



L7  
L8  
L9  
L10  
L11  
L12  
end  
#Le nom du chien est  
#de la création de l'  
monChien = Chien.new  
chien.parle)



Boot to  
Gecko (B2G)

Dans ce numéro :

4 Dossier :  
8 pages sur l'Open Source.

Dans ce numéro :

16 Code(s) :  
Ruby le langage charmant.

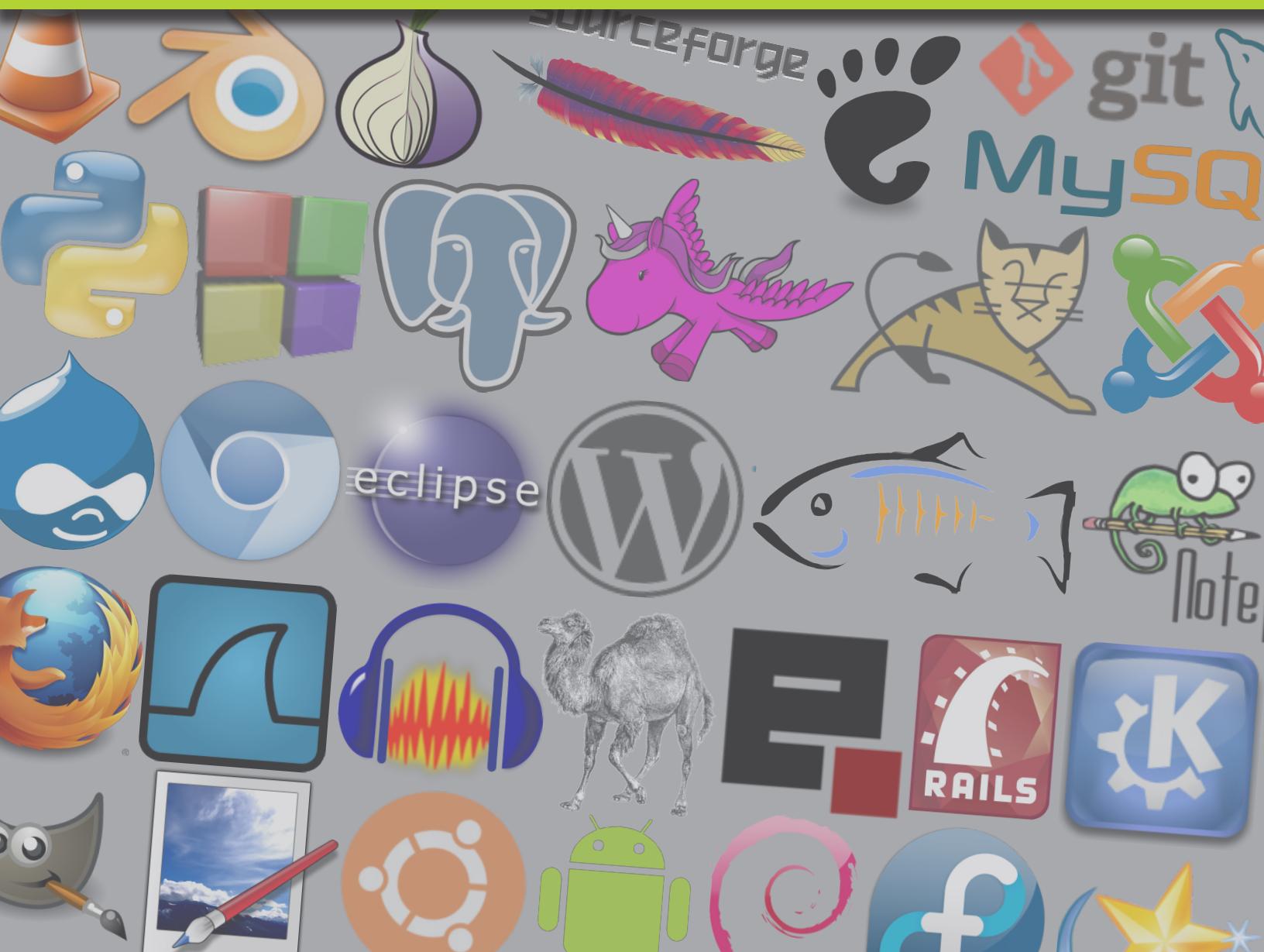
Dans ce numéro :

19 Découverte :  
B2G l'OS signé Mozilla.

# ENSI OPEN SOURCE magazine

Premier Magazine à l'ENSI

Premier magazine dédié à l'Open-Source en Tunisie



12 Dans ce numéro :  
Distributions Linux :  
Le miracle "Puppy Linux"

21 Dans ce numéro :  
Culture Générale :  
Architecture d'Android.

N°1-JUIN 2012

# Editorial



Malheureusement, il n'y a personne pour vous apprendre à aller au delà de Windows, à se demander qu'est-ce qu'il y a d'autre à part Microsoft...et Appel (pour quelques uns). L'informatique ne se limite pas à avoir le dernier Windows, manipuler Office Word ou Powerpoint, Installer DirectX, avoir le meilleur antivirus, craquer un jeu et surtout pas passer sa journée sur facebook !

Il est honteux de voir les gens reconnaître Mark Zukerberg, Bill Gates ou Steve Jobs et n'avoir jamais entendu parler de Dennis Ritchie (RIP), Tim Berners-Lee, Rasmus Lerdorf, Linus Torvalds et la liste est longue. Je trouve que c'est irrespectueux envers l'histoire et envers l'humanité même de ne pas avoir un minimum de reconnaissance envers ceux qui ont aidé à la vulgarisation du savoir numérique et continuent à le faire. Sans ces personnes là vous n'aurez pas tout le cyber-confort actuel. Vous pouvez toujours vous racheter en s'aidant de votre ami fidel Google pour connaître les logiciels et langages sur la couverture mais surtout leurs inventeurs !

Ce magazine n'a pas pour but de faire l'éloge de l'open source ni de faire la cour à Richard Stellman dont sa dernière visite est passée presque inaperçue dans le pays où le hashtag #OpenGov connaît depuis les élections un énorme succès : passivité ou hypocrisie ?!

Nous ne sommes pas Anonymous, personne ne l'est d'ailleurs, nous sommes des élèves ingénieurs à l'ENSI qui essayent de jouer le rôle de Morpheus dans la matrice où on vit pour qu'on puisse trouver un jour l'élu (ou les élus).

Bref, l'équipe vous a préparé dans ce premier numéro un ensemble d'articles espérant gagner votre intérêt et surtout vous motiver à contribuer au patrimoine open source.

Bonne lecture.

Lazâar Hamza

## Sommaire

Vous lisez dans ce premier numéro :



### Dossier

**8 pages pour tout savoir** sur les origines et le concept Open Source.

4



### Distributions Linux

**Puppy Linux**; le miracle des distributions Linux.

12



### Code(s)

**Tout le monde aime Ruby** : découvrez ce langage charmant.

16



### Découverte

**Boot To Gecko** : le plan de Mozilla pour dévorer Android ?!

19



### Culture Générale

**L'architecture d'Android** : Anatomie du Bugdroid !

21

ENSI Open-Source Magazine

#root : Lazâar Hamza

Designer: Lazâar Hamza  
@RealJohnTube

Remerciement à tous les contributeurs :

Abbassi Dhia  
Mezghani Mohamed Saïd  
Zrelly Wicem  
Ben Salem Hassen

Vive l'Open Source.

