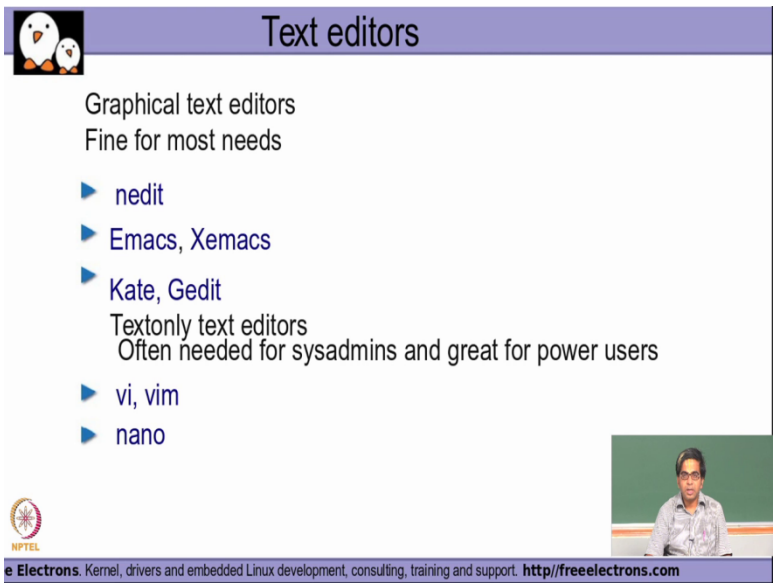
Information Security
Shri Vasan V S, Principal Consultant
Department of Computer Science and Engineering
Indian Institute of Technology Madras
Module 21
LINUX: Text Editor

This module we are going to be seeing the different type of editors that could be used on a Linux System.

So typically why would we need to use an editor so editor software or editor application is something that will be required for you to edit the data that you would possibly be programming or scripting? So later on in the course when we look at how we can actually develop shell scripts. You would need to use an editor for you to code the script appropriately save it in the editor and then after you save it in the editor and come out into the command prompt you can actually go ahead and execute the script.

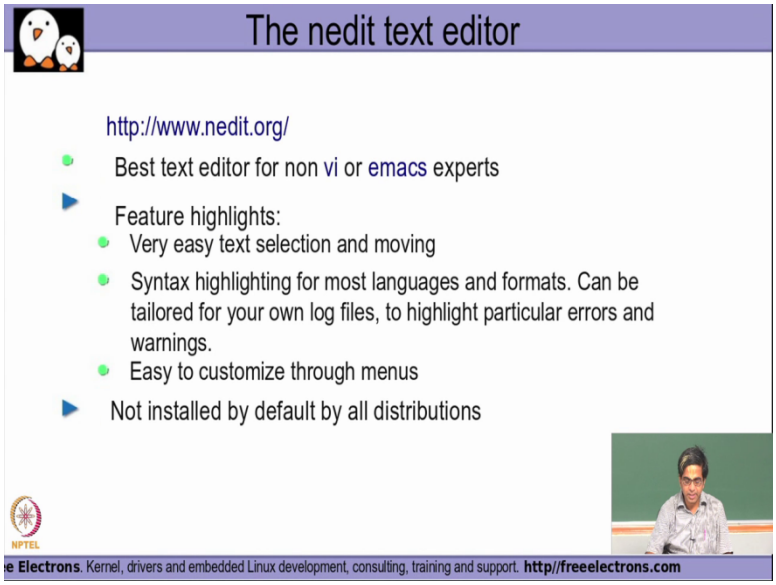
(Refer Slide Time: 01:00)



So there are different type of editors that are typically used on a Linux System. So we will take a very quick look at some of the most commonly used ones and there are lot of documentation also available for that which would be very handy for you to get an understanding of how to use editor. But predominantly editor is a application that is really required for you to do the first level of coding then for subsequent modifications like maybe adding more lines of code or deleting the existing lines of code changing the lines of code you would actually need to use the editor.

So you have different types of editor that is a available like the editor the gedit, Emac you have a Vi or vim you have Nano.

