What Makes Linux Tick as a Programming Stronghold?

Linux is widely regarded as one of the best programming platforms available today. In this article, we explore the features that make the OS a programmer's paradise. Acquire the holy weapons of development that all Linux programmers should add to their arsenal of skills!

We all know Linux is an excellent platform for application development. In this article, let us take a formal look at why programmers swear by the programming tools and flexibility that the OS offers. We shall explore the languages, libraries and tools that are available on the Linux platform; and try to discover what exactly is making the, "It's all about money, Honey," development companies take note of this open source beauty.

Let's look at all the offerings Linux has in store for programmers. Many of these come pre-installed with most Linux distributions. Wow! You've got everything to build a space jet, and you wondered where to begin!

The goodies on offer—languages

The programming language that you could use may depend on project requirements, or your personal level of comfort. Some major languages that can be used on Linux are C, C++, Java, Ruby, Perl, Python, etc. Let's take a closer look at these. Some languages are object oriented, some are structural and some are scripting languages. Table 1 summarises this information you could use to make your choice.

Please note that most of these languages are both compiled and interpreted. In Table 1, the spotlight is on which method is more popular or generally used.

TABLE 1

Language	Execution	Object oriented	Difficulty level	Comments
C	Compiled	No	Medium	The mother of many programming languages, used in application development, system development, etc.
C++	Compiled	Yes	Medium	KDE is written in C++. The language is widely used in application development
Java	Compiled/ Interpreted	Yes	Medium	Using Java, you can develop cross platform applications. It is used in both application development and Web development.
PHP	Interpreted	Yes	Easy	An important server side scripting language, PHP offers highly extensible programming and is great to write database Web applications in.
Perl	Interpreted	Yes	Easy	Perl offers powerful capabilities of handling text and strings. It is used in applications and server side scripting. A powerful combo example of this is Webmin [www.webmin.com]. Perl is the administrator's best friend.
Python	Interpreted	Yes	Easy	Most Linux distributions use Python for their set up of GUI and other GUI administration tools. Most sysconfig commands are actually Python scripts.
Ruby	Interpreted	Yes	Easy	Ruby is an innovative, new scripting language. Its main advantages are simplicity, straight-forrwardness, extensibility and portability.

Thanks to a wide community out there on the Net, that are ready to help, you will have plenty of information and resources available at your disposal, regardless of which language you use. Also, you will find many sample applications that are full-fledged, heavy-duty role models, whose code is open to all. Besides the open source model, the code for these types of applications is also documented excellently. This means that you can get down to serious business, sooner than you'd imagined.

Libraries—making the developer's life easy

A library is a collection of standard programs and subroutines that are stored and available for immediate use. You would never want to know the inner workings of graphics hardware that's used to display some text on the screen. You would just use the printf() function in C, which is part of a standard library, and be done with it.

Just like programming languages, there are a plethora of libraries available for the Linux platform. Let us take a look at the most popular ones.

GTK: GTK (the GIMP tool kit) was initially designed to create The GIMP (The GNU Image Manipulation Program). Now, even on its own, it is an excellent platform for developing GUI applications. Initially, it supported only C, but now you can also use it to create powerful graphical applications in many languages like Perl, Python, C++, PHP, C#, etc. However, most GUI code written using GTK is in the C language.

A popular example of a GUI desktop environment, built using GTK, is the GNOME desktop environment. XCDROAST [www.xcdroast.org], the CD burning application that Linux enthusiasts swear by, has been written using GTK, along with many GNOME applications. GTK is easy to learn. Just brush up your language basics, get yourself all those libraries required, and dive right in. The screenshot of a GTK application is shown in Figure 1.

QT: This is yet another popular GUI tool kit. QT is available as free and open source software only when you develop free and open source applications yourself or develop applications that you don't distribute to others. QT is considered "commercial" when you develop and distribute your own commercially-licensed (non-GPL licensed) applications based on QT.

QT is much easier to use. Our favourite desktop environment, KDE, is written using QT. QT comes with its own GUI designer, similar to GLADE, but is much more powerful and easy to use. If you are new to GUI programming, and know C++, you'll definitely opt for QT.

SDL: SDL (Simple Direct Media Layer) is a cross-platform multimedia library that helps us to write 2D and 3D (through OpenGL) applications. SDL has been programmed using the good old C language. Bindings are also available for C++, Java, PHP, etc. SDL takes Linux's multimedia and gaming capabilities to the next level.

