

NOUVELLE
FORMULESpécial
Visual Studio 2013L'invasion
des APIHacker
sa caméra
GoProBilan des conférences
hacking 2013Métier
Qui est le
Data Scientist ?

Son code Android au régime

Le quotidien du designer



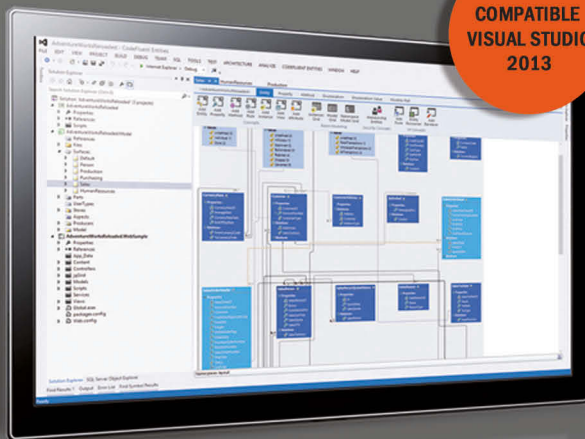
M 04319 - 171 - F: 5,95 € - RD

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



BESOIN D'AIDE?

CodeFluent Entities



ÉLÉMENTS APPLICATIFS

- . Localisation
- . Liaison aux données
- . Règles et validation
- . Concurrence d'accès
- . Sécurité
- . Cache
- . Gestion des BLOBS

ARCHITECTURES

- . SOA, Client intelligent
- . Client riche, RIA
- . Web, Webparts
- . Client/Serveur, N-Tier
- . Office, SharePoint
- . Clients mobiles
- . SaaS, Cloud

TECHNOLOGIES

- . .NET, C#, VB, Linq, Visual Studio
- . ASP.NET WebForms, MVC
- . Windows 8, WPF
- . WCF, JSON/REST, services mobiles
- . Excel, Access, SharePoint

BASES DE DONNÉES

- . Microsoft SQL Server
- . Oracle Database
- . MySQL
- . PostgreSQL
- . Microsoft SQL Azure

Téléchargez maintenant
www.softfluent.com/landings_cfe_programmez

Intégré à Visual Studio depuis 2008, CodeFluent Entities génère les fondations de votre application dans les règles de l'art et vous permet de vous concentrer sur les parties les plus intéressantes qui font votre valeur ajoutée.

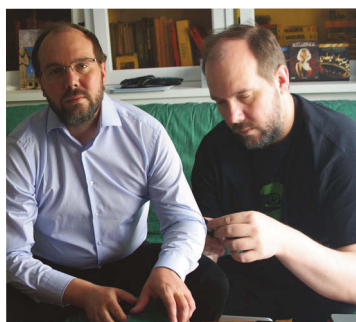
"J'ai récemment assisté à une formation d'une semaine sur Entity Framework mais CodeFluent Entities fournit bien plus de choses et est nettement plus facile à comprendre et à mettre en œuvre." *

Peter Stanford - Artefaction - Australie

* Source : <http://visualstudiogallery.msdn.microsoft.com/B6299BBF-1EF1-436D-B618-66E8C16AB410>



info@softfluent.com - www.softfluent.com - 01.75.60.04.45
 SoftFluent France - 3 rue de la Renaissance - 92160 Antony



La Force est dans l'API

C'est le buzz de 2014 : API. Le développeur connaît cette drôle de bestiole depuis longtemps qui peut ressembler à un gentil Gremlin... ou à un Alien contrarié (déjà qu'un Alien non-contrarié n'est pas ton ami, je vous laisse deviner la suite). Tout le monde en veut, tout le monde en propose. Tu veux te connecter à mon service, voici mon API. Tout le monde peut créer rapidement une (Web)API pour exposer un service.

Comme vous le verrez dans notre dossier spécial, avec un peu de méthode, les bons outils, et une idée précise, créer une API n'est pas l'exercice le plus difficile. Mais comme toujours il y a API et API. Celle qui sera propre sur elle, bien formée, rigoureuse dans son fonctionnement. Et les autres.

L'autre problème que l'on évoque trop peu souvent : la pérennité de l'API. Car une API doit vivre, être maintenue, évoluer. Il faut surveiller son bon fonctionnement au quotidien, s'assurer que son déploiement est optimal. Pour le développeur qui va consommer l'API, le problème est important, car une API cassée va rendre inutilisable une partie de l'application. De même, une API qui n'est plus mise à jour peut rendre obsolète son projet. Et personne n'est à l'abri d'un changement interne dans l'API ou de son arrêt pur et simple. Bref, qui fait quoi ? Qui garantit quoi ?

Ce mois-ci, nous allons rentrer dans le dur avec un cahier spécial Visual Studio 2013 pour mieux comprendre et utiliser les nouveautés : Code Map, Visual Studio Online, Xamarin, Windows Azure, les nouveaux outils de tests, etc. Nous revenons aussi sur Java 8 qui doit sortir (promis, juré, craché) le 18 mars prochain. Espérons que les plus gros bugs seront corrigés à temps.

Si vous avez raté les conférences françaises du hacking de 2013, nous vous proposons un compte-rendu complet qui vous donne les principales tendances...

Bonne programmation !

François Tonic
Directeur de la publication & rédacteur en chef
ftonic@programmez.com

sommaire

96 **Le coin du designer**

84 **Webmaster**

12 **Agenda**

10 **Journal d'un dév #3**

74 **Bug**

38 **Data Scientist**

4 **Les chiffres clés**

22 **Les coulisses du lancement de VS 2013**

15 **Abonnez-vous !**

6 **MariaDB**

39 **Spécial Visual Studio 2013**

16 **Matériel**

24 **Les API nous envahissent !**

68 **Java 8 2e partie**

66 **Caméra GoPro**

98 **Time Machine**

88 **Android Annotations**

92 **PowerBI 2e partie**

14 **Développeuse du mois**

71 **Framework Lucene .net**

94 **SharePoint 2013 2e partie**

8 **Une vie à la Silicon Valley**

78 **Excel & Java 2e partie**

À LIRE DANS LE PROCHAIN NUMÉRO
n° 172 en kiosque le 28 février 2014

Java 8
Mi-mars, Java 8 sera disponible, quoiqu'il arrive. Qu'apporte cette version ? Java est-il toujours un langage moderne et d'avenir ?

Google Glass
Quelles applications développer ? Un nouveau marché pour les développeurs ?

Développement embarqué
Le défi du développeur

4K

le nouveau mot à la mode pour les écrans

Android

va-t-il tuer ChromeOS sur les desktop ?

iOS 7.1

ouf, une 4e bêta en janvier 2014.
Sortie indéterminée.

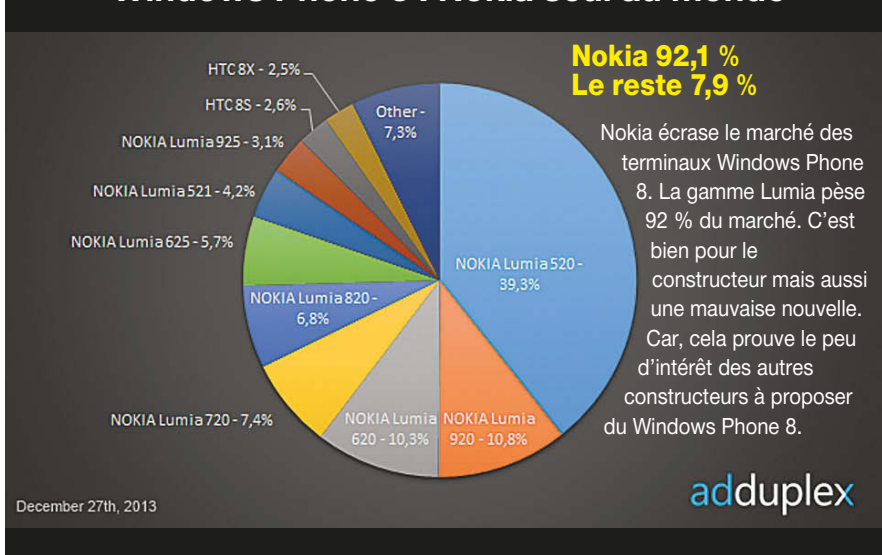
Bitcoins

friture sur la monnaie, entre taxation
et interdiction de paiement

150 000 €

ce que réclame la CNIL à Google
(soit grosso modo le salaire d'un ingénieur
logiciel expert)

Windows Phone 8 : Nokia seul au monde



Miroir, miroir, quel est le langage le plus utilisé ?

L'index TIOBE donne chaque mois les langages les plus utilisés par les développeurs.
Il s'agit d'un indicateur basé sur les recherches web. Des % à manipuler avec précaution.

Janvier 2014	Janvier 2013	Tendance	Langage	%	Evolution (en %)
1	1	=	C	17.871%	+0.02%
2	2	↓	Java	16.499%	-0.92%
3	3	↑	Objective-C	11.098%	+0.82%
4	4	↓	C++	7.548%	-1.59%
5	5	↓	C#	5.855%	-0.34%
6	6	↓	PHP	4.627%	-0.92%
7	7	↓	(Visual) Basic	2.989%	-1.76%
8	8	↓	Python	2.400%	-1.77%
9	10	↓	JavaScript	1.569%	-0.41%
10	22	↑	Transact-SQL	1.559%	+0.98%

Le trio de tête reste identique : C, Java et Objective-C. Ce dernier grignote encore quelques dixièmes. Tous les autres accusent une baisse, plus ou moins forte. T-SQL continue à gagner un peu. En un an, ce langage gagne 12 places.

1 milliard \$

l'investissement d'IBM
dans la division Watson.

M#

le nouveau langage de Microsoft
dédié au projet Midori

+ 14 mois de retard

le dérapage de l'édition tactile de Firefox
(Windows 8.x) qui n'en finit plus d'être
repoussée. Nouvelle date de disponibilité :
mars 2014. Date initiale : janvier 2013

Gratuit

Google veut imposer son codec VP9.
Il doit permettre de diffuser des vidéos 4K.
Et se pose en concurrent très sérieux
du codec H.265.

10 milliards \$

c'est le montant généré par l'App Store
d'Apple en 2013 !

Les culturels en France : 75 milliards €

L'ensemble des marchés culturels représente 74,6 milliards € (2011) et 1,2 millions d'emplois.
Le jeu vidéo pèse 5 milliards et presque 24 000 emplois...

(source : EY)

Les industries culturelles et créatives en France



Tabou : le chômage des informaticiens

Le marché informatique évite-t-il le chômage des informaticiens (tout profil) ? Non. Officiellement, ce sont +34 000 informaticiens inscrits à Pôle Emploi. Le Munci a analysé les chiffres en croisant tous les chiffres disponibles (et les différents codes ROME informatique et télécoms). Le résultat est tout autre : +76 000 demandeurs d'emploi. Le développement et les études pèsent 27,2 %

(source : Munci)

MariaDB grignote !

Est-ce que MariaDB va réussir à déloger MySQL ? Une étude menée par 451 Research, Uptime Institute et Yankee Group, montre que l'écosystème MySQL change peu à peu. En 2013, MySQL pesait assez largement : 62 %. La projection pour 2018 établit une forte croissance pour Percona, MariaDB et un tassement de MySQL (45 %)...

Combien d'employés ?

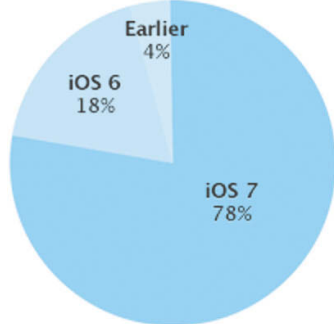
Amazon	109 800
Microsoft	99 000
Apple	80 300
Google	46 421
eBay	31 500

(source : the wall street journal)

Méchants geeks de la Silicon Valley

La mairie de San Francisco en a assez de voir les bus des grandes sociétés technologiques utiliser les arrêts des transports en commun. Le maire veut réduire les arrêts autorisés et les faire payer pour récupérer un peu d'argent. Car finalement, les bus privés utilisent une partie du réseau des transports en commun. Google veut contourner le problème avec des navettes fluviales...

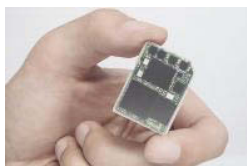
Fragmentation iOS



Source : Apple

UN ORDINATEUR DE LA TAILLE D'UNE CARTE SD ?

Challenge accepté et réussi par Intel avec le projet Edison. De la



taille d'une carte SD, Edison est une carte de développement utilisant un processeur Quark 400 Mhz gravé en 22 nm. Il intègre 2 cœurs, le wifi et le bluetooth. Il est parfaitement adapté à la domotique et aux vêtements intelligents.

McAfee

devient Intel Security

Mars

sortie probable du Samsung S5

2016

déjà des vidéos 8k...

2015

disponibilité de Windows 9 (?)

2014

les premiers mobiles sous Ubuntu Touch ?

147

le nombre de correctifs de sécurité prévu par Oracle dans la mise à jour mensuelle...

2to

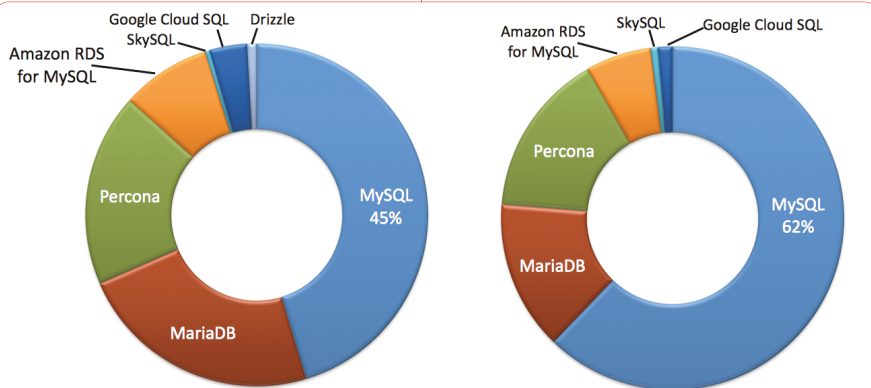
la capacité d'un SSD d'Adata, prix inconnu.

WebOS

omniprésent sur les SmartTV de LG en 2014

Combat : PS4 vs Xbox One

Sony affirme avoir vendu 4,2 millions de PS4... Dans le même temps, Microsoft annonce 3 millions de Xbox One ! 1er set : Sony. Le set retour risque d'être musclé !

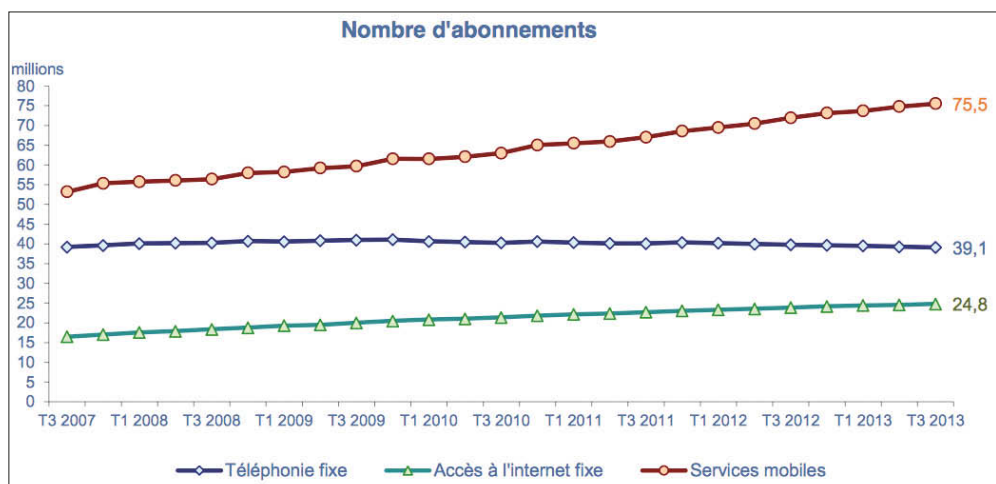


Où en est la téléphonie en France ?

L'ARCEP a publié son observatoire des marchés des communications électroniques en France pour le 3e trimestre 2013.

Quelques chiffres clés :

- **9,3 milliards €** de revenus pour les opérateurs
- **56 milliards de minutes :** trafic de la téléphonie fixe (+2,5 %)
- **45,4 milliards** de SMS et MMS
- **5,3 milliards** d'investissement sur les 9 premiers mois de 2013
- le nombre de salariés de ce secteur n'a pas changé.



MariaDB 10 : rencontre avec Monty

En décembre dernier, Michael Widenius, alias Monty, a fait une halte sur Paris pour présenter aux développeurs et à la communauté la nouvelle version de MariaDB, la version 10. Version très attendue. Programmez ! était là.

Le Meetup de décembre 2013, organisé par SkySQL et LeMug (MySQL User Group) fut l'évènement de libération du fork MariaDB. La version 10 était annoncée depuis des mois. Elle est compatible avec MySQL 5.6, tout en apportant de nombreuses nouveautés.

Les améliorations que vous y trouverez, se découpent selon différents axes :

- ▶ Les fichiers de données et de définition des tables (.frm) sont compatibles au niveau binaire.
- ▶ Toutes les API clients, les protocoles et les structures sont identiques.
- ▶ Tous les noms de fichiers, fichiers binaires, chemins, ports, sockets et etc.... devraient être les mêmes.
- ▶ Tous les connecteurs de MySQL (PHP, Perl, Python, Java.NET, MyODBC, Ruby, connecteur MySQL C etc.) travaillent de la même manière avec MariaDB.
- ▶ Le paquet mysql-client fonctionne aussi avec le serveur MariaDB.

Choisir sa version

MariaDB propose 2 versions : MariaDB 5.5 et MariaDB 10. La branche de MariaDB 5.5 est la version stable qui permet de garantir une complète compatibilité avec MySQL. De plus, elle supporte certaines fonctionnalités intégrées dans MySQL 5.6. La version MariaDB 5.6 est une version qui garantit l'ensemble des nouveautés apparues dans MySQL 5.6. Cependant, les derniers ajouts de celle-ci, non planifiés sur la roadmap, ont nécessité un délai d'implémentation. La version MariaDB 10, actuellement 10.0.7, bénéficie des nombreuses évolutions et de la compatibilité de MariaDB 5.6. La v10 est en cours d'optimisation, et le travail sera terminé durant le 1er semestre 2014. Bien entendu, certains utilisateurs ont déjà franchi le pas pour bénéficier des différents moteurs de stockage. C'est pourquoi si vous devez installer un nou-

veau serveur et que vous souhaitez une compatibilité, vous devez privilégier, pour le moment, la version 5.5.

Galera Cluster peut-il devenir MariaDB Cluster ?

"Galera Cluster est un acteur privilégié de MariaDB, et, lorsque la fondation parle de cluster, on préfère utiliser MariaDB Galera Cluster, car nous nous occupons du côté serveur", précise Monty. Par ailleurs, Percona extraDB Cluster est considéré comme l'équivalent de MariaDB Galera Cluster au niveau fonctionnel.

L'annonce de MySQL 5.7, qu'en pensez-vous ?

La réponse de Monty : "Oracle a dévoilé une partie du contenu de la version de MySQL 5.7 en septembre dernier. Les évolutions annoncées sont une bonne chose pour la richesse de l'écosystème et une bonne avancée pour la communauté.

Certaines fonctionnalités sont inspirées de MariaDB 10 pour permettre à MySQL de rester dans le cycle dynamique des bases de données, comme la réplication. Bien entendu, certaines évolutions sont propres à Oracle, comme le 'select primary'. Ces nouveautés seront supportées par MariaDB, lorsqu'elles seront données à la communauté."

Performances MySQL / MariaDB

La réponse de Monty : "il est difficile de comparer les 2 bases et encore plus quand il s'agit d'un fork et du projet originel. Suivant l'angle d'approche, du contexte d'utilisation, de la configuration des serveurs, les résultats seront différents. Mais si vous souhaitez réellement effectuer une comparaison, vous devez choisir les mêmes configurations et options, et vous verrez le résultat par vous-même que notre base de données est plus performante."



Le futur de MariaDB ?

La réponse de Monty : "avec l'arrivée de la version 10 de MariaDB, de nombreuses nouveautés et améliorations ont été apportées. C'est un grand pas pour nous, par rapport à ce que l'on proposait jusqu'à présent. La prochaine étape consiste à stabiliser et à améliorer l'existant, par exemple, les moteurs de stockage.

D'autre part, de nombreuses améliorations ne sont pas définies. Nous pouvons ainsi évoquer l'arrivée de nouveaux moteurs NoSQL; présence encore faible mais qui devrait s'accroître dans les prochaines versions. Il y a aussi des refontes des mécanismes de réplication, du sharding et du load balancing.

N'oubliez pas que la roadmap de Maria est transparente. Elle est influencée par la communauté, l'équipe de développement, et par sa base-mère, MySQL."

Conclusion

Durant l'entretien, Monty a voulu montrer que malgré la jeunesse de Maria, elle est assez mature pour répondre à de nombreuses attentes du monde du web. La preuve, de nombreux acteurs ont rejoint la fondation. La documentation, le support et les aides, en Français, sont en cours de conception et de réalisation. Cela aidera les utilisateurs et la communauté. Nous n'avons pas fini d'entendre parler de MariaDB !

Christophe Villeneuve

Consultant IT pour Neuros, auteur du livre « PHP & MySQL-MYSQLi-PDO, Construisez votre application », aux Éditions ENI. Rédacteur pour WebRIVER, membre des Teams DrupalFR, AFUP, LeMug.fr, Drupagora, PHPTV..

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



SERVEURS DÉDIÉS

PREMIÈRE MONDIALE CHEZ 1&1



TOUT NOUVEAU ET DÉJÀ CHEZ 1&1 :

INTEL® ATOM™

1&1 SERVEUR DÉDIÉ A8i

À partir de

39,99

€ HT/mois
47,99 € TTC*

NOUVEAU : 1&1 SERVEUR DÉDIÉ A8i AVEC 30 % DE PERFORMANCE EN PLUS

- Intel® Atom™ C2750
- 8 Cœurs et 8 Go de RAM
- 2 x 1 To SATA HDD
- Parallels® Plesk Panel 11
- Linux, Windows ou Clé-en-main
- Bande passante 100 Mbps
- Architecture 64 bits
- System-on-Chip (SoC) : 30 % de performance supplémentaire



☎ 0970 808 911
(appel non surtaxé)



1and1.fr

* Le serveur dédié A8i est à partir de 39,99 € HT/mois (47,99 € TTC) pour un engagement de 24 mois. Egalement disponible avec une durée d'engagement de 12 mois ou sans durée minimale d'engagement. Frais de mise en service : 49 € HT (58,80 € TTC). Conditions détaillées sur 1and1.fr. Intel, le logo Intel, Intel Atom et Intel Inside sont des marques commerciales d'Intel Corporation aux États-Unis et/ou dans d'autres pays.

Chapitre 4 : Des escadrilles de robots dans chaque foyer !

Le monde de la robotique progresse comme jamais auparavant, le challenge « DARPA Robotics » pousse l'industrie à faire des bonds de géant. Les acteurs du numérique comme Google s'engouffrent dans le domaine. Le gouvernement américain ne sous-estime absolument pas la vague asiatique dans ce domaine et soutient plus que jamais les universités, laboratoires de recherche et entreprises souhaitant y prendre toute position. Eh oui chers homo sapiens, l'ère de la robotique, de « l'Homotique » et du Robotsapient(1) est bien en marche et nous n'en voyons que les prémisses à ce jour !

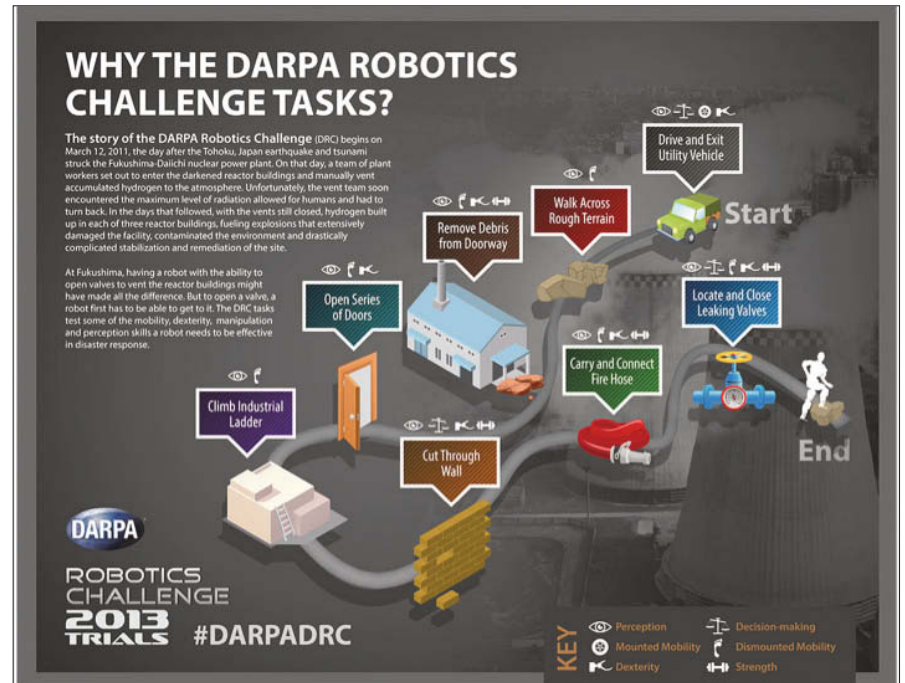
L'espèce des robots est en marche, et, à l'image de l'*humanité* désignant à la fois l'ensemble des individus appartenant à l'espèce humaine (homo sapiens) mais aussi les caractéristiques particulières qui définissent l'appartenance à cet ensemble(2), cette nouvelle espèce d'individus requiert une identité qualifiant son rôle et son importance aux côtés de l'humanité, nous l'appellerons « *Robotocité* » !

« La Robotocité peut être définie comme étant à la fois l'ensemble des individus appartenant à l'espèce robotique (robot sapiens, drones, driverLess car, ...) mais aussi comme étant les caractéristiques particulières qui définissent l'appartenance à cet ensemble ! »

A peine défini et ce terme nous semble déjà familier ! Convaincu de son existence une recherche en ligne restera pourtant creuse... Tentons ainsi de comprendre le pourquoi d'une telle vague ainsi que son statut au niveau mondial, au sein de la Silicon Valley et ses enjeux futurs !

Robocité : mais 'pourquoi' le robot ?

Revenons pour ce faire à l'ADN de la robotique, le terme robot trouve ses origines dans une pièce de théâtre tchèque créée par Karel CAPEK vers 1921, le terme signifiant en réalité « Travail Forcé », trouvant lui-même ses propres origines dans le mot Tchèque « Robota », signifiant travail ou servage. Ce terme robot allait alors prendre sa propre trajectoire en étant très vite récupéré par la science-fiction dont Isaac



Source : <http://www.theroboticschallenge.org>

Asimov avec les 3 lois de la robotique.

Les dés étaient jetés... il est dans la nature humaine que de vouloir asservir, dominer, conquérir, ... Telle est notre histoire ! Il est vain de tenter de lutter contre cet instinct et ces archétypes qui ressurgissent à tout moment du plus profond du meilleur des êtres possibles sur cette planète. La démarche de conquête devrait se trouver de nouveaux territoires ! La technologie EST un territoire de conquête, de ruptures et de désirs !!! Où, au-delà des auteurs de la science-fiction à succès, les ingénieurs hardware et software pourront laisser leur créativité et leurs talents s'exprimer ! Il n'en fallait pas moins pour donner naissance à un tsunami technologique où hardware et software devraient se marier pour donner jour à un nombre invraisemblable de projets, startups, entreprises à l'échelle mondiale mais aussi et surtout au sein de la Silicon Valley ! Et si nous, humanité, pouvions créer ce fameux robot ultime, convoitise de tout humain, individu de cette nouvelle espèce nommée *Robocité*, ayant pour *objectifs* de *servir* mais aussi et surtout de nous *aimer*, nous *aider* et nous *accompagner* dans le respect des 3 lois d'Asimov ?

Robotocité au niveau mondial

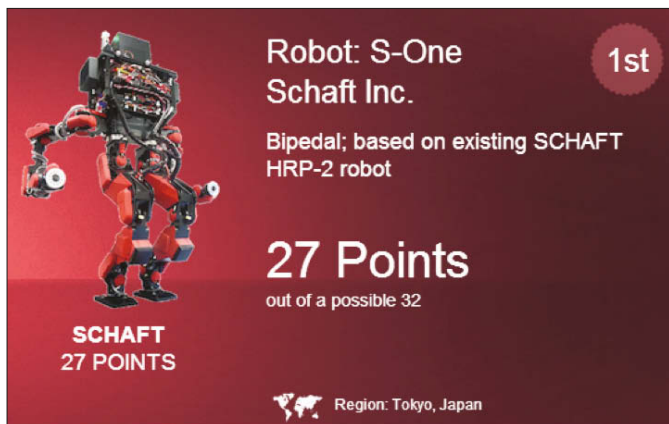
La robotique n'en est plus à ses prémisses, bien qu'encore très jeune, les résultats sont plus que prometteurs. Tout comme l'évolution de l'*humanité*, la *Robotocité* vit son propre cycle de vie ayant déjà eu un passage par différentes étapes identifiées que sont :

- L'état de *conception* avec la robotique littéraire de Science-Fiction (*premier type*) et automates.
- L'état *embryonnaire* avec la robotique industrielle de l'après-guerre (*premier type*).
- L'état de *fœtus* avec la robotique contemporaine en plein développement :
 - De ses capteurs, sensors, ... permettant de mieux comprendre son environnement (*Second Type*)
 - De son Intelligence Artificielle ... permettant actions et proactivité (*Dernier Type*)

La *Robotocité* deviendra plus que probablement le meilleur compagnon de l'humanité comme de nombreux ouvrages de science-fiction nous l'ont présenté durant ces dernières décennies... ou pas ! Explorons le challenge DARPA Robotics qui s'est déroulé en décembre 2013. L'histoire de ce challenge démarre en mars 2011 suite au tremblement de terre et tsunami de

(1) Robotsapient est une marque de la société WowWee

(2) Sources Wikipedia : <http://fr.wikipedia.org/wiki/Humanit%C3%A9>



s-One, vainqueur du DARPA Robotics Challenge 2013.



Robot Petman de Boston Dynamics - <http://youtu.be/RGZoMPXGOMI>

Fukushima-Daiichi et catastrophe nucléaire qui en découla. L'objectif étant de créer une nouvelle génération de robots capables d'opérer sur de tels sites en lieu et place d'êtres humains, les opérations devant être réalisées par un Robot sont représentées dans le schéma suivant (gérer des escaliers, ouvrir une porte, conduire un véhicule, fermer une vanne, ...). Le vainqueur est le Robot S-One de la société Schaft Inc. avec un total de 27 points sur 32. Je vous invite à voir la vidéo : http://youtu.be/Z9_hegb_JJE

Le lien avec la Silicon Valley devient très rapide lorsque l'on apprend que cette société a été tout récemment *rachetée par Google* en même temps que 7 autres sociétés également présentes au DARPA Challenge.

La Robotique au sein de la Valley !

Effectivement, la Robotique est devenue en quelques semaines, le sujet 'hot' de la Silicon Valley et le nombre de groupe Meetup ne fait qu'exploser. La société Google n'y est pas étrangère ! Après son annonce de rachat de pas moins de 8 entreprises (*Schaft Inc.*, Industrial Perception, Redwood Robotics, Meka Robotics, Holomni, Bot & Dolly, AutoFuss et enfin *Boston Dynamics*.), le ton est donné et les ingénieurs software ou hardware du domaine deviennent en quelques instants des profils clés pour entrer dans cette nouvelle course effrénée.

« *Google investit ainsi dans le futur* »

Je vous invite à effectuer une rapide recherche en ligne afin de parcourir leur contenu. Après avoir été dominée par les pays asiatiques, voici que la robotique redeviendrait de plein droit propriété américaine. Google a dominé le monde de la recherche en ligne, il semble bien décidé à aller au bout de l'accompagnement de l'être humain en lui proposant ses futurs compagnons robotiques...

Et que dire de leur dernière acquisition de l'entreprise Nest !?! La robotique est-elle réellement limitée à la notion de bipède ? Les robots seront beaucoup plus, ils seront 'tout' et partout, tout

objet connecté ayant en définitive une parcelle de software dernière génération intégrant intelligence artificielle et connecté ou non à des fermes de serveurs regorgeant de Big Data et Machine Learning. Les familles de robots se diversifient bien plus que nous n'avons de familles d'humains, la Robotique sera beaucoup plus vaste et complexe que ce que nous n'avons jamais connu dans le règne de l'humanité. Des nano-robots aux objets connectés, en passant par les robots fonctionnels (voitures, drones, surveillance, nettoyage, logistiques, livraison, ...), robots de jeux jusqu'aux robots de compagnie, sans oublier toutes les versions intermédiaires et plus encore ...

Le Maker-Faire au service de la robotique ?

Tout devient très intéressant lorsque l'on marie à la robotique la notion de Maker-Faire où s'intègrent des technologies que nous avons parcourues lors d'articles précédents telles que l'impression 3D digne représentante de ce mouvement à l'échelle mondiale du Maker-Faire.

Le Maker-Faire, mouvement lancé en 2005 avec le magazine MAKE, ouvre le champ à cette nouvelle industrie du « Do It Yourself » (DIY) qui n'aura de cesse de s'accélérer avec l'avènement du logiciel libre et de l'avènement de l'impression 3D ainsi que de composants tels que l'Arduino ou le Raspberry.

Cette nouvelle communauté, non contente de créer une réelle rupture dans le monde du prototypage et des productions en petites quantités, créée pour la plus grande liberté du peuple, de l'innovation et du progrès en général, l'explosion des positions dominantes d'acteurs industriels alors seuls acteurs à avoir les moyens de produire certains types de pièces techniques... La robotique, bien qu'encore au stade embryonnaire, fait clairement partie de ces cibles !

Un recherche rapide sur le site de plan d'impressions 3D libres « *thingiverse* » de la société MakerBot ressort à ce stade plus de 1400 plans potentiels de robots à imprimer par vous-même

depuis une imprimante 3D : <http://www.thingiverse.com/search/page:1?q=robot&sa=>

Un tri sera nécessaire mais vous y retrouverez à la fin de la journée plus d'une centaine de plans de qualité vous permettant de monter votre ou vos robots pour la plus grande joie de tout Geek et ou enfants et ainsi *créer des escadrilles de robots dans chaque foyer* !

Et après ?

Les robots, encore perçus comme des jouets ou objets de science-fiction, vont prendre une place de plus en plus importante dans nos vies. Leur compagnie et leur dépendance n'en sera que croissante, le marché de la robotique ne faisant que démarrer et ses règles n'en étant pas encore sèches ! La *Robotique*, ensemble des individus appartenant à l'espèce robotique, nous réservera de nombreuses surprises et l'adoption par le grand public et/ou les gouvernements pourrait être bien plus rapide et simple que nous ne l'imaginons à ce stade.

Les challenges et progrès sont encore nombreux afin d'aboutir au clone robotique bipède de l'être humain, mais devons-nous attendre d'atteindre ce Graal, désir ultime, pour adopter massivement ces nouveaux objets, ces nouveaux individus, ces nouveaux êtres ? Il vous reste le libre arbitre ou pas !

Liens

Karel Capek :

http://fr.wikipedia.org/wiki/Karel_%C4%8Capek

Isaac Asimov :

http://fr.wikipedia.org/wiki/Isaac_Asimov

Darpa Challenge : <http://www.theroboticschallenge.org>

Google : <http://makezine.com/magazine/robots-whats-google-up-to/>

IEEE Robotics : <http://spectrum.ieee.org/robotics>

SV Robots Startups : <http://www.quora.com/Silicon-Valley/What-are-all-the-robotics-companies-in-Silicon-Valley>

Greg

A Geek in the Silicon Valley

#3 : La difficulté de trouver un job

J'aime mon boulot. Il consiste à être ingénieur logiciel. Je n'ai pas le mot de « chef » ou de « responsable » dans mon titre. Je n'ai à faire ni d'avant-vente, ni d'infrastructure. Je n'ai pas non plus de clients à voir. Je ne fais pas de support, de réponses à des appels d'offres, de MOA/MOE ou autre. Non, mon boulot consiste simplement de mener à bien une fonction-

nalité et/ou un produit de sa conception technique jusqu'à la porte du « Microsoft Store ».

Mon lead (responsable de ma carrière) est un développeur bien meilleur que moi et le lead de mon lead l'est 10x plus. Le testeur, l'intégrateur, le manager de projet (PM) et l'ingénieur logiciel travaillent main dans la main pour « shipper » sans que l'un soit au-dessus de l'autre dans la hiérarchie. Pas de « manager » qui n'y connaît rien en code ou de commercial qui me dit ce que j'ai à faire. Bref pas de petit chef, juste de grands ingénieurs.

Travailler dans cet environnement est un vrai plaisir : des produits que j'aime, des technos que j'aime, des gens intelligents du monde entier, un environnement fabuleux avec ses lacs, ses montagnes et sa verdure et des avantages (retraite, salaire, santé, vacances, matériels,...) à faire pâlir n'importe quel développeur hors du top 5 des éditeurs logiciels. Oui mais...

Car il y a un mais !

Le premier « mais » ce sont les sacrifices qu'il a fallu faire : quitter la France et partir vivre à l'autre bout du monde avec + 9h de décalage horaire. Ce fut plus dur que je ne l'aurais pensé. La barrière de la langue la première année, le changement de culture, tous ces chocs ont été plus ou moins difficiles. Certains s'en sortent mieux que d'autres, pour moi cela aura parfois été dur. Mais, le plus difficile est de vivre dans l'insécurité américaine. Ici, le CDI n'existe pas. On peut se faire virer du jour au lendemain et toute votre protection sociale ainsi que le visa sont basés sur votre emploi. Difficile de faire des plans à long terme, acheter une maison, une voiture ou tout simplement se sentir chez soi.

Certes, ma revue de fin d'année est bonne, certes la « carte verte » (la fameuse « Green Card ») arrivera bientôt, certes on entend partout que nous sommes en pénuries d'ingénieurs logiciels et qu'on trouvera un autre travail « facilement ». La réalité est un peu différente. On cherche des ingénieurs en masse : oui, mais



quand on parle des meilleures entreprises telles que Facebook/Amazon/Google/Apple/Microsoft, elles cherchent des profils d'ingénieurs bien particuliers. C'est parfois une ligne de plusieurs candidats qui attendent pour tenter leur chance, on y trouve des gens de toutes nationalités, d'écoles connues tel que MIT, Stanford ou Princeton, et de moins connues...

Mon profil n'intéresse personne

En France, j'avais une casquette de professeur, auteur de livre, acteur de communautés et consultant (ce qu'on appelle aussi en France « expert » ou aussi « architecte »). Toutes ces choses ne pèsent pas lourd, voire, sont complètement ignorées par ces entreprises lors de l'entretien. Sans même parler du fait d'avoir été « responsable » ou « chef » quelque part. Les meilleurs experts Français ne viennent pas travailler ici, car ce ne sont pas ces profils qui sont recherchés. Si mes certifications, mon diplôme et mes années d'expériences ne comptent pas pour le poste d'ingénieur logiciel, qu'est-ce qui compte réellement ? Voici un exemple d'entretien de 8 heures (après avoir réussi un entretien technique d'une heure et un entretien RH de 30 minutes) :

► L'algorithme : éliminatoire. Si vous ne répondez pas à la question vous êtes certain de ne pas avoir d'offre.

1. Trouver tous les comptes bancaires parmi un tableau d'un million d'éléments non triés dont la somme vaut exactement N (NP-Complete).
2. Générer un B-Tree équilibré à partir d'une liste non triée.
3. Trouver tous les sous-ensembles de strings dans un string qui sont des palindromes (ex : avec une complexité de $O(n)$ en utilisant la programmation dynamique).

► L'architecture

4. Architecturer un système de réservation de billets d'avions avec UML (ex : penser à opti-

miser la data-structure pour pouvoir être capable de proposer au client un billet avec le moins d'escales possibles, avec la durée la moins longue et au prix le plus bas).

5. Architecturer un service de générations d'URL courts (ex : hashtable pour pouvoir gérer un grand nombre d'urls avec formule de hachage unique).

► Ce que vous avez déjà mis en production

6. Où puis-je tester des produits que vous avez mis en production ? Parler de votre expérience, quel est votre but dans votre carrière, en quoi travailler pour la société est important pour vous ?

► Votre façon d'être, de travailler en équipe, votre passion et créativité, savoir utiliser un contrôle de source ...

7. Vous devez créer une machine à café pour les voitures, comment faites-vous ? (Pour qui ? Pour quoi ? Comment ? A quel prix ? Parler du hardware si vous avez un background électronique...)

8. Avez-vous déjà travaillé avec un contrôle de source ? Sous quel environnement compilez-vous ? Avez-vous déjà mis en place une build avec intégration continue ? Quel est votre background au niveau des tests...

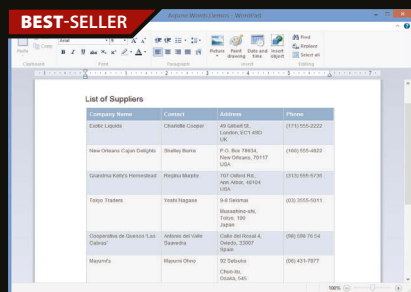
Non seulement il est difficile de décrocher un entretien et de se faire remarquer mais en plus il faut passer le test au téléphone qui peut se révéler difficile (ça dépend aussi de la chance qu'on a) et ensuite réussir tout ça sur place.

Préparez solidement les entretiens

Vous l'aurez compris, à moins d'être un bon en sciences et algo (et il y en a un paquet en France !) la tâche sera difficile. Je n'avais pas le niveau à la base et je pense qu'il serait très compliqué pour moi de quitter une équipe produit pour aller travailler dans des équipes tels que core, langage, compilateur...

Je vous recommande deux livres qu'il faut vraiment lire avant de se lancer dans tout entretien technique : « **Elements of Programming Interviews: 300 Questions and Solutions** » aux éditions Paperback et « **Introduction to Algorithms** » chez Hardcover (il existe une version Française mais je ne vous la conseille pas, histoire d'avoir le bon vocabulaire). Bon courage (ça m'a pris un an pour les lire :))

Julien Dollon - Julien.dollon.net

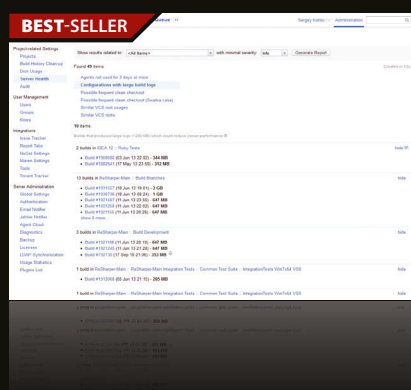


Aspose.Words for .NET à partir de € 731



Lisez, modifiez et écrivez des documents Word sans Microsoft Word.

- Création de documents, manipulation du contenu/formatage, puissante capacité de fusion de courrier et exportation en DOC/HTML
- Accès détaillé à tous les éléments d'un document par programmation
- Support les formats de fichiers: DOC, DOCX, WordprocessingML, RTF, HTML, OOXML, OpenDocument, PDF, XPS, EMF et EPUB

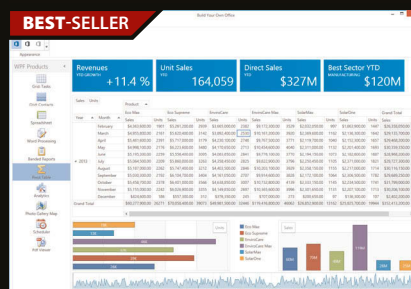


TeamCity à partir de € 1 763



Serveur d'intégration continue prêt à l'emploi, extensible et conçu pour les développeurs.

- Configuration rapide et intuitive des projets de builds
- Gestion intelligente de serveurs (usage des disques/état serveurs)
- Analyses et couvertures de code intégrées
- Extensibilité simple via REST API, messages des scripts de builds et API ouverte
- Support intégré pour une vaste gamme de technologies incluant Java, .NET, Ruby, Objective-C et Android

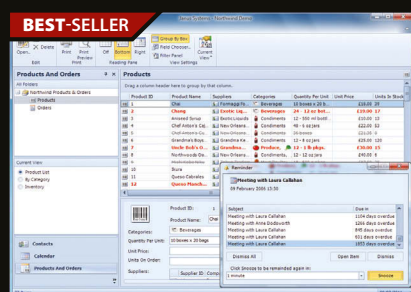


DevExpress Universal Suite à partir de € 1 611



400+ outils et contrôles WinForms, ASP.NET, WPF, Silverlight et Windows 8.

- Exploitez votre base de codes pour développer des applications tactiles multiplateformes
- Inclut un tableau de bord de visualisation interplateformes et un serveur de rapports
- Nouveaux contrôles DevExpress incluant Tableur et Carte
- Codage, débogage et refactorisation avec CodeRush pour Visual Studio
- Inclut la nouvelle galerie unifiée de modèles applicatifs DevExpress



Janus WinForms Controls Suite V4.0 à partir de € 678



Ajoutez des interfaces de style Outlook à vos applications .NET.

- Vues ruban, grille, calendrier, et barres chronologique/raccourcis
- Nouveau – Style visuel Office 2010 pour tous les contrôles
- Nouveau – Support des profils client Visual Studio 2010 et .NET Framework
- Janus Ribbon ajoute Backstage Menus et la fonctionnalité onglet comme dans Office 2010
- Prend désormais en charge la sélection de cellules multiples

février

Dart flight school :

6 février à Nantes

A l'occasion de la sortie de DART 1.0, le groupe utilisateur Google de Nantes organise une soirée spéciale pour faire découvrir le langage. Du code, du fun, des rencontres ! Site : <http://goo.gl/CybEos>

NIDays : 11 février à Paris

National Instruments organise sa traditionnelle journée. L'invité vedette en sera Joël de Rosnay, prospectiviste de renom. Il viendra parler des enjeux de la civilisation numérique. Plus de 50 ateliers et sessions techniques se succéderont : système embarqué, instrumentation, cartes d'acquisition, etc. LabView sera, comme chaque année, un des thèmes phares.

NIDays organise aussi la nouvelle édition des coupes robotiques et le concours des meilleures applications. 12 équipes de lycées s'affronteront pour la Coupe NXT (basée sur le kit LEGO MINDSTORMS NXT) et 12 équipes d'écoles supérieures pour la Coupe RIO (basée sur le Starter Kit Robotics de National Instruments). La coupe robotique réserve toujours surprises et suspense en démontrant l'habileté des jeunes étudiants. Pour en savoir plus : <http://france.ni.com/nidays/coupes-robotiques>

TechDays 2014 :

11, 12 et 13 février

La grande conférence TechDays de Microsoft revient pour une nouvelle édition qui s'annonce très riche pour les développeurs avec de nombreuses sessions sur Windows Azure, Windows, Windows Phone, les outils ALM, DevOps ! Plus de 300 sessions se dérouleront sur les 3 jours.

Quelques sessions à ne pas rater :

- ▶ Votre futur est dans le code : 11 février 9h15 – 10h30
- ▶ C# Async, 1 an après... : 11 février 11h – 11h45
- ▶ Coding4Fun : une des sessions les plus attendues. Avec de grands moments de « n'importe quoi » et les codes les plus délirants de l'année ! 11 février 17h45 – 18h30
- ▶ Geek is in da house : l'autre session que tout le monde attend ! Session toujours plus WAF, toujours plus de domotique. Que nous réservent les développeurs en folie ? Réponse le 12 février !
- ▶ Symphony2 et Windows Azure : 13 février 12h15 – 13h
- ▶ Après Coding4Fun, voici NUI4Fun : la communauté Kinect va se lâcher sur scène. 11 février 16h30 – 17h15

Programmez ! et Cloudmagazine seront partenaires de l'événement. François Tonic, directeur de la publication & rédacteur en chef, co-animera deux conférences autour du Cloud Computing et de Windows Azure. Le magazine y aura comme chaque année un stand.

Site : <http://www.microsoft.com/france/mstechdays/>

mars

Global Windows Azure Bootcamp 2e édition

Le 29 mars prochain, une journée Windows Azure aura lieu en France, à Paris et à Lyon. Le Global Windows Azure est un événement mondial avec des conférences, des sessions techniques, des rencontres. Objectif : montrer Azure, les possibilités techniques et le potentiel pour le développement. En France, c'est la communauté ZeCloud qui l'organise. Les participants devront apporter leur environnement de développement et les outils nécessaires pour participer à la journée... site : <http://goo.gl/TJW4t>



avril

DEVOXX FRANCE 2014

L'événement du monde Java se déroulera du 16 au 18 avril prochains à Paris. Cette année les principaux thèmes seront : Java, HTML 5, mobile, Cloud, big data, NoSQL, agilité, DevOps, les langages alternatifs, les startups et aussi le futur DevOxx. Avec l'arrivée de Java 8, Java sera bien entendu à l'honneur mais DevOxx s'ouvre à d'autres horizons. La première journée est dite "université". Ce sont des conférences de 3h et des ateliers. En fin de journée, des sessions courtes se succéderont. Les 17 et 18 avril, c'est la conférence proprement dite avec les multiples conférences. Site : <http://www.devovx.fr>

DEVOXX FRANCE

La Conférence des Développeurs Passionnés



SCRUMDAY 2014

L'événement phare de l'agilité et de la méthode Scrum en France aura lieu cette année les 10 et 11 avril à Disneyland Paris. Le Scrum User Group France est toujours le maître d'œuvre. Site : <http://www.scrumday.fr>

Les conférences sécurité et hacking à suivre

17 mars : journée de la sécurité des systèmes d'information.

18 mars : GS Days. De nombreuses conférences auront lieu autour du cloud, du web, du hacktivisme.

La conférence Hackito Ergo Sum revient les 24-26 avril à Paris ! Une des conférences phares du hacking en France. Site : <http://2014.hackitoergo-sum.org>



Une conférence, un salon,
l'agenda d'un groupe utilisateurs ?

Contactez-nous : redacteur@programmez.com

Rendez-vous le 17 février*

Le nouveau site web de Programmez! arrive.

Nouvelle présentation
Nouveaux contenus
Nouvelle boutique
Nouvelle newsletter

Quelques chiffres :
+ 16 semaines de développement
100 % Drupal
3 développeurs
1 designer

The screenshot shows the Programmez! website with a newsletter sign-up form on the right side. The form includes a 'Je m'inscris' button. The main content area features a 'tech-days 2014' banner for February 11-13 at the Palais des Congrès de Paris. Below the banner, there are sections for 'A LA UNE CETTE SEMAINE' (featuring Java 8 and Windows 8.1), 'AVIS D'EXPERTS' (featuring botnets), and 'LA SUITE DE L'ACTUALITES SUR PROGRAMMEZ!' (featuring Windows XP migration). The bottom of the page has a 'tech-days 2014' banner and a 'Je m'inscris' button.

