's search engine myself. Others have far more "out of the box" results.

A green rectangular banner with rounded corners. On the left is a green square icon containing a white play button symbol with a circular arrow around it. To the right of the icon, the word "DOWNLOAD" is written in white capital letters. Below that, the "PCLinuxOS" logo is displayed in large, stylized, metallic blue and silver letters. Underneath the logo, the words "Mate Desktop" are written in a large, italicized, white serif font.



# Screenshot Showcase



---

*Posted by seaplane\_tux, July 4, 2019, running KDE.*

# Casual Python, Part 7

by Peter Kelly (critter)

## And now for something completely different

This will be different because I am going to demonstrate building a graphical application without using Designer. I am also going to get a head start by using someone else's code. I am not talking here about blatant plagiarism, which is frowned upon and could even lead to a legal challenge, but of code reuse. Both Python and Qt are licensed very liberally for open source software, but commercial software, Qt at least, requires a suitable license to be sought. For what we are doing, we should be fine. The code I am going to use started as part of the Qt5 examples, but I am going to modify it extensively. Even so, I include a copy of the example's license. You should always consider this when bringing in other code. Why do I not use designer? Quite simply, designer does not support some of the features used in this application. In fact, the only thing here that designer fully supports is the QWidget we use as the base of the form.

A few years ago, I was using KDE and had Cairo clock, a smart looking analog clock, on the desktop. The analog clock supplied with the current KDE/Plasma desktop I did not like at all, and was going to re-install Cairo clock when I realised that with Python and Qt5 I could make my own and probably improve on it. This is our next project.

A desktop clock is most usually running continuously so you probably don't want it to show on the taskbar. This is only a problem with some desktops, but if it affects you, I included some code at the end to fix this. However, if you want to include that code, then you will need to install wmcctrl from the repository (I needed this on my openbox install, but you may not).

```
#####
## Copyright (C) 2013 Riverbank Computing Limited.
## Copyright (C) 2010 Nokia Corporation and/or its subsidiary(-ies).
## All rights reserved.
##
## This file is part of the examples of PyQt.
##
## $QT_BEGIN_LICENSE:BSD$
## You may use this file under the terms of the BSD license as follows:
##
## "Redistribution and use in source and binary forms, with or without
## modification, are permitted provided that the following conditions are
## met:
```

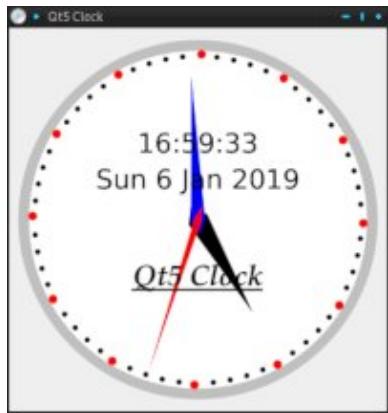
```
## * Redistributions of source code must retain the above copyright
##   notice, this list of conditions and the following disclaimer.
## * Redistributions in binary form must reproduce the above copyright
##   notice, this list of conditions and the following disclaimer in
##   the documentation and/or other materials provided with the
##   distribution.
## * Neither the name of Nokia Corporation and its Subsidiary(-ies) nor
##   the names of its contributors may be used to endorse or promote
##   products derived from this software without specific prior
##   written permission.
##
## THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
## "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
## LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
## A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
## OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
## SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
## LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
## DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
## THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
## (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
## OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE."
## $QT_END_LICENSE$
```

This is the original licence that came with the example upon which this is based.

## Q5\_aclock.py



This is the clock from the Qt5 examples. It works fine but lacks features.



My basic clock - no transparency, dots for number markers and added digital date and time and a default label. It also features a choice of sweep or stepping second hand.



Clock with Arabic numbers, batons for the minute markers and a custom label. Transparency removes the window frame.

Clock with rotated Roman numerals and a dark theme.



The screenshots give an idea of what can be achieved. It's re-sizeable, so you can have a really BIG clock if you are working away from the computer. All of the options can be passed on the command line, which also features a help prompt:

```
$:> ./qt5_aclock.py -h
usage: qt5_aclock.py [-h] [-l L] [-n] [-r] [-d] [-b] [-s] [-t] [-x X] [-y Y]
```

It provides an analog clock with sweep or stepping second hand, choice of numerals. Dots or batons for minute marks, light or dark theme and optional text field.

#### optional arguments:

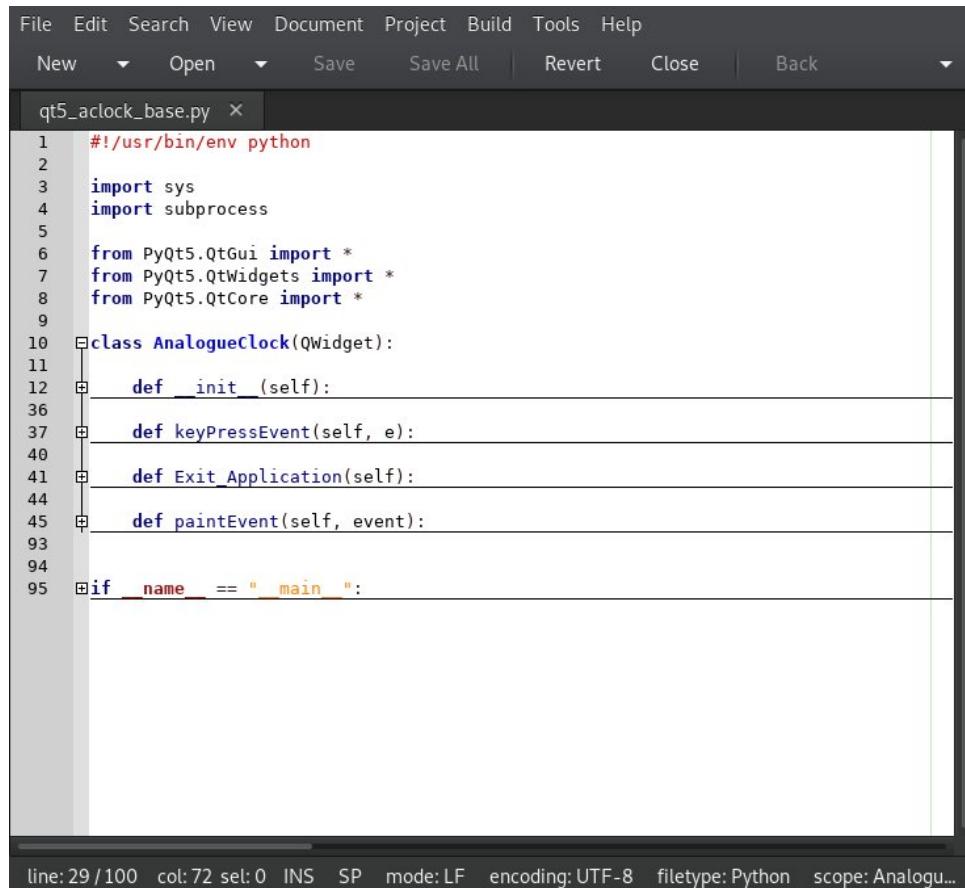
- h, --help show this help message and exit
- l L permits customization by adding text to the clockface (maximum 12 characters). quote if spaces included (default: Qt5 Clock)
- n adds Arabic numerals to the display (default: False)
- r adds Roman numerals to the display (default: False)
- d apply the dark theme (default: False)
- b Use batons for minute marks, default is dots (default: False)
- s changes the default stepping second hand to a sweeping motion (default: False)
- t enables transparency - compositing must be enabled and active (default: False)
- x X x startup position (default: 0)
- y Y y startup position (default: 0)

© Casualsoft 2018



There is a lot of code in this one, almost 500 lines in the full version! I am going to start with a fairly basic version and add features once it is up and running. This is the way most applications develop.

Here is the reduced version with the code inside the definitions collapsed in Geany (you can do this by clicking in the little boxes in the margin. It helps you to concentrate on only the parts of the code that are of current interest). This looks a lot less fearsome now.



```

File Edit Search View Document Project Build Tools Help
New Open Save Save All Revert Close Back ▾

qt5_aclock_base.py ×

1 #!/usr/bin/env python
2
3 import sys
4 import subprocess
5
6 from PyQt5.QtGui import *
7 from PyQt5.QtWidgets import *
8 from PyQt5.QtCore import *
9
10 class AnalogueClock(QWidget):
11     def __init__(self):
12     def keyPressEvent(self, e):
13     def Exit_Application(self):
14     def paintEvent(self, event):
15
16 if __name__ == "__main__":
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95

```

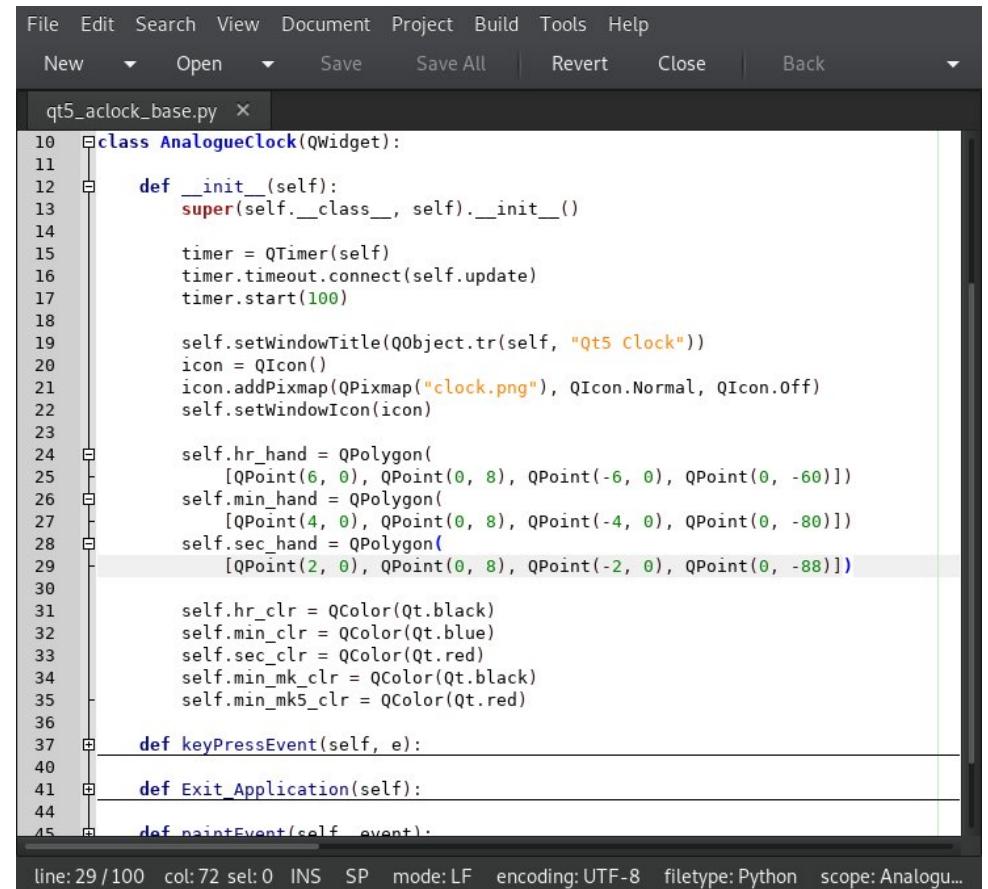
line: 29 / 100 col: 72 sel: 0 INS SP mode: LF encoding: UTF-8 filetype: Python scope: AnalogueClock

As I pointed out earlier, a clock is usually set to run on login, and then runs constantly. For this reason, I have given it a command line interface. With this, you can write a command with all of the options you want, and pass it to however you autostart things. You could also write a qt5\_aclock.desktop file, which included the options, as I discussed when building the application finder. If you don't want any of the options, then just run it without them. Another alternative is

to build a version with all of the required options set as defaults. This is the reason for building the application after all - to get something that does exactly what you want. You could just install the beautiful Cairo clock, but this is your baby. This is personal.