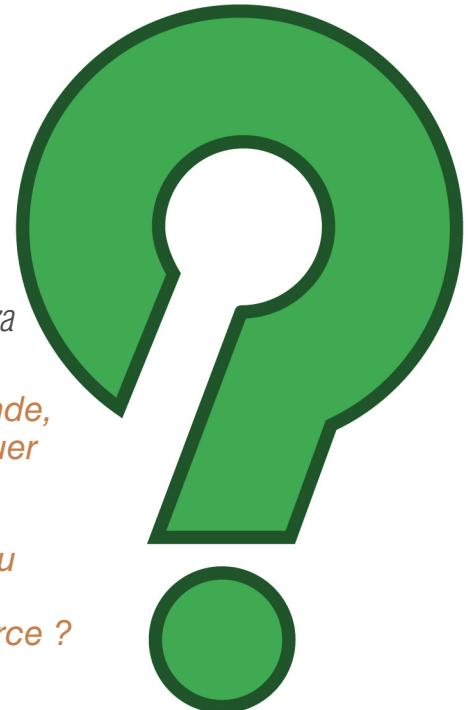
# L'Open Source : Quésaco ?

par Mezghani Mohamed Saïd et Lazâar Hamza

*L'Open source, représente une opportunité d'une coopération anonyme parfois conditionnée entre tous les développeurs du monde, un esprit de partage vaste voire illimité et une aptitude à faire évoluer le monde de l'informatique d'une façon participative. Donc c'est bien plus qu'une simple appellation molle et confuse, molle car on ne voit dans open source parfois que la transparence de "open" ou l'importance de "source" et confuse car on l'associe souvent par ignorance à ce qu'elle ne l'est pas. Alors qu'est-ce que l'open source ?*

Ces deux jolis mots venus de l'anglais, la langue des TIC par excellence sont en fait une marque déposée détenue par Bruce Perens, retenez bien ce nom car on va y revenir après. Une première traduction littérale et naïve du terme "open source" en français donnerait "source ouverte". Dans la version francophone du site officiel du projet Debian on a opté pour "informatique libre", ailleurs on peut trouver des termes comme "logiciel ouvert" ou "code source libre" et les divergences dans la dénomination ne finissent pas. Heureusement, Google translate tranchera : la notion est condamnée à l'anglicisme. Assez joué sur la forme passons au fond.

D'abord, pour commencer un logiciel open source n'est ni un Freeware, ni un Shareware. D'autre part, un code écrit en langage open source ou rendu publique ne fait pas de lui un code open source. Pour rendre une oeuvre open source, son propriétaire doit l'inclure sous une licence open source approuvée et listée par l'Open Source Initiative (OSI), l'organisme reconnu par la communauté open source pour gérer et contrôler les licences selon l'Open Source Definition (OSD). Le 30 Mars 2012 le projet Debian a rejoint l'OSI en tant qu'affilié et ceci vient renforcer le lien historique qui existait déjà entre ces deux organisations, explications dans ce qui suit.



Le "Debian Free Software Guidelines" (Principes ou Lignes de conduite du logiciel libre selon Debian) est un sorte de contrat social entre les concepteurs du système Debian et la communauté de l'informatique libre qui s'engage à produire un système totalement libre. C'est Bruce Perens qui s'est chargé de rédiger le premier brouillon de ce contrat qu'il modifia à la lumière des commentaires des développeurs Debian jusqu'à sa version 1.0 publiée le 05 juillet 1997. Un an après, Bruce Perens crée l'OSI avec Eric S.Raymond et il se contenta d'omettre toute référence à Debian dans le DFSG pour en faire la « définition de l'open source » un document adopté par l'OSI et qui fixe les critères qui déterminent si une licence mérite le label OSI approved ou pas, en d'autres termes si une licence respecte l'esprit de l'informatique libre ou pas. Le DFSG n'a subit qu'une seule modification le 26 avril 2004 alors que l'OSD en est à sa version 1.9.

Remontons un peu plus dans l'histoire jusqu'à un jour à l'MIT.

## L'histoire d'un hacker et d'une imprimante

Tout a commencé vers les débuts des années 80 quand Richard Stallman, à cette époque un hacker au département de recherche en intelligence artificielle du MIT, a eu des problèmes de

pilotes avec son imprimante Xerox et qu'il voulait accéder au code source du pilote pour réglé ce problème. Richard s'est conforté à un refus total des employés de l'entreprise de lui fournir le code source du pilote. Convaincu par la gravité de cette situation, Stallman se lança dans sa propre aventure pour combattre ce qu'on appelle aujourd'hui les logiciels propriétaires. C'est qu'en 1983 qu'il annonça le développement du projet GNU et la création de la free software foundation (FSF). Et c'est ainsi qu'il a donné naissance à un nouveau concept dans le monde de l'informatique, un concept qu'il baptisa free software ou en français logiciel libre (et non logiciel gratuit).

### GNU et FSF c'est quoi au juste?

Le nom du projet GNU est un acronyme récuratif « Gnu is Not Unix ». C'est un projet visant le développement d'un ensemble de logiciels libres qui constitue un système d'exploitation libre et qui, comme son nom l'indique, reprend le concept et le fonctionnement d'UNIX. Aujourd'hui parmi la liste des paquets GNU, on retrouve la collection de compilateurs GNU (GCC), le débogueur GDB, les outils binaires GNU, le shell Bash, la bibliothèque C GNU, les outils de base GNU, l'assembleur GNU et le noyau Hurd. Pour la protection du projet GNU il fallait avoir une structure logistique, légal et financière. C'est là qu'est apparue la Free Software Foundation (FSF) qui est une organisation à but non lucratif dont la mission est la promotion du logiciel libre et la défense de ses utilisateurs.

Depuis le départ, Stallman cherchait à protéger les libertés des utilisateurs de logiciel dans le domaine numérique. En effet, toute la philosophie des logiciels libres provient de l'application des libertés du monde réel au monde numérique. Stallman définit ainsi les 4 quatres libertés essentielle qu'un logiciel libre est censé protéger. En premier lieu, la liberté de faire tourner le programme comme vous le souhaitez. En second lieu, la liberté d'étudier le code source et de le changer. Ensuite, la liberté de partager le logiciel tel qu'il est. Et finalement, la liberté de partager

le logiciel que vous avez modifié et de distribuer votre propre version du logiciel. C'est dans cette optique qu'est apparue la licence GPL et le fameux copyleft. Mais il faut reconnaître Stallman défend la liberté des utilisateurs sur tout le domaine numérique. C'est plus qu'une histoire de code source, c'est une histoire d'éthique. A titre d'exemple vous pouvez voir les partisans du libre dénoncer la censure, la surveillance, les formats de données restreints et défendre le partage des données sur le net. Tout ça constitue la liberté indispensable d'une personne dans le cyberspace.



