PROGRAMMEZ.COM

PROGRAMMEZ!

Mensuel n°184 - Avril 2015

le magazine du développeur

Java 8

Quel bilan après 1 an ?
Java 9 : objectif 2016
20 ans de Java

3D

Unity 5.0 :
Les nouveautés

Bases de données

NoSQL et la révolution de la donnée

Xamarin

Osez les Xamarin Forms

Fière d'être développeuse !

Le premier développeur était une femme !

Défi

Développer sur Linux avec un Chromebook

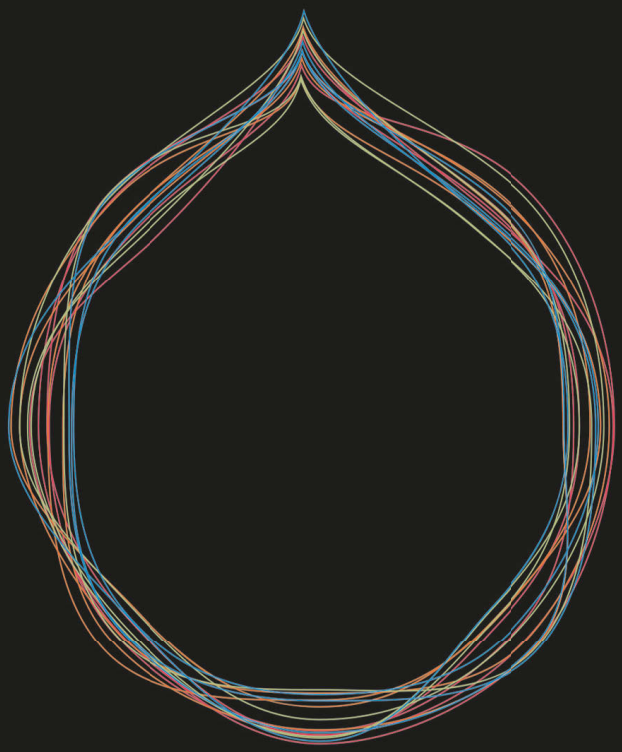
Drupal

Choisir son outil de tests

M 04319 - 184 - 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

acquia.com



Réussissez votre Transformation Digitale avec Acquia.

Optimisez et Personnalisez l'expérience utilisateur sur votre site avec Acquia Lift

Réduisez votre Time to Market avec Acquia Cloud Site Factory

Créez des experiences e-Commerce incomparables avec Acquia Commerce Cloud

Acquia | THINK AHEAD

Contact: 01 79 97 25 70

Illogique, capitaine

Fin février, une icône de la science-fiction nous a quittés. Leonard Nimoy était le mythique Mr Spock dans Star Trek. Cette figure emblématique de la culture américaine et de l'univers geek est apparue en 1966. Autour de ce personnage, tout un mode de pensée a été créé. La civilisation vulcaine a rapidement intrigué les trekkies et au-delà.

Le dernier tweet de l'acteur est aussi poétique que réaliste : « la vie est comme un jardin. On peut y trouver des moments parfaits, mais pas les préserver, excepté dans (notre) mémoire ».

Longue vie et prospérité.

Avec ce 184e numéro, Programmez ! rentre dans sa 17e année d'existence. Que le temps passe vite. Toute l'équipe est fière de travailler mois après mois pour vous proposer un contenu éclectique et de qualité.
Depuis 18 mois, Programmez ! a beaucoup changé : 1 site Web 100% Drupal, les premières apps mobiles, une nouvelle maquette (qui a été ajustée dans les n°182 et 183). Nous continuerons nos itérations durant l'année 2015, en attendant, notre prochaine version majeure : Programmez ! n°200.

Développeuse, Xamarin, Maker, NoSQL, Java...

Ce mois-ci, nous plongeons dans plusieurs (très) grosses technos : le NoSQL avec un important focus sur Cassandra (la base NoSQL de référence avec MongoDB), Xamarin (avec plusieurs articles dédiés et notamment les problèmes de compatibilité sous Android). La partie Maker se concentre sur une gestion d'une cave à vins 2.0 et la présentation du nouveau Raspberry Pi 2 (et franchement, il envoie du lourd). Pour les demi-dieux, nous vous proposons des contre-mesures pour contrer les attaques DDOS et la suite de nos aventures JavaScript, en mode Jedi.

Il y a quelques numéros de cela, nous avions dit pourquoi développeur est un beau métier. Aujourd'hui, nous allons parler aux femmes : soyez fière d'être développeuse et pourquoi être développeuse, c'est super sympa.

Autre gros morceau : Java. Impossible de ne pas parler de Java 8. Cela fait un an que cette version est proposée officiellement aux développeurs. Quel bilan ? Java est-il toujours d'actualité ? Et demain, que faut-il en attendre ?

Nous vous proposons également bien d'autres technos : Unity 5.0, Drupal et les tests unitaires, le Machine Learning, les nouvelles architectures de compilateurs avec Roslyn, le design avec les ergonomies tactiles, les API d'Office 365.

Bon code !

ftonic@programmez.com

à lire dans le prochain numéro n°185 en kiosque le 30 Avril 2015

LE FUTUR DES LANGAGES DE PROGRAMMATION
C++, Go, Dart, JavaScript, TypeScript, Java, HTML, C#, PHP, Ruby, Swift... Que nous réservent les futures versions de nos langages favoris ? Programmez ! fait le point !

DEVOXX FRANCE 2015
Retour sur la plus importante conférence Java de France

INDEX TIOBE

Pour le mois de mars, pas de changement en tête du classement. Cependant, on constate une baisse du podium. Objective-C subit une grosse chute et C++ le talonne de très près. Pour le reste, notons une belle progression de JavaScript.

mars 2015	changement	Langage	%	Evolution
1	-	C	16.642%	-0.89%
2	-	Java	15.580%	-0.83%
3	–	Objective-C	6.688%	-5.45%
4	+	C++	6.636%	+0.32%
5	-	C#	4.923%	-0.65%
6	+	PHP	3.997%	+0.30%
7	++	JavaScript	3.629%	+1.73%
8	+	Python	2.614%	+0.59%
9	+	Visual Basic .NET	2.326%	+0.46%
10	++	Visual Basic	1.949%	+1.95%

Un carte SD de 200 Go ? SanDisk l'a fait ! Prix ? Environ 400 $

Microsoft va-t-il concurrencer Docker ? Le projet Drawbridge est un indice...

Firefox OS : Orange va l'utiliser au Moyen Orient et en Afrique

Google + se coupe en deux : Google+ Photos et Google+ Streams

A chaque mois, sa faille : aujourd'hui, FREAK, héritage des années 1990 !

NAO N'EST PLUS FRANÇAIS

Durant plusieurs jours, la nouvelle n'avait guère intéressé les média. Et pourtant, il s'agissait d'un réel séisme dans la robotique française. Aldebaran passait sous contrôle de Softbank, entreprise japonaise qui avait soutenu le projet Pepper. Softbank possède 95 % du constructeur robotique. Etrangement, les politiques français n'ont pas réagi à cette annonce pourtant cruciale dans la robotique et la haute technologie française. Aldebaran sera dirigé par un Japonais. Pas de détails sur les futures orientations et les conséquences possibles de ce rachat sur le siège parisien et les équipes.

2015 LINUX JOBS REPORT : la fondation Linux analyse l'emploi dans l'open source

Ce rapport a été réalisé par DICE pour la fondation Linux. L'étude porte sur 1000 responsables IT dans le monde et totalise plus de 3400 réponses de professionnels Linux. Que retenir ?

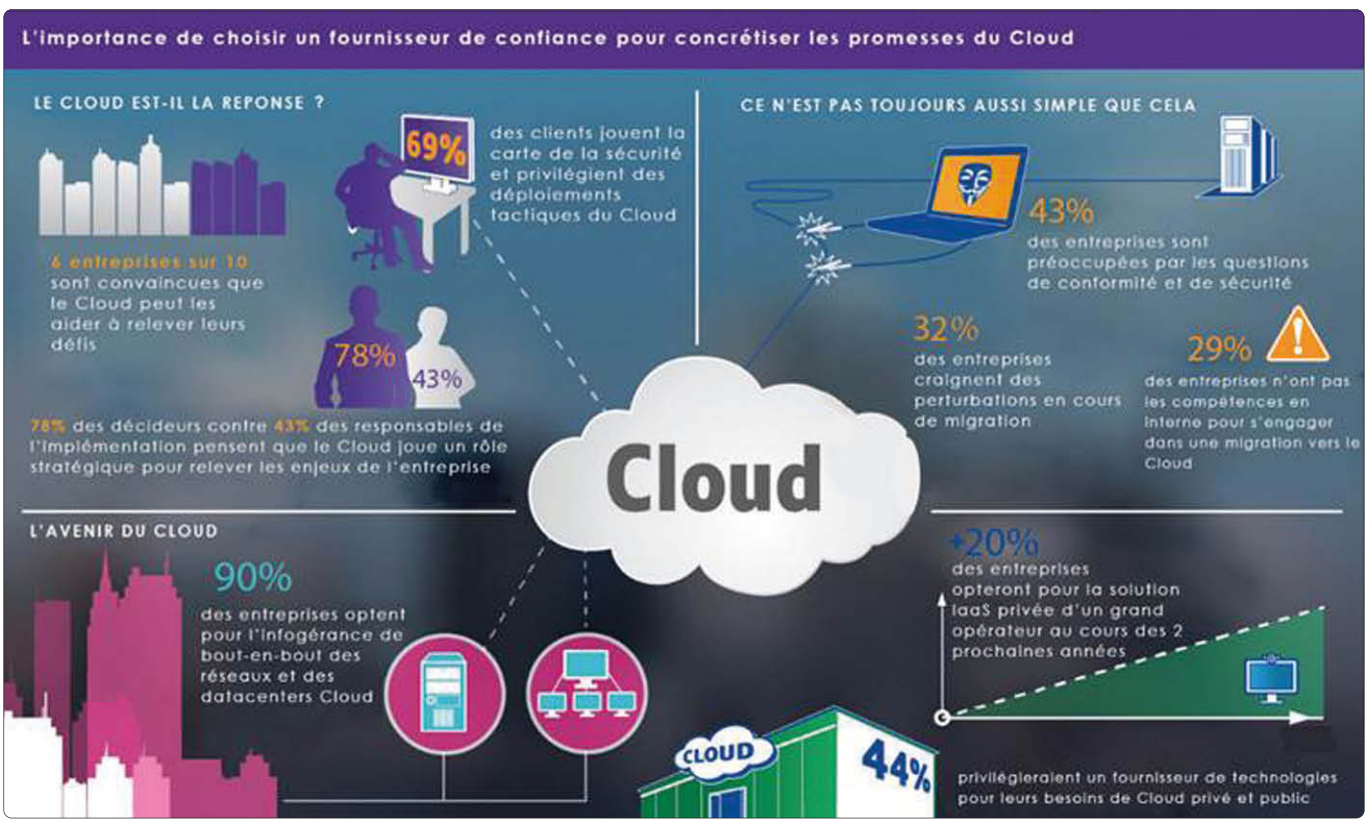
- Que les entreprises vont recruter des professionnels Linux dans les prochains mois.
- Que le Cloud va être un moteur puissant notamment OpenStack et CloudStack.
- Que les experts Linux ayant une certification vont être une cible pour les entreprises

HTTP V2 ARRIVE

http/2 a été finalisé et sa standardisation officielle ne tardera pas. Cette v2 est attendue car la précédente évolution majeure remonte à 1999. Cette nouvelle version sera compatible avec http 1.1. Les principaux atouts de http/2 sont des requêtes plus nombreuses (sur une connexion TCP), compression des en-têtes, meilleure gestion des priorités des requêtes. http/2 est supporté par Firefox et Chrome (pour tests).

JAVA 8 UPDATE : POLÉMIQUE

Oracle sortait début mars une mise à jour de Java sur OS X. Surprise, aux Etats-Unis, dans l'installation, l'installation d'une barre d'outils Ask apparaît... Une pratique que l'on croyait disparue. Il est possible de désactiver l'option mais il faut être attentif durant la procédure.



avril

Drupal developer days

Montpellier. Du 13 au 19 avril, Drupal fera son grand show durant toute une semaine. Site : http://montpellier2015.drupaldays.org

Journée française des tests logiciels 7e édition

Le 14 avril se déroulera la nouvelle édition de la journée des tests logiciels. Cette année, l'événement change de lieu. Comme à chaque édition, la journée sera ponctuée de sessions générales et plus « techniques ». Une journée sera consacrée aux tutoriels (13 avril), la journée ouverte étant celle du 14 avril. De nombreux éditeurs et SSII seront présents.
Pour en savoir plus : http://www.jftl.org

Global Azure Bootcamp

La communauté Azure se retrouvera le 25 avril dans plusieurs villes dans le monde entier. En France, il y aura des conférences à Paris et à Lyon. Sur Lyon, l'événement promet d'être animé dans la prestigieuse Manufacture des Tabacs.
La journée s'articulera autour de conférences techniques et notamment sur les technologies Open Source. Programmez ! sera présent toute la journée avec l'animation d'une conférence par François Tonic et un stand.

avril/mai

Maker Faire 2015 à Saint-Malo et à Paris

Cette année, nous aurons droit à deux Maker Faire en France, organisées par le FabLab

- 11 & 12 avril à Saint-Malo : c'est l'occasion de rencontrer des dizaines de makers et leurs créations, dans tous les domaines, pas uniquement autour de la robotique ou de l'arduino ! Vous pourrez voir une moto électrique, les lego comme support technologique ou encore comment construire une imprimante 3D ! Des exposants seront aussi là pour vous montrer toutes les facettes du monde maker ! Tarifs : 10 € (+ 12 ans), 5 € (- 12 ans). Site officiel : http://www.makerfairesaintmalo.com
- 2 & 3 mai à la Foire de Paris : après une édition 2014 déjà très réussie, Maker Faire Paris revient à la Foire de Paris ! Au programme : de l'art, de la technologie, de l'impression 3D, de l'artisanat, du design, des loisirs créatifs ! Plusieurs centaines de Makers sont attendus pour présenter leurs projets, des dizaines d'exposants. Le programme définitif sera disponible mi-avril. Les makers ont jusqu'au 2 avril pour déposer leur projet aux organisateurs. Certains sont déjà connus : MRobot, Cyborg Vegetal, Atom 2, DomoTab...

23 avril

conférence DevDays à l'école ESGI

L'ESGI organise une après-midi développeur dans ses nouveaux locaux parisiens, de 14 h à 19 h.
Au programme :

- Plyng : développement Web pour créer des jeux multijoueurs en ligne.
- Les métiers de développeur dans les sociétés de service
- Programmation NDK
- Programmation fonctionnelle avec Scala
- François Tonic (éditeur & rédacteur de Programmez) fera une intervention un peu particulière, « des hiéroglyphes au C# ».

Devoxx France 2015 : du 8 au 10 avril 2015

Devoxx France 2015, 4e édition, est placé sous le thème des 20 ans du langage Java. Les plénières, le jeudi et le vendredi, proposeront des sujets autour du Futur, des 20 prochaines années et des possibilités infinies du langage. Brian Goetz, architecte en chef de Java chez Oracle, parlera des perspectives d'évolution de la plateforme et de la JVM. Stephan Tual d'Ethereum présentera une solution logicielle basée sur les principes de Bitcoins. Eric Filiol de l'ESIA parlera sécurité et confidentialité des données, et Quentin Adam de Clevercloud proposera une nouvelle vision dans 20 ans du Cloud et du déploiement d'application.

Devoxx France propose plus de 89 conférences le jeudi et le vendredi, avec des présentations phares autour de la JVM, de Java, de Spring Boot, ou des architectures distribuées. Des sujets plus originaux comme la robotique, l'importance de l'UX dans le développement, ou l'utilisation de Docker dans le développement permettront aussi à un vaste public de trouver un sujet intéressant.

La journée du mercredi propose plus de 11 ateliers pratiques. Les participants pourront ainsi découvrir le langage Go, faire un atelier Spark avec Scala et Java 8, ou découvrir la plate-forme Google Compute Engine avec les développeurs de Google. Les Universités sont aussi des sessions longues de 3 heures, qui permettent chaque année de découvrir un thème en profondeur. Après AngularJS en 2013, puis Docker en 2014, la tendance cette année est plus partagée. Tout d'abord beaucoup de Java, avec un bilan de Java 8. Mais aussi des sujets plus pointus comme l'optimisation et la performance par Gil Tene d'Azul Systems. D'autres présentations porteront sur Mesos, Spring Boot, l'écosystème Javascript ou les Streams en Java 8.

Comme chaque année, les organisateurs de Devoxx France filment et proposeront gratuitement les conférences sur le site Parleys.com quelques semaines après la conférence.

Les organisateurs attendent plus de 2500 personnes par jour, dont 177 orateurs.

Remèdes anti-DDoS

Dans l'article « Anonymous : Internet est à nous ! » paru au numéro précédent, nous avons vu que les énergies rassemblées des membres du collectif se muent le plus souvent en actions de déni de services. L'attaque DDoS n'est pas l'apanage des hacktivistes, elle est en outre de plus en plus utilisée car assez simple à réaliser. A contrario il est bien plus difficile de s'en prémunir, son niveau de risque est quant à lui en forte augmentation.

Véronique Loquet
Fondatrice de l'agence RP AL'X Communication. **Spécialiste de l'Open Source et de la sécurité sur Twitter** @vloquet

Renaud Bidou, expert en sécurité, nous éclaire sur les différentes mesures techniques et les étapes permettant de parer au mieux ces assauts. La lutte anti-DDoS est faite de compromis et de rapports de force; faute de pouvoir déjouer l'attaque, il semble toujours possible d'en atténuer l'impact.

Principes de protection

Plusieurs hypothèses sont à prendre en compte en fonction des éléments à protéger :

- Le type d'infrastructures concernées,
- Le type de menace suivant ce que l'on considère le plus exposé, ou le plus critique,
- L'architecture système dans laquelle devront s'intégrer les éléments de sécurité,
- Les coûts.

Tour d'horizon des contre-mesures

- Les firewalls réseau, qui offrent un degré de protection limité contre certaines attaques réseau de type SYNflood,
- Les IPS, qui se concentrent sur la prévention des dénis de service réseau via des mécanismes d'analyse comportementale,
- Les firewalls applicatifs, en particulier les WAFs parfois pertinents dans la prévention des attaques au niveau applicatif,
- Les mécanismes de réputation IP, efficaces contre des attaques statiques.

Rappelons ici que la mise à jour régulière des correctifs de failles est un basic de l'hygiène informatique. Nessus, Microsoft Baseline Security Analyzer (MBSA), SARA... sont des outils pour évaluer la sécurité, il existe plusieurs services pour scanner en ligne gratuitement, comme BrowserCheck de Qualys, qui identifie en un clic les vulnérabilités des navigateurs et des plugins et propose d'en effectuer les mises à jour. Toute une gamme de freemium est disponible sur le site de l'éditeur suivant les points à vérifier. Pour boucher les trous des applications Web, Google vient par exemple de présenter la bêta d'un outil gratuit de détection de vulnérabilités pour Google App Engine.

Des techniques pour se protéger

SYNcookies

Les SYNcookies ne sont pas récents. Ils sont toutefois une solution relativement efficace de protection statique contre la saturation du TCB (Transmission Control Block) des systèmes ciblés. Le mécanisme consiste à effectuer un transfert des ressources mémoire (vecteur cible des SYNFloods) vers des ressources de traitement. Lors de l'établissement d'une connexion TCP les informations de connexion sont maintenues dans le TCB. Dans l'absolu la seule information qui n'est pas définie de manière déterministe est le numéro de séquence TCP, aléatoire, généré par le serveur. Dans le cas des SYNcookies, ce numéro de séquence est généré à partir d'une fonction déterministe dont la sortie est le hash des données suivantes :

- Les paramètres réseau de la connexion : adresse et port source, adresse et port destination,
- Un timestamp : nécessaire pour identifier un timeout,
- Une graine : indispensable pour ne pas devenir « spoofin ready », générée aléatoirement au boot du système.

Ce SYNcookie devient le numéro de séquence envoyé par le serveur au client. Il n'est donc pas utile à ce stade de créer une entrée dans le TCB. Dans le cas d'une connexion légitime, le numéro de séquence acknowledgé dans le ACK final du 3-way handshake peut être vérifié par méthode calculatoire. Dans le cas d'un SYNflood il n'y a pas de ACK final, ce qui importe peu puisque aucune entrée correspondante n'est créée dans le TCB.
Cette technique est efficace si le système mettant en œuvre les SYNcookies dispose de la puissance calculatoire nécessaire. Cela impose qu'il ne s'agit pas de la ressource à protégér (généralement occupée à d'autres tâches), et qu'il dispose de composants hardware dédiés (FPGA ou network processors) à cette tâche.
Cette technique n'est pertinente que contre les SYNfloods arrivant à destination, ce qui signifie que le volume des « petits paquets » n'a pas pulvérisé d'autres équipements en amont.

L'analyse comportementale

L'analyse comportementale est un moyen de détection visant à identifier et à caractériser une attaque en vue de son blocage. Il est important de distinguer plusieurs niveaux d'analyse :

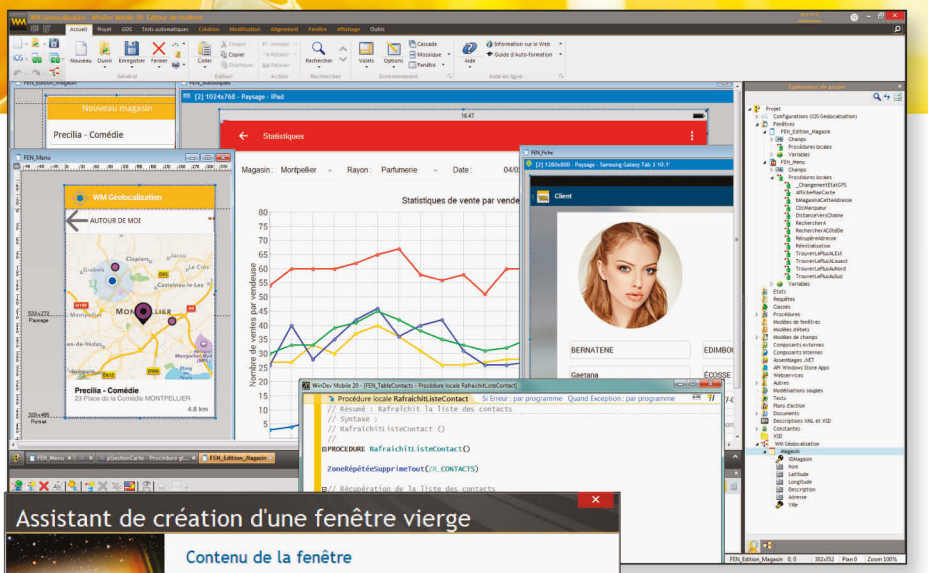
- L'analyse des flux,
- L'analyse des données réseau,
- L'analyse du contenu applicatif,

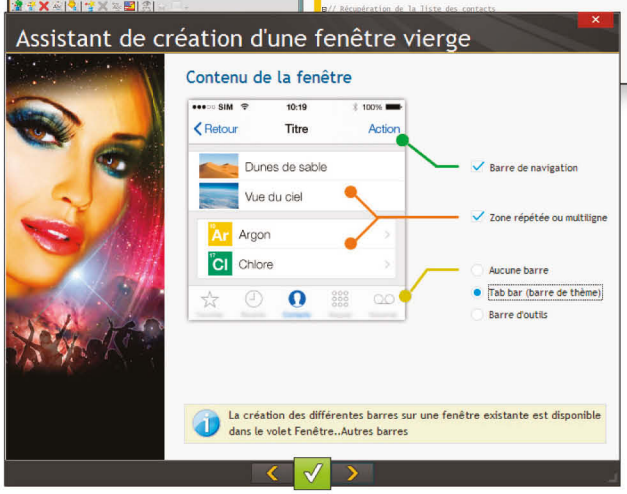
Dans tous les cas, le principe est d'identifier les critères communs à tous les éléments de trafic participant à l'attaque (en supposant qu'ils présentent effectivement de tels critères).
L'analyse de flux est une approche macroscopique consistant à identifier globalement des anomalies en termes de volume, qu'il s'agisse du nombre de paquets par seconde ou de la bande passante.
Généralement réalisée à partir de données NetFlow, cette analyse permet d'identifier la cible de tout type d'attaque basé sur le volume et de blackholer la pauvre victime. L'analyse des données réseau est plus précise, elle porte sur les éléments caractéristiques d'un paquet (ports, adresses, numéros de séquences, IPID, TTL...). En fonction du niveau de sophistication, il est possible de caractériser le type et la nature de l'attaque. Ces caractéristiques peuvent ensuite être exploitées par le moteur de blocage.
L'analyse du contenu applicatif devient nécessaire pour les attaques ne répondant pas aux critères de seuils des attaques réseau, le plus souvent construites à partir d'outils de génération de niveau applicatif. Il s'agit alors d'identifier des éléments caractéristiques de l'attaque dans le contenu d'une requête applicative (URL, headers, paramètres). Dans le même esprit que le résultat de l'analyse réseau, cette analyse de niveau applicatif doit permettre au moteur de blocage de cibler les requêtes utilisées pour le déni de service.

Seuils

Le principe de cette méthode est trivial : une valeur est définie pour une métrique, en cas de dépassement le service n'est plus rendu ou une alerte est levée. On distingue deux types de seuils : les seuils de ressources et les seuils temporels. La définition d'un seuil de ressources est un moyen simple et efficace pour prévenir certaines attaques. Le

WINDEV MOBILE 20
LE DÉVELOPPEMENT NATIF POUR TOUS LES MOBILES





Un environnement de développement complet, intégré, adapté au monde du «mobile»

PORTABILITÉ DE VOS APPLICATIONS

ANDROID, IOS, WINDOWS PHONE, WINDOWS MOBILE & CE

Avec WINDEV Mobile 20, une même application peut fonctionner sous les différents OS mobiles: iOS (iPhone, iPad), Android, Windows CE & Mobile, Windows Phone... Recompilez !

TOUS LES TYPES DE MOBILES

Développez pour tous les mobiles: téléphones, smartphones, pocket PC, terminaux, terminaux durcis, terminaux industriels, tablettes, netbook,...

CRÉEZ DES APPLICATIONS NATIVES POUR TOUS LES SYSTÈMES MOBILES

WINDEV Mobile 20 permet aux professionnels du développement de créer facilement des applications natives pour tous les mobiles: smartphones, tablettes et terminaux industriels. Et si vous possédez un existant WINDEV ou WEBDEV, vous pouvez le ré-utiliser.

UN ENVIRONNEMENT DE DÉVELOPPEMENT AUTONOME

Quels que soient le matériel cible et le système d'exploitation, la méthode de développement avec **WINDEV Mobile 20** est similaire.
L'environnement de développement est intégré, puissant, complet, intuitif, et il est adapté aux spécificités des mobiles.
Avec ou sans base de données, avec ou sans connexion au S.I. il n'a jamais été aussi facile de développer professionnellement sur mobile.

LE CYCLE DE VIE COMPLET EST GÉRÉ

WINDEV Mobile 20 est livré en standard avec tous les outils qui permettent de gérer le cycle de vie des applications: Générateur de fenêtres, Langage L5G, Débogueur, Générateur de rapports, Générateur d'installations, mais aussi Générateur d'analyses Merise et UML, Tableau de Bord du projet, Gestion-naire de Sources collaboratif, Générateur de dossier de programmation, Suivi des plannings,...

PROGRAMMEZ EN L5G: 90% DE CODE EN MOINS

Le langage de 5ème génération WLangage permet de développer plus vite qu'avec un langage traditionnel.
Ses fonctions évoluées rendent le code facile à écrire et à lire, facilitent à la fois le développement et la maintenance.

VERSION EXPRESS GRATUITE
Téléchargez-la !

Tél province: **04.67.032.032**
Tél Paris: **01.48.01.48.88**

Fournisseur Officiel de la Préparation Olympique

www.pcsoft.fr
Des centaines de témoignages sur le site

mécanisme le plus courant est la limitation du nombre de clients simultanés connectés à un serveur Web, mis en œuvre par défaut sur l'ensemble des serveurs ne travaillant pas en mode événementiel. Cependant cette implémentation n'est pas la plus pertinente car un pic de trafic de l'application peut provoquer l'atteinte de ce seuil et le rejet de connexions légitimes. Un seuil temporel peut être défini soit comme un nombre d'événements intervenant au cours d'une durée fixe, comme le nombre de paquets par seconde ; soit comme une durée maximale, par exemple la durée maximale d'une session applicative.

Signatures

Les signatures peuvent être utilisées efficacement dans le cas d'attaques précédemment caractérisées, soit par moteur d'analyse ou « manuellement ». Dans le premier cas la signature est automatiquement générée à partir d'un système d'analyse comportementale. Dans le second elle est conçue en amont pour bloquer un élément spécifique de l'attaque. Avec cette approche tous les éléments de trafic (paquets, requêtes, contenu applicatif) correspondant à cette signature sont systématiquement bloqués. Ce blocage peut être temporaire, par exemple appliqué uniquement lorsque les seuils définis pour la détection sont dépassés, ou devenir permanent. Cette dernière option, plus efficace en termes de performances et temps de réaction présente toutefois deux contreparties non-négligeables :

- Le risque de bloquer définitivement un trafic légitime qui a été temporairement détourné,
- L'accumulation de règles de filtrage rendant la maintenance plus complexe et ayant in fine un impact sur les performances.

Blackhole et Blacklist

Le principe du trou noir est de détourner tout le trafic correspondant à certaines caractéristiques vers /dev/null dès qu'il est identifié. Objectif : Sauver Ryan ! En effet certaines attaques présentent de tels volumes qu'elles peuvent avoir un impact sur l'infrastructure de transport amont, c'est-à-dire l'opérateur. Il n'est alors pas question de faire dans la dentelle : tout le trafic à destination de la cible est poubellisé dès son apparition sur le réseau. Certes la cible n'est plus accessible, mais le réseau de l'opérateur reste fonctionnel. Le sacrifice d'un seul pour le bien de tous.
A l'inverse les listes noires consistent à bloquer systématiquement le trafic provenant d'une source considérée comme menace active. C'est dans ce contexte qu'est introduite la notion de réputation d'une adresse IP, établie à partir d'infrastructures globales corrélant les informations concernant des attaques DDoS observées. Simple tant que les sources ne changent pas, ou qu'une attaque n'a pas été lancée à partir d'une adresse spoofée.

Architectures et impacts

Schémas d'architecture

Une architecture de protection contre les dénis de services est composée de plusieurs types d'éléments : les éléments de détection, les éléments de caractérisation et les éléments de blocage. Dans certains cas triviaux ces trois fonctions sont regroupées dans un seul et même composant, généralement le système que l'on cherche à protéger, par exemple le cas d'un mécanisme de timeout de session.
Pour les autres approches, plus adaptées à la prévention des dénis de services basés sur de forts volumes réseau ou applicatifs, on distingue deux types de déploiements pour des équipements dédiés à ce type de fonction :

- Le déploiement en ligne,
- Le déploiement en dérivation.

Dans le premier cas les équipements sont situés en coupure du réseau. La détection s'effectue à partir du trafic observé sur un point précis et le blocage n'intervient que sur le lien protégé. Les métriques sont donc calculées sur la base de l'intégralité du trafic observé en un point précis et la protection n'est appliquée que sur le nœud de la chaîne réseau ou applicative liée. Il s'agit d'architectures de protection adaptées à des réseaux relativement centralisés tels qu'en entreprise, sur lesquels sont déployés des IPS pour la prévention des DoS réseau et des WAFs pour la prévention des DoS applicatifs Web.
Dans le second cas les fonctions de détection et de caractérisation sont effectuées via des échantillons du trafic prélevés à différents points du réseau et « copiés » à destination des systèmes de sécurité. Il s'agit d'une analyse distribuée, adaptée à la détection d'attaques provenant de plusieurs points d'accès. En cas de détection d'une attaque, seul le trafic correspondant est routé vers les « machines à laver » en charge du nettoyage.

Architectures Cloud

Les architectures Cloud mises en œuvre par la plupart des éditeurs de solutions de prévention des DoS fournissent aux entreprises un service de nettoyage en amont de leur connexion Internet. Il s'agit de rediriger tout le trafic entrant vers ces services qui réalisent les fonctions de détection, identification et blocage. Cette approche présente quelques avantages intéressants :

- La consolidation des flux à destination de l'ensemble des clients de la plateforme permet d'optimiser le temps de réaction. Dans le cas d'une attaque à grande échelle, la première cible protégée par l'opérateur servira de référence lorsque l'attaque se propagera à d'autres systèmes,
- L'architecture de la plateforme à sécuriser n'est pas modifiée. Le déploiement peut être très rapide, voire réalisé « en live » au cours d'une attaque.

Performances

Quel est l'impact sur les performances ? Dans le cas des IPS déployés en ligne sur les réseaux d'entreprise, compter une latence supplémentaire de l'ordre de quelques millisecondes pour les moteurs d'analyse comportementale (une dizaine en cas d'attaque) et un impact quasi-nul pour les mécanismes de type SYNcookies. Pour les mécanismes de protection opérant au niveau applicatif, comme les WAFs, il faut envisager une latence d'environ 10 ou 20 millisecondes en cas d'attaque. Garder à l'esprit que ces latences sont applicables à tous flux transitant par les équipements en question. Sur les réseaux d'opérateurs les systèmes en dérivation n'ont pas d'impact notable. En cas d'attaque la latence peut atteindre plusieurs dizaines de millisecondes. Cette latence n'est applicable qu'aux flux malicieux détournés vers les systèmes de blocage, sachant que ces plateformes traitent des attaques dont les volumes sont 10 ou 100 fois ceux traités par les IPS et WAFs; raison pour laquelle les équipements en ligne déployés sur les réseaux d'opérateurs peuvent accroître la latence de quelques 100 millisecondes.
L'exploit étant déjà qu'ils résistent à l'attaque !
Enfin les solutions Cloud... Le flux étant détourné vers une plateforme tierce il est nécessaire de rajouter le temps de transit. En revanche le dimensionnement de la plateforme de protection étant calculé pour soutenir des attaques massives, cette latence reste quasi-constante dans le cas d'une attaque qui ne ciblerait qu'une seule des infrastructures protégées... quelques dizaines de millisecondes.

Conclusion

Il ne faudrait pas sous-estimer l'impact de ces attaques et leur nombre grandissant. La parade de prévention des DDoS n'est pas simple, peut être coûteuse et avoir un impact sur l'architecture et les performances. Néanmoins les solutions existent et les évolutions sont constantes en termes de qualité de détection, de rapidité de réaction et de performances. ⊞

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

Développeurs, Apps, SDK, API : les enjeux des objets connectés et du wearable

On nous parle partout d'objets connectés (ou IoT, Internet des objets), d'informatique sur soi (le fameux « wearable). Mais encore trop souvent, on oublie l'usage réel derrière l'objet technologique. Or, sans usage, ces objets ne servent à rien. Et sans apps, comment les utiliser ? Certains constructeurs ont compris l'importance de l'écosystème et particulièrement du développeur.

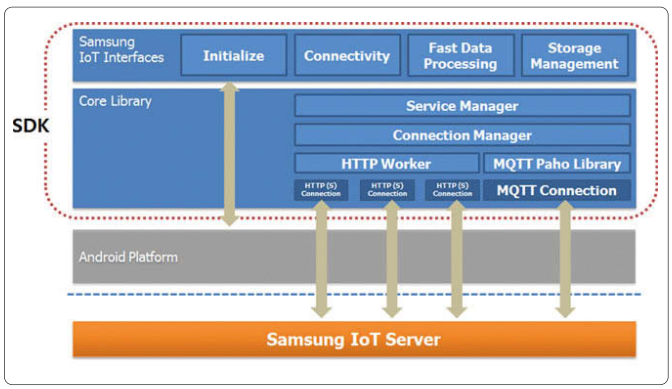
Le smartphone a montré comment et pourquoi le SDK et les apps étaient vitaux! Le succès d'Android et d'iOS repose en partie sur cette logique. Ce n'est pas un hasard si les développeurs sont mis à l'honneur, et que chaque constructeur rappelle les chiffres clés : nombre d'apps et de développeurs, les milliards redistribués, etc. Les objets connectés et le wearable ne font pas exception.

Apple sait qu'il faut des développeurs

Prenons l'Apple Watch qui fait tellement parler d'elle depuis quelques mois et qui sortira le 24 avril prochain. Dès le mois de novembre 2014, Apple a libéré WatchKit, le SDK dédié à la montre. Le constructeur veut que les développeurs sortent des apps dès la sortie officielle de l'objet. Mais comme souvent, Apple limite, dans un premier temps, les fonctions disponibles : pas d'applications natives, accès matériel très limité, durée des flux avec l'iPhone imposée, etc. Malgré tout, des apps seront disponibles dès le 24 avril et elles devraient rapidement se multiplier. Ceci même si Apple sera très stricte sur les règles de façon à ne pas nuire à l'ergonomie, et, surtout, à l'autonomie de la montre. Cependant, Apple libère peu à peu les accès au fur et à mesure des itérations des SDK. Mais l'approche d'Apple est pragmatique, et ils savent qu'un objet connecté ne peut réussir que s'il y a des développeurs et des apps. Pebble, un des piliers du marché des montres connectées, propose lui aussi un SDK et des API pour créer des apps et interagir avec Android et iOS; même si les contraintes matérielles sont très fortes, le développeur dispose d'un SDK assez complet.

MS Band : Microsoft dégaine son SDK

Le bracelet connecté de Microsoft, MS Band, est sorti, en quantité très limitée, l'automne dernier. En février 2015, l'éditeur a sorti un SDK dédié. Il permet d'accéder aux différents capteurs, créer et recevoir des notifications, étendre les fonctionnalités du bracelet. La logique applicative s'exécute sur Android, iOS (SDK pas encore disponible) et Windows Phone, mais aussi sur OS X et Windows. Il permet d'accéder aux capteurs et aux données suivants : accéléromètre, gyroscope, distance, rythme cardiaque, podomètre, UV, température de la peau.



Android est bien présent !

Côté Android, nous trouvons tout naturellement Android Wear qui peut être vu comme une extension d'Android sur les objets de type montres. On y trouve les notifications, les interactions vocales et l'accès aux capteurs de l'objet, avec accès à la couche de données (indispensable pour les apps de santé par exemple). Vous pouvez créer des applications natives. Les possibilités sont grandes là-encore et tout va dépendre des objectifs de votre app, et bien entendu des possibilités de votre wearable. De plus, selon le constructeur, vous aurez accès à plus ou moins d'API, de personnalisation, de bonnes pratiques.

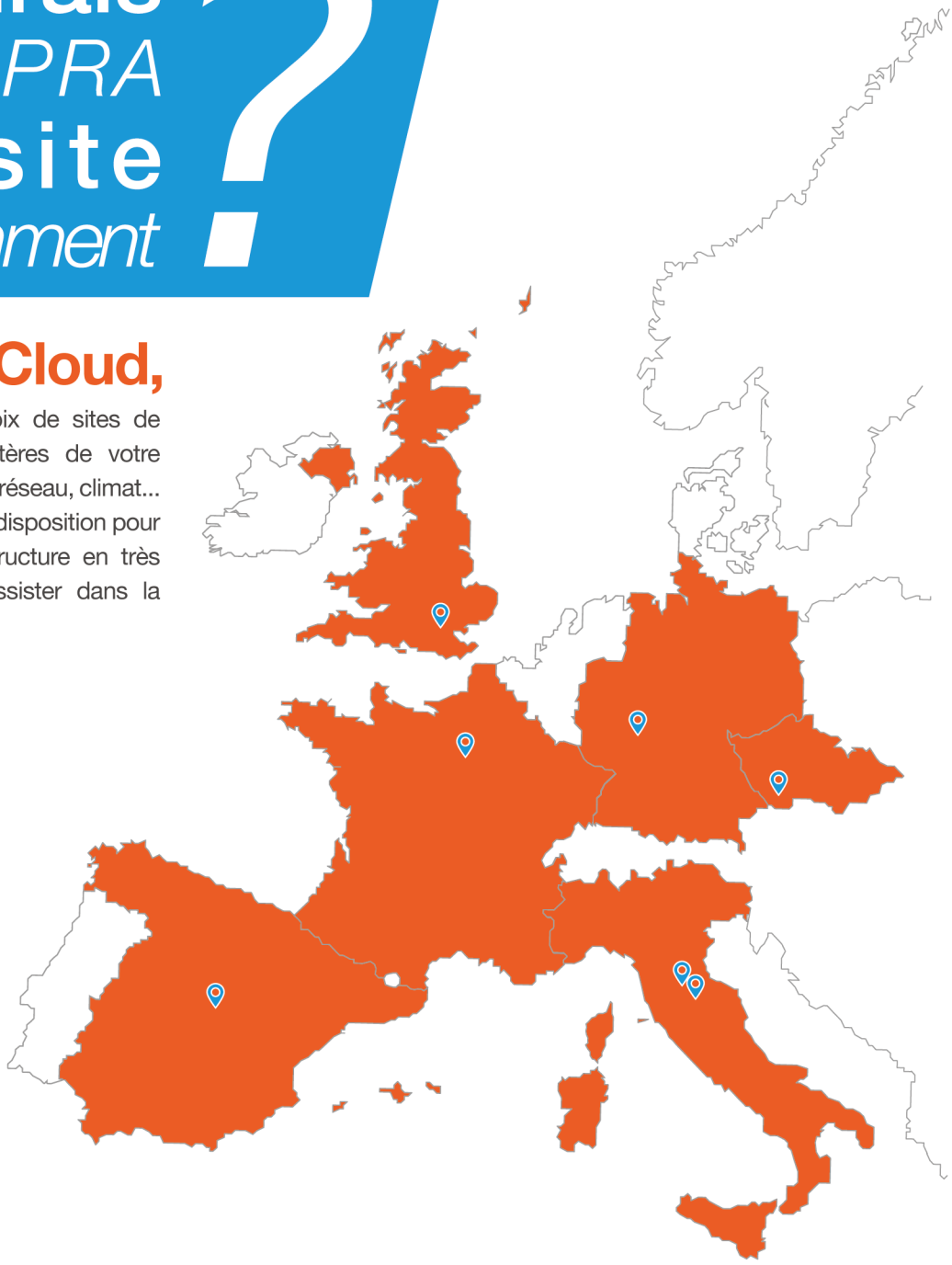
Prenons de la hauteur

Mais il n'y a pas que les SDK liés aux matériels, il y a aussi tous les environnements pour collecter, diffuser, traiter les données provenant de ces objets divers et variés. Et là, chaque fournisseur fait un peu ce qu'il veut. Samsung propose sa propre plateforme d'objets connectés : Samsung IoT SDK pour les matériels sous Android 4.1 minimum. Ce kit permet créer des connexions pour envoyer et recevoir des messages, traiter des données, stocker les données sur le service Samsung IoT Cloud Platform, etc. Microsoft veut faire la même chose avec Azure. HP a dévoilé sa plateforme Cloud : HP IoT Platform. Oracle ne fait pas autre chose avec sa plateforme IoT pour connecter, collecter et traiter les données. Ces fournisseurs misent beaucoup sur les capacités de stockages et d'analyses des données. Alors que les SDK des constructeurs s'orientent plus sur l'usage et les apps. Nous sommes sur différents niveaux d'usage et d'intégration. Les platesformes de type Azure, Oracle, HP, seront plus pour les entreprises et les traitements dits big data pour analyser les usages, le comportement. Mais ils serviront aussi à l'usage quotidien des objets.
Bref, selon la plateforme, les possibilités pour le développeur seront différentes et plus ou moins complètes. À vous de jauger le marché et de savoir si les développements souhaités seront possibles sur la plateforme choisie. Le marché est en pleine explosion même si la montre connectée, par exemple, se cherche. Les IoT vont s'imposer et se multiplier que l'utilisateur le souhaite, ou non.

Je voudrais *bâtir un PRA* **multi-site** *Je fais comment* ?

Avec Aruba Cloud,

vous disposez d'un large choix de sites de secours, en fonction des critères de votre stratégie de sécurité: proximité, réseau, climat... Nos équipes sont aussi à votre disposition pour vous aider à bâtir une infrastructure en très haute disponibilité et vous assister dans la définition de votre stratégie.



3 hyperviseurs

6 datacenters en Europe

APIs et connecteurs

70+ templates

Contrôle des coûts

Nous avons choisi Aruba Cloud car nous bénéficions d'un haut niveau de performance, à des coûts contrôlés et surtout car ils sont à dimension humaine, comme nous. Xavier Dufour - Directeur R&D - ITMP

Contactez-nous! 0810 710 300 www.arubacloud.fr

aruba CLOUD

MY COUNTRY. MY CLOUD.*

Cloud Public | Cloud Privé | Cloud Hybride | Cloud Storage | Infogérance

*MON PAYS, MON CLOUD.

1&1 SERVEUR CLOUD

NOUVELLE GÉNÉRATION

Easy to use – ready to cloud*

Le nouveau serveur Cloud 1&1 procure une combinaison parfaite entre la performance d'un hardware dédié et la flexibilité du Cloud !

FLEXIBLE & ABORDABLE

Configuration individuelle

- CPU, RAM et stockage SSD sont configurables indépendamment et en toute flexibilité afin de s'adapter au mieux à vos besoins

Transparence des coûts

- **NOUVEAU :** facturation à la minute
- **NOUVEAU :** facturation détaillée, claire et structurée pour une totale maîtrise de votre budget

SIMPLE & SÛR

1&1 Cloud Panel

- **NOUVEAU :** l'interface innovante et conviviale facilite la gestion de votre serveur

Sécurité

- Les data centers haute performance de 1&1 comptent parmi les plus sûrs en Europe
- Les sauvegardes quotidiennes et snapshots vous protègent de la perte de données
- Le firewall intégré bloque les attaques contre votre serveur

TOUT INCLUS

Haute performance

- **NOUVEAU :** votre serveur Cloud livré en moins d'1 minute
- **NOUVEAU :** technologie SSD pour une performance maximale
- **NOUVEAU :** réseau privé VLAN, API, load balancing, firewall et de nombreuses autres fonctionnalités facilement configurables
- **NOUVEAU :** virtualisation reposant sur la technologie leader VMware
- **NOUVEAU :** applications incluses, prêtes à l'emploi : WordPress, Drupal™, Magento®
- Parallels® Plesk 12
- Trafic illimité

1 MOIS POUR ESSAYER

1 CLIC POUR CHANGER DE PACK

1 APPEL UN EXPERT VOUS RÉPOND

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

Saisissez votre email et testez immédiatement

1 MOIS D'ESSAI GRATUIT !*

0970 808 911
(appel non surtaxé)

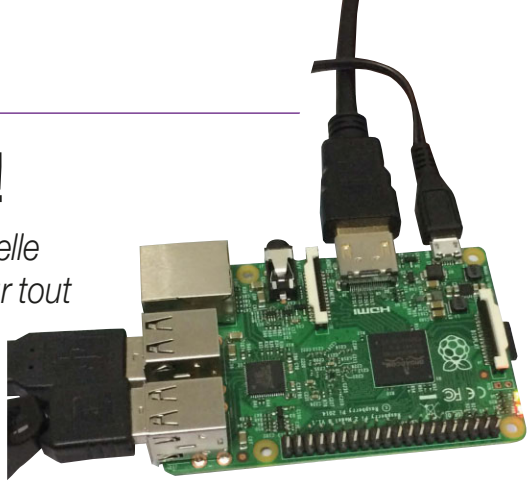
*Facile à utiliser – prêt pour le Cloud. 1&1 Serveur Cloud : 1 mois d'essai gratuit, sans demande de vos coordonnées bancaires, puis à partir de 15,84 € HT/mois (19,01 € TTC) sur la base de la configuration minimum. Pas de frais de mise en service. Conditions détaillées sur 1and1.fr.
Windows et le logo Windows sont des marques déposées de Microsoft aux Etats-Unis et dans les autres pays. Linux est une marque déposée de Linus Torvalds aux Etats-Unis et dans les autres pays.

1&1

1and1.fr

Raspberry Pi 2 : c'est du sérieux !

Raspberry Pi est devenue une icône de l'informatique bidouille nouvelle génération. Sur une petite carte, les créateurs font tenir un ordinateur tout entier. Depuis l'apparition de la Pi, ce sont 5 millions d'exemplaires qui ont été vendus. La v2 s'est déjà écoulée à plus de 500 000 exemplaires... Les nombreuses nouveautés ont beaucoup aidé !



François Tonic
Programmez !

Nous avons pu trouver la Pi 2 sur RS (distribution) au prix de 36 € (un peu plus cher que les précédents modèles), ce qui est un excellent prix. La boîte est toujours minimaliste : la carte et c'est tout. À vous d'acheter tout le reste (carte SD, alimentation, clavier, souris, etc.). Pour 36 €, on ne peut pas tout avoir...

Différence entre la Pi1 et la Pi2

Les deux versions possèdent les mêmes dimensions, les bords de la Pi2 sont arrondis et les trous de fixation ont changé de place. Si vous pensiez remplacer votre précédente carte, il faudra adapter le support: son organisation a été assez largement modifiée, par exemple, le port micro-USB a changé de place.

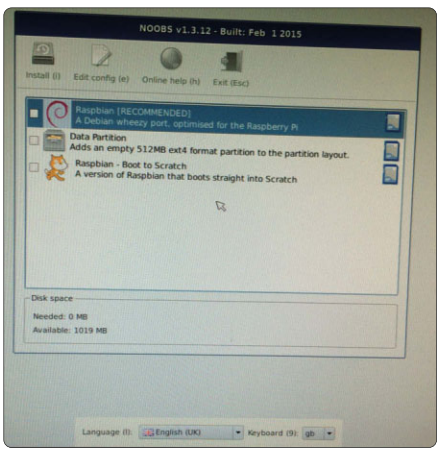
Les principaux éléments

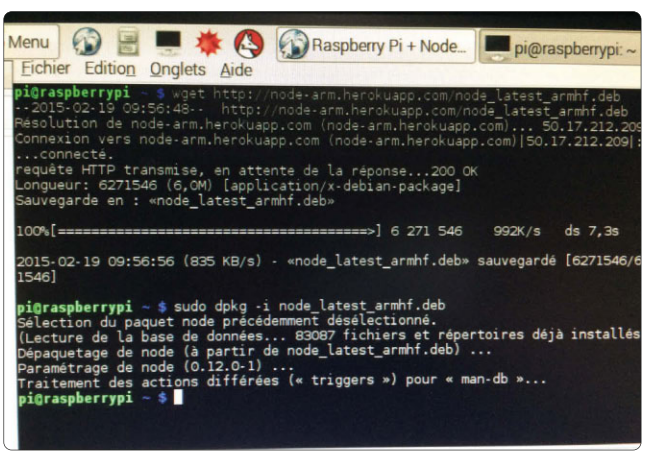
	Pi 1	Pi 2
USB	2 ports	4 ports
Réseau	1 Ethernet	1 Ethernet
Processeur	Broadcom (ARMv6) 700 MHz 256 mo (RAM), 1 cœur	Broadcom (ARMv7) 900 MHz 1 Go (RAM), 4 cœurs
Vidéo	HDMI RCA Video ports CSI Camera / DSI display	HDMI ports CSI Camera / DSI display
Extension (GPIO)	26 broches	40 broches
Carte	SD	micro-SD
Système	Linux	Linux Windows 10 (annoncé)

La compatibilité sur les extensions matérielles est assurée par le GPIO dont les 26 premières broches sont identiques aux modèles Pi 1. Incontestablement, à l'usage, le nouveau processeur et la mémoire vive de 1 Go améliorent les performances globales de la carte et logiciels installés, à condition qu'ils soient adaptés à la v2.

On installe !

Étape 1 : tout d'abord, trouvons dans notre caverne technique une micro-SD de capacité suffisante (8 Go recommandés). Formatez-la avec un outil de type SDFormatter. Plusieurs images systèmes sont disponibles sur le site officiel : Raspbian, Ubuntu Core, serveurs XBMC, Pidora, Risc OS. Décompressez l'image système puis mettez-la sur la SD. Attention : certains systèmes sont compatibles Pi 1 et 2, d'autres uniquement Pi 1 ou Pi 2. Noobs est un outil graphique pour faciliter l'installation du système sur votre Pi.





Étape 2 : il vous faudra un clavier, une souris, un câble d'alimentation micro-USB, un câble vidéo. Branchez le tout.

Étape 3 : on peut maintenant lancer un premier boot et terminer l'installation du système et sa mise à jour. Comme nous avons installé Noobs sur la SD, la Pi démarre directement dessus. L'outil est pratique et très ergonomique. Vous pouvez installer plusieurs systèmes en même temps. Nous avons opté pour Raspbian et OpenELEC Pi2.

OpenELEC propose un assistant de configuration très convivial et rapide. L'interface graphique est très fluide. Pour utiliser le Pi 2 comme un ordinateur desktop c'est une bonne solution. Pour les utilisateurs avancés, pour le développement, la Raspbian reste la référence. Le système répond bien, cependant attention aux connexions réseaux, en Ethernet vous restez en base 10/100, ce qui pèse sur les débits.

