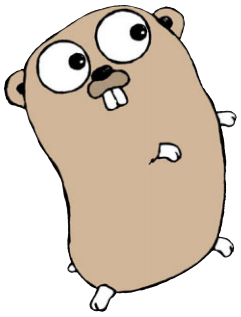
programmez!

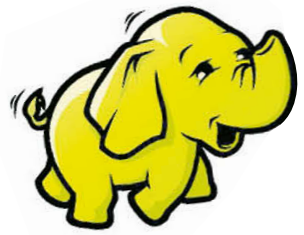
#196 - Mai 2016

le magazine des développeurs



Le langage Go

Découvrez un langage surprenant



Angular 2

Un framework magique !

3D facile

N'ayez plus peur de créer des objets 3D

Katie : 12 ans

& makeuse !

Fières d'être développeuses !

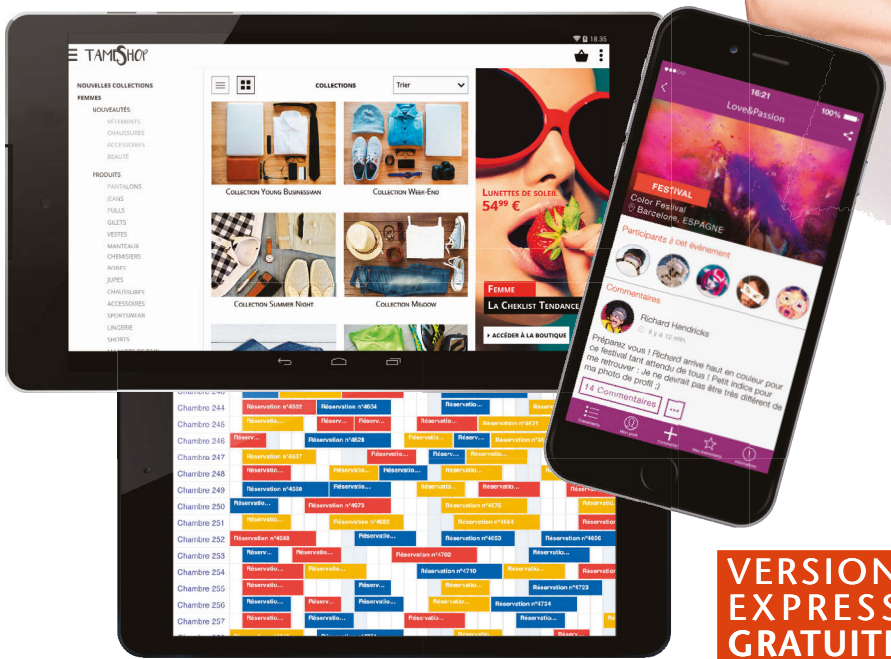
Un numéro spécial codé, débugué et compilé par la communauté Duchess France

M 04319 - 196 - F: 5,95 € - RD

Printed in EU - Imprimé en UE - BELGIQUE 6,45 € SUISSE 12 FS - LUXEMBOURG 6,45 € - DOM Surf 6,90 € Canada 8,95 $ CAN - TOM 940 XPF - MAROC 50 DH

WINDEV
NOUVELLE VERSION MOBILE

DÉVELOPPEZ MOBILE NATIF. UN SEUL CODE SOURCE, TOUTES LES CIBLES.



VERSION EXPRESS GRATUITE
Téléchargez-la !
Version non limitée dans le temps

Développez vos applis «une seule fois»

WINDEV MOBILE 21 vous permet de développer des applis mobiles **natives** pour tous les systèmes. Le code et les interfaces sont identiques.
Il suffit de recompiler le source pour obtenir des applis natives pour **Android**, pour **iOS**, pour **Windows 10 Mobile**, pour smartphones et tablettes...
Base de Données embarquée, Client/Serveur et Cloud incluse.

Vous disposez déjà d'applications WINDEV ? Elles sont compatibles. Recompilez-les simplement pour mobile !

DÉVELOPPEZ 10 FOIS PLUS VITE

www.pcsoft.fr

Des centaines de témoignages sur le site
Dossier complet gratuit sur simple demande

Lorsque François Tonic nous a parlé de son idée, que Duchess France soit rédactrice en chef de ce numéro, nous n'avons pas voulu profiter de ce numéro spécial pour en faire un magazine "girly" avec des paillettes, des fleurs et des cœurs de partout, mais plutôt réaliser un numéro qui nous ressemble, un magazine qui vous ressemble.

La technique fait partie de notre lot quotidien, donc nous avons voulu que les sujets soient divers et variés tout en étant techniques et dans l'actualité du moment.

Vous allez notamment être initié au langage GO de Google, devenir incollable sur les moteurs 3D, découvrir une techno mobile venant du pays basque "NeoMAD", connaître les derniers tips de Visual Studio, découvrir deux façons de concevoir sa pile analytique avec Spark, devenir un magicien sur le framework Angular, vous plonger dans Kafka...

Nous avons pensé à votre temps libre aussi pour pouvoir, pourquoi pas, créer une console de jeux vidéos rétro avec une Raspberry PI et vous faire découvrir le dernier Arduino, la 101.

Vous n'étiez pas présent au Startup Weekend à la Réunion et à la BUILD 2016 ? Pas de panique, nous avons pour vous un résumé.

Et cerise sur le gâteau, nous allons vous faire découvrir Katie, une jeune fille de 12 ans qui est makeuse.
Nous en sommes fans !

Nous espérons que ce numéro vous plaira. Nous y avons mis du cœur à l'ouvrage ! :-)

Aurélie Vache
Rédactrice en chef

À lire dans le prochain numéro n°197 en kiosque le 31 mai 2016

Dossier DevOps saison 3

Quoi de neuf dans le DevOps ? Revenons aux fondamentaux de la démarche DevOps et aux nouvelles tendances, notamment, avec l'utilisation des conteneurs qui apporte un autre niveau de flexibilité.

Swift : un langage surpuissant à découvrir

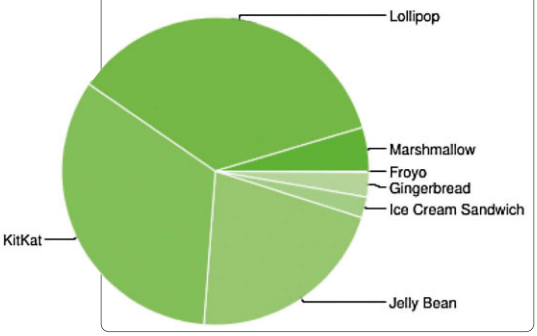
Découvrez un langage surpuissant pour Mac, iOS et Linux !

.Net : open source, open source et open source

Microsoft ouvre de plus en plus de couches techniques. Plongez dans le monde open source d'ASP.Net Core et de .Net Core !

LES VERSIONS D'ANDROID INSTALLÉES

Tous les détails sur
https://developer.android.com/about/dashboards/index.html



Amazon
retire les câbles USB type C de mauvaises qualités. Ils peuvent endommager le matériel

Xamarin,
vive les licences gratuites

Swift
va-t-il devenir le nouveau langage référence de Google, Facebook et Uber ?

Google veut vendre sa filiale
robotique

La saga judicaire
Oracle – Google
n'est toujours pas terminée. Fin mars, Oracle réclame jusqu'à 9 milliards $!

OpenJDK 9
c'est pour quand ? Toujours le 23 mars 2017

+276 000
réservations pour la Tesla Model 3

Le prochain
Doom
sera disponible le 13 mai prochain... sauf retard

Oculus et Vive
connaissent des retards de livraisons et des pénuries de composants

50 000 €,
le prix d'un robot à l'image de Scarlett Johansson

Western Digital propose un disque SSD spécialement réalisé pour le Raspberry Pi :
WD PiDrive 314 Go

Nest
stoppe les serveurs de Revolv.
Les utilisateurs apprécient

Vivaldi,
un nouveau navigateur, par des anciens d'Opera

Mais c'est quoi une licorne ?
Une licorne est une startup qui dépasse 1 milliard de valorisation et est non-cotée en bourse

Pas de nouveaux smartphones chez Microsoft avant...
2017

SPACEX : LA PROCHAINE RÉVOLUTION DANS LA CONQUÊTE DE L'ESPACE ?


D.R.

Après plusieurs échecs, la société SpaceX, l'autre défi d'Elon Musk (Tesla) a réussi à faire atterrir sa fusée, Falcon 9 sur une plateforme en pleine mer ! Falcon 9 est un vrai lanceur pouvant lancer des satellites, voire, prétendre à aller au-delà, la Lune, Mars... Ces nouvelles sociétés bousculent clairement l'industrie actuelle, notamment Ariane Espace.
Dans le même temps, la capsule Dragon de SpaceX a réussi sa fonction avec la station orbitale, ISS.

L'INDEX TIOB DU MOIS

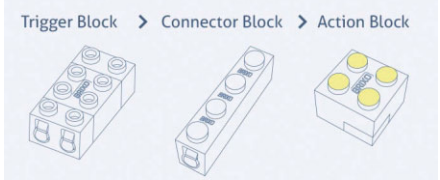
04/16	04/15	tendance	langage	%	évolution en %
1	1		Java	20.846%	+4.80%
2	2		C	13.905%	-1.84%
3	3		C++	5.918%	-1.04%
4	5	^	C#	3.796%	-1.15%
5	8	^	Python	3.330%	+0.64%
6	7	^	PHP	2.994%	-0.02%
7	6	v	JavaScript	2.566%	-0.73%
8	12	^^	Perl	2.524%	+1.18%
9	18	^^	Ruby	2.345%	+1.28%
10	10		Visual Basic .NET	2.273%	+0.15%

Peu de changements importants : Python connaît une nouvelle progression tandis que Swift ne tardera plus à rejoindre Objective-C.

QUELQUES PROJETS KICKSTARTER

OLO : la première imprimante 3D « pour » smartphone. 2398 % du montant souhaité !

Cubetto : un projet innovant pour proposer la programmation très ludique dès l'âge de 3 ans ! Simple et efficace. 1413 % des objectifs de financement.



Brixo : un projet sympathique mêlant capteurs, électroniques, moteurs et apps. Ce sont des briques électroniques. 438 % des objectifs.

Sticker Project : bien souvent, on oublie à quoi correspond telle ou telle prise. Sticker Project est là pour ça avec de petits stickers illustrés...

WEBDEV®

NOUVELLE VERSION 21

CRÉEZ FACILEMENT DES SITES «RESPONSIVE WEB DESIGN»

ACCÉDANT À VOS BASES DE DONNÉES

Rendez vos sites «Mobile Friendly».

WEBDEV 21 vous permet de rendre facilement vos sites dynamiques «Mobile Friendly». Créez un seul site pour toutes les cibles.
Les sites que vous créez sont ainsi mieux référencés par Google.
Responsive Web Design et Dynamic Serving sont à votre service dans WEBDEV 21.

Vous disposez déjà d'applications WINDEV ? Elles sont compatibles WEBDEV 21 !

DÉVELOPPEZ 10 FOIS PLUS VITE

www.pcsoft.fr

Des centaines de sociétés témoignent sur le site

mai

DevOps Day 2 - Issy-les-Moulineaux : 11 mai

Ne manquez pas le DevOps Day, le mercredi 11 mai au Centre de Conférences du Campus Microsoft. Au cours de cette matinée consacrée à la démarche DevOps, découvrez comment accélérer en confiance entre les métiers, les études et les opérations ! Vous assisterez notamment à différents témoignages clients dont celui de Christophe Goset du Crédit Agricole CIB, et pourrez repartir avec un exemplaire du livre « Découvrir DevOps » préfacé par Patrick Debois (fondateur du mouvement DevOps) et rencontrer les auteurs. Où : dans les locaux de Microsoft France, au 41 quai du Président Roosevelt, 92130 Issy-les-Moulineaux. Inscription : https://goo.gl/97LmLa

Ncrafts 2016 - Paris : 12 & 13 mai

NCrafts est une conférence unique, internationale, généraliste et indépendante qui se déroule en France depuis 2014. Il s'agit d'un évènement pour les développeurs par des développeurs. De nature éclectique Ncrafts traite autant des meilleures pratiques, des dernières technologies, de design ou encore d'architecture, le tout ficelé avec une bonne dose de bon sens, de pragmatisme software craftsmanship et beaucoup de code. Venez nous retrouver cette année avec des speakers internationaux de haut vol comme Simon Brown sur l'architecture, Liz Kheog sur BDD, Michael Feathers sur le code et le refactoring, Greg Young le papa de CQRS, Mathias Verraes sur DDD et beaucoup beaucoup d'autres sur la programmation fonctionnelle, le design, la mobilité, le Web, le devops et bigdata,.., plus d'infos sur http://ncrafts.io.

J On The Beach - Málaga (Espagne) : les 20 & 21 Mai

Vous connaissez les principales grandes conférences en France, mais connaissez vous "J On The Beach" en Espagne ? Si vous voulez profiter des plages espagnoles et des tapas, tout en assistant à des conférences de qualité autour de la Big Data (Spark, Elastic, Mesos, Java, Scala, OrientDB ...) alors n'hésitez pas ! :-)
Site : http://www.jonthebeach.com/

PHP Tour 2016 - Clermont-Ferrand : 23 & 24 mai

Deux jours pour réunir toute la communauté PHP en plein coeur de la France, deux jours pour échanger, se perfectionner, découvrir de nouvelles fonctionnalités ou approfondir ses connaissances PHP.
Site : http://event.afup.org

juin

EclipseCon France - Toulouse : du 7 au 9 juin

Cette conférence organisée par la fondation Eclipse, se déroule à Toulouse au Centre de Congrès Pierre Baudis comme ces deux dernières années et sera totalement en anglais. Au menu 3 journées bien remplies avec des ateliers et des meetups sur l'IoT et l'embarqué, le Cloud, l'optimisation des performances de l'IDE Eclipse, le DevOps, les Sciences, le Polarsys, le Java 9 ... Bref the place to be pour tout ce qui concerne Eclipse et sa communauté.
Inscription : https://www.eclipsecon.org/france2016/registration

Best Of Web 2016 - Paris : les meilleurs talks de l'année le 10 juin

En 2015, huit groupes *meetups* se sont fédérés pour proposer une conférence réunissant leurs meilleures présentations : Best Of Web. Une nouvelle édition nous est proposée en 2016, plus ambitieuse encore : cette fois ce ne sont pas huit mais douze *meetups* qui sont réunis, et dont le Best Of sera visible le 10 juin : AngularJS-Paris, Backbone JS Paris, Paris API, CSS Paris, etc.
Une première journée organisée le 9, en plus petit comité, où seront proposées des formations, ainsi que d'autres surprises. L'ambition demeure la même : proposer un événement dans l'esprit *meetup,* communautaire, abordable, participatif et convivial, différent des conférences plus classiques où s'expriment des stars. Best Of Web 2016 se déroulera les 9 et 10 juin 2015 à la Grande Crypte (Paris 16e) où 500 participants sont attendus.
Site : http://bestofweb.paris/
Twitter : https://twitter.com/bestofwebconf
Email : contact@bestofweb.paris

Riviera Dev - Nice (sophia antipolis) : les 16 et 17 juin

2 jours de conférences, 24 présentations au campus Sophia Tech a Sophia-Antipolis. Le CFP est ouvert jusqu'au 30 Avril donc le programme n'est pas encore acté mais il devrait y avoir du Java, du software craftmanshift, du Jenkins ...
Inscription : http://rivieradev.fr/

Agile France - Paris : 16 & 17 Juin

Pendant deux journées on y parle de méthodes agiles, de logiciels, de technologies, de bien être au travail, d'expérience utilisateur ... et de soi.
Plus d'infos : http://2016.conf.agile-france.org/

DOMAINES | MAIL | HÉBERGEMENT | E-COMMERCE | **SERVEURS**

APPS CLOUD READY!*

CLOUD SPECTATOR
2x NUMÉRO 1
Europe et Amérique du Nord
Test comparatif 03/2016

intel Powered by Cloud Technology

1&1 SERVEUR CLOUD
1 MOIS GRATUIT !*

1&1 Cloud App Center est le meilleur moyen de lancer vos applications ! Choisissez parmi plus de **100 applications ultra-modernes** et associez-les à la vitesse et aux performances du serveur Cloud 1&1, **numéro 1 du test comparatif Cloud Spectator !**

- ✓ **Plateforme puissante et sécurisée**
- ✓ **Pas besoin de connaissance en serveurs**
- ✓ **Facturation à la minute**

1 MOIS POUR ESSAYER
1 CLIC POUR CHANGER DE PACK
1 APPEL UN EXPERT VOUS RÉPOND

0970 808 911
(appel non surtaxé)

1&1
1and1.fr

*Applications prêtes pour le Cloud. 1&1 Serveur Cloud : 1 mois d'essai gratuit, puis à partir de 4,99 € HT/mois (5,99 € TTC) (pour la configuration du serveur Cloud S). Facturation mensuelle en fonction de la configuration choisie. Pas de durée minimum d'engagement. Des frais de mise en service de 9,99 € HT (11,99 € TTC) s'appliquent. Conditions détaillées sur 1and1.fr. Intel et le logo Intel sont des marques commerciales d'Intel Corporation aux États-Unis et/ou dans d'autres pays. 1&1 Internet SARL, RCS Sarreguemines B 431 303 775.

La culture geek et les femmes

Laurent VACHE
@vachaldo

La culture geek a, de tous temps, été orientée pour ne convenir qu'aux hommes. Les femmes qui en faisaient partie étaient soit considérées comme des OVNI, soit comme des usurpatrices ou des profiteuses.

Par exemple, si vous voyiez une femme entrer dans un magasin de comics, tous les hommes se seraient retournés vers elle en se demandant si elle s'était perdue ou si elle cherchait quelqu'un. Aucun n'aurait imaginé qu'elle était là parce qu'elle était fan de Spiderman ou de Thor. En gros, une femme qui entre dans ce monde d'hommes, c'était comme en voir une arriver à un JUG. L'étonnement et l'aspect intriguant de la chose se serait fait sentir.

Mais même si la culture geek tend à se féminiser, quelle est la véritable place faite aux femmes de nos jours ? Comment sont-elles utilisées et ont-elles évolué dans ce monde d'hommes volant presque obligatoirement au secours de la demoiselle en détresse ?

Nous allons tenter de le comprendre au travers de l'image de la femme au cinéma et dans les séries TV.

Pré-2000 : entre faire-valoir et superwoman

La culture geek a souvent été considérée comme sexiste, réservée aux hommes et où les femmes n'avaient pas leur place. Que ce soit au cinéma ou à la télévision, quand une héroïne se détachait du lot des faire-valoir sexy habituels, il lui fallait toujours un preux chevalier pour la sauver, ou alors on la cantonnait aux fourneaux.

L'exemple typique était celui de Samantha dans *Notre sorcière bien aimée*. Une femme forte avec des pouvoirs, héroïne et personnage principal de la série, mais bien loin de l'émancipation et de la liberté de penser des sœurs Halliwell quelques décennies plus tard.

En 1993, Meryl Streep prédisait que si la place des femmes au cinéma ne changeait pas, on n'en verrait plus du tout d'ici 2010. Hollywood se voyait accusé, en ce début des 90, de tomber dans le stéréotype de la femme objet et soumise. En ligne de mire le cinéma d'action, d'aventure et de science-fiction que le geek mâle affectionne et qui tourne le dos aux femmes héroïnes. Les années précédentes ont permis à certaines femmes d'être stars dans des films, mais pour les producteurs elles n'en étaient pas le sujet central. Si *Alien* ou *Terminator* ont cartonné, ce n'était pas, pour eux, grâce à Ellen Ripley ou Sarah Connor, mais grâce à l'extra-terrestre vorace ou au robot tueur.

Du côté des séries TV, la décennie 90 nous a apporté son lot de femmes fortes et de femmes faire-valoir ou sexy... parfois tout en même temps. Pour citer un exemple qui englobe un peu tout, je ne pouvais passer à côté de *Xena la guerrière*. L'exemple typique de la femme forte, mais qu'on garde quand même un peu sexy pour attirer le chaland. On pourrait aussi parler de Samantha Carter, LA femme à appeler en cas de problème informatique dans *Stargate SG-1*. Mais si on ne devait retenir que deux exemples de ces années là, je nommerais Buffy Summers et Angela Bennett.

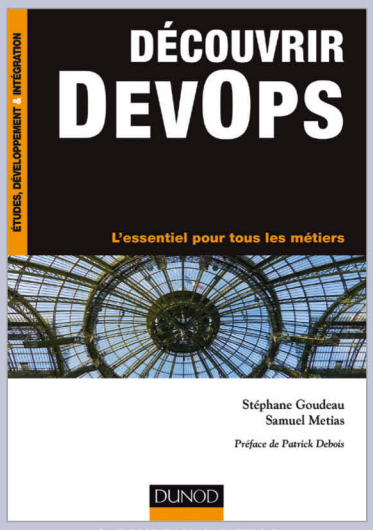
La première, dans *Buffy contre les vampires*, était une jeune femme, belle et athlétique, qui n'avait pas forcément besoin des hommes pour être et exister. Même si, tout de même, le regard approbateur de son père spirituel Giles, et les canines sexy d'Angel lui apportaient du réconfort. Mais, si on enlève le côté élue, superpouvoir etc., elle restait une ado normale à laquelle on pouvait s'identifier. Les autres femmes de la franchise n'étaient pas en reste, du moins à partir de la saison 3 et du passage aux années 2000. La petite Willow, timide et transparente dans les premières saisons, laissera place à une sorcière affirmée et indépendante. La bimbo Cordélia, qui pensait que le bouton de sauvegarde de son ordinateur était la touche « Delete », deviendra également plus affirmée pour devenir une vraie femme forte dans le spin-off *Angel*.

Et sans oublier la dernière saison où les femmes prennent presque entièrement le pouvoir face à la Force et à un Nathan Fillion misogyne à souhait.

DÉVELOPPEZ VOS **COMPÉTENCES**

S. MEYERS
9782100743919 ▪ 320 pages ▪ **29,00 €**

42 conseils pour connaître les bonnes pratiques et comprendre le fonctionnement de C++11 et de C++14.

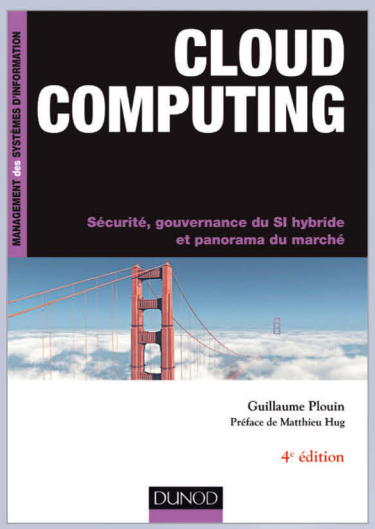


S. GOUDEAU, S. METIAS
9782100745876 ▪ 232 pages ▪ **24,90 €**

Une vision à 360° de la démarche DevOps : toutes les informations pour répondre aux problématiques de tous les métiers concernés.

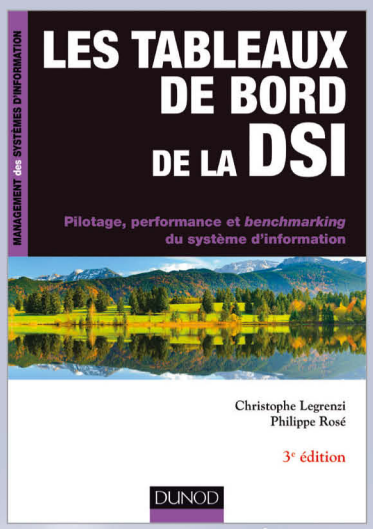
L. BRIXIUS
9782100738885 ▪ 336 pages ▪ **29,00 €**

Le guide pratique pour s'initier à SketchUp, le logiciel phare de la modélisation 3D.



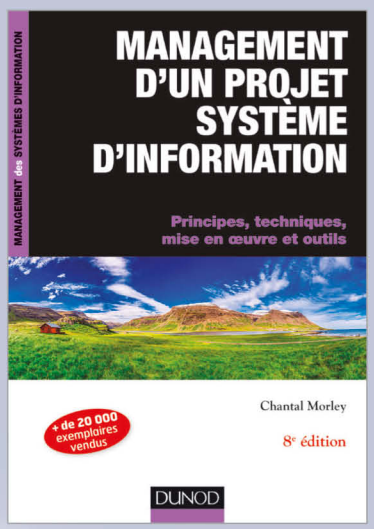
G. PLOUIN
9782100742592 ▪ 288 pages ▪ **29,50 €**

Pour comprendre les concepts et les enjeux du cloud computing.



C. LEGRENZI, P. ROSÉ
9782100742585 ▪ 272 pages ▪ **35,00 €**

Un panorama des tableaux de bord les plus couramment utilisés par les DSI pour leur donner les moyens de réfléchir aux indicateurs qu'ils utilisent au quotidien.



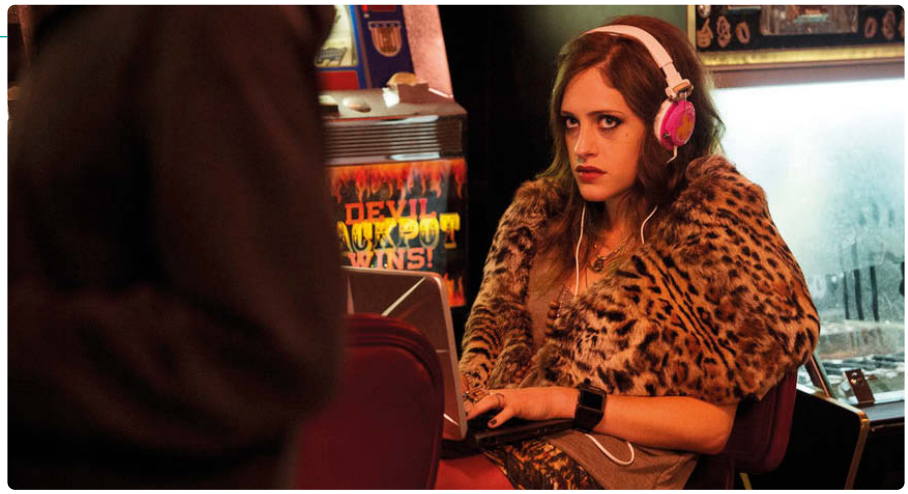
C. MORLEY
9782100747368 ▪ 512 pages ▪ **45,00 €**

L'analyse des outils et des méthodes de gestion pour conduire un projet SI efficacement.

Réalisation WIP

www.dunod.com

DUNOD
ÉDITEUR DE SAVOIRS

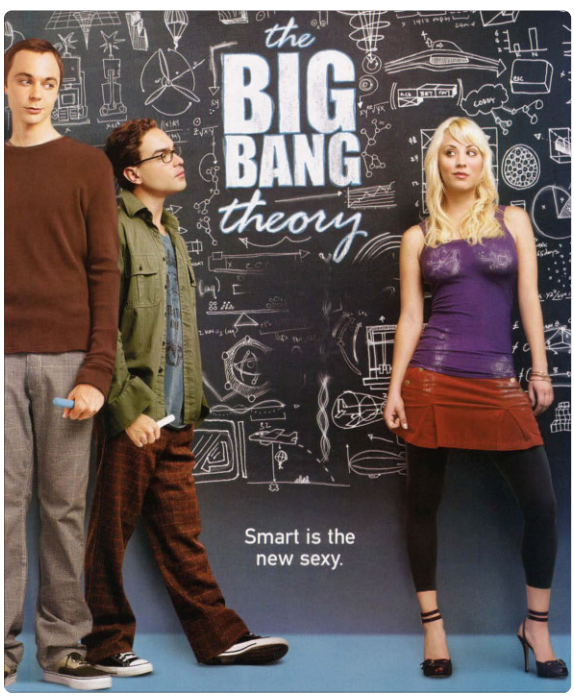


Et que dire donc d'Angela Bennett, la séduisante informaticienne du film *Traque sur internet*, incarnée à merveille par Sandra Bullock. On pouvait retrouver là une héroïne plausible, au comportement normal pour une geek de cette époque et avec laquelle le téléspectateur pouvait s'identifier. Angela Bennett est intelligente, obnubilée par son métier et par les nouvelles technologies, et ne peut pas partir en vacances sans son ordinateur portable. Un peu comme beaucoup de personnes de nos jours, geek ou pas, qui ne peuvent pas réserver un hôtel sans vérifier que le wifi y sera bien présent.

Post 2000 : Plus proches de la réalité ?

Depuis les années 2000, on commence à tendre vers une féminisation de la culture geek, et une place plus importante des femmes dans les séries et les films. Finie la belle demoiselle en détresse ou la cruche ne pouvant rien faire sans un mâle pour l'aider. On voit désormais apparaître de vraies femmes fortes qui n'ont besoin de rien ni personne pour exister. Les séries TV ont jeté le pavé dans la marre qui a permis cette révolution. Les exemples sont légion avec Angela dans *Bones*, Toshiko Sato dans *Torchwood* ou encore Chloé Sullivan dans *24*. Ce sont des scientifiques qui ont une vie personnelle riche et qui sont douée dans leur travail. Loin du cliché de l'assistante du professeur qui doit juste prendre les notes. D'ailleurs, au tout début des années 2000, ce cliché est inversé dans la série Sydney Fox, l'aventurière où le personnage principal est un Indiana Jones au féminin qui a comme assistant Nigel, un homme qu'elle doit sauver presque à chaque épisode d'un danger ou d'une situation délicate. Cette évolution peut se sentir également dans des séries longues qui ont vu passer les années et les différentes visions de la société. Doctor Who était l'exemple type dans les années 60 à 80 où la femme qui accompagnait le héros ne servait limite à rien d'autre qu'à se faire capturer pour être sauvée par la suite. Alors que de nos jours, la compagne du docteur n'est pas forcément l'héroïne à part entière de la série, mais elle reste en tout cas le personnage central qui « contrôle » et raisonne le seigneur du temps.

D'autres séries de la culture geek permettent aux femmes de sortir leur épingle du jeu, et de ne plus être de simples faire valoir sexy. C'est le cas de *The Big Bang Theory* qui permet aux femmes qui y sont présente d'exister autrement qu'au travers des hommes. Penny, qui a démarré la série comme étant un personnage frivole et intellectuellement limitée va peu à peu se transformer en une jeune femme maligne et intégrant des concepts geeks jusqu'à alors étrangers pour elle. Elle ira même jusqu'à citer des scènes de Star Wars ou parler fièrement de « beta test ». Une série qui au départ pouvait laisser penser à une suite de sketchs entre mecs, et faite pour les mecs, va, au fil des saisons, voir se former en parallèle un trio de femmes parlant entre elles de problèmes féminins. Ce trio féminin apporte un équilibre à la série.



D'autres séries en revanche continuent de cultiver le machisme ambiant de la culture geek. C'est le cas de *Silicon Valley*. Les scénaristes ont oublié d'insérer des personnages féminins important. On peut y apercevoir l'assistante du grand patron qui n'est là que pour tomber sous le charme du héros. Les autres exemples restant l'apparition d'une strip-teaseuse ou encore d'une entrepreneuse dont la seule fonction est d'arnaquer les programmeurs naïfs pour qu'ils codent son application à sa place. Un retour en arrière ?

Heureusement que des OVNI existent. *Halt and Catch Fire* tend à corriger ce constat. On peut y retrouver deux femmes, Donna et Cameron, qui sont placées sur le devant d'une scène occupée de nos jours par les hommes. Deux leaders féminins coincés dans un monde masculin. Donna, qui est plus âgée d'une dizaine d'années a dû se battre pour tout, alors que Cameron commence à récolter les fruits de ce combat. Mais rien n'est gagné pour autant pour toutes deux qui vont devoir survivre dans un monde dans lequel elles n'ont pas forcément été invitées.

Que ce soit à la télévision ou au cinéma, la culture geek regorge de nos jours de nombreuses figures féminines de premier plan. Scarlett Johansson en est une, notamment grâce à son interprétation de La Veuve Noire de Marvel, ou encore Halle Berry qui a non seulement interprété la Tornade des X-Men mais également une scientifique et astronaute dans la série futuriste *Extant*. Sans oublier qu'elle fut l'une des premières James Bond girl à ne pas avoir forcément besoin du séduisant british pour se sortir de situations délicates.

Conclusion

L'évolution de la femme dans la culture geek peut presque entièrement se résumer à l'évolution de Lara Croft depuis sa naissance en 1996 dans le premier opus de Tomb Raider. Nous avions une femme pulpeuse, modélisée pour attirer l'argent du plus grand nombre de testostérones ambulantes. Le paroxysme de cette image de femme objet sexy, dont on se souvient d'avantage de sa plastique que du scénario des films, fut atteint quand Angelina Jolie l'incarna au cinéma. Alors que de nos jours, cette image tend à se déliter avec la nouvelle mouture de Lara Croft. Un personnage de femme moins pulpeuse, mais toujours aussi forte et intelligente, qui arpente les nouveaux Tomb Raider non pas seulement dans un but de plaire aux hommes et à leur porte-monnaie, mais à toute une génération de geek dont les femmes font maintenant intégralement partie.

Sur abonnement ou en kiosque

Le magazine des pros de l'IT

Mais aussi sur le web

Ou encore sur votre tablette

L'INFORMATICIEN
www.linformaticien.com

Duchess France

Créée en 2010, Duchess France est une association destinée à valoriser et promouvoir les femmes ayant des profils techniques dans l'informatique. En leur donnant plus de visibilité, elle aspire à faire connaître ces métiers et créer des vocations auprès d'autres femmes et jeunes filles.

Notre association organise des événements techniques (meetups, hands-on) destinés à tous les passionné-e-s de l'IT, homme ou femme, débutant ou confirmé, autour de diverses technologies pour apprendre et partager des connaissances (pour vous inscrire à nos évènements rendez-vous sur notre page Meetup).

L'une des missions de Duchess France est de voir plus de femmes animer des conférences. Pourquoi ? Tout simplement parce qu'il existe beaucoup de talents féminins qui restent dans l'ombre. En étant davantage visibles, on montre aux jeunes filles qu'elles peuvent s'épanouir autant que les garçons dans un métier technique et que c'est une voie qu'elles peuvent envisager au même titre qu'eux.

Pour les encourager, nous organisons des ateliers de préparation aux Call For Papers, ainsi que des sessions de coaching pour répéter les présentations (via hanghout ou en réel).

Nous organisons également des ateliers de prise de parole en public.

Cette année une dizaine de femmes sont speakers à Devoxx, sur 200 speakers annoncés : le chemin est encore long !

Duchess France est partenaire de plusieurs grandes conférences techniques telles que Devoxx, les Dot conférences, les user groups (PJUG, PSUG ...), l'EclipseCon France et d'autres associations (Girls in Tech, Simplon, Rails Girls, etc.

Marrainage

Nous avons lancé il y a quelques mois notre initiative de marrainage de mentorat : #AdoptADuchess

Encourager les femmes à aller vers les métiers techniques, c'est bien. Faire en sorte qu'elles s'y sentent bien, les aider à développer leurs compétences efficacement et leur permettre d'élargir leur réseau, c'est encore mieux ! C'est un programme qui permet d'encourager, soutenir, conseiller, valoriser et élargir le réseau de femmes qui débutent dans les métiers techniques de l'informatique.

En 3 mois, déjà 20+ marraines et 50+ filleules inscrites ! Une communauté d'entraide commence à se former petit à petit et nous comptons sur vous pour l'agrandir ! :-)

Pour intégrer le programme, il suffit de vous inscrire dans la rubrique Marrainage de notre site (http://duchess-france.org).

Célia (marraine) : *"Quand les Duchess ont lancé l'opération de marrainage, j'y ai vu l'occasion de transmettre et partager ma passion pour ce métier avec Alexa, ma filleule, mais aussi d'échanger avec d'autres professionnelles du domaine. Cela me permet aussi, à mon niveau, de m'engager pour la promotion des femmes dans les métiers du numérique."*

Alexa (filleule) : *"De mon côté, c'est mon chef de département qui m'a fait connaître ce système de marrainage. J'y ai vu l'occasion de m'engager à mon niveau pour l'égalité des genres et la diversité dans le monde de l'informatique, mais aussi d'apprendre, de me construire un réseau, et de partager mes idées et ma curiosité."*

Stéphanie (marraine) : *"Maëlle a eu quelques difficultés au cours de sa 3ème année de formation d'ingénieur. Via le réseau des marraines, une ancienne élève ingénieure issue de cette école a pu appeler directement la scolarité pour comprendre d'où venait le problème et aider Maëlle à corriger le tir : le réseau c'est pas du flan !"*

Maëlle (filleule) : *"J'ai eu la chance de faire partie de l'association des Duchess qui m'a apporté un encadrement dont j'ai besoin. J'ai eu quelques soucis à mon école d'ingénieur et grâce à ma marraine et une Duchess du réseau j'ai eu le soutien et le conseil que je cherchais. Ça fait toujours du bien d'avoir quelqu'un qui nous écoute, nous oriente et surtout qu'il soit connaisseur du milieu professionnel."*

Quelques informations :

- Google Group : 230+ membres (https://groups.google.com/forum/#!forum/duchessfr) ;
- Meetup : 1100+ membres (http://www.meetup.com/fr-FR/Duchess-France-Meetup/) ;
- Twitter : 3500+ abonnés (https://twitter.com/duchessfr) ;
- Site : environ 10 000 pages vues par mois (http://www.duchess-france.org) ;
- 12 événements organisés en 2015 ;
- Présentes dans plusieurs grandes villes : Paris, Lyon, Marseille, Aix, Limoges et Toulouse ;
- Concours : Moov Simply, Code Story, Google Hash Code ...

Suivez nos événements sur l'agenda de notre site et notre compte Twitter @duchessfr.
Devenez membre en vous inscrivant sur notre Google Group.

#AdoptADuchess

L'informatique, une affaire de femmes ?

Duchess France
et remerciements spéciaux
à Duchess Lyon

En France, la communauté des développeurs compte à peine 10% de femmes.

La transformation digitale et le progrès de la mixité dans le domaine de l'informatique fait apparaître une "massive perte de chances" pour la communauté féminine. Les acteurs du système souhaitent néanmoins désespérément féminiser leurs effectifs.

Seulement, le stéréotype de "l'ingénieur informaticien" n'aide pas les femmes à entrer dans un monde qui offre pourtant d'énormes chances d'employabilité.

Les développeuses se sentent souvent pointées du doigt, reconnues pour leur statut de femme et non pour leur capacité technique. Peu crédibles pour la hiérarchie, les femmes sont à des postes quelles ne souhaitent pas, ou décident de partir discrètement vers un domaine où elles pourront s'épanouir pour ce qu'elles font et non pour ce qu'elles sont.

Le premier développeur est une femme

L'AMC (Association for Computing Machinery) a créé le Prix Turing pour récompenser les plus grands informaticiens. Attribué depuis 1966, il aura fallu attendre quarante ans avant de voir une femme récompensée par cet équivalent du prix Nobel en Informatique. Pourquoi les femmes sont elles peu visibles dans l'informatique ? Faisons un retour dans le temps pour revenir au premier développeur de l'histoire : Ada Lovelace.

Ada Lovelace (1815-1852)

Fille d'un célèbre poète britannique Lord Byron et d'une admirable intellectuelle Annabella Millebank, Ada Lovelace acquit une solide formation en mathématiques. A l'époque Victorienne, l'effort intellectuel étant une affaire d'hommes, elle fut observée pour évaluer les conséquences d'une telle formation sur une femme.

A 17 ans, Ada Lovelace fit la rencontre de Charles Babbage, un célèbre mathématicien de l'université de Cambridge. Elle rajoutera des notes à l'article de ce dernier sur la machine analytique de Babbage ce qui triplera sa taille. Ceci lui valut d'être considérée comme le premier programmeur de l'histoire.

Avec cent ans d'avance, Ada Lovelace avait décrit les notions de base de l'informatique telles que les entrées/sorties, l'unité centrale et la mémoire. Elle précise aussi comment utiliser la machine pour gérer nombres, lettres et symboles. Ada était une visionnaire qui a su entrevoir le potentiel de la machine et du calculateur universel, avant les travaux d'Alain Turing. Malheureusement, elle n'est toujours pas citée dans les manuels scolaires de l'informatique.

Une image à changer

Le métier de développeur et d'informaticien au masculin a déjà une mauvaise image, la plupart des personnes l'associent à cette personne solitaire qui parle toute seule devant un PC avec des caractères verts sur un écran noir.

Ce personnage est déjà dévalorisant pour un homme, mais alors qu'une femme choisisse ce métier, impossible ! L'été dernier, une entreprise avait choisi de faire une campagne de recrutement en faisant poser des développeurs de l'entreprise. Parmi ces personnes, une femme : Isis Anchalee. La publicité a fait le tour des réseaux sociaux avec des commentaires très agressif du style : "encore une publicité mensongère pour attirer les mecs", "elle est trop sexy pour développer". Isis a dû écrire un article pour justifier sa propre existence en tant que développeuse et prouver qu'elle travaillait bien dans l'entreprise. Elle a du coup lancé le hashtag #ILookLikeAnEngineer, qui a fait un buzz, en invitant les femmes à montrer qu'il existe des développeuses, et de tout type.

Pour faire changer les mentalités, il faudrait pour commencer que l'on parle réellement du métier, pour les hommes comme pour les femmes. Lorsque l'on explique notre métier à des collégiens, lycéens ou même à des adultes, on se rend compte que ce métier est peu connu alors que la technologie et le numérique sont au coeur des principales activités quotidiennes. Il faudrait aussi arrêter de limiter le développement à "un travail sur ordinateur", parce qu'on le sait tous, concevoir un logiciel, une application, un site, une brique logicielle, ne signifie pas faire du Word ou de l'Excel. Contrairement à ce que l'on pense, le métier de développeur est un métier de création et de communication et il faudrait en parler dès le plus jeune âge.

Il y a deux aspects à mettre en avant pour le

développement : l'aspect créatif d'un côté et l'aspect humain de l'autre. Nous n'utilisons pas les sites Web, les applications pour mobile, nous les fabriquons ! Et nous ne sommes pas seuls devant un écran noir, nous construisons en équipe, en échangeant quotidiennement avec nos clients et en discutant avec les administrateurs systèmes et réseaux qui nous fournissent les plateformes de production. Les jeunes filles aiment voir l'aspect humain dans les métiers, et c'est pour cette raison qu'elles se dirigent souvent vers des métiers autour de la santé ou dans le social. Or dans le développement, le travail se fait le plus souvent en équipe, dans une ambiance conviviale, et c'est un des aspects du métier qui gagnerait vraiment à être plus connu.

Malheureusement, le fait qu'il y ait peu de femmes, contribue aussi à ce qu'il y en ait encore moins. Étant peu nombreuses, elles sont souvent des bêtes curieuses pour les hommes. Il y a soit les misogynes qui remettent en cause en permanence les compétences : "T'as quel diplôme ? T'as fait quelle école ? Toi t'es comme nous, t'es pas une vraie fille". Il y a ceux qui veulent être agréables mais qui sans le faire exprès marginalisent les développeuses : "Une femme ici ? C'est rare, c'est bien que tu les représentes, on aimerait en avoir plus, comment ça se fait que tu sois la seule ?". C'est très gentil, mais nous les développeuses, on adore coder et on ne sait pas pourquoi les autres n'aiment pas cela. Et surtout on aimerait parler technique, d'Angular, de Docker, de Spark et de Spring-boot et pas systématiquement devoir expliquer et presque s'excuser d'être la seule femme ou alors d'avoir la pression au quotidien de représenter sans le vouloir toutes les femmes dans chaque acte, chaque parole. Cette pression peut faire que les femmes ne se sentent pas à leur place, et il arrive que quelques unes quittent totalement l'informatique, ce qui est vraiment dommage.

Avec la venue de solutions offshore, beaucoup d'écoles et de sociétés présentent le métier de développeur comme un métier de petite main ou sans avenir "car les indiens nous prendront notre boulot". Ce qui a beaucoup desservi notre métier, c'est donc tout naturellement que les managers et RH poussent leurs consultant(e)s vers la gestion de projet ou les postes de product owner (PO). Il y a aussi l'expertise technique mais souvent la hiérarchie force les femmes à quitter le monde technique, car leur image ne reflète pas le métier. Cela peut aussi paraître moins crédible, une consultante venue résoudre une fuite mémoire ou faire un audit de sécurité ? La population déjà très faible de femmes développeurs diminue donc encore plus suite à ces réorientations.

L'ordinateur en tant qu'objet reste très masculin, c'est un effet de la société. En effet encore aujourd'hui, autant il est largement accepté que les jeunes garçons passent des heures sur les consoles et les jeux vidéos, autant les jeunes filles qui passent du temps derrière un écran sont déjà considérées comme des anomalies qui font des choses "de garçon", ou sont catalogués de "geek", terme souvent perçu négativement, alors que dans d'autre pays "geek is the new chic".

Et cela se confirme après dans les écoles, beaucoup de jeunes hommes sont là grâce a leur passion pour les jeux vidéos, même si dans leur vie professionnelle plus tard, ils ne travailleront pas du tout dans ce domaine.

Alors comment rendre l'ordinateur mixte ? Comment faire de la programmation une activité pour les filles comme pour les garçons ? Il faut faire découvrir le développement au plus tôt, avant que les stéréotypes ne puissent s'installer dans la tête de nos enfants.

Commencer dés le plus jeune âge

Il existe beaucoup d'initiatives pour démocratiser la programmation : les goûters du code (coding gouter), les mix-teen, devoxx4kids, code.org... Grâce à ces initiatives les filles peuvent se rendre compte qu'elles arrivent à faire aussi bien que les garçons sur un ordinateur, et inversement les garçons dès le plus jeune âge vont se rendre compte qu'il s'agit d'une chose normale, les filles peuvent faire du code. Il existe également des associations comme Ingénieur(e) et technicien(ne) demain (ou encore FACE) qui interviennent dans les lycées pour faire passer le message de façon subliminale que les métiers techniques sont aussi pour les filles : l'association présente le parcours d'étude scientifique dans les études supérieures de façon neutre, mais toutes les intervenantes sont des femmes. Il y a le témoignage d'une femme ingénieure en activité et d'une femme étudiante dans des métiers scientifiques.

Journée e-girl au lycée Descartes

Il y a quelques jours au lycée Descartes à Champs-sur-Marne avait lieu la journée e-girl, Faisant un petit retour sur cette super journée.

Concrètement, le lycée a nommé un référent égalité filles-garçons : Mohammed Grimej qui a organisé une journée e-girl le 21 mars. La journée s'est déroulée en trois temps.

D'abord une conférence donnée par Isabelle Collet - maître de conférence en sciences de l'éducation - pour les professeurs. Plusieurs points ont été évoqués :

- Des taux de mixités paritaires en informatique dans d'autres pays ;
- Évocation de femmes qui ont marqué l'histoire de l'informatique comme Grace Hopper ;
- Des pièges à éviter. A vouloir obtenir plus de mixité on peut provoquer l'effet inverse : en voulant aider spécifiquement une fille en minorité dans un groupe de garçons, en la pointant du doigt on peut provoquer le syndrome de l'imposteur ;
- Un exemple de mise en place de quota qui a bien fonctionné à la Norwegian University of Science and Technology.

L'après midi plusieurs présentations ont été données dans un amphithéâtre devant un public d'étudiants et lycéens. Des témoignages des lycéennes : Cyndie, Sylvie, Audrey, Elif, Manon et Arielle qui ont présenté leur formation, puis les anciennes élèves Berivan, Laurie et Amour ont présenté leurs parcours, et pour finir deux présentations de professionnelles : une tech-lead qui travaille à la Casden et une responsable du groupe L'Oréal. Puis les associations : Prologin, CyberElles et Duchess France ont présenté leurs actions. C'est pour finir Isabelle Collet qui a clôturé la journée avec un talk percutant s'adressant aux garçons de l'assemblée :

"Anecdote : une jeune femme reçoit une remarque déplacée d'un de ses collègues, elle est la seule femme de l'open space. Les autres hommes de l'open space n'ont pas réagi !

Isabelle Collet s'adressant aux garçons présents en amphithéâtre : "Quand vous assistez à un cas comme ça, montrez votre désaccord, ou au moins donnez un mot de soutien à votre collègue qui se sentira moins isolée".

Éviter la fuite des cerveaux féminins

Alors que peut-on faire dans les entreprises pour garder les femmes qui y sont ?

Il faudrait pour commencer rendre l'environnement neutre. Il ne devrait plus y avoir de femmes dénudées en guise d'écran de chargement, ou sur des sites de monitoring d'application par exemple.

Utiliser un vocabulaire neutre (non genré) quand c'est possible. Dans les annonces de recrutement par exemple : "nous recrutons une personne, cette personne doit connaître..." et pas "nous recrutons un développeur...il doit connaître...il....il". Ou expliciter les deux genres : un-e développeu-se-r. Il faudrait que les RH et les manager arrêtent de forcer les femmes à tenir des postes qu'elles ne souhaitent pas faire : c'est pas parce qu'elle savent communiquer qu'il faut les mettre chef de projet ou fonctionnelle. Lorsque l'on aime développer, on est très malheureuse à ces postes.

De la même façon, ne pas pousser systématiquement les développeuses vers les tâches de présentation/de graphisme (chartes graphiques, css, html) parce que l'on suppose qu'elles savent accorder les couleurs, que c'est "dans leurs gènes"...

Au contraire, si vous avez la chance d'avoir des développeuses dans votre équipe, encouragez-les à prendre des challenges, à progresser et à se mettre en avant.

Pour celles qui ne supportent plus d'être pointées du doigt, ou pour celles qui ne trouvent plus leur place, l'entrepreneuriat est souvent un eldorado pour l'émancipation des développeuses. Même si seulement 10% des startups sont créées par des femmes en France, ce chiffre a tendance à s'améliorer notamment grâce aux actions des structures d'accompagnement comme lespionnieres.org mais aussi le gouvernement qui a compris que le numérique passe par la construction de nouveaux modèles, et une nouvelle image.

Axelle Lemaire, secrétaire d'état en charge du numérique succède à Fleur Pellerin. Cette succession de femmes à ce poste n'est pas un hasard car les femmes y ont leur place et sont le moteur dans la révolution numérique.

Conclusion

Le code n'a pas de sexe. Il n'y a pas de cerveau masculin ou de cerveau féminin. Les développeuses s'épanouissent dans leur travail au quotidien si on ne les pointe pas du doigt ou qu'on ne les rabaisse pas, ce sont des développeurs comme les autres.

Cet article traite de la place des femmes dans les métiers de l'informatique, mais il ne faut pas oublier les autres groupes sous-représentés dans ces métiers. Donc plus de mixité c'est bien, mais plus de diversité c'est mieux !

Elles se sont lancées !

Grâce à Duchess France, plusieurs oratrices ont osé et se sont lancées. Elles vous livrent leur vécu, leur retour et leur expérience.

Aurélie Vache

Développeuse Web Full-Stack chez atchikservices

Fin novembre dernier je me suis lancée. J'ai présenté la technologie Google BigQuery, une techno liée à la Big Data, lors d'un meetup du Toulouse Data Science. Le talk a duré environ 45 minutes et il y a eu énormément de questions à la fin.

Un peu moins d'un an auparavant, lors d'un meetup, on m'a demandé si je ne voulais pas faire un talk. Ma première réaction a été : Moi, faire un talk ? Euh, vous êtes sûr ?

A la base, faire un talk, pour moi c'était une idée inimaginable. Je bégaie et je pensais que je n'allais jamais être capable de le faire.

On m'a re-demandé plusieurs fois si je voulais faire un talk ; j'avais un sujet, donc l'idée a fini par mûrir dans ma tête, j'ai accepté. J'ai commencé par me pencher sur les slides en premier, que j'ai re-travaillées encore et encore, et, ensuite, je me suis attelée au discours que j'aurais pour chacune des slides. L'idée était d'avoir une trame qui allait me donner un rythme. Même si je n'allais pas réciter mon discours par coeur, il fallait que j'en apprenne une grande partie. J'ai répété plusieurs fois chez moi, je révisais même dans le métro.