Newsletter

Accueil

actus

The screenshot shows the Programmez! website with a main content area on the left and a newsletter sign-up form on the right. The main content area features a 'tech-days 2014' banner for February 11-13 at the Palais des Congrès de Paris. Below the banner, there are sections for 'ACTUALITES' (featuring Java 8 and Windows 8.1), 'AVIS D'EXPERTS' (featuring botnets), and 'EN CE MOMENT SUR LE FORUM' (featuring Windows XP migration). The right side of the page has a newsletter sign-up form with a 'Je m'inscris' button and a 'Retrouvez-nous sur Facebook' section.

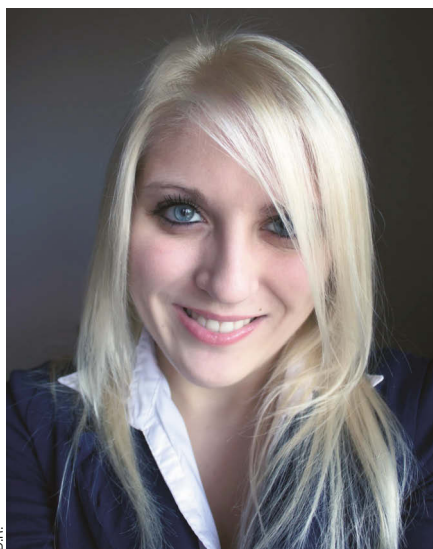
The screenshot shows the Programmez! website with a main content area on the left and a newsletter sign-up form on the right. The main content area features a 'tech-days 2014' banner for February 11-13 at the Palais des Congrès de Paris. Below the banner, there are sections for 'ACTUALITES' (featuring Java 8 and Windows 8.1), 'AVIS D'EXPERTS' (featuring botnets), and 'EN CE MOMENT SUR LE FORUM' (featuring Windows XP migration). The right side of the page has a newsletter sign-up form with a 'Je m'inscris' button and a 'Retrouvez-nous sur Facebook' section.

Tout change sauf le code

Programmez ! expert du code depuis 1998

* certaines fonctions ne seront pas complètes. Version bêta

D.R.



Peux-tu te présenter en quelques mots ?

Je m'appelle Zoé Caudroy, j'ai 22 ans je prépare actuellement un diplôme d'ingénieur à l'ESIEE Paris en apprentissage avec l'Agence Nationale de la Sécurité des Systèmes d'Informations (ANSSI). Mon travail consiste à concevoir et mettre en œuvre des produits et des systèmes d'informations sécurisés au profit des ministères et des opérateurs d'importance vitale. A côté de ça, je fais partie des communautés Microsoft en tant que Microsoft Student Partner et organisatrice des Afterworks. J'anime également un blog où vous pouvez trouver quelques tutoriaux et des retours de conférences.

Comment es-tu tombée dans le monde informatique et en particulier dans l'univers des développeurs ?

J'avoue que j'ai toujours pensé à suivre un cursus informatique le plus tôt possible. D'abord parce que j'ai toujours eu des ordinateurs depuis que je suis très jeune, et parce qu'au final, durant les années de collège, j'ai eu beaucoup de sensibilisation à l'informatique. J'ai eu mon premier ordinateur quand j'avais 8 ans. J'étais une grande fan de jeux vidéo et mon rêve était de développer un jour, à mon tour, des jeux qui passionneraient des personnes comme cela me passionne. Au final, après mon passage à l'Epitech je me suis rendue compte que je ne voulais pas faire du développement un métier, mais plutôt une passion. Je me suis donc orientée dans le secteur de la sécurité des systèmes d'informations.

Comment vois-tu notre univers ?

Je pense que les développeurs ne sont pas assez mis en avant. Pour moi, un développeur est une personne logique, organisée et très créative. Les capacités des développeurs sont soumises, la plupart du temps, à des contraintes imposées par les supérieurs. Comme par exemple le manque de liberté sur

Zoé, de la sécurité aux soirées communautaires !

Un parcours détonnant et de l'enthousiasme

les projets. La plupart des clients ne sont jamais contents du résultat et il ont beaucoup de mal à réaliser la quantité de boulot qu'il y a derrière les applications. Je pense qu'il faut vraiment être passionné pour faire ce métier.

Est-ce que tu te disais au début : "mais dans quel monde suis-je tombée ou qu'est-ce que je fais là ?"

J'ai sûrement du y être confrontée inconsciemment, mais j'ai toujours tenu tête, tenu bon, et je me suis toujours battue pour défendre mes visions.

Tu as de nombreuses activités. Tu t'occupes notamment des AfterWorks. C'est quoi ? A quoi ça sert ?

Les Afterworks des communautés Microsoft ont été créés par Laurent Miltgen-Delinchamp et Williams Bories afin de rassembler de manière mensuelle, tous les mois, dans un même lieu convivial, l'ensemble des acteurs du monde communautaire Microsoft. Ces événements sont ouverts à tous les UG (Groupes Utilisateurs), aux clubs d'informatique, aux communautés en ligne, aux MVP (Most Valuable Professionals), aux MSP (Microsoft Student Partners), aux partenaires Microsoft, mais éga-

lement à toute personne souhaitant échanger autour des technologies et des produits Microsoft.

Le geek est très la mode, un peu le nerd aussi. Qu'est-ce que tu en penses ?

Je ne pense pas que le geek / nerd soit vraiment à la mode. Dans mes différentes formations à l'école et au travail j'ai plutôt bien souvent rencontré des hommes communicants, et non pas que des nerds ou des geeks comme le laisse croire le stéréotype du développeur.

Tu vas travailler à l'ANSSI. Tu as envie de faire de la sécurité informatique ?

Le domaine de la sécurité informatique est très dynamique, l'évolution des technologies est très rapide. Il faut avoir un excellent esprit de synthèse et de curiosité intellectuelle. On ne doit pas avoir peur du changement. C'est exactement ce que je recherche aujourd'hui. La sécurité informatique est devenue une nécessité. C'est pourquoi j'ai choisi d'intégrer l'ANSSI pour mon cycle d'ingénieur en apprentissage. Elle est l'autorité nationale en matière de sécurité et de défense des systèmes d'information.



D.R.

Une année pleine
de technologies
et de codes
avec

PROGRAMMEZ!
le magazine du développeur www.programmez.com

1 an 11 numéros

49€

seulement (*)

2 ans 22 numéros

78€

seulement (*)

Complétez votre collection

Spécial étudiant
39€
1 an 11 numéros

réservée à la France
Métropolitaine

(*) Tarifs France métropolitaine

Toutes nos offres sur www.programmez.com

Oui, je m'abonne

ABONNEMENT retourner avec votre règlement à
Programmez, 17, route des Boulangers 78926 Yvelines cedex 9

- ☐ **Abonnement 1 an au magazine** : 49 € (au lieu de 65,45 €, prix au numéro)
☐ **Abonnement 2 ans au magazine** : 78 € (au lieu de 130,9 €, prix au numéro)
☐ **Abonnement spécial étudiant 1 ans au magazine** : 39 €
 Photocopie de la carte d'étudiant à joindre

Tarifs France métropolitaine

- ☐ **168** : ☐ exemplaire(s)
☐ **169** : ☐ exemplaire(s)
☐ **170** : ☐ exemplaire(s)

TOTAL : x 6 € = €

Prix unité : 6 € (frais postaux inclus),
France métropolitaine uniquement.

Adresse d'envoi de la commande :
Programmez!
7, avenue Roger Chambonnet
91220 Brétigny sur Orge

nouveau

☐ M. ☐ Mme ☐ Mlle Entreprise : Fonction :

Prénom : Nom :

Adresse :

Code postal : Ville :

Tél : (Attention, e-mail indispensable pour les archives sur internet et offre Eni)

E-mail : @

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

☐ Je souhaite régler à réception de facture

Kano : un ordinateur en kit pour l'été 2014

Kano est un ordinateur en kit à monter soi-même. Le projet a tellement intéressé que les scores Kickstarter ont explosé. Au départ les fondateurs de Kano demandaient 100 000 \$. A l'arrivée, ce sont 1 522 160 \$ qui ont été donnés pour soutenir ce projet un peu fou ! À Programmer ! nous l'avons tout de suite aimé !

Le principe est assez simple. Dans une boîte rectangulaire, on retrouve tous les éléments nécessaires pour créer un ordinateur. L'ensemble se base sur un Raspberry Pi modèle B. On dispose d'une alimentation, d'un dongle WiFi, des câbles nécessaires, d'une carte SD contenant le système (Kano OS), d'un clavier spécifiquement créé, de guides d'utilisation et de montage et d'un boîtier en kit, ainsi qu'un module audio.

Kano OS est un dérivé de Debian Linux. L'équipe a adapté et redéveloppé une partie du système durant plus de 6 mois pour que le système soit le plus performant possible et implémente les fonctions Kano. Le travail a été assez considérable :

- nouvelle interface inspirée des interfaces mobiles
- boot ultra-rapide, -10 secondes
- image système pesant à peine 1 Go, avec redimensionnement

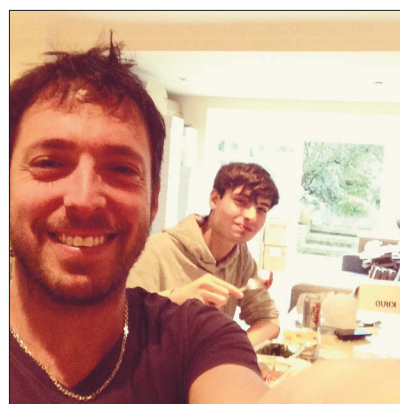
- dynamique de l'image au 1er démarrage
- différents logiciels dédiés Kano comme Minecraft
 - mécanisme de mise à jour
 - outils en ligne de commande

Un système complet

Cependant attention, Kano n'est pas forcément un ordinateur facile à utiliser. Car malgré les modifications et la nouvelle interface, il faudra toujours user de la ligne de commande pour certaines opérations. Mais pour l'amateur, le bidouilleur, Kano sera un formidable outil. Surtout, cela donne un usage concret au Raspberry, sans (trop) se casser la tête à trouver un usage. C'est la remarque que

l'on nous fait souvent. « Bon ok tu as un Raspberry, tu en fais quoi ? ». Souvent, on ne sait pas quoi répondre... Kano est la réponse...

Kano viendra aussi avec un langage de programmation léger : Kano Blocks. Il s'inspire du projet Google Blockly. Le but est de proposer un langage visuel, orienté jeux, basé sur des blocs que l'on assemble et que l'on configure très simplement. Il faut aussi que les programmes créés s'exécutent très rapidement sur la machine. Mais l'ambition



Alex et Yonatan

est que l'exécution se fasse dans un navigateur, jouable sur le web et donc partout. Kano proposera du développement pur et dur avec Python et JavaScript.

Un kit à 99 \$

Le kit complet sera proposé à 99 \$ (Kano Kit). Il sera disponible vers juillet 2014. Une version Kano Lab sera proposée à partir de 999 \$ et inclura 10 kits. Mais attention, il sera très difficile de trouver un Kano Kit dès sa sortie officielle, car les premiers soutiens sur Kickstarter sont prioritaires. Le Raspberry Pi nu est vendu environ 25-35 \$.

Quelques liens

Site officiel : <http://www.kano.me>

Projet sur Kickstarter :

<http://goo.gl/DKqhyP>



DANS LA JUNGLE DU CLOUD, MIEUX VAUT CHOISIR LE BON PARTENAIRE.



Aruba Cloud, les solutions IaaS qui répondent à chacun de vos besoins.

CLOUD COMPUTING

- Créez, activez et gérez vos VM.
- Choisissez parmi nos 3 hyperviseurs.
- Maîtrisez et planifiez vos ressources CPU, RAM et espace disque.
- Uptime 99,95% garanti par SLA.

CLOUD OBJECT STORAGE

- Créez vos espaces et stockez vos données en toute sécurité.
- Une solution qui s'adapte à vos besoins : Pay as you Go, ou formule prête à l'emploi.
- Bande passante et requêtes illimitées.

LE CLOUD PAR ARUBA

- Ubiquité : choisissez votre pays et datacenter.
- Interopérabilité : API et connecteurs.
- Agnosticisme : choisissez votre hyperviseur.
- Scalabilité : étendez votre infrastructure à l'infini.
- Transparence : pas de coûts d'activation, ni coût caché.
- Pay as you Go : ne payez que ce que vous consommez.

Aruba, le bon partenaire pour bénéficier de la puissance d'un acteur majeur qui considère que chaque client, dans chaque pays, est unique. **MY COUNTRY. MY CLOUD.**

aruba
CLOUD
arubacloud.fr | TÉL : 0810 710 300
(COÛT D'UN APPEL LOCAL)



PRIMO : du bois, des cases, du code !

Comment apprendre la programmation avec une boîte en bois, des cases, des blocs et un petit robot amusant ? C'est tout le concept original de Primo. Le matériel est animé par une carte Arduino. Et la première étape est de construire son robot, Cubetto Robot. Pour piloter cette drôle de boîte en bois, il nous faut l'Interface Board. Cette console possède des cases que l'on remplit avec des blocs de différentes couleurs. Ces blocs donnent des instructions. Ils sont munis de capteurs. L'enfant construit la logique de fonctionnement directement avec la board. Le concept est drôle et très ludique. Primo sera disponible à partir d'août. La mauvaise nouvelle est le prix : 170 £ (+ 200 €) ! site : <http://primo.io>

Lenovo : une nouvelle tablette pro

Le constructeur Lenovo a dévoilé lors du grand salon CES la tablette ThinkPad 8. Elle se destine aux entreprises. Elle peut se connecter à un clavier (un peu comme la Surface de Microsoft). Equipée d'un port micro-USB 3, de 128 Go de stockage, de Windows 8 Professionnel, d'un écran 1920x1200 et d'une connexion réseau LTE (téléphonie), elle sera disponible courant février, pour un prix de base de 377 € H.T.

HP met Android sur un PC

Et si Android était un véritable système pour ordinateurs et pas uniquement pour les terminaux mobiles ?

HP et Lenovo le pensent avec le Slate 21 All-in-One, ordinateur tout-en-un de 21" (écran tactile). Il fonctionne avec Android 4.3 avec un prix très agressif ; moins de 400 \$. À ce prix, les capacités de la machine seront en standard limitées : 2 Go de Ram, 16 Go de stockage. Il fonctionne avec un processeur Tegra 4. Il embarque 3 ports USB 2, Ethernet 10/100, prise audio, lecteur SD, Bluetooth et WiFi. La machine sera disponible courant de l'été. Pour HP, le choix d'Android, au détriment de ChromeOS, s'explique par la personnalisation du système et la richesse de Google Play.

Carrefour teste l'impression 3D

Un hypermarché Carrefour (Flins, 78) a mis en place et teste un service d'impression 3D avec le matériel A4 Technologie. Ce test grandeur nature a été réalisé les 10 et 11 janvier. Il s'agissait de mettre en place 4 imprimantes 3D. Auchan propose l'impression 3D près de Roissy. Ce service repossur

l'aide technique de CKAB, distributeur des imprimantes MakerBot. Le tarif dépend du poids de l'objet, allant de 4,99 à 29,99 € (sur devis pour des objets de +465 g).

Des modèles sont disponibles par défaut, mais il est possible d'imprimer son propre modèle. Mais il faut être patient, car l'impression 3D demeure lente, très lente.

Parrot fait des bonds

Durant le salon CES, le constructeur français a dévoilé deux nouveaux gadgets : Jumping Sumo et Mini Drone. Mini Drone est une déclinaison de l'AR/Drone que l'on contrôlera avec un appareil mobile. Il possède toujours 4 moteurs. Mais à la différence de son aîné, il possède deux roues pour « rouler » où il veut. À voir si la résistance sera au rendez-vous. Par contre l'autonomie risque d'être assez médiocre. Jumping Symo est un engin sur deux roues ; il apparaît comme très maniable et capable de faire des bonds. Aucune date ni tarif n'ont été dévoilés.

Une clé USB à double connectique

Integral propose une clé USB 3 à double connectique : USB classique (USB 3) et micro-USB. Pratique. « Alliant rapidité et efficacité, la clé USB 3.0 OTG (On-The-Go) est spécialement dédiée aux Smartphones et tablettes sous Android compatible. Le transfert de fichiers entre les terminaux mobiles et l'ordinateur est désormais simplifié, car il n'est plus nécessaire de disposer d'un réseau Wifi ou Bluetooth. » Dixit le constructeur. Cette clé est proposée en 16, 32 et 64 Go. À partir de 19,90 € H.T.

LaCie ose la boule



Google avait sa sphère, LaCie propose la sienne : LaCie Sphère. Le design est dû à Christofle. Il s'agit d'un disque dur de 1 To en USB 3 (autoalimenté), dommage pour le Thunderbolt. La boule argentée est vendue 390 €. Pour profiter d'un port Thunderbolt 2 (par exemple avec le nouveau Mac Pro), tournez-vous vers le Little Big Disk. Il propose des configurations SSD SATA III avec des débits théoriques de 635 Mo/s... Dommage que les tarifs en SSD soient élevés et limités à 1 To.

« SecConf » : La France qui cherche les trous ...

SSTIC, Hackito Ergo Sum, NoSuchCon, Grehack, Hack-in-Paris autant de manifestations ayant eu lieu en 2013 et très attendues pour 2014. Le moyen de connaître en avant-première les faiblesses potentielles des matériels et logiciels constituant les SI et donc de limiter les risques grâce à la connaissance...

L'année sécurité 2013 s'est achevée avec deux événements importants : Botconf, à Nantes, première conférence Européenne (et probablement mondiale) exclusivement consacrée à l'analyse et la lutte contre les Botnets, et Grehack à Grenoble. A cet inventaire il faut ajouter la mère de toutes les manifestations du genre, les SSTIC (Symposium sur la sécurité des technologies de l'information et des communications) de Rennes, Hackito Ergo Sum (alias HES) qui s'est déroulée cette année dans les amphithéâtres du Parc de la Villette, dans le nord de Paris, Hack in Paris, qui prenait ses quartiers de printemps à Eurodisney fidèle à son habitude, et NoSuchCon (NSC) qui démarrait sa première édition dans l'espace Niemeyer, Place du Colonel Fabien dans le Xème arrondissement de la capitale.

Si la répartition Paris/Province a pu être considérée comme équilibrée, ce n'est pas franchement le cas des calendriers : HES du 2 au 4 mai, NSC du 15 au 17 mai, SSTIC du 5 au 7 juin, Hack in Paris du 17 au 21 juin. Grehack (15 novembre) et Botconf (5 décembre, deux semaines avant la CCC Berlinoise mondialement courue) ont préféré se démarquer des grandes manifestations de printemps, et c'est un bien.

Grehack

Universitaire mais pas que. Ce que l'on retire de Grehack ? Un savant mélange de recherche pure. Par exemple cette année : développement étude et analyse d'un « Remote Access Tool » par Markku-Juhani et Olavi Saarinen, leçon d'analyse binaire par le célèbre Halvar Flake et travaux sur des attaques plus quotidiennes, sur des risques plus immédiats... Ainsi Mathieu Cunch, avec un exposé intitulé « Je connais votre adresse MAC, tracking ciblé de personnes par leurs terminaux mobiles Wifi ». L'adresse MAC d'une carte 802.11 est pratiquement une pièce d'identité, explique Cunch. Elle est propre à un et un seul appareil, donc à une et une seule personne. Et elle est transmise en clair non seulement durant les échanges avec un point d'accès, mais également lors des tentatives de découverte de réseau qui s'effectuent en permanence. Mieux encore, il est tout à fait pensable de faire « parler » un équipement Wifi et lui faire avouer quels sont les points d'accès auxquels il est déjà inscrit, et ceci avec une simple « replay attack ». La géolocalisation de person-



A l'occasion de NSC, Itzik Kotler présentait Pythonect, un DSL (Domain-Specific Language) utilisé pour construire Small (Simple Malware Analysis Language), script chargé de fouiller dans du code nuisible pour en extraire des chaînes de caractères significatives, des adresses IP, des messages camouflés.

ne, le traçage de ses habitudes de déplacement et donc de consommation, les lieux qu'elle fréquente, les hot-spots qu'elle utilise, il n'est pas nécessaire d'avoir beaucoup d'imagination pour savoir qui peut être intéressé par ce genre de métadonnées.

Peu conventionnelle également la présentation d'Alejandro Nolla, qui explique comment utiliser les grands serveurs de jeu que l'on trouve sur Internet pour conduire une attaque par amplification en déni de service distribué. La capacité de réponse de ces serveurs est telle que rien ne peut contenir leur puissance de « flood », et tout contribue à masquer l'attaquant.

Encore plus près de notre quotidien, l'analyse de MM Jagdish Acharya, James-Douglas Lefruit, Vincent Roca, Claude Castelluccia sur les fuites d'information des applications de la RATP parisienne. Si la carte Navigo est une référence en matière de sécurité RFID et de bonnes pratiques d'intégration, ce n'est pas franchement le cas des appliquelettes destinées aux usagers de smartphones. UDID, géolocalisation de l'utilisateur, parfois même le nom de l'utilisateur...le monde du développement d'applications en est, d'un point de vue sécurité, au même stade que l'était celui des applications Web il y a 10 ans, au début des premières campagnes de l'Owasp.

Hack in Paris, l'affaire Snowden avant l'heure

Hack in Paris est une « spin off » de la Nuit du Hack, probablement le plus grand CTF (concours de hacking) de France. Une spin-off plus sage, plus « rangée », s'adressant à des administrateurs réseau et développeurs. Et s'il ne fallait retenir qu'une seule chose de l'édition 2013, ce serait bien l'intervention d'Eric Filiol, professeur à l'ESIEA.

Une intervention intitulée "Le contrôle des technologies par les états-nations, passé, présent et futur" ou la cryptologie et la sécurité des systèmes d'information. Une bonne heure durant, à force d'arguments tant techniques que logiques, Eric Filiol a démontré à quel point l'espionnage sur Internet avait évolué, allant jusqu'à mettre en doute les outils de chiffrement mondialement utilisés par les entreprises.

Du programme de surveillance Echelon à Carnivore, en passant par des outils tels que Magic Lantern et autres projets connus de la CIA et de la NSA, l'arsenal de cybersurveillance nord-américain (UK-USA) a été passé au peigne fin. C'était plus de 3 mois avant que n'éclate le scandale Edward Snowden et l'aveu de compromission de certains outils vendus par RSA, ainsi que l'infection par des virus d'écoute de la majorité des dorsales numériques Euro-

péennes... que le contenu de celles-ci soit chiffré ou non. La conférence peut encore être vue sur Youtube (<http://bit.ly/1aNgZF>). Sans entrer dans les détails, il semble que les autorités US jouent avec une forme « d'obsolescence programmée » des outils de chiffrement, obsolescence provoquée par des erreurs d'intégration, involontaires ou non, la question reste posée. Une question qui en soulève une autre, celle de la totale dépendance des utilisateurs européens (et par conséquent Français), aux logiciels et équipements de sécurité provenant majoritairement d'Outre Atlantique.

Ce à quoi il faut ajouter une trop grande confiance dans les outils permettant de générer des aléas (notamment les processeurs de la sphère Intel/AMD) dont la perfection est très relative. Or, un générateur d'aléa faillible entraîne le risque d'un chiffrement prédictif.

Hackito, une bonne dose de matériel

S'il ne fallait retenir qu'une seule chose de HES 2013, ce serait son orientation résolument « hardware »... Enfin... pourrait-on presque ajouter. Car depuis ces 25 dernières années, les vulnérabilités, les attaques, les correctifs procédaient essentiellement (voir exclusivement) du domaine logiciel.

Youssef Touil, l'auteur de SDR-Sharp (<http://sdr-sharp.com/>) venait en compagnie de Benjamin Vernoux présenter une carte de hacking sans fil (émission et réception) très large bande, capable tant d'écouter (et de décoder) que d'émettre (replay attack, spoofing, DoSsing). HackRF est un véritable outil de pentesting couvrant de 30 MHz à 6 GHz sans trou. Qu'il s'agisse de l'identification d'un avion de ligne, d'une liaison Wifi, d'un lien entre un automate programmable et sa machine de supervision ou... de Yutu, le rover lunaire envoyé par la Chine et actif depuis décembre dernier : HackRF peut l'entendre et enregistrer les informations transmises, voir même interférer dans l'échange de données.

Comme pour confirmer ces propos, Adam Laurie, a fait une présentation intitulée « hacker les fréquences situées en dessous de 1 GHz » (<http://adamsblog.aperturlabs.com/2013/03/you-can-ring-my-bell-adventures-in-sub.html>). Passer de l'onde radio au chronodiagramme et à l'analyse de bus est simple, la plupart des outils étant gratuits, parfois même open source.

La démonstration de Laurie n'a porté que sur l'activation d'une sonnette de porte, preuve de fonctionnement totalement innocente, mais pouvant être étendue à d'autres portes... de voi-

tures de luxe par exemple. Ou de portiers sécurisés « par RFID ».

Et c'est encore au tour d'Adam Laurie, accompagné de Zak Franken cette fois, d'inquiéter l'assistance en décrivant la manière d'analyser les entrailles de circuits intégrés à coup d'acide fluorhydrique (l'un des plus dangereux qui soit et achetable sur eBay).

Tous les détails sur le blog de Franken pour la partie « acide » (<http://zacsblog.aperturlabs.com/2013/02/decapping-integrated-circuits-using.html>) et sur celui de Laurie pour l'aspect décodage de ROM (<http://adamsblog.aperturlabs.com/2013/01/fun-with-masked-roms.html>).

Certes, l'on pourrait considérer ce genre d'attaque comme purement livresque, exceptionnelle, voire totalement impossible à reproduire... mais la question mérite tout de même d'être posée à certains fabricants de téléphones cellulaires, de jouets, d'équipements automobiles de seconde monte dont les clones s'achètent déjà à quelques euros sur eBay.

Un genre de sport qui ne se pratique pas uniquement en Chine...

Ajoutons à ces interventions remarquées celles de Daniel Mende, qui a démontré comment « voler » à distance les photos d'un appareil photo Canon professionnel (et, au passage, espionner son propriétaire), Philippe Teuven qui effectuait une revue de détail des vulnérabilités réelles et imaginables autour des RFID (un spé-

cialiste travaillant pour le compte de NXP, ex-Philips, acteur important du monde « sans contact ») ou encore Glenn Wilkinson, sur le suivi et le flicage des porteurs d'équipement de mobilité (tablettes, téléphones...). Ah, si Orwell avait pu imaginer tout ça...

No Such Con : la French 0% Bullshit Conference.

Créée par un groupe d'anciens organisateurs de Hackito Ergo Sum, NoSuchcon se veut une conférence destinée aux professionnels (ou souhaitant le devenir) de la sécurité numérique. La première édition a véritablement tenu cette promesse, se plaçant, avec des intervenants de prestige, au niveau d'une Hack-in-the-Box, d'une CanSecWest ou d'une DefCon bonne cuvée. Nicolas Grégoire, chercheur Français, a donné un cours doctoral sur l'art du fuzzing intelligent autour du moteur XSLT (EXtensible Stylesheet Language), http://www.nosuchcon.org/talks/D1_04_Nicolas_Gregoire_XSLT_Fuzzing.pdf.

D'autres présentations, telle celle de Saumil Shah, laissaient entrevoir des champs de vulnérabilités monstrueusement indéfrichables. Avec son « pixel de la mort qui tue » (http://www.nosuchcon.org/talks/D1_05_Saumil_Deadly_Pixels.pdf), il prouve qu'il est envisageable de camoufler n'importe quel code exploitable et actif à l'intérieur d'une image. Et n'allons pas imaginer que l'auteur ait repris les veilles techniques de sté-



Nicolas Grégoire : fuzzing intelligent autour de XSLT (EXtensible Stylesheet Language).

ganographie : il « code » chaque bit à transmettre en fonction, par exemple, du degré de transparence de chaque pixel constituant une illustration. Un code java de lancement, et le virus est dans la place. Le principe peut être étendu à tout ce qu'un antivirus ne contrôle pas : fichier audio ou autre format directement géré et interprété par un navigateur html.

Luigi Auriema, le plus grand chasseur de faille au monde (détenteur du titre depuis plus de 10 ans) démontrait comment « hacker » les moteurs de jeux (consoles, jeux en ligne) qui peuvent aisément constituer d'importants réseaux d'attaque ou de communication occulte. Ce n'est qu'en décembre dernier que les autorités US ont entamé une campagne d'infiltration de ces plateformes pour savoir si, oui ou non, des groupements terroristes pouvaient tirer parti de ces voies de communication un peu particulières. Travis Goodspeed, le père des « GoodFET » (<http://goodfet.sourceforge.net/>), venait, quant à lui, décortiquer en public le «

reverse » d'un système à base de microcontrôleur. « Un microcontrôleur, c'est un système intégré, sans protection particulière (ASLR, DEP, No-Execute par exemple), utilisant généralement un socle de traitement connu, directement dérivé du 8051 ou du Z80, et dont les



Karsten Nohl : une approche algorithmique complexe pour le vol de voitures de luxe ...

GPIO peuvent s'analyser, dans le pire des cas, avec une simple LED, au mieux avec un analyseur logique coûtant moins de 50 euros. Ne parlons pas des protections souvent illusoires des fusibles « read only », des composants à la réf-

rence grattée ou des codes logés en ROM aisément lisibles une fois l'enveloppe du circuit intégré éliminée.

Karsten Nohl, un des gourous du hacking sans-fil d'Outre Rhin, expliquait quant à lui comment se plonger dans le « reverse » des clefs Hi-Tag2 de NXP. Ces clefs sont chargées d'émettre un code chiffré assez complexe permettant d'ouvrir la majorité des grosses berlines de fabrication Allemande, de Audi ou BMW par exemple. « *Tout ceci n'est que vanité de hacker*, concluait en substance Nohl, *après un exposé relativement technique. Dans la vraie vie, celle du vol de véhicule, on n'analyse pas de signal, on ne lance pas d'attaques contre un processeur. Souvent par hasard, le voleur découvre qu'une simple mise à la masse d'un fil alimentant le plafonnier du coffre ou un phare est capable d'initialiser le processeur chargé de la sécurité périmétrique. Nous atteignons là la limite qui sépare le domaine de la recherche pure de la découverte fortuite et de l'empirisme. Il est rare, très rare que les « méchants » s'inspirent de travaux des conférences de sécurité pour perpétrer leurs forfaits. La censure qu'exercent certains éditeurs sur les chercheurs les plus prolifiques, une forme de « sécurité par l'obscurantisme » ne diminue en rien ces risques d'attaque. Seule une information publique non biaisée et complète des risques liés à une technique ou une technologie assurera un renforcement des défenses* »



Paul Rascagnères, universitaire Luxembourgeois, présentait lors de Hackito Ergo Sum une bibliothèque de malware doublée d'analyses techniques sur ces codes.

Visual Studio 2013, les coulisses d'un lancement

Le lancement mondial de la dernière version de Visual Studio a été organisé le 13 novembre 2013, et nous avons eu la chance d'être invités et de participer à cet événement majeur de la fin d'année dernière. Comme à son habitude, Microsoft a fait les choses en grand, et l'on peut dire que les équipes « Corp », qui ont orchestré cette journée, ont réussi leur coup.



Tout d'abord, le lieu : New York, la ville qui ne dort jamais et qui électrise tous ceux qui s'y rendent. Ce fut notre cas. Le quartier ensuite : Meatpacking, le quartier à la mode du moment, qui doit son nom aux anciens abattoirs qui ont fait sa réputation.

Le lieu ensuite, un ancien abattoir - justement - reconverti en studio TV et offrant de nombreuses salles de retransmission et de conférences. Ce lieu idéalement choisi a été totalement personnalisé aux couleurs de Microsoft et plus particulièrement de Visual Studio.

Le parterre d'invités triés sur le volet était éclectique et on pouvait pêle-mêle rencontrer à la fois des partenaires (dont Cellenza) comme des clients, de toutes tailles et de toutes nationalités, puisqu'il s'agissait bien du lancement mondial. Ce lancement était d'ailleurs le point de départ des lancements nationaux qui ont suivi, la France ayant organisé son propre événement le 26 Novembre dans la foulée de celui de New York. La journée a démarré avec une session plénière diffusée sur Internet et en live, avec un site web dédié, et des speakers de renom (S. Somasegar, Brian Harry, Scott Guthrie, Scott Hanselmann). Ces présentations sont d'ailleurs disponibles sur Internet: <http://events.visualstudio.com>.

Nous avons ensuite pu assister à des sessions interactives en comité plus restreint. L'occasion de pouvoir s'entretenir directement avec quelques pointures (Scott Guthrie ou Brian Harry notamment dont il faut souligner la gentillesse et le professionnalisme) et de partager avec eux sur les évolutions des outils. Ce fut également un moment privilégié pour mieux

comprendre la façon dont les équipes Produits chez Microsoft travaillent, en mode « full agile » bien évidemment !

Après plusieurs sessions et tours de tables, la journée s'est achevée par un cocktail sur un Rooftop (toit-terrasse en langue de Molière !) qui nous offrait à la fois une superbe vue sur l'Empire State Building, mais aussi et surtout, l'occasion de découvrir nombre de solutions de partenaires, ainsi que la façon dont Visual Studio pouvait être mis en œuvre. J'ai personnellement particulièrement apprécié de voir fonctionner une imprimante 3D.



Fig.A

Le lancement a ensuite été relayé en France, dans les locaux de Microsoft à Issy Les Moulineaux, le 26 Novembre 2013.

Le « Visual Studio Paris Summit 2013 » a été un événement majeur de la communauté française, et s'est révélé être un franc succès avec plusieurs centaines de personnes présentes dans les locaux du Centre de Conférence de Microsoft. Brian Harry avait fait le déplacement pour nous présenter les évolutions de l'outil en personne.

Xamarin + Visual Studio 2013 = .NET, Android et iOS

Dans le cadre de la présentation de Visual Studio 2013, Microsoft a mis en avant un nouveau partenariat signé avec Xamarin (<http://xamarin.com>). Xamarin est une start-up créée après le rachat de Novell par Attachmate, dont l'objectif est de faire progresser le projet open source "Mono", qui fournit une implémentation de C# et du CLR cross-plateforme (<http://www.mono-project.com>), tout en proposant des solutions commerciales spécifiques (aujourd'hui dédiées aux plates-formes mobiles Android et iOS). L'intégration des outils de Xamarin au sein de Visual Studio n'est pas une nouveauté. Cependant, le fait que Microsoft s'associe à cette démarche est un message fort lancé à la communauté quant à la portée de cette suite d'outils de développement (<http://blog.xamarin.com/microsoft-and-xamarin-partner-globally>).

Visual Studio, en plus de permettre la création d'applications .NET pour Windows et Windows Phone, est en mesure de créer des logiciels pour les terminaux mobiles Android et iOS (Fig.A). Le tout avec l'approbation et le soutien de Microsoft. Xamarin précise d'ailleurs que l'utilisation conjointe des produits des deux éditeurs permet aujourd'hui de créer des applications pouvant être déployées sur 2,5 milliards de terminaux. Même Java ne peut se targuer de disposer d'une telle audience... En plus du support de Visual Studio 2013, les deux acteurs vont travailler de concert à l'évolution du langage C#, et à la mise au point de bibliothèques portables permettant de faciliter la création d'applications multiplateformes.



Jason De Oliveira
CTO | MVP C#
Cellenza - Software
Development Done Right
Son Blog :
<http://jasondeoliveira.com>



Avec Crosscut®, vous accédez en quelques clics à la bonne information, partout, tout le temps

Notre philosophie : la **SIMPLEXITÉ**, ou l'art de masquer la complexité de la technologie pour simplifier la vie de vos consommateurs et de vos collaborateurs.

➤ Digitaliser les usages

Crosscut® est une plateforme d'intégration qui permet d'acquérir, d'échanger et de scénariser l'information de manière fluide et efficace.

Vous reliez facilement les objets connectés du point de vente physique à vos solutions informatiques existantes (CRM, e-commerce, ERP, logiciel de caisse...).

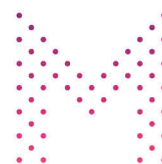
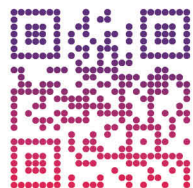
➤ Déplacer la technologie dans le Cloud

Crosscut® est le meilleur outil de la transformation de votre SI à demeure.

Hébergé dans le Cloud, Crosscut® crée un lien agile entre n'importe quel type d'applications, où qu'elles se trouvent dans le monde et quelles que soient les technologies utilisées.

Crosscut® : le plus court chemin vers l'intégration d'application

Nous contacter



MOSKITOS
Managed services. For real.

L'invasion des API



Trois lettres qui affolent l'univers informatique, les éditeurs et les entreprises. Comme si on avait inventé quelque chose de révolutionnaire. Et pourtant, non. L'API est tout sauf une nouveauté pour le développeur. Depuis 15 ans, nous en parlons tout le temps dans votre magazine préféré. Et pour cause, l'API est une des bases du modèle de développement. Une conférence est même organisée rien que pour elle : les API Days !

Il faut dire que ces petites bêtes sont partout. Le moindre objet a son API ou son kit de développement. Tous les services web proposent des API pour créer une intégration (une glue plutôt) avec son application, son site web... Ainsi pour utiliser les paiements PayPal et personnaliser l'affichage et les fonctionnalités, on passe par des API (en fait des WebAPI).

L'API pour les gouverner tous !

Saviez-vous que la Maison-Blanche (États-Unis) avait publié un document référence : « gouvernement numérique : construire la plateforme du 21e siècle pour mieux servir le peuple américain » ? Le document s'articule autour de 4 notions fortes : l'information, la plateforme partagée, l'utilisateur et la sécurité (et la vie privée). Les API, et particulièrement les WebAPI, tiennent une place importante dans la

première couche sur « l'information », au même titre que l'open data. Les WebAPI sont indispensables pour utiliser les données ouvertes et les contenus. Le gouvernement doit fournir les API (et donc les développer, les déployer, les maintenir). Et il est crucial d'utiliser des standards et une définition commune de l'API.

Mais attention, il ne faut pas non plus que ce marché en pleine explosion se transforme en foire où tout le monde fait n'importe quoi. Et c'est bien le risque ! Il faut que l'API soit bien formée, qu'elle réponde à un réel service et surtout, que l'entreprise ou l'éditeur en fasse la maintenance et son support. Car qui me dit que dans 12 mois, l'API Y existera encore ou si le modèle de l'API ne va pas changer dans 6 mois, etc. Et c'est bien là une des faiblesses de ce marché.

La question économique est importante pour certaines API. Où et comment monétiser l'API ? Le composant informatique ou les services utilisés ? Les possibilités sont nombreuses : paiement à l'usage, freemium, forfait, abonnement, etc.

Programmez ! vous propose de faire le point.

🔴 François Tonic

Pourquoi les API sont-elles importantes ?

Une API permet à un logiciel d'exposer tout ou partie de ses fonctionnalités de façon à être utilisé par d'autres programmes. Au sens strict, les API existent depuis que les programmes informatiques existent. Au cours du temps, elles ont revêtu une importance croissante, notamment avec l'avènement des architectures distribuées et des architectures « Orientées Services » (SOA).

Plus récemment, avec l'essor fulgurant des plateformes de réseau social, la montée en puissance de l'Internet des objets connectés, et la disponibilité d'un nombre de plus en plus important d'applications et de services sur le nuage, les API sont désormais omniprésentes. On peut donc dire aujourd'hui, qu'elles représentent un *lien agile* indispensable à la communication entre une application et le reste du monde.

Mais ce n'est pas tout.

Elles contribuent également au succès commercial de votre application et vous préparent aux grandes évolutions de demain.

L'API comme source de valeur

Concevoir les API d'une application implique une réflexion plus aboutie sur l'architecture logicielle, et la mise en place de processus de supervision de l'exécution. On citera notamment Steve Yegge qui, à l'occasion d'un mémorandum⁽¹⁾ très pertinent sur l'architecture « orientée services » et la notion d'API, recommande que toutes les équipes Google prennent en compte, dès la conception, la possibilité de rendre accessible tout nouveau produit à l'extérieur de l'entreprise via une API. Y compris pour un produit pourtant initialement destiné à un usage interne.

Mais, dans le même temps, Yegge insiste sur les précautions à prendre dans la gestion de la performance : prévoir une infrastructure de communication adéquate, définir une série d'indicateurs de mesures, et identifier clairement les points de contention.

Ce faisant, on constate que cette démarche contribue à l'amélioration de la qualité du logiciel, les équipes devant prendre en compte, notamment, les points suivants :

- ▶ Politique de quotas et de ralentissement (*throttling*) pour l'équilibrage de charge.
- ▶ Démarche d'assurance qualité la plus complète possible pour une supervision fine de l'exécution en production.

- ▶ Automatisation et normalisation des tests unitaires de l'API en isolation.

L'API comme facteur d'agilité

L'avènement du *Cloud Computing* et des applications « en tant que services » implique une véritable remise en question de l'ensemble du secteur de l'informatique et, tout particulièrement, des modes de consommation de l'information par les utilisateurs ou les clients.

Cette véritable « Révolution Industrielle Informatique », dont le concept est cher à Louis Naugès⁽²⁾, porte en elle une transformation de l'industrie du logiciel, qui va basculer d'un mode de travail qui repose sur le cahier des charges à un mode basé sur l'offre permettant des cycles beaucoup plus rapides.

Aujourd'hui, au grand bénéfice des sociétés de services et d'ingénierie en informatique, l'industrie du logiciel suit une démarche immuable depuis plusieurs dizaines d'années :

- 1) réunion d'un groupe de travail pour élaborer un cahier des charges qui traduira les exigences métiers, 2) sélection d'une équipe pour la mise en œuvre – au terme d'une procédure d'appel d'offres, généralement assez longue – 3) choix des outils de construction et, enfin, 4) construction de la solution.

Tout cela pour aboutir à la construction d'une solution logicielle « unique au monde », caractéristique dans cette approche que Louis Naugès qualifie de « monde préindustriel ».

Au contraire, la Révolution Industrielle Informatique consiste à promouvoir une démarche innovante – pourtant « universelle et normale dans tous les [autres] mondes industriels » – en donnant priorité à l'offre.

Sur la base des besoins, dont les grandes lignes sont exprimées clairement par les intervenants métiers, cette démarche consiste à placer la Direction des Services Informatiques comme un spécialiste de l'offre, c'est-à-dire un interlocuteur en capacité de connaître les différentes solutions disponibles sur le marché et proposer la plus à même de répondre aux besoins.

Le délai qui sépare l'expression des besoins à la mise en œuvre d'une solution est souvent considérablement réduit, comparé à la démarche décrite précédemment. Il s'exprime souvent en quelques semaines, et non pas en plusieurs mois.

Et l'API dans tout cela ?

L'API est la condition *sine qua non* à l'adoption de cette démarche et à la réussite commerciale de votre passage au *Cloud Computing*, car c'est elle qui vous permet d'exposer la valeur ajoutée de votre produit ou de votre application sur le nuage.

De nouvelles sources de valeurs s'offrent alors à vous :

- ▶ Gain de productivité : développer une fois l'API permet à votre produit d'être réutilisé plusieurs fois dans des contextes différents, augmentant ainsi sa valeur commerciale.
- ▶ Partenariats innovants : avoir un produit accessible par API rend plus simple les logiques de collaborations économiques entre fournisseurs de services et d'applications sur le nuage. Par exemple, Moskitos et notre partenaire, Captain Pass, mettent en commun leur solution pour proposer des services innovants à leurs clients.
- ▶ Composition : mettre à disposition la valeur ajoutée contenue dans votre produit ou votre application à l'extérieur, permet à vos clients de créer une « application composite » dont la valeur ajoutée est plus grande que la somme des valeurs ajoutées des applications qui la composent.



● Bertrand Masson –
Founder @Moskitos.
Directeur de Moskitos.
<http://maximelabelle.wordpress.com>
Twitter : @Bertrand_Masson



● Maxime Labelle –
Managed Services Architect @Moskitos.
Responsable de l'usine logicielle chez Moskitos.
<http://jeremiedevillard.wordpress.com>
Twitter : @jeremiedev



● Jérémie Devillard –
Managed Services Architect @Moskitos.
Responsable de la Recherche et du Développement chez Moskitos.
<http://maximelabelle.wordpress.com>
Twitter : @SpringComp

(1) Steve Yegge's Google Platforms Rant. Via Rip Rowan - <http://bit.ly/1hCu2EH>

(2) D'une démarche pilotée par la demande à une approche pilotée par l'offre : une rupture rendue possible par la R2I, Révolution Industrielle Informatique. Louis Naugès - <http://bit.ly/1euee1v>

APIGILITY : créez vos API en quelques clics

En juin 2013, Matthew Weier O'Phinney, lead-developer sur le framework Zend, était en discussion avec Andi Gutmans et Zeev Suraski sur ce qui pourrait et/ou devrait être inclus dans le Zend Framework 3. Ils ne désiraient pas réinventer une énième version du pattern MVC, ne voulaient pas sur-complexifier le framework. Ils se demandaient ce dont l'industrie, les développeurs avaient besoin à l'heure actuelle. La réponse est naturellement venue : des outils.

Dans un contexte de développement applicatif web de plus en plus orienté « API Centric », « Mobile First » et « Agile Delivery », le développement d'API se révèle complexe en mise en place et chronophage en maintenance.

Pour avoir moi-même, en tant que consultant architecte, travaillé pendant ces 2 dernières années sur des problématiques émergentes de mise en place d'API pour différents comptes sur Paris, j'ai constaté que beaucoup de développeurs et responsables techniques, en fonction de leur sensibilité, cursus, expérience, avaient différentes approches sur comment devait être construite une API. Au-delà de l'approche structurelle, beaucoup de questions restaient en suspens : quel format de réponse préférer, XML ou JSON ? Comment gérer les erreurs de requête et/ou de réponse ? Comment appréhender les requêtes HTTP ? Comment formater la réponse ? Quelle méthode/protocole choisir pour la sécurité et les autorisations d'accès ?

Apigility vous permet de rapidement et efficacement construire et définir vos propres API RESTful visuellement, et de les exposer pour des applications, services tiers, ou bien même de manière publique pour différentes audiences. Présenté officiellement lors de la ZendCon 2013 en octobre dernier, Apigility est actuellement en cours de développement. A l'écriture de cet article en janvier, la dernière version stable était la 0.8. Les développements sont très actifs, et la version 1.0 est attendue dans les semaines à venir.

Installation et configuration

Comme de nombreux projets open-source, Apigility est disponible publiquement sur Github. L'équipe de développement propose une application basée sur Zend Framework 2 et AngularJS, appelée « ZF Apigility Skeleton ». Il vous est possible de l'installer localement de trois manières :

- ▮ une version archivée désynchronisée (zip, tar).
- ▮ un « clone » du dépôt via Git.
- ▮ via Composer.

Nous allons utiliser cette dernière méthode, qui présente l'avantage d'être simple et automatique.

Ouvrez votre terminal, naviguez à la racine où vous désirez que votre projet soit installé, et tapez :

```
curl -sS https://getcomposer.org/installer | php

php composer.phar create-project -sdev zfcampus/zf-apigility
-skeleton myapi
```

- ▮ « create-project » permet de cloner le projet passé en paramètre depuis sa correspondance sur packagist, puis de déclencher une commande « composer install » dessus.
- ▮ « sdev » quant à lui précise le niveau minimum de stabilité, ici « dev ». En effet, Apigility ne dispose à ce jour pas encore d'une version stable.

- ▮ « myapi » ou quelque paramètre suivant l'url du dépôt détermine le nom du dossier dans lequel sera contenu le projet. Si omis, c'est le nom canonique du dépôt qui sera utilisé.

Composer installe le Zend Framework 2 dans sa dernière version de développement ainsi que 19 modules ZF2, dont 12 propres au projet Apigility. Nous reviendrons sur chacun d'entre eux au cours de cet article.

Pour pouvoir lancer l'interface de gestion d'Apigility, vous devez configurer le projet en « mode développement ». Pour ce faire, toujours en ligne de commande, tapez :

```
cd myapi
php public/index.php development enable
```

Cette commande va dupliquer le fichier de configuration myapi/config/development.config.php.dist et ainsi activer trois des douze modules préalablement installés :

- ▮ ZFTool : permet de gérer les modules ZF2 (création et configuration de modules, contrôleurs, actions...)
- ▮ ZF\Apigility\Admin : fournit l'interface de gestion des API du projet
- ▮ ZF\Configuration : ajoute des fonctionnalités à ZFTool pour la gestion des modules

Apigility nécessite PHP 5.4.8+. Puisqu'il génère des fichiers de configuration plutôt que de stocker celle-ci dans une base de données, il est recommandé de désactiver le cache OpCode ou APC sur votre système pendant que vous travaillez avec l'interface d'administration. Maintenant que le projet est installé et accessible, vous avez 2 possibilités :

- ▮ Définir un hôte virtuel sur votre serveur web existant. Il faudra alors définir le DocumentRoot sur le dossier « public ».
- ▮ Démarrer le projet grâce au serveur intégré à PHP (à ne pas utiliser en production !)

Nous allons utiliser ce dernier. Toujours dans le dossier racine de votre projet (myapi), tapez :

```
php -S 0:8080 -t public public index.php
```

Lancez ensuite votre navigateur sur <http://localhost:8080>, vous arrivez sur la page d'accueil d'Apigility. Un clic sur « Get started » vous permet d'accéder au dashboard, duquel vous avez la possibilité de créer et configurer vos API [Fig.1](#).

Création d'une API

Pour exposer vos données via une API, la première étape est de créer un adaptateur qui permet à cette API de se connecter à la source de données. De multiples adaptateurs sont disponibles au sein d'Apigility, dont les différentes déclinaisons de PDO. Pour notre exemple, nous utiliserons SQLite, avec la structure et les données suivantes :

```
CREATE TABLE user (
    id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
    firstname TEXT,
    lastname TEXT,
    email TEXT NOT NULL,
    created_at TEXT NOT NULL
);

INSERT INTO user (firstname, lastname, email, created_at)
VALUES ("Matthew", "Weier O'Phinney", "matthew@mwop.net", "2013-12-20");

INSERT INTO user (firstname, lastname, email, created_at)
VALUES ("Enrico", "Zimuel", "enrico@zimuel.it", "2014-01-02");

INSERT INTO user (firstname, lastname, email, created_at)
VALUES ("Rob", "Allen", "rob@akrabat.com", "2014-01-13");

INSERT INTO user (firstname, lastname, email, created_at)
VALUES ("Ralph", "Schindler", "ralph@ralphschindler.com", "2014-01-13");
```

Enregistrez cette structure au format SQLite sur votre disque, idéalement au sein de votre projet. Nous utiliserons le chemin `myapi/data/db/team.db`.

Astuce : sur un OS *nix, enregistrez ce fichier au format SQL, par exemple `myapi/data/db/team_sql.sql` puis saisissez dans un terminal dans le dossier `myapi/data/db` la commande suivante :

```
cat team_sql.sql | sqlite3 team.db
```

Pour gérer vos fichiers sqlite, je vous conseille l'utilitaire SQLite Database Browser, un projet open source multi-plateformes disponible à l'adresse <http://sourceforge.net/projects/sqlitebrowser>

Depuis le dashboard d'Apigility, naviguez depuis le menu de gauche dans « Database Adapters » puis cliquez sur Create New DB Adapter. Renseignez le nom (DB\Team), le type (Pdo_Sqlite), puis le chemin complet du fichier `team.db`, les autres paramètres étant inutiles pour une base SQLite. Validez en bas à droite, vous retrouvez votre "adapter" dans la liste.

