



Full Circle

THE INDEPENDENT MAGAZINE FOR THE UBUNTU LINUX COMMUNITY

ISSUE #48 - April 2011



LINUX LABS
SWAPPINESS - PT. 1



REMASTERSYS YOUR MACHINE

CREATE YOUR OWN PERSONAL UBUNTU LIVE CD

full circle magazine is neither affiliated with, nor endorsed by, Canonical Ltd.

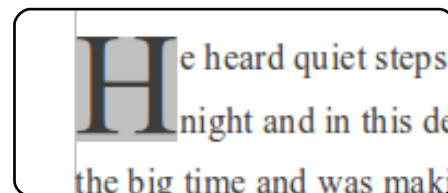


Linux News

p.04



Program In Python Pt22 p.07



LibreOffice Pt3

p.16



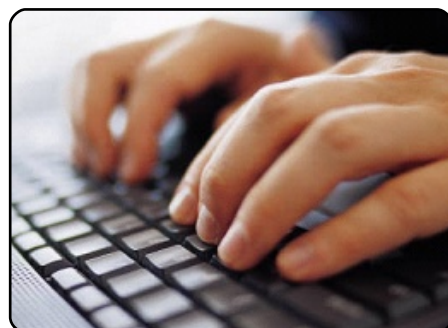
Finding Ebooks

p.19



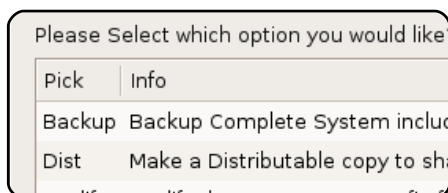
Full Circle

THE INDEPENDENT MAGAZINE FOR THE UBUNTU LINUX COMMUNITY



Write For Full Circle p.24

Guidelines for submitting an article to Full Circle. We rely on reader submissions so please write!



Review - Remastersys p.31



Linux Lab p.25

Robin Catling explains swap files in this the first of two articles about swappiness.



Letters

p.35



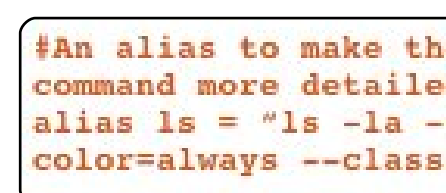
Ubuntu Women

p.37

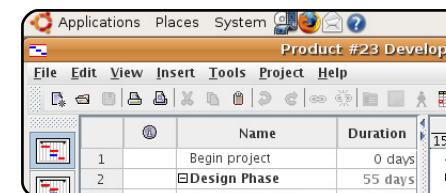


Ubuntu Games

p.39



Command & Conquer p.05



Top 5

p.44



The articles contained in this magazine are released under the Creative Commons Attribution-Share Alike 3.0 Unported license. This means you can adapt, copy, distribute and transmit the articles but only under the following conditions: You must attribute the work to the original author in some way (at least a name, email or URL) and to this magazine by name ('full circle magazine') and the URL www.fullcirclemagazine.org (but not attribute the article(s) in any way that suggests that they endorse you or your use of the work). If you alter, transform, or build upon this work, you must distribute the resulting work under the same, similar or a compatible license.

Full Circle magazine is entirely independent of Canonical, the sponsor of the Ubuntu projects, and the views and opinions in the magazine should in no way be assumed to have Canonical endorsement.



Welcome to another issue of Full Circle!

There are several notable things happening this month. First, and most important, is that we're four! Each April issue marks another milestone in the history of FCM, and this is now the fourth year of FCM. As I do every year (and hopefully every month), I want to thank all the people who help me put together FCM each month. You'll find their names at the end of this (and every) issue of FCM. Another important thing this month is 11.04. I'm sure, by the time you read this, many of you will have upgraded to Natty Narwhal. The last major thing of note, which you can read more about in the [Linux News](#) page, is that Canonical have announced the end of ShipIt. This means that people won't be able to request a free Ubuntu CD. I'm sure most of the world has broadband Internet, but I know many of our readers do not. I hope the removal of ShipIt doesn't stop people from getting Ubuntu.

Since it's our fourth birthday, I've added a fourth HowTo article about developing a circuit using the Arduino board. I'm hoping that, from this issue onwards, I can keep four HowTo's each month. **Daniel Holbach** has kindly offered to write a series of articles on developing for Ubuntu. The first article describes the development process of Ubuntu itself, and then moves on, in later parts, to how you can help develop Ubuntu with fixes and bug reports. It's very insightful, and it'll start next month. This month, however, we have more Python, the third piece on LibreOffice, and a piece on finding ebooks for free. If you're fed up with reinstalling Ubuntu, putting your desktop back to the way you like it, and reinstalling all your applications, then you might want to take a look at Art's review (and HowTo, all in one handy dandy article) of Remastersys. It lets you take a new Ubuntu install, and remaster it to become your own customized Live CD.

I hope you enjoy the issue, and I'll see you all next month.

All the best, and keep in touch.

Ronnie

ronnie@fullcirclemagazine.org



This magazine was created using :



Full Circle Podcast

Released every two weeks, each episode covers all the latest Ubuntu news, opinions, reviews, interviews and listener feedback. The Side-Pod is a new addition, it's an extra (irregular) short-form podcast which is intended to be a branch of the main podcast. It's somewhere to put all the general technology and non-Ubuntu stuff that doesn't fit in the main podcast.

Hosts:

Robin Catling
Ed Hewitt
Dave Wilkins

<http://fullcirclemagazine.org>



AUDIO MP3



AUDIO OGG



ShiIt comes to an end

Canonical has announced the end of the ShiIt program. As Gerry Carr explained on the Canonical blog:

It's with some regret that we are announcing the end of the ShiIt Programme and the CD distributor programme. When we started ShiIt in 2005 broadband was still a marketing promise even in the most connected parts of the most developed nations. We knew that this represented a significant stumbling block to the adoption of a new technology like Ubuntu. So we invested in making the CDs free and freely delivered to anywhere in the world [...] but for Ubuntu 11.04 you will no longer be able to go to our website and apply for a free CD.

We are going to make large numbers of CDs available to the Ubuntu Local Communities (LoCos) through a shiIt-lite program.

Source: <http://blog.canonical.com>

C64 Returns!

It's back... and better than ever! The new Commodore 64 is a modern functional PC as close to the original in design as humanly possible. It houses a modern mini-ITX PC motherboard featuring a Dual Core 525 Atom processor and the latest nVidia ION2 graphics chipset. It comes in the original taupe brown/beige color, with other colors to follow.

Commodore OS 1.0, along with emulation functionality and classic game package, will be mailed to purchasers when available. In the meantime, units come with the Ubuntu 10.04 LTS operating system on CD ready to install.

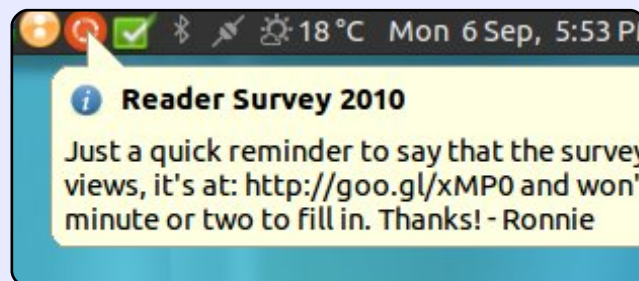
Source: commodoreusa.net



Full Circle Notifier

Our very own **Full Circle Notifier** is now at 1.0.2. FCN is a small application that sits in your system tray and will not only announce issue/podcast releases, and can be set to automatically download them for you too! Several people have created packages of FCN and translations are starting.

For more info, see the **FCN Google Group**:
<http://goo.gl/4Ob4>

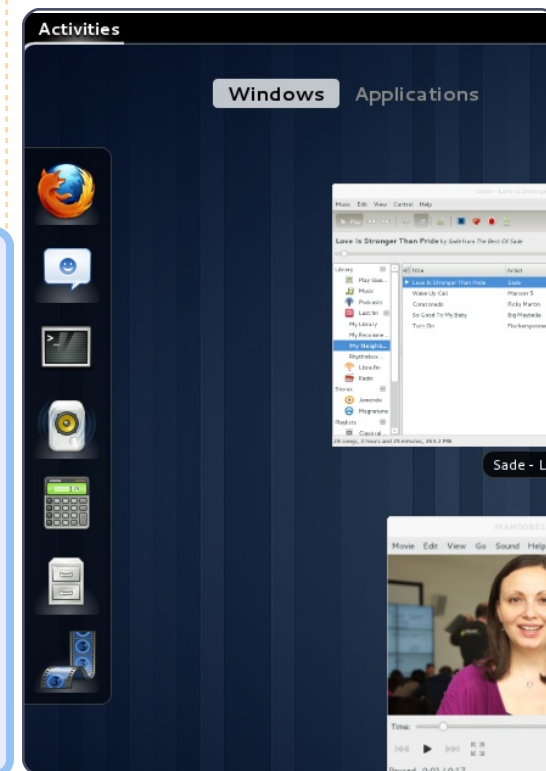


GNOME 3 Released!

GNOME 3 was released on 6th April 2011. If you want to try it, you can run a live version from a CD/DVD or USB stick.

The 'live version' comes in both OpenSUSE and Fedora versions. No Ubuntu download though as Ubuntu, from 11.04, will come with its own Unity interface.

Source: <http://gnome3.org/>





This month, I felt I would share with you something I just recently learned about. The topics I'll be covering apply only to those readers who either use iBus/SCIM and aren't happy with it, or who have it running and are happy with it - but whose Japanese/Chinese/etc. doesn't appear in a legible font in rxvt-unicode. Also, I'd like to take a moment to announce that next month I hope to do a question and answer session for C&C readers. If you have questions about Linux in general, the command-line, or me as an author, feel free to send your questions to lswest34@gmail.com before the 28th of April. I will be selecting a bunch of questions to answer next month. Requests for articles are also welcome.

As some of you probably know, I wrote an article on iBus in issue #43 of FCM. I hadn't used iBus since I was comfortable with SCIM. However, an update recently disabled SCIM, and so I tried iBus. What really got me was that I couldn't switch between hiragana

and katakana easily, so I decided to take a suggestion from a friend of mine and tried out uim. Surprisingly, uim doesn't block my dead keys in rxvt-unicode, and allows easy switching between hiragana and katakana. Below is how I configured it for use.

uim & uim-fep:

From the homepage (<http://code.google.com/p/uim/>):
"uim's goal is to provide simple, easily extensible and high code-quality input method development platform, and useful input method environment for users of desktop and embedded platforms. See what's uim? for further information."

First, you'll need to install it:

```
sudo apt-get install uim uim-gtk2.0 uim-qt uim-qt3 uim-fep uim-anthy
```

This should cover uim support for terminals, QT applications, and GTK applications using anthy. There are a number of other

packages offering applets, different dictionaries, and utilities, that may be of interest to some people.

Once you've installed it, running uim-toolbar-gtk-systray will give you a system tray icon. Right-click on it and choose preferences. Here, I would adjust the list of enabled input languages to only the ones you need, and adjust the global key bindings to your preferences. If you find that the system tray icon is practically invisible, it's because too much information is being displayed in the one "icon" width. To adjust this, open the preferences, and, under "Toolbar", uncheck everything, and set the enabled toolbar buttons per language that you use to just "Input Mode". This will reduce it to one icon - making it readable again. Also, in order to get it working, you'll need to add the following to /etc/profile (or .bashrc, or .zshrc):

```
export XMODIFIERS=@im=uim
```

```
export GTK_IM_MODULE="uim"
```

```
export QT_IM_MODULE="uim"
```

Once you've set these variables, you should run the following in a terminal:

```
gtk-query-immodules-2.0 > /etc/gtk-2.0/gtk.immodules
```

This will re-create the gtk.immodules file, which specifies to GTK programs which Input Method types are available.

Uim-fep is a Front-End Processor for terminal emulators. Basically, it allows you to type Japanese in a terminal emulator (rxvt-unicode in my case), without relying on uim-xim (which is a bit of a resource hog). In order to get it working, you'll need to add uim-fep to the end of your .bashrc, or your .zshrc, or whatever shell you're using. If you get a warning that uim-fep is already running, you can add "clear" (without the quotes) after it, so that it hides the message. Once it's running, you'll have a line at the end of your terminal that looks something like

this:

```
anthy-utf8[- ]
```

Using the global shortcut for uim will result in the icon at the end changing to the input method, and allows you to type Japanese in-line in the terminal.

Rxvt-unicode:

In case you have the problem that your Japanese is nearly unreadable in rxvt-unicode (this may apply to other terminal emulators as well, but I haven't tested it), then you can add the following to your .Xdefaults:

```
URxvt.preeditType:  
OnTheSpot,None
```

```
URxvt.imLocale: ja_JP.UTF-8
```

```
URxvt.font: xft:Anonymous  
Pro:size=11:antialias=true:autohint=false,xft:IPAGothic:size=11:antialias=true
```

```
URxvt.boldFont: xft:Anonymous  
Pro:size=11:weight=Bold:antialias=true:autohint=false,xft:IPAGothic:size=11:weight=Bold:antialias=true
```

This, basically, tells urxvt to

expect Japanese input from uim. The fonts are actually a list of two, as you can see. Anonymous Pro is the terminal font I use for everything, but if rxvt-unicode can't find the symbols for something in that font, it will move on to the next one in the list (or a fallback font if there is no such symbol in any font listed). This allows you to have support for multiple languages without compromising the readability of Latin symbols. Also, you may see some people using urxvt.* instead of URxvt.* - which can be problematic if you set the name of your terminal from a shortcut (i.e. urxvt -name ncmpcpp -e ncmpcpp). The first section of these preferences tells the system that the WM_CLASS of the program is that we want to affect, and the lowercase "urxvt" is the first of the list, which is set using the -name argument. If, instead, you use "URxvt", then it will not change depending on the -name switch. To see what I mean, enter the following command into a terminal, and click on rxvt-unicode.

```
xprop | grep "^WM_CLASS"
```

Which gives you something like this:

```
WM_CLASS(STRING) = "urxvt",  
"URxvt"
```

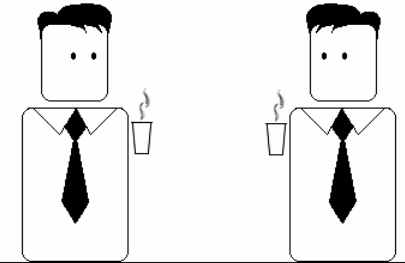
Now you should have a fully functional uim setup, and shouldn't have had to compromise any functionality in your terminal either. If you have any suggestions, or requests for articles, feel free to email me at lswest34@gmail.com. Also, don't forget your questions! I will need the questions sent in before the 28th of April!



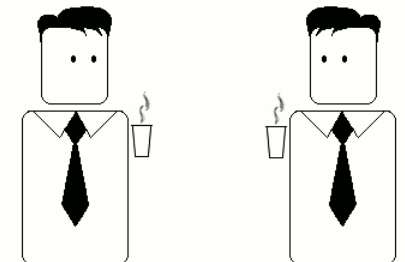
Lucas has learned all he knows from repeatedly breaking his system, then having no other option but to discover how to fix it. You can email Lucas at: lswest34@gmail.com.

A New Job

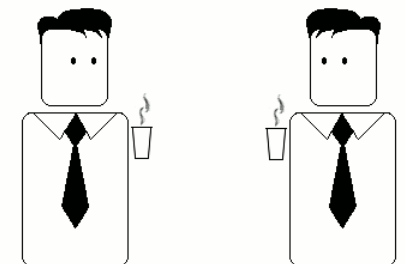
I have serious doubts about my capabilities, you know. I almost never have any good ideas.



Does this mean I'll never make it as an engineer?



No, of course not. It just means you'll fit in well.



by Richard Redei



Correction

Last month, in part 21, you were told to save what you have as "PlaylistMaker.glade", but, in the code, it was referred to as: "playlistmaker.glade". I'm sure you noticed that one has capitals and the other does not. The code will run only if you use both the call and file name with, or both without, the capitals.

To start off on the right foot, you need to have the playlistmaker.glade and playlistmaker.py from last month. If you don't, jump over to the last issue and get the goodies. Before we get to the code, let's take a look at what a playlist file is. There are multiple versions of play lists, and they all have different extensions. The one we will be creating will be a *.m3u type playlist. In its simplest form, it's just a text file that starts with "#EXTM3U", and then has an entry for each song file you want to play -

including the full path. There's also an extension that can be added before each entry that includes the length of the song, the album name the song comes from, the track number, and the song name. We'll bypass the extension for now and just concentrate on the basic version.

Here is an example of a M3U playlist file..

```
.
#EXTM3U
Adult Contemporary/Chris
Rea/Collection/02 - On The
Beach.mp3
Adult Contemporary/Chris
Rea/Collection/07 - Fool (If
You Think It's Over).mp3
Adult Contemporary/Chris
Rea/Collection/11 - Looking
For The Summer.mp3
```

All path names are relative to the location of the playlist file.

OK...now let's get to coding. Shown right is the opening of the source code from last month.

Now, we need to create an event handler routine for each of our events that we have set up. Notice that on_MainWindow_destroy and

```
#!/usr/bin/env python
import sys
from mutagen.mp3 import MP3
try:
    import pygtk
    pygtk.require("2.0")
except:
    pass
try:
    import gtk
    import gtk.glade
except:
    sys.exit(1)
```

next the class definition

```
class PlayListCreator:
    def __init__(self):
        self.gladefile = "playlistmaker.glade"
        self.wTree = gtk.glade.XML(self.gladefile, "MainWindow")
```

and the main routine

```
if __name__ == "__main__":
    plc = PlayListCreator()
    gtk.main()
```

Next, we have our dictionary which should go after the __init__ routine.

```
def SetEventDictionary(self):
    dict = {"on_MainWindow_destroy": gtk.main_quit,
            "on_tbtnQuit_clicked": gtk.main_quit,
            "on_tbtnAdd_clicked": self.on_tbtnAdd_clicked,
            "on_tbtnDelete_clicked": self.on_tbtnDelete_clicked,
            "on_tbtnClearAll_clicked": self.on_tbtnClearAll_clicked,
            "on_tbtnMoveToTop_clicked": self.on_tbtnMoveToTop_clicked,
            "on_tbtnMoveUp_clicked": self.on_tbtnMoveUp_clicked,
            "on_tbtnMoveDown_clicked": self.on_tbtnMoveDown_clicked,
            "on_tbtnMoveToBottom_clicked": self.on_tbtnMoveToBottom_clicked,
            "on_tbtnAbout_clicked": self.on_tbtnAbout_clicked,
            "on_btnGetFolder_clicked": self.on_btnGetFolder_clicked,
            "on_btnSavePlaylist_clicked": self.on_btnSavePlaylist_clicked}
    self.wTree.signal_autoconnect(dict)
```

on_tbtnQuit_clicked are already done for us, so we need to have only 10 more (shown top right). Just make stubs for now.