Je ne vous cache pas que plus la date approchait et plus le stress montait. Parler avec les duchess m'a fait du bien, car je me suis rendu compte que même les speakeuses habituées aux confs stressaient, que le stress était une réaction normale, c'est comme cela que le corps se prépare.

Le Jour J arriva. La veille j'avais testé les slides dans la salle, la présentation s'affichait correctement, cela rassure MAIS le plus dur restait à faire : présenter devant des personnes.

Les premières minutes ont été dures mais au fur et à mesure la fluidité est venue et j'ai même mis 2-3 touches d'humour. Faire rire volontairement le public c'est une sensation assez agréable et les entendre applaudir à la fin, encore plus.

Cette petite expérience m'a fait prendre un peu plus confiance en moi. J'ai été capable de présenter une technologie, de partager mes connaissances.

Oui le fait de bégayer m'a rajouté du stress et du travail lors de ma préparation mais au final j'ai aimé partagé mes connaissances et même si je ne me sentais pas prête pour proposer mon talk a Devoxx cette année, j'aimerai bien en refaire à l'avenir.

Si vous avez un sujet et que vous souhaitez le faire, lancez-vous, vous êtes légitimes !

Stéphanie Moallic

Développeuse Web & Mobile en freelance

Lorsque j'ai commencé à développer sous Ionic, on m'a demandé de faire une présentation de ce framework lors d'une session du FinistJUG. Ça faisait un moment que le responsable du FinistJUG me demandait si je ne voulais pas me lancer et parler d'un sujet technique qui me tenait à coeur. Cette fois-là, je lui ai répondu OK et je me suis lancée, j'ai fait une répétition devant lui et plusieurs personnes de sa boîte avec les slides qui passaient sur leur téléviseur.

Le plus dur pour moi a été de démarrer mon discours, j'ai recommencé plusieurs fois mon lancement car je bloquais mais une fois passée cette difficulté, ça se déroulait tout seul. Le jour J, j'ai eu un problème technique (je ne pouvais pas lancer les slides depuis mon PC), j'ai donc basculé sur une autre machine que je ne connaissais pas (sachant que que je faisais une partie de live coding) et j'ai commencé ma présentation. Comme j'avais peur d'avoir les mêmes soucis que lors de la répétition, j'avais écrit le début de mon discours et ensuite tout a roulé tout seul. La première fois passée, j'ai soumis une proposition pour Devoxx France qui a été acceptée. Là, j'ai été prise de trac : un amphi où le talk était filmé, ça n'a pas été ma meilleure performance mais ça m'a permis de mieux appréhender mon trac les fois suivantes, même si parfois, j'ai eu encore des problèmes techniques (coupure électrique, plus de micro, difficulté à trouver la salle, oubli d'une commande...). Mais ça met un peu de piquant à l'exercice. Maintenant, pour chaque talk, je répète mon intro ce qui me permet de me sentir plus à l'aise et de moins stresser, j'ai toujours un peu chaud, mais je me sens à l'aise et j'apprécie de plus en plus l'exercice de partager des retours ou des sujets plus techniques avec le public.

Pauline Iogna

Tech Lead à la Casden

« En ce qui me concerne, être speaker était loin d'être une vocation. Les duchesses m'ont incité à plusieurs reprises à présenter un sujet et... j'ai fini par accepter.

J'ai commencé par préparer un quickie (présentation de 10-15min). Un format court est idéal pour commencer, c'est plus rassurant.

J'ai écrit mes premières slides et le premier jet n'était pas terrible. Après une première relecture, je choisis de proposer ma présentation aux Human Talks, elle est sélectionnée ! Dans la foulée, je la propose également à Devoxx France sans y croire vraiment. À ma grande surprise : la présentation a été également acceptée !

La plus grande difficulté rencontrée a été de fluidifier mon discours. Mais avec une bonne préparation, on gagne en aisance et en maîtrise du sujet. Raconter une histoire, suivre un fil conducteur donnera de la clarté à au discours. Présenter votre talk à un interlocuteur qui ne connaît pas du tout le sujet est un bon exercice, cela apportera de la pédagogie à la présentation. Faites relire votre présentation par vos collègues ou responsables techniques.

Participer aux ateliers de préparations organisés par Duchess France est aussi un excellent entraînement.

Pas besoin d'être un expert dans tous les domaines pour mériter sa place dans les conférences. Si vous avez un sujet attractif, que vous le maîtrisez et avez envie de partager votre expérience, alors devenez speaker.

Une fois passé le premier talk, il ne faut pas hésiter à faire le tour des conférences.

C'est aussi l'occasion d'adapter sa présentation pour passer sur de nouveaux formats.

Aujourd'hui je ne regrette pas de m'être lancée. Depuis, je continue régulièrement de proposer des sujets.

Certains ont été refusés, d'autres ont été acceptés. Plus on en fait, plus on apprécie. »

Amira Lakhal

Développeuse Agile Java chez Valtech

« Lors de mon cursus scolaire, j'ai fait des présentations diverses. J'étais très contente de montrer mon travail à toute la classe. Mais en grandissant, je perdis cette fierté et l'angoisse du public s'installa petit à petit.

Ayant acquis des compétences dans le cadre professionnel, j'ai voulu partager mes connaissances. Pour ma première présentation, j'ai co-animé un atelier technique avec deux experts en la matière. Je me souviens d'avoir été très stressée. Heureusement, le fait d'avoir co-animé m'a permis d'avoir un temps de parole partagé.

Si vous voulez présenter un sujet, faîtes-le en binôme la première fois! Après cette première expérience, j'ai gagné plus de confiance en moi et j'ai animé d'autres ateliers, toujours à plusieurs. J'ai aussi animé des sessions seule mais les sujets étaient généralises et simples. C'était l'occasion d'affronter la peur du public en solo.

Essayez d'animer des présentations sur des sujets généralistes et simples!

Ayant appris à faire face à un public, j'avais envie d'animer une présentation technique. Première étape: il fallait trouver un sujet qui me plaise et où je me sente à l'aise.

Dès que le sujet est défini, il faut trouver une conférence, et, ensuite envoyer la soumission du sujet. Si le sujet n'est pas pris, il ne faut pas se dire qu'il n'est pas bien, mais il faut retenter sa chance ailleurs et plus précisément dans les événements communautaires. Ce sont des soirées qui se déroulent avec un public variant de 30 à 200 personnes et on peut avoir énormément de retours constructifs afin d'améliorer l'idée initiale.

Présentez des sujets aux événements techniques communautaires!

Et vint le jour où ma soumission a été acceptée à une grande conférence. J'étais consciente de l'opportunité qui s'offrait à moi, il ne fallait pas la gâcher. J'ai mis en place un planning pour préparer mon sujet. J'ai répété en petit comité et dans des événements communautaires. Suite aux retours très constructifs, j'ai remanié encore ma présentation.

Le jour J arriva, j'ai peu dormi, le stress était à son max. Et j'ai animé ma première présentation technique seule, dans une conférence internationale le tout en anglais. Une vraie première pour moi. Passés les premiers instants, le stress a disparu et je me sentais de plus en plus à l'aise.

Résultat : ma présentation a eu tellement de succès que j'ai été invitée à non pas une mais plusieurs conférences internationales. Je suis devenue plus à l'aise et ai oublié mes anciennes peurs et appréhensions. J'éprouve un grand plaisir à partager ce que j'ai appris en espérant aider d'autres personnes ou même créer des vocations.

Lancez-vous, partagez vos connaissances! »

Deux duchesses au Sénat

L'histoire a commencé par un email reçu en juin en provenance du Sénat. Ils avaient besoin de recruter 2 informaticiens de profil développeur pour compléter leur équipe à la DSI. Pour ce faire, et comme pour d'autres profils, ils organisent un concours et cherchent des personnes externes pouvant être membres du jury.

Aurélie Vache
Développeuse Web Full-Stack chez atchikservices
Duchess France Leader
@aurelievache

Agnès Crépet
Java Champion
Co-fondatrice de Ninja Squad
Co-fondatrice de la conférence Mix-IT
Duchess France Leader
@agnes_crepet

Le hasard a bien fait les choses, car nous avons été contactées pour faire partie de ce jury, étant toutes deux membres du board de Duchess France. Trois autres personnes hors Sénat ont été contactées pour faire partie du jury : une professeure à l'Université Paris-Diderot, un professeur à l'Université Paris-Sud ainsi qu'un consultant en recrutement. Quelques chiffres avant de vous raconter les coulisses de ce concours :

- 102 candidats préinscrits
- 62 candidats inscrits ;
- 57 hommes (92%) ;
- 46 candidats présents lors de la 1ère épreuve (74.2%) ;
- 43 candidats présents lors de la 2ème épreuve (69.4%) ;
- Diplôme : 10% de doctorat, 35% Master 2 / DESS / DEA, 45% école d'ingénieur, 10% autre.

Réunion de jury du 23 octobre 2015

Grâce un tour de table nous avons pu nous apercevoir que la parité était respectée et qu'il y avait 5 personnes extérieures au Sénat sur les 10 membres.

La première réunion avait pour but de faire un point sur les candidatures, l'organisation et la répartition des épreuves, et de déterminer les épreuves d'admissibilité. C'est là que nous avons été assez surprises du protocole de recrutement et de "l'ambiance" : au Sénat, la DSI ouvre la séance et c'est elle qui la ferme. Entre collègues, beaucoup se vouvoient. On a nos petits noms sur des étiquettes, et tous les membres du Sénat sont super bien habillés, on sent bien que nous ne sommes pas chez Google ("you can be serious without a suit") et heureusement qu'on n'avait pas mis nos baskets ! :-)

Les épreuves d'admissibilité devaient comporter un Quiz, des questions d'algorithmique ainsi que de gestion de projet. Pour les questions algorithmiques, le candidat pouvait répondre soit en Java soit en C.

Chaque membre du jury a pu donner son avis lors de l'élaboration des épreuves d'admissibilité, cette complémentarité s'est révélée cruciale lors de toutes les étapes de ce concours.

Entre les deux réunions de jury, certains membres du jury ont dû corriger les épreuves d'admissibilité, et d'autres ont dû rédiger et/ou donner leur avis/leurs remarques sur les épreuves d'admission.

Réunion de jury du 18 décembre 2015

La deuxième réunion avait comme ordre du jour :

- L'examen des résultats des épreuves d'admissibilité ;
- La fixation de la liste des candidats admissibles ;
- L'adoption des sujets de l'épreuve orale d'admission portant sur des épreuves techniques ;
- L'organisation des épreuves d'admission ainsi que les modalités d'interrogation des candidats à l'oral.

Après évaluation des notes des candidats, nous avons choisi d'en garder 12 et nous avons organisé les épreuves d'admission en conséquence.

Les 4 sujets choisis pour les épreuves d'admission étaient variés, ils faisaient appel à des connaissances de développement Web, d'applications mobiles, de bases de données, de problèmes NP complexes, de hashage, d'annuaire LDAP, de clés privées et publiques, de gestion de projet, d'agilité...

Épreuves orales d'admission en janvier 2016

Cette fois-ci nous avons enfin rencontré les 12 candidats : 2 journées intenses qui démarraient à 7h45 et se terminaient vers 19h30.

Pendant la première journée chaque candidat avait 20 minutes pour préparer un sujet imposé et 10 minutes pour nous le présenter. A la fin de sa présentation, une question algorithmique lui était posée ainsi que d'autres questions, il avait 30 minutes pour y répondre.

Cette première journée nous a permis de rencontrer les candidats et de juger notamment leurs compétentes techniques.

Le vendredi nous avons fait passer un entretien libre de 30 minutes aux candidats vus la veille. Chaque candidat devait exposer une expérience de son choix puis nous avons posé des questions diverses et variées, notamment techniques si nous avions un doute sur le candidat à l'issue de sa prestation de la veille.

Le niveau d'études des candidats était diversifié : parmi les 12 candidats il y en avait 6 qui avaient fait une école d'ingénieur, 4 Master 2/DESS/DEA et 2 doctorats.

La force du jury, avec sa mixité, des personnes compétentes et complémentaires, a fait que les délibérations finales n'ont pas été houleuses et n'ont pas duré une éternité. 4 personnes se sont distinguées du reste des candidats, l'entretien libre puis les échanges entre membres du jury ayant permis de nous mettre tous d'accord sur la pertinence des profils retenus par rapport aux fonctions recherchées au Sénat.

Conclusion

Cela a été une expérience très enrichissante. Faire partie d'un jury d'un concours spécial développeurs, d'assister à toutes les phases de l'aide à la création des sujets, à la rencontre des candidats et enfin à la délibération finale.

Au premier abord on peut se dire que travailler pour le Sénat cela peut être "poussiéreux", on peut penser que les développeurs qui y travaillent utilisent des vieilles technologies. En réalité ce n'est pas le cas. Nous avons eu la chance de pouvoir parler aux membres de la DSI du Sénat. La diversité des problèmes et des solutions qu'ils doivent mettre en œuvre au quotidien fait que les développeurs au Sénat ne peuvent pas s'embêter et doivent utiliser des technologies diverses, variées et au goût du jour.

Ils n'utilisent pas la dernière version de Java (après notre passage, on espère bien qu'ils migrent à Java 8 :p) mais ils utilisent les méthodes agiles et font de la veille technologique.

Nous avons aimé le fait que le Sénat concocte un jury mixte, la parité étant respectée, et fasse appel à des professionnels externes.

Ils ont mis tous les moyens en œuvre pour ce recrutement, ont investi pour faire venir des personnes extérieures et cela a permis l'élaboration d'un concours de qualité. Et si cela vous dit de travailler pour une institution publique, lancez-vous ! Il y a vraiment matière à s'éclater ! :) ⊞

Zen to Done : comment organiser sa vie de geekette ?

En tant que développeur, on peut facilement s'investir dans de multiples projets. Et qui parmi nous n'a pas au moins un projet vide initié ou un projet forké sans nouvelle modification sur Github ?
Zen To Done est une méthode d'organisation créée par Leo Babuta dans le but d'améliorer la méthode GTD (Getting Things Done) de David Allen. Car GTD se concentre trop sur les tâches à réaliser tout de suite, et moins sur nos objectifs à long terme. Il n'est également pas suffisamment structuré pour des gens qui ont besoin de plus d'organisation dans leur journée et qui ont besoin de temps pour intégrer une habitude.

© coffeee-in

Ellène Dijoux Siber
Développeur freelance
Duchess France Leader
@ElleneSiber

Avant de lister les 10 habitudes, il faut savoir que la plupart des gens ont besoin d'environ un mois pour qu'une tâche devienne une habitude. Leo Babuta recommande d'adopter une habitude à la fois si possible et surtout ne pas tenter plus de 3 habitudes à la fois.

1 • Recueillir

Notre cerveau n'est jamais au repos, beaucoup d'idées fusent tout le temps, les mails et les papiers s'entassent. Nous nous engageons sur des projets, notons des e-mails ou numéros de téléphone alors que nous avons l'esprit déjà occupé. Et souvent nous oublions toutes ces informations au moment où elles sont indispensables.
Le but de cette habitude est donc de définir un seul endroit pour recueillir tous les papiers et de noter systématiquement toutes les choses auxquelles on pense.
Ainsi au boulot, vous pouvez utiliser un carnet pour y noter tout ce qui vous passe par la tête :

- "Faire la revue du code de Nicolas",
- "Envoyer le fichier de configuration à Vincent" ...

Bref tout ce qui peut vous alléger l'esprit. Evitez les Post-its collés sur votre écran, les feuilles volantes sur votre bureau ou "rangées" dans votre tiroir. Si votre métier requiert beaucoup de paperasses, ayez une corbeille à papier pour tout collecter.
Il vous faudra également une corbeille à courrier chez vous. Dans laquelle vous collecterez tout : les factures, notes de frais, le courrier reçu, les documents administratifs, vraiment tout. C'est là une très bonne méthode pour ne plus rien perdre.
Pour les e-mails, l'idéal est de n'avoir qu'une seule boîte de réception, en tout cas le moins possible. Les forums, les réseaux sociaux et professionnels devront être traités comme des e-mails également : moins on en aura, plus simple ce sera.
Pour tout le reste ayez toujours quelque chose sur lequel noter tout ce qui vous passe par la tête, les projets auxquels vous aspirez et tout ce que l'on vous demande de faire, d'acheter. Le plus simple est d'utiliser un carnet de note et stylo qui ne nécessite ni réseau, ni batterie et est transportable partout mais libre à vous de choisir. Et surtout emmenez-le partout avec vous car les bonnes idées arrivent au moment où on s'y attend le moins ! Cette première étape vous permettra de ne plus oublier, ni perdre et de vous alléger l'esprit.

2 • Traiter

Si vous avez intégré la première habitude, vous pouvez très vite vous sentir débordé par cette pile d'éléments qui grossit. Pas de panique ! Cette habitude vous apprendra à traiter toutes ces notes et documents que vous avez rigoureusement collectés. Le but est donc de traiter chaque élément un à un et tout devra être traité. Pour ce faire, vous allez devoir prendre des décisions très rapides sans tergiverser :

- Commencez toujours par le dessus de la pile

et prenez une décision rapide, ne le mettez pas de côté ou en dessous de la pile.
- Jeter doit être le choix par défaut surtout si vous n'en avez pas besoin.
- Déléguez. On vous a proposé de présenter votre métier à un forum d'étudiant mais vous n'êtes pas inspiré alors proposez-le à quelqu'un de plus motivé. Et surtout n'oubliez pas d'archiver ou de supprimer le document !
- Si la tâche prend moins de deux minutes, faîtes-le tout de suite puis archivez ou jetez le document.
- Si l'action prend plus de deux minutes, planifiez-la.
- Archivez si c'est quelque chose dont vous aurez besoin pour référence comme un numéro de licence ou une note de frais par exemple. Prenez un classeur à séparateur et éviter les classeurs génériques du type "Divers" ou "A classer"
- Ne laissez donc jamais un élément dans votre boîte de réception : jetez ou archivez. Si vous en avez vraiment beaucoup, planifiez quelques heures pour tout traiter.
- Répétez ce processus pour garder votre boîte de réception vide. Ne le faites pas non plus de manière obsessionnelle ! Et appréciez d'avoir un "Zero Inbox".

Vérifiez et traitez vos boites de réception au moins une fois par jour pour éviter que les choses s'accumulent.

3 • Planifier

Nous sommes malheureusement souvent exposés aux urgences, que ce soit au boulot ou à la maison. Celles-ci nous poussent à y répondre au lieu de prendre du temps pour analyser leur importance. Pour ce faire, concentrez-vous d'abord sur les tâches qui sont vraiment importantes à vos yeux que l'on appelle également « les grosses pierres ». Planifiez-les dans votre agenda de la semaine puis comblez le reste avec les plus petites tâches. Puis tous les matins définissez quelles sont les tâches les plus importantes de votre journée et commencez toujours par celles-ci le plus tôt possible dans la journée. Plus tôt, elles seront faites et moins les urgences de dernière minute de la journée vous freineront. Une fois faites, appréciez le travail accompli et pourquoi pas, récompensez-vous.

4 • Faire

Faîtes une tâche à la fois et évitez de paralléliser. A l'abri de toute nuisance et dans un endroit épuré, minutez-vous tel un pomodoro pour réaliser vos tâches. Lorsqu'une urgence, une interruption ou une envie se présente, notez-là dans votre carnet, histoire de vous libérer l'esprit. Si vraiment vous ne pouvez pas passer outre cette interruption, notez où vous en étiez pour ne pas perdre le fil la fois suivante.

5 • Toujours faire simple

Il est important de faire une liste contextuelle avec un système que vous vous approprierez facilement. Pourquoi a-t-on besoin de contexte ? Parce que nous ne pouvons réaliser les mêmes tâches à la maison et au travail par exemple. On pourrait donc séparer les tâches de la façon suivante :

- travail, pour tout ce qui concerne le travail ;
- personnel, pour tout ce qui concerne la maison ;
- courses, pour votre liste de courses ;
- téléphone, pour les coup de fil à passer ;
- en attente, pour celles en attente et à ne pas oublier.

Et pourquoi pas un @projet, pour les projets que vous ne pouvez pas réaliser maintenant mais un jour peut-être ! L'essentiel est d'avoir une liste contextuelle à votre image. Pour gérer cette liste, vous pouvez utiliser des outils mais surtout rester simple : un carnet de notes peut suffire avec Google Agenda, et, pour les papiers, un système de dossier pourrait faire l'affaire.
Dans la pratique, prenez l'habitude de passer en revue vos listes quotidiennement. Cela doit faire partie de vos habitudes.

6 • Organiser

Le rangement est un des piliers du Zen To Done, car un bureau rangé/épuré vous aidera à vous libérer l'esprit et vous concentrer sur vos actions. Vous diminuerez aussi le risque de perdre des documents. La règle de cette habitude est donc de tout ranger à sa place. Trouvez une place pour chaque chose et rangez immédiatement au bon endroit. Faîtes-en une habitude : concentrez-vous 30 jours de suite sur cette habitude jusqu'à ce qu'elle devienne un automatisme. Gardez toujours vos surfaces dégagées pour avoir envie de travailler à nouveau sur votre bureau. Une fois cette habitude intégrée vous pouvez maintenant revoir l'organisation de vos rangements et pensez à vous débarrasser du superflu.

7 • Examiner

Chaque semaine vous allez examiner vos objectifs. Commencez par choisir un objectif à long terme et un autre à court terme, celle de la semaine, un suffit largement pour débuter, vous pourrez en choisir plus lorsque vous aurez pris confiance en vous et assimilé cette habitude. Passez aussi en revue votre agenda pour la semaine passée et la semaine à venir pour vérifier qu'un évènement précédent n'en entraine pas un autre et pour bien avoir en tête ce qui vous attend dans la semaine. Passez en revue vos listes et rayez les tâches déjà réalisées.

8 • Simplifier

En tant que développeur, nous avons beaucoup d'idées géniales pour changer le monde ! Mais la longueur de la liste des choses serait tellement longue que ça en devient décourageant surtout quand on doit jongler avec le boulot, les communautés et la vie personnelle. Et comme nous n'avons que 24 heures dans une journée, il va falloir donc apprendre à éliminer les tâches superflues et déterminer ce qui est important et en rapport avec vos objectifs. Réduisez vos engagements et apprenez à dire non pour ne garder que les choses qui vous tiennent vraiment à coeur. Réduisez vos abonnements aux newsletters et autres publicités pour y voir un peu plus clair dans vos mails. Gardez surtout les tâches à forte valeur ajoutée, celles qui vous aideront à atteindre vos objectifs.

9 • Routines

Créer des routines ou des rituels quotidiens et hebdomadaires vous aideront à mieux contrôler votre journée et à faire face aux imprévus. Pour commencer, vous pouvez faire la liste des tâches professionnelles et personnelles qui vous paraissent vraiment importantes comme "vérifier la couverture du code par les tests" ou encore "faire la litière du chat". Regroupez ensuite toutes ces tâches et détectez celles que vous réalisez quotidiennement et plusieurs fois par semaine. Répartissez-les dans la semaine (mais pas plus de 3 tâches par jour !) et testez ces routines la semaine suivante. Une fois la semaine passée : examinez ce qui n'a pas fonctionné et simplifiez si vous avez voulu en faire trop. Trouvez votre rythme de croisière et ne lâchez pas ces routines 30 jours durant. Le temps nécessaire pour qu'elles s'intègrent dans votre quotidien.

10 • Trouver votre passion

Sa passion ? Mais on parle de méthode d'organisation ici ! Eh bien tout simplement parce que la passion est le premier carburant de notre motivation à travailler. Sans cette motivation, vous aurez beau adopter les 9 habitudes précédentes, cela ne vous rendra pas plus heureux. Alors cherchez votre passion et demandez-vous ce que vous aimez déjà faire, lire. Si vous avez une idée, testez-la au plus vite et expérimentez-la jusqu'à trouver votre graal. ⊞

Pour aller plus loin :

http://www.mamansorganise.com/zen-to-done-version-minimaliste/
http://www.habitudes-zen.fr/2009/zen-to-done-ztd-lultime-systeme-simple-de-productivite/

Le métier de CTO décrypté pour vous

CTO de SoFizz, l'appli mobile destinée aux femmes qui veulent partager leurs activités : pour trouver des personnes pour prendre un verre, aller à une expo, en soirée, courir... et pouvoir profiter de bon plans !

Mathilde Lemée
Duchess France Leader

Si tu es un tant soit peu visible sur Linkedin, tu as déjà dû recevoir des offres pour être 'CTO' dans une startup. Ça peut paraître un peu abstrait au début. Voilà donc les questions les plus fréquemment posées !

Mais c'est quoi un CTO ?

Déjà, il n'y a pas un rôle type pour le poste de CTO ! Le CTO ou Chief Technical Officer est en fait un acronyme qui définit un peu tout et n'importe quoi. C'est, dans une startup, le responsable technique. Immédiatement en dessous du CEO (Chief Executive Officer), le n° 1 de l'entreprise, c'est un des piliers de la startup. On va l'appeler le DSI (Directeur des Systèmes d'Information) ou Directeur Technique.

Et il fait quoi exactement comme job ?

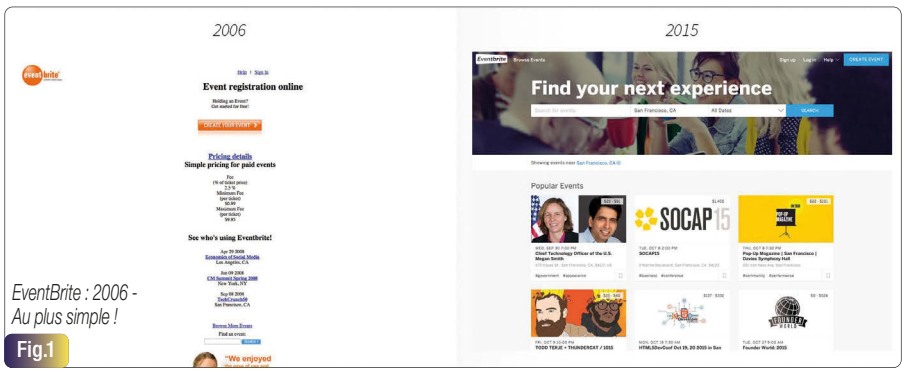
Chaque startup est différente. En fonction de l'expérience du CTO, du CEO, l'avancement du produit, les financements, le rôle du CTO va être adapté... On peut voir 3 tendances, en fonction de la maturité de la startup : Fig.1.

Les premiers pas

Le premier job d'un CTO dans une startup qui débute est finalement assez similaire à un job de lead developer auquel on va demander en plus une connaissance métier et produit avancée. Il aura pour mission de choisir les frameworks utilisés, l'architecture globale du projet et surtout ... de les mettre en place. Mettre à jour les updates de sécurité, fixer les anomalies, développer les fonctionnalités... Sauf qu'à la différence d'un travail de lead développer dans un grand compte, le code produit n'est pas supposé rester. C'est souvent synonyme de patch 'à l'arrache'... Le projet est ajusté à chaque étape (alpha, beta, lancement...), parfois (comprendre souvent) en profondeur.

C'est le stade du **Minimum Viable Product :** construire le plus petit produit possible mais viable pour aller à la rencontre du marché le plus tôt possible et de pouvoir pivoter pour construire THE produit. Pour un développeur, ça veut dire ne pas chercher à créer l'architecture parfaite, le produit parfait ou un produit qui scale parfaitement et surtout ne pas être amoureux de son code ! Mais au contraire, aller vite, mettre en avant ce qui peut bouger facilement et accepter de mettre tout (ou une bonne partie) de son code à la poubelle.

Être CTO dans une startup qui se lance, c'est bien sûr une réalisation technique mais aussi un travail sur le produit : le définir, proposer des améliorations, savoir chiffrer le développement des fonctionnalités, proposer des outils et des indicateurs pour comprendre l'utilisation qu'en font les premiers clients ... L'important n'est pas de construire le produit tel qu'il l'a été dans l'idée de départ, mais celui qui correspond au marché.



EventBrite : 2006 - Au plus simple !

Fig.1

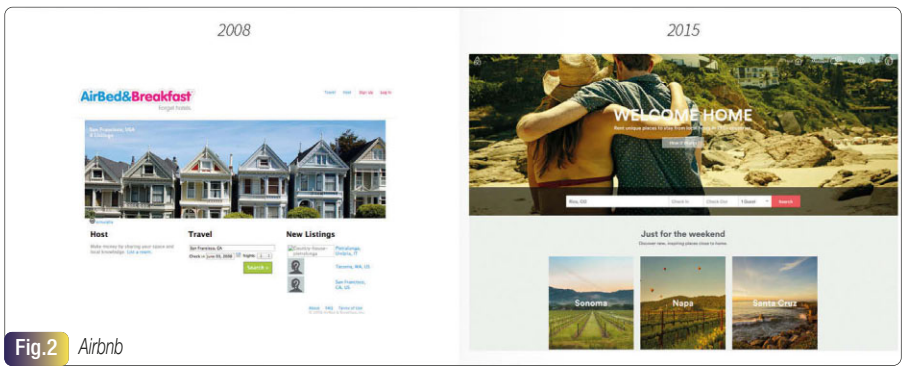


Fig.2 *Airbnb*

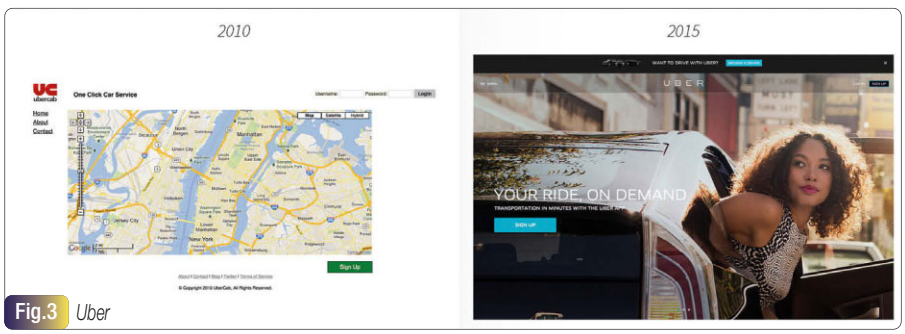


Fig.3 *Uber*

L'adolescence

Le projet a commencé à rencontrer son marché, la startup a de plus en plus de clients. L'équipe technique s'agrandit. Le CTO qui était jusqu'à présent seul commence à devoir gérer une petite équipe et à se poser des questions sur le moyen terme. Ce qui se traduit par la nécessité de faire des tâches très diverses : recruter, gérer le planning, faire le suivi RH de son équipe tout en continuant à participer à la définition du produit et à son évolution ... On commence à mettre en place des codes reviews, refaire des pans entiers de fonctionnalités pour les rendre maintenables et scalables. Le CTO a appris à s'entourer d'experts plus compétents que lui ! Il continue encore à coder un peu mais de moins

en moins au fur et à mesure que l'équipe technique grossit. Il fait beaucoup de veille pour voir comment les nouvelles technologies peuvent aider son projet. Fig.2.

La maturité

C'est un DSI. Il est là pour donner une vision, un angle et faire en sorte de mener la barque dans la bonne direction. Il peut être un ancien développeur, ou non, mais il reste le garant de la culture technique. Fig.3.

Est-ce qu'il faut être spécialiste pour être CTO ?

Rarement ! Le rôle premier du CTO sera de développer le produit minimum viable. Donc il doit savoir coder s'il n'y a que 3 personnes dans la startup ! On lui demandera donc d'être plutôt touche à tout. Cela sera ensuite à lui de s'entourer de développeurs plus experts que lui (des développeurs back, des développeurs front, des ops ...), de faciliter le passage de connaissance et la diffusion des bonnes pratiques. Sans oublier de former le CEO et de vulgariser pour l'aider à prendre les bonnes décisions.

Est-ce que cela demande un gros investissement humain ?

Oui. Surtout en phase 1. Le temps est précieux au début, c'est important de pouvoir aider sa startup à aller le plus vite possible à la rencontre de son marché. Et comme l'équipe sera minimale, cela sera compliqué de décrocher. Au fur et à mesure, on commence à avoir assez de temps pour gérer tout un peu mieux (plus de tests, redondance des serveurs, systèmes de monitoring, test de perfs ...) et on devient plus serein. Malgré l'investissement, il n'est pas incompatible avec une vie de famille, car dans tous les cas, cela reste important de faire des 'breaks', sous peine de burn-out !

Est-ce que le CTO a forcément des parts ?

Pas forcément mais la plupart du temps oui. En fonction du moment où le CTO arrive sur le projet, il pourra négocier plus ou moins de parts. Si l'idée et les investissements de départ sont communs, habituellement le partage des parts est identique. Si le CTO est trouvé, après une v1 faite par un stagiaire ou une agence de com, cela sera encore moins, idem si c'est après une levée ou si un des associés met beaucoup plus que les autres au capital. On peut compter entre 15 et 30% de parts pour le CTO. C'est une fourchette à adapter ! En général, on évite le 50-50, pour éviter le blocage. Tout est question de négociation ! Il est courant d'attribuer cela via des BSPCE (BONS DE SOUSCRIPTION DE PARTS DE CREATEUR D'ENTREPRISE) si la société a déjà été créée qui seront soumis à des conditions (5% pour la première milestone, 10% pour la suivante ...).

INTERVIEW DE KATHRYN GREER CTO chez SmartAngels

Plateforme de crowdfunding qui permet aux investisseurs particuliers comme professionnels de financer des startups et PME de croissance en capital.

Peux-tu te présenter ?

Je suis Kathryn Greer, j'ai 25 ans, ancienne étudiante d'Epitech et je suis Directrice Technique chez SmartAngels depuis un peu plus d'un an. Avant ça, j'étais développeuse Symfony chez SensioLabs.

Tu as fait le choix de rejoindre SmartAngels avec 2 ans d'expérience, peux-tu nous expliquer ton cheminement ?

Je ne cherchais pas particulièrement à partir de mon ancienne entreprise chez qui je me plaisais aussi, mais une connaissance qui cherchait un CTO pour SmartAngels m'a parlé de la startup et j'ai été immédiatement séduite. En effet, le crowdfunding est un sujet qui m'a toujours attirée. Aussi, le fait de devoir surmonter tous les challenges d'un projet, ainsi que de le voir évoluer au fil du temps, me plaisait beaucoup. Je ne regrette pas mon choix car cela m'a permis d'apprendre énormément de choses que je n'aurais pas eu l'occasion de voir dans mes précédents emplois ; et je suis ravie de côtoyer quotidiennement la super équipe de SmartAngels.

Comment se déroule une de tes journées types ? Pour toi c'est quoi le métier de CTO ?

Chaque jour chez SmartAngels est différent ! En moyenne, une journée commence par le petit point quotidien avec les développeurs, puis j'enchaîne avec une mise en prod, ou un entretien avec un candidat par exemple. Je m'occupe aussi de préparer les sprints et les côtés techniques des projets en général avec les chefs de projets. Je passe beaucoup de temps en R & D de différentes technologies ou techniques pour anticiper les besoins de SmartAngels qui grandit vite (en dev et/ou devops) !

Le métier de CTO est différent d'une entreprise à l'autre, car nous devons tous gérer des problématiques différentes. Si je devais généraliser, un CTO dans une startup française a comme mission de :

- Créer et motiver continuellement la meilleure équipe technique reflétant les besoins de la société ;
- Connaître tous les aspects techniques de la start-up afin de pouvoir répondre à tous les besoins métiers dans les préparations des projets ;
- Savoir anticiper les besoins techniques de la startup, de faire les recherches et développements en conséquence avec pour but de transmettre les nouvelles connaissances à l'équipe ;
- Représenter l'image et l'esprit de l'entreprise dans la communauté technique ;
- Apprendre et se remettre en question continuellement, afin de mieux avancer.

Tes 5 outils/frameworks préférés ?

Je n'ai pas vraiment de préférés, mais voici ceux que j'utilise le plus souvent...

Dans le désordre :

- PHPStorm
- Symfony2
- AngularJS
- Github
- RabbitMQ

Un message à faire passer à ceux qui n'osent pas se lancer dans une startup ?

N'ayez crainte ! Vous rencontrerez de nombreux challenges très intéressants, il y aura des hauts et des bas, mais vous adorerez participer à l'évolution rapide de la startup avec des personnes toutes aussi passionnées que vous !

Les avantages ?

C'est vivre le projet à 200% : hyper stimulant et très enrichissant. C'est également la possibilité de sortir de sa zone de confort et de découvrir d'autres aspects très enrichissants (marketing, juridique), celui d'amener son projet de l'idée à la maturation, du premier client au millionième, de commencer from scratch pour arriver à un produit stable et robuste :).

Les inconvénients ?

Comme tout ce qui est vécu à 200%, c'est très intense. On peut passer d'une très mauvaise nouvelle à une très bonne, on vit dans une sorte de yoyo émotionnel. Garder en tête qu'à peu près 10% des entrepreneurs ont des signes de pré burn-out et 20% se sentent soumis à un travail excessif et compulsif (source : étude du cabinet de prévention des risques psychosociaux Technologia 2013). L'autre inconvénient y est lié, la quantité de travail.

Katie Denton : 12 ans & makeuse

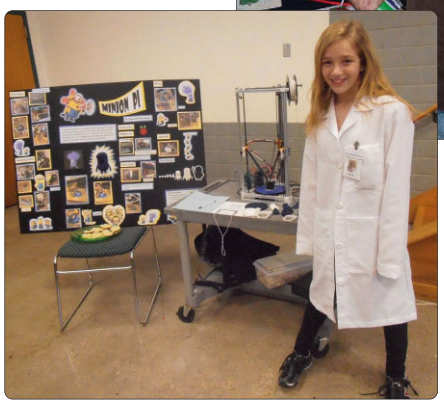
Katie Denton a 12 ans. Elle vit au Texas et elle est actuellement en classe de 5e mais suit les cours de maths de 3e (impensable en France). Elle tient régulièrement un blog et une chaîne YouTube : MakerKatie STEM. Portrait d'une makeuse en herbe.

Interview de Aurélie Vache, traduit par Luce Desblancs

Katie avec le créateur de la Raspberry Pi, Eben Upton

Depuis combien de temps es-tu "maker", comment es-tu tombée dedans ?

J'ai eu mon premier Rasberry Pi pour mes 10 ans et j'ai commencé à coder en Python pour allumer des LED. Mon premier gros projet était une mini imprimante 3D (Kossel Mini 3D) customisée, que j'ai terminée il y a un an. Je voulais une imprimante 3D depuis longtemps, et puis j'ai eu à faire un "Projet Passion" pour l'école, et mon père m'a proposé de la construire à cette occasion. Cela m'a pris plusieurs mois pour la terminer et cela a été énormément de travail mais j'ai appris beaucoup de choses.



Tu as eu la chance de rencontrer le créateur du Raspberry Pi, Eben Upton. C'était à quelle occasion ?

J'ai rencontré le créateur du Raspberry PI par hasard. Je lui avais posé des questions sur l'utilisation d'un Raspberry pour mon projet d'imprimante 3D. Il devait venir au Texas pour le SXCreated du South by South West pour faire une présentation et il m'a invitée à le rencontrer. J'ai pu lui parler pendant environ 1h et il m'a donné de nombreuses idées.

J'ai vu que tu as suivi le programme d'été de tutorat Girls and Boys de Women@NASA. Peux-tu nous en dire plus ?

J'ai été choisie pour participer à ce programme de tutorat l'été dernier et c'était vraiment génial. Chaque semaine je devais échanger avec un ingénieur de la NASA par Skype ou FaceTime et je devais choisir une activité pour la journée. Les projets étaient basés sur le STEM (Science, Technologie, Ingénierie et Maths), avec une thématique différente chaque semaine. Certains projets étaient des activités en ligne, d'autres incluaient la construction d'une main robotisée et d'un modèle du premier avion des frères Wright.

En France on compte environ 10% de développeuses. Ce nombre est semblable en Europe ainsi qu'aux USA. Qu'en penses-tu ?

Je pense que ce chiffre est bien trop bas ! On devrait encourager les filles à l'école à faire ce qu'elles veulent faire et à ne pas être découragées s'il n'y a que des garçons. C'est ce qui est arrivé par exemple au concours de robotique de notre école. Nous avions formé des équipes et avec une de mes amies, nous avions convaincu d'autres filles de former une équipe uniquement féminine. Au concours, il y avait trente huit équipes, et presque toutes étaient des équipes composées uniquement de garçons. Nous sommes arrivées en demi finales ! Ma classe de robotique est presque exclusivement composée de garçons. Dans un autre cours de technologie, j'étais la seule fille de 5e. La plupart de ces cours sont considérés comme étant des "activités de garçons", et ce n'est pas une bonne chose.

Question ouverte à Katie

Pour le "Projet Passion" de cette année, je construis un super ordinateur grâce à 16 Raspberry Pi 3 et je devrai probablement en rajouter d'autres. C'est un projet sur le long terme et ça sera le cerveau de mon armée de robots, qui commencera avec mon Robot Rapiro (Raspberry Pi Robot = robot humanoïde personnalisable grâce à l'impression 3D) et GoPiGo2. Pour commencer je vais installer plusieurs nodes avec OpenCV pour que mes robots puissent faire de la reconnaissance faciale. Je vais installer plusieurs nodes avec JASPER pour que je puisse travailler sur le contrôle vocal de mes robots. Ensuite plusieurs nodes lanceront un cluster Hadoop, ce qui me permettra d'avoir un Big Data storage et ce qui me permettra peut-être de commencer des projets de Machine Learning. Mon but, ça serait de le faire monter à 64 nodes et de voir à quelle vitesse il peut aller. Mon équipe de robotique 100% filles va continuer sa route l'année prochaine, nous sommes déjà prêtes. Le challenge VEX de l'an prochain sera annoncé fin avril et nous pensons commencer la conception de notre robot cet été. J'ai récemment visité Silicon Labs lors d'un voyage scolaire, et nous irons découvrir Google. C'est toujours sympa comme visite. Plus tard, j'aimerais continuer à travailler sur les domaines de la programmation et de la robotique. Je pense que la robotique, le Machine Learning et l'intelligence artificielle (AI) seront très importants dans le futur.

Merci beaucoup Katie ! :-) ⊞

Un Startup Weekend à la Réunion

Du 11 au 13 décembre 2015 a eu lieu la 6ème édition du Startup Weekend de l'île de la Réunion. Cet événement, organisé par l'association locale Webcup, s'est déroulé dans la ville de Saint-Denis dans les locaux de l'ESIROI, l'école d'ingénieur de l'Université de la Réunion.

Anaïs Payet
Développeuse mobile à
Impraise

Bien que le concept ne soit plus tout récent et ayant déjà eu l'occasion de participer à quelques hackatons, je n'avais pourtant jamais tenté l'expérience du Startup Weekend : retour sur cet événement qui s'est déroulé en plein milieu de l'océan indien depuis mon point de vue de développeuse mobile.

Qu'est-ce qu'un Startup Weekend ?

Si l'on devait résumer le concept en une phrase ça serait ainsi : 54H pour former une équipe avec des inconnus dans le but de créer une entreprise à partir d'une idée et être capable la vendre en 5 min devant un Jury d'experts. Fondés à l'origine en 2007 aux Etat-Unis, les Startup Weekends (ou SW) sont des événements à but non lucratif et de portée internationale dont le concept s'est exporté en France en 2010. Aujourd'hui dans le monde, d'après les sources du site officiel du Startup Weekend, c'est déjà plus de 2900 évènements organisés dans plus de 150 pays soient 23 000 équipes formées pour un total de 193 000 participants.

Le Startup Weekend à la Réunion et l'association Webcup

Le SW est l'évènement majeur réunionnais de la création d'entreprise innovante. Depuis le premier organisé en 2011 par la Webcup, de plus en plus de personnes ont exprimé leur volonté de participer à cet événement à tel point qu'en 2015 une 2ème édition dans le sud en complément de celle du nord a été créée. L'association Webcup, organisatrice de ces weekends n'en est pas à son coup d'essai. Elle s'occupe et organise régulièrement la plupart des événements en lien avec le numérique à la Réunion : ateliers de programmation, hackatons, promotion des filières du numérique, et bien sûr le concours de la Webcup dont elle tire son nom, un événement qui met en compétition divers pays de l'océan Indien sur la création d'un site Web.

Grâce au bénévoles de la Webcup, l'apprentissage du code pour tous dans l'île est rendu possible sans limite d'âge ou de sexe.

Qui participe au Startup Weekend ?

On y trouve un public très hétérogène : étudiants, personnes actives ou retraités qu'ils viennent en tant que entrepreneur, développeur, designer, marqueteur ou coach, chacun y est le bienvenue du moment qu'il est motivé. On peut décider d'y venir pour présenter son idée à pitcher mais ce n'est pas obligatoire. Pour ma part, j'ai décidé de venir sans idée et de rejoindre une équipe pour y apporter mes compétences. Pour cette 6ème édition nous étions 113 participants, dont 43 femmes. Environ la moitié des personnes présentes n'avait jamais participé à un startup weekend auparavant.

Venir en tant que développeur au Startup Weekend

En tant que développeur vous êtes peut-être un habitué des hackatons où l'on vient généralement entre geeks pour coder non-stop, manger de la pizza et présenter un prototype qui déboite à la fin du temps imparti. Vivre l'expérience du SW vous plongera dans une toute autre ambiance. Certes vos heures de sommeil seront limitées et on y trouvera probablement de la pizza mais ça n'a pas grand chose à voir.
La première des différences et l'une des plus notables est que la plupart des personnes présentes ne possède aucune ou alors très peu de notions techniques. Aussi, si vous vous attendiez à venir coder tout le weekend, vous pouvez passer votre chemin. A ma plus grande surprise, je n'ai pas écrit une seule ligne de code durant ces 54H ! J'ai fait des estimations de projet, parlé de mon expertise, donné des conseils et pris le temps de réfléchir à la faisabilité d'un produit. Quand on est développeur, on ne s'en rend pas forcément compte, mais on a tendance à ne voir que le côté technique. Ce weekend m'a permis d'élargir mes horizons et de sortir de ma zone de confort. J'ai pu y rencontrer d'autres personnes hors de mon cœur de métier et collaborer avec elles, chose que je n'aurai jamais pu faire dans un autre contexte.

Déroulement du weekend

1er Jour : Vendredi soir

L'arrivée des participants se fait entre 17H et 18H, chacun confirme sa participation, récupère son tee-shirt et une ou plusieurs gommettes de couleur selon sa ou ses compétences :

- Bleu : Développeur / Technique
- Vert : Graphiste
- Jaune : Communication/ Marketing
- Rouge : Business/Finance

Pendant les premières heures nous assistons à la présentation du programme du weekend et à une conférence d'un entrepreneur sur son retour d'expérience de gestion d'entreprise. Puis vient le moment de pitcher son idée. Ceux qui le souhaitent possèdent alors une minute pour convaincre et pas une de plus. Je suis assez impressionnée par le nombre de volontaires : ce n'est pas moins de 40 personnes qui défilent pour présenter leur idée. Les uns derrières les autres les candidats exposent leurs besoins en compétences et

tentent de motiver par leurs concepts.
Après les votes, 13 équipes sont finalement retenues et on nous demande de nous répartir dans l'équipe de notre choix. Il est honnêtement très difficile de choisir car il y a beaucoup d'idées séduisantes, chacune méritant d'être approfondie.
Je choisi de rejoindre un projet dont l'idée est de créer une application mobile pour le tourisme à la Réunion mais sous forme de chasse au trésor géolocalisée. Je suis la seule développeuse dans mon équipe.
Quand vers minuit, les équipes sont enfin toutes créées, on nous assigne notre table pour les deux jours qui vont suivre. Il est déjà bien tard alors nous nous donnons rendez-vous le lendemain de bonne heure pour monter notre « entreprise ». La nuit va être courte et la suite promet d'être intense et pleine de surprises.

2ème Jour : Samedi

A notre arrivée, croissants et pains au chocolat nous attendent déjà pour bien commencer la journée.
Nous prenons le temps de faire mieux connaissance au sein de mon équipe et nous répartissons les tâches par ordre de priorité.
Des « coachs » sont présents durant les 54h pour accompagner toutes les équipes. Ces professionnels confirmés questionnent et conseillent les équipes afin de faire progresser leur projet. Ce sont le plus souvent des entrepreneurs mais aussi des experts comptables, des juristes ou ingénieurs. Leurs compétences sont grandement appréciées et leur conseils très précieux afin de créer un business model qui tient la route.
Durant la journée se déroulent également trois conférences portant sur la création d'entreprise. On peut librement y assister pour approfondir ses connaissances.
La pause déjeuner est très appréciable car la nourriture est typique de la Réunion et cuisiné par les étudiants de l'école. Le reste de l'après-midi est consacré pour ma part à la définition de toutes les fonctionnalités détaillées de l'application et l'ensemble de la soirée à l'élaboration de maquettes.

3ème Jour : Dimanche

La pression monte. A 13h tout doit être bouclé et prêt pour la présentation devant le Jury. Tous les efforts sont mis dans le PowerPoint final.
En début d'après-midi, les unes à la suite des autres via un tirage au sort, les équipes présentent leur projet avec un chronomètre de 5 minutes.
La délibération des Jurys est plutôt longue ce qui rend l'attente interminable.
Au final les gagnants sont annoncés, il ne s'agit pas de mon équipe mais c'est le jeu ! Les gagnants portent un projet qui ambitionne de sublimer les fruits réunionnais sous forme de desserts glacés originaux : l'économie locale à l'honneur. Une dernière photo de groupe pour la route et le buffet de clôture est servi. Tout le monde se dit au revoir en n'oubliant pas de s'échanger ses coordonnées pour des collaborations futures. Des idées ont germé durant le weekend et probablement de nouvelles entreprises sont en train de se créer.
Le SW est un terrain parfait pour expérimenter la viabilité de son idée. La porteuse de projet de mon équipe compte bien continuer sur sa lancée et elle n'est pas la seule. Que ces projets d'un weekend deviennent des projets concrets, c'est tout ce que je leur souhaite, les statistiques sont bonnes.

Bilan

Les Startup Weekends sont des évènements qui encouragent l'innovation et l'entrepreneuriat mais ce sont surtout des week-ends où le partage des compétences, l'apprentissage et la créativité sont les maîtres mots.
Événements ouverts à tous, qui permettent le développement du tissu économique local, les Startup weekends ont pour leitmotiv la conviction que tout le monde peut créer son entreprise et que l'esprit d'équipe est la clef de voûte de tous les projets.
Les SW valorisent les idées et offrent un environnement sécurisant pour les porteurs de projets en mettant toutes les chances de leur côté pour réussir dans l'aventure excitante qu'est la création d'entreprise.
Pour conclure, même si nous n'avons pas gagné avec mon équipe, ce n'était pas le plus important. Personne n'en est sorti déçu, bien au contraire. Nous avons tous le sentiment d'avoir rencontré des personnalités formidables et dotées d'une grande énergie positive. Il est encourageant de voir qu'à partir d'une simple idée on arrive à construire quelque chose même avec des inconnus.
Si vous ne l'avez pas encore fait, je vous recommande grandement de participer à un Startup weekend au moins une fois dans votre vie. C'est une expérience dont vous ressortirez définitivement transformé, et, qui sait, réveillera peut-être votre âme d'entrepreneur.

Abonnez-vous à programmez!

le magazine des développeurs

Nos classiques...