En haut à droite de l'interface, cliquez sur « Create new API », puis renseignez son nom. Choisissez un nom en un seul mot et sans caractères spéciaux. Ce nom servira d'espace de noms au sein du code pour votre API ainsi que pour sa négociation de contenu.

Après avoir validé, un accès à cette API est disponible depuis la navigation sous le logo, un module ZF2 est créé avec le nom saisi et est activé dans l'application globale. (`myapi/config/application.config.php`) L'interface est redirigée vers sa page de configuration. Depuis cette page, vous pouvez :

- Créer des services REST et RPC
- Gérer les versions de votre API (qui par défaut est à 1)

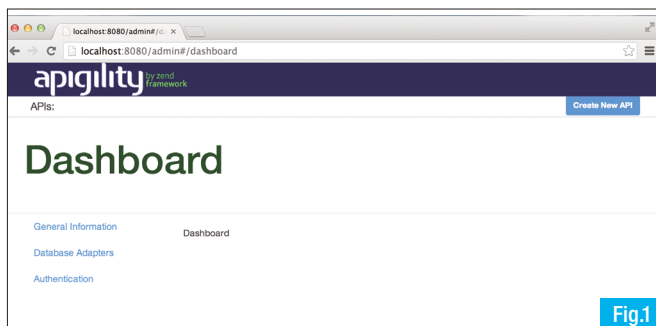


Fig.1

- Définir des autorisations d'accès Fig.2

En regardant les fichiers générés par cette création, vous constatez que ce module contient une configuration vide, et une architecture de dossiers qui regroupera les services que vous créerez.

(`myapi/module/Team`) Fig.3

Création d'un service RPC

RPC est l'acronyme de « Remote Procedure Call ». Ce protocole standardisé RFC (<http://tools.ietf.org/html/rfc707>) peut être considéré comme l'ancêtre de REST. Il permet simplement des appels de fonctions dans un modèle client-serveur.

Apigility permet d'implémenter ce protocole au sein de notre API, via son module « `zf-rpc` ». Nous en donnerons un exemple minimaliste.

Depuis le menu sur la gauche de notre API, cliquez sur « RPC Services » puis sur « Create New RPC Service ». Saisissez « Registration » pour le nom et « `/register` » pour la route. Validez.

Dans l'onglet « Source code », vous remarquez qu'une classe « `RegistrationController` », héritant de « `AbstractActionController` », qui est le contrôleur de base du MVC de ZF2, a été créée.

Ouvrez ce fichier avec un éditeur de texte et ajoutez l'instruction dans la méthode `registrationAction` comme suit :

```
RegistrationController.php
1 <?php
2 namespace Team\V1\Rpc\Registration;
3
4 use Zend\Mvc\Controller\AbstractActionController;
5
6 class RegistrationController extends AbstractActionController
7 {
8     public function registrationAction()
9     {
10         return ['message' => 'Registrations are closed for now.'];
11     }
12 }
```

Un test en ligne de commande via :

```
curl --header "Accept:application/json" http://localhost:8080/register
```

Donne la réponse :

```
{"message":"Registrations are closed for now."}
```

Création d'un service REST

Toujours dans le menu latéral, cliquez sur « REST Services », puis sur « Create New REST Service ». Deux choix s'offrent à vous :

- « Code-Connected » : l'interface générera des « stubs », c'est-à-dire des squelettes de classes dont les méthodes seront vides de toute

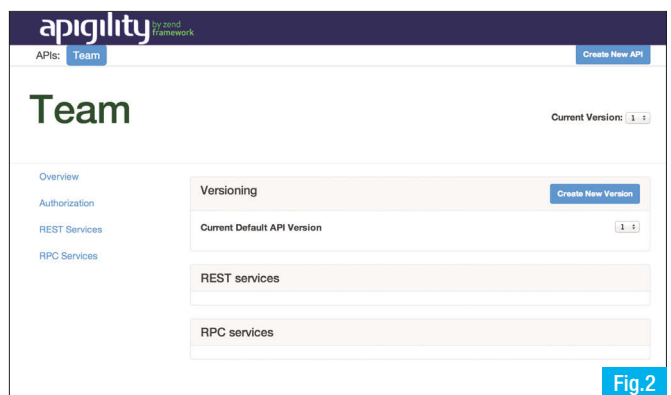


Fig.2

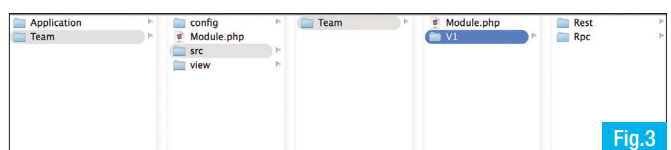


Fig.3

fonctionnalité, mais dont la structure sera configurée pour répondre à celle d'un service REST. Cette approche vous offre une grande liberté pour exposer vos ressources avec votre propre logique d'implémentation, voire d'inclure du code fonctionnel existant.

► « DB-Connected » : cette méthode de génération de service REST vous permet de rapidement et simplement travailler avec votre source de données relationnelles existantes. Elle se base sur le motif de conception « Table Data Gateway » de Zend Framework 2 pour assurer les requêtes en base de données et retourner les résultats.

Une définition simple de ce motif pourrait n'être qu'*un objet qui assure la liaison entre le code et une table dans la base de données*. Cette instance gère les lignes de cette table.

Choisissez l'onglet DB-Connected, puis sélectionnez l'adaptateur précédemment créé dans la liste et saisissez le nom de la table que vous désirez utiliser (user pour notre exemple) pour ce service. Validez. Vous le retrouvez dans la liste des services REST. Vous devriez lire « Team\V1\Rest\User\UserEntity ». Cliquez ensuite sur ce nom, vous accédez à la configuration de ce service. De là, un peu de théorie s'impose pour les néophytes...

Comme évoqué en préambule, Apigility vous permet de construire vos API RESTful. Derrière ce terme se cache un modèle d'architecture émergeant sur les webservices : REST, acronyme de REpresentational State Transfer, n'est pas un standard en soit, puisqu'il n'existe pas de spécifications, mais plutôt une normalisation, se basant elle-même sur des standards existants :

- URI : syntaxe qui permet à une ressource d'être adressable
- HTTP : protocole sans état avec des opérations disponibles (GET, POST, PUT, PATCH, DELETE, OPTIONS)
- HAL : Hypermedia Application Language, représentation du contenu et navigation au sein de l'application. Nous y reviendrons plus tard.
- MIME : types et sous-types de contenu (text/html, image/jpeg, application/json, etc.)

Pour résumer, « consommer » une API RESTful revient à spécifier une ressource via une URI sur une méthode HTTP qui retourne une réponse au format HAL pour un MIME donné.

REST est le style d'architecture. RESTful est l'adjectif apposé aux API respectant l'ensemble des standards définis ci-dessus.

Exemple :

```
curl --header "Accept:application/json" https://api.github.com/repos/zfcampus/zf-apigility-skeleton
```

Cela nous donne accès en lecture (HTTP) aux informations (HAL) du projet zf-apigility-skeleton (URI) sur Github au format JSON (MIME). Apigility offre la possibilité de configurer chacun de ces paramètres et ainsi de créer vos API RESTful.

Configuration de l'API

De retour au service REST User, 4 onglets de configuration sont disponibles. Le premier propose un résumé basique/avancé de la configuration de base de notre service. Le quatrième permet de voir les fichiers générés pour le service : UserCollection (qui représente une collection d'entités user) étend Zend\Paginator et UserEntity (qui représente un objet UserEntity) étend simplement ArrayObject.

Le second permet de modifier tout ou partie de cette configuration.

Parmi les nombreux paramètres, modifiez la route comme suit :

/users/:user_id] et définissez la pagination à 2. Décochez les méthodes pour n'accepter que GET pour les entités et collections. Laissez les autres paramètres tels quels. Validez en bas de page. Nous reviendrons sur le troisième onglet un peu plus loin.

Afin de pouvoir tester notre API, différentes options sont disponibles. Le

besoin est de pouvoir faire des requêtes HTTP avec la possibilité de spécifier des en-têtes et le corps. Je vous propose d'utiliser une excellente extension pour le navigateur Google Chrome appelée « Postman ».

Après l'avoir installée (depuis le Chrome Webstore), précisez l'URL <http://localhost:8080/users> en laissant les autres paramètres vides puis cliquez sur « Send ». Vous devriez voir le résultat s'afficher en dessous Fig.4.

La réponse contient la collection automatiquement paginée de tous les utilisateurs présents dans la base SQLite. Cette réponse de collection contient deux parties :

- Une partie « _links » détaillant les liens de pagination (courant, suivant, premier, dernier)
- Une partie « _embedded » contenant la liste des entités. Chacune de ces entités contient les champs, ainsi que le champ « _links / _self ». Mais d'où vient ce format ?

Le format HAL

Ce format ouvert, acronyme de Hypermedia Application Language, décrit de façon générique une structure pour les ressources RESTful, complétant ainsi le niveau 3 du « Richardson Maturity Model » (<http://martinfowler.com/articles/richardsonMaturityModel.html>). Il assure que chaque ressource contienne des liens relationnels, et que chaque ressource intégrée soit une structure standard, identifiable et adressable. HAL peut être interprété via deux types, XML et JSON.

Pour les ressources JSON, le paramètre minimal à fournir est la propriété « _links » contenant un lien relationnel « self ».

Pour plus d'informations, visitez <http://tools.ietf.org/html/draft-kelly-json-hal-06>

Le module « zf-hal » d'Apigility est en charge de générer le rendu des résultats au format HAL via une stratégie de réponse dédiée (voir à ce sujet : <http://framework.zend.com/manual/2.2/en/modules/zend.view.quickstart.html#creating-and-registering-alternate-rendering-and-response-strategies>).

Ce module génère les liens relationnels, le contenu des ressources et l'en-tête « application/hal+json ».

Son paramétrage est fonction de la configuration du service REST, que vous retrouvez dans myapi/config/module.config.php, à la clé « zf-hal ».

```
'zf-hal' => array(
    'metadata_map' => array(
        'Team\V1\Rest\User\UserEntity' => array(
            'route_identifier_name' => 'user_id',
            'entity_identifier_name' => 'id',
            'route_name' => 'team.rest.user',
```

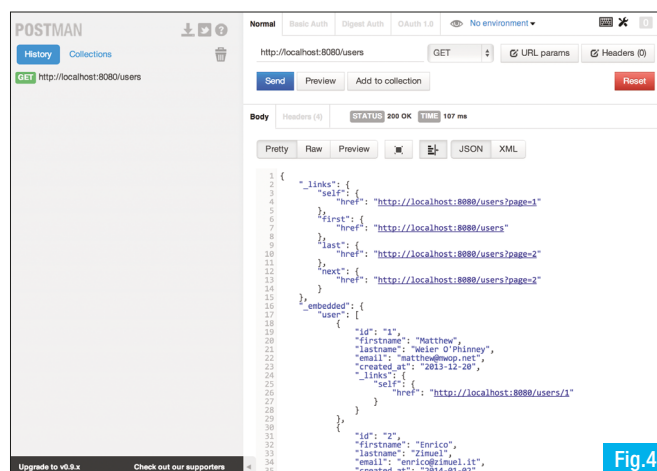


Fig.4

Ce faisant, vous créez une chaîne de validation dont vous pouvez, ainsi que pour les filtres, redéfinir l'ordre d'exécution en déplaçant leurs blocs. Pour le champ « created_at », cochez « Required? ».

Ajoutez :

- Un validateur « notEmpty » avec les options « breakchainonfailure » à « true » ainsi que « message » à « creation date is mandatory ».
- Un validateur « regex » avec l'option « pattern » à « `/^[0-9]{4}-(0[1-9]|1[0-2])-(0[1-9]|[1-2][0-9]|3[0-1])$/` »
- Deux filtres stringtrim et striptags.

Dans Postman, définissez l'URL sur <http://localhost:8080/v2/users>. Ajoutez une en-tête « Content-Type: application/json » puis sélectionnez l'onglet « raw ». Dans le corps de l'objet de requête, saisissez puis validez :

```
{
  "firstname": "Jane",
  "lastname": "Doe",
  "email": "toto"
}
```

L'email est erroné et la date de création omise. Le retour est « 422 / Unprocessable entity ». En effet, les règles de validation telles que définies précédemment ne sont pas respectées.

Vous remarquez que les messages personnalisés apparaissent pour chaque règle et que le reste de la réponse suit le schéma du module « zf-api-problem ».

La correction des champs en email : jane@doe.com et created_at : 2014-01-14 renvoie une réponse 201 / Created avec l'objet correspondant.

Authentification

Depuis l'écran d'accueil d'Apigility, vous avez la possibilité de définir la méthode d'authentification que vous souhaitez implémenter au sein de votre API.

Trois méthodes vous sont proposées :

- HTTP Basic Authentication : vous fournissez un nom d'utilisateur et un mot de passe, en clair, dans vos en-têtes HTTP. Rapide et simple à implémenter, mais il est fortement recommandé d'ajouter une couche SSL sur ces échanges.
- HTTP Digest Authentication : l'authentification nécessite deux requêtes, la deuxième étant cryptée avec un hash du nom d'utilisateur, du mot de passe et du « realm ». Plus sécurisé que la version « basic » même sans SSL, mais pour chaque appel, deux sont nécessaires. Il existe également des points de vulnérabilité.

Fig.5

► OAuth2 : protocole introduit en 2012 par la RFC

(<http://tools.ietf.org/html/rfc6749> et <http://tools.ietf.org/html/rfc6750>) et utilisé par de nombreux fournisseurs (Facebook, Twitter, Google, Github, Outlook.com, etc...). Il permet de donner à des utilisateurs la possibilité de s'authentifier auprès d'un site (consumer) via le service d'un autre site (provider) via différentes méthodes.

OAuth2 étant de plus en plus utilisé dans l'industrie et offrant de nombreuses fonctionnalités en fonction du contexte d'utilisation, nous l'utiliserons pour notre exemple.

Apigility utilise pour cela le module PHP développé par Brent Shaffer (v) et y ajoute un module permettant de l'interfacer avec vos API via « zf-oauth2 ».

Première étape nécessaire à l'utilisation d'OAuth2 comme protocole d'authentification de votre API, mettre en place et définir la structure des données de son implémentation. (A date, ces étapes sont manuelles. L'équipe de développement et les contributeurs étudient la meilleure façon de pouvoir inclure ces étapes dans l'interface d'administration) Vous trouverez dans le dossier myapi/vendor/zfcampus/zf-oauth2/data la source SQL pour une base MySQL.

Incluez ces instructions SQL dans un fichier au format SQLite.

```
CREATE TABLE oauth_access_tokens (
  access_token VARCHAR(40) NOT NULL,
  client_id VARCHAR(80) NOT NULL,
  user_id VARCHAR(255),
  expires TIMESTAMP NOT NULL,
  scope VARCHAR(2000),
  CONSTRAINT access_token_pk PRIMARY KEY (access_token)
);

CREATE TABLE oauth_authorization_codes (
  authorization_code VARCHAR(40) NOT NULL,
  client_id VARCHAR(80) NOT NULL,
  user_id VARCHAR(255),
  redirect_uri VARCHAR(2000),
  expires TIMESTAMP NOT NULL,
  scope VARCHAR(2000),
  CONSTRAINT auth_code_pk PRIMARY KEY (authorization_code)
);

CREATE TABLE oauth_clients (client_id VARCHAR(80) PRIMARY KEY,
  client_secret VARCHAR(80), redirect_uri VARCHAR(2000), grant_types VARCHAR(80), scope VARCHAR(2000), user_id VARCHAR(255));

CREATE TABLE oauth_jwt (
  client_id VARCHAR(80) NOT NULL,
  subject VARCHAR(80),
  public_key VARCHAR(2000),
  CONSTRAINT client_id_pk PRIMARY KEY (client_id)
);

CREATE TABLE oauth_refresh_tokens (
  refresh_token VARCHAR(40) NOT NULL,
  client_id VARCHAR(80) NOT NULL,
  user_id VARCHAR(255),
  expires TIMESTAMP NOT NULL,
  scope VARCHAR(2000),
  CONSTRAINT refresh_token_pk PRIMARY KEY (refresh_token)
);

CREATE TABLE oauth_scopes (
  type VARCHAR(255) NOT NULL DEFAULT "supported",
  scope VARCHAR(2000),
  client_id VARCHAR(80)
, `is_default` TINYINT(1) NULL DEFAULT NULL);
```

```
CREATE TABLE oauth_users (
    username VARCHAR(255) NOT NULL,
    password VARCHAR(2000),
    first_name VARCHAR(255),
    last_name VARCHAR(255),
    CONSTRAINT username_pk PRIMARY KEY (username)
);
INSERT INTO oauth_clients VALUES('team-api','','/oauth/receive
code','password',NULL,NULL);
INSERT INTO oauth_users VALUES('coolguy','$2y$14$cbiQsiBq.3
715KAC7CgKBuGLXlpsinHolq0bO3T1hRHZWsLJibP66','Cool','Guy');
```

Les 2 dernières lignes ajoutent un client public à notre API (pas de client_secret) qui a le droit de s'identifier avec le workflow « password » ainsi qu'un utilisateur « coolguy » avec le mot de passe « superpass » encrypté via BCrypt. Cet algorithme d'encryptage est actuellement reconnu comme une méthode sûre par rapport aux méthodes de hachage telles que MD5 ou SHA1. Voir <http://codahale.com/how-to-safely-store-a-password> pour plus d'informations.

Cliquez donc dans la partie « Authentication » sur « Setup OAuth2 Authentication » et spécifiez le chemin absolu du fichier SQLite en préfixant ce chemin par « sqlite: ». Vous pouvez préciser la route sur laquelle pointeront les requêtes d'authentification. Par défaut, elle sera « /oauth »

La source d'authentification OAuth2 mise en place, vous pouvez désormais définir les règles d'autorisation d'accès à vos API, et ce par service et par méthode HTTP. Naviguez jusqu'à votre API « Team » dans le menu « Authorization ».

Vous allez restreindre, par autorisation, l'accès au service « user » en lecture seule. Cochez la colonne « GET » qui présélectionnera tous les contrôleurs définis puis sauvegardez en bas à droite.

Dans Postman, faites une requête sur la version de 2 du service « user » de l'API Team. Comme attendue, vous recevez une réponse 403 / Forbidden.

Afin d'obtenir l'autorisation de lecture, vous devez obtenir auparavant un jeton d'accès. Vous avez la possibilité d'utiliser différents flux d'autorisation (client_credential, authorization_code, etc...)

Dans un flux par mot de passe comme celui que nous allons utiliser, vous avez besoin d'un nom d'utilisateur, un mot de passe et de l'identifiant du client via lequel vous demandez l'accès (nous avons défini l'API comme publique, donc sans client_secret).

Définissez dans Postman une nouvelle requête, sur <http://localhost:8080/oauth/en> « POST », avec comme en-têtes « Accept: application/json » et « Content-Type: application/json ». Dans le corps « raw », ajouter les paramètres suivants :

```
{
  "username": "coolguy",
  "password": "superpass",
  "grant_type": "password",
  "client_id": "team-api"
}
```

Vous obtenez une réponse 200 / OK avec comme contenu (par exemple) :

```
{
  "access_token": "7bdceddbc5a054ced630ad5a3f4cec043a1a1d4b",
  "expires_in": 3600,
  "token_type": "Bearer",
  "scope": null,
```

```
  "refresh_token": "d8a3c87c421d6c1afbb0e631b6e64d9fc10afb47"
}
```

Votre jeton d'accès (access_token) vous autorise désormais l'accès aux services de l'API, et ce pour 1h (3600 secondes).

Pour le rafraîchir, faites une requête « POST » sur la même URL avec les paramètres suivants :

```
{
  "refresh_token": "d8a3c87c421d6c1afbb0e631b6e64d9fc10afb47",
  "grant_type": "refresh_token",
  "client_id": "team-api"
}
```

Ce jeton d'accès est de type « Bearer ». Pour l'utiliser, ajoutez l'en-tête « Authorization: Bearer 7bdceddbc5a054ced630ad5a3f4cec043a1a1d4b » à une requête sur le service V2 User. Celle-ci retourne une collection d'entités User [Fig.6](#).

Version 1.0

Au-delà de l'amélioration de la stabilité globale et de la correction de bugs résiduels, les fonctionnalités restant à développer jusqu'à la version 1.0 porteront principalement sur la documentation applicative et sur l'export de la documentation des API configurées. Les systèmes existants en cours d'étude sont RAML (<http://raml.org>), Swagger (<http://developers.helloverb.com/swagger>) et API Blueprint (<http://apiblueprint.org>). Un lifting de l'interface est également prévu.

Un dernier mot concernant Doctrine. Tom Anderson a beaucoup travaillé sur la partie serveur de l'implémentation de Doctrine pour Apigility (ORM & ODM). Vous pouvez consulter l'avancement de ses travaux sur <https://github.com/soliantconsulting/zf-apigility-doctrine>. Je commence à collaborer avec lui sur l'intégration des fonctionnalités au sein de l'interface d'administration. « zf-apigility-doctrine » sera intégré dans l'admin aux horizons de la version 1.0, voire 1.1.

Participez !

Apigility est un projet open-source. A ce titre, toute contribution est la bienvenue via Github sur <https://github.com/zfcampus>

Un forum IRC de discussion sur les développements est disponible sur le canal #apigility-dev.

A vos claviers !

 **Julien Guittard**
Consultant architecte formateur PHP/Zend
Zend Certified Engineer
<http://twitter.com/julienguittard>

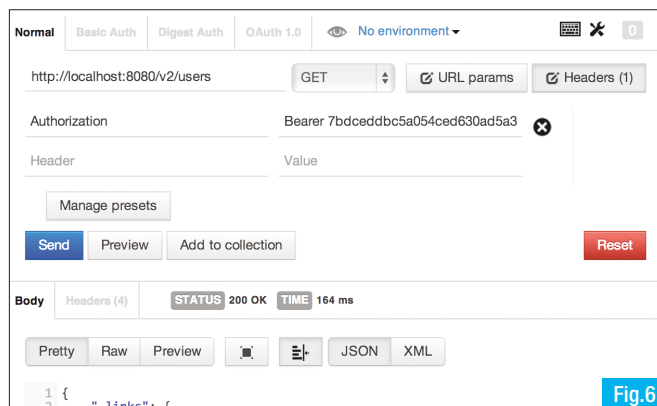


Fig.6

API : créer, déployer, gérer

Nous allons tenter au travers de cet article de faire toute la lumière sur la notion d'API. Nous allons, dans un premier temps définir cet acronyme et lui donner un sens, ensuite nous verrons la démarche à suivre pour bâtir une API. Puis, nous aborderons également les différentes problématiques qu'il faut adresser pour déployer, gérer et maintenir une API. Enfin, nous présenterons les Marketplaces.

API : APPLICATION PROGRAMMING INTERFACE

Une API est une interface de programmation. Cette traduction littérale lève déjà un peu le voile. L'objectif d'une API vise à assurer l'interopérabilité entre un système qui fournit le service et un autre qui se positionne en tant que consommateur de ce dernier. Pour mettre en œuvre cette interopérabilité, une API fournit un ensemble de classes, fonctions ou procédures.

Historique

C'est dans les années 1980 que les premières API ont fait leur apparition. À l'origine, elles étaient principalement utilisées au sein des architectures logicielles et hardware. Elles assuraient les échanges entre les différents composants. Un exemple éloquent est le modèle de communication OSI (1984). Pour rappel, OSI est un modèle de communication entre ordinateurs. Il définit une architecture en couches. Chaque couche a un rôle bien précis et communique à l'aide de primitives de service. Dans cet exemple, les primitives de service jouent le rôle d'API. Avec l'explosion du web et plus récemment du Cloud computing, les « Modern Web APIs » ont fait leurs apparitions. Leur histoire est plus récente, une dizaine d'années.

API vs bibliothèque vs Framework

Par abus de langage, il arrive que ces trois termes soient employés les uns à la place des autres. Parfois la frontière entre une API et une bibliothèque est très fine. C'est pour cette raison que nous allons explorer brièvement la sémantique qui se cache derrière chacun de ces termes afin

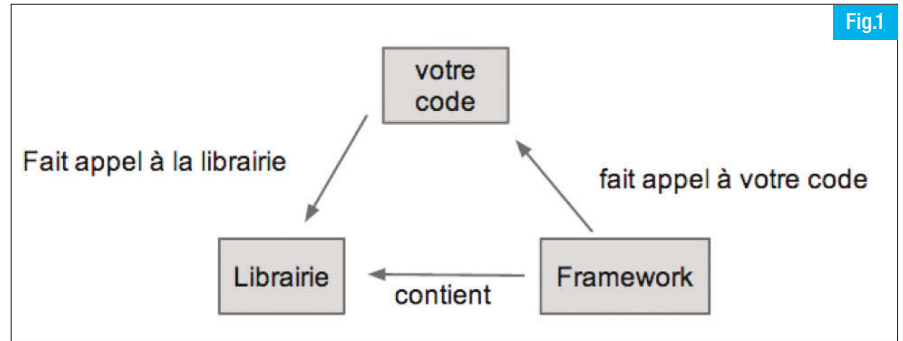


Fig.1

de mieux comprendre les tenants et les aboutissants de chacun. Le terme qui se détache parmi les trois est le framework. Je pense qu'il présente le moins de confusion. Comment pourrait-on définir le terme de framework ? La définition dans sa version la plus simple est qu'un framework offre un cadre de développement standard à un besoin technique (et/ou fonctionnel) précis. Il est composé de plusieurs librairies ayant chacune une responsabilité qui lui est propre. Au sein d'un logiciel, le framework imposera au développeur l'utilisation de certains patterns. De par son côté structurant, il permet d'améliorer la productivité de travail. Lorsqu'on compare les termes « librairie » et « framework », on évoque « l'inversion de contrôle ». Dans une application, le framework aura un rôle actif, c'est lui sera à l'initiative des interactions avec le code créé par le développeur. Les mécanismes mis en place, en règle générale, pour leurs interactions sont :

- Fonction de callback,
- Gestion d'événement,
- Interface.

La librairie, quant à elle, a un rôle passif, les interactions avec l'application dans laquelle elle est utilisée sont à l'initiative du développeur. Une librairie est un ensemble de classes, de fonctions, permettant d'adresser une problématique précise. Le principe d'une librairie est d'éviter aux développeurs d'avoir à recréer des choses qui existent déjà. Vous avez sûrement déjà entendu le célèbre adage : « ne pas réinventer la roue ». Aujourd'hui, on peut trouver une multitude de librairies adressant des problématiques très variées. Une représentation graphique permettant de mieux comprendre la relation entre une librairie, votre code et un framework Fig.1. Maintenant que nous avons une vision plus claire des termes, il est plus facile de comprendre les différences. Une API a pour vocation d'exposer (de façon interne ou externe) des données ou des fonctionnalités à un tiers. Contrairement aux frameworks et aux librairies, une API sera une interface permettant d'accéder à un service ou une ressource tierce.

Les différents types d'API

Les API peuvent être classifiées, selon une série de critères, en plusieurs types. Dans ce qui suit, seules les classifications techniques et fonctionnelles seront détaillées.

Classification technique

La classification technique se focalise sur l'aspect technique des API, autrement dit, la manière selon laquelle elles vont être exposées, indépendamment de la fonctionnalité qu'elles vont proposer :

- **Web** : une API web est consommée par un client léger. Le mode de communication se base généralement sur le protocole HTTP (soit REST, XML, clef/valeur...). Les exemples sont nombreux : les API twitter, google plus, Facebook...
- **Native** : une API native est une API qui est consommée directement par le même langage, ou un langage dérivé de celui de l'API. Exemple : l'API Log4j est une API java de journalisation ; cette API ne peut être consommée que par un programme écrit en java.
- **Distribuée** : une API distribuée est consommée par un programme tiers, sans être obligatoirement exécutée sur la même machine (liée physiquement) et sans qu'ils soient écrits avec le même langage. Ces API sont généralement exposées en SOAP ou REST.

L'un n'exclut pas l'autre, une API peut en même temps être une API web, native et distribuée. Cela dépend toujours des besoins derrière le développement de cette API.

Classification fonctionnelle

La classification fonctionnelle, à l'opposé de la classification technique, se focalise plutôt sur l'aspect fonctionnel des API (les fonctionnalités, les features...), indépendamment du volet technique (la façon dont elles sont exposées) :

- **Graphique** : une API graphique permet de créer plus aisément des interfaces graphiques. Exemple : l'API bootstrap qui permet de créer des composants graphiques web en faisant une abstraction sur la complexité de CSS.

► **Fonctionnelle** : une API fonctionnelle permet de traiter des problèmes liés à une fonctionnalité donnée. Par exemple : le calcul du nombre de personnes connectées sur Twitter.

► **Données** : une API de données (ou API data) permet de manipuler des données : extraction, transformation, fouille, stockage....

► **Sécurité** : une API de sécurité permet de gérer la sécurité dans un programme : authentification, chiffrement, hashage....

Dans cette classification (fonctionnelle), le découpage de la classification dépend de la vision de la fonctionnalité exposée par l'API, cette liste n'est pas exhaustive.

COMMENT CRÉER UNE API ?

Pour construire une API, il est nécessaire de respecter certaines étapes. Nous allons voir dans cette section les principales étapes qu'il est bon de mettre en œuvre pour assurer la pérennité et le succès d'une API.

Le succès et la pérennité peuvent se trouver à différents endroits, c'est-à-dire que selon le type d'API que l'on réalise, qu'elle soit publique, privée, sous forme de service Cloud ou proposée au sein de la librairie, la finalité est différente. La liste que nous allons voir ci-dessous décrit un certain nombre d'étapes que l'on peut mettre en œuvre ou non selon le contexte.

La prise en main d'une API

Pour assurer un bon démarrage aux futurs utilisateurs, il est nécessaire de mettre à disposition un guide de démarrage (Getting started) permettant de voir comment utiliser l'outil au travers de cas nominaux. Une explication des bonnes pratiques à respecter est aussi une aide précieuse au début. Bien souvent lorsqu'on débute, des questions récurrentes sont posées, pour les adresser de manière efficace, mettre en place une FAQ (Foire Aux Questions).

Une documentation claire

La documentation est un élément central pour faire en sorte que les personnes adoptent le projet. Il est important que cette documentation soit rédigée par le(s) développeur(s) de l'API. Cette documentation doit également rester ouverte afin que des contributeurs puissent venir y ajouter leurs points de vue. Un autre élément important est une gestion propre des messages d'erreurs ; Il faut que l'ensemble des messages d'erreurs pouvant être remontés par l'API puissent être facilement identifiables afin d'en trouver la cause. L'utilisation de code d'erreur est une bonne pratique. Pour permettre d'avoir une vue d'ensemble à l'utilisateur de l'API, il est utile de lister de façon exhaustive l'ensemble des points d'entrées.

Voici une liste d'outil permettant de mettre en œuvre une documentation efficace :

► **CARTE** (<https://github.com/devo-ps/carte>) : Outil largement inspiré des deux outils suivants. Il permet de décrire la structure d'une API web.

► **I/O Docs** (<https://github.com/mashery/iodocs>). Outil permettant de créer une documentation interactive.

► **Swagger Ui** (<http://swagger.wordnik.com/>). Swagger permet de produire et visualiser la structure complète de l'API web documentée.

Le code source

Cet élément pourrait faire partie du bloc ci-dessus. Il est important de proposer des exemples de code à la manière de « How To ». Cette pratique permet de pouvoir rapidement utiliser l'API pour des problèmes précis. L'hébergement du code source doit se faire sur une plateforme faisant converger les outils de gestion de versions et les réseaux sociaux. Aujourd'hui la plateforme la plus utilisée est Github (ce point concerne aussi bien un projet public que privé). Github propose un service de snippet de code. Cela permet de stocker dans Github (sur le Cloud) des morceaux de code. Ces bouts de code peuvent être privés ou publics.

Support

Bien que l'on ait mis à disposition plusieurs moyens d'apprentissage (exemple de code, guide de démarrage rapide, FAQ), il n'est pas rare d'être confronté à des problèmes liés au contexte dans lequel nous souhaitons utiliser l'API. Pour permettre à la population gravitant autour du projet d'échanger et de s'entraider, il existe les groupes communautaires, les forums de discussions, etc.. Sur le plan logiciel, il existe un outil nommé **StashBoard** (<http://www.stash-board.org/>) ; il permet de créer un tableau de bord qui aura comme objectif de monitorer l'API web sous-jacent. C'est un outil générique qui peut adresser pas mal de problématiques.

Communication

Tout au long de la vie d'un projet, il est important d'informer les développeurs/utilisateurs des évolutions en cours ou à venir.

Suivre les évolutions

Il est toujours intéressant d'avoir une vue sur l'ensemble des évolutions passées et à venir d'une API. Pour cela il est nécessaire de fournir le canal permettant de remonter dans le temps et voir les différentes versions qui ont jalonné la vie de l'API. Cette information doit être mise à disposition au travers d'un « change log ». Ensuite, il faut donner la vision future aux développeurs en fournissant une roadmap du projet à moyen/long terme.

Licence

Lorsque l'on publie une API, il est nécessaire d'indiquer au public les droits qu'ils ont vis-à-vis de l'outil et du code source sous-jacent. Pour cela, il est nécessaire de définir la licence s'appliquant au projet ainsi que les conditions d'utilisation d'un point de vue commercial.

Financement/rentabilité

Pour qu'un projet puisse vivre dans le temps, il est forcément question d'argent à un moment donné. Lorsque cette question entre en jeu, cela veut dire que le projet prend une autre tournure. Selon le type de projet (amateur ou professionnel), il faut définir le moyen par lequel celui-ci se financera. Plusieurs options sont possibles. Par exemple, en définissant un quota de consommation ou un coût d'une licence, ou encore une contribution libre.

DÉPLOIEMENT

Une API est tout d'abord une Interface qui permet à une application ou à un système autre que le vôtre de pouvoir interagir avec un autre système.

Cette approche permet d'ouvrir votre système ainsi que vos données (toutes ou partiellement) vers le monde extérieur.

C'est donc une approche architecturale et une solution pour avoir le moins de couplage entre les différents services qu'ils soient internes à votre entreprise ou externes.

L'une des premières entreprises à adopter l'exposition de ses services en API est Amazon. Afin d'éviter de donner des accès directs aux informations ainsi qu'aux services d'une entreprise à une autre, les départements informatiques des entreprises ont donc préféré publier des API, ce qui leur permet d'avoir le contrôle sur leurs données.

De la simplicité ainsi que de la popularité grandissante envers les API, les entreprises, et différentes équipes de développement, ont même commencé à publier des API pour leur utilisation interne ; ceci ayant donc pour effet de minimiser les coûts de développement et de maintenance.

Il est bien connu que faire évoluer les systèmes d'information pour s'attacher à suivre les besoins et la complexité toujours grandissante du business est une tâche ardue et consommatrice de ressources ; les API semblent être un bon compromis et même un sérieux booster pour faire des économies tout en facilitant l'adaptation rapide aux nouveaux besoins. Dans ce qui suit il est question des « web API » ce qui signifie toute URL/URI permettant d'agir sur un système distant afin d'interagir dessus ou d'en extraire des informations.

Protocole de communication

Lors de la conception de vos API, vous allez être amené à passer par plusieurs étapes de la demande fonctionnelle à la gestion des différentes versions en passant par les optimisations ainsi que la sécurisation des accès.

En premier lieu il faut choisir un protocole d'échange, pour cela pas la peine de chercher très loin, HTTP sera votre meilleur allié. Mis à part le fait que HTTP est compatible avec tous les navigateurs web, c'est le protocole par défaut de communication à utiliser dans le web. Ceci a l'avantage de vous faciliter la vie, tout en vous évitant de développer un protocole propriétaire qui rende vos API compatibles avec le web. HTTP permet par exemple via l'utilisation des verbes (**HTTP verbs**) de distinguer le type de requête sur une ressource publiée.

Supposons L'URL <http://mydomain/clients/01>; en effectuant une requête HTTP sur cette URL avec le verbe « **GET** » on aura probablement une réponse qui nous informera sur des détails du client 01. La même requête HTTP sur la même URL, mais en utilisant cette fois le verbe « **DELETE** », appellera la fonction permettant la suppression du client 01 du système. De cette façon de combiner URL et mots HTTP est né un style d'architecture nommée REST (Representational State Transfert) popularisée par le web et adoptée par tous les développeurs. Cependant, ce n'est pas suffisant ! Il faudra aussi définir un format d'échanges de données. Pour cela on utilise le format JSON (pour JavaScript Object Notation), lui aussi popularisé par le web. Cet acronyme ne correspond plus toutefois à ce que l'on a voulu faire de ce format au départ puisque l'on peut très simplement publier et consommer JSON sur une autre plateforme que JavaScript. JSON permet de créer des structures simples ou complexes (par imbrications), qui présentent l'avantage d'être facilement « **debuggables** », compressibles et humainement lisibles. Vous pouvez bien évidemment choisir un autre format (XML, TEXT, HTML). Cependant, les API sont de plus en plus consommées par des téléphones portables ou des tablettes avec des connexions plus lentes et généralement limitées (fair use) : chaque byte compte. Le XML par exemple présente l'avantage qu'il sait être interprété par tous les navigateurs, mais cet avantage a un prix : la taille du document.

STRATÉGIE DE DEPLOIEMENT

Vous venez juste de finir la conception et la publication de vos API, votre format d'échange de données est JSON, il ne vous reste qu'à publier ou déployer vos API. Pour cela, il existe plusieurs approches suivant les ressources dont vous disposez. Si vous ou votre équipe

avez les connaissances techniques nécessaires, c'est la route à prendre, sinon il faudrait probablement utiliser des ressources externalisées à votre disposition.

Les solutions clé en main

Les entreprises utilisent des progiciels en interne comme des bases de données, des tableurs ou encore des frameworks techniques. Chacun de ces outils offre une solution permettant d'exposer une API. Dans ce cas, il est intéressant de s'orienter vers les solutions mises à votre disposition; en effet ce choix impacte directement le coût de montée en compétence sur les outils.

Plateforme as a Service (PaaS)

Ce type de service, hébergé dans le Cloud, permet de bénéficier de la puissance technique d'une plateforme ainsi que des outils proposés par celle-ci, pour développer plus facilement sa propre API. Nous pouvons prendre comme exemple le service **AppEngine** proposé par Google. Il offre, en plus des ressources physiques, un point d'entrée permettant de développer et d'exposer facilement des API. Le service offre également une console d'administration permettant de monitorer les API exposées.

API as a Service (AaaS)

Le service de type **AaaS** a pour objectif de fournir des API visant à être consommées par des applications tierces. Elles pourront être accessibles avec ou sans authentification. Nous pouvons prendre comme exemple, les API comme Twitter, Google plus, Facebook, etc.

Le « sur mesure »

Les API dites sur mesure correspondent à des développements internes à une entreprise. L'objectif de ce type de service est de répondre à une problématique fonctionnelle, ou technique, très spécialisée sur le système d'information de l'entreprise. Si nous prenons en exemple une entreprise de transport, on peut imaginer que dans le cadre de l'activité de la société, le système d'information ait besoin de fournir à plusieurs applications le nombre de kilomètres parcourus par l'ensemble des véhicules. Les informations exposées pourront être interrogées selon certains critères. Ce besoin est très spécifique au besoin de l'entreprise. Par conséquent, il faudra mettre au point une API permettant d'effectuer des requêtes permettant d'obtenir ces informations.

Sécurisation des API

Pour sécuriser l'accès à l'API, les fournisseurs de services utilisent des mécanismes de sécurisation. Ces mécanismes peuvent être implémentés par des protocoles standardisés ou

non, selon les besoins :

- **Le protocole standardisé comme Single Sign-On (SSO)** : ce type de protocole permet d'avoir un point d'entrée unique sur le système d'information sous-jacent. C'est-à-dire que l'on devra s'authentifier une fois, et que l'on pourra accéder à l'ensemble des applications contenues dans le périmètre couvert par le SSO. Plusieurs mécanismes existent pour mettre en place le SSO : OAuth2, openId, SAML,...
- **Son propre protocole d'authentification** : ici, utiliser un protocole standard, mais implémenter selon les besoins du fournisseur un mécanisme adapté.

Monétisation de l'utilisation d'une API

Dans le monde des API, il existe plus d'une technique pour rentabiliser l'utilisation d'un service. Nous allons voir 3 moyens que l'on peut mettre en œuvre :

- **Quota** : la consommation d'un service va générer l'utilisation de plusieurs ressources sur le service distant. Typiquement, de la puissance machine, de la bande passante, de l'espace disque. Les quotas dont nous parlons vont s'appliquer pour déterminer le coût de fonctionnement. Ce type de facturation présente un avantage. Il permet de payer uniquement ce que l'on consomme. Aussi, lors de la mise en place d'une nouvelle API si celle-ci n'est pas beaucoup utilisée, le coût d'infrastructure sera négligeable. En revanche avec la popularité, les chiffres de consommation varieront. En fonction de cette variation et des quotas, le prix d'utilisation variera également. Le fonctionnement par quota est très utile et souple.
- **Abonnement** : le principe de l'abonnement, à l'instar de la gestion par quotas, fixe un forfait mensuel (voire annuel). C'est-à-dire qu'il faut payer pour utiliser le service (comme dans le cas précédent) et ceci de manière régulière. L'avantage que l'on peut y voir est qu'en terme de planification de budget il est facile d'anticiper sur les dépenses liées à l'utilisation de ce service.
- **Service gratuit** : comme l'intitulé l'indique, celui-ci n'engendre aucun coût lié à son utilisation. Cependant, il est d'usage de fournir un canal permettant aux utilisateurs de contribuer financièrement au projet.

Scalabilité d'une API

Une API exposée publiquement doit être capable d'encaisser une charge importante de requêtes. La répartition en termes de charge dans le temps n'est pas toujours homogène. Selon les périodes la charge peut varier. Il faut donc être capable d'adapter les ressources en fonction de la charge. Cette régulation doit se

faire en temps réel, ceci demande une réactivité importante du système. Il existe plusieurs manières de gérer cette problématique. Nous allons voir ci-dessous une liste non exhaustive des solutions possibles :

- ▮ **Load balancing** : cette technique permet de répartir dynamiquement la charge de requêtes sur plusieurs nœuds de la ferme de serveurs. Le load balancer se trouve en amont de la chaîne de traitement d'une requête. En fonction de la charge existante sur l'ensemble des nœuds, le répartiteur (load balancer) va diriger la requête vers le nœud le moins encombré. Cette solution est couramment implantée dans les entreprises.
- ▮ **Cloud** : la solution Cloud permet de s'abstraire de toutes les problématiques d'infrastructure. En effet, lorsque vous déployez votre API sur le Cloud (Google, Amazon, Heroku...), c'est à la charge du fournisseur d'assurer cette gestion des pics de charge.

GÉRER UNE API

Gestion de versions

Avec l'évolution rapide des systèmes d'information, la gestion de versions devient une nécessité. Les API ne font pas exception, même si le concept donne l'illusion qu'elles sont dispensées du versionning : après tout, ces dernières ne sont exposées que pour donner un ensemble de services, et ce, indépendamment de l'évolution des systèmes appelants.

Versionning pour le développement

Les règles de gestion de versions d'une API respectent les règles standard en termes de numéro de version. À moins que le projet n'adopte une manière spécifique, les règles en vigueur sont les suivantes :

Une version mineure implique la correction de bugs ou alors des évolutions ne mettant pas en jeu la compatibilité de l'API. Le changement de version dans le cas d'une version mineure se fait de la manière suivante : 1.6 -> 1.6.1
Une version majeure implique un changement important dans l'API mettant en cause la compatibilité avec des versions antérieures. Généralement, cela correspond à l'ajout d'une nouvelle fonctionnalité et/ou des évolutions sur une fonctionnalité existante. 1.6 -> 2.0

Versionning pour la consommation

À cause de certaines contraintes, le client d'une API est obligé de travailler avec une version historique et ne peut pas encore utiliser la dernière version majeure. Pour permettre aux utilisateurs de l'API de sélectionner la version, le fournisseur ajoute un paramètre permettant de distin-

guer la version appelée. Plusieurs techniques existent :

- ▮ **La gestion dans l'URL/URI** : c'est la méthode la plus utilisée, la version est gérée dans le lien de l'appel. Exemple : <http://www.nonDuSite.com/api/v1/service>. Ici on appelle la version v1 de l'API service. Ce mécanisme est utilisé par exemple par les endpoints de Google, Twitter...
- ▮ **La gestion dans le HTTP header** : avec ce type de gestion, la version à utiliser est passée dans le header HTTP.
- ▮ **La gestion en dur** : dans ce type de gestion, la version à utiliser est passée dans le corps du message.

Compatibilité ascendante et descendante

En exposant une API, elle sera consommée par des composantes qui peuvent être dans le même système, ou un système externe (indépendant de l'évolution à mettre en œuvre). D'où l'obligation de garder la capacité à gérer les appels basés sur une version antérieure... Deux solutions se présentent pour adresser ce problème et garder la compatibilité de l'API :

- ▮ **La compatibilité ascendante** consiste à ce que l'API, en évoluant, soit capable de gérer les appels des systèmes basés sur l'ancienne version. Ce mode de compatibilité est réalisé en tenant compte de l'ancienne version et en développant la nouvelle.
- ▮ **La compatibilité descendante** : Consiste à ce que l'API soit capable d'être flexible avec les appels futurs. C'est-à-dire un appel d'un système développé pour consommer une future version de l'API. Ce mode de compatibilité est réalisé en utilisant des fichiers de configuration pour avoir un meta-model.

Dans les deux cas, une bonne conception de l'API, en se basant sur des patrons de conception (design patterns), permet une meilleure gestion de la compatibilité de l'API. Néanmoins, la vie n'est pas toujours aussi facile pour une API ! Le cap doit être passé parfois pour mettre fin à une version précédente devenue obsolète, ou pour enlever une fonctionnalité qui n'est plus utilisée. Une prévision de cette dépréciation est à mentionner (de préférence) dans les sous-versions précédant la mise au placard d'une version donnée, et ce, afin de donner du temps aux consommateurs de l'API pour changer leurs appels à cette dernière.

LES MARKETPLACES

La première idée qui me traverse l'esprit en parlant des « API Marketplaces » c'est iTunes. Quand on veut acheter une chanson sur iTunes, on paie Apple et non pas Sony Music, ce qui fait de Apple et donc iTunes un « Broker » et repré-

sente donc l'unique intermédiaire ou interface par laquelle on passe pour consommer de la musique (il y en a d'autres heureusement). Les entreprises ont donc tout intérêt à consommer les API des autres entreprises — principe DRY —, mais surtout de publier leurs propres API afin que d'autres entreprises puissent les consommer — Google, Twitter, etc. — et notamment via l'utilisation toujours grandissante des périphériques mobiles. Plusieurs problématiques se posent alors : comment peut-on s'assurer que l'URL de l'API que l'on consomme sera toujours la bonne — lien cassé — ? Comment gérer la multitude des contrats d'utilisation de ses ressources, faudrait-il accepter un à un chacun d'entre eux ? Comment faire quand le client fait appel à différentes API avec chacune un SLA différent ? La solution est une centralisation permettant un accès unifié aux différentes API qui, sous la tutelle de « Marketplace », auront les mêmes contrats et les mêmes SLA. L'application cliente aura donc un interlocuteur unique qui certifie de la qualité de l'API. L'API Marketplace se comportera donc comme une centrale d'achat où le développeur fera son « shopping » d'API nécessaires pour son application cible sans avoir à se préoccuper de l'hétérogénéité des contrats des différentes API, car son seul interlocuteur — et donc son seul contrat — c'est avec la Marketplace. Il existe un nombre important d'API Marketplaces, publiques ou sur abonnement, spécialisées (domaine bancaire, médical, etc.) ou généralistes (réseaux sociaux, gestion de contenus, etc) parmi elles :

Mashape
Programmableweb

Conclusion

Il reste bien évidemment beaucoup à dire concernant l'écosystème autour des API, un écosystème en continuelle évolution. Ce qu'il faut retenir à mon avis c'est que désormais, tout système d'information doit « impérativement » inclure dans sa phase de conception, d'évolution ou de refonte, un volet permettant sa publication de tout ou partie sous forme d'API ; il devrait aussi être en mesure de consommer des API internes ou externes. Ceci permet une « scalabilité » horizontale de l'infrastructure informatique de l'entreprise, et permet de rendre pérenne l'investissement des entreprises en matière d'IT.

- 🔴 Fabrice Sznajderman Senior Java & Scala développeur à SFEIR
- 🔴 Hamdi Bouhanef Concepteur développeur java EE à SFEIR
- 🔴 Abderrazak Bouadma Software Engineer à SFEIR

APIdays, happy days

Les API, tout le monde en parle. En moins de deux ans, cet écho s'est propagé bien au-delà du seul cercle des développeurs car ces fameuses Application Programming Interfaces semblent être la solution la plus souple et efficiente pour mettre à disposition un service via Internet.

Stéphane Distinguin, fondateur et PDG de Fabernovel, agence internationale dédiée à l'innovation et organisatrice des APIdays, un des plus gros événements mondiaux consacrés au sujet, le souligne : "les API sont la meilleure interface possible entre le numérique et le physique, la technique et le business.

Les API incarnent au mieux cette révolution technologique et la capacité qu'ont des personnes qui ne sont pas uniquement des informaticiens, à utiliser ces éléments de façon à développer des services, à inventer de nouvelles applications."

Dans le cadre de ces deux journées de conférences, une équipe de démonstration formée d'étudiants d'Epitech était invitée à mettre en pratique et démontrer tout l'intérêt d'une API pendant ces deux jours APIdays.

Leur challenge : créer un système d'information complet ainsi qu'une application utilisant cette infrastructure. Ils nous en parlent eux-mêmes.

Infrastructure

L'infrastructure déployée ne suivait pas un fonctionnement figé, mais plutôt "élastique", « adaptable » ou « scalable » en anglais. Elle permettait une adaptation des ressources en fonction du besoin, respectant les fameuses et connues lois du "Cloud" : celles des ressources à la demande. De façon plus concrète, plus les serveurs sont chargés, plus nous déployons de serveurs pour traiter ces requêtes.

Voir le schéma.

