

GeeksZine

Open Source is Fun

FREE AS IN FREEDOM

RICHARD STALLMAN'S
CRUSADE FOR FREE SOFTWARE



Issue : June 2010

Website : www.richnusgeeks.com

Publisher : RichNusGeeks Consulting

Table of Content

Just two words 3

Is that so? 4

Banishing a ghost 7

Surprising the master 11

Learning the hard way 12

The last rap 16

Just two words, *an editorial*

Hello everybody once again! I'm feeling very delighted while presenting another issue of the zine. The last issue of the zine recorded more than 800 views and a big thanks to all of you for the same. I got many responses regarding the first issue of the zine and those were very encouraging. Some people liked the zine and some people hated it for its plain and simple layout. In fact, the zine has got two new contributors this month. Please welcome Ankur Aggarwal and Anirudhh Dev on board. I hope together we shall be able to make the zine more beautiful, look and content wise, in the very near future.

You may have noticed, the front page of the zine has the image of the cover of a book on Richard Stallman, it's free online at <http://oreilly.com/openbook/freedom/>. This is just a small token of gratitude to the legend who infused the fun in the computing by starting the wave of free computing. This issue of zine contains some cool tricks with unexplored shell command and tool to extract more fun and profit from your GNU/Linux box. There are contributed articles on Richard Stallman's very own Emacs and IRC. We have also covered Zenity, a tool to create very nice Graphical User Interfaces (GUIs) with utmost easiness without any GUI programming.

This zine is from geeks, for the geeks and of the geeks. So if anyone of you think of any weird section and stuff to add to zine or want to provide feedbacks then please drop us a line. We would be more than indebted to add your section and stuff to the zine and work upon your feedbacks. So all the geekheads, we hope to see you every month with more improved GeeksZine and add more fun to this world with open source software.

Keep hacking

Ankur Kumar
RichNusGeeks Consulting

geekszine@yahoo.in
geekszine@gmail.com

Is that so?, *for the first timers to floss*

Wassup everyone? Now it is the time to explore more after making a transition to GNU/Linux from your closed source platform as demonstrated in the last month issue of the zine. This is my first article in the zine and I decided to write about something which helps in fulfilling one of the basic needs of all the geeks in the world. That is, talking to each other and obviously not in a way most of the other non-geek humans do. Enter the world of IRC, let me start with an introduction to it with brief history.

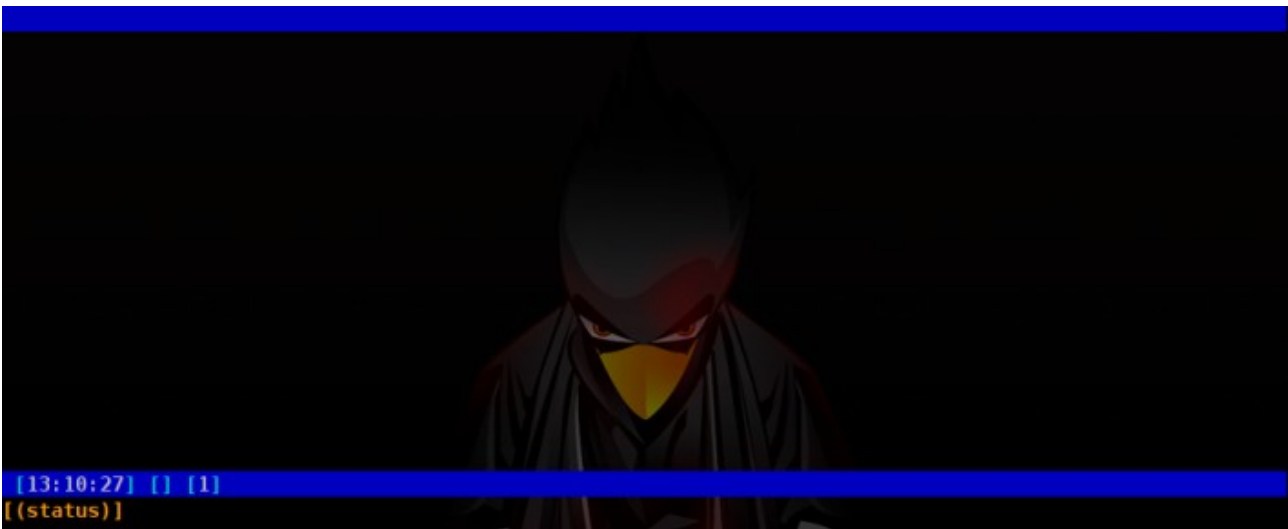
Internet Relay Chat (IRC) was first started in August of 1988, when Jarkko Oikarinen, an employee of the Department of Information Processing Science at the University of Oulu, had some spare time on his hands and tried to figure out means of communication based on BBS (Bulletin Board System). IRC really gained a lot of popularity and public exposure during the Gulf War. People would gather around their computers to hear the latest news updates, often before any World News Headlines had hit the airwaves. Family members would gather around their computers to be able to keep in contact with loved ones. Once IRC got rolling, it simply gathered momentum until it became the huge success as it is today.