Looking at the code as shown in the image, the only thing that has changed from the previous examples is the name of the class and the new paintEvent method. Also there is no user interface module imported, as we haven't used designer to create one. Because of this, the class definition calls only the QWidget class inside its parentheses. At this stage, it has only the attributes and methods of that widget.

In fact, the only changes in here turn out to be in the `__init__` method and the new `paintEvent` method. The `keyPressEvent` and `Exit_Application` are exactly the same as in previous examples. Even the code following line 95 has no surprises.



```

File Edit Search View Document Project Build Tools Help
New Open Save Save All Revert Close Back ▾

qt5_aclock_base.py ×

10 class AnalogueClock(QWidget):
11     def __init__(self):
12         super(self.__class__, self).__init__()
13
14         timer = QTimer(self)
15         timer.timeout.connect(self.update)
16         timer.start(100)
17
18         self.setWindowTitle(QObject.tr(self, "Qt5 Clock"))
19         icon = QIcon()
20         icon.addPixmap(QPixmap("clock.png"), QIcon.Normal, QIcon.Off)
21         self.setWindowIcon(icon)
22
23         self.hr_hand = QPolygon(
24             [QPoint(6, 0), QPoint(0, 8), QPoint(-6, 0), QPoint(0, -60)])
25         self.min_hand = QPolygon(
26             [QPoint(4, 0), QPoint(0, 8), QPoint(-4, 0), QPoint(0, -80)])
27         self.sec_hand = QPolygon(
28             [QPoint(2, 0), QPoint(0, 8), QPoint(-2, 0), QPoint(0, -88)])
29
30         self.hr_clr = QColor(Qt.black)
31         self.min_clr = QColor(Qt.blue)
32         self.sec_clr = QColor(Qt.red)
33         self.min_mk_clr = QColor(Qt.black)
34         self.min_mk5_clr = QColor(Qt.red)
35
36
37     def keyPressEvent(self, e):
38
39     def Exit_Application(self):
40
41     def paintEvent(self, event):
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95

```

line: 29 / 100 col: 72 sel: 0 INS SP mode: LF encoding: UTF-8 filetype: Python scope: AnalogueClock



Here is the `__init__` method unfolded.

Lines 15 - 17 call up a QTimer object which is assigned to the variable timer. A QTimer object continually times out every interval and emits a signal which is connected to the QWidgets update method. Line 17 starts the timer with an interval of 100 milliseconds. All this means is that the class is updated 10 times every second.

Lines 19 - 22 set the window title and icon.

Lines 24 - 29 set the shapes of the three hands. Qt knows how to draw shapes of three or more sides by using its QPolygon method. This method requires x, y coordinates for each corner point and these are supplied in the form of QPoint objects which are designed for the job. A QPoint requires two integers, no floating point numbers. (From the official Qt documentation: 'The QPoint class defines a point in the plane using integer precision'.)

You may just be able to see a thin, blue, vertical line down the right hand side of the image. This line I have set at column 80 which is the convention for the maximum width of code text. This convention is actually a throwback to the days when terminals had a maximum of 80 columns. If we stick to the convention, then anybody reading the code is happy. Nothing bad will happen if you exceed this limit. To make these lines fit, I have split them after the first parenthesis. Python accepts this as a line continuation but if you need to split a line where there are no parentheses, use a backslash as the last character of the line and continue on the next line. When splitting lines you should consider readability. I could also have done something like this:

```
self.hr_hand = QPolygon([QPoint(6, 0),
                        QPoint(0, 8),
                        QPoint(-6, 0),
                        QPoint(0, -60)])
```

To Python it is the same.

Lines 31 - 35 set the colors for the three hands, the minute marks and the five-minute marks. I have used standard color names, but most conventions are accepted such as '#00ff00' for green (must be quoted). That's the `__init__` method done.

The next two methods have no changes from those we have used previously.

```
def keyPressEvent(self, e):
    if e.key() == Qt.Key_Escape:
        self.Exit_Application()
```

```
def Exit_Application(self):
    self.close()
    sys.exit()
```

The paintEvent method is where most of the work is done:

```
qt5_aclock_base.py

44
45 def paintEvent(self, event):
46     side = min(self.width(), self.height())
47     time = QTime.currentTime()
48     qp = QPainter()
49     qp.begin(self)
50     qp.setRenderHint(QPainter.Antialiasing)
51     qp.translate(self.width() / 2, self.height() / 2)
52     qp.scale(side / 210.0, side / 210.0)

53
54     qp.setPen(QPen(Qt.lightGray, 6)) # draw the background
55     style = Qt.BrushStyle(Qt.SolidPattern)
56     qp.setBrush(QBrush(QColor(255, 255, 255), style))
57     qp.drawEllipse(QRect(-97, -97, 194, 194))

58
59     qp.setPen(Qt.NoPen) # draw the hour hand
60     qp.setBrush(QBrush(self.hr_clr))
61     qp.save()
62     qp.rotate(30.0 * ((time.hour() + time.minute() / 60.0)))
63     qp.drawConvexPolygon(self.hr_hand)
64     qp.restore()

65
66     qp.setPen(QPen(self.min_mk5_clr)) # draw the 5 minute marks
67     qp.setBrush(QBrush(self.min_mk5_clr))
68     for i in range(0, 12):
69         qp.drawEllipse(90, 0, 4, 4)
70         qp.rotate(30.0)
71     qp.setPen(Qt.NoPen)
72     qp.setBrush(QBrush(self.min_clr))

73
74     qp.save() # draw the minute hand
75     qp.rotate(6.0 * (time.minute() + time.second() / 60.0))
76     qp.drawConvexPolygon(self.min_hand)
77     qp.restore()
78     qp.setPen(Qt.NoPen)
79     qp.setBrush(QBrush(self.sec_clr))

80
81     qp.save() # draw the second hand
82     qp.rotate(6.0 * (time.second()))
83     qp.drawConvexPolygon(self.sec_hand)
84     qp.restore()

85
86     qp.setBrush(QBrush(self.min_mk_clr)) # draw the minute marks
87     qp.setPen(QPen(self.min_mk_clr))
88     for j in range(0, 60):
89         if j % 5:
90             qp.drawEllipse(90, 0, 2, 2)
91             qp.rotate(6.0)
92     qp.end()
```

The clock is going to be painted on to the form provided by the QWidget. As we want the clock to be circular, we must constrain it to fit within the smallest side of the forms display area. To do this, we can use the standard Python function min(), which returns the smallest item of the supplied arguments.

The variable time is set to the value returned by QTime class's currenttime method, which is retrieved from the system clock.

The QPainter class does the painting on the canvas of the form (yes, just like a real painter).

Line 48 assigns an instance of this class to the variable qp.

All painting is performed in code placed between the QPainters begin and end methods.

Lines 49 and 92 call these methods.

Line 50 tells qp how to render the images it paints. Antialiasing will give smoother edges to our lines and radii.

Line 51 moves (translates in geometric terms) the painter to the center of the canvas.

Line 52 Scales the coordinate system (really, just play with these values to see what they do).

To paint on the canvas, qp requires a pen for the outlines and a brush to fill in the shapes. A QPen has a color, a style (line type: solid, dotted etc), a width which is relative to the object size, a cap style and a join style. If a separate outline is not required, then the special object Qt.NoPen can be used. A QBrush has a style, color, gradient and a texture.

Lines 54 - 56 set these attributes, where not set the default is used. Try changing the brush style to Qt.DiagCrossPattern and see what happens.

Line 57 draws the ellipse (a circle is an ellipse with the major and minor axes equal) inside a rectangle, starting at -97, 97, 194 wide and 194 high. These values are not pixels, but are relative to the scale in line 52.

The rest of the code in this method is fairly repetitive, but with special features that I will explain.

Line 59 calls the setPen method with NoPen. The hour hand is simply a colored diamond shape with no outline, unlike the clock body which has a light gray rim or bezel.

Line 61 saves the current state of qp.

Line 62 calculates the angle of the hour hand for the current time and rotates the QPainter by this amount. For example, at 3:05 the angle is:

$$30 * (3 + 5 / 60) = 30 * 3.083 = 92.9 \text{ or, just past the 3 on the dial (12 o'clock is 0 degrees).}$$

Line 63 actually draws the hour hand.

Line 64 restores the previous state of the painter, removing any rotation.

Line 66 to 70 draw 12 little circles at a radius of 90 and size 4 rotating 30 degrees between each. There is no need for a save and restore here, as a complete 360 degrees is completed.

Lines 72 - 77 draw the minute hand calculating the angle from current minute and second values. The seconds were ignored when calculating the hour hand, as the difference would be so very small.

Lines 79 to 84 draw the second hand.

The minute marks are drawn in a similar manner to the 5 minute marks skipping each five minute position. To do this, the modulo operator is used in an if statement. Modulo 5 returns the remainder after division.

1 % 5	<b>==&gt; 1</b>	<b>draw mark</b>
2 % 5	<b>==&gt; 2</b>	<b>draw mark</b>
3 % 5	<b>==&gt; 3</b>	<b>draw mark</b>
4 % 5	<b>==&gt; 5</b>	<b>draw mark</b>
5 % 5	<b>==&gt; 0</b>	<b>skip mark</b>
6 % 5	<b>==&gt; 1</b>	<b>draw mark</b>
...		