1 an ..... **49 €**
11 numéros

2 ans ..... **79 €**
22 numéros

PDF ..... **30 €(*)**
1 an - 11 numéros
(*) Souscription sur le site internet

Etudiant ..... **39 €**
1 an - 11 numéros

Vous souhaitez abonner vos équipes ? Demandez nos tarifs dédiés* : redaction@programmez.com (* à partir de 5 personnes)

OFFRE SPÉCIALE

1 an ..... **60 €** +
11 numéros

2 ans ..... **90 €** +
22 numéros

** Cette offre inclut l'abonnement à Programmez !, le pack Maker et les frais logistiques et postaux. Pack Maker non vendu séparément de l'abonnement. Livré sans documentation. Offre valable uniquement pour la France métropolitaine. (Photo non contractuelle) Quantité limitée.*

Contenu du pack Mission MAKER

*1 carte Arduino NANO CH340 (avec son câble USB)***
1 planche à pain
1 écran LCD (sans headers)
1 lot de headers
1 lot de fils
1 capteur de température
1 lot de résistances

*** l'Arduino CH340 nécessite des pilotes spécifiques à installer.*

Toutes nos offres sur www.programmez.com

Tarifs France métropolitaine

Oui, je m'abonne

ABONNEMENT à retourner avec votre règlement à :
Service Abonnements PROGRAMMEZ, 4 Rue de Mouchy, 60438 Noailles Cedex.

- ☐ ***Abonnement 1 an au magazine :*** 49 €
- ☐ ***Abonnement 2 ans au magazine :*** 79 €
- ☐ ***Abonnement étudiant 1 an au magazine :*** 39 € Photocopie de la carte d'étudiant à joindre
- ☐ ***Abonnement 1 an au magazine + Mission MAKER :*** 60 €
- ☐ ***Abonnement 2 ans au magazine + Mission MAKER :*** 90 €

PROG 196
Offre limitée, valable jusqu'au 31 mai 2016

☐ M. ☐ Mme ☐ Mlle Entreprise : Fonction :

Prénom : Nom :

Adresse :

Code postal : Ville :

email indispensable pour l'envoi d'informations relatives à votre abonnement

E-mail : @

☐ Je joins mon règlement par chèque à l'ordre de Programmez !

☐ Je souhaite régler à réception de facture

** Tarifs France métropolitaine*

Back from //Build 2016

Du 30 Mars au 1er Avril a eu lieu la //Build de Microsoft, la grand-messe des développeurs au cours de laquelle la firme présente l'ensemble des nouveautés à venir, sur une grande partie de ses produits : Windows, bien évidemment, mais aussi Azure, XBox, HoloLens, etc. Voici donc un tour d'horizon de l'ensemble des annonces faites au cours de cette période, que ce soit lors de « keynotes » ou simplement lors des sessions réalisées tout au long de l'évènement...

Thomas LEBRUN
tlebrun@infinitesquare.com
http://blogs.infinitesquare.com/b/tom

Maxime CAROUL
mcaroul@infinitesquare.com
http://blogs.infinitesquare.com/b/mcaroul

Quoi de neuf sur Windows ?

On commence par la partie « Windows » qui ne symbolise plus simplement le système d'exploitation que l'on connaissait sur PC : le système de la firme de Redmond devient de plus en plus central, utilisé sur les PC, tablettes, téléphones, XBox, etc.

A ce titre, Microsoft annonce qu'une grande mise à jour (Anniversary Update) arrivera au cours de l'été 2016 et permettra de profiter de tout un ensemble d'améliorations et nouveautés majeur.

Ainsi, les tuiles (introduites avec Windows Phone 7) évoluent pour devenir des « Chaseable Live Tiles » : si l'on connait aujourd'hui les « Live Tiles », qui permettent de s'animer pour pousser de l'information à l'utilisateur, on est un peu déçu de s'apercevoir qu'en cliquant sur la tuile, on est redirigé vers la page d'accueil de l'application correspondante. Dans la mise à jour, un clic sur une « Chaseable Live Tile » permettra d'arriver directement sur le contenu associé, permettant de mieux se focaliser sur le contenu proposé à l'utilisateur.

Les notifications vont également évoluer, afin de proposer de nouveaux modèles utilisables dans les applications. Ces nouvelles notifications permettront de proposer plus d'interactions avec l'application source, sans que vous ayez besoin d'ouvrir celle-ci.

Le centre de notification va également avoir droit à de nouvelles optimisations. Déjà, un ensemble de modifications graphiques vont être apportées :

- Possibilité de voir les photos des contacts ;
- Possibilité d'avoir plus d'interaction avec les notifications ;
- Un plus grand contenu sera affiché/visible ;
- Etc.

Mais ce n'est pas tout ! Les notifications sur les tuiles et les notifications affichées dans le centre de notification seront synchronisées. Ainsi, si vous avez 3 notifications en attente et que vous en effacez une, la tuile se mettra à jour pour afficher le bon nombre de notifications en attente.

Enfin, une nouvelle fonctionnalité va faire son apparition : « l'Universal Dismiss ». Derrière ce nom un peu chic se cache une demande importante de la part des utilisateurs : la possibilité de pouvoir enlever une notification sur un périphérique et, via un mécanisme de synchronisation dans le Cloud, d'avoir cette notification synchronisée (et donc enlevée) sur l'ensemble des autres périphériques.

Enfin, parce que Windows ne serait pas Windows sans sa barre des tâches, celle-ci évolue aussi pour permettre, aux applications UWP, d'afficher un nombre (également appelé « badge ») pouvant contenir un compteur de notifications, de messages non-lus, de flux RSS, etc.

Cortana va également être amené à évoluer, avec la mise à jour de cet été, pour devenir plus intelligente et apprendre à vous connaître de mieux en mieux. Ainsi, avec votre permission (cela restera une autorisation à donner, de la part de l'utilisateur), il lui sera possible de scanner vos différents documents : emails, calendriers, documents, etc. A partir des données récoltées, cela lui permettra de vos proposer des rappels ou des actions à réaliser, tout en étant contextualisé à vos propres données. Il lui sera également possible de suivre les notifications sur l'ensemble de vos périphériques et, nouveautés, il lui sera possible d'envoyer des SMS, sur votre téléphone Android, depuis votre PC (cette fonctionnalité existant déjà pour Windows 10 Mobile mais se trouve maintenant présente pour Android).

Fig.1

De plus, dans un souci d'homogénéisation de l'ensemble de la plateforme, Cortana se verra présente sur la XBox, qui s'ouvrira à l'ensemble des développeurs en leur permettant, via le téléchargement d'une application depuis le Store, de transformer leur console classique (à savoir achetée dans le commerce) en kit de développement leur permettant de développer et déployer des applications UWP ! Il s'agit là d'une avancée majeure, le développement d'applications XBox ayant été réservé uniquement à certains partenaires pour le moment.

Si vous êtes utilisateur d'une tablette Surface ou d'un autre périphérique utilisable avec un stylet, sachez que Microsoft ne vous a pas oublié et à pensé à vous avec « Windows Ink ». L'objectif de « Windows Ink » est d'inciter les utilisateurs à oublier le côté « papier/stylo » et de les emmener sur la voie du stylet, en s'intégrant à tous les niveaux de Windows 10. Fig.1.

Ainsi, il sera possible de prendre ses notes directement avec le stylet, de disposer d'une règle, de dessiner et calculer des itinéraires via l'application « Cartes », de sélectionner/manipuler des paragraphes ou mots dans Microsoft Word, etc. Cette fonctionnalité semble vraiment importante pour Microsoft qui a annoncé que les développeurs pourront intégrer cette fonctionnalité dans leur propres applications UWP, au moyen de quelques lignes de code !

Enfin, et cela a fait un peu l'effet d'une bombe au sein de la communauté des utilisateurs Windows, Microsoft a annoncé l'intégration native, dans l'OS, de Bash. Si vous avez déjà ouvert une ligne de commandes sur un système d'exploitation de type Linux/Unix, vous avez déjà utilisé Bash et ses fameuses lignes de commandes (grep, etc.) qui intègre, en natif, des

fonctionnalités comme Telnet, SSH, etc. Il sera maintenant possible d'utiliser ces lignes de commandes Linux directement depuis Windows : on retrouvera donc un peu d'Ubuntu au sein de Windows !

Du nouveau dans le Web ?

Microsoft Edge, le nouveau navigateur de la firme américaine, arrivera lui aussi avec son lot de nouveautés et d'évolutions. Tout d'abord, et cela a été très demandé par les développeurs/utilisateurs, les extensions seront présentes mais, surtout, elles auront la possibilité d'interagir avec d'autres applications natives de Windows 10. En termes de solution et d'implémentation technique, il semblerait que cela repose sur la mise à disposition d'un répertoire partagé et accessible aux extensions/applications. Peu de détails ont été donnés sur cette partie donc il y a fort à parier que nous en saurons plus dans les mois à venir.
Au niveau des nouvelles fonctionnalités de Edge, Microsoft a fait une démonstration de l'utilisation de Windows Hello, le mécanisme d'authentification biométrique (par reconnaissance faciale ou empreinte digitale) avec les systèmes de sécurité présents sur le Web. Lors de la demande de connexion à un site Web, plus besoin de nom d'utilisateur et/ou mot de passe : il faut simplement se présenter devant son PC ou utiliser son empreinte digitale pour réaliser la connexion au site !
La dernière, mais non des moindres, des fonctionnalités présentées sur Microsoft Edge concerne les notifications : le navigateur supportera les notifications Web comme le fait Chrome. De plus, l'ensemble de ces notifications sera visible dans le centre de notifications de Windows : plus de raisons pour vous de passer à côté.

Xamarin, Bot Framework et bien plus encore

Parmi les grandes annonces effectuées au cours de l'évènement, l'une a été particulièrement applaudie : la mise à disposition, gratuitement, de Xamarin ! Microsoft a frappé fort en annonçant le rachat de la société de Miguel De Icaza mais ils ont été plus loin en déclarant que Xamarin sera dorénavant gratuit et accessible depuis l'ensemble des versions de Visual Studio : de la version « Community Edition » à la version « Enterprise ».
C'est un énorme pas en avant pour le produit qui, jusqu'à maintenant, avait une bonne cote auprès des développeurs mais pour lequel on reprochait son coût assez conséquent ($2000 par développeur et par plateforme). Mais Microsoft ne s'arrête pas là et a également annoncé que les SDKs de Xamarin pour Android, iOS et Mac seraient rendus Open Source, sous licence MIT : si quelque chose ne vous plait pas dans Xamarin, vous serez libre de le corriger/changer par vous-même et d'embarquer votre propre version du moteur Xamarin dans vos applications !
Le côté « Intelligence Artificielle », que l'on retrouve dans le slogan « Conversations As A Platform » de Microsoft, prend également tout son sens avec la mise à disposition du Bot Framework (https://dev.botframework.com) : un ensemble d'outils permettant de développer des agents de communication (des « bots ») capables de comprendre les questions posées et, en fonction du contexte, de proposer des résultats adéquats. Ainsi, lors de la conférence, Microsoft a fait la démonstration d'un agent capable de prendre la commande d'une pizza, en permettant à l'utilisateur de spécifier les différentes options, l'adresse de livraison, etc. Il est donc possible de développer des agents doués d'une certaine intelligence, capables de comprendre vos utilisateurs et capables d'apprendre et de s'améliorer, notamment grâce au Machine Learning et à des services externes tels que LUIS (Language Understanding Intelligent Service, https://www.luis.ai).
Dernière grosse nouveauté annoncée, sur la partie globale/Windows : la mise à disposition du « Desktop App Converter ». En soit, il ne s'agit pas d'une nouvelle annonce car Microsoft en avait déjà parlé lors de la Build précédente. Cependant, le mode de fonctionnement a évolué et les démonstrations réalisées montrent la puissance de l'outil, à tel point qu'on est vite impatient de vouloir le tester. Fig.2.
L'objectif du convertisseur est de prendre un MSI traditionnel (tel que celui de Pain.NET, celui de Photoshop ou celui de n'importe quelle application Win32), de le passer dans une moulinette et d'obtenir, au final, un installeur permettant d'installer l'application en mode UWP, donc en passant par le Store ou un déploiement privé.
L'intérêt est ici vraiment majeur : plus besoin de se compliquer la vie pour faire les mises à jour, tout se fait via le Store et il est même possible, pour vos applications Win32, d'utiliser des APIs Windows 10 (comme les « Live Tiles », les notifications, etc.) et donc de migrer, petit à petit vers la plateforme UWP.

Azure n'est pas oublié

Leader en PaaS et sur la couverture mondiale

Avant d'aborder les différentes annonces de cette conférence, il convient de rappeler ces quelques chiffres sur le Cloud Azure :

- Présent dans 22 régions du monde, et 8 autres seront bientôt annoncées. C'est plus que Google et AWS réunis ;
- 85% des entreprises du Fortune 500 utilisent Azure ;
- Plus de 120 000 nouvelles souscriptions par mois. C'est 20 000 de plus qu'annoncé à la précédente conférence Build ;
- 1,4 million de bases de données SQL gérées ;
- 2 milliards de messages par semaine sont traités par Azure IoT ;
- 5 millions d'entreprises utilisent Azure Active Directory ;

D'ailleurs, le cabinet Gartner a récemment reconnu Microsoft Azure comme un leader sur les solutions PaaS pour la troisième année consécutive.

Focus sur l'IoT, les webjobs as a service et les micro-services

Azure IoT tente d'offrir une solution simple, clef en main qui permet aux entreprises de gérer à distance et d'interagir avec ses périphériques en supportant le protocole open source LWM2M. Microsoft fournit aussi un SDK qui permet d'optimiser et de traiter les données avant de les envoyer sur Azure pour bénéficier d'une latence minimale, d'un faible coût de bande passante et d'une sécurité renforcée. Mais son atout majeur est qu'il permet de connecter une infrastructure existante avec Azure IoT !
Certainement l'annonce la plus inattendue de cette conférence Azure Functions, actuellement en preview publique, est un service basé sur le

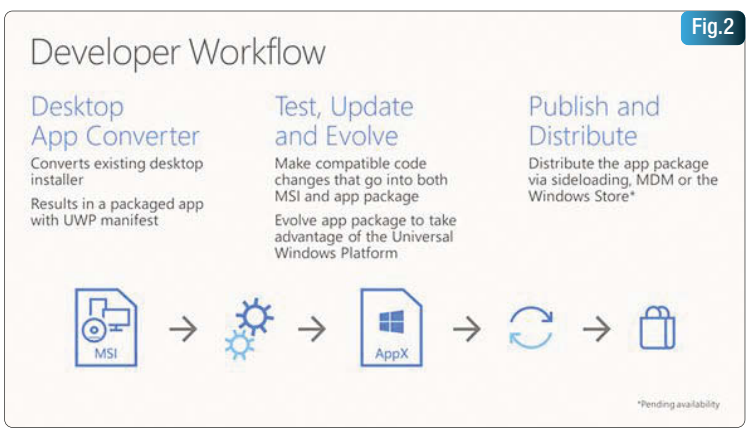


Fig.2

SDK des WebJobs. Il permet de définir des fonctions, en d'autres termes votre logique métier, en réponse à certains évènements (une planification, une requête http, un message reçu dans service bus, etc...) sans se soucier de l'hébergement ni de la mise à l'échelle. Ceci avec une tarification réellement basée sur votre usage, en tenant compte du temps d'exécution de votre code ainsi que la quantité de mémoire allouée, grâce au plan de service dynamique.
Cette offre de service est adaptée pour le traitement des données, l'internet des objets, la création de simples API ou micro-services. Pour le moment, seuls les langages C# et Node.js sont supportés nativement. D'autres langages tel que F#, Python, PHP ainsi que les scripts Bash et PowerShell sont supportés en mode expérimental.
Azure Service Fabric qui sert à gérer et déployer des applications micro-service hautement évolutives et fiables passe en disponibilité générale. Sachez que cette plateforme est au cœur de nombreux services Cloud tel qu'Azure SQL Database, Cortana, Skype Entreprise, Intune, Event Hubs, DocumentDB et Azure Data Factory. Sachez également qu'Azure Service Fabric est également disponible en mode on-premise et donc aussi dans d'autre Cloud, d'abord sur Windows et bientôt sur Linux.

Des solutions de bases de données mises à jour

DocumentDB a bénéficié d'un nombre important d'annonces. Tout d'abord de nouvelles options de tarification ont fait leur apparition permettant d'allouer séparément l'espace de stockage en gigaoctets et le débit via les unités de requêtes.
La fonctionnalité « Global Database » propose une réplication sur différentes régions Azure. Cette dernière est en preview et nécessite de s'inscrire au programme pour pouvoir en profiter. La fonctionnalité Time-To-Leave sur les documents, déjà présente dans MongoDB, devrait faire son apparition ce mois d'Avril.
Pour finir, DocumentDB supporte (en preview) de manière transparente les drivers et API MongoDB.
Concernant SQL Database, l'offre est en général en retard au niveau fonctionnel comparé à la dernière version SQL Server on-premise disponible.
Mais cette tendance commence à s'inverser avec les fonctionnalités comme le masquage dynamique des données, le mode Always Encrypted ou le filtrage des données selon le contexte utilisateur (Row Level Security) qui ne seront disponibles qu'à la prochaine version de SQL Server (SQL Server 2016).
D'ailleurs, Microsoft a profité de cette conférence pour annoncer la disponibilité en preview publique des tables temporelles.
Cette fonctionnalité permet de stocker l'historique complet des changements appliqués aux données et offre ainsi de nombreuses possibilités comme une analyse chronologique des données ou même la possibilité de reconstruire ses données à partir de n'importe quel moment dans le passé.

Dans un environnement toujours plus automatisé, sécurisé et maitrisé

Application Insights bénéficie de trois nouvelles fonctionnalités :

- Analytics : permet de requêter vos données de télémétrie de manière avancée ;
- Live Stream Metrics : permet de suivre les métriques de votre application presque en temps réel ;
- Application Map : découvre automatiquement la topologie de votre application et présente une carte descriptive afin de vous aider à identifier les problèmes de performance.

Azure Resource manager n'échappe pas aux nouveautés avec le support d'Azure Key Vault (nécessite le SDK Azure 2.9) afin de lier la valeur des paramètres de déploiement aux secrets stockés et ainsi ne pas les sauvegarder dans le contrôleur de code source. Il est également possible d'exporter le template à partir d'un groupe de ressources ou d'un précédent déploiement.
Pour finir, une fonctionnalité de debug fait son apparition lors des déploiements via PowerShell ou CLI en ayant accès au contenu de la requête et de la réponse pour chaque opération.
Le service de chiffrement sécurise vos stockages blob de manière transparente, inclut la gestion des clefs et n'ajoute aucun surcoût supplémentaire. Cette fonctionnalité actuellement en preview n'est disponible que pour les comptes de stockage non classique présent en Asie de l'est (en standard) et à l'est du Japon (en premium).

.NET / C#, plus présents et importants

Les nouveautés concernant Visual Studio 15 et C# 7 sont très nombreuses et ont été pensées pour améliorer encore davantage la productivité des développeurs. La tendance étant clairement à l'open-source et au cross-plateforme.
Il n'y a eu aucune annonce particulière du côté d'EF Core 1.0 mais le Framework toujours en développement a tout de même bénéficié d'une couverture à travers cette conférence. Le but de Microsoft est d'offrir un Framework d'accès aux données adapté à :

- Toutes les plateformes supportant .NET Core (Windows, OSX ou Linux) ;
- Tous les fournisseurs de base données relationnelles, non-relationnelles tel que Redis ou Azure Table Storage ;
- Tous les types de devices (téléphone, tablette, Xbox, PC et serveur).

Pour cela, EF Core 1.0 repose sur une nouvelle base de code offrant une API cœur légère et un ensemble d'extensions modulaires via des paquets NuGet. Ainsi vous ne chargez que ce dont vous avez besoin avec un impact mémoire et CPU optimisé, c'est le principe du « pay-per-play ». Du côté des "nouveautés", citons le support du mode batch de manière transparente lors de l'appel à la méthode SaveChanges ainsi que l'évaluation côté client des requêtes LINQ.
Attention cependant, EF Core 1.0 n'a pas vocation à remplacer la dernière version majeure d'Entity Framework. D'ailleurs sachez que toutes les fonctionnalités d'EF 6.X ne seront pas implémentées dans EF Core 1.0. Actuellement, EF Core est en version pre-release et représente une « vraie » 1ère version avec :

- Des fonctionnalités manquantes : « lazy loading », mapping de procédures stockées, etc. ;
- Des fonctionnalités limitées : translation des requêtes LINQ ;
- Un support de fournisseur de base de données réduit.

C'est pourquoi il est important d'évaluer avant de migrer de EF 6.X vers EF Core 1.0. En effet, certaines API ont changé et certains comportements sont différents pour une même méthode.
En ce qui concerne C#, le langage évolue toujours au fur et à mesure des décisions prises par Microsoft, en fonction des retours des membres de la communauté (l'équipe ayant l'habitude de poster les notes de ces meetings internes sur Github, afin d'avoir des retours « terrain »).
Au programme, on retrouvera des choses assez sympathiques comme le « Binary literal » :

```
int[] numbers = { 0b1, 0b10, 0b100 };
```

Ou les "Nested Functions" (les fonctions imbriquées dans des fonctions) :

```
static void Main(string[] args)
```

```
{
   int NestedFunction()
   {
      // ...
      return 0;
   }

   int result = NestedFunction();
}
```

Conclusion

Comme nous pouvons le constater, la //Build de cette année a permis à Microsoft d'annoncer un grand nombre de nouveautés. Même si l'on pouvait se douter de certaines d'entre elles, d'autres ont été de véritables surprises, pour l'ensemble des participants à la conférence. Il est à noter qu'aucune date n'a été annoncée pour la prochaine édition : il est possible que Microsoft préfère attendre un peu d'avoir un peu plus de contenu intéressant à proposer pour une prochaine édition... ou bien ils font durer le suspense, à voir dans les prochains mois !

Bash Linux sur Windows 10

C'est sans doute la grosse nouveauté de Windows 10 annoncée durant la conférence BUILD : une dose de Linux directement dans Windows ! Pour cette fonctionnalité, Microsoft a travaillé avec Canonical, éditeur de la distribution Ubuntu.

La rédaction

Attention : il s'agit d'une fonction bêta. Toutes les fonctions ne sont pas encore présentes et la stabilité est aléatoire.

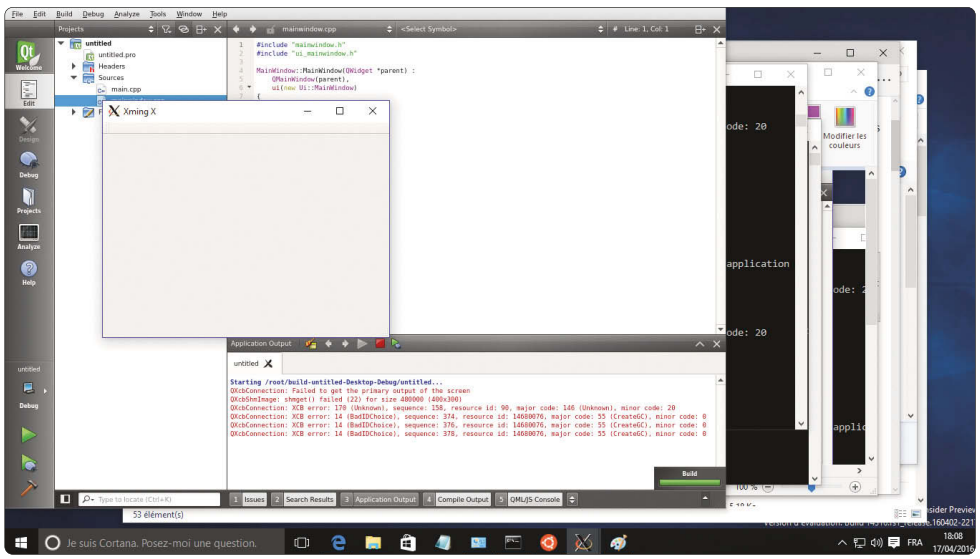
On dispose de l'application Bash. Bash est un shell avec un langage de commandes. On le retrouve sur Ubuntu, OS X et d'autres Linux. Il permet d'exécuter des commandes et des scripts. Lorsque la partie Linux est active, le Bash devient Ubuntu on Windows car en réalité, il dispose d'un user-mode Ubuntu. Le Bash et les outils de lignes de commandes sont ceux d'Ubuntu. Cette fonction repose sur un sous-ensemble Windows pour Linux (WSL), c'est le cœur du système. Surtout, le WSL est natif à Windows. Ce n'est pas une machine virtuelle ou une émulation mais du 100 % natif ce qui assure de bonnes performances des applications Linux !

Une installation très simple

Actuellement, cette fonction est uniquement accessible aux abonnés Windows Insider. Vous devrez activer la récupération des builds Insider dans les paramètres Windows et mettre à jour. Il faut une build 14316, ou supérieure.

Il faut ensuite que l'option « mode développeur » soit activé (paramètres -> mise à jour et sécurité -> pour les développeurs). Après tous les redémarrages, vous devrez, dans « programmes et fonctionnalités », activer l'option Windows Subsystem for Linux (Beta). Et un petit redémarrage...

Puis, dans « invite de commandes », on tape bash. On confirme et l'outil s'occupe de tout. L'opération prend quelques minutes si cela semble tourner dans le vide, recommencer le bash après un redémarrage. C'est tout !



Qt Creator utilisant Xming pour la partie interface.

On installe et on exécute !

Le bash fonctionne directement dans Windows et se présente comme une application. Dans la ligne de commande, on accède aux commandes nécessaires. Pour installer un éditeur de type vim, on peut faire :

```
Apt-get install vim
```

Puis taper vim ; ceci ouvre alors l'éditeur.

Bref du pur Linux.

Par défaut, bash n'inclut pas de serveur X (serveur servant aux interfaces graphiques). Vous pouvez utiliser Xorg ou Xming. Nous avons testé Qt Creator et Firefox. Attention à bien utiliser la commande display :=

La stabilité est parfois très aléatoire en mode graphique mais cela montre tout le potentiel du bash et surtout la possibilité, dans le futur, de coder directement en mode Linux depuis son Linux ou de tester des apps et des sites Web nativement. Ou encore d'utiliser des applications Linux directement sous Windows avec de bonnes performances.

Des limites à ne pas oublier

Vous ne disposez pas d'un Linux complet et vous ne pourrez pas exécuter toutes les applications Linux. Des limites sur les serveurs X, sur les applications nécessitant une GUI, existent et vous ne pourrez pas tout utiliser. Les outils et solutions serveurs ne sont pas recommandés par Microsoft. WSL est une approche poste de travail / desktop et surtout orienté développeur.

Nous sommes dans un mode d'exécution 64 bits. Bien entendu, vous pouvez désactiver et désinstaller WSL, en ligne de commandes ou via l'interface.

Pour en savoir plus sur WSL : https://msdn.microsoft.com/en-us/commandline/wsl/about

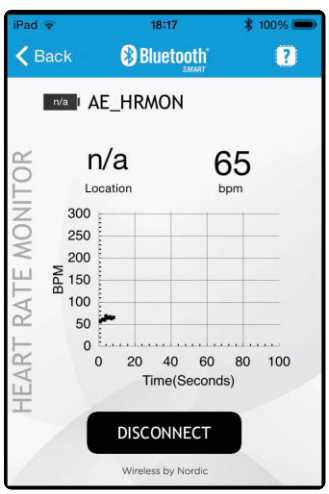
Nous sommes impatients de voir comment cette fonction va évoluer dans les futures builds.

Arduino / Genuino 101 : la nouvelle Arduino veut remplacer la célèbre Uno

On l'attendait avec une certaine impatience. La nouvelle carte Arduino, basée sur Intel Curie, est enfin disponible. La Genuino 101, tel est son nom en Europe, a la lourde mission de remplacer la Uno, best-seller incontournable des makers. Il y a de bonnes surprises, mais aussi de mauvaises.

François Tonic
Programmez!

La 101 est une Arduino comme la Uno : même programmation, même qualité, open hardware, mais avec un microcontrôleur plus puissant et des capteurs intégrées. Le SoC Intel est un Curie 32 bits basse consommation, embarquant les capteurs et capacités BLE. Cette carte pourrait se comparer à la carte Intel Galileo que nous avions beaucoup aimée. Mais elle était parfois instable et vendue à 60 €.
La carte arrive à peine en France et elle est encore difficile à trouver (et à un prix un peu élevé). Et il faudra attendre un peu avant de voir apparaître des clones pas chers comme avec les Uno. Le microcontrôleur risque d'être un frein. Pour le prix, malheureusement, elle sera plus proche des 40-45 € que des 30.

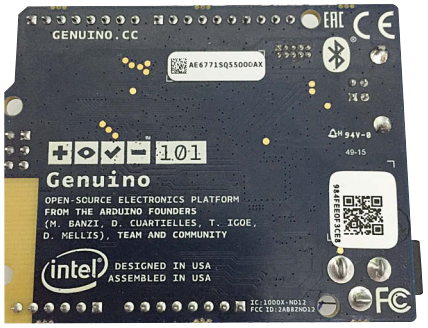


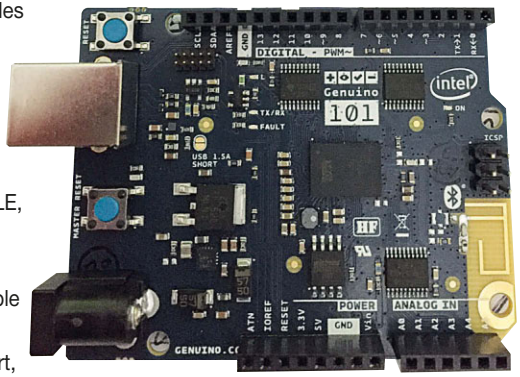
Premières impressions

Nous avons trouvé une Genuino 101 officielle. Livrée nue... heureusement que nous possédons câble USB et alimentation. La carte reprend les mêmes dimensions et l'emplacement des trous ne change pas (ouf !). Par contre, on constate immédiatement des différences : 2 boutons reset, des composants plus nombreux, une puce Intel plus compacte, une antenne pour le Bluetooth.
La carte est très propre, comme toujours avec les cartes officielles.

Passons aux choses sérieuses

Pour utiliser la 101, vous pouvez utiliser l'outil Arduino IDE, mais minimum en version 1.6.7. Pas pratique quand certaines librairies, ou clones fonctionnent uniquement avec des versions antérieures. Nous trouvons les dernières versions moins rapides et plus grosses en installation !
Ensuite, vous devez ajouter la carte 101 via le gestionnaire des cartes. L'opération est simple et rapide.
Alimentez la carte par une alimentation externe, parfois, le port USB n'est pas assez puissant. Nous avons eu des déconnexions de la board, obligeant à reseter régulièrement la carte. Des exemples de codes sont présents pour les nouvelles fonctions (BLE, accéléromètres et le gyroscope). Le gyro est assez sensible, mais semble assez précis d'après nos premiers tests. D'autre part, la carte sera compatible avec les shields Arduino actuels.
Petit détail : sur certains exemples officiels (depuis le site Arduino), le nom des librairies est incorrect, corrigez manuellement. Par exemple : CurieBLE.h et non CurieBle.h... Arduino manquait de capteurs par défaut. La 101 est une bonne surprise avec des capteurs intégrés et la possibilité de les mixer : combiner la connexion BLE avec une app mobile telle que nRF Toolbox et un capteur de fréquence cardiaque (de type Keyes HrAT). Le code exemple est très rapide à trouver et à charger sur la 101. On connecte l'app mobile à la 101 et c'est tout ! Bien entendu, le résultat est imprécis, mais cela vous démontre le potentiel de la nouvelle carte et des capteurs intégrés qui vont simplifier certains projets.





Bref !

Pour le moment nous ne sommes ni satisfaits ni déçus. Attendons de voir à l'usage et en espérant que la communauté propose de belles choses sur la 101, et, surtout, que les clones puissent rapidement arriver. Pour exécuter pour la première fois le code chargé depuis l'IDE, la carte est plus lente.
Pour étendre les fonctionnalités, Intel a développé un système spécifique à la carte (ViperOS). Mais en réalité, les fonctions avancées ne sont pas disponibles actuellement. L'accès à ces fonctions, et au code du système, était annoncé pour mars. Espérons que l'on pourra rapidement étendre les fonctionnalités de base.
De nouveaux systèmes légers devraient toutefois voir le jour. Ainsi, il existe le projet Zephyr (fondation Linux) pour IoT et supportant notamment la 101. Nous vous conseillons la prudence et de disposer d'une carte supplémentaire...
Quelques projets sur hackster.io : https://www.hackster.io/arduino/products/arduino-101-genuino-101

	Uno	101
microcontrôleur	ATmega328	Intel Curie 2 cœurs
Voltage de la carte	5V	3,3V (avec tolérance à 5)
Compatibilité Uno	N/A	Oui
Capteurs et modules	Aucun	Accéléromètre Gyroscope - Ethernet BLE
Outil de développement	Arduino IDE	Arduino IDE
Librairies dédiées supplémentaires	Non	Oui
Tarif	env. 15 €	env. 35-40 €

LES +

- Les capteurs intégrés
- BLE
- Microcontrôleur Intel
- Nouvelles librairies
- Compatibilité
- Open Source

LES –

- La tolérance 5V
- Prix
- Pas de clone
- Instabilités durant nos tests
- Lenteurs

PHARAON + programmez! le magazine des développeurs

Défi de programmation 2016

Comment concilier archéologie, histoire, voyage et programmation ? Les magazines Pharaon et Programmez ! lancent un défi de programmation pour les développeurs, les étudiants en informatique et tous les passionnés !

L'objectif est simple :

Créer une app mobile capable de prendre une photo des cartouches royaux des pharaons pour lire automatiquement les noms écrits en hiéroglyphes et afficher le nom du pharaon.

L'app doit fonctionner sur smartphone ou tablette et être disponible, au moins, sur une des 3 plateformes du marché : **Android, iOS, Windows Mobile.**

Agenda

Le défi est ouvert depuis le 15 mars pour une durée de 3 mois.
Le 15 juin, les développeurs devront envoyer leurs apps pour que la rédaction les teste et choisisse les gagnants qui seront annoncés dans le numéro d'été et/ou sur le site Web du magazine fin juin.
Trois apps gagnantes seront retenues.

Les gagnants recevront :

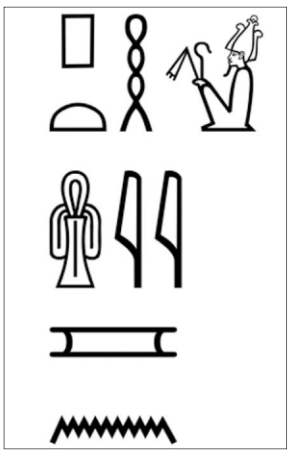
- 1 Intel NUC,
- 1 carte RIoTBoard
- 1 clé USB Programmez !
- 1 clé USB Pharaon Magazine,
- 1 visite du département égyptien du Musée du Louvre avec François Tonic (historien, éditeur & rédaction en chef de Programmez ! et de Pharaon Magazine),
- 2 livres : la tombe de Ramose, la tombe royale d'Akhenaton,
- Articles dans Programmez ! et Pharaon Magazine.

TOUS LES DÉTAILS SUR
WWW.PROGRAMMEZ.COM

EXEMPLE

1 *- L'utilisateur prend la photo du cartouche :*

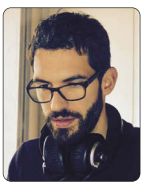
2 *- L'app reconnaît les deux cartouches et signes hiéroglyphiques :*



3 *- Réponse donnée à l'utilisateur :*
Menmaâtrê
L'Osiris, celui de Seth, l'aimée de Ptah
= le pharaon Séthy Ier
(hiéroglyphes réalisés avec l'outil JSesh)

Comment Angular 2 rend vos applications plus performantes : la magie du framework révélée

Développer avec AngularJS 1.X dégageait un sentiment de "magie", Angular 2 procure toujours ce même effet : tu entres une valeur dans un input et hop!, comme par magie, toute la page se met à jour en conséquence. J'adore la magie, mais je préfère comprendre comment fonctionnent les outils que j'utilise. Si tu es comme moi, je pense que cette partie devrait t'intéresser : on va se pencher sur le fonctionnement interne d'Angular 2 ! Mais d'abord, laisse-moi t'expliquer comment fonctionne AngularJS 1.x, ce qui devrait être intéressant, même si tu n'en as jamais fait.



Cédric Exbrayat
Développeur et co-fondateur de la société **Ninja Squad**

Tous les frameworks JavaScript fonctionnent d'une façon assez similaire : ils aident le développeur à réagir aux évènements de l'application, à mettre à jour son état et à rafraîchir la page (le DOM) en conséquence. Mais ils n'ont pas forcément tous le même moyen d'y parvenir.
EmberJS par exemple, demande aux développeurs d'utiliser des setters sur les objets manipulés pour que le framework puisse "intercepter" l'appel au setter, connaître ainsi les changements appliqués au modèle, et pouvoir modifier le DOM en conséquence. React, lui, a opté pour recalculer tout le DOM à chaque changement mais, comme mettre à jour tout le DOM est une opération coûteuse, il fait d'abord ce rendu dans un DOM virtuel, puis n'applique que la différence entre le DOM virtuel et le DOM réel.
Angular, lui, n'utilise pas de setter, ni de DOM virtuel. **Alors comment fait-il pour savoir ce qui doit être mis à jour ?**

AngularJS 1.x et le digest cycle

La première étape est donc de **détecter une modification du modèle**. Un changement est forcément déclenché par un évènement provenant soit directement de l'utilisateur (par exemple un clic sur un bouton, ou une valeur entrée dans un champ de formulaire) soit par le code applicatif (une réponse HTTP, une exécution de méthode asynchrone après un timeout, etc.).
Comment fait donc le framework pour savoir qu'un évènement est survenu ? Eh bien c'est simple, il nous force à utiliser ses directives, par exemple *ng-click* pour surveiller un clic, ou *ng-model* pour surveiller un input, et ses services, par exemple *$http* pour les appels HTTP ou *$timeout* pour exécuter du code asynchrone.
Le fait d'utiliser ses directives et services permet au framework d'être parfaitement informé du fait qu'un événement vient de se produire. C'est ça la première partie de la magie ! Et c'est cette première partie qui va déclencher la seconde : il faut maintenant que le framework analyse le changement qui vient de survenir, et puisse déterminer quelle partie du DOM doit être mise à jour.
Pour cela, en version 1.x, le framework maintient une liste de *watchers* (des observateurs) qui représente la liste de ce qu'il doit surveiller. Pour simplifier, un *watcher* est créé pour chaque expression dynamique utilisée dans un template. Il y a donc un watcher pour chaque petit bout dynamique de l'application, et on peut donc avoir facilement plusieurs centaines de watchers dans une page.

Ces watchers ont un rôle central dans AngularJS 1.x : ils sont la mémoire du framework pour connaître l'état de notre application.
Ensuite, à chaque fois que le framework détecte un changement, il déclenche ce que l'on appelle le *digest* (la digestion).
Ce digest évalue alors toutes les expressions stockées dans les watchers et compare leur nouvelle valeur avec leur ancienne valeur. Si un changement est constaté entre la nouvelle valeur d'une expression et son ancienne valeur, alors le framework sait que l'expression en question doit être remplacée par sa nouvelle valeur dans le DOM pour refléter ce changement. Cette technique est ce que l'on appelle du *dirty checking*.

Pendant ce digest, AngularJS **parcourt toute la liste des watchers**, et évalue chacun d'eux pour connaître la nouvelle valeur de l'expression surveillée. Avec une subtilité de taille : ce cycle va être effectué dans son intégralité tant que les résultats de tous les watchers ne sont pas stables, c'est à dire tant que la dernière valeur calculée n'est pas la même que la nouvelle valeur. Car bien sûr, dans une vraie application, le résultat d'une expression, donc d'un watcher, déclenche parfois un callback qui va lui-même modifier à nouveau le modèle et donc changer la valeur d'une ou plusieurs expressions surveillées !
Prenons un exemple minimaliste : une page avec deux champs à remplir par l'utilisateur, son nom et son mot de passe, et un indice de robustesse du mot de passe qui surveille le mot de passe, recalculé à chaque fois que celui-ci change.
Nous avons alors cette liste de watchers après leur première évaluation, lorsque l'utilisateur a saisi le premier caractère de son mot de passe :

```
$$watchers (expression -> value)
- "user.name" -> "Cédric"
- "user.password" -> "h"
- "passwordStrength" -> 2
```

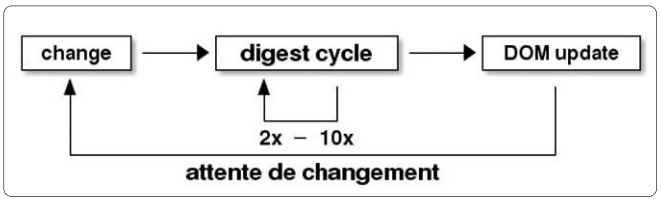
Le cycle va être effectué une seconde fois pour voir si les résultats sont stables, car certains résultats dépendent peut-être d'une autre expression.

```
$$watchers
- "user.name" -> "Cédric"
- "user.password" -> "h"
- "passwordStrength" -> 3
```

C'est le cas : la force du mot de passe dépend de la valeur de celui-ci. Comme les résultats ne sont pas encore stables (une valeur ayant changé entre les deux cycles), le cycle est une nouvelle fois évalué en entier.

```
$$watchers
- "user.name" -> "Cédric"
- "user.password" -> "h"
- "passwordStrength" -> 3
```

Cette fois, c'est stable. C'est seulement à ce moment-là qu'AngularJS 1.x applique les résultats sur le DOM. Cette boucle de digest est donc jouée au moins 2 fois à chaque changement dans l'application. Elle peut être jouée jusqu'à 10 fois mais pas plus : après 10 cycles, si les résultats ne sont toujours pas stables, le framework considère qu'il y a une boucle infinie et lance une exception.
Donc mon schéma précédent ressemble en fait plus à :



Et j'insiste : c'est ce qui se passe après chaque évènement de notre application. Cela veut dire que si l'utilisateur entre 5 caractères dans un champ, le digest sera lancé 5 fois, avec 3 boucles à chaque fois dans notre petit exemple, soit 15 boucles d'exécution. Dans une application réelle, on peut avoir facilement plusieurs centaines de watchers et donc plusieurs milliers d'évaluations d'expressions à chaque changement. Même si ça semble fou, cela marche très bien car les navigateurs modernes sont vraiment rapides, et il est toujours possible d'optimiser deux ou trois choses si nécessaire.
Tout cela signifie deux choses pour AngularJS 1.x :

- Il faut utiliser les services et les directives du framework pour tout ce qui peut déclencher un changement ;
- Modifier le modèle à la suite d'un évènement non géré par Angular nous oblige à déclencher nous-même le mécanisme de détection de changement (en ajoutant le fameux $scope.$apply() qui lance le digest). Par exemple, si l'on veut faire une requête HTTP sans utiliser le service $http, il faut appeler $scope.$apply() dans notre callback de réponse pour dire au framework : "Hé, j'ai des nouvelles données, lance le digest s'il te plaît !".

Et la magie du framework peut être scindée en deux parties :

- Le déclenchement de la détection de changement à chaque événement ;
- La détection de changement elle-même, grâce au digest, et à ses multiples cycles.

Maintenant voyons comment Angular 2 fonctionne, et quelle est la différence.

ANGULAR 2 ET LES ZONES

Angular 2 conserve les mêmes principes, **mais les implémente d'une façon différente**, et on pourrait même dire, plus intelligente.
Pour la première partie du problème — le déclenchement de la détection de changement — l'équipe Angular a construit un petit projet annexe appelé *Zone.js*. Ce projet n'est pas forcément lié à Angular, car les zones sont un outil qui peut être utilisé dans d'autres projets. Les zones ne sont pas vraiment un nouveau concept : elles existent dans le langage Dart (un autre projet Google) depuis quelques temps déjà. Elles ont aussi quelques similarités avec les Domains de Node.js (abandonnés depuis) ou les ThreadLocal en Java. Mais c'est probablement la première fois que l'on voit les Zones en JavaScript : pas d'inquiétude, on va les découvrir ensemble.

Zones

Une zone est un contexte d'exécution. Ce contexte va recevoir du code à exécuter, et ce code peut être synchrone ou asynchrone.
Une zone va nous apporter quelques petits bonus :

- Une façon d'exécuter du code avant et après le code reçu à exécuter ;
- Une façon d'intercepter les erreurs éventuelles d'exécution du code reçu ;
- Une façon de stocker des variables locales à ce contexte.

Prenons un exemple. Si j'ai du code dans une application qui ressemble à :

```
// calcul du score -> synchrone
let score = computeScore();
// mettre à jour le joueur avec son nouveau score -> synchrone
updatePlayer(player, score);
```

Quand on exécute ce code, on obtient :

```
computeScore: new score: 1000
udpatePlayer: player 1 has 1000 points
```

Admettons que je veuille savoir combien de temps prend un tel code. Je peux faire quelque chose comme ça :

```
startTimer();
let score = computeScore();
updatePlayer(player, score);
stopTimer();
```

Et cela produirait ce résultat :

```
start
computeScore: new score: 1000
udpatePlayer: player 1 has 1000 points
stop: 12ms
```

Facile. Mais maintenant, que se passe-t-il si updatePlayer est une fonction asynchrone ? JavaScript fonctionne de façon assez particulière : les opérations asynchrones sont placées à la fin de la queue d'exécution, et seront donc exécutées après les opérations synchrones.
Donc cette fois mon code précédent :

```
startTimer();
let score = computeScore();
updatePlayer(player, score); // asynchrone
stopTimer();
```

va en fait donner :

```
start
computeScore: new score: 1000
stop: 5ms
udpatePlayer: player 1 has 1000 points
```

Mon temps d'exécution n'est plus bon du tout, vu qu'il ne mesure que le code synchrone ! Et c'est par exemple là que les zones peuvent être utiles. On va lancer le code en question en utilisant zone.js pour l'exécuter dans une zone :

```
let scoreZone = zone.fork();
scoreZone.run(() => {
  let score = computeScore();
  updatePlayer(player, score); // asynchrone
});
```

Pourquoi est-ce que cela nous aide dans notre cas ? Eh bien, si la librairie zone.js est chargée dans notre navigateur, elle va commencer par patcher toutes les méthodes asynchrones de celui-ci. Donc à chaque fois que l'on fera un setTimeout(), un setInterval(), que l'on utilisera une API asynchrone comme les Promises, XMLHttpRequest, WebSocket, FileReader, GeoLocation... on appellera en fait la version patchée de zone.js. Zone.js connaît alors exactement quand le code asynchrone est terminé, et permet aux développeurs comme nous d'exécuter du code à ce moment là, par le biais d'un hook.
Une zone offre plusieurs hooks possibles :

- *beforeTask* qui sera appelé avant l'exécution du code encapsulé dans la zone ;
- *afterTask* qui sera appelé après l'exécution du code encapsulé dans la zone ;
- *onError* qui sera appelé dès que l'exécution du code encapsulé dans la zone lance une erreur ;
- *onZoneCreated* qui sera appelé à la création de la zone.

On peut donc utiliser une zone et ses hooks pour mesurer le temps d'exécution de mon code asynchrone :

```
let scoreZone = zone.fork({
  beforeTask: startTimer,
  afterTask: stopTimer
});
scoreZone.run(() => {
  let score = computeScore();
  updatePlayer(player, score);
});
```

Et cette fois-ci ça marche !

```
start
computeScore: new score: 1000
udpatePlayer: player 1 has 1000 points
stop: 12ms
```

Vous voyez maintenant peut-être le lien qu'il peut y avoir avec Angular 2. En effet, le premier problème du framework est de savoir quand la détection de changement doit être lancée. En utilisant les zones, et en faisant tourner le code que nous écrivons dans une zone, le framework a une très bonne vue de ce qui est en train de se passer. Il est ainsi capable de gérer les erreurs assez finement, mais surtout de lancer la détection de changement dès qu'un appel asynchrone est terminé !
Pour simplifier, Angular 2 fait quelque chose comme :

```
let scoreZone = zone.fork({
  afterTask: triggerChangeDetection
});
scoreZone.run(() => {
  // your application code
});
```

Et le premier problème est ainsi résolu ! C'est pour cela qu'en Angular 2, contrairement à AngularJS 1.x, il n'est pas nécessaire d'utiliser des services spéciaux pour profiter de la détection de changements. Vous pouvez utiliser ce que vous voulez, les zones se chargeront du reste.
A noter que **les zones sont en voie de standardisation**, et pourraient faire partie de la spécification officielle ECMAScript dans un futur proche.
Autre information intéressante, l'implémentation actuelle de zone.js embarque également des informations pour WTF (qui ne veut pas dire What The Fuck ici, mais Web Tracing Framework). Cette librairie permet de profiler votre application en mode développement, et de savoir exactement quel temps a été passé dans chaque partie de votre application et du framework. Bref, plein d'outils pour analyser les performances si besoin !

La détection de changement en Angular 2

La seconde partie du problème est la détection de changement en elle-même. C'est bien beau de savoir quand on doit la lancer, mais comment fonctionne-t-elle ?
Tout d'abord, il faut se rappeler qu'une application Angular 2 est un arbre de composants. Lorsque la détection de changement se lance, le framework va parcourir l'arbre de ces composants pour voir si les composants ont subi des changements qui impactent leurs templates. Si c'est le cas, le DOM du composant en question sera mis à jour (seulement la petite portion impactée par le changement, pas le composant en entier). Ce parcours d'arbre se fait de la racine vers les enfants, et contrairement à AngularJS 1.x, il ne se fait qu'une seule fois. **Car il y a maintenant une grande différence** : la détection de changement en Angular 2 ne change pas le modèle de l'application, là où un watcher en AngularJS 1.x pouvait changer le modèle lors de cette phase. Et en Angular 2, un composant ne peut maintenant modifier que le modèle de ses composants enfants et pas de son parent. **Finis les changements de modèle en cascade !**
La détection de changement est donc seulement là pour vérifier les changements et modifier le DOM en conséquence. Il n'y a plus besoin de faire plusieurs passages comme c'était le cas dans la version 1.x, puisque le modèle n'aura pas changé !
Pour être tout à fait exact, ce parcours se fait deux fois lorsque l'on est en mode développement pour vérifier qu'il n'y a pas d'effet de bords indésirables (par exemple un composant enfant modifiant le modèle utilisé par son composant parent). Si le second passage détecte un changement, une exception est lancée pour avertir le développeur.
Ce fonctionnement a plusieurs avantages :

- il est plus facile de raisonner sur nos applications, car on ne peut plus avoir de cas où un composant parent passe des informations à un composant enfant qui lui aussi passe des informations à son parent. Maintenant les données sont transmises dans un seul sens ;
- la détection de changement ne peut plus avoir de boucle infinie ;
- la détection de changement est bien plus rapide.

Sur ce dernier point, c'est assez simple à visualiser : là où précédemment la version effectuait **(M watchers) * (N cycles)** vérifications, la version 2 ne fait plus que **M vérifications**.
Mais un autre paramètre entre en compte dans l'amélioration des performances d'Angular 2 : le temps qu'il faut au framework pour faire cette vérification. Là encore, l'équipe de Google fait parler sa connaissance profonde des sciences informatiques et des machines virtuelles.
Pour cela, il faut se pencher sur la façon dont sont comparées deux valeurs en AngularJS 1.x et en Angular 2. Dans la version 1.x, le mécanisme est très générique : il y a une méthode dans le framework qui est appelée pour chaque watcher et qui est capable de comparer l'ancienne valeur et la nouvelle. Seulement, les machines virtuelles, comme celle qui exécute le code JavaScript dans notre navigateur (V8 si tu utilises Google Chrome par exemple), n'aiment pas vraiment le code générique.