*Richard Stallman*

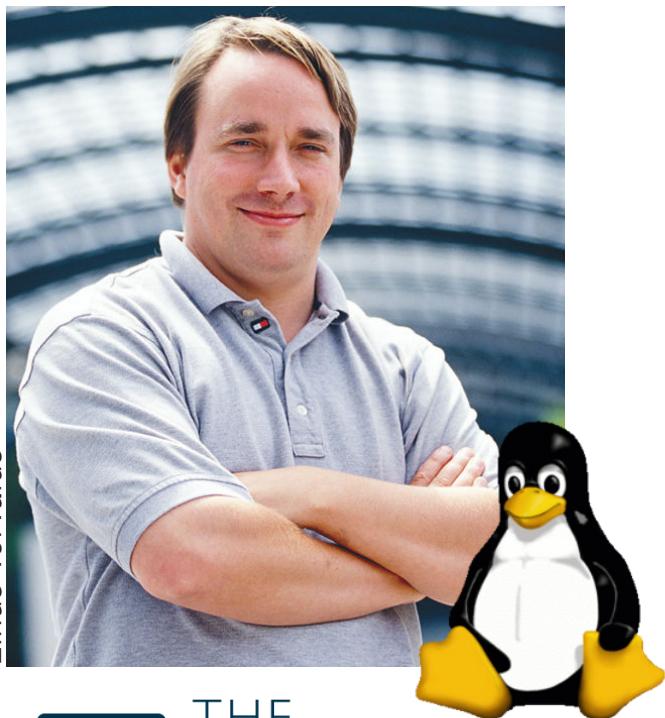


*Mascotte GNU*

## Et cerise sur le gâteau

Au début des années 1990, le projet GNU s'est lancé dans le développement d'un noyau de système d'exploitation intitulé Hurd qui va permettre à l'utilisateur d'exploiter au mieux les composants matériels de la machine. Mais, son développement rencontrait beaucoup de difficultés et ce noyau ne fonctionnait pas comme prévu. C'est là que, de l'autre côté du globe et plus précisément au Finlande, Linus Torvalds donna naissance au noyau Linux. Ce jeune étudiant a trouvé son inspiration dans le système d'exploitation Minix développé par Andrew Tanenbaum pour offrir au projet GNU le noyau attendu. L'une des raisons de la réussite de Linux est son rythme de développement. En effet, depuis la publication de son code source sur Usenet, ce noyau a suscité l'intérêt de beaucoup des utilisateurs de Minix. Plus tard des milliers de programmeurs bénévoles ont participé à ce projet. Aujourd'hui les principaux contributeurs sont un ensemble d'entreprises, souvent concurrentes, comme Red Hat, Novell, IBM ou Intel et plus récemment Microsoft.

Linus Torvalds



# THE CATHEDRAL & THE BAZAAR

MUSINGS ON LINUX AND OPEN SOURCE  
BY AN ACCIDENTAL REVOLUTIONARY



**ERIC S. RAYMOND**



Le succès foudroyant de Linux a fait de Linus l'un des programmeurs les plus populaires dans la communauté d'utilisateur de logiciel libre. Il est devenu même un symbole de ce mouvement. Malgré ce statut, Linus se moquait royalement de la liberté de Linux, tout ce qui compte pour lui c'était ce qui découle de cette liberté, à savoir une communauté de développeurs bénévoles qui produit le meilleur logiciel possible. Et c'est ce qui constitue le point de désaccord majeur avec Richard Stallman et sa philosophie des logiciels libres. C'est ainsi qu'en 1998 Eric Raymond, qui était fasciné par la manière avec laquelle le système GNU/Linux évoluait et en même temps par son incroyable efficacité, sortit un livre intitulé : « La cathédrale et le bazar ».

A l'occasion de ce livre Raymond oppose le processus de développement de Linux en le comparant à un bazar au processus de développement des logiciels propriétaires en les comparant à la structure d'une cathédrale. Par la même occasion il introduit un nouveau concept : les logiciels open source. Le choix de cette appellation s'est fait de telle sorte qu'on a pu lever l'ambiguïté introduite par free dans free software et ainsi former une nouvelle classe de logiciels. Et de la même façon que dans le cas du mouvement de logiciel libre, il y a eu la création de l'Open Source Initiative (OSI), organisation dévouée à la promotion des logiciels open source, et le postulat de l'Open Source Définition qui définit la condition d'utilisation des logiciels open source. Et ce cadre convenait parfaitement à Linus Torvalds pour le développement de son noyau Linux.



Bruce Perens

# Open...



# Du libre à l'open source :

Code éthique ou modèle économique ?

**O**n confond souvent l'appellation logiciel libre et logiciel open source, pourtant ce sont deux sortes de logiciels différents de point de vue purement éthique. En lisant l'open source definition et la free software definition, on remarque une identité quasi-totale dans les droits donnés aux utilisateurs. Mais la différence fondamentale entre ces deux types de logiciel reste le critère d'évaluation du logiciel.

## Open Source Definition :

### Introduction :

Open source ne signifie pas seulement l'accès au code source. Les conditions de distribution de logiciels open-source doivent se conformer aux critères suivants:

1. Redistribution libre.
2. Code source.
3. Les œuvres dérivées.
4. Intégrité du code source de l'auteur.
5. Pas de discrimination contre les personnes ou les groupes.
6. Pas de discrimination contre les domaines d'application.
7. Distribution de la licence.
8. La licence ne doit pas être spécifique à un produit.
9. La licence ne doit pas contaminer d'autres logiciels.
10. La licence doit être technologiquement neutre .



## Free Software Definition :

En gros, les utilisateurs ont la liberté d'exécuter, de copier, de distribuer, d'étudier, de modifier et d'améliorer le logiciel. [...]

Un programme est un logiciel libre si vous, en tant qu'utilisateur de ce programme avez les quatre libertés essentielles :

- La liberté d'exécuter le programme, pour tous les usages (liberté 0).
- La liberté d'étudier le fonctionnement du programme, et de le modifier pour qu'il fasse votre travail informatique comme vous le souhaitez (liberté 1). Pour ceci l'accès au code source est une condition nécessaire.
- La liberté de redistribuer des copies, donc d'aider votre voisin (liberté 2).
- La liberté de distribuer aux autres des copies de vos versions modifiées (liberté 3). En faisant cela, vous pouvez faire profiter toute la communauté de vos changements. L'accès au code source est une condition nécessaire.

Pour la FSF, le système des logiciels libres a pour finalité la protection de la liberté en matière informatique. Elle se positionne donc davantage d'un point de vue éthique et politique qu'économique. Son objectif final est d'éviter toute appropriation abusive du logiciel.

Face à ce positionnement, l'OSI est plus orientée « marché » et privilégie une approche plus économique en élargissant le champ de l'open source. D'ailleurs à sa création, le terme Open Source devait lever l'ambiguité du free de Free Software. En anglais, Free signifie à la fois libre et gratuit. Hors, s'il l'est souvent, un logiciel libre n'est pas toujours gratuit.

Dans la pratique, un logiciel libre est toujours Open Source; un logiciel Open Source, généralement libre. On voit donc que la proposition de l'OSI recouvre un usage plus large car elle envisage l'utilisation des logiciels libres dans l'activité professionnelle. La FSF fût, dans un premier temps, opposée à cette ouverture vers le monde de l'entreprise car ils pensaient que cela serait préjudiciable aux respects de liberté et d'éthique des logiciels libres. L'OSI au contraire a vu cette ouverture aux entreprises comme la possibilité de diffuser largement les logiciels, de permettre un développement sur le long terme des projets. Ce modèle a prouvé son efficacité dans les grandes entreprises comme Redhat.

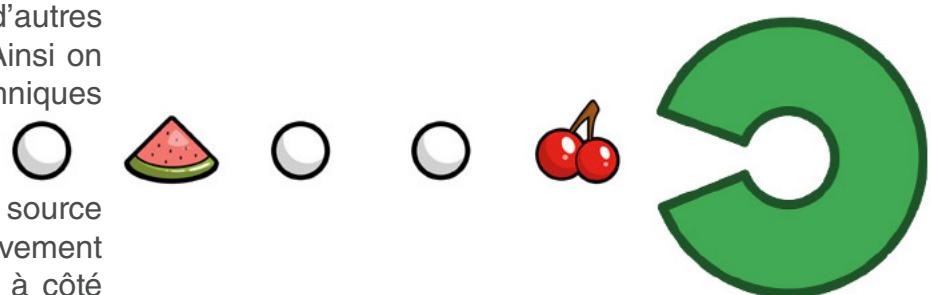
Les partisans des logiciels libres évaluent la valeur d'un logiciel par le respect du développeur de la liberté des utilisateurs et de leurs vies privées. Alors que les partisans de l'open source évaluent le logiciel par sa capacité d'effectuer les tâches souhaitées même s'il utilise d'autres logiciels ou bibliothèques propriétaires. Ainsi on opte à la mise en avant des principes techniques aux dépens de l'éthique.