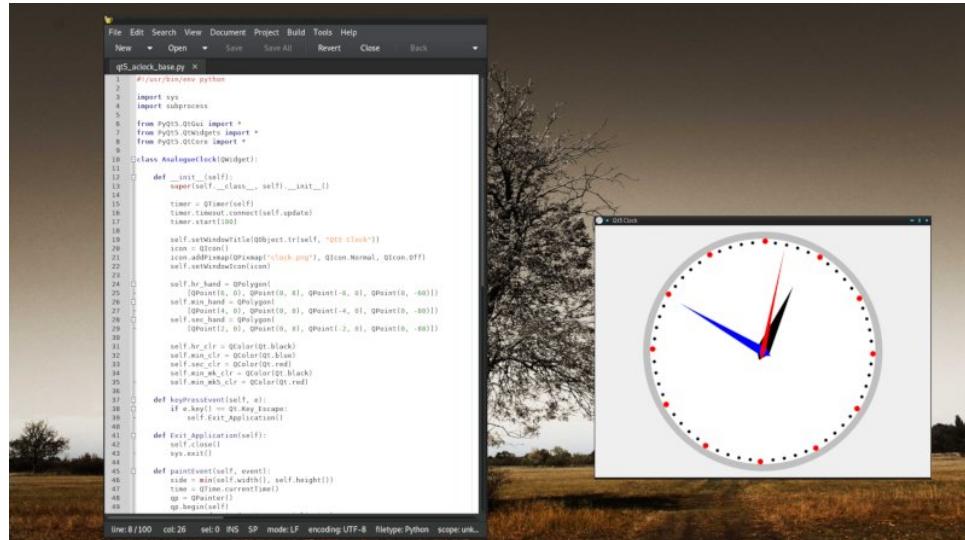
All of the above is repeated to redraw the clock every time the timer fires, ten times a second. If the timer was set to once a second, we might just miss the update time by a microsecond or so and this would then not always move the second hand by the correct amount. If you do not want the second hand, remove the code for drawing it, and replace the timer value in line 17 with 1000. This will reduce the load on the system (although the load is quite small, even with the second hand).

The final section of the code you should recognize:

```
if __name__ == "__main__":
    app = QApplication(sys.argv)
    clock = AnalogueClock()
    clock.show()
    sys.exit(app.exec_())
```

And that completes the basic clock.

If you run this code, you will see a rather large clock placed where the window manager decided it would be best to place it.



To regain control, add a line after `self.setWindowTitle(...)` which is in the `__init__` definition.

```
self.resize(120, 120)
```

To give a smaller clock.

Try resizing the window or make it full screen.

## Adding features

I am going to add the new features one by one, and then test each. If you are using Geany, this is easy to do. Modify the code and then press the F5 key. The application will hopefully launch, and a small console window will open. The console window is to display any messages from Python. The default terminal in Geany is xterm. If you do not have xterm installed, or would rather use a different one, this can be set from the Edit menu – Preferences – Tools.

For example: `konsole -e "/bin/sh %c"`

If the Application successfully launches and the changes are as they should be, simply press the escape key to close the application, and then press the return key to close the terminal window. If unsuccessful, read any messages in the terminal window to try and understand what went wrong. The routine is: F5, escape, return, edit code and then repeat the sequence. The code file is automatically saved each time you press F5.

## Transparency

First of all, let's get rid of the window frame and make the background of the frame transparent. Add these two lines somewhere near the beginning of the `__init__` method. Where exactly doesn't matter, as long as it is in that method.

```
self.setWindowFlags(Qt.FramelessWindowHint)
self.setAttribute(Qt.WA_TranslucentBackground, True)
```

I probably should have told you this before, but a method or function definition ends when the indentation ends. This is also true for for loops and other control structures, and this is why the four space indentation is so important.

Now that you have no window frame, you will be glad we have the `keyPressEvent`, as you now have no close button. Press Escape to close the clock.

We can also make this an option in the code.

Change the two lines we added to `__init__` to read

```
if transparency == 'transparent':
    self.setWindowFlags(Qt.FramelessWindowHint)
    self.setAttribute(Qt.WA_TranslucentBackground, True)
```

The 'if' should be in line with 'super' and the other two lines indented a further four spaces.

After the line:

```
if __name__ == "__main__":
```

Add, indented four spaces, This line:

```
    transparency = 'transparent'
```

Now, if we do not want transparency, change the line above to

```
    transparency = ''
```

I will use a similar method to add other options, and then finally, make them switchable from the command line or application startup.

## Batons

To change the minute marks from dots to batons, add this line after the transparency line:

```
transparency = 'transparent'
marks = 'batons'
```

Change the code for drawing the 5 minute marks to:

```
qp.setPen(QPen(self.min_mk5_clr))
qp.setBrush(QBrush(self.min_mk5_clr))
for i in range(0, 12):
    if marks == 'batons':
        qp.drawLine(85, 0, 95, 0)
    else:
        qp.drawEllipse(90, 0, 4, 4)
    qp.rotate(30.0)
```

And the code for the minute marks to:

```
qp.setBrush(QBrush(self.min_mk_clr))
qp.setPen(QPen(self.min_mk_clr))
for j in range(0, 60):
    if j % 5:
        if marks == 'batons':
            qp.drawLine(90, 0, 95, 0)
        else:
            qp.drawEllipse(90, 0, 2, 2)
    qp.rotate(6.0)
```

Now if you change

```
marks = 'batons'
```

to

```
marks = ''
```

You can have the dots back.

## Sweep second hand

Add

```
movement = 'sweep'
```

After

```
marks = 'batons'
```

Change the code to draw the second hand to:

```
qp.save()
if movement == 'sweep':
    qp.rotate(6.0 * (time.second() + time.msec() / 1000))
else:
    qp.rotate(6.0 * (time.second()))
qp.drawConvexPolygon(self.sec_hand)
qp.restore()
```

This adds milliseconds to the equation to get smaller increments. This is not true, smooth movement but as the position is redrawn ten times a second it looks pretty close. Look closely at a real wall clock with sweep second hand. Most use the same method.

## Legend

Add

```
legend = 'Qt5 Clock'
```

After

```
movement = 'sweep'
```



Add

```

font = QFont()
font.setFamily("Arial") # or whatever you prefer / is installed
font.setPointSize(12)
font.setBold(False)
font.setItalic(True)
font.setUnderline(True)
font.setWeight(75)
qp.setFont(font)
qp.drawText(-50, 25, 100, 16, Qt.AlignCenter, legend)

```

As a new block somewhere between the qp.begin and qp.end statements. Make sure that you use a font that is installed on your system. Change the other font attributes to your personal preference. This takes care of drawing the legend.

## Start up position

After

```
legend = 'Qt5 Clock'
```

Add

```
xpos = 20
ypos = 20
```

Change the code after

```
clock.show()
```

To:

```

clock.show()
if transparency == 'transparent':
    clock.move(xpos, ypos)
else:
    clock.move(xpos, ypos + 20)

```

Note the double == in if transparency == 'transparent': This not an assignment. Here you are testing for equality.

You need a little more in the y direction to accommodate the title bar if transparency is not enabled.

## Dark theme

After

```
ypos = 20
```

Add

```
theme = 'light'
```

In the \_\_init\_\_ method locate what originally were lines 31 to35 and looked like this:

```

self.hr_clr = QColor(Qt.black)
self.min_clr = QColor(Qt.blue)
self.sec_clr = QColor(Qt.red)
self.min_mk_clr = QColor(Qt.black)
self.min_mk5_clr = QColor(Qt.red)

```

And change it to look like this:

```

if theme == 'light':
    self.hr_clr = QColor(Qt.black)
    self.min_clr = QColor(Qt.blue)
    self.sec_clr = QColor(Qt.red)
    self.min_mk_clr = QColor(Qt.black)
    self.min_mk5_clr = QColor(Qt.red)
else:
    self.hr_clr = QColor(Qt.lightGray)
    self.min_clr = QColor(Qt.cyan)
    self.sec_clr = QColor(Qt.red)
    self.min_mk_clr = QColor(Qt.white)
    self.min_mk5_clr = Qt.cyan

```

Now go to the code in the paintEvent method that paints the background and looks like this:

```

qp.setPen(QPen(Qt.lightGray, 6))
style = Qt.BrushStyle(Qt.SolidPattern)
qp.setBrush(QBrush(QColor(255, 255, 255), style))
qp.drawEllipse(QRect(-97, -97, 194, 194))

```

And change that to look like this:

```

if theme == 'light':
    qp.setPen(QPen(Qt.lightGray, 6))
    style = Qt.BrushStyle(Qt.SolidPattern)
    qp.setBrush(QBrush(QColor(255, 255, 255), style))

```



```

else:
    qp.setPen(QPen(QColor(189, 183, 107), 4))
    style = Qt.BrushStyle(Qt.SolidPattern)
    gradient = QRadialGradient(QPoint(0, 0), 75)
    gradient.setColorAt(0, QColor(0,0,0))
    gradient.setColorAt(1, QColor(22,8,0))
    qp.setBrush(gradient)
    qp.drawEllipse(QRect(-97, -97, 194, 194))

```

Try changing the theme = 'light' statement to theme = 'dark' to see the alternative style.

I have added a radial gradient to the dark theme background to demonstrate the feature. On a small clock it is not very noticeable. Try it full screen or use different colors.

In the next section I'll show how to put the numbers on the clock, how to rotate the Roman numerals as they go around the clock and how to make all of these options selectable from the application start up command.

*Editor's Note:* All of the code for the Casual Python article series is available for download from [here](#).



## The PCLinuxOS Magazine Special Editions!

Get Your Free Copies Today!

## *International Community PCLinuxOS Sites*



*Like Us On Facebook!  
The PCLinuxOS Magazine  
PCLinuxOS Fan Club*



## **Screenshot Showcase**

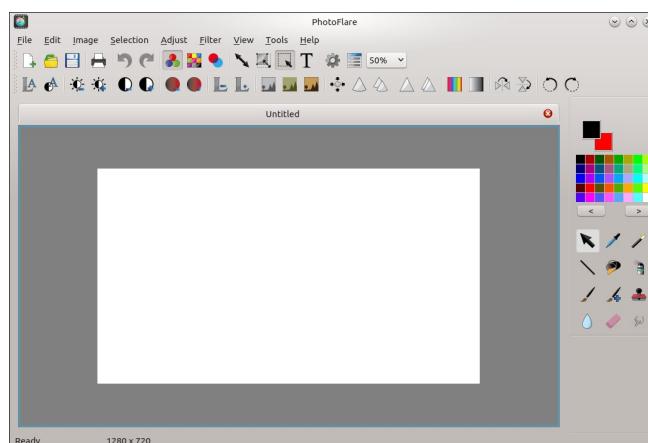


*Posted by itshondo, July 25, 2019, running KDE.*

# Repo Review: PhotoFlare

by CgBoy

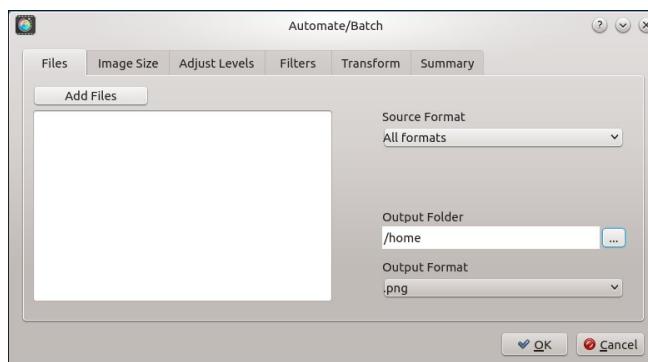
This month I'm taking a quick look at PhotoFlare, a simple, but powerful image editor intended to be very quick and easy to use. Some of its features include color levels adjustment, image filters, batch image processing, automatic cropping, and many more. It currently does not support any RAW image formats, or image layers, though.