There are various clients like xchat, konversation to connect to irc servers but my favourite one is irssi (<http://irssi.org>). It's so geeky to chat inside a terminal emulator in these days of yahoo messenger and gtalk and also it's being highly customizable through perl scripts. So lets start with the basics of irssi :

1. install irssi from your distribution's repository or from source (*sudo apt-get install irssi* for debian and its derivatives),
2. type *irssi* in the your favourite terminal emulator to start it,
3. you will see two blue bars and a window labeled *status* as shown in the screenshot below,
4. everything you will type will go into status window(area underneath the bottom blue bar).

Commands to get you started :

1. `/connect server_name` to connect to a server eg: `/connect irc.freenode.net`,
2. `/join #channel_name` to connect to a channel eg: `/join #ubuntu` (if a channel doesn't exist it will be created and you will be the op),
3. `/nick` name you want to use
4. `/part` to close a channel window,
5. `/quit` to close irssi and come back to shell prompt,
6. `/help` is the Swiss army knife of the irssi. `/help` alone will list all the help commands in window1 (Alt + 1). Type `/help + command` to know about the command.



The status bar should be showing active network, time, window, number, channel name now. If you want to connect to another server with a new network connection, type `/connect server_name`. Now you can switch between the two server windows using Ctrl + x. If you want to connect to another server on the same connection use command `/server server_name`. This will disconnect you from your current server and connect to the new one. To join another channel type `/join #channel_name`. After this you can switch between channels by using Alt + # (where # is 0-9) window number. The status window is by default window number 1. To switch between windows in irssi, use Alt + #, where # is 0-9. Windows start at number 1, and Alt + 0 goes to window number 10. You can use Alt + q through Alt + o for windows 11 to 19. It is possible to have more than 19 windows. Use Alt with the arrow keys to scroll through the windows. Use

pageUp, pageDown keys to scroll through text.

You could explore more by going through a huge list of irc commands and emotions you will come across (knowing emotions is fun :-)) provided at <http://www.ircbeginner.com> . So hopefully I got you started with irc and irssi and now you are on your way to explore more things in this world. Make new friends, give help and get help to do the things we never imagine. Hack and enjoy!