SDL is used in video playback software, games and emulators. The award winning game *Civilization: Call to Power* was ported to Linux using SDL. Another application using SDL is the game, Frozen Bubbles. My favourite SDL application is JESS [http://arquier.free.fr/]—the visualisation plug-in for XMMS (http://www.xmms.org). If you are a game programmer or come from a DirectX background, you'll love to develop applications using SDL.

ALSA: ALSA (Advanced Linux Sound Architecture) is basically an audio and MIDI functionality provider for the Linux operating system. ALSA comes with its own driver API and library API. If you are a game or multimedia

HOW TO GET STARTED WITH PROGRAMMING ON LINUX

The best part about most applications that run on Linux is that their source code is available for all kinds of experimentation and innovation. If you are a wannabe programmer, here is how you should approach your task. Let us assume that you are interested in the development of Web servers on Linux. The steps below will help you get started.

- You should first search for a Web server that could act as a role model—a server that can handle any load, is extensible, and is used by millions. A typical example would be Apache.
- You now need to brush up your programming fundamentals. C is an excellent language to start with. Also, familiarise yourself with compilation, debugging tools and interfaces—GCC and DDD, for example.
- When you are well versed with GNU development tools, download the latest source code for Apache, and try to delve deep into it. You might find it tough to digest the huge code-base at first.
- Now, visit the developer support site for the Apache Web server [http://httpd.apache.org/dev/], and look at the material available for helping developers — project guidelines, release guidelines, coding style guidelines, etc. This will give you a feel for industry standard coding requirements and conventions.

Get innovative and do something that you deem is enough to put your name in the credits file. The community will be thankful to you for contributing that special feature to an already popular Web server.

developer, or wish to write a sound output or special DSP plug-in on the Linux platform, then ALSA is the choice to make. Previously, there was an open sound system (OSS) available in Linux. But OSS was dropped with the release of the 2.6 series of the Linux kernel in order to make way for the more advanced ALSA. To maintain compatibility, ALSA supports OSS emulation.

PVM & MPI: Some people are path-breakers. For them Linux must offer PVM (Parallel Virtual Machine) and MPI (Message Passing Interface) libraries without fail. Both of these are used in clustering environments. Now these are also used in conjunction with the popular clustering software called Oscar. Both libraries enable developers to build programs, which can run on multiple nodes simultaneously. Oscar is actually a collection of PVM, MPI and other software, which will help you build advanced applications.

Programming tools

These tools include compilers, debuggers and everything that is concerned with development. IDEs or integrated developer environments should also come under the programming tools category, but we shall discuss this separately.

GCC: GCC (the GNU Compiler Collection) includes compilers for different languages like C, C++, Java, etc. GCC not only supports multiple languages, but even

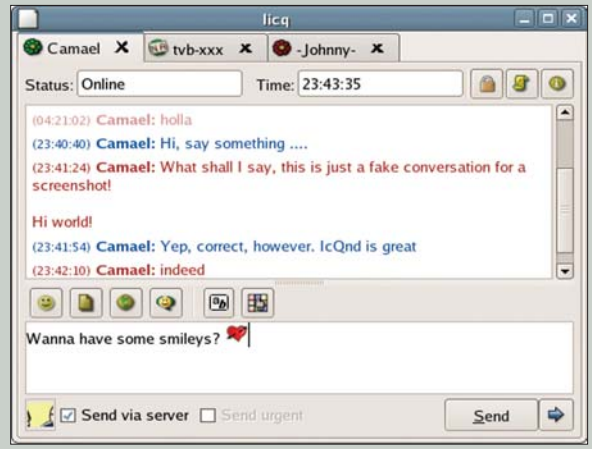

Figure 1: An example of a GTK application

Figure 2: An example of a QT application

Figure 3: An example of an SDL application

multiple targets and hosts at the same time. It also supports multiple operating platforms and cross-compilation technologies. GCC brings the following innovative features to Linux.

- GCC can work on multiple CPU architectures like x86, x64 (AMD64), SPARC, IA64, etc.
- You can use one architecture to develop programs for an entirely different architecture (say, compiling a 64-bit program on a 32-bit computer).