Et après ?

Comme toujours avec ces boards, c'est à vous d'imaginer et de trouver l'usage. Il peut parfaitement se transformer en mini-serveur, en serveur média ou pour prototyper un objet connecté, faire de la domotique, etc. Le processeur et la mémoire vie le rendront bien plus intéressant. Sa compatibilité GPIO sur les cartes d'extensions et les accessoires de la Pi 1 (attention tout de même aux librairies tierces) permet une transition en douceur. Il ne fait aucun doute que la Pi 2 va permettre l'apparition de nouveaux shields et capteurs. Adafruit et SeeedStudio (constructeur des excellents capteurs Grove) proposent des composants et kits complets pour la Pi.

Par contre, on peut regretter que la documentation (notamment technique) soit toujours aussi faible et pas toujours facile à trouver. Heureusement que la communauté propose de nombreux tutoriels, mais le constructeur doit absolument investir dans la documentation.

Et le Wi-Fi ?

Par défaut, vous disposez uniquement d'Ethernet. Le réseau sans fil sera cependant utile pour de nombreux usages. Pour ce faire, vous pouvez rajouter un dongle Wi-Fi, un des plus utilisés est le EW-7811Un (Edimax). Il faudra jouer du terminal et de la configuration pour rendre opérationnel le matériel. Le tutoriel suivant est très bien et facile à suivre (perso, je le préfère à l'interface graphique proposée par Raspbian) :
http://raspberrypihq.com/how-to-add-wifi-to-the-raspberry-pi/

Offre spéciale abonnement !

et aussi...

PROGRAMMEZ!
le magazine du développeur

devolo
The Network Innovation

Pour un abonnement 2 ans, Devolo et Programmez! vous offrent un kit complet CPL dLAN 550
d'une valeur de 79,90 €

Pour seulement 94,90 €

(au lieu de : 158,90 €, abonnement 2 ans / 22 numéros : 79 € + 15,90 € de frais logistiques et postaux)

Attention :
- *cette offre est strictement limitée à la France Métropolitaine et à la Corse.*
- *quantité limitée, jusqu'à épuisement des stocks. cette offre est susceptible de s'arrêter à tout moment.*

2 ans 22 numéros
94,90€

Spécial étudiant
39€(*)
1 an 11 numéros

1 an 11 numéros
49€
seulement (*)

2 ans 22 numéros
79€
seulement (*)

(*) Tarifs France métropolitaine

ABONNEZ-VOUS !

Toutes nos offres sur www.programmez.com

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 € (au lieu de 65,45 €, prix au numéro)

☐ ***Abonnement 2 ans au magazine :*** 94,90 € (au lieu de 158,90 €, kit CPL dLAN offert)

☐ ***Abonnement spécial étudiant 1 an au magazine :*** 39 €
Photocopie de la carte d'étudiant à joindre

Offre abonnement + clé USB Programmez!
☐ 1 an (11 numéros) + clé USB : **60 €**
Clé USB contenant tous les numéros de Programmez! depuis le n°100, valeur : 29,90 €
Tarifs France métropolitaine

☐ M. ☐ Mme ☐ Mlle Entreprise : Fonction :

Prénom : Nom :

Adresse :

Code postal : Ville :

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

E-mail : @

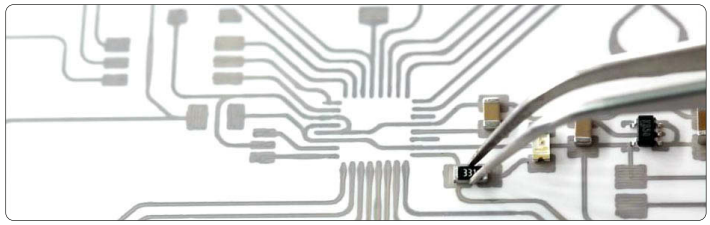
☐ Je joins mon règlement par chèque à l'ordre de Programmez !

☐ Je souhaite régler à réception de facture

** Tarifs France métropolitaine*

PROG 184
***Offre limitée**, valable jusqu'au 28 Avril 2015*

Circuits Scribe, interview



Dans Programmez 182, nous vous avions présenté l'étonnant Circuits Scribe. Ce mois-ci, petite interview d'un des co-fondateurs, Michael Bell.

Pourquoi avoir créé ce projet ?

Michael : Nous avons créé ce projet à partir de notre travail de doctorat à l'université de l'Illinois et à Harvard, autour des encres conductrices. Nos encres peuvent être utilisées pour des usages tous publics et Analisa Russo, un des co-fondateurs, est venue avec l'idée de mettre cette encre dans un stylo de type rollerball. A partir de là nous avons continué à innover sur l'encre puis à concevoir les modules, les tutoriels et les kits.

De nombreux kits et modules sont régulièrement en rupture de stock, est-ce le signe d'un grand succès ?

Michael : Oui, Circuit Scribe est un grand succès. Nous n'avons pas anticipé de si bonnes ventes sur les modules individuels. Nombre d'entre eux sont en rupture avant même de pouvoir les réapprovisionner. Nous avons le même succès sur les différents kits. Le succès a été immédiat dès notre campagne Kickstarter. Nous demandions 80 000 $, à la fin de la campagne, nous avions récolté 674 000 $!

Les modules sont disponibles sur Github. Pourquoi avoir choisi l'open hardware ?

Michael : Effectivement, tous nos modules sont Open Source Hardware (ou OSHW). Depuis que nous essayons d'enseigner l'électronique, il n'est pas nécessaire de cacher comment les différents éléments sont réalisés. Les étudiants et les enseignants devraient être en mesure d'expliquer comment chaque module fonctionne, les modifier, et les hacker.

Quand on utilise Circuits Scribe, il n'est pas toujours facile de trouver la documentation et des exemples avancés, par exemple sur les possibilités d'utiliser Circuits avec une Arduino : allez-vous améliorer cette partie du projet ?

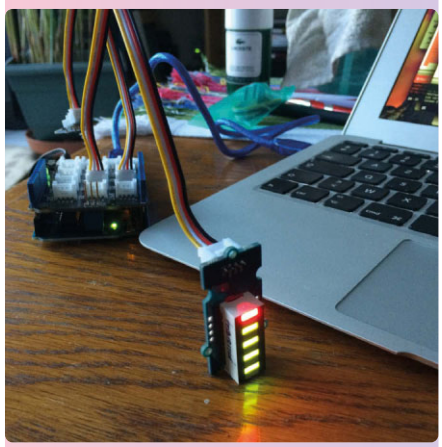
Michael : Nous avons un partenariat avec Autodesk qui développe un simulateur en ligne pour nos modules et l'encre (http://123d.circuits.io/circuitscribe/). Des milliers de circuits ont été conçus mais la fonction d'exploration n'a pas été rendue disponible. Pour aider les professeurs à utiliser nos kits et modules en classe, nous avons créé des cours spécifiques. Nous travaillons un contenu pour les débutants et le grand public, le travail est en cours, mais certains éléments sont disponibles sur : https://projectignite.autodesk.com/app/project/1/Sketching-Simple-Circuits-Basic-Kit-Edition-/overview/
Nous travaillons aussi sur un module Arduino depuis quelques mois. Ce n'est que récemment que nous avons pu nous concentrer sur la documentation et de nouveaux modules, et non plus uniquement sur les commandes...

Allez-vous créer des API, des outils, des SDK pour étendre les possibilités de Circuits Scribe ?

Michael : Nous travaillons sur différents outils, notamment pour améliorer l'utilisation de notre stylo. Mais effectivement, nous ne travaillons pas assez sur la partie logicielle et la disponibilité d'un SDK. Mais tout ceci pourrait changer avec Arduino. Nous prévoyons d'être totalement ouverts (vers cette plateforme, NDLR).

D.R.

Intel Galileo & capteurs pour Arduino : attention à la compatibilité



La carte Galileo est compatible Arduino : on peut utiliser des modules et shields pour Arduino, avec taux de compatible élevé. Il peut donc poser problème. Sur notre projet « tasse connectée pour geek », présentée aux TechDays, nous avions utilisé une Galileo gen2 et principalement le kit Grove (fabriqué par Seeed). Lors de la refonte du projet, version vNext, il a été impossible d'utiliser correctement la barre 10 leds Grove. Le même code, et la même librairie, fonctionnent parfaitement sur une carte Arduino Uno. Même remarque avec l'accéléromètre Grove 16g. Les données du capteur n'étaient pas correctes sur la Galileo. Conclusion : pour plus de stabilité, nous migrons vers une board Arduino Uno.

Chromebook : développer sur Linux pour – 200 €

Pour se lancer dans l'aventure du développement logiciel, on fait comment ? Il faut s'équiper en matériel et là, ça peut coûter très cher. Des solutions alternatives existent. Lesquelles ? On en parle ici.

Christophe PICHAUD
.NET Rangers by Sogeti
Consultant sur les technologies Microsoft
christophepichaud@hotmail.com
www.windowscpp.net

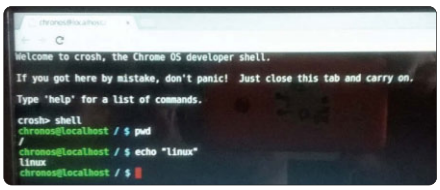
Vous avez entendu parler des Google ChromeBook ? C'est un terminal à partir de 199 € et il est possible de le débrider. Avoir un ChromeBook pour le débrider, ça sert à quoi ? A accéder au système Linux qui est dessous ! Et une fois qu'on a un shell sous linux, on peut faire des choses. Par exemple, installer un compiler GCC, G++ ou même eclipse pour faire du Java ou eclipse-cdt pour faire du C/C++. Bref, on va se servir du ChromeBook comme d'un device léger mais en mode développeur. Et comment ?

Téléchargeons les scripts

Il faut maintenant télécharger un script magique qui se nomme crouton disponible sur goo.gl/fd3zc. Ce script va nous permettre de choisir la distribution Linux de notre choix avec l'environnement de fenêtrage désiré. Le script va se mettre dans le répertoire Downloads de l'utilisateur courant.

Passer en mode développeur

Premièrement, il faut débrider le ChromeBook. La procédure est simple : il suffit de faire Esc + Refresh + Boot Button. Le périphérique va redémarrer et indique que la vérification de l'OS est Off. Il faut faire CTRL ^D pour annuler le message d'erreur. A partir de là, le périphérique mouline pendant 5 à 6 minutes puis, vous êtes en mode développeur. Lorsque vous êtes sur le ChromeBook, appuyez sur CTRL + ALT + T, vous tombez sur une console indiquant que vous êtes sur le Chrome OS developer shell. Entrez « shell ». Et là, bingo, vous êtes sur Linux. A nous l'aventure !



Installer un gestionnaire de fenêtres

Il faut maintenant installer un environnement graphique pas trop lourd. Prenons Xfce. L'installation se fait en mode admin (via sudo) et la commande est la suivante :

```
« sudo sh -e ~/Downloads/crouton -t xfce »
```

C'est le script crouton qui fait le paramétrage. Wait & See.... Les packages se téléchargent les uns derrière les autres et on se demande si ça va vraiment marcher. Au bout de quelques minutes, le script va vous demander un nom d'utilisateur et un mot de passe. Une fois arrivé, vous tapez : « sudo startxfce4 » et voilà ! Fig.1.

Il est aussi possible d'installer un autre gestionnaire de fenêtres comme lxde :

```
« sudo sh -e ~/Downloads/crouton -r trusty -t lxde »
```

Il est possible de charger d'autres desktop comme unity via

```
« sudo sh -e ~/Downloads/crouton –u -t unity-desktop »
```

Installer les outils de développement

Bref, vous êtes sous Linux et donc vous pouvez faire ce qu'il vous plait.

Voici mes commandes préférées pour avoir un environnement de développement parfait :

Installation des compilateurs C et C++ de GCC :

```
sudo apt-get install gcc
sudo apt-get install g++
```

Installation de CMake et LLVM:

```
sudo apt-get install cmake
sudo apt-get install clang
```

Installation de l'IDE Eclipse en mode Java et C/C++:

```
sudo apt-get install eclipse
sudo apt-get install eclipse-cdt
```

L'installation de GCC, le compilateur C, G++ le compilateur C++, cmake et Clang (avec LLVM) puis l'IDE Eclipse et sa version pour le C/C++ Eclipse-cdt permet de combler les besoins de développement Fig.2. SI vous avez des besoins d'une suite bureautique, il suffit d'installer LibreOffice :

```
sudo apt-get install libreoffice
```

Et voilà un ChromeBook devient une véritable station de travail pour le développement logiciel sous Linux. Pour plus de confort, installons firefox et GIMP (logiciel graphique) via :

```
sudo apt-get install firefox
sudo apt-get install gimp
```

Si vous souhaitez retourner en mode ChromeBook, il suffit de se déconnecter (logout) et c'est tout ! ChromeBook OS est en fait comme un Windows Manager qui tourne au-dessus de Linux.

Installer Mono pour faire du C#

Pour faire du mono (le .NET sous Linux) il faut ajouter de 2 commandes :

```
sudo apt-get install mono-complete
sudo apt-get install monodevelop
```

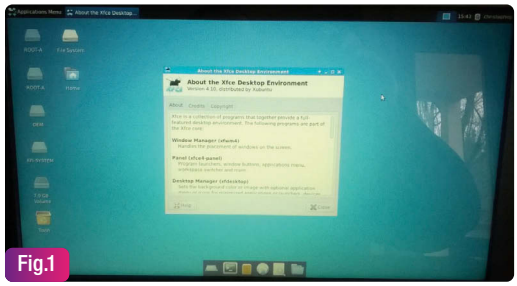


Fig.1

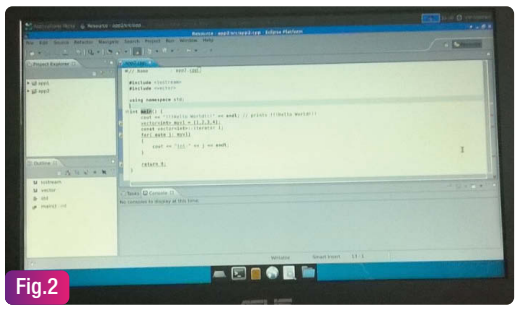


Fig.2

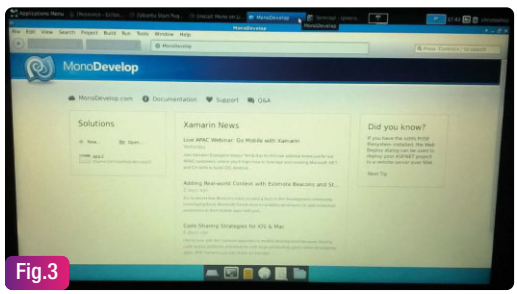


Fig.3

Le lancement de l'IDE Mono se fait en tapant monodevelop: Fig.3.

Conclusion

L'accès au socle Linux du ChromeBook démontre que nous avons affaire à un socle Linux standard. Les packages s'installent avec sudo apt-get. Récapitulons la liste des langages disponibles dans la distribution et suite aux diverses commandes exécutées ci-dessus :

- C, C++
- Java
- Perl, Python
- Mono C#

La gestion des packages évite d'installer des choses à la main ou d'aller dans des fichiers de configurations exotiques. Un « sudo apt-get » garantit que le système restera stable. Pour 199€, le chrome book est un formidable matériel car c'est un laptop doté de 4GB avec un processeur Intel Celeron dual-core. Il y a 32 Go de stockage ce qui est suffisant pour développer pour s'amuser ou pour apprendre. Le ChromeBook ne possède qu'un écran de 11' mais c'est le parfait compagnon pour la mobilité.

NoSQL

Apparue à la fin des années 90, la notion de NoSQL n'a commencé à s'étendre qu'après 2010. S'opposant au principe des bases relationnelles, le NoSQL ne vise pas à remplacer mais plutôt à compléter les SGBDR.

Jean-Noël Durand
Epitech

NoSQL (litteralement "Not only SQL") est une catégorie de Système de Gestion de Base de Données (SGBD) qui laisse de coté le principe des bases relationnelles. On ne parle donc plus de tables ; les données, quant à elles, ne sont plus manipulées par SQL.
NoSQL est principalement utilisé par les géants du Web tels que Google, Amazon, Facebook... pour répondre à des problématiques en termes de charges et de volumétrie. Alliant performance et simplicité, cette solution permet de traiter d'énormes quantités de données sans réelle structuration relationnelle. On retrouve alors l'utilisation de NoSQL dans tout ce qui est stockage de logs, statistiques etc. Ceci là où les SGBDR exigeraient du matériel coûteux ainsi que des compétences peu répandues pour optimiser le traitement de telles quantités de données.

5 Choses à savoir avant tout !

Nous venons de parler des géants du Web mais le NoSQL est toutefois présent aussi dans les entreprises de taille moyenne. De plus, il est open source. Tout le monde peut donc y accéder, sauf que cette technologie est encore très jeune et du même coup encore peu répandue. Enfin, à l'heure actuelle on ne trouve pas de solutions NoSQL embarquées et il y a très peu d'hébergeurs qui les proposent.
Certes NoSQL n'est pas relationnel mais cela ne veut pas dire que l'on ne doit pas réfléchir un minimum à son modèle de données. Un bon modèle de données offrira de bonnes performances.
NoSQL ne remplace pas SQL, c'est un complément aux bases relationnelles et répond à des problématiques spécifiques.
Pour saisir l'engouement qui se développe autour des solutions NoSQL, nous devons d'abord comprendre pourquoi les bases relationnelles sont les plus répandues.

Petit rappel sur les bases relationnelles

Les données sont structurées en tables, reliées par des clés, le tout suivant un schéma fixe. Nous pouvons alors spécifier ce que l'on cherche à travers des requêtes SQL. Le système de bases de données relationnelles se charge de convertir cette requête en une séquence d'opérations sur les tables telles que des jointures, des sélections, des suppressions etc. Le SGBDR permet alors d'alléger le code applicatif en se chargeant de mettre en place le schéma de recherche des données tout en séparant la structure des données de l'application. Le plus important est que le SGBDR permette de garder un certaine cohérence des données en appliquant des contraintes d'intégrité en fonction du schéma prédéfini par l'utilisateur et en protégeant les données des accès concurrentiels. Cependant on retrouve certaines difficultés à l'utilisation de cette solution comme celle de répartir les traitements d'un SGBDR sur un grand nombre de nœuds serveurs. Le nombre d'échanges entre ces derniers devient alors trop important et les performances s'en trouvent réduites. De plus les SGBDR ne sont pas réellement faits pour être utilisés avec de la programmation orientée objet qui manipule des grappes de données hiérarchisées. Sauvegarder ce type de données dans des tables devient alors d'une réelle difficulté. On pense alors aux bases orientées objets...

Le NoSQL à la rescousse

Puis, avec l'arrivée des sites e-commerce et des réseaux sociaux, les besoins ont changé. Les exigences de montée en charge et de disponibilité sont désormais beaucoup plus importantes que les exigences d'intégrité qui font l'utilité des bases relationnelles.
Pour des sites e-commerce, une indisponibilité ne serait-ce que de quelques instants signifie une perte plus ou moins importante en fonction du périmètre de l'entreprise, on laisse donc de côté l'isolation des transactions au profit des performances. Sachant aussi que les données valorisées au sein d'un système d'information ne sont souvent pas structurées, ce sont des logs, des fichiers Excel, des statistiques etc. ; des données pour lesquelles les SGBDR n'apportent aucune aide. C'est alors que l'on retrouve les solutions NoSQL et leurs caractéristiques très intéressantes pour ce genre de problématiques comme la possibilité de facilement distribuer le stockage et le traitement des données sur un grand nombre de machines ; d'où le fait que cette solution soit dépourvue de schémas et qui plus est, soit non relationnelle, ceci sans compter que le NoSQL est open source.

Côté technique

On retrouve plusieurs bases NoSQL. Elles peuvent se répartir en 4 groupes :

- Clé/valeur : c'est la base NoSQL la plus simple, les données sont effet simplement représentées par un couple clé/valeur, la valeur pouvant être n'importe quoi... Cela signifie que la valeur n'a techniquement aucune structure, cette dernière étant accessible grâce à la clé. On peut l'assimiler à une table de hachage. Le code applicatif interroge directement la BD et se résume aux opérations PUT, GET et DELETE. Cette méthode est la plus simple et aussi la plus rapide.
- **Orientée colonnes :** ce type de base ressemble à un SGBDR classique, cependant, dans une base relationnelle, le nombre de colonnes est fixe tandis qu'en NoSQL, ce dernier est dynamique. On peut donc avoir des colonnes différentes pour chaque ligne et avoir une certaine flexibilité quant aux données à stocker.
- **Orientée documents :** cette solution s'appuie sur un système clé/valeur mais ici, les valeurs sont stockées dans un fichier de type JSON, XML, ou autre... On peut alors, à l'aide d'une seule clé, accéder à un ensemble de données structurées très facilement là où un SGBDR classique demanderait plusieurs jointures.
- **Orientée graphes :** l'utilisation principale de ce type de base est le stockage des informations des réseaux sociaux. Cette solution permet de stocker et de manipuler des données liées par des relations variables. Dans le cadre d'un réseau social, par exemple, la navigation entre les entités se fait beaucoup plus efficacement que sur un SGBDR.

Ci dessous un schéma représentant schématiquement ces 4 types de bases : Fig.1. Nous allons nous attarder sur deux bases de types NoSQL, les bases MongoDB et Cassandra.

MongoDB

MongoDB est une base orientée document permettant de manipuler des objets au format BJSON sans schéma relationnel. Les données sont donc des documents que l'on peut comparer aux enregistrements d'une base de donnée relationnelle, ces derniers sont enregistrés dans ce que

Maker Faire® Paris

2 & 3 mai - Foire de Paris
Paris expo Porte de Versailles

Démonstrations

Conférences

Ateliers

Avec le parrainage
du ministère
de la Culture
et de la
Communication

Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

Ministère
Culture
Communication

Make:

leFabShop

LEROY MERLIN
...et vos envies prennent Vie!

FOIRE DE PARIS
LE MEILLEUR EST ICI
29 AVRIL - 10 MAI 2015

l'on appelle des collections (comparables aux tables dans un SGBDR). L'absence de schéma permet une grande liberté dans les enregistrements, le seul champ commun et obligatoire est le champ clé "id", les autres champs peuvent être différents d'un enregistrement à l'autre et ce, même au sein d'une même collection.
Là où une table SQL ressemblerait à ça :

ID	Article	Prix
1	poire	10
2	pomme	5

Une collection MongoDB pourrait ressembler à ça.
(on notera la présence d'un champ "poids" supplémentaire dans le deuxième enregistrement) :

```
{
"_id": ObjectId("4sdf54sd65464zaefz5d5sc7"),
"Article": "poire",
"Prix": "10"
},
```

```
{
"_id": ObjectId("4sdf54sd65464zaefz5d5sc8"),
"Article": "pomme",
"Prix": "5",
"poids": "2"
},
```

On remarque le champ obligatoire "_id" qui est un index unique généré automatiquement par MongoDB servant à identifier le document.
Ces documents sont structurés de manière très simple, ils se composent de clefs et de valeurs séparées d'un signe ":". La valeur peut être du texte, un nombre, une image, ou une imbrication de documents (voir ci dessous le champ "données" qui contient un document de champs "Prix" et "Poids").

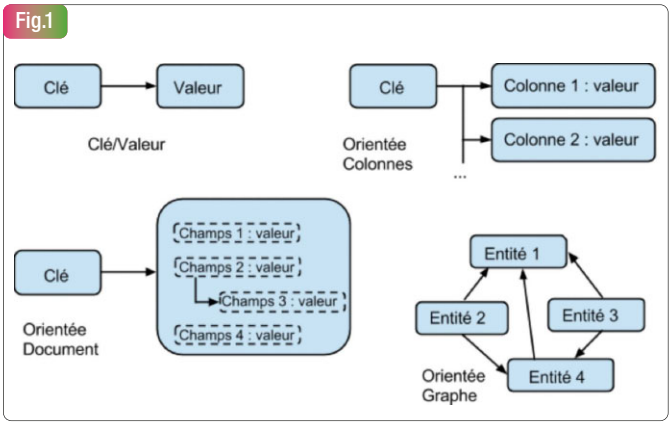
```
{
"_id": ObjectId("4sdf54sd65464zaefz5d5sc7"),
"Article": "poire",
"Données":
            {
             "Prix": "10",
             "Poids": "2"
            }
            },
```

Pour ce qui est de la portabilité, MongoDB est utilisable nativement avec les principaux langages via les drivers fournis, à savoir : Java, .NET, Ruby, PHP, JavaScript, node.js, Python, Perl, PHP, Scala etc.
Outre sa fonction de base de données, MongoDB peut être utilisé comme système de fichiers dans un but de réplication/répartition des fichiers sur un ensemble de serveurs afin de disposer d'une certaine tolérance aux pannes.

Cassandra

Cassandra est une base NoSQL orientée colonnes et son architecture se présente sous la forme de " nœuds" (chaque nœud est une machine sur laquelle tourne une instance de Cassandra) qui sont eux même regroupés en "clusters" (groupe d'au moins deux nœuds) qui sont quant

Fig.1



à eux regroupés en data center (possibilité de répliquer les data centers). Tous les nœuds sont "égaux", on ne retrouve ni machine maître, ni machine esclave. Ce modèle permet d'assurer la réplication des données entre tous les nœuds, ce qui, dans le cas où un nœud serait en panne, permet de toujours avoir accès aux données à travers d'autres nœuds. Le remplacement, l'ajout ou la suppression de nœuds peut donc être effectué sans provoquer l'indisponibilité du service. Dans le modèle de Cassandra une colonne se présente sous la forme Nom / Valeur / Timestamp (ce dernier détermine la mise à jour la plus récente).
Le nom se limite à un poids de 64Ko tandis que le champ Valeur peut contenir jusqu'à 2 Go de données. Cependant la valeur n'est pas obligatoire et il n'est pas rare de retrouver des champs Nom qui présentent le nom suivi de la valeur.
La valeur peut aussi contenir "plusieurs valeurs" comme par exemple un ensemble de chaînes de caractères.

Fruits
Article:Poire Poids:2 Prix:10
Timestamp

Les colonnes sont regroupées en lignes. Une ligne est un ensemble de colonnes identifiée par une clé. Tout comme le nom d'une colonne, une clé peut contenir au maximum 64Ko de données tandis qu'une ligne peut contenir plusieurs milliards de colonnes.
Le nombre de colonnes n'est pas forcément le même pour chaque ligne et les colonnes ne sont pas forcément les mêmes, comme illustré ci dessous :

Ligne 1	Article	Poids	Prix
	Poire	2	10

Ligne 2	Article	Origine
	Poire	France

Ces lignes peuvent êtres regroupées en familles de colonnes (que l'on peut comparer à une table dans un SGBDR). On distingue deux types de familles :
Les familles de colonnes statiques : colonnes définies lors de la création ou la modification de la famille de colonnes.
Les familles de colonnes dynamiques : colonnes définies lors de la création ou la modification d'une ligne.

Famille	Ligne 1	Article	Poids	Prix
		Poire	2	10
	Ligne 2	Article	Origine	
		Poire	France	

Les familles de colonnes peuvent quand à elles être regroupées en keyspace (comparable au schéma d'un SGBDR).

Cassandra : partez sur une bonne base !

Cassandra est une base NoSQL orientée colonne et créée à l'origine par Facebook en s'appuyant sur deux papiers de recherche : BigTable de Google, DynamoDB d'Amazon.
Le projet a été libéré en 2008 et inclus à la fondation Apache dès 2009. Il est maintenant un top level project utilisé par un grand nombre d'entreprises. Nous pouvons citer par exemple NetFlix, Apple ou encore le CERN.

Matthieu Nantern,
Consultant chez Xebia

Concepts de base

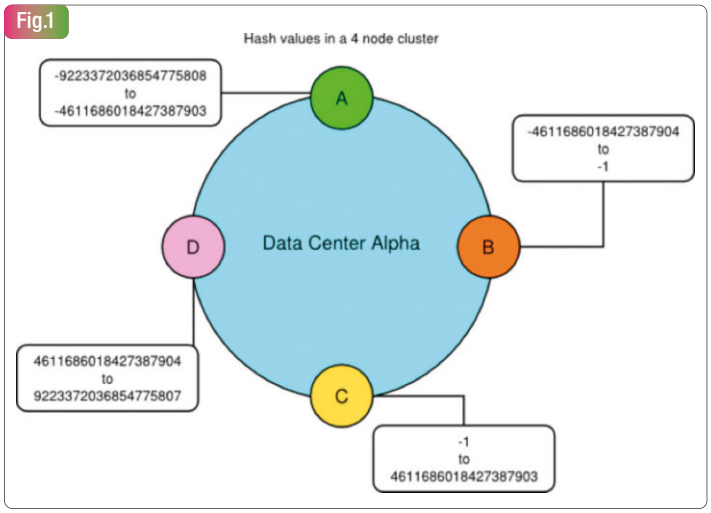
Dans ce chapitre, nous allons voir les principes de base de Cassandra, ceux qui en font une base NoSQL de plus en plus utilisée. La première caractéristique est sa scalabilite linéaire. Si vous avez besoin de servir plus de requêtes, il vous suffit d'ajouter plus de serveurs, pas besoin de mettre en place des mécanismes de réplication (compliqués) ou de re-écrire du code. C'est aussi valable pour la quantité de données : si vous voulez stocker plus de données il suffit d'ajouter des machines, et les données seront réparties sur votre cluster automatiquement.
Cette fonctionnalité est très intéressante dans le cas où vous ne connaissez pas la charge que vous aurez dans 1 ou 5 ans. Il existe à l'heure actuelle en production des clusters Cassandra allant de 3 machines à plus de 1000 (Netflix ou Apple) et gérant des péta-octets de données. La montée en charge est facilitée par une deuxième caractéristique de Cassandra : chaque serveur a le même rôle (architecture "masterless"). Il n'y a pas de serveur principal ou de serveur gérant la coordination. Un serveur assure l'ensemble des fonctionnalités de Cassandra. Ainsi, monter en charge revient toujours à ajouter le même type de serveur. De plus, Cassandra étant une base facile à installer (un jar et deux fichiers de configuration), elle est très bien acceptée par les équipes opérationnelles. Cassandra est également une base haute disponibilité qui peut tolérer (suivant votre configuration) la perte de plusieurs nœuds sans difficulté. Pour cela, comme de nombreuses autres bases NoSQL, Cassandra s'appuie sur la réplication des données sur de multiples nœuds. La réplication (asynchrone) entre plusieurs datacenters est même prévue nativement dans Cassandra : ainsi même la perte d'un datacenter n'entraînera pas d'interruption de votre service ! Ces avantages entraînent quelques contraintes au niveau des requêtes pouvant être effectuées sur Cassandra, ceci étant dû à sa nature distribuée. En effet, il n'est pas possible de réaliser des transactions ou des jointures sur notre cluster. Néanmoins, une bonne modélisation et une dénormalisation adaptée des données permettent le plus souvent de pallier à ces limitations.

Distribution des données

Chaque noeud du cluster Cassandra est en charge de gérer une partie des données. Cette répartition est effectuée via un hash de la clé de partition. Une clé primaire peut être composée d'une ou plusieurs colonnes (dans ce cas il s'agit d'une clé composée).
Le premier élément de la clé primaire, appelé clé de partition, détermine sur quel noeud du cluster la donnée sera stockée.
Par exemple, prenons les clés de partitions et les hash suivants :

Clé de partitionnement	Hash
jim	-2245462676723223822
suzy	1168604627387940318

Fig.1



Ici, la donnée ayant comme clé de partitionnement "jim" sera placée sur le noeud A (qui gère l'intervalle dans lequel est compris le hash de la clé) et la donnée "suzy" sur le noeud C Fig.1.
La réplication a ensuite lieu afin de copier les données sur d'autres noeuds afin d'assurer la haute disponibilité.
Les autres éléments d'une clé primaire composée, appelés "clustering columns" servent ensuite à ordonner les données sur un même noeud.
Voici un exemple :

```
CREATE TABLE grades (
 grade_id uuid,
 student_id  uuid,
 obtained_at  timestamp,
 grade   text,
 subject  text,
 PRIMARY KEY (student_id, obtained_at, grade_id)
);
```

Dans ce cas, la clé de partitionnement est "student_id" et les clustering columns sont "obtained_at" et "grade_id". Il est possible d'avoir plusieurs "clustering columns" pour une même table.
Les données appartenant à cette table sont placées sur un noeud en fonction de la colonne student_id puis triées sur le disque en fonction des colonnes "obtained_at" puis "grade_id".
Ainsi pour une table représentant les notes d'un élève, toutes celles-ci seront gérées par une même machine et classées en fonction de leur date d'obtention (de la plus ancienne à la plus récente) puis de leur identifiant.
Un dernier point important à souligner : il est possible de spécifier pour chaque requête le niveau de cohérence requis.
Comme tout système distribué, Cassandra est régi par le théorème CAP (voir encadré) et se place dans les domaines AP (Availability et Partition tolerance).

Le théorème CAP également connu sous le nom de théorème de Brewer montre qu'il est impossible pour un système distribué de satisfaire de manière simultanée les trois contraintes suivantes :

- Cohérence ("Consistency") : tous les noeuds du système voient la même donnée au même moment,
- Disponibilité ("Availability") : chaque requête recevra une réponse (que cela soit un succès ou un échec),
- Résistance à la partition ("Partition tolerance") : le système continue de fonctionner malgré des défaillances pouvant aller jusqu'à la séparation du cluster en sous-système.

Les bases relationnelles sont de type CA (Cohérence et Disponibilité), les bases NoSQL comme MongoDB ou HBase sont de type CP tandis que Cassandra ou Riak sont de type AP.

Il existe plusieurs types de cohérence utilisables pour chaque requête. Les plus communément utilisés sont :

- ALL : le cluster attend la réponse de tous les noeuds devant écrire ou lire la donnée avant de répondre au client,
- ONE : le cluster attend seulement la réponse d'un des noeuds avant de répondre au client,
- QUORUM : le quorum est défini comme la majorité absolue des réplicas. Si la donnée est répliquée 3 fois, alors le cluster attendra la réponse de 2 noeuds avant de répondre au client. Si la donnée est répliquée 4 fois alors le quorum est constitué de 3 machines.

Le choix de la cohérence est important et doit être déterminé en fonction des cas métiers et de la performance attendue. Le niveau ONE est le plus performant mais se fait au détriment de la fraîcheur des données, tandis que le niveau ALL offre la plus forte cohérence mais au détriment de la haute disponibilité. Pour que cela soit plus parlant, prenons un exemple : nous répliquons les données 3 fois au sein de notre cluster pour notre table "grades" définie ci-dessus.

Cas 1 : On écrit et on lit avec un niveau de cohérence "ONE": Fig.2.
Si un client écrit la valeur B sur l'un des noeuds et qu'un autre client lit la valeur alors que la réplication est toujours en cours il lira une ancienne valeur (A en l'occurrence). L'écriture tout comme la lecture sera extrêmement performante (on n'attend la réponse que d'un seul serveur) mais avec le risque de ne pas toujours avoir la donnée la plus récente.

Cas 2: On écrit avec le niveau "ONE" et on lit avec le niveau "ALL" Fig.3.
Ici, on conserve la performance en écriture (un seul noeud doit persister la donnée avant de répondre au client) et on obtient toujours la donnée la plus fraîche en lecture. Cependant, ceci a un coût non négligeable, le cluster doit attendre d'avoir la réponse des trois noeuds avant de répondre au client; en plus, si l'on perd un serveur, il ne pourra plus honorer la lecture (car Cassandra attend la réponse de 3 serveurs avant de répondre au client).

Cas 3: On écrit et on lit avec le niveau "QUORUM" Fig.4.
Ce cas représente un bon compromis entre la performance et la disponibilité des données. Les écritures sont un peu moins performantes que précédemment (le cluster attend deux réponses au lieu d'une dans les cas précédents), mais les lectures ont l'assurance d'obtenir la donnée la plus à jour et il est possible de perdre machine sans que cela affecte la disponibilité des données.

Installation

Passons maintenant à un peu plus de pratique ! Nous avons vu dans le chapitre précédent que la facilité d'installation de Cassandra était un de ses points forts, réalisons donc cette installation sur une Ubuntu 14.04. La façon la plus rapide est d'utiliser le dépôt de paquets mis à disposition par Datastax. Les seuls pré-requis sont un JRE en version 7 (celui d'Oracle est recommandé) et JNA pour les installations de production. On ajoute ensuite le dépôt community, la clé publique, puis on installe Cassandra après avoir mis à jour la liste de nos dépôts :

```
$ echo "deb http://debian.datastax.com/community stable main" | sudo tee -a /etc/apt/sources.list.d/cassandra.sources.list
$ curl -L http://debian.datastax.com/debian/repo_key | sudo apt-key add -
$ sudo apt-get update
$ sudo apt-get install cassandra=2.0.12
```

Et voilà, Cassandra est installé et démarré ! On va maintenant s'y connecter et créer notre keyspace :

```
$ cqlsh
Connected to trace_consistency at 1.2.3.4:9160.
[cqlsh 4.1.1 | Cassandra 2.0.9 | CQL spec 3.1.1 | Thrift protocol 19.39.0]
Use HELP for help.
cqlsh> CREATE KEYSPACE school WITH REPLICATION = { 'class' : 'SimpleStrategy', 'replication_factor' : 1 };
```

Comme nous n'avons qu'une machine dans notre cluster nous demandons à Cassandra de ne pas répliquer la donnée (paramètre 'replication_factor'). Le langage utilisé est proche du SQL et se nomme CQL ("Cassandra Query Language"). Nous avons maintenant une base, il ne nous reste plus qu'à l'utiliser dans notre code.

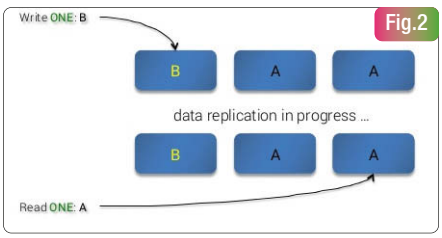
Premiers tests

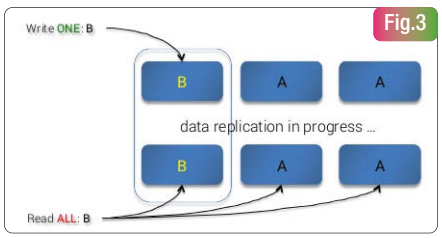
De nombreux drivers existent pour s'interfacer avec Cassandra, C#, Java, NodeJs, Ruby, Python, etc.
Pour la suite de cet article nous allons utiliser le driver Java développé par Datastax qui offre de nombreuses fonctionnalités :

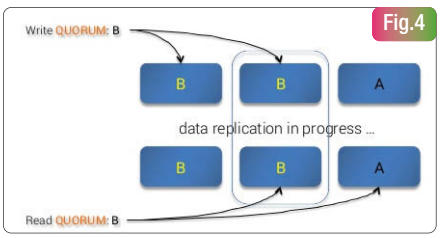
- Découverte de la topologie du cluster : le driver en se connectant à un cluster identifie l'ensemble des machines, et peut ainsi s'adresser directement au noeud contenant la donnée requêtée,
- Failover transparent : en cas de défaillance d'un nœud, le driver essaye automatiquement d'autres machines, puis tente de joindre à nouveau la machine défaillante en tâche de fond,
- Asynchrone : il est possible avec le driver de profiter des fonctionnalités asynchrones offertes par Cassandra,
- Compatible avec toutes les fonctionnalités de Cassandra.

Notre cas d'utilisation sera simple : nous allons nous connecter à notre machine Cassandra, créer une table, y insérer des données puis les lire avec le driver Java. La première étape est d'ajouter le driver à notre projet. Cela se fait simplement via une dépendance Maven :

```
<dependency>
 <groupId>com.datastax.cassandra</groupId>
```







```
<artifactId>cassandra-driver-core</artifactId>
<version>2.1.4</version>
</dependency>
```

Pour nous connecter à notre machine Cassandra, définissons deux méthodes :

- "connect" : permettant d'ouvrir la connexion vers un cluster Cassandra. Cette méthode utilise un Builder auquel on ajoute la liste des noeuds de notre cluster. Il n'est pas nécessaire de tous les indiquer, le driver prenant à sa charge la découverte complète du cluster.

```
public void connect(String node) {
  cluster = Cluster.builder()
      .addContactPoint(node)
      .build();
}
```

- "close" : fermant la connexion.

```
public void close() {
  cluster.close();
}
```

Ensuite, intégrez la classe complète avec un "main" permettant de l'exécuter :

```
package fr.xebia.cassandra;

import com.datastax.driver.core.Cluster;
import com.datastax.driver.core.Host;
import com.datastax.driver.core.Metadata;

public class SimpleClient {
 private Cluster cluster;

 public void connect(String node) {
   cluster = Cluster.builder()
       .addContactPoint(node)
       .build();
 }

 public void close() {
   cluster.close();
 }

 public static void main(String[] args) {
   SimpleClient client = new SimpleClient();
   client.connect("127.0.0.1");
   client.close();
 }
}
```

Cette classe va se connecter à notre machine puis fermer la connexion. Pas très utile... Ajoutons donc la création d'une table au keyspace "school" déclaré plus haut. Pour exécuter des requêtes, il nous faut obtenir une session du cluster déclaré ci-dessus. Cela se fait par le biais de l'appel de la méthode "connect" sur notre objet Cluster :

```
public void connect(String node) {
  cluster = Cluster.builder()
      .addContactPoint(node)
      .build();
  session = cluster.connect();
}
```

Une fois notre session obtenue, nous pouvons créer une table destinée à stocker les notes de nos élèves :

```
public void createSchema() {
  session.execute(
      "CREATE TABLE IF NOT EXISTS school.grades (" +
          "grade_id uuid," +
          "student_id uuid," +
          "obtained_at timestamp," +
          "grade text," +
          "subject text," +
          "PRIMARY KEY (student_id, obtained_at, grade_id)" +
          ");");
}
```

Puis insérez une donnée :

```
public void loadData() {
  session.execute(
      "INSERT INTO school.grades (grade_id, student_id, obtained_at, grade, subject) " +
      "VALUES (" +
          "756716f7-2e54-4715-9f00-91dcbea6cf50," +
          "5dad4ece-b8bb-46b0-ab5d-22cce12527dd," +
          "'2015-01-12T16:02:51Z'," +
          "'B+'," +
          "'chinese')" +
          ";");
}
```

Enfin, intégrez la classe complète :

```
package fr.xebia.cassandra;

import com.datastax.driver.core.Cluster;
import com.datastax.driver.core.Host;
import com.datastax.driver.core.ResultSet;
import com.datastax.driver.core.Row;
import com.datastax.driver.core.Session;

public class SimpleClient {
 private Cluster cluster;
 private Session session;

 public Session getSession() {
   return this.session;
 }

 public void connect(String node) {
   cluster = Cluster.builder()
       .addContactPoint(node)
       .build();
   session = cluster.connect();
 }

 public void createSchema() {
   session.execute(
       "CREATE TABLE IF NOT EXISTS school.grades (" +
           "grade_id uuid," +
           "student_id uuid," +
           "obtained_at time," +
           "grade text," +
```

```
            "subject text," +
            "PRIMARY KEY (student_id, obtained_at, grade_id)" +
            ");");
  }

  public void loadData() {
    session.execute(
        "INSERT INTO school.grades (grade_id, student_id, obtained_at, grade, subject) " +
        "VALUES (" +
            "756716f7-2e54-4715-9f00-91dcbea6cf50," +
            "5dad4ece-b8bb-46b0-ab5d-22cce12527dd," +
            "'2015-01-12T16:02:51Z'," +
            "'B+'," +
            "'chinese')" +
            ";");
  }

  public void close() {
    session.close();
    cluster.close();
  }

  public static void main(String[] args) {
    SimpleClient client = new SimpleClient();
    client.connect("127.0.0.1");
    client.createSchema();
    client.loadData();
    client.close();
  }
}
```

Nous allons maintenant lire la donnée que l'on vient d'insérer. Cette fois-ci nous utiliserons plutôt une requête préparée qui n'a besoin d'être parsée qu'une seule fois par le cluster, ce qui améliore les performances.
Trois étapes sont nécessaires :
1 - Déclarer la requête préparée,
2 - Associer des valeurs au paramètre,
3 - Exécuter la requête préparée.
Ces trois étapes sont réalisées via la méthode ci-dessous :

```
public void queryData() {
  // Etape 1
  PreparedStatement statement = getSession().prepare(
    "SELECT * from school.grades WHERE student_id = ?;");

  // Etape 2
  BoundStatement boundStatement = new BoundStatement(statement);
  boundStatement.bind(UUID.fromString("5dad4ece-b8bb-46b0-ab5d-22cce12527dd"));

  // Etape 3
  ResultSet rs = getSession().execute(boundStatement);
  for (Row row : rs) {
    System.out.println(row.getString("subject")+" - "+ row.getString("grade"));
  }
}
```

Un cas pratique

Maintenant que nous avons installé et vu les bases de la programmation avec Cassandra, passons à un cas pratique. Nous souhaitons modéliser des élèves qui disposeront de quelques champs :

- Un nom,
- Un prénom,
- Un email unique servant d'identifiant et permettant d'accéder à notre site,
- Une liste de cours.

Ces élèves auront des notes, et l'on souhaite pouvoir obtenir l'ensemble des notes d'un élève pour une plage de temps (un trimestre ou une année par exemple). La première étape et la plus importante est de modéliser correctement les tables sous Cassandra.

Modélisation des tables

Un point crucial avec Cassandra, trop souvent oublié, est la nécessité d'adapter la structure des tables de notre base aux requêtes que l'on souhaite faire. Une bonne modélisation est cruciale afin d'obtenir des requêtes performantes et de profiter des points forts de Cassandra. Ainsi les écritures sont peu coûteuses et de ce fait, il vaut mieux privilégier la duplication de données immutables afin d'éviter de faire une lecture supplémentaire.

Gestion des notes

Nous souhaitons maintenant gérer dans notre système les notes des élèves. Chaque objet Note est composé d'un identifiant technique, d'un élève à qui elle est liée, d'une valeur (la note), d'une matière et de la date d'obtention de la note. Nous souhaitons pouvoir consulter toutes les notes d'un élève, éventuellement entre deux dates définies.
Dans une modélisation pour un SGBDR classique, nous aurions probablement une table note ayant :

- Comme clé primaire l'identifiant technique,
- Comme clé étrangère l'identifiant de l'élève lié,
- Et ensuite une colonne pour les champs note, matière et date.

Dans Cassandra, une telle modélisation est possible mais sera au mieux lente. Si une telle modélisation était utilisée dans notre cas, nous aurions alors une ligne par note et obtenir toutes les notes d'un élève obligerait à utiliser un index secondaire :

```
CREATE TABLE grades (
 grade_id uuid,
 student_id  uuid,
 obtained_at  timestamp,
 grade   text,
 subject  text,
 PRIMARY KEY (grade_id)
);
CREATE INDEX student_id ON grades (student_id);
```

Voici la requête correspondante :

```
SELECT * FROM grades WHERE student_id = "<my_id>"
```

Cela fonctionne, mais ce ne sera ni performant, ni scalable car l'index secondaire oblige à interroger tous les noeuds de notre cluster Cassandra afin de vérifier qu'il dispose de données. Il faut vraiment limiter l'utilisation des index secondaires à des cas exceptionnels et pour des tables ne comprenant que peu d'éléments.
Une bien meilleure modélisation est de profiter de la performance de Cassandra sur des lignes ayant de nombreuses colonnes et donc de ranger chaque note d'un élève dans une colonne, puis d'utiliser l'identifiant de l'élève comme clé de partitionnement comme dans notre table ci-dessous :

```
CREATE TABLE grades (
 grade_id uuid,
 student_id  uuid,
```

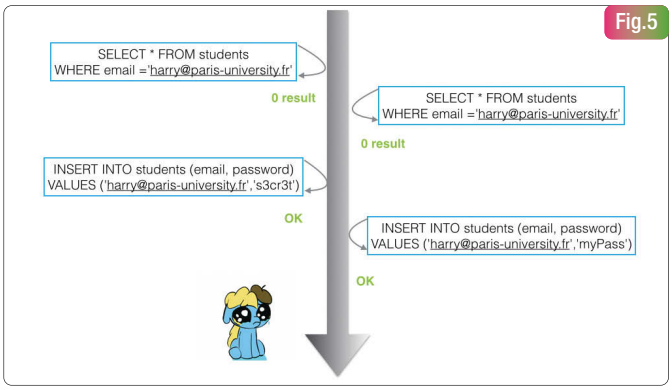


Fig.5

```
obtained_at timestamp,
grade  text,
subject  text,
PRIMARY KEY (student_id, obtained_at, grade_id)
);
```

On ajoute également en tant que clustering column “obtained_at” afin que les notes soient triées sur disque en fonction de la date d’obtention de la note. Cela permet de s'appuyer sur la manière dont Cassandra range les données afin d'offrir de bonnes performances pour notre requête. Enfin, la dernière clustering column “grade_id” est un identifiant technique permettant de s’assurer que notre clé primaire est bien unique.
Obtenir toutes les notes d’un élève est tout aussi facile qu’avec un index secondaire mais n’interroge plus qu’un noeud du cluster (celui hébergeant les données pour l’étudiant) :

```
SELECT * FROM grades WHERE student_id = “<my_id>”
```

Obtenir toutes les notes d'un élève pour les deux premiers mois de l’année est tout aussi aisé :

```
SELECT * FROM grades WHERE student_id = “<my_id>”
AND obtained_at >= “2015-01-01 00:00:00”
AND obtained_at < “2015-03-01 00:00:00”
```

Cette requête est performante car la clé de partitionnement permet à Cassandra de déterminer le serveur du cluster hébergeant la donnée. Ensuite, comme les données sont triées sur disque dans l’ordre des “clustering columns”, réaliser des requêtes sur la date d’obtention (notre première clustering column) revient à obtenir une partie contiguë d'un fichier de données. La lecture séquentielle est bien plus performante sur un disque dur que des lectures aléatoires, ce qui permet d’obtenir une bonne latence sur cette interrogation. Ce type de requête est appelée une "slice query" et doit être privilégiée le plus possible.

Gestion des étudiants

Nous avons maintenant une structure robuste et scalable afin de gérer les requêtes nous ayant été demandées sur les notes d’un élève. Afin de remplir complètement notre cas pratique, il nous reste à gérer notre table étudiant avec deux contraintes à respecter :

- Il ne peut pas y avoir deux étudiants avec le même email dans notre base Cassandra,
- On veut pouvoir ajouter des cours à un étudiant.

Pour gérer notre première contrainte une implémentation naïve serait l’algorithme suivant:

- On réalise un SELECT afin de vérifier que l’email de l’étudiant n’est pas encore utilisé,
- Si le SELECT ne renvoie pas de valeur on peut alors réaliser l’INSERT afin de créer le nouvel étudiant.

Cela fonctionne bien tant que deux étudiants n’essayent pas de s’inscrire en même temps. Sinon, nous nous retrouvons dans la situation suivante : Fig.5. Une réponse positive a été envoyée au premier étudiant car l’insertion a été réalisée avec succès. Cependant, il va ensuite tenter de se loguer à son compte et cela ne fonctionnera pas.
Dans Cassandra, le dernier processus ayant écrit a raison.
Afin d’éviter ce cas problématique il existe depuis le début de la version 2 de Cassandra, les lightweight transactions. Ce mécanisme, se basant sur Paxos, permet de s’assurer que l’insertion n’a lieu que si l’étudiant n’existe pas. Il permet l’écriture d'une requête de ce type :

```
INSERT into students VALUES (...) IF NOT EXISTS;
```

Dans le cas où l’étudiant existe déjà, une réponse négative sera envoyée au client au lieu d’écraser les valeurs précédemment insérées.
Il est important de ne pas abuser de cette fonctionnalité, qui implique quatre allers-retours réseaux au lieu d’un seul au sein du cluster Cassandra. La dernière tâche à effectuer avant de considérer notre cas pratique comme résolu est de permettre l’ajout d’un cours à un étudiant.
Nous allons tenter l’approche naïve :

- On lit dans Cassandra les données de notre étudiant puis l’on ajoute le nouveau cours à notre utilisateur,
- On insère les nouvelles données dans Cassandra.

Lire avant d’écrire est considéré comme un anti-pattern dans Cassandra et cela pour deux raisons :

- D’un point de vue performance, cela oblige à effectuer deux requêtes sur le cluster,
- D’un point de vue cohérence des données, comme vu dans le paragraphe précédent, on peut perdre certaines mises à jour. La bonne façon de traiter ce cas est d’utiliser les données de type collection dans Cassandra. Par exemple un Set (une liste unique d’éléments) sera particulièrement adapté pour la gestion des cours de notre élève :

```
CREATE TABLE students (
 email text PRIMARY KEY,
 first_name text,
 last_name text,
 courses set<text>
);
```

Avec une telle structure, ajouter un cours peut se faire sans lecture préalable :

```
UPDATE students
 SET courses = courses + {'chinese'} WHERE email = 'harry@paris-university.fr';
```

De la même manière la suppression peut se faire facilement :

```
UPDATE students
 SET courses = courses - {'chinese'} WHERE email = 'harry@paris-university.fr';
```

Et voilà! Plus de lecture avant les écritures !