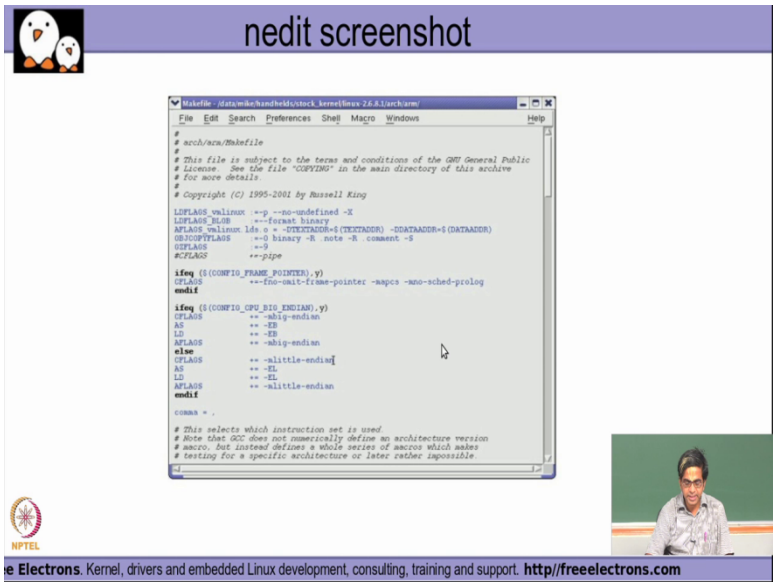
(Refer Slide Time: 01:50)



So you have different type editors that is actually available. So nedit text editor is something that is very very simple to make use of without too much of sophisticated features that is actually have it actually does the basic job very clearly so its it helps you to basically select the text easily and then sort of select them for moving it into a different location within the same file. It also tries to highlight the syntax is for most of the programming languages and format.

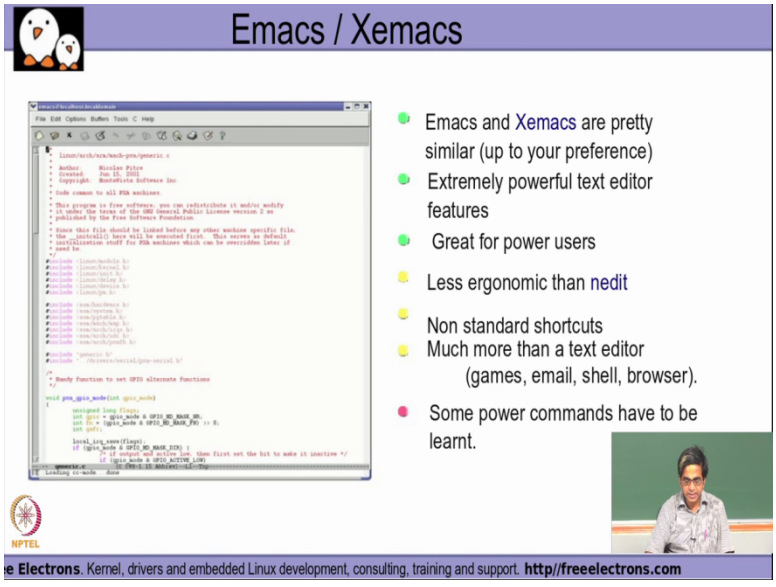
So when we know that we are expecting a particular colour or a particular format to be accepted in that particular programming language we would really be expecting the editor to be showing it in a particular colour and if we don't see that colour when you are editing the file we know that we are actually made some mistake which needs to be corrected before that particular file or script get successfully executed. So there are also different menus that are actually available with this and that also comes in handy to customize the behaviour to suit our needs.

(Refer Slide Time: 03:03)



So this is just a screen shot of how the nedit basically looks like when you edit the file. So you have a search option available here which you can make use of to search for different strings. So different type of customisation could be actually set here as part of enabling the preferences sub menu likewise you have pretty simple features that you would need to make use of an editor at a very very basic level.

(Refer Slide Time: 03:33)

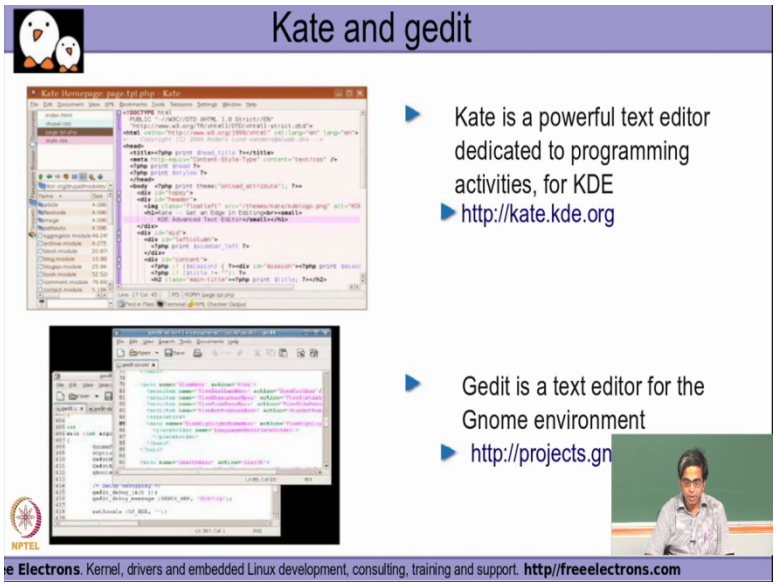


So the next comes emacs and the X version that is the version with x Windows support is called as an xEmacs these are very very powerful editor tools and you could actually do lot of

sophisticated editing functionalities with this tool it's just that it is little bit cryptic to make use of unless and until we get used to this editor over a period of time we will not be able to leverage the complete sophisticated set of functionalities that this editor actually provides us.