We'll modify these stubbed routines in a few minutes. For now, this should get us up and running with an application, and we can test things as we go. But, we need to add one more line to the `__init__` routine before we can run the app. After the `self.wTree` line, add...

```
self.SetEventDictionary()
```

Now, you can run the application, see the window, and click the Quit toolbar button to exit the application properly. Save the code as "playlistmaker-1a.py" and give it a try. Remember to save it in the same folder as the glade file we created last time, or copy the glade file into the folder you saved this code in.

We also need to define a few variables for future use. Add these after the `SetEventDictionary` call in the `__init__` function.

```
self.CurrentPath = ""
self.CurrentRow = 0
self.RowCount = 0
```

Now, we will create a function that allows us to display a popup dialog box whenever we need to give some information to our user. There is a built-in set of routines that we will use, but we'll make a routine of our own to make it easier for us. It is the `gtk.MessageDialog` routine, and the syntax is as follows...

```
gtk.MessageDialog(parent, flags, MessageType, Buttons, message)
```

Some discussion is needed before we go too much further. The message type can be one of the following...

```
GTK_MESSAGE_INFO - Informational message
GTK_MESSAGE_WARNING - Nonfatal warning message
GTK_MESSAGE_QUESTION - Question requiring a choice
GTK_MESSAGE_ERROR - Fatal error message
```

And the button types are...

```
GTK_BUTTONS_NONE - no buttons at all
GTK_BUTTONS_OK - an OK button
GTK_BUTTONS_CLOSE - a Close button
GTK_BUTTONS_CANCEL - a
```

```
def on_tbtnAdd_clicked(self,widget):
    pass
def on_tbtnDelete_clicked(self,widget):
    pass
def on_tbtnClearAll_clicked(self,widget):
    pass
def on_tbtnMoveToTop_clicked(self,widget):
    pass
def on_tbtnMoveUp_clicked(self,widget):
    pass
def on_tbtnMoveDown_clicked(self,widget):
    pass
def on_tbtnMoveToBottom_clicked(self,widget):
    pass
def on_tbtnAbout_clicked(self,widget):
    pass
def on_btnGetFolder_clicked(self,widget):
    pass
def on_btnSavePlaylist_clicked(self,widget):
    pass
```

```
Cancel button
GTK_BUTTONS_YES_NO - Yes and No buttons
GTK_BUTTONS_OK_CANCEL - OK and Cancel Buttons
```

Normally, you would use the following code, or similar, to create the dialog, display it, wait for a response, and then destroy it.

```
dlg =
gtk.MessageDialog(None,0,gtk.
MESSAGE_INFO,gtk.BUTTONS_OK,"
This is a test message...")
response = dlg.run()
dlg.destroy()
```

However, if you want to display a message box to the user more

than once or twice, that's a LOT of typing. The general rule of thumb is that if you write a series of lines-of-code more than once or twice, it's usually better to create a function and then call that. Think of it this way: If we want to display a message dialog to the user, say ten times in your application, that's 10 X 3 (or 30) lines of code. By making a function to do this for us (using the example I just presented), we would have 10 + 3 (or 13) lines of code to write. The more we call a dialog, the less code we actually have to type, and the more readable our code is. Our

function (top right) will allow us to call any of the four message dialog types with just one routine using different parameters.

This is a very simple function that we would then call like this...

```
self.MessageBox("info", "The button QUIT was clicked")
```

Notice that if we choose to use the MESSAGE_QUESTION type of dialog, there are two possible responses that will be returned by the message dialog - a "Yes" or a "No". Whichever button the user clicks, we will receive the information back in our code. To use the question dialog, the call would be something like this...

```
response =  
self.MessageBox("question", "Are you sure you want to do this now?")
```

```
if response ==  
gtk.RESPONSE_YES:
```

```
    print "Yes was clicked"
```

```
elif response ==  
gtk.RESPONSE_NO:
```

```
    print "NO was clicked"
```

You can see how you can check the value of the button returned. So

```
def MessageBox(self, level, text):  
    if level == "info":  
        dlg = gtk.MessageDialog(None, 0, gtk.MESSAGE_INFO, gtk.BUTTONS_OK, text)  
    elif level == "warning":  
        dlg = gtk.MessageDialog(None, 0, gtk.MESSAGE_WARNING, gtk.BUTTONS_OK, text)  
    elif level == "error":  
        dlg = gtk.MessageDialog(None, 0, gtk.MESSAGE_ERROR, gtk.BUTTONS_OK, text)  
    elif level == "question":  
        dlg = gtk.MessageDialog(None, 0, gtk.MESSAGE_QUESTION, gtk.BUTTONS_YES_NO, text)  
    if level == "question":  
        resp = dlg.run()  
        dlg.destroy()  
        return resp  
    else:  
        resp = dlg.run()  
        dlg.destroy()
```

now, replace the "pass" call in each of our event handler routines with something like that shown below right.

We won't keep it like this, but this gives you a visual indication that the buttons work the way we want. Save the code now as "playlistmaker-1b.py", and test your program. Now we are going to create a function to set our widget references. This routine is going to be called only once, but it will make our

```
def on_tbtnAdd_clicked(self, widget):  
    self.MessageBox("info", "Button Add was clicked...")  
def on_tbtnDelete_clicked(self, widget):  
    self.MessageBox("info", "Button Delete was clicked...")  
def on_tbtnClearAll_clicked(self, widget):  
    self.MessageBox("info", "Button ClearAll was clicked...")  
def on_tbtnMoveToTop_clicked(self, widget):  
    self.MessageBox("info", "Button MoveToTop was clicked...")  
def on_tbtnMoveUp_clicked(self, widget):  
    self.MessageBox("info", "Button MoveUp was clicked...")  
def on_tbtnMoveDown_clicked(self, widget):  
    self.MessageBox("info", "Button MoveDown was clicked...")  
def on_tbtnMoveToBottom_clicked(self, widget):  
    self.MessageBox("info", "Button MoveToBottom was clicked...")  
def on_tbtnAbout_clicked(self, widget):  
    self.MessageBox("info", "Button About was clicked...")  
def on_btnGetFolder_clicked(self, widget):  
    self.MessageBox("info", "Button GetFolder was clicked...")  
def on_btnSavePlaylist_clicked(self, widget):  
    self.MessageBox("info", "Button SavePlaylist was clicked...")
```

code much more manageable and readable. Basically, we want to create local variables that

reference the widgets in our glade window - so we can make calls to them whenever (if ever) we need

to. Put this function (above right) below the SetEventDictionary function.

Please notice that there is one thing that isn't referenced in our routine. That would be the treeview widget. We'll make that reference when we set up the treeview itself. Also of note is the last line of our routine. In order to use the status bar, we need to refer to it by its context id. We'll be using this later on.

Next, let's set up the function that displays the "about" dialog when we click the About toolbar button. Again, there is a built-in routine to do this provided by the GTK library. Put this after the MessageBox function. Here's the code, below right.

Save your code and then give it a try. You should see a pop-up box, centered in our application, that displays everything we have set. There are more attributes that you can set for the about box (which can be found at <http://www.pygtk.org/docs/pygtk/class-gtkaboutdialog.html>), but these are what I would consider a minimum set.

```
def SetWidgetReferences(self):
    self.txtFilename = self.wTree.get_widget("txtFilename")
    self.txtPath = self.wTree.get_widget("txtPath")
    self.tbtnAdd = self.wTree.get_widget("tbtnAdd")
    self.tbtnDelete = self.wTree.get_widget("tbtnDelete")
    self.tbtnClearAll = self.wTree.get_widget("tbtnClearAll")
    self.tbtnQuit = self.wTree.get_widget("tbtnQuit")
    self.tbtnAbout = self.wTree.get_widget("tbtnAbout")
    self.tbtnMoveToTop = self.wTree.get_widget("tbtnMoveToTop")
    self.tbtnMoveUp = self.wTree.get_widget("tbtnMoveUp")
    self.tbtnMoveDown = self.wTree.get_widget("tbtnMoveDown")
    self.tbtnMoveToBottom = self.wTree.get_widget("tbtnMoveToBottom")
    self.btnGetFolder = self.wTree.get_widget("btnGetFolder")
    self.btnSavePlaylist = self.wTree.get_widget("btnSavePlaylist")
    self.sbar = self.wTree.get_widget("statusbar1")
    self.context_id = self.sbar.get_context_id("Statusbar")
```

and then add a call to it right after the self.SetEventDictionary() call in the __init__ function.

```
self.SetWidgetReferences()
```

Before we go on, we need to discuss exactly what will happen from here. The general idea is that the user will click on the "Add" toolbar button, we'll pop up a file dialog box to allow them to add files to the playlist, and then display the file information into our treeview widget. From there, they can add more files, delete single file entries, delete all file entries, move a file entry up, down, or to the top or down to the bottom of the

```
def ShowAbout(self):
    about = gtk.AboutDialog()
    about.set_program_name("Playlist Maker")
    about.set_version("1.0")
    about.set_copyright("(c) 2011 by Greg Walters")
    about.set_comments("Written for Full Circle Magazine")
    about.set_website("http://thedesignatedgeek.com")
    about.run()
    about.destroy()
```

Now, comment out (or simply remove) the messagebox call in the on_tbtnAbout_clicked routine, and replace it with a call to the ShowAbout function. Make it look like this.

```
def on_tbtnAbout_clicked(self,widget):
    #self.MessageBox("info","Button About was clicked...")
    self.ShowAbout()
```


treeview. Eventually, they'll set the path that the file will be saved to, provide a filename with a "m3u" extension, and click the save file button. While this seems simple enough, there's a lot that happens behind the scenes. The magic all happens in the treeview widget, so let's discuss that. This will get pretty deep, so you might want to read carefully, since an understanding of this will keep you from making mistakes later on.

A treeview can be something as simple as a columnar list of data like a spreadsheet or database representation, or it could be more complex like a file-folder listing with parents and children, where the folder would be the parent and the files in that folder would be the children, or something even more complex. For this project, we'll use the first example, a columnar list. In the list, there will be three columns. One is for the name of the music file, one is for the extension of the file (mp3, ogg, wav, etc) and the final column is for the path. Combining this into a string (path, filename, extension) gives us the entry into the playlist we will be writing. You could, of course, add more columns as you wish, but for

now, we'll deal with just three.

A treeview is simply a visual storage container that holds and displays a model. The model is the actual "device" that holds and manipulates our data. There are two different pre-defined models that are used with a treeview, but you can certainly create your own. That having been said, for 98% of your work, one of the two pre-defined models will do what you need. The two types are `GTKListStore` and `GTKTreeStore`. As their names suggest, the `ListStore` model is usually used for lists, the `TreeStore` is used for Trees. For our application, we will be using a `GTKListStore`. The basic steps are:

- Create a reference to the `TreeView` widget.
- Add the columns.
- Set the type of renderer to use.
- Create the `ListStore`.
- Set the model attribute in the `Treeview` to our model.
- Fill in the data.

The third step is to set up the type

```
def SetupTreeview(self):
    self.cFName = 0
    self.cFType = 1
    self.cFPath = 2
    self.sFName = "Filename"
    self.sFType = "Type"
    self.sFPath = "Folder"
    self.treeview = self.wTree.get_widget("treeview1")
    self.AddPlaylistColumn(self.sFName,self.cFName)
    self.AddPlaylistColumn(self.sFType,self.cFType)
    self.AddPlaylistColumn(self.sFPath,self.cFPath)
    self.playList = gtk.ListStore(str,str,str)
    self.treeview.set_model(self.playList)
    self.treeview.set_grid_lines(gtk.TREE_VIEW_GRID_LINES_BOTH)
```

of renderer the column will use to display the data. This is simply a routine that is used to draw the data into the tree model. There are many different cell renderers that come with GTK, but most of the ones that you would normally use include `GtkCellRenderText` and `GtkCellRendererToggle`.

So, let's create a function (shown above) that sets up our `TreeView` widget. We'll call it `SetupTreeview`. First we'll define some variables for our columns, set the variable reference of the `TreeView` itself, add the columns, set up the `ListStore`, and set the model. Here's the code for the function. Put it after the `SetWidgetReferences` function.

The variables `cFName`, `cFType` and `cFPath` define the column numbers. The variables `sFName`, `sFType` and `sFPath` will hold the column names in our displayed view. The seventh line sets the variable reference of the treeview widget as named in our glade file.

Next we call a routine (next page, top right), which we'll create in just a moment, for each column we want. Then we define our `GTKListStore` with three text fields, and finally set the model attribute of our `TreeView` widget to our `GTKListStore`. Let's create the `AddPlaylistColumn` function next. Put it after the `SetupTreeview` function.

Each column is created with this

function. We pass in the title of the column (what's displayed on the top line of each column) and a columnID. In this case, the variables we set up earlier (sFName and cFname) will be passed here. We then create a column in our TreeView widget giving the title, what kind of cell renderer it will be using, and, finally, the id of the column. We then set the column to be resizable, set the sort id, and finally append the column into the TreeView.

Add these two functions to your code. I choose to put them right after the SetWidgetReferences function, but you can put it anywhere within the PlaylistCreator class. Add the following line after the call to SetWidgetReferences() in the __init__ function to call the function.

```
self.SetupTreeview()
```

Save and run your program, and you will see that we now have three columns with headers in our TreeView widget.

There are so many things left to do. We have to have a way to get

the music filenames from the user and put them into the TreeView as rows of data. We have to create our Delete, ClearAll, movement functions, save routine, and file path routines, plus a few "pretty" things that will make our application look more professional. Let's start with the Add routine. After all, that's the first button on our toolbar. When the user clicks the Add button, we want to pop up a "standard" open-file dialog that allows for multiple selections. Once the user has made their selection, we then want to take this data and add it into the treeview, as I stated above. So the first logical thing to do is work on the File Dialog. Again, GTK provides us a way to call a "standard" file dialog in code. We could hard code this as just lines in the on_tbtnAdd_clicked event handler, but let's make a separate class to handle this. While we are at it, we can make this class handle not only a file OPEN dialog, but a folder SELECT dialog as well. As before with the MessageBox function, you

```
def AddPlaylistColumn(self,title,columnId):
    column = gtk.TreeViewColumn(title,gtk.CellRendererText(),text=columnId)
    column.set_resizable(True)
    column.set_sort_column_id(columnId)
    self.treeview.append_column(column)
```

can pull this into a snippet file that has all kinds of reusable routines for later use.

We'll start by defining a new class called FileDialog which will have only one function called ShowDialog. That function will take two parameters, one called 'which' (a '0' or a '1'), that designates whether we are creating an open-file or select-folder dialog, and the other is the path that should be used for the default view of the dialog called CurrentPath. Create this class just before our main code at the bottom of the source file.

```
class FileDialog:
    def ShowDialog(self,which,CurrentPath):
```

The first part of our code should

be an IF statement

```
if which == 0: # file
chooser
...
else:         # folder chooser
...
```

Before going any further, let's explore how the file/folder dialog is actually called and used. The syntax of the dialog is as follows

```
gtk.FileChooserDialog(title,p
arent,action,buttons,backend)
```

and returns a dialog object. Our first line (under if which == 0) will be the line shown below.

As you can see, the title is "Select files to add...", the parent is set to None. We are requesting a File Open type dialog (action), and we want a Cancel and an Open button, both using "stock" type icons. We

```
dialog = gtk.FileChooserDialog("Select files to add...",None,
gtk.FILE_CHOOSER_ACTION_OPEN,
(gtk.STOCK_CANCEL, gtk.RESPONSE_CANCEL,
gtk.STOCK_OPEN, gtk.RESPONSE_OK))
```

are also setting the return codes of `gtk.RESPONSE_CANCEL` and `gtk.RESPONSE_OK` for when the user makes their selections. The call for our Folder Chooser under the Else clause is similar.

Basically, the only thing that changed between the two definitions are the title (shown above right) and the action type. So our code for the class should now be the code shown middle right.

These set the default response to be the OK button, and then to turn on the multiple select feature so the user can select (you guessed it) multiple files to add. If we didn't set this, the dialog would only allow one file to be selected at a time, since `set_select_multiple` is set to False by default. Our next lines are setting the current path, and then displaying the dialog itself. Before we type in the code, let me explain why we want to deal with the current path. Every time you pop up a file dialog box, and you DON'T set a path, the default is to the folder where our application resides. So, let's say that the music files that the user would be looking for are in `/media/music_files/`, and are then

broken down by genre, and further by artist, and further by album. Let's further assume that the user has installed our application in `/home/user2/playlistmaker`. Each time we pop up the dialog, the starting folder would be `/home/user2/playlistmaker`. Quickly, the user would become frustrated by this, wanting the last folder he was in to be the starting folder next time. Make sense? OK. So, bottom right are our next lines of code.

Here we check the responses sent back. If the user clicked the 'Open' button which sends back a `gtk.RESPONSE_OK`, we get the name or names of the files the user selected, set the current path to the folder we are in, destroy the dialog, and then return the data back to the calling routine. If, on the other hand, the user clicked on the 'Cancel' button, we simply destroy the dialog. I put the print

```
dialog = gtk.FileChooserDialog("Select Save Folder..",None,
                                gtk.FILE_CHOOSER_ACTION_SELECT_FOLDER,
                                (gtk.STOCK_CANCEL, gtk.RESPONSE_CANCEL,
                                 gtk.STOCK_OPEN, gtk.RESPONSE_OK))
```

```
class FileDialog:
    def ShowDialog(self,which,CurrentPath):
        if which == 0: #file chooser
            #gtk.FileChooserDialog(title,parent,action,buttons,backend)
            dialog = gtk.FileChooserDialog("Select files to add...",None,
                                            gtk.FILE_CHOOSER_ACTION_OPEN,
                                            (gtk.STOCK_CANCEL, gtk.RESPONSE_CANCEL,
                                             gtk.STOCK_OPEN, gtk.RESPONSE_OK))

        else:          #folder chooser
            dialog = gtk.FileChooserDialog("Select Save Folder..",None,
                                            gtk.FILE_CHOOSER_ACTION_SELECT_FOLDER,
                                            (gtk.STOCK_CANCEL, gtk.RESPONSE_CANCEL,
                                             gtk.STOCK_OPEN, gtk.RESPONSE_OK))
```

The next two lines will be (outside of the IF/ELSE statement)...

```
dialog.set_default_response(gtk.RESPONSE_OK)
dialog.set_select_multiple(True)
```

```
if CurrentPath != "":
    dialog.set_current_folder(CurrentPath)
response = dialog.run()
```

Next, we need to handle the response from the dialog.

```
if response == gtk.RESPONSE_OK:
    fileselection = dialog.get_filenames()
    CurrentPath = dialog.get_current_folder()
    dialog.destroy()
    return (fileselection,CurrentPath)
elif response == gtk.RESPONSE_CANCEL:
    print 'Closed, no files selected'
    dialog.destroy()
```


statement in there just to show you that the button press worked. You can leave it or take it out. Notice that when we return from the Open button part of the routine, we are returning two sets of values. 'fileselection' is a list of the files selected by the user, as well as the CurrentPath.