The interface is well designed, making the program very easy to use, and the layout is also customizable. Except for a few tools, most of PhotoFlare's image editing features can be easily accessed from the top and side toolbars.

The toolbar on the side contains the color selector and some of the basic image manipulation tools. You have access to the pointer tool for making selections, a color picker, the magic wand tool for making automated selections, a line drawing tool, a paint fill bucket, a spray paint tool, two kinds of paintbrushes, a cloning tool, a blur tool, an eraser, and a smudge tool.

The toolbars at the top contain the buttons for adjusting the brightness, contrast, saturation, gamma, hue, and image size and orientation. Text can be inserted, and a number of filters, such as grayscale, sepia, noise reduction, softening, and sharpening, can also be applied from the toolbar. All the rest of PhotoFlare's filters can be accessed from the Filter menu.



PhotoFlare has a useful tool for batch image processing, allowing you to apply filters and transformations to multiple images at once, and then save them to an output folder in the format of your choice. You can set it to adjust the image size, levels (brightness, contrast, saturation, and gamma), image orientation, and you can apply any of the available filters.

## Summary

If all you need is a simple program for editing and adjusting your images, I think PhotoFlare is an excellent choice. I did run into a few bugs occasionally, but for the most part it worked well enough. GIMP is obviously still a better option if you need something more advanced, but PhotoFlare is much simpler and easier to use.

A collection of PCLinuxOS merchandise items. It includes a white polo shirt, a black t-shirt, a white mug with the text "PCLinuxOS radically simple", a tan baseball cap, a pair of white shorts, and a small rectangular sticker or patch with the PCLinuxOS logo. Below the items is a promotional message.

Support PCLinuxOS! Get Your Official  
**PCLinuxOS**  
Merchandise Today!

# PCLinuxOS Family Member Spotlight: MarekTux

As told to YouCanToo



**What is your name/username?**

Markus Henseleit/MarekTux

**How old are you?**

44 years old.

**Are you married, single?**

Married to a Filipina, but she is still in the Philippines, so currently I'm living alone.

**How about Kids, Grandkids (names and ages)?**

Nope.

**Do you have pets, what is your favorite?**

Nope.

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

Still working. After a brief period of unemployment, I'm currently occupied as an employee in the IT department of a small company dealing with digital forms and digitalization in general,

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

My home is where I hang my hat - wait, that's a hatstand...

Currently here in Berlin, Germany, Europe, Sol III



Berlin



Berlin

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

I went to school here in Germany, Northern Germany to be exact. I reached the equivalent as graduating from high school.



Brandenberg Gate

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

I enjoy all kind of history, I often go to museums and read or hear audio books in a rather large field (mystery, crime, science-fiction, fantasy...), preferably in English, but also some in German.

**Why and when did you start using Linux?**

Along time ago (in a galaxy far, far away) - actually, my last Windows at home was Windows 2000... Eventually, after a long period of distro-hopping, I stuck to PCLinuxOS, first for the 32-bit edition - and I haven't regretted it.

**What specific equipment do currently use with PCLinuxOS?**

On my desktop.



Linux Docs  
Linux Man Pages

Do you feel that your use of Linux influences the reactions you receive from your computer peers or family? If so, how?

Not too much, to be honest. I've no intention any more to persuade them to another OS, even if most things are easier.

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

Great idea. But honestly, I cannot think of anything that is truly lacking within PCLinuxOS.

*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.*



**TorrentFreak**  
The place where **breaking news**,  
**BitTorrent** and **copyright collide**

## Screenshot Showcase



Posted by Mr Cranky Pants - YouCanToo, July 1, 2019, running KDE.



# ***Short Topix: Vulnerability Discovered Masquerading As Gnome Extension***

by Paul Arnote (parnote)

**Mozilla, Firefox Branded “Internet Villains”**



On July 2, 2019, the U.K.’s Internet Service Providers’ Association (ISPA) [announced](#) their candidates for Internet Hero of the Year, as well as their candidates for Internet Villain of the Year. Amazingly, Mozilla/Firefox was listed among the “villains.” Here is a snippet from their announcement:

*‘Mozilla – for their proposed approach to introduce DNS-over-HTTPS in such a way as to bypass UK filtering obligations and parental controls, undermining internet safety standards in the UK.*

So, what is DNS-over-HTTPS? Well, according to [Wikipedia](#), **DNS over HTTPS (DoH)** is a protocol for performing remote Domain Name System (DNS) resolution via the HTTPS protocol. A goal of the method is to increase user privacy and security by preventing eavesdropping and manipulation of DNS data by man-in-the-middle attacks.

Typically, DNS information is passed between computers in an unencrypted state. Because of this, users can be more easily tracked, and their actions monitored. According to a TechCrunch [article](#), under the law in the U.K., websites can be blocked for facilitating the infringement of copyrighted or trademarked material, or if they are deemed to contain terrorist material or child abuse imagery.

By encrypting DNS queries, it’s claimed that it will make it more difficult for internet providers to filter their subscribers’ internet access. The ISPA isn’t alone. U.K. spy agency GCHQ (the U.S. NSA’s U.K. brethren) and the Internet Watch Foundation, which maintains the U.K.’s internet blocklist, have criticized the move to roll out encrypted DNS features to the browser.

Really? This is what you get when you have ISPs caught between protecting user privacy and policing users to remain in compliance with heavy handed government regulations. Any reasonable person would expect that user privacy would be front and center. But then again, if they do not remain in compliance with the heavy handed government regulations, they won’t be in business any longer either. So, in effect, it is a Catch-22, a complete “damned-if-you-do, damned-if-you-don’t” situation.

Mozilla isn’t the first to implement DoH. Cloudflare has been offering DoH since April, 2018 with their 1.1.1.1 DNS service. We first told you about it [here](#),

in the May, 2018 issue of The PCLinuxOS Magazine. Millions of users are currently waiting for Cloudflare to roll out its recently announced forthcoming VPN service, so there is little doubt that users are clamoring for improved internet security.

**More Than 1,300 Android Apps Circumvent Permissions**



According to a multinational research [document](#) (PDF) from the International Computer Science Institute, called “50 Ways to Leak Your Data: An Exploration of Apps’ Circumvention of the Android Permissions System,” 1,325 Android apps

## Short Topix: Vulnerability Discovered Masquerading As Gnome Extension

circumvent the permissions granted to them by users. They looked at 88,000 popular apps from the Google Play store, across all categories. You would think that once a user denies or grants specific permissions to an app, that would be the end of the discussion.

But ... Nooooo! Many of the 1,325 apps either continued on as if the user never denied permissions (such as access to location data, WiFi, contacts, camera, etc.), or just piggy-backed their data extraction from other legitimate apps that were given permissions. The information was presented to the attendees of the Federal Trade Commission's PrivacyCon in late June, 2019, after being reported to both Google and the FTC in September, 2018.

One app that was used as an example in the report was the Shutterfly app. Even after permissions for location sharing were denied, GPS coordinates for individual photos were still uploaded to Shutterfly's servers, and stored there. You will be in for quite a surprise if you look at the EXIF data from photos taken with your phone's camera.

From the report:

*We observed that the Shutterfly app (com.shutterfly) sends precise geolocation data to its own server (apcmobile.thislife.com) without holding a location permission. Instead, it sent photo metadata from the photo library, which included the phone's precise location in its exchangeable image file format (EXIF) data. The app actually processed the image file: it parsed the EXIF metadata—including location—into a JSON object with labelled latitude and longitude fields and transmitted it to their server.*

*While this app may not be intending to circumvent the permission system, this technique can be exploited by a malicious actor to gain access to the user's location. Whenever a new picture is taken by the user with geolocation enabled, any app with read access to the photo library (i.e., READ\_EXTERNAL\_STORAGE) can learn the user's precise location when said picture was*

*taken. Furthermore, it also allows obtaining historical geolocation fixes with timestamps from the user, which could later be used to infer sensitive information about that user.*

Yessiree. Encoded within each photo taken with your cellphone's camera is the exact GPS coordinates within the EXIF data that is stored in the JPG files detailing exactly where that particular photo was shot. It won't show up if you just look with the EXIV2 command line tool, which seems to promote itself as an enhanced EXIF tool. You must look with the EXIF tool, specifically. Install the EXIF tool from the PCLinuxOS repository, and (from a normal command line), run **exif -m yourphoto.jpg**. You're likely to be as surprised as I was. To be totally honest, I had never really given it any thought before that my photos taken with my cellphone's camera contained geolocation information that virtually anyone could access.

But the Shutterfly app wasn't just guilty of uploading the EXIF information that's stored in the header of each JPG file. Nope. The Shutterfly app specifically extracted the GPS data (along with other EXIF data) and uploaded a JSON object to the Shutterfly servers. While Shutterfly isn't using this information for nefarious or evil purposes, it perfectly illustrates a vulnerability that most users won't even have a cursory or glancing thought about.

Google has responded to the report, and said that these issues will be addressed (at the earliest) in Android Q, which is due to be released later this year.



Vulnerability Discovered Masquerading As Gnome Extension



CAUTION! There is a new vulnerability for Linux masquerading as a GNOME shell extension. Fortunately for PCLinuxOS users, the incidence should be minuscule, due to the fact that GNOME isn't one of the desktops offered on PCLinuxOS. I first saw this [reported](#) on BleepingComputer.com, stumbling upon subsequent articles on [OfficialHacker.com](#) and [TheHackerNews.com](#).

What makes things worse is that it is NOT detected by any of the major virus scanners. Called EvilGnome, it's installed as a self-extracting archive, camouflaging itself as a GNOME shell extension. Once installed, it sets up cron jobs that take

## Short Topix: Vulnerability Discovered Masquerading As Gnome Extension

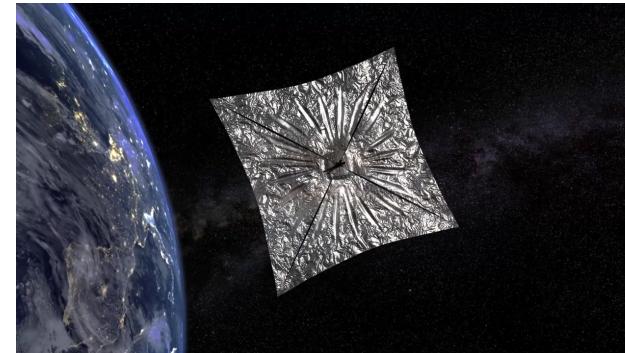
screenshots of the user's computer, captures audio from the user's microphone, and looks for newly created files on the user's computer. It then sends that gathered data to the owner's command-and-control server. It can also ping the owner's server, and download and execute additional payloads. The malware also contains an uncompleted and yet-to-be-implemented keylogger.