Et si tu le permets, je vais faire une petite parenthèse sur le fonctionnement des machines virtuelles. Avoue que tu ne t'y attendais pas dans un article sur un framework JavaScript ! Les machines virtuelles sont des programmes assez extraordinaires : on leur donne un bout de code et elles sont capables de l'exécuter sur n'importe quelle machine. Vu que peu d'entres nous (certainement pas moi) sont capables de produire du code machine performant, c'est quand même assez pratique. On code avec notre langage de haut niveau, et on laisse la VM se préoccuper du reste. Evidemment, les VMs ne se contentent pas de traduire le code, elles vont aussi chercher à l'optimiser. Et elles sont plutôt fortes à ce jeu là, à tel point que les meilleures VMs ont des performances aussi bonnes que du code machine optimisé (voire bien meilleures, car elle peuvent profiter d'informations au runtime, qu'il est plus difficile voire impossible de connaître à l'avance quand on fait l'optimisation à la compilation).
Pour améliorer les performances, les machines virtuelles, notamment celles qui font tourner des langages dynamiques comme JavaScript, utilisent un concept nommé inline caching. C'est une technique très ancienne (inventée pour SmallTalk je crois, depuis près de 40 ans, une éternité en informatique), pour un principe finalement assez simple : si un programme appelle une méthode beaucoup de fois avec le même type d'objet, la VM devrait se rappeler de quelle façon elle évalue les propriétés des objets en question. Il y a donc un cache qui est créé, d'où le nom, inline caching. La VM commence donc par regarder dans le cache si elle connaît le type d'objet qu'elle reçoit, et si c'est le cas, utilise la méthode optimisée de chargement.
Ce genre de cache ne fonctionne vraiment que si les arguments de la méthode ont la même forme. Par exemple {name: 'Cédric'} et {name: 'Cyril'} ont la même forme. Par contre {name: 'JB', skills: []} n'a pas la même forme.
Lorsque les arguments ont toujours la même forme, on dit que le cache est **monomorphique**, un bien grand mot pour dire qu'il n'a qu'une seule entrée, ce qui donne des résultats très rapides. S'il a quelques entrées, on dit qu'il est polymorphique, cela veut dire que la méthode peut être appelée avec des types d'objets différents, et le code est un peu plus lent. Enfin, il arrive que la VM laisse tomber le cache s'il y a trop de types d'objet différents, c'est ce qu'on appelle un état mégamorphique. Et tu l'as compris, c'est le cas le moins performant.
Si j'en reviens à notre détection de changement en AngularJS 1.x, on comprend vite que la méthode générique utilisée n'est pas optimisable par la machine virtuelle : on est dans un état mégamorphique, là où le code est le plus lent. Et même si les navigateurs et machines modernes permettent de faire plusieurs milliers de vérifications de watchers par seconde, on pouvait quand même atteindre les limites.
D'où l'idée de faire un peu différemment en Angular 2 ! Cette fois, plutôt qu'avoir une seule méthode capable de comparer tous les types d'objet, l'équipe Google a pris le parti de générer dynamiquement des comparateurs pour chaque type. Cela veut dire qu'au démarrage de l'application, le framework va parcourir l'arbre des composants et générer un arbre de *ChangeDetectors* spécifiques.
Par exemple, pour un composant User avec un champ name affiché dans le template, on aura un ChangeDetector qui ressemble à :

```
class User_ChangeDetector {
  detectChanges() {
    if (this.name !== this.previousName) {
      this.previousName = this.name;
      isChanged = true;
    }
  }
}
```

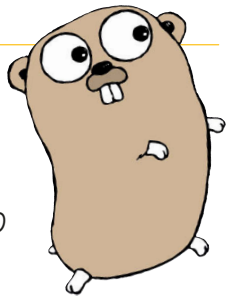
Un peu comme si on avait écrit le code de comparaison à la main. Ce code est du coup très rapide (monomorphique si vous suivez), permet de savoir si le composant a changé, et donc de mettre à jour le DOM en conséquence.

Donc non seulement Angular 2 fait moins de comparaisons que la version 1.x (une seule passe suffit) mais en plus ces comparaisons sont beaucoup plus rapides !

Depuis le début, l'équipe Google surveille d'ailleurs les performances, avec des benchmarks entre AngularJS 1.x, Angular 2 et même React sur des cas d'utilisation un peu tordus afin de voir si la nouvelle version est toujours la plus rapide. Il est même possible d'aller encore plus loin, puisque la stratégie de ChangeDetection peut même être modifiée de sa valeur par défaut, et être encore plus rapide dans certains cas. Mais ça c'est pour une autre fois !

Si vous désirez en savoir plus sur le sujet, l'auteur de cet article est également l'auteur de l'ebook **Devenez un ninja avec Angular 2**, un ebook à prix libre en vente à l'adresse https://books.ninja-squad.com.
Et comme on aime les lecteurs de Programmez, on vous propose d'en gagner en répondant à la question suivante : "Qu'est-ce que génère Angular 2 au démarrage de l'application pour améliorer les performances?". Twittez votre réponse en mentionnant "@NinjaSquad" et le hashtag "#programmez" et gagnez l'un des trois ebooks en jeu ! Le concours est ouvert jusqu'à fin Mai, bonne chance !

Découverte de Golang



Face à des problématiques de complexité croissante (maintenance, relecture du code) et de lenteur des temps de build sur des applications monolithiques écrites en C++, Rob Pike et Ken Thomson créèrent Go en 2007. Ce langage repose sur un typage statique avec une syntaxe dérivée du C, avec des fonctionnalités additionnelles telles qu'un ramasse-miettes et des capacités de typage dynamique. Il vient avec une librairie standard riche permettant de réaliser rapidement des projets Web. L'intention de Go est de permettre de résoudre des problèmes complexes avec des outils simples.

Henri LEPIC
Fondateur de Digikong.io, ingénierie & formation aux technologies Go-centriques

CONCEPTS CLÉS DU LANGAGE

Variables, constantes et pointeurs

Les variables

Go est un langage fortement typé où des types natifs sont inclus tels que l'int, le string ou le booléen. D'autres types peuvent être créés en réalisant des structures de types natifs. La déclaration d'une variable peut se faire simplement par inférence automatique de type avec l'opérateur :=.

```
name := "Scrat"
```

Dans cet exemple la variable name sera de type string, cette simplification est réalisée par le compilateur lui-même qui est capable de reconnaître les types natifs et composés. Cependant, il est toujours possible de déclarer une variable avec une déclaration explicite du type. Cet usage peut être utile dans la mesure où au runtime, on n'a pas toujours à disposition la valeur à assigner dans la variable. Voici une déclaration sans inférence de type, suivie de l'assignation d'une valeur à la variable.

```
var name string
name = "Scrat"
```

Nous pouvons aussi initialiser une valeur à une variable de manière explicite :

```
var name string = "Scrat"
```

La déclaration d'un ensemble de variables peut être regroupée avec des parenthèses :

```
var (
  name string = "Scrat"
  age int
)
```

La déclaration d'un ensemble de variables peut ou non faire l'objet d'une initialisation. Par défaut l'initialisation d'une variable se fait sur sa valeur 0. Dans notre exemple la valeur de la variable age aura comme valeur par défaut 0 qui est la valeur par défaut du type int.
La portée d'une variable est descendante. C'est-à-dire que tous les blocs de code enfants en héritent. À l'inverse si la déclaration est dans un sous-bloc de code ; les blocs parents n'en bénéficient pas. Un sous-bloc peut déclarer une variable de même nom que celle déclarée dans un bloc parent mais dont la portée sera limitée à ce sous-bloc :

```
package main

import "fmt"

func main() {
  var i int = 0 // output : i declared and not used
  {
    var i int = 1
    fmt.Println(i)
  }
}
```

Dans l'exemple précédent, nous obtenons une erreur de compilation : la variable i au niveau de la fonction main() n'est pas utilisée par le programme. Le compilateur Go détecte toujours les variables non utilisées et renverra une erreur afin de ne pas affecter inutilement de la mémoire.

Les constantes

Pour rappel, une constante ne peut changer de valeur une fois initialisée. Globalement, une constante a le même comportement qu'une variable avec en plus sa capacité à gérer des nombres avec une plus grande précision, puisqu'elle peut être codée sur 256 bits. Une autre différence avec les variables est qu'il n'est pas possible d'utiliser l'inférence de type avec les constantes.

```
const (
  HELLO = "world"
)
```

Avec l'utilisation de iota, il est possible de générer une suite très simplement.

```
const (
  ONE = iota +1
  TWO
  THREE // output 3
)
```

Les pointeurs

La syntaxe d'utilisation des pointeurs reprend celle du langage C : l'opérateur & renvoie l'adresse d'une valeur et * va permettre de déréférencer un pointeur vers la valeur pointée.

```
monTexte := "texte à partager"
print(&monTexte) // affiche une adresse mémoire, ex : 0x10327f80
```

Dans cet exemple, la valeur &monTexte est un pointeur vers l'adresse mémoire de la valeur de la variable monTexte. Si je déréférence cette valeur, j'obtiens la valeur de ma variable :

```
newVar := &monTexte
print(*newVar)  // sortie : texte à partager
```

On peut par conséquent créer une variable avec une inférence de type et récupérer son pointeur directement.

```
maVar := &string{"texte à partager"}
```

Attention : dans ce cas le type de maVar sera de type pointeur d'une string, par conséquent, &string. Dans un cadre d'usage classique, il est recommandé de créer des variables par valeur et non par pointeur afin qu'à la relecture, le partage par pointeur de la valeur soit plus explicite.

Tableaux, slices et maps

Les tableaux

Les tableaux de variables Go permettent d'organiser des suites de variables de manière contiguë dans un espace mémoire. L'initialisation d'un tableau se fait de la façon suivante :

```
var Tableau [2]int
```

Il est possible de créer un tableau avec l'usage de l'inférence de type et d'initialiser sur la même ligne les valeurs du tableau. Nous pouvons donc écrire :

```
Tableau := [2]int{1,2}
```

Nous avons créé ici un tableau d'une longueur de 2 éléments (nombre entre les crochets), tableau qui contient des valeurs de type int qui sont initialisées à 1 et 2 pour les positions respectives de 0 et 1 du tableau. Le nombre d'éléments que contient le tableau peut être omis et remplacé par "...". Le compilateur calculera alors le nombre d'éléments à initialiser dans le tableau, soit :

```
var Tableau [...]int{1,2}
print(len(Tableau)) // output : 2
```

La contrainte d'un tableau est qu'il ne peut pas changer de taille.

Les slices

Une slice est un type référentiel vers un tableau dont la taille peut évoluer. En d'autres termes, c'est un pointeur sur un tableau. Ce qui veut dire que le passage par paramètres dans une fonction est une copie du pointeur, par conséquent très légère.
L'opération de création d'une slice est semblable au tableau, il faut juste omettre le nombre d'éléments qu'elle contient.

```
var Slice []int
```

J'ai ici une slice de variable de type int appelée slice. Par défaut cette slice est initialisée avec une longueur de 0 et une capacité de 0. Pour ajouter ou soustraire des éléments dans ma slice, je dispose de la fonction append().

```
Slice = append(Slice, 1)
```

Une slice peut être copiée en sous-slice comme dans l'exemple suivant :

```
SliceA := []string{"Scrat","Sid","Ellie","Manny"}
SliceB := SliceA[1:2]
fmt.Print(SliceB) // ouput : [Sid]
```

La première valeur entre les crochets est la position de départ de la copie, la seconde valeur est la position de l'élément où s'arrête la copie. Dans la mesure où une slice est un type référentiel, la copie de la slice fait référence au même tableau. En d'autres termes, si je modifie un des éléments de SliceB, il sera également modifié dans SliceA.

```
SliceB[0] = "Diego"
fmt.Println(SliceA) // output : [Scrat Diego Ellie Manny]
```

Lors de la capture d'une slice, un troisième paramètre optionnel permet de gérer la capacité :

```
SliceC := SliceA[1:2:2]
fmt.Print(cap(SliceC)) // ouput : 2
```

En gérant ainsi la capacité, il est possible après l'utilisation d'append() de détacher le tableau de référence d'origine. Cette opération va attribuer un nouveau tableau à la slice. La référence du tableau de SliceC ne sera plus la même que celle de la SliceA.

```
SliceA := []string{"Scrat","Sid","Ellie","Manny"}
SliceB := SliceA[1:2:2]
SliceB = append(SliceB,"Rob")
fmt.Println(SliceA,SliceB) // output : [Scrat Sid Ellie Manny] [Sid Rob]
```

On remarque dans l'exemple que la SliceA n'est pas affectée par l'ajout d'un élément dans la SliceB. Pour simplifier, si vous désirez réaliser une copie originale, il est possible d'utiliser la fonction copy() qui va automatiquement créer une nouvelle référence de tableau.

```
sliceA := []string{"Scrat","Sid","Ellie","Manny"}
sliceB := make([]string, len(sliceA),cap(sliceA))
copy(sliceB, sliceA)
sliceB[1] = "Rob"
fmt.Println(sliceA,sliceB) // output : [Scrat Sid Ellie Manny] [Scrat Rob Ellie Manny]
```

Nous pouvons remarquer l'usage de make() qui va permettre d'allouer une slice en spécifiant sa longueur et capacité.

Les maps

Une map est une liste de valeurs accessibles par une clé. La création d'une map nécessite l'usage du mot clé make.

```
var MaMap := make([string]string)
```

Une map est encore plus souple qu'une slice : il est possible d'ajouter une valeur sans avoir à changer la taille d'un tableau.

```
MaMap["Manny"] = "Quand est-ce que tu as perdu la tête ?"
```

À partir du dernier exemple, la récupération d'une valeur se fait comme suit :

```
citation := MaMap["Manny"]
```

Afin de supprimer une valeur dans une map, nous pouvons faire appel à la fonction delete()

```
delete(MaMap, "Manny")
```

Pour s'assurer qu'il y a bien une valeur récupérable dans une map, une deuxième variable booléenne est accessible :

```
citation, ok := MaMap["Manny"]
if !ok {
  // il n'y a pas de données dans la map à la position Manny
  ...
}
```

Dans notre cas, la première valeur citation est une string vide et la seconde valeur est de type bool et permet de savoir si la donnée dans la map existe. Ce deuxième paramètre peut être pratique pour différencier des valeurs de type par défaut avec de véritables valeurs.

Les signatures de fonction

Une signature de fonction comporte le ou les types qu'elle a en paramètre et le ou les types qu'elle retourne. La déclaration d'une fonction se fait via

le mot clé func. Dans l'exemple suivant, nous utilisons le package errors afin de créer un message d'erreur.

```
func Print(m string) error {
  if len(m)==0{
     return errors.New("no message to display")
}
fmt.Println(m)
return nil
}
```

Ici dans notre exemple, nous avons une fonction appelée Print. Cette fonction récupère en paramètre un message de type string qui retourne une error. À l'intérieur de la fonction, nous avons la condition if qui évalue si la chaîne envoyée en paramètre est vide, auquel cas la fonction renvoie une erreur avec le message "no message to display". Si la chaîne envoyée en paramètre de la fonction n'est pas vide, alors la fonction affiche le message via la fonction Println() du package fmt.

Les paramètres de fonction

Une fonction peut recevoir plusieurs paramètres du même type défini une fois :

```
func Concat(a, b, c string) string {
  return a + b + c
}
```

Dans le cas où les variables ont un même type, il n'est pas obligatoire d'expliciter répéter leur type. Une fonction peut aussi être variadique et avoir un nombre indéfini de paramètres :

```
func ConcatN(n ...string) string {
  var res string
  for _, v := range n {
     res += v
  }
  return res
}
```

La fonction ConcatN() pourrait être appelée avec autant de paramètres que nécessaire. Chaque paramètre doit être séparé par une virgule ou par ... pour le cas d'une slice passée en paramètre.

```
resA := ConcatN("a","b","c")
resB := ConcatN([]string{"a","b","c"}...)
fmt.Println(resA,resB) // output : abc abc
```

Les retours de fonction

Les retours peuvent omettre de spécifier les variables renvoyées si elles sont déjà nommées dans la liste des variables de retour dans le prototype de la fonction :

```
func GiveAge(a int) (age int, err error) {
  age = a
  if age < 0 {
     err = errors.New("are you born yet?")
  }
  return
}
```

La récupération de multiples valeurs de retour se fait dans le même ordre que la signature :

```
a, err := GiveAge(30)
```

Nous récupérons ici l'âge dans la variable a et une erreur err. Dans la majorité des cas une fonction n'a pas plus de deux variables de retour. Par convention, la première est la donnée à récupérer et la seconde est une erreur à récupérer si besoin.

Structures et compositions

Les structures

Il est possible de créer ses propres structures (types composites) à partir des types natifs.

```
type user struct{
  name string
  age int
}
```

Après la déclaration d'une structure, nous pouvons utiliser ce nouveau type avec des valeurs.

```
var u user
u.name = "Scrat"
u.age = 14
fmt.Println(u) // output : {Scrat 14}
```

Notez l'utilisation de nom de variable courte dans notre exemple u. Cet usage idiomatique incite le développeur à se focaliser sur des petites portions de code. Nous pouvons aussi, avec une inférence de type, créer la variable u et par là-même, initialiser ses valeurs telles que :

```
u := user{"Scrat",14}
fmt.Println(u) // output : {Scrat 14}
```

Composition

Une structure de type peut être enrichie avec d'autres types existants via la composition de type. Nous voulons par exemple créer un nouveau type admin à partir du type user adjoint d'une autre propriété rightLevel de type int.

```
type admin struct{
  user
  rightLevel int
}
```

Tous les champs de user seront alors disponibles avec en plus le nouveau champ rightLevel

```
a := admin{"Sid",15,1}
fmt.Println(a) // output : {Sid 15 1}
```

Méthodes et receveurs

Tel que nous l'avons déjà vu nous pouvons en Go créer des structures qui sont des types composites. Les méthodes ne sont que des fonctions rattachées à ces types.

Les méthodes

Dans le cas suivant nous avons un type user avec comme propriétés name et age de type respectivement string et int. Une fonction sayHello() est déclarée ayant comme paramètre user, qui permet de récupérer la propriété name de user afin d'afficher un message.

```
package main

import "fmt"

type user struct {
  name string
```

```
  age int
}

func sayHello(u user) {
  fmt.Println("Hi I'm", u.name, "!")
}

func main() {
  u := user{
     "Scrat",
     14,
  }
  sayHello(u)
}
```

Nous pouvons réécrire la fonction sayHello() telle que :

```
func (u user)sayHello() {
  fmt.Println("Hi I'm", u.name, "!")
}
```

La fonction sayHello() est dorénavant une méthode attachée au type user, ce qui lui permettra d'appeler la fonction directement depuis une variable de type user.

```
u.sayHello()
```

À savoir sur les méthodes : tout nom de méthode commençant par une majuscule est exporté et accessible à tout code utilisateur du package. Rendre visibles sélectivement certaines parties de l'extérieur du package permet de penser le package sous forme d'API.

Le receveur pointeur

Un receveur d'une méthode peut être de type pointeur.

```
func (u *user) changeName(n string) {
  u.name = n
}
```

Plusieurs raisons sont valables pour utiliser un type pointeur. D'une part le pointeur tel que déjà évoqué est plus léger qu'une copie de valeur lors de l'appel de la méthode. D'autre part, le receveur pointeur permet de modifier des membres du receveur.

Gestion des erreurs

Dans une communication réseau ou lors d'un traitement complexe, des erreurs peuvent survenir. Le développeur doit pouvoir décider de la bonne stratégie à adopter. Par exemple, dans le cas de la validation d'un formulaire Web, si une des données est mal formée, le programme doit pouvoir renvoyer une erreur à l'utilisateur. La manière la plus courante de retourner les erreurs est d'utiliser le package errors. Ce dernier package peut être étendu pour un usage plus avancé.

```
package validform

import (
  "errors"
  "fmt"
)

func main() {
  fmt.Println(validateName("Su"))
}

func validateName(e string) (bool, error) {
  if len(e) < 2 {
     return false, errors.New("too short")
  }
  return true, nil
}
```

Dans une fonction, la façon la plus idiomatique de retourner une erreur est de l'avoir en seconde valeur de retour. De plus, il arrive que des messages d'erreur soient régulièrement utilisés, ce pourquoi nous pouvons les stocker au préalable dans des déclarations de variables groupées.

```
var (
  ErrInvalidTooShort = errors.New("validform: too short")
  ErrInvalidEmail = errors.New("validform: invalid email")
)
```

Remarquez ici l'utilisation de noms de variables plus longs et explicites, permettant une meilleure compréhension. La première expression validform dans le constructeur de l'erreur permet de faire référence au package concerné, ce qui peut s'avérer pratique si l'information est historisée. Enfin l'usage d'une déclaration groupée d'erreurs est une bonne pratique afin de faciliter la réalisation de tests unitaires.

Les Interfaces Go

Même si le concept d'interface en Go partage des notions avec ce que l'on connaît dans d'autres langages, sa logique sous-jacente est particulière. Une interface Go est un type définissant simplement un ensemble de méthodes. Dans le concept OO classique, tout type voulant se conformer à une interface le signale explicitement dans sa déclaration par un lien d'héritage ou avec un mot clé. Dans Go, tout type est automatiquement conforme à une interface s'il dispose d'un ensemble méthode identique à celui listé dans la définition de cette interface.

L'interface vide

L'interface vide, notée interface{}, pourrait s'apparenter à une classe de base comme java.lang.Object. Son ensemble de méthodes étant vide, une valeur interface vide peut se voir affecter n'importe quelle valeur de type :

```
var i interface{}
i = "a"
i = 1
print(i) // output : 1
```

Définition des méthodes d'une interface

Une interface est un ensemble de méthodes qui peut être vide ou comprendre une ou plusieurs méthodes. D'un type qui implémente le même ensemble de méthodes, on peut dire qu'il réalise cette interface. C'est-à-dire qu'il pourra être affecté à une valeur interface de ce type d'interface.

```
type Stringer interface{
  String()string
}
```

Utilisation d'une interface

"Si C++ et Java sont concentrés sur les hiérarchies de types et leurs taxonomies, Go rime plus avec composition."

– Rob Pike

Nous pouvons utiliser une interface de la manière suivante :

```
package main

import (
  "fmt"
)

type user struct{
  Name string
}

func (u user) String() string{
  return "Hi I'm " + u.Name
}

func main() {
  u := user{"Rob"}
  fmt.Println(u) // output : Hi I'm Rob
}
```

Cet exemple permet de démontrer la simplicité par laquelle il est possible d'implémenter une interface. Si la fonction `(u user) String() string` n'existait pas, la sortie aurait été `{ Rob }`. Le simple fait d'avoir créé cette méthode permet au type `user` d'être compatible avec le contrat de l'interface `Stringer` utilisé dans le package `fmt`.

Composition d'interfaces

Il est possible de composer plusieurs interfaces entre elles de la même manière qu'une structure.

```
type ReaderWriter interface{
Reader
Writer
}
```

Dans ce cas, ce sont toutes les méthodes appartenant à `Reader` et `Writer` qui sont incluses dans le type d'interface `ReaderWriter`. Ce qui veut dire que tous les types qui veulent utiliser cette interface doivent implémenter les méthodes de `Reader` et `Writer`.

Goroutine, mutex, atomic, channel

Go est aussi connu pour son mot clé `go` qui permet de lancer des traitements asynchrones : on parle alors de concurrence. La concurrence s'apparente à un thread léger. Par défaut, il est possible de créer jusqu'à 100 000 goroutines en même temps dans un même programme. Ce nombre astronomique est dû à la légèreté d'une goroutine (environ 2Ko), contrairement à ce que pourrait être un thread nécessitant plus de mémoire. Afin de simplifier les actions d'écriture et de lecture sur les mêmes variables, Go dispose de différents moyens à cet effet.

Les mutex

Les mutex vont suspendre les partages de ressources entre les goroutines. Une seule goroutine sera en mesure d'exécuter la portion de code entre les deux actions de Mutex `Lock()` et `Unlock()`.

```
package main

import (
  "fmt"
  "sync"
)

var (
  mu        sync.Mutex
  increment  int64
  nGoroutines int = 10
  wg        sync.WaitGroup
)

func main() {
  wg.Add(nGoroutines)
  for i := 0; i < nGoroutines; i++ {
     go CostCall()
  }
  wg.Wait()
  fmt.Println(increment)
}

func CostCall() {
  mu.Lock()
  {
     increment++
     wg.Done()
  }
  mu.Unlock()
}
```

Dans cet exemple nous avons bien évidement l'usage des `Mutex`, mais aussi des `WaitGroups` sans lesquels le code exécuté ne pourrait pas attendre le retour des goroutines avant d'avoir fini. Comme nous avons pu le voir précédemment, les actions réalisées avec les goroutines sont asynchrones. Ce qui veut dire que la fonction principale `main()` continuera son déroulement d'exécution jusqu'à sortir de la fonction avant d'avoir attendu la fin des appels de la fonction `CostCall()`.

Les fonctions atomiques

Dans certaines opérations de lecture et d'écriture avec des entiers, il est préférable d'utiliser les fonctions atomiques. Les actions réalisées avec le package `atomic` sont pré-écrites en assembleur et optimisées par type d'OS et de CPU. En utilisant l'exemple précédent, nous aurions étés capables d'écrire la fonction `CostCall()` comme suit après l'ajout du package `sync/atomic` et la suppression des Mutex.

```
func CostCall() {
  atomic.Addint64(&increment,1)
wg.Done()
}
```

Les fonctions atomiques permettent de lire et d'écrire de manière concurrente dans des variables de type `int32` et `int64` uniquement. L'usage des fonctions atomiques reste donc limité à des compteurs.

Les channels

Si le concept du channel est relativement simple, son usage requiert en revanche une pratique idiomatique. Pour comprendre le fonctionnement d'un channel, on peut penser à un conduit dans lequel on envoie un ou plusieurs éléments. On parle de channel non bufferisé quand il s'agit d'un seul élément et de channel bufferisé quand on désire écrire plusieurs éléments. Un channel permet de synchroniser les accès à la mémoire, et de ce fait d'éviter des problèmes d'écriture lecture en cas de concurrence. Pour créer un channel, nous avons recours au mot clé `make()`.

```
ch := make(chan int)
```

Pour reprendre le premier exemple des Mutex, nous pouvons obtenir le code suivant en utilisant les channels :

```
package main

import (
  "fmt"
  "sync"
)

var (
  nGoroutines int = 10
  wg        sync.WaitGroup
)

func main() {
  c := make(chan int, 1)
  wg.Add(nGoroutines)
  for i := 0; i < nGoroutines; i++ {
     go CostCall(c)
  }
  c <- 0
  wg.Wait()
  close(c)
  fmt.Println(<-c)
}

func CostCall(c chan int) {
  i := <-c
  i++
  c <- i
  wg.Done()
}
```

Dans cet exemple, nous pouvons remarquer que l'utilisation d'un channel se fait via l'opérateur <-. Cet opérateur permet d'obtenir la valeur du channel s'il lui précède et également d'affecter une valeur au channel s'il lui succède. Nous utilisons ici un channel bidirectionnel qui envoie et reçoit une donnée de type int.
Un autre type de channel existe le channel unidirectionnel. Il envoie ou reçoit des valeurs.

```
out := make( chan<- int)  // utilisé uniquement pour envoyer un int
in := make(<-chan int)   // utilisé uniquement pour recevoir un int
```

RÉALISER SA PREMIÈRE APP

Nous allons, pour cet exercice, créer une application simple.

Installer Go sur sa machine

Rendez-vous sur golang.org/dl dans la rubrique "Featured downloads" ;
Téléchargez l'installeur qui correspond à votre machine (Win, Mac, Linux) ;
Une fois le fichier téléchargé, exécutez-le et suivez les étapes d'installation ;
Pour la configuration, suivez les instructions présentes sur golang.org/doc/install et testez.

Compositions dans *L'Âge de glace*

Description de l'exercice

Pour cet exercice, nous allons approfondir le concept d'interface et de composition. Notre exemple utilise Scrat et Manny, personnages emblématiques de *L'Âge de glace*. Scrat et Manny partagent tous deux des propriétés communes comme leur prénom et ont la capacité de réaliser des choses telles que voyager ou manger. L'application que nous allons réaliser va nous permettre de créer des actions concrètes à partir d'interfaces abstraites.

Créer des structures

Manny est un mammouth, et Scrat un écureuil.

```
type Mammoth struct {
  Name string
}

type Squirrel struct {
  Name string
}
```

Leur propriété commune Name leur permet de décrire leur identité à chacun.

```
type Identity struct{ Name string }

type Mammoth struct {
  Identity
}

type Squirrel struct {
  Identity
}
```

Attacher des méthodes et implémenter l'interface Stringer

Nous pouvons commencer par les aider à décliner leur identité en implémentant l'interface Stringer du package fmt.

```
func (m Mammoth) String() string {
  return "Hello, I'm a Mammoth and my name is " + m.Name + "."
}

func (sq Squirrel) String() string {
  return "Hi, I'm a Squirrel and my name is " + sq.Name + "."
}
```

Avec ces méthodes, si nous utilisons la fonction fmt.Println(), ils auront un message approprié à leur espèce.

```
m := Mammoth{Identity{"Manny"}}
s := Squirrel{Identity{"Scrat"}}
fmt.Println(m) // output : Hello, I'm a Mammoth and my name is Manny.
fmt.Println(s) // output : Hi, I'm a Squirrel and my name is Scrat.
```

Création d'une première interface

Aussi, nous pouvons dire que Manny et Scrat sont tous deux des animaux et qu'ils savent tous deux décliner leur identité.

```
type Animal interface {
  String() string
}
```

Ce qui nous permet de créer une fonction qui prend en paramètre un type abstrait Animal.

```
func Talk(a Animal) {
  fmt.Println(a)
}
```

Cette fonction peut être appelée avec un des animaux en paramètre. Les types Mammoth et Squirrel implémentent la fonction Talk() dans son type concret. L'interface Animal peut par conséquent se réaliser.

```
Talk(m) // output : Hello, I'm a Mammoth and my name is Manny.
Talk(s) // output : Hi, I'm a Squirrel and my name is Scrat.
```

Une nouvelle possibilité est d'avoir la capacité de stocker dans une même slice les types Mammoth et Squirrel. Ce qui nous permet de boucler dans la liste en appelant la même fonction Talk().

```
as := []AnimalAction{&m, &s}
for _, a := range as {
  Talk(a)
}
```

Composer des interfaces avec des types

Maintenant, nous aimerions attacher des actions aux types Mammoth et Squirrel. Pour plus de souplesse, nous voulons attacher les actions au runtime et non à la compilation. Nous allons créer une nouvelle interface Action. avec une méthode abstraite Do().

```
type Action interface {
  Do()
}

type Mammoth struct {
  Identity
  Action
}

type Squirrel struct {
  Identity
  Action
}
```

Pour le moment la fonction Do() est de type abstrait. Nous allons implémenter cette fonction afin de lui donner des actions concrètes à réaliser. Nous voulons ajouter deux comportements possibles qui peuvent être interchangeables entre les deux types Mammoth et Squirrel.

```
type Eat struct {
  Food string
}

func (e Eat) Do() {
  fmt.Println(e.Food + " is verry testy")
}

type Travel struct {
  Goal string
}

func (t Travel) Do() {
  fmt.Println("Ok I'm going to " + t.Goal)
}
```

Composer plusieurs interfaces

Okay, tout est prêt ! Enfin, presque ! Il nous faut, pour finir, créer une interface AnimalAction qui compose les deux interfaces Animal et Action. Mais pourquoi, me direz-vous, compliquer les choses ainsi ? Vous vous rappelez de la liste d'animaux que nous avions créée ? Cette liste pourra de cette manière bénéficier de comportements polymorphiques.

```
type AnimalAction interface {
  Animal
  Action
}
...
func main(){
  ...
  as := []AnimalAction{&m, &s}

  for _, a := range as {
    Talk(a)
    Do(a)
  }
}

func Do(a Action) {
  a.Do()
}
```

Récapitulatif du code

Vous pouvez retrouver l'ensemble de l'exemple sur la zone du play http://play.golang.org/p/HwWvFnHjwU

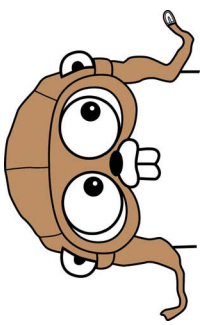
Cliquez sur le bouton Run en haut à gauche afin d'exécuter le code et d'afficher dans le navigateur les sorties du programme. Ce programme nous a permis d'expérimenter les capacités de découplage de Go avec la composition d'interfaces. Nous pourrions dans un second temps renommer le package par agedeglace et ajouter des types et des méthodes concrètes sans casser le code existant dans un nouveau package afin d'en étendre les usages.

CONCLUSION

Beaucoup de développeurs se plaignent de ne pas retrouver dans Go certaines features présentes dans leur langage de prédilection. Souvent, le débat revient sur le sujet des génériques, mais il est difficile de croire qu'un jour ces features arriveront dans Go, au regard des positions fermes de la Go Team. Cependant, cela n'a pas empêché des sites à fort trafic de l'adopter, tels que Le Bon Coin, Twenga, ou Dailymotion pour ne citer qu'eux. Ce qui a sans doute conféré le plus de crédibilité à Go, ce sont des sociétés comme Hashicorp et la formidable communauté open source, qui a produit de nombreux projets ouverts tels que Docker, Kubernetes, ou encore Go-kit. On pourrait légitimement se poser la question suivante : quel type de développeur faut-il être pour écrire du Go, au regard des projets qui, pour la plupart, sont encore très orientés système et devops ? Mon meilleur conseil est que vous vous fassiez votre propre avis sur la question en allant plus loin avec la lecture d'un ouvrage comme *Go in Action* de William Kennedy, ou, dans un registre plus académique, *The Go Programming Language*, de Brian W. Kernighan et Alan A. Donovan.

Ouvrages recommandés

- Go in action de William Kennedy
- The Go programming Language de Alan A. Donovan (Author), Brian W. Kernighan



Sites internet

- le tour de go (https://go-tour-fr.appspot.com/welcome/1)
- goinggo.net
- Talks (https://talks.golang.org/)

Forums

- Golang nuts (https://groups.google.com/forum/#!forum/golang-nuts)
- GolangBridge (https://forum.golangbridge.org/)

Démystifier la création des graphiques en 3D

La 3D envahit notre quotidien visuel ; à travers les jeux vidéo, les publicités à la télé, le cinéma, les documentaires et aussi la présentation de projets Architecturaux... elle est partout !

Zhe (Aurore) Li
Ingénieur Big Data - **Centre Aquitain des Technologies de l'Information et Electroniques**

Mais alors comment fait-on de la 3D ? Est-ce accessible pour les particuliers, ou réservé aux professionnels ? Faut-il nécessairement posséder un logiciel très cher ? Cet article est là pour répondre à toutes ces questions et pour vous aider à faire les premiers pas vers le monde de la 3D.

Les éléments essentiels pour orchestrer une scène 3D

L'ordinateur travaille sur le rendu d'une scène 3D à partir du point de vue de la caméra, en tenant compte des effets de la lumière venant des différentes directions et de la façon dont les lumières interagissent avec les objets 3D dans la scène qui sont décrits par des géométries et par des matériaux. Ce procédé est similaire à la prise d'une photo.

Autrement dit, une scène 3D est composée d'objets 3D, de la lumière et d'une caméra.

Le système de coordination (World Space) :

Avant de parler des détails des éléments dans une scène 3D, on doit savoir comment on positionne ces éléments dans l'espace. Dans l'infographie 3D, on utilise souvent le système de coordonnées cartésiennes. Dans ce système, on définit un point d'origine auquel tous les éléments de l'espace font référence et trois directions X, Y et Z comme le montre l'image ci- dessous : Fig.1.

Cela permet de définir la position de n'importe quel point dans l'espace en donnant les positions sur les trois axes X,Y et Z comme (2,3,4). Fig.2.

Il existe deux règles pour définir le devant (front) de l'objet 3D. Les Coordonnées cartésiennes 3D sont en RHS. L'axe X pointe vers la droite, l'axe Y pointe vers le haut, et l'axe Z est orienté sur l'écran. RHS est dans le sens anti-horaire (CCW). Certains logiciels graphiques (tels que Microsoft Direct3D) utilisent la règle de la main gauche (LHS), où l'axe Z est inversé. LHS est dans le sens horaire (CW).

La caméra :

La caméra représente l'endroit où l'on se place pour regarder le monde. Il y a plusieurs façons de le faire. Ici, nous allons voir trois types de caméras différentes : Fisheye, caméra perspective et caméra orthogonale. Fig.3.

FishEye peut être utilisé pour le rendu de la scène 3D mais en pratique on l'utilise rarement.

Notre œil est habitué à la projection en perspective, car les objets distants apparaissent plus petits. Fig.4.

La projection orthographique semble un peu étrange au début, car les objets conservent les mêmes dimensions quel que soit leur éloignement. C'est une technique souvent utilisée pour dessiner les plans des architectes par exemple. Fig.5.

L'orientation de la caméra :

Une caméra est aussi un objet placé dans l'espace. Fig.6.

Définir sa **position** est trivial en donnant (0,0,2) par exemple. Mais nous avons besoin également de définir l'orientation de la caméra pour savoir vers où la caméra regarde le monde. Pour cela, nous utilisons une technique appelée "LookAt" système. Cela consiste à définir la position d'un point fixe (**target**) par exemple (0,0,0) dans l'espace vers lequel la caméra regarde. Ensuite, nous devons savoir comment **le haut de la caméra** s'oriente également en donnant un vecteur (0,1,0) (qui est l'axis Y+) par exemple.

Avec ces trois éléments, nous obtenons une référence complète de notre caméra dans l'espace. Mais pour le rendu d'une scène 3D, la position de la caméra dans l'espace ne nous intéresse pas. Nous voulons positionner l'espace par rapport à la caméra parce que nous regardons le monde au travers de la caméra. Cela veut dire que tous les objets positionnés dans l'espace nécessitent une transformation des coordonnées faisant référence à l'espace vu de la caméra. Fig.7.

Ce travail s'effectue dans le pipeline graphique par le GPU.

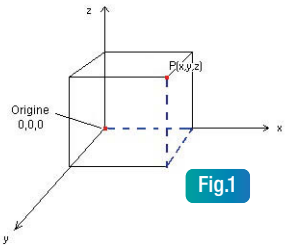


Fig.1

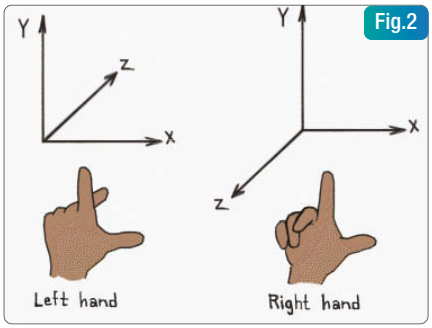


Fig.2

[http://viz.aset.psu.edu/gho/sem_notes/3d_fundamentals/html/3d_coordinates.html]


Fig.3 © J.F. Stuefer de Europe

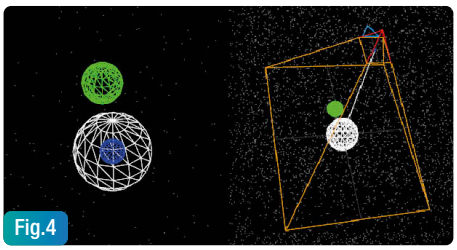


Fig.4

[http://threejs.org/examples/#webgl_camera]

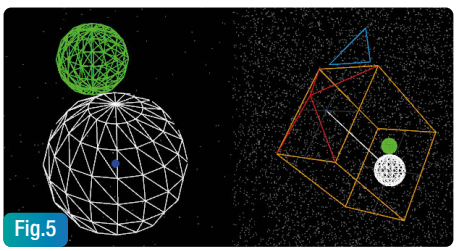


Fig.5

[http://threejs.org/examples/#webgl_camera]

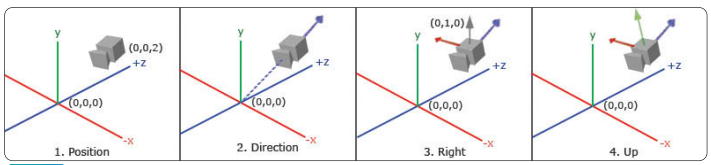


Fig.6 [http://learnopengl.com/#!Getting-started/Camera]

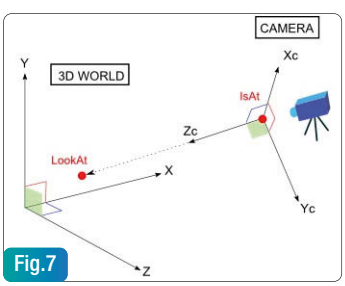


Fig.7

Détails sur la caméra perspective :

Dans cet article, nous allons voir plus de détails sur la caméra perspective parce que la vue présentée par cette technique est proche de la façon dont on regarde le monde réellement. (Fig.8) :

- Le ratio d'aspect : le ratio entre la largeur et la hauteur de la zone rectangulaire qui sera la cible de la projection ;
- Le champ vertical : l'angle vertical de la caméra au travers duquel nous regardons le monde ;
- La position du plan Z proche qui nous permet de découper les objets trop proches de la caméra ;
- La position du plan Z lointain qui nous permet de découper les objets trop loin de la caméra.

La lumière

Maintenant, nous allons explorer la lumière dans le rendu 3D.
Il y a plusieurs types de sources de lumière dans le monde de l'infographie :
La lumière qui vient d'une direction spécifique qui n'éclaire que les objets situés dans cette direction comme le soleil. Nous l'appellerons lumière directionnelle (**Directional light**).
La lumière qui vient de partout et éclaire tout de façon uniforme, indépendamment de la face exposée... On l'appelle lumière ambiante (**Ambient light**).
La lumière qui projette dans une direction avec un faisceau elliptique comme un spot. On l'appelle projecteur (**Spot light**).
La lumière qui vient d'un point central comme une ampoule et projette dans toutes les directions. On l'appelle lumière omnidirectionnelle. (**Point light**) Fig.9.
Pour définir les lumières plus concrètement avec Three.js, on peut définir la couleur, l'intensité et la position de la lumière.

```
ambientLight = new THREE.AmbientLight( 0x444444 );
scene.add( ambientLight );
pointLight = new THREE.PointLight( 0xffffff, 1.25, 1000 );
pointLight.position.set( 0, 0, 600 );
scene.add( pointLight );
directionalLight = new THREE.DirectionalLight( 0xffffff );
directionalLight.position.set( 1, -0.5, -1 );
scene.add( directionalLight );
```

Les objets 3D :

Le dernier élément essentiel pour le rendu d'une scène 3D ce sont les objets. Nous pouvons définir les formes, les tailles et les orientations de ces objets via les outils de modélisation ou par les programmes. Les objets 3D sont constitués de sommets, d'arêtes et de faces organisés en polygones sous forme de fil de fer. On l'appelle aussi le **maillage** qui représente la surface d'un objet 3D. Chaque sommet peut contenir des informations comme la position, la couleur, les coordonnées de la texture et le vecteur normal à un plan. Fig.10.

```
//Restons sur un matériel basique
var material = newTHREE.MeshBasicMaterial({ color: 0xaaaaaa });
//Créer les différentes géométries
cube = new THREE.BoxGeometry( 90, 90, 90 );
sphere = new THREE.SphereGeometry( size, 24, 16 );
cylinder = new THREE.CylinderGeometry( 50, 50, 110, 16 );
torus = newTHREE.TorusGeometry( 50, 20, 8, 20 );
mesh1 = new THREE.Mesh( cube, material );
 scene.add( mesh1 );
mesh2 = new THREE.Mesh( sphere, material );
scene.add( mesh2 );
mesh3 = new THREE.Mesh( cylinder, material );
scene.add( mesh3 );
mesh4 = new THREE.Mesh( torus, material );
scene.add( mesh4 );
```

Nous pouvons habiller les formes 3D à l'aide de matériaux et de textures pour qu'on puisse avoir un rendu qui proche du monde réel.

Les matériaux :

Le GPU calcule les couleurs pour chaque pixel en prenant en compte les matériaux des objets. Pourquoi ? Parce qu'avec des matériaux différents, les couleurs d'un objet peuvent changer en fonction des lumières diverses dans la scène. Il y a quelques matériaux qui sont réflecteurs de la lumière, tels que le bois brut, le journal et le béton. Nous les appelons les matériaux spéculaires.
Il peut y avoir plusieurs couleurs différentes suite aux interactions entre les lumières et les matériaux : Emissive, Ambient, Diffuse et Spéculaire.
Ambient: cette couleur est ce que l'objet reflète lorsqu'il est éclairé par la lumière ambiante plutôt que la lumière directe.
Diffuse: c'est la couleur naturelle de l'objet sous une lumière blanche.
Emissive: c'est la couleur que l'objet émet.
Spéculaire : c'est la couleur issue d'une réflexion spéculaire. Souvent c'est un objet d'une surface brillante.
Un exemple de code pour créer les matériaux dans three.js :

```
var materialBasic = new THREE.MeshBasicMaterial({ color: 0xaaaaaa });
var materialLambert = new THREE.MeshLambertMaterial({color:0xaaaaaa,shading: THREE.Smooth
Shading });
```

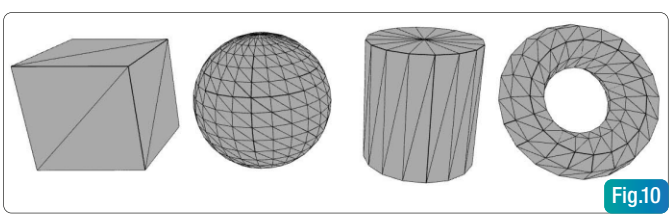
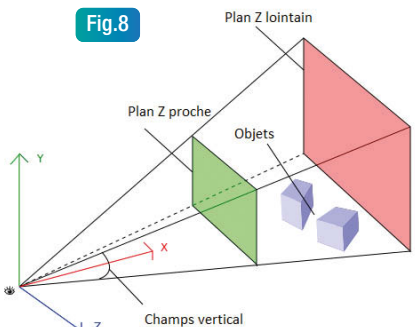


Fig.10

Fig.8



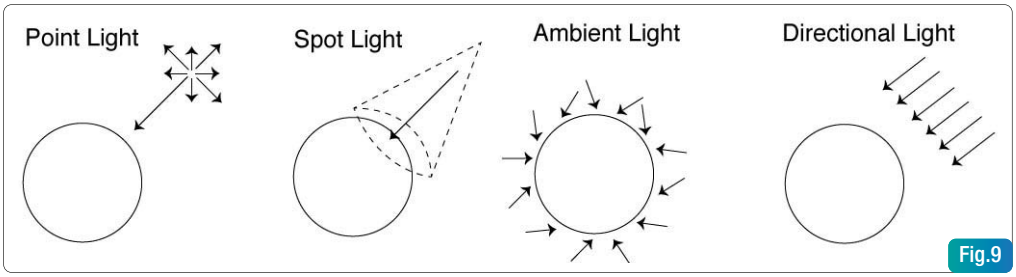


Fig.9

```
var materialPhong = new THREE.MeshPhongMaterial({color:0xaaaaaa,shininess:40, shading: THREE.
SmoothShading });
var materialNormal = new THREE.MeshNormalMaterial({shading:THREE.SmoothShading });
```

Shader :

Un ***shader*** ou **nuanceur** sert à décrire l'absorption et la diffusion de la lumière, la texture à utiliser, les réflexions et réfractions, l'ombrage, etc.
Les *shaders* sont flexibles et efficaces : des surfaces apparemment compliquées peuvent être rendues à partir de géométrie simple. Par exemple, un *shader* peut être utilisé pour générer les théières différentes à partir d'un simple maillage comme dans l'image en dessous. Fig.13.
Il existe plusieurs techniques de simulation d'éclairage dans un rendu 3D :

- Ombrage à plat : il consiste simplement à affecter à chacune des faces de l'objet un éclairage proportionnel à l'angle entre la normale à la face, estimée plane, et la direction de la source lumineuse qui éclaire la scène comme vous pouvez le voir sur la partie gauche dans l'image ; les facettes des objets 3D sont toujours très visibles.
- Ombrage de Gouraud : Il consiste à interpoler linéairement la luminosité entre les trois sommets d'un polygone. Appliqué à chaque sommet, cet ombrage plus réaliste va adoucir les angles comment vous pouvez constater dans la partie droite dans l'image. Fig.14.
- Ombrage de Phong : Le principal problème de l'ombrage de Gouraud est qu'il ne calcule que les sommets des polygones : une source lumineuse spéculaire placée au centre d'un triangle n'apparaîtra pas. À la différence de l'ombrage de Gouraud, l'ombrage de Phong calcule ce fait pour tous les points d'une surface.

Vous pouvez remarquer les différents rendus de ces techniques dans l'image des théières.

La texture :

Nous pouvons aussi appliquer une texture aux objets 3D. Une **texture** est une image en deux dimensions (2D) que l'on va appliquer sur une surface (2D) ou un volume en trois dimensions (3D) de manière à habiller cette surface ou ce volume. Lorsque l'on applique une texture sur un modèle 3D, on a besoin d'une méthode pour indiquer à l'ordinateur quelle partie de l'image doit être utilisée pour chaque triangle. Cela se fait grâce aux coordonnées UV. Voici un exemple (avec Three.js) que je trouve intuitif pour comprendre le “UV Mapping” :

On charge l'image dans le matériel qu'on va appliquer à l'objet

```
var material = new THREE.MeshPhongMaterial( { map: THREE.ImageUtils.loadTexture('images/texture.jpg') } );
```

On définit les coordonnées UV sur l'image. On part du point en bas à gauche (0,0) et on termine au point en haut à droite. (1,1) :

```
var bricks = [new THREE.Vector2(0, .666), new THREE.Vector2(.5, .666), new THREE.Vector2(.5, 1),
new THREE.Vector2(0, 1)];
var clouds = [new THREE.Vector2(.5, .666), new THREE.Vector2(1, .666), new THREE.Vector2(1, 1),
new THREE.Vector2(.5, 1)];
var crate = [new THREE.Vector2(0, .333), new THREE.Vector2(.5, .333), new THREE.Vector2(.5, .666),
new THREE.Vector2(0, .666)];
var stone = [new THREE.Vector2(.5, .333), new THREE.Vector2(1, .333), new THREE.Vector2(1, .666),
new THREE.Vector2(.5, .666)];
var water = [new THREE.Vector2(0, 0), new THREE.Vector2(.5, 0), new THREE.Vector2(.5, .333), new
THREE.Vector2(0, .333)];
var wood = [new THREE.Vector2(.5, 0), new THREE.Vector2(1, 0), new THREE.Vector2(1, .333), new
THREE.Vector2(.5, .333)];
```

On crée une géométrie d'un cube Fig.15.

```
var geometry = new THREE.CubeGeometry( 80, 80, 80);
```

On applique les coordonnées à notre géométrie pour appliquer la texture

```
geometry.faceVertexUvs[0] = [];
geometry.faceVertexUvs[0][0] = [ bricks[0], bricks[1], bricks[3] ];
geometry.faceVertexUvs[0][1] = [ bricks[1], bricks[2], bricks[3] ];
geometry.faceVertexUvs[0][2] = [ clouds[0], clouds[1], clouds[3] ];
geometry.faceVertexUvs[0][3] = [ clouds[1], clouds[2], clouds[3] ];
geometry.faceVertexUvs[0][4] = [ crate[0], crate[1], crate[3] ];
geometry.faceVertexUvs[0][5] = [ crate[1], crate[2], crate[3] ];
geometry.faceVertexUvs[0][6] = [ stone[0], stone[1], stone[3] ];
geometry.faceVertexUvs[0][7] = [ stone[1], stone[2], stone[3] ];
geometry.faceVertexUvs[0][8] = [ water[0], water[1], water[3] ];
geometry.faceVertexUvs[0][9] = [ water[1], water[2], water[3] ];
geometry.faceVertexUvs[0][10] = [ wood[0], wood[1], wood[3] ];
geometry.faceVertexUvs[0][11] = [ wood[1], wood[2], wood[3] ];
```

On passe la géométrie et le matériel à notre maillage :

```
mesh = new THREE.Mesh(geometry,  material);
```

Voici le résultat : Fig.16.
Il existe plusieurs techniques pour donner des variations à la surface d'un objet :
Normal mapping est une technique utilisée pour feindre le relief d'une texture. Elle est utilisée pour simuler graphiquement des détails géométriques, sans ajouter de polygones à la géométrie réelle.
Specular Mapping est une méthode utilisée pour définir les zones brillantes ou réfléchissantes sur une surface.
Alpha mapping est une technique qui applique une image sur un objet 3D, et désigne certaines zones de l'objet à être transparente. Fig.17 et 18.

