Why I Use Linux

Today’s article will be short and simple. I’ve wanted to write it for a while, and it may end up being another one of those articles that gets updates over time.

I tend to use Lubuntu and that’s my preferred desktop system – even after the change to LXQt. It’s visually simple and familiar. It’s light on resources, even though I have plenty.

I don’t use Linux because I hate Microsoft. You’ll never see me call them a derogatory name. I don’t have any major anger towards Microsoft or their products.

I don’t even care if the product I’m using is opensource. Being closed-source doesn’t bother me. I care that the application works and lets me do the things for which I installed it.

I am using a closed-source browser as I type this. Like Linux, it just works for me. It gets out of my way and lets me accomplish my computing goals.

That’s what I really like about Lubuntu. It just gets out of my way. Once it is configured, I don’t have to keep tinkering with it. I don’t have to continually pay attention to the operating system – it just keeps working and I just keep updating it.

I love the simplicity and efficacy of the terminal. When I boot my computer, a terminal emulator is one of the first things I open. I often have two or three of them open at once.

I never have any hardware issues that I can’t resolve. Sure, it may be a bit difficult to find your wireless driver – but once you do then you needn’t worry about it again. In the mean time, tether your phone and connect that way. Then again, I always have an adapter that works well enough for me to get

the drivers for any built-in hardware.

I don’t see the process as any different than Windows. You have to put some effort in to make it work. But, once it’s installed, all of my updates are done at the same time and with but a single command. The concept of a package manager is fantastic and you get a wide variety to choose from.

I like both the sense of community and the community. If I really want to put the effort in, I can find the person what wrote the driver for my wireless adapter. I don’t suppose that’s really true with any other operating system. With Linux, I can find the person(s) who put my OS together – and, in fact, I do. I talk with them at least once a week at the online team meeting. I recognize many of the names I see across the ‘net and I’ve known some of them for many years.

I suppose that I do like having access to the source code. I don’t tend to make (m)any changes, as my programming skills aren’t that great these days. Still, I do sometimes make a quick change, apply my own patch, and compile applications on my own.

I like that I have the freedom to have as much, or as little, operating system as I want. I can have a distro with everything installed or I can have a distro that barely has a terminal installed – and you’ve gotta compile that yourself. There are so many choices. There’s a Linux for everyone and, if you’re willing to learn, there are seemingly infinite combinations. I like being able to pick my desktop environment, favorite terminal emulator, favorite window manager, etc…

I like that it’s always changing. I legitimately like systemd, for example. I like learning Netplan. I like learning the new features. I like understanding what’s going on under the hood – or having access to people that can actually explain it. I also like that no matter how hard I try, I will never truly

understand everything. There's always something new to learn. There's always something new to 'geek out' with.

I guess, with the above, you could say that I like the constant innovation. Sure, sometimes Linux is trying to 'keep up' with the other mainstream operating systems – and sometimes it goes out on the edge and the community does things you simply can't find elsewhere.

Linux isn't perfect. There are bugs aplenty and flaws we'd maybe not tolerate in an operating system we paid for. Sure, we overlook the warts and call it our own – but we can call it our own. We can meaningfully contribute to a project, to a distro, to an organization, and to the community. There are so many ways that we can give back, and that is awesome.

Anyhow, I don't want to make this too long. Feel free to write a sentence or two below, letting me know why you like Linux. I don't think there's a wrong answer to this question – unless you try REALLY hard to write a wrong answer. Instead of responding where you'd normally find me, respond here so that folks can see this in the future and see your contributions to the subject.

Like always, feedback is awesome and the newsletter works. I've been wanting to write this for a while and, well, it's my site. So, I get to do things like that! If you want your own site, that can be arranged. If you want to contribute here, that too can be arranged. Until next time…

How To: Use 'find' to List

.iso files in a Directory.

In this article, we're going to do a real world exercise involving the 'find' command, because you really shouldn't process the output of the 'ls' command.

Using The Linux Find Command:

The conventional wisdom is that processing the 'ls' command's output is a Bad Idea®. Don't take my word for it, read this link (which will save me the time of repeating it). Seriously, read it. It's well worth the read.

If you absolutely refuse to read the above link, then the reason you shouldn't parse the output from 'ls' is because there are file names that will screw up the output something fierce and you won't get accurate output. On the other hand, if you take care with how you name your files then it really shouldn't be much of an issue.