EvilGnome has been linked back to a Russian threat group, known as Gamaredon Group. They are an active and persistent threat group that has been around since 2013.

Since it has thus far evaded the major virus checkers, just how do you know if you've been compromised? Fortunately, that's fairly easy. Take a look in `~/.cache/gnome-software/ gnome-shell-extensions`, and look for the "gnome-shell-ext" executable. If you don't find it, then you're in the clear. The malware stores its collected data in the `~/.cache/gnome-software/gnome-shell-extensions/tmp/` folder on infected Linux computers. EvilGnome connects to its command-and-control servers via SSH, over port 3436.

This newest exploit perfectly makes the case for how dangerous it is to download and install software from outside of the official PCLinuxOS repository. By sticking to the official offerings in the PCLinuxOS repository, you virtually eliminate the possibility of becoming one of its victims.

## LightSail 2 Set To Prove Value Of Solar Sails



The Planetary Society launched a small cubesat on June 25, 2019, from a Falcon Heavy rocket that launched from Kennedy Space Center in Florida. The satellite was deployed July 2, 2019 from another satellite that was designed by Georgia Tech students. The solar sail was successfully deployed on July 23, 2019.

Roughly thirty days after deployment of the solar sail, the mission plan is to raise the satellite's orbit using the solar sail as the only propulsion. Solar sails work by using only the impact of the Sun's photons upon the surface of the extended sail material.

LightSail is a crowd-funded venture (raising over \$1 million with a KickStarter campaign) that sets out to prove the value and usefulness of using solar sails as the sole propulsion source of small cubesat satellites. The Planetary Society has advocated for the use of solar sails for decades. They found their first practical success in 2005 with the launch of Cosmos 1, a solar sailing spacecraft. However, it was lost due to a rocket failure. In 2015, LightSail 1 successfully completed its test flight, paving the way for this current endeavor.

Anyone with an internet connection can follow along with LightSail 2's progress by [visiting](#) its Mission Control Center. The Planetary Society, under the

leadership of CEO Bill Nye (yes, THAT Bill Nye, of television fame from the Bill Nye The Science Guy fame), is the world's largest and most influential non-profit space organization. The organization is supported by over 50,000 members in over 100 countries, and by hundreds of volunteers around the world.

## Mad Magazine Closes Shop After 67 Years



After 67 years, [MAD Magazine](#) is essentially closing down. Well, not completely, but at least in a very significant way compared to how we have come to know them. Alfred E. Neuman and company's monthly satirical review will sell its last monthly magazine on newsstands in August, 2019.

Started in 1952, the magazine features irreverent, satirical humor targeting current events and societal trends. Nothing is or was sacred. They even often poked fun at themselves. Along the way, it has wielded significant influence in helping either fuel the current debate(s), or shaping the debate in a different light. But, in today's culture where runaway, rampant political correctness rules the discussions, it seems many people can't take a joke or see the humor in many topics of the day.

Want to keep up on the latest that's going on with PCLinuxOS?

Follow PCLinuxOS on Twitter!

<http://twitter.com/iluvpclinuxos>

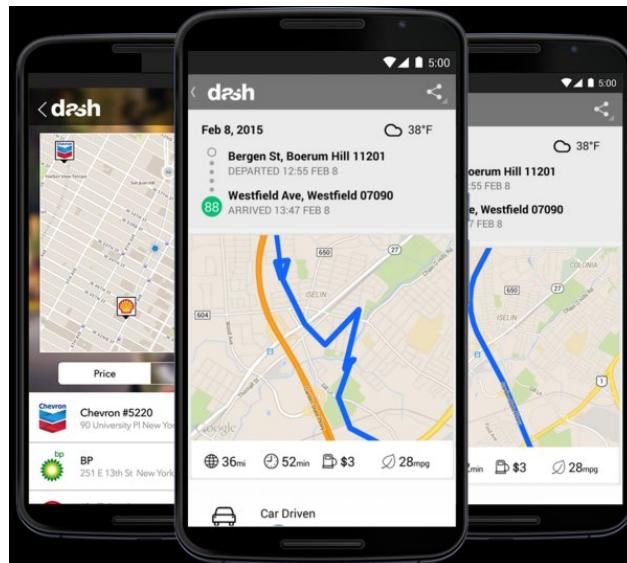
## Short Topix: Vulnerability Discovered Masquerading As Gnome Extension

Starting with 2019's issue 10, it will only be available in comic book shops, and mailed to subscribers. Starting with 2019's issue 11, there will no longer be any new content, except for end-of-the-year specials. From issue 11 onward, they will only feature previously published content, drawn from its massive vault of material gathered from the past 67 years. DC Comics, however, will continue to publish MAD books and special collections.

Although it started out as a comic book, it quickly changed its format to a bimonthly magazine. Its covers, featuring Alfred E. Neuman with his gap-toothed smile, were eagerly anticipated by its fans. It was very influential for successive generations of comedians, writers, artists and performers.

With the state of the world being what it is, we need the irreverent and satirical humor of MAD Magazine more than ever. It's saddening that such a publication will be closing up shop. We need you A. E. Neuman, more than ever!

### Are Notifications On Google Maps Driving You Crazy On Your Android Device?



If you use Google Maps on iOS, you won't have this problem; for some reason, iOS users are immune from this problem. But, if you use Google Maps on an Android device, this problem is enough to DRIVE. YOU. CRAZY. There are 54 different notifications for Google Maps, and by default, they are ALL turned on. Oh, yay! What a treat for users of Google's own OS!

These notifications from Google Maps can quickly overwhelm users. At minimum, they crowd out more important notifications, becoming more of an annoyance than being helpful.

You could tap on the hamburger menu at the left upper corner of the Google Maps window, select Settings, then Notifications, and then begin to plow through all of the notifications that Google Maps has turned on. If you choose this route (or forced to because of the version of Android you use), be prepared for an arduous journey through "notification hell." Many of the notifications are obscurely described, so you may never be sure if they are turning off something helpful, annoying, or that doesn't even apply to you and your life. You'd probably have better odds playing craps at the local casino.

But, if you use Android 9 Pie, there is an easier and faster way to turn off whole groups of notifications. By turning off whole groups of notifications, you don't have to wade through each and every one of the 54 different notifications.

To accomplish this, follow the steps below.

First, "long press" the Google Maps icon.

Second, in the little window that pops up, click on the "information" icon near the top. It looks like an "i" with a circle around it, a lot like a copyright or trademark symbol. This will take you directly to the "app info" screen.

Third, tap on "Notifications" in the "app info" screen.

Fourth, and lastly, you should find all of the notifications divided up into categories. This makes it much easier to turn off entire groups of notifications that you don't want to be bothered with.

For example, I never use public transportation, so that whole group of notifications was a no-brainer for me. Another one for me was Google Maps telling me, every morning when I go to work, how many minutes it will take to get there. Uhm ... that never, ever changes, and it's just a nuisance notification. So those types also got turned off, too.

Now, I'm hoping that I'm feeling a lot less "love" from Google Maps. I've tested this, and it does work (I'm running Android 9 Pie, on a Google Pixel 3 phone). The nice thing about using this method is that if you turn off notifications that you find useful, you can repeat the steps above to turn them back on.



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

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

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

# The Ruby Programming Language: Blocks, Modules and Other Interesting Things

by phorneker

## Recap on Control Statements

So far, we have learned how to use:

- **if/then/else**
- **while/do**
- **unless/do**
- **for/do**
- **until/do**

with the **end** statement indicating the end of the control statement. We also learned that while the keyword **do** is *optional*, it is a good programming practice to use the keyword to make it clear where the code to be executed begins for that control block.

## Generic Blocks

Did you know that blocks of code in Ruby *do not have to be inside of a control statement or loop*?

**Generic blocks** are blocks of code that are given a name. *This is unique to Ruby*. When you define a function or procedure (method in object oriented terms), we use this form:

```
def <function name>  
<place code here>  
end
```

Ruby provides another kind of **generic block** that is defined using the same function name as you used when you defined the function or procedure. The difference here is that this form is used:

```
def <function name>
```

```
<place code here>
```

```
end
```

```
<function name> { <generic block code placed here> }
```

The **yield** statement invokes the code embedded in the **{}**. This is useful for statements and functions *used inside the function or procedure we are defining* that tend to repeat. One useful application for this is to display progress of a function or procedure. For this to happen, we will need to include a parameter in the **{}** like this:

```
<function name> {|<parameter>| <generic block code placed here> }
```

## Hello World revisited

Let us take that famous “Hello World” program we started this series with, and rewrite it using a **generic block**.

```
#!/usr/bin/ruby  
  
def test  
  yield  
end  
test { puts "Hello World!" }
```

This code executes the same as the original “Hello World” program at the beginning of this series. If you think this is a weird way of implementing a hello world program, we could write this another way:

```
#!/usr/bin/ruby  
  
def test(&block)  
  block.call  
end  
test { puts "Hello World!" }
```

The **&block** defines a parameter that happens to be a parameter containing the pointer where the **generic block** is defined. Like everything else in Ruby, **block**



# The Ruby Programming Language: Blocks, Modules and Other Interesting Things

is an object, which happens to contain a method that is defined where **test** is invoked. The **call** is a method that tells Ruby to execute the code contained in the generic block defined where **test** is invoked.

*With this style of coding, it is possible to call **test** using various generic blocks redefining the method being called every time **test** is executed.*

**Mind your step when using this type of coding, as it could lead to hard to debug errors in your Ruby code, unless you pay close attention to what is contained in the generic block for each instance of test being invoked.**

## BEGIN and END are special blocks

The keywords **BEGIN** and **END** are considered to be **reserved words** in Ruby, and are spelled **in all capital letters**.

Blocks that are defined as **BEGIN** are executed **before** the main block of code is executed. Likewise, at the end of the main block, Ruby executes blocks that are defined with **END**.

*You can have more than one block called **BEGIN** and more than one block called **END** in your Ruby code.* Ruby scans the source code for blocks that are defined with **BEGIN** first and are executed **in the order they are encountered**. Ruby also scans for **END** statements and when the main block is finished, the **END** blocks are executed **starting with the last END block defined and going in reverse direction until the last END block is executed**.