Tripartite

Techniquement trois composants étaient présents. La partie client, comprenait une application web et une application Android qui interagissaient avec notre infrastructure via une API. Le serveur backend, le deuxième composant, desservait et traitait les données en interagissant avec la base de données. Dans le but de créer une architecture pouvant tenir la charge, nous avons choisi des technologies pou-

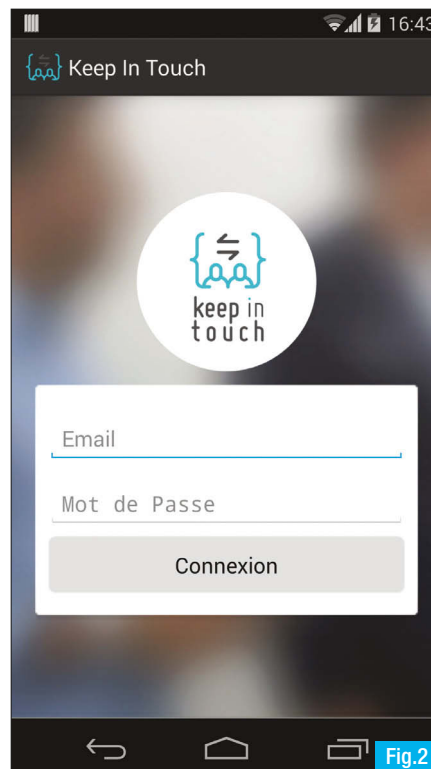


Fig.2

vant être répliquées horizontalement. Pour cela nous avons opté pour le langage Javascript, ou plutôt NodeJS ainsi qu'une base de données NoSQL, ici MongoDB.

Finalement, en collaboration proche avec HP pour la gestion de l'infrastructure, nous avons utilisé OpenStack. Grâce à son API nous pouvions demander de nouvelles instances, dupliquées à partir de celles déjà présentes (cf. voir Fig.1).

Répartition de charge interne aux machines

Étant donné que le javascript, de nature monothreadé, n'utilise pas, nativement, tous les processeurs d'un même serveur, nous avons utilisé un utilitaire développé par votre serveurur, pm2 (<https://github.com/Unitech/pm2>).

Il permet de dupliquer un programme NodeJS

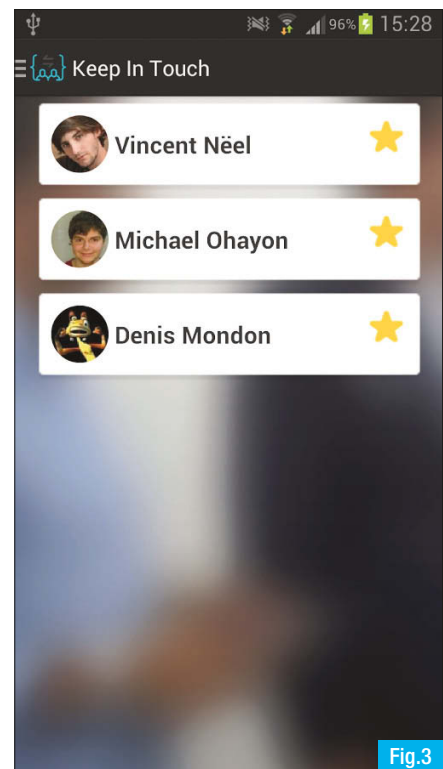


Fig.3

sur tous les processeurs disponibles et répartir la charge sur chaque processeur.

Orchestrateur

Cette partie "intelligente" permettait de "moniter" les serveurs utilisés et de considérer leur charge. Si l'orchestrateur voyait que tous les serveurs étaient en surcharge, il demandait à OpenStack de créer un nouveau serveur (scale up). A l'inverse, s'il considérait que les serveurs déployés étaient en surplus par rapport à la demande, alors, cette fois, il enlevait des serveurs (scale down).

Cet orchestrateur comprenait aussi la logique d'ajout des adresses IP à chaud dans le load balancer. Concernant celui-ci, nous avons choisi Nginx, même si HA Proxy aurait été plus adapté.

Descriptif technique de l'application Keep in Touch

Gestion Connexion Utilisateur

Pour que l'utilisateur puisse se connecter à l'application, notre API utilise un système de token d'identification. A chaque fois qu'une connexion est effectuée, un token est attribué à l'utilisateur. Ensuite, pour chaque requête

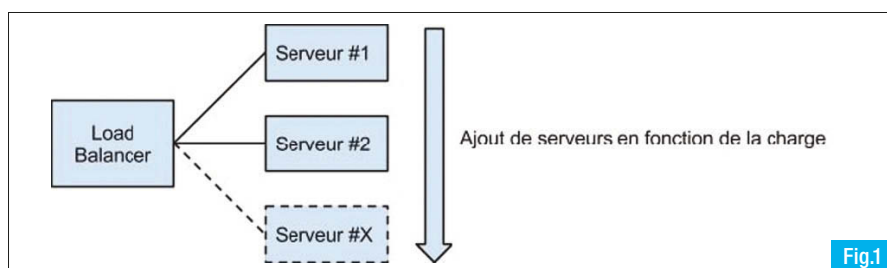


Fig.1

envoyée au serveur, le token est transmis au serveur via la méthode POST pour vérifier que l'utilisateur est connu [Fig.2](#).

Récupération des informations

Pour récupérer les informations, nous faisons appel à l'API et lui demandons la liste des contacts. Les données sont au format JSON (JavaScript Object Notation) et sont converties en objets Java grâce à la bibliothèque Gson. Comme nous disposons de plusieurs centaines de contacts, nous ne les récupérons pas tous en même temps. Nous mettons en place un système de pagination. Nous récupérons en effet un flux réduit de données (quelques dizaines de membres) et nous téléchargeons la suite de la liste au fur et à mesure que l'utilisateur fait défiler la liste des membres [Fig.3](#).

Gestion des permissions des données

Chaque donnée peut être partagée différemment à chaque contact. Par exemple, il est possible de partager son profil LinkedIn avec une personne et son Vimeo avec une autre. Nous avons pris l'initiative de présenter ce choix de permissions via un système de drag & drop. En faisant glisser son profil LinkedIn dans la liste des données accessibles, nous indiquons à l'API de rendre l'information lisible par le contact, par défaut, seule l'adresse email est partagée avec tous les participants de la conférence [Fig.4](#).

Code source du projet : www.programmez.com

FM

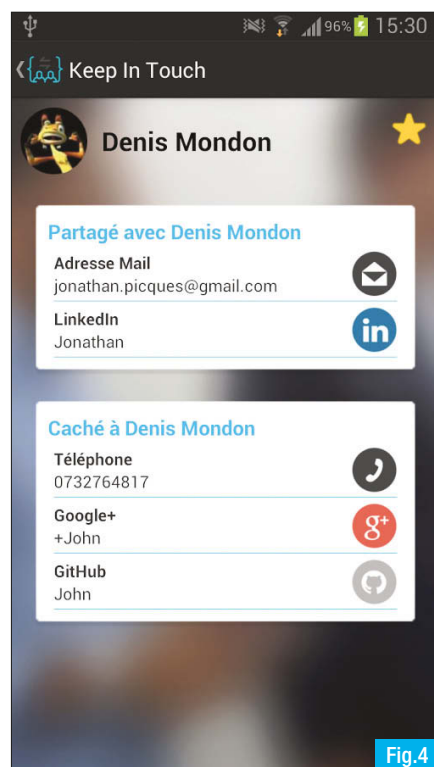


Fig.4

Interview de Niels Frier, responsable du Cloud dans l'Epitech Innovation Hub, pour remettre en perspective les API dans l'histoire de l'informatique.

C'est quoi une API ?

En gros, c'est un contrat qui permet de définir entre deux personnes, deux entités, deux logiciels, le processus de communication. Afin de définir la manière dont ils vont communiquer ensemble. Par exemple, il y en a un qui explique : « si tu m'appelles de telle façon, je pourrai te répondre ça ». Ce qui veut dire que l'autre sait alors que, s'il a besoin d'obtenir le « ça » en question, il va pouvoir communiquer avec la première entité, en passant les bons paramètres. En général, il s'agit de déporter du fonctionnel, cas que l'on rencontre fréquemment en architecture informatique où, très généralement, si l'on veut développer quelque chose de compliqué et mettant du temps pour être développé sur plusieurs plateformes, on va pouvoir le faire en une fois sur un serveur central et l'exploiter par le biais d'une API. Ce qui veut dire que n'importe quelle instance qui connaît le bon contrat de communication pourra utiliser cette fonctionnalité-là.

Pour résumer, une API rend interopérable deux systèmes entre eux ?

Voilà, c'est une bonne façon de faire sauf qu'il s'agit d'un mécanisme parmi d'autres, même si celui-ci en

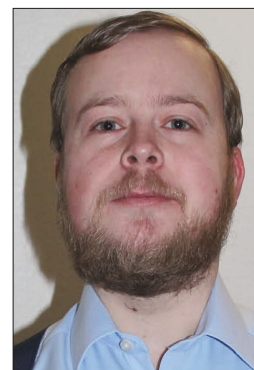
particulier a beaucoup séduit. Alors qu'en lui-même, c'est un concept qui peut être utilisé de bien des manières différentes : on peut avoir des API sur Internet comme on peut avoir d'autres API sur un seul PC, qui permettent de faire communiquer deux processus. C'est ce qu'on appelait « IPC » depuis les débuts de l'informatique et qu'on appelle aujourd'hui API, et plus généralement le fait de mettre à disposition un service via Internet.

L'IPC ?

C'est l'inter-process communication, quand on voulait faire communiquer deux processus sur les vieux PC. Mais cela se pratique encore sur ceux d'aujourd'hui : on définissait un contrat pour que chaque logiciel installé sur le PC soit capable de se dire « je vais communiquer en fonction de tel contrat ». Une API, c'est l'évolution de ce principe-là, déporté sur Internet.

D'où l'explosion du marché du Cloud...

Ce qui est une évolution logique. Le fait d'utiliser des serveurs pour héberger des sites, et plus globalement des services sur Internet, existe depuis longtemps, toujours selon la même idée : centraliser les ressources. N'importe qui, de l'étudiant ou de la multinationale, peut développer un service sur le Cloud pour quelques euros par mois et s'il rencontre le



succès avec la fonctionnalité qu'il expose, il pourra très facilement soutenir la charge.

Si je me souviens de la conférence de Laurent Solly lors de ces APIdays, c'est un genre de solution parfaite pour une plateforme type Facebook – entre le Cloud, l'interopérabilité et les services liés à la plateforme – non ?

Encore une fois, tout est lié : avec ce qu'on appelle les SOA, les architectures orientées services, l'idée est effectivement de pouvoir déporter des services fonctionnels sur diverses entités à travers Internet. Le Cloud a juste permis d'exploser ceci parce qu'à l'inverse d'avant, où les besoins professionnels représentaient des budgets assez lourds, avec une administration système, une plateforme réseau et ainsi de suite jusqu'à obtenir une architecture complète, aujourd'hui avec la technologie à disposition, le premier « étudiant venu » est tout à fait capable de développer cela seul. Soit pour monter sa boîte, soit tout simplement pour externaliser un service qu'il pourra utiliser en interne dans le cadre de l'entreprise.

Le Data Scientist n'est-il pas l'expert en Business Intelligence ?

Habituellement, quand on parle de « Big Data », on évoque fréquemment son côté stockage et peu son côté exploitation. Mais ce dont on parle très peu, c'est de son côté business ainsi que des acteurs impliqués au sein d'une entreprise. C'est la raison qui m'amène à faire un zoom sur le métier de Data Scientist, qui, selon les grands de l'informatique comme Google ou IBM, est la perle rare des métiers d'avenir.

Confusions et Redéfinitions

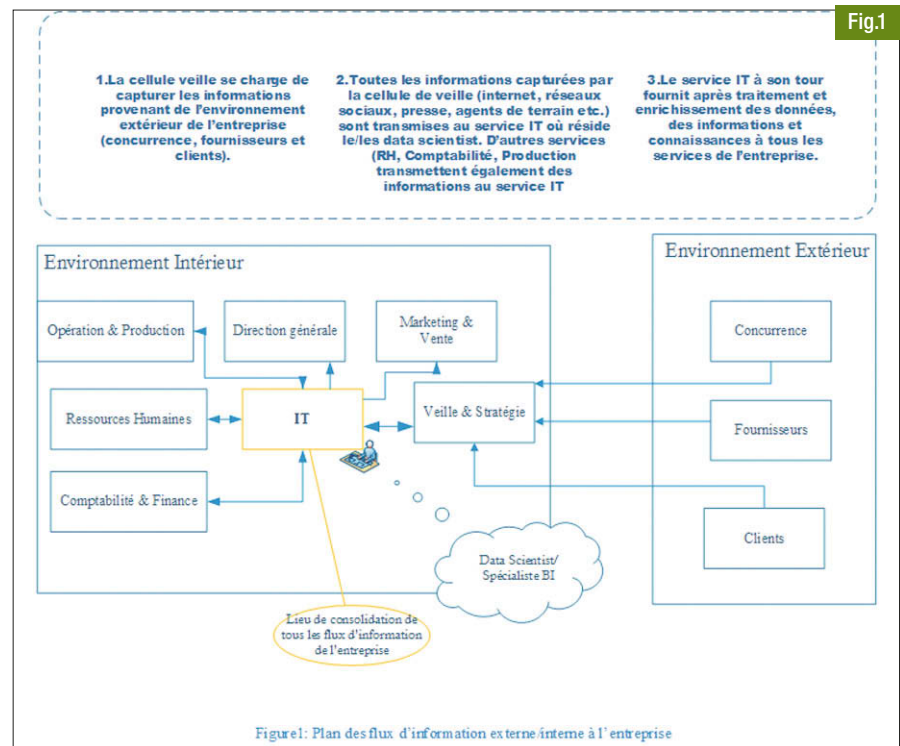
Lorsque vous demandez à quelqu'un de vous définir le métier de Data Scientist, vous vous apercevez assez rapidement que les définitions divergent suivant le domaine d'activité, voire même le pays d'appartenance de la personne. Par exemple, les Anglo-saxons rapprochent ce métier à celui de Business Analyst alors que les Francophones le rapprochent du métier de statisticien. Quoi qu'il en soit, c'est un poste qui requiert beaucoup de compétences simultanées (mathématiques, informatique, base de données, statistique, data mining), ce qui en fait une « denrée rare » sur le marché de l'informatique. Je vais certainement vous surprendre en vous disant que ce métier existe sous une forme plus connue : « Le Spécialiste Business Intelligence ». Avant de faire un zoom sur ce profil souvent méconnu par ses fonctions réelles au sein de l'entreprise, intéressons-nous au terme « Business Intelligence », souvent victime de quelques définitions fourre-tout ou inadéquates. Mieux vaut donc partir de son étymologie. Business Intelligence est un terme composé de deux mots : Business et Intelligence.

Business

Business est une contraction de « Busy » et « Ness » qui signifie littéralement « Ce que l'on est occupé à faire ». Autrement dit, il s'agit de « ce que l'on sait faire ». Ou de manière plus propre, son « Activité ». Par exemple, le business d'une banque est : la gestion des capitaux (du détail à l'investissement). Dans les faits, le Business qui au départ, définissait l'activité d'une entreprise a pris une dimension capitaliste après la seconde guerre mondiale et pendant la guerre froide pour être défini comme étant « ce que l'on sait faire moyennant de l'argent ». Le business d'un agriculteur est, par exemple, son savoir-faire (l'agriculture) moyennant de l'argent. Malheureusement, depuis l'introduction de la dimension financière, le business est parfois considéré comme étant l'argent rapporté par une entreprise.

Intelligence

Après la définition de Business, intéressons-



nous à l'intelligence. A première vue, l'intelligence paraît triviale à tout le monde. Mais si on vous demande de la définir, ça paraît moins évident. De manière basique, l'intelligence est : la faculté de lier des choses qui auparavant ne l'étaient pas. Par exemple, considérez les trois objets suivants : une table, une chaise et une lampe de chevet. Alors si quelque chose vous fait penser aux études, c'est que vous avez fait un lien entre ces trois objets. Vous venez alors de manifester une forme d'intelligence. L'intelligence concerne plusieurs domaines :

- Dans le domaine linguistique, on dira qu'une personne est intelligente linguistiquement, si elle arrive à faire des phrases très équilibrées à l'aide de quelques mots liés (exemple : les poètes).
- Dans le domaine musical, on dira qu'une personne est intelligente musicalement, si elle arrive à produire du son agréable avec quelques liens ou juxtapositions de notes de musique.
- Egalement dans le domaine scientifique, on qualifiera d'intelligence logique, la capacité à

enchaîner des séquences logiques aboutissant à des démonstrations élaborées, voire même des postulats ou des théorèmes.

Tous ces exemples illustrent un seul principe : celui du lien établi entre les objets. La Business Intelligence n'échappe pas à ce principe : « établir des liens entre les différents éléments de votre activité ». En d'autres termes, il s'agit de découvrir de l'information ou de la connaissance en établissant des liens entre les données de votre activité. Vous faites donc de la Business Intelligence à partir du moment où vous croisez vos données afin d'en découvrir de la connaissance au service de votre activité.

Spécialiste Business Intelligence = Data Scientist ?

A ce stade, vous vous demandez certainement, quel est le lien entre « Business Intelligence » et « Data Scientist ». En effet, le Data Scientist, défini comme étant le chirurgien des données, est la personne qui manipule les données en profondeur, dans le seul but de répondre à ses doutes et intuitions. Le spécialiste en Business



Tout savoir sur

Visual Studio®



*Les nouveautés
Visual Studio Online - Les tests
ALM - DevOps*

Coder et déployer partout

Le développeur manipule de plus en plus de technologies, de frameworks, de bibliothèques et d'outils. Il doit supporter de plus en plus de terminaux (desktop, tablettes, mobiles, consoles de jeux, SmartTV, le Cloud Computing, le Web, etc.), tout en fournissant un code de qualité et le plus optimisé possible. L'IDE doit intégrer tout cela et simplifier au maximum le travail du développeur, tout le moins, l'aider à coder plus rapidement et à automatiser au mieux les tâches répétitives.

Visual Studio 2013 est plus que jamais la plateforme de développement de référence pour Windows et le Cloud Computing. Deux fonctions phares se détachent : Code Lens et Code Map. Ce dernier est particulièrement utile dans les gros projets. Visual Studio 2013 supporte Windows 8.1, Windows Store, Windows Phone. Il n'a jamais été aussi simple de développer pour les tablettes et les smartphones. Visual Studio s'ouvre aussi à d'autres technologies et plateformes. Grâce au nouveau partenariat avec Xamarin, il est possible de créer des applications mobiles Android et iOS !

Une des grosses nouveautés est l'approche DevOps qui tient désormais une place stratégique. Dans le même temps, l'offre ALM se complète et rajeunit (Visual Studio et Visual Studio Online). L'idée est d'intégrer l'ensemble des évolutions technologiques et marchés : mobilité, multi-terminaux, les services Cloud, les fonctions sociales, la qualité et la livraison continue... ALM et DevOps ne feront qu'un. Nous retrouvons l'idée d'agilité technique et d'agilité business. Mais le paradigme ne change pas : planifier, développer, tester, livrer, opérer. Les services Cloud vont s'intégrer de plus en plus à Visual Studio. Ainsi, il est, dès à présent, possible de générer vos applications en utilisant le service Build de Visual Studio Online et d'automatiser et de suivre les déploiements avec le service Release Management.

Comment développer et mettre en production plus rapidement ? Quelles sont les techniques de déploiement ? Comment être plus efficace sur les tests et la qualité logicielle ? Réponse dans les pages qui suivent.

 François Tonic



sommaire

Éditorial	40
MSDN Platforms	41
Offre globale Visual Studio 2013	42
Tous les environnements, tous les écrans et le multiplateforme	44
De Visual Studio 2010 à 2013	46
Gagner du temps avec Visual Studio 2013	47
Du prototypage à la production	50
DevOps : de l'agilité dans votre process de test et de déploiement	54
Agilité, tests et qualité	57
Ressources	62

MSDN Platforms

Avec la sortie de la version 2013 de sa gamme d'outils de développement Visual Studio, Microsoft en a profité pour mettre à jour ses offres MSDN avec la nouvelle MSDN Platforms. Voyons ce que cet abonnement apporte.

Avant de rentrer dans le vif du sujet, faisons un petit rappel sur ce que sont les abonnements MSDN (Microsoft Developer Network). Lorsque l'on développe une application pour la plateforme Microsoft, nous avons généralement besoin de deux choses : des outils de développement et un ou plusieurs environnements de tests. C'est ce qu'apporte l'abonnement MSDN aux développeurs, une licence Visual Studio pour le développement et l'accès à l'ensemble des logiciels et systèmes d'exploitation Microsoft pour réaliser des tests ou évaluer/prototyper de nouveaux logiciels. MSDN permet aussi de disposer des futures versions de Visual Studio.



pendant un court laps de temps son application sur différents environnements.

Enfin on retrouve d'autres bénéfices liés cette fois-ci au support, à la formation et aux ressources techniques. Au niveau du support, chaque abonnement inclut la possibilité d'ouvrir deux incidents auprès du support technique (très utile lorsque l'on rencontre un problème complexe sur un produit Microsoft), mais aussi un support prioritaire sur les forums MSDN et TechNet (les abonnements TechNet n'existent plus, mais l'ensemble des ressources, forums, sites, blogs... sont toujours présents) afin de trouver des solutions à nos problèmes avant d'ouvrir un incident au support Microsoft. L'abonnement contient aussi l'accès gratuit à deux cours sur la plateforme Microsoft E-Learning afin de se former sur de nouvelles technologies, d'améliorer ses connaissances sur une technologie connue ou se préparer pour passer une certification. On notera aussi la présence d'un abonnement à MSDN Magazine afin de rester au courant des dernières technologies et de profiter de tutoriaux autour des développements. L'ensemble de ces avantages sont un réel plus et sont maintenant accessibles sans la licence Visual Studio permettant aux professionnels IT de retrouver un accès aux logiciels et systèmes d'exploitation de Microsoft

MSDN Platforms

Avec la disparition des abonnements TechNet (pendant de MSDN pour les professionnels IT) et l'essor des développements multiplateformes, l'offre classique MSDN avait un inconvénient pour une partie des professionnels : celui d'intégrer obligatoirement une licence Visual Studio qui pouvait ne pas être utile. Pour pallier à ce problème, Microsoft propose désormais l'abonnement MSDN Platforms. Parmi les avantages de l'abonnement MSDN Platforms, on retrouve l'accès à une grande partie des logiciels Microsoft pour des usages hors production comme les différentes versions du système d'exploitation Windows mais aussi SQL Server, les produits de la gamme Dynamics ou encore SharePoint et Exchange. On notera toutefois l'absence de licence pour la suite Office et l'ajout depuis peu de la possibilité d'installer ces logiciels sur Azure afin d'avoir une plus grande flexibilité dans la mise en place d'environnement de tests.

En plus de ces bénéfices pour un usage hors production, l'abonnement MSDN Platforms contient des licences de production, cette fois pour la suite ALM de Microsoft ; à savoir une licence pour Team Foundation Server ainsi qu'une CAL pour les environnements sur site et une licence pour Visual Studio Online, la



version dans le Cloud. Ces licences pourront être très utiles pour les développeurs travaillant sur d'autres plateformes que celle de Microsoft mais souhaitant utiliser Team Foundation Server ou Visual Studio Online.

Comme indiqué plus haut, les licences fournies via l'abonnement MSDN peuvent maintenant être utilisées dans un environnement Azure. De plus, l'utilisation d'Azure est simplifiée par l'intégration d'un crédit de 75€ par mois pour les détenteurs d'un abonnement MSDN Platforms. Ce crédit permet à l'abonné de pouvoir utiliser l'ensemble des services Azure gratuitement dans la limite de 75€ par mois ; très pratique lorsque l'on veut avoir une machine virtuelle rapidement pour tester

pour tester, évaluer et prototyper, et permettant aux développeurs multiplateformes d'avoir, en plus de l'accès aux logiciels, accès aux services Microsoft comme la suite ALM Team Foundation Server / Visual Studio Online. Cela permet d'avoir un unique processus de développement pour l'ensemble des technologies utilisées.



Guillaume Rouchon
Consultant Senior ALM |
MVP Visual Studio ALM chez
Cellenza
Software Development Done
Right

Son Blog: <http://blog.getza.net>

Offre globale Visual Studio 2013

Visual Studio 2013 est disponible depuis novembre dernier. L'environnement étoffe l'offre proposée en 2012 avec une version Online qui n'est autre que le produit « Team Foundation Service ». Nous vous proposons de découvrir en détail l'offre 2013.



Fig.1

L'offre « Visual Studio 2013 » proposée par Microsoft reprend la structure de celle de 2012 : nous allons retrouver les versions Professional, Premium et Ultimate. Elle propose également, une version « Visual Studio Online » qui remplace « Team Foundation Service », version Cloud de la plateforme ALM « Team Foundation Server » (TFS) de Microsoft.

Nous vous proposons de passer en revue ces différentes versions de Visual Studio en allant de la plus basique à la plus complète avec un focus sur les nouveautés apportées. Puis de découvrir l'offre Visual Studio Online. (Fig.1)

VISUAL STUDIO 2013

Abonnement MSDN

Chacune de ces offres donne droit à un abonnement MSDN de niveau équivalent : VS Ultimate donne droit à un abonnement Ultimate par exemple. L'abonnement donne accès à des milliers de produits Microsoft et autres services supplémentaires. Vous disposez ainsi de tout le nécessaire pour concevoir, développer et tester vos applications sur diverses plateformes, y compris Windows Azure.

Visual Studio Professional

Grâce à cette version, vous pouvez créer des applications et des services de qualité avec une expérience de développement cohérente et

cela, que vous vous atteliez à la création d'expériences innovantes ou à la modernisation de vos applications métier. Vous aurez les outils dont vous avez besoin pour développer :

- Des applications Cloud et Web basées sur des standards actuels et de prochaine génération, tels que HTML5 et CSS3, qui s'adaptent aux navigateurs mobiles, modernes et traditionnels. Visual Studio 2013 inclut un support complet de JavaS-

cript et jQuery pour créer des applications.

- Des applications mobiles pour Windows Phone 8, et même iOS et Android avec Xamarin
- Des applications desktop avec la nouvelle plateforme Windows 8.1, dont des applications Windows Store, pour étendre vos solutions métier existantes. Avec la capacité de développer des applications métier Office 365 basées sur le Cloud, vous pouvez aider les utilisateurs à être plus productifs où qu'ils se trouvent.
- Mais également des applications Business Intelligence avec SQL Server...

L'amélioration de votre IDE et les interactions avec l'ensemble des outils Microsoft vous permettront également d'augmenter votre productivité :

- « Peek Definition » permet de consulter directement les méthodes appelées sans avoir à ouvrir une autre fenêtre (Fig.2).
- Le Browser Link, nouveauté concernant le développement web, vous permettra de synchroniser vos modifications entre les fenêtres du navigateur et l'éditeur de code.
- Les « Portable Class Library » permettront de créer des composants fonctionnant sur un ensemble hétérogène de plateformes : vous

n'avez donc plus à dupliquer vos codes et vos tests, améliorant ainsi la qualité et simplifiant la maintenance (Fig.3).

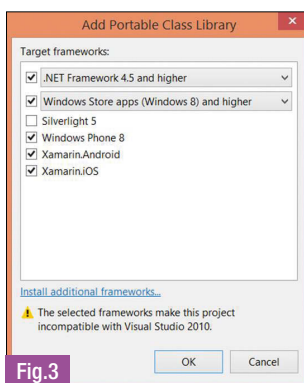


Fig.3

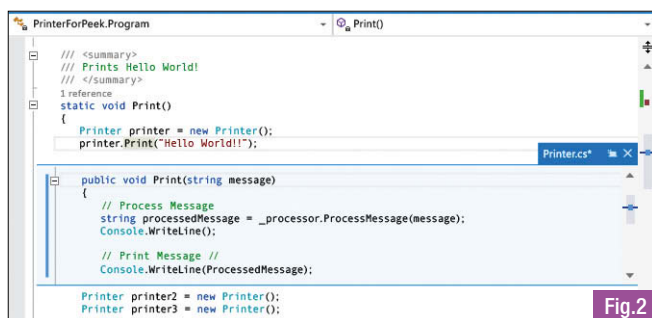


Fig.2

tant ainsi la qualité et simplifiant la maintenance (Fig.3).

Visual Studio Test Professional

Elle est conçue pour les professionnels du test et de la qualité. Cela englobe les campagnes de tests, la gestion, l'organisation et l'exécution du suivi des cahiers de recettes.

Elle vous propose une gamme d'outils adaptés à chaque acteur :

- Le « Test Runner », entièrement instrumenté et configurable, vous permettra d'effectuer la capture des enregistrements détaillés des étapes exécutées. Vous pourrez par la suite en automatiser certains et interrompre rapidement leur exécution pour remonter un bug (sous forme de « Work Item ») directement dans TFS.
- Avec le « Gestionnaire de cas de test », vous pourrez définir, saisir, valider et suivre l'état de vos différents cas de tests (manuels, automatiques ou exploratoires) que vous pourrez organiser sous forme de campagne de test (ou cahier de recette).
- « Lab Management » vous permettra de créer vos environnements et d'automatiser leur déploie-

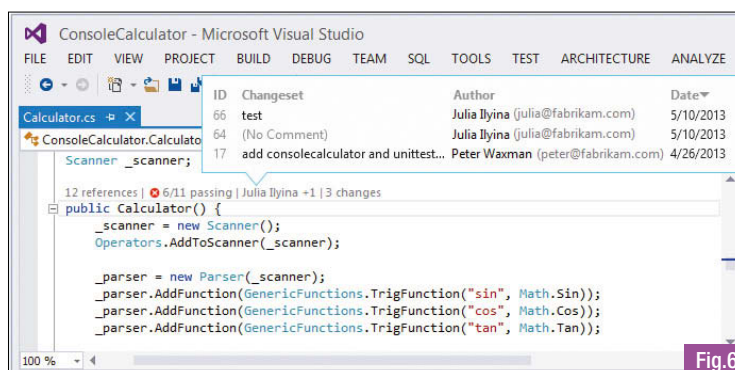


Fig.6

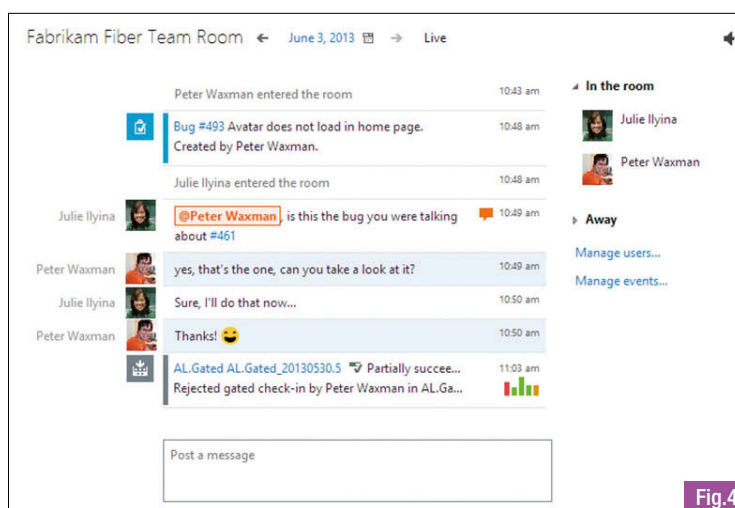


Fig.4

ment pour les testeurs et les développeurs. Enfin, vous pourrez également lancer des campagnes de « feedback » pour tester rapidement une fonctionnalité par vos utilisateurs finaux ou vos équipes marketing afin de les prendre en compte plus rapidement dans vos développements.

Visual Studio Premium

La version Premium inclut toutes les fonctionnalités des versions précédemment présentées. Elle inclut deux types de fonctionnalités en plus : celles directement intégrées à votre IDE et celles liées à l'utilisation de TFS ou Visual Studio Online :

Dans votre IDE, vous aurez :

- ▶ Un arsenal complet d'outils permettant de calculer et mesurer la qualité de vos applications : vous pourrez lancer des analyses statiques, mesurer la couverture de code et détecter les « copier/coller » dans vos applications.
- ▶ Le « Fakes Framework » vous aidera à améliorer vos tests unitaires en simplifiant l'isolation des dépendances.
- ▶ La possibilité de faire l'automatisation des tests graphiques (CodedUI tests) vous permettra de compléter votre arsenal de tests.
- ▶ Développement d'applications Office et SharePoint.

Dans TFS et Visual Studio Online, vous aurez accès à la quasi-totalité des fonctionnalités de ces produits simplifiant et fluidifiant grandement le travail d'équipe. Par exemple, vous pourrez utiliser :

- ▶ Le « Code Review » pour valider et échanger les points de vue sur vos différents développements tout en conservant l'historique.
- ▶ L'« Agile Portfolio Management » vous permettra de mieux découper et organiser de

manière hiérarchique vos backlogs ce qui s'avère très utile lorsque vous avez plusieurs équipes qui travaillent sur des périmètres différents du même projet.

- ▶ Votre portail web TFS sera également enrichi de graphiques permettant de visualiser rapidement le nombre de bugs, le statut de vos Builds, l'état des tâches. Cela vous permettra d'avoir une vision globale de l'avancement de votre projet.
- ▶ « Team Room » : chat puissant permettant à la fois la communication entre différents acteurs « humains » et services techniques. Il sera, par exemple, possible de permettre à TFS d'afficher des messages techniques liés au statut de vos Build, le changement d'état d'un « Work Item » dans le même fil de discussion entièrement configurable (Fig.4).

Visual Studio Ultimate

Cette version représente la meilleure des offres car elle englobe toutes les possibilités offertes par la version Premium en y ajoutant des fonctionnalités liées à la haute disponibilité, les performances, à la montée en charge et à la complexité des applications d'entreprise.

Vous aurez à votre disposition :

- ▶ De puissants outils pour faire des tests de charge et de performance. Vous détecterez plus facilement les goulots d'étranglement dans vos applications pour les rendre plus performantes et pourrez évaluer si votre application tiendra la charge cible.
- ▶ « Intelitrace » permettra de collecter toutes les informations liées à vos sessions de débog que vous jugerez utiles (trace système, événements, état de la mémoire,...) simplifiant ainsi vos diagnostics.
- ▶ Créer des diagrammes UML et d'architecture

pour rendre plus lisible l'architecture et le fonctionnement de vos applications.

- ▶ Améliorer votre productivité avec les deux nouveautés que sont « Code Map » (Fig.5) et « Code Lens » (Fig.6) qui enrichissent encore plus le feedback apporté.

VISUAL STUDIO ONLINE

Visual Studio Online, est le nouveau nom de « Team Foundation Service ». Cette version offre quasiment les mêmes fonctionnalités qu'un TFS mais entièrement hébergé dans le Cloud : vous n'aurez donc plus de souci avec la gestion de l'infrastructure (serveurs, agents, licences,...) et pourrez en quelques secondes, créer un environnement d'équipe opérationnel et accessible de presque n'importe où.

Cette version se décline en 3 offres :

- ▶ Basic : Elle vous permettra de réaliser les tâches les plus simples telles que le stockage de vos sources, la gestion de votre Backlog et l'exécution de Builds.
- ▶ Professional : elle vous permet de connecter 10 utilisateurs en plus des utilisateurs Basic et inclut un abonnement mensuel à l'IDE Visual Studio Professional.
- ▶ Advanced : cette version vous permet la gestion de projets plus importants et développés sur différents IDE tels que Visual Studio, Eclipse et XCode. Vous pourrez également utiliser la gestion de portefeuille Agile pour diviser les projets volumineux en travaux que vous pouvez répartir entre plusieurs équipes. Les salles d'équipe fournissent des mises à jour en temps réel des événements du projet, ce qui permet à vos équipes de développement de rester connectées.

CONCLUSION

On constate que Visual Studio et TFS (ou Visual Studio Online) sont de plus en plus imbriqués : certaines fonctionnalités TFS sont directement liées à votre niveau de licence Visual Studio. Cette interdépendance offre des possibilités uniques car elle permet l'intégration d'« aspects » complexes liés à la production logicielle tels que les tests fonctionnels, l'intégration des méthodologies Agiles, ou encore des éléments liés à DevOps. Enfin, le développement de ces outils dans le « Cloud » rend ces bonnes pratiques accessibles aux entreprises de toutes tailles.



Fathi Bellahcene
Manager | MVP C#
chez Cellenza
Software
Development Done Right
Son Blog :

<http://blogs.codes-sources.com/fathi>

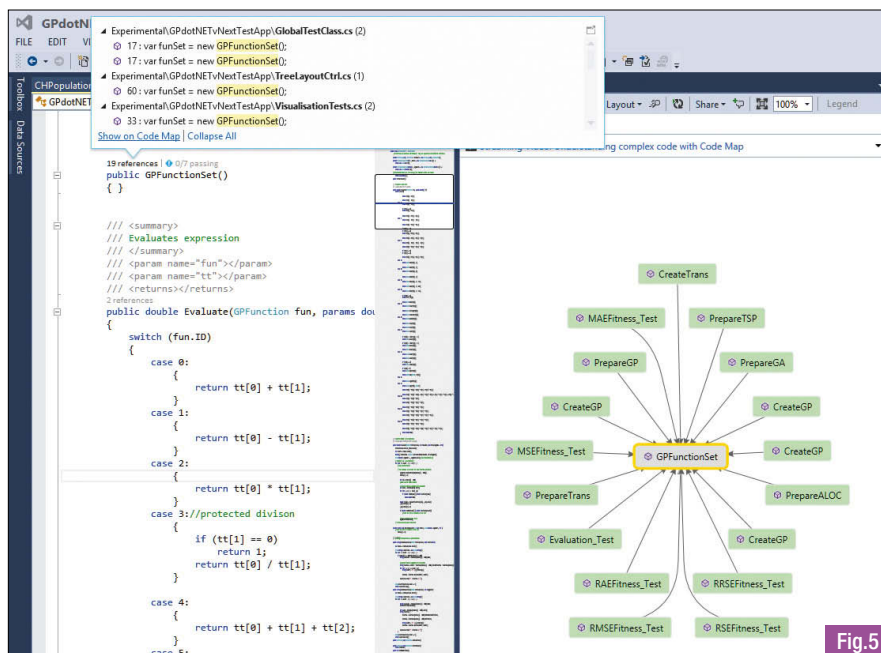


Fig.5

Tous les environnements, tous les écrans et le multiplateforme

Visual Studio 2013 est la version la plus évoluée à ce jour des outils de développement de Microsoft, non seulement au niveau des outils, mais aussi des plateformes qu'il cible. Revue générale de ce qui est livré dans la boîte.

Le 64 bits !

Cela peut sembler bizarre, mais Visual Studio 2013 est la première version de Visual Studio où il est possible de faire de l'« edit & continue » en mode 64 bits. L'évolution de Windows et du matériel informatique impose de plus en plus de développer en 64 bits. C'était donc très pénalisant du point de vue de la productivité de ne pas avoir accès à cette fonctionnalité.

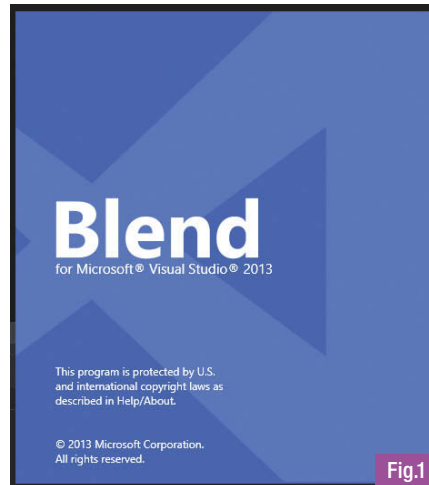
Le desktop

Le winform est la technologie historique pour produire des applications desktop. Bien que celle-ci soit dépassée par les technologies récentes (HTML 5 pour les tablettes Win RT et WPF pour les PC sous Windows 8), celle-ci reste supportée par cette version de Visual Studio (elle ne reçoit bien évidemment plus d'évolution). En managé, WPF tient toujours le haut du pavé et est toujours massivement supporté par Microsoft et les éditeurs tiers. Le Framework .NET 4.5.1 étant supporté à partir de Windows Vista SP2 & Windows Server 2008 SP2, c'est la plateforme de choix pour vos applications .NET en mode desktop. Il est, bien sûr, toujours possible de développer des applications en C++ — qui vient de passer en version 11 —, mais nous verrons que les plus gros changements concernant ce langage ne sont disponibles que pour les tablettes et téléphones. Côté design, Blend est maintenant intégré à la distribution de Visual Studio 2013 (Fig.1).

Le web

Le développement des applications web a énormément évolué en 4 ans depuis l'arrivée du Framework ASP.Net MVC. Le webform est toujours disponible et subit le même sort que winform : aucune évolution, mais vous pourrez toujours produire des applications en ASP.NET. L'ASP.NET MVC s'appuie sur les standards du web que ce soit au niveau des patterns de développement ou des frameworks et outils utilisés :

- ▶ JSON
- ▶ REST
- ▶ jQuery
- ▶ Bootstrap



- ▶ HTML 5
- ▶ SignalR
- ▶ OAuth
- ▶ OWIN/Katana

Visual Studio 2013 fournit un nouvel assistant de création de sites web. Avant, il fallait faire un choix entre ASP.NET classique et MVC ou web API, maintenant, il n'y a plus qu'un seul type de projet et c'est dans l'assistant que l'on sélectionne les composants (y compris les méthodes d'authentification) que l'on veut pour nos applications (Fig.2). ASP.NET MVC 5 s'appuie maintenant sur le framework Katana qui permet, entre autres, d'ajouter des systèmes d'authentification externes comme Facebook ou plus « corporate » comme Office 365.

Bootstrap est inclus de base dans le modèle de projet pour fournir la gestion des applications mobiles pour vos designs. Microsoft est clairement engagé à utiliser les frameworks les plus reconnus dans ses applications.

Silverlight

Toujours présent, contrairement à ses détracteurs, Silverlight répond à des besoins qui sont incompatibles avec les autres technologies. Il reste un élément non négligeable de la galaxie XAML.

Windows Phone

Plusieurs types d'applications Windows Phones sont supportés (en version 8) :

- ▶ .NET
- ▶ C++/XAML
- ▶ HTML5/Javascript

Il est même possible d'y intégrer Direct3D (essentiellement pour les jeux). L'intégration avec le store est incluse dans Visual Studio, ce qui permet de débloquent les téléphones pour développer. Tout récemment, Microsoft a publié de nouveaux émulateurs HD pour développer des applications sur les tous nouveaux téléphones Full HD comme le Nokia 1520.

Windows Store App

Comme pour Windows Phone, les technologies disponibles sont les suivantes :

- ▶ .NET
- ▶ C++/XAML
- ▶ HTML5/Javascript

Les applications utilisent le framework WinRT. Une partie de ce framework est cependant utilisable pour les applications de type Desktop. Attention toutefois, Visual Studio 2013 ne supportant que le développement d'applications Windows 8.1 il faudra rester sur Visual Studio 2012 pour les développements visant Windows 8.

Microsoft Office/Sharepoint

Visual Studio 2013 supporte les versions 2013 d'Office et 2013/2010 de Sharepoint. En plus du modèle classique d'application Office, Visual Studio 2013 apporte une nouveauté : les « office et sharepoint apps ». Contrairement aux addin classiques, ces applications sont en pur HTML/JavaScript et peuvent s'intégrer dans les versions web des outils, comme Outlook Web Access.

Windows Azure

Visual Studio permet de créer des services Cloud. Pour les web roles, Azure se base sur les dernières versions d'ASP.NET MVC. Tout ceci est donc très cohérent avec ce que l'on peut faire sur IIS « on premise ». Une grande nouveauté de Visual Studio 2013 est l'intégration d'Azure dans l'IDE. Il est maintenant possible, une fois le compte lié à Windows Azure renseigné dans Visual Studio, de modifier la configuration de son

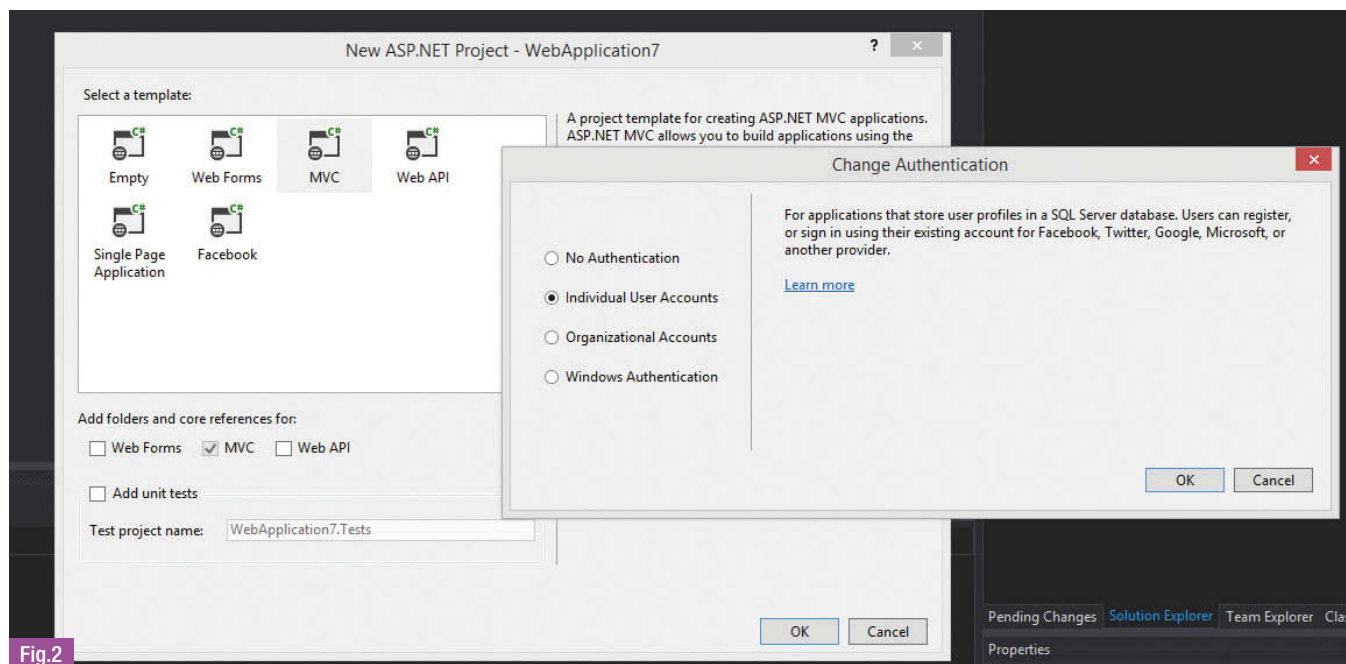


Fig.2

abonnement sans passer par l'interface web. Un peu comme cela était déjà possible avec le « Server Explorer » dans les versions précédentes. Cette interface n'a pas vocation à complètement remplacer les fonctionnalités de l'interface web de Windows Azure, mais peut remplacer les fonctionnalités liées au développement : création de services de type Cloud, de sites web, etc.

Lightswitch

Lightswitch s'étoffe version après version. Il est maintenant disponible en version Silverlight ou HTML5. Cet outil répond au besoin d'applications types formulaires de données. Ce sont typiquement des applications que l'on aurait écrites avant avec Access. L'avantage du web dans Lightswitch est qu'il va pouvoir s'adapter aux différents types de terminaux comme les mobiles et les tablettes.

iOS/Android via Xamarin

Cela a été annoncé au lancement de Visual Studio 2013. Microsoft et Xamarin s'associent pour fournir directement dans l'IDE les outils de développement pour iOS et Android. Forcément, il faudra toujours un Mac pour compiler et déployer sur un iPhone ou un iPad, mais tout sera pilotable depuis Visual Studio (Fig.3).

Plusieurs plateformes à la fois : les « Portable Class Libraries » (PCL)

Avec toutes ces plateformes, il est évident que certains de nos codes sources vont se retrouver sur plusieurs plateformes simultanément. À moins de recopier les fichiers

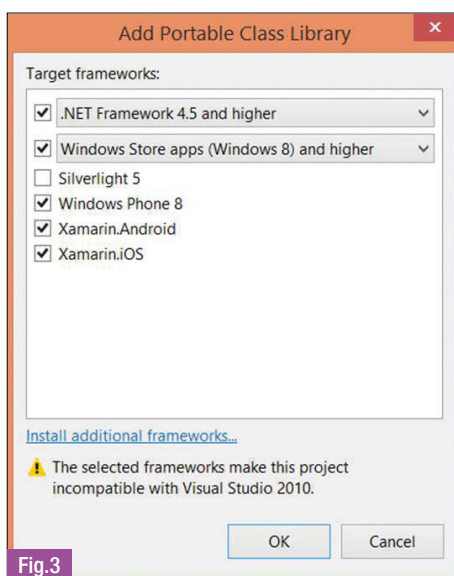


Fig.3

ou de passer par des liens, il va falloir dupliquer les projets, ce qui va à l'encontre de la qualité logicielle. Cette problématique est maintenant résolue avec les « Portable Class Libraries ». Avec les « Portable Class Libraries », vous pourrez créer des composants qui pourront être utilisés sur plusieurs plateformes différentes. Vos développements seront ainsi mutualisés et les plateformes supportées sont entre autres :

- ▶ .NET Framework (à partir du 4)
- ▶ Windows Store App
- ▶ Windows Phone 8

SQL Server

Pendant longtemps, SQL Server avait une version de retard sur Visual Studio. Avec Visual Studio 2012, Microsoft avait sorti une nouvelle génération d'outils :

- ▶ SQL Server Data Tools « SSDT »

SQL Server Data Tools for Business Intelligence « SSDT BI »

Le premier cible le moteur relationnel, et apporte pratiquement la même richesse de développement qu'un langage compilé. Le second cible SSIS, SSRS et SSAS. Au moment de l'écriture de cet article, seule la partie SSDT est disponible sur Visual Studio 2013. Mais elle permet de cibler les bases SQL Server classiques comme la base SQL Azure. La partie BI sera disponible, mais la date n'est pas encore annoncée. Une fois gérée sous forme de projet, une base de données peut être traitée à la manière de code source classique : compilation, validation statique, refactoring. Il est même possible de comparer une base à un schéma généré (au format DACPAC) et de créer les scripts de migration.

Conclusion

Visual Studio est un outil complet qui permet de répondre avec un seul IDE à l'ensemble des problématiques et interfaces de l'entreprise. Contrairement aux premières versions de Visual Studio, les choix technologiques sont maintenant à faire du côté des développeurs qui ont la possibilité de composer les différents types de projets et de langages. La possibilité d'ouvrir des projets d'anciennes versions de Visual Studio sans que cela migre les projets permet en plus une transition en douceur vers cette version.



Michel Perfetti
Manager | MVP Visual Studio ALM chez
Cellenza
Software Development Done Right
Son Blog: <http://www.buildmeinfamous.net>
Twitter: @miiitch

De VS 2010 à 2013 : migration des projets existants, installation

La première étape avant de pouvoir utiliser Visual Studio 2013 est bien sûr de l'installer. Vient ensuite la phase de migration des projets existants pour les mettre à jour. Ce sont ces différents aspects que nous verrons ici.

INSTALLATION

Prérequis

La première vérification à effectuer avant l'installation est le support ou non de la plateforme cible. Mais ce point ne devrait pas être bloquant, car l'installation peut s'effectuer aussi bien sur Windows 7 que 8. Concernant les systèmes serveurs, Windows Server 2008 R2 ou versions supérieures, sont supportés. Ensuite, il faudra faire les vérifications d'usage telles que la place disponible pour l'installation.