Aller plus loin

Au cours de cet article nous avons vu les bases nécessaires afin de bien débuter avec Cassandra : les concepts de cette base NoSQL, comment les données sont stockées au sein du cluster, comment installer et utiliser notre base avec un client Java, et, enfin un cas de modélisation des données avec les erreurs classiques à éviter. Cependant, il est possible de faire bien plus avec Cassandra :

- Déployer et faire évoluer son cluster en production afin de gérer toujours plus de trafic,
- Mettre en place du Spark avec Cassandra afin de faire du machine learning sur toutes nos données,
- Faire de l’élection de leader avec Cassandra.

Nous vous laissons découvrir ceci.

Devenez développeuse !

Peu de personnes le savent, mais le premier programmeur était une femme. En effet en 1843, Ada Lovelace (fille du poète anglais Lord Byron) a publié le premier algorithme pouvant être exécuté par une machine. Au début des années 80, le langage orienté objet Ada a été nommé ainsi en son honneur.


D.R.

Dans les années 50, Grace Murray Hopper, une militaire informaticienne a créé le premier compilateur (A-0 System) puis le langage COBOL en 1959. Elle est aussi connue pour avoir découvert le "premier cas réel de découverte d'insecte" lors de la panne de l'ordinateur Harvard Mark II causée par un insecte pris dans un relais. Cette découverte a popularisé le terme de "bug informatique".

Avant l'avènement de l'ordinateur personnel, la saisie informatique et la programmation étaient principalement l'apanage des femmes. Au début des années 80, avec l'arrivée des PC dans beaucoup de foyers, les hommes ont commencé à investir le terrain et à devenir majoritaires dans ce domaine. Cette période fût aussi celle des premiers pas de l'informatique à l'école avec le "plan Informatique pour tous" en 1985, destiné à équiper les écoles d'outils informatiques avec notamment les célèbres Thomson MO5 et Thomson TO7.

Enfin, on peut citer Frances Allen, la première femme à avoir reçu le prix Turing (équivalent au prix Nobel de l'informatique), et une pionnière de la compilation optimisée, de l'optimisation de code et du calcul parallèle. Et ce ne sont là que quelques noms de femmes informaticiennes célèbres; il en existe bien d'autres, ce qu'on a tendance à oublier aujourd'hui vu le pourcentage actuel de femmes dans l'informatique qui reste faible.

Aujourd'hui, où en sommes-nous?

En Europe, le nombre d'étudiantes en informatique est faible, mais une fois diplômées, le nombre de femmes dans l'informatique diminue encore : elles représentent seulement 9 % des concepteurs d'applications mobiles, 19 % des responsables dans le secteur des technologies de l'information et de la communication (TIC) (contre 45 % dans les autres secteurs de services), 19 % des entrepreneurs dans les TIC (contre 54 % dans les autres services) et 3 % des femmes diplômées le sont en informatique (contre 10 % des hommes).

En France, 20% des postes dans l'informatique sont occupés par des femmes (elles étaient 31% dans le début des années 80), parmi celles-ci, il y aurait 9% de développeuses. On constate aussi qu'après quelques années dans le développement, on propose souvent aux développeuses d'évoluer vers le management, la maîtrise d'ouvrage, le test logiciel, ce qui réduit encore le nombre.

Aux États-Unis, les étudiantes sont moins de 1% à penser que le développement informatique pourrait représenter leur avenir, 12% des femmes diplômées le sont en informatique (en 1984 elles représentaient 37%). Alors qu'en Indonésie ou en Iran, plus de la moitié des personnes ayant une activité informatique sont des femmes. Nous aurions donc, dans nos pays occidentaux, à nous inspirer d'autres cultures où globalement les femmes sont souvent plus exposées aux tâches et métiers techniques.

Des femmes qui rockent

Plusieurs femmes participent au développement, à l'évolution et l'innovation de l'informatique. Parmi elles, Radia Perlman, souvent appelée "Mère de l'Internet", s'est distinguée avec son invention du protocole Spanning Tree (protocole pour le fonctionnement des ponts en infrastructure réseau). Mais aussi Anita Borg qui y a également participé, et a développé MECCA, un système automatique centralisé de communication par listes de diffusion d'emails permettant de communiquer avec une communauté virtuelle. Barbara Liskov a, quant à elle, apporté énormément au domaine de la conception des langages orientés objet.

Groupe de développeuses/tech en France et dans le monde

Pour valoriser et promouvoir les femmes dans la technique ou encore susciter des vocations dans les métiers de l'informatique, plusieurs associations et communautés se mobilisent. Parmi elles, l'association Duchess France, qui met à l'honneur des développeuses, les encourage à intervenir lors de conférences et met en avant le métier de développeuse auprès de tous.

D'autres initiatives internationales comme Girls Who Code lancent des programmes d'atelier de code pour des adolescentes en espérant les inciter à poursuivre des études en informatique et susciter des vocations.

En bref, l'envie d'impliquer d'avantage de femmes dans le milieu de l'informatique est bien présent.Il reste à être patient pour que cela se répercute sur les prochaines générations.

Parce que le code n'est pas une affaire de genre mais bel et bien celle d'une passion comme en témoigne les portraits de développeuses qui suivent.

Les Duchesses

AMIRA LAKHAL

développeuse Java chez Valtech, Paris

J'ai découvert l'informatique à l'âge de douze ans. J'étais intriguée par cette bête qui jonchait dans le salon et qu'on dénommait PC, il me fallait la dompter. Ma première expérience, je l'ai vécue avec le "Prince of Persia", jeu sur ma disquette 3,5 pouces et que je démarrais à l'aide d'une commande DOS. Je suis devenue accro, j'y ai passé des heures entières, je suis devenue... "geek"!

Ensuite l'aventure a continué au lycée. J'ai eu la chance de pouvoir y découvrir l'algorithmique, le HTML et les débuts d'Internet. Après un bac scientifique, j'ai poursuivi mon cursus pour obtenir un DUT Génie Logiciel. J'ai omis de préciser un détail important : je suis née à Tunis et j'y ai vécu et étudié jusqu'à l'obtention de mon DUT. En Tunisie, la question ne se pose pas sur les métiers et le genre : il y a autant de garçons que de filles qui étudient l'informatique. Imaginez ma stupéfaction en intégrant mon école d'ingénieur en Bretagne : dans ma promotion, nous étions 3 filles sur 30 étudiants. La situation n'avait pas l'air de surprendre mes camarades de classe. C'était normal et j'étais l'intruse. J'y ai passé des bonnes années et j'ai appris les bases de l'algorithmique grâce au "B-Book", la bible du développeur.

Mon diplôme d'ingénieur en poche, j'ai quitté la Bretagne pour m'installer à Paris et rejoindre le monde professionnel. J'ai commencé à aller aux événements communautaires, à suivre de plus en plus de blogs techniques, à faire de la veille.

Après une courte période dans une startup, j'ai rejoint Valtech en tant que développeur Java. J'ai découvert de nouvelles librairies autour du langage Java et j'ai appris à utiliser des nouvelles méthodologies: TDD, BDD, XP, SCRUM...

De jour en jour, j'améliore mon code et je gagne plus d'expérience et de bonnes pratiques. De plus, le fait de changer de mission me permet de traiter des contextes fonctionnels différents et d'intégrer des nouvelles équipes à chaque fois.

Malheureusement, je rencontre rarement des femmes développeurs mais ma présence a toujours été bien acceptée.

D'ailleurs, je suis membre de l'association Duchess France ou j'aide à rendre plus visibles les femmes dans l'IT.

Aujourd'hui, je m'intéresse aux problématiques liées au Big Data et j'étudie les diverses solutions ainsi que certains langages fonctionnels. L'informatique est un monde très passionnant car les technologies et langages évoluent de jour en jour et je ne m'en lasse pas.

STÉPHANIE HERTRICH,

développeuse évangéliste chez Microsoft

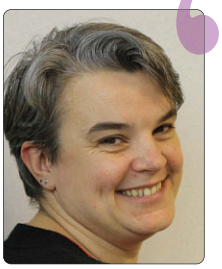
Je suis évangéliste technique chez Microsoft depuis 4 ans, spécialisée dans le développement d'applications mobiles en XAML/C#.

Auparavant, j'ai passé 13 ans dans l'industrie en tant que développeuse puis lead dev. Cela fait donc un bon moment que je code !

Bientôt 20 ans ! J'ai passé plusieurs années à faire du développement temps réel embarqué en C++ avant de bifurquer vers les technologies Microsoft. J'ai choisi de rester dans la technique plutôt que d'aller vers le management ou le marketing, même si ce n'est pas toujours mis en valeur en France. Le côté technique de mon job me plaît. Pour ma part j'y suis venue tôt, j'ai eu ma 1ère machine vers 10 ans - à l'époque c'était un Oric Atmos (48K !), puis Amstrad CPC 6128, puis Amiga. Entre 14 et 16 ans j'ai fait partie de plusieurs groupes de démos mais je ne codais pas encore : j'étais "swapper", c'est à dire quelqu'un qui ne sait ni coder, ni faire du graphisme, ni composer de la musique et qui est donc réduit à diffuser les démos : la loose ! Je voyais les autres coder en ASM 68000 et cracker des jeux, ça m'épatait ! Puis j'ai commencé des études d'informatique... et ça m'a beaucoup plu : moi aussi j'avais des super-pouvoirs. En fait j'aime coder pour les mêmes raisons que les garçons : plus petite je jouais beaucoup aux legos techniques, ça me plaisait de construire des trucs. Bref, ça vient de loin.

STÉPHANIE MOALLIC,

développeuse informatique à Brest



J'ai eu mon premier ordinateur vers 8 ou 9 ans, c'était un Amstrad CPC 6128. Mais un peu avant j'avais eu la chance à l'école de pouvoir faire mes premiers pas en programmation sur un MO5 et un TO7 avec le langage LOGO et sa petite tortue à laquelle on donnait des ordres pour faire des petits dessins. Après j'ai tenté de faire de la programmation en Basic sur mon CPC mais j'oubliais toujours la commande pour enregistrer tout ce que je venais de recopier. Quand j'y repense, je crois que j'ai toujours voulu programmer même si je ne savais pas toujours ce que cela signifiait vraiment. Puis il y a eu le choix de l'orientation où personne ne savait quoi me répondre : bac informatique, bac scientifique, ce n'est pas pour toi, ça n'existe pas...Alors j'ai décidé : lycée technique, bac scientifique option technologie et BTS Informatique Industrielle ! Le BTS en poche, j'ai commencé à travailler (en 1998) en région parisienne où j'ai vraiment découvert le métier de développeur : spécifications fonctionnelles/techniques, règles de codages, gestion des priorités...Ce premier emploi m'a confortée dans ce choix de vouloir être développeuse.

J'ai aussi découvert que pour progresser dans ce métier il faut toujours être prêt à apprendre de nouvelles choses, on ne peut pas toujours rester sur nos acquis. Mais j'ai la chance de faire de ma passion mon métier. Après cette première expérience de 2 ans, j'ai quitté Paris pour aller m'installer en Bretagne et découvrir encore de nouveaux aspects du métier, en codant dans des langages que certains considèrent comme dépassés (le COBOL), mais aussi en commençant à développer en Java (dans sa version 1.2) pour créer des interfaces graphiques qui n'étaient pas des applets.

Aujourd'hui, je continue à développer. Je m'intéresse aux technologies autour du Web mais aussi autour des interfaces et de l'expérience utilisateur. Je commence aussi à jouer avec le développement d'applications mobiles pour la vie de tous les jours.

AGNÈS CREPET

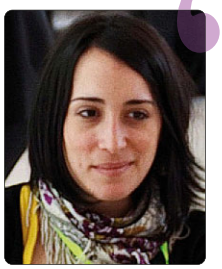
Java Champion, co-fondatrice de Ninja Squad, Lyon

Pas d'ordinateur dans ma famille quand j'étais petite, mais j'ai eu la chance d'avoir des cours de LOGO en primaire : vive l'apprentissage du code à l'école! Les petites filles ne continueront à rêver qu'aux princesses si on continue à ne leur mettre que des Barbies dans les mains... Moi j'ai eu la chance, même si je n'avais pas d'ordinateur à la maison, de pouvoir jouer énormément aux légos : je suis sûre que ça a joué dans mon appétence pour les sciences de l'ingénieur! Puis, plus tard, quelques copains m'ont fait découvrir le monde des logiciels libres, et là, je me suis dit qu'il fallait que je travaille dans ce domaine! Après mes études dans le domaine des réseaux de neurones artificiels et du génie logiciel, j'ai commencé à travailler en 2001 comme développeur chez un éditeur, puis je suis restée plusieurs années dans une SSII en tant que responsable d'une cellule d'architectes Java/JEE. J'ai ensuite basculé du côté client. J'ai intégré une DSI d'un laboratoire pharmaceutique en 2008 pour me lancer dans une refonte complète du Système d'information : ESB et SOA au programme (on ne parlait pas de micro-services à l'époque, dommage!). Enfin en 2012, avec 3 autres développeurs passionnés, je me lance dans la création d'entreprise et on monte Ninja Squad, une société coopérative sans manager ni commercial et où on essaie de développer, au delà de la prestation, nos propres applications... le pied ;-)

En 2012 j'ai également eu l'honneur d'être nommée Java Champion. C'est un titre décerné à un peu plus de cent personnes dans le monde œuvrant pour la communauté Java. Nous sommes moins d'une dizaine en France et il y a peu de femmes : je suis la seule française mais j'espère que cela va changer ! Depuis quelques années, je suis investie dans plusieurs communautés de développeurs : l'équipe organisatrice du Lyon Java User Group, celle de Duchess France et j'ai co-fondé la conférence Mix-IT. La richesse de ces communautés est une facette incroyable de notre métier : l'échange de savoirs, la curiosité, le fait de se remettre tout le temps en question. C'est ce qui me fait dire aujourd'hui que j'ai l'impression de faire un métier formidable! Il y a certes peu de filles à mes côtés au quotidien, surtout sur la partie technique, mais j'ai bon espoir que cela change, en tout cas on essaie de tout faire pour avec l'équipe de Duchess France ;-)

KATIA ARESTI :

développeuse Freelance,Duchess FR Board member, MongoDB User Group



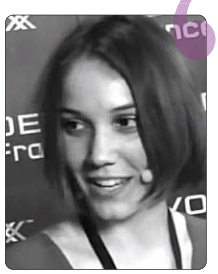
"INGÉNIEUR !!!" voilà ce que je répondais, quand j'étais enfant, à la question "Qu'est-ce que tu veux faire quand tu seras grande ?". Certainement influencée par mon père qui m'avait mis l'idée en tête car il voulait "le meilleur métier" pour moi. Cela explique peut-être qu'en plus de mes poupées mes parents m'ont offert beaucoup de LEGOS (après réflexion c'était peut-être une excuse de mon père pour pouvoir y jouer librement...). J'ai donc passé des heures infinies pendant toute mon enfance à donner vie à des personnages et histoires imaginaires et à construire avec des petites briques !

Même s'il est important d'écouter nos parents, dans la vie nous devons choisir notre propre voie. Malgré mon attirance pour l'art dramatique, la danse et la peinture, mon parcours à l'école m'a montré que je suis plus scientifique que littéraire. A 17 ans, au moment fatidique où il faut décider quelles études choisir après le BAC, en ayant perdu beaucoup de mon enthousiasme d'enfance, je disais encore que j'allais être Ingénieur. Ingénieur ? Mais au fait, ça veut dire quoi "être Ingénieur" ? ça consiste en quoi ce métier ? Et puis, Ingénieur de quoi d'abord ? En réalité, j'étais très perdue ...

Mon premier élément de réponse je l'ai eu lorsque j'ai choisi la matière optionnelle informatique. J'avais déjà un ordinateur à la maison, je surfais sur le net et je voulais en apprendre d'avantage. Durant 3 mois notre professeur nous a appris à coder en Pascal. Ce fut une vraie révélation, et une fois le BAC obtenu, je suis partie en école d'ingénieur en Informatique en espérant découvrir le métier. Les 5 années passées à l'école n'ont fait que confirmer mon choix. J'adore tout ce qui attrait à la création de software.
Aujourd'hui cela fait déjà 10 ans que je travaille dans l'IT. Je suis développeuse plutôt backend mais je touche aussi le front-end. J'adore le challenge qui nous force à se réinventer et se remettre en question en permanence. J'ai découvert un métier riche, rempli de personnes passionnées, dans lequel on peut s'épanouir en prenant des responsabilités techniques et fonctionnelles, en tant que coach, consultant ou "pompier", dans tous les secteurs et tous les domaines. J'adore le coté à la fois créatif et technique, social et individuel, amusant et rigoureux du métier. Finalement je ne sais pas si j'ai trouvé la réponse à ce que c'est d'être Ingénieur, mais j'ai découvert être "Développeur", et j'espère pouvoir continuer dans cette voie pendant longtemps.

LUDWINE PROBST,

Data Engineer chez Cityzen Data, Duchess France leader Brest



Pour moi l'informatique n'a pas été une évidence. Plus jeune jamais je n'aurai imaginé travailler dans ce secteur. Jusqu'à mes 25 ans, je n'avais pas Internet et vivais très bien sans technologie. Et puis j'ai dû me mettre à écrire quelques lignes de code se résumant à des boucles for et while (en scilab) pour une épreuve de l'agrégation de mathématiques. Premier contact difficile avec le "code" et les ordinateurs. Mais comment marche un ordinateur ? Comment fait-il pour comprendre ce que j'écris ? Tout cela restait bien mystérieux pour moi...

A l'obtention de mon Master Recherche en Mathématiques en 2010 est arrivé le dur choix de décider quoi faire de ma vie. Enseignante en maths ? Animatrice de films d'animation ? Faire le tour du monde ? Sans trop savoir à quoi m'attendre, j'ai parcouru des offres d'emploi sur le Web et je que je suis tombée sur la seule offre d'emploi collant en tout point à mon maigre CV : développeur informatique dans une SSII, offre accessible à des débutants en informatique avec formation à la clé.
Vivre à Paris, un de mes rêves ! Et le code pour

moi c'était écrire des algorithmes, réfléchir à des problèmes, et trouver des solutions à rédiger dans un certain langage...En fait ça m'évoquait des démarches similaires à celles que j'utilisais en faisant des maths à la différence près que souris, écran, clavier et Internet allaient remplacer papier, gomme, stylo et livres. Alors en 2 semaines je quitte Besançon et me lance dans cette nouvelle aventure où j'apprends le Java. Consciente de mes lacunes ou retards j'accumule les participations aux events tech de la communauté parisienne et me forme via des tutoriaux sur le net.
Aujourd'hui, je suis développeuse backend spécialisée dans les données (traitement, analyse, machine learning), ce qui me donne une certaine proximité avec les mathématiques. Je travaille avec Apache Spark en ce moment et suis confrontée à des challenges de calculs distribués, processing et analyse de très gros volume de données. En bref, je suis très heureuse de me casser la tête sur des problématiques chaque jour et de pouvoir apprendre de nouvelles choses. Pour moi l'informatique est un monde en évolution permanente, où innovation, créativité et curiosité s'entremêlent.

AURÉLIE VACHÉ,

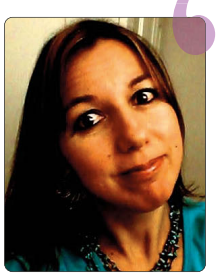
developpeuse Web chez atchikservices, Toulouse

Petite j'ai toujours aimé jouer au Lego et dessiner. Partir de rien pour créer quelque chose grâce à de la technique et de l'imagination c'est ce qui me plait.
Lorsque l'on a eu notre premier ordinateur a la maison, un Packard Bell avec un P1 a 133Mhz, de suite je m'y suis intéressée et je voulais savoir comment créer des logiciels et des jeux. Puis dès le collège je me suis plongée dedans, je participais à des ateliers "informatique" le midi et c'est la que j'ai commencé à écrire mes premières lignes de HTML puis mes premières lignes en Turbo Pascal à la maison.
Pour mes études, je voulais "travailler dans l'informatique", créer des logiciels, donc on m'a orienté vers un BTS STT option "Informatique de Gestion" puis une Licence Professionnelle option "Développeur d'Applications E-Business" à Montpellier.
J'ai commencé chez atchikservices il y a 9 ans en tant que Développeuse Java/JEE (J2EE à l'époque), je faisais partie de l'équipe de dev qui concevait des Chats, pour des opérateurs de téléphonie mobile, pour les interfaces SMS, MMS, WAP (WML, XHTML, OML) et Web. Quelques années plus tard je suis allée dans une équipe qui s'occupait du déploiement des applications et de tout l'aspect opérationnel. Cela m'a permis au fil des années de devenir une Développeuse multi casquettes.

Ce que j'aime dans mon métier de dev Full-Stack, c'est que je touche à toutes les couches d'un projet. Ça va de l'administration système, aux bases de données, au développement d'applications métiers, sociales, de sites Internet, à la gestion de projet technique, en passant par la qualité de service, l'opérationnel (DevOps), le monitoring, la sécurité, les campagnes de tests, le déploiement, le SEO, l'ergonomie (UX) et le Big Data.
Chaque semaine et je dirais même, chaque journée est différente. Le matin je peux travailler sur l'optimisation des performances de la récupération de données Big Data en asynchrone via Google BigQuery d'une appli en Java/GWT et quelques heures après sur la création ou la maintenance d'une application sur Facebook en PHP/HTML5/CSS3/JS/MySQL et puis quelques minutes plus tard je peux passer au fine-tuning d'un serveur Apache par exemple. J'aime le fait que dans la journée je peux être amenée à travailler sur des missions différentes les unes des autres, il faut constamment faire preuve d'adaptation et être en veille permanente :-).

ARÁNZAZU SAN JUAN LLANO,

développeuse Senior Java Agile chez PALO IT, Paris :



A la fin 2002, j'ai déménagé sur Madrid afin de reprendre mes études d'Ingénierie en Informatique. Mon premier contact avec la programmation, s'était produit quelque temps avant, à l'école d'ingénieur, où j'écrivais mes algorithmes *Pascal*, *C*, *C++* ou *Haskell* sur papier, et où j'attendais longuement pour accéder à un PC puisque à l'époque je n'en disposais pas. Mon premier ordinateur n'arriva qu'après mes 20 ans, juste avant l'interruption de mes études dans des circonstances complexes, ce qui ne n'a pas entamé mon enthousiasme en découvrant *Linux* à travers la distribution Slackware et les articles des magazines Todo Linux ou PC Pro avec ses inoubliables CDs *Red Hat, Ubuntu ou Fedora*. Essayer d'étudier l'informatique sans avoir un ordinateur avait été la « folie » de cette fille qui grandit dans un environnement technologique dont la plus grande expression était la télévision en noir et blanc. Cette fille d'un petit village de Cantabria, dans le Nord de l'Espagne, à qui plaisaient autant les mathématiques que la poésie, des passions conciliées dans la *découverte disruptive* de la personne d' *Ada Byron*.

Le déménagement sur Madrid sera le premier « pivot » d'une carrière professionnelle construite en mode *Lean Startup* sans le savoir. Madrid sera la vie en *mode multithreading* menant en parallèle, mes études à temps partiel, un volontariat, des formations complémentaires, et de multiples emplois dont développeuse *Javascript*. À la fin 2005, je *pivote* encore, en réalisant une année d'études en Belgique. À mon retour, j'obtiens mon premier CDI chez Sopra, en travaillant dans des *projets nearshore en Français* comportant de nombreux déplacements en France. Le troisième « pivot » arrivera en 2012 avec ma mutation sur Paris où je serai conquise à jamais par les *méthodes agiles*.

J'ai surtout exercé les rôles de développeuse et technical lead *Java/J2EE,* et, actuellement je travaille en tant que consultante PALO IT sur un projet d'orchestration de services *REST* chez Cloudwatt pour lequel j'utilise *Arch Linux* comme environnement de développement. Je partage mon activité professionnelle avec la finalisation de mon Master en Ingénierie Web, je fais aussi de la veille technologique sur *Android/NodeJS/AngularJS* et j'espère continuer à coder *dignement* durant tous les « pivots à venir ».

Témoignage de Daphné Popin, développeuse chez Marmelab (Nancy)

marmelab
ATELIER D'INNOVATION DIGITALE

- Et toi tu fais quoi ?
- Ben, je suis dev' aussi.
- Sérieux ?
- Ben oui, pourquoi ?

J'imagine que toutes les développeuses ont, comme moi, déjà eu affaire à cette réaction où le développeur en face de toi réagit comme si tu venais de faire une blague. Je sais que la part des femmes dans la technique est plus que faible, mais quand même les mecs !

```
<alert>Troll mode</alert>
```

Qu'est-ce qu'un développeur ?

Un développeur ne se sent bien qu'avec un ordinateur entre les mains. Dans sa pyramide de Maslow, le Wi-Fi est tout en bas. Il passe ses journées et ses nuits à coder. Il a le teint pâle parce qu'il ne voit la lumière du jour que très rarement, lorsqu'il est obligé d'aller se ravitailler en café/pizzas/bières. Il n'a pas d'amis « IRL » parce que ça n'est pas son genre de sortir le samedi soir. Non, il joue aux jeux vidéo le soir. Et bien sûr, il est célibataire. Et il sent mauvais.

Qu'est-ce qu'une développeuse ?

Il en existe deux sortes. La première, c'est celle qui essaie en tout point d'être comme son homologue masculin. Elle ne ressemble pas à une « vraie fille ». En fait, c'est comme un développeur, sauf qu'elle se lave un peu plus souvent. La seconde, on dirait vraiment une fille, elle porte même quelquefois des robes et des chaussures à talons. Et tout le monde est très sympa avec elle, donc elle en profite. De toute façon, dès qu'elle a un problème, tout le monde accourt pour l'aider. Il faut dire que ce n'est qu'une fille, elle ne peut pas régler son bug toute seule.

Voilà, maintenant que je me suis mise les développeurs et les développeuses à dos, je vais pouvoir continuer mon propos. De la même manière que je viens d'énoncer un amas de clichés liés à notre profession, je ne pense pas que la question « Qu'est-ce qu'une femme peut apporter au métier ? » soit pertinente. Je pense que ce qu'on peut apporter dans une équipe et pour un projet n'est pas lié à notre sexe mais à notre personnalité.

Je ne pense pas non plus que les développeurs soient sexistes, et nous considèrent inférieures à eux. Mais il est vrai qu'évoluer dans un milieu d'hommes, c'est tout un art ! Bien sûr, j'ai déjà eu droit à de bonnes blagues machos qui font rire tout l'open space, mais ça me fait rire moi aussi après tout. Il suffit de répondre avec une phrase cinglante qui leur cloue le bec, au risque de passer pour une digne héritière de Simone de Beauvoir. Et puis, il y a des avantages assez inattendus : les conférences techniques sont quand même le seul endroit où la file d'attente pour les toilettes à la pause est du côté des hommes !

On m'a souvent demandé pourquoi j'ai choisi ce métier. J'avoue que ce n'était pas une vocation, même si vu mon nom [POPIN], il faut croire que c'était ma destinée ! Arrivée en première année de Licence de Mathématiques, on nous annonce l'emploi du temps. Cours d'algorithmique. Je me suis dit, c'est quoi ce truc ?! Au début, tout ça me paraissait totalement abstrait. Et puis au final, je me suis rendue compte que c'était principalement de la logique et qu'il fallait juste faire travailler ses méninges. À la fin de ma première année de fac, le changement d'orientation était devenu une évidence. Trois ans plus tard, un DUT et une Licence en poche, me voilà « Développeuse Web ».

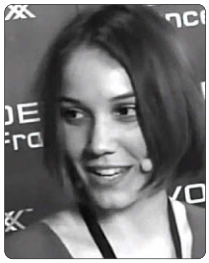
Et j'adore mon métier ! J'ai commencé en agence chez Nurun où j'ai principalement travaillé sur des projets PHP, avec une gestion de projets classiques en cycle en V. Depuis un peu plus d'un an, je suis chez Marmelab, atelier d'innovation digitale basé à Nancy et Paris. On y travaille sur des projets Web innovants, avec souvent une forte incertitude métier et/ou technique, avec des technologies qui font

rêver, et toujours en mode agile ! On applique la méthodologie SCRUM, qui nous donne plus de pouvoirs en tant que développeuse(eur). Mais comme chacun sait, « With great power comes great responsibility ». Il me semble que ça augmente notre implication et nous responsabilise davantage. Par exemple, au niveau des poker plannings où chacun participe aux estimations de tâches. De même pour les démos, où chacun présente à l'équipe ce qu'il a réalisé durant le sprint. Mais de mon point de vue, le principal atout est qu'avec ce mode de fonctionnement, l'échange entre les développeurs est accru, on se tire vers le haut. Si bien que je ne m'imagine plus commencer ma journée sans le rituel du stand-up !

Mais revenons-en au sujet... Le nombre de filles dans les filières menant aux métiers du développement augmente tous les ans. J'espère donc que dans quelques années ce genre d'article ne nous paraîtra plus d'actualité. Et qu'au-delà de ce problème de parité, les clichés liés à notre métier n'existeront plus. J'ai l'impression que de plus en plus de femmes osent prendre la parole aux conférences, je les admire ! D'ailleurs, chers Développeurs, n'hésitez pas à venir nous parler aussi à tous ces événements, on n'est pas différentes de vous, on est souvent intimidées et on ne risque pas de vous manger !

Paroles de Ludwine et de Stéphanie


Stéphanie Moallic


Ludwine Probst

Ludwine Probst, développeuse chez Cityzen Data (https://twitter.com/nivdul), à Brest, et Stéphanie Moallic, développeuse chez B&B Hotels, à Brest, (https://twitter.com/steffy_29) évoquent pour Programmez ! le métier et le marché du développement.

Qu'est-ce que c'est qu'être développeur en 2015 ?

Le monde numérique évolue à vitesse grand V, le développeur doit lui aussi être en évolution constante afin de s'adapter et d'apprendre les nouvelles technologies ainsi que les nouvelles tendances. C'est ce qui fait aussi que le métier de développeur est si varié. Aujourd'hui, le monde numérique s'invite partout ! Nous sommes entourés de plus en plus de données et d'objets connectés : services, santé, sport, vie de tous les jours...

Quels profils émergent ?

Avec l'explosion du nombre de données et les enjeux que ces données représentent (stockage, analyse, restitution), de nouveaux profils dans la "Data Science" voient le jour et sont de plus en plus demandés. Les données aujourd'hui représentent de vrais enjeux business, par exemple pour des sites de e-commerce avec des systèmes de recommandation ou encore personnalisation de contenu par utilisateur.

Après avoir longtemps été négligés, les utilisateurs redeviennent aujourd'hui un centre d'attention avec un intérêt grandissant pour l'UX et l'UI design. En effet les applications proposées étaient souvent peu pratiques, liées au peu de concurrence, mais aussi au peu d'intérêt porté à cet aspect. Une attention particulière est portée aux utilisateurs, et surtout à leur expérience, lors de l'utilisation d'applications ou sites, afin que celles-ci soient les plus naturelles et efficaces possible.
L'utilisateur revient au centre, il doit être séduit et fidélisé.

Et le nombre des développeuses est-il en augmentation ?

D'après plusieurs études et les derniers chiffres, la réponse est non. Il y a aujourd'hui autour de 10-15% de femmes présentes dans les filières informatiques en France. Ce chiffre ces dernières années semble stagner, voire diminuer suivant les différentes études et sources. Et l'on dénombre autour de 10% de femmes développeuses ou techniques (les chiffres exacts sont difficiles à obtenir).
Les initiatives à travers le monde se multiplient pour initier les filles et femmes au code, et former des communautés de développeuses pour éviter l'isolement.
En France, on peut citer Devoxx4Kids ou Coding Goûter qui proposent des ateliers de découvertes au code pour les enfants et adolescents.
L'association Duchess France met en avant des femmes développeuses en tant que rôles modèles, avec pour ambition d'inspirer et représenter des femmes techniques dans l'informatique. La Web@cadémie, quant à elle, propose une formation de développeur Web en 2 ans pour des jeunes filles et garçons de 18 à 25 ans, sortis du système scolaire sans qualification. La féminisation des promotions est une préoccupation et une ambition de cette école.
Il ne s'agit pas d'atteindre nécessairement une parité parfaite dans l'informatique mais de permettre aux jeunes filles, mais aussi aux femmes, de mieux se projeter dans ce milieu et de s'y affirmer; le milieu de l'informatique souffrant encore aujourd'hui de clichés et d'un manque de modèles féminins.

Etude mutationelles http://www.global-contact.net/etudes/
Duchess France www.duchess-france.org
Web@cadémie http://webacademie.org/

Le mouvement "apprendre le code" à l'école peut-il aider à sensibiliser les filles à l'informatique ?

Le mouvement "apprendre le code" est une initiative intéressante mais qui semble limitée si elle se traduit par le simple apprentissage d'un langage ou outil. Ce qui nous semble le plus important dans le code et l'informatique en général, c'est le raisonnement, la capacité à poser et résoudre un problème et la compréhension de l'algorithmique. Les mathématiques sont justement enseignés à la base dans ce sens au collège et lycée : ils nous initient aux démonstrations de théorèmes, à la modélisation et résolution de problèmes.

Le monde actuel étant de plus en plus ancré dans le numérique, il semble désormais important d'y sensibiliser les enfants à l'école afin qu'ils comprennent au mieux le monde qui les entoure, leur donner les armes et connaissances nécessaires pour qu'ils saisissent notre monde actuel et puissent s'y impliquer au mieux.

Pourquoi devenir développeuse ?

> **Stéphanie :** et pourquoi pas ? J'ai toujours trouvé fascinant ce que l'on pouvait faire avec un ordinateur et quelques lignes de code : de la musique, de la génération de fractales, des sites Internet, commander la mise en route d'un chauffage dans une pièce...Les domaines d'applications du métier de développeur/développeuse sont vastes. Et j'apprécie le côté créatif du métier.

> **Ludwine :** le métier de développeur évolue au rythme du monde qui nous entoure.

L'informatique et plus globalement le numérique, sont de plus en plus présents dans notre quotidien, ils modèlent et impactent clairement notre façon de vivre. Aujourd'hui pour moi, un développeur n'est plus un simple exécutant, il devient créateur et innovateur. Et les hommes autant que les femmes peuvent s'y épanouir et y trouver leur place.

Alizée Penel, développeuse chez Genymobile

L'informatique est reconnue pour être un secteur majoritairement masculin, qui forme et emploie peu de femmes. En France, 20 % des développeurs seraient des femmes et seulement 10 % d'entre elles seraient ingénieurs systèmes. Cependant, les femmes sont bien présentes à des postes clés de l'IT, et plusieurs communautés et associations voient le jour afin de les mettre en avant.
Dans ce contexte, qu'est-ce qu'être développeuse système ? Nous parlerons de ce métier à travers l'expérience d'Alizée Penel qui partage son temps entre l'entreprise et le professorat.

Qui es-tu ? Que fais-tu ?

I am an (embedded) system developer(1), en poste chez Genymobile, une entreprise d'expertise et de solutions Android professionnelles. Je travaille sur des problématiques de virtualisation, de stack graphique sous Linux et de ROM cooking Android. J'ai également le plaisir de dispenser des cours de système Android à EPITA aux futurs ingénieurs spécialisés dans les systèmes embarqués.

Pourquoi développeur ? Qu'est-ce qui t'a motivée, attirée dans ce métier ?

J'ai découvert les joies du code très tard, à 21 ans. À l'origine, je souhaitais être vétérinaire et par conséquent, j'ai suivi trois ans de classes préparatoires. N'ayant pas réussi les concours, je me suis inscrite en Licence de Sciences de la Vie à Jussieu où j'ai eu la chance de découvrir la programmation grâce à un cours d'introduction qui était obligatoire : une révélation ! L'année d'après, j'intégrais EPITA, avec une seule idée en tête : devenir chef de projet.

Quelles sont les raisons pour lesquelles tu es restée dans la technique plutôt que la gestion de projet ?

Ce sont les jours et les nuits passés devant mon écran à apprendre comment les choses fonctionnent et à les faire fonctionner qui m'ont fait changer d'avis. Tous les jours, on apprend, on crée, on répare : c'est sans fin. La constante évolution des technologies me tient en haleine. L'open-source aussi a joué un rôle : des personnes d'origines et de sensibilités différentes travaillent ensemble pour converger vers les mêmes objectifs, c'est-à-dire partager et sans cesse améliorer. Une philosophie qui m'a séduite tout de suite.

Et demain ?

Tant que la technique et le développement me captiveront, je poursuivrai. Question de passion. Continuer à enseigner aussi, et plus si possible.

Est-ce que dans ton travail quotidien, le fait d'être une femme a déjà posé des problèmes ?

Malheureusement, oui, c'est déjà arrivé : des réflexions, des comportements, parfois même en public. Ce qui me choque encore, c'est le silence de l'auditoire. Le recours à ces attitudes, pour démontrer nos torts éventuels ou pour nous faire taire, révèle la faiblesse d'esprit qui persiste encore chez certains hommes. La peur qu'ils cherchent à installer pour asseoir leur autorité montre en réalité leurs craintes et leur propre malaise vis-à-vis de nous. Mais soyons clairs, c'est rare et ponctuel. Ce n'est pas le milieu informatique qui est en cause : il y a des gens peu intelligents partout. Sinon, au quotidien, RAS. Je suis l'égale de mes collègues. Ils me taquinent tous les jours et je le leur rends bien. Nous sommes plutôt avantagées en vérité : il n'y a jamais la queue aux toilettes.

Que penses-tu des récents mouvements de femmes dans les technologies (Duchess, OPW) ? T'y associes-tu ? Pourquoi ?

Ces mouvements me laissent un peu perplexe, et me dérangent sous un certain aspect, surtout OPW. Normalement, l'open-source ne fait pas état de celui ou de celle qui contribue mais de la valeur et de l'intérêt de ce qui est partagé. En posant la contrainte d'être une femme pour participer au programme OPW, c'est un contresens total. Personnellement, je souhaite me démarquer, non pas parce que je suis une femme, mais parce que je le mérite par la qualité de mon travail. Nettement plus gratifiant !

Dans l'école où tu enseignes, est-ce que tu réfléchis à des actions/attitudes en faveur des élèves filles ? Est-ce que toi-même tu as des a priori (inconscients ou non) face à une élève fille ?

Si une seule chose change, c'est la forme de mon discours et non pas le fond, mais ça s'arrête là : traiter les filles différemment des garçons, ce ne serait pas leur faire honneur ni leur rendre service. Je prodigue les mêmes conseils aux deux : si tu es réellement motivé, donne-toi les moyens de réussir, ne lâche rien, pose des questions et si la réponse ne te satisfait pas, dis-le. Un peu comme le fait un coach sportif.

Quels projets/technologies te font rêver ? Auquels souhaiterais-tu participer ? Pourquoi ?

Que des projets bas-niveau : noyau Linux, QEMU, Xen, etc. Ainsi, travailler sur Genymotion satisfait mes attentes puisqu'il utilise les mêmes technologies, et j'ai la chance d'avoir pu rejoindre ce projet. Il y a également la robotique et la domotique qui me passionnent car cela réunit l'informatique, la physique et l'électronique. Etre au plus près du cœur de la machine, avoir les mains dans le cambouis est intellectuellement excitant.

Vim ou emacs ? Et est-ce que tu as un color scheme rose ?

J'étais sous emacs il y a six mois encore. Je suis passée à Vim depuis. Non, je ne trollerai pas, sinon on n'est pas sorti de l'auberge ! Et pour info, je n'aime pas le rose.

T'es plutôt char* ou String ?

En ma qualité de dev système qui a fait du kernel, char*, évidemment. Les strings sont réservé(e)s à mes collègues javaistes !

(1) L'avantage de l'anglais sur la langue de Molière, c'est que les noms communs sont neutres. Ainsi, un métier reste un métier qui définit ce que je fais et non qui je suis.

Unity 5.0 : les nouveautés

La nouvelle version majeure d'Unity est sortie et on peut dire qu'ils ont bien travaillé. Dans cette version 5, énormément de nouveautés comme on pouvait s'y attendre avec de gros changements au niveau du rendu graphique, de la gestion du son et l'intégration de WebGL.

Fig.1

Maxime FRAPPAT
et Jonathan ANTOINE
Consultants
Infinite Square

Personal Edition vs Professional Edition

Il existe toujours deux versions différentes avec leurs lots de différences. On retrouvera pour la version professionnelle, un abonnement offert d'un an à *Unity Cloud Build Pro*. Un autre outil intéressant est la mise à disposition gratuite d'*Unity Analytics Pro*. Autre nouveauté pour la version professionnelle, l'accès à un nouveau store baptisé « Level 11 » où beaucoup de plugins sont disponibles gratuitement ou à un prix réduit. Il en coûtera 19$ par mois aux utilisateurs de la version personnelle afin d'accéder à ce programme.

Niveau prix, la version personnelle est entièrement gratuite et comprend toutes les plateformes mais est limitée aux entreprises ayant obtenu un revenu brut inférieur à 100 000$. La version professionnelle commence elle à 57€ par mois, auxquels il faut rajouter la même somme si vous souhaitez publier sur Android ou iOS en version pro. Aucune de ces deux versions ne propose l'accès au code source, ce qui peut se comprendre, mais quand on sait qu'*Epic Games* a mis à disposition celles d'*Unreal Engine 4* il y a peu gratuitement, cela peut peser dans la balance lors du choix du moteur à utiliser.

Build automatique dans les nuages

Depuis peu, Unity propose un service de build automatique dans le Cloud. Pour le moment, les seules plateformes supportées sont iOS, Android et le Web Player. Le fonctionnement est simple :

- Vous renseignez l'adresse du gestionnaire de source que vous utilisez,
- A chaque nouveau changement, le projet est recompilé,
- Une fois la compilation terminée, un email est envoyé pour que vous puissiez installer votre projet sur votre device.

Simple, rapide, efficace. Notons tout de même que seuls les gestionnaires de sources Git, SVN ou Perforce sont compatibles.

Des statistiques, plein de statistiques !

Encore en Beta ouverte, ce nouveau service est en quelque sorte un Google Analytics à la sauce Unity. Le SDK fourni peut être utilisé sur les plateformes Android, Windows Store Apps, Windows, Windows Phone 8, iOS, Linux/Steam OS, WebGL et le Web Player.

Au programme, des métriques précises sur vos utilisateurs qui vous donnent, par exemple, la possibilité de comparer l'usage que peuvent faire de votre jeu les joueurs Android et les joueurs iOS. La création d'évènements personnalisés est également de la partie, vous pourrez donc enfin connaître précisément qui a battu le dernier boss du jeu en utilisant un sort précis.

Un rendu toujours plus réaliste

Si Unity 4 permettait de créer des scènes 3D de grande qualité et très réalistes, Unity 5 met la barre encore plus haut en ajoutant quelques ingrédients magiques :

- *Physically-based Standard Shader* (PBR) : Comme ses concurrents, Unity permet désormais l'utilisation des shaders basés sur la physique pour une plus grande cohérence entre les éclairages et les matériaux,
- *High Dynamic Range Rendering* (HDR) : Cette fonctionnalité, qui permet d'ajuster les couleurs dans une scène pour avoir un rendu plus réaliste, devient encore plus puissante avec l'ajout du composant *ReflectionProbes* et du support des textures HDR aux formats .exr et .hdr,
- Ajout du *FrameDebugger* pour connaître précisément comment est fait le rendu d'une frame en visualisant les *draw calls,*
- Meilleure gestion des ombres,
- *Real-time Global Illumination* (GI) : Une gestion de la lumière en temps réel qui permet aux scènes d'être encore plus réalistes.

Fig.1.

Les sound designer vont être contents

La gestion des sons dans Unity était plutôt restreinte, ce n'est plus le cas avec l'apparition de *l'AudioMixer*.

- Une nouvelle fenêtre permet de créer des mix ainsi que de gérer les effets sonores
- Paramétrage des sons en mode édition ou directement en jeu,
- Possibilité d'enregistrer les paramètres audio afin de les réutiliser,
- Amélioration de l'Audio Profiler avec de meilleures statistiques et plus d'informations
- Support de l'édition multiple pour la classe *AudioClip,*
- Le chargement des données d'un *AudioClip* peut se faire à la demande via un script. Le chargement peut quant à lui se faire en arrière-plan sans bloquer la boucle de jeu.

Des animations encore plus paramétrables

Les animations sont très importantes dans un jeu et Unity Technologies l'a bien compris. Avec cette nouvelle version, l'éditeur va plus loin dans son système de « machine à états » et nous propose quelques petites nouveautés afin de nous simplifier le développement :

- Le composant *StateMachineBehaviours* fait son apparition, se reposant sur le même principe que le *MonoBehaviour* mais appliqué à une machine à états pour lui permettre d'avoir accès très facilement aux callbacks : *OnStateEnter*, *OnStateExit*, *OnStateUpdate*, *OnStateMove*, *OnStateIK*, *OnStateMachineEnter* et *OnStateMachineExit,*
- Refonte du *Blend Tree* pour pouvoir contrôler le poids de chaque animation indépendamment,
- L'ajout de deux nœuds de transitions (*Entry*, *Exit*) qui permettent de définir comment doit se comporter la machine à état en entrant et en sortant Fig.2,
- Les composants tels que *Controllers*, *StateMachines*, *BlendTree*, ... peuvent être créés et édités directement dans l'éditeur.

Un éditeur 64 bits pour encore plus de puissance

La grande nouveauté au niveau de l'éditeur c'est le passage aux 64 bits. La version 32 bits reste accessible pour la plateforme Windows, pour iOS tout est désormais en 64 bits. Ce changement n'est pas sans effet car il affecte tous les plugins natifs utilisés dans l'éditeur : ils doivent aussi passer au 64 bits. D'autres fonctionnalités font leur apparition dont certaines attendues de longue date :

- La possibilité de merger les scènes et les *prefabs* grâce à l'ajout d'un outil en ligne de commande,
- Intégration de l'outil de merge avec le gestionnaire de source utilisé dans Unity. L'activation se fait via le menu *EditorSettings,*

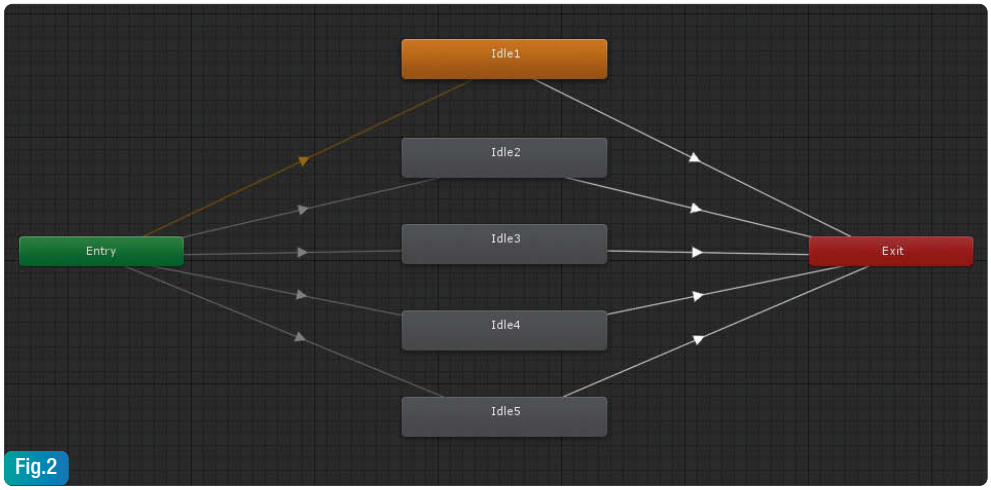


Fig.2

- Les plugins ne sont plus obligés d'être placés dans des dossiers spécifiques comme *Assets/Plugins/WP8*, *Assets/Plugins/X86*. Vous pouvez désormais les placer n'importe où et paramétrer le plugin en cliquant simplement dessus,
- Chaque thread s'affiche indépendamment dans la fenêtre *Timeline*, très pratique pour du multi-thread.

PhysX au service du mobile

L'intégration de PhysX 3 permet d'obtenir de meilleures performances sur les processeurs multi-cœurs et plus spécifiquement sur les mobiles. La physique 2D a elle aussi eu son lot de changements avec notamment :

- L'ajout des composants *ConstantForce2D*, *PointEffector2D*, *AreaEffector2D*, *PlatformEffector2D* and *SurfaceEffector2D,*
- L'ajout des méthodes Physics2D.IsTouching, *IsTouchingLayer* and *Collider2D.IsTouching*, *IsTouchingLayer,*
- La propriété *center* des *BoxCollider2D* et *CircleCollider2D* est remplacée par la propriété *offset.*

Une mise à jour de l'API transparente

L'API ayant évolué, Unity a mis en place un script automatique qui permet de modifier les scripts utilisant des méthodes obsolètes afin de les remplacer par les nouvelles. A noter que ces changements s'effectuent de manière transparente pour le développeur.
Parmi ces changements on retrouve un point sur lequel les équipes techniques d'Unity sont en désaccord depuis longtemps : les raccourcis vers les composants tels que *rigidbody, renderer, transform, etc.*

De ce fait, le code suivant :

```
void Start()
{
  rigidBody.velocity = Vector3.up;
}
```

Sera remplacé par :

```
void Start()
{
  GetComponent<Rigidbody>().velocity = Vector3.up;
}
```

WebGL & IL2CPP : la preuve que ça fonctionne

La fenêtre de compilation a un nouvel élément : WebGL ! Encore en preview, il est supporté par les dernières versions des navigateurs Chrome et Firefox. La technologie derrière tout ça est la même que pour iOS 64 bits : IL2CPP.
Unity a fait le pari de refondre petit à petit son architecture technique et de créer sa propre implémentation de la CLI (*Common Language Infrastructure*) comme .NET ou Mono.
Techniquement cela se traduit comme ceci :

- IL2CPP convertit les assemblies en code C++,
- Le compilateur C++ produit ensuite des binaires natifs.

Au lancement, le code est exécuté grâce à la machine virtuelle d'IL2CPP qui permet d'ajouter des fonctionnalités comme le Garbage Collector.

Conclusion

Cette nouvelle version permet à Unity de rester au contact de ses concurrents en misant beaucoup sur la possibilité de faire un rendu ultra réaliste. Elle n'oublie pas non plus les *sound designer* grâce à l'ajout de l'*AudioMixer*. L'introduction de WebGL laisse à penser qu'IL2CPP tient la route, à voir comment cela se passera pour les autres plateformes !

Xamarin Forms pour le développeur XAML

Xamarin Forms est un framework proposé par Xamarin qui, en plus du partage de code classiquement proposé par Xamarin, permet de partager sa logique de présentation via une API spéciale, un moteur de layout et une série de contrôles. Xamarin Forms a pour particularité de permettre l'utilisation du langage XAML pour décrire ses interfaces.

John Thiriet - *Consultant .NET / Formateur*
Cellenza
@johnthiriet - http://blog.cellenza.com

Cellenza

C'est donc tout naturellement que le développeur XAML sous Windows Phone ou WPF s'intéressera à Xamarin Forms dont la dernière version stable au moment où nous écrivons ces lignes est la *1.4.0.6341*.

XAML

Dans l'absolu, XAML n'est qu'un langage de sérialisation d'objets. C'est ainsi que chaque plateforme utilisant XAML a apporté ses propres variations et améliorations. Depuis WPF jusqu'aux applications Windows universelles, il n'a jamais vraiment été possible de réutiliser notre code XAML à 100%. Xamarin Forms ne fait pas exception, et je dirais même que c'est une variation majeure qui forcera le développeur à remettre en cause certains de ses acquis et à réapprendre certaines façons de faire. Je classerais donc les éléments du XAML de Xamarin Forms en trois grands ensembles.

Utilisable directement et connu

L'utilisation de XAML faite par Xamarin Forms induit un certain nombre de comportements connus et s'utilisant de la même manière que dans au moins un autre plateforme XAML.
On citera entre autres :

- Les pages et leur code-behind,
- Le Binding,
- Les Styles,
- Les ressources et les dictionnaires de ressources,
- Les Converters,
- Certains contrôles comme la Grid,
- Les ItemTemplates,
- La déclaration et l'usage des espaces de noms. Fig.3

Variations mineures

Certains éléments ressemblent à s'y méprendre à des éléments connus, et on apprend facilement à les utiliser, et ce, sans même lire la documentation. Cependant, ils contiennent suffisamment de différences pour qu'il soit intéressant d'y faire attention.

- Les Buttons n'ont pas de propriété Content mais juste une propriété Text,
- On utilise StackLayout en lieu et place de StackPanel,
- On écrit Label au lieu de TextBlock,
- Entry est utilisée au lieu de TextBox pour les champs texte d'une seule ligne,
- BindingContext s'utilise à la place de DataContext.

Fig.3

```
<ContentPage.Resources>
  <ResourceDictionary>
    <converters:InvertBooleanConverter x:Key="InvertBooleanConverter" />
  </ResourceDictionary>
</ContentPage.Resources>
```

Variations majeures

Ces éléments nécessitent de passer un temps important à lire la documentation et à se familiariser avec leurs utilisations tellement celles-ci sont différentes de l'usage connu ou attendu. Je reviendrai sur certains d'entre eux dans la suite de cet article.