So, how did we get here?

Winter is ebbing and it was not a very spectacular season this year. As winter winds down, it means that it's time for Spring Cleaning. I highly recommend folks Spring Clean their digital lives as well as their real lives. Come on now, you and I both know that you have files that can easily be deleted and save you some space. In the days of large-capacity storage, we've become digital hoarders.

Once in a while you should muck out the stalls. Which is, coincidentally, what I was doing when I decided that this would make a good article. I had close to 300 .iso files, most of which were for varied Linux distros and most of which were no longer current. Hell, some of them were for distros that don't even exist anymore. (Let's just say that it had been a while since I cleaned out that directory.)

So, I went to town and cleaned out about 200 files. Some of those files had outlasted entire hard drives, being backed up, backed up again, and backed up some more – migrating like herds of reindeer across the steppes just above the taiga. Unlike the reindeer, these files had no significant value to me and it pained me exactly none to delete them.

Now, all the files are more or less sensibly named. I could easily have just used the 'ls' command, but it's just bad form to do so. For example, I could have used:

[code]ls -la | grep iso[/code]

I could have output them to a handy text file for storage with:

[code]ls -la | grep iso >> linux_images.txt[/code]

I did not. No. No, I did not. Instead, I behaved myself and used the 'find' command. Using 'find' is a bit more complex, but it's worth learning how to use it.

The find command is a holdover from Unix. It first showed up in 1976 with Version 5. It was designed along with, and meant to be used in unison with 'cpio', which oddly didn't appear in Unix until Version 7. The cpio was all about archiving and actually still exists though you've probably never used it.

Anyhow, 'find' is defined as thus:

> *find – search for files in a directory hierarchy*

That bit about a directory hierarchy? That's actually a bit of a sticking point with the 'find' command. It insists you include the directory in the command. You can't just open the terminal, navigate to the desired directory, and run the command. You might just as well stay right there in your home directory and run the command right there! Going anywhere ain't gonna help!

I wanted to list the .iso files in the directory and so I ended up with this command:

[code]find /media/kgiii/elements4/Distros -iname "*.iso"[/code]

Yes, yes I do have 4 WD Elements external drives connected to the PC where I do a bunch of my work. Don't you judge me! It's 2021! I'll do what I want!

Anyhow, of all the modifiers to that command, the important one for this exercise is the '-iname'. That is pretty handy and it means you don't have to worry about case sensitivity.

> *-iname*
> *Like -name, but the match is case insensitive. For example, the patterns `fo*' and `F??' match the file names `Foo', `FOO', `foo', `fOo', etc. The pattern `*foo*` will also match a file called '.foobar'*

The rest is self-explanatory. I obviously want anything that ends with .iso. The wildcard '*' indicates that I want any characters in front of the .iso to be included for the purposes of making my list.

NOTE: As you can see, I used quotes around the text I wanted to be found. That's mandatory. You have to quote 'em, else 'find' doesn't do its thing.

So, what can you do with this? You can count them, if you'd like. You would have learned that back when we were talking about 'awk' and 'putting it all together'. You just need to use a pipe and then 'wc -l'. So, that command would look like:

[code]find /media/kgiii/elements4/Distros -iname "*.iso" | wc -l[/code]

(If you're curious, I'm down to just 98 .iso files in that directory. That's a pretty impressive cleaning job, if I do

say so myself!)

And, what else can you do with it? Anything you want! You tell me what else you can do with it. How else can you apply this? How do you use the ‘find’ command? How am I supposed to know what else you do with it?!? You get to decide how you use this information. Seriously, leave a comment letting me know how you can use the ‘find’ command in your daily computing.

Like always, thanks for reading. This article is a bit longer than it really needs to be. Still, you can sign up for the newsletter and get silly articles like this at least every other day. I write ’em ahead of time and use the scheduling feature that WordPress has. I’m not sure why, but this seems to make writing articles on a schedule easier. Maybe it takes the pressure off because it gives me at least two days to write an article? Dunno, I ain’t that kinda doctor!

How To: Time a Command

Have you ever wanted to know how long it takes to complete a command that you entered in the Linux terminal? Well, wonder no more!

This is going to just be a pretty quick article and easy to follow. There’s not a whole lot to explain and it’s pretty straightforward. Like often, let’s crack open your default terminal by pressing CTRL + ALT + T on your keyboard.