*Mind your step here, too, as there could be a slight delay in execution of the Ruby code as Ruby has to scan the source code for BEGIN and END blocks before beginning execution of your program. The more BEGIN and END blocks you include in your program, the better chance there will be a delay in your program's execution.*

## Modules: The Moment we all have been waiting for.

So far, we have learned to write Ruby code as a single source code file containing methods (as well as functions and procedures as we refer to them in with traditional programming), global and local variables, constants, and classes.

Think of a **class** as a container that encapsulates methods and variables local to that class. But, what if there was a container that packaged global variables, constants and classes into a single entity?

Ruby provides just that...in the form of **modules**.

The concept of Ruby **modules** is nothing new.

In C and C++, we compile this type of grouping into **libraries**. For PCLinuxOS, using these libraries in a C or C++ program requires installation of the library **and its associated “-devel” package** from Synaptic before that library can be used.

For Python, its modules are available in Synaptic with the module name prefixed by either **python-** or **python3-** (depending on which version of Python you are using). The same goes for PERL, with the module name prefixed with **perl-**.

In FreePascal, modules are compiled as **units** (with the **.ppu** extension). The concept of **units** in Pascal was introduced with Turbo Pascal 4.0 for DOS. (Files here are compiled with the **.TPU** extension).

For Java, modules are stored in Java Archives (those files with the **.jar** extension).

Implementing **modules** is similar to implementing **classes** as far as source code goes. Instead of typing **class <class name>**, we type in **module <module name>**. A good structure for a module is as follows:

```
module <module name>
<constants defined here>
<variables local to the module defined here>
<classes defined here>
end # <module name>
```

For the **end** statement, I recommend including the module name here so we know what this particular **end** statement refers to.

The Ruby source file can have the same name as the name for the module being defined. This would be ideal if that module is *the only module implemented in the source file*. However, there **can be more than one module defined in a Ruby source file**.

As with classes, invoking methods (i.e. functions or procedures) in modules is accomplished by typing:

```
<module name>. <method>
```



# The Ruby Programming Language: Blocks, Modules and Other Interesting Things

This only works when the module being accessed is in the same source code file as the block of code that called the method inside the module.

But what if the module is located in another source code file? The **require** statement attempts to resolve that problem. Think of **require** as an **include** statement in C or C++, or as the **import** statement in Java.

The **require** statement (guess what) requires a name of a source code file that contains the module required by your Ruby program. **If no file extension is supplied, Ruby will assume that the filename has the .rb extension when retrieving the source file.**

Hence, it is imperative that all Ruby source files have the .rb extension when such files are saved.

Near the beginning of your program's source file, just below the "#!/usr/bin/ruby" statement, you will need to include a **\$LOAD\_PATH** constant as follows:

```
$LOAD_PATH << '<path to source files>'
```

where *path to source files* is a UNIX path, not unlike what we use in **bash(1)**. A single dot ('.') tells Ruby to search the current directory for a file containing the requested **module**.

A variant on the **require** statement, called **require\_relative** tells Ruby to search for the requested file or module **relative to the current directory the Ruby program is being executed from**.

## One more thing about modules

Just as we use the **firejail** package from Synaptic to launch applications inside their own environments (kept away from each others space), Ruby **modules** are invoked in a similar way. Each **module** is executed in its own environment provided by the Ruby interpreter.

It is possible for two **modules** to have the same name for a defined method. By running the modules in separate environments, each method executes in its own space **without interfering with each others internal data or code**.

## Classes and Modules: What's the Difference?

The **include** statement requires the name of a module, rather than the name of a source file as in C and C++. If the required **module** is contained in another

source file, the **include** statement **must be preceded by a require statement containing the name of the source file where the module resides**.

When **include** is used inside a **class** definition, it is possible for a **class** to inherit properties from *multiple modules*.

So how is a **module** different from a **class**?

First, **modules**, as discussed earlier, are executed separately from one another, in separate environments provided by the Ruby interpreter. **Classes** are created separately from one another, but they are created *in the same environment* provided by the Ruby interpreter.

**Classes** are extensible entities in the sense that *descendants* of classes *inherit* the properties of the parent class. **Modules**, however, are **not** extensible. The Ruby way to extend a module is to create a **class** that includes the original module(s) and then define the extensions in the class.

## Don't Forget the Main Block

We can have a boat load of classes and modules in our Ruby program. **But nothing happens here until we implement a main block to bring it all together.**

Remember the "Hello World" program at the beginning of this series? This is an example of a **main block**, or that block of Ruby code that *actually gets executed* when we run the Ruby program.

Usually, this block appears near the end of the source code file for the program that we are developing.

*This code does not begin with anything, nor does it end with anything, unlike control loops, classes, modules, and generic blocks. You simply enter the Ruby code needed to implement the program that controls all classes, modules and everything else.*

It is *this code* that Ruby executes when the program is run, **unless there are defined blocks that start with BEGIN (in all capital letters)**, in which case all such blocks are executed first **before** the main block is executed.

At the end of the main block execution, Ruby will end the program, **unless there are defined blocks that begin with END (in all capital letters)**, in which case, execution of these blocks follows the execution of the main block of the Ruby program.



# The Ruby Programming Language: Blocks, Modules and Other Interesting Things

As stated earlier, Ruby executed END blocks *starting from the last END block encountered by the interpreter* and working its way backwards through the source code.

## File and Directory Management in Ruby

One of the modules that is a part of the Ruby distribution contains the **IO Class**. Up to now, we have already used some functions from this class, namely: **puts**, **gets**, **putc**, **getc**, and **print**.

These methods in the **IO** class get their input and output from UNIX **stdin** and **stdout** devices, which are normally assigned to the keyboard and screen respectively.

At the beginning of this series, when a Ruby source file is assigned the **executable** attribute, i.e. we type:

```
chmod a+x <name of Ruby program>
```

on a command line, **bash(1)** will automatically launch the Ruby interpreter and load the source file for invocation. As one would expect, the < and > work as expected from the command line using the specified files for input and output respectively.

Mind though, this only works when the Ruby source file has the **executable** attribute set.

In addition to I/O programming using methods that use UNIX **stdin** and **stdout**, Ruby provides methods for working directly with files and directories. The latter is preferred for working with disk files as the basic I/O methods that use UNIX **stdin** and **stdout** (i.e. **putc**, **puts**, **gets**, **getc**, and **print**) only works with ASCII text files.

(After all, we could develop applications that need to read files in binary formats such as those from LibreOffice, GIMP, WordPerfect, Audacity, or even compressed tarball archives.)

**File** and **Dir** are objects that are part of the **IO Class**. For the following methods, **name** refers to the UNIX filename to be used for the file. The parameter **mode** can be one of the following: **r** for *read only*, **w** for *write only*, **r+** for *read file* with the option to create the file if the file specified by **name** does not already exist, **a** for *append*, i.e. open to the end of the existing file to append output.

The following methods belong to the **File** class.

**File.new(name,mode)**

This method is used to **create a new disk file**. This function returns a **file handle** (usually an integer) if the file could be created, or **false** otherwise (a result of zero).

**File.open(name,mode)**

This method opens a disk file or whatever the parameter **name** is associated with.

**File.open(name,'r+')** is similar to **File.new(name,'w')** as they do the same thing when executed. But that is where the similarity ends.

First, when a new file is created with the **File.new** method, the intention is to create a disk file for **writing**. As everything in UNIX is a file, it is possible to use **File.new** to create a UNIX pipe for communication with another application.

However, **File.new** cannot be used to create files that represent block devices. You can create disk files on a hard drive, but you cannot access partitions of the hard drive with **File.new!**

The method **File.open** however, can be used to open anything that is represented as a UNIX filename, e.g. disk files, TCP/IP ports, UNIX sockets, UNIX pipes, serial ports (old school), USB devices (e.g. /dev/ttyUSB0 to access a Handspring Visor through a USB to Serial adapter).

If successful, the method will return a **file handle** (usually an integer), or **false** (a result of zero) otherwise. For these methods, **you will need to assign these functions to a variable in order to be able to work with disk files or devices in Ruby**.

**Filevar.sysread(bytes)**

When successful, this method will read the number of bytes specified by **bytes** from the file or device, and will return the result of that reading to the variable assigned to this method.

**Filevar.syswrite(content)**

When successful, **content** is written to the disk or device.

**Filevar.each\_byte{|parameter| generic block }**



# The Ruby Programming Language: Blocks, Modules and Other Interesting Things

This method processes the entire file, byte by byte, executing Ruby code implemented as a **generic block** with one parameter, namely the variable the file will read to and use for input to the **generic block**.

The next two methods were intended to be used with ASCII text files, and no **file handle** is needed to use these methods.

`IO.readlines(filename)`

This method returns an array of strings. This method is assigned to an array variable where the index for the array is the line number of the text file (starting at zero). Use `array_variable.size` to get the number of lines contained in the text file.

`IO.foreach(filename){|parameter| generic block }`

Similar to `IO.readlines`, this method reads ASCII text files, line by line. However, a **generic block** is used to process the text file, and **parameter** contains the line currently read from the text file, and the **generic block** is executed one time for each line in the text file. Execution ends when the EOF (end of file) character is read.

The next method is the Ruby implementation of the **chmod** command in **bash(1)**.

`Filevar.chmod(permission)`

The parameter **permissions** is an **octal** number (base 8) that just happens to be the same octal number we use to change permissions in the command line.

For example: `Filevar.chmod( 0755 )` is the same as `chmod 0755 filename` on the command line.

## File inquiry functions in Ruby

`Filevar::file?(filename)`

This function tests to see if **filename** really is a **disk file**, and not something else such as a TCP/IP port, or a pipe or socket, or a physical device.

`Filevar::directory?( directory name )`

This function tests directory name to see if it is a **directory** (or folder as some people call it) on disk.

`Filevar::readable?(filename)`  
`Filevar::writeable?(filename)`  
`Filevar::executable?(filename)`

These functions test for permissions to read, write, and execute the file respectively. The parameter passed to these functions does not have to be a disk file. It could be a directory, a physical device, a TCP/IP port, socket, or a pipe.

*In the case of a directory, the test for **executable** returns true if permission to **search** the directory is granted. The **executable** test was intended for disk files. It does not make any sense to set executable permission on a port, socket or pipe.*

`Filevar::zero?(filename)`

This function tests to see if there is anything in the file or directory, i.e. the function will return **true** if the reported size is **zero**.

When downloading large files In Firefox, two files are created. The first is the filename that will be assigned to the file when the downloading is finished. The other filename is the same as the first, except it has a **.part** appended to the name. This file reports how many bytes were downloaded so far, and changes during the download.

The `::zero?()` function will return **true** for files that are currently being downloaded from the Internet, such as files without the **.part** file extension. This is particularly true when downloading PCLinuxOS ISO files using Firefox.

`Filevar::size?(filename)`

This function returns the size of **filename** in bytes.

`Filevar::ftype(filename)`

This function returns the type of file associated with **filename**. Valid values returned by this function are: **file**, **directory**, **characterSpecial**, **blockSpecial**, **fifo**, **link**, **socket**, or **unknown**.