Par contre, dans le cas de développement Windows Store ou Windows Phone, il faudra faire attention au système qui hébergera Visual Studio. Le système le plus adapté est Windows 8, et ce pour deux raisons :

- ▶ La première est relativement évidente : seul Windows 8 peut exécuter des applications Windows Store, ce qui le rend de fait quasiment indispensable.
- ▶ La seconde concerne le développement d'application Windows Phone qui fera appel à l'émulateur pour tester les applications. Pour fonctionner, cet émulateur a besoin d'Hyper-V, ce qui rend une nouvelle fois Windows 8 inévitable. Mais si pour différentes raisons (édition standard de Windows, non prise en charge de la virtualisation par le processeur...) il n'est pas possible d'utiliser Hyper-V, il est néanmoins possible de tester et de déboguer les applications directement sur un téléphone.

Installation

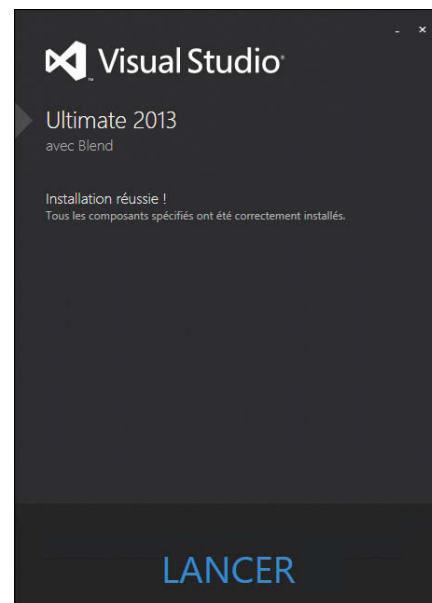
Au fil des versions, l'installation de Visual Studio s'est simplifiée. Avec la version 2013, elle est devenue très aisée, elle ne comporte en effet plus que trois étapes. Dans un premier temps on choisit le répertoire d'installation, puis on choisit les composants à installer et l'installation est terminée. Difficile de faire plus simple.

Windows Azure

Il existe un autre moyen d'utiliser Visual Studio 2013 : Windows Azure. Des machines virtuelles préconfigurées contenant une installation prête à l'emploi sont disponibles dans la galerie. Même si cette solution ne conviendra probablement pas à une utilisation quotidienne, elle permet par exemple d'évaluer cette nouvelle



Installation : les options



Installation terminée

version. Imaginons que vous souhaitiez migrer vos postes, vous pouvez avec cette solution vérifier que tous vos projets seront correctement pris en charge. Vous pouvez également vous en servir en déplacement pour toujours avoir sous la main un environnement de travail fonctionnel.

Premier lancement

Lors du premier lancement, un choix s'offre à vous : soit vous connecter en utilisant un compte Microsoft, soit configurer Visual Studio comme auparavant.

Si vous choisissez la deuxième solution, vous aurez le même comportement qu'auparavant, c'est-à-dire choisir pour chaque installation le paramétrage de votre environnement. Par contre si vous choisissez de lier Visual Studio avec votre compte Microsoft, vous profiterez de plusieurs avantages.

Tout d'abord, tous vos paramètres seront stockés en ligne et partagés entre toutes vos installations, plus besoin de les ressaisir lors d'une installation sur un poste différent par exemple. Ensuite, si votre compte Microsoft est associé à un compte MSDN, la licence sera automatiquement installée.

Enfin si vous avez un compte Visual Studio Online, il sera directement accessible sans avoir à ressaisir vos identifiants.

Migration

Cette action va permettre de mettre à niveau l'ensemble des projets afin de les rendre compatibles avec cette nouvelle version. On peut la diviser en deux étapes. La première qui est obligatoire consiste à charger avec la version 2013 son ancienne solution. Lors de ce chargement, Visual Studio va convertir automatiquement les anciens projets. Lorsque cette conversion sera terminée, vous pourrez voir votre projet s'ouvrir, et consulter l'ensemble des fichiers. À ce moment-là, je vous recommande de lancer une compilation, cette dernière validera que la conversion ait réussi. Mis à part un éventuel problème au niveau des dépendances des projets, la compilation devrait aboutir sans erreur. La deuxième qui, elle, est facultative, consiste à mettre à jour le Framework utilisé. En effet, il est probable que lors de la migration des projets, vous effectuiez un test complet de votre application pour vérifier son bon fonctionnement ; autant en profiter pour mettre à jour le Framework et ainsi tirer profit des dernières nouveautés du langage.



Pierre-Henri Gache
Consultant chez Cellenza
Software Development Done
Right
Son Blog :
<http://www.pierrehenrigache.com>

Gagner du temps avec Visual Studio 2013

Visual Studio 2013 s'étoffe, comme à chaque version, de nouvelles fonctionnalités. Microsoft a toujours porté une attention toute particulière à la productivité et au confort de développement. Cet article montre comment gagner toujours plus de temps avec cette nouvelle version de l'IDE star de Microsoft.

BROWSER LINK

Microsoft s'attaque à l'une des plus grosses hantises des développeurs Web : la compatibilité multi-navigateurs de leurs applications. Pour aider à surmonter cette contrainte, des solutions existent déjà, mais sont souvent lourdes et très peu ou pas intégrées à Visual Studio. Parmi les plus connues :

- ▶ **Microsoft Expression Web SuperView** qui permet de tester notre application sur différents navigateurs en même temps,
- ▶ **BrowserStack** qui est la référence dans le domaine,
- ▶ **Modern.IE**.

Il est à noter que BrowserStack et Modern.IE offrent la possibilité de choisir l'environnement cible où notre site sera testé. Avec Visual Studio 2013, Microsoft fournit une nouvelle solution nommée **Browser Link**. Celle-ci contrairement aux solutions citées précédemment est totalement intégrée à Visual Studio, et est de surcroît beaucoup plus pratique et facile à utiliser.

Qu'est-ce que Browser Link et qu'a-t-il de plus à offrir ?

Browser Link n'est en réalité qu'un canal entre Visual Studio et un navigateur ou un simulateur qui permet l'échange dynamique de données en temps-réel. Basé sur le framework open source Microsoft ASP.NET SignalR (en version 2.0 avec l'arrivée de Visual Studio 2013), **BrowserLink** permet ainsi de gérer des situations où deux machines communiquent via une des quatre technologies ou techniques suivantes :

- ▶ WebSockets,
- ▶ Server-Sent Events,
- ▶ ForeverFrame,
- ▶ Long Polling

Grâce à ce choix technique, **Browser Link** devient une solution modulable et évolutive qui étouffera sa base de support gérée au fil du temps.

À l'utilisation ça donne quoi ?

Pour utiliser **Browser Link** (dans le but de tester une application Web sous différents moteurs), il faut dans un premier temps installer tous les navigateurs sur lesquels l'application Web devra

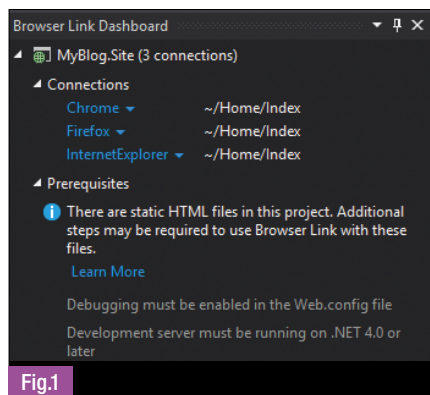
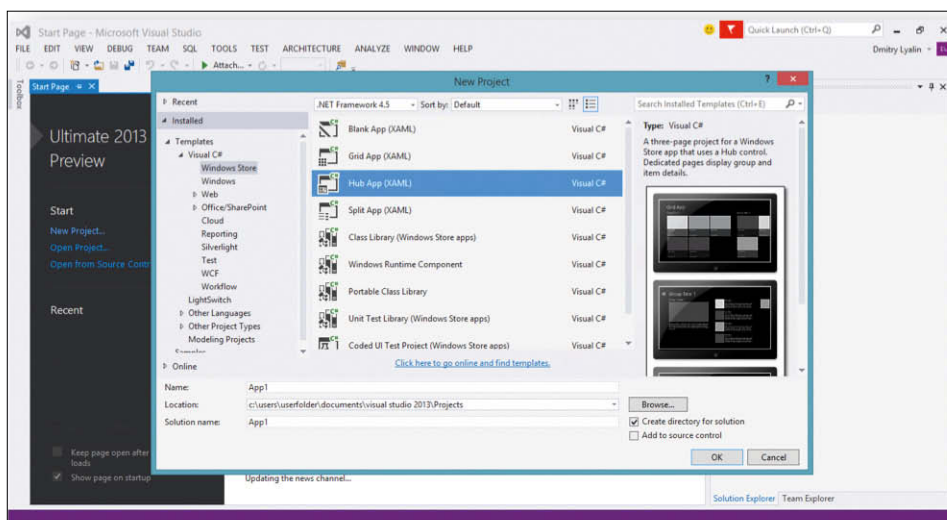


Fig.1

être testée (il est à noter que depuis la version 11 d'Internet Explorer, le navigateur de Microsoft est capable de switcher entre une quinzaine de user agent). Ensuite tout est géré : une fois qu'un site en cours de Debug est ouvert sur un navigateur (ou un onglet du navigateur), ce dernier est automatiquement détecté par Visual Studio 2013. Il devient visible dans la fenêtre **Browser Link Dashboard**, avec en face de chaque navigateur, la page ciblée **Fig.1**. Cette fenêtre, en plus de fournir une visu sur tous les navigateurs détectés par Visual Studio, permet de rafraîchir chacun d'eux indépendamment en cliquant sur l'un d'eux puis sur **Refresh**. Plus intéressant il est possible de rafraîchir tous les navigateurs en même temps en cliquant sur le bouton **Refresh** qui s'ajoute par défaut à côté du bouton de Debug dans Visual Studio, ou de façon plus pratique en utilisant le raccourci clavier **CTRL + ALT + Entrée**. **Browser Link** est une solution adaptée

aux contraintes actuelles. L'usage de l'informatique se tourne de plus en plus vers la mobilité et il n'est donc plus rare de devoir réaliser un site dit **responsive**. Pour aider au développement face à cette nouvelle contrainte, **Browser Link** permet de visualiser le rendu de son application Web sur iPhone et iPad via le simulateur pour Windows **Electric Mobile Studio** (gratuit pendant 7 jours). Pour utiliser ce simulateur, il faut dans un premier temps le télécharger sur <http://www.asp.net/mobile/device-simulators>. Une fois installé, il suffit d'ajouter le simulateur aux références de **Browser Link** via la petite flèche à côté du bouton **Refresh** -> **Browse with** -> **Add**. La fenêtre qui apparaît permet de configurer l'application supportée par **Browser Link** :

- ▶ **Program** : Chemin vers l'exécutable du support visé (dans notre cas **Electric Mobile Studio C:\Program Files (x86)\Electric Plum\Electric Mobile Studio 2012\ElectricMobileStudio2012.exe**)
- ▶ **Arguments** : arguments à passer pour le lancement (vide pour **Electric Mobile Studio**)
- ▶ **Friendly name** : Nom qui apparaîtra dans la liste des supports gérés

AMÉLIORATION DE L'ÉDITEUR XAML

Après le développement Web, Microsoft s'attaque au XAML. Base des applications Windows 8 et Windows Phone, le XAML est devenu quasi-incontournable dans le développement Microsoft. Une fonctionnalité

(déjà présente dans le très populaire **ReSharper** et dans diverses extensions) semblant anodine, mais pourtant fortement attendues par les développeurs s'ajoutent à l'IDE. Il est désormais plus aisé de modifier le type d'un contrôle XAML. Auparavant la modification du type du contrôle à la balise ouvrante ne modifiait pas la balise fermante (et vice-versa), ceci était fortement désagréable dans de gros fichiers XAML. Désormais une modification sur l'une des balises répercutera la modification sur l'autre. Microsoft a également amélioré le support de l'IntelliSense dans du code XAML, notamment lors d'un Binding. Comme certaines extensions le proposaient déjà, il est désormais possible d'accéder par auto-complétion aux propriétés d'un de nos modèles MVVM dans du code XAML. Avec **Browser Link** et les améliorations apportées sur la partie XAML, Microsoft montre son intérêt au confort et à la productivité. Dans la même optique, il est possible depuis Visual Studio 2010 de modifier le code d'une application durant une phase de debugging via la fonctionnalité **Edit And Continue**, et de voir en direct les résultats des modifications.

X64 EDIT AND CONTINUE

La fonctionnalité **Edit and Continue** était auparavant réservée au développement d'applications X86, elle s'adapte désormais au développement d'applications x64. Cette fonctionnalité aide les développeurs à modifier un comportement ou à corriger en direct un dysfonctionnement survenu lors du débogage sans pour autant relancer l'application pour appliquer les modifications.

Comment l'activer ?

Par défaut, cette fonctionnalité est activée dans Visual Studio 2013. Elle peut toutefois être activée ou désactivée manuellement en cochant la case **Edit and Continue** dans **Tools**

-> **Options...** -> **Debugging** -> **Edit and Continue**

Note : *Petite subtilité pour les applications Web, si une fenêtre d'alerte apparaît et empêche de modifier le code en mode Debug, cela signifie qu'Edit and Continue n'est pas activé pour le projet Web. La fonctionnalité est activable dans l'onglet Web des propriétés de projet et en cochant la case **Enable Edit and Continue** (cette option est désormais activée par défaut).*

Concernant Azure, seuls les Web Roles sont supportés, mais Microsoft prévoit de fournir un support des Worker Roles dans l'avenir.

Comment l'utiliser ?

Rien de plus simple, il suffit d'insérer un point d'arrêt à l'endroit souhaité de l'application et il sera possible de modifier le code en direct.

PEEK DEFINITION

Place au confort d'utilisation : **Peek Definition** à la manière du **Go To Definition (F12)** permet de naviguer vers une méthode, à la différence près qu'il permet de ne pas changer de page ou d'onglet en s'insérant directement dans une pop-in appelée **Peek Window** au niveau du contexte courant. Pour profiter de cette fonctionnalité il suffit de presser **ALT+F12** ou de faire un clic droit -> **Peek Definition** sur la méthode désirée. Afin d'aller encore plus en profondeur dans le code il est aussi possible de lancer un **Peek Definition** dans... une **Peek Window**. Une barre de navigation apparaît au-dessus de la **Peek Window** pour connaître le niveau de définition courant et pouvoir remonter dans le code.

BARRE DE DEFILEMENT

Toujours sur le thème de la navigation : aucune surprise pour les personnes utilisant les **PowerTools** auparavant, nouveauté pour les autres, Visual Studio 2013 offre une nouvelle jeunesse à sa barre de défilement.

Elle permet désormais de naviguer plus facilement dans le code. Celle-ci ne se résume pas à sa fonction première à savoir... scroller, elle s'éttoffe et devient plus intelligente en permettant désormais :

- ▀ de voir l'intégralité du code source en mode réduit (de la même manière que sur l'éditeur de texte **Sublime Text**),
- ▀ de voir via une ligne bleue où l'utilisateur se trouve dans le code,
- ▀ de voir via un indicateur jaune tous les endroits où le code a été modifié mais pas sauvegardé,
- ▀ d'avoir une prévisualisation sous forme de zoom d'une partie du code de la page au passage de la souris sur la barre de défilement.

Ces petites fonctions semblent gadgets, mais elles permettent d'avoir accès à différentes informations sans changer son contexte et cumuler des va et vient fréquents entre les pages.

Comment l'activer ?

Clic droit sur la barre de défilement puis **Scroll bar option**, cocher **Use map mode for vertical scroll bar** ainsi que **Show Preview Tooltip** Fig.2.

MANAGED RETURN VALUE

Fonctionnalité demandée par de nombreux développeurs via le site **Visual Studio**

```
class Program
{
    static void Main(string[] args)
    {
        int result = Add(One(), Two(), 42);
    }

    public static int One()
    {
        return 1;
    }

    public static int Two()
    {
        return 2;
    }

    public static int Add(params int[] values)
    {
        return values.Sum();
    }
}
```

Fig.3

```
ods.Ftp.UploadFile());
```

```
using (FileStream stream = File.OpenRead(p))
{
    byte[] buffer = new byte[stream.Length];
    stream.Read(buffer, 0, buffer.Length);
    stream.Close();
    using (Stream reqStream = request.GetRequestStream())
    {
```

Fig.2

Uservice. Il est désormais possible d'obtenir facilement le retour des méthodes lors du Debug via le **Managed Return Value**. Déjà, présente pour le C++, elle est généralisée aux langages .NET. Lors du debugging, le retour des dernières méthodes est visible dans la fenêtre **Autos** ou en utilisant la pseudo-variable **\$ ReturnValue** dans la fenêtre d'exécution **Fig.3.**

CODE MAP

Code Map est une fonctionnalité très intéressante pour le debugging. Auparavant pour voir les références d'une méthode dans le code il y avait un classique clic droit -> **View call hierarchy**. Désormais il y a **Code Map** ! **Code Map** permet d'avoir une vue beaucoup plus large et pratique à l'utilisation de la hiérarchie du code. Pour utiliser **Code Map** : clic droit sur la méthode cible -> **Find all reference on Code Map**.

Code Map se présente sous la forme d'un schéma composé de nœuds qui représentent les méthodes. Il est très facile de distinguer les appelants des méthodes ainsi que la hiérarchie des appels. Il est possible de demander à **Code Map** de montrer quelles méthodes appellent une méthode cible, remonter dans la hiérarchie du code, ou montrer toutes les références dans le but d'étoffer le schéma. Sur les gros projets, le résultat peut être anarchique et vite ressembler à un sac de nœuds. Il est possible de déplacer les nœuds manuellement, ou utiliser une mise en disposition rapide qui réorganisera de façon plus harmonieuse les différents nœuds (onglet **disposition**). Enfin et pour une meilleure communication, il est possible d'ajouter des commentaires sur des nœuds, enregistrer le **Code Map** et l'envoyer par email sous format XPS ou image **Fig.4.**

CODELENS

CodeLens est une nouvelle fonctionnalité utilisant des données issues de TFS concernant le projet. Il prend la forme d'annotations affichées directement dans l'éditeur de code. Elle permet de connaître d'un coup d'œil et sans changer de fenêtre :

- ▶ le nombre de références faites dans le projet à la méthode ciblée
- ▶ le nombre de tests passés et échoués
- ▶ le nombre de modifications réalisées sur la méthode
- ▶ l'historique des modifications...

Ces informations étaient auparavant visibles par les développeurs mais nécessitaient plusieurs manipulations et contraignaient les développeurs à changer d'onglet ou d'application **Fig.5.**

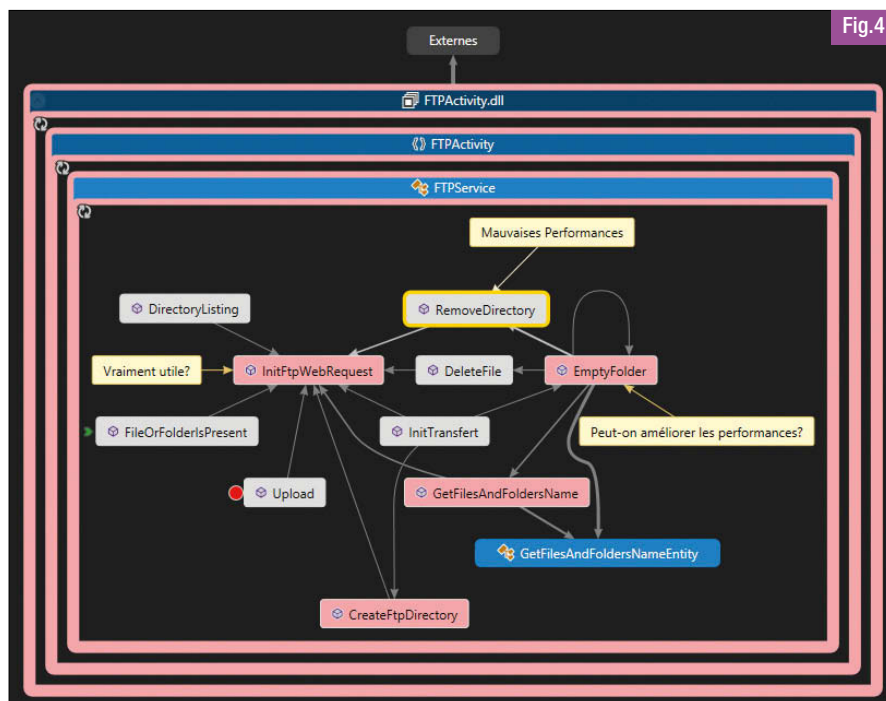


Fig.4

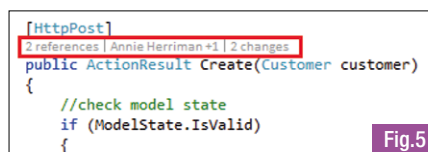


Fig.5

Comment activer et personnaliser les informations remontées par CodeLens ?

Par défaut toutes les informations possibles sont remontées par **CodeLens**. Malgré le nombre important d'informations remontées, l'impact sur les performances est quasi nul. En effet **CodeLens** fonctionne en mode asynchrone et n'est chargé pour une page seulement si celle-ci est ouverte. Donc aucune raison de s'en priver ! La personnalisation des informations remontées par **CodeLens** est disponible dans **Tools -> Option -> Text Editor -> All Languages -> CodeLens**. Par une simple ligne d'informations, **CodeLens** offre un grand nombre de fonctionnalités.

Chercher les références d'une méthode

En cliquant sur le nombre de références ou via le raccourci **ALT+2**, **CodeLens** affiche toutes les méthodes qui référencent votre méthode. Toujours dans un souci de confort, il est possible d'obtenir une preview de chacune des méthodes en les survolant, ou y accéder en cliquant dessus.

Vérifier l'état des tests unitaires

Via le raccourci **ALT+3**, les tests ayant échoués sont clairement signalés par une croix rouge.

Avoir des informations sur les derniers check-in

Il peut arriver qu'un test ne fonctionne pas à la suite d'un check-in réalisé trop rapidement par un développeur ou par une modification du code par vous-même. Pour éclaircir la situation sur la source du problème, **CodeLens** (grâce à sa liaison avec TFS) permet d'obtenir de nombreuses informations utiles comme :

- ▶ Voir l'historique des modifications effectuées et pour chacune des modifications une différence du code ou avoir plus d'informations sur le check-in,
 - ▶ Obtenir le pseudonyme de la dernière personne ayant réalisé un check-in.
- Il peut être aussi intéressant de discuter en direct avec le développeur d'un check-in. **CodeLens** permet d'envoyer un mail d'un seul clic ou de lancer une conversation Lync avec un contact du projet.

Avec **CodeLens**, Microsoft offre une fonctionnalité profitant d'une parfaite synergie entre TFS et Visual Studio dans le but d'offrir un confort d'utilisation et d'améliorer la communication.

CONCLUSION

Ces nouvelles fonctionnalités deviendront, pour beaucoup de développeurs, rapidement indispensables. Suivant la même logique que **Team Foundation Server** et **Visual Studio Online**, l'IDE Visual Studio profite désormais d'un cycle de livraison fortement raccourci, de nouvelles fonctionnalités seront donc à découvrir d'ici peu.

○ Jérémie Landon
Infinite Square



Du prototypage à la production : développer mieux, plus vite, pour une meilleure qualité

On a souvent tendance à partir tête baissée dans la mise en place de l'architecture et du développement en omettant une étape importante qui permet très souvent de s'assurer de la bonne compréhension du besoin et d'éviter ainsi que le résultat de notre dur labeur n'ait comme réponse « ce n'est pas ce que je voulais. ». Cette étape est le prototypage.

Celui-ci se décline sous plusieurs formes pour répondre à des besoins différents :

- Le storyboard pour valider les écrans et la navigation.
- Le développement d'un prototype fonctionnel (POC) pour tester une idée ou un algorithme.

PowerPoint

La création de storyboard permet de s'assurer, en amont des phases de définition de l'architecture et de développement, que l'on a bien compris le besoin des utilisateurs en termes d'écrans, de navigation et de fonctionnalités. Un storyboard est une représentation graphique « au brouillon » des différents écrans et de la navigation. Ils peuvent être réalisés au crayon sur de simples feuilles de papier ou via des outils spécifiques. Ils ont l'avantage d'être très visuels, ce qui simplifie la compréhension par rapport à un ensemble de diagrammes UML par exemple. Microsoft propose depuis 2012 un add-in à PowerPoint permettant de réaliser des storyboards. Celui-ci ajoute un onglet

au menu et propose une bibliothèque de formes reprenant les contrôles standards pour les applications Windows, Windows Store et Windows Phone (Fig.1). Il est bien entendu possible d'étendre cette bibliothèque avec vos propres formes adaptées à votre charte graphique ou avec des formes adaptées à d'autres plateformes (vous pouvez trouver de nouvelles formes sur la galerie Visual Studio à l'adresse <http://bit.ly/1ah4mFR>).

Une fois vos formes et arrière-plans prêts, il suffit de glisser/déposer les formes sur vos slides, d'adapter la taille, de remplir les zones de texte et d'utiliser les fonctions de mise en forme de PowerPoint. Toutes ces manipulations sont des manipulations standard à PowerPoint facilitant grandement la courbe d'apprentissage et la prise en main. On peut utiliser les fonctions de master page de PowerPoint pour, par exemple, définir des modèles avec les éléments standard à tous nos écrans (fond, bandeau, bas de page...) pour ne pas avoir à les recréer à chaque slide, vous pourrez également utiliser

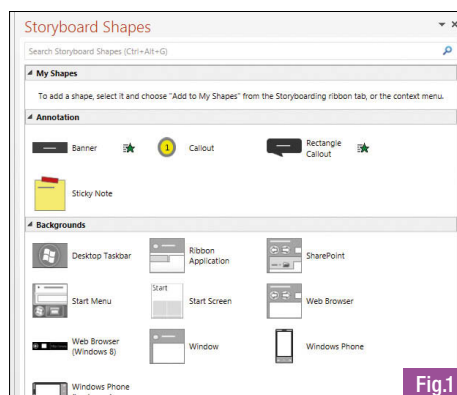


Fig.1

les fonctions d'animation afin de présenter les interactions et la dynamique de navigation (en passant d'une slide à une autre). (Fig.2).

Sketchflow

Pour les applications WPF ou Silverlight, Microsoft propose un autre outil pour la réalisation de prototypage interactif : Sketchflow. Cet outil, initialement proposé dans la gamme de produits Expression, permet de réaliser des prototypes d'interface graphique et de navigation directement depuis l'éditeur Blend (Fig.3). Contrairement à l'add-in PowerPoint, Sketchflow est beaucoup plus proche de l'écriture d'écran XAML et donc plus adapté à des utilisateurs connaissant Blend. Il permet de définir des écrans, de les designer en XAML et d'ajouter facilement, sans code, la navigation entre ceux-ci (au besoin le projet peut s'ouvrir dans Visual Studio pour y ajouter du code). On

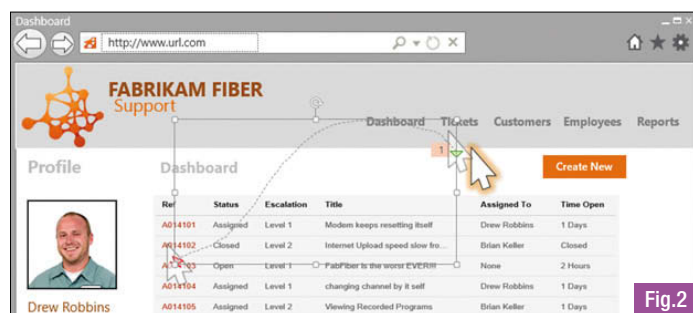


Fig.2

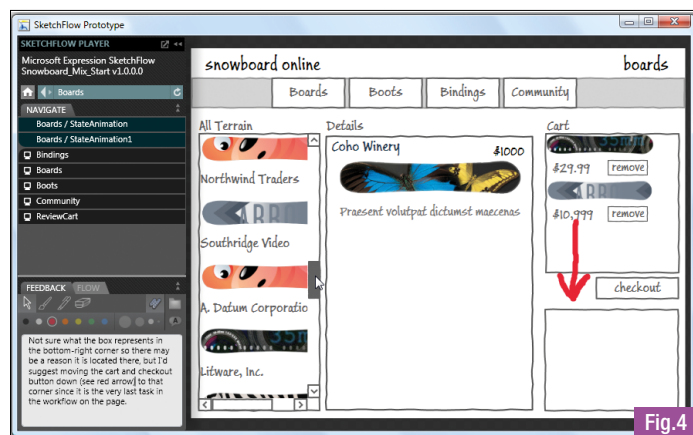


Fig.4

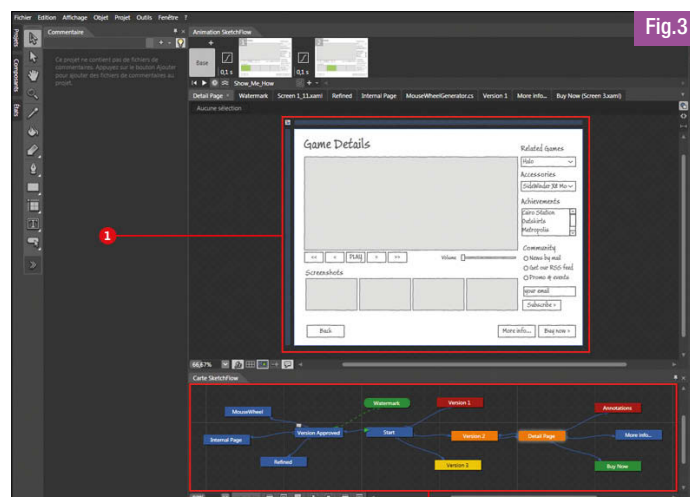


Fig.3

retrouve toute la richesse de Blend pour la réalisation de ces écrans :

- ▶ Bibliothèque de contrôles
- ▶ Styles
- ▶ Animations
- ▶ Behaviors
- ▶ Shapes
- ▶ UserControl

Une fois les différents écrans réalisés, il faut compiler le projet et on peut alors l'exporter sous la forme d'un package Sketchflow qui pourra être distribué et lu avec un lecteur spécifique (Fig.4). Ce lecteur va permettre à son utilisateur non seulement de voir les différents écrans, de naviguer entre eux et de voir les différentes animations mises en place, mais aussi de voir la carte des écrans, d'écrire des remarques et d'annoter directement sur les écrans. Ces remarques pourront alors être retransmises à l'équipe de développement pour qu'elle les prenne en compte. Une fois le prototypage validé, ce prototype étant en XAML, les développeurs peuvent réutiliser le code afin de limiter la réécriture.

CodeFirst/Scaffolding

L'autre mode de prototypage que l'on rencontre est l'écriture rapide d'un bout de l'application afin de valider une idée ou un algorithme. Visual Studio et .NET apportent des solutions simples afin de passer de l'idée à l'application en

quelques secondes : Entity Framework avec Code First et le Scaffolding ASP.NET MVC. Code First permet de mettre facilement en place notre couche d'accès aux données en se basant purement sur des entités métier POCO et une simple classe héritant de DbContext afin d'avoir nos repositories. Code First s'occupera pour nous de créer ou mettre à jour la base de données et l'accès à celle-ci via Entity Framework ; en 5 minutes on obtient notre couche d'accès aux données sans écrire une seule ligne de SQL ! Pour l'interface graphique, Microsoft propose pour ASP.NET MVC un mécanisme de scaffolding permettant de réaliser en quelques clics des écrans pour la gestion d'une entité métier (ajout, édition, visualisation et suppression). Très pratique lorsque l'on souhaite prototyper une application basée sur les données. On pourra ensuite au fur et à mesure des retours utilisateurs améliorer ces interfaces pour avoir une idée correcte du besoin.

DÉVELOPPEMENT

On y est, les développements démarrent... mais pas n'importe comment ! Un bon projet est un projet cadré ! L'agilité impose un cadre de travail clair pour permettre aux équipes de s'engager. Pour faire simple : il n'y a pas de place pour l'improvisation en agile !

L'utilisation d'outils adaptés permet un gain de temps et là aussi Microsoft apporte de nombreuses solutions.

de sécurité que le reste de TFS.

Vous avez en plus la possibilité d'avoir plusieurs « repository » Git pour un même « team project » et la possibilité d'attacher des « work items » aux commits afin d'avoir une traçabilité au niveau de vos exigences.

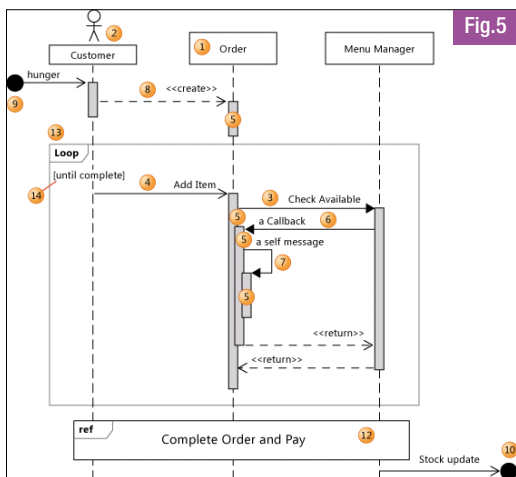
Architecture

L'une des contraintes principales d'une application est sa structure ! Même si elle est bien connue de l'équipe, c'est souvent la plus dure à faire respecter. Pour aider dans sa mise en place et son suivi, il y a plusieurs pistes. On commence par du classique : les diagrammes UML (Fig.5). Depuis Visual Studio 2010 un nouveau type de projet est disponible afin de gérer ces diagrammes. On va pouvoir modéliser son application pour fournir une structure cohérente, centralisée et standardisée via les diagrammes suivants :

- ▶ Diagramme de cas d'usage
- ▶ Diagramme de séquence
- ▶ Diagramme d'activités
- ▶ Diagramme de composants
- ▶ Diagramme de classes

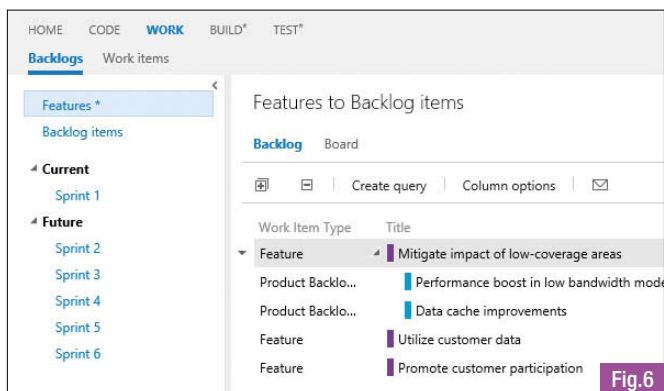
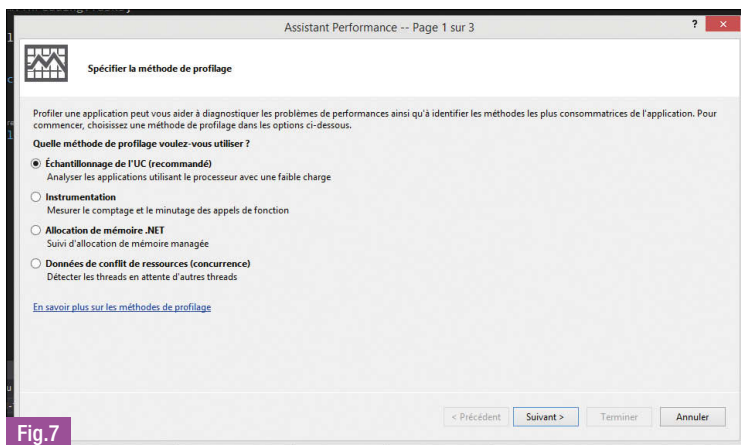
Notre projet va contenir un dictionnaire d'éléments (acteur, cas d'usage, sous-système, classe, interface...) que l'on pourra utiliser dans les différents diagrammes. Comme pour le prototypage il va être possible de rattacher les éléments du modèle ou les diagrammes à des « work items » afin d'attester de la traçabilité dans notre projet.

Jusque-là on restait dans une configuration classique, mais Microsoft propose aussi un autre type de diagramme inédit : les diagrammes de couche (« layer diagram »). Ces diagrammes permettent de définir des contraintes sur la structure du logiciel (nom des composants, espaces de noms, dépendances) qui pourront être validées à la compilation et au moment de la build ! Cette validation est une grande avancée car généralement ce genre de diagramme traîne dans une obscure documentation qui n'est jamais à jour, voire jamais consultée. Comme pour les éléments UML, ces



Contrôleur de source

Visual Studio et Team Foundation Server 2013 supportent, en plus du contrôle de source classique (nommé maintenant « TFVC » pour « Team Foundation Version Control »), Git dont le modèle distribué apporte énormément de flexibilité. Git est intégré, comme TFVC, dans les bases de données de TFS : vous n'aurez donc pas besoin d'un système de backup particulier. De plus, TFS apporte une fonctionnalité très importante à Git : le système d'autorisation. Même si les types de droits sont différents avec TFVC, l'accès à Git utilise le même mécanisme



diagrammes peuvent être attachés à certains éléments des « work items » qui permettront de connaître l'intention de certaines contraintes.

Suivi de projet

Depuis la version 2012, Visual Studio et TFS vivent une révolution : les outils de gestion de projets sont de plus en plus tournés vers l'agilité avec le support des artefacts suivants (en suivant la terminologie Scrum) :

- Le backlog,
- Le taskboard,
- La gestion des sprints.

Avec la version 2013, nous avons maintenant un niveau supplémentaire d'abstraction : la gestion des fonctionnalités (Fig.6). Cela nous permet d'agréger les éléments du backlog sous un même élément. Cette information apporte encore plus de qualité car il est même possible d'avoir une fonctionnalité qui va être développée par plusieurs équipes simultanément via des « User Stories » séparées. Par ailleurs, on va pouvoir ainsi exposer un backlog plus haut niveau pour le management et les sponsors sans rentrer dans le détail.

Depuis 2012, les « work items » en dehors des éléments agiles et en particulier les requêtes étaient moins mis en avant. Maintenant, il est possible de réaliser des graphiques directement depuis le portail web à partir de requêtes, sans avoir besoin de passer par Excel ou par reporting services.

L'utilisation des différents types de « work items » pour le suivi de projet va permettre une hiérarchisation de l'information et une traçabilité complète facilitant les développements et améliorant la qualité globale du produit. On pourra effectivement partir d'une fonctionnalité haut niveau à laquelle on aura associé quelques storyboards qui sera déclinée sous forme de « User Stories » basées sur ces storyboards puis de tâches techniques et de cas de tests.

SQL Server Data Tools

SQL Server Data Tools permet de développer pour SQL Server avec les mêmes processus garantissant la qualité, qu'avec les outils classiques de développement (intégration continue, tests unitaires...). SQL Server souffre du fait même de son fonctionnement : le code est mélangé aux données et la mise à jour se fait élément par élément. SSDT apporte une validation globale du schéma non fourni par SQL Server. En plus de ce service, SSDT fournit de l'autocomplétion, la possibilité de travailler soit en design soit en code pur, du refactoring et même des tests unitaires.

Qualité

Visual Studio est doté d'outils permettant de garantir la qualité du logiciel délivré. L'analyse statique du code généré permet d'identifier de futurs problèmes (performances, architectures) qui peuvent être réglés très tôt dans le cycle de développement. Il ne faut pas oublier que plus un problème est réglé tôt, moins il est coûteux à corriger. Visual Studio possède aussi toute une gamme d'outils d'analyse de performance qui s'étoffe d'année en année. En plus des classiques analyses par échantillon ou instrumentation, on retrouve maintenant l'analyse de la mémoire allouée et l'analyse du comportement lors de l'accès concurrent à des ressources (Fig.7).

Portable Libraries

Il est maintenant possible de partager des composants entre l'ensemble des plateformes de Visual Studio. Cette initiative est assez ancienne, mais il y a à présent des composants du framework, comme la pile HttpClient, qui sont livrés dans ce format. Ce qui veut dire que vous pouvez utiliser le même jeu de composants pour vos applications multiplateformes sans avoir à dupliquer votre code et vos tests (Fig.8).

INTÉGRATION

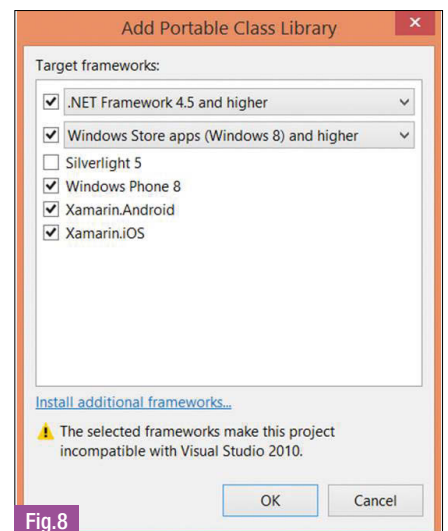
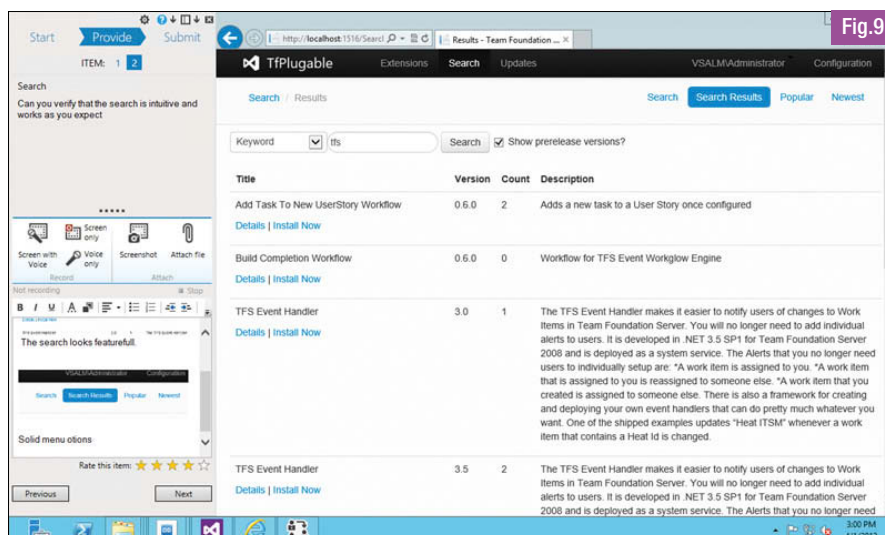
Une fois le travail du développeur terminé, il faut l'intégrer à celui de l'équipe. Cette partie est critique car la qualité du logiciel dépend surtout de la qualité des tests et des analyses réalisées après le développement. Plus l'analyse est poussée et automatique, plus tôt le logiciel est prêt à être livré aux utilisateurs et validé.

Build

La build reste l'élément essentiel de l'intégration. Team Foundation Server 2013 est livré avec, en plus des processus de Build 2012, de deux nouveaux processus simplifiés : un pour TFVC et un autre pour Git. Ce processus de build va permettre aux équipes de générer un package prêt à être installé pour arriver dans la phase de test. Bien entendu on ne va pas que réaliser le package, mais on s'assurera aussi d'un minimum de qualité des développements via l'exécution de tests unitaires et d'outils tiers de qualimétrie tel que NDepend ou StyleCop. On notera qu'il est toujours possible de réaliser des tests fonctionnels automatisés (tests d'interface codés) via l'utilisation de Lab Management.

Tests manuels

De plus en plus d'équipes commencent à mettre en place des tests fonctionnels afin de valider la qualité des livrables. Pour ces équipes, l'écriture en amont de storyboards leur apportent généralement un plus. En effet celle-ci a un support simple, validé par les utilisateurs, qui peut leur servir pour l'écriture des différents cas de tests autour de la navigation ou des interactions au niveau des écrans. On pourra aussi inclure lors de cette phase les designers à l'origine des storyboards et les graphistes à l'origine des maquettes graphiques leur permettant de valider que l'aspect graphique (alignement, marge, pictos...) soit lui aussi respecté.



Feedback client

Avec la mise en place de tests et le prototypage de son application, on ne devrait pas avoir de surprise. Mais le test ultime est bien entendu le commentaire des utilisateurs. Pour cela, Microsoft propose un outil de feedback intégré à Team Foundation Server. Avec cet outil, il est possible de demander un feedback aux utilisateurs pour une partie spécifique. Cela permet d'avoir un retour en continu et de prendre en compte les remarques des utilisateurs dans les plus brefs délais.

Lorsque les utilisateurs reçoivent l'email de la part de l'équipe de développement, ils pourront télécharger un outil gratuit leur permettant de voir la demande de feedback et d'enregistrer leurs retours (Fig.9) avant de renvoyer le tout dans Team Foundation Server.

Ce mécanisme permet de boucler la boucle avec le prototypage du début du projet. L'utilisateur pourra confronter sa vision, normalement retranscrite dans les storyboards et autres prototypes, avec ce qui a été réalisé.

DÉPLOIEMENT

Le déploiement automatique est un challenge pour les équipes. Il y a de nombreux obstacles :

- La complexité du logiciel,
- Le nombre de personnes concernées : les administrateurs base de données, réseaux, les équipes de support de production...
- Du temps dédié à cette automatisation.

Visual Studio propose différents outils que l'on soit « on premises » ou dans le Cloud. Ces outils vont permettre de réaliser régulièrement des déploiements fiables dans les différents environnements jusqu'à la production et ainsi compléter le cycle commençant par le prototypage et finissant par une mise en production et des utilisateurs contents avec un produit adapté à leurs besoins !

CONCLUSION

Plus qu'un outil de développement, Visual Studio et ses plateformes (Team Foundation Server 2013 et Visual Studio Online) sont maintenant les plaques tournantes de vos applications.

Tout le cycle de vie du logiciel est géré et ne concerne plus seulement les développeurs. Du fonctionnel via les prototypes sur PowerPoint en passant par les Product Owners via le portail agile, Visual Studio touche maintenant le monde de l'IT et fournit un suivi et de bout en bout.




 Michel Perfetti
Manager | MVP Visual Studio ALM chez Cellenza
Software Development Done Right

Son Blog : <http://www.buildmeinfamous.net>

Twitter : @miitch



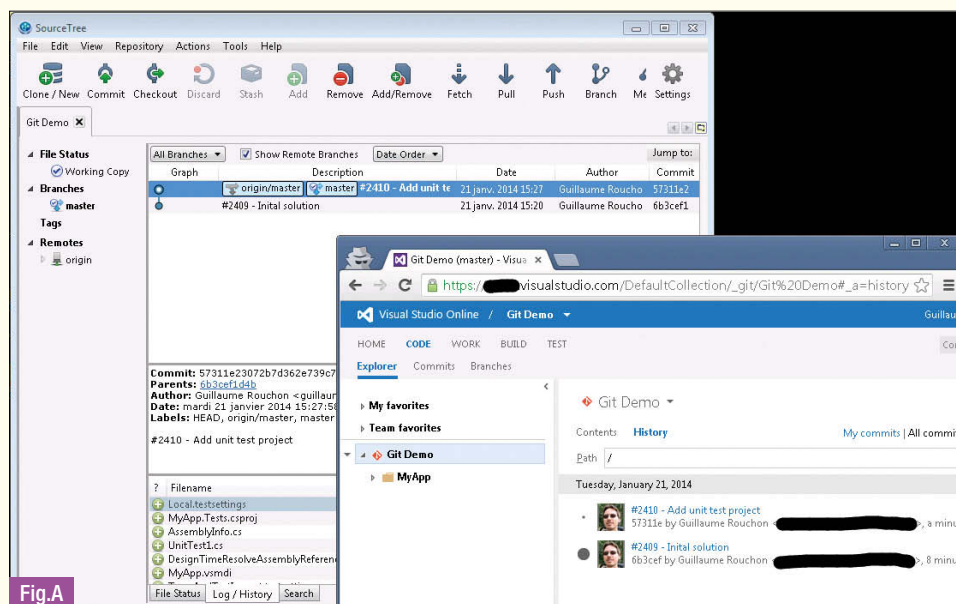
 Guillaume Rouchon
Consultant Senior ALM | MVP Visual Studio ALM chez Cellenza
Software Development Done Right

Son Blog : <http://blog.getza.net>

TFS et Git

Microsoft propose avec Visual Studio Online et Team Foundation Server, en plus de son système de gestion des sources centralisé, l'intégration de Git en temps que système de gestion des sources distribué.

Git est un système de gestion des sources distribué Open Source bien connu. Les systèmes distribués étant de plus en plus populaires, Microsoft a décidé, plutôt que de redévelopper une telle solution, d'intégrer Git. On parle bien ici d'intégration et non de réécriture/modifications ce qui apporte de nombreux avantages dont le principal est qu'il soit compatible avec l'ensemble des clients Git existants, toutes plateformes confondues. TFS expose les interfaces standards de communication de Git permettant alors à n'importe quel client (ligne de commande, XCode, Eclipse, Source Tree (Fig.A...)) de se connecter et de réaliser différentes opérations



(commit, push, pull, clone...). Il est ainsi possible de centraliser l'ensemble de ses développements dans une unique solution et de bénéficier à la fois des avantages de TFS, de Git et des outils de développement que l'on utilise. Afin de s'intégrer avec les outils Git, aucune modification n'a été apportée au protocole. L'ensemble de l'intégration avec les autres composants se fait donc au niveau du serveur. Le repository est persisté dans la base de

données de TFS et une gestion des droits a été ajoutée. Avec le gestionnaire de sources classique de TFS il est possible lors d'un checkin d'associer celui-ci à un « work item » afin d'offrir une traçabilité entre les tâches projet et le code. Pour assurer cette traçabilité avec Git, comme contrôleur de code source dans TFS, il suffit de rajouter comme information dans le commentaire du commit # suivi du numéro du work item que l'on souhaite associer au commit. Cette fonc-

tionnalité est ainsi disponible depuis n'importe quel client Git. Pour la partie Build, Microsoft propose un template de build spécifique se basant sur Git et la manière dont il gère les sources. Il est possible de spécifier la ou les branches à surveiller pour déclencher un build. Avec cette intégration standard, Microsoft fait un premier pas vers l'ouverture aux autres outils de développements, permettant ainsi d'avoir un processus partagé, quelle que soit la technologie.

DevOps : de l'agilité dans votre process de test et de déploiement

Un projet de développement logiciel est normalement toujours composé de 3 grandes phases distinctes :

- Le métier, dont le rôle est d'apporter une réponse à une problématique ou à un besoin utilisateur.
- Le développement dont la mission est de répondre dans un premier temps au plus vite aux demandes du métier, en livrant souvent et le plus rapidement possible l'implémentation de fonctionnalités de qualité.

Et enfin, les opérations, qui mettent en place le développement en production pour répondre au besoin métier initial. Le problème est souvent que chacune de ces trois phases, composées d'équipes complètement différentes, possède un langage qui lui est propre.

De ce fait, une frontière se crée naturellement entre chacun d'eux, empêchant la bonne communication. Les conséquences sont multiples :

- Un développement qui ne répond pas au besoin du métier,
- Un développement qui se soucie peu de la production,
- Des retours des opérations rarement complets pour le développement,
- Une vraie difficulté à créer des cycles de développement fluides.

Pour pallier en partie à ces problèmes, les méthodes agiles sont apparues. Celles-ci permettent entre autres une meilleure communication entre le métier et le développement.