Figure 4: An example of a PVM and MPI application

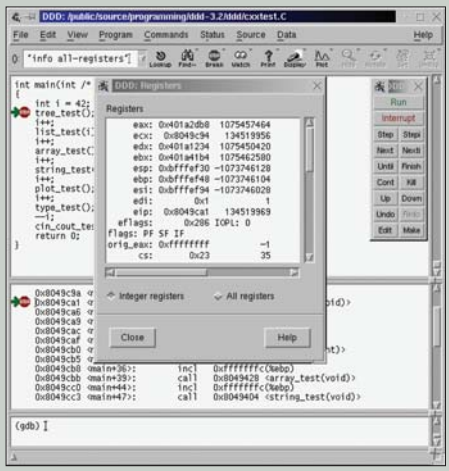

Figure 5: The GNU DDD in action

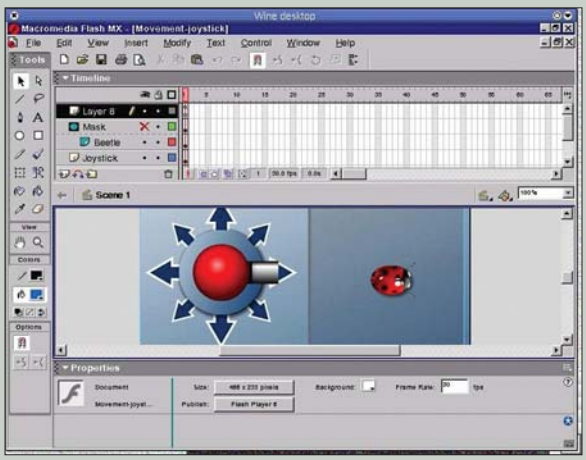

Figure 6: WINE running a popular Windows application

- Multiple software platforms are supported by GCC. There are ports of GCC that are also available for the Windows platform, which help you in maintaining source and binary level compatibility between Windows and Linux. You can even program a Windows application from within Linux.

GNU debuggers: To make bug hunting easy and effective, GNU has devised the GDB (GNU debugger), which is the core and default debugger on most Linux systems. However, GDB is a command-oriented debugger. Numerous GUI front-ends are available for this debugger, including one from GNU called the DDD (Data Display Debugger). An earlier article in LFY has already covered this front-end in detail. Many popular IDEs also provide integrated debugging. Figure 5 presents a screenshot.

WINE: WINE is my personal favourite. It is an amazing tool that helps you bridge the development gap between the Linux and Windows operating systems. WINE is an implementation of the Win32 API on the Linux platform. WINE's code is 100 per cent Microsoft free. WINE also provides a development library called WineLib, which is used to compile native Linux executables out of Windows source codes. Figure 6 shows WINE in action.

Version tracking, collaborative development and bug tracking

Any robust development platform must support collaborative development. The majority of large software projects involve more than one developer. There are teams working on projects, so we need tools that support multi-developer development and team development. Linux is, again, in a much better position than any other platform to support this need. We have the CVS (Concurrent Versioning System) and DCVS (Distributed CVS) for version tracking in Linux. Collabnet and Sourceforge.net are collaborative development systems, and Bugzilla is a very effective bug-tracking system.

CVS: CVS (Concurrent Versioning System) is a version control system similar to visual source safe (available with visual studio). It is a cross-platform solution. Basically, it is a command-line application, but when it is combined with CVS front-end tools like Cervisia (a KDE CVS front-end), WinCVS (a CVS GUI for Windows systems) or LinCVS, it becomes a powerful version control system. These days, almost all open source projects rely on CVS.

Bugzilla: Bugzilla is a bug-tracking system. This system enables individuals or teams of developers to keep track of outstanding bugs in projects. Bugzilla can also communicate with team members, can manage quality and carry on patch management. Bugzilla helps in improving software, decreasing downtime and in effective software bug removal. Bugzilla also features multiple authentication methods, time tracking, group support, localisation, full text searches and numerous other features.

Bugzilla is currently being used by many projects, including the popular Mozilla project [https://

bugzilla.mozilla.org], creators of the popular Firefox browser and Thunderbird mail client, the Linux kernel, KDE [http://bugs.kde.org], OpenOffice, Eclipse, GNOME and many other projects.

Sourceforge.net: Sourceforge.net is the world's largest open source software development site hosting more than 1,00,000 projects. Sourceforge.net is so involved with open source software development that many crucial and important software packages come out of its stable. Some major examples are DRI (Direct Rendering Infrastructure), the Linux USB project, the LIRC project and the ALSA (Advanced Linux Sound Architecture). Many of these projects are now part of the standard kernel source tree. Many other applications like Gaim, phpMyAdmin, amarok, etc are being developed on this site.

Integrated development environments

When we talk about Linux, we talk about innovation, possibilities and creativity. Linux has some of the world's best-integrated development environments. There are plenty of development environments available for both open source and commercial software. If you are preparing to mint money using your Linux environment, it is OK to purchase commercial software, too.