Depuis l'apparition du concept de l'open source Stallmann n'a pas cessé de critiquer vivement ce mouvement. Il pense qu'il est passé à côté des vrais problèmes soulevés par le logiciel libre. En contre parti la communauté open source voit

que Stellman se pose de faux problèmes philosophiques. Sur le sujet de la diffusion des logiciels libres, RMS développe depuis des années un discours plutôt axé sur les valeurs fondamentales du libre. Il estime en effet que l'utilisation des applications par un large public qui ne partage pas ou ne connaît pas les valeurs du libre va à l'encontre du mouvement open source.

Cet argumentaire l'éloigne d'ailleurs peu à peu de la réalité de l'open source qui aujourd'hui dépend de plus en plus des entreprises qui jouent un rôle majeur dans le développement du libre (plus de 70% du kernel Linux est développé par des salariés). Néanmoins, Richard Stallman bénéficie d'un respect (légitime) et d'une reconnaissance des contributeurs et bénéficie toujours d'une écoute favorable lors d'événements centrés sur le libre.

Mais malgré ce désaccord entre ces mouvements, ils restent très proches et un travail conjoint fait la réussite du modèle économique proposé par l'OSI et assure la protection du patrimoine libre défendu par la FSF. Donc même si ces deux mouvement s'entrechoquent parfois, ils vont dans le même sens, celui du développement du libre dans le monde. C'est ainsi que l'expression Free/Libre and Open Source Software, abrégée en FLOSS est née plus mondialisée et plus globale que open source seulement ou free software seulement. C'est avec cet esprit d'union et d'unité que nous pouvons ensemble combattre ACTA ou PIPA/SOPA et protéger Internet et faire parvenir la technologie à toute la planète.



# Les licences open sources

On dit qu'ils sont écrites et lues que par les hommes en costumes et faites pour eux et leur argent...Vrai ou faux ?!



Voici la liste des licences les plus populaires et les plus utilisées d'après le site de l'OSI (opensource.org) :

- Apache License, 2.0 (Apache-2.0)
- BSD 3-Clause "New" or "Revised" license (BSD-3-Clause)
- BSD 3-Clause "Simplified" or "FreeBSD" license (BSD-2-Clause)
- GNU General Public License (GPL)
- GNU Library or "Lesser" General Public License (LGPL)
- MIT license (MIT)
- Mozilla Public License 2.0 (MPL-2.0)
- Common Development and Distribution License (CDDL-1.0)
- Eclipse Public License (EPL-1.0)

## Copyleft vs. Permissive

«Copyleft» ou «gauche d'auteur» pour les français, se réfère à des licences qui permettent aux utilisateurs la création d'oeuvres dérivées, mais les obligent à utiliser la même licence que l'œuvre originale. Entre une oeuvre "originale" sous copyleft et les créations qui y dérivent une continuité dans l'héritage de la licence copyleft doit être assurée. Il existe deux types de licences copyleft : standard ou fort (ex:GNU General Public License) et faible (LGPL "Lesser GPL" et non pas "Library GPL") qui permet .



La plupart des licences copyleft sont open source, mais pas toutes les licences open source sont copyleft. Lorsqu'une licence open source n'est pas copyleft, ce qui signifie des logiciels publiés sous cette licence peuvent être modifiés puis distribués sous d'autres licences, y compris les licences propriétaires. Ces licences sont généralement appelées soit "non-copyleft" ou "permissives" mais toujours open source. L'exemple le plus évident des licences les plus permissives sont les licences BSD (Berkeley Software Distribution).



copyleft © all rights reserved

# La famille open source

Une famille qui s'élargit de jour en jour.

OSI Affiliates, June 18, 2012



**SOURCEFORGE**

Avec plus de 4.000.000 téléchargements par jour c'est le plus fameux des sites qui hébergent majoritairement des projets libres. Il est aussi reconnu pour les outils de gestion qu'il offre aux développeurs (VCS, wiki, base de données MySQL, URL de sous-domaine unique, etc.).

Software Freedom Law Center

Organisation qui procure assistance et défense juridique aux projets de logiciels libres. Son plus gros projet à ce jour est la rédaction de la licence GNU General Public License (GPL), version 3.



Software Freedom Conservancy

Organisation à but non-lucratif qui aide à promouvoir, améliorer, développer et défendre les projets libres et open source (FLOSS). Parmi ces projets on retrouve Boost, Git, Inkspace, PyPy, Wine, etc.



# Les 12 bonnes raisons pour se mettre au “Puppy Linux”

par Mezghani Mohamed Saïd

L'œuvre de Barry Kauler, Puppy Linux est une distribution spéciale de Linux destinée à rendre l'informatique facile et rapide. Puppy est couverte par la licence GPL/ LGPL. La première version de Puppy Linux a été publiée en Juin 2003. La communauté réunie autour de ce projet est complètement ouverte, sans aucun agenda ni structure. Le nom de cette distribution vient du chien de son développeur. En fait Puppy était un chien de petit taille mais coriace. C'est de celà qu'est venue l'inspiration du nom et un peu du contenu de cette distribution. Aujourd'hui Puppy Linux est la distribution Linux la plus rapide de tous les systèmes d'exploitation et la plus fiable.

## 1. Vitesse

Puppy est une distribution à base de noyau Linux qui est caractérisé à la fois par sa simplicité d'utilisation et par sa rapidité. En effet pour charger toute les composantes de son noyau il suffit d'avoir 100MB de mémoire. Et oui vous avez bien lu 100MB de mémoire. La totalité du système d'exploitation est chargé en mémoire ce qui fait que vous n'avez pas besoin de disque dur. Donc en évitant les accès disque inutiles on peut gagner beaucoup en terme de vitesse. Donc bonne nouvelle pour vos dinosaures ! Dorénavant vous pouvez exploiter vos anciens ordinateurs à l'aide de la distribution Puppy Linux. Et celà sans avoir aucun souci de rapidité ni de fiabilité.



Puppy

↑  
↓



## 4. Une interface conviviale

En regardant la taille de tous le système de Puppy qui ne dépasse pas les 200 MB, on s'attend à un graphique pixelisé et à des icônes qui datent de la première interface graphique développée. Mais figurez vous que bien au contraire, l'interface graphique de Puppy est digne d'une distribution à base de noyau Linux. En effet, malgré que Puppy ne soit pas gourmand en terme de ressources matérielles, on arrive à produire une interface qui séduit de plus en plus de personnes. Et c'est là toute la magie de Puppy.