The values **file** and **directory** are self explanatory.

The values **characterSpecial** and **blockSpecial** refer to physical devices such as **/dev/ttyUSB0** (usually associated with a Serial to USB adapter).

The value **fifo** is another term for a UNIX pipe. (fifo means **First In First Out** when it comes to data transport)



# The Ruby Programming Language: Blocks, Modules and Other Interesting Things

The value **link** means that this file is a **symbolic link** to a file located elsewhere in the system.

The value **socket** refers to a UNIX socket.

The last value **unknown** could mean a TCP/IP port, a UDP/IP port, or even something that **simply does not exist on the system**.

## Directory Methods in Ruby

### `Dir.chdir( directory )`

This method does the same thing as typing **chdir directory** on a command line.

### `Dir.pwd`

This method returns a string containing the name of the current working directory. This is the same as typing **pwd** on the command line, except the name of the directory is assigned to a variable or output with the **puts** method.

### `Dir.entries( directory )`

This method returns an array of strings with each string containing the name of a file that resides in *directory*.

For instance, `Dir.entries("/home/patrick").join(' ')`

will list all files in my home directory on my laptop *with a space separating each entry in the list*.

Another (cleaner) way of listing files in a directory is with the following Ruby code:

```
Dir.foreach("/home/patrick") do |entry|
  puts entry
end
```

The latter displays the contents of my home directory, except each entry is on a line by itself rather than each entry being separated by spaces.

The next two files actually belong to the **File** class, but the functions they do suggest they be placed with the directory management files.

```
File.rename( old filename, new filename )
File.delete( filename )
```

These methods rename and delete files from the directory respectively.

### `Dir.delete( directory )`

does for entire directories what **File.delete** does for individual files.

Ruby provides two aliases for this method, namely **Dir.unlink** and **Dir.rmdir** which do exactly the same thing. The former was intended to remove symbolic links to directories, whereas **Dir.rmdir** is the complement to the next method: **Dir.mkdir**

### `Dir.mkdir( directory )`

This method, as we can guess, is the same as typing **mkdir directory** on the command line.

## Not So Common File and Directory Functions

### `Filevar.rewind`

This method moves the file pointer to the beginning of the file.

### **...and another thing**

Why is there no **Filevar.close** method for files and directories? It turns out that Ruby **automatically** closes open files when a class, module, or even the program itself terminates *for any reason*.

Actually, there is a **close** method, **but only for temporary files** (read that as one time use) **that exist only when the Ruby program is executing**. (On PCLinuxOS, such files are stored in **/tmp**.)

To use anything for temporary files, you need to include **requires 'tempfile'** near the beginning of the source code file before these I/O functions can be used.

Also, **tempfile** is a separate module that comes with Ruby and is not a part of the system module that includes the **IO class** and the **File class**.

At one time, there was a function called **Filevar::exists?(filename)**. This function has been deprecated and may be removed in future versions of Ruby. To get around this, simply assign a variable to the **File.open** method and test that variable to see if there is a non-zero value. (A zero value here means that the file could for some reason not be opened, hence no file handle was assigned.)



# The Ruby Programming Language: Blocks, Modules and Other Interesting Things

The **fileutils** module comes with Ruby and contains a more complete set of file and directory functions than what you get with the **IO** and **File** classes from the system module (i.e. what is built in to the Ruby kernel). To use these functions, you will need to place a **require 'fileutils'** statement near the beginning of the source code file.

## What is a good way to design a Ruby program?

In my opinion, the top down design for programming is the best way to implement software **in any language**. Top down design has the advantage of the **divide and conquer** approach to problem solving with Ruby.

A Ruby program is divided into tasks, which could be implemented as **modules** or as **classes**, and as we have seen, can be contained in more than one source code file.

By having more than one source code file, we can implement some classes in one source file and encapsulate those classes into a single **module**, which can then be **reused by other Ruby programs**.

Whether the program is a single source file, or a series of source files, **there needs to be one source file that contains the main program block**, and that is the file that encapsulates the entire series of files, blocks, modules and classes (through **require** and **include** statements) into one program that can be executed by simply typing its name.

## The Main Source File

Having said that, one good way to design the main source file is shown here:

```
#!/usr/bin/ruby

#
# Program Name: name of your Ruby program
# Version: version number
# Author: your name here
# Filename: name of main source file here
#
$LOAD_PATH = '.' # use current directory for source files

# Global variables go here
```

# These variables are accessible to everything in the program.

# Global constants go here

# Constants are generally named in all capital letters.

# Place your "require" and "include" statements here

Require 'module name'

include 'source file'

# BEGIN blocks are executed first in the order listed here

BEGIN {

  # place initialization code here

}

# END blocks are executed in reverse order from what is listed here

END {

  # place closing code here

}

# place your local **module** and **classes** here

# place your local functions and procedures (methods) here

# ... and finally, place your main program block here

**# end program source file**

The last line in this file, **#end program source file** is *not required* but it is a good practice to make it clear that this is the end of the program source code file.

Another good practice is to name the main program source file the same as the name of the program itself. Also, be sure to save the file with the **.rb** extension so everyone knows it is a Ruby program.

Modules and classes save in different files should be **uniquely named** so as not to confuse the Ruby interpreter, not to mention ourselves, as to which source file is the main file to be executed.



# The Ruby Programming Language: Blocks, Modules and Other Interesting Things

## Design considerations

One good idea is to implement **modules** that intend to be reused in future Ruby programs.

```
#!/usr/bin/ruby

#
# Program Name: name of your Ruby module
# Version: version number
# Author: your name here
# Filename: name of main source file here
#
module modulename
# module wide constants declared here
# module wide variables declared here
# classes and local functions that are part of the module
# go here
end # end modulename
```

Another good idea is to implement classes you intend to reuse in a separate file and use the **include** statement to incorporate the classes into your main program.

Other good ideas for Ruby programming are:

- Keep the number of global variables to an absolute minimum, reducing the chance the resulting program would be difficult to debug.
- When using global variables, be sure they **really need to be global variables**. One good example of a global variable is an array or hash allocated for data storage to be used by the entire Ruby program, for example, the contents of an entire text file.
- Constants should be named in all capital letters, and those of a global nature **need to be defined only once**. After all, any global variable is accessible from anywhere in the program.
- With top down design, focus on getting the structure of the program correct, starting with the main block. Use dummy functions, i.e. those that do not really

do anything, but are useful for testing the main part of the program to be sure it is working, so that no matter what function replaces the dummy functions, the main program block **will always work**. Once you get the main block working, then focus on the smaller, but higher level blocks of code. Eventually, you will get to the minute details of the program. By then, most of the program **will work correctly**, and hence when you finish the program, there is a great chance the program will work as intended.

- Test! Test! Test! There is no better way to assure the quality of a program than to constantly test each part of the program. (After all, how many problems do large corporations face every day because the software they are running was not thoroughly tested?)
- Always document what you are coding. That is why much of the source code we see is flooded with comments. By documenting what you code, anyone else, including yourself some time in the future will be able to see what the code is intended to do.

## The Next Thing To Do...

We have learned what it is like to program in Ruby, some of it with traditional programming techniques (read as tried and true). Now, we are ready to start a real world example of Ruby programming.



**PCLOS-Talk**  
Instant Messaging Server  
*Sign up TODAY!* <http://pclostalk.pclosusers.com>



**Like Us On Facebook!**  
**The PCLinuxOS Magazine**  
**PCLinuxOS Fan Club**



**LinuxQuestions.org**





**Like us on Facebook!**

*PCLinuxOS Magazine*

*PCLinuxOS Fan Club*



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

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

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

 **PCLOS-Cloud** secure private simple-to-use  
Sign up TODAY! [pclousers.com/services-signup.php](http://pclousers.com/services-signup.php)

## Screenshot Showcase



*Posted by Snubbi, July 2, 2019, running Mate.*

# ***ms\_meme's Nook: It's Linux Time***



**MP3**

**OGG**

PCLOS I'm installing it's Linux Time  
Over the net I hear it calling it's Linux Time  
Trying to figure out GParted so I can get started  
Delighting in Linux Time

Darkness deepens on my partitions create or delete  
Resize grow or move I need a cheat sheet  
Can't figure out GParted I never will get started  
Need help with Linux Time

To the forum I now log in  
And kindly ask for help with the task  
All the members are so very kind  
Help to me they never ever decline

PCLOS I'm a using it's Linux Time  
Nothing about it is confusing it's Linux Time  
Every day I'm happy to be booting it with you  
Together at last it's Linux Time

# PCLinuxOS Recipe Corner Bonus



*from the kitchen of  
ramchu*

## Mediterranean Shrimp Linguine

### Ingredients

- 4 cups Zucchini noodles
- 2 pounds uncooked medium shrimp, peeled and deveined
- 1 medium onion, chopped
- 6 tablespoons olive oil
- 4 garlic cloves, minced
- 1 cup chopped roasted sweet red peppers
- 2 cans (2-1/4 ounces each) sliced ripe olives, drained
- 1/2 cup minced fresh parsley
- 1/2 cup white wine or chicken broth
- 1/2 teaspoon crushed red pepper flakes
- 1/2 teaspoon kosher salt
- 1/2 teaspoon dried oregano
- 1/2 teaspoon pepper
- 3/4 cup crumbled feta cheese
- 2 tablespoons lemon juice

### Directions

1. Cook Zucchini noodles in boiling water for 3 minutes, drain and rinse in cold water to stop cooking, reserve 1/2 cup cooking water.
2. In a large skillet, saute shrimp and onion in oil until shrimp turns pink. Add garlic; cook 1 minute

longer. Stir in the red peppers, olives, parsley, wine, pepper flakes, salt, oregano and pepper. Reduce heat.

3. Add Zucchini noodles and reserved water to the skillet. Stir in cheese and lemon juice; cook and stir until cheese is melted.



**The PCLinuxOS  
Magazine**

*Created with  
Scribus*



 Want to keep up on the latest that's going on with PCLinuxOS?

Follow PCLinuxOS on Twitter!