In order to get the routine to do something, add the following line under the on_tbtnAdd_click routine...

```
fd = FileDialog()

selectedfiles, self.CurrentPath = fd.ShowDialog(0, self.CurrentPath)
```

Here we retrieve the two return values that are sent from our return call. For now, add the following code to see what the information returned will look like.

```
for f in selectedfiles:

    print "User selected %s" % f

print "Current path is %s" % self.CurrentPath
```

When you run the program, click on the 'Add' button. You'll see the file dialog. Now move to

somewhere where you have some files and select them. You can hold down the [ctrl] key and click on multiple files to select them individually, or the [shift] key to select multiple contiguous files. Click on the 'Open' button, and look at the response in your terminal window. Please note that if you click on the 'Cancel' button right now, you'll get an error message. That's because the above code assumes that there are no files selected. Don't worry about that right now - we'll handle that in a little bit. I just wanted to let you see what comes back if the 'Open' button is pressed. One thing we should do is add a filter to our file-open dialog. Since we expect the user to normally select music files, we should (1) give the option to display only music files, and (2) give the option to show all files just-in-case. We do this by using the filefilter attributes of the dialog. Here's the code for that which should go in the which == 0 section right after the dialog set line.

```
filter = gtk.FileFilter()
filter.set_name("Music Files")
filter.add_pattern("*.mp3")
filter.add_pattern("*.ogg")
filter.add_pattern("*.wav")
```

```
dialog.add_filter(filter)
filter = gtk.FileFilter()
filter.set_name("All files")
filter.add_pattern("*")
dialog.add_filter(filter)
```

We are setting up two "groups", one for music files (filter.set_name("Music Files")), and the other for all files. We use a pattern to define the types of files we want. I have defined three patterns, but you can add or delete any that you wish. I put the music filter first, since that's what we will assume the user is going to be mainly concerned with. So the steps are...

- Define a filter variable.
- Set the name.
- Add a pattern.
- Add the filter to the dialog.

You can have as many or as few filters as you wish. Also notice that once you have added the filter to the dialog, you can re-use the variable for the filter.

Back in the on_tbtnAdd_clicked routine, comment out the last lines we added and replace them with this one line.

```
self.AddFilesToTreeview(selectedfiles)
```

so our routine now looks like the code shown on the next page.

So, when we get the response back from file dialog, we will send the list containing the selected files to this routine. Once here, we set up a counter variable (how many files we are adding), then parse the list. Remember that each entry contains the fully qualified filename with path and extension. We'll want to split the filename into path, filename, and extension. First we get the very last 'period' from the filename and assume that is the beginning of the extension and assign its position in the string to extStart. Next we find the very last '/' in the filename to determine the beginning of the filename. Then we break up the string into extension, filename and file path. We then stuff these values into a list named 'data' and append this into our playlist ListStore. We increment the counter since we have done all the work. Finally we increment the variable RowCount which holds the total number of rows in our ListStore, and then we print a message to the status bar.

Now you can run the application

```
def on_tbtnAdd_clicked(self,widget):
    fd = FileDialog()
    selectedfiles,self.CurrentPath =
fd.ShowDialog(0,self.CurrentPath)
    self.AddFilesToTreeview(selectedfiles)
```

We now have to create the function that we just put the call to. Put this function after the on_btnSavePlaylist_clicked routine.

```
def AddFilesToTreeview(self,FileList):
    counter = 0
    for f in FileList:
        extStart = f.rfind(".")
        fnameStart = f.rfind("/")
        extension = f[extStart+1:]
        fname = f[fnameStart+1:extStart]
        fpath = f[:fnameStart]
        data = [fname,extension,fpath]
        self.playList.append(data)
        counter += 1
    self.RowCount += counter
    self.sbar.push(self.context_id,"%s files added
for a total of %d" % (counter,self.RowCount))
```

and see the data in the TreeView.

As always, the full code can be found at

<http://pastebin.com/JtrhuE71>.

Next time, we'll finalize our application, filling in the missing routines, etc.



Greg Walters is owner of RainyDay Solutions, LLC, a consulting company in Aurora, Colorado, and has been programming since 1972. He enjoys cooking, hiking, music, and spending time with his family.

EXTRA! EXTRA! READ ALL ABOUT IT!



THE PERFECT SERVER SPECIAL EDITION

This is a special edition of Full Circle that is a direct reprint of the Perfect Server articles that were first published in FCM#31-#34.

<http://fullcirclemagazine.org/special-edition-1-the-perfect-server/>

Full Circle Special Editions Released On Unsuspecting World*



PYTHON SPECIAL EDITION #01

This is a reprint of Beginning Python Parts 01 – 08 by Greg Walters.

<http://fullcirclemagazine.org/python-special-edition-1/>

* Neither Full Circle magazine, nor its makers, apologize for any hysteria caused in the release of its publications.



HOW-TO

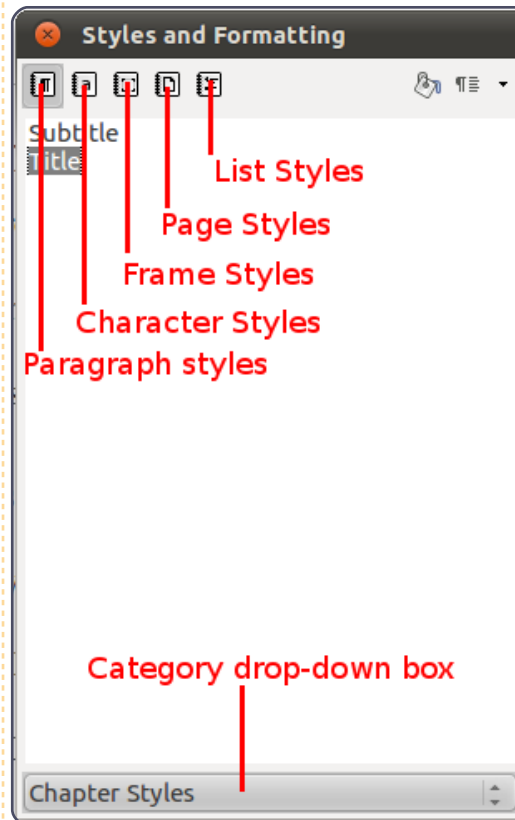
Written by Elmer Perry

Libre Office - Part 3

In my last article, I wrote about changing the layout of paragraphs to format your document. While this approach is okay on short documents, it creates a lot of work should you decide to change something in a large document. This is where the use of styles will make things easier.

LibreOffice writer has five different style types: paragraph, character, frame, page, and list. You can access all the styles by clicking on the Styles and Formatting button on the formatting toolbar. This will pop up the Styles and Formatting window. You can dock the Styles and Formatting window on the left by holding down the Ctrl key and double-clicking the empty space in the Styles and Formatting window toolbar.

The styles toolbar (right) has seven icons. The first five give you access to the different style types. In order from the left, they are paragraph, character, frame, page, and list. We will concentrate on



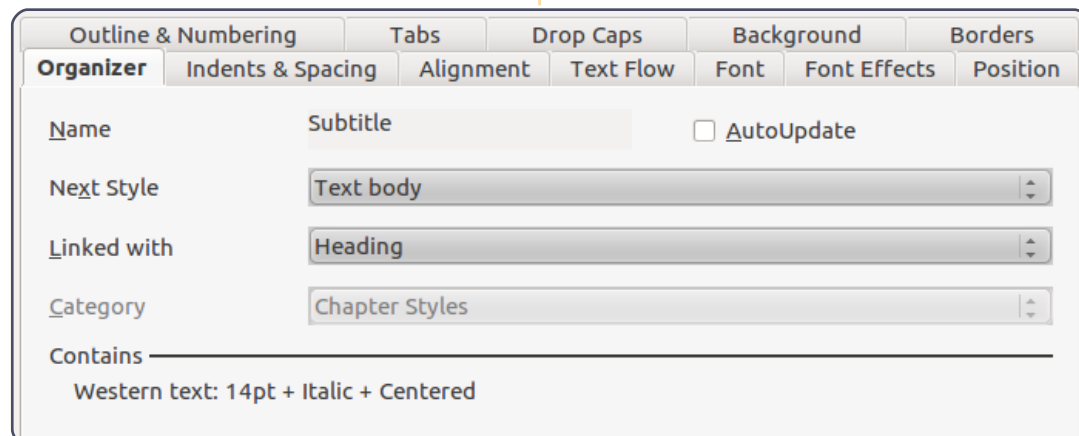
paragraph and character styles in this article.

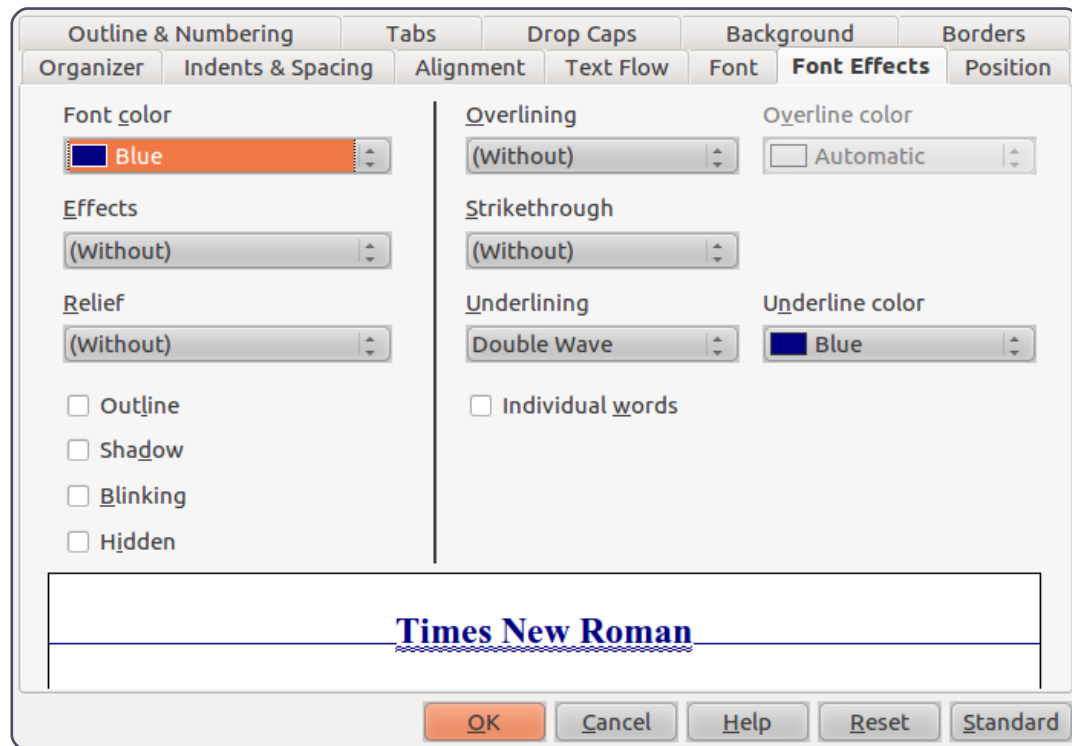
Open a new text document and type in a title. Open the Styles and Formatting window. The paragraph icon should be selected by default. At the bottom of the Styles and Formatting Window is a drop-down box. Click on the box and select Chapter Styles. Double-

click "title". Your title will center, enlarge, and become bold. Now, let's change the default styling for titles. In the Styles and Formatting window, right-click on the "title" style and select modify. The dialog that pops up looks a lot like the dialog from the last article, but there are a few new tabs that are not in the standard paragraph dialog. The first is the organizer. The organizer (below) shows you the name for the style, the next style to use, and the linked style. You will see that the next style is "subtitle", but we don't want to use a subtitle, so we will change this to the "text body" style. This makes it so that when we hit Enter to start a new paragraph the next paragraph will use the "text body"

style. The "title" style is linked with the "Heading" style. When styles are linked, any changes to the parent style affects the styles linked to it. As an example, if you change the text in the "Heading" to blue, all the styles linked to it will have blue text as well.

Now, let's format our title differently from the default. Click on the Font Effects tab. The Font Effects (next page, top left) allow you to change the look of the font, including color, strike-through, underline, shadow, and relief. The dialog shows you how the effects make your text look. Change the color to blue, underlining to Double Wave, and underline color to blue. Click OK.





Press Enter to start a new paragraph. Notice the style changed to "text body" just like we set up in the organizer tab. Now, type in three paragraphs of text to use for our example document.

Next, we will modify the "text body" style and create two new ones based on the "text body" style. Back in the Styles and Formatting window, click the drop-down box and select the Text Styles category. Right-click on "text body" and select modify. On the Indents & Spacing tab, change

the line spacing to 1.5 lines, and the First Line to 0.50. Click OK. Notice that our changes affected all three paragraphs.

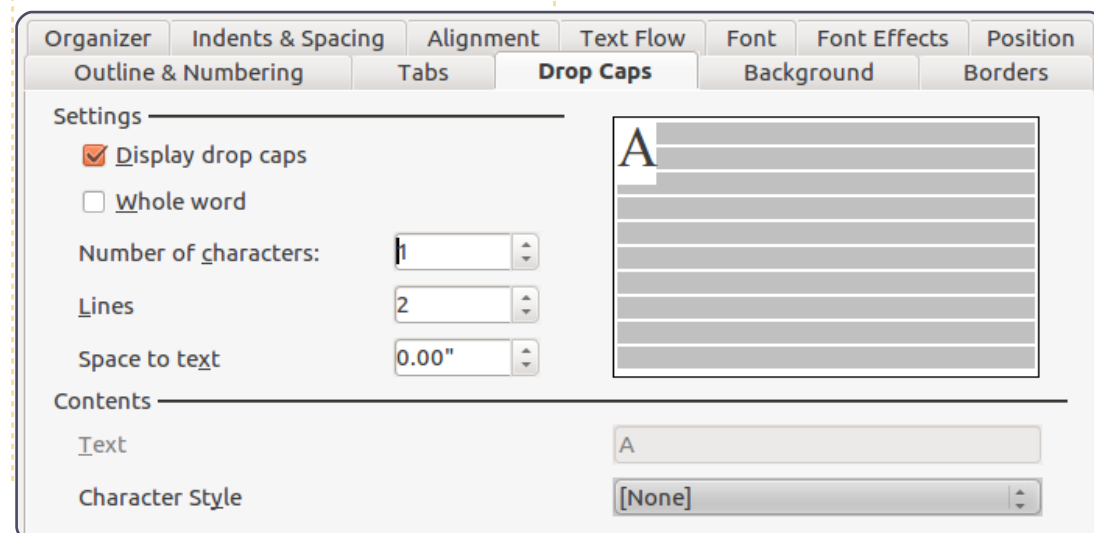
Now, let's create a paragraph for long quotes. Move the cursor to anywhere in the second paragraph. In the styles window, right-click text body and select New. On the organizer tab, give it the name of "Blockquote." Change the next style to "text body", as we rarely have two long quotes in a row. You will notice that because we created the new style by right-clicking on text body, it is

automatically linked to "text body". To create a new style not linked to another, change the Linked with to "None".

Now, let's change the formatting of our new style. On the Indents & Spacing tab, change the "before text" and "after text" to 0.50. Change the First Line back to 0.00. On the Font tab, change the typeface to italic. Click OK, and you will notice a new paragraph named "Blockquote" has been added to your list. Again, move your cursor to anywhere in the second paragraph and double-click "Blockquote." Now, you will see the first line indent has been taken away, the paragraph is indented on both sides, and the text is italicized.

Now, we want to change the first paragraph, giving it some drop caps. Since we want the first paragraph of each chapter to look this way, we will create another style. Again, right-click on "text body" and select New. Name the new style "First Paragraph", and change the next style to "Text Body." On the Indents & Spacing tab, change the First Line back to 0.00. On the Drop Caps tab (shown below), check "Display drop caps", set "Number of characters" to 1 and set "Lines" to 2. Click OK. Again, no changes are seen yet. Move your cursor into the first paragraph and double-click your new style.

We need this new paragraph style to follow every new chapter



title. Modify the "title" style so the next style is "First Paragraph."

Character styles affect only selected text rather than entire paragraphs. In the third paragraph, select some of the text. Click on the character style icon in the styles window, and double-click "Emphasis." This will italicize the text you have selected. You can modify the character styles much

in the same way you do the paragraph styles.

The key advantage to styles is making the formatting of like text the same throughout a document. In the next article, we will talk about adding frames to your document.



Elmer Perry is a children's minister in Asheville, North Carolina whose hobbies include web design, programming, and writing.

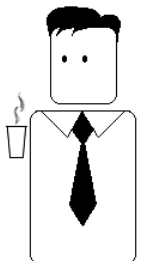
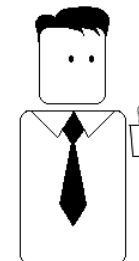
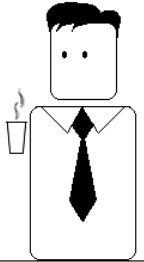
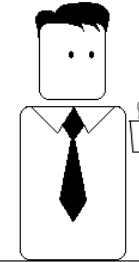
Paragraph Styles Example

He heard quiet steps behind him. That didn't bode well. Who could be following him this late at night and in this deadbeat part of town? And at this particular moment, just after he pulled off the big time and was making off with the greenbacks. Was there another crook who'd had the same idea, and was now watching him and waiting for a chance to grab the fruit of his labor? Or did the steps behind him mean that one of many law officers in town was on to him and just waiting to pounce and snap those cuffs on his wrists? He nervously looked all around. Suddenly he saw the alley.