Wary Puppy 5.3 : la dernière version officielle >>>

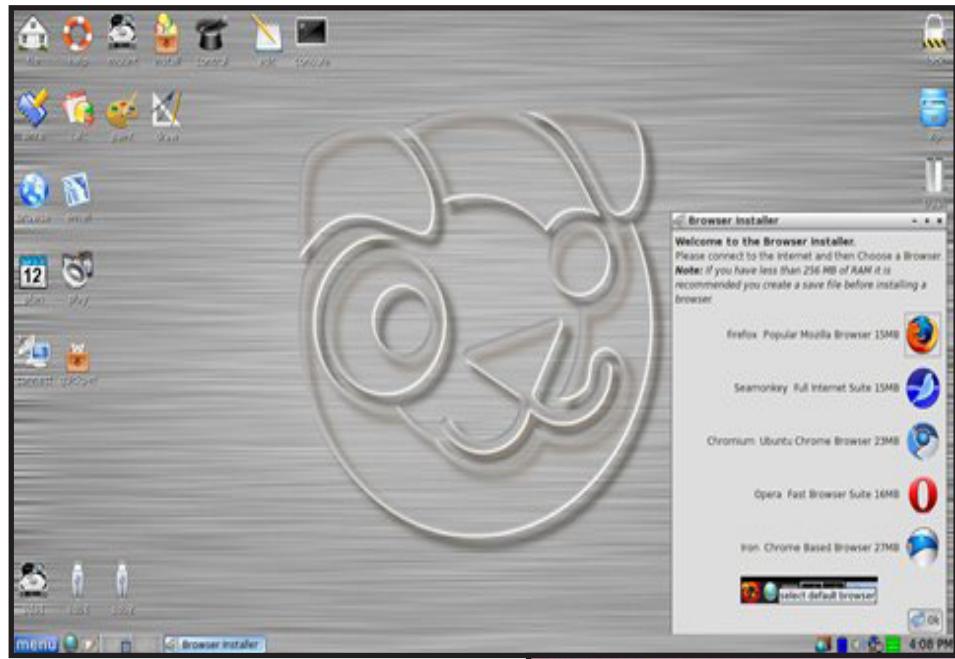
## 2. Sécurité

Comme toutes les distributions Linux Puppy utilise le système de gestion de fichier Unix qui assure une très grande sécurité des données. Et cela en divisant les utilisateurs de l'ordinateur en groupes fixant ainsi des droit d'accès en lecture, écriture et exécution qui ne peuvent pas être contournés.

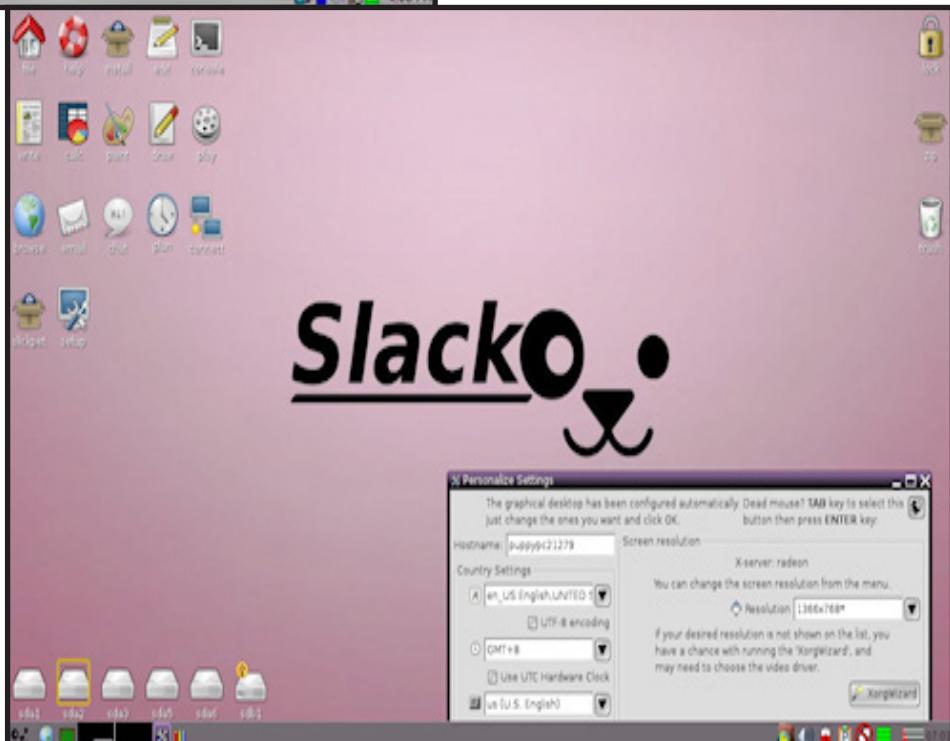
## 3. Open-Source

Le projet de Puppy est un projet open source, ce qui fait qu'il y a toute une communauté réunie autour de ce projet qui assure un rythme de développement rapide par rapport au système d'exploitation non-open source. Cette communauté assure aussi une aide à tout moment à travers le forum et le site de la communauté. Si vous rencontrez un problème, il suffit de le signaler sur le forum pour qu'une armée d'utilisateurs déchaînés vous réponde à l'instant.





Slacko Puppy 5.3.3: compatible avec Slackware. >>>



Il existe aussi un outil très puissant qui s'appelle Woof qui vous permet de compiler votre propre version de Puppy avec compatibilité et les paquets que vous désirez. Mais cet outil nécessite un minimum de connaissances sur l'utilisation des scripts et sur les systèmes de gestion de fichiers.

## 5. Compatibilité

La communauté d'utilisateur de Puppy, qui est de plus en plus large, assure la disponibilité d'un très grand nombre de logiciels et celà sous forme de paquets d'extension pet que vous pouvez facilement installer sur votre système. Il est à noter qu'il y a des versions de Puppy, disponible sur le site, qui sont compatibles avec les paquets de Ubuntu et de Slackware.

<<< Lucid Puppy 5.2.8 : compatible avec Ubuntu.

## 6. Facilité d'utilisation

Aujourd'hui Puppy assure une prise en charge d'une large gamme de périphériques audio, vidéo et même de connexion sans fil. Et ce qui est fascinant avec ça c'est que vous avez tout un bouquet de logiciels de bureautique (tableur, traitement de texte,...) de connexion internet (courrier, chat, navigateur) et même un lecteur de vidéo, de son et d'images. Bref, il y a tout ce qu'il faut pour faire tourner un ordinateur.

## 8. La portabilité

Puppy peut facilement être installé sur une clé USB sur un disque dur ou n'importe quels autres supports de stockage. Si vous utilisez un CD (ou un DVD) Puppy est chargé totalement en mémoire de telle sorte que vous pouvez éjecter le CD (ou le DVD) et utiliser le lecteur à votre guise. Vous pouvez aussi enregistrer votre session sur le support de stockage. Vous pouvez aussi utiliser Puppy à partir d'une clé USB, vous pouvez ainsi emporter vos données personnelles n'importe où et enregistrer votre session toute en assurant une grande protection de ces données.

## 10. Être au point de l'évolution du système

Puppy vous permet d'avoir toujours les dernières options disponibles et cela à travers les mises à jour ou à travers l'installation de nouveaux paquets. Ce qui fait que si vous migrez d'une version à une autre vous n'aurez aucun problème de compatibilité ni de prise en main.

## 12. Stabilité du système

Puppy est caractérisée par une très grande stabilité notamment sur les nouvelles machines. Pour les dinosaures la stabilité du système dépend énormément des ressources matérielles disponibles sur la machine. Mais elle reste beaucoup plus stable et beaucoup plus rapide que les versions des autres systèmes d'exploitation que ces dinosaures sont supposés faire tourner.

## 7. Économiser l'argent

Aujourd'hui, Puppy vous permet d'économiser beaucoup d'argent. D'une part, en évitant d'acheter les licences de logiciels souvent inutiles et d'autre part en assurant un bon fonctionnement avec des ressources matérielles limitées. Plus besoin de formater le disque dur et d'installer le système d'exploitation, vous pouvez charger la totalité du système de Puppy en mémoire. Finis les problèmes d'espace sur le disque dur, désormais vous pouvez consacrer la totalité de l'espace sur le disque dur exclusivement pour vos fichiers personnels.