KDevelop: KDevelop is an excellent IDE for Linux—initially written for the development of QT/KDE applications—and now a full-fledged application development environment supporting 15 programming languages, including Ada, C, C++, Objective C, SQL, Fortran, Haskell, Java, PHP, Pascal, Perl, Python, Ruby, Bash and XUL. KDevelop has many great features. The latest version is KDevelop 3.2, which supports 15 languages, has an integrated debugger and many other great features.

Eclipse: Eclipse was one of the biggest contributions to the open source world. It was developed by IBM, and then donated to the Eclipse foundation [www.eclipse.org]. The Eclipse foundation is a not-for-profit organisation formed to support the Eclipse platform. Eclipse began as a Java IDE, but has come much farther. People now use Eclipse to build their own software. Numerous open source and commercial projects are taking to the Eclipse software development platform. Check out [http://www.eclipseplugincentral.com]. As an IDE, Eclipse has many great features that are normally found in only commercial grade frameworks. Eclipse provides multiple language support, a native look and feel to Java applications, and many other modern IDE features like UML modelling, Web application development support, Intellisense, code folding, code refactoring, etc

Linux—the OS for weird experiments!

Linux has always been an open platform; and never hides anything from users and wannabe hackers. This is precisely why Linux is not only about serious development, but also some highly unusual and experimental stuff. Some

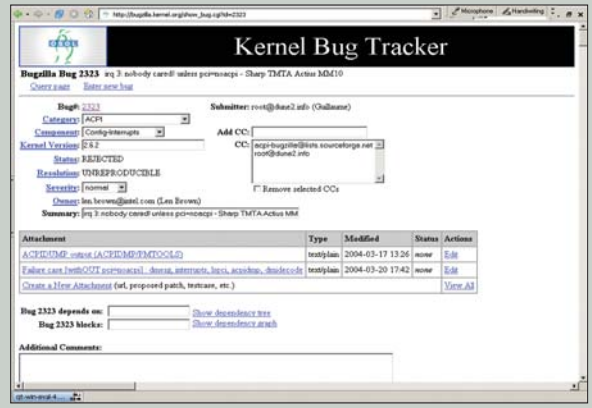

Figure 7: Bugzilla

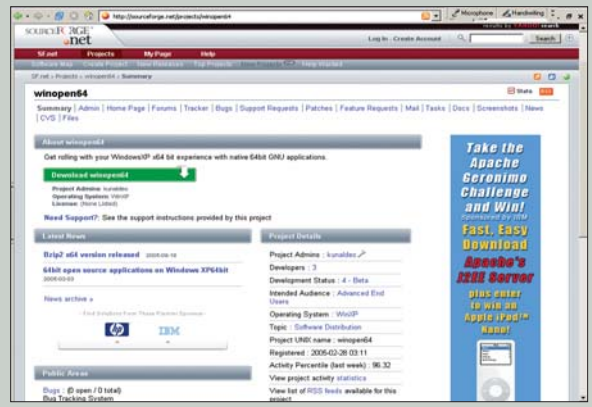

Figure 8: Sourceforge.net hosting an open source project

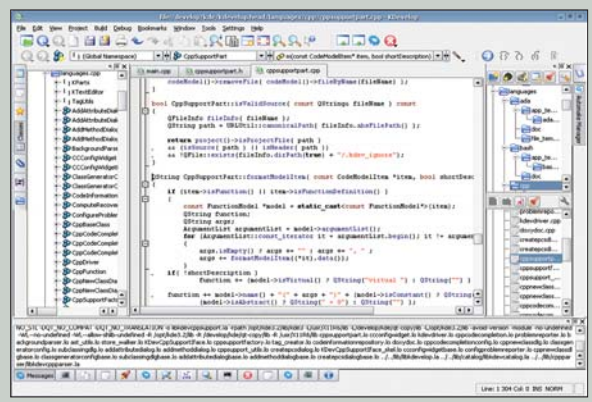

Figure 9: KDevelop in action

game freaks even turn their X-Box console to a Linux machine. Some people use Linux to even brew their coffee [http://www.tldp.org/HOWTO/Coffee.html]. You may be just a normal programmer on the Windows platform, but on Linux, you could be the next Linus Torvalds or Alan Cox. So get ready for your moment of glory! END

By: Kunal Deo. *The author is an open source developer and technical writer. He is employed with SPI Offshore Pvt Ltd.*