- Triggers,
- Behaviors,
- Navigation,
- Cycle de vie,
- ListView,
- Les propriétés de dépendances et propriétés attachées,
- Les contrôles personnalisés.

Note : *Resharper est un produit qu'on ne présente plus et qui, dans sa version 9, ajoute le support du XAML de Xamarin Forms. Le support fourni par défaut dans Visual Studio ou Xamarin Studio est relativement limité et je ne peux que vous conseiller d'utiliser Resharper si vous pensez faire plus qu'un simple test en Xamarin Forms.*

Xamarin Forms a été créé avec le support aisé du MVVM en tête. Toutes les connaissances acquises lors de précédents projets sont réutilisables sans trop de remises en questions. Un moteur de binding, la gestion des commandes, les propriétés de dépendances ou encore l'interface INotifyPropertyChanged, tout est présent pour réutiliser un mode de

Fig.1

```
0 references | John Thiriet | 13 changes
public static class ViewModelLocator
{
    0 references | John Thiriet | 1 change
    public static LoginPageViewModel LoginPageViewModel
    {
        get { return Resolver.Resolve<LoginPageViewModel>(); }
    }
}
```

Fig.2

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             xmlns:viewModels="clr-namespace:MyProject.Mobile.ViewModels;assembly=MyProject.Mobile"
             x:Class="MyProject.Mobile.Pages.LoginPage"
             BindingContext="{x:Static viewModels:ViewModelLocator.LoginPageViewModel}"
             Title="{Binding PageTitle}"
             IsBusy="{Binding IsBusy}">
</ContentPage>
```

développement classique. Comme pour les autres plateformes XAML, l'utilisation de MVVM n'est en rien obligatoire et n'oblige pas à avoir un code-behind, cependant le gain est tellement notable qu'il serait dommage de s'en priver.
Il existe différents frameworks MVVM fonctionnant avec Xamarin Forms parmi lesquels les plus connus sont MVVMCross et MVVMLight. Il existe un support limité mais suffisant pour pas mal de projets de MVVM dans le projet XLabs dont il est question en fin de cet article.

Note : *Si comme moi vous utilisez un ViewModelLocator dans vos projets sa déclaration est légèrement différente. En effet, il vaut mieux déclarer ce dernier de manière statique.*

Fig.1 - Fig.2

Injection de dépendance

Tout comme Xamarin classique, Xamarin Forms permet d'accéder à toutes les fonctionnalités de la plateforme sous-jacente. Il est aussi possible d'accéder aux éléments graphiques natifs de la plateforme grâce à ce que l'on appelle les renderers. Pour ce faire, un petit moteur d'injection, le DependencyService, a été intégré. En dehors de cet usage, il n'est cependant pas vraiment utile. Dans le cadre de l'utilisation de MVVM, on lui préfèrera un vrai moteur d'injection tel que celui présent dans MVVMLight ou un de ceux proposés par les XLabs.

Cycle de vie d'une page et navigation

Le cycle de vie d'une page Xamarin Forms est différent du cycle de vie des plateformes sous-jacentes. Là où en Windows Phone on "surdéfinissait" les méthodes OnNavigatedTo et OnNavigatingFrom pour respectivement initialiser et nettoyer nos viewmodels, en Xamarin Forms on travaillera avec les méthodes OnAppearing et OnDisappearing.
Attention ! Ces méthodes ne sont pas équivalentes car on ne dispose pas, à la différence de Windows Phone, des informations liées au « sens » de la navigation. Cela pose un souci lorsque l'on utilise un contrôle Picker. Sous Xamarin Forms, le contrôle Picker permet de sélectionner un élément dans une liste. Sous Windows Phone, il est implémenté avec un ListPicker. Ce dernier a pour particularité de naviguer vers une nouvelle page pour afficher la liste des éléments, ce qui déclenchera les évènements OnAppearing et OnDisappearing de la page le contenant. Il faut donc mettre un soin tout particulier à la gestion des ces évènements.

```
/// <summary>
/// MarkupExtension utilisée pour la gestion de la localisation de l'application.
/// </summary>
[ContentProperty("Text")]
2 references | John Thiriet | 3 changes
public class TranslateExtension : IMarkupExtension
{
    private readonly ITextProvider _textProvider;

    0 references | John Thiriet | 1 change
    public TranslateExtension()
    {
        _textProvider = Resolver.Resolve<ITextProvider>();
    }

    /// <summary>
    /// Ressource à utiliser.
    /// </summary>
    2 references | John Thiriet | 3 changes
    public KnownResources Text { get; set; }

    /// <summary>
    /// Effectue la récupération du texte associée à la propriété <see cref="Text"/>.
    /// </summary>
    /// <param name="serviceProvider">Fournisseur de service.</param>
    /// <returns>Texte traduit.</returns>
    0 references | John Thiriet | 3 changes
    public object ProvideValue(IServiceProvider serviceProvider)
    {
        var translation = _textProvider.GetString(Text);
        return translation;
    }
}
```

Fig.4

Markup Extensions

Les markup extensions sont présentes en XAML depuis longtemps, ce sont les éléments entre accolades.
Par exemple, StaticResource ou DynamicResource sont des markup extensions.
En Xamarin Forms, on peut créer nos propres extensions. La documentation officielle de Xamarin utilise un très bon exemple qui est celui de la gestion de la localisation d'une application. Partant de la documentation officielle, on peut étendre un peu le modèle Fig.4.
On remarquera 2 éléments importants :

- La propriété Text n'est pas de type string mais d'une énumération KnownResources,
- On utilise l'injection de dépendance pour récupérer un objet se chargeant de fournir les textes traduits (ITextProvider).

Fig.5

Notre markup extension n'a donc ici aucune logique, cette dernière est à la charge de fournisseur de texte et cette logique est réutilisable.
Utiliser une énumération permet aussi d'avoir l'IntelliSense sur la propriété Text, ce qui évite de nombreuses fautes de frappe et permet de voir quelles ressources sont utilisées ou non.

Les UserControls

Les UserControls sont, en XAML, des agrégats de contrôles existants. Ils disposent d'un fichier XAML et de son code-behind. Ils sont en cela différents des contrôles dont il est fait mention dans cet article. Il n'existe pas en Xamarin Forms de template de fichier pour créer un UserControl. Il est cependant très aisé d'y arriver.
Pour cela il faut juste :

- Créer une page XAML.
- Changer le type de base dans le XAML et dans son code-behind par
- ContentView au lieu de ContentPage.

Les Behaviors

Les behaviors sont des morceaux de code-behind réutilisables à différents endroits de l'application. Ils sont extrêmement utiles et utilisés dans le cadre de MVVM. L'utilisation type d'un behavior est de faire le pont entre des événements de la vue et des commandes présentes dans le viewmodel. Cependant, ils ont à l'heure actuelle plusieurs problèmes en Xamarin Forms :

```
<ListView ItemsSource="{Binding ItemList}">
  <ListView.Behaviors>
    <behaviors:ListViewSelectionChangedToCommandBehavior
     Command="{Binding ShowDetailCommand,
     Source={x:Static viewModels:ViewModelLocator.MasterPageViewModel}}"/>
  </ListView.Behaviors>
  <ListView.ItemTemplate>
    <DataTemplate>
      <TextCell Text="{Binding Name}"/>
    </DataTemplate>
  </ListView.ItemTemplate>
</ListView>
```

Fig.6

```
public enum KnownResources
{
    HomePage_Title,
    HomePage_SubmitButton_Text,
}
```

Fig.5

- Ils n'héritent pas du BindingContext de l'objet sur lequel ils sont attachés,
- Certains types de propriétés de dépendances ne sont pas résolus correctement,
- Ils ne sont instanciés qu'une seule fois par vue et ce, quel que soit le nombre de fois où ils ont été déclarés.

Tous ces soucis font qu'ils sont, pour l'instant, assez difficiles à utiliser et cela ne serait-ce que pour faire un lien entre un changement d'objets sélectionnés dans une liste et une commande.

Fig.6

Propriétés attachées

Lors des débuts de WPF et ce avant que les behaviors n'existent, on utilisait des propriétés attachées pour ajouter des comportements à certains contrôles et factoriser le code-behind. A l'inverse des behaviors, cette technique fonctionne parfaitement en Xamarin Forms.
Ainsi, on peut, par exemple, rendre un Picker compatible avec MVVM ce qu'il n'est pas par défaut. Fig.7

Triggers

Trigger se traduit par déclencheur en français. Comme leur nom l'indique, ils se déclenchent suite à une modification de l'état de l'application et permettent d'effectuer un certain nombre d'actions. En WPF, lorsque l'on n'avait pas encore les visualstates, on changeait l'apparence de nos contrôles grâce aux triggers.
Il existe plusieurs types de triggers :

- Un EventTrigger se déclenche suite à événement,
- Un PropertyTrigger se déclenche suite à une modification d'une propriété d'un contrôle,
- Un DataTrigger se déclenche suite à une modification de données bindées,
- Un MultiTrigger se déclenche suite à de multiples modifications potentiellement conditionnelles.

En Xamarin Forms, si l'on souhaite changer la couleur d'un formulaire s'il est dans un état invalide ou même la couleur d'un bouton lorsque l'on appuie dessus, il faut penser trigger.
Attention cependant : certains styles systèmes peuvent surdéfinir ce que vous avez écrit dans vos trigger surtout sur Windows Phone.

Styles

Il y a deux niveaux de styles dans Xamarin Forms :

- Les styles Xamarin Forms.
- Les styles natifs.

Lorsqu'un style Xamarin Forms s'applique, il le fait après le style natif, ce qui peut avoir un impact notamment pour les triggers.
J'ai pris pour habitude d'aller modifier les contrôles par défaut de la plateforme de la manière la plus simple possible. Par exemple, sous Windows Phone, cela peut consister en la suppression des visualstates du template des buttons, de façon à ce qu'ils ne rentrent en conflit avec aucun de mes styles Xamarin Forms.
D'autre part, de la même manière qu'en Windows Phone, on peut déclarer un fichier App.xaml en Xamarin Forms et y stocker les ressources globales de l'application ce qui permet de les centraliser.

Contrôles et renderers

Lorsqu'il n'est pas possible de passer par un UserControl ou que les styles ne suffisent pas pour obtenir l'apparence souhaitée, il ne reste plus qu'à créer ses propres contrôles et renderers.
Les contrôles en XAML classique sont ce que l'on appelle des « lookless controls ». Ils ne sont en fait qu'une série de comportements. En XAML classique, on affecte une apparence à ces contrôles grâce à des templates de contrôle. En Xamarin Forms, le rôle des templates de contrôle est joué par les renderers.
Un renderer se situe dans le projet natif de chaque plateforme. Il y a donc autant de renderers pour un contrôle que de plateformes cibles. Il a accès à toutes les API natives et peut donc modifier à sa guise l'apparence du contrôle, et ce, en n'ayant pour seules limites que celles de la plateforme. Les renderers sont affectés aux contrôles grâce au DependencyService. Il est ainsi possible de modifier le rendu d'un contrôle Xamarin Forms sans modifier ce contrôle !
Bien que très puissants et probablement ce qu'il y a de plus performants, les renderers nécessitent de très bien connaître les API graphiques de chaque plateforme, ce qui les réserve aux plus expérimentés.

XLabs

Il arrive parfois que Xamarin Forms ne propose pas une fonctionnalité indispensable à votre application. Dans ce cas, il peut être intéressant de regarder ce qu'offre ce projet. Le projet XLabs regroupe un ensemble de fonctionnalités natives à chaque plateforme, mais exposées sous forme d'interface utilisable directement en Xamarin Forms.
Il offre, par exemple, un accès aisé aux fonctions de géolocalisation ou à la prise de photos. Il regroupe un certain nombre de contrôles personnalisés avec leurs renderers associés tels que le ImageButton ou le ExtendedLabel.
Il dispose aussi d'un certain nombre de services d'infrastructure et de modules permettant d'ajouter facilement des moteurs d'injection de dépendances, de cache ou de sérialisations plus complets et reconnus.
Il comporte aussi un framework MVVM assez simple viewmodel-first.

Conclusion

Xamarin Forms est un framework avec un cycle de développement très rapide. Il s'améliore presque toutes les semaines et son spectre d'application s'en trouve élargi. D'abord réservé à du prototypage, il est maintenant prêt à être mis en production pour des applications de gestion et de saisie de données de type formulaires.
Si familier et pourtant si différent, j'ai voulu dresser un portrait rapide des points importants à connaître en Xamarin Forms afin d'éviter à d'autres de tomber dans les mêmes pièges que moi lorsque j'ai débuté.

Fig.7

```
<Picker x:Name="picker"
        Title="{markupExtensions:Translate LoginPage_CountryPicker_Title}"
        IsEnabled="{Binding IsBusy, Converter={StaticResource InvertBooleanConverter}}"
        attachedProperties:PickerItemsSource.HandleSelectionEvent="True"
        attachedProperties:PickerItemsSource.SelectedItem="{Binding SelectedCountry, Mode=TwoWay}"
        attachedProperties:PickerItemsSource.ItemsSource="{Binding CountryList}"
        attachedProperties:PickerItemsSource.DisplayMemberPath="FormattedName"/>
```

Xamarin : gérer la rétro-compatibilité d'une application Android Lollipop et Material Design.

Dans cet article, nous irons étape par étape dans la création d'une application Android 5.0 (Lollipop) utilisant quelques recettes du MD (Material Design), rétro-compatible jusqu'à Android 4.0 (Jelly Bean) grâce aux bibliothèques Support prévues à cet effet. Pour rappel, MD est un guide détaillé sous forme de langage visuel, qui agrège les meilleures pratiques de design et d'expérience utilisateur pour des applications Web, bureau et mobile.

Andreï TALANTSY-VIYENE
Cellenza

Cellenza

Télécharger la dernière version du SDK

Depuis l'éditeur Xamarin Studio ou Visual Studio, sélectionner l'outil "Android SDK Manager" et vérifier que les éléments suivants sont installés et à jour :

- Android SDK Tools
- Android SDK Platform-tools
- Android 5.0.1 (API 21)
- SDK Platform

Ensuite, revenant à l'éditeur, créer un nouveau projet de type Application Android. Depuis les propriétés du projet, dans le menu Application Android, modifier la version Android minimale supportée par l'application : choisir Android 4.0.

Référencer la (les) bibliothèque(s) Support

Pour supporter les nouveautés des dernières versions du SDK Android, dont Lollipop, sur les versions antérieures jusqu'à la 4.0, il est nécessaire de référencer les libs Support fournies par Google et adaptées en C# :

- Depuis le component Store : saisir "Support Library" dans la recherche.
- Sélectionner et installer : "Android Support Library v7 AppCompat".

Remarque : *Il existe plusieurs libs préfixées v7, chacune apportant des composants supplémentaires ajoutés dans les derniers SDK Android.*

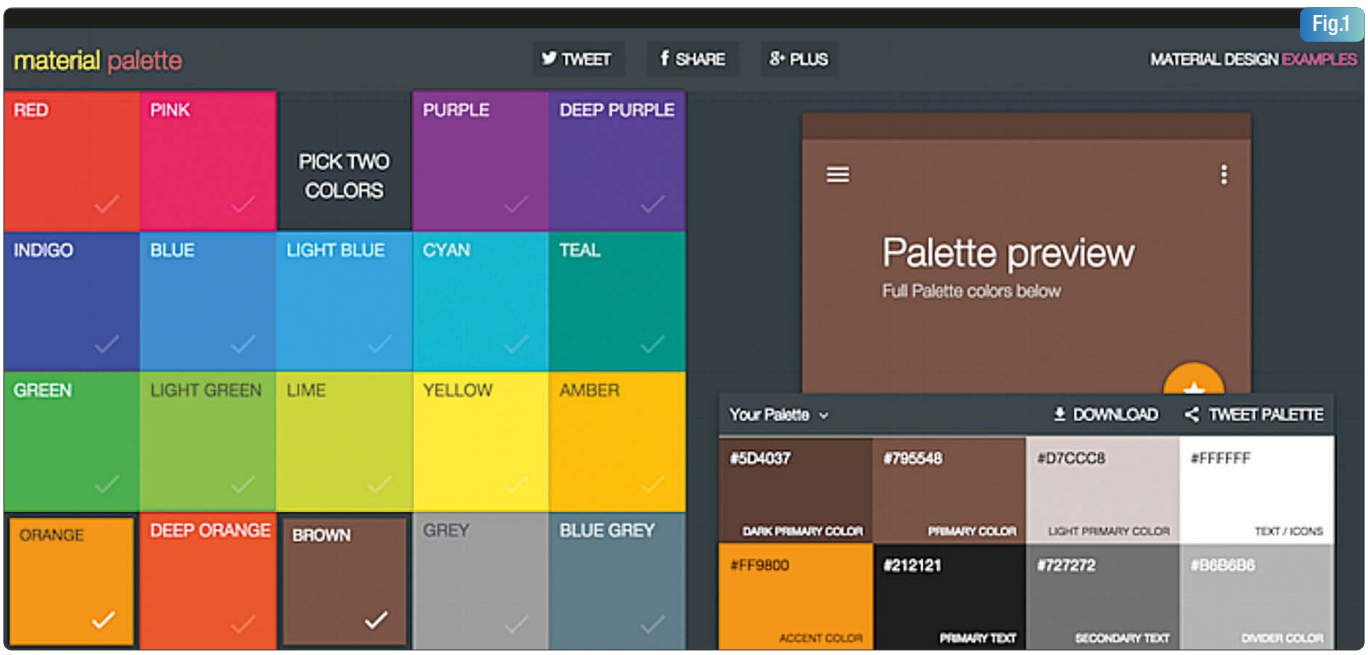
Créer un thème.

D'après les règles du MD, nous avons besoin d'identifier au minimum 3 couleurs primaires pour constituer la palette de couleurs de notre application. Il s'agit des couleurs de type : Primary, PrimaryDark et Accent. Le choix de ces couleurs étant bien codifié, nous utiliserons donc un outil en ligne pour nous aider :
http://www.materialpalette.com/ nous permet de sélectionner deux couleurs, et génère pour nous une palette complète : Fig.1.
Dans le répertoire Resources, il existe un dossier Values dans lequel nous ajouterons les couleurs et thèmes de l'application. Nous créerons également un dossier Values-v21 qui contiendra des ressources spécifiques à Lollipop. Les fichiers à créer ou à modifier sont :

- Resources\values\Colors.xml pour y ajouter les couleurs sélectionnées précédemment :

```
<resources>
  <color name="color_primary">#795548</color>
  <color name="color_primaryDark">#5D4037</color>
  <color name="color_accent">#FF9800</color>
</resources>
```

Fig.1



- Resources\values\Styles.xml pour ajouter un thème :

```
<resources>
<style name="AppThemeBase" parent="Theme.AppCompat.Light.NoActionBar">
  <item name="colorPrimary">@color/color_primary</item>
  <item name="colorPrimaryDark">@color/color_primaryDark</item>
  <item name="colorAccent">@color/color_accent</item>
</style>

<style name="AppTheme" parent="AppThemeBase">
</style>
</resources>
```

Il est intéressant de noter que nous avons défini un thème parent : AppThemBase et un autre qui en hérite. Le principe est que AppThemeBase s'applique quelle que soit la version du framework, et les sous-thèmes ajouteront éventuellement des propriétés spécifiques à la version d'API cible. Le fichier qui suit en est un exemple :

- Resources\values-v21\Styles.xml: dans ce fichier, nous ajouterons des paramètres liés à la transition entre vues (spécifiques à Lollipop) :

```
<resources>
<style name="AppTheme" parent="AppThemeBase">
  <item name="android:windowContentTransitions">true</item>
  <item name="android:windowAllowEnterTransitionOverlap">true</item>
  <item name="android:windowAllowReturnTransitionOverlap">true</item>
  <item name="android:windowSharedElementEnterTransition">@android:transition/move</item>
  <item name="android:windowSharedElementExitTransition">@android:transition/move</item>
</style>
</resources>
```

Il faut préciser également que le thème final (utilisé dans la vue) doit être présent au moins dans values\Styles.xml par défaut, sinon la compilation échouera pour des versions d'API autres que la 21.

Remarque: *le thème déclaré est de type NoActionBar, ce qui sous-entend que la barre de titre et de menu du haut ne sera pas incluse dans la vue par défaut. Cette barre est traditionnellement gérée par le composant système ActionBar, mais ce dernier tend à être délaissé au profit du widget Toolbar, plus personnalisable et réutilisable.*

La prochaine action est accessoire. Nous allons simplement ajouter du texte en ressources. Pour ce faire, nous créerons le fichier Resources\Strings.xml s'il n'existe pas et ajouterons le contenu suivant :

```
<resources>
  <string name="main_title">HELLO MATERIAL</string>
  <string name="main_subtitle">lorem ipsum dolor</string>
</resources>
```

Déclarer une ressource de vue ou layout

Il s'agit ici de déclarer et de disposer les composants que nous manipulerons dans le code de vue. Créer une ressource de type **layout**, Main.axml sur le projet :

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  android:orientation="vertical"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent">
  <android.support.v7.widget.Toolbar
    android:id="@+id/toolbar"
    android:layout_width="match_parent"
    android:layout_height="128dp"
    android:paddingLeft="72dp"
    android:paddingBottom="16dp"
    android:gravity="bottom"
    android:background="?attr/colorPrimary"
    app:titleTextAppearance="@style/TextAppearance.Widget.AppCompat.Toolbar.Title"
    app:subtitleTextAppearance="@style/TextAppearance.Widget.AppCompat.Toolbar.Subtitle"
    app:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar"
    app:popupTheme="@style/ThemeOverlay.AppCompat.Light" />

  <LinearLayout
      android:layout_width="fill_parent"
      android:layout_height="wrap_content">

    <android.support.v7.widget.SwitchCompat
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"/>
    <android.support.v7.widget.SwitchCompat
      android:layout_marginLeft="16dp"
      android:checked="true"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content" />
  </LinearLayout>

  <LinearLayout
      android:layout_width="fill_parent"
      android:layout_height="wrap_content">

      <CheckBox android:layout_width="wrap_content"
          android:layout_height="wrap_content"/>
      <CheckBox android:layout_marginLeft="16dp"
          android:checked="true"
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"/>
  </LinearLayout>

    <EditText
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Saisir votre texte"/>
</LinearLayout>
```

Analysons en détail les composants déclarés ci-dessus :

- xmlns:app : cette définition de namespace permet au compilateur d'étendre la référence de ressources à celles venant des libs externes, dont la lib Support v7 notamment, qui apporte des propriétés supplémentaires appliquées dans ce **layout**.
- android:background : Sur cette propriété, nous appliquons l'attribut système colorPrimary. qui prendra sa valeur dans le thème déclaré précédemment.
- app:* : Les propriétés valorisées sous ce namespace sur le widget Toolbar permettent d'appliquer un style sur (respectivement) :
  - Le titre et sous-titre du composant,
  - Le composant lui-même, avec les propriétés d'un **ActionBar,**
  - Le menu déroulant lorsqu'il est ajouté.

Nous avons ensuite ajouté en plus un Switch (version améliorée par rapport à celle par défaut), un Checkbox et un champ texte. Nous verrons dans le rendu final, comment ces controls appliquent le thème défini précédemment.

Ecrire le code de vue ou Activity

Nous pouvons à présent terminer le travail de tuyauterie par du code dans la classe MainActivity (à créer si non présente).
Pour pouvoir utiliser les fonctionnalités de l'**ActionBar,** nous allons faire hériter la vue de la classe ActionBarActivity, fournie par la lib support v7 :

```
[Activity (Label = "@string/main_title", Theme = "@style/AppTheme", MainLauncher = true, Icon = "@drawable/icon")]
public class MainActivity : Android.Support.V7.App.ActionBarActivity
{
  protected override void OnCreate (Bundle bundle)
  {
    base.OnCreate (bundle);

    SetContentView (Resource.Layout.Main);
    var toolbar = FindViewById<Android.Support.V7.Widget.Toolbar> (Resource.Id.toolbar);
    SetSupportActionBar (toolbar);
    toolbar.SetSubtitle (Resource.String.main_subtitle);
  }
    public override bool OnCreateOptionsMenu (Android.Views.IMenu menu)
    {
      var actionItem = menu.Add(new Java.Lang.String ("Partage"));

      actionItem.SetIcon(Android.Resource.Drawable.IcMenuSearch);
      MenuItemCompat.SetShowAsAction(actionItem, MenuItemCompat.ShowAsActionIfRoom);

      menu.Add (new Java.Lang.String ("Plus"));
      return true;
    }
}
```

Entrons un peu plus dans le détail de ce code :

- Theme : cette propriété de l'attribut Activity nous permet de faire référence au thème déclaré précédemment,
- SetSupportActionBar : cette méthode nous permet d'activer un **ActionBar** en remplaçant le composant système par le widget **Toolbar** déclaré dans le layout,
- Dans la méthode OnCreateOptionsMenu : nous ajoutons deux options de menu. La particularité ici c'est l'utilisation de la classe MenuItemCompat, qui n'ajoute rien de plus que le composant de base, mais inclut la rétro-compatibilité liée à cette fonctionnalité.

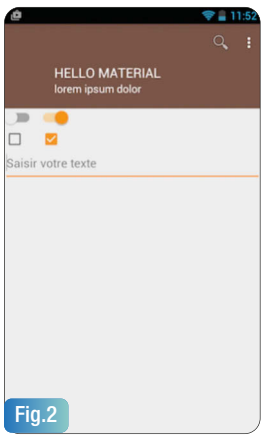


Fig.2

Nous pouvons à présent exécuter l'application depuis le simulateur ou le terminal Android de votre choix, et elle devrait ressembler à cela : Fig.2.

Pour aller plus loin

Le résultat de cette configuration est bien modeste, mais elle pose la base pour le support des dernières nouveautés du framework Android sur vos applications. Veuillez noter que toutes les nouveautés de Lollipop n'ont pas été portées sur les bibliothèques Support, notamment les nouveaux types de ressources comme les animations ou les images vectorielles. Mais en attendant qu'elles le soient, il y a bien des sujets à explorer comme la palette de couleurs, les animations de transition et bien d'autres.

Voici quelques références qui reprennent plus en détail les concepts énoncés dans cet article:
http://android-developers.blogspot.fr/2014/10/appcompat-v21-material-design-for-pre.html
http://www.google.com/design/spec/material-design/introduction.html
http://www.materialup.com/

© samsung

En partenariat technique avec Infinite Square, Guyzmo, EBLM

Nouveau
PROGRAMMEZ !
sur mobile et desktop

ANDROID APP ON Google play
ANDROID

WINDOWS 8.X

WINDOWS PHONE

Java 8 : le jour d'après

Java 8 est disponible depuis un an. Et cette année, Java fête son 20e anniversaire... Eh oui ! Déjà! sur la couverture du premier numéro de Programmez ! (1998), Java était à l'honneur, avec un serveur Java. Depuis, nous n'avons jamais cessé de parler de ce langage.

Les avis sont souvent très partagés sur Java, ceux qui aiment, ceux qui détestent, et, au milieu, ceux qui hésitent. Malgré tous ses défauts, Java s'est imposé comme un langage incontournable des écoles d'informatique à l'entreprise... Des millions de codes s'exécutent chaque minute. Il fait partie du patrimoine informatique et il le restera encore longtemps.

Java est comme le C++ ou Cobol. Maintes fois sa mort a été annoncée, et, à chaque fois, il a résisté... De nombreuses polémiques et des failles de sécurité l'ont fortement ébranlé, jusqu'à pousser des éditeurs à désactiver par défaut Java des navigateurs Web ! Il a perdu beaucoup de terrain sur les sites Web au profit de JavaScript et de HTML 5.

Depuis le rachat de Sun par Oracle, et donc de l'univers Java, de nombreux doutes se sont fait jour, parfois à juste raison : agenda incertain, retards successifs de Java 8, décalage de fonctions, un Java EE incertain, procès Oracle – Google, etc. Malgré une communauté toujours active (il n'y a qu'à voir l'effervescence autour de Devoxx), Java traverse des turbulences. Il est omniprésent sur Android et sur Hadoop, mais ne s'agit-il pas d'arbres qui cachent une certaine misère ? Si Android abandonne un jour Java, l'impact en serait considérable.

Oracle va tenter de tenir la cadence des nouvelles versions : une version majeure tous les 2 ans. Java 9 est attendue pour l'automne 2016... Les premières pré-versions sont disponibles sur java.net.

Java doit-il se repenser, un peu comme Apple avec son nouveau langage Swift, pour proposer une alternative plus « facile » et plus claire que Objective-C ? Java 7 avait débuté un nettoyage que Java 8 a poursuivi. Il en faudra sans doute bien plus pour que Java garde sa place.

La rédaction.

Salaires des développeurs Java

Java reste un profil très recherché, notamment à cause du patrimoine Java existant dans les entreprises et la mobilité avec Android. Dans tous les index de popularité, Java est dans les premiers langages.

Côté salaire, peu de surprises. Selon les chiffres de chooseyourboss.com :

- 0 à 1 an d'expérience (junior) : 35 à 40 000 € / an
- 2 à 5 ans d'expérience (intermédiaire / confirmé / senior) : 40 à 50 000 € / an
- 6 ans et + (expert) : 50 à 75 000 € / an

Les dernières études d'urbanlinker.com présentent des salaires inférieurs :

- junior : 35-39 k€
- senior : 39-52 k€
- expert : 48-70 k€

On peut tout de même trouver des offres d'emploi pour un développeur confirmé à 31-36 000 € (région Bordeaux) ou encore de 40 à 55 k€ pour un développeur avec 2 ans d'expérience (Paris). Les salaires proposés seront donc très hétérogènes. Soyez vigilant.

Les 20 ans de Java

1991-1995 : LA NAISSANCE

Sun Microsystem veut construire des objets connectés. Le constructeur a besoin d'une plateforme logicielle complète. James Gosling travaille dessus, avec une petite équipe. C'est dans cette ambiance que naît le projet Oak, un nouveau langage de programmation pour ces objets et matériels intelligents. Initialement, le C++ était envisagé. En 1992, l'équipe dévoile un projet d'assistant numérique (le PDA) avec un nouveau système et le langage Oak. En 1994, un problème juridique oblige Sun à choisir un nouveau nom pour le langage Oak. Ce sera Java !

1994 : AUX ORIGINES DE HOTJAVA

En 1994, l'équipe de Gasling développe un navigateur Web : Webrunner. Il s'appellera HotJava et sera basé sur Java. Sa première démo officielle aura lieu en 1995.

FÉVRIER 1995 : « JAVA : AN OVERVIEW » PAR JAMES GOSLING

Pour beaucoup, cet article est l'acte fondateur de Java. Gosling décrit ainsi Java : un langage simple, orienté objet, distribué, interprété, robuste, sécurisé, avec une architecture neutre, portable, capable de hautes performances, un système multithread. Et il s'agit d'un langage dynamique. Le but est faire plus simple que C et C++. La notion de « ramasse-miette » est importante car elle doit simplifier la gestion mémoire, cauchemar de nombreux développeurs.

PRINTEMPS 1995

Java 1.0 est disponible. Java sera intégré par défaut dans le navigateur Netscape. La page java.sun.com apparaît sur Internet.

1996

Une nouvelle version arrive. L'année est marquée la première conférence développeur Java : la JDC, connue aujourd'hui sous le nom de JavaOne.

1997

Java 1.1. Symantec sort son outil visuel de développement Java : Visual Café (for Java). Borland sort JBuilder 1.0.

1998

Java 1.2 sort officiellement en décembre 1998. Son appellation change aussi pour : Java 2 Platform Standard Edition. Java 1.2 est en réalité Java 2 ! Cette version supporte pour la première fois la bibliothèque d'interface Swing. Et les premiers compilateurs JIT (Just in Time) apparaissent !

Cette année-là, Sun décide de créer un organe très important : le Java Community Process. Son objectif est de coordonner et de faire évoluer le langage et l'ensemble de la plateforme Java.

Durant la conférence JDC, la bague Java est montrée sur scène, la fameuse Java Ring. Sun dévoile un autre projet : Java Professionnal Edition (JPE)

1999

En Décembre 1999, Sun sort la version 1.2 de l'édition entreprise, J2EE. Elle inclut déjà de nombreux modules et composants : servlet Java, JMS, JSP, JAF, EJB. L'évolution de J2EE sera plus lente que la version standard. HotSpot s'installe.

2001

Eclipse 1.0 est disponible sur Windows et Linux.

2004

Java 5.0 va être un des grands tournants du monde Java. Cette version fait un peu le ménage dans les API et le langage. Elle introduit les génériques, les annotations.

2006

Java SE 6 supporte maintenant les langages de script. La première version du serveur d'applications de Sun, GlassFish, est disponible.

2008

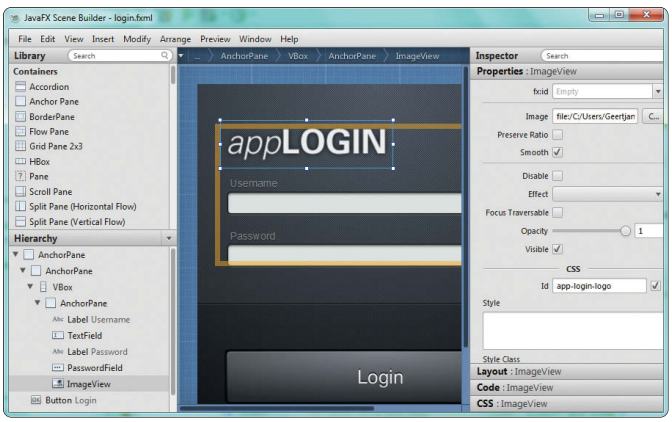
Dans une mise à jour de Java SE 6, JavaFX, la boite multimédia, arrive...

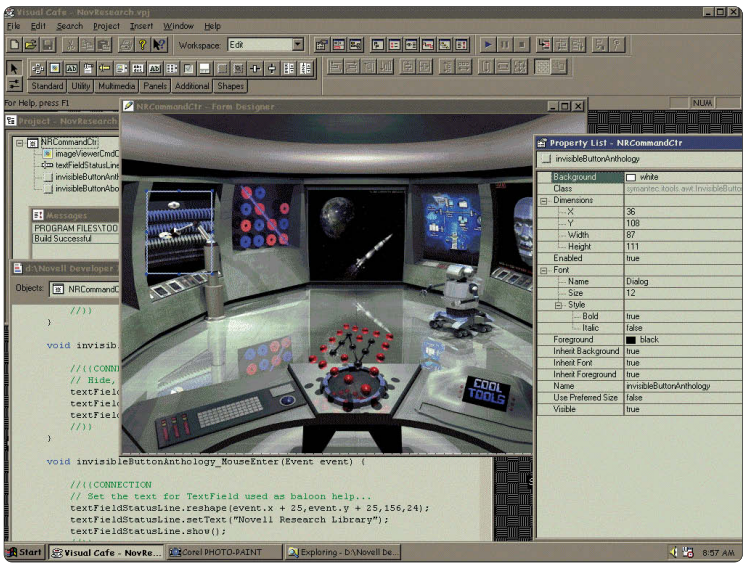
2011

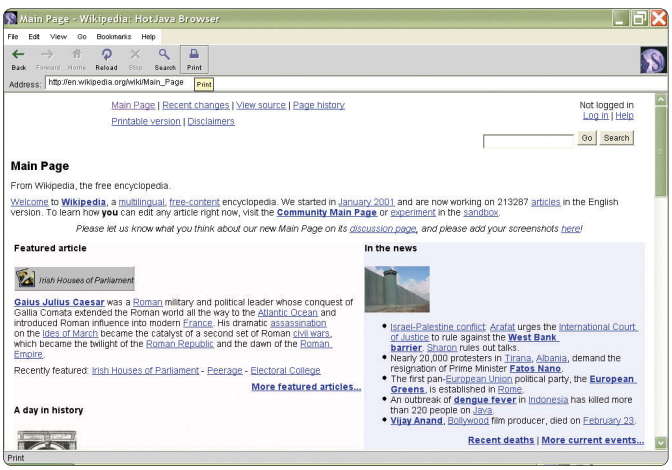
Java SE 7 sort. Parmi les grosses nouveautés, la machine virtuelle Java accepte les langages dynamiques.

MARS 2014

Java 8 sort enfin après de multiples retards et des coupes dans les fonctions.







Bilan et perspectives un an après la sortie de Java 8

Le 18 Mars 2014, Java 8 débarquait enfin après plus de 3 ans d'attente depuis la version 7 du langage. Cette longue attente aura fait naître un engouement certain auprès des développeurs Java désireux de bénéficier au plus vite des nouveautés majeures que constituaient les Lambdas et les Streams. Un an après, avec un peu de recul, il est temps de tirer un premier bilan de cette mouture tant attendue, et d'aller plus loin en s'attardant sur les perspectives concernant l'avenir de Java.

Sylvain SAUREL
Ingénieur d'Etudes Java / Android
sylvain.saurel@gmail.com – www.all4android.net

Lors de la sortie de Java 8 en Mars 2014, la communauté Java était excitée pour 2 raisons principales. La première était l'arrivée des Lambdas expressions mettant à portée des développeurs Java la programmation fonctionnelle. Cette direction prise par Oracle était un signal fort envoyé à la communauté puisque validant les principes au cœur du succès de langages comme Scala, Erlang ou Haskell. La seconde raison étant les importants efforts réalisés par Oracle pour embrasser le monde du multi-cœur. Tous les acteurs du monde Java s'accordent à dire que Java 8 connaît une adoption rapide depuis son lancement. De nombreux signes sont là, tels que le nombre important de projets open source basculant vers Java 8, ou encore la distribution Linux Fedora travaillant sur la mise en place de Java 8 comme runtime Java par défaut. Néanmoins, cela reste simplement des signes, et il est bon de confronter ces impressions à des chiffres concrets. La société Typesafe, responsable du développement et du support du langage pour la JVM Scala, a donc réalisé une enquête auprès de 3000 professionnels du monde Java en fin d'année 2014 afin d'avoir un retour concret sur l'adoption de Java 8. Les résultats obtenus confirment la tendance d'une adoption rapide. En effet, près de 30% des équipes interrogées sont déjà passées à Java 8. En sus, plus de 20% des équipes restantes envisagent de le faire dans un délai de 6 mois. L'année 2015 devrait donc confirmer cette tendance puisque les obstacles rencontrés par une mise à jour sont plutôt liés à des incompatibilités avec les infrastructures en place. Les retours de l'enquête confirment bien également que les Lambdas étaient la fonctionnalité la plus attendue de Java 8 puisque plus de la moitié des professionnels sondés les ont déjà mis en place, ou sont en phase de test pour étudier leurs avantages en termes de productivité qu'elles apportent. Le rapport révèle également que la communauté attend beaucoup d'Oracle pour les versions futures de Java, avec une attente particulièrement forte autour de la modularité et du projet Jigsaw sur lequel nous reviendrons plus tard dans cet article.

Lambdas

Nouveauté majeure de Java 8, les Lambdas expressions auraient pu être un simple changement syntaxique, mais elles sont en réalité bien plus que ça. Les architectes en charge du langage ont ainsi fait un gros travail autour des Lambdas pour impacter en profondeur le langage, les bibliothèques, mais également la JVM. Elles ont un impact positif sur les performances, tout en apportant concision et productivité aux développeurs. Mieux encore, elles amènent la touche de programmation fonctionnelle qui faisait tant défaut au langage. La montée en puissance de langages basés sur la JVM tels que Scala ou Groovy aura ainsi eu pour effet de pousser Oracle à enfin intégrer ce paradigme. Concrètement, avant les Lambdas, le tri d'une liste de strings se faisait à l'aide d'une instance de Comparator comme suit :

```
class LengthComparator implements Comparator<String> {
 public int compare(String first, String second) {
  return Integer.compare(first.length(), second.length());
 }
}

LengthComparator lengthComparator = new LengthComparator();
Arrays.sort(strings, lengthComparator);
```

En utilisant une Lambda expression, ce code peut être simplifié de la sorte :

```
Arrays.sort(strings, (String first, String second) -> Integer.compare(first.length(), second.length());
```

Streams

Les Lambdas limitées à cette utilisation auraient pu être considérées comme un simple changement syntaxique. A contrario, pour tirer pleinement parti de la puissance des Lambdas, les Streams ont également été ajoutés au langage. Un Stream représente une séquence de valeurs, et expose un ensemble d'opérations d'agrégations autorisant le développeur à exprimer des manipulations classiques sur ces valeurs de manière simple et claire. L'API Collections du JDK a ainsi été entièrement mise à jour pour intégrer cette notion particulièrement puissante. L'ensemble du JDK a été impacté et c'est un travail considérable qui a été réalisé par les équipes d'Oracle. Pour faciliter ce travail, les architectes du langage ont introduit les méthodes virtuelles d'extension aux interfaces existantes. En pratique, les Streams donnent la possibilité aux développeurs de réaliser simplement des opérations chaînées au sein de pipelines :

```
shapes.stream().filter(s -> s.getColor() == BLUE).forEach(s -> s.setColor(RED);
```

Ici, la méthode stream de la Collection shapes produit une vue Stream des éléments de la collection. La méthode filter produit un Stream filtré ne gardant que les objets Shape bleus, et, enfin, on utilise une itération interne via un appel à forEach pour appliquer la couleur rouge sur ces instances de Shape. On le voit clairement, la combinaison des Streams et des Lambdas permet des constructions centrées sur le code métier. Outre ces possibilités, les Streams offrent également des opérations de type map / reduce, ce qui s'avère particulièrement précieux pour des opérations sur des listes ou des maps. On peut ainsi considérer l'exemple suivant :

```
double average = roster.stream()
        .filter(p -> p.getGender() == Person.Sex.MALE)
        .mapToInt(Person::getAge)
        .average()
        .getAsDouble();
```

Etant donné une liste de personnes représentées dans l'objet roster, on souhaite calculer l'âge moyen des personnes de type masculin. Une fois le Stream de la liste récupéré, on applique un filtrage sur les personnes de type MALE avant d'obtenir le Stream correspondant aux âges des personnes filtrées; ceci grâce à l'utilisation de l'opération intermédiaire mapToInt avec en entrée la référence de méthode Person::getAge. Reste ensuite à calculer la moyenne du Stream d'entiers en appelant l'opération terminale average.

Streams parallèles

La volonté affichée d'Oracle de voir la plateforme embrasser les enjeux du multi-cœur en rendant le parallélisme plus accessible aux développeurs, aura conduit à l'introduction des Streams parallèles avec Java 8. Le but des Streams parallèles est de laisser les développeurs se concentrer sur la description des opérations à réaliser en les abstrayant de la manière dont elles seront exécutées. Le tout visant à faciliter les traitements parallèles au maximum sans toutefois les rendre complètement invisibles pour les développeurs. Suivant cette volonté, les opérations intermédiaires des Streams qui sont traitées au sein de pipelines peuvent être exécutées en série ou en parallèle. Le basculement entre ces modes se faisant via un simple choix au moment de la création du Stream. Par défaut, les implémentations au sein du JDK retournent des Streams séquentiels. Pour accéder à un Stream parallèle, il faut invoquer la méthode parallelStream(). En reprenant l'exemple précédent sur le calcul de l'âge moyen des personnes de type masculin, son exécution en parallèle se fait comme ceci :

```
double average = roster.parallelStream()
        .filter(p -> p.getGender() == Person.Sex.MALE)
        .mapToInt(Person::getAge)
        .average()
        .getAsDouble();
```

Cet accès facilité à la programmation parallèle ne devra cependant pas faire perdre de vue aux développeurs que les sources de données associées aux Streams sont généralement mutables, et qu'il existe donc des possibilités d'interférence si la source est modifiée pendant l'exécution des traitements d'un Stream. Le contenu d'une source de données associé à un Stream devra donc rester constant lors de l'exécution de ses opérations.

Interfaces fonctionnelles

L'introduction de la programmation fonctionnelle dans Java 8 aura mis en exergue la nécessité d'avoir un certain nombre d'interfaces fonctionnelles prêtes à l'emploi dans le JDK standard. Ces interfaces s'avèrent essentielles lors des opérations de type map / reduce ou filter sur des Streams. Les interfaces fonctionnelles proposées en standard sont organisées dans les catégories suivantes :

- Function<T, R> : prend en entrée un paramètre T et retourne un objet R,
- Predicate<T> : prend en entrée un paramètre T et retourne un booléen en sortie,
- Consumer<T> : prend en entrée un paramètre T, effectue un certain nombre de traitements et ne retourne rien,
- Supplier<T> : aucun paramètre en entrée et retourne un objet T,
- BiFunction<T, U, R> : prend 2 paramètres en entrée respectivement de type T et U, renvoie un objet R,

Dans nos exemples précédents, nous avons ainsi eu recours à l'interface fonctionnelle Predicate pour filtrer les personnes. Elles sont donc bien indissociables des Lambdas et des Streams.

API Date and Time

Historiquement, la plateforme Java a toujours souffert d'un support des dates et des heures tout juste passable. Apparues durant les premières années de Java, l'API java.util.Date et l'API java.util.Calendar souffrent de manques historiques liés à des erreurs de design qui rendent la vie impossible aux développeurs Java depuis de nombreuses années. Après une longue attente, la lumière est enfin apparue au bout du tunnel avec l'introduction de l'API Date and Time au sein de Java 8. Basée sur la bibliothèque Joda-Time, qui s'était imposée de facto comme la solution de référence durant les années 2000, cette API aura permis de repartir de zéro et de s'intéresser avant tout au temps de manière conceptuelle. Le résultat est une API parfaitement conçue et articulée autour de 3 idées centrales :

- Classes immutables. Une des principales faiblesses des formateurs de l'API date originelle provenait du fait qu'ils n'étaient pas thread-safe. Ceci imposait aux développeurs de devoir les utiliser de manière thread-safe et avait tendance à complexifier leur développement quotidien.
- Conception liée au domaine. La nouvelle API modélise parfaitement son domaine avec des classes aux noms adéquats, et représentant correctement les différents cas d'utilisation que ce soit pour les dates ou le temps. Finies les confusions liées à l'ancienne API qui proposait une classe Date pour manipuler un concept n'étant pas une date et une méthode toString() suggérant une gestion des fuseaux horaires alors que ceux-ci n'étaient pas gérés.
- Séparation des chronologies. La nouvelle API autorise les développeurs à travailler avec différents systèmes de calendriers afin de prendre en compte les besoins des utilisateurs dans le monde entier.

La nouvelle API s'articule autour de 5 packages :

- java.time qui est le package de base qui contient les objets de type valeur,
- java.time.chrono qui donne accès aux différents systèmes de calendriers,
- java.time.format pour le formatage et le parsing des dates et des heures,
- java.time.temporal représentant le bas niveau du framework et qui permet d'étendre ses fonctionnalités,
- java.time.zone fournissant les classes de support pour la gestion des fuseaux horaires.

Au niveau conceptuel, l'API s'articule autour des concepts de date, d'heure, d'instant, de durée et de période temps, tout en supportant les systèmes de calendriers alternatifs ainsi que les notions de fuseaux horaires. En outre, des classes utilitaires facilitant le formatage et le parsing sont proposées. Le résultat final est un succès, et l'API a déjà été largement adoptée. De nombreuses équipes ont désormais laissé Joda-Time de côté pour revenir au nouveau standard proposé par Java 8.

Nashorn

L'apparition du moteur JavaScript Nashorn pour la JVM constitue la dernière nouveauté majeure apportée par Java 8 pour les développeurs. Jusqu'à Java SE 7, le JDK proposait une implémentation basée sur le moteur Rhino de Mozilla. Le nouveau moteur Nashorn tire quant à lui parti de la JSR 292 de Java 7 et des possibilités offertes par invokedynamic. Il offre également un meilleur support de la norme ECMA et de meilleures performances puisque basé sur invokedynamic. Concrètement, Nashorn peut s'utiliser en mode ligne de commande via l'exécutable jjs fourni par le JDK, ou bien sous la forme d'un interpréteur embarqué au sein d'une application Java. Le mode ligne de commande permet par exemple de créer un code Javascript utilisant des APIs Java standards et proposant une IHM basée sur Swing ou JavaFX ! Les possibilités n'ont de fait nulle autre limite que l'imagination des développeurs Java. Au niveau d'une application Java, l'utilisation se fera peu ou prou de la même manière que ce qui se faisait avec Rhino puisque que l'API permettant d'utiliser un moteur de scripting n'a pas changé. On peut donc proposer l'exemple suivant pour déclarer une fonction Javascript réalisant la somme de 2 nombres et appeler cette fonction :

```
import javax.script.ScriptEngine;
import javax.script.ScriptEngineManager;

public class Hello {

  public static void main(String... args) throws Throwable {
    ScriptEngineManager engineManager =
```

```
new ScriptEngineManager();
    ScriptEngine engine =
engineManager.getEngineByName("nashorn");
    engine.eval("function sum(a, b) { return a + b; }");
    System.out.println(engine.eval("sum(1, 2);"));
  }
}
```

Un an après la sortie de Java 8, on ne peut pas dire que Nashorn aura été une réelle révolution. Une évolution intéressante pour la plateforme tout au plus. Beaucoup de développeurs n'étant pas forcément impactés puisque les cas d'utilisation ne sont pas forcément très courants dans les applications Java d'entreprise.

Amélioration de la sécurité et des performances

Moins connu du grand public, Java 8 aura également permis un grand pas en avant pour la sécurité de la plateforme avec des améliorations importantes de la cryptographie dans le cadre d'un projet de modernisation de la cryptographie initié par la NSA. Des améliorations au niveau de la gestion des keystores ont également été apportées. Enfin, la classe SecureRandom a été mise à jour avec l'introduction de la méthode getInstanceStrong() qui retourne une instance de SecureRandom plus sécurisée, car offrant un meilleur support pour l'entropie des nombres aléatoires. Côté performance, le JDK 8 gagne en efficacité grâce au nouveau garbage collector G1 et à la suppression de la Permanent Generation. Avec l'essor du Big Data et des applications à fortes volumétries, ces optimisations étaient cruciales pour Oracle. Elles constituent bien souvent des raisons supplémentaires pour pousser les entreprises à migrer vers une nouvelle version de Java.

Outils tiers

L'adoption d'une nouvelle version majeure d'un langage comme Java 8 passe également par les outils tiers autour de la plateforme. Ici, par outils tiers on entend tout ce qui est lié à Java. Ceci allant donc des frameworks, des bibliothèques tierces aux outils comme les IDE en passant par les serveurs et les langages basés sur la JVM. Au rayon framework, on pourra citer l'exemple de Spring dont la version 4 aura été entièrement conçue en se basant sur Java 8. La nouvelle version du framework a ainsi été disponible avec un support de ce dernier dès la sortie en Mars 2014. De son côté, Apache a réalisé un gros travail pour se mettre à niveau au cours de l'année 2014. Ceci se traduisant par exemple par le support de Java 8 et de ses nouveautés par Tomcat. Le serveur Web léger Jetty était également compatible quasiment dès la sortie. Cette rapidité à évoluer aura d'ailleurs permis à ces serveurs de gagner en popularité par rapport à des mastodontes comme JBoss ou WebLogic dont la date de support n'est toujours pas déterminée à l'heure actuelle. En ce qui concerne les IDE, NetBeans, en tant qu'éditeur de référence pour Java poussé par Oracle, supportait les nouveautés de Java 8 dès la sortie de la nouvelle mouture. En effet, Oracle avait profité de la sortie de Java 8 pour sortir la version 8 de NetBeans. Du côté d'Eclipse et d'IntelliJ Idea, le support aura également été rapide et aura permis à la communauté des développeurs de rapidement pouvoir prendre en main les nouveautés de Java 8. Enfin, au niveau des langages basés sur la JVM, un effort important aura également été réalisé, symbolisé notamment par le langage Scala dont les architectes ont fait le choix de ne plus supporter que Java 8 à partir de la version 2.12 du langage.

Mises à jour

Un frein certain à la montée de version vers Java 8 en 2014 aura été la détection rapide de bugs faite dès la sortie de la nouvelle mouture. Les impacts de ces bugs étaient divers mais ils auront au final fait grand bruit et pu nuire au processus d'adoption dans certaines entreprises. Néanmoins, il faut bien se rendre compte que la détection si rapide des bugs dans Java 8 tient plus au fait qu'il s'agit d'un langage ouvert porté par une large communauté d'utilisateurs que d'un problème de qualité. De fait, les bugs sont détectés et remontés très rapidement. Il s'agit donc plutôt d'une bonne chose. D'ailleurs, il faut se rappeler que les versions majeures antérieures Java 6 et Java 7 avaient également eu leur lot de bugs sans que cela au final n'ait de conséquences majeures sur la pérennité de la plateforme. Pour répondre aux bugs détectés, Oracle aura fourni un gros effort pour effectuer des mises à jour rapides en 2014 avec près d'une demi-douzaine de publications. Dernier élément à prendre en compte pour les entreprises hésitant encore à migrer vers Java 8, Java 6 n'est plus maintenu et Java 7 sera bientôt dans le même cas avec une EOL (End Of Life) prévue pour avril 2015. Il est donc temps de passer à Java 8 !