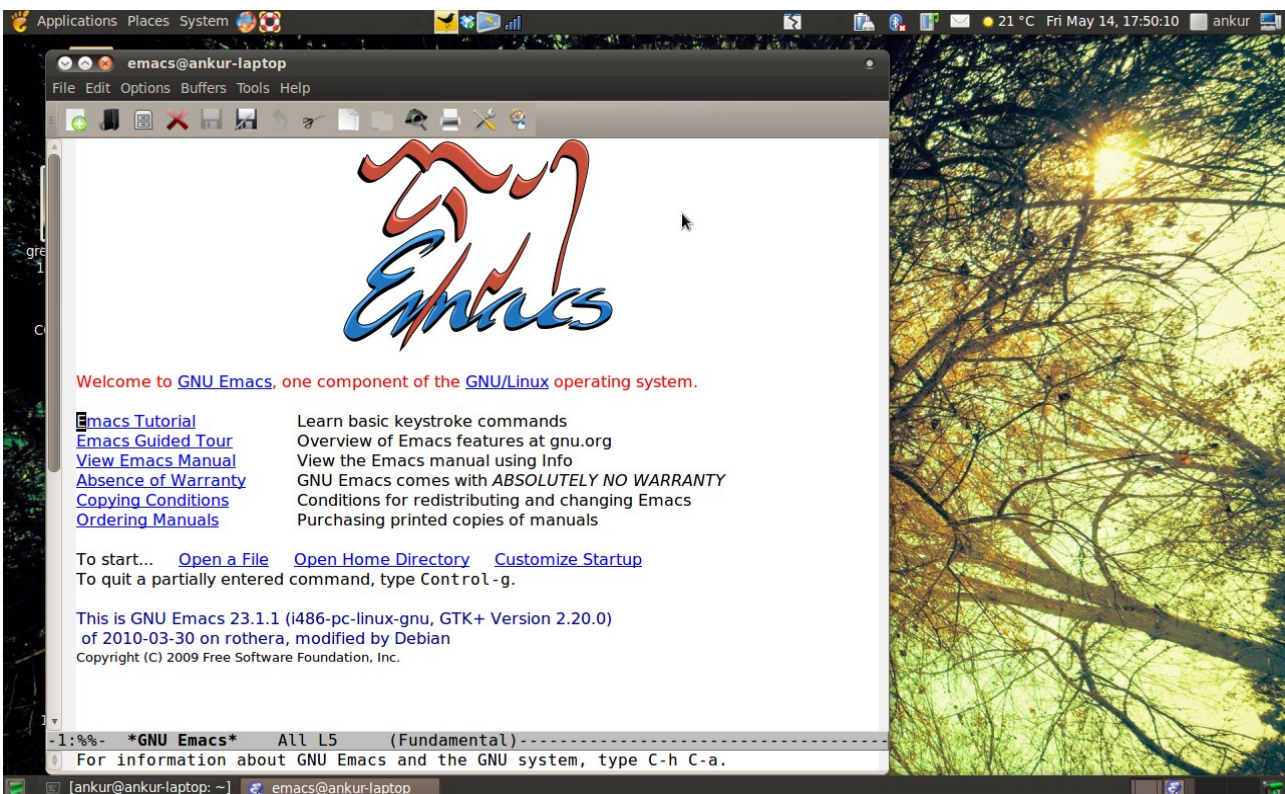
Anirudhh Dev
m33cfw@gmail.com

Banishing a ghost, *transition to floss*

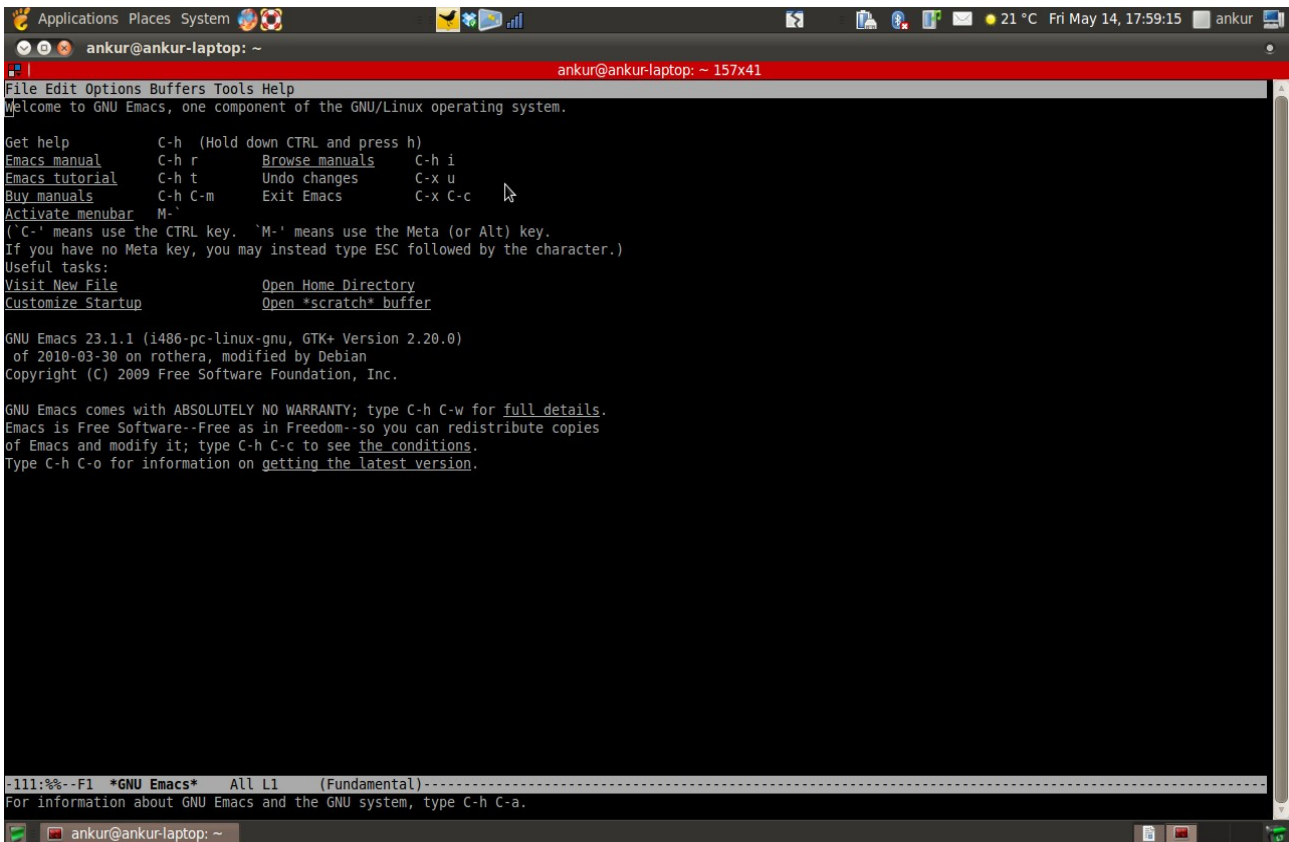
Text Editors are very important tool for everyone, especially for the programmers. So let's start playing with them and I am going to discuss the basics of Emacs text editor in this article.