Now, let’s take the command:

[code]ls -la[/code]

Unless you have a lot of files, that completes pretty quickly. But, how fast does it really take? Well, simply add the ‘time‘

command before it. Time is simply described in the man page as:

> *time — run programs and summarize system resource usage*

And, for today, it's going to be pretty easy to use that command. To find out how long it took to list all the files and folders in a directory, you could use:

[code]time ls -la[/code]

Note how it tells you the time beneath the results and, if you want to try something bigger, you can take a look at this command:

[code]for i in {0..99999}; do echo "I love LinuxTips!"; done[/code]

That should take a just a little more time, but you can actually see how long it really took by adding 'time' in front of it. So:

[code]time for i in {0..99999}; do echo "I love LinuxTips!"; done[/code]

The output at the end is something like this:

> *real 0m0.566s*
> *user 0m0.423s*
> *sys 0m0.143s*

The 'real' is how much time it really took. The 'user' is how much time it took for the user. The 'sys' is how much time it took for the system – the amount of time that the kernel actually devoted to it.

So, there you have it! You can use the time command to find out how long it takes to run stuff in your terminal. If you're playing with scripting and you're looking to optimize it, this

is a valuable tool. If you're just a bit curious, then you now have a new tool.

I told you that it'd be quick and easy! Like always, thanks for reading. Feel free to sign up for the newsletter. I promise to not spam you or sell your email address.

Ubuntu Restricted Extras

As I trawl the 'net, I see some confusion about the Ubuntu Restricted Extras package. This post is meant to clear up any confusion. Given the nature of the beast, I assume this is also valid for Ubuntu derivatives – but I've not explored all of them personally (and there are many).

So, what's in the package? It's a package with more than one thing in it and will also download (if allowed) more data along the way. This is what's inside:

```
gstreamer0.10-ffmpeg
gstreamer0.10-fluendo-mp3
gstreamer0.10-pitfdll
gstreamer0.10-plugins-bad
gstreamer0.10-plugins-ugly
gstreamer0.10-plugins-bad-multiverse
gstreamer0.10-plugins-ugly-multiverse
icedtea6-plugin
libavcodec-extra-52
libmp4v2-0
ttf-mscorefonts-installer
unrar
```

Basically, it's a bunch of fonts, codecs (so that you can play

patent-encumbered media files), and the ability to open .rar compressed files and use Java applets in the browser. It’s some handy stuff, but because it’s all non-free Ubuntu doesn’t include it by default. Indeed, you won’t even find it unless you enable the ‘multiverse’ repository, like so:

[code]sudo add-apt-repository multiverse[/code]