Limitations

Nouveautés majeures, adoption rapide, correction de bugs à un rythme soutenu, le bilan concernant Java 8 semble idyllique. Il ne l'est bien entendu pas vraiment puisque certaines limitations ont également été remontées depuis 1 an. Au niveau du langage, certains développeurs ont ainsi critiqué les problèmes liés à la surcharge d'opérateurs lors de l'emploi de génériques qui sont amplifiés avec les Lambdas expressions dans certains cas d'emploi. Un autre problème souvent remonté concerne les méthodes virtuelles d'extension qui ne supportent pas tous les mots clés du langage et notamment les mots clés "final" ou "synchronized". Des voix se sont également élevées contre l'ambigüité créée par les noms choisis pour les interfaces fonctionnelles proposées en standard par le SDK. Au niveau exécution, certains benchmarks sont venus mettre en avant des problèmes de performance lors de l'emploi des Streams parallèles notamment. Ces benchmarks ont néanmoins fait ressortir une évidence, il n'y pas de solution miracle en informatique. Chaque solution a ses avantages et ses inconvénients, et il convient donc d'adapter son code au contexte d'utilisation. Globalement, si les limitations rencontrées sont bien réelles, il faut les relativiser puisque leur portée reste restreinte. Il faut plutôt voir dans ces limitations comme des pistes d'améliorations éventuelles pour le futur de Java. Et le futur de Java, c'est justement ce donc nous allons parler dans la suite de cet article.

Java 9

Alors que Java 8 n'a qu'un an, la future mouture de Java, la version 9, est déjà attendue de pied ferme par la communauté pour le courant de l'année 2016. Cette dernière s'annonçant également comme une version majeure pour la plateforme avec la modularisation du JDK. Connu sous le nom de projet Jigsaw, cette modularisation a été maintes fois reportée depuis Java 7, mais il semblerait que ce soit la bonne cette fois !

Projet Jigsaw

Afin d'assurer la sortie de Java 8, les architectes en charge de Java côté Oracle ont décidé de reporter le projet Jigsaw à Java 9. Néanmoins, un travail préparatoire a été réalisé avec l'introduction des profils Compact au sein de Java 8. Ces derniers se basent sur des packages complets et constituent des ensembles fermés, ce qui implique que les références à des classes non présentes au sein d'un profil sont interdites. Ils sont au nombre de 4 : Compact 1 (10 Mb), Compact 2 (17 Mb), Compact 3 (24 Mb) et le JRE complet (140 Mb). Ces derniers vont permettre de choisir à la compilation le profil adéquat le plus restreint à utiliser. Le tout visant bien entendu à réduire l'empreinte mémoire à l'exécution.
Au sein de Java 9, le projet Jigsaw va avant tout amener une modularité du code source du JRE et du JDK qui sera organisé en modules interopérables

ce qui permettra la création d'exécutables facilement scalables et donc exécutables sur des périphériques contraints. Bien que n'étant pas partie prenante du projet Jigsaw, le nouveau système de cache de code segmenté de Java 9 supportera ces mécanismes de modularisation. Ce nouveau système prendra des décisions intelligentes afin de compiler les segments de code les plus fréquemment utilisés en code natif pour le stocker en vue de futures exécutions. Le heap sera également segmenté en 3 unités distinctes : le code non lié à des méthodes sera stocké de manière permanente dans le cache, le code ayant un cycle de vie potentiellement long connu comme "code non-profilé", et, enfin, le code transient connu sous le nom de "code profilé". Le système de build sera également amélioré afin de faciliter la compilation en modules distincts. Au niveau du déploiement, des outils seront fournis au sein de Jigsaw permettant le support des modules, de leurs contraintes ainsi que de leurs dépendances à l'exécution. Les motivations poursuivies par le projet Jigsaw sont nombreuses mais on pourra notamment citer la volonté de supporter des périphériques plus limités en mémoire afin de tirer parti de l'Internet des Objets (Internet of Things ou IoT), l'amélioration de la sécurité et des performances, et, enfin, faciliter la vie des développeurs en charge de créer et de maintenir des bibliothèques.

Gestion des Processus

A l'heure actuelle, les possibilités offertes pour contrôler et gérer les processus du système d'exploitation sous-jacent à l'exécution sont limitées. Ainsi, à l'heure de Java 8, il est par exemple très difficile d'obtenir le PID d'un processus. Des solutions de contournement existent, mais elles ne sont pas portables. Ainsi, avec Java 8, la récupération du PID d'un processus Linux se fera avec le code spécifique suivant :

```
public static void main(String[] args) throws Exception {
 Process proc = Runtime.getRuntime().exec(new String[]{ "/bin/sh", "-c", "echo $PPID" });

 if (proc.waitFor() == 0) {
  InputStream in = proc.getInputStream();
  int available = in.available();
  byte[] outputBytes = new byte[available];
  in.read(outputBytes);
  String pid = new String(outputBytes);
  System.out.println("Your pid is " + pid);
 }
}
```

La nouvelle API ajoutée à Java 9 proposera une forme beaucoup plus concise et surtout portable :

```
System.out.println("Your pid is " + Process.getCurrentPid());
```

Cette API permettra plus généralement aux développeurs Java d'interagir directement avec l'OS sous-jacent, avec la possibilité de récupérer le PID d'un processus donc, mais également son nom, son état, ou encore d'énumérer les JVMs en cours d'exécution.

Money and Currency API

Portée par la JSR 354, l'API Money and Currency possède déjà une implémentation de référence disponible sur GitHub. Sur le même principe que l'API Date and Time introduite dans Java 8, elle apporte une réponse aux problématiques de gestion monétaire. Basée sur un noyau permettant la représentation de montants et devises, elle facilite les opérations monétaires, le formatage, mais également les conversions tout en étant extensible. La représentation des monnaies se fera par l'intermédiaire des classes Money et FastMoney :

```
Money amt1 = Money.of(10.1234556123456789, "USD");
FastMoney amt2 = FastMoney.of(123456789, "USD");
Money total = amt1.add(amt2);
```

La mise en forme d'une monnaie se faisant quant à elle en regard d'un format monétaire lié à un pays :

```
MonetaryAmountFormat germanFormat = MonetaryFormats.getAmountFormat(Locale.
GERMANY);
System.out.println(germanFormat.format(monetaryAmount));
```

Là encore, le JDK tire parti d'une création de Stephen Colebourne puisque l'API Money and Currency se base sur l'API Joda-Money devenue une référence de facto.

Suite du Projet Coin

Source d'évolutions syntaxiques au sein du langage avec Java 7, le projet Coin avait permis le support des String au sein des blocs switch, la gestion automatique des ressources, l'amélioration de l'inférence de type avec l'introduction de l'opérateur diamant, ou encore, le support des exceptions multiples. Cette nouvelle mouture du Projet Coin ne se veut pas une version 2.0, mais plutôt un complément à ce qui avait été introduit dans Java 7. Ainsi, l'annotation @SafeVargs pourra être employée sur des méthodes d'instances privées. Les variables de type final pourront être utilisées au sein des blocs try-with-resources utilisés pour la gestion automatique des ressources. On parle également d'un support pour les méthodes privées au sein des interfaces. Rien d'extraordinaire à attendre mais des petits changements qu'il sera bon de connaître.

Améliorations de la JVM

Comme à chaque nouvelle version du JDK, les ingénieurs de la plateforme en profitent pour améliorer sensiblement la JVM. Java 9 ne fera pas exception à la règle avec l'introduction d'un système de log unifié pour tous les composants de la JVM. Il permettra de faciliter le diagnostic des causes des problèmes de performances ou de crashs de la JVM. Un meilleur contrôle du compilateur est également attendu pour descendre jusqu'au niveau des méthodes. Cette mise à jour ouvrant la voie à des options d'optimisations du compilateur JIT sur des méthodes. Les problèmes de performances liés à des conflits de locks sont légion dans les applications Java multithreads. Java 9 se propose d'améliorer la performance des monitors Java afin d'augmenter les performances de la JVM pour relever le seuil à partir duquel ces conflits de locks viennent ralentir l'exécution des applications Java. Ces nouveautés de l'ombre, bien que peu attendues des développeurs Java, contribuent à maintenir la plateforme Java à niveau, tout en lui permettant de répondre aux nouveaux enjeux auxquelles les applications doivent faire face.

Conclusion

A l'image de Java 5 qui avait introduit les génériques au sein du langage, Java 8 aura été une version majeure de Java dont les impacts vont modifier en profondeur la manière de travailler des développeurs Java. Les Lambdas expressions et les Streams ouvrant ainsi la voie à la programmation fonctionnelle dans le monde Java. Autre gros morceau attendu de Java 8, le projet Jigsaw avait de nouveau été reporté. Ce projet visant à modulariser le JDK sera désormais la figure de proue de la future mouture du langage d'Oracle. Attendu pour le courant de l'année 2016, Java 9 ne pourra se permettre de décevoir le monde Java une fois de plus en ce qui concerne le projet Jigsaw. L'avenir de Java dépendant fortement de sa capacité à se modulariser pour répondre aux nouveaux enjeux du monde mobile ou de l'Internet Of Things.

Pourquoi des entreprises choisissent Java ?

Depuis quelques années, Java est en perte de vitesse en termes de popularité. La faute à qui ? Deux facteurs sont mis en avant : la mode du développement des applis mobiles, dont la base langagière tourne autour de la famille C, et l'explosion des services Cloud positionnés PHP, ASP.Net et Javascript. Si l'on s'adresse à la communauté des programmeurs, il est difficile d'avoir des avis objectifs, car le choix de langage est souvent passionnel, voire fusionnel pour certains codeurs. PROGRAMMEZ a interrogé, Brice Cornet, CEO de Simple CRM (http://crm-pour-pme.fr), Simple CRM a toujours historiquement développé en PHP mais ils ont pourtant annoncé récemment l'arrivée d'une application en Java. Pourquoi ce choix ?

PROGRAMMEZ : avais-tu des a priori à lancer les équipes dans du développement Java ?

Je pense que cette question va me mettre toute la communauté Java à dos. J'avoue avoir considéré, depuis quelques années, Java comme le nouveau COBOL, avec toute la lourdeur que cela sous-entend, y compris le « Compile Only Because Of Luck » ! Même les amoureux de Java se doivent de reconnaître que les 4000 classes qui composent ce langage étaient réellement lourdes.

Heureusement, Oracle a besoin de se positionner sur le Cloud. Lorsque qu'en février 2014, Oracle a annoncé son intention forte de dépoussiérer les classes et le JDK, tout en supprimant les freins liés aux JMX, j'avoue avoir eu envie de réviser mon jugement. Mais il a encore fallu de longs mois avant que cette révision soit effective.

Pourquoi ?

Il y a deux raisons. La première c'est qu'avec Oracle, il peut se passer 5 ans entre l'annonce et la réalité dans l'IDE.

La deuxième est que les failles de sécurité Java ont fait la une de la presse, et pas uniquement des magazines spécialisés. Quand on est un service Cloud et que l'on avance dans les arguments de vente 4 backup par jour des données et une connexion SSL 4096 bits, il est impossible de justifier l'utilisation d'un langage que certains systèmes d'exploitation comme Mac, vont jusqu'à bannir de leur environnement.

Il faut dire aussi qu'Oracle se tire sans cesse des balles dans le pied. Depuis janvier 2015, quand tu installes la JVM, Oracle installe dans ton navigateur Web ASK comme moteur de recherche par défaut, avec en plus une barre d'outils ridicule ! Cela donne juste une image cheap de l'univers Java ; on a vraiment l'impression d'être face à un éditeur de shareware.

Et pourtant au final, tu as quand même choisi le Java pour une partie du développement de Simple CRM. Comment fais-tu pour redonner confiance à tes clients après la campagne de communication très impopulaire qu'a essuyée ce langage ?

C'est assez simple : il faut leur dire la vérité. Oui ce langage a des failles mais c'est aussi un fait évident, il n'existe aucun langage infaillible, sans cela tout le monde l'utiliserait et nous vendrions tous des logiciels 100% bug free.

Ensuite, je mets en lumière qu'ils utilisent tous les jours Java sans le savoir et que de nombreux systèmes du monde de l'aérospatiale et de la défense militaire assurent des missions extrêmement complexes grâce au Safety-Critical Java. Enfin, je leur explique aussi que la partie codée en Java n'est pas en connexion directe avec leur Simple CRM, mais que Java sert juste au formatage et au transfert des données de leur mobile vers leur CRM, avec, entre les deux, une zone tampon qui supprime tout risque de faille.

Pourquoi avoir finalement choisi Java et pourquoi faire exactement ?

Simple CRM est un logiciel 100% Cloud. Il est donc impossible de travailler sans connexion Internet. Tu peux bien entendu consulter ton agenda via une synchro mobile / tablette ou calendrier compatible iCal, tu peux également consulter tes contacts via une synchro également disponible sur mobile et tablette mais pour ce qui est la création de données, nous n'avions aucune solution.

Il nous fallait donc créer un client léger compatible à la fois pour Windows, Mac, Linux, IOS, Android, BlackBerry et Windows Mobile. On avait deux possibilités : soit on réduisait nos ambitions et on limitait le client à deux systèmes d'exploitation, soit on trouvait une solution magique. J'ai fait pas mal de recherches et finalement, j'ai opté pour Java, sous l'IDE Eclipse, couplé à Codename One (http://www.codenameone.com). Tu codes une version de ton app et tu peux compiler pour

Windows, Mac, Linux, IOS, Android, BlackBerry et Windows Mobile. Cela divise par 7 le temps de développement et pas 7 le temps alloué à la maintenance ; c'est totalement prodigieux !

Quelle version de Java est utilisée ?

On a opté pour le JDK 8. La raison est la nature même de l'application. Cette app formate des données afin de créer du contenu dans Simple CRM. On traite donc pas mal de chaînes de caractères afin de les formater correctement. Or JDK 8 a passé à la trappe l'ancien système de concaténation des chaînes de caractères et propose maintenant StringJoiner.add() qui a l'immense avantage de permettre la spécification de délimiteur entre les éléments concaténés, ce qui nous facilite bien la vie. Enfin, le JDK 8 c'est aussi et surtout l'abandon de l'ancestral GregorianCalendar au profit de la famille java.time qui te permet de gérer en natif les formats, les zones, les fonctions de chronos, et bien d'autres éléments qui nous ont permis, dans l'immédiat, de simplifier le code, et surtout de prévoir le futur avec l'insertion de nouvelles fonctionnalités de time tracking en mode déconnecté.

Est-ce que cette expérience t'a fait réviser ton avis sur le langage ?

J'ai toujours estimé que les soi-disant débats autour des langages ou des OS étaient avant tout des guerres d'égo. C'est aussi utile que de comparer la masse d'une personne qui démolit des murs, avec le marteau d'un forgeron, le marteau d'un bijoutier et le maillet d'un menuisier. Chaque langage répond à des cas précis et sur le développement mobile, Java a réellement un avenir prometteur si Oracle continue la modernisation du langage et met en avant d'excellents produits partenaires comme Codename. Donc est-ce que mon avis est révisé ? Oui ! Car il y a une réelle volonté d'améliorer les choses et une entreprise qui a les moyens de le faire. Maintenant il faut rester clair : nous sommes dans un monde qui fonctionne à la vitesse de l'e-mail. Si Oracle veut que Java reprenne sa place du numéro 1 mondial, ils doivent accélérer leur politique de modernisation. ⊞

Eclipse : bien plus qu'un IDE Java

Eclipse est souvent associée à son outillage Java. Mais ce n'est qu'une petite partie de ce que peut faire la plateforme. Retour en arrière sur cette solution si extensible.

Freddy Allilaire
Obeo

IBM offre Eclipse à la communauté

Eclipse était à la base une technologie propriétaire d'OTI, filiale d'IBM. Le but était de réduire le nombre d'environnements de développement proposé aux clients d'IBM et d'améliorer la réutilisation de composants. Après plusieurs tentatives infructueuses sur le problème de l'extensibilité, une petite équipe d'experts IBM se lança dans la conception d'un nouvel outillage. Eclipse était né ! C'était une plate-forme d'intégration d'outils de développement, extensible grâce au mécanisme de plug-ins.

Durant la conférence EclipseCon 2005, Lee Nackman d'IBM Rational Software confirma que le nom Eclipse faisait référence à la volonté d'IBM d'éclipser Microsoft, principal concurrent à l'époque. Pas de clin d'œil à Sun comme le veut pourtant la croyance populaire.

La mise en Open Source allait être la suite logique pour permettre une large adoption. Elle a été annoncée en novembre 2001. Très vite, des dizaines d'applications commerciales ont été créées à partir de la plateforme. À la fin 2003, le consortium initial était composé de plus de 80 membres. Ainsi début 2004, le comité Eclipse annonça une réorganisation avec la création d'une fondation.

Le rôle de la fondation Eclipse

La fondation Eclipse, organisme à but non lucratif, aide la communauté à se développer. Elle est financée par les cotisations de ses membres et dirigée par un conseil d'administration. Sa mission n'est pas de créer et maintenir les projets Eclipse. C'est le rôle des committers qui sont généralement employés par des organisations ou qui sont des développeurs indépendants qui donnent bénévolement de leur temps.

La fondation propose quatre services principaux : une infrastructure informatique, la gestion de la propriété intellectuelle, des processus de développement, et, enfin, la gestion de l'écosystème. La licence EPL a permis de créer cette communauté aussi ouverte, car elle autorise la libre utilisation de technologies dans des produits commerciaux ou des applications internes.

Un outillage Java qui va devenir référence

Eclipse va très vite se faire sa renommée avec son IDE Java. Avec la montée en puissance du langage Java, les développeurs étaient à la recherche d'un outillage de haut niveau. Le Java Development Toolkit a su s'imposer grâce à sa richesse et sa qualité : système de complétion, debugger, formatage de code, refactoring, etc. En ajoutant les facteurs Open Source, gratuité et extensibilité, tout était réuni pour ce succès. De nombreux produits venaient compléter la plate-forme : serveur J2EE, système de gestion de version, etc.

Cependant, au fil des années, le projet JDT s'est essoufflé, notamment avec le désengagement partiel d'IBM. Le passage à Eclipse 4 fut également chaotique et la plateforme devenait moins performante, résultant l'apparition d'un phénomène d'Eclipse bashing. Cette gronde provoqua beaucoup de discussions au sein de la communauté. De nouvelles directions ont été données pour répondre aux attentes et renouer le dialogue avec les utilisateurs. L'édition Luna en 2014 a permis de rattraper le retard. Même si le choix d'un IDE a sa part de subjectivité, il faudra encore du travail pour convaincre la totalité du monde Java. Cependant, chaque nouvelle version génère toujours des millions de téléchargements aux quatre coins du globe.

Un large écosystème dans l'ombre du JDT

Eclipse est communément connue comme un outil de développement Java, mais la plateforme est aussi utilisée par des applications telles que Lotus Notes ou le logiciel de commande et contrôle du Rover de la NASA. Avec 270 projets et 2000 modules dans son marketplace, Eclipse possède une collection impressionnante d'outils. Ces projets Open Source couvrent un large éventail de technologies nouvelles et novatrices, dont beaucoup ne sont pas liées au développement. En fait, Eclipse est devenue un bien commun qui est utilisé par des milliers d'entreprises pour construire des produits logiciels.

Pour finir, une des plus grandes réussites d'Eclipse est le release train, qui est la sortie simultanée annuelle de ses principaux projets. Pour la dernière édition, ce sont 76 projets qui ont participé et livré le même jour une nouvelle version, avec 340 committers impliqués et 61 millions de lignes de code contribuées.

PROGRAMMEZ! A BESOIN DE VOUS !

Vous êtes développeur(se), expert(e), passionné(e), partagez votre passion du code et de la technologie dans Programmez!

Envoyez-nous vos idées et articles :
ftonic@programmez.com / redaction@programmez.com

source : 07-14-14 © dirante

Java peut-il mourir ?

Il y a un an, la tant attendue version 8 de Java sortait. Considérée par certains comme la plus importante version depuis Java 1, elle ne suffit pas à mettre tout le monde d'accord sur la question : "Java est-il en train de mourir ?"

Daniel PETISME
@danielpetisme

It's Not the Big That Eat the Small...It's the Fast That Eat the Slow: How to Use Speed as a Competitive Tool in Business ?

Cette reprise de la formule de Jean de Lafontaine par Jason Jennings est illustrée par WhatsApp. L'application de messagerie instantanée est venue détrôner tous les acteurs établis en seulement quelques années. La clé de leur réussite : une capacité d'adaptation typique des startups. L'objectif est d'offrir à l'utilisateur ce qu'il veut, au moment où il le veut. Java serait-il, à l'instar d'une grande entreprise, devenu victime de son inertie ? Le fameux "Time to Market" serait-il trop long ?
La plus grosse nouveauté de Java 8 est sans aucun doute les lambdas. Avec ce sucre syntaxique, les développeurs peuvent ajouter un peu de fonctionnel dans leur code. "Enfin !" diront certains. En effet, cela faisait longtemps que les développeurs Java jalousaient leurs amis fan de Ruby, JavaScript ou encore C# (seulement pour les lambdas, il ne faut pas pousser). Combien d'entre vous ont cédé aux sirènes des langages alternatifs de la JVM (Groovy, Scala, etc.) pour combler leurs besoins de Map/Reduce et autres Monades ? Les plus puristes ont adopté (sans trop de difficulté) l'excellente bibliothèque Guava de Google pour avoir un avant-goût de fonctionnel.
Depuis 1998, le Java Community Process gère les améliorations de la plateforme. Cette organisation permet de s'assurer que celles-ci répondent aux attentes du marché et des développeurs qui le composent. Chaque demande (Java Specification Request) doit respecter un processus formel et passer des jalons de validation obligatoires. Cette mécanique est certes un gage de confiance de la plateforme mais les délais associés peuvent faire passer le langage comme "en retard" sur certains concepts, comme ce fût le cas avec les lambdas. Et si, finalement, le sentiment de "mort de Java" venait du simple fait que les développeurs veulent des langages "hypes" (et notamment dans le monde du front-end) ?
Mais si Java n'est pas au top de la modernité, pourquoi finit-il toujours dans les short-lists des choix technologiques ?

JSRs: Java Specification Requests
JSR 335: Lambda Expressions for the Java™ Programming Language

Stage	Access	Start	Finish
Final Release	Download page	04 Mar, 2014	
Final Approval Ballot	View results	18 Feb, 2014	03 Mar, 2014
Proposed Final Draft	Download page	14 Jan, 2014	
Public Review Ballot	View results	10 Dec, 2013	23 Dec, 2013
Public Review	Download page	04 Nov, 2013	04 Dec, 2013
Early Draft Review 3	Download page	31 Jan, 2013	02 Mar, 2013
Early Draft Review 2	Download page	19 Jun, 2012	19 Jul, 2012
Early Draft Review	Download page	15 Nov, 2011	15 Dec, 2011
Expert Group Formation		07 Dec, 2010	01 Aug, 2011
JSR Review Ballot	View results	16 Nov, 2010	06 Dec, 2010

Le résumé des 4 années de travail qui a permis d'ajouter les expressions lambda dans le langage Java (https://jcp.org/en/jsr/detail?id=335)

J'en ai plein le CRUD ! Rendez-moi mon Java!

Les langages "hypes" sont très bons dans les "Todo-like apps en moins de 10 minutes". Souvent parce qu'ils viennent avec une architecture et des conventions optimisées pour les démarrages projet, notamment dans les mises en place de CRUD. Le problème récurrent de ces langages est que leur facilité de prise en main (qui séduit les développeurs) disparaît dès que l'on sort des tutoriels. Le prix du ticket d'entrée de Java est certainement un peu plus élevé, de plus le langage ne propose pas forcément les dernières fonctionnalités à la mode, mais préfère s'inscrire dans le temps. Comment ? En inspirant la confiance.
Java est utilisé depuis des années comme pilier de l'enseignement informatique. Je me rappelle avoir fait du Lisp pour apprendre le fonctionnel, du Fortran pour faire du calcul et du C pour faire du système. Mais quand il faut un langage couvrant toutes les problématiques de l'informatique de gestion, le choix s'oriente vers Java. C'est bien là qu'est la force de Java, tout le monde sait écrire du Java. Avec 9 millions de développeurs (plus les prochaines générations à venir), les DSI sont sereins face à la maintenance de leurs parcs applicatifs pour les 20 prochaines années. Pour autant, le langage ne suffit pas à expliquer le succès de Java. Java c'est avant tout une communauté qui, à l'image de Linux dans le monde système, a pris le pari de l'Open Source pour construire un écosystème riche et fiable sur lequel les entreprises n'hésitent plus à s'appuyer (Apache Tomcat étant le plus bel exemple). C'est également la communauté Java qui a été précurseur dans les pratiques de développement logiciel. Java a été le porte-drapeau de l'automatisation avec Apache Ant et Apache Maven, des tests unitaires avec JUnit et de l'intégration continue avec Hudson/Jenkins. L'apport de Java dans le monde informatique ne s'arrête pas là.

"Most people talk about Java the language, and this may sound odd coming from me, but I could hardly care less. At the core of the Java ecosystem is the JVM."

James Gosling, Créateur du langage de programmation Java (2011, TheServerSide)

La machine virtuelle Java est un bac à sable parfait pour créer un langage (et surtout depuis Java 1.7 et l'introduction de invokedynamic) comme l'illustre très bien Groovy, JRuby ou récemment Golo. L'avantage des langages alternatifs est de générer du bytecode s'exécutant sur la Java Virtual Machine. Ces langages sont donc tous inter-compatibles et s'intègrent parfaitement avec du code Java. Le développeur polyglotte peut alors choisir le langage le plus adapté à son problème et le faire communiquer avec le reste du code Java de l'application. En profitant de l'expérience des langages alternatifs, Java est sure de proposer à ses développeurs le meilleur de l'état de l' art.

Conclusion

Finalement, est-ce que Java est en train de mourir ? Non, je ne pense pas. Son organisation à l'image d'une entreprise impose une rigueur afin d'inscrire les évolutions du langage dans le temps. A l'heure où "l'innovation" est un buzzword, le "time-to-market" de Java serait peut-être à revoir. Cependant, son écosystème, et notamment les langages alternatifs de la JVM beaucoup plus réactifs, donnent aux développeurs l'opportunité de tester les derniers concepts et les intégrer facilement dans leurs projets. Java est ancré dans nos SI pour encore un bout de temps !

Le Cloud sauvera-t'il Java EE?

Malgré les évolutions récentes, Java EE souffre toujours d'une image de complexité. Cette image est elle toujours justifiée avec les dernières nouveautés de Java et de Java EE ? Comment le Cloud et notamment le PaaS peuvent simplifier le quotidien des équipes Java EE ?

Fabien Amico
Expert Java, DG et CTO de Treeptik, société spécialisée dans les architectures java et cloud qui développent un CloudUnit le PaaS spécialisé en Java EE.
Professeur vacataire en troisièmes année d'ingénieur à EPITECH.

Le principaux acteurs du Cloud

Le culte de la complexité

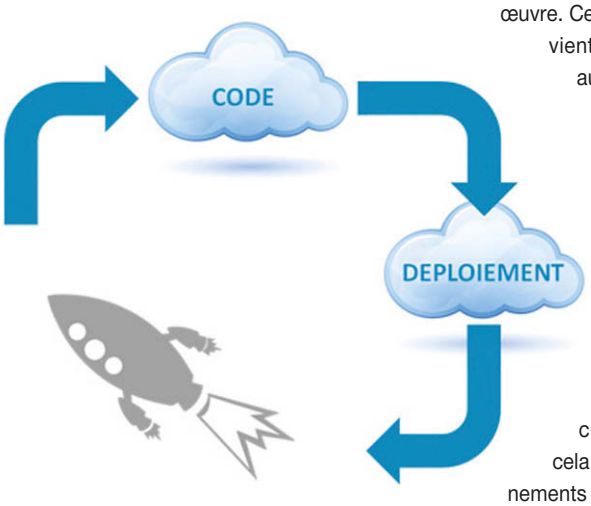
Java EE est la spécification d'Oracle qui permet le développement d'applications d'entreprise. Les applications d'entreprises sont réputées pour avoir un certain nombre de contraintes liées notamment à la sécurité ainsi qu'à une approche multi-niveaux et distribuée. Afin de répondre à ces problématiques, la spécification est constituée d'un grand nombre de standards qui la traite spécifiquement.

Cette grande diversité de standards a longtemps induit une couche de complexité importante souvent considérée par l'écosystème Java comme un mal nécessaire afin de pouvoir développer des applications réparties et sécurisées. Ce culte de la complexité a longtemps fait le bonheur d'ingénieurs capables d'implémenter les solutions à base de JEE 1.4 ou 1.5 qui étaient alors très convoitées.

Une simplification bien réelle

Cette complexité qui générait un coût important sur les projets a permis l'émergence dans l'écosystème Java de nouveaux frameworks venus simplifier le développement d'un projet d'entreprise. Ces projets Open Source ont été au fil de l'eau intégrés dans la spécification JEE. C'est ainsi que le projet Hibernate a permis l'émergence du standard JPA et que le projet Spring (IOC) a inspiré le standard CDI.

Aujourd'hui, il est possible de dire que, d'un point de vue "développement", Java EE s'est bel est bien simplifié, même si la plupart des projets n'utilisent qu'un sous-ensemble (JSP, Servlet, JPA, etc.) et non pas l'ensemble des APIs comme les EJBS ou JMS.



Une mise en œuvre parfois difficile

Comme vu précédemment, le code des applications s'est réellement simplifié avec l'intégration d'un nouveau standard mais pour autant la Java entreprise édition comporte encore un certain nombre de freins qui complexifient cette mise en œuvre. Cette complexité de mise en œuvre ne vient pas seulement du standard mais aussi du contexte applicatif lui-même.

Il n'est pas rare par exemple de voir des applications Java EE composées d'une multitude d'outils. On peut facilement trouver des applications constituées de plusieurs serveurs d'applications qui s'échangent des messages JMS, des bases de données et des outils de cache, sans parler des serveurs Web. Tous ces outils sont complexes et longs à configurer et cela doit se faire à minima sur les environnements de développement, d'intégration et de production. Certains projets peuvent nécessiter huit environnements de validation, recette, qualification, etc. On peut aussi avoir besoin de créer des environnements d'exécution jetables pour des démos chez des clients, ou même pour des environnements de formation.

Le Cloud : enfin une vraie simplification

Les dernières avancées du Cloud Computing permettent de s'attaquer à un aspect jamais encore abordé dans la Java Entreprise Edition, c'est-à-dire la simplification du déploiement de toute l'infrastructure d'exécution des applications. Le Cloud Computing et notamment l'émergence des PaaS (Plateforme as a Service) permettent aujourd'hui de s'affranchir de ces problématiques. Les PaaS permettent la mise à disposition rapide d'environnements d'exécution, configurés et managés.

Ils automatisent le déploiement et le provisionnement d'environnements complexes et permettent parfois de les dupliquer à l'infini dans des Cloud publics ou privés.

Ces outils permettent aux projets Java EE d'arriver enfin à des niveaux de productivité acceptables et permettent ainsi à tous les membres de l'équipe de se concentrer sur leur cœur de métier : c'est-à-dire produire de nouvelles fonctionnalités pour les développeurs et le monitoring des infrastructures pour les admin systèmes.

Visual Studio 2015 Preview et le compilateur .NET "Roslyn"

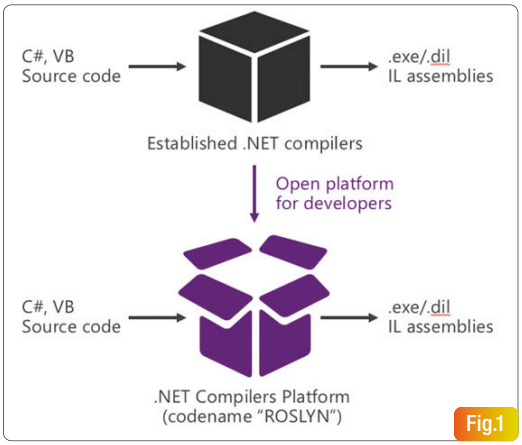


Fig.1

Il y a encore peu de temps, les compilateurs C# et VB étaient de véritables boîtes noires. Afin d'offrir des extensions Visual Studio permettant à la fois de détecter et de proposer des corrections à des problèmes syntaxiques et sémantiques, les éditeurs avaient recours à un lourd développement interne visant à analyser le code source d'une application. C'était du moins le cas jusqu'à l'arrivée en Avril dernier de « Roslyn », la plateforme de compilation .NET. Fig.1

Hugo Carnicelli
Consultant **SoftFluent**

Cette plateforme permet d'accéder de manière unifiée aux structures de données et algorithmes nécessaires à la compilation. Microsoft souhaite au travers de cette plateforme de permettre aux développeurs d'extensions d'accroître leur productivité et faciliter un travail parfois hautement complexe et fastidieux. Porté par l'équipe Microsoft Open Technologies, le projet est en pleine expansion, ainsi qu'en attente de retour d'expériences concernant les améliorations à y apporter.

Architecture

Avec l'ouverture de cette plateforme, dont bénéficient les développeurs C# et VB, il est possible d'accéder à une toute nouvelle API : ses apports sont de pouvoir travailler sur les aspects « analyse » et « diagnostic » mais aussi d'implémenter de nouvelle fonctionnalités afin de compléter le langage C# ou bien VB selon ses propres besoins Fig.2.

Ce graphique montre comment fonctionne le compilateur .NET. La première couche permet d'indiquer les fonctionnalités liées à un élement : le format, la déclaration d'une méthode, le navigateur d'objets, etc. La seconde couche expose la manière dont est organisée l'API du compilateur. La dernière concerne le traitement effectué par le processus de compilation : l'analyse syntaxique, la séparation des symboles et des métadonnées, la liaison et enfin la génération de code IL (« Intermediate Language », langage machine) Fig.3.

Comme on peut le voir sur ce second graphique, il est possible d'accéder au compilateur en fonction de la précision de son besoin. Il est possible de résoudre des besoins fonctionnels concernant le refactoring ou les diagnostics avec proposition de correctifs. Des API liées à l'espace de travail permettent de naviguer au sein d'un projet et des références d'une fonction ou d'une variable. Il est aussi possible de gérer le format du code source. Enfin, les API du compilateur permettent d'analyser les arbres syntaxiques, les différents symboles (instruction, paramètre, déclaration, appel) et les liaisons entre les différents éléments du programme.

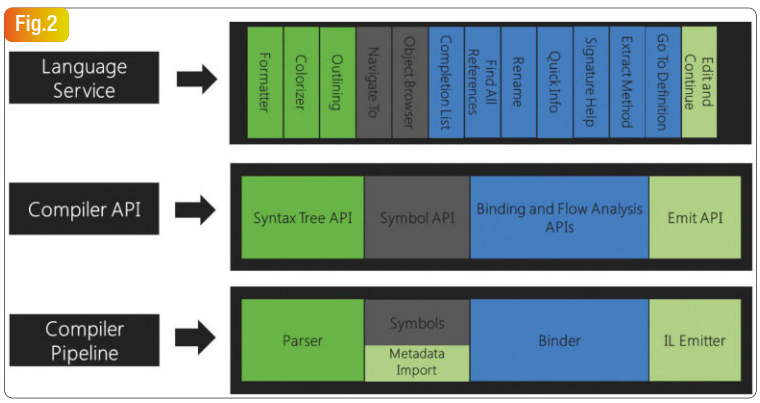


Fig.2

Installation

Pour commencer et si ce n'est pas déjà fait, vous aurez besoin d'installer Visual Studio 2015 Preview ainsi que le SDK associé :
http://www.visualstudio.com/downloads/visual-studio-2015-downloads-vs
Ensuite, il ne vous reste plus qu'à installer Roslyn. Pour ce faire, direction GitHub, où le projet vient d'être migré officiellement depuis CodePlex :
https://github.com/dotnet/roslyn

Qu'est-ce-que cela apporte ?

Une fois les templates de projets Visual Studio installés, ainsi que l'analyseur syntaxique, vous devriez voir apparaître dans la fenêtre de création de projet une nouvelle catégorie « Extensibility » via laquelle vous accédez à 3 types de projets :

- Une application console
- Une extension liée au Code Refactoring
- Une extension liée au Code Fix Fig.4.

Il va donc nous être possible de réaliser de nouveaux produits sous forme de package VSIX ou bien de package Nuget. Prenons comme exemple un nouveau projet de type « Diagnostic with Code Fix » : Fig.5.

On remarque d'emblée une coloration syntaxique améliorée propre à Visual Studio 2015 en version Preview (et donc Ultimate). Les propositions de refactoring et autres analyses liées à ces nouvelles fonctionnalités sont disponibles via le raccourcis « Ctrl + ; ». Afin de faciliter le développement d'extensions Visual Studio en rapport avec le code refactoring, Roslyn met

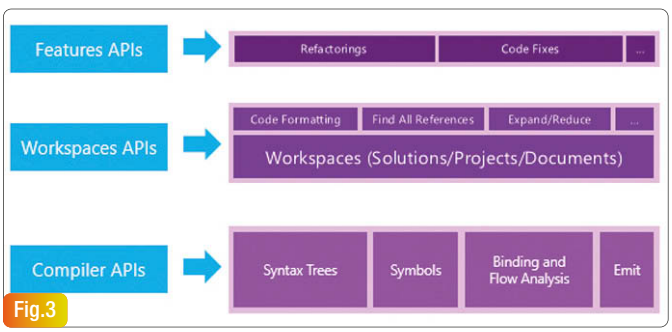


Fig.3

à disposition quelques fonctionnalités :

- Les analyses sémantiques
- Les analyses syntaxiques
- Les transformations syntaxiques

Fonctionnalité simple

Afin d'agrémenter Visual Studio, nous allons prendre comme exemple une implémentation de refactoring basée sur l'utilisation ou non d'une variable locale. Dans un premier cas de figure, la déclaration et l'affectation de valeur d'une variable inutilisée, l'analyse de base détecte sans problème la factorisation adéquate : Fig.6.

Toutefois, il est possible que sa vigilance soit mise à l'épreuve dans un cas légèrement plus complexe : Fig.7.

En effet, la variable étant passée en référence, elle est considérée comme « utile » puisque l'analyse semble se limiter au scope local dans ce cas précis. Nous allons donc implémenter une règle personnalisée afin de détecter l'inutilité d'une variable dans l'ensemble de l'arbre syntaxique.

Eléments syntaxiques

À l'heure actuelle, Roslyn ne bénéficie pas d'une documentation technique particulièrement exhaustive puisqu'il s'agit encore d'un projet en cours de développement. En revanche, le dépôt GitHub possède un nombre important d'informations essentielles (https://github.com/dotnet/roslyn/wiki/Samples-and-Walkthroughs). Parmi celles-ci, la liste des membres syntaxiques. Nous allons nous contenter dans cet article d'utiliser les types de nœuds syntaxiques suivants :

- LocalDeclarationStatementSyntax
- InvocationExpressionSyntax
- ArgumentSyntax
- MethodDeclarationSyntax
- ParameterSyntax
- StatementSyntax
- IdentifierNameSyntax

Nous allons voir comment exploiter l'arbre syntaxique fin d'analyser l'utilisation d'une variable.

Implémentation

Pour commencer, nous allons réaliser un exemple de code minimaliste :

```
var classSourceCode = @"
  using System;
  using System.Collections.Generic;
  using System.Linq;
  using System.Text;
  using System.Threading.Tasks;
  using System.Diagnostics;

  namespace ConsoleApplication
  {
    class ClasseUtile
    {
      public static void Utile()
      {
        int inutile = 0;
        MasqueInutile(ref inutile);
        // traitement
      }
      public static void MasqueInutile(ref int i)
      {
        if (true || false || i > 0)
        {
          // traitement
        }
      }
    }
  }";
SyntaxTree tree = CSharpSyntaxTree.ParseText(classSourceCode);
SyntaxNode root = (CompilationUnitSyntax)tree.GetRoot();

if (CheckVariableDeclarations(root))
{
```

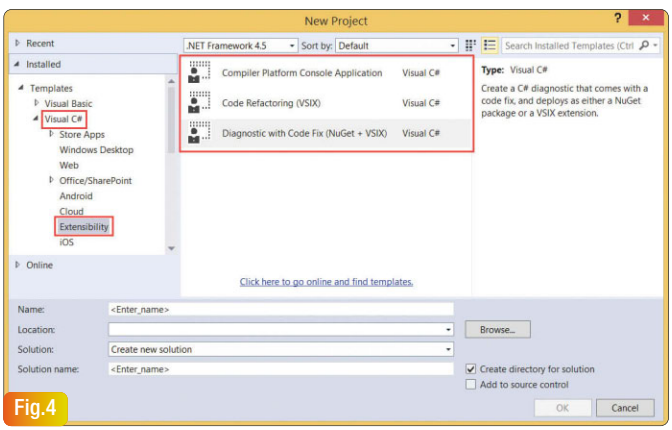


Fig.4

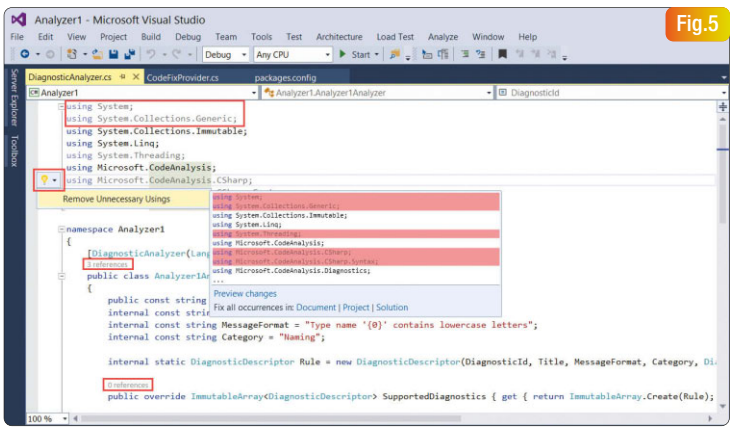


Fig.5

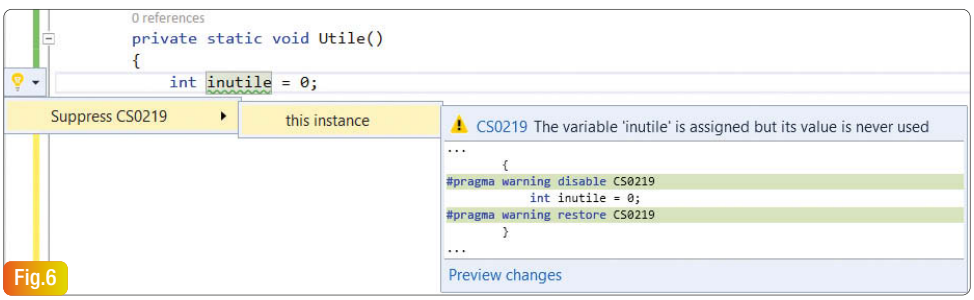


Fig.6

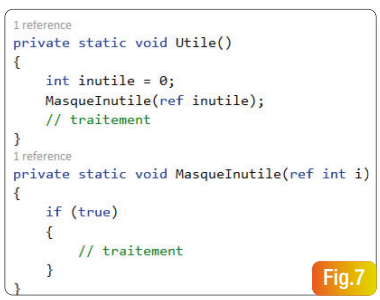


Fig.7

```
    Console.WriteLine("Code optimisable...");
}
else
{
    Console.WriteLine("Code non optimisable !");
}
```

La fonction « CheckVariableDeclarations() » va simplement aller retrouver, dans le code source fourni en entrée du compilateur via la variable « classSourceCode », l'utilité de la déclaration des différentes variables.
Voici son contenu :

```
private bool CheckVariableDeclarations(SyntaxNode root)
{
    bool find = false;

    // pour chaque déclaration de variable locale
    foreach (LocalDeclarationStatementSyntax variable in root.DescendantNodes().OfType<LocalDeclarationStatementSyntax>())
    {
        Console.WriteLine(string.Format("Variable utilisée ? {0}", variable.Declaration.Variables.First().Identifier.ToString()));
        if (CheckVariableDeclaration(variable))
        {
            Console.WriteLine(string.Format("Variable utilisée ! {0}", variable.Declaration.Variables.First().Identifier.ToString()));
            find = true;
        }
    }

    return find;
}
```

Elle ne fait que parcourir la liste des déclarations de variables locales, et pour chacune, appeler la méthode *CheckVariableDeclaration*.
Celle-ci, plus complexe, va vérifier si l'on retrouve en paramètre de l'appel de toutes les méthodes locales la variable en cours d'analyse :

```
private bool CheckVariableDeclaration(LocalDeclarationStatementSyntax variable)
{
    bool find = false;

    // retrouver les méthodes appelant cette variable
    // 1. liste des méthodes appelées
    foreach (InvocationExpressionSyntax method in variable.Parent.DescendantNodes().OfType<InvocationExpressionSyntax>())
    {
        // 2. liste des arguments
        foreach (ArgumentSyntax argument in method.ArgumentList.Arguments)
        {
            // 3. arguments trouvé
            if (argument.Expression.ToString() == variable.Declaration.Variables.First().Identifier.ToString())
            {
                Console.WriteLine(string.Format("Argument utilisé ? {0}", argument.Expression.ToString()));
                if (CheckArgument(argument, method, method.ArgumentList.Arguments.IndexOf(argument)))
                {
                    Console.WriteLine(string.Format("Argument utilisé ! {0}", argument.Expression.ToString()));
                    find = true;
                }
            }
        }
    }

    return find;
}
```

Une fois les méthodes identifiées, elle appelle la fonction *CheckArgument* qui va se charger de vérifier l'utilité de chacun des paramètres de la méthode concernée :

```
private bool CheckArgument(ArgumentSyntax variable, InvocationExpressionSyntax method, int parameterIndex)
{
    bool find = false;

    // vérifier si méthode appelée utilise la variable
    // 1. liste des méthodes déclarées
    foreach (MethodDeclarationSyntax methodDeclaration in variable.SyntaxTree.GetRoot().DescendantNodes().OfType<MethodDeclarationSyntax>())
    {
        // 2. arguments trouvé
        if (method.Expression.ToString() == methodDeclaration.Identifier.ToString())
        {
            Console.WriteLine(string.Format("Parametre utilisé ? {0}", variable.Expression.ToString()));
            if (CheckMethodBody(methodDeclaration.ParameterList.DescendantNodes().OfType<ParameterSyntax>().ElementAt(parameterIndex), methodDeclaration.Body.Statements))
            {
                Console.WriteLine(string.Format("Parametre utilisé ! {0}", variable.Expression.ToString()));
                find = true;
            }
        }
    }

    return find;
}
```

Dernière étape, le corps de la méthode est à son tour analysé afin de déterminer si l'une des instructions utilise le paramètre donné :

```
private bool CheckMethodBody(ParameterSyntax param, SyntaxList<StatementSyntax> statements)
{
    bool find = false;

    // liste des instructions
    foreach (StatementSyntax statement in statements)
    {
        foreach (IdentifierNameSyntax expression in statement.DescendantNodes().OfType<IdentifierNameSyntax>())
        {
            Console.WriteLine(string.Format("Parametre utilisé dans l'expression ? {0}", param.Identifier.ToString()));
            if (expression.Identifier.ToString() == param.Identifier.ToString())
            {
```

```
            Console.WriteLine(string.Format("Parametre dans l'expression ! {0}", param.Identifier.
ToString()));
            find = true;
        }
    }
  }

  return find;
}
```

Vous remarquerez que l'ensemble de ces méthodes propage un retour booléen de manière à déterminer si le code source initial est optimisable.

Conclusion

Dans notre exemple, le code suivant sera déterminé comme correct (notez l'utilisation du paramètre « i » dans le test de la méthode *UselessMask*) :

```
class UsefulClass
{
  public static void Useful()
  {
    int useless = 0;
    UselessMask(ref useless);
    // traitement
  }
  public static void UselessMask(ref int i)
  {
    if (true || false || i > 0)
    {
      // traitement
    }
  }
}
```

Au contraire, si l'on enlève cette partie de l'évaluation, le code sera déclaré comme pouvant être optimisé :

```
class UsefulClass
{
  public static void Useful()
  {
    int useless= 0;
    UselessMask(ref useless);
    // traitement
  }
  public static void UselessMask(ref int i)
  {
    if (true || false)
    {
      // traitement
    }
  }
}
```

Pour aller plus loin

L'implémentation présentée ici n'est bien entendu pas complète et ne couvre pas certains aspects comme :

- La récupération du code source du projet ouvert dans Visual Studio au lieu d'une variable fixe
- La création et l'installation d'un package Nuget ou VSIX
- L'initialisation de la variable « inutile » par une méthode, faisant elle-même quelque chose de concret
- La nécessité de recourir à un mécanisme récursif afin de parcourir convenablement l'ensemble de l'arbre syntaxique
- Proposer un CodeFix
- ... et probablement beaucoup d'autres choses !

Si cette démonstration vous a plu, vous pouvez essayer de ré-implémenter vos propres fonctionnalités et peut être proposer une extension Visual Studio pour concurrencer les plus grands noms du marché :

- ReSharper de JetBrains (https://www.jetbrains.com/resharper/)
- JustCode de Telerik (http://www.telerik.com/products/justcode.aspx)
- CodeRush de DevExpress (https://www.devexpress.com/Products/CodeRush/)
- VSCommands de Squared Infinity (http://vscommands.squaredinfinity.com/)

Il est intéressant de noter que JetBrains n'a pas prévu d'utiliser Roslyn, puisque cette plateforme se limite aux langages C# et VB et que ReSharper va beaucoup plus loin que ceux là (http://blog.jetbrains.com/dotnet/2014/04/10/resharper-and-roslyn-qa/).

LLVM : l'autre refonte du compilateur

Dans l'univers Open Source, on parle beaucoup du LLVM (anciennement : Low Level Virtual Machine, machine virtuelle de bas niveau). LLVM est une architecture de compilation et de toolchain (chaîne de compilation). Elle se veut souple, modulaire et réutilisable. LLVM est désormais le nom officiel. Le projet fut initié par l'université de l'Illinois pour proposer un compilateur supportant le dynamique et le statique. Le projet est sous licence permissive de type BSD.

LLVM repose sur plusieurs modules, en voici quelques-uns :

- LLVM Core : librairie de base de l'architecture. Elle permet une optimisation indépendante de la plate-forme et support les processeurs les plus utilisés.
- Clang : compilateur natif C, C++ et Objective-C
- Dragoneff : comprends le code générateur, l'optimisation et les parsers GCC (le compilateur C le plus répandu en open source).
- LLDB : librairie de debug
- Libc++ / libc++ ABI : implémentation du C++ Standard Library, support complet de C++ 11
- OpenMP : implémentation compatible avec Clang. Dédié aux hautes performances sur plusieurs processeurs

Actuellement, LLVM est en version 3.6 (version stable). Il évolue indépendamment des implémentations. Parmi les gros utilisateurs de cette architecture, nous retrouvons Apple et Google. Il est massivement utilisé par la compilation des apps mobiles, mais aussi pour les systèmes et les services web. Apple a fait le mouvement LLVM dès 2008.

LLVM est-il plus performant et plus efficace que GCC ? Les communautés ne sont parfois pas tendres entre elles. LLVM n'est pas forcément aussi complet que GCC, mais LLVM se révèle souvent plus rapide en compilation (https://vmakarov.fedorapeople.org/spec/2014/2014.html). Mais cela va dépendre du processeur cible, du code, du langage utilisé. Richard Stallman n'est pas un grand fan de LLVM et particulièrement de Clang (https://gcc.gnu.org/ml/gcc/2014-01/msg00247.html).

Gérer sa cave à vin avec Microsoft Azure et des QR Code

Laurent Ellerbach
Audience Marketing and Technical Evangelist Lead
Microsoft Central and Eastern Europe
laurelle@microsoft.com
http://blogs.msdn.com/laurelle

Je suis un amateur de vin et je possède “quelques” bouteilles. J’en bois beaucoup avec mes amis, ma famille. Ce qu’il fait que j’achète de nombreuses bouteilles tout au long de l’année. Comme pour toute gestion de stock, il est compliqué de savoir où j’en suis. Entre les entrées et les sorties, mon fichier Excel, imprimé, n’y suffit plus. Imaginez à quoi les 2 feuilles ressemblent au bout de quelques mois quand elles sont remplies de graffitis. Sans compter qu’il faut aussi noter à la main les nouvelles bouteilles arrivant dans le stock. De plus, je ne suis pas le seul à aller chercher les bouteilles, nous sommes 5 et donc il y a de nombreuses erreurs avec le temps. Du coup, je suis obligé de faire un inventaire complet 2 à 3 fois par an. Au-delà de l’exemple lié à ma cave à vin, c’est le problème classique de gestion de stock.

Il y a quelques années déjà j’avais réfléchi à une solution autonome utilisant des code-barres. Mais je n’ai jamais pris le temps de réaliser cette solution. Bien m’en a pris, car la solution que j’ai mise en place aujourd’hui est bien plus simple. Elle se base sur des QR Code imprimés et collés sur les bouteilles, un scan avec un smartphone, une infrastructure Cloud basée sur un site Web dans Microsoft Azure et une base de données SQL Azure. Le site Web utilise ASP.NET + MVC + EntityFramework. Le tout aurait pu être également réalisé en PHP ou Java ou à peu près n’importe quelle technologie Web. Microsoft Azure est une plateforme ouverte supportant de très nombreuses technologies. À travers 8 étapes simples, nous allons donc voir comment construire cette application de gestion de cave.