Emacs was developed by Richard Stallman under the GNU project and he released his Emacs source code under a similar free copyleft software license called the EMACS Public License. C and Emacs Lisp programming languages were used to create this editor. This editor is also known as “The King Of Editors”. Believe me, the title matches exactly with its unique features. The Emacs environment doesn't seem intuitive at first glance and it doesn't work like other editors and word processors but learning Emacs is very easy. The latest version of Emacs is 23.2 (released on 8th May). You can check your version after installing Emacs by typing `emacs --version` in the terminal. Emacs is a cross platform software and I have tried it on windows as well. Emacs was designed by keeping in mind that users don't have to waste their time in clicking their mouse (start using Emacs and you will feel the same).

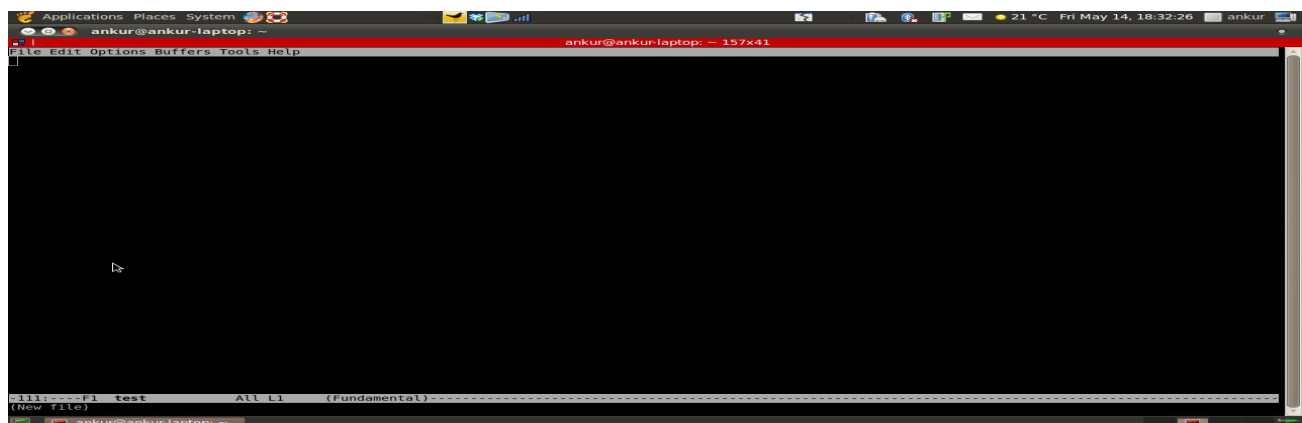
Emacs has two modes - GUI and CLI. Type `emacs` in the terminal to get the graphical mode. Below is the snapshot of the graphical mode :



Graphical mode can be useful if you are not much familiar with the command line terminal of GNU/Linux. Our main focus in this article is on the CLI mode. You can start the CLI mode by typing *emacs -nw* in the terminal. You should get something like this :