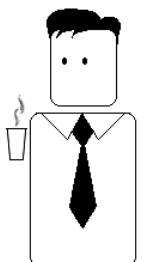
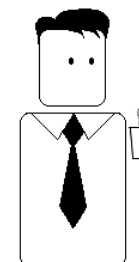
Like lightning he darted off to the left and disappeared between the two warehouses almost falling over the trash can lying in the middle of the sidewalk. He tried to nervously tap his way along in the inky darkness and suddenly stiffened: it was a dead-end, he would have to go back the way he had come. The steps got louder and louder, he saw the black outline of a figure coming around the corner. Is this the end of the line? he thought pressing himself back against the wall trying to make himself invisible in the

Customer Service

'How can I help you?' is a dangerous question to ask.



Most people react by telling the story of their lives.



by Richard Redei



When you are looking for E-books, there are a number of factors you should take into account: reader device, formats, DRM, and price are some that I intend to cover in this article.

Readers

The first decision you need to make is what device you intend to read your E-books on. The three large online booksellers in the United States - Amazon (Kindle), Barnes & Noble (Nook), and Borders (Kobo) - each offers a dedicated device for reading E-books. And consumer electronics companies like Sony (Reader) are beginning to offer devices. Among the advantages of these devices is that they offer a form factor close to that of a real printed book; with e-ink they are often easier to read; and with e-ink they offer really long battery life (as much as two weeks between charges). Their disadvantages are that they are one more device to carry around; they cost between \$100 and \$200

each; and they are limited in the formats they can handle.

The next option that is becoming more popular is to use an e-reader application running on a tablet device, such as the iPad or one of the many Android tablets. All three of the above booksellers offer free applications for tablet computers. These allow you to search for and purchase books from your tablet, and then download them to it.

The last option, and the one I use, is to use software on my Android phone. I covered the application Aldiko in my previous article, which has both free and inexpensive paid versions. I also have the three bookseller applications installed. I personally find this the best option for one simple reason: I always have my phone with me. So any time I have a few minutes to kill, I can pull out my phone and do a little reading.

Formats

Sadly, there is no uniformity for



formats in E-books. Some of the formats have been around for a while, like the PDB format used by the Palm Pilot. Others are proprietary, such as Amazon's AZW format. In my previous article, I showed you how to convert books between most unprotected formats using Calibre. Since my primary e-reader software (Aldiko) prefers Epub, I look for books in that format, or books that are unprotected that I can convert to that format. You need to see which formats can be read in deciding on a device or on e-reader software.

DRM

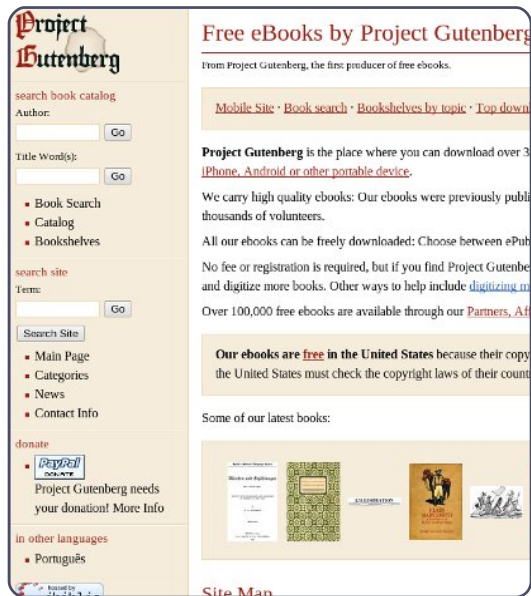
We seem to be going through a great deal of difficulty with publishers of all media over digital restrictions. It started with the music industry, which now seems to be wising up to how to keep their customers by providing music in convenient files that are not locked down. Between E-music and Amazon, I purchase all of the music I want in plain, unprotected MP3 files. Unfortunately, book

publishers are still competing to see who can be the most annoying to their customers. While there are various illegal methods for finding books, I do not intend to explain how to do that. I will point out that these "alternatives" tend to be low-quality and not very satisfactory. And I have found I can obtain high-quality books quite legally that are DRM-free.

Price

If you go to an online bookseller like Amazon and look around, you will see that most books being published these days are being offered in electronic formats at the same time as the dead-tree versions come out. The negative is that they tend to be offered at prices that are as high, if not higher, as the hardcover editions. But there are exceptions, some of which are quite nice. And there are places where you can get books for free, quite legally.

Project Gutenberg



If you know your history, you may recognize the name Gutenberg as the person who is credited with developing (in the West, I know) the printing press. This online project (http://www.gutenberg.org/wiki/Main_Page) took his name for creating a library of free books which are in the public domain. Public domain books are no longer subject to copyright, and there are a lot of good books, including many of the classics, that fall into this category. Here are just a few of the books you can find, which were selected by Wired magazine (October 2010) as the best free E-books you can find here:

- *A Connecticut Yankee in King Arthur's Court*, by Mark Twain
- *A Princess of Mars*, by Edgar Rice Burroughs
- *Frankenstein: Or, the Modern Prometheus*, by Mary Wollstonecraft Shelley
- *Gulliver's Travels*, by Jonathan Swift
- *The Adventures of Sherlock Holmes*, by Arthur Conan Doyle
- *Alice's Adventures in Wonderland*, by Lewis Carroll

Project Gutenberg has 33,000 books available, so you won't lack for good books to read. These books are not, of course, the latest best-sellers.

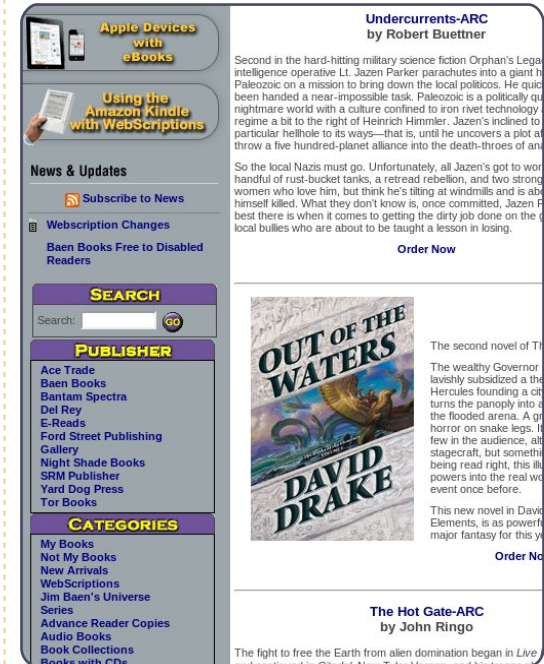
Baen Books



Baen Books (<http://www.baen.com/>) is doing something really good, and I hope it works well for them. They have created something called the Baen Free Library, in which they offer high-quality, absolutely free, and non-DRMed books for download. If you are a fan of Science Fiction and Fantasy (and I'd guess most of the folks reading this article are), this is a great way to start building your library. They offer a variety of the most popular formats as well.

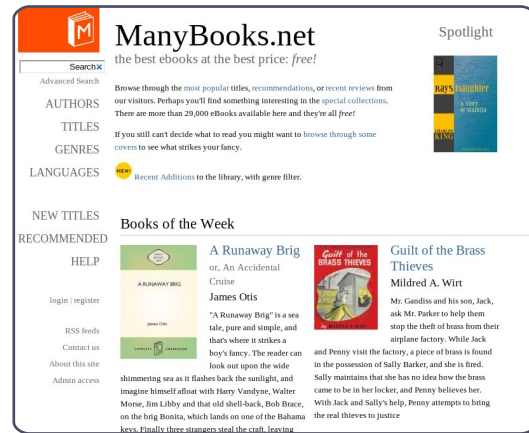
Now you might wonder what their business model is. And the answer is that the Free Library is just a selection from their much larger line of books. For example, you can download the first book of David Weber's popular Honor Harrington series, *On Basilisk Station*, for free. If you like it, they have at least 9 more, and they are available for only \$6 each, in high-quality non-DRMed files. I have downloaded a bunch of the free books, but I have also purchased a half-dozen books from them. I check their site periodically to see what is new. Their site is also worth visiting for other free content and author interviews.

WebScription



This (<http://www.webscription.net/>) is an offshoot of Baen's ebook site, but it offers books from a variety of other publishers as well. Ace Trade, Del Rey, and Tor are among the larger publishers getting on board, as well as some of the smaller specialty publishers like Subterranean Press and Nightshade Books. As you might expect from the Baen connection, the selections all appear to be Science Fiction or Fantasy

ManyBooks.net



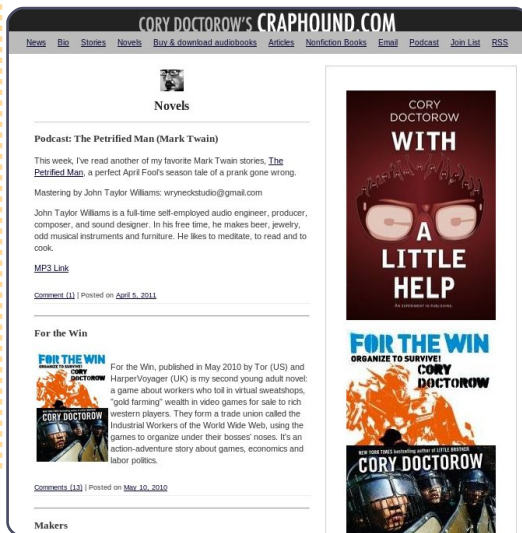
This site draws a lot of its content from Project Gutenberg, but also adds some things that are not Public Domain, but which have been made available. I have found books there that are current (e.g. Accelerando, by Charles Stross), so it is worth checking out.

Fictionwise



Fictionwise (<http://www.fictionwise.com/ebooks/multiformat.htm>) offers books (and some magazines) at very reasonable prices, and without DRM. The selection is heavy on Science Fiction and Fantasy, but does go beyond. They seem to have a lot of Romance, if that is your thing. I'd rate this higher except it looks like they have their own proprietary format and require you to use their reader. The reader is free to download, but, they don't have an Android version, and so I have not tried them out. Still, the rest of the deal looks pretty good, so check it out if you are interested.

Cory Doctorow



Cory Doctorow (<http://crapbound.com/?cat=5>) deserves a mention all by himself because of his stance on freedom. He insists that his books be available without DRM, and is opposed to DRM in all forms. He has made free e-book versions of his works available on his website, and, despite this, his sales keep going up. His view is that the two are related: the more people are exposed to his work, the more likely they are to buy his books when they get the chance. Right now you can download quality versions, without DRM, free of charge from his site, and he won't even let you put money in a tip jar. He says that if you want to support him, buy a paper copy and give it to a library. Pretty good advice, I think.

Summary

This is just a snapshot of the market as it is right now (I am writing this in early 2011). I'd bet things would be better in a year. Many of the authors realize that expensive DRMed e-books are not doing them any favors, and are just restricting their market. Just as musicians discovered - building a

fan base is much better in the long run. So I think we will start to see publishers try out offering books more conveniently.

But, until that day comes, we do have some options. I'd like to point out, as well, that sites like Project Gutenberg and ManyBooks ask for donations. If you are using them to get free e-books, give them a donation to help expand the offerings they can provide. It is just the right thing to do.



HOW-TO

Written by David W. Mawdsley

Arduino Traffic Lights

With the Heathkit company gone, and with a modern computer using Ubuntu 10.04 LTS, I found a website detailing a micro-controller named Arduino at www.arduino.cc (not .com). It used a USB cable for power and its computer connection. Arduino Uno was affordable at \$30. A simple traffic light simulation on a breadboard seemed just the fun experiment to try.

Getting things to work involved just three main tasks:

- Installing and configuring the Arduino IDE (Integrated Development Environment),
- Figuring out the wiring for the breadboard and the Arduino ports needed, and
- Writing a simple C program followed by an upload of the compiled code to the Arduino micro-controller.

Task 1 was simple, 2 was routine with some refresher study, and 3 required a small learning curve about C procedures - along

with some new commands specific to the Arduino ports. Uploading the compiled code to the micro-controller was easy. Within a few days of fussing with things, my system finally worked as designed. Later, I updated my code to include a buzzer in the circuit.

Part 1: Installing the Arduino IDE

(for details see <http://arduino.cc/playground/Linux/Ubuntu>)

I used the "Ubuntu without 'arduino' package" section of the page for my install. (Your installation may vary.)

Download the .gtz file, and install it with Archive Manager: *aduino-0022.gtz* (from <http://arduino.cc/en/Main/Software> using the Linux 32-bit package)

Install the compiler and the libraries packages:

```
sudo apt-get install gcc-avr
avr-libc
```

For those using the USB port to dialog, add yourself to the group 'dialout' to have write permissions at that port with:

```
sudo usermod -aG dialout
<your username>
```

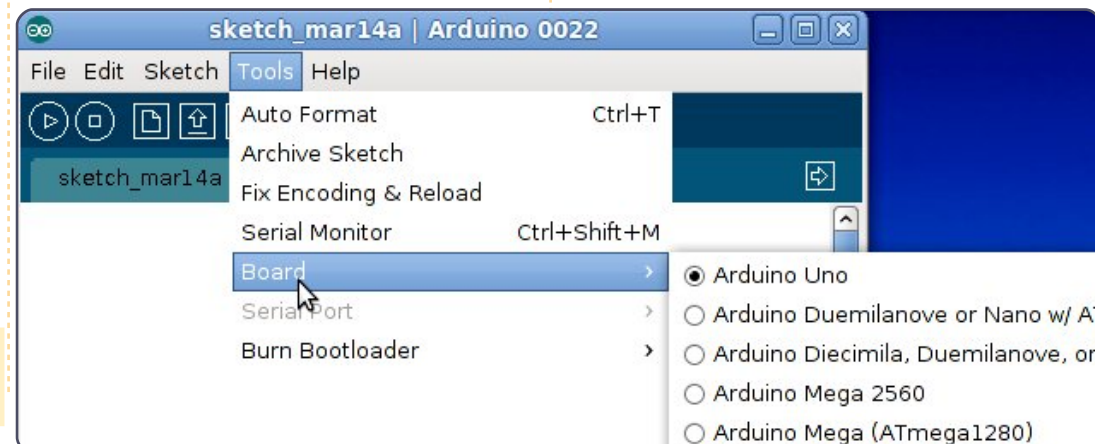
Next, to run the application, open the "arduino-0022" folder, right click "arduino" and choose "Run". Once the IDE is running, select your board (mine was the Arduino Uno) from Tools > Board.

Part 2: Wiring the breadboard and connecting it to the Arduino

Here are the parts I used to complete the wiring (mostly from Radio Shack):

- breadboard and a few spools of 22-gauge, insulated, solid-core wire
- 2.1mm power plug, and 9-volt battery cap (solder red lead to the center tap, black lead to the outside tap)
- 9-volt battery
- LEDs: 2 red, 2 green, 1 yellow (approx 2.1 mA forward current each. Note the orientation!)
- breadboard push button switch
- 3VDC Mini Buzzer (Radio Shack 273-0053)
- resistors 5-220 Ohm, 2-150 Ohm, 1-10K Ohm (all 1/8 Watt okay)

Shown below is the wiring



HOWTO - ARDUINO TRAFFIC LIGHTS

schematic, and a view of my Arduino mounted on a wood frame with the breadboard and the wires. (The Lego board was used to hold things loosely together.)

Part 3: Writing the code using C, and uploading the compiled code to the Arduino Uno

After some reading at the Arduino website, and with some borrowed code, the project code was one page in length using gedit. The file was then renamed

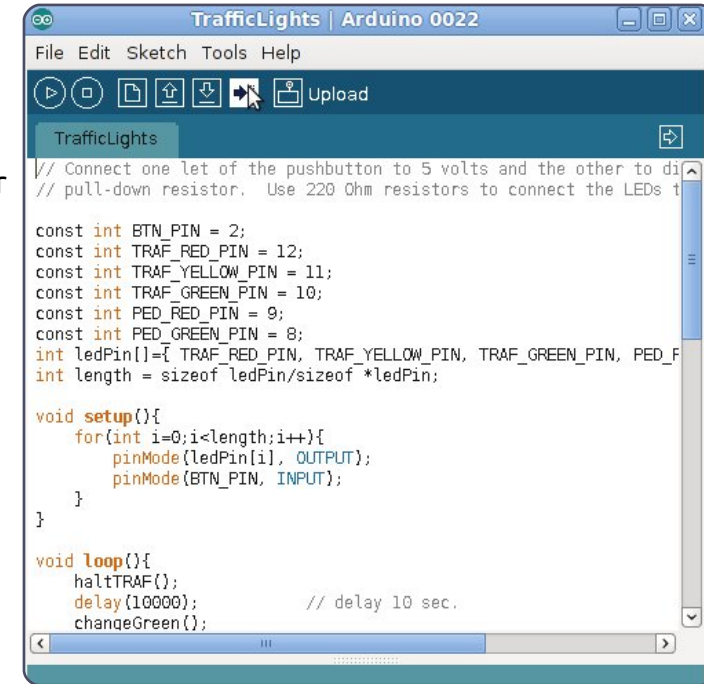
'trafficlightsound.pde' inside a folder named 'trafficlightsound' -- the name extension and corresponding folder name that the Arduino IDE seemed to require. Clicking in the IDE's start arrow at the left made the application verify and compile the code of 1468 bytes.

The source code for the Arduino traffic lights can be found at: <http://pastebin.com/ACk9u937>

After connecting a USB cable between the computer and the micro-controller, one click on the right arrow on the second row square of the IDE started the

upload of bytecode to the microcontroller. Once loaded, the Arduino Uno ran the various lights and buzzer as designed.

After disconnecting the USB cable and connecting the 9-volt battery, the project ran independently from the computer. The program will continue to run in a loop until the power is removed.