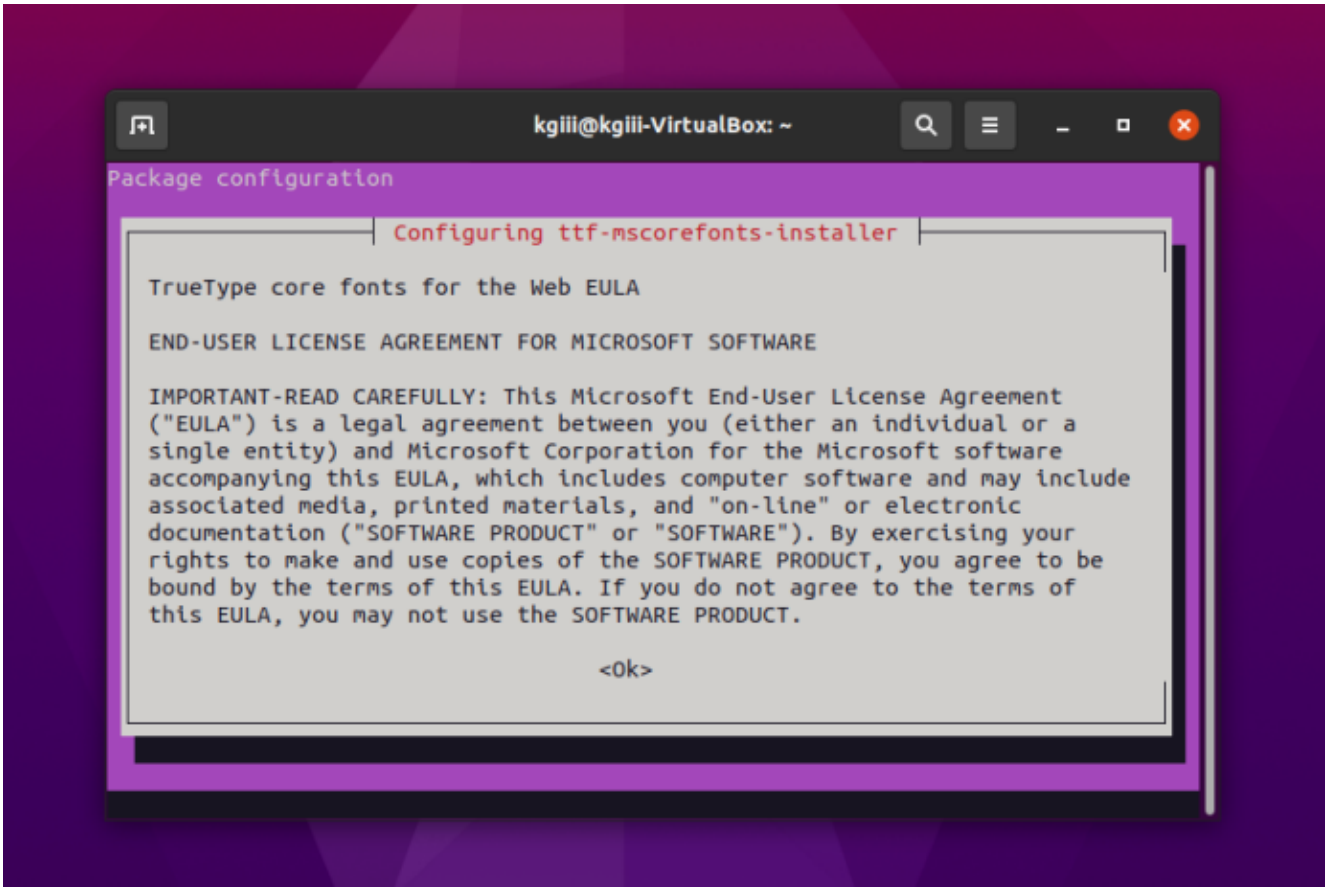
Then, you’ll need to make sure that your system knows (though some more modern releases will do this automatically) about the new software choices.

[code]sudo apt update[/code]

Follow that up with:

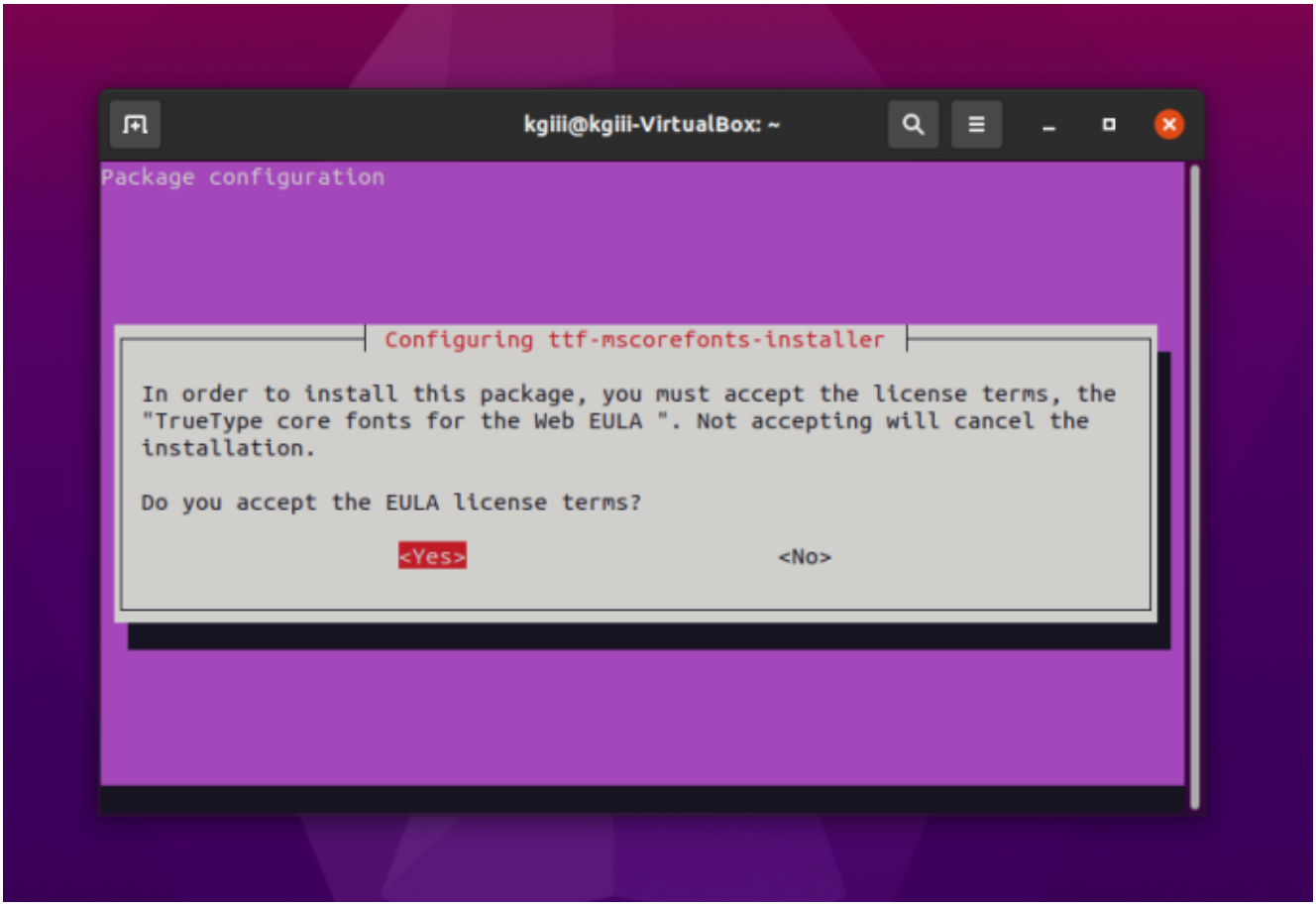
[code]sudo apt install ubuntu-restricted-extras[/code]

That’s going to download a bunch of stuff and show a screen that trips up a lot of newer users. There’s no obvious way to use a mouse and, indeed, you have to use your keyboard to agree to the user agreement. It looks like this:



Yes, you have to agree to proceed.

It’s okay. Don’t give up! You can use your TAB (or arrows) key to make selections and use the ENTER button to submit your answers. You’ll do it again on the next screen.



Use the tab or the arrow keys and the enter to agree.

At that point, it'll download some more files with fonts in them and extract them to the right directories so that you can use them. You'll also be able to use the other included files/applications, though they are not 'free' in the sense of the license agreement (which is the reason they're not included by default and you have to jump through hoops to install them).

Put it all Together

This is just a quick article that may show one of the reasons I write this material. We're just going to put together a couple of pages from this site in an example of real-world use.

So, if you remember, I told you how to use dpkg to get a list of installed applications. The command I used on that page was:

[code]dpkg -l > Documents/installed_apps.txt[/code]

Well, the output from that is more information than I really need. It looks like this:

[code]ii abiword 3.0.2-6 amd64 efficient, featureful word processor with collaboration
ii abiword-common 3.0.2-6 all efficient, featureful word processor with collaboration – common files
ii accountsservice 0.6.45-1ubuntu1.3 amd64 query and manipulate user account information
ii ack 2.22-1 all grep-like program specifically for large source trees[/code]

That's plenty informative – but I want just the name of the applications. Sure enough, we can use 'AWK' to do this.

On that page, we have this command:

[code]awk '{ print $2 }' countries.txt > finished.txt[/code]

So, let's mix the two together! We can do that!

Let's see… I'll obviously need to change the paths and file names. Coincidentally, I'll not need to change the column (the bit about { print $2 }) because I still want the second column.

What does it end up looking like?

[code]awk '{ print $2 }' Documents/installed_apps.txt > Documents/InstalledApps.txt[/code]

Now, navigate to your documents folder and open InstalledApps.txt with your favorite text editor. You'll see that it looks a bit like this:

[code]abiword
abiword-common
accountsservice
ack[/code]

You’ll still have some unwanted text at the top of the page, but it works well enough to get the job done. It’s reasons like that which motivate me to write this material.