Start a new file in Emacs by typing *emacs -nw filename* in the terminal and you should see a blank buffer. Emacs works on buffer system. Any thing you write will be there on your buffer. When you save the buffer, it gets permanently written on your disk. Default Buffer in it is known as SCRATCH BUFFER. New file buffer should look something like the screenshot shown below :



At the top is the menu bar like in the GUI interfaces but you hardly use it in the CLI. Below that is your buffer area where you can write your contents and data. Mode line is below the buffer. You can call it as the status bar of the editor. It gives you important information about your Emacs session and about the current buffer displayed in the window above it, including whether your latest changes were saved to disk, what line number the cursor is at, how far into the buffer the bottom of the screen is (in percentage of the whole) and the current active features and settings of Emacs. Mini Buffer is below the modeline where Emacs displays the message relevant to any operations.

You don't need to use the mouse in Emacs as there are many key bindings (combination of keys) in this editor. Key bindings starts with two most important and common key that are Ctrl and Meta keys. Meta key in case is Alt key or Esc key. I have Already told you how to start the Emacs and now going to tell how to Exit it because it doesn't have a normal exit. To exit the emacs key combination used is Ctrl-x Ctrl-c. It means press Ctrl key then press x while still holding the Ctrl key and then press c while holding the Ctrl key. It exits the emacs command line and you will again be in the terminal. In the mini buffer you will see the progress of your command while you are typing it. I am listing some important commands below that will help you to start your work initially in the emacs :

<u>Key Combination</u>	<u>Function</u>
C-x C-s	Saves the content in the buffer
C-x C-f	Find a file, Start a file by giving location
C-x k	Kill buffer
C-x b	Switch between multiple buffer
C-p	Move the cursor to previous line
C-n	Move the cursor to the next line
C-f	Move the cursor one letter forward
C-b	Move the cursor one letter backward
M-f	Move the cursor one word forward
M-b	Move the cursor one word backward
C-a	Move to beginning of line
C-e	Move to end of line

M-a	Move to beginning of sentence
M-e	Move to end of sentence
C-_	Undo the change

C = Ctrl, M = Meta (Alt or Esc)

I hope this will be enough for you to start with the emacs. Believe me guys, it's worth giving a try! In the next issues of the zine we will take a look about different modes and functions used in emacs and their usages. Till then, explore more material on internet about it and have fun the geeks way with Emacs.

Ankur Aggarwal
coolankur2006@gmail.com

Surprising the master, *cool tricks*

- All the *IX systems have man command to show text terminal based help on various topics. We also want to try the things in a command terminal most of the time while reading the man documents. So usually we have to open a separate text terminal to try the things we read in the manual. This is very clumsy when you have a lot of windows opened in the graphical mode. The situation is more difficult if you work in console mode without any tiling window manager. Surprise! Man pages have a inbuilt solution to solve these problems. If you press ! key while browsing the man pages, you get a new subshell in the man terminal and you can try the things there, one line of command with every ! key press. If you press Enter key then you get back to the man page you were browsing. Try it and sharpen your working skills while reading the man pages ;-)
- Sometimes we want our GNU/Linux box to speak to us. This speaking ability is very useful to make announcements and alerts in case of many critical events in our machines or just for sheer fun. There exists a command line speech synthesizer known as *espeak* that comes preinstalled with Ubuntu. You could also install it on debian systems by typing the command *sudo apt-get install espeak* in a text console. The usage of *espeak* is pretty straight forward, just type *espeak "text to announce"* in a text console and listen your computer speaking the text. Some basic options to control *espeak* are :

-a <integer> sets the amplitude/loudness of the speech,
-f <filename> speaks the content of the given file,
-p <integer> sets the pitch/frequency of the speech,
-s <integer> sets the speed of the speech in words/minute.