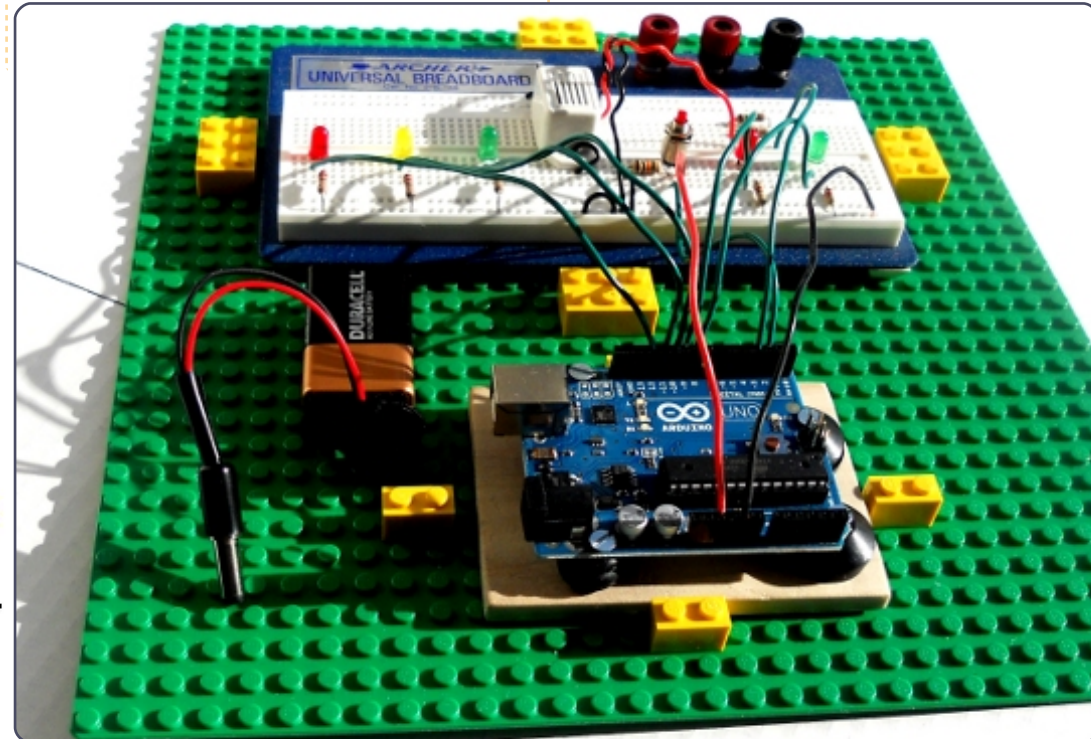
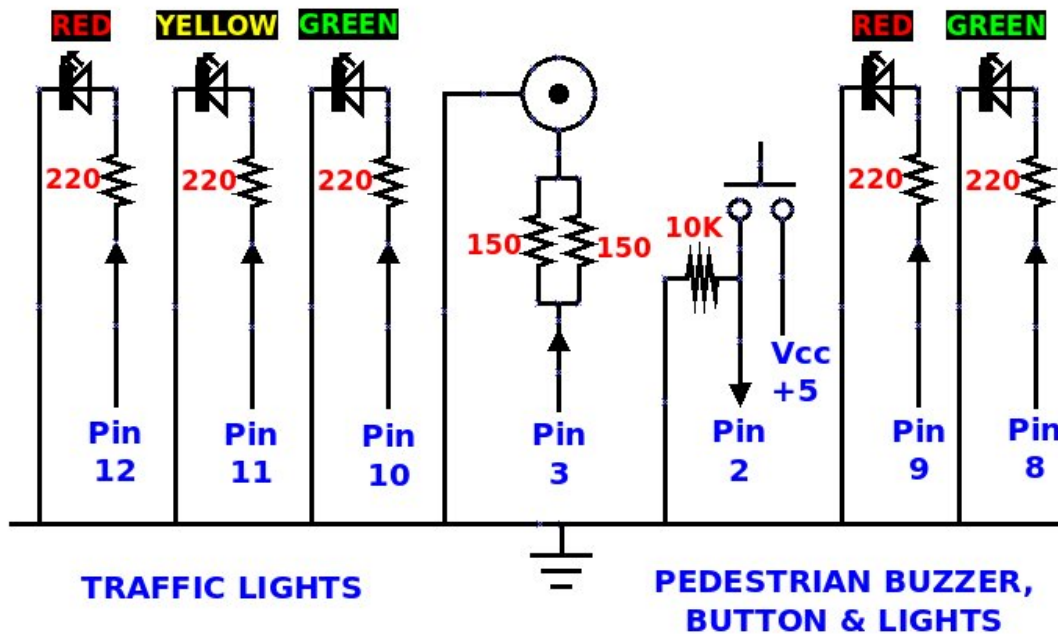


```
File Edit Sketch Tools Help
TrafficLights
// Connect one let of the pushbutton to 5 volts and the other to di
// pull-down resistor. Use 220 Ohm resistors to connect the LEDs t

const int BTN_PIN = 2;
const int TRAF_RED_PIN = 12;
const int TRAF_YELLOW_PIN = 11;
const int TRAF_GREEN_PIN = 10;
const int PED_RED_PIN = 9;
const int PED_GREEN_PIN = 8;
int ledPin[]={ TRAF_RED_PIN, TRAF_YELLOW_PIN, TRAF_GREEN_PIN, PED_F
int length = sizeof ledPin/sizeof *ledPin;

void setup(){
  for(int i=0;i<length;i++){
    pinMode(ledPin[i], OUTPUT);
    pinMode(BTN_PIN, INPUT);
  }
}

void loop(){
  haltTRAF();
  delay(10000); // delay 10 sec.
  changeGreen();
}
```





Guidelines

The single rule for an article is that **it must somehow be linked to Ubuntu or one of the many derivatives of Ubuntu** (Kubuntu, Xubuntu, Lubuntu, etc).

Write your article in whichever software you choose. I would recommend OpenOffice, but **PLEASE SPELL AND GRAMMAR CHECK IT!**

Writing

In your article, please indicate where you would like a particular image to be placed. Please do not embed images into your Open Office document.

Images

Images should be JPG with low compression.

Regarding image sizes: if in doubt, send a full size screengrab and we will crop the image.

If you are writing a review, please follow the guidelines shown here.

For a more detailed list of the style rules and common pitfalls please refer to: <https://wiki.ubuntu.com/UbuntuMagazine/Style> - in short: US spelling, no l33t speak and no smilies.

When you are ready to submit your article please email it to: articles@fullcirclemagazine.org

If you can't write articles, but hang out in Ubuntu Forums, send us interesting forum threads that we could print.

Non-English Writers

If your native language is not English, don't worry. Write your article, and one of the proofreaders will read it for you and correct any grammatical or spelling errors. Not only are you helping the magazine and the community, but we'll help you with your English!

REVIEWS

Games/Applications

When reviewing games/applications please state clearly:

- title of the game
- who makes the game
- is it free, or a paid download?
- where to get it from (give download/homepage URL)
- is it Linux native, or did you use Wine?
- your marks out of five
- a summary with positive and negative points

Hardware

When reviewing hardware please state clearly:

- make and model of the hardware
- what category would you put this hardware into?
- any glitches that you may have had while using the hardware?
- easy to get the hardware working in Linux?
- did you have to use Windows drivers?
- marks out of five
- a summary with positive and negative points

You don't need to be an expert to write an article - write about the games, applications and hardware that you use every day.



I am prompted by my co-podcaster and fellow columnist Ed Hewitt to amend a statement I made in File-systems Part 1, which stated that you won't get through a Linux install without defining a swap partition. Whilst most of the installers these days will let you through with a warning of how this is inadvisable, few will stop you in your tracks for this 'sin'. Let's step back for a moment.

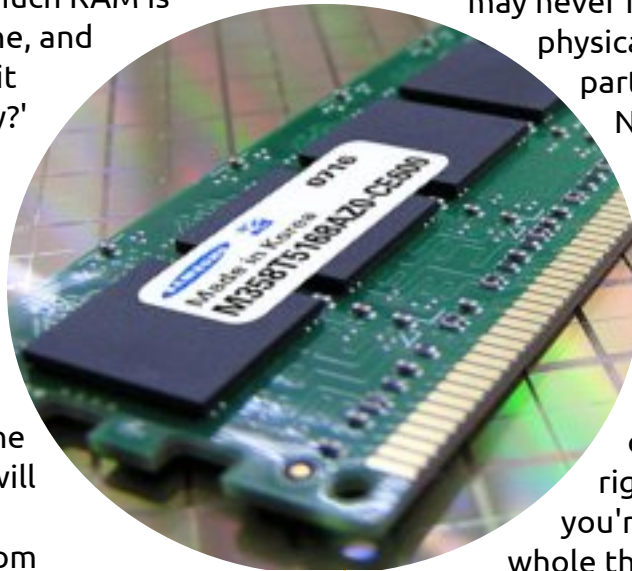
Why do I need a SWAP partition for Linux?

Swap partitions are necessary for those computers which have less physical memory (RAM) than the applications need. Think of a swap partition as temporary storage which is used when all the physical memory is in use - with no further space for data and programs. Given the complex operating systems we now run, with advanced graphics, large programs, and multi-tasking, you can soon use your physical memory resources fully. In this case, the operating system will

swap out some of the programs and data to temporary storage. With plenty of physical memory available, the swap partition may never be used and the space would be wasted. So the question is really 'how much RAM is in my machine, and do I ever fill it past capacity?' Answer yes, and you could benefit from a Swap partition. It may not be needed all the time, but it will help your computer from freezing at times of over-capacity.

How do I know if I need one or not?

Apologies for answering a question with more questions, but what's your use-case? What's your operating system and your peak workload?



Got a netbook, with 1GB RAM and Ubuntu Netbook edition; mostly surfing the web, writing emails, and the odd wordprocessed document? You may never fully use all the physical memory. Swap partition needed? No. However, jump onto Skype for a conference call with 50 tabs open in Firefox, you'll probably roll-over into Swap right there. Unless you're Ed and the whole thing locks up.

Insert smiley face here.

My old Toshiba Satellite has only 196MB RAM. Running a light-weight Linux such as Crunchbang or DSL for some light surfing, it's fine with no Swap. Step up to Ubuntu 10.10 with LibreOffice and Firefox running, now I roll over into Swap.

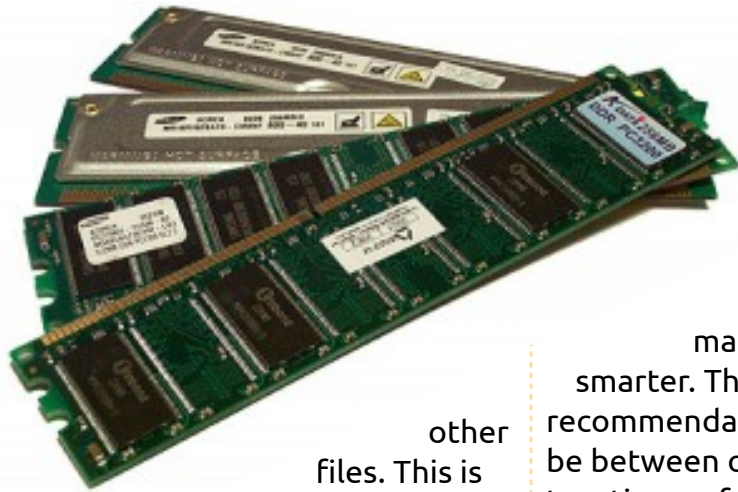
My Dell 6400 with 4GB RAM

and a fully loaded Ubuntu 10.10 is fine with Firefox, Chrome, and OpenOffice multi-tasking together, using no Swap for whole sessions at a time. Launch Audacity sound editor, and OpenShot video editor for some work on the Podcast, and YouTube Hi-Definition, and I'm back into using Swap.

Any machine running a current, full-size operating system (not a light-weight), with a small amount of physical memory, maybe 256MB or less, will need a Swap partition. As workload rises, with more programs open working on larger data files, you'll push through your maximum physical memory threshold - be it 512MB, 1GB, 2GB or higher - into Swap.

Why a Whole Partition?

There is an alternate approach to this 'virtual' memory management, it's called Page Files. In both Windows and some Linux configurations, page files reside on the main program- or data-partition, alongside all your



other files. This is often considered, shall we say, sub-optimal, both for performance and for data security. Page files can be very large, demanding fast on-demand writes to disk. It can cause an Input/Output bottleneck, and when you've had a Windows pagefile trash a chunk of your active partition - losing precious data and programs, you appreciate the Linux approach - segregating Swap from everything else.

What size swap do I need?

As we've seen, maybe none. When we do need one, the trick is to balance the use of smaller, faster RAM against slower larger disk, so that you (or rather the operating system kernel) get the

best performance out of that combination. The truth is that - with modern kernels like we have in the Debian 2.6 family - memory

management is a lot smarter. The old recommendation that swap should be between one-and-a-half and two times of the physical memory is probably over-generous for a desktop machine, but barely sufficient for a server. Setting a swap space between half and equal the amount of physical RAM should be adequate. If you have a laptop and set it to 'suspend' to disk, then you need swap space equal or greater than physical RAM. If in doubt, choose a larger swap, since a couple of gigabytes won't be missed - unless you have a Solid-State Drive, but that's another story...

In Part Two, Virtual Memory Management, Swap-On, Swap-Off.



A PLEA ON BEHALF OF THE PODCAST PARTY

As you'll heard in episode #15 of the podcast, we're calling for opinion topics for that section of the show.

Instead of us having a rant about whatever strikes us, why not prompt us with a topic and watch for the mushroom clouds over the horizon! It's highly unlikely that the three of us will agree.

Or, an even more radical thought, send us an opinion by way of a contribution!

You can post comments and opinions on the podcast page at fullcirclemagazine.org, in our Ubuntu Forums section, or email podcast@fullcirclemagazine.org. You can also send us a comment by recording an audio clip of no more than 30 seconds and sending it to the same address. **Comments and audio may be edited for length. Please remember this is a family-friendly show.**

It would be great to have contributors come on the show and express an opinion in person.

Robin





MY STORY

I just completed my first year as a Ubuntu user and was asked to write an article about my experience. The invitation to write caused me to look back on the year and ask myself, what have I done this year? Well, the first year has taken me through three different Ubuntu operating systems starting with 9.10 and ending with 10.10.

Not knowing much about the terminal when first using Ubuntu I focused on the GUI. As I learned how to install themes and customize the Ubuntu desktop, I began to help other new users with this process. The task of figuring what packages need to be extracted, how and where they are installed is simple after you've done it, but potentially confusing to a new user.

I would like to mention the sticky in the desktop environments section of Ubuntu Forums as it is a nice list of resources for new users interested in customizing their desktops [1]. I mention this because many of the questions I

have answered could have been avoided by using this resource.

It's great feeling to help another Ubuntu user and when they proudly post a screenshot of their personalized desktop or send a thank you it's even better.

Frogs Hair

[1]:
<http://ubuntuforums.org/showthread.php?t=809695>

I have been using Ubuntu for more than 3 years and am almost completely Window's free and cannot speak highly enough of Ubuntu.

Pretty much all computers are sold with Windows pre-installed and if you want to use a Linux distro you have to install it with a dual boot or completely remove Windows. I tried a dual boot at the onset of my incursion into the world of Linux and was not happy with the problems that appeared. A full Ubuntu installation is definitely the way to go. Under

Windows, I used several programs that will not work in Ubuntu (not under WINE either) but have all but conquered this annoyance by using similar Ubuntu software and some web based applications that do the job just as well and, in some cases, better.

There is a stigma attached to Ubuntu and Linux users in general; that is to use it you have to be a computer expert to get it to work. If you check computer hardware supplier's websites the majority state that Linux is for computer experts, implying that Windows is simple. There is no mention of the extra software you have to install and pay for to stop an attack from the Internet.

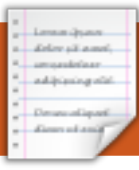
The Linux stigma is therefore unwarranted. I do not consider myself to be a computer expert and am made to accept this fact by some FCM articles that go over my head. What one does have to accept is that there is a distinct learning curve. I remember my very first Ubuntu installation and the message that I needed to run a

command in a terminal. At the time it was baffling but is not a problem now.

In the February edition of FCM the article about LibreOffice was of great interest. I had actually installed it prior to reading the article. It is a lot faster and has a lot of features that OOo does not have. I highly recommend it. It is being supported by Canonical so it is the way to go.

I have tried, on many occasions, to convince my work colleagues to use Ubuntu but they have their head in the sand and stay with XP. However, recently my grandson showed me an article that had been presented at his school related to Linux. It was extremely well written and listed the advantages of Linux in the classroom. I was very surprised to read such an article and asked who had written it. It was amazingly my grandson which was all the more surprising as he is only nine years old.

Allan Hambidge



MY OPINION

Written by Allan J. Smithie

Remember the scene in Star Trek IV where Scotty tries to operate a Mac workstation?

I may be mis-remembering the dialogue, but it goes something like this:

McCoy: *"You have to use the mouse."*

Scotty (picks up mouse and speaks into it): *"Hello, computer."*

McCoy: *"Just use the keyboard."*

Two things: One, I apologize for the Star Trek spring-board. Two, evidently the Mac is not as intuitive as we all thought, even to a starship engineer from the twenty-third century.

Heralded as a great leap forward, the Graphical User Interface (GUI) that came out of Xerox PARC, Palo Alto in the seventies - yes, almost forty years ago - pioneered the use of WIMPs - Windows, Icons, Mice, Pull-down Menus. Although strictly that should be Mouses, as Mice is the plural for small rodents - not

computer pointing devices.

Yes, the GUI was a huge improvement on what came before. In the seventies, the Data Processing Department relied on punch-cards and paper tape. In the eighties, we got as far as green-screen terminals and mind-numbing keying of program code. Try type-setting a magazine on one. No, don't. I did. I believe that is why I am now gray-haired. The advent of the cheap(-ish) PC brought computing to the (relatively well-off) masses. The GUI accelerated the take-up, not as the catalyst, once the machines became powerful enough to run a GUI with some application programs atop it.

Xerox may have invented the GUI as a corporate tool, but it was Macintosh and PC that "liberated" it from the then-walled garden of the Unix X-windowing system. Here at last were machines for the rest of us, that we could all use. Kinda.

I may be showing both my age

and my ignorance of factual computing history, but it's never stopped me in the past. I can say with some certainty that there is nothing intuitive about the current or previous generations of GUIs. None of them. We all have to be shown how to use them. The operation of mouses and menus is not an innate behavior to the human primate, any more than language - another subject I argue about with my friends in teaching. A certain well-known US TV presenter recently learned how to Tweet. First he had to be taught how to operate the Twitter website with a mouse, which he first tried to touch to the screen. It's true. Bright people don't necessarily get it. I believe the number of mouse buttons affects the learning curve in inverse measure. I buried a former friend under my patio for having owned a Logitech super-mouse with seven buttons and a scroll-wheel. Not really, but I thought of it on every visit to his office.