1 Créer le site Web Azure depuis Visual Studio 2013

Je suis un grand fan de Visual Studio 2013; je l'utilise beaucoup et le compare souvent avec d’autres et c’est un environnement vraiment complet. Visual Studio est disponible dans de nombreuses versions notamment Visual Studio Community qui est une version gratuite destinée aux hobbyists et personnes participant à du code communautaire (http://www.visualstudio.com/products/visual-studio-community-vs). Pour commencer, si vous n’en avez pas encore, il faut ouvrir une souscription à Microsoft Azure. Pour réaliser l’application et la base de données, l’utilisation de Microsoft Azure est gratuite. L’hébergement gratuit et la base de données de 1Go suffiront largement. Lors de l’enregistrement à Azure, vous devrez utiliser un Microsoft Account (ex-Live ID) et une carte de crédit vous sera demandée. Elle est utilisée à titre de vérification uniquement. Vous ne serez pas débité si vous restez dans l’utilisation gratuite. Pour vous lancer, c’est ici que ça se passe : http://azure.microsoft.com/pricing/free-trial/.

Pour créer un site Web dans Azure à l’aide de Visual Studio 2013, vous pouvez suivre le pas-à-pas ici : http://azure.microsoft.com/documentation/articles/web-sites-dotnet-get-started/. Par contre, il faudra que vous fassiez attention, il est nécessaire de garder l’authentification dans le projet. Nous en aurons besoin pour être sûrs que personne ne vienne modifier votre stock de bouteilles. Sélectionnez “Individual User Accounts” dans l’écran d’authentification. C’est la seule différence avec le tutoriel. À noter que si vous utilisez une version précédente de Visual Studio, ce sont à peu près les mêmes étapes à suivre Fig.1. Le projet est maintenant créé, et vous pouvez dès maintenant ajouter des utilisateurs; il y a aussi quelques pages créées, le site est fonctionnel. En regardant la liste des connexions aux bases de données, vous verrez des tables AspNet. Ces tables sont utilisées pour la gestion des utilisateurs et des rôles Fig.2. À chaque fois qu’un utilisateur s’enregistre, il arrive dans la table AspNetUsers. La page d’enregistrement est simple et ne contient que le mail et le mot de passe, aussi la table ne contiendra que ces informations. À noter que le mot de passe n’est pas stocké en clair, mais seulement son hash code. C’est une bonne pratique que de ne pas stocker les mots de passe des utilisateurs directement en base Fig.3.

Fig.1

2 Créer la table des vins

Je n’ai créé qu’une seule table “ListeVins” qui contient uniquement un Id, une région, une description, un lieu de rangement (je les créerai en texte), une année, une quantité de bouteilles et le temps que je dois garder la bouteille avant de la boire en année. Pour ce faire, il faut aller dans la gestion des serveurs, sélectionner Azure, puis accéder dans les bases de données Azure et sélectionner "les gérer". Il devient possible de faire des opérations comme créer une nouvelle table Fig.4.

Fig.2

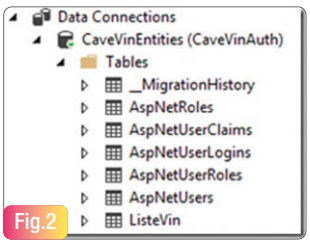


Fig.4

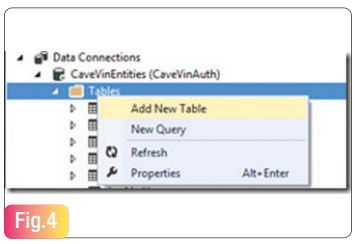


Fig.5

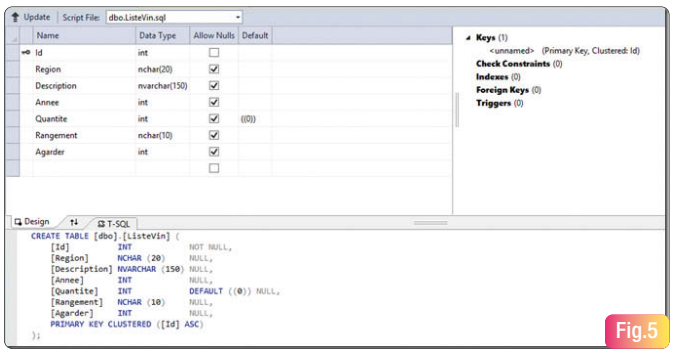


Fig.3

	Id	Email	EmailConfirmed	PasswordHash	SecurityStamp	PhoneNumber	PhoneNumber...	TwoFactorEna...	LockoutEndDa...	LockoutEnabled	AccessFa
▶	45-494c-976c-96a	laurent@ellerba...	False	AEjrhrsjxuiocpY...	b4b5f918-10fa-...	NULL	False	False	NULL	True	0

Il est maintenant possible de créer des colonnes. Soit en écrivant le Transac SQL, soit avec l'outil graphique, au choix. J'aurais pu créer un modèle plus compliqué avec une date de création du lot de bouteille dans le stock, ajouter une table traquant les changements dans la base, quel utilisateur a fait des changements, etc. Mais je vais faire très simple. Vous pourrez toujours compléter cette application avec ces éléments Fig.5. Une fois terminé, cliquer sur mettre à jour (update) pour que les modifications soient faites dans la base SQL Azure.

3 Créer un "Model" basé sur la table ListeVins

MVC veut dire "Model View Controller". Le "Model" est la liaison avec la base de données. Le "View" est la partie graphique, la page HMTL en tant que telle avec du code. Le "controller" se trouve au milieu et permet d'effectuer des opérations plus complexes comme la création de graphiques, gérer les problématiques de sécurité, etc. La première étape va donc consister à ajouter un "Model" à notre projet. Pour cela, il faut donc cliquer avec le bouton droit sur le répertoire Model du projet dans Visual Studio, puis sélectionner ajouter, puis nouvel élément "ADO.NET Entity Data Model" Fig.6. Donnez-lui un nom tel que ListeVin puis cliquez sur ajouter et sélectionnez "EF Designer from database" Fig.7. Si vous n'avez pas encore effectué une connexion à la base de données, c'est le moment de le faire. Sélectionnez "Nouvelle connexion" Fig.8. Vous pouvez obtenir le nom de votre serveur soit dans les propriétés de la base de données SQL Azure dans Visual Studio (replacez les x par le nom), soit dans la console de management de Azure (https://manage.windowsazure.com). Mettez le login et le mot de passe que vous avez définis quand vous avez créé le site Web. Sélectionnez la base de données qui contient votre table ListVins. Vous pouvez ensuite décider de mettre les informations de login soit dans le code, soit dans le fichier de configuration. Je décide de le mettre dans le fichier de configuration Fig.9. Dans l'étape suivante, il est possible de sélectionner les tables, vues, et autres fonctions que l'on veut pouvoir avoir dans notre Model. Le Model utilise l'Entity Framework qui permet d'obtenir une abstraction des données et d'effectuer des opérations sur la base de données de façon très simple sans pour autant mettre de côté les performances Fig.10. Une fois fini, le Model est ajouté. Beaucoup de code C# a été généré. Pas la peine de faire la moindre modification, il a été généré automatiquement. Cela fait gagner un temps fou pour l'accès aux données Fig.11. Il est nécessaire de compiler entièrement le projet afin que le Model soit reconnu à l'étape suivante.

4 Créer le Controller et le View

Cliquez avec le bouton droit sur "Controllers" dans les répertoires de projet, sélectionner "Controller..." puis "MVC 5 Controller with views, using Entity Framework" et cliquez sur ajouter Fig.12. Sélectionnez le modèle créé précédemment dans la liste déroulante et la base. Nommez le "ListeVinsController", sélectionnez au moins "Generate views" et ajoutez. Vous pouvez également sélectionner la mise en forme, ou vous pouvez le faire plus tard dans le code des pages Fig.13. Vous venez donc de créer automatiquement tout ce qui est nécessaire pour insérer de nouvelles bouteilles, obtenir le détail, les modifier. C'est le moment donc de tester tout cela en cliquant sur F5.Vous arrivez sur la page par défaut. Changez l'url en replaçant "/home/index" par "/ListeVins". Il faut noter que le nom du Controller créé par défaut se fait en enlevant "Controller"

Fig.13

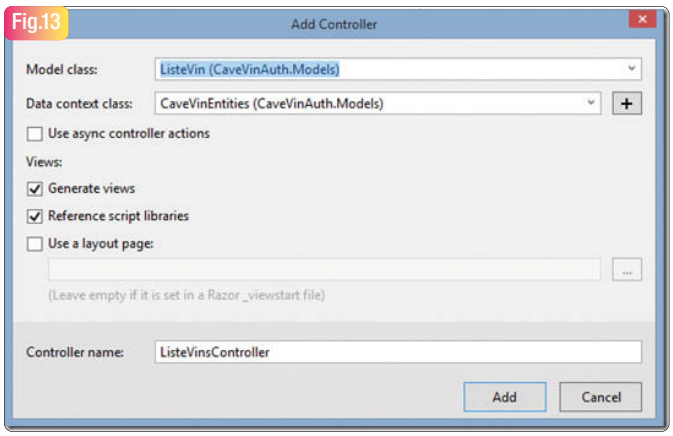


Fig.8

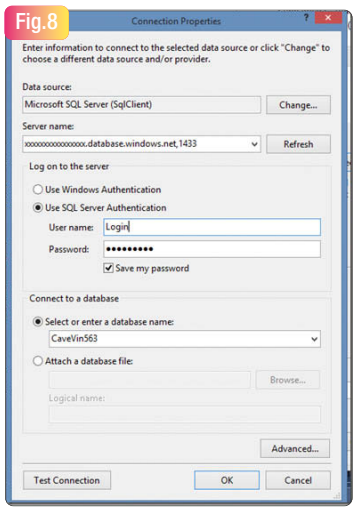


Fig.11

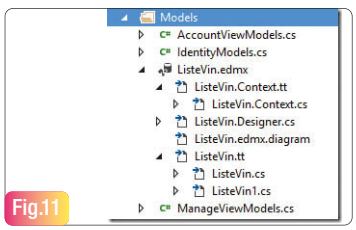


Fig.6

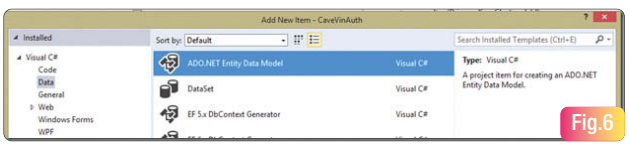


Fig.10

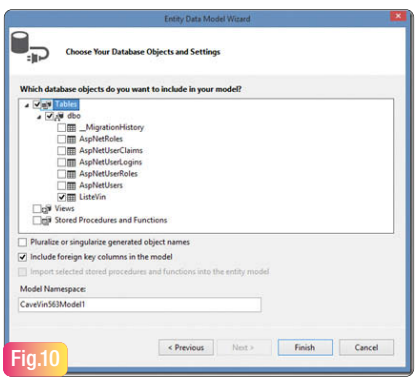


Fig.7

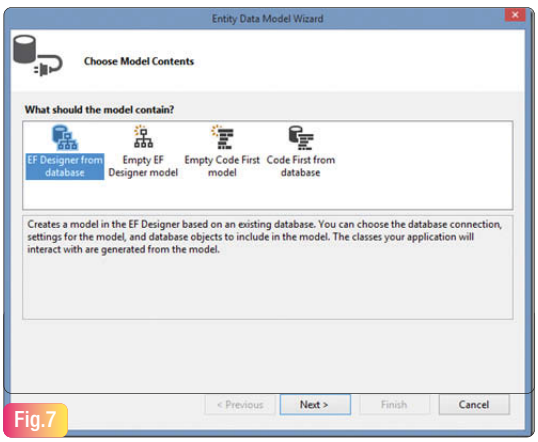


Fig.9

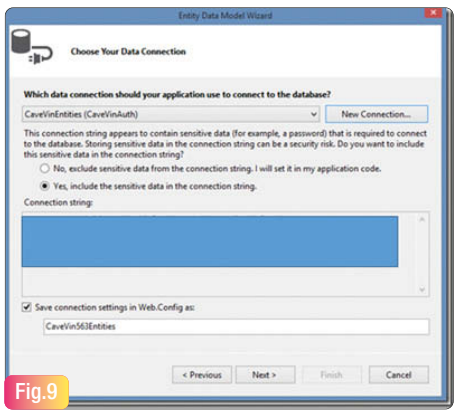
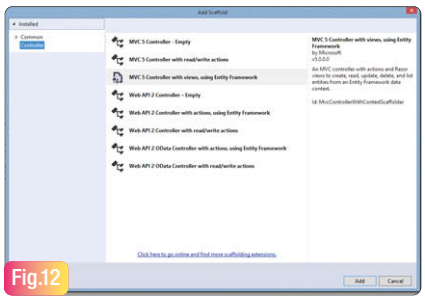


Fig.12



du nom que vous avez rentré précédemment Fig.14. Amusez-vous à rentrer quelques informations sur votre nouveau stock. Et le tout sans avoir encore écrit la moindre ligne de code. C'est ce que j'aime avec ASP.NET et MVC. Sachant que l'ajout de code et la modification du code existant se font vraiment naturellement; nous allons le faire dans une prochaine étape. Pour l'instant, il n'y a aucune sécurité d'implémentée et c'est ce que nous allons implémenter dans l'étape qui suit.

5 Ajouter de la sécurité

Comme expliqué dans la première partie, plusieurs tables sont utilisées pour gérer les utilisateurs et les rôles. AspNetRoles est la table qui gère les rôles. Vous pouvez créer autant de rôles que vous voulez et les nommer comme vous le souhaitez. Vous pouvez ensuite les utiliser pour gérer la sécurité d'accès Fig.15. J'ai créé 2 rôles, un rôle administrateur et un rôle "CanEdit". Je les utiliserai plus loin dans le code. Attention, ils sont sensibles aux minuscules/majuscule. Ensuite, dans la table AspNetUsersRoles, il faut relier les utilisateurs aux rôles qui viennent d'être créés. Dans mon cas, j'associe les 2 rôles à mon utilisateur Fig.16. Je fais cela manuellement, car je n'ai que 5 utilisateurs. Il est bien sûr possible de faire cela avec des pages Web que vous pouvez créer de la même façon que l'on a créé les pages View et Controller, mais en s'appuyant sur les Model sécurité des utilisateurs. Un minimum de code est nécessaire pour cela. Pour appliquer la sécurité dans le code, vous pouvez soit le faire de façon déclarative sur les fonctions,soit directement dans le code, ou les deux. Et cela à la fois sur les utilisateurs et/ou les rôles. [Authorize(Roles = "Admin")] donnera accès uniquement aux utilisateurs référencés comme dans le rôle "Admin". Par défaut, toutes les connexions anonymes sont acceptées.

```
    // POST: ListeVins/Create
    // To protect from overposting attacks, please enable the specific properties you want to bind to, for
    // more details see http://go.microsoft.com/fwlink/?LinkId=317598.
    [HttpPost]
    [ValidateAntiForgeryToken]
    [Authorize(Roles="Admin")]
    public ActionResult Create([Bind(Include = "Id,Region,Description,Annee,Quantite,Rangement,Agarder")] ListeVin listeVin)
    {
      if (ModelState.IsValid)
      {
        db.ListeVin.Add(listeVin);
        db.SaveChanges();
        return RedirectToAction("Index");
      }

      return View(listeVin);
    }
```

Il est possible de définir plusieurs rôles en les séparant, sans espace par des virgules. Et vous pouvez faire de même pour les utilisateurs.

```
[Authorize(Roles = "Admin,CanEdit")]
public ActionResult Edit(int? id)
{
  if (id == null)
  {
    return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
  }
  ListeVin listeVin = db.ListeVin.Find(id);
  if (listeVin == null)
  {
    return HttpNotFound();
  }
  return View(listeVin);
}
```

J'ai appliqué de la sécurité sur les fonctions Create, Delete et Edit, elles sont clés dans le projet. Si vous n'êtes pas un utilisateur enregistré ou si vous n'avez pas les bons droits, vous serez automatiquement redirigés vers la page de login Fig.17. Une fois connectés avec le bon rôle associé au profil, vous pouvez accéder ici à la page Create Fig.18.

6 Supprimer 1 bouteille de l'inventaire

J'ai créé une fonction spécifique dans le Controller pour enlever une bouteille de l'inventaire.

```
// GET: ListeVins/Decrease/5
[Authorize(Roles = "CanEdit")]
public ActionResult Decrease(int? id)
{
  if (id == null)
  {
    return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
  }
  ListeVin listeVin = db.ListeVin.Find(id);
  if (listeVin == null)
  {
    return HttpNotFound();
  }
  listeVin.Quantite -= 1;
  db.SaveChanges();
```

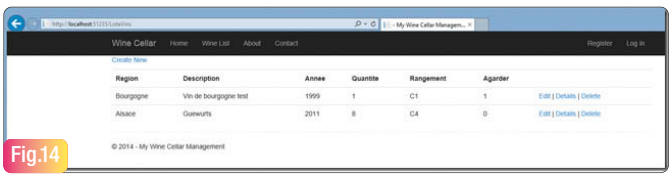


Fig.14

Id	Name
2	Admin
1	CanEdit
NULL	NULL

Fig.15

UserId	RoleId
2662739c-ac45-...	2
2662739c-ac45-...	1
NULL	NULL

Fig.16

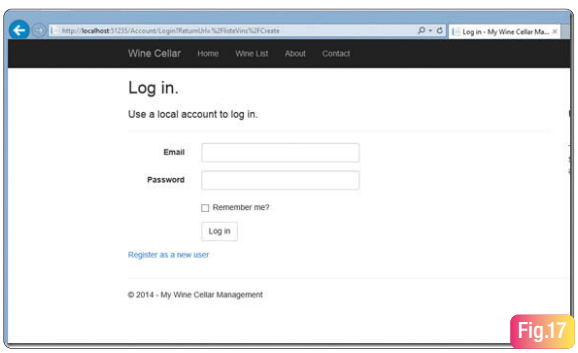


Fig.17

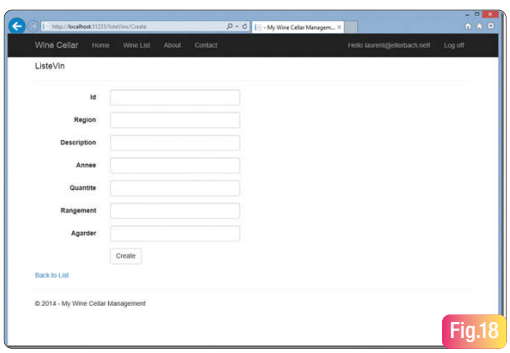


Fig.18

```
        return RedirectToAction("Index");
    }
```

Le code est très similaire au code permettant de supprimer une série de bouteilles. Je diminue la quantité de bouteilles disponible de 1, je ne fais aucun test sur les quantités, je sauve les modifications dans la base de données. Cette fonction n'est disponible que pour les utilisateurs possédant le rôle CanEdit. J'aurais pu ajouter un joli message de confirmation, mais pour faire simple, je redirige simplement sur la page affichant l'ensemble des bouteilles. Pour supprimer une bouteille d'un lot, il faut donc aller sur l'URL http://yoursite/listevins/decrease/5. Ici, 1 bouteille sera supprimée du lot de bouteilles 5 (remplacez yoursite par l'URL de votre site Web). Autant dire que tout cela n'est pas super KAF (Kids Acceptation Factor) ni WAF (Woman Acceptation Factor) donc pas très facile d'utilisation pour les femmes et les enfants que j'envoie très régulièrement me chercher du vin à la cave. Et il faut que ce soit simple pour eux si je veux qu'ils continuent d'aller me chercher des bouteilles et noter ce qu'ils ramènent. D'où l'idée d'utiliser des QR code sur les bouteilles et leurs smartphone respectifs pour décrémenter le nombre de bouteilles.

7 Ajouter les QR codes avec redirection sur la bonne URL

Il faut impérativement que tout le monde à la maison puisse facilement scanner le QR Code. LA bonne nouvelle c'est qu'ils ont tous des Windows Phone (OK, ça marche aussi avec n'importe quel smartphone). Ce qu'il faut c'est que je puisse générer dynamiquement des QR Code de façon à ce que je puisse les imprimer et les coller sur les bouteilles. Oui, c'est une opération qui prend un peu de temps, mais c'est un bon investissement, car il n'est nécessaire qu'une seule fois. Et quand j'ai de nouveaux arrivages de vin, je n'ai également besoin de le faire qu'initialement. Pas besoin a priori de coller de nouvelles étiquettes pour les 20 prochaines années. J'ai trouvé un projet open source de génération de QR Code sur CodePlex: http://qrcodenet.codeplex.com/
Une fois téléchargée et référencée dans mon projet, l'utilisation est très simple :

```
    public string GetQrCode(int? id)
    {
        return "<img src=\"/ListeVins/QrCode?id=" + id + "\" />";
    }

    public FileResult QRCode(int? id)
    {
        QrEncoder encoder = new QrEncoder(ErrorCorrectionLevel.M);
        QrCode qrCode;
        MemoryStream ms = new MemoryStream();
        encoder.TryEncode("https://yoursite/listevins/decrease/"+id, out qrCode);


        var render = new GraphicsRenderer(new FixedModuleSize(10, QuietZoneModules.
Two));
        render.WriteToStream(qrCode.Matrix, ImageFormat.Png, ms);
        return File(ms.GetBuffer(), @"image/png"); ;

    }
```

La fonction GetQrCode retourne simplement une chaîne de caractères qui contient un tag HTML d'image. L'image est en fait la seconde fonction qui va réellement la créer. L'image est créée en mémoire et renvoyée dans un stream. Le stream est renvoyé tel qu'un fichier. Pour que cela puisse fonctionner, remplacez "yoursite" par votre propre URL. Maintenant, je vais modifier le fichier "Details.cshtml" pour ajouter le QR Code dans la liste telle que la région, la quantité, l'année ou le nombre d'années pendant lesquelles garder le vin. Le tag image contient l'appel à la fonction qui permet de créer l'image du QR Code.

```
<dd>
  @Html.Action("GetQrCode", "ListeVins", new { id = Model.Id });
</dd>
```

Fig.19.

Fig.19

```
            <dt>
                @Html.DisplayNameFor(model => model.Agarder)
            </dt>

            <dd>
                @Html.DisplayFor(model => model.Agarder)
            </dd>
            <dd>
                @Html.Action("GetQrCode", "ListeVins", new { id = Model.Id });
            </dd>
        </dl>
    </div>
    <p>
        @Html.ActionLink("Edit", "Edit", new { id = Model.Id }) |
        @Html.ActionLink("Back to List", "Index")
    </p>
</body>
</html>
```

Et le résultat permet de créer le QR Code dans la page de détail Fig.20. La seule chose qu'il reste à faire, c'est de copier/coller le bon nombre d'images de QR Code sur une planche d'étiquettes et de les coller sur les bouteilles ! Comme vous pouvez le constater sur la capture d'écran, j'ai personnalisé les menus, le pied de page et quelques autres éléments. Vous pouvez également le faire en modifiant les fichiers cshtml du répertoire home et shares.

8 Scanner et boire

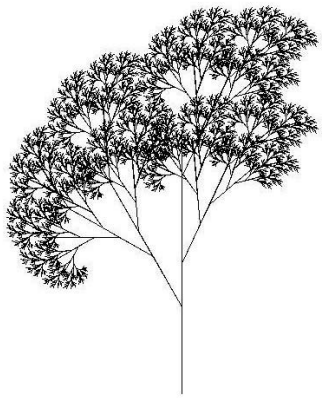
C'est la dernière étape et la plus importante! Quand quelqu'un de la maison va à la cave et récupère une bouteille, il a juste besoin de scanner le QR Code avec son téléphone. Cela fonctionne avec tous les smartphones ! Et au pire, s'ils oublient leur téléphone ou ne veulent pas le faire pour une excuse bidon, je peux toujours le faire a posteriori quand la bouteille est sur la table ou même à la poubelle. Et pour ceux qui auraient installé Cortana sur leur téléphone, la recherche visuelle n'est plus directement disponible. L'application QR for Cortana permet de créer un filtre dans les filtres photo et donne un accès direct à la recherche visuelle de Bing. Application disponible ici : http://www.windowsphone.com/store/app/qr-for-cortana/3517e8f0-33c6-4c4c-a293-14fd08722364 Fig.21

Fig.20

Fig.21

La première fois qu'ils scannent un QR Code, ils doivent s'authentifier avec leur login et leur mot de passe. Cela n'est à faire qu'une seule fois en cochant la case "se souvenir de moi" et c'est bon pour de très nombreuses utilisations. C'est donc très simple et très efficace. Maintenant, je peux boire (avec modération bien sûr) de nombreuses bouteilles de vin sans avoir à refaire mon inventaire trop régulièrement. Cet article vous a donc permis de créer un site Web de gestion d'inventaire. Vous pouvez bien entendu étendre cette application et l'utiliser pour n'importe quel type de gestion de stock. L'avantage est la simplicité à la fois de l'application, du code et surtout du côté utilisateur. L'effort d'inventaire est à faire principalement lors de la création des éléments dans la base de données et lors du collage de l'autocollant contenant le QR Code sur l'objet.

La notion d'arbres en informatique



Les arbres informatiques sont parmi les plus utiles - et les plus fascinantes - structures de données évoluées que l'on puisse manipuler.

Steve Begelman
Steve.Begelman@gmail.com

Les arbres sont utiles pour modéliser et éventuellement persister des données aussi différentes, par exemple, que :

- La décomposition en pièces ou en "composants" d'un objet industriel aussi complexe que l'aile d'un avion Boeing 747, soit plusieurs millions de pièces,
- La structure d'un système de gestion de fichiers du type UNIX, dont on a un bel exemple grâce à Wikipédia est reproduite sur la page Fig.1 ,
- L'arbre généalogique de votre famille sur une longue période,
- Une recette gastronomique de la cuisine française faisant appel à des "sous-recettes" ou une formule chimique, par exemple un plat classique à sauce du répertoire d'Auguste Escoffier :Volaille->Garniture->Accompagnement->Sauce->...

On devrait se poser de suite la question principale : "*Qu'est-ce que un 'nœud' dans un arbre informatique?*" La réponse est simple: c'est un objet appartenant à l'arbre, parce qu'il possède quatre propriétés: un **Niveau**, un **Rang**, et un **Père** (parfois dénommé "propriétaire" ou bien "ascendant") ; il possède également, bien sûr, une **Valeur**.

Explications

Examinons l'exemple d'arbre simple suivant à seulement 5 niveaux (figure 2).

Pour bien comprendre le code C#, on peut commencer par matérialiser cet exemple d'arbre à 14 nœuds à travers les données du tableau suivant, qui réalise la transformation « arbre » vers « tableau » :

Niveau	Rang	Propriétaire	Valeur	Remarques
1	1	0	45337	Tête de l'Arbre : pas de propriétaire
2	1	1	54677	
2	2	1	12319	Nœud Intermédiaire : a des dépendants
3	1	1	4545	
3	2	1	4733	
3	3	1	33	Nœud Feuille sur son niveau
3	4	2	13	
3	5	2	760	
4	1	4	3411	
4	2	4	11	
5	1	2	911	Nœud Feuille au niveau le plus profond
5	2	2	34	Nœud Feuille au niveau le plus profond
5	3	2	6777	Nœud Feuille au niveau le plus profond
5	4	2	4370	Nœud Feuille au niveau le plus profond
5	4	2	4370	Doublon : Erreur
4	1	4	4901	Adresse (N,R) en Doublon : Erreur

Il est très important de comprendre que les deux dernières lignes de ce tableau vont provoquer des erreurs : l'une parce que la ligne est dupliquée

Fig.1

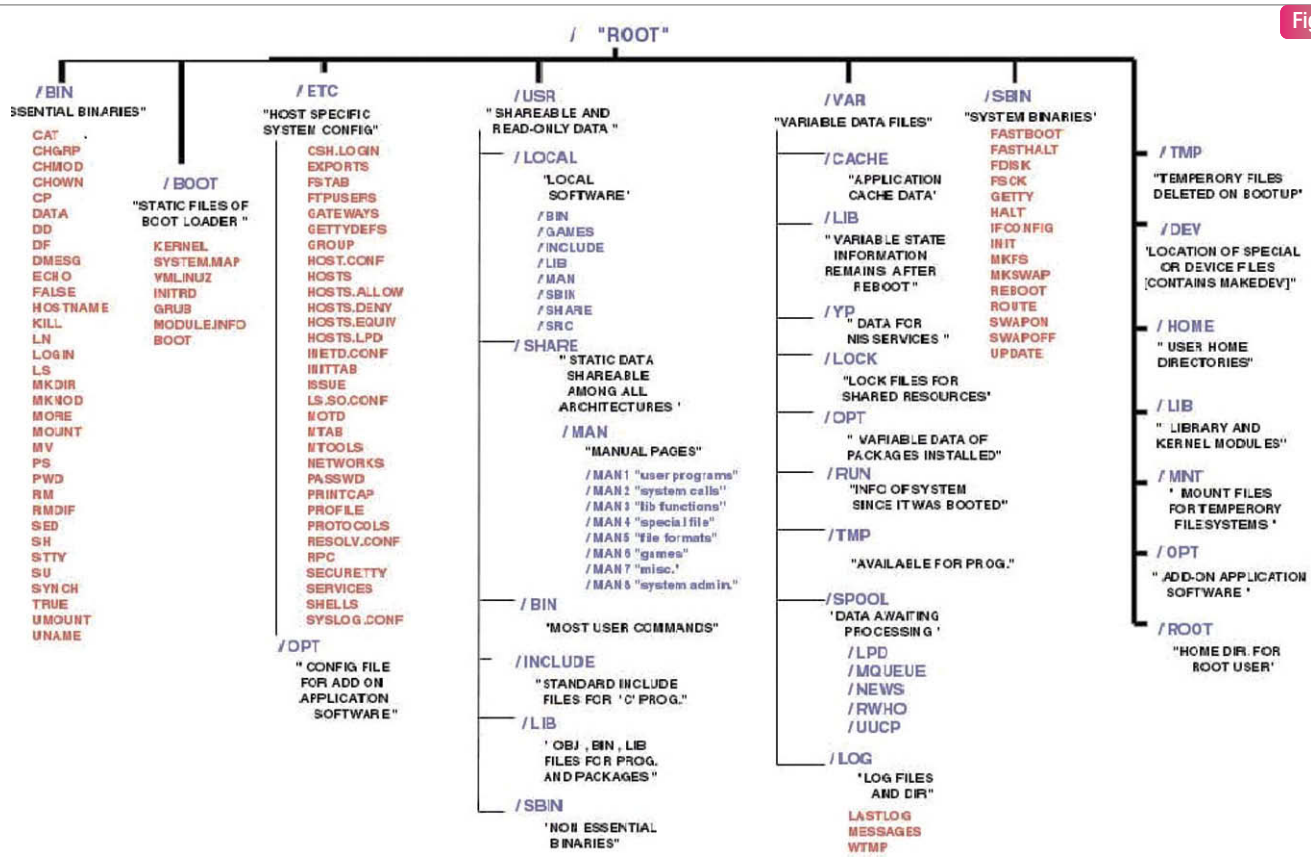
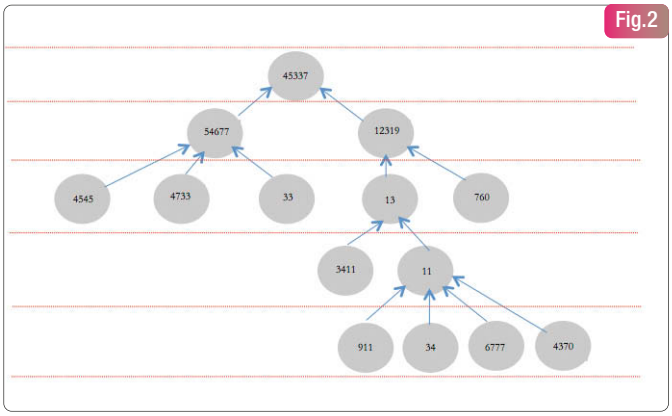


Fig.2



(interdite), l'autre parce que l'adresse « 4 1 » existe déjà, également interdite.

A strictement parler, l'arbre n'est pas une structure générique de données simple au même type que la Liste, la Table de Hachage ou le Dictionnaire, par exemple : c'est plutôt une "superstructure" des données, faisant appel à certaines structures de données génériques.

La preuve, c'est que ni le STL C++ ni les génériques C# n'en contiennent. Notre solution utilise intensément les <List> et les Templates.

Pour reprendre l'exemple de l'aile d'un avion Boeing 747, seule une gestion arborescente permet de répondre à des questions du type *: Si la "pièce" XYZ n'est pas en stock aujourd'hui, quels sont les composants que je ne peux pas fabriquer ou assembler, car embarquant cette même pièce ?*

Dans le monde réel, l'information qui décrit les caractéristiques du nœud (la pièce dans notre exemple) résidera quasi-systématiquement dans une table référentielle d'une base de données :

à vous d'instancier et gérer l'arbre hiérarchique des dépendances et relations entre pièces.

Notons que le CLR (Common Language Runtime) contient un objet de type "TreeView", mais il ne s'agit *pas* ici d'une librairie facilitant l'affichage des Arbres dans des pages Web ou dans Windows Forms : il s'agit d'une bibliothèque extensible de bas niveau permettant de manipuler des arbres en mémoire indépendamment de leur présentation à l'utilisateur.

Une des caractéristiques les plus saisissantes de l'arbre, c'est que, à la différence d'autres structures de données évoluées du type "liste dichotomique" ou liste à clef de hachage, les arbres existent dans la nature, tels quels, et nous n'avons qu'à nous en inspirer.

C'est à double titre que les arbres nous intéressent : aussi bien les branches et les feuilles (la partie visible) que la partie cachée et invisible consistant en racines et nodules, nous inspirent.

Ce qui nous frappe d'emblée lorsque nous examinons un arbre centenaire, c'est que nous ne voyons pas combien de branches et de feuilles il contient.

C'est là la première contrainte d'une bonne implémentation d'un arbre : on doit pouvoir modéliser un arbre de complexité très grande, dont nous ne connaissons ni le nombre de "branches" (les "niveaux"), ni le nombre de feuilles de la branche.

Exemples en C#

Dans l'implémentation C#/.Net que j'ai confectionnée, les contraintes habituelles de l'arbre sont respectées à la lettre, c'est-à-dire :

1) Nous modélisons un arbre de *profondeur arbitraire*; en réalité, le nombre de nœuds dans l'arbre n'est limité que par le nombre d'éléments que l'on puisse stocker dans un objet de type Liste<T> via les génériques, soit un nombre très grand,
2) L'arbre est générique, c'est-à-dire que la valeur stockée dans chaque nœud de l'arbre est faiblement typée, de telle manière qu'il puisse stocker soit une chaîne de caractères: soit, par exemple, "AX1233CXC T22244", une des pièces (un vérin poussoir, en l'occurrence) entrant dans la composition de notre aile d'avion Boeing cité plus haut, soit un entier du type "349422339", la clef numérique d'une pièce référencée dans le référentiel d'une base de données,
3) La valeur de la clef n'est pas unique : dans un arbre, la clef peut figurer à différents endroits d'un arbre, et l'arbre est faiblement typé,
4) Les contraintes purement techniques et "formelles" sont implémentées, ce qui veut dire que :
   - Chaque nœud n'est rattaché qu'à un seul et unique père-propriétaire au niveau immédiatement supérieure
   - L'intégrité physique de l'arbre est préservée, c'est-à-dire qu'il n'y a pas d'orphelins (nœuds sans père), ni de suppression d'un nœud tant qu'il a des nœuds dépendants,
   - Chaque arbre a un seul nœud « tête ».

   Notons que dans notre implémentation l'ordre d'insertion des nœuds n'est pas imposée, avec comme corollaire qu'un parcours séquentiel de la liste ne donne aucune information vraiment utile sur l'arborescence au sens strict, bien qu'il donne des informations importantes sur les valeurs stockées individuellement dans l'arbre.

   Qui plus est, l'intégrité de l'arbre est garantie à tout moment, lors des insertions et suppressions; si ceci n'était pas le cas, on devrait déléguer aux méthodes de parcours de l'arbre une détection d'anomalie dans la structure arborescente, une véritable "boîte à Pandore" qui accroît la complexité...
5) Toutes les opérations classiques (insertion, suppression, modification) sont supportées, en respectant les contraintes de l'intégrité physique et logique notées ci-dessus; c'est donc un choix « ceinture et bretelles », car les méthodes d'accès s'appuient sur le fait que l'arbre est intègre vis-à-vis des accesseurs...

L'implémentation

Notre implémentation s'appuie essentiellement sur un structure C#/.Net du type :

List<Tree<T>.TreeNode>

Nous utilisons donc une Liste contenant des "structures anonymes" de type TreeNode, dont la clef est une template T. Bien qu'il y ait bien d'autres structures adaptables, ce type est particulièrement performant, car sa caractéristique principale, c'est qu'il est très performant pour la gestion d'arbres de grande taille, faisant l'objet d'insertions et modifications importantes, car le coût d'une insertion/modification/suppression, c'est le coût d'une Objet.Add(...) élémentaire C#, soit un coût très faible.

Ceux qui connaissent déjà bien les arbres remarqueront que mon objectif n'a pas été de gérer un "B-Tree", dont la propriété principale est l'équilibre binaire de l'arbre : chaque "niveau" de l'arbre n'a que deux et seulement deux nœuds, de manière à ce que l'accès en lecture via une recherche dite dichotomique soit excessivement rapide et déterministe, donnée par la célèbre formule :

$$Nombre\_De\_Recherches = O(\log_b n) opérations$$

Bien que exceptionnellement utile dans certains cas d'analyse, les "Balanced Binary Tree" (arbre binaire équilibré) ont souvent un coût très élevé en mise à jour : l'insertion d'un nouveau nœud peut entraîner la réorganisation complète de l'arbre.

Notre approche est à la fois plus intuitive et plus pragmatique: nous nous basons sur une pile ("stack" en anglais), et on se contente "d'empiler" de nouveaux nœuds dessus, sans s'occuper de l'ordre d'empilement.

Toutes les méthodes habituelles pour parcourir l'arbre sont implémentées, soit de "haut en bas" via une méthode du type :

List<Tree<T>.TreeNode> traverseByAddressFull()

Soit "de bas en haut", via une méthode du type :

List<Tree<T>.TreeNode> ascendByAddress()

C'est le "Saint Graal" de la gestion des arbres : parcourir l'arbre dans n'importe quel sens, à partir de n'importe quel endroit, simplement en fournissant soit son adresse (Niveau,Rang,Proprietaire,Valeur), soit en fournissant sa valeur de clef de type T, par exemple : "349422339", tout en s'assurant du respect des contraintes d'intégrité.

Nous remarquerons dans la littérature académique hyper-abondante sur la gestion des arbres, que la plupart des exemples d'implémentation sont faits en C/C++. Pourquoi? Tout simplement parce que, d'une part, à la différence du C#/.Net, le C/C++ supporte depuis son origine des variables et méthodes-membres "statiques" au niveau de la Méthode, alors que le C# ne les supporte qu'au niveau de la Classe. D'autre part, le C/C++ supporte des pointeurs génériques, et bien que le C# les supporte, il m'a semblé plus prudent de stocker le père d'un nœud non pas via pointeur, mais dans une structure anonyme du type :

```
public struct TreeNode {
    public int level;
    public int rank;
    public int owner_node;
    public T theValue;
}
```

En quoi ceci est-il si important ? Parce que pour résoudre le problème technique central de l'arbre (nous ne connaissons ni sa "profondeur", ni le nombre de ses nœuds, tous les deux étant dynamiques et en principe non-bornés) il faut une "ruse" pour gérer cela, et cette ruse est en réalité assez simple : *utiliser des méthodes récursives qui manipulent des structures statiques, et des variables d'état statiques*, afin d'empiler "à l'infini" des nœuds intermédiaires lors de notre parcours de l'arbre.

Si vous croyez donc que la récursivité est une chose peu utile à laquelle n'y adhèrent que les universitaires, vous avez tort : tout algorithme qui doit "stocker" des informations qui seront traitées « à leur tour » et dans un ordre inconnu, tire parti efficacement de la récursivité. Utiliser la récursivité pour calculer un factoriel ne m'a jamais enchanté (très "vieille école", je préfère le calcul itératif pour les tâches triviales, car propre et facilement lisible), mais pour les arbres, entre autres, elle est quasi indispensable.

L'enjeu principal de la récursivité? Savoir quand il faut s'*arrêter* d'appeler la méthode récursive, pour que cela ne "cycle" pas, c'est-à-dire entrer dans une boucle infinie, car aucune solution n'est trouvée; j'ai tenté de mettre les garde-fous nécessaires dans mon code pour empêcher qu'une méthode "cycle".

Il y a aussi la question des optimisations : certaines optimisations sont très simples. Par exemple, dans un arbre d'un million de pièces, où on veut trouver les descendants de la pièce "XYZ", la première chose à faire, avant même d'effectuer un parcours récursif, c'est de vérifier que la pièce existe, pour éviter un parcours inutile. Quand cela me semblait pertinent, j'ai rajouté ces tests qui optimisent le code.

Certains disent donc que le C/C++ est plus adapté à la gestion des arbres, car la "staticité est mieux faite et plus souple, mais je ne suis pas de cet avis. Vous verrez que dans notre implémentation vous ne déclarez ni ne manipulez des structures statiques dans le C# : cela est fait pour vous dans les constructeurs de la Classe principale. Qui plus est, l'exceptionnelle richesse et rapidité des génériques "tem platées" qui existent en natif dans le CLR compense largement cet inconvénient.

Vous remarquerez que j'ai fait le choix d'encapsuler toutes les structures et variables statiques, au prix d'un appel obligatoire à ma méthode void reinitializeTrees(); l'alternative m'aurait contraint de rajouter des structures statiques dans la signature de mes principales méthodes, mais cette option me paraissait pas élégante et de surcroît contraire aux principes de l'encapsulation.

Bien évidemment, toute cette complexité est cachée : le développeur se contentera d'utiliser les membres définis dans l'interface.

Mais il y a un effet négatif qui est une conséquence du caractère statique de la gestion via des « statiques » : il faut impérativement "nettoyer" les structures statiques (ou bien les réallouer sur une nouvelle instance) entre chaque "parcours" de l'arbre, et cela est illustré dans les exemples d'appels fournis dans le driver.

Il y a des alternatives à la solution que je vous propose, et notamment le stockage des nœuds dans un document XML, avec des accesseurs en XSLT et des contrôles en XSD, mais à mon avis, la solution que je vous propose est plus efficace (par plusieurs ordre de grandeur!) et est sensiblement plus souple. L'approche XML pour moi vient, en fait, "en aval" de notre solution, car il est assez facile de XMLiser ou HTMLiser le résultat d'un query du type :

List<Tree<T>.TreeNode> traverseByAddressFull()

On peut ainsi l'intégrer sans complications dans une page Web ou un Web Service.

Description de l'implémentation

L'objectif de mon implémentation, c'est de produire une bibliothèque du style .dll ou .lib (si vous utilisez occasionnellement, comme moi, les Framework MONO sous UNIX). On peut utiliser n'importe quelle version de Visual-Studio (voir l'en-tête des sources pour une description des prérequis).

La solution que nous avons fournie se décompose en 5 fichiers source : un qui décrit les Interfaces formelles, et un qui implémente les Interfaces. Deux fichiers décrivent l'un l'interface des méthodes de tri paramétrable, l'autre l'implémentation des routines de tri. C'est tous ces quatre fichiers qui doivent produire le "Trees.ddl" que vous inclurez et référencerez dans votre projet « Visual-Studio ».

Le fichier TestTreeDriver.cs est hyper utile, mais il n'est pas nécessaire pour générer une DLL sous Windows ou avec MONO : il s'agit d'un "driver", avec une méthode Main(), qui permet de démontrer simplement – ou plutôt tester – toutes les fonctionnalités de l'interface, à travers des exemples pédagogiques.

Ainsi, dans la solution (.sln) actuelle que vous pouvez facilement télécharger à l'adresse http://dotnet.developpez.com/telecharger/detail/id/4324/Gestion-des-Arbres-en-C-Net. il s'agit d'un projet du type "console exécutable", ne faisant pas appel à l'indispensable Framework de tests dont on a besoin dans le monde réel : la solution actuelle peut être transformée en simple "bibliothèque DLL" en moins d'une minute. En quelques clics, il suffit de télécharger la solution, cliquer le fichier de solution « Trees.sln », puis builder, afin de debugger et tester le driver.

Il vous restera ensuite à mettre tous mes tests du driver actuel dans un Framework de tests du type N-Unit.

Il est possible que vous ayez besoin de modifier le code, afin de modifier le comportement de la librairie : à titre d'exemple, j'ai expliqué au début de cet article que chaque nœud de mon arbre stocke une clef non-unique, mais il est possible que votre arbre n'autorise que des clefs uniques, et cela nécessitera des modifications dans mes méthodes de validation; si

mon travail est bien fait, il n'y aura pas de modifications à effectuer dans les structures que je manipule.
Comme il se doit, tout est dit dans l'Interface, dont la description suit. Si on comprend bien l'interface, on sera à même de l'étendre en cas de besoin, soit :

```
// Implementation File : TreeImplementation.cs
// ----------------------------------------
using System;
using System.Collections.Generic;
using System.Linq;

namespace your_inverted_domain.utilities.trees
{
  // All Interface Members:
  public interface ITree<T>
  {
    int countElementsInTree();

    int count { get; set; }
    Tree<T>.TreeNode originValue { get; set; }

    T[] getValues();
    List<T> getValuesList();

    int addNodeToTree(int level, int rank, int owner_node, T newElement);
    int updateNodeInTree(int level, int rank, int owner_node, T modifiedElement);

    bool getIsUniqueValue(T inputValue);

    Tree<T>.TreeNode getNodeFromUniqueValue(T inputValue);

    T getNodeValueByAddress(int level, int rank, int owner_node);

    int[] getNodeAddressFromUniqueValue(T inputValue);

    bool deleteLeafNodeByAddress(int level, int rank, int owner_node);
    bool deleteLeafNodeByUniqueValue(T value);

    List<Tree<T>.TreeNode> getAllNodes();

    int getMaxDepth();
    int getMaxWidthAtLevel(int level);

    List<Tree<T>.TreeNode> traverseByAddressFull(int? level, int? rank, int? owner_node,
T theValue);
    List<Tree<T>.TreeNode> traverseByUniqueFull(T theValue);

    List<Tree<T>.TreeNode> traverseByAddressOneLevel(int? level, int? rank, int? owner
_node, T theValue);
    List<Tree<T>.TreeNode> traverseByUniqueOneLevel(T theValue);

    List<Tree<T>.TreeNode> ascendByAddress(int? level, int? rank, int? owner_node, T the
Value);
    List<Tree<T>.TreeNode> ascenByUnique(T theValue);

    bool deleteBranchInTreeByNode(Tree<T>.TreeNode value, Tree<T> myTree);

    void reinitializeTrees();
  }
}
```

Quant à l'interface pour les tris, c'est simple, avec 4 variantes de la méthode Compare() :

```
using System;
using System.Collections.Generic;
using System.Linq;

namespace your_inverted_domain.utilities.trees
{
  public interface IComparer<T>
  {
    int Compare(Tree<T>.TreeNode x, Tree<T>.TreeNode y);
  }
}
```

Si vous debuggez le code, vous constaterez que les méthodes récursives ne sont pas « tail récursifs », c'est-à-dire qu'il y a un empilement des appels de fonction, avec comme résultat, à première vue assez bizarre, que le statement « return » ne renvoie pas le résultat et ne « sort pas », et c'est un peu déroutant au début, même si le résultat final est correct.
Il y a une solution qui s'articule autour du « trampoleening » pour faire (une simulation) de la « tail récusion » en C#, mais qui n'est pas nécessaire dans le cas de notre implémentation. On peut espérer qu'une future CLR .Net comprendra une annotation C# du type :
[TailRecursion]
Après avoir testé mon implémentation des arbres, je me suis rendu compte que, pour de très grandes arbres, il aurait été intéressant de persister l'image d'un arbre sur disque, afin de ne pas avoir à le recréer de toutes pièces au démarrage d'une application : cela plairait, par exemple, aux gestionnaires de portefeuilles qui gèrent des "méta-portefeuilles" (un portefeuilles contenant N autres sous-portefeuilles, chacun avec des milliers de titres), car cela permet de faire des simulations d'impact très rapides, sans avoir à recréer l'arbre de toutes pièces.
C'est la Sérialisation .Net qui est parfaitement adaptée à cette tâche.
Quand cela me semblait utile, et notamment pour les algorithmes un peu complexes des parcours, j'ai commenté le code pour que le lecteur l'appréhende mieux.

Toute l'actualité des technologies et du développement sur www.programmez.com

Ma première application pour Office 365

Office 365, vous en avez certainement déjà entendu parler. Que ce soit dans sa version pour le particulier ou pour l'entreprise, cette offre de Microsoft permet de bénéficier de la suite Office et de services en ligne pour un abonnement mensuel (https://products.office.com/fr-fr/products). Dans sa déclinaison pour les professionnels, ce sont plusieurs produits serveurs tels qu'Exchange (messagerie), Lync (messagerie instantanée et conférence en ligne) et SharePoint (espaces collaboratifs, portail, gestion documentaire) qui deviennent ainsi disponibles en mode "Online", c'est-à-dire hébergés et administrables dans le Cloud Microsoft.

Gaëtan BOUVERET
Consultant SharePoint
gbouveret@infinitesquare.com
INFINITE SQUARE

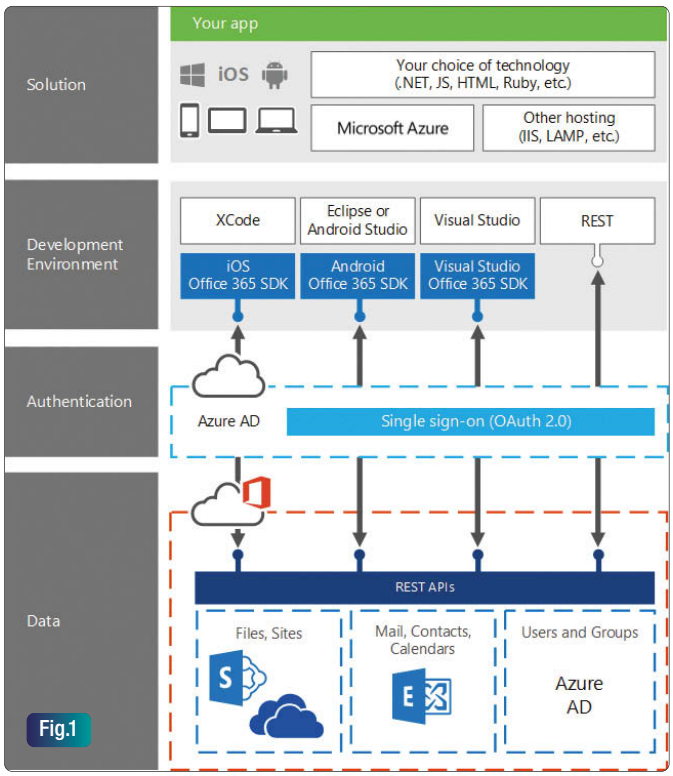


Figure 1: Schéma général de l'API

Historiquement, chaque produit bénéficie de ses propres composants d'administration et de possibilités d'extension, généralement sous la forme de commandes PowerShell et de Web services. Mais depuis quelques mois, une API a été mise à disposition afin d'offrir une interface de programmation transverse, ne nécessitant plus une connaissance spécifique du logiciel ni un environnement de développement particulier, en se basant sur les standards REST et OData 4.0 pour la représentation des données, et OAuth 2.0 via Azure Active Directory (AAD) pour la partie authentification et gestion des autorisations. Des SDK ont été publiés pour les différents IDE (Visual Studio, Eclipse, Android Studio et XCode) pour simplifier la manipulation de l'API par encapsulation de toute la plomberie REST. Enfin, on retrouve plusieurs Starter Kits sur Github (https://github.com/OfficeDev) dont nous allons nous servir pour cet article : des applications .NET (MVC, Windows 8) mais aussi iOS ou Android **Fig.1**.

Les services disponibles

Actuellement l'API Office 365 propose la gestion des emails, des contacts et calendriers, la gestion des fichiers de son espace OneDrive et plus largement la manipulation des sites SharePoint et de leurs contenus. A côté de cela, Azure AD offre la gestion des utilisateurs et des groupes et surtout la partie authentification / SSO.