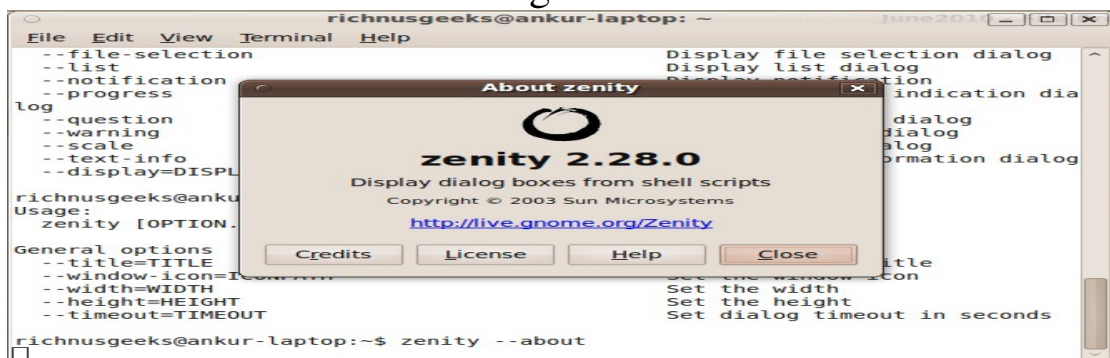
Espeak is a cross platform utility that runs on more than half a dozen of OS platforms. So go through its man page by typing *man espeak* in a text console, explore it more and make you machine speak.

Learning the hard way, *cool floss software tools*

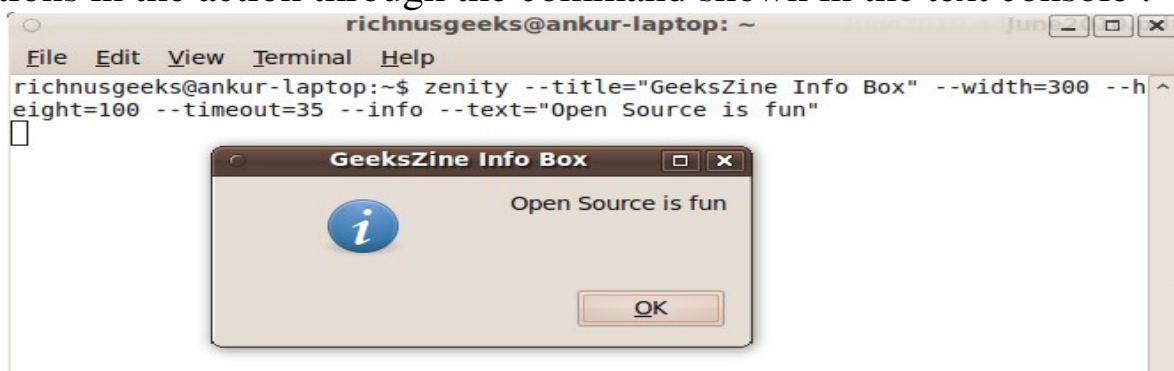
We all love GUIs as those are more user friendly and look more appealing. Adding a GUI to a new or existing application takes a lot of time and efforts as you have to choose a new GUI toolkit, learn about it and new programming concepts etc. Creating GUIs using a GUI toolkit approach is most of the time is overkill as we only need to get few inputs from the users, process those inputs through our program and finally reflecting the results. Ok! No more tears as we shall explore a foss tool known as Zenity that enables us to create nice instant GUIs without any GUI toolkit programming at all. We can create new GUI based applications through it or morph our existing command line based applications into graphical mode GUI applications.

Zenity is a GNOME desktop environment based tool created using Gtk+ GUI toolkit. It comes preinstalled with the latest versions of Ubuntu and you could install it on the debian systems using the command `sudo apt-get install zenity`. To install it from the source tarballs, please browse the webpage of zenity at <http://live.gnome.org/Zenity> for the necessary instructions. It mainly provides a simple set of various dialog box widgets that are capable of taking user inputs in various forms and reflecting the desired outputs. The current set of zenity contains *calendar*, *entry*, *error*, *info*, *file-selection*, *list*, *notification*, *progress*, *question*, *warning*, *scale* and *text-info* dialog boxes.