The problem worsens over time, whereas one expects it to

get better. The sheer multiplicity of devices, copyrighted, patented, and trademarked, each with its attendant software, also copyrighted, patented, and trademarked, is making it more difficult to be productive, not less. Consistency would be a boon, only commerce won't allow it. Yes, we have standards beyond individual platforms such as i-OS, Windows, and, thank you Hewlett-Packard, Web-OS. Linux has Open Desktop.org. Not that you would know it between Gnome, KDE, Xfce, LXDE, Sugar, Linpus, Chrome, and many other re-badged deviations. Android is splintered whilst Meego falters, and Symbian... does whatever Symbian does in the market these days. Double-tap, pinch-to-zoom, tap-to-focus - all work in slightly different ways, and the menus of any two smart-phones are seldom the same.

Icons. These are religious works of art. The universal language of icons in computing is not universal. Nor is it a language. There are no standards and little permanence, since icons are either about



MY OPINION

creative artistic interpretation or marketing and branding. Here endeth the lesson.

Minority Report

Yes, I do mean Tom Cruise with a data glove, waving his arms about in front of a holographic projection of data. Anyone who's tried a data-glove and VR headset knows just what a fool they look (hey, got the t-shirt from VR-expo London in 1995). I'll take it if it means no more mice. Or labyrinths of menus. In recent months, we see the first signs with the Kinect and the Wii-Motion controllers. Yes, you still look like a fool.

'User-Friendly'

A term so vague, it never lost common currency, it never held common currency. I refer back to my previous point about icons. 'User friendly' is what we all want, despite the fact that we can't define it and it, too, changes over time. I submit that the next generation of GUI has to be:

- flexible, accommodating all tastes, abilities, handicaps and

cognitive dissonances (bingo players may take a drink now);

- consistent in the learning and operation;
- layered in its inevitable complexity, simple in the common day-to-day operations, and as complex as needed to achieve those more advanced;
- and task-oriented - really, don't just say it, do it.

What does it look like? I have no idea. It's not OS-X, Ice-Yeti, or KDE 4.7. It goes beyond i-OS, Android, and Windows Phone 7. Just because our kids are really fast with them, doesn't mean they're good. These have all evolved from the past, and it has to be a break with the past. It has to be designed around real people and not around the obstacles the software engineers find with the available hardware. Just imagine.



Allan J. Smithie is a journalist and commentator based in Dubai. His blog is at:
<http://allanjsmithie.wordpress.com>



FREE UBUNTU CD FOR EVERY READER! *

- 1 - Print this page using a modern, colour, printer. This is a critical step.
- 2 - Carefully, using sharp scissors, cut around the perimeter of the above CD. If you have not printed this page you will find scissor scratches almost impossible to remove from your desktop/laptop screen.
- 3 - Foolishly place your new paper 'CD' in your CD/DVD device.
- 4 - Scratch your head, baffled as to why your CD won't boot.
- 5 - Direct all complaints to mrmondav@fullcirclemagazine.org as he probably knows nothing about this.

WARNING: YOU MUST HAVE A SUPERVISING ADULT PRESENT WHEN YOU ARE USING SCISSORS! ESPECIALLY SHARP SCISSORS.

* Neither Full Circle magazine, nor it's creators, will be held responsible if your CD/DVD device eats your paper CD.

MORE UBUNTU!

Can't get enough Ubuntu?
We've got a whole lot more!
DON'T MISS ANOTHER ISSUE!

Ubuntu 10.04
Kubuntu 10.04
on a double-sided DVD

ubuntu 10.04 Lucid Lynx

UBUNTU
user
EXPLORING THE WORLD OF UBUNTU

TOTALLY LUCID

THE LYNX LEAPS
What's new in Ubuntu 10.04?

HUGE SAVINGS OFF THE NEWSSTAND PRICE!
SUBSCRIBE NOW!

TUNEUP FOR STARTUP
Find out why Lucid boots faster

Getting around in Launchpad
New ink: Exploring OpenOffice 3.2
Create your own e-books

DISCOVERY GUIDE



WWW.UBUNTU-USER.COM/SUBSCRIBE-NOW



REVIEW

Written by Art Schreckengost

Remastersys

Remastersys is one of the mysterious programs Ubuntu users may know of but rarely try, and this is unfortunate since it's a program with a lot to offer.

The official website, <http://geekconnections.org/remastersys/>, is the only place to get valid information, and I advise all to go there. Tony Brijeski is the developer, and claims his program is "*A unique Backup-to-Live-Media Tool for Debian and Ubuntu*" (from the website).

Be wary of other websites that offer instructions and downloads that are usually outdated. Stick with the developer in this case as menus and choices have changed over time.

Do not confuse this with APTonCD, another program that is designed to backup applications, not the OS with applications. Remastersys backs up everything.

What is required? Ubuntu, or a variant using the base Ubuntu

code. All desktop styles are invited to the party too. That's easy enough.

Unless you have a top end OS package like openArtist, chances are you'll have to install remastersys. Go to the website mentioned above, and follow the instructions (you'll have to add a software source to Synaptic).

Before you happily click on the new menu entry, do some preliminary work. Please do not ignore this information!

Use Ubuntu Tweak or Computer Janitor to clean your system of garbage files, unneeded cache items, and discontinued kernels. My last cleanup operation cleaned out over 1GB.

Now it's time to brush up on your elementary percentages.

Remastersys can create a final file of no more than 4GB (this is a limit of the genisoimage protocol in Ubuntu), but this is misleading because that's the limitation of

the final compressed file, not the size of your occupied hard drive.

Open Disk Usage Analyzer (DUA) on the main menu, and review the numbers. Figure one is total hard drive space, and the second is OS occupied space. For example, you may have a 250GB hard drive with only 4GB occupied.

Remastersys takes the second figure and compresses it to a much smaller file, usually down to 33-50% of original size. This is where the math gets a little fuzzy.

Some files are already compressed and can't be squeezed further. MP3 files are about as small as they're going to get, so you'll have to remove them or pay the consequences.

How so? If you have 8GB on the hard drive, but 3GB of that are music files, only the 5GB balance will be reduced. That could put you over the 4GB compressed file limit.

Take it from a person who has made this mistake. Move

multimedia files off the hard drive to an external drive. This might explain why Ubuntu variations come with few, if any, such files.

In short, keep your hard drive space to no more than 8GB, since anything above that is pushing the limits (my personal best is 12GB but that was really on the limit).

And how big is 8GB? The average Ubuntu installation takes up about 4GB, so you've got plenty of wiggle room for programs, files, etc. ArtistX with 2,500 included packages and programs is still under 4GB compressed, even though it balloons to nearly 12GB once installed (and they used remastersys to create the download file on their website).

Housecleaning still isn't done quite yet. Shut down internet and Bluetooth connections, if any. Stop playing music files, and plug in your laptop if it's on battery power. Disconnect external hard or thumb drives, and remove that SD card you forgot about. Get rid of the CD or DVD in the drive, if any.



Finally, disable any screensavers that may pop up after a period of disuse.

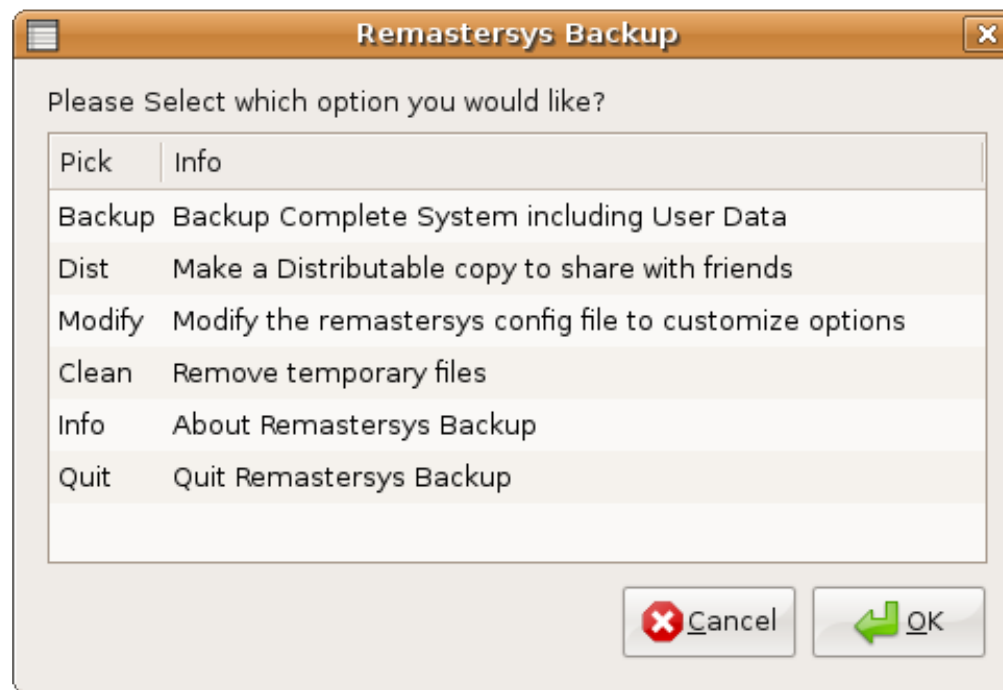
Why all this work before you even open the program? DUA is a great program, but often overlooks anything past the hard drive. External drives and cards are easy enough to forget, and your compressed math figure could be incorrect as a result.

In addition, remastersys is going to take over your computer for the better part of a half hour, so plan on some idle time. Anything that can interrupt the project, such as a screensaver popping up, can cause termination (although it rarely does).

Now you can open remastersys.

You should get this standard warning:
It is necessary to close all other windows and unmount any network shares while running Remastersys Backup. Please do so now and then click OK when you are ready to continue.

Essentially, make sure you have nothing running in the



background. Simple enough, and clicking okay will take you to the first menu.

Do not click on anything in the main menu until you read below!

The main menu is straightforward but does have entries that merit closer definition. You will see Backup, Dist, Modify, Clean, Info, and Quit, along with a couple of other headings, but included explanations may seem cryptic - so here's a common sense, plain English, listing:

Backup – Clones everything, warts and all (including those files I advised to get rid of earlier – now you know why).

• **Dist** – Allows you to make a copy for distribution to friends by stripping personal info and related data. Essentially, it backs up your programs, but not your data.

• **Modify** – Allows users to modify settings and exclude files. More on this below.

• **Clean** – Cleans out temp files

from previous Remastersys attempts. Read below.

• **Info** – All about the program - as if you didn't already know.

• **Quit** – Self explanatory.

I purposely left out Distcdfs and Distiso, two entries that make little sense given Dist offered above (if you must know, one creates a CD file system, and the other creates an .iso image, but Dist alone does most of this without having to fool with these). Previous versions did not have these two headings, so don't worry if they're not there on your version.

At this point, users can make a mistake by clicking on Backup or Dist! Clicking either starts the process immediately without modification or intervention. Keep reading before taking any action!

Default is Backup but you can use the tab keys, cursor, or your mouse, to jump to any other category. Clicking okay activates the heading you have highlighted.

Clean is strictly for previous remastersys users, and allows for

prior attempts to be removed so they don't add to the new clone.

The only problem with Clean is that it specializes in temp files, and might not touch an existing clone you created earlier. In these cases, you may have to visit the remastersys folder to manually remove it (right click the file and move it to trash).

The next command, Modify, is the one that should be clicked first!

Of all the options, the one most useful is Modify, because it branches out to a sub-menu consisting of:

- **Username** – Defaults to “custom”, but clicking on this allows changes.
- **Title** – Same as above, but usually self-titles as “Custom Live CD” - even if it is DVD size.
- **Filename** – Defaults to custom.iso. Leave well enough alone here since you're after an .iso image.
- **Working Directory** – Defaults to

the Remastersys folder, but users can change this to Desktop for ease of finding it later.

- **Files to Exclude** – Allows files to be removed from the mix, but read below before going here.

- **Go back to main menu** – A.k.a. quit.

Modify does have a flaw from a user standpoint, and that's in the Files to Exclude heading. The box that opens does not point to any file folders - or even offer to. Users are expected to cut and paste, or manually enter file-paths (leaving a space in between for each entry). Make a typo and you've got problems.

Hmmm, maybe moving files to a flash drive doesn't sound so bad now, does it?

It's Modify that often confuses the most. Don't be spooked into thinking you've just butchered the OS. The only modification is to the final product, not the active OS.

Once you've got this settled, click on okay, go back to the main window, and click either Backup or Dist to start the process. Backup is for personal use, while Dist is a copy stripped of personal data so you can give copies to friends or post online.

As a matter of preference, I almost always opt for Dist. That way, if a disc gets in the wrong hands, my user-ID and password aren't compromised.

Once you're sure, click on okay, and sit back for a while. Depending upon the size to be compressed

you've got about 15 to 30 minutes to kill.

Monitor the pop up window that appears! Down at the bottom you'll see a notification of final file size. If it's more than 4GB, click to close the window to stop operations!

This is an unfortunate snafu in remastersys. It blindly follows your command even if the final file is too large for completion. Of course, you do get a little nasty pop-up box stating the final file was too large, but you don't get it until the bitter end!

Back to square one, and this is where Clean comes into action. Click on it to see if any residual temp files are hanging around. Consider putting more files on external drives or storage, and look for other programs to dump.

Once the pop up window shows the final file will not exceed 4GB, you're home free.

If all goes well you should see: *our custom .iso and custom.iso.md5 files are ready in /home/remastersys/remastersys. It*



is recommended to test in a virtual machine or on a rewritable cd/dvd to ensure it works as desired. Click on OK to return to the main menu.

If the size was right you'll have the .iso file sitting in the Remastersys folder (or the location you picked). You have two realistic options:

- Burn an .iso image CD or DVD. As stupid as it may sound, make sure you use an .iso burner. Burning a video or data disc isn't going to work.
- Virtualbox - if you plan to use the clone as a secondary, not main, OS on another computer; however, you still have to burn a disc, use a flash drive, or transmit across the network to transport the file.

Another potential option is UNetbootin and a flash drive, but that procedure has been hit or miss for me. Don't know if it's the file, or a bug in the program (either UNetbootin or remastersys), but UNetbootin often can't find the image file - even when I park it on the desktop. It did work occasionally, but I eventually found it easier to just burn a DVD and be done with it.

Those itching to test the final product - without burning a DVD or moving the physical file - can open Virtualbox and load the OS for review, but it may seem odd to have a copy of your current running OS next to the original (still running, too).

To boot to the image file on CD/DVD or flash drive, reset your BIOS to boot from the media used, and you should see a menu at boot giving you several options:

- **Start Custom Live CD in Graphical Mode** – essentially Ubuntu live mode.
- **Start Custom Live CD in Safe Graphical Mode** – mainly for computers that won't boot otherwise.
- **Install Custom Live CD** – complete installation.
- **Check the CD/DVD for defects** – if it's defective, will you get this far? Maybe!
- **Memory Test**
- **Boot the First Hard Disk**

I always recommend live mode before any installation - only because no two computers are the same. Drivers and codecs are bound to be missing, and this may be the time you suddenly realize you should have removed those compromising pictures from that folder.

By the way, that compressed file in the remastersys folder is not going away anytime soon. It'll stay there until you remove it, and failing to do so and using remastersys again will result in that file possibly being included in your second attempt. Oops!

In other words, burn your discs, and then delete the iso once satisfied.

Anybody who has ever used Windows-based cloning programs can attest to the fact that remastersys is one of the easiest programs to use, and one of the quickest once the initial tricks and roadblocks are conquered.

As for a rating:

- **Ease of use.** Once past the initial

jitters, it's a good program. Some online information is downright incorrect if not hazardous - but this is not a fault of the program. Dinged a diamond for a rather Spartan website although I did find it amusing that the program is visually appealing compared to the website.

• **Operations.** Dropped a mark for a command-line file-exclusion process that is twitchy, and a propensity to create non-existent files if the final file size is too large (it should halt the process, and warn users before starting).

• **Overall rating.** There just isn't anything much else out there for Ubuntu and variants in this area, but this program works just as well as much more expensive Windows and OS X software. Just needs a little polish to be excellent.



Dell a No-Go?

I just checked the Dell site (per the Q&A in Issue #43) and found out that Dell is no longer offering any laptops or desktops with Ubuntu Linux (at least not in the United States).

For that, Dell is a no-go for me (as far as future laptops) unless they decide to start offering Ubuntu as an option.

Patrick

No Copy, No Paste

I just want to let you know that I too have a problem with copy and paste on both Windows XP and Ubuntu 10.10.

I use an older version of Foxit Reader (2.3) on XP and the standard software on Ubuntu, which is a standard install.

Grant

I love FCM and promote it heavily through our public library and the open source events I do there as a volunteer.

I just read FCM#47 and happened upon the feedback on not being able to copy and paste. Curious, I decided to try it on FCM#47 and I could not! I'm using Mint 9 and tried to copy and paste from Evince and from the dynamic version displayed via Firefox; no dice. I tried the two special editions and yes, I can in those. I then went back through through progressively older issues and found that I could not copy and paste back to #36. So, I decided to try opening some issues using Okular; same thing - still couldn't do it. Thought you might find this interesting.

MarkB

I just tried this with Kubuntu 10.04 and Ocular. I can copy to clipboard and file as an image only. The copy as text to

clipboard won't work. Up to issue 39 I could and I was able to in issue 40 (some rights reserved) where the text is not in columns. Something happened with issue 40.

I have never liked Ubuntu as I couldn't find how to set the refresh of my CRT monitor and it was real easy in Kubuntu. I started with 5.10 but only really got into using it full time since 7.04. I stuck with 8.04 on my desktop until 10.04 came out but had tried 8.10 and later on my laptop using KDE 4.x and have been happy with it.

I know it's an Ubuntu magazine but it would be helpful to explain things about Kubuntu as well, as sometimes it is different between the two, especially system settings like networking.

Neville Friedrich

Ronnie says: *Quite a few readers emailed to say that they can't copy and paste from FCM. Having asked assistance on the Scribus forums I think (hope!) it's fixed for this, and future, issues. @Neville: Although*

Join us on:



*we predominantly feature Ubuntu we are open to articles about all *buntu derivatives. I also use Kubuntu and hope to write some Kubuntu articles shortly.*

Beta Not

Today I installed the first of two betas, before the final release of Ubuntu 11.04 that gets released this month, and I must say it is in no way ready for the casual user. This version is one to keep the newbies away from.