<http://twitter.com/iluvpclinuxos>

# PCLinuxOS Puzzled Partitions

	<b>1</b>	<b>8</b>	<b>2</b>			<b>4</b>	<b>5</b>
<b>9</b>				<b>1</b>		<b>6</b>	<b>2</b>
		<b>3</b>			<b>1</b>		
<b>9</b>	<b>2</b>						
		<b>8</b>			<b>9</b>	<b>2</b>	
			<b>3</b>				
	<b>7</b>			<b>4</b>			<b>3</b>
<b>8</b>			<b>6</b>	<b>2</b>			<b>9</b>
<b>5</b>							

**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 250, average score 175.

# PCLinuxOS Word Find: August 2019

## Household Items

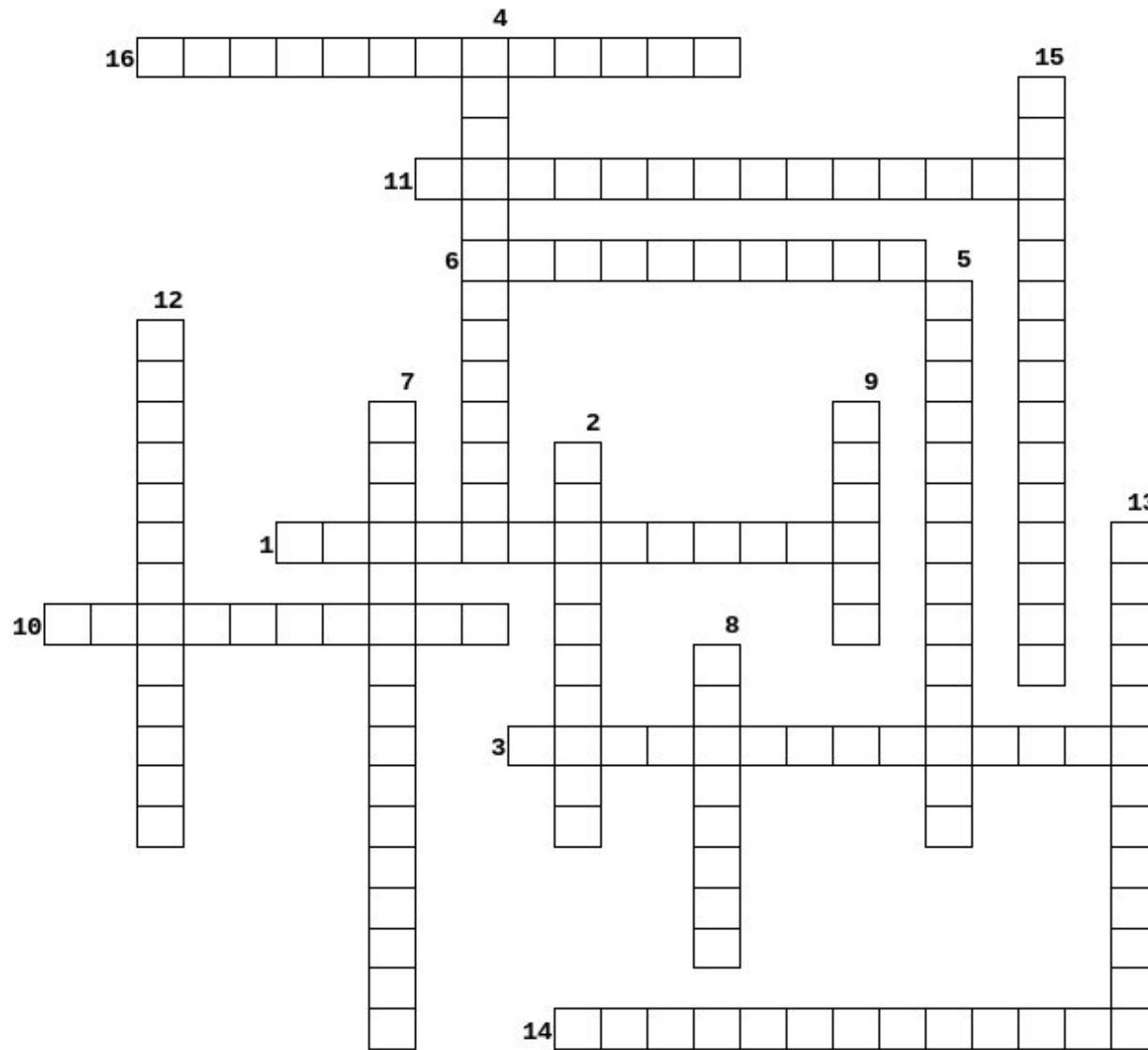
S M O K E D E T E C T O R N K K E R O S E N E H E A T E R O  
 N U S R A Q C Q S D A J F P A T T B F Z Z Y P W L C Y A X V  
 E J F I Y N V F I R E E X T I N G U I S H E R A J B F G N R  
 W V R P M M X R N E V O R E T S A O T B C H A F Q F H M A O  
 F V O F I F Z E Z W P A H R D R W K M O Z C L F X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

air conditioner	alarm clock
answering machine	barbecue grill
burglar alarm	camera
ceiling fan	cell phone
coffee grinder	convection oven
crockpot	curling iron
dishwasher	doorbell
espresso maker	fire extinguisher
food processor	freezer
hair dryer	humidifier
hot plate	ice cream maker
kerosene heater	leaf blower
microwave oven	mousetrap
percolator	pressure cooker
refrigerator	smoke detector
toaster oven	trash compactor
vacuum cleaner	video camera
waffle iron	

[Download Puzzle Solutions Here](#)



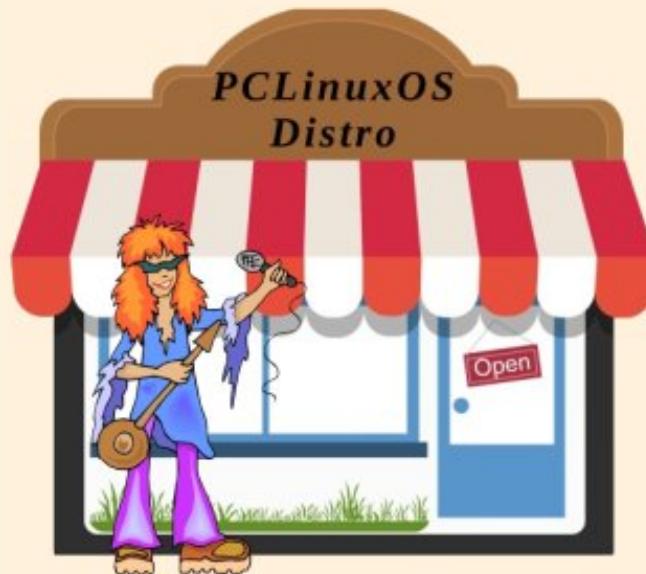
# Household Items Crossword



1. creates new fashion
2. adds moisture to the house, if needed
3. changes the temperature in your warm house
4. can heat up most any food
5. cleans the clothes
6. makes a great breakfast
7. used with your phone before voice mail
8. also called a slow cooker
9. to remember those sights you've seen
10. extra, turning addition to the barbecue grill
11. cooks dinner in record time
12. safety feature in the house that rings  
if there's a fire
13. chops up that coleslaw
14. cleans the rugs
15. liquefies food scraps
16. turns cream and flavors into a frozen treat

[Download Puzzle Solutions Here](#)

# Mixed-Up-Meme Scrambler

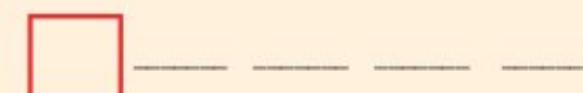


"I thought we were going  
— — — —  
dancing."

Fool  
**IDOTI**



Map  
**TARCH**



Ever  
**SLAWYA**



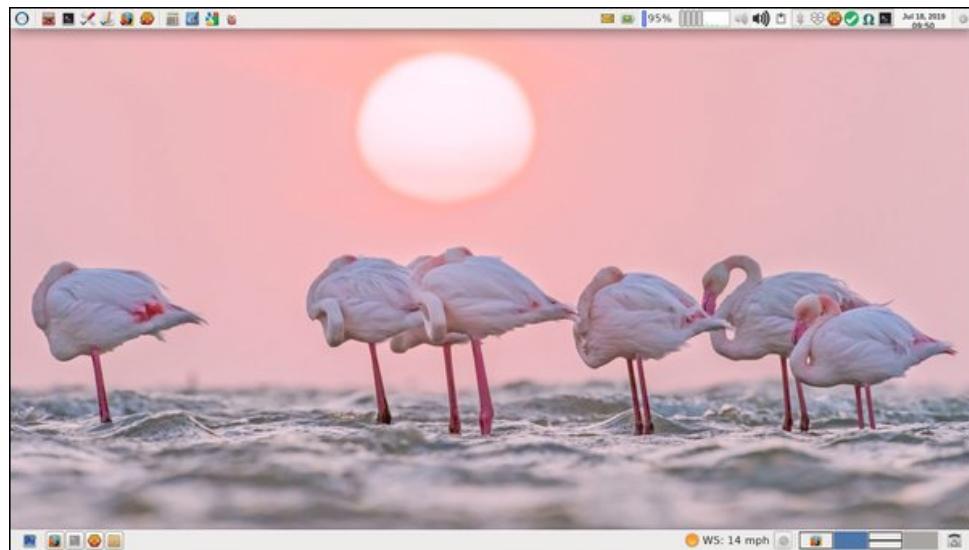
Now  
**OTADY**



Use the clues to unmix the letters to make a new word. Remix the letters in the red boxes to solve the puzzle.

[Download Puzzle Solutions Here](#)

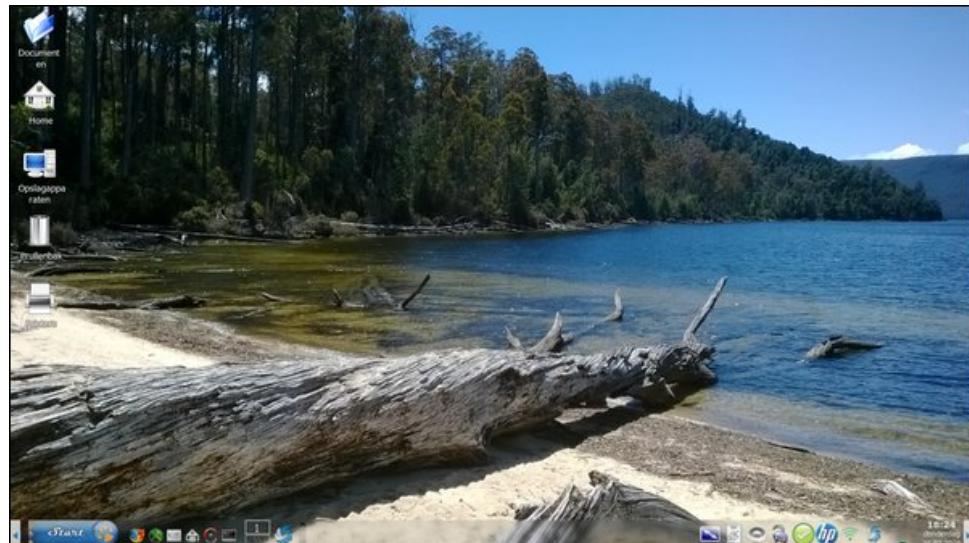
# More Screenshot Showcase



Posted by parnote, July 18, 2019, running Xfce.



Posted by OnlyHuman, July 16, 2019, running e17.



Posted by mutse, July 25, 2019, running Trinity.



Posted by Meemaw, July 16, 2019, running Xfce.