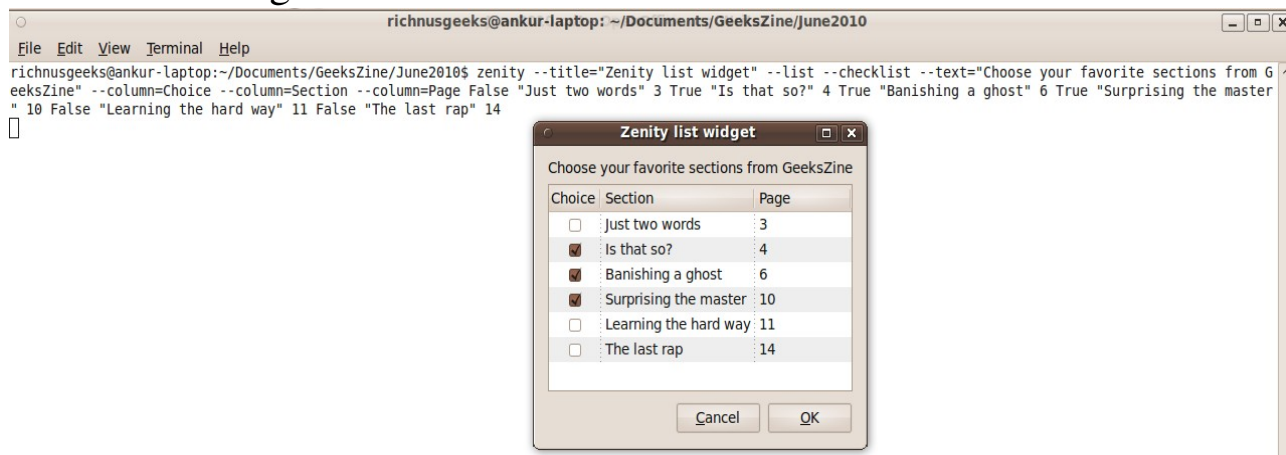
The usage pattern of zenity is `zenity [general options] dialog-widget [dialog-widget options]` where general options determine the *title*, *window-icon*, *width*, *height* and *timeout* for the dialog created. The dialog-widget options tweak the looks and behavior of the various dialog widgets created through zenity. Type `zenity --about` in a text console to see it quickly in the action. You should see an info dialog box as shown below :



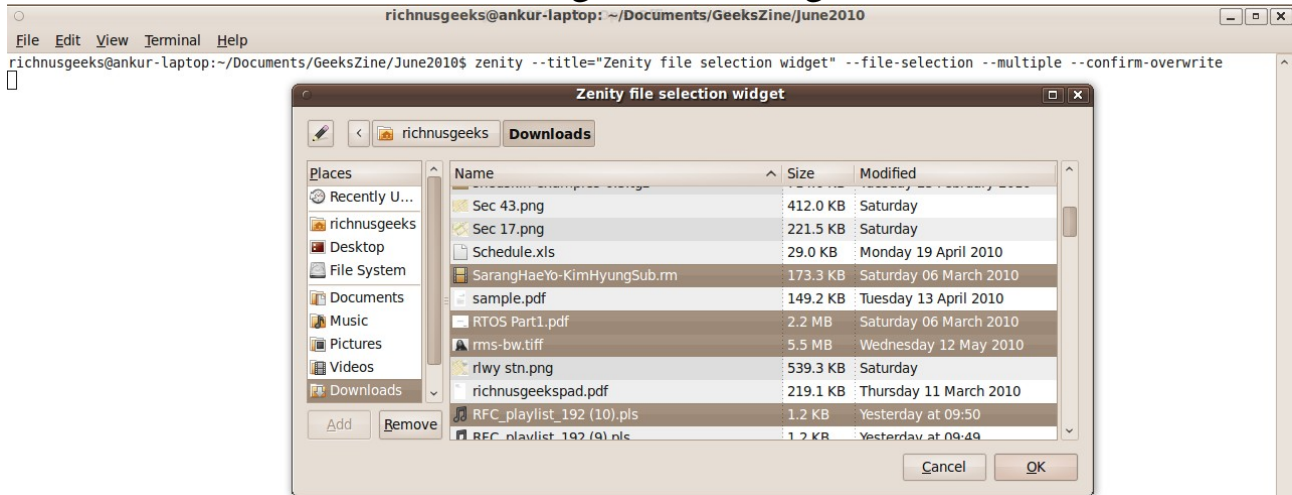
Now we explore more about general options and widgets provided by zenity. Zenity dialogue widgets have a default title but we can change it using `--title="title text"` general option. We can also change the default width and height through `--width=<width integer>` & `--height=<height integer>` general options. To make the dialog widgets to return automatically after a predefined amount of seconds use `--timeout=<seconds>` general option. Widgets take a `--text` option to display their text. See the below shown screenshot to see these general options in the action through the command shown in the text console :



Now it's the time to play with some zenity widgets to realize its full potential. You could open zenity man page in a text console and use the *!trick* mentioned in the surprising the master section (page 10) to tweak with various zenity commands. Zenity provides a powerful `--list` widget through which you can show a user selection dialog in many different configurations. This widget lets you show multiple column of items selectable through radio buttons or check boxes through `--column <title>`, `--radiolist` and `--checklist` widget options respectively. A *True* checks a radiobutton or checkbox and *False* does uncheck. See the screenshot below to learn the usage of it :