Upon installing on a clean machine I was met with a loss of Metacity window decoration, even though it was present during the live USB test run. After some trouble shooting I was able to get

Unity 3D and Compiz working together. However, I experienced app crashes and lock ups through most of the run on the test machine.

I know its just the first beta but with a just a few weeks left for bug fixes and freezes I dont think this year's first release of Ubuntu is for the faint hearted more for people who enjoy trouble shooting. Yes, Unity is workable, but I'm afraid I have to agree with Robin Catling it's not ready.

I haven't experienced a version of Ubuntu, even in beta, that was this frustrating since the days of 6.x. Part of it is, of course, the new interface which will take time to learn. But, coupled with these bugs, it makes for a rough ride that I don't think new users, or people accustomed to just a GUI, will like. My advice? Stay away from it until 11.10, or they do a service pack to 11.04. Its not going to be released hassle free and new-user friendly.

Dougn Redhammer

Linux On Mac

I'd be interested to read about people who use a Mac, and dual or triple boot Apple's OS with Linux. Could someone write about this?

I'm personally thinking of getting a Mac for the stability, but want to do multiple booting to have the freedom of Linux's customization, and some software I have a proprietary Linux license for.

Ludo Beckers

Mail Server

I am glad you guys are working so hard to spread info for people like us who want to learn Linux especially Ubuntu. Will you be running any articles about creating a mail

server? As I want to learn how to create one using Ubuntu Server Edition. I hope someone can help.

Leo Marloe Dicang

Ronnie says: *Anyone out there want to write an article, or several, on creating a mail server? If so,*

please email me a brief outline of your article, and let's help Leo (and others) get a mail server up and running.

Paul proudly presents his first do-it-yourself toner refill to his wife, Wilma.....



costantinos.bourboulas@oracle.com Apr-11

Modern Times



UBUNTU WOMEN

Written by Elizabeth Krumbach



Elizabeth Krumbach: Please tell us a little about yourself.

Jessica Ledbetter: Hello everyone, I'm Jessica Ledbetter (<https://wiki.ubuntu.com/jledbetter>). I've been a web developer for a Department of Energy lab in Virginia for about 10 years, and I code primarily in Java and ColdFusion, plus freelance in PHP. I was the first in my family to go to college, and, so far, the only one to finish. I worked while getting my Bachelor of Science in Computer Science, and later a Master's degree in Information

Technology.

EK: What inspired you to get involved in the Ubuntu community?

JL: I have used *nix for over 15 years, and, even though I'm a visual person, I really like the command-line interface for compiling, finding documents, and the like. I looked into Linux distributions as a better programming environment though I really enjoyed my Mac for design work. My partner was a huge fan of Ubuntu, so we went to a Linux Fest in Florida where I met more people from the Ubuntu community. I was hooked instantly. There are a lot of distributions out there, but I think Ubuntu has one of the most amazing communities around. It's hard not to be involved.

EK: What are your roles within the Ubuntu community and what plans do you have for the future?

JL: Right now, I'm one of the leaders of the Virginia Local Community Team in the United

States, a member of Ubuntu Women, and a member of Ubuntu Beginners Team. Also, I have co-lead a session for Ubuntu Open Week Maverick. Recently, I was honored with being on the nomination list for the Beginners Team Council.

As a current master in the Beginners Team, I hope to funnel new developers into the Ubuntu project. Through that position, as well as future screencasts that I have sitting on my desktop, I want to be able to help answer one of the most frequent questions I see asked and have asked: "I'm an [insert language here] programmer. How do I contribute to Ubuntu?" In addition to these roles, I'm also working on a short session in Peer2Peer University (<http://p2pu.org/>) about how to contribute code to an open-source project.

EK: Have you hit any barriers with getting involved, and what can you recommend to newcomers?

JL: There's so much information that sometimes it's hard to know

where to start, what questions to ask, and where to ask them. I began by lurking in the Ubuntu Women IRC and Florida LoCo channels, then asked questions of those who seemed most approachable. And from there, I started to venture out based on people and projects I learned of via those channels. My advice is to ask if you're unsure of what something means or how to get involved. Everyone can contribute - you don't have to be a coder! Though, if you want to learn how to program or contribute as a programmer, there are lots of ways to do that too! A new gateway for new developers is coming together at <http://developer.ubuntu.com/>, and there's also the Beginner's Team that helps beginners get involved <https://wiki.ubuntu.com/BeginnersTeam>.

EK: Is there anything you feel the project could do better with when it comes to new folks coming to the project?

JL: Sometimes it seems like there

is too much information but sometimes there's not enough. I remember trying to find out how to get involved in development, but running into lots of weird vocabulary like "MOTU," "packaging," "blueprints," and "triage." I come from a web background, so a lot of the desktop applications were not only in a new language but also a new way of developing. With that in mind, I think we can improve by remembering we're a very diverse community when communicating, creating documentation, and doing training. Some improvements are already in progress. For new folks, we should give more overviews so that people can find where to contribute, and be funnelled into those areas. We should define our technical terms, and we should make it easy to ask for assistance if a new person ever feels uncomfortable.

EK: What other things are you interested in outside of open-source and Ubuntu?

JL: Most of my free time is happily spent programming or designing. Recently, I took a few courses via the open-learning project Peer2Peer University (p2pu.org).

It's a great platform to help people learn from their peers. Going hand-in-hand with obtainable education for all, I serve as the Public Relations chair on the Board of Trustees of my city's library. Also, I'm a huge animal lover, vegetarian, and part of a leadership and speaking organization called Toastmasters (toastmasters.org).

QUICK TIP - No internet with 10.x (IPv6)

I had Ubuntu 9.04 as my main system, and as I prefer to do a fresh install and miss out the intermediate versions, I duly downloaded 10.04.1 LTS and burned it to a CD. The basic installation went very well, as had all my previous versions, but horror of horror, I found that I couldn't get onto the internet with web addresses. FTP appeared to be working and I could 'ping' any address with either the IP address or it's URL (so DNS was working), but run Firefox (FF) and request a site, no chance. I even upgraded FF, still no joy.

I messed about for ages with Network Connections. Nothing worked. Eventually I tried my Live CD of 9.04, and I immediately got access to the web.

Where do you start, it's so frustrating. I resorted to Ubuntu Forums and had a good search around. Nothing. It can't just be me surely. So I put up a new Post in 'Absolute Beginners Talk' and waited. Quite a number of people read the post but I didn't get a single reply. So, I tried again in 'Networking & Wireless'. Bingo! A response from wojox that resolved the problem which was with an IPv6 setting.

Solution: Within Firefox

1. Type about:config in the address bar, press Enter.
2. Find network.dns.disableIPv6 in the list.
3. Right-click > Toggle to select True (i.e. disable IPv6)
4. Restart Firefox and try again.

This solution is now mentioned on the Firefox tutorial site:-

<http://firefox-tutorials.blogspot.com/2010/05/common-issues-solutions.html> but you'd have to know that FF settings are the problem before you'd even consider hunting for them, and if you had a non-working v10.x that's not much help!

For me perseverance paid off, other less enthusiastic users may have given up.

I still don't understand why the setting for IPv6 has changed from True to False in the newer versions of FF or why it causes my system to fail in this way. There must be more to it as I still can't get a manual network connection to work, but at least I have a working copy of v10.04 now.

Laidback



News

• Uplink & Darwina now in USC

– UK Indie Developer, Introversion, has released two of its popular titles, Uplink and Darwina, into the Ubuntu Software Centre.

Volley Brawl has been recently released exclusively to Ubuntu by publisher Ohso. It is a simple Volley Ball game between two players trying to hit the ball over the net. To score a point, the ball must hit the floor on the opposing side.

There are two modes to Volley Brawl. Single Player allows you to play a quick match against the computer, with either a score limit or time limit, each with unlimited options. Multiplayer offers plenty of options including Local multiplayer - using either the same computer or other computers on the LAN. There is decent support for Online multiplayer, although

there seems to be a dearth of players at the moment. There are different teams you can select to play as, but they offer only cosmetic differences. Apart from those modes, there are no other features of Volley Brawl to mention. It is severely lacking in content, which does not help its replay value. Online/Local leader boards, Challenge Mode, or even Achievements, could help Volley Brawl in this area.

Volley Brawl is a very quick and easy game to pick up and starting playing. The controls are simple, and the gameplay has some nice fluidity. You will be scoring points and winning games very quickly. The CPU AI does a decent job, but still can be easy to beat. Varying difficulty levels would be nice to have, again to help the replay value.

The graphics and look of the game are crisp and pleasant for the style of the game, but nothing ground-breaking. However, the same cannot be said for the sound. The background music is awful,

and becomes annoying quickly - even though the sound effects in the game are not too bad. Volley Brawl ran very well and is certainly a good candidate for Netbook gaming. Supporting quick load times, it is certainly a game you can quickly start playing. Although Full Screen mode is not recommended on large displays, Volley Brawl seems to support low resolutions.

Volley Brawl is an easy game to starting playing, and you will be mastering it in no time. Although the game is priced at a low \$2.99, it still lacks replay value due to lack of game modes and easy AI. Volley Brawl does support many multiplayer options, even though the main selling point of this game is the online multiplayer, which lacks any players at the moment. With the promise of future content updates, the game is likely to get better but at the moment Volley Brawl is a fairly weak title. Volley Brawl can be purchased exclusively from the Ubuntu Software Center for \$2.99.

Score: 5/10

Good:

- Solid Game play
- Online multiplayer
- Promise of future content updates

Bad:

- Very simple game
- No real longevity
- Lacks game modes
- Sound is awful



Ed Hewitt, aka *chewit* (when playing games), is a keen PC gamer and sometimes enjoys console gaming. He is also co-host of the Full Circle Podcast!



Q&A

Compiled by Gord Campbell

If you have Ubuntu-related questions, email them to: questions@fullcirclemagazine.org, and Gord will answer them in a future issue. Please include as much information as you can about your problem.

Q I use and maintain 6-8 computers for myself, family, friends, and businesses. They are a mixture of Windows XP and 7, and Ubuntu's Lucid and Maverick. My personal desktop and laptop are dual-boot machines. There are straight-Ubuntu and straight-Windows machines in the mix. I have at least a half-dozen loose hard drives that are used for storage and transfer. My question is how to format the hard drives for maximum compatibility. It is imperative that they operate across OSes and file system boundaries as much as possible. There is a mix of documents, pictures, and music files. The drives range from 10GB to 1.5TB.

A The consensus appears to be, use FAT32 for drives smaller than 255 GB, including flash drives, and use NTFS for larger drives. However, if there are individual files larger than 2 GB, use NTFS.

Q When trying to access resources on my home network, I get "Failed to retrieve share list from server".

A Many people have found solutions in this message thread: <http://ubuntuforums.org/showthread.php?t=1169149>

Q Is there some way I can make an image from my machine? I want to use it in Virtualbox.

A (Thanks to *Rob_H* in the Ubuntu forums) Even if you can create the image, you may encounter problems with hardware drivers. VirtualBox emulates a small set of devices, which are probably not the same as the actual hardware you've got in your system.

My advice is to just do a fresh install in VirtualBox and copy over

the data you need. You'll probably save time in the long run.

Q When I run an ID Software game such as Quake 4, the sound is badly delayed.

A Use the ALSA wrapper for OSS, following the instructions found in this message: <http://ubuntuforums.org/showthread.php?t=1705760>

Q How can I copy a VCD?

A Run Sounds & Video/Brasero, and select "Disc copy."

Q I am using Ubuntu 10.04, and want to install Netgen, but it does not appear in the

repositories.

A It's in the Universe repository in 9.10 and 10.10, but somehow it is not in 10.04. However, the source file is available. You could file a bug, and it will probably then appear. <https://help.ubuntu.com/community/ReportingBugs>

Q I am trying to share a USB external hard drive from my Ubuntu install to my Windows systems (Win 7 and 1 Win XP.) The drive is formatted as NTFS. But I can't browse to it.

A Open Accessories/Terminal and enter this command:

```
gksudo gedit /etc/samba/smb.conf
```

Scroll down to this line:
`guest ok = yes`
and insert this line after it:
`force user = (your ubuntu user name)`

Q & A

Save the file, exit, and enter this command:

```
sudo service smbd restart
```

Q I have an Hitachi 3 TB HDD that I use in an icy dock external enclosure so I can backup all of my video files, etc. When I hook up the HDD to my computer it is detected as an 802 GB HDD.

A Some external enclosures have a 32-bit limit, which means they will not work properly with a hard drive larger than 2 TB. The drive had 2.5 TB of space when installed as an internal hard drive, partitioned and formatted with Gparted.

Q How can I play WebGL videos in Firefox 4 with an old video card?

A After installing Firefox 4, read the first message here: <http://ubuntuforums.org/showthread.php?t=1713184&highlight=libOSMesa.so>

Q I wonder if it is good practice to always download and install the available software from the update manager, or does it slow down the computer?

A I would go ahead and install all of the updates that are available in the Update Manager. These will contain bug fixes, and I don't see how continually updating is going to make your system slower. Ubuntu does not have a registry, which is the source of many slowdowns in Windows.

Q I can't play a DVD.

A Install the ubuntu-restricted-extras and libdvdrread4, and run:

```
sudo /usr/share/doc/libdvdrread4/install-css.sh.
```

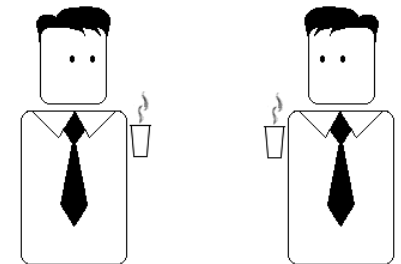
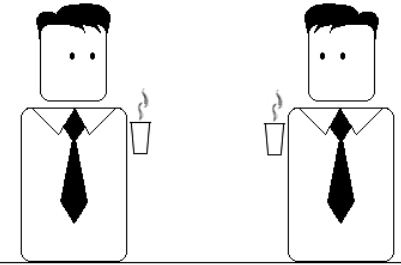
Q I use Xubuntu 10.04. When I run Settings > Appearance, most of the themes that are available in /usr/share/themes & ~/.themes are not seen.

A To change the window decoration for XFCE, you need to go to XFCE setting manager (it is called Setting Manager, and it will probably be in System > Settings under XFCE). From there, you can select Window Manager, and then change the decoration style.

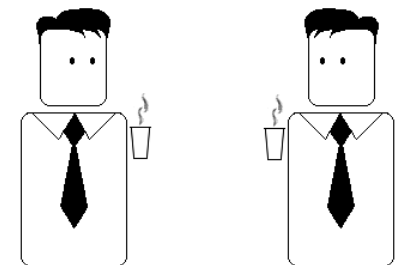
Q I just installed Ubuntu on my brand new system that I built. I put in my CD for my HD 5770 video card, but I can't get it to do anything.

A That disc contains Windows software, which is not useful for Ubuntu. Instead, you should run Administration > Additional Drivers.

How come that iPads are getting cheaper when even potatoes are getting more expensive?



Apparently potatoes are more useful.

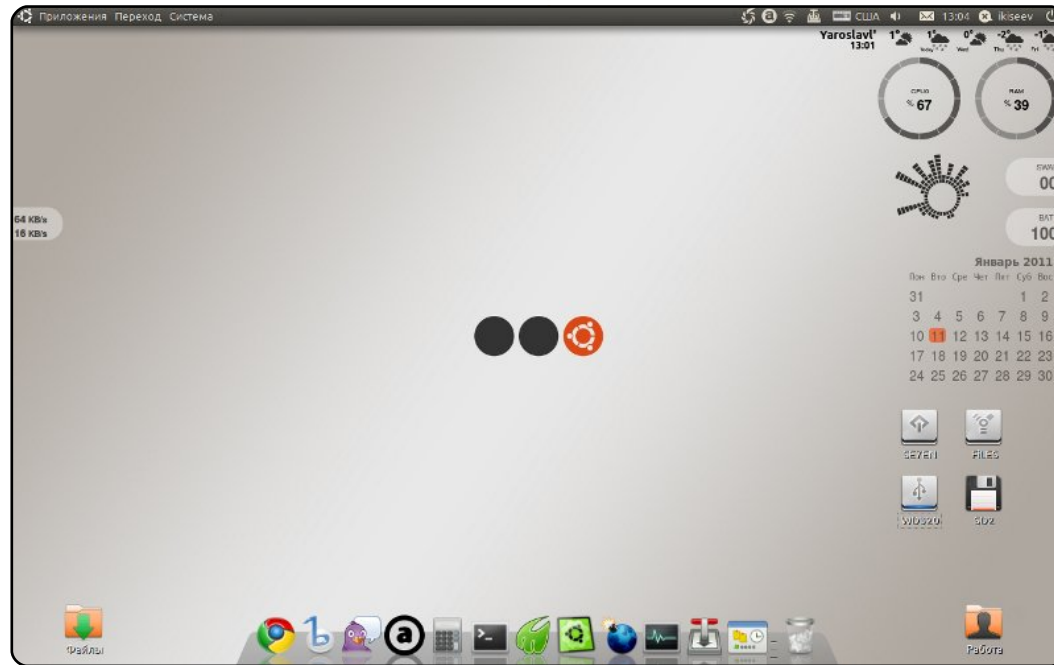


by Richard Redei



MY DESKTOP

Your chance to show the world your desktop or PC. Email your screenshots and photos to: misc@fullcirclemagazine.org and include a brief paragraph about your desktop, your PC's specs and any other interesting tidbits about your setup.



This is my desktop on a Dell INSPIRON 1501 laptop. For the design of my desktop. I used standard theme Ubuntu 10.10 (Ambiance), a set of widgets Screenlets, dock-panel Dockey, and wall-paper Ubuntu SpotLite3. I very much like the simplicity and logicity of my desktop.

Антон Киселев



I'm using screenlets, which I learned about from FCM. The background image for the panels I painted myself with GIMP. To save space on the panel, I put the package of window-picker-applet, which reflects the list of windows in a grid. I use a menu of Mint Linux. The icon theme is Tango-Blue-Materia.