Plus concrètement avec Three.js :

```
var textureLoader = new THREE.TextureLoader();
var material = new THREE.MeshPhongMaterial( {
```

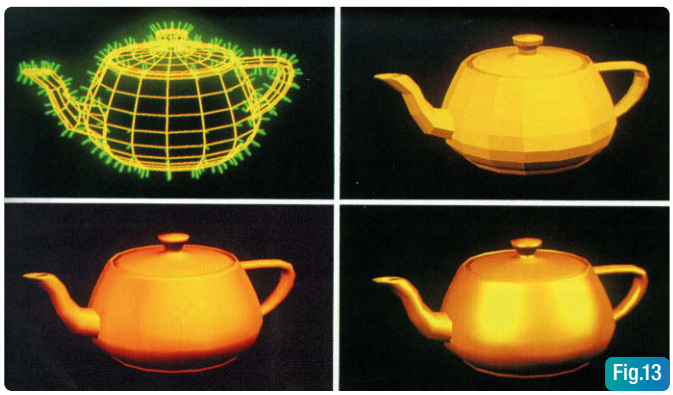


Fig.13

[http://www.viz.tamu.edu/faculty/parke/ends375f03/notes/sec7_3.html]
Haut-gauche: pas de l'ombrage; Haut-droite:l'ombrage à plat ; Bas-Gauche : Ombrage de Gouraud; Bas-droite : Ombrage de Phong

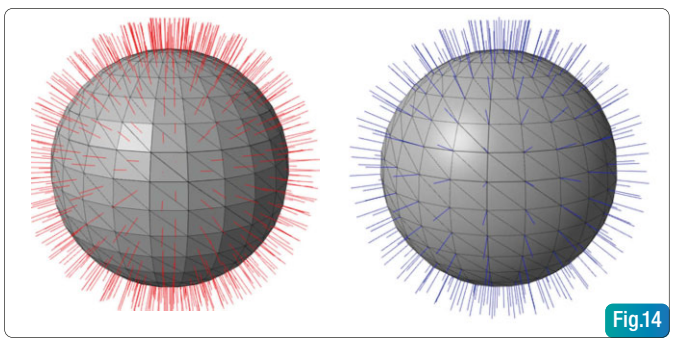


Fig.14

```
    color: 0xdddddd,
    specular: 0x222222,
    shininess: 35,
    map: textureLoader.load( "image1.jpg" ),
    specularMap: textureLoader.load( "image2.jpg" ),
    normalMap:textureLoader.load("image3.jpg" )
} );
```

Animation :

Avoir un beau rendu 3D est très bien mais nous aimons aller plus loin et interagir avec notre scène 3D, voir la scène s'animer.

Prenons un exemple simple :

Dans ma scène 3D qui est remplie de cubes, un cube devient lumineux lorsque l'on passe la souris dessus. Fig.19.

Une façon de le faire est de projeter un rayon 3d depuis la souris, à travers la caméra, dans la scène, et ensuite de vérifier si ce rayon intersecte avec des objets. Fig.20.

Concrètement en code avec Three.js :

```
var geometry = new THREE.BoxBufferGeometry( 20, 20, 20 );
```

```
for ( var i = 0; i < 2000; i ++ ) {
//Créer les cubes dans la scène
var object = new THREE.Mesh(geometry,new THREE.MeshLambertMaterial({color:  Math.
random() * 0xffffff } ) );
object.position.x = Math.random() * 800 - 400;
object.position.y = Math.random() * 800 - 400;
  object.position.z = Math.random() * 800 - 400;
  object.rotation.x = Math.random() * 2 * Math.PI;
        object.rotation.y = Math.random() * 2 * Math.PI;
          object.rotation.z = Math.random() * 2 * Math.PI;
            object.scale.x = Math.random() + 0.5;
object.scale.y = Math.random() + 0.5;
object.scale.z = Math.random() + 0.5;
scene.add( object );
}
//Définir un rayon 3D
```

```
var raycaster=new THREE.Raycaster();
//Définir la souris
var mouse = new THREE.Vector2(), INTERSECTED;
//Initialiser le moteur de rendu
renderer = new THREE.WebGLRenderer();
//Ajouter l'événement onDocumentMouseMove dans la DOM. La fonction est définie plus loin
document.addEventListener( 'mousemove', onDocumentMouseMove, false );
//Trouver les intersections avec les cubes
raycaster.setFromCamera( mouse, camera );
var intersects = raycaster.intersectObjects( scene.children );
if (intersects.length > 0 ) {
  if ( INTERSECTED!= intersects[ 0 ].object ) {
  ....
  //Rendre les cubes croisés brillants
  INTERSECTED.material.emissive.setHex( 0xff0000 );
 }
}
renderer.render( scene, camera );

//On récupère la position de la souris lorsqu'on clique sur l'ecran
function onDocumentMouseMove( event ) {
  event.preventDefault();
  mouse.x = ( event.clientX / window.innerWidth ) * 2 - 1;
  mouse.y = - ( event.clientY / window.innerHeight ) * 2 + 1;
}
```

Il existe beaucoup de techniques d'animation dans l'infographie 3D par exemple le système des particules, la capture des mouvements, la détec-

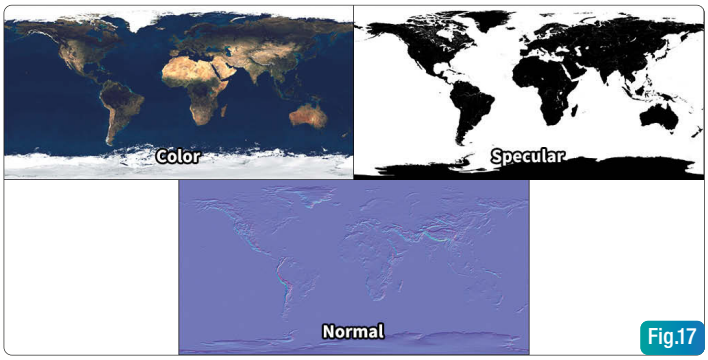


Fig.17

[http://threejs.org/]


Fig.18

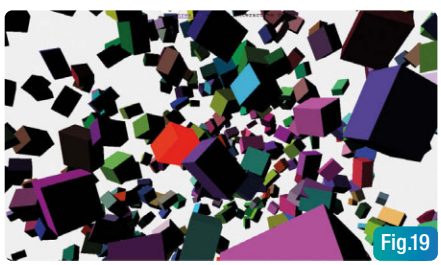


Fig.19

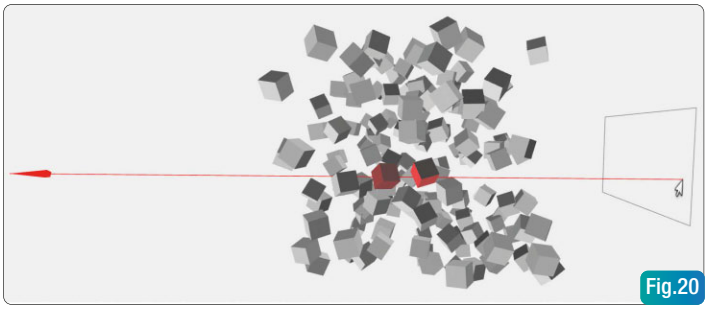


Fig.20

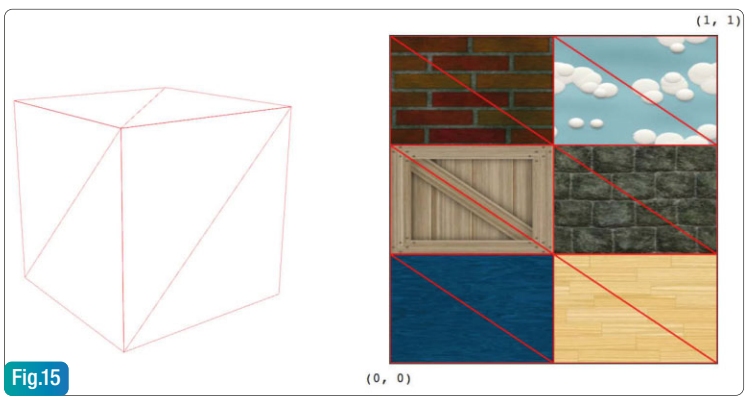


Fig.15

Fig.16

tion de collision, etc. Pour implémenter ces animations, on pourrait se baser sur des frameworks existants pour se faciliter la tâche.

Contrôles de la caméra :

Nous avons la possibilité de contrôler la position de la caméra avec la souris ou le clavier, Dans three.js cela peut être réalisé à l'aide d'**orbitcontrols.js** qu'on peut inclure dans notre fichier html :

```
<script src="three/controls/OrbitControls.js"></script>
camera = new THREE.PerspectiveCamera( 60, window.innerWidth / window.innerHeight, 1, 1000 );
camera.position.z = 500;
controls = new THREE.OrbitControls( camera );
controls.damping = 0.2;
controls.addEventListener( 'change', render );
```

Pipeline des graphiques 3D

Jusqu'à présent, on a vu les éléments essentiels pour faire une scène 3D. Mais comment les moteurs 3D réceptionnent toutes ces informations et produisent une scène 3D, autrement dit une image ?

Le pipeline graphique est un concept fondamental de l'infographie 3D. Il accepte la description des objets 3D avec les sommets (tels que le triangle, le point, la ligne et le quadrangle), et produit la couleur et la position pour les pixels sur l'écran. Voici une image qui présente une version simple du pipeline graphique 3D : Fig.21.

- Traitement des sommets : il s'agit de positionner la caméra et transformer les objets 3D par exemple dans l'espace vu de la caméra ;
- Rastérisation : convertir chaque primitif (sommets reliés en triangle par exemple) en un ensemble de fragments. Un fragment peut être traité comme un pixel dans des espaces 3D, qui est aligné dans le primitif comme tous les autres, avec des attributs tels que la position, la couleur, la texture normale etc. La rastérisation est le procédé le plus utilisé pour les jeux vidéo. Le peu de temps nécessaire pour calculer une image selon ce procédé permet d'obtenir en continu des images à un rythme suffisamment élevé pour faire du temps réel.
- Traitement des fragments : le traitement des fragments se concentre sur la texture et l'éclairage, ce qui a le plus grand impact sur la qualité de l'image finale.
- 3D Combine : c'est l'étape pour déterminer quel pixel se trouve au-dessus de quel autre et pour savoir quelles parties de quel(s) élément(s) sont visibles en utilisant la technique ***Z-buffer***. Cette étape peut aussi déterminer la transparence du fragment en utilisant l'***alpha-value***.

Dans le GPU moderne, l'étape de traitement des sommets et l'étape de traitement du fragment sont programmables. Vous pouvez écrire des programmes, connus sous les noms de **vertex shader** et de **fragment shader** pour effectuer votre transformation personnalisée pour les sommets et les fragments. Les programmes de shaders sont écrits en GLSL (OpenGL Shading Language), HLSL (High-Level Shading Language pour Microsoft Direct3D), ou Cg (C Graphics par NVIDIA).

Par exemple avec Three.js, il est possible d'ajouter les shaders personnalisés si vous voulez implémenter les effets personnalisés ou combiner plusieurs objets dans une seule géométrie pour améliorer la performance :

```
var material = new THREE.ShaderMaterial({
    uniforms: {
        time: { type: "f", value: 1.0 },
        resolution: { type: "v2", value: new THREE.Vector2() }
    },
    attributes: {
        vertexOpacity: { type: 'f', value: [] }
    },
    vertexShader: document.getElementById( 'vertexShader' ).textContent,
    fragmentShader: document.getElementById( 'fragment_shader' ).textContent
});
```

- **Uniforms** sont des variables qui ont les valeurs partagées par tous les sommets comme la lumière ou l'ombre mapping. Uniforms peut être accédé par **vertex shader** et **fragment shader**.
- **Attributes** sont des variables associées à chaque sommet comme la position, la face normale, la couleur. Attributes est accessible uniquement au **vertex shader**.

Les contenus des shaders peuvent être :

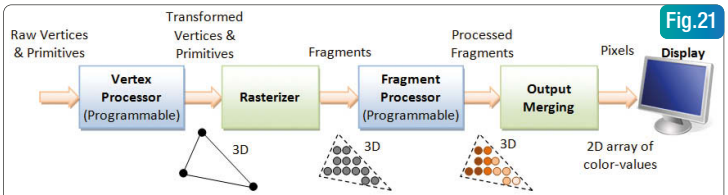
```
<script id="fragment_shader" type="x-shader/x-fragment">
    uniform float time;
    uniform vec2 resolution;
    varying vec2 vUv;
    void main( void ) {
        vec2 position = -1.0 + 2.0 * vUv;
        float red = abs( sin( position.x * position.y + time / 5.0 ) );
        float green = abs( sin( position.x * position.y + time / 4.0 ) );
        float blue = abs( sin( position.x * position.y + time / 3.0 ) );
        gl_FragColor = vec4( red, green, blue, 1.0 );
    }
</script>
<script id="vertexShader" type="x-shader/x-vertex">
    varying vec2 vUv;
    void main()
    {
        vUv = uv;
        vec4 mvPosition = modelViewMatrix * vec4( position, 1.0 );
        gl_Position = projectionMatrix * mvPosition;
    }
</script>
```

- Le **vertex shader** s'exécute en premier ; il réceptionne les **attributes**, calcule/manipule la position de chaque sommet, et passe les données supplémentaires (**varying**s) au **fragment shader ;**
- Le **fragment shader** s'exécute en second ; il applique la couleur de chaque pixel pour faire le rendu sur l'écran.

Il est difficile de montrer les effets incroyables de shader personnalisé en image statique dans l'article. Vous pouvez regarder l'exemple fourni par Three.js à l'adresse : http://threejs.org/examples/#webgl_shader2

Il existe plusieurs procédés pour faire le rendu 3D. Le lancer de rayon est aussi l'une des techniques les plus utilisées. Il consiste à calculer les éclairages de la caméra vers les objets, puis vers les lumières, alors que dans la réalité, la lumière va de la scène vers l'œil. Cette technique permet d'obtenir une image d'une qualité supérieure à celle de la rastérisation mais il

Fig.21



3D Graphics Rendering Pipeline: Output of one stage is fed as input of the next stage. A vertex has attributes such as (x, y, z) position, color (RGB or RGBA), vertex-normal (n_x, n_y, n_z), and texture. A primitive is made up of one or more vertices. The rasterizer raster-scans each primitive to produce a set of grid-aligned fragments, by interpolating the vertices.

[https://www.ntu.edu.sg/home/ehchua/programming/opengl/CG_BasicsTheory.html]

nécessite beaucoup de temps et de puissance de calcul. Son usage se voit plus dans les réalisations des images fixes et les films d'animation.
Une caractéristique importante du pipeline graphique est chaque étape peut s'exécuter en même temps indépendamment. Cela permet un travail sur le rendu d'une image en parallèle pour accélérer le traitement.

OpenGL

Pour le développement d'un moteur 3D, on pourrait se baser sur OpenGL. **OpenGL** (***Open Graphics Library***) est un ensemble normalisé de fonctions de calcul d'images 2D ou 3D lancé par Silicon Graphics en 1992. Cette interface de programmation est disponible sur de nombreuses plate-formes où elle est utilisée pour des applications qui vont du jeu vidéo jusqu'à la CAO en passant par la modélisation.

Three.js

Avant de parler de Three.js, il faudrait que l'on parle un peu de **WebGL**.
WebGL permet d'afficher, de créer et de gérer dynamiquement des éléments graphiques complexes en 3D dans la fenêtre du navigateur Web d'un client. Il est actuellement implémenté dans la plupart des grands navigateurs modernes. Il permet d'exploiter les pilotes de la carte graphique et de tirer ainsi profit de l'accélération matérielle, un atout majeur pour des rendus performants de formes et/ou de textures complexes.
Mais il n'est pas évident d'utiliser WebGL directement à cause des calculs matriciels complexes. Des librairies Javascript existent pour nous faciliter la tâche, Three.js en fait partie.
Le code source de Three.js est hébergé sur le Github de son créateur, mrDboob : https://github.com/mrdoob/three.js/. La bibliothèque est ainsi accessible à tout le monde.
Voici un exemple de code qui permet de créer la Terre avec Three.js :

```
<!DOCTYPE html>
<html>
<body>
//l'élément du DOM dans lequel nous allons mettre notre scène en 3D
<div id="container"></div>

//Il faut inclure le librarie Three.js
<script src="https://cdnjs.cloudflare.com/ajax/libs/three.js/r75/three.min.js">
</script>
<script src="three/Detector.js"></script>
<script>
var container;
  var camera, scene, renderer;
  var group;

  init();
  animate();

  function init() {
     //Pour savoir si notre navigateur supporte le WebGL
     if ( ! Detector.webgl ) Detector.addGetWebGLMessage();

container = document.getElementById( 'container' );
     //Définition de perspective de la caméra avec les paramètres : champs vertical,  le ratio d'aspect, les positions du plan Z proche et loin
     camera = new THREE.PerspectiveCamera( 60, window.innerWidth / window.innerHeight, 1, 2000 );
     //la position de la caméra (0,0,500)
     camera.position.z = 500;
     //Définition de la scène 3D
     scene = new THREE.Scene();

     //Création l'objet 3D -- Charger la texture
     var loader = new THREE.TextureLoader();
     loader.load( 'PathToEarthImage', function(texture)
{
     //Création de la géométrie
     var geometry = new THREE.SphereGeometry( 200, 20, 20 );
     //Création de la matière
     var material = new THREE.MeshLambertMaterial( { map: texture, overdraw: 0.5 } );
     //Création du maillage
     var mesh = new THREE.Mesh( geometry, material );
     //Ajouter l'objet dans la scène
     scene.add( mesh );
     });

     //Initialiser le moteur de rendu
     renderer = new THREE.WebGLRenderer();
     renderer.setClearColor( 0x000000 );
     renderer.setSize( window.innerWidth, window.innerHeight );
     //Ajouter le rendu dans la page Html
     container.appendChild( renderer.domElement );
  }

//la fonction animate() qui sera appelée récursivement pour mettre à jour les attributs de l'objet à animer.
  function animate() {
     requestAnimationFrame( animate );
     render();
  }

  function render() {
     //Un peu d'animation : Rotation de la terre
     group.rotation.y -= 0.005;
     //Lancer le processus du rendu
     renderer.render( scene, camera );
  }
  </script>
</body>
</html>
```

Si tout se passe bien, on devrait avoir une terre 3D qui tourne sur elle-même dans notre scène !

"This is not the end"

La création des graphiques 3D est un sujet très vaste et varié. On n'a pas pu tout couvrir dans cet article. J'espère que vous avez passé un moment agréable en le lisant et que cet article vous a donné envie de plonger dans cet univers incroyable !
Si ce sujet vous intéresse, je vous invite à regarder la présentation que j'ai donnée a Devoxx France 2015 qui explique comment créer des scènes 3D dans un navigateur Web avec Three.js : https://www.youtube.com/watch?v=763XrkxwNuw ainsi qu'un site que j'ai créé afin de montrer les orbites de certains satellites en 3D et en 2D qui est réalisé avec Three.js (Work In Progress) : http://followsats.herokuapp.com/

Décoder **le passé**

Kiosque | Abonnement | PDF

www.pharaon-magazine.fr

Développez des applications mobiles cross-plateformes avec NeoMAD !

1ère partie

Je souhaiterais vous faire découvrir un produit de cross-platform français. C'est un outil qui a vu le jour sur la côte Basque il y a quelques années. Je l'ai testé pour vous, il m'a séduite et j'ai envie de vous le faire partager.

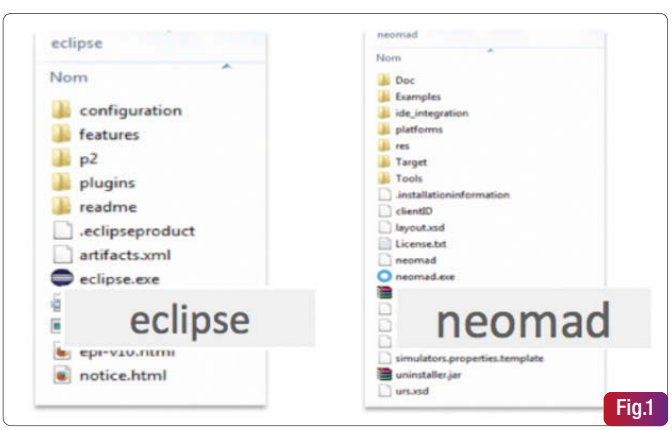
Alice Barralon
Scrum Master et consultant indépendant
a.barralon@oriions.com

NeoMAD, voyons qu'est-ce que ça signifie ? C'est la traduction littérale de l'anglais : 'nouveau' + 'fous'... Quel nom bizarre... Mais pourquoi choisir un nom qui ne se prend pas au sérieux pour un projet si ambitieux et si prometteur ? Est-ce un brin de 'folie' de la part de l'équipe du projet de vouloir jouer dans la cour des grands, de se positionner aux côté de PhoneGap, IONIC, Xamarin (Microsoft), et de tous les autres qui se placent sur le marché des applications mobiles multi-plateformes ?
NeoMAD est un outil qui permet de générer du code **natif** en JAVA. Ce n'est pas de l'hybride ou une surcouche graphique qui laisse à croire qu'on est dans du natif. L'utilisation et l'écriture du code ressemblent d'ailleurs fortement à du JAVA pour Android.
NeoMAD est téléchargeable gratuitement. Cela lève un premier frein car je peux donc le tester sans aucun frais et sans avoir le couperet du temps limité aux 30 jours d'essai que proposent certains éditeurs. Je vais donc sur le site neomades.com et je me lance, c'est parti ! Let's have fun with NeoMAD !

Installer son environnement de développement

Il faut tout d'abord télécharger la version starter qui est gratuite et accessible depuis la page d'accueil neomades (http://neomades.com/fr/). La version inclut les cibles Android et iOS ; elle package NeoMAD dans un Eclipse Bundle qui rend l'outil prêt à l'emploi. Pour ceux qui préfèrent travailler avec d'autres IDE, un outil est livré en parallèle qui permet de lancer NeoMAD en ligne de commande.
Lorsque NeoMAD est installé (jusqu'ici l'installation n'a pas duré plus de 5 minutes), il faut s'assurer que les 3 outils suivants sont présents sur le poste du développeur :

- Un JDK supérieur à la version 6 ;
- Un SDK Android (si votre cible est sur Android) ;
- XCode (qui peut être installé uniquement sur un Mac).



Contenu du ZIP NeoMAD

A qui s'adresse NeoMAD ?

NeoMAD se destine plus aux développeurs back-end qui maîtrisent JAVA et la POO. Les développeurs front-end iront peut-être plus spontanément vers des technologies qui s'appuient sur du JS.
Pourquoi choisir NeoMAD ?
Parce que vos équipes connaissent la POO et particulièrement JAVA et vous avez besoin de développer une application mobile multiplateforme mais vous n'avez pas envie de dupliquer des applications natives (IOS, Android).

Pour installer l'outil Bundle, commencez par déziper le ZIP du bundle précédemment téléchargé. Mettez-le dans votre espace dédié au développement. Le zip contient deux dossiers :

- Eclipse (LUNA)
- Neomad (les docs, examples, etc.) Fig.1.

Voilà, vous êtes prêt à développer votre première application avec NeoMAD ! Facile n'est-ce pas ? On termine quelques ajustements dans l'environnement qui vont nous permettre de déployer l'application développée sur les plateformes ciblées.
Pour remplir les préférences : **NeoMAD Eclipse > Preferences > NeoMAD** ou alors on peut passer par le fichier de config **: neomad/neomad.properties**
NeoMAD.properties file content :

```
# ANDROID SDK path (use '/' as separator)
# Folder that contains "platforms" folder
# Windows e.g. C:/Android/android-sdk
# MacOS e.g. /Library/Android/android-sdk-macosx
ANDROIDPATH=xxxxxxxxxx

# JAVA JDK path (use '/' as separator)
# Folder that contains "bin" folder
# Windows e.g. C:/Program Files/Java/jdk1.8.0_31
# MacOS e.g. /Library/Java/JavaVirtualMachines/jdk1.7.0_07.jdk/Contents/Home
JDKPATH=xxxxxxxxxxxxxxxxxx
```

Pour plus de détails concernant l'installation de l'AVD pour Android et de la configuration pour iOS, je vous invite à vous rendre sur le site neomades.com

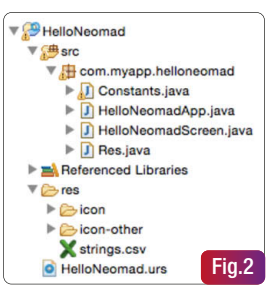


*Figure SEQ Figure * ARABIC 1 - Organisation d'un projet NeoMAD*

Construire sa première app NeoMAD

Je vous propose maintenant de démarrer la première application NeoMAD.
Pour cela nous allons créer un nouveau projet dans Eclipse : **File > New > NeoMAD Project** Fig.2.
Un projet NeoMAD est identique à n'importe quel projet JAVA, avec ses sources

structurées en packages. On retrouvera donc :

- src : les sources Java
- res : les ressources
- out : output
- urs : spécificité NeoMAD (description du projet)

Nous allons commencer par dessiner l'interface graphique (GUI). Elle n'est pas WYSIWYG et doit être écrite à la main dans un fichier XML. Le fichier de description sera stocké dans **res/layout**. Pour le créer : **File>New>NeoMAD XML Layout File**. Fig.3.

Extrait de code qui décrit la première interface graphique que nous allons construire :

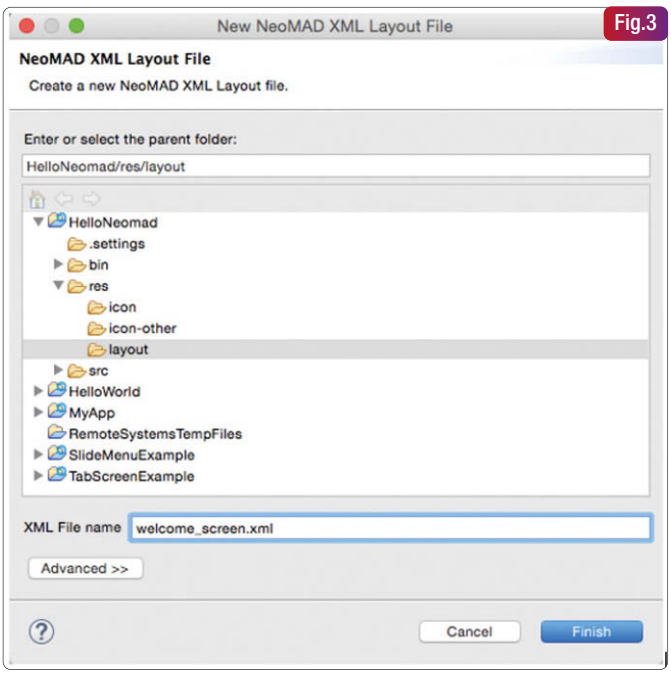
```
<?xml version="1.0"?>
<!-- A VerticalLayout represents a group of views arranged vertically. We want the Vertical
Layout to fill all the screen size and the content to be centered into the VerticalLayout. -->
<VerticalLayout xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://www.neomades.com/XSD/3.8.0/layout.xsd"
  stretchHorizontalMode="MATCH_PARENT"
  stretchVerticalMode="MATCH_PARENT"
  contentAlignment="HCENTER|VCENTER">

<!-- A TextLabel allows to display a simple text on the screen. We want the TextLabel size
 to match the text. -->
<TextLabel
      stretchHorizontalMode="MATCH_CONTENT"
      stretchVerticalMode="MATCH_CONTENT"
     text="@string/WELCOME_SCREEN" />
</VerticalLayout>
```

Ce nouveau fichier doit être déclaré dans le fichier des URS (HelloNeomad.urs)

```
<?xml version="1.0"?>
<urs xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation=
"http://www.neomades.com/XSD/3.8.0/urs.xsd">
 <parameters>
```

Fig.3



Ajout d'un fichier graphique dans le layout

```
<mainclassname>HelloNeomadApp</mainclassname>
  <applicationname>HelloNeomad</applicationname>
  <packagename>com.example.helloneomad</packagename>
     <icon path="${ICON_PATH}" />
     <vendor>vendor name</vendor>
     <version>1.0.0</version>
     <srcpath>src</srcpath>
     <outputpath>out</outputpath>
 </parameters>
 <strings path="res/strings.csv" />
 <resourcelot>
   <layout name="WELCOME_SCREEN"
      path="res/layout/welcome_screen.xml"/>
 </resourcelot>
</urs>
```

Le fichier des URS contient des informations générales concernant le projet. Les attributs des balises XML sont les suivants :

- `mainclassname` : le nom du point d'entrée de l'application ;
- `Applicationname` : le nom qui sera affiché dans le menu de l'appareil mobile ;
- `Packagename` : nom de l'application root package ;
- `Vendor` : nom du vendeur ;
- `Description` : description ;
- `Icon` : chemin d'accès à l'icône ;
- `Version` : la version ;
- `Srcpath` : chemin d'accès au répertoire ;
- `Outputpath` : chemin d'accès vers le répertoire ou seront générés les sources.

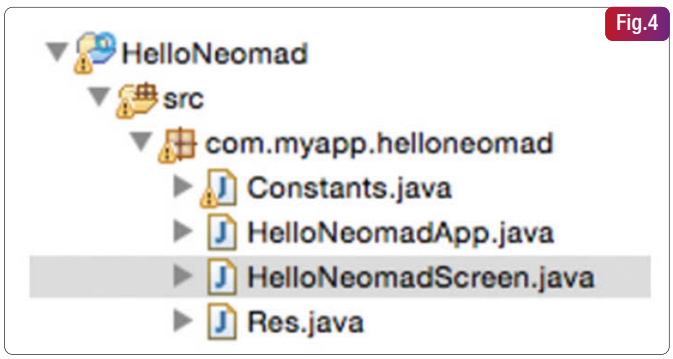
A titre d'information, tous les tags sont obligatoires à l'exception du tag 'description'.

Création d'une simple interface utilisateur

Vous pouvez maintenant travailler sur la partie graphique dans le fichier JAVA (HelloNeomadScreen.java). A titre de test, je vous propose d'ajouter un message de bienvenue comme suit : Fig.4.

```
/**
 * Application page screen
 */
public final class HelloNeomadScreen extends Screen {
  protected void onCreate() {
     // Fill the screen with the HELLOWORLD_SCREEN layout
     setContent(Res.layout.WELCOME_SCREEN);
  }
}
```

Fig.4



Pour gérer le multilingue, NeoMAD propose de stocker les textes des labels dans le fichier string.csv. Nous allons ainsi ajouter un nouveau message : Fig.5.

La structure du fichier CSV se construit comme suit :

- La première ligne décrit le langage
- Chaque ligne qui suit décrit un label : <Label ID> ;<Lang 1> ;<Lang 2>

```
ID;en;fr;
WELCOME_SCREEN;"Greeting";"Félicitations";
```

NeoMAD va créer automatiquement une constante dans Res.java qui correspondra à chaque label ajouté dans le fichier.

Aperçu du fichier source Res.java généré :

```
//Res.java Generated code
public final class Res {
  public static final class string {
     public static final int WELCOME_SCREEN = 0;
  }
}
```

Je vous propose d'ajouter maintenant un bouton dans l'écran. Pour cela, deux façons s'offrent à nous :

- Ajout dans le fichier de description xml ;
- Ajout dans la classe Java.

Si on ajoute le code dans le fichier XML (welcome-screen.xml)

```
<Button text="@string/WELCOME_GO" stretchHorizontalMode="MATCH_PARENT"
stretchVerticalMode="MATCH_CONTENT" id="@+id/button_welcome_go" margin="10" />
```

Ajouter le label dans le fichier texte comme vu précédemment (string.csv)

```
WELCOME_GO;"GO";"GO";
```

Ajouter l'écouteur du bouton dans le code JAVA (HelloNeomadScreen.java) :

- Attacher un évènement clickListener au bouton ;
- Implementer la classe ClickListener ;
- Ecrire le contenu de la method onClick.

```
public final class HelloNeomadScreen extends Screen implements ClickListener{
  protected void onCreate() {
     // Fill the screen with the HELLOWORLD_SCREEN layout
     setContent(Res.layout.WELCOME_SCREEN);
     // Attach ClickListener to buttons
     Button goButton = (Button) findView(Res.id.button_welcome_go);
     goButton.setClickListener(this);
  }
  public void onClick(View view) {
     switch (view.getId()) {
     case Res.id.button_welcome_go:
        break;
     default:
        break;
     }
  }
}
```

Si on déclare un nouvel écran :

```
public void onClick(View view) {
     switch (view.getId()) {
     case Res.id.button_welcome_go:controller.pushScreen(DisplayScreen.class);
        break;
     default:
        break;
     }
}
```

et qu'on le créé en suivant :

```
package com.myapp.helloneomad;

import com.neomades.app.Screen;

public final class DisplayScreen extends Screen {
  protected void onCreate() {
     setContent(Res.layout.DISPLAY_SCREEN);
  }
}
```

qu'on remplit sa partie graphique :

```
<?xml version="1.0"?>
<VerticalLayout xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://www.neomades.com/XSD/3.8.0/layout.xsd"
stretchHorizontalMode="MATCH_PARENT"
stretchVerticalMode="MATCH_PARENT"
contentAlignment="HCENTER|VCENTER">

<!-- A TextLabel allows to display a simple text on the screen.
We want the TextLabel size to match the text. -->
<TextLabel
stretchHorizontalMode="MATCH_CONTENT"
stretchVerticalMode="MATCH_CONTENT"
text="@string/DISPLAY_SCREEN" />

</VerticalLayout>
```

Pour tester l'application

Le lancement de l'application fonctionne comme le lancement de tout projet Java dans Eclipse : un clic droit sur le projet **Run As > Run Configurations ...** Fig.6.

Vous pouvez choisir de lancer l'application :

- Sur un appareil mobile connecté en USB ;
- Sur un Android Virtual Device (AVD) ;

Fig.5

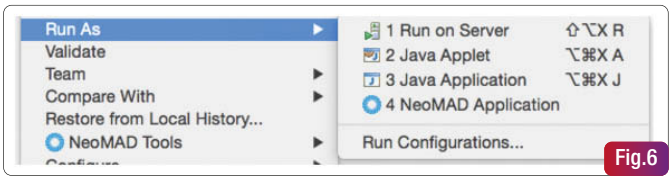


Fig.6

- Sur un émulateur iOS. Fig.7.

Exemples NeoMAD

Si vous cherchez à pratiquer, vous pouvez jouer avec un ensemble d'exemples présents dans le dossier : neomad/Examples.

- AnimationExample (une image animée) ;
- AsyncListExample (une list asynchrone) ;
- CameraExample (pour prendre des photos) ;
- ConditionalCoding ;
- ContactExample (pour récupérer des infos depuis le carnet de contact) ;
- HelloCloneWorld (pour cloner une appli existante) ;
- HelloWorld ;
- HttpUploadExample (pour uploader un fichier) ;
- IconsExample (une adaptation des icons de chaque plateforme) ;
- JSONExample (pour lire/écrire un flux json) ;
- NotificationExample ;
- PlatformSpecificCode ;
- RSSReaderExample ;
- SensorExample (pour piloter l'accéléromètre, le gyromètre) ;
- ServiceExample ;
- SlideMenuExample (un slide menu) ;
- TabScreenExample ;
- XmlPullParserExample.

Pour importer un exemple existant : **File > New > Other**
Choisir NeoMAD Example : Fig.8.

Voilà, j'espère vous avoir convaincu et donné envie de tester NeoMAD. Pour les experts Android qui souhaitent développer sur cross-plateforme, la prise en main NeoMAD sera très rapide et efficace car vous constaterez qu'il y a beaucoup de similitudes (XML, classes Java, etc.). A vous de jouer maintenant !

Suite le mois prochain.

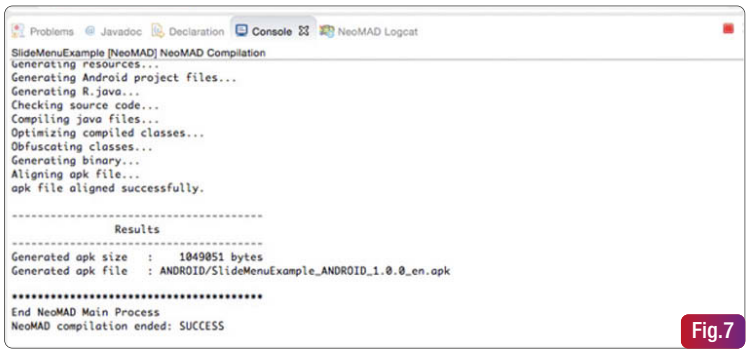


Fig.7

Logs au lancement de l'appli NeoMAD

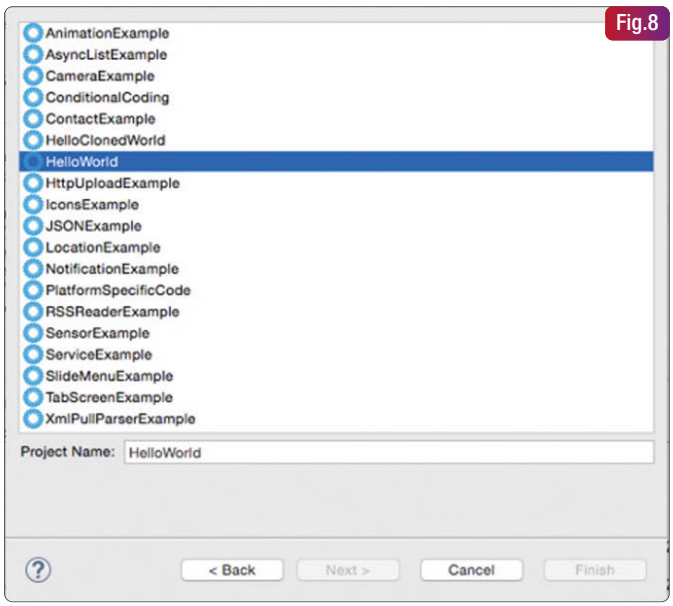


Fig.8

Ouverture d'un exemple NeoMAD dans Eclipse

Pot (pas si) pourri d'astuces pour Visual Studio

Entre les nouveautés de C# 6.0, Roslyn et Visual Studio, voici quelques pépites à côté desquelles vous êtes peut-être passé. Vous trouverez également quelques fonctionnalités souvent méconnues qui existent pourtant depuis bien longtemps !

Stéphanie Hertrich
Evangéliste technique @Microsoft France
Duchess France Leader
Twitter : @stepheUp
Blog : blogs.msdn.com/stephe

Les immanquables dans l'éditeur

La barre de recherche de Visual Studio

Parmi la myriade de commandes et options dispersées à travers les menus, fenêtres et sous-fenêtres, il est bien compliqué d'accéder rapidement à une fonctionnalité voulue. La barre de recherche intégrée dans le bord même de la fenêtre de Visual Studio (et donc visible en permanence) vous permet d'y accéder directement en tapant ses premières lettres :
Ex : pour trouver l'option permettant de changer le niveau de verbe (verbosity) du compilateur qui se trouve dans un 4ème niveau de sous-menu (Menu/Options/Projects&Solution/Build&Run) Fig.1.

La sélection en rectangle

En maintenant la touche Alt vous pouvez sélectionner du texte au format rectangulaire. Pratique lorsque vous copiez-collez du code provenant d'un site web où il est parfois préfixé par des numéros de ligne. Fig.2.

Mise en forme du code

Le raccourci Ctrl-K-D vous permet de remettre votre code en forme (indentations, ...) et cela y compris pour le code XAML. Corrigez les erreurs de compilation auparavant pour un résultat optimal. Fig.3.
Ctrl-K-D Fig.4.

Conservez les positions de vos fenêtres

Vous pouvez sauvegarder votre contexte de fenêtres et en créer par exemple autant que vous avez d'environnements de travail différents (pro, perso, double-écran, résolution, type de projet...). Ces contextes sont associés à votre profil utilisateur et sont donc répliqués via roaming sur vos différentes machines. Fig.5.

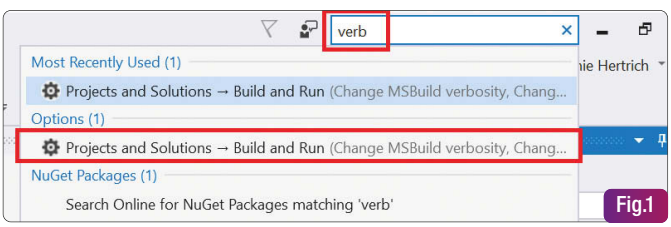


Fig.1

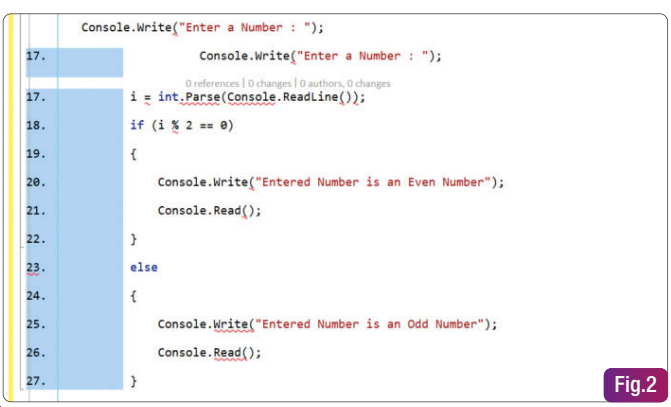


Fig.2

Paste as JSON

Saviez-vous que vous pouviez générer des classes à partir d'un bloc JSON ou XML en un simple clic ?
Copiez le bloc de code JSON ou XML dans le presse-papier Fig.6.
Puis choisissez « Edit/Paste Special » dans Visual Studio. Fig.7.
Les classes correspondantes seront générées directement dans le fichier cible.

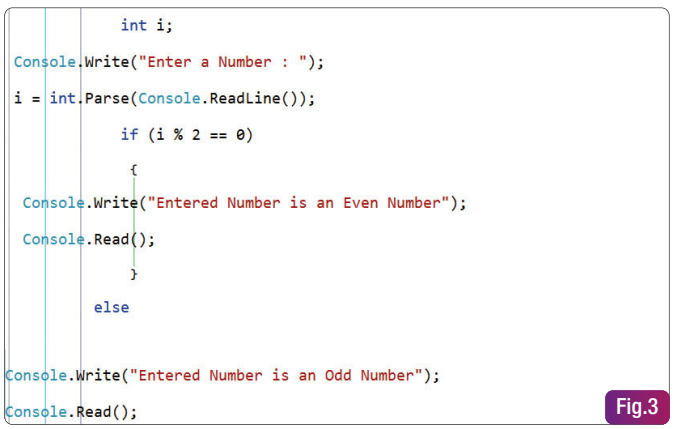


Fig.3

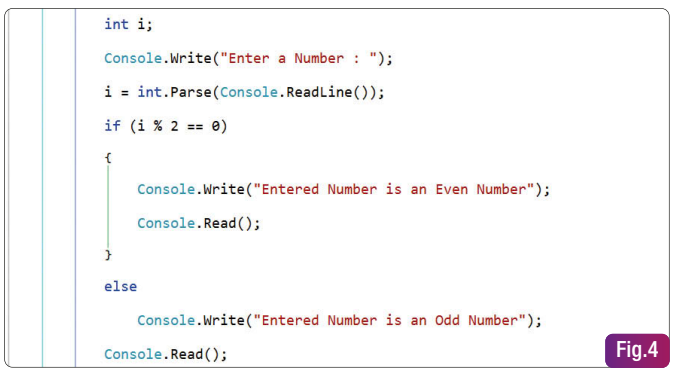


Fig.4

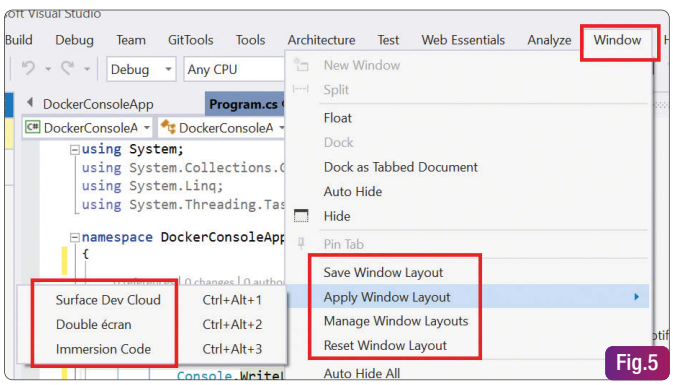


Fig.5

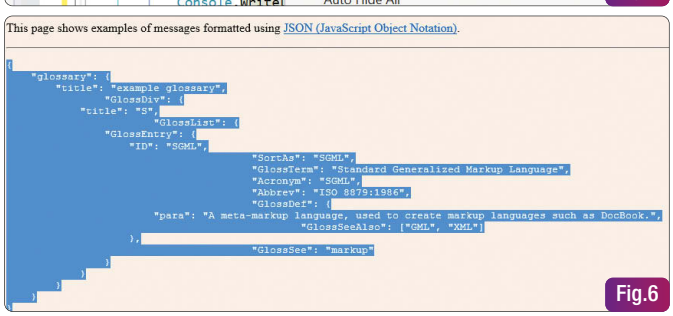


Fig.6

Générer un GUID

Certainement la fonctionnalité la plus ancienne de cette liste puisqu'elle existe depuis Visual Studio 2005 : vous pouvez créer un GUID très facilement depuis le menu Tools : Fig.8.

Nouveautés du langage C# 6.0

Expression bodied members

Votre méthode retourne une simple ligne de code ? Vous pouvez dorénavant utiliser une syntaxe simplifiée, à la mode des expressions lambda. Cerise sur le gâteau : ça marche aussi pour les getters !

```
public double Dist
{
  get
  {
  return Math.Sqrt(X * X + Y * Y);
  }
}
```

devient

```
public double Dist => Math.Sqrt(X * X + Y * Y);
```

L'opérateur « null-conditional »

Il s'agit de simplifier les situations qui impliquent de tester la nullité d'une variable/membre pour accéder à un attribut ou une méthode.
Dans le code suivant :

```
static void Main(string[] args)
{
  switch (args?.Length)
  {
    // ...
  }
}
```

La condition du switch correspond à la logique de code suivant (où args ne serait par contre évalué qu'une seule fois).

```
(args != null) ? (int?)args.Length : null
```

Notez que le type de retour de cet opérateur appliqué à un type T est un Nullable<T>.
Il est très pratique dans les cas où l'on va chaîner les appels nécessitant des éléments ou propriétés non nuls :

```
string[] names = person?.Name?.Split(' ');
```

Autre cas d'application intéressant : simplifier l'invocation de delegate permettant ainsi de supprimer le test à nul qui le précède habituellement. On obtient :

```
OnTemperatureChanged?.Invoke(this, value);
```

Mais l'opérateur fonctionne aussi avec les index en utilisant « ?[

```
if(json == null ||
  json["x"] == null ||
  json["x"].Type != JTokenType.Integer ||
  json["y"] == null ||
  json["y"].Type != JTokenType.Integer)
{
  throw new ArgumentException("Parameter is not shaped like a Point", "json");
}
...
```

devient

```
if (json?["x"]?.Type != JTokenType.Integer ||
   json?["y"]?.Type != JTokenType.Integer)
{
  throw new ArgumentException("Parameter is not shaped like a Point", "json");
}
```

Nameof

C'est une des nouveautés très populaires de C# 6.0 : on peut désormais faire référence au nom d'une variable, type ou membre grâce à l'opérateur « nameof » permettant ainsi d'avoir une vérification de son existence dans le scope lors de la phase de compilation. La dernière ligne du snipet servant d'exemple pour l'opérateur Null-conditional devient

```
throw new ArgumentException("The parameter is not shaped like a Point", nameof(json))
```

Entre autres, c'est très pratique pour l'implémentation de l'interface INotifyPropertyChanged.

Adieu String.Format !

C'est possible grâce aux chaînes de caractères interpolées qui permettent une simplification d'écriture bien pratique pour le formatage de chaînes. Elles permettent de référencer des expressions directement là où on les utilise dans la construction d'une chaîne de caractères. C'est applicable partout où vous utilisez des chaînes de caractères littérales.

```
msg = string.Format("Hello!  My name is {0} {1} and I am {2} years old.",
          person.FirstName, person.LastName, person.Age);
```

devient

```
msg = $"Hello!  My name is {person.FirstName} {person.LastName} and I am
  { person.Age} years old.";
```

Les extensions

Accessibles via le menu Tools/Extensions and Updates, elles permettent d'étendre les fonctionnalités de Visual Studio.

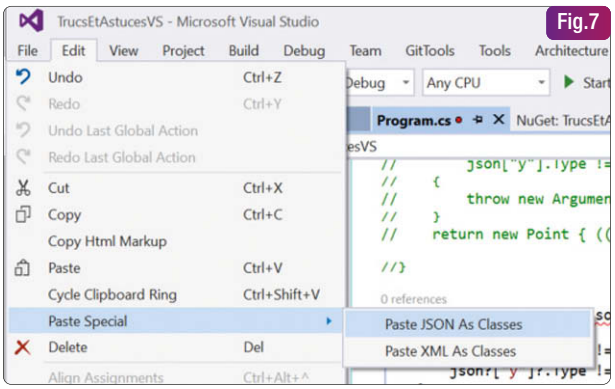


Fig.7

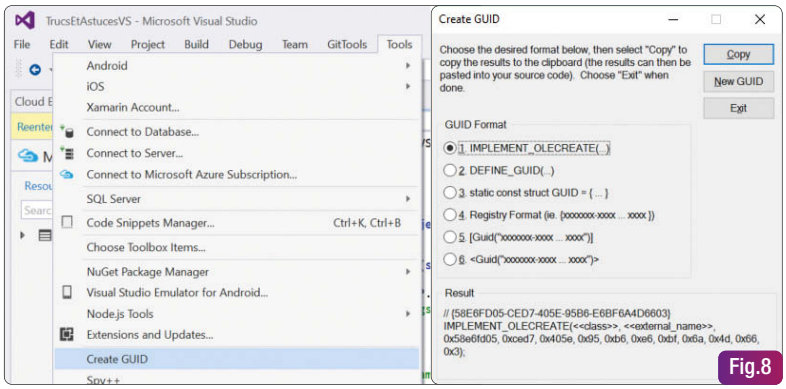


Fig.8

Developer Assistant

C'est une extension très intéressante de Visual Studio, notamment depuis qu'elle a bénéficié de l'intégration de **Github**. Elle vous donne accès à différents niveaux assistance pour améliorer votre productivité au quotidien. Fig.9.

Grâce à l' « **API Assistant** », vous aurez accès à des millions d'exemples de code, notamment pour l'utilisation des APIs à la volée pendant que vous codez. Fig.10.

Le « **Developer Assistant** » vous permet de formuler votre recherche en langage naturel : Fig.11.

Le « **Compiler Error Assistant** » fournit de l'aide contextuelle pour corriger les erreurs de compilation.

Duocode

Cette extension permet de **cross-compiler du code C# 6.0 en Javascript** avec le support de TypeScript. Elle vous permet de réutiliser vos compétences de développeur .Net et de partager du code entre des applications Windows et Javascript. Fig.12. L'extension est payante mais vous pouvez la tester pendant 30 jours gratuitement. Elle fonctionne avec Visual Studio 2012, 2013 et 2015. Rendez-vous sur http://duoco.de/

Alive

Alive est une extension qui vous permet de visualiser le résultat de votre code en temps réel, au fur et à mesure que vous le tapez et ceci directement dans l'éditeur. Bref, **votre code devient vivant** lors de la phase de développement. L'extension est payante mais vous pouvez la tester gratuitement pour 30 jours sur https://comealive.io. Fig.13.