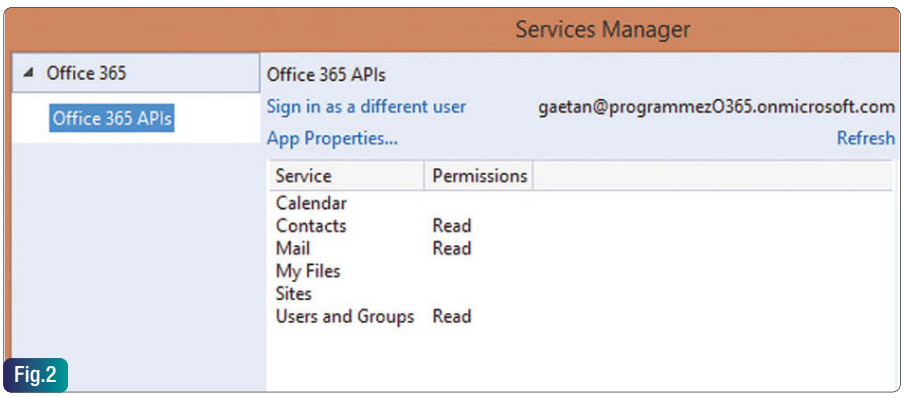
Un dernier service, le « Discovery service » permet de trouver les informations sur les Web services disponibles pour un environnement Office 365, pratique dans les scénarii d'application multi-tenant pour lesquels nous ne connaissons pas les URL à l'avance.

Prérequis

Avant de nous lancer dans le développement de l'application, quelques prérequis sont nécessaires.

Tout d'abord, il vous faudra évidemment un compte Office 365. Si vous n'en avez pas, la version d'essai d'une durée de 30 jours est disponible en s'inscrivant à l'adresse suivante (sans nécessiter d'entrer un numéro de carte bleue): http://go.microsoft.com/fwlink/p/?LinkID=403802. Il est conseillé aussi d'avoir une souscription sur Azure, qui pourra être associée au tenant Office 365. Une version d'essai est disponible sur http://azure.microsoft.com/fr-fr/; vous pouvez aussi y accéder depuis la section « Active Directory » dans l'administration Office 365.

Pour finir, téléchargez et installez la dernière version de l'extension « Microsoft Office 365 API Tools » pour Visual Studio 2013 (http://aka.ms/OfficeDevToolsForVS2013) afin de pouvoir utiliser le SDK.



Initialisation du projet

Commencez par récupérer le projet « O365-WebApp-MultiTenant » sur Github (https://github.com/OfficeDev/O365-WebApp-MultiTenant) et ouvrez la solution avec Visual Studio. Mettez à jour les références manquantes à l'aide de Nuget en compilant le

projet. Il faut ensuite paramétrer les permissions qui seront octroyées à l'application par l'utilisateur. Pour cela, un clic droit sur le projet, puis « Add » / « Connected Service » pour enregistrer l'application et sélectionner les droits d'accès aux services d'Office 365 qui seront nécessaires. Connectez-vous en cliquant sur « Register the app », puis cochez dans « Contacts » et « Mail » la lecture des éléments de l'utilisateur, et dans « Users and Groups » le single sign-on et la lecture du profil. Vous devriez avoir ceci : Fig.2.

Enfin, cliquez sur « App Properties » pour pouvoir désigner votre application comme étant multi-tenant en sélectionnant « Multiple Organization » : cela signifie que l'application pourra être disponible pour les environnements Office 365 de n'importe quelle société. Vous pouvez aussi changer le nom de l'application si vous le désirez Fig.3.

Validez le tout et exécutez le code en mode debug. Le starter kit intègre déjà le listing des contacts Outlook (pensez à en ajouter quelques uns depuis Office 365). Connectez-vous via le lien « Sign-in » : à la première connection il vous sera demandé d'accepter la demande d'autorisation de l'application : Fig.4.

Rendez-vous ensuite sur « My Contacts » pour vérifier que le service fonctionne correctement Fig.5.

A noter que contrairement à ce qui est affiché, le starter kit ne contient pas l'implémentation de la création, modification et suppression des contacts.

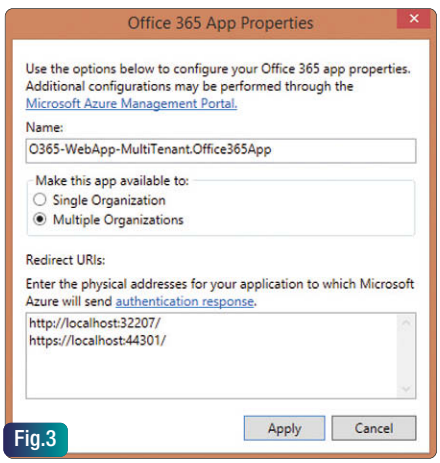


Fig.3

```
public class MyMessage
{
    public string Subject { get; set; }

    public string Sender { get; set; }

    public DateTime Received { get; set; }
}
```
Fig.6

Aller un peu plus loin

Maintenant que tout fonctionne, nous allons réaliser une page récupérant les 10 derniers emails reçus.

Commencez par créer une nouvelle classe « MyMessage » dans le répertoire « Models » avec les propriétés suivantes : Fig.6.

Ensuite, ajoutez un nouveau contrôleur vide appelé « MessagesController » et recopiez-y le code de « ContactsController.cs » en prenant soin de spécifier les bons namespaces; nous allons le modifier et en profiter pour le commenter un peu.

Tout d'abord, la première partie permet de récupérer le contexte d'authentification à l'aide de l'identité de l'utilisateur et l'identifiant de son tenant. A noter que le projet intègre un système de gestion de cache en base de données des jetons issus de l'Azure AD Authentication Library (« ADAL »), ceci afin d'éviter de redemander à chaque page le jeton. Ceci est effectué à l'aide de la classe « ADALTokenCache » et d'une base de données locale Fig.7.

Ensuite, étant donné que nous sommes en mode multi-tenant, nous devons découvrir l'URL du Web service via le « Discovery service ». Vous noterez que nous devons fournir à chaque fois le jeton d'authentification, qui sera passé dans les appels REST sous-jacents Fig.8.

Nous demandons la capacité « Mail » ce coup-ci, car ce sont les messages électroniques qui nous intéressent, mais vous avez aussi « Contacts », ou « MyFiles » selon le service

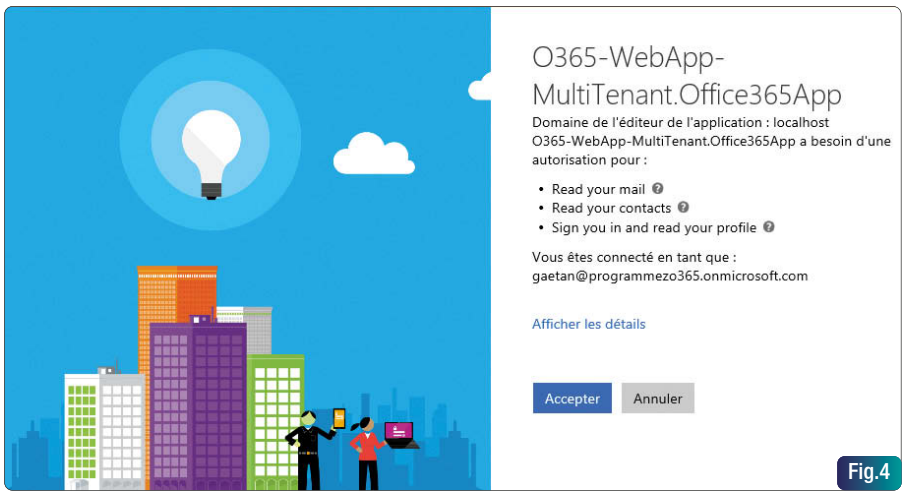


Fig.4

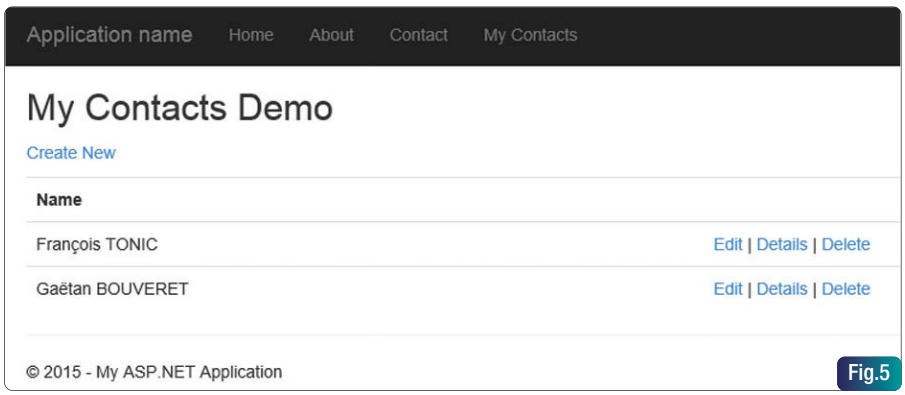


Fig.5

```
var signInUserId = ClaimsPrincipal.Current.FindFirst(ClaimTypes.NameIdentifier).Value;
var userObjectId = ClaimsPrincipal.Current.FindFirst("http://schemas.microsoft.com/identity/claims/objectidentifier").Value;
var tenantId = ClaimsPrincipal.Current.FindFirst("http://schemas.microsoft.com/identity/claims/tenantid").Value;

AuthenticationContext authContext = new AuthenticationContext(string.Format("{0}/{1}", SettingsHelper.AuthorizationUri, tenantId),
    new ADALTokenCache(signInUserId));
```
Fig.7

```
DiscoveryClient discClient = new DiscoveryClient(SettingsHelper.DiscoveryServiceEndpointUri,
    async () =>
    {
        var authResult = await authContext.AcquireTokenSilentAsync(SettingsHelper.DiscoveryServiceResourceId,
            new ClientCredential(SettingsHelper.ClientId, SettingsHelper.AppKey),
            new UserIdentifier(userObjectId, UserIdentifierType.UniqueId));

        return authResult.AccessToken;
    });

var dcr = await discClient.DiscoverCapabilityAsync("Mail");
```
Fig.8

désiré. L'URL étant désormais connue, elle sera utilisée pour initialiser le client Outlook afin de récupérer les messages de l'utilisateur courant (« Me ») : Fig.9.
Nous trions par date de réception décroissante (« OrderByDescending(..) ») et spécifions une pagination de 10 éléments (« Take(10) »). Nous ne récupèrerons ici que la première page.

```
var messagesResult = await exClient.Me.Messages.OrderByDescending(m => m.DateTimeReceived).Take(10).ExecuteAsync();

var messages = messagesResult.CurrentPage;
foreach (var msg in messages)
{
    myMessages.Add(new MyMessage
    {
        Subject = msg.Subject,
        Sender = msg.Sender.EmailAddress.Address,
        Received = msg.DateTimeReceived.Value.DateTime
    });
}
```

Fig.9

Remarque : les messages sont issus par défaut de la boîte de réception (« Inbox »), mais il est possible de les récupérer depuis d'autres répertoires, spéciaux ou non, par exemple depuis les brouillons comme ceci : Fig.10.

```
await exClient.Me.Folders.GetById("Drafts").Messages.ExecuteAsync();
```

Fig.10

Créez ensuite la vue correspondante pour le rendu des messages en prenant soin d'utiliser le bon modèle, ajouter le lien vers l'action du nouveau contrôleur dans la navigation située dans le fichier _Layout.cshtml. Ré-exécutez le projet et rendez-vous sur votre nouvelle page pour consulter les derniers emails reçus :Fig.11.

Mes emails

Received	Sender	Subject
17/02/2015 11:56:19 1	gbouveret@infinitesquare.com	Démarrage projet O365
17/02/2015 13:16:06	gbouveret@infinitesquare.com	RE: Démarrage projet O365
17/02/2015 13:16:21	[redacted]	Demande de contact

Fig.11

En cas de souci avec les permissions, par exemple si l'application a changé ses droits, connectez-vous sur https://myapps.microsoft.com avec votre compte Office 365. Vous y retrouverez toutes les applications ajoutées, et pourrez les retirer : Fig.12.
A la prochaine connexion, vous devrez de nouveau autoriser l'application, cette fois-ci avec les permissions à jour.

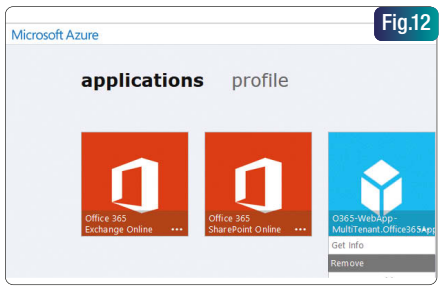


Fig.12

Conclusion

Vous voilà maintenant prêt à manipuler l'API d'Office 365 pour intégrer les données de vos utilisateurs au sein de vos applications afin, par exemple, de proposer des alertes ou de sauvegarder automatiquement des informations dans Office 365 (contacts, documents etc...).
Vous pourrez aussi tester les autres Starter Kits pour développer une application mobile.
Sachez enfin qu'il existe un site permettant de tester vos requêtes REST et consulter votre jeton OAuth : https://oauthplay.azurewebsites.net/.

Tout Programmez!
sur une clé USB

Tous les numéros de Programmez! depuis le n°100.

29,90 €*

Clé USB 2 Go. Photo non contractuelle. Testé sur Linux, OS X, Windows. Les magazines sont au format PDF.

** tarif pour l'Europe uniquement. Pour les autres pays, voir la boutique en ligne*

Commandez directement sur notre site internet : www.programmez.com

Vos premiers pas en Machine Learning

“The goal is to turn data into information, and information into insight.” Carly Fiorina

Thomas OUNNAS,
Data scientist chez Xebia

Yoann BENOIT,
Data scientist chez Xebia

Au travers de cet article, nous vous proposons une initiation au Machine Learning, qui est un sous-ensemble de l'Intelligence Artificielle. Cette discipline permet l'analyse et la construction d'algorithmes capables d'apprendre à partir de données d'entrée. Grâce à deux Use Cases simples représentant deux cas de figure classiques, nous allons vous décrire la démarche de résolution d'un problème dans ce domaine via deux branches principales :

- L'apprentissage supervise,
- L'apprentissage non-supervise.

Sachant le résultat d'un phénomène, l'apprentissage supervisé permet de comprendre son mécanisme afin d'être capable de bien prédire les résultats à venir et de prendre des décisions pertinentes.

Par phénomène, on peut envisager plusieurs choses, par exemple si le fait qu'un e-mail soit un spam ou non, le prix du mètre carré dans une ville donnée, l'identification d'un code postal, la probabilité qu'un client achète tel ou tel produit, etc.

Derrière cette idée, on identifie une variable Y (prix, label spam/non-spam, etc.) représentant le résultat que l'on cherche à prédire et que l'on appelle la cible. On dispose aussi d'un ensemble de variables dites features, qui permet de capter et décrire le phénomène étudié.

L'apprentissage non-supervisé, quant à lui, ne suppose pas la connaissance d'une variable cible. L'idée n'est donc pas de comprendre ou prédire une cible précédemment identifiée mais plutôt de trouver une structure, un pattern dans les données. On parle alors de clustering : l'idée est de former des groupes de points homogènes, sachant qu'un point est défini par un ensemble de features. La manière de regrouper ces points reste à être définie et correspond au choix d'une métrique. Il est courant d'utiliser la distance euclidienne, mais il existe une multitude de choix possibles, souvent associés à des cas d'études précis.

A - Clustering pour la compression d'image

Le clustering est un type d'apprentissage non-supervisé permettant de trouver des patterns dans les données en les segmentant en plusieurs groupes homogènes. Nous avons donc à notre disposition un dataset composé de plusieurs features, sans target. Il s'agit alors, dans la plupart des algorithmes de clustering, de choisir le nombre de groupes K que l'on veut créer, et de rassembler les données en K groupes distincts. Les données au sein d'un même groupe doivent être proches, en relation avec la métrique choisie.

Un exemple d'utilisation du clustering consiste à compresser la taille d'une image, composée de plusieurs milliers de couleurs, en réduisant leur nombre à travers des regroupements. Typiquement, il peut y avoir plusieurs tons de rouge dans une image, le but est alors de repérer ces différents tons et de les regrouper en seulement quelques groupes de rouges (voir un seul groupe).

Attention, nous sommes loin ici des techniques de compression classiques utilisées dans le traitement d'image. Nous cherchons simplement à diminuer le nombre de couleurs dans une image, tout en gardant un aspect visuel correct, pour diminuer sa taille en mémoire.

Le Clustering en pratique :

Chargement d'une image

Chargeons maintenant une image pour mieux se rendre compte du problème.

```
from PIL import Image # Image loading
im = Image.open('perroquet.jpeg', 'r')
width, height = im.size
print width, height
```

L'image a pour dimensions 1024x768 pixels, elle comporte 238135 couleurs distinctes et sa taille est de 122 Ko au format .jpeg Fig.1.

Fig.1

Création du dataset en lien avec l'image

Pour réduire le nombre de couleurs, regroupons celles qui se ressemblent dans les 238135 présentes. Pour ce faire, il nous faut donc construire un dataset qui va représenter la couleur de chaque pixel. En utilisant la codification RGB, il nous faut donc trois colonnes dans notre dataset et autant de lignes qu'il y a de pixel, soit 1024*768 = 786432 lignes. Ce dataset se construit rapidement à partir de l'image chargée en appelant la fonction getdata() :

```
import numpy as np # Manipulate arrays
image_array = np.array(im.getdata()) / 255.
```

On obtient un tableau de la forme suivante :

R	G	B
0.266667	0.364706	0.756863
0.262745	0.360784	0.752941
0.282353	0.380392	0.772549

Notez que l'on a divisé les données du tableau par 255 simplement pour avoir des données comprises entre 0 et 1 (on parle alors de rescaling), car c'est un pré-requis de la fonction qui va permettre d'afficher les images par la suite.

Une méthode de clustering : K-Means

Nous allons maintenant pouvoir regrouper les couleurs proches grâce à une méthode de clustering appelée K-Means.

Avant de tenter de former des groupes, il est nécessaire de définir la notion de proximité entre des points, c'est-à-dire de se fixer une distance adaptée au sujet. Dans notre cas, la distance la plus intuitive à utiliser est la distance euclidienne. Deux points vont donc être considérés comme proches lorsque leur distance sera faible. En considérant deux points A(xA, yA, zA) et B(xB, yB, zB), elle est définie de la manière suivante :

L'algorithme du K-Means va chercher à construire des centroïdes pour chaque groupe déterminé. Ces centroïdes ne sont pas des points appartenant au dataset, mais des points fictifs représentant les barycentres de chaque groupe de points formé.

Le K-Means est un algorithme itératif, c'est à dire qu'en partant de centroïdes initialisés aléatoirement, on va mettre à jour à chaque itération leurs

coordonnées, comme représentées dans l'image ci-dessous : Fig.2.
L'image (a) montre les données de départ qui sont visiblement séparables en deux groupes distincts. Deux centroïdes vont alors être initialisés de façon aléatoire (figure (b)). Les points sont alors affectés à son centroïde le plus proche, formant des clusters (figure (c)). Les centroïdes se replacent en calculant la distance moyenne de tous les points qui lui sont affectés (figure (d)). Au fil des mises à jour, certains points seront certainement plus proches d'autres centroïdes, ce qui modifiera leur appartenance aux clusters (figure (e)).. Le processus de mise à jour des centroïdes et d'affectation des points dans les clusters est itéré plusieurs fois jusqu'à ce que les clusters ne changent plus (ou très peu). On obtient alors nos groupes de points finaux, comme montré sur la figure (f).

Clustering des couleurs de l'image

Maintenant que nous savons comment fonctionne l'algorithme du K-Means, appliquonsle sur le dataset représentant notre image initiale. Comme beaucoup d'autres algorithmes de Machine Learning, le K-Means est implémenté dans scikit-learn, et son utilisation devient très simple. Un seul paramètre reste à définir : le nombre de groupes souhaité, c'est-à-dire le nombre de couleurs que l'on va utiliser dans l'image.
L'implémentation se fait alors de la manière suivante :

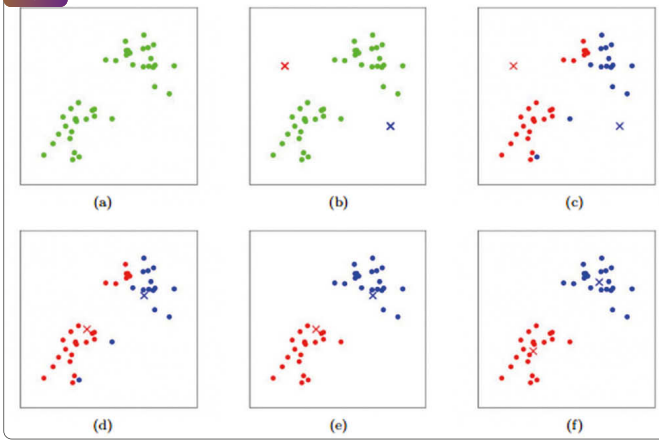
```
# Selection of the desired number of colors
n_colors = 64
# Taking a random sample of points of the data
image_array_sample = shuffle(image_array, random_state=0)[:10000]
# Fitting the K-Means
kmeans = KMeans(n_clusters=n_colors,random_state=0).fit(image_array_sample)
# Affecting each pixel to a cluster
labels = kmeans.predict(image_array)
```

Le paramètre random_state=0 n'est pas obligatoire et permet de régler l'initialisation aléatoire.

Reconstruction de l'image compressée

Nous avons alors à disposition les coordonnées RGB de chaque centroïde (présentes dans l'attribut *kmeans.cluster_centers_*), ainsi que le label du groupe dans lequel chaque pixel est affecté (présent dans labels).
A présent, nous devons reconstruire un tableau de même dimension que l'image originale, car pour le moment nous avons simplement un tableau de trois colonnes et d'autant de lignes que de pixels. Nous allons donc construire une fonction permettant de reconstruire l'image avec ces différentes contraintes. Elle a donc pour paramètres les centroïdes, les labels de chaque pixel, ainsi que la largeur et la hauteur de l'image.

Fig.2




```
def recreate_image(cluster_centers, labels, width, height):
    # Dimension of a center (3 in our case)
    d = cluster_centers.shape[1]
    # Initialization of the compressed image
    image_comp = np.zeros((height, width, d))
    # Initialization of a counter on the pixels' indexes
    label_idx = 0
    # Reconstruction of the image, pixel by pixel
    for i in range(height):
        for j in range(width):
            image_comp[i][j] = cluster_centers[labels[label_idx]]
            label_idx += 1
    return image_comp
```

Tout est maintenant à notre disposition pour reconstruire l'image avec un nombre de couleurs réduit.

```
# Recreation of the array with the initial image's dimensions
image_comp = recreate_image(kmeans.cluster_centers_, labels, width, height)

# Plotting the image
plt.figure(2)
plt.clf()
ax = plt.axes([0, 0, 1, 1])
plt.axis('off')
plt.title('Compressed Image (K=64)')
plt.imshow(image_comp);
```

Fig.3

Voici le résultat obtenu en se restreignant à 64 couleurs sur l'image : Fig.3.
En se restreignant à 64 couleurs, l'image garde donc une bonne qualité. Le perroquet est bien reproduit grâce à des couleurs très marquées, à la différence du fond (mer et nuages) du fait de l'importance de la continuité des couleurs (du bleu clair au bleu foncé). De ce fait, en se restreignant à un plus petit nombre de couleurs, l'impression de continuité entre les couleurs est plus difficilement réalisable, d'où ces "coupures" entre chaque couleur.
Si on compare les tailles des images, l'image originale faisait 122 Ko, alors que l'image compressée réduite à 64 couleurs ne fait plus que 52 Ko.
Voici les résultats obtenus pour plusieurs K. On peut voir que notre oeil a rapidement l'impression que l'image est l'originale (à partir de 64 ou 128), alors que le nombre de couleurs est beaucoup plus faible Fig.4.
Nous avons ici illustré un algorithme de clustering dans le cadre d'une réduction de nombre de couleurs dans une image. Il y a cependant de nombreuses autres applications au clustering, comme par exemple chercher à regrouper des utilisateurs d'un site Internet selon leur comportement sur celui-ci, ou bien créer des zones pour optimiser les circuits de distribution d'une entreprise. Pour plus d'informations, se référer au tutoriel sur le même sujet sur le site de scikit-learn.

B - Régression pour la prédiction de location de vélos

Une application typique dans un contexte d'apprentissage supervisé consiste à faire de la prédiction (de ventes, de trafic, de pannes, etc.). Nous

vous proposons ici d'appliquer un modèle de régression pour prédire la demande de location de vélos dans une ville. Le but est alors de faire émerger une variable cible (notre target), qui n'est autre que le nombre de vélos loués dans une journée, en prenant en compte différentes variables (jour de la semaine, température, météo, etc.). Pour cela, nous allons appliquer une démarche systématique en Data Science :

- Exploration des données : compréhension et connaissance des données à disposition.
- Feature Engineering : création des features qui vont servir de base pour nos modèles.
- Splitting : séparation des données en trois jeux différents, un pour l'apprentissage (training set), un pour le tunning des paramètres (validation set) et le dernier pour l'évaluation du modèle (test set).
- Modelling: création et paramétrage d'un modèle de Machine Learning pour pouvoir réaliser la prédiction.

Chargement des données

Avant toute chose, nous allons charger les données en allant les chercher à leur source.

```
import pandas as pd
from StringIO import StringIO
from zipfile import ZipFile
from urllib import urlopen

url = urlopen("https://archive.ics.uci.edu/ml/machine-learning-databases/00275/Bike-Sharing-Dataset.zip")
zipfile = ZipFile(StringIO(url.read()))
data = pd.read_csv(zipfile.open("day.csv"), parse_dates = ['dteday'])
```

Voyons à quoi ressemblent les données pour mieux comprendre comment les traiter par la suite :

```
data.head()
```

Fig.4

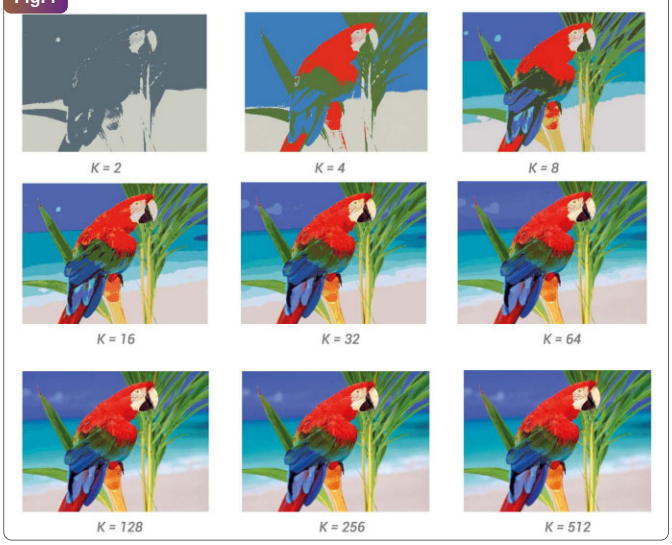


Fig.5. Plusieurs features sont à notre disposition pour l'analyse : certaines correspondent à la date (saison, année, mois, etc.), d'autres au temps (météo, température, humidité, etc.). Les trois dernières colonnes sont des variables cibles . "Casual" correspond au nombre de locations par des utilisateurs occasionnels, "Registered" au nombre de locations par des utilisateurs enregistrés et "Count" au nombre total de locations. Pour la suite de notre étude, nous allons nous concentrer sur les features season, *mnth, holiday, weekday, workingday, weathersit, temp, hum* et *windspeed*, et tenter de prédire la target casual. Restreignons donc notre dataset :

```
target = data.casual
list_features = ['season','mnth','holiday','weekday','workingday','weathersit','temp','hum','windspeed']
X = data[list_features]
```

Exploration des données et Feature Engineering

Deux types de variables sont à notre disposition :

- Des variables catégorielles : elles représentent un nombre fini de cas possibles, sans qu'il n'y ait forcément de relation d'ordre entre les états (typiquement les jours de la semaine)
- Des variables continues : le nombre de possibilités est infini et il existe une relation d'ordre entre les valeurs observées (typiquement la température)

Variables catégorielles

Observons à présent les différentes modalités des variables catégorielles à notre disposition.

```
print "Nombre de modalités de la variable %s: %d" % ('season',len(X['season'].unique()))
print "Nombre de modalités de la variable %s: %d" % ('mnth',len(X['mnth'].unique()))
print "Nombre de modalités de la variable %s: %d" % ('holiday',len(X['holiday'].unique()))
print "Nombre de modalités de la variable %s: %d" % ('weekday',len(X['weekday'].unique()))
print "Nombre de modalités de la variable %s: %d" % ('workingday',len(X['workingday'].unique()))
print "Nombre de modalités de la variable %s: %d" % ('weathersit',len(X['weathersit'].unique()))

Nombre de modalités de la variable season: 4
Nombre de modalités de la variable mnth: 12
Nombre de modalités de la variable holiday: 2
Nombre de modalités de la variable weekday: 7
Nombre de modalités de la variable workingday: 2
Nombre de modalités de la variable weathersit: 3
```

Comme on peut le constater, les variables holiday et workingday ne possèdent que deux modalités : c'est ce que l'on appelle des **dummies** (ou variables binaires). Ainsi, la variable workingday va prendre la valeur 1 s'il s'agit d'un jour ouvré en dehors des vacances, sinon la valeur 0.

Les autres variables possèdent plusieurs modalités, sans relation d'ordre entre elles : la valeur 1 de weekday n'est en aucun cas inférieure à la valeur 5. Il est donc nécessaire d'encoder ces variables de manière différente afin d'éviter une relation hiérarchique implicite entre les modalités. On parle

	instant	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	casual	registered	cnt
0	1	2011-01-01	1	0	1	0	6	0	2	0.344167	0.363625	0.805833	0.160446	331	654	985
1	2	2011-01-02	1	0	1	0	0	0	2	0.363478	0.353739	0.696087	0.248539	131	670	801
2	3	2011-01-03	1	0	1	0	1	1	1	0.196364	0.189405	0.437273	0.248309	120	1229	1349
3	4	2011-01-04	1	0	1	0	2	1	1	0.200000	0.212122	0.590435	0.160296	108	1454	1562
4	5	2011-01-05	1	0	1	0	3	1	1	0.226957	0.229270	0.436957	0.186900	82	1518	1600

Fig.5

alors de **dummisation** (ou d'encodage disjonctif complet) : c'est le procédé consistant à réécrire une variable catégorielle en plusieurs variables binaires (dummies). Par exemple, la variable season possède 4 modalités. Nous allons donc créer 4 nouvelles variables *season_1, season_2, season_3 et season_4* qui vaudront 0 ou 1 suivant la valeur considérée de *season*. Appliquons maintenant le processus de dummisation à toutes les variables catégorielles.

```
list_dummies = ['season', 'mnth', 'weekday', 'weathersit']
for v in list_dummies:
    dummies = pd.get_dummies(X[v], prefix=v+"_")
    X = pd.concat([X, dummies], axis=1)
    del X[v]
pd.set_option('max_columns', None)
X.head()
```

Fig.7.

	holiday	workingday	temp	hum	windspeed	season__1	season__2	season__3	season__4	mnth__1	mnth__2	m
0	0	0	0.344167	0.805833	0.160446	1	0	0	0	1	0	0
1	0	0	0.363478	0.696087	0.248539	1	0	0	0	1	0	0
2	0	1	0.196364	0.437273	0.248309	1	0	0	0	1	0	0
3	0	1	0.200000	0.590435	0.160296	1	0	0	0	1	0	0
4	0	1	0.226957	0.436957	0.186900	1	0	0	0	1	0	0

Variables continues

```
X[['temp','hum','windspeed']].describe()
```

Fig.8.

	temp	hum	windspeed
count	731.000000	731.000000	731.000000
mean	0.495385	0.627894	0.190486
std	0.183051	0.142429	0.077498
min	0.059130	0.000000	0.022392
25%	0.337083	0.520000	0.134950
50%	0.498333	0.626667	0.180975
75%	0.655417	0.730209	0.233214
max	0.861667	0.972500	0.507463

Regarder la description statistique globale des variables continues dans notre dataset est une étape importante car elle permet de l'explorer ou de le modifier lorsqu'il contient des valeurs manquantes ou des valeurs aberrantes. Cela permet aussi d'avoir une idée des ordres de grandeur des variables ainsi que de leur distribution.

Splitting

Ce concept consiste à séparer les données en au moins deux parties : une base d'apprentissage nommée **training set** et une deuxième qui servira à tester notre apprentissage sur des données nouvelles encore jamais observées, on parle de **test set**.

Il est aussi recommandé d'ajouter une troisième partie, appelée **validation set**; il va être utilisé pour valider un modèle et ses paramètres lors de sa construction (à noter qu'il existe d'autres méthodes d'évaluation qui ne nécessitent pas la présence de ce validation set, parmi lesquelles la cross-validation ou encore le principe du Bootstrap, mais dont l'explication sortirait du cadre de cet article). Le test set servira alors pour la validation finale du modèle, nous donnant ainsi une estimation fiable de notre erreur.

Construisons maintenant nos training, validation et test set :

```
import numpy as np
np.random.seed(seed=1234)
ind_train = np.random.choice(len(X),0.5*len(X),replace=False)
ind_val_test = list(set(range(len(X))) - set(ind_train))
ind_val = np.random.choice(ind_val_test,0.7*len(ind_val_test),replace=False)
ind_test = list(set(ind_val_test) - set(ind_val))

X_train, X_val, X_test, target_train, target_val, target_test = X.iloc[ind_train], X.iloc
[ind_val], X.iloc[ind_test], target.iloc[ind_train], target.iloc[ind_val], target.iloc[ind_test]
```

Modelling

Cette phase permet de construire un modèle de régression et de prédire la demande de vélos en fonction des features présentées précédemment. Le modèle étudié ici sera un arbre de décision, dont le principe est relativement intuitif et qui permet une bonne interprétation des résultats. Il se construit au fur et à mesure en se définissant des règles, qui vont soit nous donner un résultat, soit nous mener à une autre règle. Prenons un exemple : je prends mon lait le matin. Première chose, je regarde sa date de validité, si elle est bonne je peux le boire, sinon, je regarde s'il est périmé de moins d'une semaine. Si oui, je peux le boire, sinon, je l'ouvre et regarde son aspect pour savoir si je peux le boire ou non.

L'un des gros avantages de la librairie Scikit-Learn de Python est la simplicité d'utilisation des différentes méthodes de Machine Learning. Ainsi, une fois que les différents datasets sont construits, il devient relativement simple de passer à la phase d'apprentissage et de prédiction.

```
from sklearn.tree import DecisionTreeRegressor
model = DecisionTreeRegressor()
model.fit(X_train, target_train) # On fit le modèle sur le training set
prediction = model.predict(X_test) # On effectue les prédictions sur le test set
```

Une fois que les prédictions sont faites, il est important de pouvoir mesurer leur qualité. Pour cela, il faut avant tout se fixer une mesure de l'erreur. Dans notre cas, nous allons utiliser le **Root Mean Squared Error**. C'est une mesure classique de l'erreur faite par un modèle de régression. Nous allons donc préalablement définir la fonction de calcul de l'erreur pour l'appliquer ensuite aux prédictions faites sur le training set, ainsi que sur le test set.

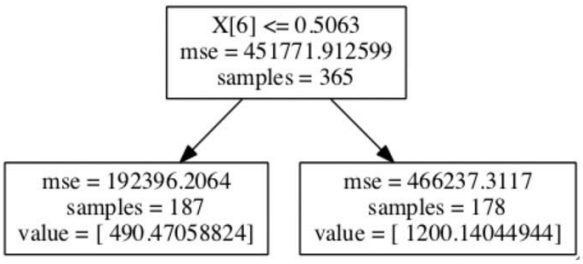
```
def rmse(obs, pred):
    return np.sqrt(np.mean((obs-pred)**2))

rmse_train = rmse(target_train, model.predict(X_train))
rmse_test = rmse(target_test, prediction)

print "Train Error: %.2f" % rmse_train
print "Test Error: %.2f" % rmse_test
Train Error: 0.00
Test Error: 501.31
```

Remarque : *il existe déjà une implémentation du mse (sans la racine) dans sklearn mais nous avons pris le parti de le réécrire dans un but didactique.*

Avec un Train Error faible et un Test Error élevé, nous nous retrouvons dans une situation délicate appelée l'**overfitting**. Du fait qu'il n'existe aucune erreur sur le training set, l'overfitting peut être vu comme de l'apprentissage par coeur. L'arbre de décision, auquel nous n'avons donné aucune restriction dans les paramètres, a réussi à trouver des combinaisons de caractéristiques lui permettant à tous les coups de prédire exactement la bonne valeur sur les données qu'il a apprises. Le problème est que dès que de nouvelles données, encore jamais observées, sont soumises au modèle, il va très souvent faire de fortes erreurs, comme ont peut le constater sur le rmse du test set. Afin d'éviter ce phénomène, il faut faire ce que l'on appelle du **tunning** : trouver les paramètres à donner au modèle qui vont minimiser l'erreur sur des données non observées. Pour notre arbre de décision, l'un des paramètres sur lesquels il est possible de jouer est sa profondeur maximale, soit son nombre de règles. En effet, plus un arbre est profond, moins il y a de données dans ses feuilles et plus il y a de chance de faire de l'overfitting. En fixant par exemple la profondeur maximale à 1, nous obtenons l'arbre suivant :



Nous n'avons autorisé ici qu'une profondeur de 1 : l'arbre a donc été contraint de trouver la meilleure caractéristique et son seuil, ceci pour séparer en deux les données puis estimer la valeur à donner.
Appliquons maintenant une démarche complète de tunning qui va nous permettre de déterminer quelle est la profondeur optimale de l'arbre nous permettant de minimiser l'erreur sur le validation set.

```
scores = []
for max_depth in [1, 2, 3, 4, 5]:
    model = DecisionTreeRegressor(max_depth=max_depth)
    model.fit(X_train, target_train)
    pred = model.predict(X_val) # Prediction on the validation set
    scores.append((max_depth, rmse(target_val, pred)))
tab_scores = pd.DataFrame(scores, columns=['max_depth', 'rmse']).sort('rmse')
tab_scores
```

	max_depth	rmse
3	4	405.900402
2	3	414.233804
4	5	418.921402
1	2	465.677455
0	1	613.281642

Remarque: *encore une fois, il existe une fonction (gridSearch) déjà implémentée dans sklearn qui peut, à l'aide du paramètre n_jobs, paralléliser le tunning. Cependant nous avons pris le parti d'en écrire une version simplifiée afin de mieux illustrer le principe sous-jacent.*

Une fois le tunning fait, nous pouvons reconstruire un arbre de décision plus stable.

```
model = DecisionTreeRegressor(max_depth=tab_scores.max_depth.iloc[0])
model.fit(pd.concat([X_train, X_val]), pd.concat([target_train,target_val]))
prediction = model.predict(X_test) # Prediction on the test set

rmse_train = rmse(pd.concat([target_train,target_val]), model.predict(pd.concat([X_
train, X_val])))
rmse_test = rmse(target_test, prediction)

print "Train Error: %.2f" % rmse_train
print "Test Error: %.2f" % rmse_test

Train Error: 319.07
Test Error: 383.05
```

Comme on peut le constater, l'erreur sur le train set a considérablement augmenté en comparaison avec la prédiction sans tunning. En revanche, l'erreur constatée sur le test set est largement inférieure, démontrant ainsi la plus grande capacité de généralisation de notre nouvel arbre. Il faut en effet accepter de ne pas pouvoir prédire à 100% les données sur lesquelles on a appris le modèle pour pouvoir avoir un score correct sur de nouvelles données. Nous sommes dorénavant capables grâce à ce modèle de faire des prédictions correctes sur le nombre de vélos qui vont être loués.

Pour aller plus loin

Maintenant que la démarche globale en Data Science est bien comprise, il est possible d'utiliser des modèles plus complexes afin d'améliorer les scores de prédiction sur le test set. Une évolution classique lorsque l'on utilise un arbre de décision est la **Random Forest**. Une Random Forest consiste à modéliser plusieurs arbres et à moyenner leurs résultats pour être plus robuste qu'un simple arbre de décision. Pour chaque arbre de la Random Forest, seulement une partie des variables vont lui être proposées, et les données sont sélectionnées aléatoirement avec remise. Cela permet de rajouter plus d'aléatoire dans les données et dans la sélection des features, ce qui est statistiquement plus viable dans le but d'avoir une meilleure capacité de généralisation. Encore une fois, il est nécessaire de faire un bon tunning pour éviter l'overfitting. Pour une Random Forest, il est classique d'optimiser la profondeur maximale de chaque arbre, ainsi que le nombre minimal d'éléments par feuille. C'est ce que nous allons faire maintenant avec une Random Forest composée de 2000 arbres.

```
from sklearn.ensemble import RandomForestRegressor
scores = []
for max_depth in [2, 4, 6]:
    for min_samples_leaf in [3, 5, 7]:
        model = RandomForestRegressor(n_estimators=2000, max_depth=max_depth,
min_samples_leaf=min_samples_leaf)
        model.fit(X_train, target_train)
        pred = model.predict(X_val)
        scores.append((max_depth, min_samples_leaf, rmse(target_val, pred)))
tab_scores = pd.DataFrame(scores, columns=['max_depth','min_samples_leaf', 'rmse'])
.sort('rmse')
tab_scores
```

Une fois les meilleurs paramètres déterminés, nous pouvons finalement les utiliser pour construire notre modèle et faire nos prédictions. Maintenant que nous n'avons plus besoin de tester plusieurs jeux de paramètres, nous pouvons encore plus augmenter le nombre d'arbres, ce qui permet en général d'obtenir de meilleurs résultats.

```
max_depth_best = tab_scores.max_depth.iloc[0]
min_samples_leaf_best = tab_scores.min_samples_leaf.iloc[0]

model = RandomForestRegressor(n_estimators=5000, max_depth=max_depth_best,
min_samples_leaf=min_samples_leaf_best)
model.fit(pd.concat([X_train, X_val]), pd.concat([target_train, target_val]))
prediction = model.predict(X_test)

rmse_train = rmse(pd.concat([target_train, target_val]), model.predict(pd.concat([X_
train, X_val])))
rmse_test = rmse(target_test, prediction)

print "Train Error: %.2f" % rmse_train
print "Test Error: %.2f" % rmse_test

Train Error: 235.60
Test Error: 340.87
```

Comme on peut le voir, l'utilisation de modèles plus complexes tels qu'une Random Forest permet d'améliorer les scores de prédiction, à la fois sur le training set et sur le test set.

Conclusion Générale

Nous avons travaillé à travers cet article sur deux cas d'applications en Data Science. L'un portait sur de l'apprentissage non-supervisé, et l'autre sur de l'apprentissage supervisé. On peut toutefois citer un troisième volet d'application : Le Reinforcement Learning. Cela regroupe un ensemble d'algorithmes qui vont faire leurs prédictions en apprenant de leurs erreurs au fur et à mesure, et s'adapter aux éventuels changements. A noter cependant que les deux premières branches de la Data Science sont les plus sollicitées.

Tests unitaires en Drupal

Une journée de travail passe très vite lorsque vous développez. Cependant il est souvent difficile de prendre un peu de temps pour écrire quelques lignes supplémentaires afin de tester votre code. Même si au premier abord, cela ne semble pas visible, l'opération est nécessaire si vous ajoutez de nouvelles fonctionnalités à votre projet Drupal...

Christophe Villeneuve
Consultant IT pour Neuros, auteur du livre "Drupal7 en mode avancé" aux éditions Eyrolles et auteur Editions ENI, Rédacteur pour WebRIVER, membre des Teams DrupalFR, AFUP, LeMug.fr, Drupagora, elePHPant...

Il existe de nombreux frameworks PHP spécialisés dans les tests unitaires. Les plus connus en France sont : SimpleTest, PHPUnit, Atoum.

Module

L'article ne vous expliquera pas comment construire un module, mais nous utiliserons le hook_help() de l'API Drupal pour vérifier avec chaque framework que le bouton Help correspond bien au contenu de chaque module. Le module de base 'Programmez' servira de fondation et le hook_help() se présente de la manière suivante :

```
<?php
// Fichier programmez.module

/**
 * Implementation hook_help()
 */
function programmez_help($path, $arg) {
  switch ($path) {
    case 'admin/help#programmez':
      $output = '<h3>Votre magazine mensuel</h3>';
      $output .= '<p>Vous vous trouvez sur la page HELP du module Programmez</p>';
      return $output;
  }
}

?>
```

Vous avez aussi besoin du fichier *.info pour faire fonctionner celui-ci, il se présente de la manière suivante :

```
; Fichier programmez.info

name = Programmez
description = Module simple de base avec l'utilisation du hook_help()
package = Programmez
core = 7.x

files[] = programmez.module

;configure =
version = "7.x-1.0"
```

SimpleTest

SimpleTest est depuis de nombreuses années le framework de tests unitaires PHP pour Drupal. Son but est de vous faciliter la validation fonctionnelle de vos pages Web, mais aussi de vos différents modules réalisées en PHP.

Installation

Depuis la version 7 de Drupal, SimpleTest est disponible dans la partie CORE de celui-ci et il vous suffit de cocher la ligne Testing (à partir du menu module) pour le rendre disponible et utilisable.
Ainsi, après avoir activé cette ligne, vous pouvez accéder à l'interface de configuration de celui-ci (admin/config/development/testing) pour jouer les différents scénarios disponibles et déjà prévus.

Créer vos propres tests unitaires avec SimpleTests

Pour mettre en place vos propres tests dans vos modules, un certain nombre d'opérations sont nécessaires pour que vos tests soient pris en compte. Tout d'abord, vous créez le module : 'programmez_simpletest' sur le même principe que le module générique et ajoutez la ligne suivante dans votre fichier *.info :

```
files[] = programmez_simpletest.test
```

Vous pouvez garder la même structure pour le fichier *.module. La seule différence que vous réalisez concerne le contenu de la page aide. Maintenant, pour rendre testable ce module, vous créez un autre fichier appelé 'programmez_simpletest.test'. Il vous sera utile car vous allez étendre la classe DrupaUnitTestCase pour que vous puissiez tester le module, de la manière suivante :

```
<?php
class programmez_simpletestTestCase extends DrupalWebTestCase
{
// votre code
}
?>
```

Ensuite, certaines fonctionnalités indispensables doivent être présentes pour que les tests soient pris en compte avec le module Core 'Testing', comme la fonction appelée getInfo :

```
public static function getInfo()
{
  return array(
    'name' => 'Help',
    'description' => 'Exemple de tests function Help - module Programmez SimpleTest',
    'group' => t('Programmez'),
  );
}
```

Cette fonctionnalité a pour but de fournir les informations de ce module et avoir une trace dans votre cas de test. Ensuite, l'autre fonction indispensable concerne l'appel aux tests, c'est à dire le point de démarrage de la classe, qui se présente de la manière suivante :

```
public function setUp()
{
  parent::setUp('programmez_simpletest');
}
```

Pour vérifier que votre classe soit prise en compte, cliquez sur le bouton « nettoyer l'environnement » de la page de configuration des tests (admin/config/development/testing). Maintenant que votre module est identifié dans la liste des modules de la page des tests, vous pouvez créer vos propres fonctions de tests. Pour cela, vous créez ceci :

```
public function testProgrammezSimpletest() {
  $this->drupalGet('admin/help/programmez_simpletest');
 }
```

Cette fonction permet de vérifier si la page existe. La dernière étape consiste à exécuter les tests. A partir de la page testing (admin/config/development/testing), vous lancez les tests pour obtenir un résultat de synthèse. Le résultat montre que tout s'est bien exécuté. En cas de problème, vous aurez eu des messages d'alerte avec la ligne qui pose problème. Enfin, vous pouvez aller plus loin avec ce framework, soit à partir du site de celui-ci, soit à partir de la page dédiée de Drupal.
source : http://www.simpletest.org
exemple avec SimpleTest https://www.drupal.org/project/examples
https://www.drupal.org/simpletest-tutorial-drupal7

PhpUnit

PHPUnit est un autre framework de tests unitaires PHP; ce projet est très actif. Actuellement, il s'utilise avec Drupal sous la forme d'un module externe. La technique présentée fonctionne aussi bien pour Drupal 7 ou 8. Pour cela, vous utiliserez le Bootstrap du CMS en attendant de voir la disponibilité de modules spécifiques.

Nous choisirons la version 3.7.28 (dernière de la branche) et surtout compatible avec les versions antérieures à PHP 5.3 pour Drupal 7. Il existe PHP Unit 4.x qui nécessite PHP 5.3 et +

L'installation de PHPUnit

PHPUnit s'utilise avec le paquet PEAR et par conséquent vous installez celui-ci avec PHP. Pour cela, vous effectuez les opérations suivantes (si vous êtes sous Linux) :

```
$ sudo pear upgrade pear
$ pear config-set auto_discover 1
$ pear install pear.phpunit.de/PHPUnit
```

Depuis le 31 Décembre 2014, l'installation de PHPUnit s'utilise avec l'extension Phar (PHP 5.3).

Utilisation de PHPUnit

Pour communiquer entre PHPunit et Drupal, nous créons un fichier test 'programmez.test.php'. La méthode choisie, utilisera le fichier Bootstrap proposé par Drupal, que vous effectuez de la façon suivante :

```
<?php
$_SERVER['REMOTE_ADDR'] = '127.0.0.1';

define('DS', DIRECTORY_SEPARATOR);
define ('PATH', DS . '..' . DS . '..' . DS . '..' . DS . '..' . DS );
define('DRUPAL_ROOT', realpath(dirname(__FILE__) ) . PATH );
set_include_path(DRUPAL_ROOT . get_include_path());

include_once DRUPAL_ROOT . 'includes/bootstrap.inc';
drupal_bootstrap(DRUPAL_BOOTSTRAP_FULL);
?>
```

Ce script s'occupe de rechercher le fichier 'boostrap' dans Drupal pour pouvoir utiliser les différents Hooks utilisés dans votre module. La deuxième étape permet d'étendre la classe PHPUnit_Framework_TestCase, qui accueillera l'ensemble des tests que vous désirez effectuer de la manière suivante :

```
class programmez extends PHPUnit_Framework_TestCase
{
// votre code
 }
?>
```

Il ne reste plus qu'à insérer l'ensemble des fonctionnalités à tester que vous souhaitez opérer dans cette classe, c'est à dire comme vous avez l'habitude de le faire avec PHPUnit. Vous allez maintenant vérifier si le hook_help() retourne bien le bon contenu par rapport à celui du module. Pour cela, vous écrivez la fonction suivante :

```
<?php
 public function test_function_help()
 {
   $output = '<h3>Votre magazine mensuel</h3>';
   $output .= '<p>Exemple de module pour différents tests unitaires avec PhpUnit</p>';

  $this->assertEquals(programmez_phpunit_help('admin/help#programmez_phpunit'),
$output);
 }
?>
```

La dernière étape vérifie si le test passe à partir de votre terminal (console) de la manière suivante :

```
$ cd sites/all/modules/programmez_phpunit/
$ phpunit programmez_phpunit.test
```

Si le résultat est différent d'un point '.', le test est mal passé pour différentes raisons. Enfin, le framework PHPUnit propose une documentation très complète avec des exemples.
Site : http://phpunit.de/

Atoum

Atoum est un Framework français, il vous permet aussi d'effectuer des tests unitaires en PHP. Il s'agit d'une bonne alternative aux autres frameworks présentés. Atoum exploite les formats natifs proposés par PHP 5.3, en l'occurence le format d'archive PHAR que nous utiliserons. Avec l'arrivée de Drupal 8, il y a un grand intérêt à commencer à l'utiliser quel que soit votre système d'exploitation (Windows, Linux, OS X) car de nombreux frameworks génériques sont déjà compatibles; aujourd'hui, pour bien commencer vous en verrez une orientation de son utilisation.

La technique proposée aujourd'hui est une des possibilités que l'auteur de l'article a retenue et qu'il vous présente en avant première.

Installation

L'installation s'effectue en 2 parties. Tout d'abord vous téléchargez l'archive directement sur le site officiel pour obtenir la dernière version stable d'Atoum ou directement à partir du lien suivant :
http://downloads.atoum.org/nightly/mageekguy.atoum.phar
Ce fichier doit se placer à la racine de votre site Web.
Ensuite pour effectuer l'installation sous linux, vous passez en ligne de commande (à partir du terminal) la ligne suivante :

```
$ php -d phar.readonly=0 mageekguy.atoum.phar --update
```

pour obtenir le résultat suivant :

```
Checking if a new version is available... Done !
There is no new version available !
```

Maintenant que le Framework Atoum est installé, vous passez au module Drupal.