Zenity has a *--file-selection* widget that provides the functionalities of single or multiple files selection, only directory selection, file saving, existing file overwriting confirmation through *--multiple*, *--directory*, *--save* and *--confirm-overwrite* widget options respectively. See the screenshot below to learn the usage of this widget :



Now we play with another interesting but damn simple to use widget provided by zenity, *--notification*. This widget shows an icon in your desktop menubar and shows a text string on placing your mouse cursor over it. You can change the notification icon through *--window-icon=<path to a png , jpg image>* general option. Try command `zenity --notification --text="GeeksZine June 2010 issue is out"` in a text console and notice your menubar. Zenity has other widgets too but those are very easy to explore with the knowledge we have attained till now. So go through the zenity man page to cover it completely.

You can use zenity from shell scripts or any programming language like C, C++, Python, Ruby, Perl etc. that provides the functionality to create subprocesses. The return value of a zenity command is 0 if Ok button is pressed or the command is successful and 1 if Cancel is pressed. To add zenity based GUIs to your new or existing programs, the steps are :

1. Construct a zenity command with the required widget along with the necessary general and widget options,
2. Execute that command in a subprocess and collect the command return value and/or widget output. In shell scripts, the command could be executed as `r=$(command string)`. Then the zenity command return value is collected in ? shell variable and the widget

output in r.

3. Take decisions based on the values of command return and/or do processing on widget output.

A python function to create subprocesses is shown in the screenshot below. To execute any command through this function, construct a command string with all the arguments separated by a | (pipe character). The widget output is collected in the 1st element (0th index) and the return value of the command is collected in the 3rd element (2nd index) of the tuple returned by this python function. The screenshot shows a small python script that asks for a password through edit widget of zenity and notifies about the success or failure of the operation :

```
richnusgeeks@ankur-laptop: ~/Documents/GeeksZine/June2010
File Edit View Terminal Help
richnusgeeks@ankur-laptop:~/Documents/GeeksZine/June2010$ cat askpass_zenity.py
#!/usr/bin/env python2.6

from subprocess import Popen, PIPE, STDOUT

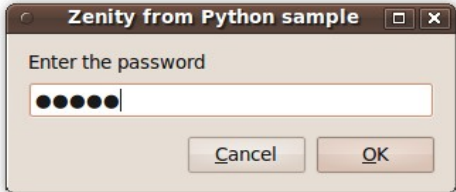
def execute(scommand):
    l = scommand.split('|')
    p = Popen(l, stdout=PIPE, stderr=STDOUT)
    toe = p.communicate()

    return (toe[0].strip(), toe[1], p.returncode)

szedit = 'zenity|--title=Zenity from Python sample|--entry|--text=Enter the password|--hide-text'
serror = 'zenity|--title=Zenity error|--error|--text=Incorrect password entered'
sinfo = 'zenity|--title=Zenity info|--info|--text=Correct password entered'

if '__main__' == __name__:
    tp = execute(szedit)
    if 0 == tp[2]:
        if 'geekszine' != tp[0]:
            execute(serror)
        else:
            execute(sinfo)

richnusgeeks@ankur-laptop:~/Documents/GeeksZine/June2010$ python askpass_zenity.py
```



You can see and realize how easily with few simple programming techniques we can learn and use zenity in just few minutes to create or add nice GUIs to our applications? In fact, many popular applications like Ubuntu Start (<https://launchpad.net/ubuntustart>) etc. are already using zenity for their GUI functionalities. So now it's the time to dirty your hands with zenity and create new GUI applications or decorate your old command line applications with it and to show the world your geek caliber.

The last rap, *an epilogue*

All the thoughts and the information in this zine is based upon the various freely and openly available resources on the internet and the personal experiences. So we don't guarantee the fitment of the opinions and the software mentioned for some particular purposes. Please try the information provided in the zine on your risk only and we are not responsible for any damage and loss caused by that.

We are putting this work in Public Domain and you are free to use and distribute the information anyway you like, with or without any attribution. If you like the work then we encourage you to share it more and more in various forms.

This entire document was produced with the FLOSS using OpenOffice.org 3.1 on Ubuntu 9.10 64-bit Desktop edition.