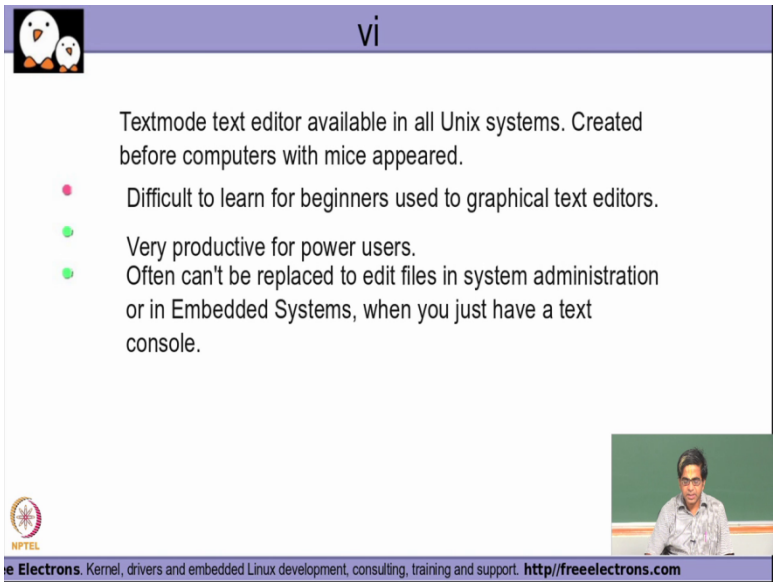
So there are lot of shortcut keys that are again possible which will help us to quickly do the different kind of Editing operations that we would need to do in the emacs editor.

(Refer Slide Time: 04:21)



So then you have a gedit with is very similar to our nedit and you also have something called as a kate which is again a very powerful text editor with all these editors have some minor differences here and there but essentially all of them help you to sort out editor file so editor file here would be creating the file in data in the file from scratch and also at the same time modifying the contents whatever was actually supposed to be present there and also whatever needs to be corrected in the existing file.

(Refer Slide Time: 05:00)

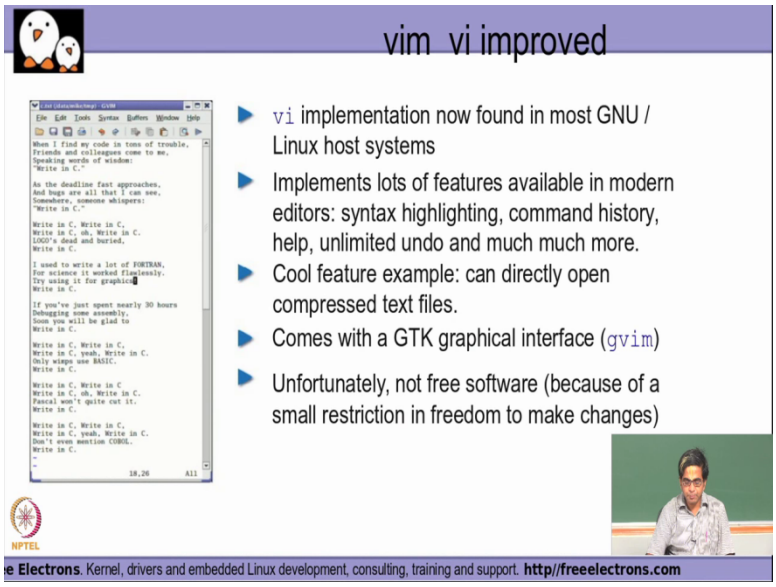


Then the most common editor that is sort of used by the maximum number of people today in the Unix world is what is called as a vi or vim that is basically the improved VI. So it is a text mode or the text editor available in all kinds of Linux System irrespective of the Unix flavour of the Unix distribution, sort of like sort of difficult for the beginners to learn like our Emacs editor it's little bit cryptic in nature with too many shortcut key combinations and everything but the main advantage with that is the kind of advanced features editing features that you could do.