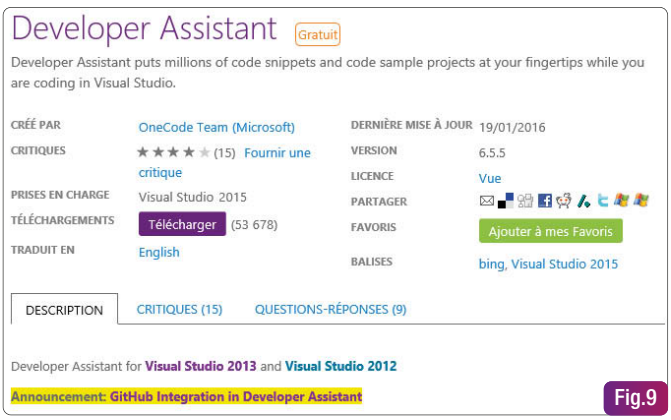


Fig.9

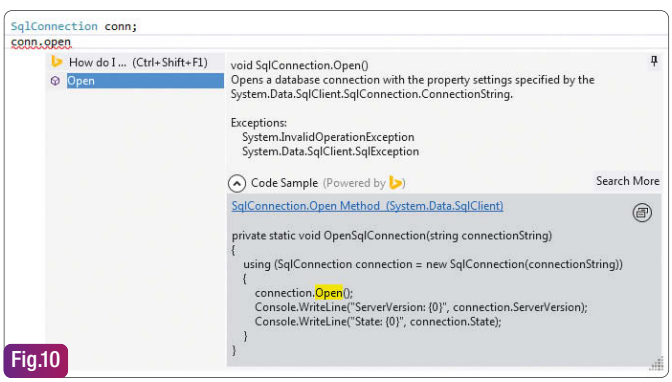


Fig.10

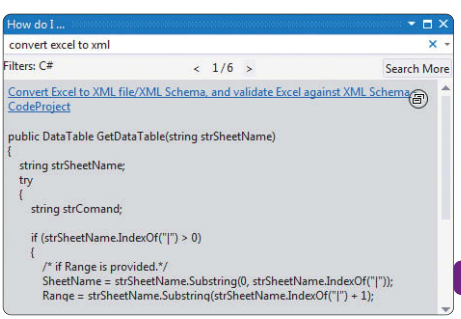


Fig.11

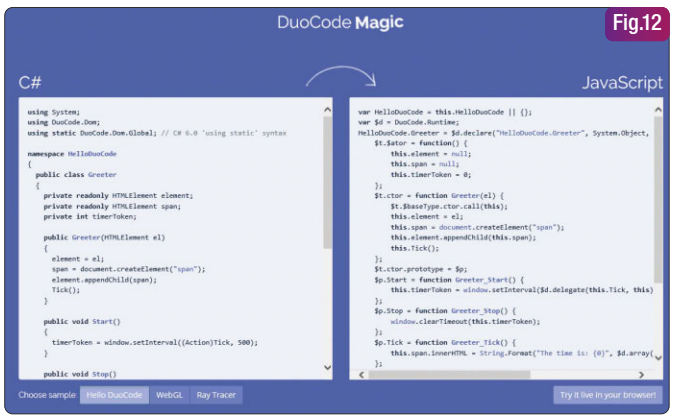


Fig.12

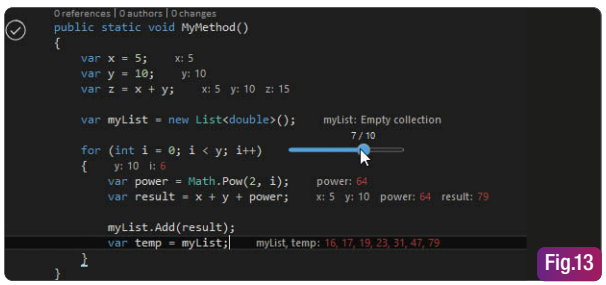


Fig.13

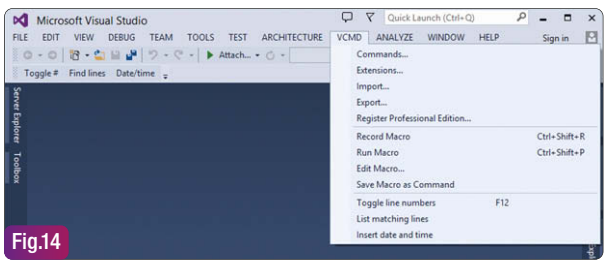


Fig.14

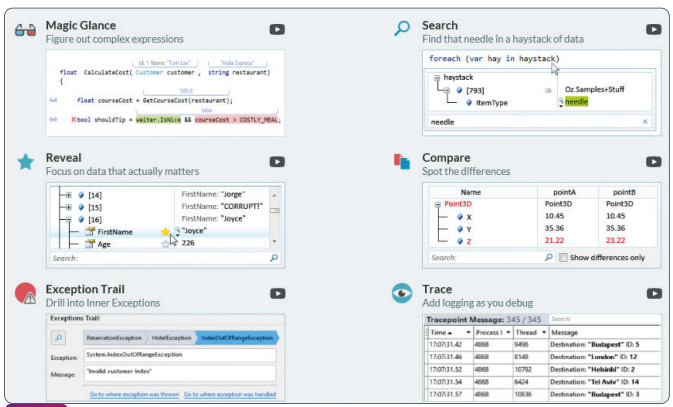


Fig.15

l'extension fonctionne aussi sur les précédentes versions de l'IDE et ce à partir de Visual Studio 2010 ! Fig.15.

Pour les applications UWP

L'extension UWP Tile Generator

Bien pratique pour les développeurs d'applications UWP, voici une extension permettant de générer des images aux différents formats recommandés pour les vignettes. Fig.16.

Le package nuget « PlatformSpecificAnalyser »

Créé avec Roslyn, il surveille votre utilisation des APIs spécifiques à une plateforme et vous alerte des problèmes potentiels tout en vous proposant une suggestion de correction. Fig.17.

Vous le trouverez ici : https://www.nuget.org/packages/PlatformSpecific.Analyzer

La marketplace de la suite Visual Studio

Saviez-vous qu'il existe aussi des extensions pour les autres produits de la suite Visual Studio comme **Visual Studio Team Services** et **Visual Studio Code** ? Une marketplace leur est dédiée : https://marketplace.visualstudio.com Fig.18.

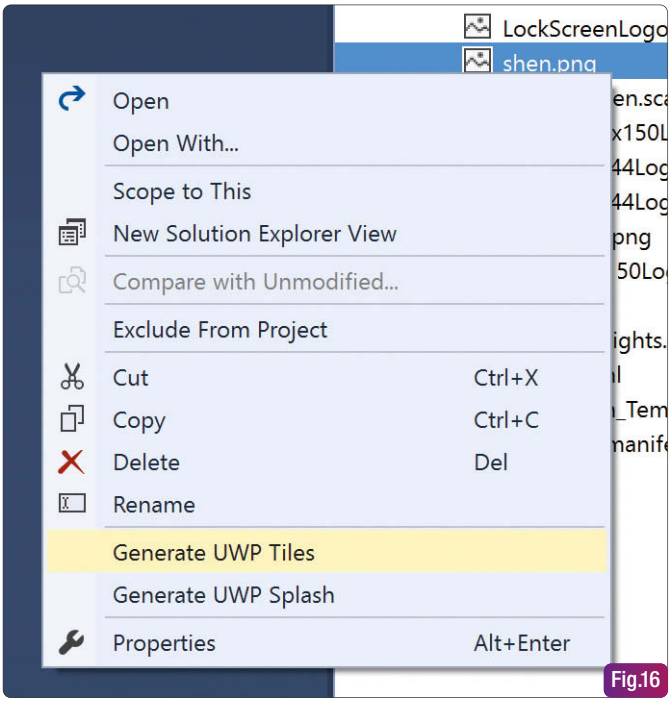


Fig.16

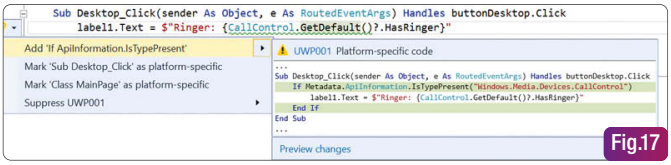


Fig.17

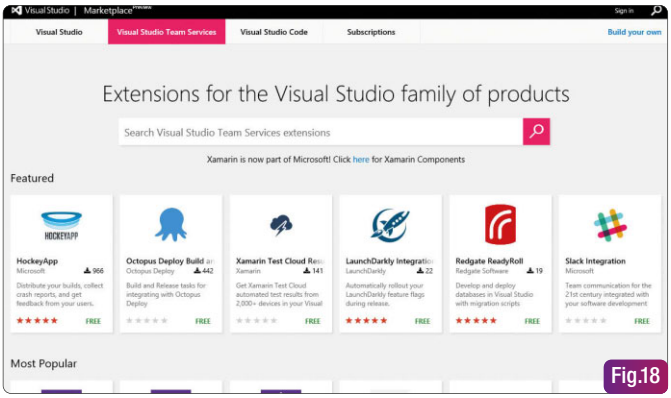


Fig.18

Les annonces de la //build

Visual Studio 2015 Update 2

Durant la conférence //build, **Visual Studio 2015 Update 2** a été annoncé avec de multiples améliorations et optimisations ainsi que de nouveaux outils d'analyse. Voici quelques morceaux choisis :

Execute in Interactive

Permet de charger la fenêtre interactive dans un contexte donné Fig.19.

Mises à jour automatiques des extensions

Il est possible de désactiver cette fonctionnalité, séparément par extension.

Xamarin

L'annonce qui a fait le plus de bruit concerne Xamarin récemment racheté par Microsoft. Pour rappel, la société édite une suite d'outils pour le développement mobile cross-plateforme en .Net.

Xamarin (y compris Xamarin.Forms) est intégré à toutes les éditions de Visual Studio, version Community incluse. Fig.20.

A titre de comparaison avec les offres actuelles :

- Avec Visual Studio Professionnel, vous aurez accès aux fonctionnalités correspondant à la version Business de Xamarin.
- Avec Visual Studio Entreprise, vous avez accès aux fonctionnalités correspondant à la version Enterprise de Xamarin.
- Xamarin Studio est désormais disponible sur Mac dans une édition Community.

La prochaine version : Visual Studio « 15 » (à ne pas confondre avec « 2015 ») est téléchargeable dès maintenant en preview : https://www.visualstudio.com/news/vs15-preview-vs

Quelques exemples de nouveautés sympathiques : en activant de nouvelles extensions du langage C#, vous pourrez utiliser du **pattern-matching** et déclarer des **fonctions locales**. Fig.21.

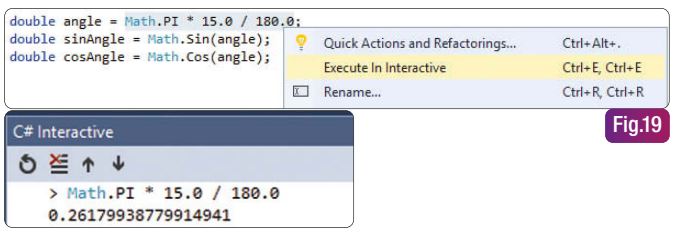


Fig.19

Fig.20

Fig.21

Comment Spark et Redshift ont changé notre quotidien

Après deux ans passés à construire des applications BI pour leboncoin à l'aide d'ETL et de bases de données relationnelles, le constat est criant : les limites frustrent le développeur, l'analyste, et, in fine, l'utilisateur. Les outils dits big data semblent être la solution. Qu'en est-il réellement? Comment migrer vers la myriade d'outils proposés?

Stéphanie Baltus
Responsable Data Engineering chez **leboncoin**
@steph_baltus

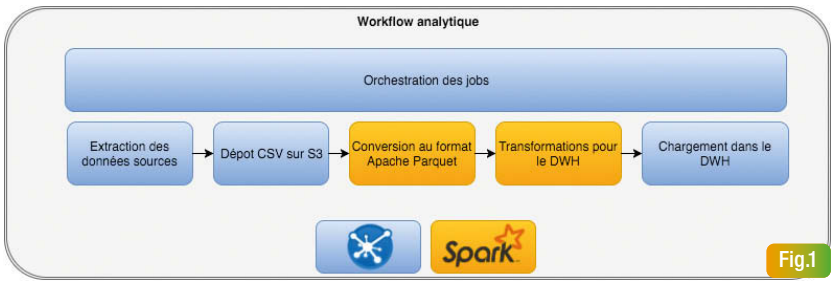


Fig.1

Durant ces deux années, la maintenance s'est montrée de plus en plus douloureuse, les évolutions très chronophages, et les tests automatiques impossibles à mettre en place avec un ETL classique et une montagne de données. Notre datawarehouse déjà restreint en était à plus de 6 To de données, avec des tables dépassant le To. Les tentatives d'optimisation, allant des flux ETL aux modèles de données, en passant par l'utilisation d'extensions et de type spéciaux (hll, range...) n'ont fait que repousser l'échéance de l'inévitable, le volume stocké étant devenu très difficile à requêter par les analystes, notamment en raison des délais de retour des requêtes.

Pour remplacer notre architecture composée de PostgreSQL, MonetDB et Pentaho Data Integration (PDI), nous avons choisi une migration en douceur avec PDI, Apache Spark et Amazon Redshift. Fig.1.

Cet article présente un retour d'expérience de notre transition vers les outils distribués.

Dans un premier temps, le mode d'extraction des données sources sera brièvement abordé, puis nous détaillerons notre usage et quelques écueils rencontrés avec Spark et Redshift. Quelques exemples de code (Java et SQL) seront fournis à titre d'exemple sur les points jugés importants.

Une extraction unique

Plusieurs de nos applications, décisionnelles ou opérationnelles, se basent sur les mêmes sources, mais ont un niveau de transformation différent.

Par ailleurs, il nous est souvent arrivé de devoir récupérer un historique depuis les bases de données sources, en raison d'anomalies dans les transformations ou de nouveaux besoins. Ces deux points génèrent des sollicitations inutiles des bases de données source.

Nous avons opté pour une extraction unique, table par table, avec des transformations basiques en utilisant PDI, les possibilités de multithreading offertes par l'outil nous permettant des extractions massives et rapides des données, tout en maitrisant l'impact sur l'infrastructure source.

Les fichiers sont stockés sur S3 en CSV partitionné, sur la clé primaire de la table.

Nous avons deux types d'extractions :

- Le "snapshot" qui récupère l'ensemble d'une base, idéal pour une récupération d'historique ;
- L'incrémental, qui ne récupère que les nouvelles données depuis la dernière extraction.

Transformations avec Spark

Pour gérer nos transformations analytiques et opérationnelles nous avons monté un cluster Spark en standalone, disposant de 8 noeuds et 8 cœurs, avec 16 Go de RAM et 320 Go d'espace disque. Cette configuration nous permet de gérer rapidement les chargements incrémentaux, tout en servant à d'autres applications opérationnelles.

Lorsque nous devons charger des "snapshots", nous passons sur une configuration où chaque nœud a 16 cœurs et 122 Go de RAM.

Même si Spark gère les transformations "in-memory", il peut arriver que certaines ne tiennent pas en mémoire, dans ce cas Spark écrit sur disque, d'où le besoin d'espace disque conséquent.

Du fait du passé de développeur Java majoritaire dans l'équipe, nous avons choisi d'utiliser l'API Spark en Java, et en version 1.8 pour bénéficier des lambdas.

Dans cette partie, nous aborderons les points clés qui ont fait la réussite de cette transition : le format Parquet, les UDF, les broadcasts variables, la communauté autour de ce framework et le point de douleur récurrent, les entrées/sorties avec S3.

Un datalake en Parquet

Le format de stockage est décisif lorsque l'on souhaite traiter des fichiers volumineux, le but étant de pouvoir distribuer efficacement les calculs, et minimiser l'utilisation inutile de données, un simple fichier plat ne semble pas adapté.

Notre choix s'est vite orienté vers Apache Parquet. Parquet est un format de stockage en colonnes, embarquant le schéma de données. Conçu en 2012 par Twitter et Cloudera, ce format est bien connu pour booster les performances de calculs lorsque l'on souhaite faire des agrégations avec Spark SQL, et est utilisable sur HDFS comme sur Amazon S3.

Ce format a également le bon goût d'être interopérable : utilisable pour du machine learning ou de la query ad hoc.

A ce jour, il n'est pas possible de stocker un fichier Parquet sur S3 directement depuis PDI. La solution fut de stocker des fichiers CSV, puis de les transformer en Parquet avec Spark.

Se pose alors la question de la performance globale, mais plusieurs éléments justifient cette conversion :

- En cas d'erreur dans les traitements, les données peuvent être traitées plusieurs fois dans un même but ;
- Au-delà du simple process de chargement de base de données, le datalake peut être utilisé par les data analysts qui souhaitent parfois utiliser la donnée brute, via Spark ou via d'autres outils.

Au vu du nombre de fichiers différents à convertir, une classe de conver-

sion générique, prenant en paramètre le schéma du fichier était nécessaire. Par ailleurs, il est nécessaire de gérer le mono header et multi-header. En effet, le datalake est composé de fichiers partitionnés contenant un entête par partition, et de fichiers peu volumineux, non partitionnés, avec un seul en-tête.

Le fichier source étant un CSV, il faut pour chaque élément créer et maintenir une classe contenant son schéma (ceci ne serait pas nécessaire avec un fichier en Parquet, AVRO ou JSON).

```
public class ElementOneDescriptor extends DataFrameDescriptor{

    public ElementOneDescriptor(String path) {
        super(path);
    }
    @Override
    public void buildSchema(PairList<String, DataType> schema) {
        schema.add("elementOne_id", DataTypes.LongType);
        schema.add("name", DataTypes.StringType);
        schema.add("value", DataTypes.StringType);
    }
}
```

Classe contenant le schéma du fichier

```
for (ExtractLoadElement element : elements) {
  String inputURI = "s3a://" + (get("bucketName") + "/"  + "csv";
  String outputURI = "s3a://" + (get("bucketName") + "/"  + "parquet";
  String parquetPath = element.getCsvPath() + "parquet/";
  DataFrameDescriptor descriptor = null;
  switch (element.getType()) {
      case "elementOne":
          descriptor = new ElementOneDescriptor(inputURI).setMultiHeader(true);
          break;
      case /*
           ...*/
          break;
  }
  descriptor.getDF(sqlContext()).coalesce(Integer.parseInt(get(PARAM_NB_PARTITIONS))).write()
                                .format("parquet").mode(SaveMode.Overwrite).save(outputURI);
}
```

Extrait de la classe de conversion en Parquet

Les performances constatées sont très satisfaisantes : 30 minutes pour convertir 1 To de données (l'archive d'une année).

Lecture des fichiers Parquet

Depuis la version 1.4.1 de Spark, rien de plus simple avec les dataframes, que de lire et d'utiliser un fichier stocké en Parquet.

L'exploitation des données est également très facile : il suffit d'interroger la table temporaire en SQL.

Il est possible notamment d'y inclure l'utilisation d'UDF (User Defined Fonction, voir détails plus bas). Le schéma étant inscrit dans le fichier Parquet, il est découvert automatiquement, aucun parsing n'est nécessaire. Dans l'exemple ci-dessous, nous sauvegardons les résultats en JSON, Redshift ne supportant que le chargement de fichier CSV, JSON, AVRO. Fig.2.

Broadcast variables et UDF

Les broadcasts variables sont des variables partagées par tous les nœuds du cluster. Au cours de l'exécution d'un job, elles sont immuables et doivent tenir en mémoire sur un seul nœud sans gêner l'exécution du job. C'est exactement le cas d'utilisation de la résolution de dimensions lors du chargement d'un datawarehouse.

En effet, les données brutes que nous avons en source contiennent souvent des valeurs ou un identifiant fonctionnel, plutôt qu'un identifiant technique. Nous avons dans le datawarehouse, les tables de références contenant un identifiant technique, l'identifiant fonctionnel présent dans la source, et, enfin, la valeur. Ces tables ont une très faible cardinalité. Nous avons donc tout intérêt à les broadcaster, plutôt que de faire de multiples appels à la base de données (à savoir à chaque ligne rencontrée). La mise en œuvre est assez simple : Fig.3.

La récupération d'identifiant technique est une tâche que nous aurons très souvent à réaliser, avec des données différentes. Pour éviter de dupliquer du code, il suffit de créer une UDF, qui recherche une valeur dans une table à partir d'une Map<clé, valeur> comme suit : Fig.4.

Plus globalement, nous avons créé une classe contenant toutes nos UDF (UDFHelper).

Lecture et écriture sur S3

Spark s'appuie sur les bibliothèques Hadoop pour la gestion des entrées / sorties sur S3. Hadoop propose trois clients pour accéder à S3:

- S3 plus simple à utiliser avec EMR (service managé par Amazon) ;
- S3n étant assez limité ;
- S3a (recommandé), remplaçant de S3n, levant ses limitations, mais nécessitant beaucoup de configuration.

Malheureusement, même avec S3a et la dernière version d'Hadoop, toutes les fonctionnalités d'AWS nécessaires à un bon support de S3 ne sont pas implémentées, notamment l'authentification temporaire via STS. Plus de détails sur ces trois clients et leurs spécificités ici https://hadoop.apache.org/docs/stable/hadoop-aws/tools/hadoop-aws/index.html.

Et la communauté ?

Spark attire de plus en plus de contributeurs (1000 fin 2015 contre 20 en 2013) et bénéficie d'une communauté d'utilisateurs de plus en plus large et connus (NetFlix, AirBnb,...). De ce fait, de nouvelles releases voient le jour tous les trimestres, apportant ainsi de nouvelles fonctionnalités, très souvent nécessaires à la bonne marche ou à l'industrialisation d'un projet. De plus, la documentation officielle est assez riche et mise à jour à chaque release, mais il peut s'avérer nécessaire, notamment pour de l'optimisation de lire le code source de l'API Spark pour en comprendre le fonctionnement. Par ailleurs, grâce à cette communauté grandissante, se débloquer d'un problème n'est qu'une question de jours : les forums, blogs et mailing-lists sont très actifs.

La transition à Spark en temps qu'ETL a été assez simple, la réalisation du code n'est clairement pas ce qui nous a pris le plus de temps. En revanche, il est important de prendre en compte, que l'utilisation d'un cluster Spark nécessite beaucoup de configuration pour arriver à un résul-

```
// Lecture du fichier parquet, et sauvegarde dans une table temporaire stockée en mémoire
sqlContext().read().format("parquet").load(get(PARAM_GENERIC_TABLE_PATH)).registerTempTable("site_table");
// Stockage des résultats dans une DataFrame
DataFrame results = sqlContext()
                    .sql("SELECT " +
                         "store_id ," +
                         // Utilisation d'une UDF
                         "getOccupationalCategoryId(occupational_category_name) AS OC_id ," +
                         // Utilisation d'une méthode de la classe SparkETL
                         get(PARAM_LOAD_PRIORITY) + " AS load_priority " +
                         "FROM  site_table "+
                         "WHERE element_type = 'pro' ");
// Stockage en JSON
results.write().format("json").mode(SaveMode.Overwrite).save(get(PARAM_OUTPUT_PATH));
```

Fig.2

```
private UDFHelper resolveDimensionsFromDWH(DatawarehouseDAO dwhDao) throws SQLException {
    // Récupération dans des map, des clés et valeur recherchées dans les référentiels
    final Map<Integer, Integer>                occCat        = dwhDao.getOCId();

    // Diffusion des variables sur tous les noeuds du cluster via le java spark context (jsc)
    final Broadcast<Map<Integer, Integer>>                   bvarOC    = jsc().broadcast(occCat);

UDFHelper udfDict = new UDFHelper(sqlContext());
    udfDictionary.registerLookup("getOCId", bvarOC, DataTypes.IntegerType, 1);
```

Fig.3

```
public <K, V> void registerLookup(String udfName, Broadcast<Map<K, V>> map, DataType retType, V unknownTag) {
  sqlContext.udf().register(udfName,
    (K key) -> {
      V value = map.value().get(key);
      if (value == null) return unknownTag;
      return value;
    },
    retType);
}
```

Fig.4

tat satisfaisant en termes de performance et de stabilité ; il est toutefois difficile de donner une "configuration idéale", celle-ci dépendant de beaucoup trop de paramètres.

Redshift, usages et spécificités

Redshift est une base de données analytique distribuée, hébergée et gérée par Amazon, basée sur PostgreSQL 8.0.2. Son stockage orienté colonnes lui permet une forte compression des données et une utilisation optimale des I/O, c'est idéal pour la lecture de masse et l'agrégation.

Après quelques tests, nous avons opté pour un cluster de 5 nœuds disposant chacun de 2 To d'espace disque et 32 Go de RAM. Nous savons que nos besoins évolueront avec l'agrandissement du périmètre et c'est là qu'opère la magie du service géré par Amazon.

- Pour ajouter un nœud avec une configuration équivalente aux nœud en place, cela se fait en quelques clics et en live. Le transfert de données à ce nouveau nœud s'opère automatiquement.
- Pour configurer un nouveau type de nœud, on est contraint de passer le cluster en maintenance pour quelques heures, le temps de la copie des données d'un cluster à un autre.

Cette flexibilité est un véritable avantage, surtout en phase de transition.

Modélisation et optimisations

Le mode de stockage de Redshift, rend obsolètes certaines règles que nous avions pu nous fixer par le passé. Par exemple, nous nous étions interdit dans la table des annonces, d'avoir une table contenant tous les critères possibles quelle que soit la catégorie. Si dans un premier temps, les analystes ont pu s'en contenter, le besoin d'analyses plus approfondies et spécialisées s'est très vite fait ressentir. Et il est difficile de trouver illégitime qu'un analyste se demande si les annonces Voitures de Leboncoin sont plutôt des Essence ou des Diesel, si des critères d'annonces sont sous-utilisés... Avec Redshift, nous avons pu, sans pénaliser le moteur de stockage ou les performances, fournir une table à plat contenant toutes ces informations. Cela offre également aux chefs de marché et de produit, des cubes par verticaux (Automobile, Immobilier, ...) plus adaptés et plus riches. En termes d'optimisation du modèle de données, deux options sont à aborder : sort key et distribution style.

Les sort keys permettent d'ordonner les données d'une table au sein du cluster afin d'améliorer les performances des requêtes de types range, agrégations, tris et les "window functions" effectuées sur cette clé. Cela évite de parcourir toute une table inutilement.

Elles sont spécifiées à la création de la table, et si elles sont multiples, deux types peuvent être utilisés :

- **Compounded sort keys** : ce type induit que l'ordre des champs est très important. Il est donc utile si les requêtes sont filtrées par toutes ces clés et dans l'ordre spécifié. Par ailleurs, les compounded sort keys améliorent la compression des données. En revanche, elles obligent en cas de chargement de volume important, à s'astreindre à des VACUUM et des ANALYZE fréquents, dans le cas contraire, les performances des requêtes peuvent être lourdement impactées.
- **Interleaved sort keys** : à l'inverse, chaque élément composant la sort key a un poids équivalent. L'usage de ces sort keys nécessite de passer régulièrement des VACUUM REINDEX sur les tables concernées.

Le distribution style est à définir à la création sur chaque table pour définir la répartition des lignes sur le cluster. Trois modes sont disponibles :

- **Even** : c'est le style de distribution par défaut, les lignes sont distribuées en round robin. Il est adapté lorsque la table n'est pas spécialement utilisée pour faire des jointures, qu'une clé évidente ne se détache pas ou qu'il n'y a pas d'intérêt à utiliser le style ALL.
- **All** : chaque nœud dispose de toutes les données d'une table. La colocation peut apporter des performances supplémentaires aux requêtes, mais cela se fait au détriment des temps de chargement et de l'espace de stockage. Cela peut donc être très utile et peu coûteux, lorsqu'il s'agit de tables de dimensions.

```
CREATE TABLE  common_layer.cl_payment_method
(
  payment_method_id integer NOT NULL,
  /**
  ...
  **/
)
DISTSTYLE ALL
```

- **Key** : on définit une clé fonctionnelle sur laquelle les jointures sont le plus souvent réalisées. Redshift répartira les données en fonction d'un modulo sur cette clé.

```
CREATE TABLE layer_1.l1_ad_price_history
(
ad_id integer NOT NULL,
  /**
  ...
  **/
)
DISTSTYLE KEY distkey(ad_id)
```

Même si l'utilisation de ces deux paramètres peut s'avérer contraignante en termes de maintenance, elle a apporté un véritable gain en termes de performance en rendant les plans de requêtes beaucoup plus cohérents. Il nous reste quelques optimisations à apporter quant au choix entre compounded key et interleaved sort key.

L'insertion de données dans Redshift

L'update dans Redshift est moins performant du fait du stockage en colonnes, Amazon recommande donc de passer par un COPY, quand il s'agit de charger / mettre à jour un volume important. Cela permet de charger le lot complet en une seule transaction et de compresser automatiquement les données. L'insert lui, peut se faire en un simple "Bulk".

Pour répondre à nos besoins, nous devons procéder par un "Annule et Remplace" en 4 temps :

Fig.5
```
COPY  staging_layer.tmp_el_one_staging_${current_work_load_id}
FROM 's3://${s3_staging_bucket}/${s3_staging_root}${current_load_id}/dwh/transform_el_one'
CREDENTIALS 'aws_access_key_id=${key_id};aws_secret_access_key=${access_key}'
JSON 'auto' -- Format source. L'option auto oblige à ce que les noms du schéma matchent le nom des champs cible
TRUNCATECOLUMNS; -- Tronque la valeur du champs source pour s'adapter au format du champs cible
```

Fig.6
```
DELETE FROM layer_1.l1_el_one
USING staging_layer.tmp_el_one_staging_${current_work_load_id}
WHERE staging_layer.tmp_el_one_staging_${current_work_load_id}.el_one_id = layer_1.l1_el_one.el_one_id
AND staging_layer.tmp_el_one_staging_${current_work_load_id}.load_priority >= layer_1.l1_el_one.load_priority;
```

Fig.7
```
FROM staging_layer.tmp_el_one_staging_${current_work_load_id} AS new_el_one
LEFT OUTER JOIN layer_1.l1_el_one AS former_el_one ON new_el_one.el_one_id = former_el_one.el_one_id
WHERE former_el_one.el_one_id IS NULL);
```

1 Création d'une table temporaire ;
2 Alimentation de cette table avec le fichier JSON généré par Spark ;

Pour cela, il suffit de spécifier la table cible, le chemin du fichier source, les informations d'authentification et le format du fichier source et d'autres paramètres. Fig.5.

3 Suppression des données à mettre à jour ;

Le point important ici, est de décider ce que l'on supprime ou pas. C'est là qu'intervient notre orchestrateur de chargement, qui définit une priorité des données en fonction de sa fraîcheur et de son origine (schéma source en cours d'utilisation ou archivé). Fig.6.

4 Chargement de la table cible.

La table est chargée depuis la table temporaire. Le point important est encore de ne charger que les données 'utiles'. Fig.7.

Maintenance & Monitoring

Contrairement à PostrgeSQL, pas d'AUTO-VACUUM avec Redshift. Il faut donc les lancer manuellement et très régulièrement, surtout lorsque l'on utilise les sort key. En revanche, avantage considérable, les VACUUM sont incrémentaux.

Par ailleurs, Amazon offre la possibilité de spécifier des profils de requêtes, permettant de prioriser les accès au cluster en fonction de ces profils.

Les backups sont gérés par Amazon, qui fournit de base, un outil de monitoring du cluster assez poussé, fournissant l'état de santé du cluster (extrait ci-dessous) et un suivi très illustré, des plans et de l'exécution des requêtes. Fig.8.

Les usages non adaptés

Redshift est parfaitement adapté pour des requêtes d'agrégation, récupérer la date minimale d'une annonce parmi 1 milliard d'annonces se fait en moins d'une minute sur notre cluster. En revanche, avoir les attributs d'une annonce en spécifiant son identifiant prend un peu plus de temps. Les requêtes opérationnelles ne sont clairement pas le cas d'utilisation.

Mais...

Malgré la puissance apportée par Amazon Redshift, certains points sont à déplorer, en termes de types de données et d'optimisation de requêtes.

Bien que Redshift propose des fonctions pour requêter un JSON, le type JSON n'existe pas, il faut passer par un type VARCHAR(65000), ce qui, en soi, n'est pas très propre, surtout que Redshift allouera de l'espace pour 65000 caractères, même si la colonne est vide.

Le type TEXT quant à lui est stocké comme un type VARCHAR(256) et a donc très peu d'intérêt.

Les types ARRAY et RANGE ne sont plus disponibles, au grand dam de nos analystes.

Quant aux optimisations de requêtes, impossible d'utiliser les CTE (Common Table Expression), on est donc obligés de passer par des sous-requêtes.

La liste complète des types, fonctions non-supportés est ici :
http://docs.aws.amazon.com/fr_fr/redshift/latest/dg/c_redshift-and-postgres-sql.html

Conclusion

Les outils distribués ouvrent clairement le champ des possibles de l'analytique. En effet, des formats de stockage tels que Parquet, couplés à Spark permettent des calculs d'agrégation beaucoup plus performants sur de fortes volumétries. Redshift de son côté, permet de requêter, agréger rapidement des volumes très importants. Son moteur de stockage colonne permet une modélisation de données moins contraignante.

Spark, bien que récent, a une forte communauté de contributeurs et d'utilisateurs dans le monde entier, cela permet notamment d'avoir un retour facilement, en cas de blocage. Les releases, à peu près tous les trimestres, apportent toujours un lot de nouvelles fonctionnalités intéressantes. Certes, ces outils sont encore en pleine évolution. Ainsi, on note quelques manquements dans les types de données et les fonctionnalités de Redshift, des migrations fréquentes et quelques peu forcées, une intégration à Redshift encore balbutiante et des dépendances à Hadoop parfois obsolètes du coté de Spark.

Malgré tout, l'utilisation de ces nouvelles technologies a conquis les développeurs comme les utilisateurs de notre stack analytique, de par la flexibilité apportée, et l'ouverture de nouvelles possibilités.

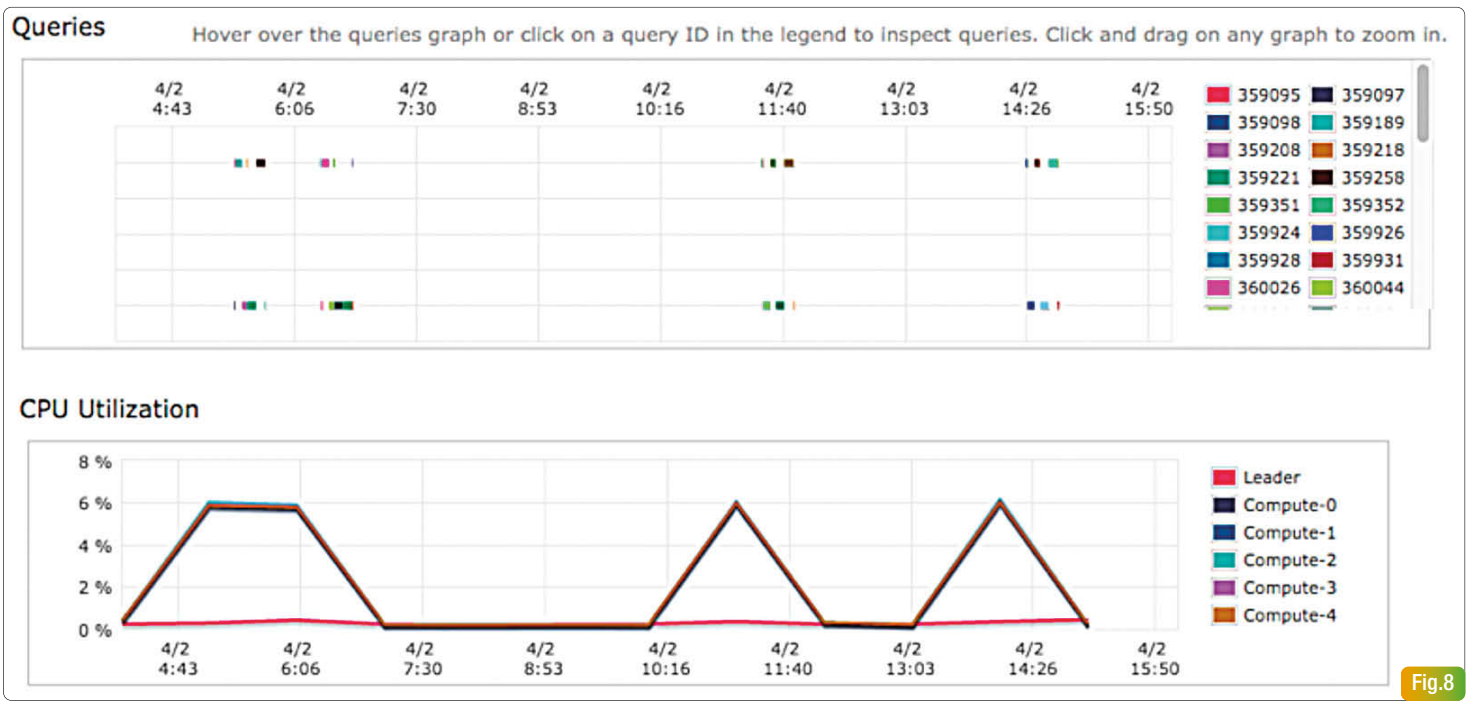


Fig.8

A la découverte de Spark Streaming

Il existe plusieurs systèmes de calculs distribués permettant de traiter une volumétrie importante de données Big Data, le tout en temps réel. L'une des solutions les plus connues, c'est Spark.

Amira LAKHAL
Développeuse Agile en Java et Scala chez Valtech
Duchess France Leader
@MiraLak

Apache Spark est un framework open source totalement écrit en Scala, permettant le traitement de données volumineuses à travers des calculs distribués en mémoire.
L'écosystème de Spark se compose de Spark Core qui est le projet principal effectuant les divers calculs distribués. On retrouve ensuite un ensemble de modules ajoutant des nouvelles fonctionnalités :

- Spark Sql permettant le traitement de données structurées en utilisant un langage très similaire à SQL ;
- GraphX permet l'exploration des graphes ;
- MLib qui contient toute une bibliothèque d'algorithmes de Machine Learning pour réaliser de la classification, de la recommandation... ;
- Spark streaming pour l'analyse en temps réel de gros volumes de données. Fig.1.

Du temps réel

Spark excelle dans l'analyse en temps réel grâce au module Spark Streaming. Spark Streaming permet ainsi d'analyser des flux de données en temps réel. Contrairement à Apache Storm qui gère le flux de données d'un coup (*one at a time*), Spark Streaming découpe le flux de données en plusieurs petits batchs à intervalles de temps réguliers avant de les traiter. Il faut noter qu'avec Storm Trident, il est possible aujourd'hui de faire du micro-batch sur un flux de données (*stream*) en s'assurant d'avoir une source fiable, alors que Spark Streaming gère très bien le cas d'une source de données avec des échecs. Fig.2.

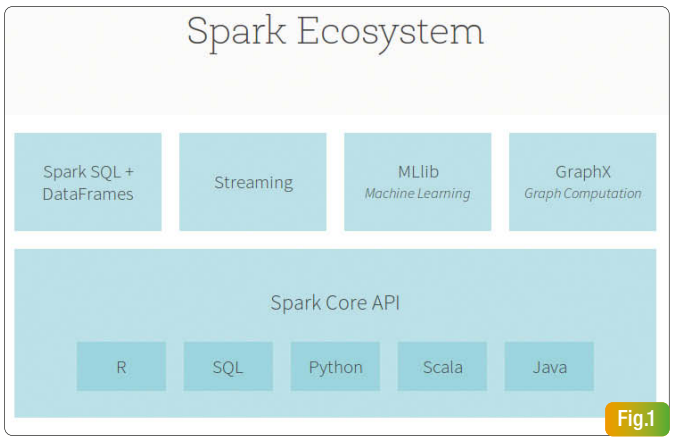
DStream

Spark Streaming se base sur une abstraction appelée DStream pour *Discritized Stream*. DStream représente un batch de RDD (*Resilient Distributed Datasets*).
RDD est l'abstraction de collection utilisée par Spark pour le traitement des données et sur laquelle on réalise les diverses opérations de calcul que ce soient des transformations telles que :

- *map()* transformer un élément en un autre élément ;
- *flatmap()* découper un élément en plusieurs autres éléments ;
- *filter()* filtrer les éléments en ne gardant que ceux qui vérifient la condition spécifiée ;
- *reduceByKey()* agréger des éléments entre eux ;
- mapToPair() création d'un tuple clé-valeur pour chaque élément ;
- Etc.

Ou des actions telles que :

- *count()* calcul des éléments finaux ;
- *collect()* retourner les éléments sous forme d'une collection Java ;
- *saveAsTextFile()* pour sauvegarder les résultats dans un fichier texte ;
- *cache()* pour sauvegarder le résultat en mémoire ;
- *persist()* pour sauvegarder le résultat sur disque ;
- Etc.



L'écosystème Spark, source databricks.com

Le process de Spark Streaming, source databricks.com

Data sources

Spark Streaming s'intègre avec plusieurs sources de données telles que :

- Apache Kafka : un système de messaging distribué ;
- Apache Flume : un système de collecte de logs ;
- Twitter : en récupérant les tweets avec l'API Twitter de streaming ;
- TCP sockets.

Spark Streaming propose aussi une intégration pour les autres data sources qui ne bénéficient pas d'une intégration prédéfinie. Il suffit d'implémenter la classe *Receiver*.

Multi-langage

Apache Spark supporte plusieurs langages : Scala, Java, Python et R.
Pour les exemples suivants, le langage utilisé est Scala.
Scala est un langage de programmation multi-paradigme qui s'intègre avec les paradigmes de programmation objet et de programmation fonctionnelle. Il se caractérise par un typage statique, l'inférence de type ou les conversions implicites qui rendent l'utilisation d'Apache Spark plus simple.

Exemple: *word count*

L'objectif de cet exemple est de calculer le nombre d'occurrence de mots en temps réel. La première étape est d'initialiser le contexte Spark Streaming :

```
import org.apache.spark._
import org.apache.spark.streaming._

val conf = new SparkConf()
             .setAppName("myFirstApp")
             .setMaster(local[*])

val ssc = new StreamingContext(conf, Seconds(2))
```

On crée le contexte Spark en précisant le nom de notre application ainsi

que le cluster manager. Dans cet exemple, le mode de lancement de Spark est en standalone. C'est la solution la plus simple qui se base sur *Akka* pour les échanges et sur *Zookeeper* pour la haute disponibilité du nœud *master*. L'exécuteur a été initialisé avec [*] , ce qui veut dire autant de threads que possible, en sachant qu'il faudra toujours avoir au moins deux threads : un pour la lecture des données et un autre pour le traitement des données. Spark peut faire appel à d'autres gestionnaires de cluster tel que Mesos ou YARN.
Après l'initialisation du contexte Spark, on initialise le spark streaming context à partir de ce dernier et on précise la période de streaming qui est de 2 secondes pour cet exemple.
L'étape suivante est de définir le flux de données à utiliser. Il existe plusieurs possibilités de flux de données. Ici, on va présenter quelques types de flux pour notre exemple simple : le calcul du nombre d'occurrences de chaque mot d'un texte.

Socket stream

Les sockets sont des connecteurs réseau qui permettent d'utiliser le protocole TCP ou UDP. Ici, ça sera plutôt le protocole TCP (*Transmission Control Protocol*). L'idée est de récupérer le flux de données en temps réel. Les données récupérées sont sous la forme d'un texte.
L'idée ici est de calculer le nombre d'occurrences de chacun des mots toutes les deux secondes.

```
val lines = ssc.socketTextStream(host, port, StorageLevel.MEMORY_AND_DISK_SER
val words = lines.flatMap(_.split(" "))
val wordCounts = words.map(x => (x, 1)).reduceByKey(_ + _)
wordCounts.print()
ssc.start()
ssc.awaitTermination()
```

Après avoir décrit les actions à faire pour le flux, on démarre le streaming en appelant l'action *ssc.start().* Ensuite, le traitement est lancé sur des threads différents. On peut arrêter le thread principal soit en précisant un événement particulier, soit en attendant la fin de l'exécution *ssc.awaitTermination()*.

Kafka stream

Kafka est système de messagerie distribué. Il a pour but de conserver les messages de façon temporaire (comme pour un système de Queuing) permettant de garantir, lors de forts pics de charge, l'intégrité du message. Je vous conseille de lire l'article dessus qui est inclus dans ce numéro pour avoir plus d'informations.

```
// Create direct kafka stream with brokers and topics
val kafkaParams = Map[String, String]("metadata.broker.list" -> brokers)
val messages = KafkaUtils.createDirectStream[String, String, StringDecoder, StringDecoder](
  ssc, kafkaParams, topicsSet)

// Get the lines, split them into words, count the words and print
val lines = messages.map(_._2)
val words = lines.flatMap(_.split(" "))
val wordCounts = words.map(x => (x, 1L)).reduceByKey(_ + _)
wordCounts.print()

// Start the computation
ssc.start()
ssc.awaitTermination()
```

Pour écouter les messages, il faut tout d'abord configurer kafka : spécifier les brokers et la liste des topics. Ensuite, pour chaque message, on récupère les mots et on calcule le nombre d'occurrences de chaque mot.

Custom receiver

Si on souhaite récupérer des données dont il n'existe pas d'intégration directe dans Spark Streaming, on peut créer notre propre *Receiver*.

```
object CustomReceiver {

    val sparkConf = new SparkConf().setAppName("CustomReceiver")
    val ssc = new StreamingContext(sparkConf, Seconds(2))

    val lines = ssc.receiverStream(new CustomReceiver(myHostname, port))
    val words = lines.flatMap(_.split(" "))
    val wordCounts = words.map(x => (x, 1)).reduceByKey(_ + _)
    wordCounts.print()

    ssc.start()
    ssc.awaitTermination()
}
```

L'initialisation est la même que précédemment sauf que l'on fait appel à la méthode *receiverStream* de Spark Streaming qui prend en paramètre le *Custom receiver*. Le Custom receiver nous renvoie comme résultat des lignes de texte sur lesquelles on applique les mêmes transformations que précédemment pour obtenir le nombre d'occurrences des mots.
Si on regarde de plus près le *Custom Receiver*:

```
class CustomReceiver(host: String, port: Int)
  extends Receiver[String](StorageLevel.MEMORY_AND_DISK_2) with Logging {

  def onStart() {
    new Thread("Socket Receiver") {
      override def run() { receive() }
    }.start()
  }

  def onStop() {
  }
}
```

La classe *Custom Receiver* étend l'interface *Receiver* fournie par Spark Streaming et qui contient les deux méthodes : *start()* et *stop()*.

```
private def receive() {
 var socket: Socket = null
 var userInput: String = null
 try {
   socket = new Socket(host, port)
   val reader = new BufferedReader(
     new InputStreamReader(socket.getInputStream(), StandardCharsets.UTF_8))
   userInput = reader.readLine()
   while(!isStopped && userInput != null) {
     store(userInput)
     userInput = reader.readLine()
   }
   reader.close()
   socket.close()
   logInfo("Stopped receiving")
   restart("Trying to connect again")
 } catch {
   case e: java.net.ConnectException =>
     restart("Error connecting to " + host + ":" + port, e)
   case t: Throwable =>
     restart("Error receiving data", t)
 }
}
```

Dans la méthode *start()*, on précise la source de données qui sera utilisée. Ici, dans la méthode *receive()*, on lit les données d'un InputStream et on sauvegarde chacune des lignes lues pendant la durée du batch qui est de 2 secondes.
Les lignes récupérées via le *Custom Receiver* seront traitées et un nouveau batch va aussitôt se lancer pour récupérer des nouvelles lignes et les analyser et ainsi de suite.

DataFrames

En plus de la lecture de flux de données en temps réel, Spark Streaming peut aussi récupérer un flux de données et le transformer pour le stocker dans une base de données. Dans ce cas, on initialise en plus un Spark SQL context en se basant sur Spark Context, et on crée un Spark SQL Dataframe.

```
val lines = ssc.socketTextStream(hostname, port, StorageLevel.MEMORY_AND_DISK_SER)
val words = lines.flatMap(_.split(" "))

// Convert RDDs of the words DStream to DataFrame and run SQL query
 words.foreachRDD((rdd: RDD[String], time: Time) => {
// Get the singleton instance of SQLContext
val sqlContext = SQLContextSingleton.getInstance(rdd.sparkContext)
import sqlContext.implicits._

// Convert RDD[String] to RDD[case class] to DataFrame
val wordsDataFrame = rdd.map(w => Record(w)).toDF()

// Register as table
wordsDataFrame.registerTempTable("words")

// Do word count on table using SQL and print it
 val wordCountsDataFrame =
  sqlContext.sql("select word, count(*) as total from words group by word")
  println(s"========= $time =========")
  wordCountsDataFrame.show()
 })

ssc.start()
ssc.awaitTermination()
```

On transforme ainsi les données en Dataframe et ensuite, on effectue le calcul du nombre d'occurrences en faisant appel à une simple requête SQL.
Attention à un détail, le Spark SQL context doit être un singleton :

```
/** Lazily instantiated singleton instance of SQLContext */
object SQLContextSingleton {

 @transient  private var instance: SQLContext = _

 def getInstance(sparkContext: SparkContext): SQLContext = {
 if (instance == null) {
  instance = new SQLContext(sparkContext)
 }
    instance
  }
}
```

Le cas d'usage peut être mis en place dans le cas où on a besoin de récupérer des informations et de les sauvegarder pour servir dans des traitements ultérieurs.

Apache Flume

Apache Flume est un système distribué qui permet d'agréger un grand volume de données en temps réel. Les données peuvent être des logs ou des événements.

On change l'exemple précédent qui était le calcul du nombre d'occurrences d'un mot pour calculer le nombre d'événement reçus. Les étapes initiales sont les mêmes : on initialise un context spark et spark streaming comme précédemment en précisant l'intervalle de batch.

```
// Create the context and set the batch size
val sparkConf = new SparkConf().setAppName("FlumeEventCount")
val ssc = new StreamingContext(sparkConf, Milliseconds(2000))

// Create a flume stream
val stream = FlumeUtils.createStream(ssc, host, port, StorageLevel.MEMORY_ONLY_SER_2)

// Print out the count of events received from this server in each batch
stream.count().map(cnt => "Received " + cnt + " flume events." ).print()

ssc.start()
ssc.awaitTermination()
```

Ensuite, on fait appel à *FlumeUtils* pour créer le flux de données et réaliser le traitement souhaité : le calcul du nombre d'événements reçus.

Cas d'utilisation

Spark Streaming est utilisé dans différents cas :

- Streaming ETL : les données sont nettoyées et agrégées avant d'être sauvegardées ;
- Triggers : détection des comportements suspects en temps réel (fraude) et enclenchement des actions nécessaires ;
- Enrichissement des données : les données temps réel sont enrichies avec d'autres informations afin de compléter l'analyse ;
- Apprentissage continu et session : regroupement et analyse des événements liés à une session active (authentification à un site Web) et potentiellement utilisation des informations pour mettre à jour les models de machine learning.

Conclusion

Spark Streaming est l'une des meilleures solutions actuelles pour le traitement de flux de données en temps réel et en continu. La prise en main est très facile car Spark Streaming propose déjà l'intégration de plusieurs types de sources de données.

References

http://spark.apache.org/docs/latest/streaming-programming-guide.html
https://github.com/apache/spark/tree/master/streaming
https://github.com/apache/spark/tree/master/examples/src/main/scala/org/apache/spark/examples/streaming

Rabbitmq, le broker de message

Rabbitmq est un MOM (Message-Oriented Middleware) open source distribué sous licence "Mozilla Public Licence" et développé en Erlang. Il s'appuie sur le pattern de broker de messages : des Producers envoient des messages à un point central qui les distribue ensuite dans des files où ils seront récupérés par des Consumers. ActiveMQ, Kafka ou WebSphere MQ (anciennement MQSeries) sont des solutions similaires.