Pour la communication entre le développement et l'opérationnel, il y a DevOps ! Mais qu'est-ce que DevOps ?

DevOps est la contraction des mots **Development** et **Operations**. C'est avant tout un mouvement, une culture, un processus agile ayant pour objectif la cohésion du système d'information sur les besoins de l'entreprise.

Son but principal est d'améliorer la communication en brisant la frontière qui sépare les différents services et d'industrialiser les processus de livraison afin d'améliorer la qualité, réduire les coûts et augmenter la productivité (et implicitement de répondre plus vite aux besoins du métier).

La gamme de produits Visual Studio ALM propose beaucoup d'outils depuis 2005, notamment pour fluidifier la communication Métier > développeurs > testeurs.

De la souplesse côté développement : Visual Studio Online

Team Foundation Service laisse place à Visual Studio Online. Ceci permet d'éviter les nombreuses confusions qui subsistent avec Team Foundation Server dont il partageait jusqu'alors l'acronyme : TFS. Visual Studio Online, comme son prédécesseur est un ensemble de services proposant entre autres la gestion de code source GIT et TFS Version Control, la

planification et le suivi de projets, l'intégration et le déploiement continu, ainsi que des

outils de test et de collaboration d'équipe. Visual Studio Online est hébergé dans le Cloud, et distribué en mode Software as a Service : aucune installation, peu de configuration, des mises à jour automatiques et une haute disponibilité. C'est avant tout le cœur de l'ALM version Microsoft, et logiquement l'un des points de départ de l'offre DevOps de l'éditeur.

Une meilleure communication entre les équipes : Team Room et Portfolio Management.

En 2013, apparaît dans le portail Visual Studio Online le concept de « Team Room ». Il s'agit tout simplement de la Timeline du projet.

Les participants à la Team Room peuvent ainsi voir l'évolution du projet en temps réel et discuter de celui-ci.

Cette partie de Visual Studio Online, ou TFS, est entièrement personnalisable par l'administrateur ; il peut y gérer les droits des participants et personnaliser les informations remontées telles que les notifications de build, les modifications du code source, la mise à jour d'un work item, une demande de révision de code...

La Team Room a pour vocation de devenir l'un des points centraux de la communication au sujet d'un projet d'équipe en cours de développement.

Ces avantages sont nombreux : historisation des conversations, suivi temporel du projet, réactivité accrue et personnalisable selon le profil de l'utilisateur. Son but est d'être accessible par tous : métier, développement et production !

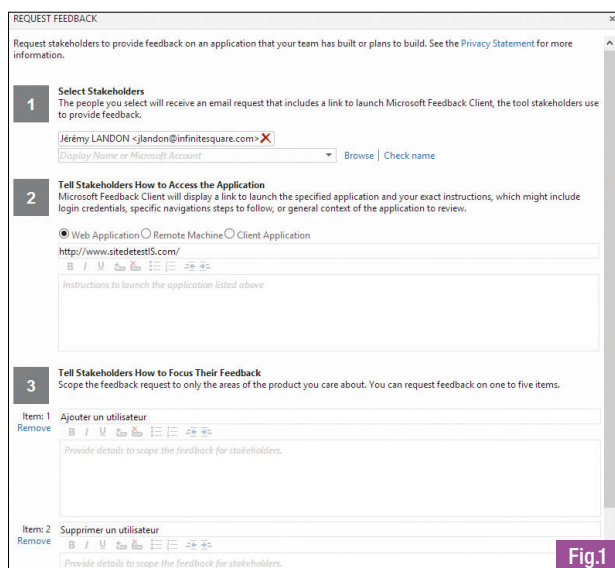


Fig.1

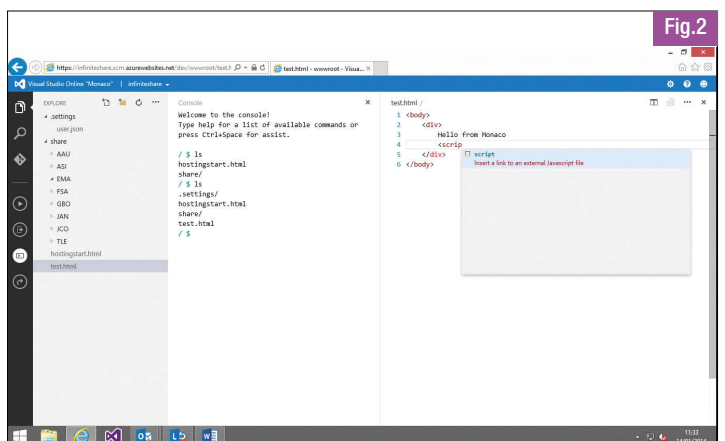


Fig.2

Jusqu'à 2013, la gestion de projet Agile dans TFS se limitait à la hiérarchisation Backlog Items et Tâches. En 2013, la notation de « Portfolio Management » permet de remonter plus haut dans l'organisation fonctionnelle du travail, en consolidant les Backlog Items dans des Features pour avoir une vision plus globale, tout en proposant un tableau de bord de gestion fonctionnelle.

Pour des feedbacks simples, efficaces et productifs : Feedback Manager

Les Feedbacks utilisateurs représentent une partie très importante dans la réalisation logicielle, ils assurent notamment la bonne réponse du développement aux attentes de l'utilisateur final.

Auparavant cette récolte d'éléments était réalisée soit via un simple fichier Excel, Word ou par échange d'Email, sans intégration automatique dans le projet. Avec Feedback Manager, Microsoft propose une solution permettant d'effectuer des demandes de Feedback auprès de n'importe quel intervenant non technique.

L'outil de Feedback récolte simplement l'avis de l'utilisateur, des screenshots, vidéos, divers fichiers, des commentaires sonores ainsi qu'une note allant de 1 à 5. Les feedbacks sont directement retranscrits en Work Item de type Feedback, et deviennent directement visibles par tous dans TFS ou Visual Studio Online **Fig.1**.

Vers un environnement de développement en ligne : Monaco

L'agilité du processus global passe peut-être également par une meilleure accessibilité de l'environnement de développement.

Visual Studio Online propose ainsi en version Preview un éditeur de code en

ligne, qui porte pour l'instant le nom de code Monaco.

Celui-ci est pensé pour simplifier l'édition de vues HTML/CSS, de fichiers XML et de scripts de code. Il a la particularité de permettre de faire de la modification « à chaud » sur un environnement.

L'éditeur Monaco propose également une invite de commande en ligne, permettant d'exécuter directement des instructions Git, PowerShell, Msbuild depuis son interface Web.

Par exemple, il peut s'avérer très utile pour une équipe de développement afin de faire des modifications en séance lors d'une démonstration auprès du métier, sans devoir déclencher le processus global de livraison. Un environnement de développement réduisant considérablement le risque d'erreur par rapport à un simple éditeur de texte, il peut être également utile pour les équipes de production lorsqu'il est requis d'effectuer des modifications urgentes en production, sur des fichiers de configuration ou sur de simples pages **Fig.2**.

Plus d'informations sur ce qui se passe en production : Application Insights

Le principal manque d'efficacité dans la collaboration Dev et Ops est lié à la collecte d'incidents de production. Mettre en place un mécanisme capable de tracer avec précision les incidents en environnement de production sans dégrader les performances de l'application n'est pas simple. La mise en place de logs applicatifs par les équipes de développement est le minimum requis. Faut-il encore pouvoir avoir accès simplement à ces logs et disposer d'un contexte de reproduction simple : que s'est-il vraiment passé ?

Depuis novembre 2013, Microsoft a annoncé une extension à Visual Studio

Online nommée Application Insights. Cet outil est actuellement disponible en version Preview, sur demande d'inscription depuis le portail Visual Studio Online.

Application Insights a pour objectif d'automatiser la mise à disposition auprès des développeurs des incidents de production. Par incident, il ne s'agit pas uniquement des logs d'erreurs, mais également de pouvoir disposer des dumps pour pouvoir débbugger, pas à pas, un incident de production en environnement de développement.

Pour permettre ceci, Application Insights s'appuie sur les logs d'IntelliTrace.

IntelliTrace, également appelé débbuggeur historique, est un outil présent depuis 2010 dans Visual Studio Ultimate. Cet outil permet de débbugger pas à pas un code s'étant déjà exécuté et propose donc la même expérience de débbug que sur du débbug classique : pas à pas en avant, mais également pour reculer dans le temps, valeur des différentes variables, liste des exceptions, call stack détaillé...

Application Insights propose également d'autres fonctionnalités intéressantes qui étaient jusque-là manquantes dans l'outillage Microsoft telles que les analyses de disponibilité ou les statistiques d'utilisation.

Pour les analyses de disponibilité, il est possible en quelques clics de configurer Application Insights pour « surveiller » une application Web. Application Insights va ainsi, à intervalle régulier, effectuer plusieurs requêtes sur l'applicatif et enregistrer le temps de réponse, proposant ainsi dans le temps des graphiques d'analyses de performances. Pour les statistiques d'utilisation, Application Insights propose un SDK qui permet d'intégrer des marqueurs dans un applicatif Web, une application Windows 8 ou une application Windows Phone. Ces marqueurs s'occupent d'analyser dans le détail l'utilisation de l'applicatif et de proposer des informations détaillées dans l'interface d'analyse centrale. Par exemple, pour une application mobile, les informations récoltées sont très nombreuses : Nombre d'utilisateurs, détail des pages vues, chemin de navigation des utilisateurs, fidélisation des utilisateurs (combien de temps avant qu'un utilisateur revienne sur l'application), type, langue et version du téléphone utilisé, mode de connexion (WiFi, GSM, offline... car les trackers fonctionnent en offline et consolide au retour du réseau)... **Fig.3**.

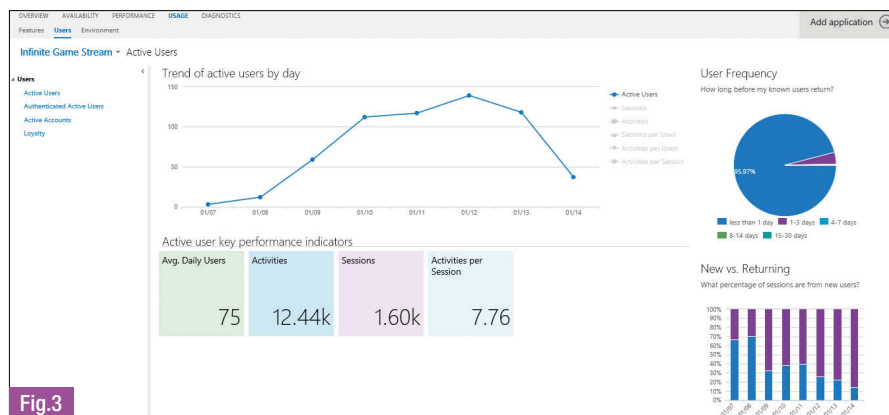


Fig.3

De l'agilité avec le Cloud pour vos environnements de développement et de test

DevOps, c'est aussi la possibilité de disposer plus simplement et plus rapidement d'environnements de développement ou de tests pour « reproduire » la production.

Aujourd'hui, disposer rapidement d'un environnement de dev ou de test n'est pas une mince affaire. Sauf quelques exceptions, les délais IT pour commander et mettre en place un environnement de développement sont souvent conséquents.

C'est là où le Cloud, et particulièrement Windows Azure, apporte de la souplesse intéressante au DevOps.

Besoin d'une machine virtuelle avec un environnement de développement pour essayer une mise à jour de Framework ? Les abonnés MSDN disposent maintenant d'avantages Azure leur permettant d'accéder à un forfait de crédit pour consommer n'importe quelle ressource Azure dont des machines virtuelles préinstallées avec Visual Studio 2013 ! Besoin d'un environnement de test pour déployer en intégration continue ? Là aussi, avec Azure, un administrateur système a la possibilité de provisionner à la demande, en quelques minutes, un nouvel environnement composé d'une ou plusieurs VM. Avec un peu de scripting, cet environnement peut même être configuré pour ne rester actif que dans les périodes de temps pour lesquelles l'équipe l'utilise, et être par exemple éteint automatiquement la nuit, le week-end et tout simplement quand le projet est terminé. Enfin, besoin de reproduire temporairement un environnement de préproduction composé de nombreuses machines pour reproduire un incident de production ? Azure et le Cloud permettent de disposer de ressources illimitées de manière éphémère, sans augmenter son investissement initial en termes de machines physiques.

L'IT Pro chargé de gérer un abonnement va donc faire le nécessaire pour simplifier la création et la reproduction d'environnements de développement, test, production, souvent en créant ses propres scripts. Les ressources informatiques pour faire tourner les applications devenant ainsi une « simple configuration ».

Pour le développeur, l'utilisation du Cloud est de plus en plus simplifiée, avec une gestion des ressources Azure directement depuis Visual Studio, sans jamais se connecter sur le portail Azure et avec des connaissances très réduites sur Azure. Par exemple, déployer un applicatif Web sur un serveur de développement IIS dans Azure peut s'effectuer en quelques clics depuis Visual Studio 2013 grâce au service de Website Azure.

Le Cloud pour les tests de charge

Reproduire un environnement proche de la production est donc simple avec Windows Azure. Par contre, pour reproduire un incident de production, il peut s'avérer parfois nécessaire de reproduire également la même charge utilisateur. Et là aussi, monter un environnement de test de charge est souvent une affaire coûteuse, en plus de l'environnement à tester, il faut créer un ensemble de machines chargé de générer la charge. Plus la charge à gérer va être élevée, plus il va y avoir besoin d'injecteurs et donc plus l'environnement va être coûteux à mettre en place. Visual Studio permet de faire des tests de charge depuis sa version 2005, et met à disposition des agents de charge pouvant être installés sur plusieurs machines. Avec la version 2013 et Visual Studio Online, ces agents sont déportés dans le Cloud, et il est ainsi possible d'effectuer des tests de charge sans avoir à installer d'infrastructure de charge !

Une simple configuration dans un projet de test de charge permet d'indiquer le nombre d'utilisateurs à simuler, Visual Studio et Azure se chargeant de configurer les injecteurs pour générer la charge nécessaire au test. Là aussi, chaque abonné MSDN dispose d'un crédit permettant de générer un certain nombre d'utilisateurs, qui peut bien sûr être augmenté pour les très gros tests de charge.

Plus de facilité pour l'automatisation des déploiements : Release Management

L'un des principaux buts du mouvement DevOps est l'industrialisation du processus de livraison pour permettre de gagner en productivité et en qualité. Auparavant, il n'existait pas de solution Microsoft pour répondre à ce besoin, désormais il y a Release Management ! Fig.4.

Release Management est issue du produit InRelease, solution développée par la société InCycle.

La solution a été rachetée par Microsoft courant 2013 afin d'en faire son produit de référence pour la mise en œuvre des scénarios de livraison.

Release Management fonctionne avec Team Foundation Server 2010, 2012 et 2013 et permet de façon simple et entièrement personnalisable de modéliser et réaliser les processus de livraison et d'effectuer des suivis d'approbation adaptés à chaque phase de livraison. La solution Release Management est décomposée en 3 modules distincts à savoir :


- Un agent de déploiement : Release Management Agent
- Un client WPF pour modéliser les processus de livraison via un système très proche de la réalisation de Workflow. Et une interface Web pour interagir avec la requête d'approbation mise en place : Release Management Client

Le point central de Release Management, contenant la base de données SQL et exposant différents services utilisés par d'autres composants : Release Management Server. Chacun des modules étant indépendant, la nouvelle solution de Microsoft se veut complètement flexible.

Conclusion

ALM, Agilité, Cloud... et maintenant DevOps. Les outils et processus de travail du métier, des développeurs et de la production sont plus que jamais en amélioration constante.

Derrière tout ceci, l'objectif reste toujours le même : simplifier notre collaboration, fluidifier nos échanges, gagner en efficacité pour pouvoir sortir des logiciels de meilleure qualité et plus rapidement pour le bonheur de nos utilisateurs !

 **Jeremy Landon**
et **Florent Santin**
Infinite Square



INFINITE SQUARE

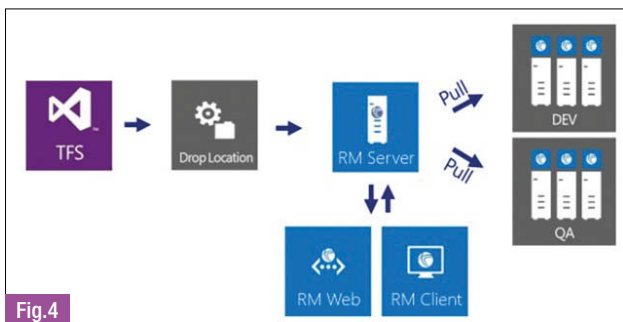


Fig.4

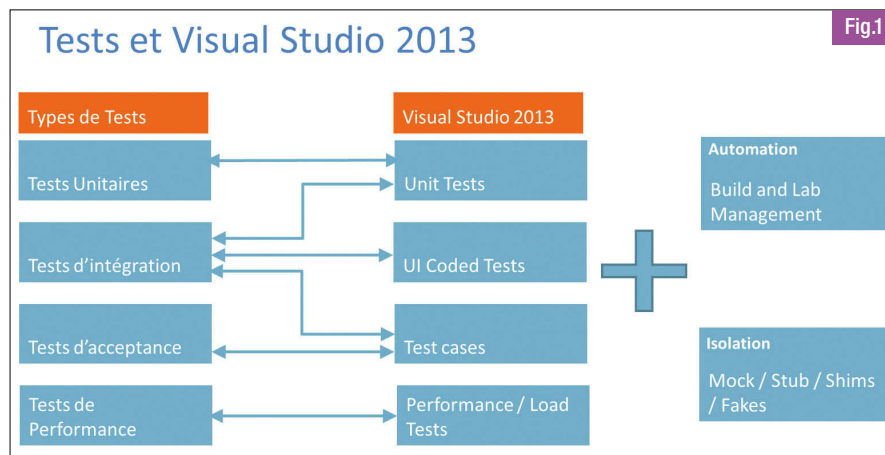
Agilité, tests, qualité

Dans cet article nous allons montrer la puissance et la facilité de Visual Studio 2013 pour le développement d'applications d'un haut niveau de qualité au travers d'un large panel d'outils méthodologiques, d'écriture de code, de productivité, de tests, et d'industrialisation des développements, formant un tout cohésif permettant la mise en œuvre d'une véritable démarche Dev-Tests.

Pour cela nous définirons quels sont les processus et outils nécessaires à une activité de Tests, puis nous verrons dans quel cadre cela nous permet de mettre en place une stratégie qualité, le tout en nous appuyant sur le paradigme agile et plus spécifiquement SCRUM. Enfin nous établirons les bases d'une approche Dev Tests en nous appuyant sur l'offre Visual Studio 2013.

Activité de Tests

Longtemps, l'activité de tests est restée le parent pauvre des projets informatiques, tant sur le plan des outils que des compétences des testeurs. Pourtant de nombreux exemples sont là pour nous rappeler l'importance de cette activité. De nos jours, beaucoup d'agilistes commencent à penser que l'écriture des tests est plus importante que l'écriture du code production, car du code non testé n'a pas de valeur et peut engendrer des situations particulièrement catastrophiques comme celle décrite ci-dessous avec le vol inaugural d'Ariane 5 en 1997, illustrant le bug le plus coûteux de l'histoire de l'informatique : la réutilisation de code du programme Ariane 4 non retesté a entraîné une trajectoire différente du fait d'un design différent du Booster. La conversion d'un float 64 bit en int 16 bits a causé une erreur de type *overflow exception* engendrant la destruction de la fusée, le coût du Bug : plus de 370 millions de dollars (1996). La morale de cette histoire est que l'activité de test doit faire partie intégrante du cycle de développement et de la stratégie qualité. Comme nous le verrons tout au long de l'article, Visual Studio 2013 d'une part et les approches agiles, et plus particulièrement SCRUM vont apporter l'ensemble des ingrédients tant au niveau des pratiques d'ingénieries que méthodologiques permettant l'implémentation et la livraison d'applications d'un haut niveau de qualité et de ce fait, obtenir la satisfaction du client, qui reste de loin le critère numéro 1 de réussite d'un projet. La phase de planification et d'écriture des tests est une activité de premier plan pour tout type de projet, nous allons avant tout définir les types de tests nécessaires à la mise en place d'une stratégie qualité digne de ce nom. On distingue généralement trois catégories de tests :



Les tests techniques (tests unitaires, tests d'intégration et tests d'interface automatisés) : Ces tests vont permettre de travailler sur la non-régression technique et fournir une métrique clé : la couverture de code. Les tests unitaires sont généralement mockés (stubs ou shims) et donc joués en isolation et testent les méthodes unitairement, contrairement aux tests d'intégration qui vont tester des méthodes de plus haut niveau et qui ne sont pas mockés afin de tester le système complet. Enfin les tests d'interface automatisés (CodedUI tests) permettent de valider des scénarios en manipulant l'interface utilisateur de manière interactive avec la capacité de valider un résultat attendu, telle que la valeur dans un champ par exemple. L'ensemble de ces tests pourront être consommés dans le cadre d'un workflow BDT (Build Deploy Test) si l'on a investi dans un Lab Management. Nous reviendrons plus loin dans cet article sur ce point.

Les tests fonctionnels (tests manuels ou semi automatisés, tests d'intégration fonctionnels, tests d'interface automatisés et tests exploratoires) : ces tests ont pour objectif de valider la conformité métier des fonctionnalités développées et feront l'objet d'un plan de tests et relieront des cas de tests aux fonctionnalités testées. Pour des raisons de productivité, le plus souvent ces tests feront l'objet d'une automatisation partielle ou totale afin de garantir des campagnes de tests de non-régression et surtout augmenter la productivité des équipes de tests. Les tests d'intégration fonctionnels permettent d'appeler des fonctions métiers de haut niveau exposées par

la couche métier sans passer par l'interface utilisateur (mode FitNesse). Enfin, les tests exploratoires sont au cœur des pratiques agiles de tests et vont permettre aux testeurs de jouer des tests avec plus de créativité et également trouver plus rapidement des bugs ou dysfonctionnements. Cette activité est à mettre en balance avec l'écriture de scripts de tests.

Les tests de performance et tests de charge.

La performance reste un point clé de réussite de tout projet, les tests de performance côté client et côté serveurs doivent faire partie impérativement de la stratégie qualité et donc des plans de tests afin de détecter au plus tôt les potentiels goulots d'étranglement. Bien évidemment Visual Studio 2013 propose une gamme complète d'outils permettant de créer et exécuter l'intégralité des tests précités. Le diagramme ci-dessus fait le mapping de ces différents types de tests avec les technologies de la plateforme Visual Studio 2013 : Fig.1.

Visual Studio 2013 : L'outillage pour les Tests

L'offre Visual studio 2013 ne propose pas moins de 4 modules répondant à l'ensemble des besoins en termes d'écriture et d'exécution des Tests. Nous allons les passer en revue.

Microsoft Test Manager 2013 (MTM) est un client lourd pour l'écriture et l'exécution des cas de tests fonctionnels encore appelés tests d'acceptance. MTM va permettre également de conduire des sessions de tests exploratoires. Le point fort de ce module est la capacité à enregistrer les actions effectuées par le testeur pour générer automatiquement

un cas de test à partir de la session exploratoire, de façon à pouvoir les répéter **Fig.2**.

MTM va permettre également une automatisation partielle ou complète de l'exécution des tests via une fonctionnalité d'enregistrement des actions effectuées par le testeur **Fig.3**.

L'intérêt est que le testeur pourra alors sélectionner plusieurs actions et lancer contextuellement la commande Play pour voir ces actions jouées automatiquement, lui permettant d'arriver plus rapidement à l'étape de validation du Test **Fig.4**.

La grande nouveauté apportée par Visual Studio 2013 est le support Web pour la gestion et l'exécution des tests avec Web Access 2013 permettant, à l'instar de Test Manager, les actions suivantes :

- La création de plans de test.
- La gestion des suites de test.
- La création de cas de tests incluant le support des étapes partagées et l'utilisation des paramètres
- L'entrée et l'édition en masse des cas de test à l'aide d'une Grid View
- La copie de cas de tests à partir d'Excel vers un Grid View et vice versa.
- L'exécution des tests.
- L'édition de tests lors de l'exécution.

Tester des Applications Windows Store sur un périphérique distant (tablette, téléphone) Sur la partie test technique, Visual Studio 2013 permet l'écriture des tests unitaires avec la possibilité d'utiliser plusieurs Frameworks de tests qui sont pluggables dans l'IDE et fournit notamment l'ensemble des fonctionnalités de refactoring nécessaires à la mise en œuvre d'une approche TDD (Test Driven Development), pratique par laquelle le développeur va procéder à l'écriture des tests avant l'écriture du code d'implémentation.

Nous allons démontrer à l'aide d'un exemple extrêmement simple basé sur une classe de calcul mathématique cette approche TDD avec Visual Studio 2013.

Nous partons d'une solution contenant un projet de type Class Library Math et son projet de Test UnitTestMath. On notera qu'au début, aucun type n'est présent dans le projet Math, tout sera généré par les fonctions de refactoring de Visual Studio 2013 : **Fig.5**.

Le principe TDD consiste à décrire les fonctions avant leur implémentation via les tests, ce qui a pour effet dans un premier temps de ne plus compiler, comme montré ci-dessous, après avoir écrit le test de la fonction Addition de la classe Operation qui n'existe pas encore, nous allons ensuite générer le type Operation et la fonction Addition : **Fig.6**.

À l'aide des fonctions de refactoring, nous allons dans un premier temps générer le type Operation dans le projet Math comme montré dans la capture d'écran ci-dessus. Puis nous allons générer dans ce nouveau Type Operation, la méthode Addition. Les opérations de refactoring sont lancées à partir des SmartTag comme montré ci-dessous **Fig.7**.

On se retrouve à l'étape où la méthode est générée sous la forme d'un stub, et le test qui ne passe pas : **Fig.8**.

Il suffit d'écrire l'implémentation de la méthode Addition et relancer le test pour terminer le cycle TDD, le test passe alors en vert **Fig.9**.

Enfin Visual Studio va également permettre d'écrire des tests unitaires « fonctionnels » qui seront joués en tant que tests d'acceptance via Test Manager. Généralement, ces tests appellent des fonctions de haut niveau qui vont combiner l'appel de plusieurs fonctions au travers des différentes couches de l'application, on les appelle parfois des tests « sous cutanés », car ils réalisent des appels juste au-dessous de l'interface utilisateur. Ces tests pouvant s'apparenter à des tests fonctionnels, il est possible de les inclure dans des cas de test. Pour cela, ouvrir le cas de test concerné dans Visual Studio, activer l'onglet « Associated Automation » puis sélectionner le test d'intégration et valider. Ce test pourra être joué de manière automatique dans vos scénarios BDT et également lancé manuellement. L'avantage de cette technique est qu'elle va permettre de limiter l'écriture de « CodedUI Tests » ou tests d'interfaces automatisés, pouvant être plus complexes et consommateurs de temps en termes de maintenance, le mix des deux est

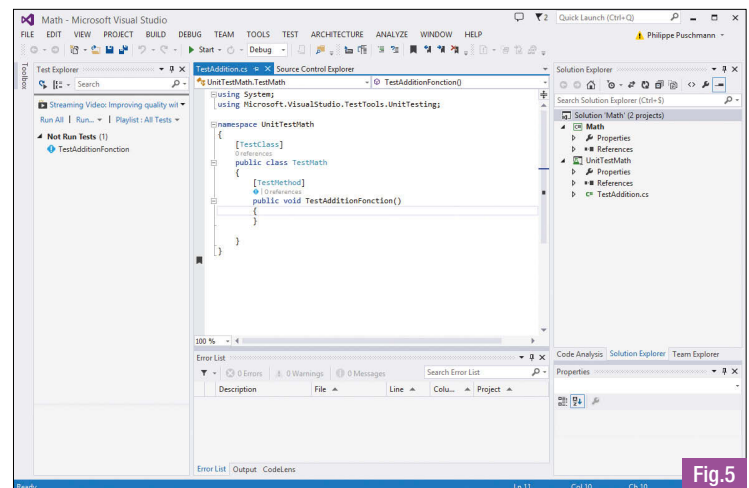


Fig.5

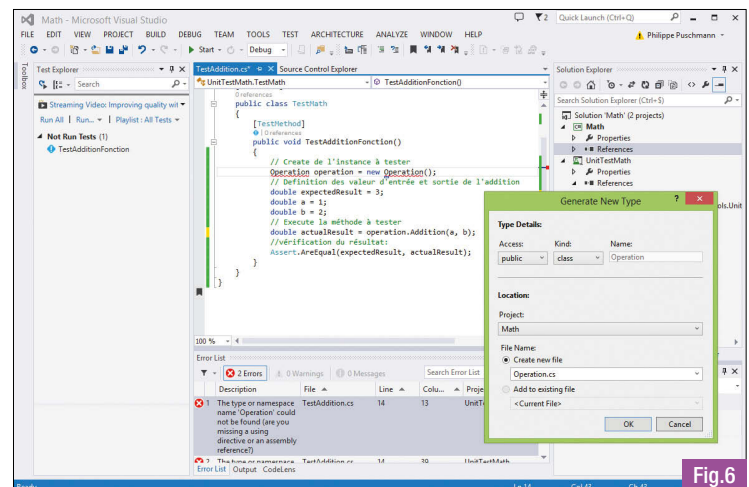


Fig.6

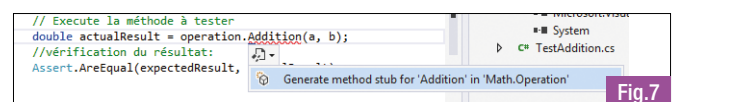


Fig.7

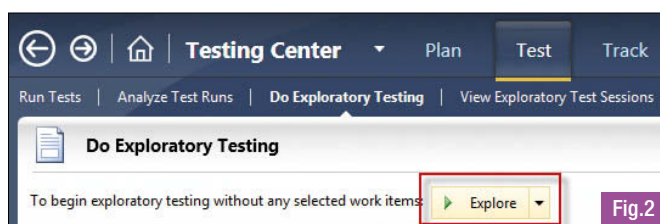


Fig.2

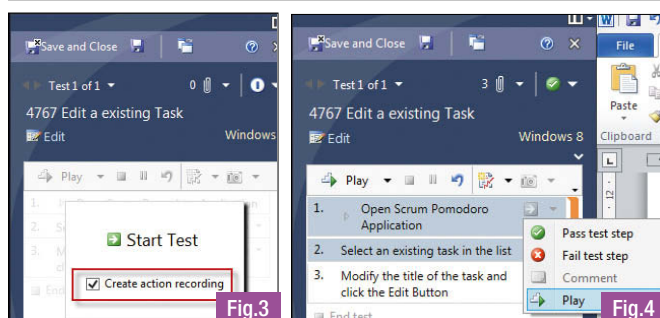


Fig.3

Fig.4

généralement le meilleur scénario **Fig.10**. Enfin les tests d'interfaces automatisés ou CodedUI Tests vont permettre de créer des tests interactifs avec l'interface utilisateur. L'intérêt de ces tests réside dans la simplicité de mise en œuvre grâce à un enregistreur d'actions, permettant à tout membre de l'équipe d'en créer, y compris des profils fonctionnels. Ils sont maintenables et modifiables, car cette technologie génère du code .Net. La maintenance en revanche va requérir des compétences de développement. Note : il est important de sensibiliser les développeurs au respect des normes de développement pour produire des interfaces testables notamment en respectant des conventions de nommage des contrôles et la fixation systématique des UI Automation Properties dans les balises XAML en WPF. Les gains apportés en retour par l'automatisation des tests rembourseront largement cet investissement.

QUALITÉ ET AGILITÉ

La qualité a de multiples facettes, et pourra être définie de manière différente en fonction des objectifs et du contexte d'un projet. De plus, la qualité est difficile à mesurer. L'agilité depuis près de 20 ans a largement

démontré au travers de nombreux rapports statistiques tels que ceux du Standish Group (Chaos Report) l'efficacité de la qualité vis-à-vis de la réussite des projets. Le cycle de vie dans le cadre d'un développement en mode Agile et plus spécifiquement SCRUM se présente comme suit : **Fig.11**.

L'équipe va garantir, en s'appuyant sur une approche incrémentale, itérative, et des cycles courts, une convergence avec le besoin métier et également un haut niveau de qualité, étant donné une activité soutenue de tests et la livraison d'un incrément potentiellement livrable en production à chaque fin d'itération. Ce processus, pouvant aller de 1 à 4 semaines maximum, implique une inspection des fonctionnalités produites afin de vérifier la conformité aux critères d'acceptation, ainsi que de la qualité desdits livrables au travers des tests et du contrôle de qualité de code. Si un des points n'est pas satisfait, le livrable est rejeté et replacé dans le Product Backlog pour finalisation au Sprint suivant. Ceci est décidé au cours de la session de Sprint Review, durant laquelle, l'équipe va présenter au Product Owner et aux parties prenantes du projet, les fonctionnalités développées lors du

Sprint. Le critère de passage pour rappel est la conformité avec la « Definition of Done » ainsi que la validation du Product Owner.

L'activité de tests est donc bien évidemment au centre d'une stratégie qualité, qui, en fait, englobe un plus large panel d'activités dont la plupart viennent du monde agile. Nous allons donc dans un premier temps définir quels sont les concepts clés à prendre en compte dans une stratégie qualité et nous mapperons le concept avec le ou les outils proposés par Visual Studio 2013.

La DOD : Definition of Done

L'état terminé qui se définit comme l'état où la fonctionnalité d'un incrément logiciel, devient potentiellement candidat à une mise en production. La Définition of Done (DOD) est une simple check-list auditable définie par l'équipe de développement et qui peut être influencée par les standards de l'organisation du projet (standard d'architecture, normes qualité, etc.). La DOD est un des éléments qui va permettre de valider comme terminé le développement d'un PBI (Product Backlog

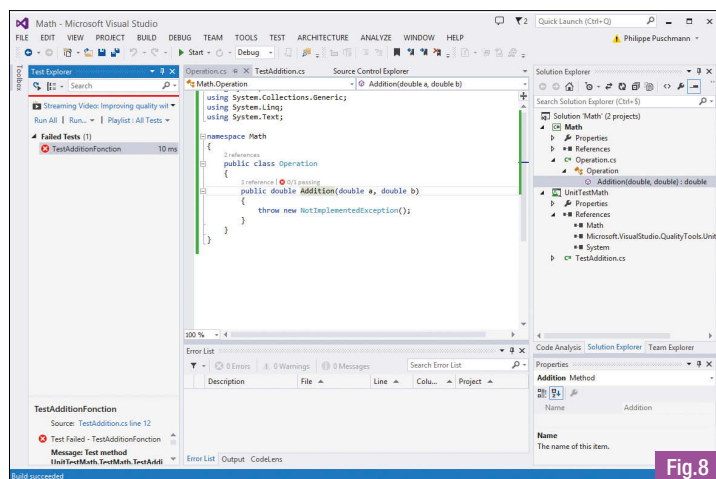


Fig.8

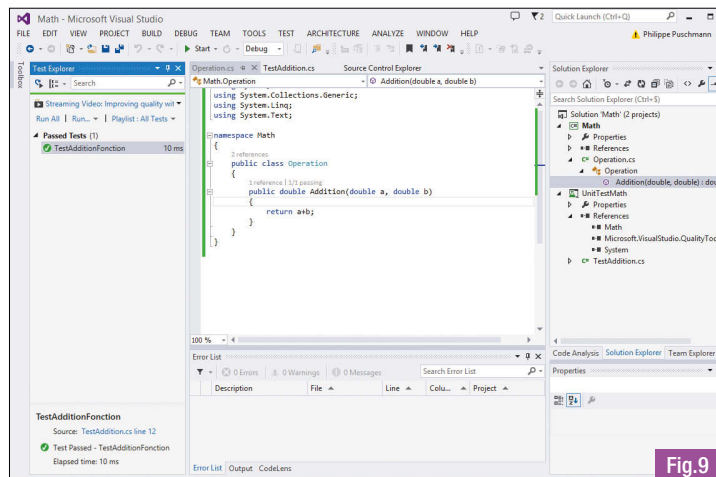


Fig.9

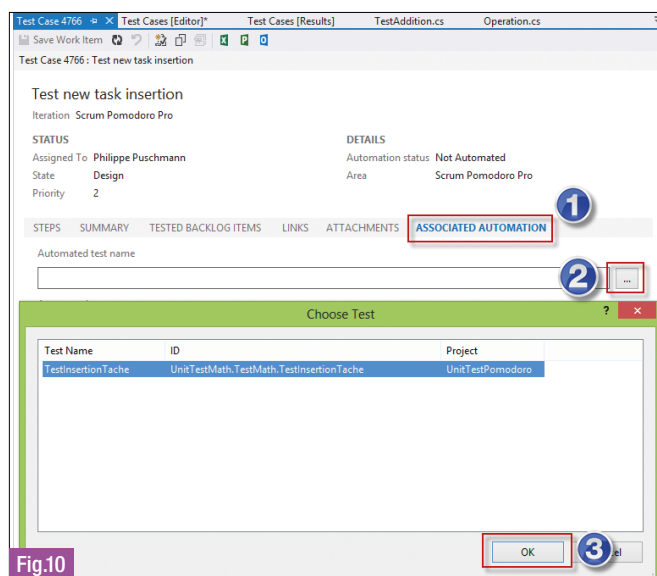


Fig.10

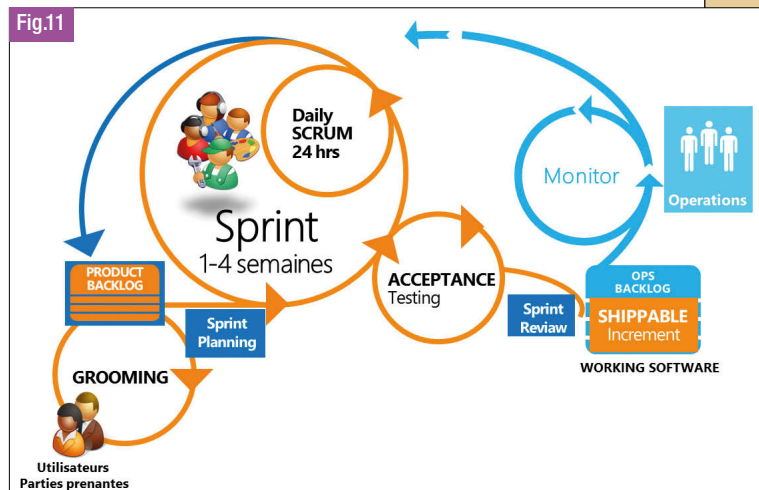


Fig.11

Item). Ci-dessous un exemple de DOD :

- ▀ Code complété, compilé et archivé dans le contrôle de code source
- ▀ Tous les tests passent (Tests unitaires, acceptance, non-régression)
- ▀ Les critères d'acceptance du Product Owner sont validés
- ▀ Nombre de Bugs critiques ouverts : 0
- ▀ La couverture de code est $\geq 70\%$
- ▀ Le nombre de warnings FXCop ≤ 20

Côté Visual Studio 2013, la DOD fera appel au Contrôle de Code Source TFS, les tests unitaires, le calcul de couverture de code, les builds automatisés, le module de gestion de projet Agile avec Web Access, le système de suivi des bugs, et Microsoft Test Manager.

La gestion de la dette technique

La dette technique est une métaphore inventée par Ward Cunningham dans les années 90. Il compare le concept existant de dette dans le domaine financier et l'applique au domaine du développement logiciel. Par définition, la dette technique est un ensemble d'accumulation d'imperfections techniques liées à un projet dans sa globalité, le plus souvent non visibles par l'utilisateur. À l'instar d'une dette financière, Ward Cunningham parle du point de non-retour, à partir duquel nous ne serons plus capables de rembourser la dette technique. Les causes principales sont généralement une mauvaise qualité du code, un manque de tests et/ou d'automatisation des tests, l'absence de validation à fréquence régulière en se basant sur une liste DOD par exemple. Un processus de contrôle qualité et des cycles courts agiles avec le respect d'une Definition of Done seront très efficaces pour gérer cette dette et/ou l'endiguer. D'autres pratiques d'ingénierie agile tel que le refactoring, le Pair Programming, la mise en œuvre d'analyse de code statique et dynamique ainsi que des tests de performance participeront également au contrôle de cette dette et permettront d'éviter le point de non-

retour. Visual Studio 2013 fournit l'ensemble des métriques permettant de piloter une dette technique et permet, au travers de la plateforme de reporting, de produire les rapports ad hoc.

Les points de contrôle et de Reporting

La mise en place de tableaux de bord ou rapports de statuts permettent de fournir les métriques qui seront vérifiées à chaque fin d'itération. Cette pratique est parfaitement en conformité avec Scrum dont les trois piliers sont la transparence, l'inspection et l'adaptation avec notamment l'opportunité à chaque fin de Sprint d'inspecter le niveau de qualité de l'incrément et potentiellement de l'adapter pour être conformes aux niveaux exigés par le projet avec par exemple, le niveau de Warning FXCop, le nombre de tests unitaires ainsi que la couverture de code associée, l'état de la Build tout au long du Sprint, et un rapport sur la couverture et les résultats des tests fonctionnels. Visual Studio 2013 fournit une plateforme de reporting et également un cube OLAP qui permettra de suivre les métriques du plan qualité et de les présenter dans un rapport de statut.

Le Refactoring : cette pratique d'amélioration du code et de son design consiste à changer la structure interne du code sans en affecter le comportement externe, en d'autres mots, améliorer le code sans casser les tests unitaires. Le refactoring ne doit être fait que pour minimiser la dette technique, ajouter de la valeur métier ou atteindre la « Definition of Done » de l'équipe. L'IDE Visual Studio fournit un outil de refactoring intégré et permet également d'intégrer sous forme de plug-in des outils tiers tels que Resharper par exemple

Le Pair Programming : Lors de cette pratique, deux développeurs vont travailler sur une tâche de développement et alterner à fréquence

constante. Pour cela, les développeurs pourront utiliser la méthode Pomodoro par exemple. Souvent controversée, cette méthode apporte pourtant de nombreux avantages tels qu'une meilleure qualité due à une inspection et revue du code continue. On notera également une augmentation des compétences et le partage des connaissances avec également une meilleure cohésion dans l'équipe. Généralement le code produit est plus rapidement conforme à la Definition of Done.

Test Driven Development (TDD) : cette pratique combine la technique de refactoring avec le fait d'écrire les tests unitaires avant le code d'implémentation. Le développeur écrit donc les spécifications détaillées de l'application et obtiendra de manière générale un meilleur design de son code. C'est une technique puissante pour éviter les défauts dans une application et garantir un haut niveau de couverture de code. Le couplage des tests unitaires et du refactoring font que Visual Studio 2013 est un outil parfaitement adapté à une approche TDD.

Analyse de code statique : cette fonction va relever la présence de défauts dans le code et lever des warnings ou des erreurs lors de la compilation. Le principe est de confronter le code écrit aux 200 règles et bonnes pratiques édictées par Microsoft couvrant les domaines suivants : Design, globalisation, interopérabilité, maintenabilité, conventions de nommage, performance, stabilité, sécurité, et usage de code.

L'analyse statique de code doit être utilisée dans le cadre du processus d'intégration continue, mais également dans le cas de développement externalisé ou encore dans le cadre d'audits. Avec le module d'analyse de code statique, il est possible dans Visual Studio ainsi que dans le processus de Build d'intégrer une politique qualité personnalisable basée sur des templates : **Fig.12**.

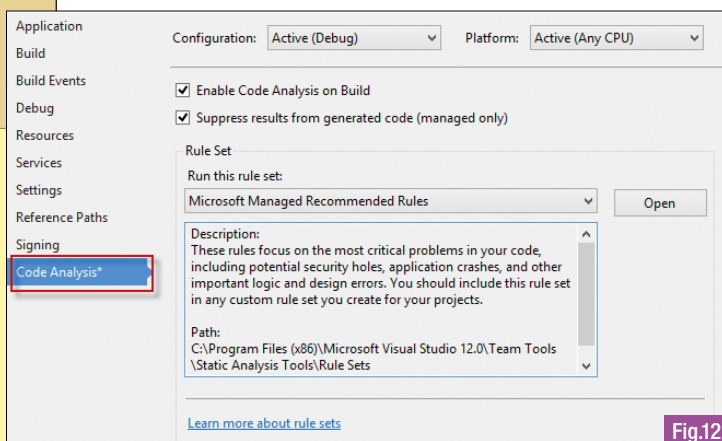


Fig.12

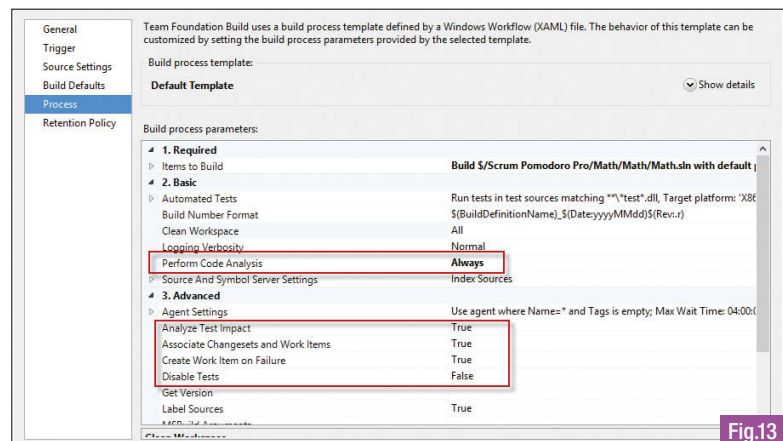


Fig.13

Sprint Reviews : le Sprint Review est un évènement qui clôture un Sprint qui a pour vocation l'inspection des fonctionnalités délivrées afin de valider si l'incrément est accepté ou replacé dans le Product Backlog. Cet évènement fait partie de la stratégie qualité, car c'est à ce moment que le Product Owner inspecte en présence des parties prenantes les fonctionnalités livrées pour validation ou non.

Intégration continue (CI) : cette pratique implique que les développements intégrés dans le contrôle de code source doivent faire l'objet à fréquence fixe, ou lors d'évènements tels que le check-in d'un développeur, d'une compilation globale avec un jeu complet des tests unitaires ainsi que d'un processus d'analyse de code statique. Dans Visual Studio 2013, le support de l'intégration continue repose sur le service de Build, permettant de créer des builds des solutions composant un projet avec le jeu des tests unitaires et le contrôle de l'analyse statique du code. Une fonctionnalité particulièrement intéressante avec Visual Studio est le Gated Check in. Sur le principe, dès qu'un développeur va soumettre son code pour archivage, l'archivage ne sera accepté qu'à la condition que le code fusionne et compile avec succès avec le code côté serveur. Techniquement, TFS crée un Shelveset côté serveur, fusionne le code proposé par le développeur puis le compile. En cas d'échec, l'archivage est rejeté, dans le cas contraire, le code est intégré au contrôle de code source. Ci-dessous la configuration d'un build avec l'analyse de code et le jeu des tests unitaires : **Fig.13**.

Déploiement continu (CD) : ce concept va permettre, à l'issue d'un build, de déployer automatiquement l'application afin que les

testeurs aient en permanence la dernière version à tester. Le déploiement continu constitue une automatisation à très haute valeur ajoutée, qui, de plus, va engendrer des économies importantes de temps à l'équipe, car il s'agit très souvent d'une tâche fastidieuse et répétitive. Visual Studio 2013 va proposer cette fonctionnalité de déploiement continu en se basant sur la brique Lab Management via les workflows BDT (Build Deploy Test). Dans ce cas la solution est compilée, testée unitairement, puis déployée sur un environnement cible, puis des tests automatisés ou manuels pourront être joués sur ces environnements. Une autre possibilité consistera à modifier les templates de builds pour invoquer des déploiements tels que Click Once et Web Deploy à la fin d'un build, si les tests ont été réalisés avec succès. Un template de Build est décrit via la création d'un workflow d'activités avec la technologie Workflow Foundation de Microsoft.

Automatisation des Tests : cette pratique va consister simplement à intégrer le jeu des tests de non-régression au processus d'intégration continue. L'automatisation pourra également inclure des tests automatisés d'interfaces utilisateurs. Visual Studio avec la plateforme de build et la mise en œuvre de scénario BDT avec Lab Management vient outiller ce scénario de bout en bout.

Gestion des Bugs : quand bien même, en agile, la devise pour les bugs est : « Fixez-les, ne les gérez pas ! », un système de gestion d'anomalies est au centre d'une stratégie qualité, ceci particulièrement lorsque l'application est utilisée par des équipes autres que l'équipe de développement, ou encore lorsque certains bugs ne pourront pas être

gérés dans le Sprint en cours. À ce moment-là, ils pourront être tracés dans le système de suivi d'anomalies. Un des points intéressants du suivi des bugs, est aussi le suivi de la volumétrie des bugs et également celui des bugs critiques qui potentiellement peuvent être bloquants pour la validation d'un Sprint, avec, par exemple, l'inclusion dans la Definition of Done, de la clause 0 bug critique. Avec Visual Studio 2013, deux outils permettront la création et la gestion des bugs : pour les testeurs, il s'agit de Test Runner, qui va permettre la création de bugs riches grâce au collecteur intégré, pour toutes les autres parties prenantes, la plateforme Web Access va permettre la création et le suivi des bugs au travers du tableau de bord du portail Web Access et aussi des requêtes de work items. Côté développeur, Visual Studio 2013 apporte de nombreuses fonctionnalités qui vont permettre un meilleur contrôle de la qualité du code avec :

Code Lens : Cet outil confère un affichage de type tableau de bord dans l'IDE au niveau fonction, mais aussi au niveau du type permettant au développeur de disposer d'informations contextuelles, de manière configurable via le menu options, avec :

- Statut des Tests unitaires
- Testé par
- Références
- Auteur
- Changements
- Work items/bugs
- Code Reviews

Le développeur peut non seulement consulter les informations, mais aussi accéder à ces informations en cliquant sur les hyperliens.

Code Map : cette fonctionnalité permet de retranscrire des portions de code en format visuel et analyser efficacement les dépendances du code. Il permet d'appréhender la complexité par le visuel et de mieux gérer la structure du code pour mener à bien une tâche de refactoring par exemple **Fig.14**.

La richesse fonctionnelle de la plateforme Visual Studio 2013 va permettre d'intégrer ses équipes dans une réelle approche Dev-Tests au travers d'outils performants centrés sur la qualité, l'agilité, l'automatisation et la productivité. L'outillage des méthodes et pratiques explicitées dans cet article avec Visual Studio 2013 permettra d'optimiser la probabilité de réussite de tout projet informatique.

Philippe Puschmann

Architecte Solution chez Avanade France.