For example trying to copy from copy set of lines from one file to another file at the same time trying to run different kind of Shell commands inside my editor trying to map of certain keys in the keyboard to doing very complex editing operations like these I have different ways by which we could do a lot of very powerful editing operations in this editor and for that simple reason this editor is actually very very productive for users who want to be extremely quick in editing and then completing the assigned job.

But the the flip side is that because of the so many different features it is actually available if you have to sort of leverage on the entire set of functionality and use its full power it will take time for a user to become an expert user on the VI , but once they become very comfortable with it users will find it extremely handy for getting their job done pretty quickly.

(Refer Slide Time: 06:52)

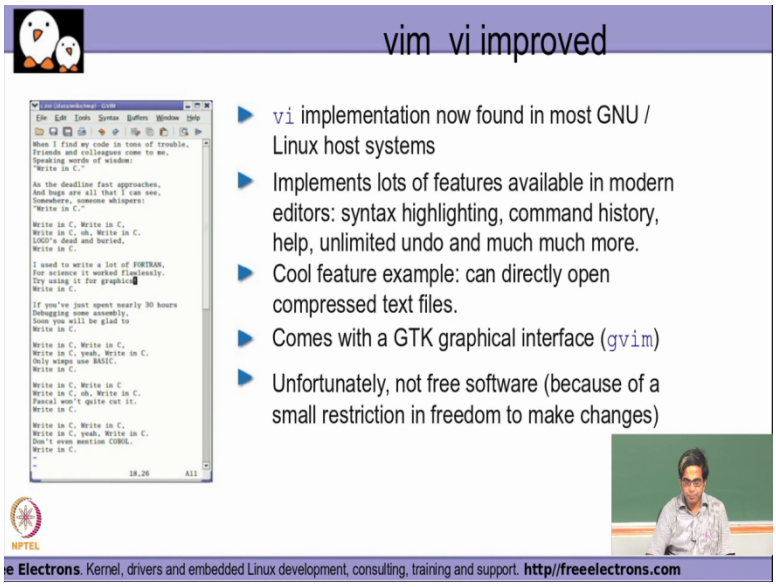


So we vim is basically an improved version of vi typically found in new gnu Linux source based systems it has a lot of functionality that could be done with the window in mechanism like for example the syntax highlighting having an unlimited number of undo operation.

So undo operation here is if I want to basically revoke whatever I had actually typed in the last time from the previous insertion mode I could actually do an undo and in the older versions of vi there were certain only a limited number of undo operations that I could basically revoke back where as with something like the vim which is improved version the vi I could actually keep going sort of undoing all the kind of changes that I had done till now.

So I could actually open up a compressed the text file also or directly when I use a vim because it detects what is the kind of compression that has been done on that file, or does the uncompression of it and then opens a file directly.

(Refer Slide Time: 07:56)



So there are very common set of commands that you would typically need to make use of and this set of basic commands actually available here if you need to have this is a sort of the cheat sheet the same documentation will also be available as a cheat sheet in this particular URL that is actually mentioned in this particular slide.

So with the vi I could actually I typically have two different modes what is called as an escape for a command mode in which every character that I type on my keyboard in the escape of the command mode is actually treated by the vi as a command for itself. And then the second mode is what is called as an insertion or append Mode where whatever whatever is actually typed in the keyboard in this particular is actually going in as part of the data of that file that is getting currently edited.

So vi basically operates in this two modes, so one mode which is Escape the command mode in which I basically give commands to the vi editor to make it behave in a particular format as per my requirement at that instant of time. Second mode is basically the insertion of the appending mode where everything that I type in this phone is actually treated by the vi to be made as part of the data of that particular file.

So when I am in the escape Mode I could basically give it instruction to insert characters in the current position insert a new line either after the current line in which the person is present or before the current line the person is present, I could basically yank, yank in vi terminology really means copying, so I could yank and paste the contents of whatever I had

copied into another location either in the same file or in a different File right? So like this we have different kind of options that are possible I could actually do a search and replace where and I will be able to quickly find out a particular string and sort of replace that particular string with another string pattern either from a particular line number to another particular line number inside the file or I could also go ahead into it globally across the entire file.

So like this I have a mechanism by which I will be able to do all the kind of basic editing operations as well as very complicated editing operations inside the VI editor. Likewise I could for example run a command inside my editor that command will actually be Run in a temporary shell that is created and then so I would be able to actually go ahead and replace the content error in a portion of the file or from the last line to the last line.

So likewise we will actually have a huge set of both basic operations as well as very complicated operations that I could do for the editing point of view by using the vi command as my preferred editor Thank you!