Utilisation

Pour utiliser Atoum avec Drupal, vous devez séparer les fonctionnalités nécessaires pour utiliser un module avec le framework.
Pour cela, vous réutilisez le module de base (présenté un peu plus haut) de la manière suivante :

```
<?php
// Fichier : programmez_atoum.php

require_once (realpath(dirname(__FILE__)).'/src/Programmez.php');

/**
 * Implements hook_help().
 */
function programmez_atoum_help($path, $arg)
{

$c=new Programmez;
$resultat= $c->help ($path,$arg='');
return $resultat;
}
```

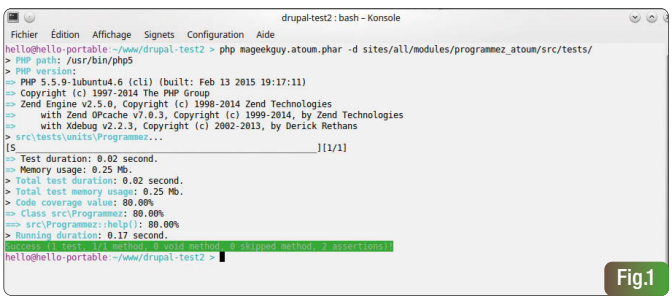
Tout d'abord, vous devez inclure les différentes fonctions utilisées pour les tests d'Atoum et vous appelez ce fichier sous la forme d'un objet. Le résultat obtenu restera identique.
Le fichier que vous appelez se décomposera la manière suivante :

```
<?php
namespace src;

class Programmez
{
  public static function help($path, $arg)
  {
    switch ($path)
    {
      case 'admin/help#programmez_atoum':
    $output = '<h3>Votre magazine mensuel</h3>';
    $output .= '<p>Exemple de module pour différents tests unitaires avec Atoum</p>';
    return $output;
    }
  }
}
```

Vous retrouvez la même fonctionnalité que vous connaissez sauf qu'il se trouve encapsulé dans une classe, qui est nécessaire pour Atoum. Comme il s'agit, d'une classe, vous pouvez toujours l'exécuter de la même façon comme une programmation objet.

```
<?php
$c=new Programmez;
$path='admin/help#programmez_atoum';
$resultat= $c->help ($path,$arg='');
echo $resultat;
?>
```



Et vous l'exécutez en ligne de commande :

```
$ cd votreSiteDrupal
$ php sites/all/modules/programmez_atoum/src/Programmez.php
```

Pour en obtenir le résultat suivant :

```
<h3>Votre magazine mensuel</h3><p>Exemple de module pour différents tests unitaires avec Atoum</p>
```

Pour l'étape 2, vous préparez un fichier de tests appelé 'Programmez.php' dans le chemin 'scr/tests/units'. Ce fichier se présente de la façon suivante :

```
<?php

namespace src\tests\units;
require_once __DIR__ . '/../../Programmez.php';
use \atoum;

class Programmez extends atoum
{
  public function testhelp()
  {
     $verif = new \src\Programmez();
     $this
     ->string($verif->help('admin/help#programmez_atoum',''))
     ->isEqualTo('<h3>Votre magazine mensuel</h3><p>Exemple de module pour différents tests unitaires avec Atoum</p>')
     ;
  }
}
```

Tout d'abord, vous déclarez les namespaces nécessaires, le chemin du fichier et la connexion avec Atoum. Ensuite chaque fonctionnalité commence par 'test' suivie du nom de votre fonction. Enfin à l'intérieur, vous appelez la classe pour tester ici si la valeur retourne une zone de texte, et, après, si elle est égale au texte. Pour exécuter votre test, vous effectuez l'opération suivante en ligne de commande :

```
$ php mageekguy.atoum.phar -d sites/all/modules/programmez_atoum/tests/
```

Le framework s'occupe de faire le reste et vous obtenez le résultat suivant : Fig.1. Comme le montre l'image le test s'est correctement déroulé. Si le résultat est différent de la lettre S, comme un E, cela signifie une erreur. Enfin, le framework Atoum propose une documentation en Français, très précise pour l'utiliser dans vos projets.
http://docs.atoum.org/fr/latest/index.html

Conclusion

La roadmap des tests unitaires pour Drupal se décompose de la manière suivante : SimpleTest jouera son rôle jusqu'à Drupal 8 avec PHP UNIT pour laisser la place à celui-ci dans Drupal 9. Mais comme vous le savez le Web évolue et il faudra aussi compter sur la présence d'ATOUM comme une autre solution au niveau des tests unitaires.
Enfin, quel que soit le framework de tests unitaires que vous utilisez tous les jours dans vos différents développements, vous pouvez aussi les porter sur ce CMS.

JavaScript pour les Jedi, épisode II : l'attaque des closures

Dans ce second épisode consacré à l'apprentissage des fondamentaux de JavaScript pour les Jedi, nous allons aborder un autre aspect offert par ce langage : les fermetures (ou closures en anglais). Nous allons d'abord tenter de définir et comprendre les fermetures, ensuite nous verrons comment ces dernières facilitent nettement le développement JavaScript, en les exploitant pour résoudre des problèmes courants liés à la portée et aux portées des fonctions.

Wassim Chegham,
JavaScript ninja et consultant en technologies Web chez **SII Paris**

Petite introduction

Etroitement liées aux fonctions dont nous avons longuement discuté lors du premier épisode [1], les fermetures sont un des trois piliers de JavaScript. Pour rappel, ces trois piliers sont : les objets, les fonctions et les fermetures.

Historiquement, les fermetures étaient exclusivement réservées aux langages fonctionnels "purs" (par exemple Haskell), et c'est donc très encourageant de les avoir dans un langage grand public comme JavaScript. Ne soyez pas très surpris de voir les fermetures dans la plupart des librairies et frameworks JavaScript ; dites-vous que si les JavaScript Jedis sont très friands des fermetures, c'est parce que ces dernières permettent de simplifier de manière drastique des opérations très complexes.

Tentons donc de définir ce que sont les fermetures.

Comment fonctionnent les fermetures

Si nous devions définir les fermetures en une phrase : une fermeture est une fonction possédant des propriétés particulières, qui lui permettent d'accéder et de manipuler des variables se trouvant en dehors de la portée créée par cette fonction.

C'est assez clair ? Disons qu'une fermeture permet à une fonction "foo" d'accéder à toutes les variables et fonctions qui sont dans le contexte de déclaration (et non d'invocation) de cette fonction "foo".

Prenons un simple exemple :

```
var jedi = "JavaScript";
function foo(){
  console.log(jedi); // JavaScript
}
foo();
```

Dans cet exemple, nous avons déclaré une variable "jedi", et une fonction "foo" dans le même contexte — dans ce cas le contexte global du "window". Lorsque nous exécutons la fonction "foo", celle-ci a bien accès à la variable "jedi". Je parie que vous avez écrit ce genre de code une centaine de fois sans vous rendre compte que vous étiez en train de manipuler des fermetures ! Si vous pensez que cet exemple est trop simple, c'est sûrement parce que vous avez remarqué que la variable et la fonction sont déclarées dans le contexte global qui, tant que la page est chargée, est toujours accessible et ne change pas. Prenons un exemple un peu plus intéressant.

```
var luke = "luke";
var jedi;
function foo(){
  var vador = "je suis ton père";
  function bar(){
    console.log(luke, vador);
  }
  jedi = bar;
}
foo();
jedi();
```

Analysons le comportement de la fonction "bar" car il est plus intéressant. Nous exécutons la sous-fonction "bar" en différé, après l'invocation de la fonction "foo" , via la copie de la référence de "bar" vers "jedi". Remarquez que lorsque cette sous-fonction est exécutée, le contexte créé par cette dernière n'est plus disponible, ce qui voudrait dire que la variable "vador" n'est plus accessible. N'est-ce pas ?

Sauriez-vous me dire donc quel serait le résultat de "console.log(luke, vador);" ?

Pensez-vous que la réponse est "luke undefined" ? Si oui, alors vous serez surpris si je vous dis que non. Le résultat est bel est bien "luke je suis ton père". Quelle magie a donc permis à la variable "vador" d'être toujours accessible, même après la finalisation du contexte créé par la fonction "foo" ? La réponse est bien sûr, les fermetures.

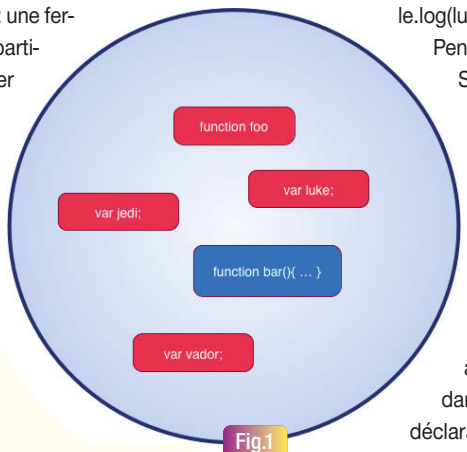


Fig.1

Les fermetures créent donc une sorte de "bulle" avec toutes les variables et fonctions — ainsi que la liste de leurs arguments — qui sont dans le contexte de la fonction au moment de sa déclaration, ainsi cette dernière aura tout ce dont elle aura besoin lors de son invocation. Cela permet donc de sécuriser ces variables et fonctions en leur évitant d'être détruites par le ramasse miette Fig.1.

A noter tout de même que cette bulle ou structure n'est pas un objet JavaScript auquel on peut accéder ou que l'on peut inspecter ou déboguer aussi facilement.

Cependant, Google Chrome intègre une toute petite fonctionnalité dans le Chrome DevTools permettant l'inspection de ces fermetures (dans le sous onglet Closure), comme indiqué dans la capture d'écran : Fig.2.

L'utilisation des fermetures présente tout de même des inconvénients : il faut bien stocker toutes ces informations en mémoire. Rappelez-vous que chaque fonction ayant accès à des informations via des fermetures, doit vivre avec cette bulle que l'on pourrait qualifier de "boulet". Toutes ses informations doivent être mise en mémoire et y rester durant toute la vie de

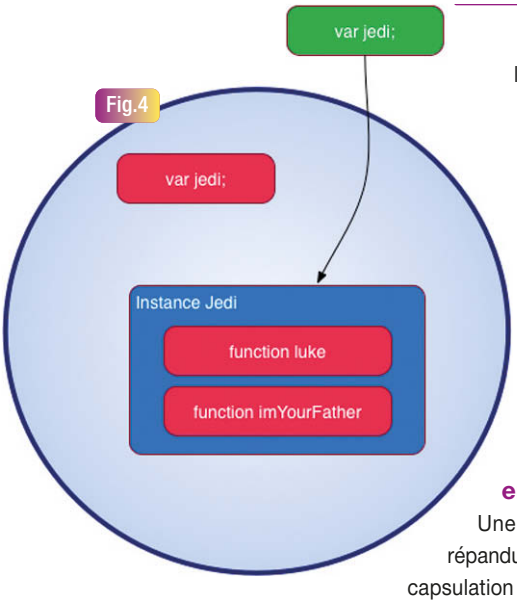


Fig.4

la fonction. Un conseil donc : utilisez les fermetures avec modération, et uniquement là où il y en a besoin !

Cas d'usage

Voici une liste des différents cas d'usage illustrant l'utilisation des fermetures...

Variables privées en encapsulation

Une des utilisations les plus répandues des fermeture est l'encapsulation d'information en tant que "variables privées", pour limiter la portée de ces variables. La Programmation Orientée Objet en JavaScript ne permet pas d'avoir des variables privées : des propriétés d'un objet non accessibles depuis l'extérieur. Mais en utilisant les fermetures, nous pouvons reproduire ce comportement. Voyons cela en code.

```
function Jedi(){
  var jedi = "";
  this.luke = function(){
    jedi = "Luke";
    return this;
  }
  this.imYourFather = function(){
    return jedi + ", Je suis ton père";
  };
};
var jedi = new Jedi();
jedi.luke().imYourFather(); // Luke, Je suis ton père
```

Dans cet exemple, nous avons défini une variable locale "jedi" dans le constructeur "Jedi". Comme nous l'avons vu dans le premier épisode, JavaScript limite la portée des variables aux fonctions. Afin de pouvoir accéder à cette variable depuis l'extérieur, nous avons défini deux méthodes permettant de modifier et lire cette variable.
Après invocation du constructeur, nous invoquons les deux méthodes, et, en résultat, nous avons bien le comportement attendu Fig.3.

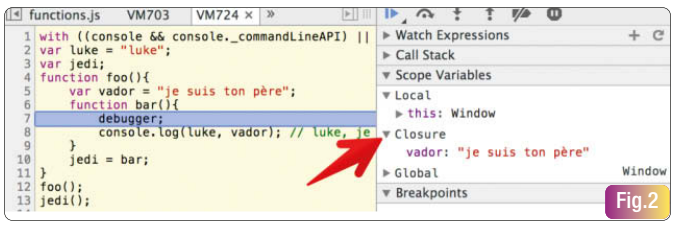


Fig.2

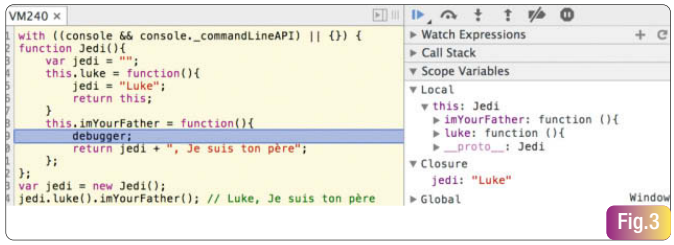


Fig.3

Nous avons donc encapsulé la variable en limitant sa portée et cela grâce aux fermetures. Voici ce que cela donne avec notre schéma de bulle : Fig.4.
Ceci était donc un simple et rapide aperçu de ce que l'on peut faire avec les fermetures et l'orienté objet. Nous explorerons en détail le monde du JavaScript Orienté Objet dans un prochain épisode.

Fonction de callback et timers

Un autre cas d'usage des fermetures est l'utilisation des fonctions de callback et des timers. Dans ces deux cas, une fonction sera appelée de manière asynchrone à un moment donné, et dans laquelle nous avons — souvent — besoin d'accéder à des informations se trouvant en dehors. Prenons l'exemple suivant :

```
<div id="content"></div>
<button id="load">Vador says...</button>
<script>
//
  var content = document.querySelector('#content');
  document.querySelector('#load').addEventListener('click', function(){
    content.innerHTML = 'Luke, ...';
    var xhr = new XMLHttpRequest();
    xhr.open('GET', '/api/users/luke', false);
    xhr.onload = function(data){
      // data.response : I'm your father!
      content.innerHTML = data.response;
    };
    xhr.send();
  });
//
</script>
```

Ici, nous avons une DIV dans laquelle nous venons insérer du texte chargé depuis un serveur distant, via AJAX. Pour cela, nous avons référencé la DIV en dehors de la fonction de callback du click. Ainsi, lorsque le click est traité par le navigateur et que la fonction est invoquée, elle a accès à la variable content, référençant la DIV.
La plupart des développeurs JavaScript sont familiers avec ce type de code. Et pourtant, si JavaScript nous permet d'écrire ce genre de code, c'est bien grâce aux fermetures.
Passons maintenant à un autre exemple plus intéressant, faisant cette fois-ci intervenir des "timers". Les "timers" sont souvent utilisés par les librairies JavaScript pour réaliser des effets ou des animations.

```
<div id="content"> □ </div>
  <button id="animateRight">Animate Right</button>
<script>
//
  var animateRight = function(){
    var tick = 0;
    var content = document.querySelector('#content');
    var timer = setInterval(function(){
      if(tick < 200){
        content.style.left = tick + 'px';
        tick += 1;
      }
      else {
        clearInterval(timer);
      }
    }, 15);
```

```
  };
  var click = function(){
    animateRight();
  };
  document.querySelector('#animateRight').addEventListener('click', click);
//
</script>
```

Ce qui est intéressant dans cet exemple, c'est l'utilisation d'une seule fonction anonyme dans "setInterval" pour réaliser l'animation ; cette fonction de callback a accès aux trois variables : "content", la référence vers l'élément DOM, le compteur "tick", et la référence "timer" vers la fonction du "timer". Ces trois variables contrôlant l'état de l'animation ne doivent pas être mises dans l'espace global. La raison est simple : si vous tentez d'animer plus d'un élément DOM, vous allez devoir maintenir N groupes de ces trois variables, soit un groupe par animation. Dit autrement, en n'ayant que trois variables globales pour les N animations, je vous laisse donc imaginer les dégâts.

Grâce aux fermetures, nous pouvons déclarer ces variables au sein de la fonction "animateRight", et nous pouvons compter sur les fermetures pour les rendre accessibles aux invocations des fonctions anonymes de chaque animation. Nous pouvons illustrer cela comme suit : Fig.5.

Chaque animation reçoit donc une encapsulation dynamique de l'état de son contexte d'invocation. Par dynamique, je veux dire que la fonction peut non seulement accéder à ces variables (qui existent dans le contexte d'invocation) mais également les modifier, et cela pendant toute la durée de vie de la fermeture. Et ce n'est pas tout.

En plus d'accéder et de modifier les variables du contexte d'invocation, il est possible de forcer un contexte en particulier grâce à la fonction "Function.prototype.bind()", comme discuté dans l'épisode 1 sur les fonctions [1].

Fonction partielle

L'application partielle d'une fonction est une technique très intéressante. Cela consiste à invoquer une fonction qui, une fois exécutée, retourne à son tour une autre fonction, destinée à être exécutée ultérieurement.

Plus simplement dit, on exécute une fonction qui fait une partie des traitements puis retourne une fonction qui, quand elle sera exécutée, fera la suite des traitements. Plus généralement, le rôle de la première fonction est de configurer/préparer les paramètres de la seconde fonction retournée. En mathématiques [2], cette technique est appelée "currying" et nous pouvons retrouver ce concept dans la plupart des langages "fonctionnels" (Haskell, Scheme, Scala, Python...).

Voici un exemple :

```
function curry(fn) {
  var restArgs = Array.prototype.slice.call(arguments, 1);
  return function() {
    var arg = Array.prototype.slice.call(arguments);
    return fn.apply(null, arg.concat(restArgs));
  };
};
```

Cet exemple est simple mais il illustre très bien l'utilisation des fonctions partielles. Explications...

[1] Le concept des fonctions a été publié dans le numéro 183 de programmez.
[2] http://bit.ly/MathCurrying
[3] http://bit.ly/JavaScriptDesignPatterns

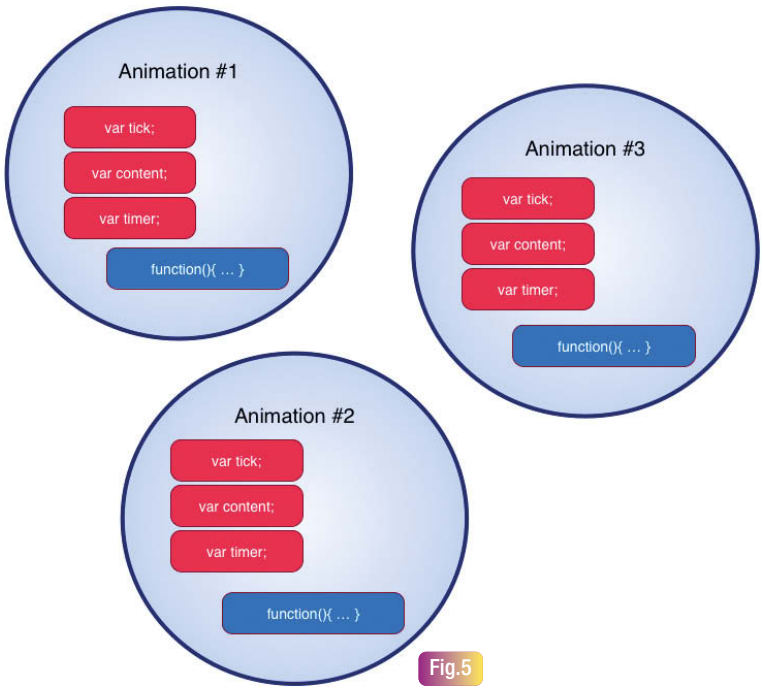


Fig.5

Nous avons créé une fonction "curry()" qui prend en paramètre une fonction (en première position) et une suite de paramètres (le reste des paramètres). La fonction "curry()", accède à la liste des paramètres (en excluant le premier, le paramètre "fn" qui a été nommé explicitement) puis retourne une fonction qui va être exécutée ultérieurement.

Cette fonction retournée a pour mission d'invoquer la fonction "fn" — via un apply() — qui a été passé à la fonction "curry()" en lui fournissant la liste des paramètres précédemment renseignés. Cette dernière étape est rendue possible grâce aux fermetures.

Un exemple d'utilisation de cette fonction "curry()" pourrait être le suivant :

```
var delay = curry(setTimeout, 1000);
delay(function() {
  alert("POWNED!!");
});
```

Grâce à la fonction "curry()" nous avons créé une fonction "delay()" que nous avons configurée comme étant une référence vers la fonction "setTimeout" avec un délai d'une seconde. Grosso modo, nous avons maintenant une fonction qui nous permet d'exécuter une action avec une seconde de délai. Très utile, non ? :)

IIFE (Immediate Invocation Function Expression)

Les fermetures sont également utilisées au sein de nombreux patrons de conception [3] très présents en JavaScript.

Parmi ces patrons, nous pouvons citer le patron IIFE, en français cela donne "les fonctions auto-invoquées". Ce patron est principalement utilisé pour encapsuler des objets ou du comportement dans des modules JavaScript (on parle également de Module Pattern). Etudions d'abord les bases des IIFE qu'on pourrait réduire à cette ligne de code :

```
(function(){})()
```

Analysons cette petite ligne de code. Tout d'abord, ignorons le contenu du premier ensemble de parenthèses, pour ne garder que ceci :

```
(...)()
```

Si vous vous rappelez ce que nous avons dit lors du premier épisode, JavaScript traite les fonctions comme des fonctions de première ordre, c'est-à-dire que nous pouvons référencer une fonction depuis une variable, et nous pouvons invoquer cette variable, comme ceci :

```
var luke = function(){};
var jedi = luke();
```

En JavaScript, les fonctions sont considérées comme étant des expressions, et nous pouvons exécuter ces expressions en utilisant l'opérateur (). Mais là où les choses peuvent paraître un peu confuses, c'est que les parenthèses peuvent également être utilisées pour délimiter les expressions. C'est-à-dire que le code suivant est tout à fait valide :

```
var luke = function(){};
var jedi = (luke)();
```

Du coup, nous pouvons omettre la déclaration de la variable, et passer par une fonction anonyme :

```
var jedi = (function(){})();
Ajoutons un peu de code :
var jedi = (function(what){

 var say = "I'm your "+what+"!";

 return (function(){
  return "Luke, "+say;
 })();

})("father");
console.log(jedi); // "Luke, I'm your father!"
```

Dans l'exemple ci-dessus, nous avons imbriqué deux IIFE, ajouté quelques fermetures, et fourni un paramètre à la première IIFE. Le résultat de cet exemple est une expression qui produit ceci, elle :

- Crée une première instance de fonction,
- Exécute la fonction,
- Crée une seconde instance de fonction,
- Exécute cette seconde fonction et retourne le résultat,
- Se débarrasse de cette seconde fonction (car elle n'est référencée nulle part),
- Retourne le résultat,
- Se débarrasse de la première fonction (car elle n'est référencée nulle part non plus).

De plus, grâce aux fermetures, la seconde fonction accède à tout le contexte de la première fonction, ainsi qu'à la liste des paramètres.

On se rend compte du coup, que cette construction assez simple peut s'avérer très puissante et très utile, dans certains cas. Voyons un autre exemple.

Il arrive des fois où vous devez attacher des évènements à plusieurs éléments du DOM. Bon, normalement si tel est le cas, je vous recommande de passer par la délégation d'événements, qui est un patron de conception adapté à ce cas d'usage ; mais ce n'est pas le sujet de cet article.

Vous seriez surement tenté de faire quelque chose du style :

```
var elements = document.querySelectorAll('button');
var len = elements.length; // 8 par exemple
for(var index=0; index<len; index+=1){
 elements[index].addEventListener('click', function(){
  alert(index+1)
 }, false);
}
```

Dans l'exemple précédent, imaginez que vous avez 8 boutons dans votre page. Vous vous attendez donc à voir un message s'afficher avec l'index de l'élément en question. En cliquant sur le bouton #3, vous voulez voir l'index 3 s'afficher. Mais au lieu de cela, vous avez toujours le dernier index 9.

Nous rencontrons ici un problème typique lié aux fermetures et aux boucles. Ce problème concerne dans notre cas, la variable passée à la fermeture (la variable "index"). Le fait est que cette variable est mise à jour à chaque itération.

Aussi, chaque fonction — ou "handler" — du "addEventListener" garde une référence vers chaque variable "index" passée par la fermeture. Ce qui signifie donc que chaque "handler" affichera toujours la dernière valeur stockée dans la variable "index". Beaucoup de développeurs débutants en JavaScript tombent dans ce piège.

Evitez donc de déclarer des fonctions dans les boucles, si vous n'avez pas le choix, passez par une IIFE.

Maintenant que je vous ai présenté le problème, parlons de la solution. Vous allez être étonné d'apprendre que pour résoudre ce problème introduit à la base par la fermeture, nous allons avoir besoin d'une autre fermeture (plus une IIFE). Comme on le dit, on va combattre le feu par le feu :)

```
var elements = document.querySelectorAll('button');
var len = elements.length; // 8 par exemple
for(var index=0; index<len; index+=1){
 (function(n){
  elements[n].addEventListener('click', function(){
   alert(n+1)
  }, false);
 })(index);
}
```

En utilisant une IIFE en tant que corps de la boucle, et en lui passant l'index courant, en tant que paramètre, nous créons un contexte isolé dans lequel chaque variable "index" est différente.

Voilà donc comment, grâce aux fermetures et IIFE, vous pouvez contrôler le contexte des variables et des valeurs.

Résumé

Dans ce second épisode, nous avons appris et surtout compris comment les fermetures (ou "closures"), qui sont d'ailleurs un des concepts majeurs de la programmation fonctionnelle, sont implémentées en JavaScript. J'espère que grâce à cet article vous aurez suffisamment de bagage pour votre quête ultime pour devenir un Jedi en JavaScript.

Dans notre prochain épisode, nous nous attaquerons à un autre concept de JavaScript : la programmation Orientée Objet avec les Prototypes. J'insiste sur le terme "prototypes" car il ne faut pas le confondre avec la POO à base de "classes".

Ce concept exploite le premier aspect que nous avons découvert, celui des fonctions du premier ordre, et le second concept, les fermetures. En tant que future Jedi en JavaScript, maîtriser la POO en JavaScript est une étape cruciale dans votre quête, et il est de votre devoir d'accomplir votre mission.

Que la force soit avec vous...

Conception d'interface tactile
2e partie : la souplesse du processus

Dans une première partie nous avons abordé les méthodologies liées aux applications qui nécessitent des équipes pluridisciplinaires. Les interfaces tactiles, mais plus largement celles que l'on appelle les interfaces naturelles, ont plus que tout besoin d'une phase de conception centrée utilisateurs.

Johanna Rowe Calvi
Designer d'interaction et d'expérience numérique chez Expertime
MVP Kinect for Windows

Nicolas Calvi
Consultant Informatique, Expert NUI chez Expertime
MVP Kinect for Windows

La complexité pourrait s'arrêter là si l'on ne prenait pas en compte les nouvelles demandes qui émergent. Comme une tendance grandissante, ou un problème auquel on prête enfin attention : la conception multiplateforme. Jusqu'il y a quelques années, le choix devait se faire entre application native pour chaque plateforme et site Web responsive. La problématique de la première étant le coût et celle du deuxième choix la qualité d'expérience. La question revenait souvent de savoir pourquoi ne pas faire juste un site Web « responsive ». La réponse est simple : les utilisateurs téléchargent des applications pour avoir un accès rapide au contenu qui les intéresse, que ce contenu soit de la vente, du réseau social, de l'administratif ou de l'information. Si vous faite partie de l'écran d'accueil de leur smartphone ou de leur tablette, vous êtes toujours présent dans leur quotidien.

Tout un nouveau panel de possibilités

Mais maintenant différentes solutions techniques sont disponibles, chacune avec leurs avantages et leurs inconvénients. Il fallait que des solutions émergent pour que les applications deviennent accessibles financièrement pour certains. Il fallait que les applications continuent à pouvoir être conçues de manière adaptée, et d'une haute qualité d'expérience, peu importe le prix pour d'autres. C'est sans nul doute une vision simplifiée mais c'est ce que nous, les consultants, comprenons de ce séisme qui se met en place dans le monde du développement. Ce qui est agréable pour les designers c'est de pouvoir proposer des solutions viables et réalisables pour chacun des clients selon leur besoin et leur budget. Ces solutions vous les connaissez forcément si vous êtes un lecteur habituel de Programmez! Ou si vous êtes un tant soit peu à l'écoute du monde des interfaces mobiles. Maintenant vous avez le choix entre des applications natives (à l'ancienne) où chacune est conçue et développée de A à Z pour chaque plateforme. Des applications conçues avec un Framework multiplateforme (ex : Xamarin) qui permettent de mutualiser (entre 50% et 80%) les développements et d'avoir un code écrit avec un même langage et des principes généraux communs. Enfin des applications Web (ex : PhoneGap) qui permettent d'avoir une réutilisation du code proche de 100% avec une conception simplifiée car elle n'est pas forcément liée complétement à la plateforme. Rentrons un peu plus dans le détail,

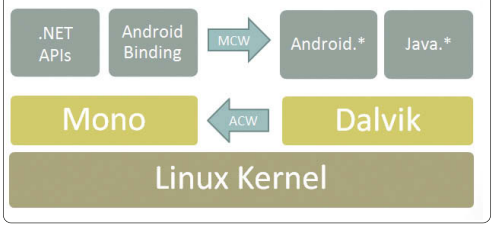
Le pur natif

Ce choix est actuellement un des plus rares du fait de son coût. Mais contrairement à ce que l'on peut s'imaginer, il reste une option sur laquelle on ne peut transiger pour certaines entreprises. La qualité d'une expérience différenciante sur chaque support est indispensable, quitte à étaler les développements sur plusieurs années. Cette solution est un peu la Rolls Royce des solutions applicatives. Lorsque l'on développe sur plusieurs plateformes (par exemple, Android, iOS et Windows Phone), devoir faire des applications natives permet d'avoir une expérience optimisée, mais surtout l'accès à toutes les ressources du périphérique. Cela demande de connaître chaque SDK de développement, et le ou les langages qui sont utilisés pour créer les applications. Pour une entreprise, il est parfois difficile d'avoir toutes les compétences en interne, et il est souvent coûteux de payer des prestataires pour faire toutes les déclinaisons natives. Mais cela reste la méthode la plus efficace si votre application demande des algorithmes pointus (traitement d'image, accès à la reconnaissance vocale, rendu 3D avancé). C'est donc un choix technique et de compétence, car si l'application n'est pas pointue, un développement avec un Framework intermédiaire est souvent la bonne solution.

Le Framework multiplateforme type Xamarin

Ce choix permet de garantir une expérience adaptée à chaque support, et permet de

Fig.1



Architecture Xamarin pour Android

concevoir en respectant les usages des utilisateurs de chaque plateforme. Le travail des designers d'interaction ainsi que celui des ergonomes est de rendre/valider que l'usage d'une solution est intuitif. Pour que la solution soit intuitive, elle doit prendre en compte les utilisateurs. Bien que certaines entreprises peuvent se permettre d'imposer une expérience unique tant leurs solutions sont consommées de manière presque addictive sur l'ensemble des supports (ex : Facebook), la plupart ne sont pas dans cette situation. La majorité des utilisateurs ne trouvant pas l'application intuitive et pertinente ne l'ouvriront sans doute plus jamais. Cette solution de natif multiplateforme est donc un bon compromis.
Le point fort de cette méthode, c'est qu'il est possible de factoriser entre 50% et 80% son code métier via des patterns de développement adaptés (comme MVVM Cross par exemple). De cette façon, il n'y a que les couches « visuelles » à développer sur chaque support. De plus, on utilise un seul et même langage sur l'ensemble des plateformes (C#/Mono dans le cas de Xamarin) ce qui permet de ne pas se perdre dans les spécificités de chaque plateforme. Cependant, cette solution a un coût. Généralement les applications de ce type sont ralenties par toutes les couches liées aux machines virtuelles. Dans la majorité des applications, cela ne sera presque pas visible. Il faut quand même le garder à l'esprit, surtout si votre application doit tourner sur des périphériques avec des performances réduites. Un autre avantage de cette solution, c'est que l'on a toujours accès de façon « semi-native » aux APIs spécifiques de chaque plateforme. Dans le cadre de Xamarin par exemple, on peut créer une « Portable Library » qui sera le code commun aux applications, ensuite pour la partie Windows Phone on restera en natif. Pour iOS et

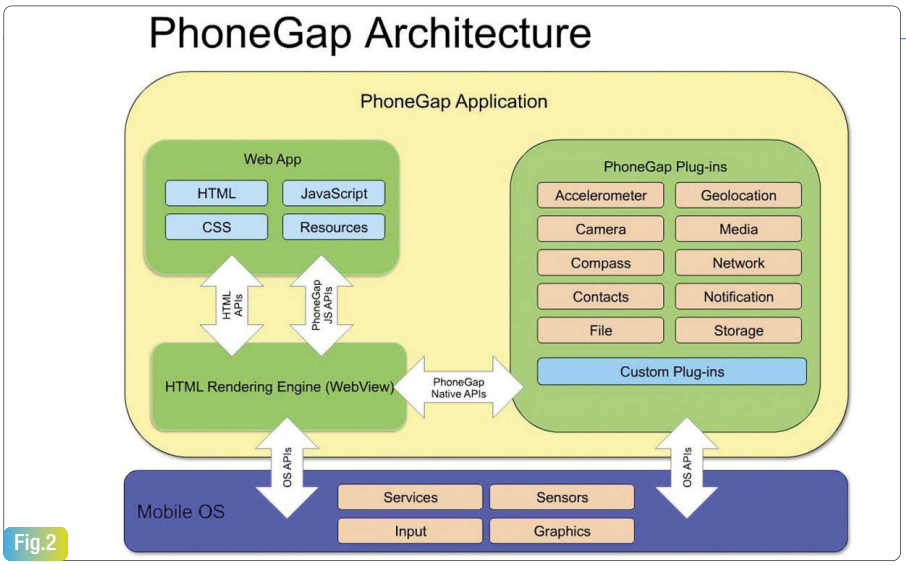


Fig.2

Architecture PhoneGap

Android, on passera par des projets Mono étendus avec des espaces de noms propres à chaque plateforme, qui feront le pont entre les APIs natives et votre code C# Fig.1.

Les applications en technologies Web

A l'heure actuelle, c'est le choix le moins glamour mais qui a le mérite d'être peu coûteux. En général lors des échanges autour de la conception de l'expérience utilisateur, les discussions sont plutôt compliquées entre qualité d'expérience et coût. C'est un choix à éviter lorsqu'on a des concurrents déjà bien présents, ou si l'entreprise souhaite se démarquer avec une qualité visuelle et d'expérience poussée. Soyons clair, il est possible d'avoir (presque) la même qualité que les applications conçues en Xamarin, mais alors l'avantage financier qu'apportait cette option n'est plus, car le travail pour chaque plateforme annule celui-ci. Cette solution repose sur le développement d'un « site Web » qui est embarqué dans une application native de la plateforme. Ce site est ensuite exécuté en local et tire parti de la puissance d'HTML 5 et CSS 3. Les Frameworks les plus utilisés pour cette technique sont PhoneGap et Titanium.
Il est possible d'accéder à des fonctions natives via des « plugins »;pour les plus courants il est possible de les trouver facilement sur la toile, sinon il va falloir les développer vous-même, il n'y a rien de prémâché de ce côté-là. De manière générale, ces Frameworks sont utilisés comme un site Web responsive, qui s'adapte à la plateforme, nous avons donc toujours la même expérience. Il est cependant possible de détecter le périphérique et de faire des expériences différentes selon celui-ci, mais cela enlève tout l'intérêt de ce type de technologie. Pour résumer, il y a trois solutions techniques entre lesquelles faire votre choix. Mais ce choix dépend de nombreux paramètres, de la qualité d'expérience aux aspects financiers en passant par la réalité du terrain et de la formation des équipes qui vont développer les applications. Il est par exemple évident que si les équipes techniques en interne d'une société sont toutes très performantes en HTML et qu'il n'est pas question d'externaliser les développements alors le choix est rapidement fait Fig.2.

Les impacts sur la phase de conception

Eh bien amis designers d'interactions, designers graphiques, préparez-vous à une nouvelle étape de votre vie de créatifs. Que l'on parle de la phase de conception, de l'expérience numérique avec les aspects de fonctionnalités et de navigation ou de conception graphique, les choix techniques ci-dessus vous impactent directement. Pour les designers d'interaction qui conçoivent l'expérience utilisateur l'approche sera différente pour chaque solution. Si l'on prend le choix d'une expérience adaptée à chaque support, alors les ateliers utilisateurs/client seront réalisés en trois étapes distinctes (chaque OS). La conception d'une expérience adaptée à chaque support signifie que les fonctionnalités et la navigation seront différentes selon les supports. Par exemple, sur Windows Phone les possibilités de deep linking (raccourcis) ou de tuiles dynamiques pourront faire partie de fonctionnalités spécifiques. Les wireframes seront aussi réalisés 3 fois et chacun validé distinctement. Pour les designers graphiques, il s'agira dans certains cas d'imaginer une création graphique unique qui sera répercutée sur l'ensemble des supports (à quelques détails près). Dans d'autres cas, il s'agira de proposer une création graphique assez proche pour limiter les coût de développement, mais tout en restant adapté aux attentes des utilisateurs iOS, Android et Windows. Et pour le dernier cas, il s'agira de concevoir une ambiance graphique unique pour chaque support, c'est cette dernière solution qui vous permettra le plus de liberté. Nous avons ici parlé uniquement de l'aspect smartphone. Si maintenant nous ajoutons l'aspect tablette alors la phase de conception en sera aussi modifiée. Il s'agira plutôt de faire des ateliers par OS (tablette/smartphone) dans le cas où vous souhaitez ne pas avoir de différences entre fonctionnalités des deux types de périphériques. Ou des ateliers par type de périphérique (smartphone puis tablette ou inversement) si vous souhaitez tirer parti à la fois des possibilités de chaque type de périphérique, ou prendre en compte la différence d'usage que les utilisateurs ont entre une tablette et un smartphone. Dans ce dernier cas, les fonctionnalités peuvent être assez différentes entre smartphone et tablette car en phase avec des usages réellement différents. Il reste cependant important avant tout de bien concevoir son interface tactile en fonction du support et de sa cible. Il faut bien faire attention à ce que chaque zone d'interaction puisse être déclenchée en un seul Tap. Il faut aussi utiliser les contrôles que les utilisateurs ont l'habitude d'utiliser (notamment vrai sur mobile). Et si l'expérience est singulière, bien tester tous les cas de figure, bien comprendre que tout le monde ne perçoit pas une interface tactile comme vous: il faut être intuitif et donc naturel.

Et pour finir

La mobilité est un vaste sujet, mais la mobilité ce ne sont pas uniquement les smartphones et les tablettes. Les solutions applicatives qu'apportent les développeurs, designers, graphistes, ergonomes sont en réponse aux demandes qui émergent du terrain en ce qui concerne les applications BtoB et principalement en réponse avec les tendances technologiques et marketing en ce qui concerne les applications BtoC. Dans un autre angle, le nombre d'applications pour le bien-être des employés ne cesse de croître... Avec toute cette digitalisation de notre quotidien d'employé, de salarié ou de client, un grand nombre d'innovations nous attendent. Lunettes connectées avec le petit nouveau : HoloLens, capteurs de mouvements comme Kinect ou Leap Motion, grandes surfaces tactiles collaboratives comme Surface Hub ou encore l'Eye tracking comme Tobbii et tant d'autres innovations à venir. Les interfaces et les interactions entre l'Homme et les technologies n'auront jamais fini de muter. Nos métiers continueront à évoluer et il ne faut pas rater le train de la mutation qui commence. Soyons souples et ne cessons pas d'innover et de créer avec toujours en tête le bien être des utilisateurs.

Timeline : 1981
Objet : le BBC d'Acorn, le plus bel ordinateur du monde

Le BBC Micro d'Acorn et son successeur, l'Archimedes, sont certainement les ordinateurs les plus puissants de leur époque. Hélas, ils n'ont jamais réussi à sortir du système éducatif britannique.

Les principaux salariés d'Acorn réunis en 2012 : Sophie Wilson (à gauche), Stephen Furber (troisième en partant de la gauche), ainsi que les fondateurs Hermann Hauser et Chris Curry (à droite).

Yann Serra

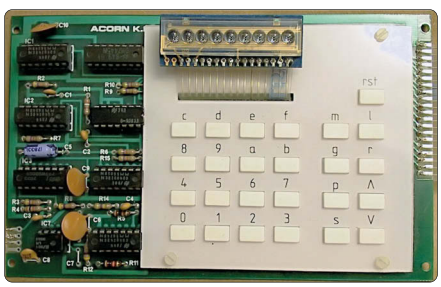
En 1978 à Cambridge, Christopher Curry et Clive Sinclair, les deux têtes pensantes de la petite société Science of Cambridge (SoC), rêvent d'étendre leur business de machines à calculer vers la fabrication de micro-ordinateurs. A l'époque, l'américain Apple vient de lancer son Apple II et le concept fait grand bruit. Problème, Curry et Sinclair ne sont pas d'accord : le premier veut développer une carte mère MK14 multifonction avec une puce 8 bits de National Semiconductor, tandis que le second préfère plancher sur un vrai micro-ordinateur, le New-Brain, basé sur le processeur Z80 de Zilog. Fâché, Curry claque la porte de SoC et part fonder une société de conseil en électronique, Cambridge Processor Unit (CPU). De manière assez amusante, on notera que les acronymes SoC et CPU serviront plus tard à signifier, respectivement, System on a Chip et Central Processing Unit, mais cela n'aura absolument rien à voir avec les entreprises de Curry et Sinclair.

Un connecteur réseau en 1980

Vendant des cartes mères MK14 aux fabricants de machines à sous le jour, Curry met au point la nuit un ordinateur pour rivaliser avec le New-Brain, sur la base cette fois-ci du processeur 6502 de MOS technology (concurrent direct du Z80). Pour ne pas mélanger la très sérieuse activité de conseil avec le hobby micro-informatique, on attribue au second une marque à part, qui sonne comme Apple. Curry choisit le nom Acorn, qui signifie en anglais le gland, le fruit du chêne. En 1980, SoC lance le ZX80 à 80 £, un petit ordinateur doté de 1 Ko de RAM, d'un processeur Z80 à 3,25 MHz et d'un mode texte N&B. Acorn répond avec un Atom à 120£. Son processeur ne fonctionne qu'à 1 MHz, mais la machine, dispose de 2 Ko de RAM et d'un affichage graphique (64x192 pixels en 4 couleurs ou 256x192 pixels en monochrome). De manière totalement avant-gardiste pour l'époque, on y trouve une connectique réseau ! Celle-ci, nommée Econet, est une invention maison, due à deux ingénieurs recrutés dans la faculté de mathématiques de Cambridge : Steve Furber et Sophie Wilson.

L'ordinateur officiel de l'école anglaise

Fort d'héberger sur son territoire des petits génies aussi valables que les américains, le parlement britannique s'enflamme rapidement sur le sujet de la démocratisation de l'informatique. Au point que le ministère de l'industrie finit par débloquer un budget pour financer à hauteur de 50% l'achat de micro-ordinateurs à toutes les écoles du Royaume-Uni. La chaîne de télévision BBC, en quête d'une image innovante, se positionne alors pour faire fabriquer les machines. Acorn remporte l'appel d'offres avec le Proton, à savoir un Atom désormais cadencé à 2 MHz, avec 16 Ko de RAM, une résolution de 160x256 en 8 couleurs, un contrôleur de disquettes et toujours sa connectique réseau Econet exclusive. Surprise : on y trouve aussi une autre invention totalement inédite pour l'époque : un slot « Tube » qui permet l'ajout d'un second processeur ! Le Proton sort en décembre 1981 sous le nom de BBC Micro. Six modèles se succèderont jusqu'en 1986. Les A, B, B+64 et B+128, où la quantité de RAM double à chaque fois. Puis le Master, comprenant un traitement de texte et un tableur en ROM ; il sera décliné en plusieurs versions, dont une turbo avec un processeur à 4 Mhz. Et enfin le Master Compact, livré avec un lecteur de disquettes 3,5 pouces et une interface graphique à la Macintosh. Au total, il se vendra plus d'un million et demi d'exemplaires du BBC Micro jusqu'en 1992.



La carte MK14.

L'archimedes

Fiasco sur les marchés traditionnels

Mais n'allons pas trop vite. En 1983, la marque Acorn engrange déjà 8,6 millions de Livres Sterling de bénéfices. Elle est si connue qu'on décide de faire disparaître la raison sociale CPU originale et de faire entrer Acorn en bourse. L'entreprise est immédiatement valorisée à 135 millions de Livres Sterling. Curry se retrouve à la tête d'une fortune de 64 millions de Livres.

Le succès est pourtant en demi-teinte. Depuis deux ans, l'essentiel des ventes de la machine se fait au travers des écoles anglaises. Les foyers préfèrent acheter les micro-ordinateurs moins chers de SoC, lequel s'est renommé entretemps Sinclair. En vedette depuis 1982, le Sinclair ZX Spectrum coûte moins de 200£. Pour rivaliser, Acorn conçoit l'Electron, un BBC Micro à prix cassé car 90% de l'électronique y est condensé dans une seule puce ULA. Problème, ce chipset est si compliqué à fabriquer qu'il retarde la sortie de l'Electron, laissant le ZX Spectrum verrouiller durablement ses parts de marché.

Seconde tentative infructueuse, lancer le BBC Micro sur le sol américain. Les écoles américaines se montrent d'abord intéressées par l'opportunité de renouveler le succès du programme éducatif britannique. Elles seraient prêtes à

investir 21 millions de dollars dans l'achat d'ordinateurs Acorn. Mais le projet échoue : on se rend compte au dernier moment que la logithèque du BBC, prévue pour afficher 256 lignes de pixels sur les écrans PAL européens, est incompatible avec les écrans NTSC américains qui ne reconnaissent que les modes graphiques en 200 lignes de pixels. En définitive, la seule réussite américaine du BBC sera de faire une apparition dans une salle de classe, lors d'une scène du film Supergirl.

Deux processeurs

Début 1984, Christopher Curry change son fusil d'épaule : adieu le marché des foyers, l'avenir du BBC se fera sur le segment professionnel, dans la lignée de l'IBM PC sorti en 1981. Il dévoile ainsi la gamme des Acorn Business Computer (ABC). Ces machines, qui intègrent leur unité centrale et un lecteur de disquettes dans le moniteur, sont en fait des BBC dotés d'un processeur supplémentaire inséré dans le fameux slot Tube. Il s'agit tantôt d'un Z80 pour exécuter le système CP/M 2.2 (modèle ABC100). Tantôt d'un processeur 16/32 bits 32016 de National Semiconductor, pour exécuter une sorte de faux Unix mono-utilisateur, le système Acorn Panos (modèle ABC210, alias Acorn Cambridge Workstation). Tantôt d'un Intel 80286, pour exécuter Concurrent DOS. Ce système est une version multi-utilisateur de CP/M pour PC, avec des extensions pour la rendre compatible MS-DOS. Tout comme l'Electron, les ABC mettront du temps à être fabriqués, et, en 1985, feront pâle figure face aux vrais PC en pleine démocratisation ou, pire, face au tout dernier Macintosh 100% graphique d'Apple.

L'invention de l'ARM

En fait, courant 1984, durant la mise au point des ABC, on prend déjà conscience chez Acorn que l'architecture du BBC ne fonctionne vraiment bien qu'avec son processeur d'origine, le 6502. Ainsi, un BBC doté d'un 6502 8 bits à 4 MHz est deux fois plus rapide qu'un ABC210 ou qu'un ABC300 pourtant pourvus d'un processeur 16/32 bits à 8 MHz. Ni le NS32016, ni l'Intel

L'Acorn Atom

Le BBC Micro

L'Electron

80286, ni même le Motorola 68000 ne parviennent à se synchroniser correctement sur le rythme des interruptions prévu pour le 6502 central. Problème, en l'état, le 6502 ne serait jamais assez puissant pour produire une machine qui rivalise avec le Macintosh.

Steve Furber et Sophie Wilson partent alors visiter Western Design Center, le fabricant américain de leurs 6502, pour connaître les projets d'évolution du processeur. Ils tombent des nues : ce fabricant n'est en réalité qu'une toute petite PME. Ils rentrent en Grande-Bretagne avec l'idée qu'ils doivent pouvoir eux aussi fabriquer leur propre microprocesseur. Sophie Wilson commence alors à imaginer le jeu d'instructions d'un 6502 32 bits. Pour dessiner le circuit de ce processeur, Steve Furber reprend les plans libres de droit du processeur SPARC, une puce RISC alors en cours de conception à l'université de Berkeley. Son nom sera l'ARM, pour Acorn Risc Machine. Les usines du prestataire VSLI produisent les premiers exemplaires de l'ARM1 en avril 1985. Pendant les deux années suivantes, il servira d'extension pour les développeurs du BBC.

Tué par les financiers

Entre le coût de développement de l'ARM, celui du nouveau BBC Master, le flop de l'Electron, le bide des ABC et les frais divers qu'il a fallu investir dans une campagne de pub américaine finalement inutile, Acorn se retrouve exsangue en 1985. Le 20 février, le fabricant italien Olivetti rachète 49% de ses parts. Puis 30% de plus en septembre. Chris Curry lâche les rênes de son entreprise. La volonté d'Olivetti semble de vider les caisses. Il n'y aura pas de campagne commerciale pour exporter le BBC Micro ailleurs en Europe, alors que c'est précisément le marché européen qui fait la fortune de Sinclair et du nouveau venu Amstrad. Olivetti laisse tout de même les ingénieurs continuer leurs travaux. Le BBC Master sort en 86. Et, en 87, Acorn lance l'Archimedes, le premier micro-ordinateur RISC 32 bits basé sur le processeur ARM. Il est conçu pour battre à plate couture les tous derniers Atari ST et Commodore Amiga 16 bits : il dispose lui aussi d'un système multitâche fenêtré et atteint 4 à 5 MIPS de performances contre 1 MIPS pour ses concurrents. Mais avec Olivetti aux commandes, l'Archimedes, n'aura jamais les moyens de sortir du territoire anglais. Ni même des écoles britanniques, d'ailleurs. Vendu avec un émulateur BBC, il y assurera la continuité de la gamme jusqu'en 1990. Jusqu'au moment où le système éducatif britannique opte majoritairement pour des Macintosh. Le département ARM, dont le processeur a séduit Apple pour construire une tablette tactile de poche (le Newton), mute en compagnie autonome en 1990. Celle-ci est devenue la plus puissante société de processeurs du monde pour appareils mobiles. L'Archimedes, lui, cesse d'être fabriqué en 1992. Et avec lui, toutes les gammes de BBC. La machine connaît une seconde vie pour une poignée de fans jusque dans les années 2000, sous le nom de RiscPC, chez différents petits cloneurs. Acorn se réoriente dans la construction de box pour TV en 1995, puis est démantelé en 1999 par ses investisseurs. ⊞

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 & rédacteur en chef : François Tonic
Ont collaboré à ce numéro : S. Saurel, Y. Serra
Secrétaire de rédaction : Olivier Pavie
Experts : V. Loquet, C. Pichaud, J-N. Durant, M. Nantern, Les Duchesses, A. Lakhal, S. Hertrich, S. Moallic, A. Crepet, K. Aresti, L. Probst, A. Vaché, Aranzazu San Juan Llano, D. Popin, A. Penel, M. Frappat, J. Antoine, J. Thiriet, A. Talantsy-Viyenne, B. Cornet, F. Allilaire, D. Petisme, H. Carnicelli, L. Ellerbach, S. Begelman, G. Bouveret, T. Ounnas, Y. Benoit, C. Villeneuve, W. Chegham, J. Rowe Calvi, N. Calvi

Crédits couverture : D.R., Oracle, Unity
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 : 1215 K 78366 - ISSN : 1627-0908

© NEFER-IT / Programmez, mars 2015
Toute reproduction intégrale ou partielle est interdite sans accord des auteurs et du directeur de la publication.

Si on disait ce qu'on pense pendant les codes reviews

Chaque semaine, de nouvelles aventures !

www.commitstrip.com

ikoula
HÉBERGEUR CLOUD

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

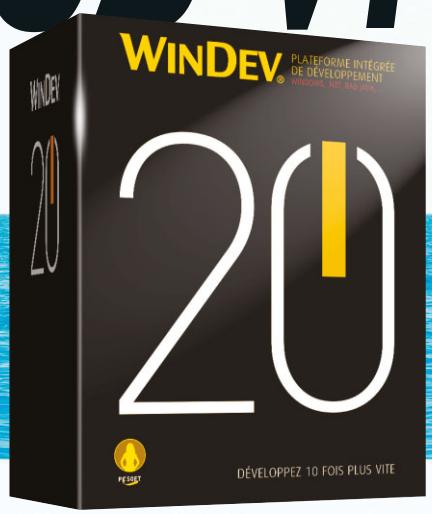
EX10

Cloud Hybride
Exchange
Lync
Sharepoint
Plateforme Collaborative

sales@ex10.biz
01 84 01 02 53
www.ex10.biz

WINDEV®

DÉVELOPPEZ 10 FOIS PLUS VITE



Elu
«Langage le plus productif du marché»

VERSION EXPRESS **GRATUITE**
Téléchargez-la !

php Java HTML5 iOS

Développez une seule fois, et recompilez pour chaque cible. Vos applications sont natives.

Tél province: **04.67.032.032**
Tél Paris: **01.48.01.48.88**

Fournisseur Officiel de la Préparation Olympique

www.pcsoft.fr
Des centaines de témoignages sur le site