PC configuration is:

4-core processor Intel (R) Core (TM) 2 Quad CPU

Q8200@2.33Ghz

4 GB RAM

Graphics card Nvidia GeForce gt 230 1.5 GB

Hard disk 360 GB

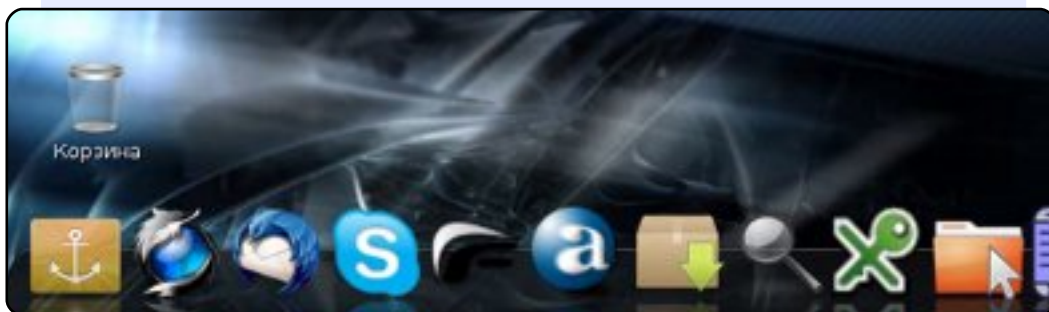
Ubuntu 10.04 LTS Lucid

Носов Артём



I'm a student of the Kiev Theological Seminary. My computer: Lenovo laptop G550 c, Pentium (R) Dual-Core CPU T4400@2.20GHz, RAM 2 GB, HDD 320 GB. The operating system is Ubuntu 10.04 Lucid. Two months before this, the OS was Windows7. I switched to Ubuntu for several reasons. I was searching for an inexpensive, or, better yet, free, alternative to Windows. What surprises me is that, in Ukraine, so few people use Linux.

Victor Potocki



I've been using Linux since 1999 and used various distributions such as Redhat (now Fedora), Mandrake (now Mandriva), Slackware, and OpenSUSE. I now use Ubuntu as my daily OS.

My Asus A42J laptop specifications:
Ubuntu 10.10 Maverick Meerkat
CPU Intel Core I5-460 2.53
2GB DDR3 RAM
NVIDIA Geforce 310 Cuda 1GB; and
500GB HD.

I use a custom theme with clear look on cotrol, evil_mac as window border, humanity for icons, and Think Linux as wallpaper. I use Conky to monitor current processes, Compiz for effects. I also use GNU/Linux in my office as a proxy server, Samba server, local web server, and other applications. I send greetings to all the people in the world from Indonesia.

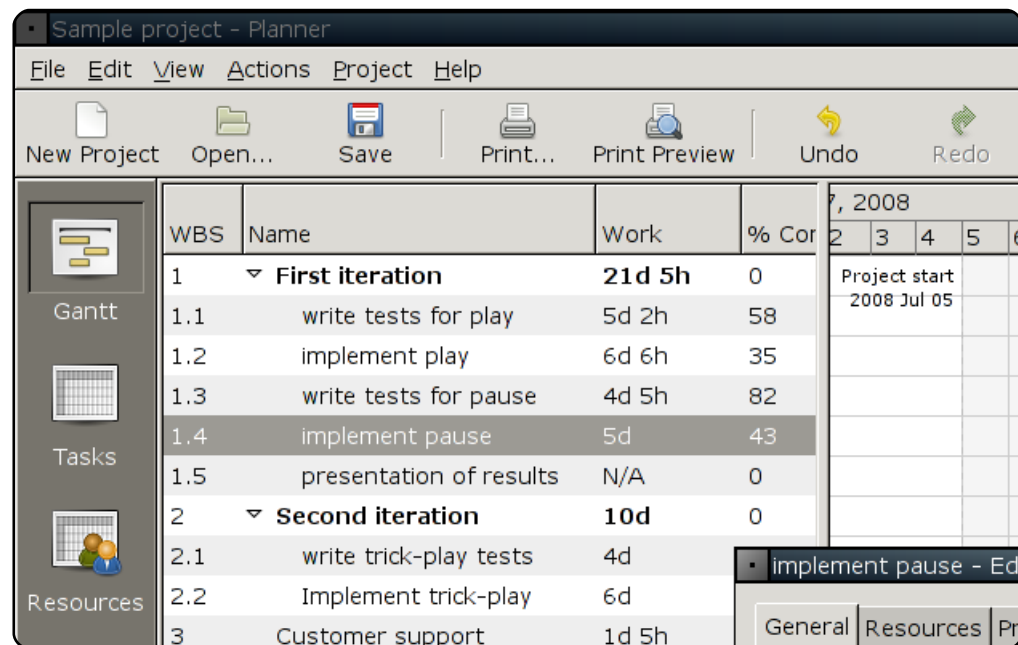
Muhammad Fahtur Rosi

Planner

Homepage: <http://live.gnome.org/Planner>

Planner is a powerful project manager written for Gnome users. You can create tasks and subtasks, create resources and assign them to various tasks (as well as include the cost of the resources), add milestones, create priorities, and input the percentage of completion. The views are also very handy - the default, a Gantt chart, displays a macro-level picture of the entire project, including the relationships between the various tasks and the resources assigned to each task. Finally, you can import Microsoft Project files, and export to both HTML and Planner.

To install Planner, use the **planner** package in the universe repository.

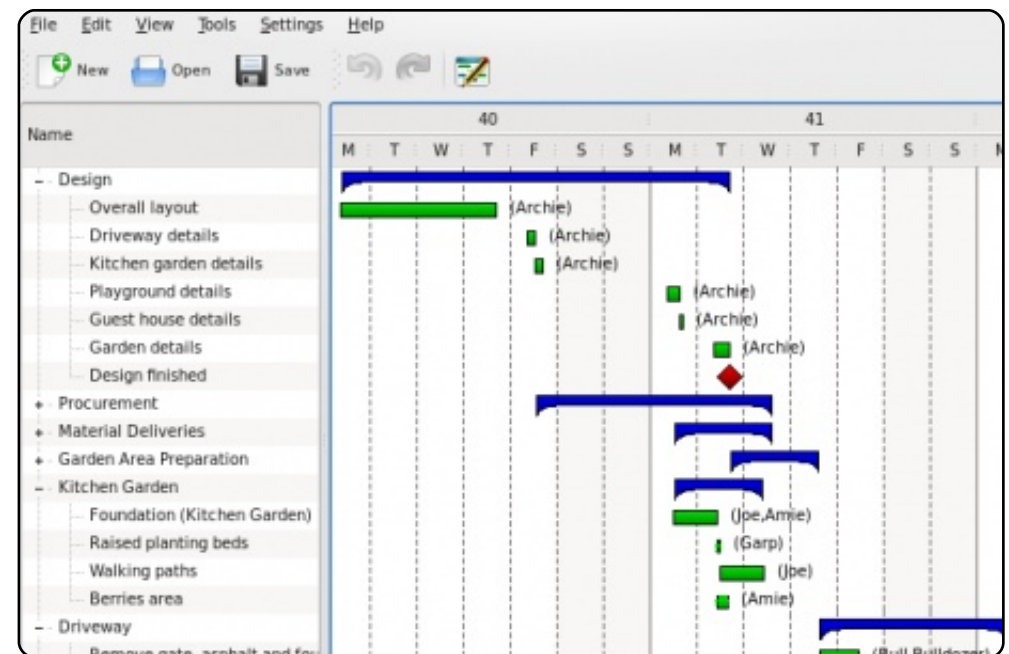


KPlato

Homepage: <http://www.koffice.org/kplato/>

If you're a KDE user, you might prefer KPlato. Its feature set is very similar to Planner's - you can set project lengths, allocate resources, and schedule and reschedule tasks. There's also a handy progress report that reports the earned value of a project. All this happens through a well-designed Gantt chart written in a beautiful Qt interface created to integrate well into the rest of KOffice.

To install KPlato, use the **kplato** package.

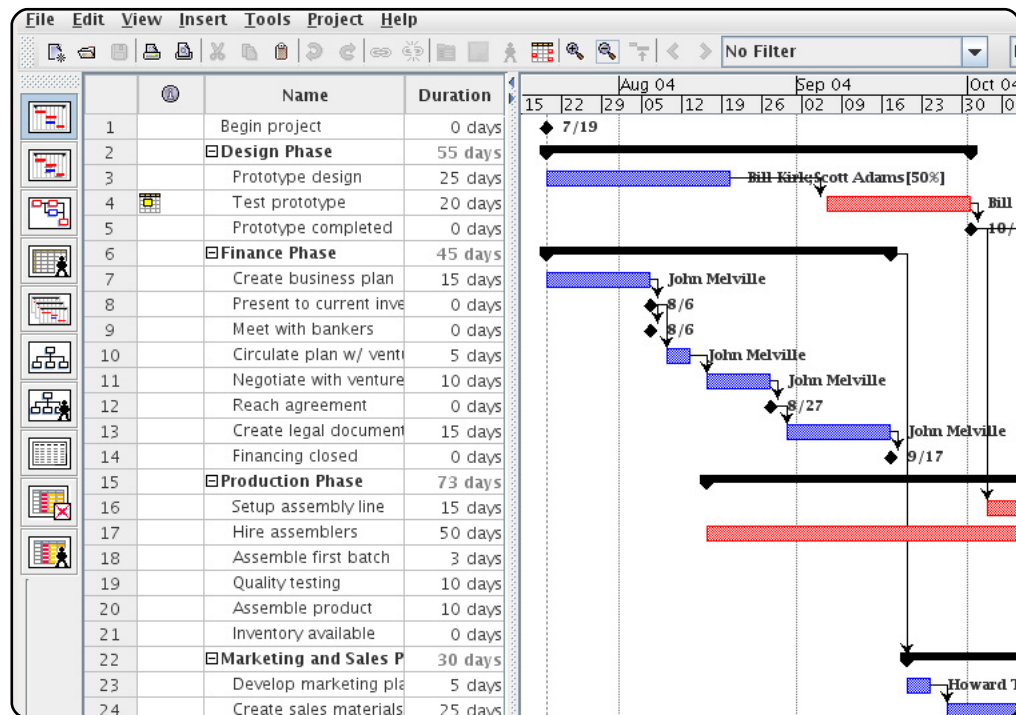


OpenProj

Homepage: <http://openproj.org/>

One of the grandfathers of open source project management, this free clone of Microsoft Project boasts over one million downloads, and for good reason. First, it's Java, a huge boon if your company uses more than one operating system. It also supports all the features everyone else has (resources, Gantt view, relationships between tasks, timelines, and reports). Finally, it supports reading both Microsoft Project and Planner files, as well as exporting to Project and PDF.

To install OpenProj, use the .deb binary provided at the download page.

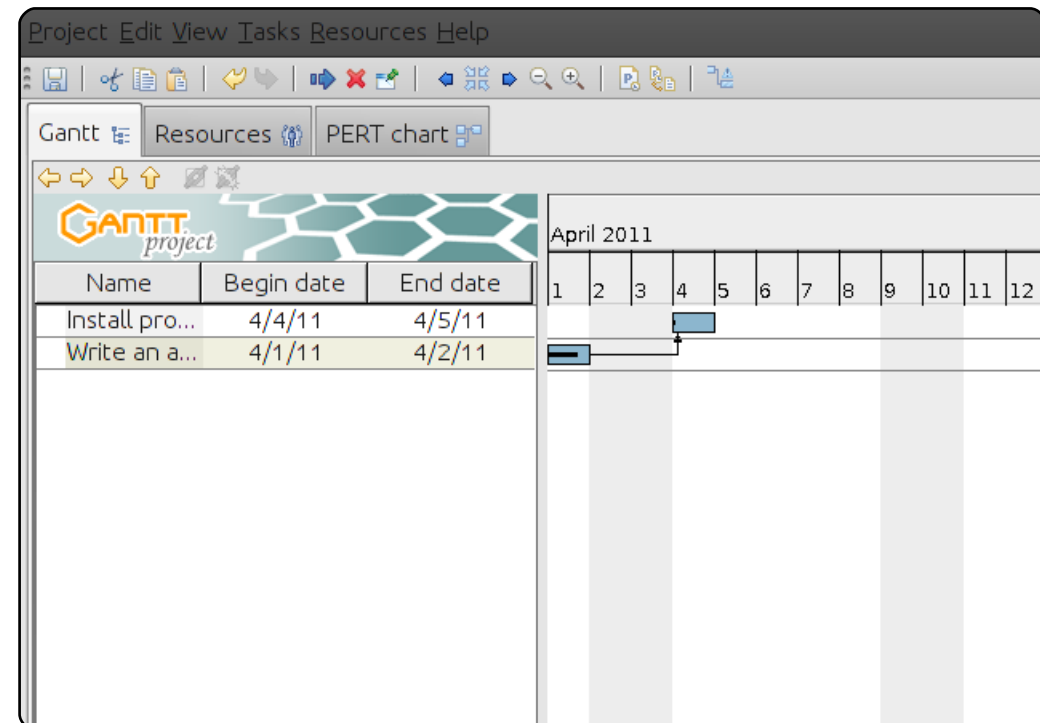


GanttProject

Homepage: <http://www.ganttproject.biz/>

If you're looking for a slightly simpler alternative, give GanttProject a try. It's another project manager built around a Gantt interface. Like the rest, it supports related tasks, progress, dates, milestones, priorities, and resources, but it's all done through a more simple interface. Another key distinguishing feature is its compatibility - it not only runs on all three major platforms, but it also provides a Java Web Start app, allowing users to run it from any computer with Java.

To install GanttProject, follow the instructions on the site's download page.

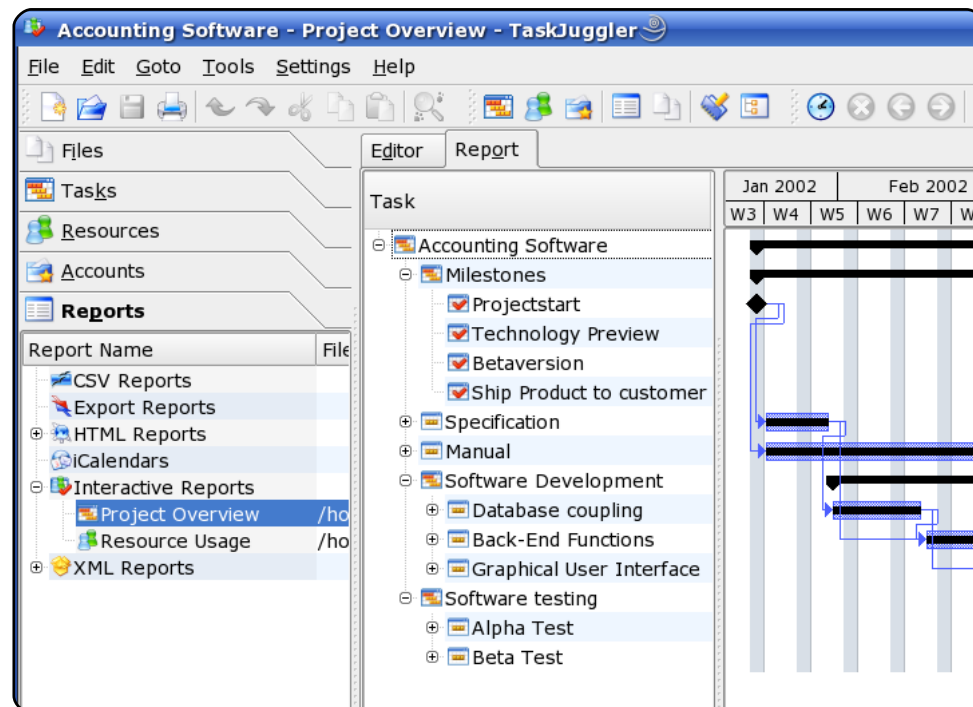


Taskjuggler

Homepage: <http://www.taskjuggler.org/>

If you're old school and prefer scripts to GUIs, give Taskjuggler a try. It's a powerful KDE-based application that turns scripts into usable data. Since it's not limited to what the customizable GUI offers, Taskjuggler supports extremely advanced features, from vacation days for your resources to accounts to calculate returns. It can also generate a wide variety of reports, including HTML tables, iCalendar files, and Gantt-style charts.

To install Taskjuggler, use the **taskjuggler** package in the universe repositories, or download it from the official homepage.



The Ubuntu UK podcast is presented by members of the United Kingdom's Ubuntu Linux community.

We aim is to provide current, topical information about, and for, Ubuntu Linux users the world over. We cover all aspects of Ubuntu Linux and Free Software, and appeal to everyone from the newest user to the oldest coder, from the command line to the latest GUI.

Because the show is produced by the Ubuntu UK community, the podcast is covered by the Ubuntu Code of Conduct and is therefore suitable for all ages.

<http://podcast.ubuntu-uk.org/>



ubuntu uk podcast

Download

Available in MP3/OGG format
in Miro or iTunes, or listen to it
directly on the site.



HOW TO CONTRIBUTE

We are always looking for new articles to include in Full Circle. For article guidelines, ideas, and for issue translation, please see our wiki:

<http://wiki.ubuntu.com/UbuntuMagazine>

Please email your articles to: articles@fullcirclemagazine.org

If you would like to submit **news**, email it to: news@fullcirclemagazine.org

Send your **comments** or Linux experiences to: letters@fullcirclemagazine.org

Hardware/software **reviews** should be sent to: reviews@fullcirclemagazine.org

Questions for Q&A should go to: questions@fullcirclemagazine.org

Desktop screens should be emailed to: misc@fullcirclemagazine.org

... or you can visit our **forum** via: www.fullcirclemagazine.org

FULL CIRCLE NEEDS YOU!

A magazine isn't a magazine without articles and Full Circle is no exception. We need your Opinions, Desktops and Stories. We also need Reviews (games, apps & hardware), How-To articles (on any K/X/Ubuntu subject) and any questions, or suggestions, you may have.

Send them to: articles@fullcirclemagazine.org

Full Circle Team



Editor - Ronnie Tucker

ronnie@fullcirclemagazine.org

Webmaster - Rob Kerfia

admin@fullcirclemagazine.org

Comms Mgr - Robert Clipsham

mrmonday@fullcirclemagazine.org

Podcast - Robin Catling

podcast@fullcirclemagazine.org

Editing & Proofreading

Mike Kennedy

David Haas

Gord Campbell

Robert Orsino

Our thanks go out to Canonical, the many translation teams around the world and to **Thorsten Wilms** for the current Full Circle logo.

Deadline for Issue #49:

Sunday 07th May 2011.

Release date for issue #49:

Friday 27th May 2011.