## 9. Récupération des données perdues

Même si vous n'êtes pas un utilisateur fidèle de Puppy vous pouvez tirer profit de ce système. Si le système que vous utilisez n'arrive plus à démarrer et que tous vos fichiers sont sur votre disque dur. Inutile de paniquer, Puppy est là pour vous permettre d'extraire vos fichiers le plus rapidement possible et avec le minimum de ressources matérielles.

## 11. Support de langues

La communauté de Puppy est de plus en plus large ce qui fait qu'elle est formée par plusieurs nationalités. Et ainsi cette communauté assure un support de la plus grande partie des langues disponibles dans les autres systèmes d'exploitation. On peut donner l'exemple de la communauté des utilisateurs francophones de Puppy qui désormais produit sa propre version de Puppy intitulée Toutou Linux totalement traduite en Français.



## Ce que vous pouvez trouver sur Puppy Linux

Ceci est une petite liste des programmes les plus connues que vous pouvez trouver sur Puppy :



**LibreOffice** : un outil de traitement de texte et tableur qui remplit parfaitement les fonctions de MS Office.



**Gimp** : un éditeur d'images équivalent d'Adobe Photoshop. Vous pouvez aussi trouver d'autres éditeurs comme Pixlr et Pinta.



**Firefox et Thunderbird** : des outils développés par Mozilla Foundation respectivement pour la navigation web et pour le courrier électronique.



**Skype** : pour la téléphonie, messagerie instantanée et vidéoconférence.



**VLC media player** : un lecteur de média populaire qui supporte un très grand nombre de formats multimédias et qui remplit une multitude de fonctionnalités.



**Audacity** : un éditeur de son.



**Wine** : un émulateur qui vous permet de faire tourner quelques applications Windows sur Puppy.

Il existe encore un très grand nombre de logiciels sur Puppy. Il faut tout de même rappeler que certaines de ces application sont encore plus gourmandes en ressource que Puppy lui-même! Chose à laquelle on ne peut pas remédier.

# Ruby, un langage à la mode ?

par Ben Salem Hassen



**Ruby...Un nom qu'on entend parfois sans savoir exactement ce que c'est. Je ne parle ni d'une version du fameux jeu Pokemon, ni de la chanson de "Kaiser Chiefs" mais plutôt d'un langage de programmation poussant un tout petit peu plus loin les limites du paradigme orienté objet; c'est un langage où tout est considéré comme objet, même les entiers !**

Dans cet article nous allons vous parler un peu de ce nouveau né de la programmation : son histoire et son utilité accompagnés de quelques petits tutoriels pour vous montrer sa syntaxe.

**C**ommençons par le tout début. Yukihiro "Matz" Matsumoto, un jeune développeur japonais a commencé l'écriture de Ruby en 1993 s'inspirant de langages telles que Perl et Smalltalk. Deux ans après, Ruby fut finalement publié sous licence libre! Par ailleurs, Ruby est totalement open source. Il est non seulement gratuit, mais son utilisation, sa copie, sa modification et sa distribution sont également libres. Depuis, ce nouveau né de la programmation a commencé à intéresser de plus en plus de programmeurs partout dans le monde, ce qui fait qu'en 2006 il a gagné une réelle reconnaissance. En ce moment, l'index TIOBE, qui mesure la popularité des langages informatiques dans les moteurs de recherche, place Ruby à la dixième place du classement des langages les plus recherchés au monde. Un classement qui est vu à la hausse dans les mois à venir.

Ruby prend son ampleur lorsqu'on parle de son framework Ruby On Rails ou RoR pour les intimes. Ce dernier a été spécialement conçu pour le développement de sites web dynamiques. Depuis sa première version stable, publiée fin 2005, ce framework a marqué le monde du développement web. Ruby On rails

permet d'écrire des applications AJAX et des sites utilisant des bases de données, sans écrire une seule ligne de SQL ou de JavaScript. Ceci grâce à une couche d'abstraction de haut niveau permettant au développeur de se concentrer sur les fonctionnalités plutôt que sur la mécanique autour de ces fonctionnalités. Aujourd'hui, de très gros sites utilisent Ruby on Rails : Twitter, Yellow Pages, Groupon, Scribd, Slide-share, Zendesk, Github, etc.



**Yukihiro "Matz" Matsumoto**

On ne va pas s'attarder sur Ruby on Rails pour l'instant vu que pour le manipuler, il faut avoir au minimum des notions de base en Ruby.

Les possibilités de Ruby sont telles qu'aujourd'hui, de nombreuses sociétés s'intéressent à recruter des développeurs spécialisés. Cependant les sociétés ne trouvent pas souvent des chaussures à leurs pieds et se trouvent contraintes de se retourner vers des spécialistes de langages de programmation plus connus. La qualification des candidats n'est pas à remettre en cause, c'est plutôt leur rareté qui fait d'eux des pokémons légendaires.



**Ruby makes me happy.**

Commençons par la préparation de notre espace de travail.

### Installer Ruby sous Linux :

Normalement Ruby est déjà installé sur votre machine et si ce n'est pas le cas, un bon vieux "apt-get install ruby" fera l'affaire.

La difficulté avec Ruby, est de trouver le bon IDE. Car vue la nature changeante du langage, il est difficile pour un IDE non spécialisé en Ruby de suivre son évolution constante. Heureusement on a FreeRIDE un IDE spécialement conçu pour ruby et totalement gratuit téléchargeable via ce lien : <http://files.rubyforge.vm.bytemark.co.uk/freeride/freeride-linux-installer-0.9.6.sh>.

Pour l'installer, il suffit d'aller avec la commande cd dans votre répertoire de téléchargement (ou dans le dossier contenant ce script fraîchement téléchargé), puis lui donner le droit de s'exécuter via la commande "chmod +x <nom du script sh>" (pour moi c'est : "freeride-linux-installer-0.9.6.sh") Il suffit finalement d'un "sudo ./<nom script>" pour lancer l'installation. Il vous sera demandé de choisir un dossier cible. Appuyez simplement sur entrée.

Si tout se passe bien, vous aurez ce message :

```
----- MANIFEST -----  
Les possibilités de Ruby sont telles qu'aujourd'hui, de  
This is version 0.9.6 of FreeRIDE, the Ruby integrated  
development environment.  
  
This version is built with the following components:  
Ruby 1.8.4  
Fox Toolkit 1.2.16  
Fox Scintilla 1.62  
FXRuby 1.2.6  
  
----- FINE -----  
FreeRIDE successfully installed.  
Start FreeRIDE with '/usr/local/FreeRIDE/freeride'
```

Maintenant , pour lancer freeRIDE, il faut exécuter la commande : "/usr/local/FreeRIDE/freeride". Mais si vous voulez ajouter une petite amélioration supplémentaire, vous pouvez visiter cette page pour apprendre à ajouter une entrée dans le menu des applications :

<http://ubuntuforums.org/showthread.php?t=1706646>. Comme ça, l'accès sera beaucoup plus rapide.

### Premier programme : Hello World !

```
1 print('Enter your name : ')
2 name = gets()
3 puts( "Hello #{name}" )
```

Comme vous pouvez le voir, il s'agit d'une simple commande "puts()". Les instructions en Ruby sont chacune sur une ligne et évidemment, puts affiche ce qui est entre cotes ou double cotes à l'écran. De la même façon, vous pouvez utiliser gets() pour lire une chaîne de caractères entrée par l'utilisateur que vous pourriez assigner à une variable. En Ruby, on n'a pas besoin de prè-déclarer les variables, ou de leur choisir un type spécifique. Ruby se charge de déterminer le type de la variable à partir de son initialisation.