Membre de l'équipe CTO, il est en charge du pôle ALM

Avanade France, et assure également la fonction de

Formateur/Coach SCRUM. MVP Visual Studio 2013, certifié

Professional SCRUM Trainer (SCRUM.org) et certifié PMP

(Project Manager Professional)

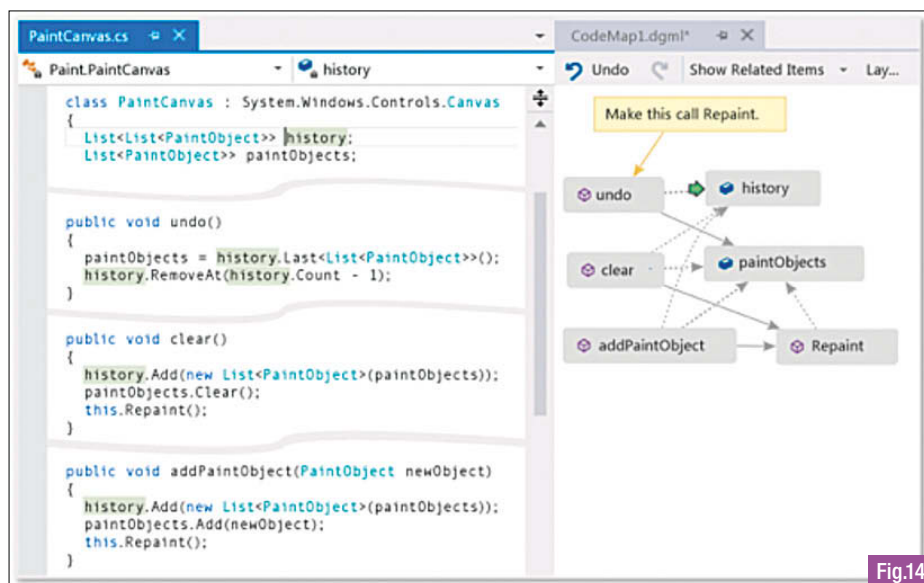


Fig.14

Ressources

par Pierre-Henri Gache (Cellenza)

Centre des Usages ALM

Sur une demi-journée ou une journée complète, vous profitez d'un environnement utilisateur innovant, équipé de plusieurs postes de travail et d'un instructeur pour vous accompagner dans l'exécution des scénarios de votre choix. <http://aka.ms/cdualmfrance>

Sur Internet

Microsoft Virtual Academy

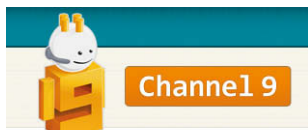
Lorsque l'on souhaite apprendre par soi-même, il est important de trouver du contenu de qualité bien organisé. C'est ce que propose ce site. On peut ainsi créer son parcours de formation personnalisé. Toutes les formations sont découpées en chapitre. Chaque chapitre comporte un support de formation qui est par exemple une vidéo et une auto-évaluation. A tous ces éléments sont attribués des points qui sont comptabilisés lorsqu'un chapitre est terminé. Ce décompte permet de suivre son avancement et de participer à un classement entre tous les utilisateurs.

Lien : <http://www.microsoftvirtualacademy.com/product-training/product-visual-studio-fr>

Channel 9

Ce site est sans conteste celui qui regroupe le plus de vidéos sur les technologies Microsoft. On peut donc y trouver un nombre incroyable de vidéos sur Visual Studio en plus de tout le reste. Les nombreux événements Microsoft qui ponctuent l'année sont aussi en général disponibles via ce site, c'est donc un très bon moyen de connaître les dernières nouveautés. A consulter sans modération.

Lien : <http://channel9.msdn.com>



Livres blancs

Assurer la qualité logicielle de bout en bout :
<http://goo.gl/atizHM>

Développer pour le Cloud :
<http://goo.gl/FAUaDm>

Réduire les coûts avec l'ALM :
<http://goo.gl/ishVFC>

Développer des applications NUI :
<http://goo.gl/fDORj2>

Atelier VM Brian Keller

Pour prendre en main Visual Studio, on a souvent besoin d'un environnement existant dans lequel on peut retrouver du contenu. C'est ce que propose Brian Keller avec ces machines virtuelles. Elles contiennent un environnement composé de Visual Studio et de Team Foundation Server prêt à l'emploi. Il propose également un ensemble d'ateliers qui couvre une large majorité des fonctionnalités disponibles. C'est un bon moyen pour se former et pour progresser rapidement.

Lien : <http://blogs.msdn.com/b/brianke/>

Developer code samples

Besoin d'aide pour utiliser une nouveauté du framework ou encore débiter un nouveau projet ?

Quoi de mieux qu'un bon exemple pour découvrir comment utiliser la fonctionnalité attendue ? C'est ce que propose ce site : un ensemble d'exemples sur des points précis qui vous permettront de progresser et d'appréhender facilement les différents aspects du langage.

Lien : <http://code.msdn.microsoft.com/>

Outils

Resharper

Resharper est certainement l'outil permettant d'améliorer la productivité le plus connu et ce à juste titre. Bien que chaque version de Visual Studio progresse dans ce domaine, il reste un incontournable.

Lien : <http://www.jetbrains.com/resharper/>

Moq

Lors de l'écriture de vos tests unitaires vous aurez besoin d'écrire des Mock. Mais si vous n'avez que la version Pro de Visual Studio vous n'aurez pas accès à Fakes. Pour le remplacer, Moq est une des solutions à envisager. Facile à

prendre en main, il saura se rendre indispensable.

Lien : <https://github.com/Moq>

BeyondCompare

Besoin d'effectuer de la comparaison de fichiers ou encore un merge entre deux branches, BeyondCompare est à mon avis le meilleur outil qui existe. Entièrement paramétrable, rapide et efficace, il sera votre allié lors des longues séances de merge.

Lien : <http://www.scootersoftware.com/>

dotPeek .Net

Cet outil permet de décompiler gra-

tuitement des assemblages .NET.

Vous pouvez ainsi vérifier que votre binaire contient bien le code désiré ou encore connaître la version de chaque référence nécessaire à son bon fonctionnement.

Lien : <http://www.jetbrains.com/decompiler/>

Notepad++

Si vous souhaitez éditer un fichier projet (csproj) à la main ou encore modifier le fichier de configuration de votre application, Notepad++ est un éditeur de texte simple et efficace qui remplira son rôle à merveille. Et il est gratuit.

Lien : <http://notepad-plus-plus.org/fr/>

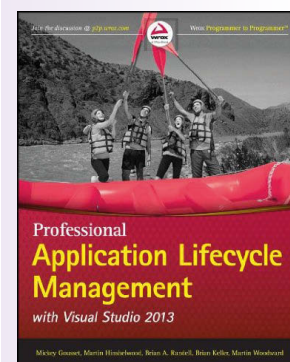
Livres

Visual Studio 2013 Concevoir et développer des projets Web, les gérer avec TFS 2013



Ce premier livre sur le développement avec Visual Studio 2013 et Team Foundation Server 2013 s'adresse à un large public de développeurs. Il a pour objectif de permettre au lecteur d'acquérir de solides bases pour démarrer dans les meilleures conditions un projet web avec les outils Microsoft, en s'appuyant sur un processus Agile. Des connaissances sur la programmation objet et le langage C# sous Visual Studio sont un prérequis minimum pour tirer le meilleur parti de ce livre.

Professional Application Lifecycle Management with Visual Studio 2013



Ce livre est également une référence lorsque l'on souhaite réaliser un projet avec Visual Studio. Par contre comme son titre l'indique, il se focalise plus sur la partie gestion du cycle de vie de l'application. C'est également un très bon choix pour celui qui souhaite passer les certifications ALM Microsoft.

DESCRIPTIF DE L'ÉTUDE DE CAS

Vous travaillez dans une entreprise de vente de produits agro-alimentaire. A l'approche de la fin d'année (période des clôtures comptables), le dirigeant et le responsable marketing vous convoquent pour assister à un de leurs débats. Le Directeur lance : « Monsieur, vous êtes le Data Scientist du département IT ? ». Vous répondez : « oui ». Il rajoute « Pouvez-vous nous aider à diagnostiquer et à comprendre ce qui se passe ? Sans en être sûr il paraîtrait que notre activité ainsi que notre image vont très mal depuis quelques mois. Bien sûr, formulé comme cela, vous serez embarrassé de répondre car la question est trop grossière. Vous avez donc besoin de décortiquer tout cela, afin de mieux comprendre toutes les fonctions que cela nécessite (voir tableau des fonctions).

Tâches du Data Scientist			
Tâche	Equipe/Livrable	Descriptif de la tâche	Fonction/Compétence
Interview	Veille & Stratégie	Récupération des données de marché (issues d'opérations veille technologique, veille concurrentielle, veille commerciale, veille environnementale) sous forme de fichiers résumés ou d'écrits papiers	
	Marketing & Vente	Récupération des données de panel (sur les produits de l'entreprise), d'enquêtes marketing ou de sondages	
	Comptabilité & Finance	Récupération des documents de bilan / compte de résultats et journaux comptables (éventuellement du PCG)	
	Opérations et Production	Récupération des bases de production et des applicatifs permettant leur compréhension	
	Ressources Humaines	Récupération du registre K-bis, du plan de formation des salariés, bilan social	
	Direction Générale	Entretiens oraux permettant de connaître la stratégie globale de l'entreprise ainsi que les objectifs à atteindre. L'entretien devra aussi permettre de comprendre la synergie des différentes équipes composant l'entreprise.	
Etude	Définition/Sélection des Business Process	Outil manuel (ou automatique via des logiciels de BPM) permettant de fournir une cartographie des flux internes et externes à l'entreprise (étude de la concurrence, étude de la e-reputation, processus de facturation, politique de détermination des prix)	Business Analyst
	Entreprise Bus Matrix	Outil manuel très pratique, permettant de mieux appréhender le fonctionnement de l'entreprise dans sa globalité (voir l'approche Kimballienne de la gestion de projet décisionnel): décomposition des Business Process sous forme de sujets (ou DataMarts) croisés avec axes d'analyses tels que le temps.	
	Architecture fonctionnelle	Ensemble des fonctionnalités que présentera le système d'information qui sera construit pour répondre aux interrogations du Directeur Général : - Indicateurs présentant l'état de réputation de l'entreprise sur Internet (réseaux sociaux, forums etc.) - Indicateurs sociaux présentant l'état de contentement des salariés (ces indicateurs sont importants pour faire leur croisement avec les indicateurs consolidés à partir des réseaux sociaux, car en général, le mécontentement intérieur peut expliquer entre autres une mauvaise réputation à l'extérieur). - Indicateurs comptables (comptabilité fournisseur, comptabilité client) pouvant expliquer une partie des mauvais chiffres en faisant ressortir les écarts entre débits et crédits, - Indicateurs de vente/prospection (prix des produits, distribution géographique des clients, statistiques sur la satisfaction client, taux de réclamations, taux de pénétration, taux d'appétence) - Indicateurs de production (nombre de transactions clients, nombre de transactions fournisseur)	
Conception & Développement	DataMarts	Conception orienté sujet, incrémentale (compatible avec les méthodes agiles tels que SCRUM). Bonne maîtrise des différents sujets concernant le business et application des designs patterns Kimballienne correspondantes (ex: gestion des indicateurs de production). Définition des granularités des faits, des dimensions, des mesures. Modélisation en Etoile/Flocon/Constellation.	Modélisateur Multidimensionnelle
	Flux ETL	Conception des flux d'alimentation à partir de la cartographie des flux de l'entreprise (connexions aux sources de fichiers plats provenant de la veille, connexions aux sources de l'ERP du service comptable, connexions aux données opérationnelles), Nettoyage, Intégration, Enrichissement, Qualification des données. Référentiel unique des données (Master Data Management) permettant de reconcilier les données de veille avec les données de panels par exemple.	Intégrateur de Données et Data Steward
	Analyses & Navigation OLAP	Conception des cubes de données, calcul des indicateurs, définition des ratios, calculs MDX , échantillonnage statistique, training des données à des fins d'analyses prédictives. C'est la phase la plus complexe et en même temps la plus passionnante car c'est là où les croisements vont se faire pour expliquer des choses au dirigeant. Vous pouvez par exemple affirmer avec précision que la baisse des ventes ne concerne que certains produits et certains segments de marché et que, c'est dû à la baisse des marges constatée sur le marché, essentiellement causée par les nouvelles ventes discount proposées par de nouveaux concurrent. Vous pouvez dire aussi que cela s'est produit durant l'été. Vous pouvez aussi à partir des éléments statistiques et de data mining, prédire une reprise approximative au début de l'année prochaine. Enfin vous pouvez expliquer la mauvaise réputation extérieure par la très mauvaise satisfaction intérieure après application des réseaux de neurones par exemple.	Mathématiques/ Statistique Datamining Business Analyst
	Reporting	Choix d'éléments visuels adaptés aux différentes équipes à des fins d'analyse. Par exemple, le dirigeant n'aura certainement pas besoin du listing tableau. Il se contentera des KPI lui donnant le comportement des différents indicateurs avec un code couleur adapté (rouge=catastrophe, orange=attention, vert=bien) par rapport aux objectifs fixés par la direction. En lui présentant un dashboard avec les indicateurs de réputation et de la politique des prix en orange et les indicateurs internes de résultat en vert, il s'inquiétera bien moins. Il saura juste qu'il a des alertes levées sur la réputation et la politique des prix mais que la progression de l'entreprise est bien sur les rails. S'il veut plus de détails, des rapports opérationnels sous forme de listing pourront lui être fournis.	Business Analyst Reporter/DataViz

Intelligence a sans aucun doute les mêmes pré-occupations : manipuler, croiser, analyser, présenter, forer, échantillonner, prédire, intégrer, gouverner les données dans le seul but de répondre aux interrogations de son entreprise (il aura donc à manipuler des concepts tels que la Data Integration, la Data Cleansing, la Data Consistency, le Master Data Management, la Data Visualisation, la Data Training, le Drilling qui sont autant de termes associés au domaine de la Business Intelligence). On voit donc nettement que le « consultant BI » autant que le « Data Scientist » ont les mêmes qualifications.

Les métiers de Business Intelligence (ou Data Scientist)

Aujourd'hui, on trouve sur le marché de la prestation informatique, des compétences BI très scindées. Cependant, il arrive parfois de trouver toutes ces compétences chez une seule personne :

Spécialiste de la modélisation de données

► Le spécialiste de la modélisation d'entrepôt de données qui se concentre essentiellement sur la modélisation dimensionnelle avec les problèmes métiers qu'elle traite. D'ailleurs dans le descriptif des types de profils recherchés, on retrouve, entre autres, des termes tels que : maîtrise de la modélisation en étoile, en flocon, en constellation ou encore approche Kimballienne exigée (en référence à Ralph Kimball considéré comme le père de la modélisation multidimensionnelle).

Spécialiste de l'intégration de données

► Le spécialiste de l'intégration de données qui aborde les problématiques de connexions aux différentes sources d'information externes ou internes à l'entreprise, de leur nettoyage, de leur mise en conformité, de la gestion de leur qualité. Dans ce type de profil, on demande la maîtrise de la cartographie des flux de l'entreprise, la maîtrise d'un outil ETL, de la bonne connaissance des outils de gestion des

données de référence (outil de Master Data Management) ainsi qu'une bonne connaissance des outils de qualité de données.

Spécialiste de la navigation des données

► Le spécialiste de la navigation de données qui s'embarque dans tous les sujets de gestion des chemins de navigation et d'analyse accessibles par les business analystes. Dans ce type de profil, on demande la maîtrise d'un moteur d'analyse (Certains parlent de moteur OLAP). Mais le moteur OLAP n'est qu'un hébergeur de cubes, contrairement à un moteur d'analyse qui héberge non seulement les cubes mais aussi les calculs de navigation, statistiques voire prédictifs, du langage MDX (langage standard de requêtage multidimensionnel), de statistique ou de datamining.

Spécialiste du Reporting

► Le spécialiste du reporting qui a en charge la restitution des données. Il doit fournir la charte visuelle des données. Cette personne est en principe un garant de l'adoption de toutes les techniques de visualisation de données (Data-Viz). Suivant le type de public à qui il doit s'adresser, sa présentation des données ne sera pas la même. Par exemple, les managers généralement très pressés pour prendre leur décision ont besoin de KPI (Indicateur Clé de Performance) alors qu'un opérationnel se contentera d'un listing de type tableau présentant les opérations traitées la veille par les membres de son équipe.

Au final, tous ces profils font partie intégrante du métier du spécialiste Business Intelligence avec une proximité forte avec le Business. En tout état de cause, le spécialiste BI doit beaucoup se rapprocher du métier de l'entreprise et bien le maîtriser, ce qui rend généralement cette fonction technico-fonctionnelle plus dure encore (les anglo-saxons mettent en avant le côté Business Analyst du profil).

Cas pratique

Pour aller jusqu'au bout des choses, essayons d'illustrer les différentes tâches de l'expert BI

(ou Data Scientist si vous voulez !) par un Business Case classique d'entreprise.

Par exemple, vous travaillez dans une entreprise de vente de produits agro-alimentaires. A l'approche de la fin d'année (période des clôtures comptables), le directeur général et le responsable marketing vous convoquent pour assister à un de leurs débats. Le dirigeant lance : « Monsieur, vous êtes le Data Scientist du département IT ? ». Vous répondez : « oui ». Il rajoute « Pouvez-vous nous aider à diagnostiquer et à comprendre ce qui se passe ? Sans en être sûr, il paraîtrait que notre activité ainsi que notre image vont très mal depuis plusieurs mois ». Bien sûr, formulé comme cela, vous serez embarrassé de répondre car la question est trop grossière. Vous avez donc besoin de décortiquer tout cela, afin de mieux comprendre toutes les fonctions que cela nécessite (voir tableau des fonctions et figure1).

Conclusion

Dans cet article, sans entrer dans les détails, nous faisons un zoom sur les fonctions qu'exerce un data scientist en entreprise. Certains peuvent voir cette discipline encore plus large que celui du spécialiste business intelligence en mettant en avant la face « Big Data ». En effet, cela peut être valable si nous partons du principe que, nous analysons en temps réel comme le fait Google, Twitter ou Facebook, des données massives venant du web ou d'autres terminaux. Si nous faisons la même chose en entreprise, cela nous ferait changer de paradigme. Par exemple, l'analyse statistique ne nécessiterait plus un échantillonnage de la population, mais une prise en considération de la totalité des données du web à des fins d'analyse. En général, la cellule veille et stratégie se charge de capturer les données nécessaires à l'entreprise. C'est donc pourquoi il s'agit dans cet article du rôle du data Scientist en entreprise.



🔴 Mohamed Taslimanka SYLLA
Architecte Business Intelligence - SFEIR

**À LIRE
DANS LE
PROCHAIN
NUMÉRO**

n° 172 en kiosque
le 28 février 2014

Java 8

Mi-mars, Java 8 sera disponible, quoiqu'il arrive.

Qu'apporte cette version ?

Java est-il toujours un langage moderne et d'avenir ?

Google Glass

Quelles applications développer ?

Un nouveau marché pour les développeurs ?

Développement embarqué

Le défi du développeur

Les objets connectés

NIDays

WORLDWIDE GRAPHICAL SYSTEM DESIGN
CONFERENCES

Le rendez-vous annuel des ingénieurs,
scientifiques et enseignants

**Au CNIT Paris La Défense,
le 11 février 2014**



Il est temps, plus que jamais, d'innover pour relever les défis du monde moderne !

Vous préférez construire l'avenir plutôt que le subir ? Alors ne manquez surtout pas l'édition 2014 de NIDays et son lot d'innovations en matière de systèmes de mesure, de contrôle, d'automatisation et embarqués :

- ▶ 70 stands de démonstrations
- ▶ 2 conférences plénières, avec le célèbre prospectiviste Joël de Rosnay en invité d'honneur
- ▶ 50 conférences techniques et métiers (dont 15 présentations d'utilisateurs)
- ▶ 12 sessions de TP pour s'initier aux matériels et aux logiciels
- ▶ 2 compétitions de robotique pour l'enseignement

Programme et inscription gratuite sur www.nidays.fr ▶ 01 57 66 24 24



GoPro hacking

Les GoPro sont réputées pour être de superbes caméras et appareils photos robustes permettant une utilisation dans toutes les conditions. Et je confirme que c'est bien le cas. Leur intérêt est d'avoir une prise de vue grand angle, de pouvoir faire des vidéos très hautes résolution jusqu'à 4K, et d'offrir aussi des prises de vues à intervalles réguliers. Les modèles depuis la version Hero 2+ et les nouvelles versions Hero 3+ ont la possibilité d'être connectées en Wifi. Et qui dit Wifi dit application, ce qui stimule forcément le besoin de hacker la bête.

L'intérêt n'est pas de reproduire ce qui existe déjà dans les applications fournies, que ce soit sur Windows Phone, iOS ou Android mais de pouvoir créer une nouvelle application pour une utilisation dédiée ou spécifique. Cet article est une introduction qui vous donne des pointeurs sur les éléments clés pour le hacking de la caméra. Rentrons donc dans le vif du sujet. La caméra possède 2 modes Wifi, un premier qui est dédié au pilotage par télécommande et un second qui permet à un appareil de se connecter à la caméra. La caméra crée son propre réseau Wifi infrastructure avec son propre SSID et mot de passe. Lors de la mise à jour des firmwares, il est possible de personnaliser son SSID et son mot de passe.

IP et TS

La caméra prendra systématiquement l'IP 10.5.5.9. Donc toute communication doit se faire avec cette IP. Cela permet notamment de détecter si une caméra est présente. Et cela impose donc d'avoir une et une seule caméra par SSID. Si vous voulez piloter plusieurs caméras, il faudra donc créer un SSID par caméra et changer de SSID pour se connecter à chaque caméra ou disposer de plusieurs cartes wifi.

La caméra est accessible à travers une interface web sur l'URL <http://10.5.5.9:8080>. Cela permet de browser la carte SD et donc de télécharger les fichiers présents, que ce soient les vidéos ou les photos. Et elle donne également accès à la preview de la caméra.

L'URL est <http://10.5.5.9:8080/live/amba.m3u8>. Les M3U8 sont des formats de fichiers utilisés par Apple notamment. Ils décrivent le/les flux qui seront lus.

Et quand on est connecté à la caméra, on voit bien ces fichiers TS. Ce sont des fichiers qui, là aussi, sont dans un format largement utilisé par Apple et qui n'est pas très simple à manipuler avec du code. En plus, les fichiers TS ont une durée de 1 seconde. C'est leur mise bout à bout qui permet d'avoir la preview, y compris avec le son. Une fois arrivée à la fin de la lecture des fichiers TS, la lecture reboucle. Ce mode est loin d'être optimal et a surtout l'inconvénient de ne pas être en temps réel. Il y a un décalage pouvant atteindre quelques secondes entre la preview et la réalité. Cela est donc contraignant et peut difficilement être utilisé pour des détections de mouvements avec prise de vue car le sujet risque d'être parti le temps que vous déclenchiez la prise de vue ou la vidéo. Mais vous pouvez avoir envie d'inclure quand même une preview dans votre application.

Si je prends l'exemple de Windows 8/8.1 et Windows Phone, le format TS est supporté pour la lecture mais le format M3U8 ne l'est pas nativement. Par contre, il existe un projet open source PhoneSM (<https://phonesm.codeplex.com/>) qui permet la lecture de ces fichiers M3U8 et TS. Le projet est



disponible avec un exemple complet pour Windows Phone. Une adaptation est nécessaire pour Windows 8 et 8.1.

Il est aussi possible de piloter et surveiller l'ensemble de la caméra. Tout cela se fait par simple appel d'URL en mode GET avec des paramètres dans l'URL. Pour avoir un statut, on envoie une commande de ce type : <http://10.5.5.9/CMD?t=PWD>

Et pour une commande : <http://10.5.5.9/CMD?t=PWD&p=%VAL>

Ce que renvoie la caméra dépend bien sûr de la demande. Du côté des commandes, il y a 2 éléments pilotables : le **bacpac** et la **camera**. Les commandes bacpac permettent d'obtenir des informations de statut sur les modes, le nom, version de la caméra; les commandes camera permettant de piloter la caméra en tant que telle afin de demander une prise de photo, changer un mode, lancer ou stopper un enregistrement.

Dans les paramètres, le mot de passe est systématiquement utilisé pour toute action de paramétrage ou de récupération de statut de la caméra. La nouvelle à la fois bonne et mauvaise, c'est qu'il existe une commande bacpac pour récupérer le mot de passe. C'est une mauvaise nouvelle d'un point de vue sécurité car cela implique qu'une fois connecté au wifi de la caméra, il est possible d'avoir accès à toutes les fonctionnalités. Le mot de passe de la caméra est le même que celui du wifi. Il faut donc au moins être connecté et avoir le mot de passe wifi pour accéder à la caméra. La bonne nouvelle, c'est que c'est une information de moins à demander à l'utilisateur.

L'appel de <http://10.5.5.9/bacpac/sd> renvoie donc un fichier contenant le mot de passe. Attention, toutes les commandes sont sensibles à la casse. La caméra peut être en mode veille pour récupérer le mot de passe. Toutes les commandes bacpac sont accessibles même quand la caméra est en veille.

Dans le second octet, le fichier renvoyé contient la longueur de la chaîne de caractères puis le mot de passe encodé en UTF8.

Comme toutes les commandes sont assez similaires et renvoient toutes un fichier de données, il est possible de faire une fonction simple comme la suivante qui permet de lire en mode asynchrone le flux venant de la caméra. Dans cette fonction, un WebRequest avec l'URL est créé, et la réponse est attendue. Ici, pour faire simple, on attend que la réponse soit vraiment entièrement là avant de la traiter. Un memory stream est créé et le buffer de lecture est vidé. Les fichiers renvoyés par la caméra sont petits et un buffer de 4K suffit très largement.

```
private async Task<byte[]> GetRequestStatusFile(string Req)
{
    byte[] buffer= new byte[4096];
    try
    {
        HttpWebRequest request = (HttpWebRequest)WebRequest.
Create(Req);
        HttpWebResponse response = (HttpWebResponse)await
request.GetResponseAsync();
        MemoryStream memsr = new MemoryStream();
        using (var sr = response.GetResponseStream())
        {
            int count = 0;
            do
            {
                count = sr.Read(buffer, 0, buffer.Length);
                memsr.Write(buffer, 0, count);
            }
            while (count != 0);
            return memsr.ToArray();
        }
    }
    catch (WebException ex)
    {
        return null;
    }
}
```

Ensuite, on peut simplement venir consommer cette fonction en l'appelant puis en récupérant ce qui a été retourné. Une chaîne de caractères est créée en transformant les byte en char et les char en string. Bon, il faut éviter de parler comme cela dans les dîners de famille car sinon, une bonne partie des convives va vous regarder étrangement.

```
mURL = «http://10.5.5.9/bacpac/sd»;
ret = await GetRequestStatusFile(mURL);
if (ret == null)
return (mConnected = false);
if (ret.Length > 2)
{
    char[] chars;
    chars = new char[ret.Length - 2];
    Decoder d = Encoding.UTF8.GetDecoder();
    d.GetChars(ret, 2, ret.Length - 2, chars, 0);
    mPassword = new string(chars);
}
```

Une fois le mot de passe récupéré, on peut par exemple s'amuser à lancer un enregistrement :

```
String mURL = http://10.5.5.9/camera/SH?t=» + mPassword
+ «&p=%01»
var ret = await GetRequestStatusFile(mURL);
if (ret.Length > 1)
{
    mIsRecording = Convert.ToBoolean(ret[1]);
}
```

Remplacez %01 par %00 pour arrêter l'enregistrement. Ce qui est retourné dans le fichier est le statut de la caméra. Si le second octet à la valeur 0,

alors la caméra n'enregistre pas. Si c'est 1 alors elle enregistre.

Voilà pour le fonctionnement de base avec un code exemple.

Il existe un projet complet en open source en .NET qui permet de piloter et d'avoir tous les statuts de la caméra sur Github : <https://github.com/Disposer/GoPro.Hero/pull/1/files>

Le code arrive également avec tous les tests unitaires pour tester l'ensemble des fonctions. Il est plutôt bien documenté. Et cerise sur le gâteau, il n'utilise que des classes compatibles avec Windows Phone et Windows 8/8.1.

L'automatisation et le pilotage à distance permet par exemple de prendre des vidéos de quelques secondes à intervalle régulier, puis des photos, puis de nouveau des vidéos ou des scénarios de ce type. N'hésitez pas à me faire part de ce que vous avez réalisé avec. Bon hack.

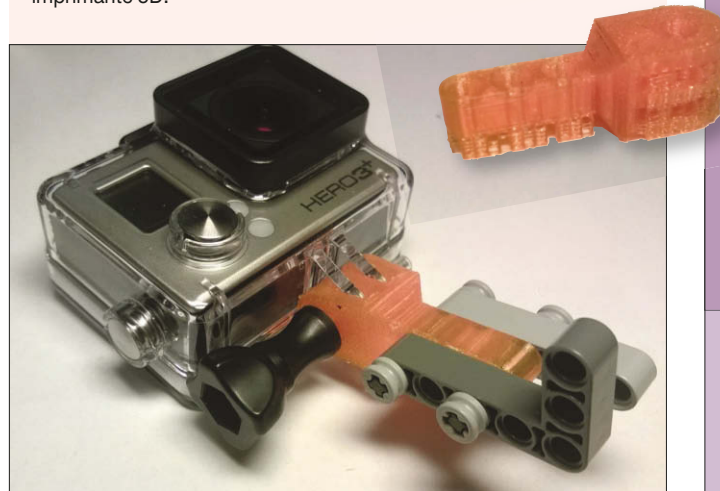


 Laurent Ellerbach
Audience Marketing Director
Microsoft EMEA
laurelle@microsoft.com

Passons au matériel

Avec la GoPro, il y a aussi des possibilités de hacking hardware. Les derniers modèles à partir de la Hero 2 possèdent une interface à l'arrière de la caméra qui permet de brancher certains accessoires comme un écran. Les pins sont très bien documentées sur ce site <http://chargeconverter.com/blog/?p=71> ainsi que le connecteur nécessaire. Il est par exemple possible de récupérer le signal vidéo (pin 2, 3 et 4), le signal USB (pin 7 et 8), le signal audio (pin 10 et 11 en sortie, 14 et 15 en entrée), de commander le bouton marche/mode (pin 12 à mettre à la masse pour activer), le bouton action (pin 13 à mettre à la masse pour activer), une entrée infrarouge (pin 16), quelques entrées numériques ainsi qu'une interface i2c pour le transfert de données (pin 28 pour DATA et 29 pour CLK). Pour ceux qui veulent se lancer dans le hacking hardware, c'est donc possible.

Et pour finir, il existe de nombreux accessoires notamment pour pouvoir la fixer, mais il n'en existe aucun permettant une fixation sur des éléments Lego. Après une rapide recherche, j'ai trouvé un objet sur Thingiverse (<http://www.thingiverse.com/thing:13518>) à imprimer sur mon imprimante 3D.



L'objet brut imprimé avec 3D Builder (application disponible gratuitement sur Windows Store) et une Replicator 2 de MakerBot
L'objet intégré avec la fixation d'un côté Lego Technic et de l'autre le boîtier GoPro !

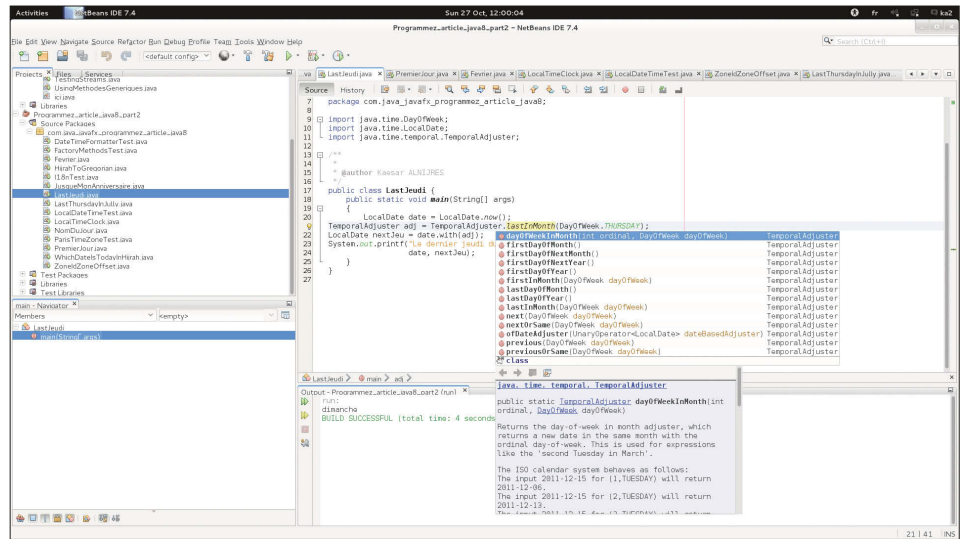
A la découverte de Java 8

2^e partie

Ce mois-ci nous continuons de découvrir les nouveautés de Java 8.

java.time

Enfin !!! Qui n'a jamais souffert de manipulation de date avec Java ? Je ne vais pas revenir sur les problèmes de date/temps dans les versions précédentes. Mais quelle heureuse surprise en découvrant les possibilités de ce package et la facilité avec laquelle on manipule les classes et les interfaces. Une petite remarque cependant, ne soyez surtout pas intimidé par le nombre très élevé de classes, interfaces, énumérations, et méthodes, ni par les nombreuses possibilités offertes pour faire quelque chose. C'est avant tout un avantage et parmi les objectifs principaux de cette nouvelle mouture. Je vais essayer de simplifier et de classer autant que possible. Il est nécessaire à mon avis de passer aux exemples après avoir une idée théorique de l'organisation du package.



Points forts

► Le design conceptuel est intuitif

Les noms donnés aux composants sont clairs et significatifs. La classe `LocalDate` par exemple manipule les dates locales, point. Pas de mélange avec le temps ou le fuseau horaire. En revanche `LocalDateTime` est à la fois pour la date et le temps. `ZonedDateTime` ajoute la notion de fuseaux horaires. `Duration` pour mesurer une durée, etc.

Autant que possible, les méthodes sont "cohérentes". Des méthodes portant un nom commun comme "now()", "from()", "plus()", existent pour la plupart des classes. Il y a des méthodes pour pratiquement tout faire.

Le temps, selon la conception de l'homme, est séparé du temps machine. L'homme calcule en jour, semaine, année, et la machine compte les nanosecondes depuis un moment donné. Pas question ici de perdre du temps pour des conversions interminables.

Exemple :

```
public static void main(String[] args)
{
    LocalDate date = LocalDate.now();
    LocalDateTime date2 = LocalDateTime.now();
    ZonedDateTime date3 = ZonedDateTime.now();
    Instant cetInstant=Instant.now();
}
```

► Immutable

La majorité de classes dans ce package sont immutables (inchangables). Pas de changement d'un objet après sa création, ce qui rend ces classes thread-safe.

► NullPointerException

La majorité de méthodes refusent de prendre null, comme argument et lancent `NullPointerException`. La majorité des méthodes ne retournent pas de null. Conséquence, un code plus robuste et la possibilité de chaîner des méthodes.

Exemple :

```
LocalDate date = LocalDate.now().plusDays(3);
```

► Utiliser les méthodes Factory

Pas de constructeurs. On construit des objet en utilisant des méthodes "factory", comme `now()`, `of()`, `from()`, `parse()`, `plus()`, `minus()`, etc.

Exemple :

```
try{
    LocalDateTime d0 = LocalDateTime.now();
    LocalDate d1 = LocalDate.of(2013,10,25);
    LocalDate d2 = LocalDate.from(d0);

    System.out.println ("d0: "+d0);
    System.out.println ("d1: "+d1);
    System.out.println ("d2: "+d2);

} catch (DateTimeException de) {
    System.out.println("Exception est lancée:"+de.getStackTrace());
}
```

Affichage :

```
d0: 2013-10-25T10:52:34.382
d1: 2013-10-25
d2: 2013-10-25
```

► Extensible

Le calendrier officiel du package est le ISO-8601, basé sur le calendrier grégorien. Mais il y a aussi des calendriers alternatifs. Vous pouvez même créer votre propre calendrier. Il est possible de personnaliser les ajusteurs de temps, les queries.

► Des classes spécialisées

Certaines manipulent la date sans le temps ni le fuseau horaire, d'autres avec le temps et/ou le fuseau horaire. Quelques classes voient le temps comme les humains, en année, mois, jour. D'autres voient le temps comme les machines où le temps est compté en nanosecondes depuis un certain moment de départ, ou époque. Date/temps est divisé en plusieurs unités

telles que date, temps, durée, etc. A vous de choisir. Deux énumérations importantes sont ajoutées pour les manipulations des jours et des mois.

► Nom de méthodes cohérentes

Les packages

Java.time et ses sous-packages :

java.time : le package principal. Le format utilisé est ISO-8601 basé sur le calendrier Grégorien et qui a la forme 2012-09-27 (pour 27/9/2012).

Attention : Le format officiel est aaaa-mm-jj.

java.time.chrono : calendrier alternatif

java.time.format : les outils nécessaires pour formater date/temps en entrée ou sortie

java.time.temporal : des interfaces et énumérations bas niveau pour les manipulations sur les unités date/temps utilisées par des classes haut niveau. Notez l'interface TemporalAdjuster et ses méthodes permettant de trouver le 1er et dernier jour d'un mois, d'une année, le prochain mercredi, etc.

java.time.zone : des classes pour les fuseaux horaires.

Une idée de classification :

Nous nous baserons sur quelques classes uniquement.

Temps humain

Dates uniquement

LocalDate, YearMonth, MonthDay, et Year.

Dates et temps

LocalDateTime

Temps uniquement

LocalTime

Date, temps et fuseaux horaires

ZonedDateTime, OffsetDateTime

Durée

Period

Pour les machines

Temps

Instant

Durée

Duration

Quelques méthodes

Construire une date directement :

► now() date/temps maintenant

► Of(), a partir des composants d'une date

► Parse(), une date sous forme d'une chaîne de caractères.

Construire avec l'arithmétique :

Des méthodes existent pour avoir des instances de date via une opération d'ajout ou de soustraction

Exemple :

```

    LocalDate d=LocalDate.now(); //maintenant
    LocalDate dd=d.plusDays(1); //demain, 1 jour de plus

    LocalDate dm= dd.minusDays(1);// 1 jour de moins

    Instant m=Instant.now();//maintenant machine.

    //ajouter une heure. Temps machine
    Instant uneHeurePlusTard= m.plusSeconds(60*60);
  
```

```

    System.out.println ("date origine: "+d);
    System.out.println("un jour de plus: "+dd);
    System.out.println("un jour de moins: "+dm);
  
```

Affichage :

d0: 2013-10-25T12:44:59.045

d1: 2013-10-25

d2: 2013-10-25

date origine: 2013-10-25

un jour de plus: 2013-10-26

un jour de moins: 2013-10-25

l'heure est: 2013-10-25T10:44:59.072Z

une heure après: 2013-10-25T11:44:59.072Z

Construire avec ajustement :

Avec with()

Exemple :

```

    TemporalAdjuster adj = TemporalAdjuster.lastInMonth(DayOfWeek.
    THURSDAY);
    LocalDate nextJeu = date.with(adj);
  
```

Convertir : la méthode from(), permet de convertir une instance en une autre, d'une nature différente

Comparer le temps : isBefore(), isAfter()

Combien de temps : avec until(), between()

Getters : plein de getters génériques ou prédéfinis sont disponibles

Des méthodes prédéfinies

Pour les méthodes très utilisées, au lieu d'utiliser des génériques, on peut utiliser les méthodes prédéfinies, telle que : getDayOfWeek() au lieu d'utiliser get(TemporalField), etc.

La méthode getDisplayName(), est utilisée intensivement avec les deux nouvelle enum DayOfWeek et Month pour présenter les jours et les langues différentes (I18N)

Exemple :

Le nom du jour

Je commence par construire un objet « de maintenant » et j'affiche le jour en français

```

    //je construis date maintenant
    LocalDate maintenant =LocalDate.now();
    //j'obtiens le nom du jour de maintenant en utilisant l'enum DayOfWeek

    DayOfWeek dotw = maintenant.getDayOfWeek();
    //j'affiche selon le format et la langue désirée
    System.out.println(dotw.getDisplayName(TextStyle.FULL, Locale.FRENCH));
  
```

Calculer une durée

Je vais calculer le temps restant jusqu'à mon anniversaire

```

    LocalDate date=LocalDate.now();
    //j'utilise until pour calculer le nombre de jour
    long nombreDeJours=date.until(LocalDate.of(2013,12,27), DAYS);
  
```

Le dernier jeudi du mois

```

    LocalDate date = LocalDate.now(); //date maintenant
    //Ajuster la date pour retourner le dernier jeudi du mois
  
```

```
TemporalAdjuster adj = TemporalAdjuster.lastInMonth(DayOfWeek.
THURSDAY);
//dernier jeudi est construit à partir de la date actuelle avec with()

LocalDate nextJeu = date.with(adj);
System.out.printf("Le dernier jeudi du mois %s est %s.%n",
    date, nextJeu);
```

Heure d'été, heure d'hiver

```
//déclarez une zone. Il suffit de changer l'ID comme
//"Asia/Tokyo", "America/Los_Angeles", ...
ZoneId zone=ZoneId.of("Europe/Paris");
//je vais calculer d'ici dix ans
for(int i=0;i<10;i++)
{
    int an=2013+i;
//on travaille avec le fuseau horaire donc ZonedDateTime
//on utilise of() pour la construction, ça nous permet d'indiquer l'année

    ZonedDateTime zdate=ZonedDateTime.of(LocalDateTime.of(an,
10,1,1,1), zone);
//ZoneRules les règles du changement d'heure, si appliqué

    ZoneRules rules=zone.getRules();
//Heure été. Je sélectionne la transition avant la date zdate 10 octobre
//la transition après c'est l'heure d'hiver

    ZoneOffsetTransition transition=rules.previousTransition
(Instant.from(zdate));
//je construis une date de la transition avant à laquelle on modifie l'heure d'été

    LocalDateTime d=LocalDateTime.ofInstant(transition.get
Instant(), zone);
    System.out.println("Année "+an);
    System.out.printf("%-14s le %s %d %s %d%n",
        "Heure d'été",
//le nom du jour du changement

        d.getDayOfWeek().getDisplayName(TextStyle.FULL, Locale.FRENCH),
//le jour

        d.getDayOfMonth(),
//le mois

        d.getMonth().getDisplayName(TextStyle.FULL, Locale.FRENCH),
//et l'année pour le calcul

        d.getYear());
//la transition après donne l'heure d'hiver

    transition=rules.nextTransition(Instant.from(zdate));
    d=LocalDateTime.ofInstant(transition.getInstant(), zone);

    System.out.printf("%-14s le %s %d %s %d%n",
        "Heure d'hiver",
        d.getDayOfWeek().getDisplayName(TextStyle.FULL, Locale.FRENCH),
        d.getDayOfMonth(),
        d.getMonth().getDisplayName(TextStyle.FULL, Locale.FRENCH),
        d.getYear());
    System.out.println("=====");
```

Une API encore plus étoffée

Depuis la création de Java, chaque nouvelle version apporte plus d'interfaces, classes et énumérations, afin de rendre le langage plus efficace, mais aussi plus facile pour les développeurs. Cette nouvelle version n'est pas une exception. A part les nouveaux paquetages (packages) cités précédemment, comme `java.util.time` et `java.util.function`, beaucoup d'autres ont été remaniés suite à l'introduction de "default methods" pour les interfaces. Des classes, interfaces, méthodes ont été ajoutées pour harmoniser l'ensemble. Les objectifs sont :

- Performance
- Syntaxe plus intuitive
- Développement fonctionnel

La liste est vraiment très longue et concerne pratiquement tout. En particulier, les Collections, Concurrency, Entrées/sorties, Reflection. Des nouvelles annotations sont ajoutées.

Collections Framework

Ce framework s'est enrichi de méthodes très utiles :

- L'apparition de la méthode `List.sort(Comparator)`, permettant de remplacer très avantageusement `Collections.sort(list, comparator)`.
- Des passerelles vers Stream : `Collection.parallelStream()` et `Collection.stream()`
- De nouvelles méthodes permettant d'utiliser les expressions Lambda, les agrégation de streams .

Nashorn remplace Rhino

Un moteur JavaScript permettant d'intégrer JavaScript dans les applications Java. Développé par Oracle et ouvert à l'open source et OpenJDK. Il est basé sur la machine de Da Vinci. Il s'intègre très bien à Java 8, spécialement les Lambda. A titre d'exemple il est possible de passer une expression Lambda à la place d'une fonction JavaScript.

Un petit exemple HelloWorld

```
import javax.script.*;

public class EvalScript {
    public static void main(String[] args) throws Exception {
        // script mangager

        ScriptEngineManager factory = new ScriptEngineManager();
        // nashorn

        ScriptEngine engine = factory.getEngineByName("nashorn");
        // eval de JavaScript
        try {
            engine.eval("print('Hello, World!');");
        } catch (final ScriptException se) { se.printStackTrace(); }
    }
}
```

Invoquer des scripts (ligne de commande)

Avec l'utilitaire `jjs` nomFichier.js. La commande `jjs` sans nom de fichier permet de lancer une session JavaScript interactive.

Bonne année

Java 8 arrive en mars 2014.

Références

<http://docs.oracle.com/javase/tutorial/index.html>

<http://download.java.net/jdk8/docs/technotes/guides/scripting/nashorn/intro.html>

https://blogs.oracle.com/nashorn/entry/nashorn_and_lambda_what_the

 Kaesar Alnijres - Développeur Java
www.java-javafx.com

Introduction à Lucene .NET

Lucene est ce que l'on appelle communément un « moteur de recherche ». Lorsqu'on emploie ce terme, beaucoup de lecteurs vont penser au moteur de recherche Google. A juste titre, il va permettre de réaliser des recherches sur vos données, mais ce n'est pas sa seule utilisation. En fait il faut plus le considérer comme une base de données avec des fonctions de recherches avancées. Par exemple sur la majorité des sites e-commerce, on utilise ce genre d'outil pour la recherche mais également pour toute la récupération des informations produits et catégories.

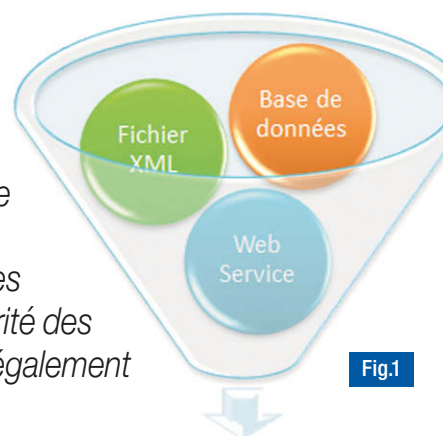


Fig.1

Lucene .Net est la version qui a été portée sur la plateforme Microsoft. Lors de la rédaction de cet article, la dernière version est la 3.0.3. Elle est téléchargeable depuis <http://lucenenet.apache.org/>.

Vous trouverez d'autres bibliothèques additionnelles telles que Lucene.Net Contrib. Elles permettent par exemple de mieux gérer la langue française, de faire des recherches par facettes, etc... Vous trouverez plus d'informations sur le site. Son utilisation est basée sur deux mécanismes : L'indexation et l'interrogation.

L'indexation

L'indexation est le fait d'écrire des données dans Lucene. Les données sont stockées au format plat dans ce que l'on appelle un document, lui-même stocké dans un index. Un index est comme une base de données, vous pouvez en avoir plusieurs. Par exemple, si nous revenons sur un site e-commerce, il pourrait y en avoir un sur les produits, et un pour les catégories.

Lors de cette phase, les données peuvent venir de différentes sources (Base de données, ERP, Fichiers, ...) : Fig.1.

Pour simplifier la lecture de cet article, nous considérerons que la couche de données est déjà réalisée avec Entity Framework sur la base AdventureWorks.

Pour rappel voici la table Product d'AdventureWorks : Fig.2.

La première chose est d'ajouter une référence à la bibliothèque Lucene.Net (téléchargée via le lien ci-dessus).

Fig.2

Column Name	Data Type	Allow Nulls
ProductID	int	<input type="checkbox"/>
Name	Name:nvarchar(50)	<input type="checkbox"/>
ProductNumber	nvarchar(25)	<input type="checkbox"/>
MakeFlag	Flag:bit	<input type="checkbox"/>
FinishedGoodsFlag	Flag:bit	<input type="checkbox"/>
Color	nvarchar(15)	<input checked="" type="checkbox"/>
SafetyStockLevel	smallint	<input type="checkbox"/>
ReorderPoint	smallint	<input type="checkbox"/>
StandardCost	money	<input type="checkbox"/>
ListPrice	money	<input type="checkbox"/>
Size	nvarchar(5)	<input checked="" type="checkbox"/>
SizeUnitMeasureCode	nchar(3)	<input checked="" type="checkbox"/>
WeightUnitMeasureCode	nchar(3)	<input checked="" type="checkbox"/>
Weight	decimal(8, 2)	<input checked="" type="checkbox"/>
DaysToManufacture	int	<input type="checkbox"/>
ProductLine	nchar(2)	<input checked="" type="checkbox"/>
Class	nchar(2)	<input checked="" type="checkbox"/>
Style	nchar(2)	<input checked="" type="checkbox"/>
ProductSubcategoryID	int	<input checked="" type="checkbox"/>
ProductModelID	int	<input checked="" type="checkbox"/>
SellStartDate	datetime	<input type="checkbox"/>
SellEndDate	datetime	<input checked="" type="checkbox"/>
DiscontinuedDate	datetime	<input checked="" type="checkbox"/>
rowguid	uniqueidentifier	<input type="checkbox"/>
ModifiedDate	datetime	<input type="checkbox"/>

Ajoutons la directive using suivante :

```
using System.Net;
```

Nous allons créer un document :

```
Documents.Document doc = new Documents.Document();
```

Le document est l'objet qui va contenir les informations d'un élément. Par rapport à une base de données, il peut être comparé à une ligne. Il faut ensuite lui ajouter les Fields. Il existe neuf constructeurs pour cet objet. Les principaux sont les suivants :

```
public Field(string name, string value, Field.Store store, Field.Index index);
public Field(string name, string value, Field.Store store, Field.Index index, Field.TermVector termVector);
```

► **name** : correspond au nom du champ.

► **value** : valeur du champ au format texte.

► **store** : indique si le champ doit être stocké dans l'index. Dans la plupart des cas, cette valeur est à Yes. Il peut être à No, lorsque l'on crée une colonne utilisée uniquement pour les tris.

► **index** : indique le mode d'indexation du champ

NO : aucune recherche ne pourra être faite sur cette colonne mais elle pourra être récupérée lors d'une recherche sur un autre champ.

ANALYZED : champ pouvant faire partie d'une recherche. Le champ est également analysé pour faire des recherches approximatives.

NOT_ANALYZED : champ pouvant faire partie d'une recherche. Le champ ne sera pas analysé, c'est-à-dire que seule une recherche exacte pourra être faite.

NOT_ANALYZED_NO_NORMS : champ pouvant faire partie d'une recherche. Le champ ne sera pas analysé. Les normes sont également désactivées, c'est à dire qu'un certain nombre de mécanismes seront désactivés. C'est le cas par exemple sur le changement de priorité d'un champ.

ANALYZED_NO_NORMS : champ pouvant faire partie d'une recherche. Le champ est également analysé. Les normes sont désactivées.

► **termVector** : indique si Lucene doit stocker les term Vector. Ce sont des statistiques sur la position et l'utilisation de contenus de documents. Ceci peut être utile pour comparer des textes.