Pauline Logna
Tech-Lead à la **Casden**
Duchess France Leader
@pauline_io

Parmi les principales caractéristiques de Rabbitmq, nous pouvons citer sa **fiabilité** nous garantit de ne pas perdre de messages grâce à un mécanisme d'**acquittement**, ou encore, le **routage** évolué des messages.
Pour garantir la **haute disponibilité**, la **répartition de charge** (avec un répartiteur de charge en amont) ou encore faciliter la **montée en charge**, il est possible de définir une architecture de type cluster associée à des politiques de pointues réplication des messages.
Aussi, RabbitMQ dispose d'une console d'administration Web, offre la possibilité de développer ses propres plugins, et est accessible depuis de multiples librairies clientes dans les principaux langages. Je vous propose de découvrir un MOM attractif quel que soit votre environnement technique.

Pourquoi RabbitMQ ?

RabbitMQ est une solution séduisante et performante à plus d'un titre.
D'abord, d'un point de vue MOM, les Architectures Orientées Messages favorisent le **découplage technique** entre applications, c'est-à-dire limitent l'adhérence entre ces dernières, ce qui rendra possible et plus simple une réelle évolutivité du SI. Ainsi, remplacer par exemple un composant écrit en Java par un autre écrit en Python, ou substituer un composant par un autre plus performant, est sans impact sur le reste de l'architecture. Dès lors, RabbitMQ s'intègre très bien dans une **architecture orientée micro-services**, dont les différents composants sont développés sous la forme de services autonomes qui s'exécutent avec leur propre processus. Consumers et Producers peuvent être développés comme des micro-services. D'autres cas concrets d'utilisation de RabbitMQ sont liés aux aspects asynchrones mis en œuvre, associés à la capacité de montée en charge. Ainsi vous pouvez migrer progressivement d'une culture batch legacy vers des traitements au fil de l'eau, ou gérer la transmission de données en masse (Big Data). De même, toutes les fonctionnalités nécessitant une haute fiabilité et non synchrones sont concernées - envoi de mails ou de SMS par exemple.
De même RabbitMQ peut fiabiliser les envois et consommations de messages, notamment par un mécanisme d'accusé de réception (ACK), par l'utilisation de listeners détectant les pertes de connexion, les messages mal routés, etc. Cette haute fiabilité est une caractéristique au cœur de RabbitMQ dont le protocole AMQP a été défini par le domaine bancaire. A titre d'information, Google a réalisé avec succès le traitement d'1 million de messages / seconde.

Protocole AMQP

Nativement RabbitMQ implémente le protocole **AMQP (Advanced Message Queuing Protocol)** développé initialement par la banque JP Morgan et standardisé ensuite par un consortium dont Red Hat et CISCO.
Néanmoins, l'ajout de plugins dédiés permet aussi à RabbitMQ de supporter d'autres protocoles dont STOMP, MQTT, HTTP.
Dans le cadre de cet article nous ciblerons le protocole natif.
L'objectif d'AMQP est de standardiser les échanges entre serveurs de messages en associant fiabilité et sécurité.
Contrairement à JMS (Java Messaging Service) qui ne définit qu'une API, AMQP est un protocole "wire-level", c'est-à-dire un protocole qui décrit une façon standard de représenter l'information au niveau applicatif.
Ainsi, par exemple, AMQP définit qu'un message est composé de métadonnées (entêtes et propriétés sous forme de clés/valeurs), et d'un corps (le contenu du message sous la forme d'un tableau de bytes).
En outre RabbitMQ utilise le pattern point à point : un message qui se trouve dans une file ne peut être retiré que par un et un seul Consumer (même si plusieurs Consumers peuvent être associés à une même file).

Client : producer ou consumer

L'utilisation de RabbitMQ se fait par le développement de clients qui s'y connectent, soit pour produire des messages (producer), soit pour les consommer (consumer).
Un producer envoie des messages vers un point central (Exchange), lequel route chaque message vers une (ou plusieurs) file(s). Ils seront alors consommés par les Consumers.
Lorsqu'un message est consommé, il est supprimé de la file par RabbitMQ après la réception de l'accusé de réception. Techniquement, la production et la consommation des messages se font donc sur un pattern de type Publish/Subscribe.

Se connecter à rabbitmq avec un client java

La première chose à faire pour un client rabbitmq, que ce soit pour produire ou consommer des messages, est de créer une connexion. L'API RabbitMQ fournit un objet `ConnectionFactory` qui va permettre de récupérer une connexion après avoir renseigné les informations nécessaires telles que le user et son mot de passe, le vhost, l'utilisation du ssl et un tableau avec les host et numéros de port de chaque noeud. Fig.1.
Une fois obtenue la connexion à partir de la factory, il suffit de créer un objet de type `Channel` chargé de produire ou consommer des messages.

Fig.1

```
public Connection createNewConnection() throws IOException, TimeoutException {
    // Build factory.
    ConnectionFactory factory = new ConnectionFactory();
    factory.setVirtualHost(vhost);
    factory.setUsername(user);
    factory.setPassword(password);
    factory.setAutomaticRecoveryEnabled(true);
    factory.setNetworkRecoveryInterval(10000);
    factory.setTopologyRecoveryEnabled(true);
    try {
        factory.useSslProtocol();
    } catch (KeyManagementException | NoSuchAlgorithmException e) {
        logger.error("Fail to use ssl protocol for rabbitmq  hosts '" + node1Host + "' and '" + node2Host + "'", e);
        throw new IllegalStateException("Could not use ssl protocol  for rabbitmq hosts '" + node1Host + "' and '" + node2Host + "'");
    }

    // Create new connection.
    Address[] addrArr = { new Address(node1Host, Integer.parseInt(node1Port)), new Address(node2Host, Integer.parseInt(node2Port)) };
    Connection connection = factory.newConnection(addrArr);
    return connection;
}
```

Produire et router des messages

La publication de messages se fait par envoi sur un exchange. Les exchanges sont en amont des files et vont router les messages vers la (ou les) file(s) techniquement associées à cet exchange.

On parle de binding entre exchange et files.

S'il est possible d'un point de vue programmation d'envoyer des messages directement sur une file sans passer explicitement par un exchange, techniquement ce message passe par un exchange défini par défaut dans Rabbitmq. Il existe différents types d'exchanges :

- Fanout

Diffuse sans condition les messages vers toutes les files auxquelles l'exchange est bindé. Fig.2.

- Direct

Associe des exchanges à des files en spécifiant des clés de routage (plusieurs files peuvent définir une même clé de routage). Fig.3.

Si on envoie un message avec la clé de routage ged, un traitement de mise en ged sera lancé ainsi qu'un envoi d'email.

- Topic

Certains messages ne peuvent être routés arbitrairement selon une clé de routage simple. En effet, en pratique on peut avoir besoin de combiner des traitements sur des ensembles fonctionnels de messages et des traitements de cas particuliers. On va alors plutôt s'orienter sur des bindings de clés composées de mots appartenant à un ensemble fonctionnel, ex : roman.policier.poche

Pour envoyer des ensembles de messages sur des files, on va alors avoir la possibilité de mettre en place un routage des messages à partir de caractères "jwild card" définissant un **pattern** dans la clé de routage.

Il suffit donc d'envoyer un message en choisissant une clé de routage qui déterminera les ensembles fonctionnels auxquels le message appartient et déclenchera les traitements associés.

Fig.5

```
public void produceMessage(byte[] message, Type type) throws Exception {

    // Get common channel.
    Channel channel = producerConnection.getChannel();

    // Publish message on queue.
    BasicProperties properties = new Builder()
            .contentType("application/octet-stream")
            .type(type.name())
            .deliveryMode(2)
            .priority(0).build();
    channel.basicPublish(exchangeName, routingKey, true, false, properties, message);
    logger.info(" [x] published on exchange " + exchangeName + " ' with routing key '" + routingKey + "' " + message + "'.");

}
```

Fig.2

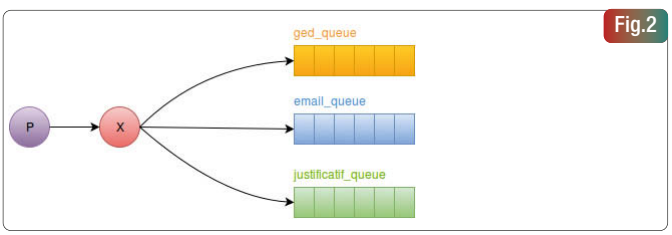


Fig.3

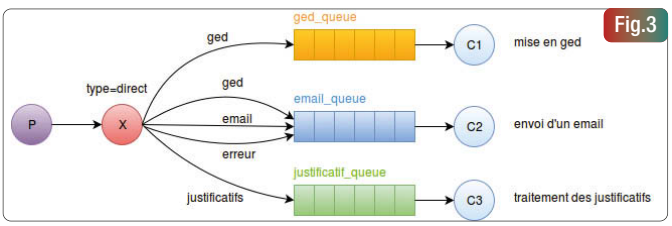
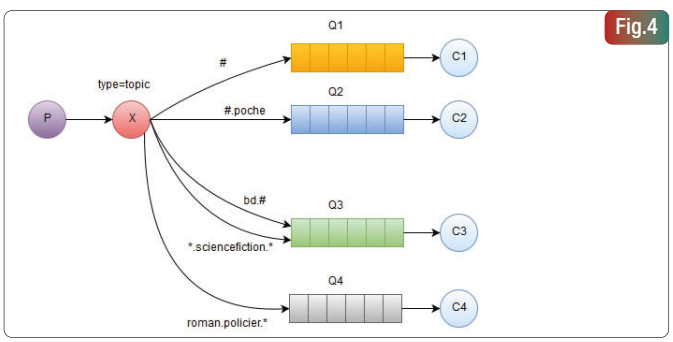


Fig.4



Dans l'exemple ci-dessous, les messages envoyés concernent des livres et les ensembles fonctionnels de messages que l'on veut distinguer sont le genre (roman, bd, ...), le sous-genre (policier, science fiction, histoire, poésie...) et le format (poche, a4, relié, broché, numérique...). Fig.4.

Dans ce cas, la clé de routage est composée de plusieurs mots. L'exchange X est bindé aux files Q1, Q2, Q3 et Q4 avec des patterns de clés.

*: peut se substituer à exactement un mot

#: peut se substituer à un ou plusieurs mots

Dans cet exemple, une clé de routage est construite sur trois mots : <genre>.<sous-genre>.<format>

Par exemple un message produit sur l'exchange X avec la clé de routage roman.aventure.poche sera envoyé sur les file Q1 et Q2.

En revanche un message produit sur cet exchange avec la clé de routage bd.fantasy.numerique sera envoyé sur les files Q1 et Q3.

La file Q4 ne recevra que les message dont le premier mot de la clé de routage est roman et le deuxième est policier alors que la file Q1 recevra tous les messages quelle que soit la clé de routage.

Produire un message sur une file en Java

La méthode produceMessage() prend en paramètre le message lui même et son type. Fig.5.

L'appel de la méthode basicPublish sur un channel ouvert provoque l'envoi du message.

Les différents paramètres à renseigner :

- L'exchange

A noter qu'on pourrait aussi publier directement sur une file de messages, alors un exchange par défaut serait utilisé par RabbitMQ.

- La clé de routage

Peut ne pas être obligatoire en fonction du type d'exchange.

- Le flag mandatory (le 3ème paramètre)

Ce flag est facultatif. Il correspond aux spécifications AMQP et permet de spécifier au serveur le comportement à avoir si après comparaison de la clé de routage et des binding le message ne peut être publié dans aucune file. Si ce flag est true, alors le message sera retourné à l'envoyeur via un listener de type ReturnListener.

Si le flag est false, le message sera perdu sans erreur ni notification.

- Le flag immediate (le 4ème paramètre)

Ce flag est facultatif et correspond aussi aux spécifications AMQP.

Si ce flag est true, alors le message est immédiatement délivré à un Consumer. Si aucun Consumer n'est connecté, le message est perdu.

Si le flag est false, alors le message reste en file jusqu'à sa consommation.

- Les propriétés

Ce sont des métadonnées du message qui correspondent aux spécifications AMQP. Dans l'exemple de code ci-dessus, on utilise l'objet BasicProperties pour typer le message. Mais elles permettent aussi d'ajouter d'autres informations telles que la date, un id d'utilisateur, une date d'expiration, etc.

Consommation de messages

Un Consumer est un client qui établit une connexion sur une file donnée

pour appliquer un traitement au message. Un message consommé une et une seulle fois, ensuite, il est supprimé de la file.

Si on souhaitait le consommer il faudrait donc logiquement le re-publier explicitement dans la file.

Quand le traitement s'est correctement déroulé, on peut envoyer manuellement un acquittement qui engendrera la suppression du message de la file par RabbitMQ.

Dans le cas où une une erreur survient lors du traitement du Consumer, on conserve la possibilité de renvoyer un acquittement négatif (NACK) . Lorsque RabbitMQ reçoit cet acquittement, il ne supprime pas le message de la file, ce message pourra donc être de nouveau consommé ultérieurement. A noter que si l'erreur rencontrée par le Consumer est réellement bloquante, alors la stratégie consistera plutôt en l'utilisation d'une file "Dead Letter Queue" (DLQ), l'équivalent d'une voie de garage pour des messages nécessitant un traitement particulier.

On peut avoir plusieurs Consumers en écoute sur une même file, ces Consumers devant être différentes instances d'un même traitement car un message consommé par l'un ne pourra pas l'être par l'autre. On aura bien à la fin un traitement par message, tout en augmentant la capacité de traitement. Enfin, il est intéressant de noter que le Channel RabbitMQ utilisé par ces Consumers peut lui aussi faire intervenir une notion de qualité de service (QoS) pour mieux répartir la charge entre les Consumers.

Consommer un message sur une file en java Fig.6.

L'appel à la méthode channel.basicConsume() permet de passer en paramètres les informations sur la façon dont va se faire la consommation au channel :

- Le nom de la file sur laquelle on va consommer les messages ;
- Le flag autoAck.

Si ce flag est à true alors l'acquittement se fera automatiquement après consommation du message.

Si ce flag est à false alors l'acquittement se fera manuellement. On aura la possibilité d'envoyer un acquittement channel.basicAck() ou un acquittement négatif channel.basicNack().

En plaçant l'acquittement après la fin du traitement, on a l'assurance que la suppression du message de la file se fait une fois le traitement effectué sans erreurs.

Si jamais une erreur est déclenchée, on a alors la possibilité d'envoyer un acquittement négatif, ainsi le message n'est pas supprimé de la file par RabbitMQ et pourra être de nouveau consommé ultérieurement.

Attention à un effet de bord : mal utiliser les acquittements peut conduire à la saturation des files.

- Un objet Consumer: il s'agit de lier un objet implémentant l'interface Consumer (ici QueuingConsumer) au channel.

L'appel à la méthode nextDelivery() exécute la consommation d'un message. Cette méthode renvoie un objet de type Delivery à partir duquel on pourra récupérer le corps du message : un tableau de bytes obtenu en appelant la méthode getBody(), mais aussi les métadonnées – inscrites dans un objet BasicProperties – obbtenues en appelant getProperties() sur l'objet delivery.

On peut noter dans notre exemple que le traitement du message task.execute(delivery.getBody()) est protégé par un try/catch générant le cas échéant un message typé error sur un exchange deadExchange.

Le pattern de la Dead Letter Queue peut alors être mis en œuvre en bindant l'Exchange deadExchange à une file d'erreurs DLQ. Ce mécanisme permet de sauvegarder les messages dans la DLQ pour les traiter avec un Consumer de gestion d'erreurs dédié.

Fig.6

```
public void consumeMessage() {
    try {
        Channel channel = consumerConnectionSingleton.getChannel();

        // consumer in channel.
        QueueingConsumer consumer = new QueueingConsumer(channel);
        channel.basicConsume(queueName, autoAck, consumer);

        // Wait for message and consume.
        logger.info(" [x] Wait for message on queue " + queueName);

        while (true) {
            QueueingConsumer.Delivery delivery = consumer.nextDelivery();
            logger.info(" [x] Received " + delivery + " on queue " + queueName);
            try {
                task.execute(delivery.getBody());
            } catch (Exception e) {
                // Send message.
                logger.error("An error occurs treating message : produce a message on exchange '" + deadProducer.getExchangeName() +"'
                deadProducer.produceMessage(delivery.getBody(), error);
            }
            // Send acknowledgement manually. basicAck or basicNack ?
            channel.basicAck(delivery.getEnvelope().getDeliveryTag(), false);
            logger.info(" [x] Done consumeing on queue " + queueName);
        }
    } catch (Exception e) {
        logger.error("An error occurs", e);
        // TODO send a baseNack ?
    }
}
```

Fig.7

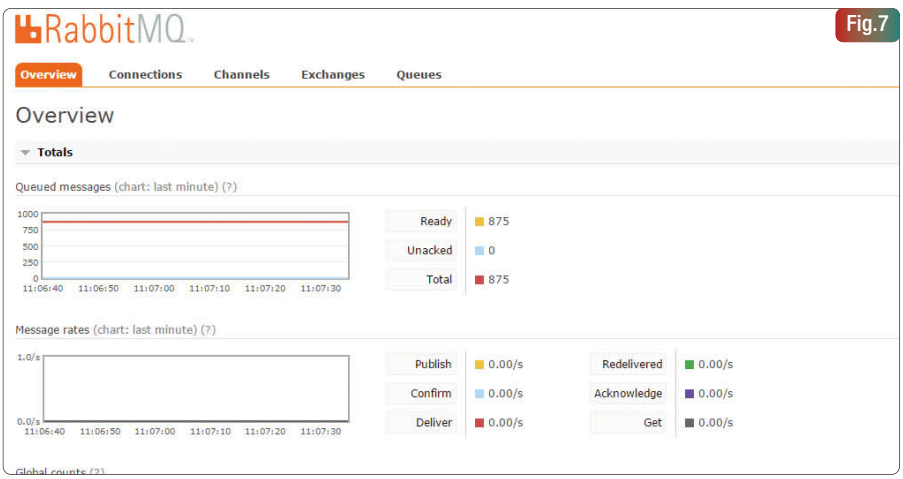


Fig.8

Administrer les files de messages

RabbitMQ fournit une interface Web permettant d'administrer et de monitorer les files de messages et les exchanges. Pour pouvoir l'utiliser il faut installer le plugin rabbitmq-management.

Overview du trafic Fig.7.
Chiffres clés comme le nombre de connexions Fig.8.

Administration files et exchanges

Des vues sur les exchanges, les files, les channels et les connexions.

A partir de cette interface, on peut créer des exchanges et des files, on peut publier des messages directement sur les files. Fig.9.

On peut également consommer des messages. Si on est amené à devoir consommer des mes-

sages directement sur une file (cas exceptionnel), il faut être vigilant et remettre ce message en file avec le champs Requeue. Car un message consommé est un message perdu, il est enlevé de la file. Fig.10.

Vhost

Un vhost définit un espace de nommage utilisé pour isoler différents "environnements" (groupes d'utilisateurs, exchanges, files, bindings).

Utilisateurs

Habilitations à la Web ui d'admin :

management

L'utilisateur peut accéder à l'ihm de management.

policymaker

L'utilisateur peut accéder à l'ihm de management et manager les policies et les paramètres dans les vhosts auxquels il a accès.

monitoring

L'utilisateur peut accéder à l'ihm de management et voir toutes les connexions et channels ainsi qu'aux informations concernant les noeuds du cluster.

administrator

L'utilisateur a accès à toutes les fonctionnalités proposées par l'ihm de management, manager les utilisateurs, les vhosts et les permissions. Il peut fermer les connexions, manager les policies et les paramètres de tous les vhosts.

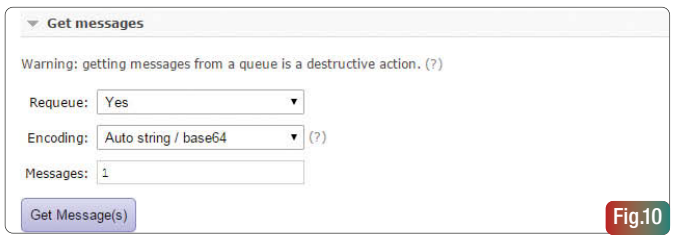


Fig.10

Gestion des policies

Les policies servent à déterminer des restrictions par vhost et comment files et exchanges vont se comporter. Sur l'image ci-dessous on peut voir que la policy nommée ged s'appliquera aux files dont le nom contient le mot ged. Le paramètre ha-mode: all signifie que des files esclaves seront définis sur tous les noeuds du cluster.

Avec les autres options on pourrait spécifier le nombre de noeuds ou une liste de noeuds à répliquer en combinant ha-mode avec ha-param. Le paramètre ha-sync-mode: automatic signifie que la synchronisation entre les files maîtres et esclaves se fait automatiquement. Fig.11.

Ligne de commande

Toutes les opérations possibles dans l'interface graphique sont également réalisables en ligne de commande.

- Se connecter à un autre host avec un autre utilisateur

```
$ rabbitmqadmin -H myserver -u simon -p simon list vhosts
+-------+
| name  |
+-------+
| /     |
+-------+
```

- Déclaration d'un exchange

```
$ rabbitmqadmin declare exchange name=my-new-exchange type=fanout
exchange declared
```

- Déclaration d'une file

```
$ rabbitmqadmin declare queue name=my-new-queue durable=false
queue declared
```

- Publication d'un message

```
$ rabbitmqadmin publish exchange=amq.default routing_key=test payload="hello, world"
Message published
```

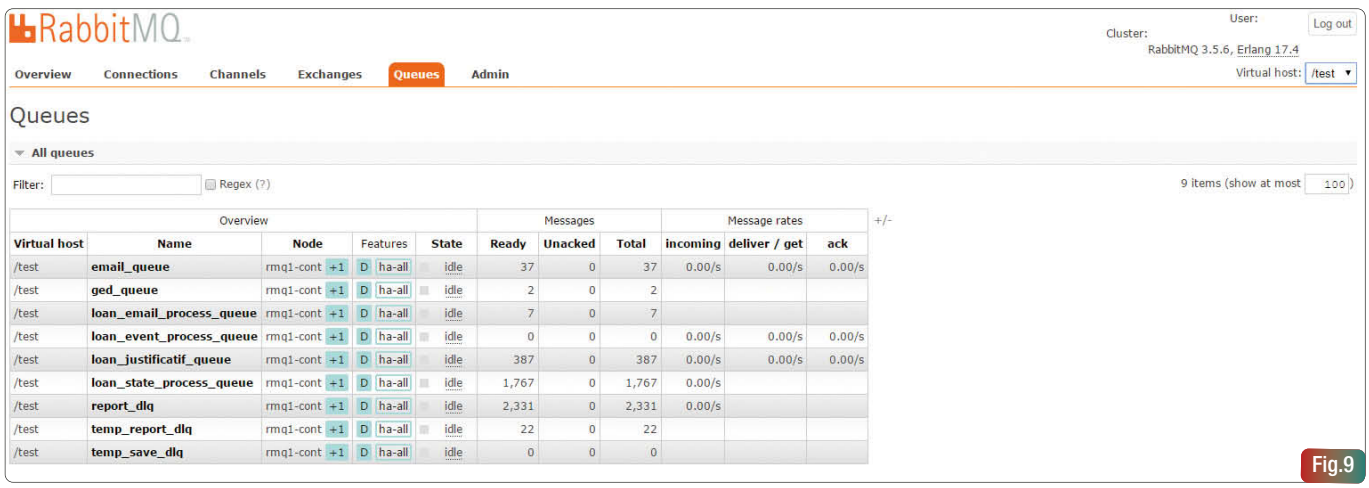


Fig.9

Fig.11

Cas pratique

Le besoin initial de la Casden était de développer le suivi d'une demande de prêt. Choix technique : introduire deux nouvelles briques dans le SI

- Bpm activiti
- Rabbitmq

Dans un premier temps il fallait offrir la possibilité d'envoyer des évènements faisant avancer un processus de suivi de demande de prêt, tout en historisant les changements d'états successifs du prêt. Les files de mise en ged et d'envoi d'email ont été implémentées par la suite.

Voici l'architecture mise en place à la Casden qui est en production depuis le mois janvier 2016. Fig.12.

D'abord nous avons mis en place des files loan_event_queue et loan_state_queue.

Des événements sont envoyés sur la file dédiée loan_event_queue. En écoute sur cette file, un micro-service embarquant la brique Activiti consomme chaque message et fait avancer les instances des processus Activiti correspondants.

A chaque passage à un nouvel état du processus, un message est envoyé sur la file loan_state_queue, et le Consumer en écoute sur la file met à jour l'état du prêt en base.

L'application de suivi de demande n'a qu'à interroger cette base pour récupérer l'état courant ou l'historique d'un prêt.

D'autre besoins ont ensuite émergé comme des envois d'emails. Une file email_queue a alors été mise en place avec un Consumer dédié qui a pour responsabilité la construction du mail à partir des données du message et son envoi par appel d'un web service. L'objectif étant qu'à terme toutes les applications du SI ayant besoin d'envoyer des emails utilisent cette file.

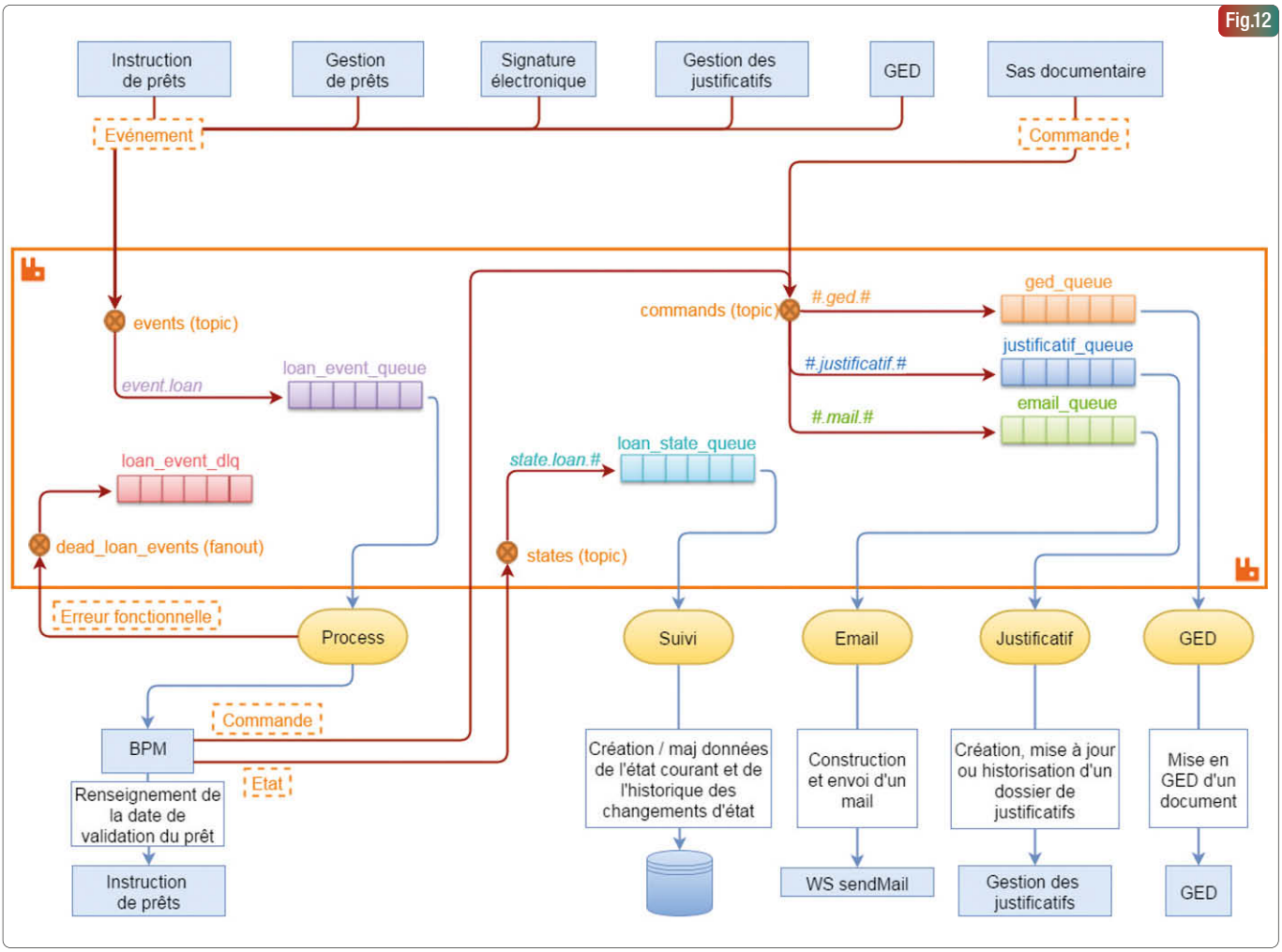
De la même façon pour rendre générique la mise en ged de documents une file ged_queue a été créée avec un Consumer de mise en ged.

Une fois ces nouvelles files en place, à chaque étape du processus, on a la possibilité de déclencher une ou plusieurs actions (envoi de mail, mise en ged, changement d'état, historisation, etc.) par des envois de messages sur un exchange de type topic bindé aux files sur des patterns (#.ged.#, #.email.#), avec une clé de routage qui va porter le choix des actions voulues. Si on veut faire une mise en ged ET un envoi de mail, la clé de routage devra matcher #.ged.# et #.email.# , exemple de clé : whatever.ged.email.anything

Pour conclure, nous utilisons RabbitMQ en production depuis le mois de janvier sans rencontrer de problèmes majeurs pour le moment, même s'il reste du travail d'amélioration et de fiabilisation à mettre en œuvre. Il couvre parfaitement notre besoin en termes de performances puisque, actuellement, le trafic est modéré (une centaine de messages / jour), mais aussi en termes de traitements fonctionnels grâce à ses différents types de routages. Pour notre cas d'usage, nous sommes particulièrement vigilants sur la perte de messages.

En effet, certains messages envoyés correspondant à des étapes de process, un message perdu, aurait pour conséquence de bloquer le process, et RabbitMQ a répondu parfaitement à nos attentes.

Fig.12



© Casden

Kafka, a walking skeleton

Vous avez entendu parler de Kafka et vous voulez en savoir plus ? En ingénierie logicielle un walking skeleton est un système minimal qui fonctionne de bout en bout, c'est le but des pages qui suivent. J'espère que cette introduction vous sera utile pour démarrer une première implémentation.

Brice Dutheil
Freelance

Kafka encore un autre broker de message ?

Ce projet vient tout droit de LinkedIn, lorsque des ingénieurs ont identifié des problèmes de performance globaux dans leurs traitements de données. Fin des années 2000, BigTable, Hadoop et MapReduce sont des buzzwords de l'IT. Cependant, les approches disponible reposent sur des traitements de type batch et ne permettent donc pas de traiter de gros volumes de données en temps réel. Afin de correspondre aux besoins de l'environnement de production de LinkedIn, les ingénieurs avaient besoin d'une solution technique répondant aux critères suivants :

- Haute Performance ;
- Durabilité des messages ;
- Scalabilité ;
- Résilience aux pannes ;
- Simplicité.

Cette solution baptisée Kafka, LinkedIn en a fait un projet Open Source.

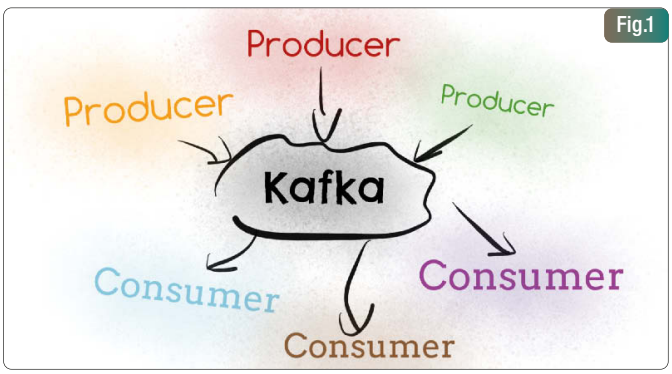
En termes de performance, Kafka devance de très loin tous les autres acteurs avec une charge encaissée de plus de 100k messages/secondes (certaines configurations de LinkedIn montent jusqu'à 2 millions/secondes). Kafka "stocke" durablement les messages, c'est-à-dire qu'ils ne sont pas effacés à la consommation ; la rétention est réglable.

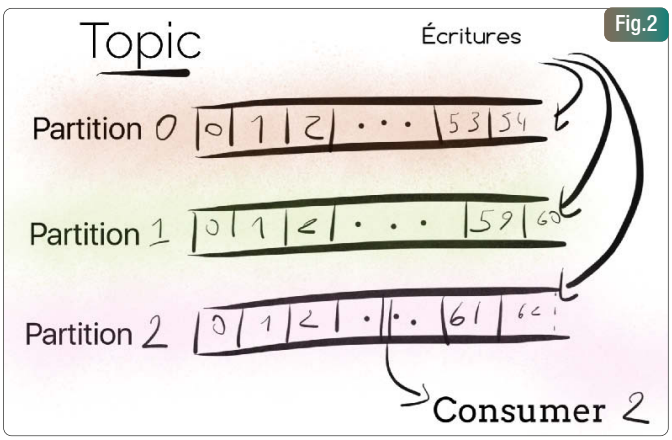
Un critère important pour un site comme LinkedIn est de pouvoir monter en charge en fonction du besoin. Pour cette raison, la topologie d'un cluster Kafka peut être modifiée : ceci permet de scaler horizontalement. Pour anticiper les pannes ou les interventions sur les machines, Kafka réplique ses données sur plusieurs serveurs, ce qui le rend relativement résilient aux pannes. Enfin Kafka est simple car sa conception se veut basée sur une primitive simple – le log ; son utilisation reste également simple car il ne propose que du queuing et du publish/subscribe.

Comment ça marche

Dans les faits le **log** est utilisé par de nombreux systèmes, par exemple dans les bases de données, cependant ces produits n'exposent pas ce log à l'utilisateur, tout au plus à l'administrateur système. En revanche avec Kafka, le log est exposé à l'utilisateur et il est au cœur de sa conception. Ces logs constituent l'abstraction de base : le *topic*. Les messages sont **publiés** dans un topic par des **producers** (producteurs) et les **consumers** (consommateurs) souscrivent à ce même topic pour traiter ces messages. Schématiquement une infrastructure utilisant Kafka peut être représentée de cette façon : Fig.1.

Techniquement un topic est un log partitionné auquel les messages sont ajoutés continuellement, et ceci, de manière ordonnée (dans une même partition). Pourquoi est-ce partitionné ? Avec Kafka il y a une relation forte entre le nombre de partitions et le nombre de consommateurs ; augmenter le nombre de partitions revient à augmenter le parallélisme des consommateurs. Fig.2.





Les producers et les consumers

Kafka fournit une abstraction appelée **consumer group** (groupe de consommateurs) qui généralise à la fois les deux patterns queuing et publish/subscribe. Chaque consommateur s'attribue un groupe, et, chaque message sera délivré à une seule instance dans le groupe.

- Tous les consumers ont le même groupe, c'est du queuing, ou la charge est répartie sur chaque consommateur.
- Les consumers ont un groupe différent, dans ce cas c'est du publish/subscribe.

À noter sur le parallélisme, au sein d'un même consumer group, pour N partitions il ne peut y avoir que N consumers actifs. Ceci dit, il peut y avoir plusieurs consumer groups abonnés à un même topic (et donc sur les même partitions). Fig.3.

À propos de la réplication

Chaque partition d'un topic est répliquée suivant la configuration donnée. Parmi tous les réplicas (les serveurs qui ont une copie des partitions d'un topic) :

- L'un sera le **leader**, c'est celui-ci qui va traiter les demandes de lectures / écritures
- Les autres sont des **followers** (suiveurs), leur but est de répliquer passivement les partitions.

Si le leader s'arrête, un des suiveurs prend le relais et devient le nouveau leader pour les partitions affectées.

À noter que chaque serveur agit en tant que leader pour certaines partitions et en tant que follower pour d'autres partitions, ce qui permet de garder un cluster équilibré. Fig.4. En rouge ce sont les partitions dont un des

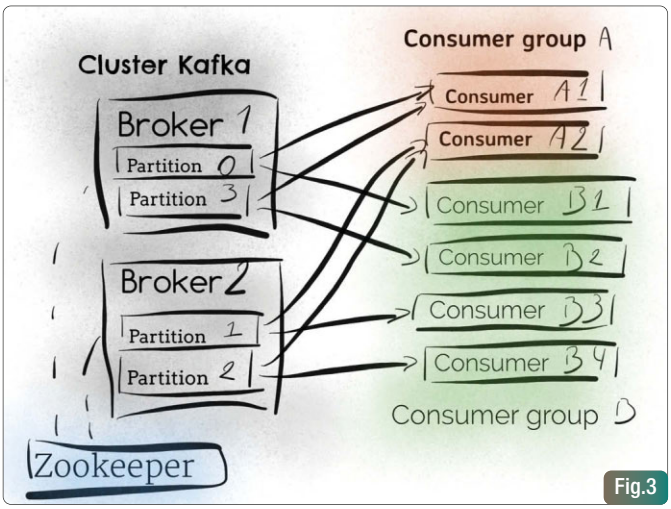


Fig.3

serveur Kafka est le leader, c'est donc la partition en rouge qui recevra les écritures des producers. En bleu sont dessinées les partitions répliquées.
Pour cette introduction, ces quelques explications devraient suffire. Il y a bien sûr davantage de ressources à la fois sur le site officiel de Kafka (kafka.apache.org) et sur le site de la société commerciale de Kafka (http://confluent.io).

Mettre en place Kafka

Cet article se base sur la version 0.9.0.x. La sous-version Scala n'a pas vraiment d'importance si vous ne faites pas de Scala. La version utilisant Scala 2.11 est d'ailleurs recommandée.

Démarrer avec 1 seule instance

Kafka utilise Zookeeper pour stocker ses métadonnées. Il faut donc le lancer en premier. Dans un terminal à part, lancez la commande :

```
# cd kafka_2.11-0.9.0.1
# ./bin/zookeeper-server-start.sh ./config/zookeeper.properties
```

Pour vérifier son état de marche, on peut envoyer le **mot de 4 lettres** stat sur le port 2181, le port de communication client.

```
# { echo stat; sleep 0.1 } | telnet 127.0.0.1 2181
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Zookeeper version: 3.4.6-1569965, built on 02/20/2014 09:09 GMT
Clients:
 /127.0.0.1:52946[0](queued=0,recved=1,sent=0)

Latency min/avg/max: 0/0/0
Received: 1
Sent: 0
Connections: 1
Outstanding: 0
Zxid: 0x0
Mode: standalone
Node count: 4
Connection closed by foreign host.
```

Si vous avez ces statistiques, alors le serveur Zookeeper tourne correctement. S'il y a un autre message, alors le cluster Zookeeper n'est pas dans la bonne configuration et n'est donc pas opérationnel.
Ensuite démarrons une instance Kafka. Il faut donner les valeurs aux propriétés suivante dans le fichier config/server.properties :

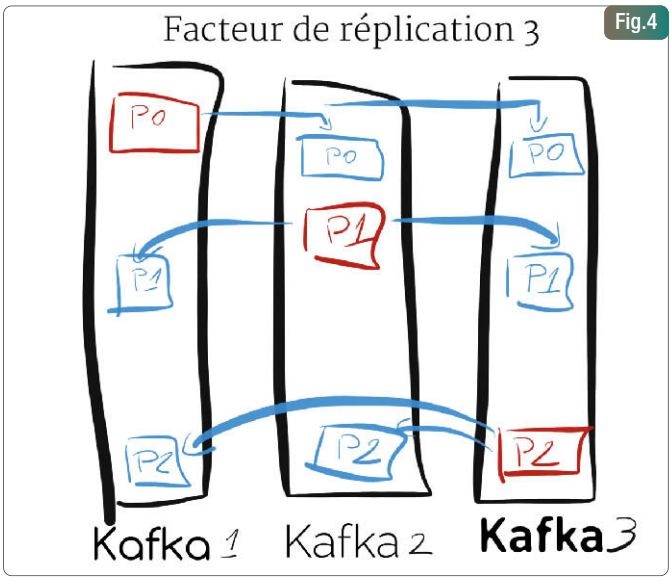


Fig.4

```
broker.id=1
port=9092
logs.dirs=/tmp/kafka-logs-1
zookeeper.connect=localhost:2181
```

Enfin lancer la première instance.

```
# ./bin/kafka-server-start.sh ./config/server.properties
```

Ensuite nous pouvons créer le premier topic, c'est en ligne de commande que ça se passe :

```
# ./bin/kafka-topics.sh --create --topic bier-bar --partition 3 --replication-factor 1 --zookeeper
 localhost:2181
```

Les métadonnées du topic sont donc créées sur le cluster Zookeeper. Notez les paramètres obligatoires que sont le nombre de partitions et le facteur de réplication. Il est possible de produire et de consommer des messages avec les outils en ligne de commande inclus dans la distribution (kafka-console-producer.sh et kafka-console-consumer.sh). Ceci dit cet article se concentre plutôt sur le code Java.

Au niveau code

Avec la version 0.9, il faut importer la dépendance org.apache.kafka:kafka-clients:0.9.0.1, elle est nécessaire pour utiliser l'API java de Kafka à la fois pour produire et consommer des messages.
En premier lieu il faut inclure au minimum trois propriétés au code pour créer un producteur de données :

- La liste des serveurs kafka ;
- Les classes de sérialisation pour la clé et la valeur.

Ensuite il suffit de produire des messages sur le topic créé.

```
public static void main(String[] args) throws InterruptedException {
  Properties props = new Properties();
  props.put(BOOTSTRAP_SERVERS_CONFIG, "localhost:9092");
  props.put(KEY_SERIALIZER_CLASS_CONFIG, StringSerializer.class.getName());
  props.put(VALUE_SERIALIZER_CLASS_CONFIG, StringSerializer.class.getName());
  KafkaProducer<String, String> producer = new KafkaProducer<>(props);
  Runtime.getRuntime().addShutdownHook(new Thread() {
    public void run() {
      System.out.println("Barman shutting down ...");
      // always close the producer
      // timeout to allow the producer to send the data to the broker
```

```
      producer.close(1000, MILLISECONDS);
    }
  });
  while (true) {
    producer.send(new ProducerRecord<>("bier-bar",
                  String.format("Bier bought at '%s'", LocalTime.now())));
    SECONDS.sleep(1);
  }
}
```

Cette classe enverra des messages simples, qu'un consommateur traitera dans un autre process de l'infrastructure. Le code minimal du consommateur suit ci-après ; la configuration minimale reste la même, tout comme les adresses des brokers Kafka, les classes de désérialisation, et, enfin l'identifiant de groupe. À noter, étant donné que ce code utilise la nouvelle API Java apparue en version 0.9, il n'y a pas besoin de gérer la connexion à Zookeeper.

```
public static void main(String[] args) {
  Properties props = new Properties();
  props.put(BOOTSTRAP_SERVERS_CONFIG, "localhost:9092");
  props.put(GROUP_ID_CONFIG, "barfly-group");
  props.put(KEY_DESERIALIZER_CLASS_CONFIG, StringDeserializer.class.getName());
  props.put(VALUE_DESERIALIZER_CLASS_CONFIG, StringDeserializer.class.getName());
  KafkaConsumer<String, String> consumer = new KafkaConsumer<>(props);
  consumer.subscribe(Collections.singletonList("bier-bar"));
  try {
    for(int i = 0; i < 10; i++) {
      ConsumerRecords<String, String> records = consumer.poll(1000);
      for (ConsumerRecord<String, String> record : records)
        System.out.println(record.offset() + ": " + record.value());
    }
  } finally {
    System.out.println("Barfly shutting down ...");
    consumer.close(); // always close the consumer
  }
}
```

Dans cette configuration, le consommateur ne fera que 10 opérations de polling, après quoi il s'arrêtera. En le lançant une seconde fois, seuls les nouveaux messages seront dépilés. Ce comportement est modifiable, ainsi plutôt que de reprendre à la position du dernier message consommé, il est possible de choisir la position, et donc, par exemple, de reprendre un topic depuis le début.

Le code présenté ici est simple, il est facile de le réutiliser pour jouer sur le parallélisme ou les topologies de consommateurs avec les **groupes de consommateurs**. Typiquement au sein d'un consumer group, une partition ne sera assignée qu'à un seul consumer. Pour un besoin métier différent il faudra qu'un consommateur ait un autre group.id.

À noter qu'un consommateur n'est pas thread-safe, il est prévu pour tourner dans un thread qui lui est dédié. Ainsi on peut imaginer soumettre des tâches de consommation de message sur un ExecutorService :

```
executorService.submit(new Barfly());
```

Cette tâche n'aurait qu'à étendre la classe Runnable :

```
public class Barfly implements Runnable {
  // ...
  @Override
  public void run() {
    System.out.printf("Starts consumer %s%n", uuid);
    consumer.subscribe(Collections.singletonList("bier-bar"));
    try {
      while (true) {
        ConsumerRecords<String, String> records = consumer.poll(1000);
        for (ConsumerRecord<String, String> record : records) {
          System.out.printf("%d:%s:%s -> %s%n", record.partition(), record.offset(),
uuid, record.value());
        }
      }
    } catch(WakeupException ignored) {
    } finally {
      System.out.println(String.format("Barfly '%s' shutting down ...", uuid));
      consumer.close(); // always close the consumer
    }
  }

  public void shutdown() {
    consumer.wakeup();
  }
}
```

Pour stopper proprement le thread du consumer, il est possible d'invoquer consumer.wakeup(). Dans cet exemple une WakeupException sera levée afin d'interrompre la boucle infinie, ce qui permet d'atteindre le bloc finally pour clore le consumer.

La sérialisation / désérialisation

Kafka ne gère que des octets, il n'a pas connaissance du contenu réel d'un message. C'est pour cette raison que les producteurs et les consommateurs ont besoin d'avoir des classes qui font la traduction entre la représentation Java et un tableau d'octets.

Par exemple StringSerializer doit juste transformer un String en tableau d'octets. Coté désérialisation, c'est le travail inverse qui est fait.

Sur des messages simples, une chaîne de caractères, utiliser le StringSerializer peut suffire, en revanche, s'il s'agit de faire passer des grappes d'objets plus complexes, alors il faut passer à autre chose.

Pour cet article imaginons que les messages doivent être sérialisés en JSON, il est possible d'écrire ce sérializeur JSON, de la façon suivante avec les librairies du projet Jackson.

```
public class JsonSerializer<T> implements Serializer<T> {
  private ObjectMapper objectMapper = new ObjectMapper().setVisibility(Property
Accessor.FIELD, Visibility.NON_PRIVATE).registerModules(new Jdk8Module(), new JavaTime
Module());
  // ...

  @Override
  public byte[] serialize(String topic, T data) {
    try {
      return objectMapper.writeValueAsBytes(data);
    } catch (JsonProcessingException e) {
      e.printStackTrace();
      throw new UncheckedIOException(e);
    }
  }
  // ...
}
```

Ce qui donnerait le message suivant :

```
0:1400 -> {"message":"Bier served","bierName":"Gallia","timestamp":[16,0,6,509000000]}
```

Notez que le message est bien sous forme de string. Le premier Barfly consommateur pourra lire ce message comme une String. En revanche pour transformer ce message dans un objet Java, il faut faire la même chose avec un désérialiseur, le code de ce désérialiseur n'en sera pas plus complexe.

Attention ceci dit JSON ou autre représentation non binaire n'est pas optimale. Si votre format de message doit être amené à évoluer, il sera plus judicieux d'utiliser Avro ou un équivalent. Avro ou d'autres solutions peuvent également avoir de l'attrait en termes d'empreinte mémoire et de performance.

Dans une infrastructure qui fait communiquer des applications entre elles, que ce soit par HTTP, avec Kafka ou une autre technologie, il faut tester fortement les outillages de sérialisation, penser aux montées de version des messages, ainsi qu'au coût que la sérialisation peut engendrer en termes de performance.

Focus sur Zookeeper

Jusqu'ici Zookeeper est assez peu évoqué. C'est un composant nécessaire de Kafka. En version 0.9 de Kafka il a deux rôles très importants :

- "Service Discovery" des instances Kafka ;
- Stockage des métadonnées des topics.

Afin de permettre un rééquilibrage de la charge des brokers, ceux-ci doivent se connaître, ils utilisent pour cela Zookeeper en tant que registre.

Comme on l'a vu plus haut, l'outillage d'administration crée un topic sur Zookeeper, car il agit là aussi comme registre de métadonnées.

Il est nécessaire qu'il soit démarré en premier et qu'il soit arrêté après Kafka. Pour ces raisons Zookeeper représente un élément sensible de l'infrastructure. Il faut donc lui accorder le plus grand soin. Connaître ce produit est plus qu'un simple bonus.

Mon avis personnel est de prendre la distribution officielle de Zookeeper plutôt que de prendre le Zookeeper livré dans la distribution Kafka, la distribution Zookeeper contient des outils additionnels intéressants pour les opérations de gestion du cluster Zookeeper.

En production, où la haute disponibilité est un prérequis, on pensera à affecter au minimum trois machines dédiées pour chacune des instances du cluster Zookeeper. L'emplacement de ces machines sera de préférence dans des armoires différentes. Cette utilisation des ressources matérielles autorisera des interventions, par exemple sur les alimentations électriques.

Sur un petit SI, où une durée d'indisponibilité est autorisée, une seule instance Zookeeper sur une machine dédiée suffira.

Dans tous les cas il faut éviter de faire tourner les brokers Kafka sur les mêmes machines que Zookeeper. Si une machine hébergeant à la fois Zookeeper et Kafka tombe, alors cela peut mener à la corruption de données de topologie du cluster Kafka et donc interrompre le service.

À noter que Zookeeper n'est pas encore équipé d'auto-discovery ni d'ajout d'instance à la volée. Le travail est en cours sur la version 3.5, il faudra donc bien dimensionner son cluster Zookeeper. Aujourd'hui en version 3.4.x il faut lancer une procédure de rolling restart, en ajoutant dans la configuration une nouvelle instance à la fois.

En production

Kafka peut manquer de maturité, mais l'outil est considéré production-ready. Il fonctionne très bien en environnement de production, des noms comme Linkedin, Twitter, Uber, Netflix, Spotify le prouvent chaque jour.

Si nécéssaire la société Confluent vend du support technique, et finance le développement de Kafka ; la société a été fondée par les ingénieurs de LinkedIn à l'origine du projet.

Pour amener Kafka en production, l'idéal est de travailler sur une fonctionnalité simple. Ainsi il est possible d'avoir des retours techniques grâce au trafic réel de production, et des retours humains par l'exploitant. Ce code devra bien sûr être activable à la demande (feature toggle) afin de ne pas compromettre la plateforme de production.

Les points d'attention

Comme pour tous les produits, il y a des points forts et des éléments à améliorer, même si Kafka tourne sur certaines des plus grosses productions du monde, il n'en reste pas moins jeune sur certains aspects :

- En phase de développement, l'outillage de test est assez maigre ;
- Comme tous les brokers, il faut que l'équipe comprenne les contraintes et la sémantique qui entourent le messaging, en particulier **at-least-once** et l'**ordre des messages** ;
- La configuration de Kafka est simple, mais si la configuration des machines est plus avancée (plusieurs interfaces réseau, noms de domaines internes, etc.) alors la documentation commence à montrer ses lacunes ;
- En environnement d'exploitation, l'expérience pourrait être améliorée. Typiquement le **rééquilibrage** d'un cluster est une opération qui pourrait être simplifiée ;
- Kafka ne vient pas avec une interface graphique d'administration, il existe le projet KafkaManager maintenu par des gens de Yahoo, mais celui-ci n'a pas exactement le même cycle de vie que Kafka.