Tout comme puts(), on pourrait utiliser print() qui a le même rôle à une différence près : après l'affichage du message, puts effectue automatiquement un retour à la ligne. Ce qui n'est pas le cas pour print().

A partir de ce bout de code, on va comprendre le comportement des variables, lors de l'affichage. Pour afficher une variable il faut l'écrire sous cette forme #{maVariable}. Et pour que notre variable soit évaluée lors de l'affichage du message, il faut impérativement écrire notre message entre double cotes. Les simples cotes afficheront le message tel qui l'est à l'origine, sans remplacer la variable par sa valeur.

### Deuxième programme : L'orientation objet

Nous allons expliquer maintenant, les détails essentiels de l'orientation objet en Ruby. Ruby est totalement orienté objet, ce qui fait que tout ce que vous utilisez comme variables est traité comme étant un objet. On va implémenter une classe Chien ayant 3 méthodes : set\_name (string), get\_name( ) et parle ( )

Comme vous pouvez le voir, notre classe est délimitée par “class” au début et “end” à la fin.

```

1 - class Chien
2 -   def set_name (nom)
3 |     @monNom = nom
4 end
5
6 -   def get_name
7 |     return @monNom
8 end
9
10 -  def parle
11 |    return 'warf!'
12 end
13 end
14
15 monChien = Chien.new
16 monChien.set_name('Rex')
17 puts(monChien.parle)
18 puts("#{monChien.get_name}")

```

Entre ces deux mots clés on pourra définir nos méthodes. Chaque méthode est précédée par “def <nom de la méthode>” et elle se termine par un “end”. Si vous faites attention à la variable @ monNom (qui est un attribut de la classe Chien), vous pouvez remarquer le “@” qu'on a ajouté. Ce symbole indique que monChien est une variable d'instance. C'est à dire que pour chaque objet créé de type Chien, le nom du chien est unique.

Il n'est pas nécessaire de mettre “return” à la fin de la méthode get\_name, car Ruby retourne immédiatement la valeur de la dernière variable évaluée.

Dans notre cas, on ne peut assigner de nom à notre objet de type Chien que par la méthode set\_nom ( string ). Si on souhaite affecter le nom chien au constructeur lors de la création de l'objet, il suffit d'ajouter une méthode “initialize(<arguments>)”. Ici , le nom est très important. Quand cette méthode existe, Ruby l'appelle automatiquement lorsque la méthode “new” est appelée lors de la création d'un nouvel objet. La méthode “initialize(<arguments>)” peut prendre de zéro à plusieurs arguments.

Ajoutons cette méthode au code précédent : Un “#” marque le début d'un commentaire sur ce qui reste de la ligne.

```

1 - class Chien
2 #ajout de la méthode initialize
3 -   def initialize(nom)
4 |     @monNom = nom
5 end
6
7 -   def set_name (nom)
8 |     @monNom = nom
9 end
10
11 -  def get_name
12 |    return @monNom
13 end
14
15 -  def parle
16 |    return 'warf!'
17 end
18 end
19
20 #Le nom du chien est affecté lors
21 #de la création de l'objet|
22 monChien = Chien.new("Rex")
23 puts(monChien.parle)
24 puts("#{monChien.get_name} ")

```

C'était un petit aperçu du langage Ruby. Vous pouvez visiter des sites comme [tryruby.org](http://tryruby.org) pour accéder à une petite initialisation de 15 minutes. Vous trouverez tous les liens d'apprentissage de Ruby sur une seule adresse :

[jeveuxapprendreruby.fr](http://jeveuxapprendreruby.fr)





# B2G : le nouveau OS mobile de Mozilla

par Zrelli Wicem

*Après l'iOS de la firme Américaine Apple, après Android de l'Open Handset Alliance, après Windows Phone 7 qui fait encore ses premiers pas un nouveau-né s'ajoute dans le domaine des OS mobiles. C'est celui de Mozilla cette fois avec la dénomination « Boot To Gecko » ou tout simplement B2G. Connue pour être derrière le fameux navigateur web Firefox, Mozilla a décidé de conquérir ce nouveau monde très concurrentiel.*

## La dénomination :

Un gecko désigne l'ensemble des lézards de la famille des geckonidés selon Wikipédia mais selon Mozilla la définition est différente, en effet Gecko est un moteur d'affichage libre de pages Web utilisé par de nombreux navigateurs Web, tels que SeaMonkey (anciennement Suite Mozilla), Firefox, Camino ou encore Netscape.

## Un nouvel OS différent :

Des grandes firmes arborant des politiques « closed source » telles que Apple ou Microsoft se bagarrent pour mettre main basse sur le secteur mobile avec parfois des produits qui coûtent très cher et qui rendent l'utilisateur dépendant de leurs gadgets; Mozilla a décidé d'intervenir.

Ainsi après avoir bien étudié l'idée, ce grand projet par lequel Mozilla cherche à révolutionner le monde des OS mobiles a été lancé. Ses mots d'ordre sont « open source » et « orientation web ». Pendant le MWC (Mobile World Congress), le nouveau OS était officiellement dévoilé avec plusieurs fonctionnalités; il prend en charge les services liés aux accéléromètres et la géolocalisation, la caméra, la messagerie et est ouvert aux applications Web. Ce qui ne fonctionne pas encore sont essentiellement les services liés aux Bluetooth, USB, et l'intégration NFC (Near Field Communication).

## Architecture :

Boot To Gecko est un système d'exploitation possédant une base Linux ( l'Open OS qui inspire tout le monde) mais tourné tout entier vers le contenu web. Inspiré par son grand succès dans ce domaine, les applications de l'OS mobile de Mozilla, y compris celles qui sont fournies avec le téléphone, seraient codées en utilisant des technologies telles que HTML5, CSS3 et JavaScript. B2G est essentiellement constitué de trois couches :

- Gonk, qui représente la couche la plus basse de l'OS et qui contient la base : le kernel Linux. C'est la couche d'abstraction matérielle et l'ensemble des composants de bas niveau comme la pile réseau, la pile téléphonie, etc. Cette couche contient des bibliothèques inspiré de l'OS Android qui touche essentiellement le GPS et la caméra.

- Le moteur de rendu Gecko accompagné d'API relatives à la caméra, batterie, l'écran, etc. et conçues pour exposer les capacités du matériel sous-jacent. Un point important à souligner est que Mozilla a travaillé sur tout un jeu d'APIs pour que les applications disposent de capacités importantes. Or, l'éditeur ne garde pas ses APIs pour lui et les a soumises au W3C pour normalisation. Mozilla reste donc dans sa philosophie de partage, mais ce n'est pas forcément le critère le plus important pour le domaine auquel elle s'attaque.

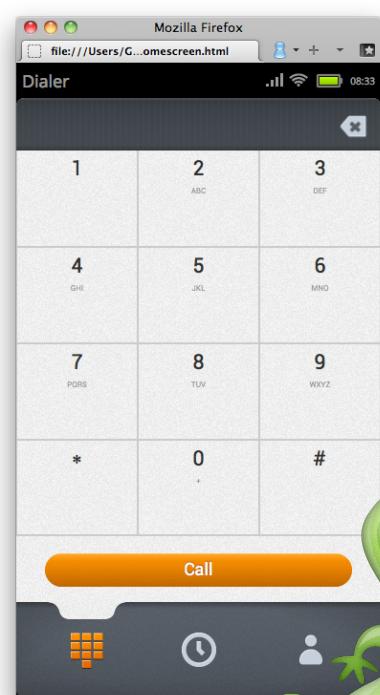
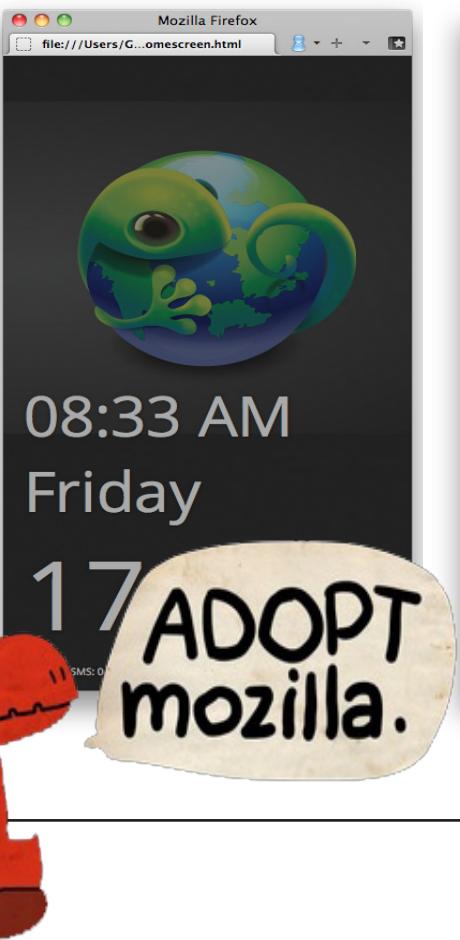
- Gaia, le moteur graphique de l'OS. Utilisant les normes HTML, CSS et JavaScript, cette couche se charge de gérer l'interface utilisateur.

### Perspectives:

Dans un monde très concurrentiel et à fort potentiel, la question qui se pose est : y a-t-il place pour un autre OS mobile face à Android (46% du marché) et iOS (avec 30% du marché) ? BlackBerry est déjà en déclin et WebOS (HP) est mort. Mozilla est sans aucun doute en retard sur le marché, mais il y a toujours une occasion pour se rattraper et pour s'imposer. L'occasion pour Mozilla et Boot-à-Gecko (B2G) réside dans un système d'exploitation qui exploite les points forts de ses rivaux, mais aussi exploite leurs faiblesses et conduit à une nouvelle catégorie d'OS mobile. C'est la politique qui a conduit au succès de Mozilla Firefox. En atten-



dant un nouveau succès pour ce nouveau système d'exploitation, Mozilla a déjà commencé à préparer le terrain et a établi un partenariat avec Telefónica et Deutsche Telekom afin de distribuer son OS au grand public tout en gardant pour cible les Smartphones à bas prix afin que cette technologie libre soit accessible à tous. Affaire à suivre !



L'inreface, les icônes et thème par défaut de B2G.



# Autopsie d'un Bugdroid

par Abbassi Dhia

**Android !? ... Il est partout, sur les mobiles, nombreux les sites web qui en parlent, les gens également. Android aujourd'hui est devenu un mot d'audience, mais que savons-nous réellement d'Android ? Dans cet article on essayera de découvrir l'architecture de ce fameux OS.**

**A**ndroid est un système d'exploitation open source pour smartphones et autres terminaux mobiles principalement tactiles lancé officiellement en 2007. Il est le premier né de l'Open Handset Alliance qui rassemble aujourd'hui près de 80 membres dont Google.

Android se place aujourd'hui comme l'OS n°1 pour les terminaux mobiles. A l'entrée de 2012 il y'a 300 millions de terminaux Android dans le monde et le nombre d'application disponibles sur GooglePlay augmente également pour atteindre désormais 450 000 applications.

Pour commencer je veux éclaircir le fait qu'Android n'est pas un système d'exploitation Linux, en fait il est basé sur un noyau Linux qui a subi plusieurs modifications dans le but de créer un système plus léger adaptable à la faiblesse des ressources matérielles (mémoire limitée, faible résolution, etc.).



open handset alliance



Android bénéficie d'une architecture en couche complète faisant de lui une plateforme riche, dédiée aux appareils mobiles, elle peut se découper en 6 parties principales : le noyau, la couche d'abstraction du matériel, les bibliothèques, l'environnement d'exécution, le Framework et enfin les applications.

Plus de détails dans la page suivante.

# APPLICATIONS

Le moteur d'exécution Linux ou le moteur Android plus connu en anglais par "Android Runtime" a pour rôle principal l'exécution des applications. Il comporte deux éléments:

1. Des bibliothèques de base: Ce sont des bibliothèques qui fournissent les fonctionnalités du langage Java essentiellement les packages JSE 1.5 (Java Plate-forme Standard Edition) qui sont utilisés pour rendre les applications portables. D'autre part on trouve les bibliothèques spécifiques à Android.

2. Une machine virtuelle:

Android dispose de sa propre machine virtuelle nommée Dalvik. Elle est désignée pour utiliser moins d'espace mémoire. Les applications développées pour Android sont compilées au format Dalvik exécutable (.dex). Un outil nommé dx compile les fichiers Java en un fichier .class puis il les convertit en un fichier .dex qui contient le bytecode (code binaire) de Dalvik.

## ANDROID RUNTIME

Les drivers du noyau sont sous licence GNU GPL (General Public License) ce qui exposerait les interfaces propriétaires des fabricants du coup les développeurs d'Android ont décidé de faire une couche qui fournira les interfaces que doivent implémenter les drivers. Cette couche s'appelle couche d'abstraction du matériel ou plus connue par HAL (Hardware Abstraction Layer). Elle joue un rôle intermédiaire entre le noyau et les bibliothèques séparant la plate-forme logique des interfaces matérielles et a pour but faciliter le portage de bibliothèques sur les différents matériels.

# HAL

Couche la plus haute de l'architecture, contient l'ensemble des applications natives, tierces ou développées présentes sur l'appareil. Ces applications sont, exécutées par le moteur d'exécution Android.

Cette couche permet de créer des applications à l'aide d'une plate-forme ouverte. Elle intéresse tout particulièrement les développeurs : ils ont un accès complet aux API utilisés par les applications de base, ainsi ils se trouvent libres de profiter du matériel (périphériques) et des informations sur la localisation et ils peuvent principalement exécuter des services (Un service est une application qui n'a aucune interaction avec l'utilisateur et qui tourne en arrière plan pendant un temps indéfini). Le Framework Android se compose des services applicatifs essentiels pour le fonctionnement de la plate-forme à savoir:

- Activity manager: il gère le cycle de vie des activités et permet le déplacement d'une application (une fois elle est fermée) à une autre (en train d'être lancée).
- View System: fournit tous les composants graphiques : listes, grille, text box, buttons, etc.
- Window Manager : il gère les fenêtres des applications et leur disposition à l'écran.
- Resource Manager : gère les ressources multimédias tels que les images, les fichiers audio, etc.
- Content Provider : gère le partage de données entre applications. Les données peuvent être partagées à travers une base de données (SQLite), des fichiers ou par réseau.

## Linux KERNEL

Le noyau utilisé dans Android est un noyau Linux modifié afin de satisfaire certains besoins spéciaux de la plate-forme tels que la gestion de l'alimentation, le partage mémoire, etc. permettant plus de performance et de fiabilité pour les appareils mobiles. Mais pourquoi le choix d'un noyau Linux ? En fait, Le noyau Linux est reconnu par la stabilité de son système de gestion de mémoire et de processus, ainsi que son modèle de sécurité robuste basé sur le système de permission (un système qui définit les droits des utilisateurs sur les systèmes). Et le plus intéressant est que le noyau est entièrement open source et il y a toute une communauté de développeurs qui veille sur son amélioration.

# FRAMEWORK