Pour finir

Kafka est encore jeune, mais offre une alternative intéressante et performante aux systèmes actuels. Réaliser un comparatif avec d'autres acteurs du marché n'est pas forcément pertinent, il faut avant tout identifier ses besoins. Si le besoin d'un broker avec des fonctionnalités plus avancées comme le support des transactions et les garanties d'ordre, alors Kafka n'est peut-être pas le bon outil. Par contre si les critères primordiaux sont de pouvoir encaisser une forte charge, de pouvoir ajouter des instances suivant le besoin, ou de laisser persister durablement les messages, alors Kafka est une option sérieuse à envisager. ⊞

Sources

http://alistair.cockburn.us/Walking+skeleton
http://kafka.apache.org/
https://engineering.linkedin.com/kafka/benchmarking-apache-kafka-2-million-writes-second-three-cheap-machines
https://www.quora.com/What-are-the-differences-between-Apache-Kafka-and-RabbitMQ?share=1&redirected_qid=625566
http://www.confluent.io
http://www.confluent.io/blog/using-logs-to-build-a-solid-data-infrastructure-or-why-dual-writes-are-a-bad-idea/
http://www.confluent.io/blog/how-to-choose-the-number-of-topicspartitions-in-a-kafka-cluster/
https://zookeeper.apache.org/doc/trunk/zookeeperAdmin.html#The+Four+Letter+Words
http://docs.confluent.io/2.0.1/kafka/post-deployment.html

Code

https://github.com/bric3/articles-kafka-walking-skeleton

Tips pour concevoir ses emailing HTML

Notre travail au quotidien ne consiste pas toujours à devoir monter une stack analytics "from scratch", a réaliser une application Web avec Java 8 et Singular, ou à se former au dernier langage à la mode ; par moments nous devons réaliser certaines tâches, qui ne paient pas de mine à première vue, mais qui sont en réalité assez "tricky" et qui vont vous permettre de monter en compétence sur les technologies HTML et CSS.

Aurélie Vache
Développeuse Web Full-Stack chez **atchikservices**
Duchess France Leader
@aurelievache

Un e-mail, comme toutes les pages Web, est codé en HTML. Néanmoins il ne faut pas partir tête baissée et coder comme on le fait un site Web avec des floats, des *margins*, des *paddings* ainsi que des *divs*. Le code HTML d'un e-mail doit respecter de nombreuses règles pour que ce dernier apparaisse correctement et soit compatible quel que soit le logiciel utilisé pour le lire et quel que soit le device de l'utilisateur. Un logiciel de messagerie intègre un moteur de rendu HTML propriétaire alors qu'un webmail interprétera à sa manière le code et modifiera certaines balises, attributs HTML et styles CSS.

Un pro des CSS inline tu seras

On vous a demandé année après année de ne plus insérer les balises de style CSS en inline, directement dans vos balises HTML, je vais vous conseiller l'inverse pour l'emailing. Il est recommandé d'insérer le style CSS en inline. Certains webmails ne tiennent pas compte du code CSS placé dans la balise *<head>* tandis que d'autres ne l'utiliseront pas si les images sont désactivées. Il est conseillé de favoriser le style *longhand* plutôt que le style *shorthand* qui n'est pas supporté par tous les clients de messagerie. Il ne faut pas écrire :

```
<span style="font: 19px bold arial, times, serif">
```

Mais :

```
<span style="font-size: 19px; font-weight: bold; font-family: 'Arial', 'times', 'sans-serif'">.
```

Même remarque pour le code couleur en hexadécimal, au lieu d'utiliser *#000* il faudra plutôt écrire *#000000*.

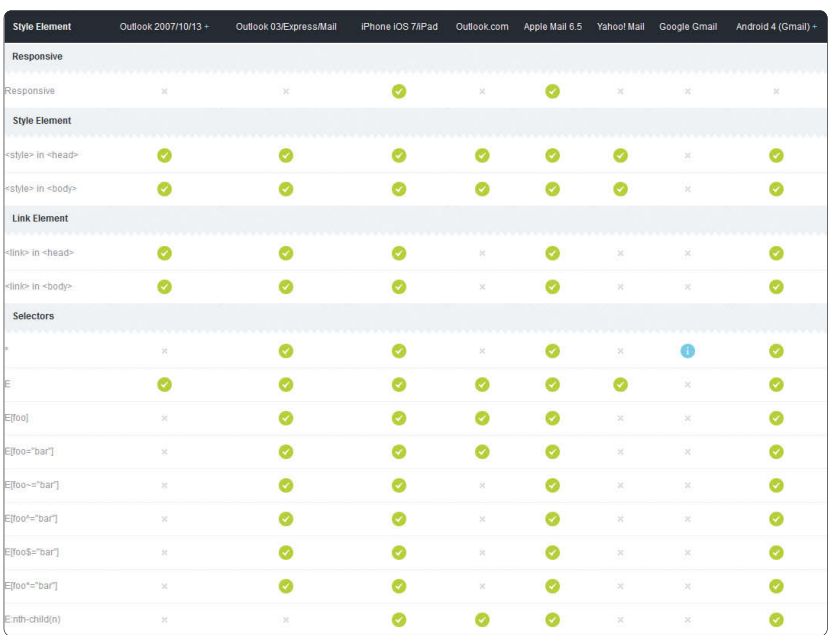


Fig.1

En règle générale dans ce style d'exercice, il faut être le plus verbeux et descriptif possible. Contrairement à ce que vous avez l'habitude de faire, il est déconseillé d'utiliser le CSS pour le positionnement, son support étant très limité ; il en résultera sûrement une mise en page cassée pour la plupart de vos destinataires.

Tous les éléments CSS ne sont pas supportés par tous les logiciels de mail, il est ainsi conseillé de vérifier leur support sur le site de Campaign Monitor (https://www.campaignmonitor.com/css/b/). Ce guide CSS liste quels styles ainsi que leur support parmi les 18 principaux clients de messagerie. Fig.1. De manière générale il faudra utiliser la version 2 des CSS et non la 3 qui n'est pas supportée par les webmails.

Mise en page

Afin que l'e-mail s'affiche correctement sur un maximum de devices, il est conseillé de ne pas dépasser **600px** de largeur. Pour maximiser la zone de prévisualisation, il est conseillé d'ajouter dans votre balise *<body>* les attributs CSS suivants :

```
bottommargin="0" topmargin="0" rightmargin="0" leftmargin="0" margin
height="0" marginwidth="0".
```

Et pour éviter aux liens de perdre leurs couleurs :

```
alink="#000000" vlink="#000000" link="#000000" bgcolor="#000000".
```

Les tableaux tu utiliseras

Oui je sais, cela fait une dizaine d'années que vous n'avez plus utilisé de balises *<table>* en HTML mais il va falloir oublier les * * si vous voulez que votre e-mail soit compatible avec la majorité des logiciels de messageries ainsi que les webmail. Si vous êtes un développeur Web Junior, même sentence, il va falloir comprendre comment fonctionnent les logiciels de mails et avoir de la patience :-). Fig.2. Lorsque l'on créé son tableau principal, il est conseillé de lui attribuer une largeur maximale de 100% :

```
<table width="100%" border="0" cellspacing="0px" cellpadding="0px">
```

Pour chaque tableau, il est recommandé de renseigner la largeur (*width)*, l'espace entre les cellules (*cellspacing)*, l'espace entre le contenu des cellules et la bordure (*cellpadding)*.
Pour gagner en précision, il est recommandé de définir la hauteur de chaque cellule. Il est préférable d'utiliser des cellules vides dont la hauteur est définie plutôt que d'utiliser des *padding et des margin*.
Il est conseillé d'ajouter l'attribut *align* dans chaque cellule car l'alignement par défaut diffère selon les clients mail : *<td align="right">*.
Il est déconseillé d'utiliser la balise *tbody*, celle-ci accroît le score "spam" de l'e-mail ce qui pourra causer sa non réception.
Même sentence pour la balise *rawspan*, celle-ci n'est pas lue par tous les clients mail contrairement à *colspan* qui l'est majoritairement.

Aux images tu feras attention et tu styliseras

Sur bon nombre de logiciels, les images seront bloquées. Donc n'utilisez pas les images pour des contenus importants comme les titres et les liens.
Tout comme pour le SEO, il est conseillé de définir un texte alternatif aux images (attribut *alt*) afin d'afficher du texte si une image ne s'affichait pas.
Petit tips : il est possible de donner un style à ce texte alternatif, mais attention ce ne sera pas compatible avec tous les logiciels de messagerie :

```
<img src="http://www.programmez.com/sites/all/themes/programmez/images/logo.png"
alt="Programmez! magazine" style="font-size: 16px; font-weight: bold; color: #555555;" />
```

Il faut spécifier les dimensions exactes de vos images grâce aux attributs *width* et *height* dans la balise *img* afin qu'elles puissent s'afficher à la bonne taille et que les images vides qui servent à votre mise en page ne soient pas étirées, détruisant ainsi votre mise en page.
Vous aimez les images de background, oubliez-les également dans un e-mail car elles ne fonctionnent pas avec tous les logiciels de mail. Si vous voulez néanmoins en utiliser, pensez à setter une couleur de fond en plus de l'image afin que le client mail puisse au moins afficher la couleur de fond. Il est conseillé d'ajouter la propriété *"display:block"* afin d'éviter l'apparition d'un trait de séparation sous vos images sur les clients Gmail et Hotmail notamment, et d'ajouter la propriété *"border: 0px;"* pour ne pas avoir une bordure qui s'affiche autour des images.
Une bonne pratique est de redimensionner vos images avec un logiciel de retouche d'image tel que GIMP afin d'utiliser la bonne taille d'image dans l'e-mail.
Une dernière chose concernant les images, le format est important. Lotus Notes ne supporte que les GIF et les JPEG, donc si votre coeur de cible utilise ce logiciel il faudra y faire attention.

Version HTML

Parce que votre e-mail peut mal s'afficher pour votre utilisateur final, il est vivement recommandé de proposer à vos lecteurs d'accéder à une version Web de l'e-mail. Le lien vers la version Web doit être placé dans l'entête de votre e-mail.

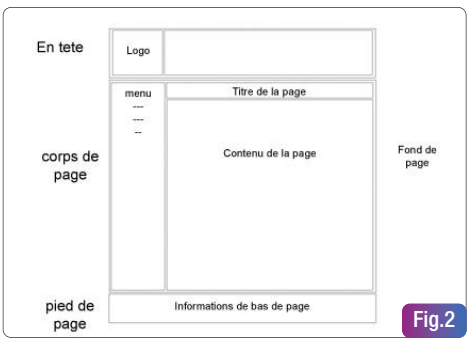


Fig.2

Tests

Avant d'envoyer votre emailing a plusieurs centaines ou milliers de personnes, une bonne pratique est d'utiliser des outils comme **Litmus** (https://litmus.com/) pour tester le rendu de votre code HTML sur plusieurs logiciels de messagerie et webmails. La version gratuite de cet outil permet de visualiser l'e-mail sur 3 plateformes différentes : Outlook 2013, Gmail et iPhone 6s. Fig.3.
Vous pouvez également tester votre email à la main sur plusieurs logiciels différents ou bien générer l'envoi d'un e-mail en HTML grâce à putsmail (https://putsmail.com).

Le HTML tu formateras correctement

Un HTML correctement formaté augmentera la compatibilité de l'affichage de votre e-mail sur les différents supports et réduira le taux de filtre en tant que spam.
Afin de vérifier cela il suffit d'utiliser le Markup validation service : https://validator.w3.org/.

Les liens tu feras styliser

Vous voulez afficher vos liens en noir, contrairement à notre réflexe, il ne faudra pas lui assigner la couleur noire (*#000000*) mais utiliser ce tips pour que Gmail notamment puisse afficher votre lien en noir : *"color: #000001".*
Une autre astuce à connaître si vous ne voulez pas que vos lien soient surlignés est de doubler l'attribut *text-decoration* comme ceci : *"text-decoration:none !important; text-decoration:none;"* (afin d'assurer la compatibilité avec Outlook). Et pour feinter certains webmails :

```
<style>
.ii a[href]{
text-decoration: none !important;
}

#yiv8915438996 a:link, #yiv8915438996 span.yiv8915438996MsoHyperlink{
text-decoration: none !important;
}
```

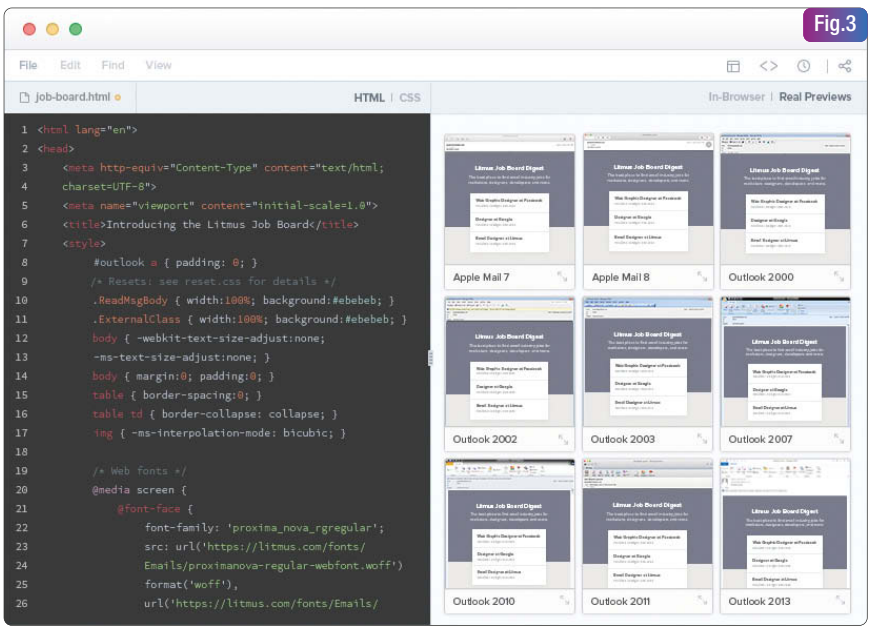


Fig.3

```
#yiv8915438996 a:visited, #yiv8915438996 span.yiv8915438996MsoHyperlinkFollowed{
text-decoration: none !important;
}
</style>
```

Encoding

Il s'agit d'une bonne pratique lorsque vous développez un site Web, il est conseillé de définir l'encoding qui sera utilisé dans l'e-mail :

```
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
```

Doctype

Comme vous le savez, chaque document HTML commence par la déclaration *<!DOCTYPE>*. Le *<!DOCTYPE>* permet de spécifier au navigateur ou au client de messagerie la version de HTML que vous prévoyez d'utiliser. Il y a un débat sur la meilleure version du doctype à utiliser dans les e-mails HTML. Selon les besoins, il est conseillé d'utiliser la version XHTML 1.0 Transitional :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional //EN" "http://www.w3.org/
TR/xhtml1/DTD/xhtml1-transitional.dtd">
ou bien la version 5 de HTML :
<!DOCTYPE html>
```

Je vous conseille de vous renseigner (https://www.campaignmonitor.com/blog/email-marketing/2010/11/correct-doctype-to-use-in-html-email/) et d'effectuer des tests avec plusieurs versions de doctype différent.

Dans les spams tu n'apparaîtras pas

Afin de ne pas apparaître dans les spam, il faut respecter le ratio 60/40, l'e-mail doit comporter le plus de texte possible (60%).

Quid de Flash et de Javascript ?

J'ai failli oublier de vous en parler, mais même si vous le savez déjà, un appel ne fait pas de mal : n'utilisez pas de **Flash** ou de **Javascript** !
Flash est déprécié et non compatible avec la majorité des terminaux mobiles, à la place vous pouvez utiliser la balise *<video>* de HTML5 par exemple qui est compatible avec quelques logiciels. Le Javascript quant à lui, sera soit supprimé, soit un message de sécurité sera affiché chez vos destinataires.

Le pre-header tu spécifieras

Votre e-mail est prêt, il et compatible avec la majorité des webmails et clients mail ? Bravo ! Mais attendez, il vous reste une chose à faire avant de l'envoyer. Afin de mettre toutes les chances de votre côté et faire en sorte que l'utilisateur aura plus de chances de cliquer sur votre e-mail et ne pas le mettre à la poubelle tout de suite, depuis quelques temps la plupart des principaux clients de messagerie incluent un petit extrait au tout début du message HTML, il s'agit du "pre-header".

Généralement il s'agit du premier texte qui existe après la balise *<body>*. Il peut arriver que ce texte ne soit pas pertinent. Pas de soucis il vous suffit d'ajouter le pre-header immédiatement juste après la balise *<body>* comme ceci :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional //EN" "http://www.w3.org
/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
<head>
<title>Mail de test</title>
<meta http-equiv="content-type" content="text/html; charset=utf-8" />
<meta charset="UTF-8" />
<meta content="width=device-width" />
</head>
<body>
  <span style="display: none !important;">
    Ceci est mon texte de preview !
  </span>
  <table class="body">
    <tr>
      <td class="center" align="center" valign="top">
        <center>
          <!-- Email Content -->
        </center>
      </td>
    </tr>
  </table>
</body>
</html>
```

La roue tu ne réinventeras pas

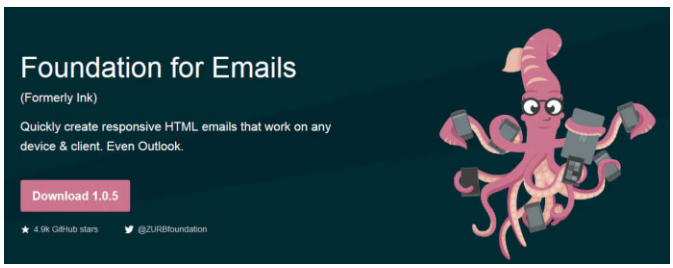
Depuis quelques années des outils sont créés afin de vous faciliter la création d'e-mails en HTML. Il y a notamment *Foundation for Emails* (http://foundation.zurb.com/emails.html), anciennement "Ink", qui fait de plus en plus parler de lui. Il s'agit d'un framework HTML/CSS créé par Zurb conçu pour aider les personnes à créer des e-mails responsive compatibles avec la majorité des webmail et clients mail et tout particulièrement Outlook. Vous pouvez télécharger le framework et créer votre e-mail responsive grâce aux templates mis à disposition. La version 2 du framework est sortie fin Mars 2016). Voici ses nouvelles fonctionnalités :

- **Sass-powered codebase :** en plus de la version CSS de Foundation, une version Sass sera disponible ;
- **Syntaxe Inky HTML :** cette nouvelle version comporte un nouveau langage de templates appelé *Inky*, qui convertit les balises HTML simples comme 'row' et 'column' en de tableaux (table) complexes nécessaire à la construction de composants "email-ready" ;
- **Un workflow de développement d'e-mail tout-en-un :** La stack ZURB Email contient une compilation Saas, la compression d'image, Handlebars templating, ainsi que l'inlining dans un puissant boilerplate.

Au vu des fonctionnalités, je pense qu'un test de *Foundation for Emails 2.0* pourra faire l'objet d'un prochain article.

Conclusion

Concevoir et maintenir un emailing en HTML n'est pas une tâche aisée mais avec de la patience ainsi que des tips, la tâche est réalisable.
Utiliser un framework tel que Foundation for Emails peut permettre de gagner du temps et éviter de s'arracher quelques cheveux mais je vous conseille dans tous les cas de tester, tester et encore tester ;-).



Concevoir une console de jeux vidéo retro avec un Raspberry PI !

Depuis l'année dernière j'ai eu envie de rejouer aux jeux vidéo de mon enfance. J'adore la Nintendo NES et j'ai eu l'agréable surprise de l'avoir à mon anniversaire. Jouer à Super Mario World, The Legend of Zelda, Donkey Kong ... quel régal ! :-)

Aurélie Vache
Développeuse Web Full-Stack chez
atchikservices
Duchess France Leader
@aurelievache

Mon mari est dans le même esprit et aimerait bien rejouer aux jeux de son enfance sur la Super Nintendo et la Nintendo 64, au minimum.
Problème, on ne va pas racheter toutes les consoles Nintendo et Séga pour pouvoir jouer à quelques jeux vidéo. Le budget à avoir serait conséquent, le retrogaming étant à la mode depuis quelques années, acheter une console avec 2 manettes revient entre 70 et 100€, il faut à cela ajouter entre 10 et 20€ par jeu et je ne vous parle pas du prix d'un Zelda qui peut monter jusqu'à 100€ en version boîte...
En ce moment je m'intéresse aux objets connectés, je me suis donc renseignée sur les Arduino et les Raspberry et du coup j'ai eu l'idée de monter ma propre console de jeux vidéo retro avec un petit objet qui fait beaucoup parler de lui : le Raspberry PI et sa version 2 modèle B embarquant un processeur 1Ghz et 4 ports USB.
Quelques semaines après mon achat du PI 2, la version 3 est sortie, mais tout ce que vous verrez dans cet article sera valable avec la version 3 du Raspberry.

Le matériel

Histoire de bien commencer, mon choix s'est porté sur un starter kit acheté chez "The Pi Hut" : Fig.1.
Pour 48.50£ (frais de port inclus) le kit comprend :

- Un Raspberry Pi 2 ;
- Une microSD de classe 10 de 8Go (avec NOOBS pré-installé) ;
- Une alimentation 5V 2A au format européen ;

Fig.3

Fig.1

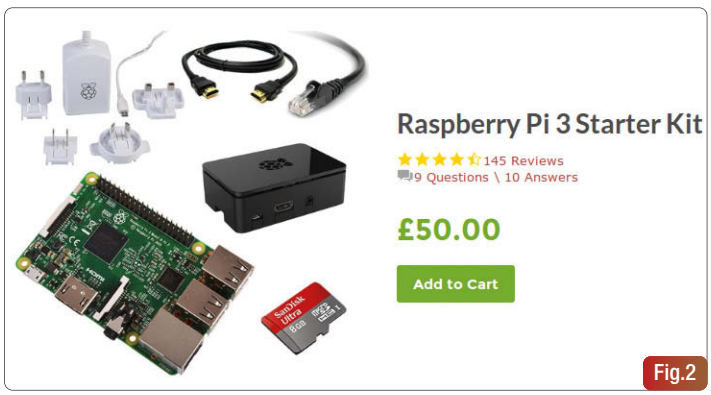


Fig.2

- Un boîtier noir pour protéger le Raspberry (ce sera plus joli pour la console rétro) ;
- Un câble HDMI ;
- Un câble Ethernet (RJ45).

A noter que depuis quelques jours (sic), le même kit est disponible incluant un Raspberry Pi 3 : Fig.2.
A cela il va falloir rajouter quelques manettes USB pour pouvoir avoir la même sensation de jeu qu'avec nos vieilles consoles. J'ai fait le choix de commander, dans un premier temps, des manettes USB reproduisant les manettes de la Super Nintendo. Si tout fonctionne bien je commanderai également des manettes USB Nintendo 64.
Si vous partez sur la distribution RetroPie, il faut également un clavier USB qu'il faudra brancher pour installer l'OS et dans tous les cas, une souris en USB.
Si vous voulez une configuration optimale pour jouer aux anciens jeux vidéo, il faut savoir que la PI communiquera avec vos jeux via la carte microSD, donc plus cette dernière a une capacité de stockage élevée et plus vous pourrez y mettre de jeux. Je conseille donc minimum d'avoir une 16Go, et une carte de classe 10 minimum (pour que la communication entre la Pi et la carte soit optimale). Pour ma part j'ai acheté, en plus du kit de base, une micro SD de 32Go de classe 10. Fig.3.

Le logiciel

En faisant le tour des distributions et des solutions qui existaient je suis tombée sur plusieurs noms qui revenaient sans cesse : **RetroPie** et **RecalBox**.
J'ai décidé d'en faire un comparatif pour que vous ayez toutes les informations avant de vous lancer.

	RetroPie	recalbox / LibRETRO / KODI / ES emulation station
Distribution/Soft	Raspbian/ EmulationStation	
Origine	Allemagne	Français
Facilité d'installation	Il faut installer la distribution ainsi que les logiciels et les configurer (terminal)	Très simple et rapide
Mise à jour	Il faut mettre à jour la version des logiciels à la main (terminal)	Mise à jour directement sur l'interface
Multi-joueurs	2 joueurs	4 joueurs

Si vous voulez bidouiller, optimiser votre configuration, alors **RetroPie** sera faite pour vous ; vous pouvez tout contrôler de bout en bout.
RecalBox est vraiment prête à l'emploi. Vous installez, suivez les installations et c'est déjà l'heure de jouer ! Cette distribution intègre même Kodi, un media center.

Les systèmes supportés

Que ce soit via RetroPie ou la RecalBox, voici la liste des systèmes supportés (à l'heure où j'écris cet article) :

- Arcade
- Arcade Classics
- Nes
- FDS (Family Computer Disk System)
- Super Nintendo
- Master System
- Playstation
- Megadrive
- Game Boy
- Game Boy Color
- Game Boy Advance
- Atari 7800
- Atari 2600
- PC Engine
- Sega SG1000
- MSX 1 / 2 / 2+
- Nintendo 64 (experimental)
- Sega 23x
- Sega CD
- ScummVM
- Game And Watch
- PRBOOM
- Vectrex
- Game Gear
- Virtual Box
- Lynx
- Wonderswan Color
- NeoGeo / NeoGeo CD
- NeoGeo Pocket Color
- SuperGrafx

L'installation

Après avoir effectué un comparatif de ces 2 OS, j'ai choisi de partir sur la recalBoxOS.
Pourquoi ? Pour sa simplicité, sa rapidité d'installation, ses mises à jour fréquentes, sa compatibilité et sa communauté.
Rendez-vous sur https://github.com/digitalLumberjack/recalbox-os/releases pour télécharger la dernière version de la distribution (actuellement il s'agit de la version 4.0.0-beta2). Fig.4.
Bonne nouvelle, la dernière version de la distribution est compatible avec le Raspberry Pi 3 !
Vous pouvez installer recalboxOS sur une carte formatée en FAT32. Utilisez pour cela le logiciel de formatage "SD Formatter" disponible à l'adresse https://www.sdcard.org/downloads/formatter_4/ pour Windows et OSX, ou gparted sous Linux. Ces logiciels sont vraiment optimisés pour les cartes SD, donc il est conseillé de les utiliser à la place du logiciel de base de votre système d'exploitation.
Décompressez le fichier recalboxOS*.zip et copiez tous les fichiers à la base de votre carte SD.
Il faut maintenant insérer la micro SD dans le Raspberry, brancher votre souris, brancher le HDMI et le relier à votre TV, puis en dernier brancher l'alimentation. Votre Raspberry démarre, charge le programme d'installation et lance automatiquement la copie des fichiers.
Vous pouvez changer la langue, qui est l'anglais par défaut, via votre souris ou la touche **L** de votre clavier si vous en avez relié un. Fig.5.
Lorsque l'installation est terminée, vous aurez un écran d'accueil avec une jolie musique de Zelda. Ca y est, votre Recalbox est prête à être utilisée. Pour ma part la langue était encore en anglais donc je l'ai changée via un réglage de la langue accessible via le menu. Fig.6.
Si vous naviguez de console en console et que vous n'apercevez pas la MegaDrive par exemple, pas de panique, l'émulateur est bien configuré mais pour qu'une console apparaisse il faut qu'il y ait au moins un jeu (une rom) dans le répertoire correspondant sur la RecalBox.

La configuration

RecalBox possède un fichier de configuration, vous allez être amené à modifier ce dernier, il se trouve ici : */recalbox/share/system/recalbox.conf*.

Configurer les manettes

Si vous avez une manette USB, branchez-la et appuyez sur le bouton START ce qui fera apparaître une fenêtre **Configure Input**. Il vous suffit d'appuyer sur un bouton de la manette, de suivre les instructions pour configurer votre manette. Le nom des touches est basé sur les boutons de la Super Nintendo donc cela tombe bien !

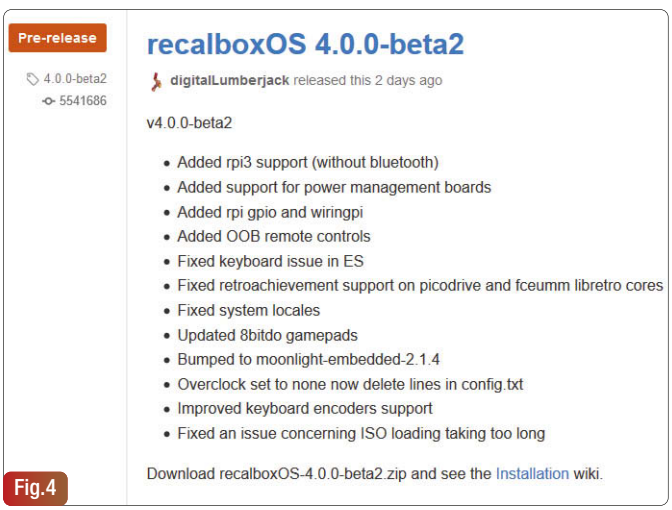


Fig.4

Il est conseillé de configurer le bouton Select pour le “Hotkey” car ce fameux Hotkey vous permettra en cours de jeu de quitter le jeu par exemple en combo avec le bouton Start.
Si vous avez une manette PS3 et une clé Bluetooth, utilisez un câble mini USB pour relier la manette à la recalbox. Patientez quelques secondes pendant que les led clignotent lentement. Débranchez la manette et appuyez sur le bouton **HOME**. Votre manette est configurée !

Contrôles

Dans l’interface :

Voici les commandes :

A → Sélectionner
B → Retour
Y → Favoris
X → Démarrer Kodi
Start → Menu
Select → Options (sur l'écran système, ouvre le menu de redémarrage)
R → Page Suivante
L → Page Précédente

Je ne connaissais pas les commandes de contrôle et du coup cela surprend lorsque le media center se démarre sans crier gare ;-).

Dans un jeu :

Lorsque vous êtes en plein jeu, des commandes spéciales sont disponibles. Appuyez en même temps sur le bouton *Hotkey* et sur un des boutons suivants :

Y → Sauvegarder l'état dans le slot sélectionné
X → Charger l'état dans le slot sélectionné
START → Quitter (très utile !)
B → Menu
UP → Sélectionner le Slot de sauvegarde -1
DOWN → Sélectionner le Slot de sauvegarde +1
L1 → Screenshot
RIGHT → Accélérer le jeu
LEFT → Rembobinage (si l'option est activée)
R2 → Shader suivant
L2 → Shader précédent

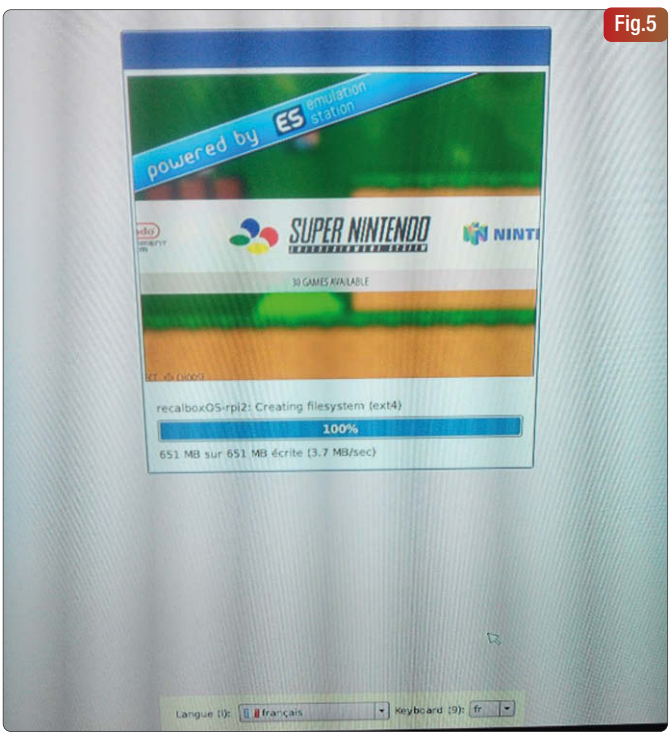


Fig.5

Et les jeux dans tout ça ?

Quelques jeux sont installés par défaut mais si vous voulez jouer aux jeux retro de votre enfance il faut que vous vous procuriez les ROMs.
Pour pouvoir les copier de votre ordinateur à votre Raspberry, il faut tout d’abord que cette dernière soit connectée à Internet/à votre réseau donc branchez un câble ethernet et redémarrez-la, ou bien branchez un dongle Wi-Fi (la PI3 en a déjà un).
Appuyez sur la touche START pour afficher le menu de RecalBox puis naviguez vers “Options réseau”. Vous pourrez voir si la Raspberry est connectée ou pas, et connaître son adresse IP.
Si vous devez configurer le Wi-Fi, saisissez le SSID et la clé de votre réseau WiFi avec un clavier. Une fois les informations validées, le WiFi sera activé et la Pi sera connectée.
Si vous ne pouvez pas entrer certains caractères nécessaires à vos informations réseau, configurez alors votre accès WiFi directement dans le fichier recalbox.conf.
Une fois la RecalBox connectée, celle-ci partage automatiquement des dossiers sur le réseau local. Sur votre ordinateur, tapez l’IP de la RecalBox dans la barre d'adresse de explorer (exemple : **\\192.168.1.30**). Ça y est, vous pouvez accéder aux fichiers de votre console retro et copier vos ROMs dans le répertoire correspondant à l’émulateur de la console d’origine.
Pour mettre à jour la liste des jeux il vous suffit d’aller dans le **Menu**, puis **Options Des Jeux** puis **Mettre A jour les listes des jeux**.

Optimiser/Configurer votre RecalBox

Pour pouvoir jouer de manière optimale à vos jeux rétro préférés quelques tips sont utiles à connaître.

Super Nintendo :

Vous voulez jouer à Bomberman ou Micro Machines à 4 joueurs et cela ne fonctionne pas ? Pas de panique, il y a une petite manipulation pour ça.
Il suffit de modifier le fichier recalbox.conf pour remplacer l'émulateur SNES par snes9x_next

```
# ------------ I - EMULATORS CHOICES ------------ #
## You can override the global configuration here
snes.core=snes9x_next
```

Enregistrez puis redémarrez proprement la box.
Il faut maintenant lancer un jeu, par exemple Super Nintendo. Ouvrez le menu Retroarch avec le bouton Hotkey et le bouton B et modifiez les paramètres suivants :

```
Setting / Configuration / Save Configuration On Exit : ON
Setting / Input / User 2 Device Type / Multitap
Quick menu / Restart Content
```

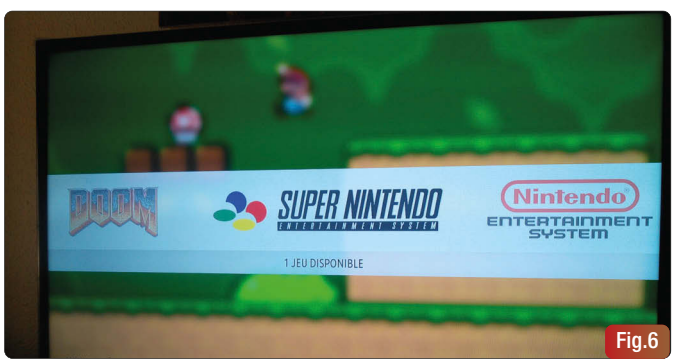


Fig.6

Vous pouvez remettre *Save Configuration On Exit* sur *OFF* après avoir redémarré le jeu. Fig.7.

Game Boy

Vous voulez jouer à 2 joueurs à certains jeux GB ? Pas de soucis :
Modifiez le fichier recalbox.conf pour remplacer l'émulateur Game Boy (et Game Boy couleur) par *tgbdual :*

```
# ------------ I - EMULATORS CHOICES ----------- #
## You can override the global configuration here
gb.core = tgbdual ;GameBoy
gb.ratio=custom
gbc.core = tgbdual ;GameBoyColor
gbc.ratio=custom
```

Enregistrez puis redémarrez votre Recalbox.
Lancez un jeu GameBoy, ouvrez le menu Retroarch avec le bouton Hotkey et le bouton B puis changez les paramètres suivants :

```
Setting / Configuration / Save Configuration On Exit : ON
Quick menu / Core Options / GB Link Enable (restart) : enabled
Quick menu / Core Options / Screen placement (restart) : horizontal
Quick menu / Core Options / Switch player screens : normal ; joueur 1 à gauche et joueur 2 à droite
Quick menu / Core Options / Show player screens : both players
Settings / Video / Aspect Ratio Index : 20:9 (1:1 PAR) ; 100% conforme ratio GB
Quick menu / Restart Content
```

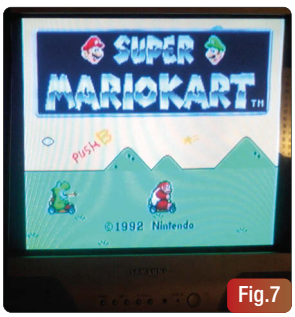


Fig.7

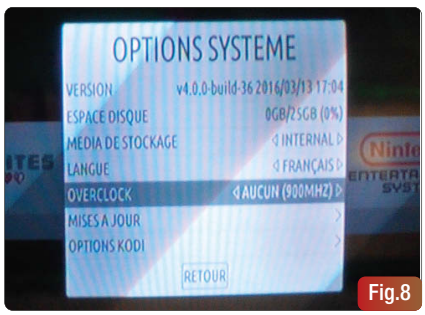


Fig.8

Vous pouvez remettre *Save Configuration On Exit* sur *OFF* après avoir redémarré le jeu.
Attention les jeux mono-joueurs vont aussi être doublés. Pas super pratique, donc vous pouvez configurer le multi-joueur pour l'émulateur gb et mono joueur sur gbc ou inversement.

Nintendo 64 :

J'avais lu qu'il fallait obligatoirement overclocker la "console" pour pouvoir jouer aux jeux Nintendo 64. Après en avoir testé plusieurs, je peux vous dire qu'il n'y a pas de soucis sur Mario Kart 64 par exemple mais il y a des ralentissements sur Goldeneye et Super Mario 64.
Soit vous êtes un Jedi et souhaitez overclocker vous-même votre Pi, pour cela je vous conseille le tuto de Hayden : http://haydenjames.io/raspberry-pi-2-overclock/.
Soit vous voulez vous laisser guider par la RecalBox, allez dans le **Menu**, dans les **Options système** et vous pourrez overclocker votre Pi simplement. Lors de la sélection du changement de la fréquence, la RecalBox redémarrera. Pour pouvoir jouer tranquillement j'ai overclocké jusqu'à 1050 Mhz (900Mhz par défaut). Fig.8.

Kodi :

Si au lieu de démarrer sur la console de jeu retro vous voulez que votre Pi lance par défaut le media center, c'est possible.
Encore une fois, il suffit de modifier *recalbox.conf* (ligne 10)

```
#Show or hide kodi in emulationstation
enable_kodi=1
#Start kodi at launch
kodi_at_start=1
```

Une utilisation de plus à rajouter au Raspberry Pi qui nous permet de rejouer aux jeux de notre enfance. A quelques semaines près j'aurais pu avoir une RecalBox tournant sur une Pi3 mais ce n'est pas grave je ne suis tout de même pas déçue de mon achat.

Les coulisses du numéro

Un numéro de Programmez! se prépare 6-7 semaines avant sa sortie en kiosque. La rédaction ne s'arrête jamais, c'est un éternel recommencement. Chaque mois, la même routine : on réfléchit aux thèmes du prochain numéro, on discute avec les contributeurs, on lit beaucoup, on assiste aux conférences.

En gris : *les commentaires d'Aurélie*
En noir : *les commentaires de François*

Depuis que je dirige Programmez!, j'aime bien avoir une vue sur 4-5 numéros, du moins, avoir les principaux thèmes même si je les change régulièrement. Et un soir, je me suis dit, pourquoi ne pas faire comme certains journaux : le rédacteur en chef d'un jour. J'ai immédiatement pensé à la communauté Duchess. L'idée était simple : que les développeuses soient les rédactrices en chef d'un numéro. Dès fin janvier, j'en ai parlé à Aurélie. Et après quelques échanges, l'idée a fait son chemin pour devenir réalité début février. Peu à peu la mécanique s'est mise en place. Mi-février, une conférence de rédaction s'est tenue pour parler des thèmes et des étapes de fabrication d'un magazine. Le numéro de fin avril a été acté.

Nous avons tout de suite été intéressées pour réaliser ce projet, nous en avons parlé fin janvier, puis, plus en détail lors de notre point mensuel début février et avons très rapidement informé notre communauté.

La fabrication du numéro

J'ai voulu rassurer les développeuses : je serai là dans mon rôle d'éditeur pour coordonner les différents intervenants qui font le magazine : SR, maquettiste, routeur, imprimeur, régleur, messagerie, etc. Et aussi pour déminer les termes barbares comme BAT, fabrication, ours, chemin de fer, folios, etc.

Du 15 février au 20 mars

Suite à notre appel sur notre Google Group, la communauté a été très réactive et nous avons eu une bonne liste d'idées d'articles, de thèmes ainsi que d'auteurs. Nous leur avions également demandé ce qu'elles voudraient dans ce numéro spécial ; toutes les idées étaient bonnes à prendre. Nous avons également demandé à notre réseau. Suite à ces échanges nous avons préparé une liste d'articles que nous avons soumise à François. Suite à la conférence de rédaction, il a fallu informer chaque auteur que leur Idée d'article avait été retenue ou non, creuser avec certains dans quelle direction partir et leur donner les consignes (nb de pages, nb de car, nb d'images max par page, format et contenu du livrable), et, bien entendu, la fameuse deadline. Le suivi a été fait grâce à un Google Sheet partagé. Durant ces 4 semaines

j'ai communiqué avec les auteurs afin de savoir ou est-ce qu'ils en étaient, s'ils avaient besoin d'une aide quelconque ... le but était d'être disponible pour eux, de les aider si le besoin s'en faisait ressentir, et de suivre l'état de chaque article. Il faut essayer d'anticiper les problèmes et au moindre problème, être réactif. J'avais la double casquette de rédactrice en chef et (co-)rédactrice de plusieurs articles. Il fallait également que le Sénat valide un texte donc j'ai essayé de prendre en compte ce temps de validation en l'envoyant le plus tôt possible. La validation a pris 3 semaines !

Dernière semaine de mars

Le rush commence à se faire sentir depuis le 24 mars et a atteint son paroxysme la semaine à cheval entre mars et avril puis la première semaine d'avril. Pour chaque article reçu, je fais une relecture obligatoire, voire plusieurs si j'ai la chance que l'on me partage l'article en cours d'écriture. La dernière semaine, avant la deadline, il y a plusieurs relectures à faire tous les jours, des échanges quotidiennes, des relances pour voir si des auteurs ont besoin d'aide et leur rappeler la deadline. Il faut essayer d'être partout et de penser à tout.

5 avril

Deadline officielle pour recevoir les articles. Mais entre-temps, plusieurs idées d'articles sont venues bouleverser quelques pages du numéro. J'en ai parlé avec Aurélie pour voir la pertinence et trouver les rédactrices...

Pour la petite histoire j'avais donné aux auteurs comme date butoir "Fin Mars" ; au final on a reçu la moitié des articles entre le 28/03 et le 03/04, puis jusqu'au 06/04, donc je pense qu'au final cette petite marge était nécessaire.

Désormais, nous comptons les jours avant le BAT final. Nous y sommes presque !

19 avril

Fraîchement arrivée la veille au soir de Toulouse, ça y est, j'assiste à mon premier BAT ! Je confirme que la/les semaines précédant le BAT sont stressantes et chronophages ^^. C'est un mélange d'anxiété et de joie car tout ne se passe pas exactement comme prévu mais le numéro prend vie petit à petit. Il a fallu vite rebondir et trouver des plans B. Ce projet commun me tient à coeur, j'ai envie que les choses se passent bien, que le numéro plaise et de ne pas décevoir François qui nous a fait confiance en nous donnant les "clés" du magazine.

Après presque 3 semaines de relectures, de maquettes, de changements, nous sommes prêts à terminer le magazine. Comme je dis toujours, un BAT qui se passe sans aucun problème, sans stress, ce n'est pas normal. La semaine qui précède le BAT est la plus difficile et la plus stressante : entre les articles en retard et ceux qui annulent leur(s) contribution(s) au dernier moment, il faut savoir être agile et réagir immédiatement. Surtout, la date de BAT est inamovible car nous devons respecter scrupuleusement le planning de sortie en kiosque, la date de livraison.

20 avril.

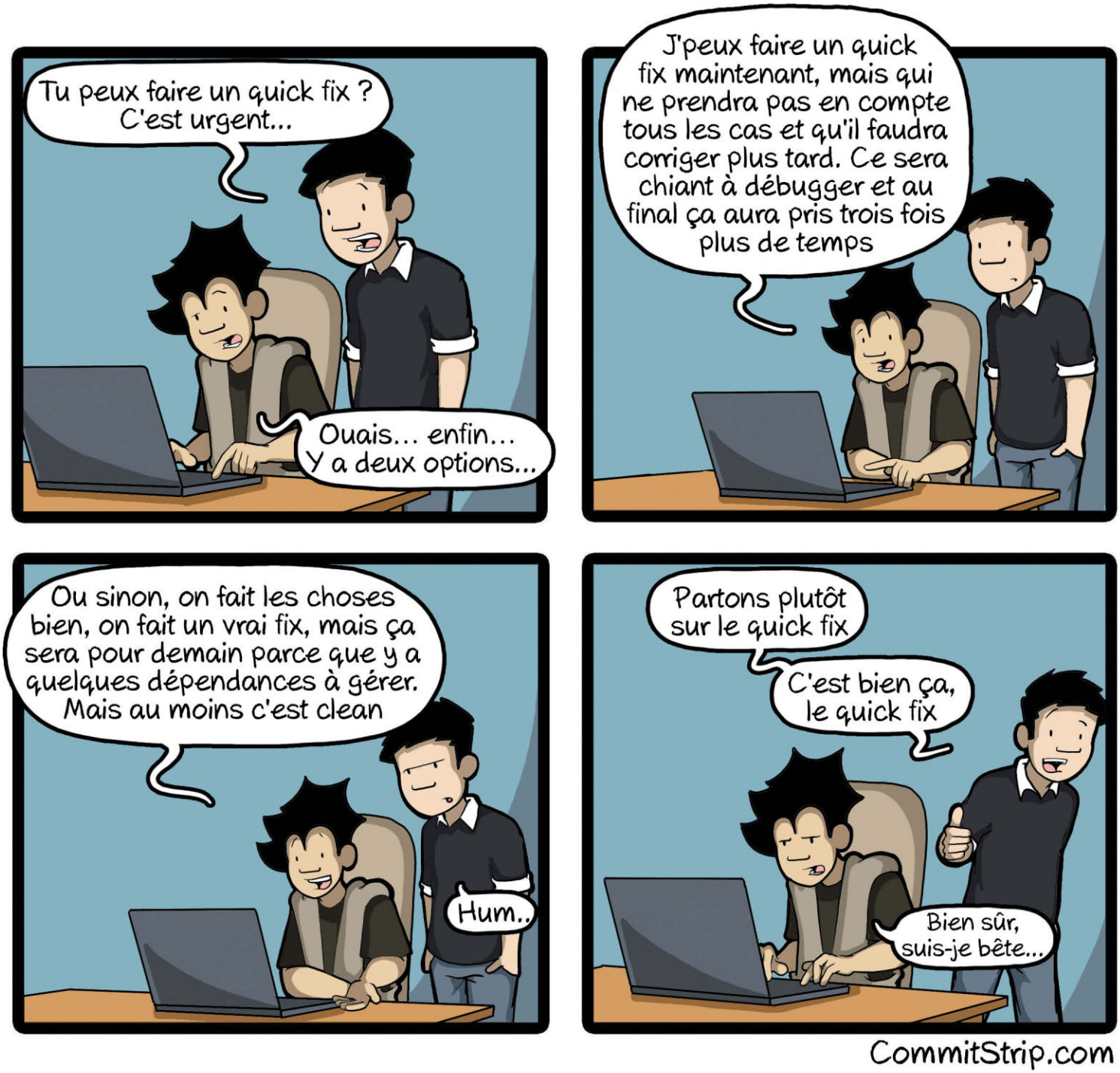
Voilà, tout est validé et les pages sont prêtes à être imprimées. Nous avons quelques jours de "repos" avant d'enchaîner sur le numéro suivant.

Conclusion

Cette expérience m'a appris qu'un rédacteur en chef doit être partout. C'est un travail stressant mais excitant également. Cela a été une expérience très enrichissante. J'ai trouvé qu'il y avait des similitudes avec la gestion (le lead) d'un projet informatique : il y a des délais/une deadline à respecter et à tenir, il faut suivre le projet ainsi que l'équipe, proposer son aide pour que l'équipe avance vers le même objectif, mettre la main à la pâte... En revanche, contrairement à une release que l'on pousserait en production, il n'y a pas de possibilité de rollback, de mise à jour en urgence ou bien de possibilité de patcher ;-). Je tiens à remercier toute la communauté et le réseau des Duchess, c'est grâce à vous que cette version 196 du magazine a pu sortir, vous pouvez en être fières ! :-)

Le défi a été relevé avec brio. ⊞

Quick fix



Abonnement : Service Abonnements PROGRAMMEZ, 4 Rue de Mouchy, 60438 Noailles Cedex. - Tél. : 01 55 56 70 55 - *abonnements.programmez @groupe-gli.com* - Fax : 01 55 56 70 91 - du lundi au jeudi de 9h30 à 12h30 et de 13h30 à 17h00, le vendredi de 9h00 à 12h00 et de 14h00 à 16h30. **Tarifs** abonnement (magazine seul) : 1 an - 11 numéros France métropolitaine : 49 € - Etudiant : 39 € CEE et Suisse : 55,82 € - Algérie, Maroc, Tunisie : 59,89 € Canada : 68,36 € - Tom : 83,65 € - Dom : 66,82 € Autres pays : nous consulter.
PDF : 30 € (Monde Entier) souscription sur www.programmez.com

Une publication **Nefer-IT**
7 avenue Roger Chambonnet
91220 Brétigny sur Orge
redaction@programmez.com
Tél. : 01 60 85 39 96
Directeur de la publication : François Tonic
Rédactrice en chef : Aurélie Vache, Duchess France
Secrétaire de rédaction : Olivier Pavie

Ont collaboré à ce numéro : Laurent Vache, Aurélie Vache, Agnès Crépet, Ellène Dijoux Siber, Mathilde Lemée, Cédric Exbrayat, Anaïs Payet, Henri Lepic, Zhe (Aurore) Li, Alice Barralon, Stéphanie Hertrich, Stéphanie Baltus, Amira Lakhal, Pauline Logna, Brice Dutheil, les Duchess Lyon, Luce Desblancs (traduction), Maxime Caroul, Thomas Lebrun, François Tonic.

Un énorme merci à toute la communauté des Duchess France.

Couverture : © kirstypargeter, © Raspberry Pi

Maquette : Pierre Sandré

Publicité : PC Presse,
Tél.: 01 74 70 16 30, Fax : 01 41 38 29 75
pub@programmez.com

Imprimeur : S.A. Corelio Nevada Printing, 30 allée de la recherche, 1070 Bruxelles, Belgique.

Marketing et promotion des ventes :
Agence BOCONSEIL - Analyse Media Etude

Directeur : Otto BORSCHA oborscha@boconseilame.fr
Responsable titre : Terry MATTARD
Téléphone : 09 67 32 09 34

Contacts
Rédacteur en chef : ftonic@programmez.com
Rédaction : redaction@programmez.com
Webmaster : webmaster@programmez.com
Publicité : pub@programmez.com
Evenements / agenda : redaction@programmez.com

Dépôt légal : à parution - Commission paritaire : 1220K78366 - ISSN : 1627-0908

ikoula
HÉBERGEUR CLOUD

TROPHÉES DUCLOUD by EuroCloud
MEILLEUR SERVICE CLOUD D'INFRASTRUCTURE 2015

LE CLOUD GAULOIS, UNE RÉALITÉ ! VENEZ TESTER SA PUISSANCE

EXPRESS HOSTING

Cloud Public
Serveur Virtuel
Serveur Dédié
Nom de domaine
Hébergement Web

sales@ikoula.com
01 84 01 02 66
express.ikoula.com

ENTERPRISE SERVICES

Cloud Privé
Infogérance
PRA/PCA
Haute disponibilité
Datacenter

sales-ies@ikoula.com
01 78 76 35 58
ies.ikoula.com

EX10

Cloud Hybride
Exchange
Lync
Sharepoint
Plateforme Collaborative

sales@ex10.biz
01 84 01 02 53
www.ex10.biz

Openska

Formations

Web & Open Source

www.openska.com