Nous allons créer 3 champs :

```
Documents.Field field1 = new Documents.Field(«ProductID»,
    item.ProductID.ToString(),
    Documents.Field.Store.YES,
    Documents.Field.Index.NOT_ANALYZED);
Documents.Field field2 = new Documents.Field(«ListPrice»,
    item.ListPrice.ToString(),
    Documents.Field.Store.YES,
    Documents.Field.Index.NO);
Documents.Field field3 = new Documents.Field(«Name»,
    item.Name,
    Documents.Field.Store.YES,
    Documents.Field.Index.ANALYZED);
```

Sur le champ 1 (l'id), nous pourrions faire une recherche exacte.

Sur le champ 2 (Le prix), nous ne pourrions pas faire de recherche, le champ remontera lors d'une recherche sur un autre champ.

Sur le champ 3 (Le nom), nous pourrions faire une recherche approximative. Maintenant, il suffit d'ajouter les champs au document :

```
doc.Add(field1);
doc.Add(field2);
```



```
doc.Add(field3);
```

Maintenant que le document est créé, il faut créer un Analyzer qui va traiter les champs précédemment indiqués comme étant à analyser.

```
Analysis.Standard.StandardAnalyzer analyser =
    new Analysis.Standard.StandardAnalyzer(Util.Version.LUCENE_30);
```

Il existe un très grand nombre d'Analyzer : [Fig.3](#).

Dans notre exemple, nous avons utilisé l'Analyzer standard. Si nous voulions indexer du texte français, nous utiliserions plutôt `Fr.FrenchAnalyzer` (disponible via la librairie `Contrib`). En effet ce dernier contiendra les spécificités liées à la langue française comme par exemple les stop words (mots à ignorer lors de l'analyse car ils sont jugés comme non pertinents : auxquelles, vous, vôtre, y, à, ça, ...).

Il faut ensuite créer l'objet qui écrira dans l'index :

```
Index.IndexWriter writer = new
    Index.IndexWriter(Store.FSDirectory.Open(@"c:\temp\Index»)
        analyser,
        true,
        Index.IndexWriter.MaxFieldLength.LIMITED);
```

Le premier paramètre est le Directory où sera stocké l'index. Il en existe plusieurs de base : [Fig.4](#).

Voici les plus utilisés :

- `FileSwitchDirectory` : permet de créer deux répertoires d'index en fonction des extensions.
- `FSDirectory` : permet de stocker l'index avec des fichiers plats dans un répertoire physique. C'est le plus utilisé car il est persistant et performant.
- `RAMDirectory` : permet de stocker l'index en mémoire. Cet index est très rapide, mais il faut faire attention à la quantité d'informations que l'on stocke avant de l'utiliser. De plus en cas de redémarrage du serveur, celui-ci sera vide.

Ici nous avons choisi de stocker notre index en fichiers. Le paramètre suivant est l'indexeur défini précédemment. Le troisième paramètre, permet d'indiquer si l'on doit créer (ou recréer) l'index. S'il est à `false`, les données seront ajoutées à la suite. Le dernier champ indique si Lucene doit stocker tous les Vector Terms ou limiter leur utilisation.

La dernière chose à faire est d'ajouter le document dans l'index.

Fig.3

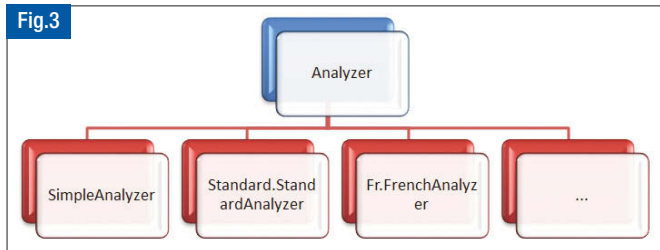
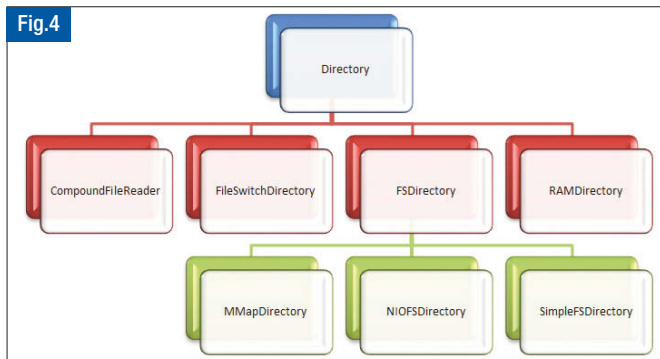


Fig.4



```
writer.AddDocument(doc);
writer.Optimize();
```

La méthode `Optimize` permet de réduire la taille du document et d'améliorer les performances. Bien sûr il faut ensuite penser à le détruire une fois tous les documents ajoutés.

```
writer.Dispose();
```

Nous avons vu comment stocker les données, nous allons maintenant nous intéresser à leur récupération.

L'interrogation

La première chose à faire est de définir l'analyseur :

```
var analyzer = new
    Analysis.Standard.StandardAnalyzer(Util.Version.LUCENE_30);
```

Ensuite, il faut définir où nous allons faire la recherche via un `IndexSearch` :

```
var searcher = new Search.IndexSearcher(
    Store.FSDirectory.Open(@"c:\temp\Index»));
```

Puis, nous définissons l'élément que nous allons rechercher :

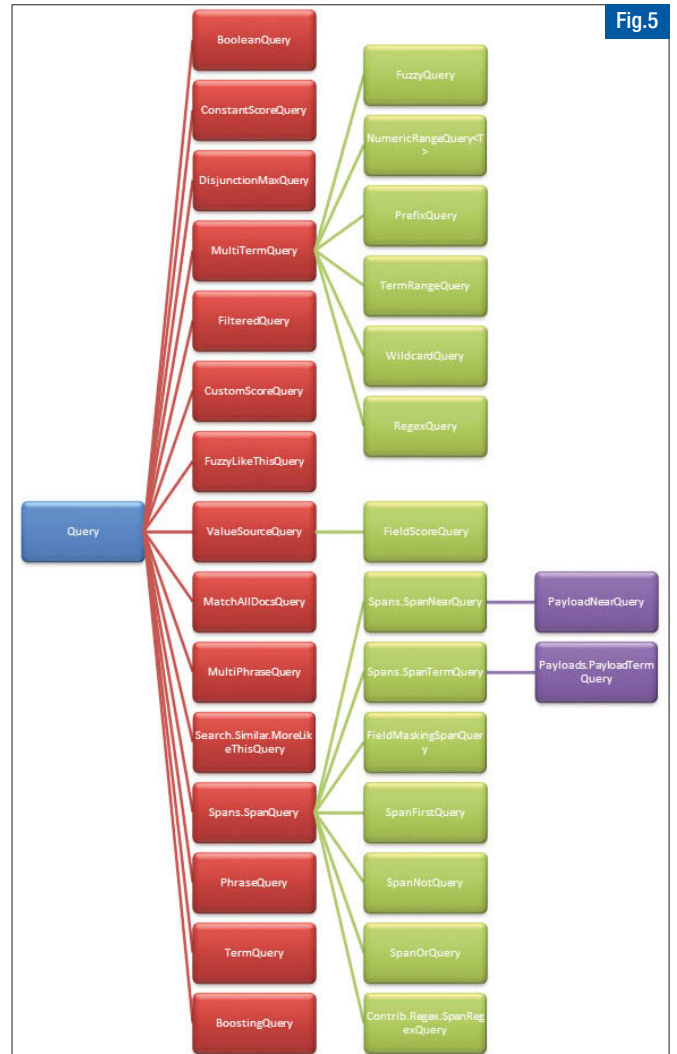
```
Index.Term term = new Index.Term(«Name», «mountain»);
```

Ici, nous allons chercher « mountain » dans le champ `Name`.

Ensuite, il faut créer une requête via l'objet `Query` : [Fig.5](#).

Il existe un très grand nombre de requêtes. Voici les plus utilisées :

Fig.5



- ▀ **TermQuery** : Permet de faire une recherche d'une valeur sur un champ.
 - ▀ **FuzzyQuery** : Permet de faire une recherche approximative (mauvaise orthographe)
 - ▀ **WildcardQuery** : Permet de faire une recherche en utilisant des méta-caractères (wild card) comme « * » ou « ? ». Ex : test* : Recherche toutes les informations commençant par test.
 - ▀ **BooleanQuery** : Permet d'exécuter plusieurs requêtes et de faire des unions ou des intersections.
- Ici, nous allons définir une requête simple :

```
Search.Query query1 = new Search.TermQuery(term);
```

Il ne reste plus qu'à faire la recherche :

```
Search.TopDocs hits = searcher.Search(query, null, 1000,
Search.Sort.RELEVANCE);
```

Le premier paramètre est la requête. Le second est le filtre, qui permet de filtrer les résultats. Le troisième indique le nombre de résultats maximum et enfin le dernier indique le tri (ici par pertinence).

La dernière étape est le parcours des résultats et la récupération de ce dernier :

```
for (int i = 0; i < hits.TotalHits && i < 1000; i++)
{
    Documents.Document doc = searcher.Doc(hits.ScoreDocs[i].Doc);
    string name = doc.Get(«Name»);
    string productID = doc.Get(«ProductID»);
    string listPrice = doc.Get(«ListPrice»);
}
```

Via la méthode Get sur Document, nous pouvons récupérer une propriété (un champ) du document.

Conclusion

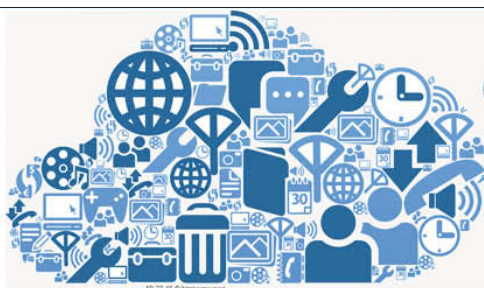
Cet article n'est qu'une introduction à Lucene, ses capacités sont beaucoup plus poussées. Il existe également d'autres moteurs de recherche basés sur Lucene, mais avec des fonctionnalités supplémentaires; c'est le cas par exemple de SolR et d'ElasticSearch.



Xavier Millot - Architecte .NET

!telios

100% nouveau, 100% Cloud



CLOUDMAGAZINE.FR

100 % cloud computing

Microsoft
tech·days
2014

11,12,13 février
au Palais des Congrès de Paris

Je m'inscris

ACTUALITES AVIS D'EXPERT ANALYSES

A la une

SAP : le cloud soutient l'éditeur en 2013

SAP vient d'annoncer les résultats pour le 4e trimestre 2013 et pour l'ensemble de l'année 2013. Le cloud et les services liés ont aidé l'éditeur à se maintenir. Globalement, SAP a généré un chiffre d'affaire de + 5 milliards € pour le dernier trimestre. Les souscriptions cloud / saas pèse 209...

CloudUnit : héberger et déployer des applications Java

un nouveau service Paas Java sera disponible dans les prochains mois : CloudUnit. Il s'agit d'un projet français pour héberger et déployer des applications Java et JEE. Les principales fonctions annoncées sont : montage des environnements Java en quelques minutes, choix technologiques standards (...)

Avis d'expert

Mesurer entre les lignes

Le réseau de compteurs intelligents 4-G sans liens physiques proposé par le Royaume Uni forme une toute nouvelle infrastructure d'importance critique : en tant que telle, elle aura besoin de procédures de tests de haut vol, annonce Marc Meulenstein...

Analyse

IBM investit dans le Cloud : 1 milliard

Quand les grands acteurs investissent dans le Cloud Computing cela fait mal, très mal. En France, on parle, pour les plus gros projets, de 150 - 200 millions. Quand IBM annonce un investissement pour renforcer son offre, l'échelle n'est plus la même...

Toute l'actualité

SFR Business Team mise sur le cloud hybride et HP

SFR Business Team est partenaire HP pour HP Bursting partner. Les entreprises utilisant CloudSystem (HP) pourront utiliser les ressources cloud public SFR localisées en France. Ce débordement doit se faire en toute transparence pour l'utilisateur. L'offre de cloud hybride répond aux différentes...

Cheops Technology : partenaire Cloudwatt

Cheops Technology annonce un partenariat avec Cloudwatt pour proposer une offre cloud hybride. Nicolas Leroy Fleuriot, Président de Cheops déclare : « Grâce aux technologies OpenStack conjointement utilisées par Cheops et Cloudwatt, nous allons répondre aux attentes du marché avec la mise en oeuvre...

Aspaway ouvre une agence sur Lyon

Aspaway (hébergement / cloud computing) ouvre une agence régionale basée à Lyon. L'objectif de l'agence de Lyon est de proposer des prestations d'hébergement aux éditeurs et entreprises lyonnaises désireuses de sélectionner un partenaire local pouvant les accompagner au quotidien dans leurs grands...

Sécurité

Cloud Computing et la sécurité

Les menaces changent avec le Cloud. Cloudwatt apporte sa vision, son analyse. vidéo de 32 minutes.

Cloud Computing et nouvelles menaces



Newsletters

Inscrivez-vous à nos newsletter !

Rechercher



Identifier et corriger certains problèmes de performance liés à l'utilisation de frameworks ORM

Un outil ORM (Object-Relational Mapping ou mapping objet-relationnel en français) offre de nombreux avantages : gain de temps de développement, lisibilité du code, réduction de répétition du code et possibilité offerte aux développeurs de rester dans le concept objet. Cependant, la solution ORM est souvent pointée du doigt lorsqu'un problème de performance survient. Dans la majorité des cas, c'est une mauvaise utilisation qui en est la cause. Cet article présente, au travers d'exemples, des problèmes rencontrés dus à une mauvaise utilisation de Doctrine, ORM pour PHP, et les outils permettant de les identifier et les corriger. La description est également valable pour les autres produits ORM.

Symfony2 étant un framework PHP très populaire, c'est avec celui-ci que l'exemple est réalisé dans cet article. Symfony2 utilise par défaut Doctrine et permet de mettre en place facilement et rapidement des formulaires liés à des entités métiers. La base de données retenue est MySQL, fréquemment utilisée en environnement PHP-Symfony2.

Présentation de l'exemple

L'exemple s'articule autour d'un bundle permettant la création d'un produit. Un bundle est un répertoire contenant un ensemble de fichiers et de sous-répertoires permettant d'implémenter une ou plusieurs fonctionnalités; fonction similaire au plugin dans Symfony1. Un produit est représenté par un libellé et est rattaché à une catégorie. Ce bundle contient donc deux entités Doctrine, Produit et Catégorie, définies avec des annotations :

```
// MonBundle\Entity\Categorie.php
/**
 * @var integer
 *
 * @ORM\Column(name="id", type="integer")
 * @ORM\Id
 * @ORM\GeneratedValue(strategy="AUTO")
 */
private $id; // identifiant de la catégorie

/**
 * @var string
 *
 * @ORM\Column(name="libelle", type="string", length=255)
 */
private $libelle; // libellé de la catégorie

// MonBundle\Entity\Produit.php
/**
 * @var integer
 *
 * @ORM\Column(name="id", type="integer")
 * @ORM\Id
 * @ORM\GeneratedValue(strategy="AUTO")
 */
private $id; // identifiant du produit

/**
 * @var string
 *
```

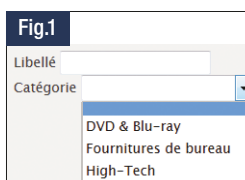
```
* @ORM\Column(name="libelle", type="string", length=255)
*/
private $libelle; // libellé du produit

/**
 * @var \MonBundle\Entity\Categorie
 *
 * @ORM\ManyToOne(targetEntity="MonBundle\Entity\Categorie")
 * @ORM\JoinColumn(name="id_categorie", referencedColumnName="id")
 */
private $categorie;
```

Ces entités sont liées par une relation ManyToOne : un produit ne peut avoir qu'une seule catégorie, et une catégorie peut être affectée à plusieurs produits. Le bundle inclut également un contrôleur gérant l'action de création du produit (redirection vers le formulaire, insertion du produit dans la base). Le formulaire de création doit permettre de saisir le libellé du produit et de sélectionner sa catégorie parmi celles déjà existantes :

```
// MonBundle\Form\ProduitType.php
public function buildForm(FormBuilderInterface $builder, array $options) {
    // champ de saisie du libellé
    $builder->add('libelle', null, array('label' => 'Libellé'));
    // champ de sélection de la catégorie
    $builder->add('categorie', null, array('label' => 'Catégorie'));
}
```

Les relations entre les entités Doctrine sont définies de sorte que l'accès à la propriété faisant référence à une autre retourne directement celle-ci et non l'id servant à faire la liaison. La propriété \$categorie de l'entité Produit est donc une entité Catégorie et non un id. Symfony2 construit alors le formulaire en fonction de ces informations et des paramètres ajoutés par le développeur. Le deuxième paramètre du champ "categorie" étant à "null" (dans la méthode add() de l'objet form), Symfony2 utilise par défaut le type de champ le plus adapté : comme il s'agit d'une entité, il crée une liste déroulante contenant toutes les catégories existantes.



Dans la vue Twig (moteur de template PHP intégré dans Symfony2), un champ texte pour le libellé du produit et une liste déroulante pour la sélection de la catégorie sont affichés avec {{ form(form) }} (Fig.1).

Au fur et à mesure de l'avancement du pro-

jet, le contenu de la table "categorie" grossit et contient plusieurs milliers d'éléments. Une liste déroulante n'est alors plus adaptée; sélectionner une entrée parmi une liste très longue étant pénible. On change donc son mode de sélection, avec un champ texte pour saisir l'identifiant de la catégorie souhaitée.

```
// Vue Twig : create.html.twig
<form action="{{ path('produit_create') }}" method="post">
    // affichage du champ du libellé du produit
    {{ form_row(form.libelle) }}
    // affichage du champ de l'identifiant de la catégorie
    <label>Catégorie </label><input id=" monbundle_produit_
categorie" name=" monbundle_produit[categorie]" type="text"
/><br />
    // affichage du champ hidden CSRF
    {{ form_widget(form.token) }}
    <input type="submit" value="Envoyer" />
</form>
```

A la soumission du formulaire on obtient une simple requête d'insertion (Fig.2).

Pour simplifier la récupération de cet identifiant, on peut ajouter un champ texte permettant cette fois de saisir le libellé de la catégorie avec une fonction d'autocomplétion. C'est cette fonction d'autocomplétion qui sera chargée de renseigner le bon identifiant dans un champ caché (Fig.3).

L'affichage du formulaire est adapté pour l'utilisateur. Cependant, la page met plusieurs secondes à s'afficher alors que rien n'est affiché par défaut (l'autocomplétion s'activant seulement après la saisie d'au moins un caractère).

Outils permettant d'identifier le problème

Pour nous aider à identifier le problème, plusieurs outils sont à notre disposition :

- la barre du développeur de Symfony2 : disponible en bas de chaque page en environnement de développement, elle affiche un résumé des données de profilage comme le nom de la route utilisée, la mémoire utilisée, les requêtes SQL, ...
- xdebug : pour le débogage pas à pas mais aussi pour la génération de fichiers de profilage. Le profilage est activé en mettant le paramètre xdebug.profiler_enable à 1 dans php.ini.
- WinCacheGrind (sous Windows) ou KCacheGrind (sous Linux) : pour l'analyse des fichiers de profilage générés par xdebug.
- le profilage de MySQL : pour les projets n'utilisant pas Symfony2 (et donc ne disposant pas de la barre du développeur), les requêtes SQL peuvent être récupérées dans un fichier de log. Il suffit d'ajouter les

options `general_log=1` et `general_log_file="chemin_du_fichier/query.log"` sous la section `[mysqld]` du fichier de configuration de MySQL.

La barre du développeur de Symfony2 a permis de mettre en évidence la récupération de la table "categorie" (Fig.4).

L'analyse de la trace générée par xdebug montre la génération d'une liste déroulante ainsi que le temps passé dessus. Le chargement de cette liste comprend la récupération et l'hydratation des données, ainsi que l'initialisation de la liste (Fig.5).

Le problème vient donc de la génération de cette liste déroulante contenant l'ensemble des catégories. L'erreur est d'avoir changé uniquement la vue Twig mais pas l'objet form, ce dernier permettant de construire le formulaire. En conséquence, la liste des catégories est de toute manière récupérée et instanciée même si elle n'est pas affichée dans la vue.

Proposition de correction

Une solution consiste à utiliser les convertisseurs de données de Symfony2. Les convertisseurs de données transforment une chaîne de caractères (par exemple, saisie par l'utilisateur) en une entité et inversement. Ceci permet d'avoir un simple champ texte pour la saisie des catégories en créant un nouveau type de champ héritant du champ texte et utilisant le convertisseur. Tout d'abord, il faut créer le convertisseur qui sera donc chargé de convertir les données :

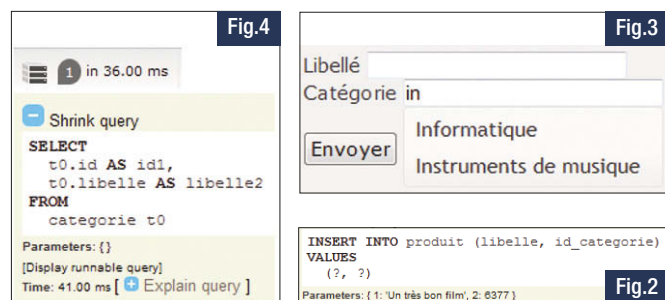
```
// MonBundle\Form\Type\CategorieToNumberTransformer.php
class CategorieToNumberTransformer implements DataTransformerInterface
{
    /** @var ObjectManager */
    private $om;

    /** @param ObjectManager $om */
    public function __construct(ObjectManager $om) {
        $this->om = $om;
    }

    /**
     * Transforme un objet (Categorie) en un nombre (identifiant).
     *
     * @param \MonBundle\Entity\Categorie|null $categorie
     * @return integer
     */
    public function transform($categorie) {
        if (null === $categorie) { // renvoie true si null est égal
            et de même type que $categorie
            return "";
        }

        return $categorie->getId();
    }

    /**
     * Transforme un nombre (identifiant) en une entité (Categorie).
     *
     * @param integer $id
     * @return \MonBundle\Entity\Categorie|null
     * @throws TransformationFailedException si l'objet (Categorie)
     n'est pas trouvé.
     */
    public function reverseTransform($id) {
        if (!$id) {
```



Function	Self	Cum.	File	Called from
Symfony\Bridge\Doctrine\Form\ChoiceList\EntityChoiceList->load	34ms	67 193ms	EntityChoiceList.php	EntityChoiceList.php (173)
Symfony\Component\Form\Extension\Core\ChoiceList\ObjectChoiceList->initialize	129ms	48 837ms	ObjectChoiceList.php	EntityChoiceList.php (430)
Symfony\Component\Form\Extension\Core\ChoiceList\ChoiceList->addChoices	2 787ms	45 320ms	ChoiceList.php	ChoiceList.php (116)
Doctrine\ORM\EntityRepository->findAll	-	18 326ms	EntityRepository.php	EntityChoiceList.php (425)
Doctrine\ORM\Internal\Hydration\AbstractHydrator->hydrateAll	109ms	18 243ms	AbstractHydrator.php	BasicEntityPersister.php (848)

```

        return null;
    }

    // récupération de la catégorie
    $categorie = $this->om
        ->getRepository('MonBundle:Catégorie')
        ->findOneBy(array('id' => $id))
    ;

    // lance une exception si l'identifiant n'a pas été trouvé
    if (null === $categorie) {
        throw new TransformationFailedException(sprintf(
            'La catégorie ayant l'identifiant "%s" n\'existe pas !',
            $id
        ));
    }

    return $categorie;
}

```

Il faut ensuite créer un nouveau type de champ de formulaire héritant du champ texte et utilisant notre convertisseur, et le déclarer en tant que service (nécessaire car une connexion à la base de données est requise). Avec l'injection de dépendance de Symfony2, l'ObjectManager sera passé en paramètre dans le constructeur de notre classe.

```

// MonBundle\Form\Type\CategorieSelectType.php
class CatégorieSelectorType extends AbstractType
{
    /** @var ObjectManager */
    private $om;

    /** @param ObjectManager $om */
    public function __construct(ObjectManager $om) {
        $this->om = $om;
    }

    public function buildForm(FormBuilderInterface $builder, array
    $options) {
        $transformer = new CatégorieToNumberTransformer($this->om);
        $builder->addModelTransformer($transformer); // Ajout du
    convertisseur
    }

    public function getParent() {
        return 'text'; // hérite du champ text
    }

    public function getName() {
        return 'categorie_selector';
    }
}

// services.xml
<service id="mon.bundle.form.type.categorieSelector" class="Mon
Bundle\Form\Type\CategorieSelectorType">
    <tag name="form.type" alias="categorie_selector" />
    <!-- ajout de l'EntityManager en argument du constructeur -->
    <argument type="service" id="doctrine.orm.entity_manager" />

```

```
</service>
```

Enfin, il faut utiliser ce type pour le champ 'categorie' :

```

$builder ->add('categorie', 'categorie_selector', array('label'
=> 'Catégorie'));

```

Avec ces modifications, la liste déroulante créée par défaut n'existe plus, de même que le problème de lenteur au chargement de cette page.

Problème dû au lazy loading

Il convient de faire attention également au lazy loading, fonctionnalité activée par défaut dans la plupart des outils ORM. Le lazy loading permet d'initialiser un objet seulement lorsqu'on y accède (par exemple via un accesseur). Mal utilisé, il peut rendre l'application moins performante en multipliant le nombre de requêtes effectuées. Prenons l'exemple d'une page d'affichage d'un produit avec sa catégorie :

```

// MonBundle\Controller\ProduitController.php
public function showProduitAction($id)
{
    // Une première requête pour récupérer le produit
    $produit = $this->getDoctrine()
        ->getRepository('MonBundle:Produit')
        ->findById($id);

    // Une deuxième pour récupérer la catégorie associée
    $categorie = $produit->getCategorie();
}

```

Dans cet exemple, avec le lazy loading activé, le premier appel génère une requête uniquement pour le produit et hydrate l'objet \$produit avec les données récupérées. Lorsque \$produit->getCategorie() est appelé, Doctrine génère une deuxième requête pour récupérer la catégorie associée au produit. Pour l'affichage d'une page de liste des produits avec leur catégorie, une requête est donc lancée pour chaque produit. Si la page doit afficher 20 produits, il y aurait au minimum 21 requêtes, ce qui est beaucoup trop ! En SQL, la récupération de ces données se fait en une seule requête avec une jointure.

Avec Doctrine, pour forcer la jointure, il suffit de rajouter l'option fetch="EAGER" sur la définition entre les entités. À chaque fois qu'un produit sera récupéré, la requête utilisera une jointure. Une seule requête sera donc lancée pour afficher la page de liste des 20 produits.

```

// MonBundle\Entity\Produit.php
/**
 * @var \MonBundle\Entity\Categorie
 *
 * @ORM\ManyToOne(targetEntity="MonBundle\Entity\Categorie",
fetch="EAGER")
 * @ORM\JoinColumn(name="id_categorie", referencedColumnName=
"id")
 */
private $categorie;

-- requêtes générées AVANT l'ajout du EAGER
SELECT t0.id AS id1, t0.libelle AS libelle2, t0.id_categorie
AS id_categorie3 FROM produit t0
SELECT t0.id AS id1, t0.libelle AS libelle2 FROM categorie t0
WHERE t0.id = '6380'

```

```

SELECT t0.id AS id1, t0.libelle AS libelle2 FROM categorie t0
WHERE t0.id = '6377'
SELECT t0.id AS id1, t0.libelle AS libelle2 FROM categorie t0
WHERE t0.id = '6401'
SELECT t0.id AS id1, t0.libelle AS libelle2 FROM categorie t0
WHERE t0.id = '6448'
SELECT t0.id AS id1, t0.libelle AS libelle2 FROM categorie t0
WHERE t0.id = '6423'
SELECT t0.id AS id1, t0.libelle AS libelle2 FROM categorie t0
WHERE t0.id = '6425'
SELECT t0.id AS id1, t0.libelle AS libelle2 FROM categorie t0
WHERE t0.id = '6412'
SELECT t0.id AS id1, t0.libelle AS libelle2 FROM categorie t0
WHERE t0.id = '6489'
SELECT t0.id AS id1, t0.libelle AS libelle2 FROM categorie t0
WHERE t0.id = '6542'
SELECT t0.id AS id1, t0.libelle AS libelle2 FROM categorie t0
WHERE t0.id = '6522'
SELECT t0.id AS id1, t0.libelle AS libelle2 FROM categorie t0
WHERE t0.id = '6525'
SELECT t0.id AS id1, t0.libelle AS libelle2 FROM categorie t0
WHERE t0.id = '6699'
SELECT t0.id AS id1, t0.libelle AS libelle2 FROM categorie t0
WHERE t0.id = '6499'

-- requête générée APRES l'ajout du EAGER

```

```

SELECT t0.id AS id1, t0.libelle AS libelle2, t0.id_categorie
AS id_categorie3, t4.id AS id5, t4.libelle AS libelle6 FROM
produit t0 LEFT JOIN categorie t4 ON t0.id_categorie = t4.id

```

Avec cet exemple, on peut penser qu'il vaut mieux systématiquement désactiver le lazy loading. Pas forcément car son comportement peut se révéler utile. En effet, si l'option "EAGER" est ajoutée sur toutes les définitions des relations, des jointures sont alors générées dès que l'on accède à des entités, y compris si on ne le souhaite pas. Par rapport à l'exemple ci-dessus, en supposant que l'on ne veuille pas afficher la catégorie d'un produit, l'ajout de l'option fetch="EAGER" va créer une requête avec jointure vers la catégorie alors que l'on n'a pas besoin de celle-ci. Les conséquences sont un temps de réponse plus long et une allocation mémoire trop importante par rapport au besoin.

Conclusion

Dans la plupart des cas, les problèmes de performance sont liés à la mauvaise utilisation des solutions ORM. Même si un de leurs avantages est de permettre aux développeurs de rester dans le concept objet, il n'empêche que leur utilisation nécessite un minimum de connaissances en SGBDR et donc en SQL. Il convient donc de vérifier les requêtes générées par l'outil



ORM et voir si elles correspondent à celles que l'on écrirait de manière optimisée en SQL.

 Sophie Sound
Ingénieur développement Osaxis

WANTED

Experts Microsoft passionnés et agiles convaincus (H/F) !



- Vous codez depuis au moins 3 ans.
- Spécialisé sur les technologies Microsoft, vous êtes à l'aise avec les technos et les frameworks récents !
- Curieux, vous bloguez et vous faites de la veille techno.
- Vous êtes attaché à la qualité logicielle et l'agilité n'a plus de secret pour vous.
- Vous aimez transmettre vos savoirs et apprendre de vos pairs !

\$ Venez découvrir nos coulisses et rejoignez-nous !
recrutement@cellenza.com

**Retrouvez-nous aux TechDays :
Stand #15**



Cellenza 

156 boulevard Haussmann
75 008 Paris
info@cellenza.com
recrutement@cellenza.com



www.cellenza.com



blog.cellenza.com

Un export Excel générique en Java basé sur les annotations et l'introspection 2^e partie

Dans cet article, nous proposons de traiter de manière générique, à l'aide d'une architecture basée sur les annotations Java, l'export Excel d'une collection d'objets. Cette architecture a été mise au point dans le cadre du développement de Keep Alert, la plateforme de surveillance de marques sur Internet (<http://www.keepalert.fr>).

Récupération des données

Avant de pouvoir récupérer nos données, il faut récupérer la classe les contenant. L'architecture de *FieldDescriptor* utilisée permet de remonter facilement dans l'arborescence, notamment à partir du *FieldDescriptor* depuis lequel nous ferons le *getValue(client)* ;.

La première étape est donc de récupérer l'instance de la classe dans laquelle se trouve la donnée. Dans la classe *FieldDescriptor-Exportable*, nous avons cette méthode :

```
private Object getFrom(Object o) throws IllegalArgumentException,
    IllegalAccessException, InvocationTargetException {
    if (o instanceof Collection) {
        List<Object> retObjects = new LinkedList<Object>();
        for (Object listObject : (Collection) o) {
            retObjects.add(getFrom(listObject));
        }
        return retObjects;
    } else {
        if (getMember().getDeclaringClass().isInstance(o))
            return o;
        FieldDescriptor firstField = this;
        while (firstField.getParent() != null
            && !firstField.getMember().getDeclaringClass()
                .isInstance(o)) {
            firstField = firstField.getParent();
        }
        if (!firstField.getMember().getDeclaringClass().isInstance(o))
            return null;
        return getFrom(firstField.getValueFrom(o));
    }
}
```

Quelques explications s'imposent :

- Nous prenons en paramètre un objet représentant un point d'entrée (ici client, mais peut-être une classe intermédiaire due à l'appel récursif)
- Dans le cas où nous sommes face à une liste, il nous faut retourner celle de tous les objets contenant notre champ,
- Autrement, on vérifie tout d'abord si on n'est pas arrivé à destination; si oui on retourne l'objet contenant la propriété recherchée, sinon on continue,
- On parcourt l'arbre des parents pour retrouver le *FieldDescriptor* permettant lui-même de retrouver la prochaine étape pour accéder à la donnée (par ex : client -> produit -> adresse fournisseur -> ville)
- Si aucun moyen de récupérer l'instance de la classe recherchée ne se présente, on retourne la valeur nulle, sinon on retourne l'instance qui nous intéresse (ici produit puis adresse fournisseur).
- La méthode *getValueFrom* retourne la valeur du champ depuis l'instance spécifiée en paramètre.

Nous allons maintenant étudier la méthode *getValueFrom* de la classe *FieldDescriptor* :

```
public Object getValueFrom(Object oThis) throws IllegalArgumentException,
    IllegalAccessException, InvocationTargetException {
    if (oThis instanceof Collection) {
        List<Object> retList = new LinkedList<Object>();
        for (Object oThisListed : (Collection) oThis) {
            retList.add(getValueFrom(oThisListed));
        }
        return retList;
    }
    Method tmpMethod = null;
    if (member instanceof Field) {
        if (((Field) member).isAccessible())
            return ((Field) member).get(oThis);
        String getterName = "get" + member.getName();
        for (Method m : member.getDeclaringClass().getMethods()) {
            if (m.getName().equalsIgnoreCase(getterName)) {
                tmpMethod = m;
                break;
            }
        }
    } else if (member instanceof Method) {
        tmpMethod = (Method) member;
    }
    if (tmpMethod != null)
        return tmpMethod.invoke(oThis);
    throw new IllegalAccessException(member.getName() + " not found in "
        + oThis.getClass().getName());
}
```

A nouveau, quelques explications :

- Dans le cas d'une liste, nous retournons celle de toutes les valeurs, grâce à un appel récursif. Nous gérons donc le cas des listes imbriquées ;
- Si le membre pointé par le descripteur est un champ public, nous le retournons directement,
- Sinon nous essayons de trouver un getter (en ignorant la casse), en recherchant une méthode nommée *getChamp*, à laquelle nous ferons appel pour récupérer la valeur du champ,
- Si le membre est une méthode, nous l'appelons directement.
- Si jamais nous ne sommes pas arrivés à trouver de getter pour le champ, nous lançons une exception.

Et voilà, nous sommes en mesure de récupérer n'importe quelle donnée annotée dans notre arborescence de classe depuis une instance de *Client* ! En effet, la fonction *getValue* de *FieldDescriptorExportable* se résumera à :

```
public Object getValue(Object from) throws IllegalArgumentException,
    IllegalAccessException, InvocationTargetException {
    from = getFrom(from);
    return getValueFrom(from);
}
```

Enfin, dans notre *FieldDescriptorCollectionExportable*, nous n'aurons plus qu'à appeler *getValue* sur chaque enfant pour peupler le jeu de résultats. Un jeu de résultat provenant d'un *FieldDescriptorCollectionExportable* va être une liste de *FieldCollectionValues* :

com.annotation::FieldCollectionValues
~field: FieldDescriptorExportable
~values: List<Object>
+getField(): FieldDescriptorExportable
+setField(FieldDescriptorExportable field): void
+getValues(): List<Object>
+setValues(List<Object> values): void

A partir de cette classe, nous pouvons retrouver le *FieldDescriptorExportable* responsable de cette entrée, ainsi que la liste de toutes les valeurs associées.

Listage des classes d'entrée

Avant de poursuivre, nous devons permettre d'identifier les classes « racines ». Ce sont les classes qui contiendront directement ou indirectement toutes les données à exporter. Dans notre exemple, une telle classe serait « Client ».

Cette étape va reposer sur le mécanisme d'introspection. Nous devons d'abord récupérer l'objet *java.lang.Class*.

Pour ce faire plusieurs choix s'offrent à nous, mais dans l'ensemble nous pouvons les regrouper dans 2 catégories : soit nous les listons « en dur » dans le code source, soit nous construisons de manière automatique une liste. Nous allons ici étudier les deux cas pour l'exemple, mais dans la suite de l'article seule la seconde méthode sera utilisée.

Listage des points d'entrée « en dur »

Pour cette solution, l'utilisation d'une énumération semble toute indiquée. En plus de permettre un parcours des différentes valeurs possibles, une initialisation simple, d'être accessible facilement, elle offre également de rattacher des informations annexes à ses différentes valeurs. Nous obtiendrions alors, dans le cas de notre exemple :

```
public enum ExportedClasses {
    CLIENT(Client.class);
    private ExportedClasses(Class<?> clazz) {
        this.clazz = clazz;
    }
    private Class<?> clazz;
    public Class<?> getClassType() {
        return clazz;
    }
}
```

Remarque : nous aurions aussi pu directement injecter l'objet « class » dans le module, mais cela impose l'écriture de code d'initialisation.

Nous n'aurions alors plus qu'à parcourir l'énumération comme une liste pour récupérer et traiter toutes les classes, comme ceci :

```
for(ExportedClasses clazz : ExportedClasses.values()) {
    Class<?> clazzToAnalyze = clazz.getClassType();
    //Extract informations...
}
```

Seulement le jour où nous voudrions créer une bibliothèque, nous ne pourrions pas utiliser cette énumération étant donné que notre code ignorerait quelles classes racine devraient être exportées.

Listage dynamique des points d'entrée

L'autre solution consiste à parcourir la liste de toutes les classes disponibles, et de repérer celles qui nous intéressent. Afin de réaliser ce parcours, nous allons ici utiliser Spring 2.5. Néanmoins, si vous ne voulez pas dépendre de Spring(1), d'autres bibliothèques Java assurent le service qui nous intéresse ici, comme la bibliothèque Reflections(2).

La classe Spring permettant de lister les classes d'un package est *ClassPathScanningCandidateComponentProvider*(3). Afin de marquer les points d'entrée, nous allons non pas utiliser une annotation, mais plutôt une interface. Je vous expliquerai ce choix plus tard. Cette interface se nommera *ExportableRootClass*, et sera pour l'instant vide. Nous allons donc avoir :

```
List<Class<?>> knownClasses = new LinkedList<Class<?>>();
ClassPathScanningCandidateComponentProvider scanner =
    new ClassPathScanningCandidateComponentProvider(true);
//On indique les classes qui nous intéressent
scanner.addIncludeFilter(new AssignableTypeFilter(ExportableRootClass.class));
for(BeansDefinition bean : scanner.findCandidateComponents(packageName)) {
    try {
        Class<?> clazz = Class.forName(bean.getBeanClassName());
        //On vérifie ici que la classe implémente directement l'interface (1)
        for(Type type : clazz.getGenericInterfaces()) {
            if(type instanceof Class<?>) {
                if(((Class<?>) type).isAssignableFrom(ExportableRootClass.class)) {
                    knownClasses.add(clazz);
                    break;
                }
            }
        }
    } catch (ClassNotFoundException e) {
        e.printStackTrace();
    }
}
```

Il est nécessaire de faire une post-vérification sur la classe récupérée ; en effet celle-ci peut indirectement implémenter l'interface si elle hérite d'une classe qui l'implémente. Dans ce cas, on pourrait avoir une prolifération inutile des classes racines.

Nous avons maintenant la liste de toutes les classes exportables, nous pouvons donc lister tous les champs marqués.

Avant de continuer...

Nous arrivons à la fin de la première partie de cet article, dont le but est d'obtenir une représentation Excel de nos données. Cependant avant d'aller plus loin, je vous invite à regarder le code source complet(4) qui nous servira de base par la suite, afin de bien comprendre le système.

N'hésitez d'ailleurs pas à modifier le code pour le tester, et en comprendre mieux le fonctionnement.

PERSONNALISATION

Nous sommes maintenant en mesure d'exporter nos données sous format Excel. Cependant, réaliser un export en utilisant directement nos classes *FieldDescriptor* serait dommage ; en effet elles ne permettent pas de gérer facilement l'ordre des champs, la personnalisation de l'affichage, etc.

Aussi, si vous voulez permettre à vos utilisateurs de personnaliser leurs feuilles Excel (colonnes à afficher, ordre, police, labels, etc.), l'introduction de nouvelles classes est nécessaire.

Dans cette partie je vais introduire ces quelques classes qui nous permettront plus tard de personnaliser l'ordre des champs plus facilement, et surtout, de sauvegarder cette personnalisation dans une base de données. La partie persistance de la personnalisation ne sera cependant pas abordée. Je vous conseille donc de vous renseigner sur Hibernate(5), ou tout autre ORM permettant la sauvegarde de collections avec conservation de l'ordre des éléments.

L'interface de personnalisation sera elle aussi oubliée de cette partie, étant donné qu'elle est très dépendante de l'environnement d'exécution. Cependant, pour vous donner une idée de la chose, nous en avons développé une entièrement en HTML 5 + Javascript + JQuery (pour la partie AJAX + stockage de donnée dans un élément DOM). Aussi si vous êtes dans le cas d'un développement d'interface web je vous invite à envisager une interface Drag'n'Drop.

Architecture

Récapitulons nos besoins :

- L'utilisateur doit pouvoir ajouter, supprimer, déplacer les colonnes,
- Il peut paramétrer le nom, certains intitulés (notamment sur les liens, par exemple)
- L'utilisateur pourrait, dans le futur (ce ne sera pas traité dans cet article) paramétrer la police de la colonne, de l'entête, les bordures, etc.

Nous aurons donc une classe *ExcelExportPersonalization*, contenant une liste ordonnée de tous les champs à exporter, avec leur personnalisation.

Notez que la personnalisation des champs ne doit pas être dans la relation « personnalisation – champ » (en d'autres mots, la relation entre la personnalisation et les champs n'est pas porteuse) ; car si l'utilisateur supprimait accidentellement une colonne, il perdrait toutes ses préférences. C'est pourquoi les personnalisations seront stockées à part, dans une autre classe.

Ainsi, l'utilisateur ne personnalisera pas vraiment *une* colonne, mais plutôt *toutes* les colonnes référençant le même *FieldDescriptor*. Sans plus tarder, voici le diagramme de classe correspondant à cette architecture : Fig.1.

Analyse

Analysons le diagramme précédent :

- Une personnalisation concerne donc un point d'entrée, identifié par le champ *className*. Si nous reprenons notre exemple, il sera égal à *com.dataclasses.Client* (soit *Client.class.getCanonicalName()*).
- La classe *ExcelExportPersonalization* possède certaines méthodes qui nous seront utiles par la suite, notamment *getPa-*

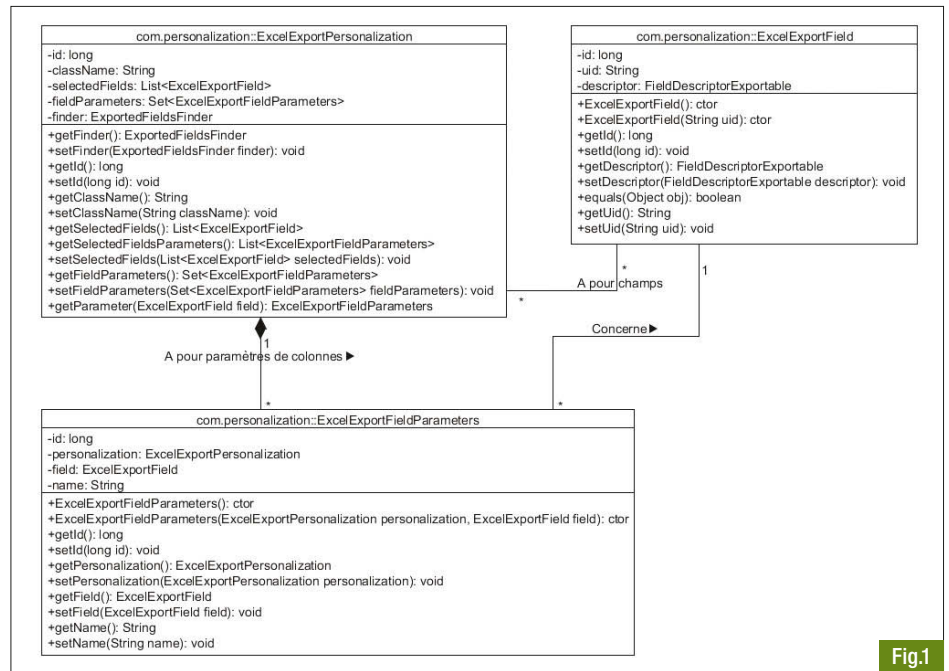


Fig.1

rameter qui retourne la personnalisation d'un *ExcelExportField* donné, et *getSelectedFieldsParameters()* qui nous retournera la liste de tous les champs sélectionnés pour l'export, de manière ordonnée.

- La classe *ExcelExportField* correspond à un *FieldDescriptorExportable*, le lien virtuel sera fait entre les deux classes grâce à leurs deux champs *UID*. La valeur de ce champ est construite à partir du *FieldDescriptor*, et correspond au schéma suivant : *className.notation.pointée.de.tous.les.champs.à.traverser*. Ce qui pourrait donner pour la ville de livraison d'un client : *com.dataclasses.Client.facturationAddress.city*.

De cette manière nous sommes sûrs de pouvoir identifier de manière unique un *FieldDescriptor*.

- La classe *ExcelExportFieldParameters* sera chargée du stockage du nom de colonne à afficher. Ce sera la classe à modifier pour ajouter d'autres paramètres à la personnalisation de nos exports. Ce sera la personnalisation d'un type de colonne.

Si la génération de l'UID peut sembler naturelle, lors d'une première implémentation nous nous sommes contentés d'une UID générée selon le schéma : *classNameCible.nomDuChamp*, ce qui fonctionne très bien lorsque nous n'avons pas deux fois la même classe exportée. Si nous reprenons notre exemple, pour le client nous avons deux adresses, celle de livraison, et celle de facturation. Question : Avec ce schéma d'UID, à quel champ correspond l'UID *Address.city*, l'adresse de livraison ou celle de facturation ? C'est pourquoi un autre schéma a été imaginé.

TODO

Tout de même un petit mot avant de passer à la suite, sur le listage des points d'entrée. Vous vous rappelez certainement que j'ai choisi une interface pour identifier ces classes. En effet tout point d'entrée doit être capable de retourner la liste ordonnée des UID à exporter par défaut. Cela permet d'avoir des configurations par défaut pour tous nos exports à réaliser. Cela nous impose malheureusement aussi d'instancier un objet pour accéder à cette liste, étant donné que Java interdit la définition de méthode statique dans une interface. Nous sommes maintenant prêts à aborder la génération de nos feuilles Excel !

EXPORT EXCEL – APACHE POI

Apache POI, c'est quoi cette bête, ça se mange ?

Apache POI(6) est le projet Apache pour la lecture, l'écriture, et la modification de documents Microsoft Office. Il est divisé en plusieurs modules. Celui qui nous intéresse est le module « ss » (pour *Spread Sheet*). Ce dernier est lui-même divisé en 2 sous-modules, le module HSSF qui gère les .xls, et le module XSSF qui gère lui les .xlsx, celui qui nous intéresse ici. L'utilisation de ces deux modules est similaire, en effet ils sont tous deux basés sur les mêmes interfaces. Seuls quelques paramètres changent (notamment au niveau des unités de mesure).

Lors de l'écriture de ce module d'export, en plus de la contrainte de personnalisation, une faible empreinte mémoire était demandée. Pour parvenir à réduire l'utilisation mémoire au maximum, la version « streaming » du module SXSSF a été utilisée. Cette variante ne conserve qu'un nombre déterminé de lignes en mémoire.

Les différents modules d'Apache POI SS et leurs fonctionnalités

API Type	HSSF (.xls)		XSSF (.xlsx)		SXSSF
	eventmodel	usermodel	eventmodel	usermodel	
	Streaming (ala SAX)	In memory tree (ala DOM)	Streaming (ala SAX)	In memory tree (ala DOM)	Buffered Streaming
CPU and Memory Efficiency	Good	Varies	Good	Varies	Good
Forward Only	Yes	No	Yes	No	Yes
Read Files	Yes	Yes	Yes	Yes	No
Write Files	No	Yes	No	Yes	Yes
Feature:					
create sheets / rows / cells	No	Yes	No	Yes	Yes
styling cells	No	Yes	No	Yes	Yes
delete sheets / rows/cells	No	Yes	No	Yes	No
shift rows	No	Yes	No	Yes	No
cloning sheets	No	Yes	No	Yes	No
formula evaluation	No	Yes	No	Yes	No
cell comments	No	Yes	No	Yes	No
pictures	No	Yes	No	Yes	Yes

Le revers de cette faible empreinte mémoire est l'impossibilité d'accéder aux champs précédents.

Précisions sur l'export

Nous allons par la suite nous efforcer d'exporter une liste de clients. Cet export sera réalisé grâce à la classe *GenericXlsxBuilder*, prenant en paramètre une liste d'objets (ici les clients), et une instance de la classe *ExcelExportPersonalization* valide (c.à.d. contenant une liste de champs, avec leurs personnalisations rattachées). En retour nous aurons un *java.io.File* représentant le fichier généré (ce sera un fichier temporaire). Libre à vous de modifier le code si vous ne voulez pas déléguer la création de ce fichier au builder.

Implémentation du générateur XLSX

Créons notre builder capable de créer une feuille Excel depuis n'importe quelle collection annotée !

Structure générale

Notre générateur sera composé des méthodes suivante :

```
public File create(@SuppressWarnings("rawtypes") Iterable data,
    ExcelExportPersonalization personalization);
protected void addRow(Row row, Object rowElement,
    List<ExcelExportFieldParameters> fields);
protected void processCell(Row row, int cellPosition, Object rowElement,
    ExcelExportFieldParameters field);
protected void setCellValue(Cell cell, Object value,
    ExcelExportFieldParameters field);
```

```
protected void processPicture(Row row, int cellPosition,
    Object rowElement,
    FieldDescriptorExportable descriptor);
protected CellPicture processPictureUnsecured(int row, int cell,
    Object value);
protected String formatCollectionExportableValue(
    FieldDescriptorCollectionExportable descriptor, Object data)
private float rowHeight(Row row);
private float getHeight(Cell cell, int fontHeight);
protected void generateColumnsHeader(Row row,
    List<ExcelExportFieldParameters> fields);
```

Voyons brièvement le rôle de chacune de ces méthodes :

- **create** – La seule méthode publique, chargée d'initialiser les polices, les différentes variables relatives à notre export. Elle sera également chargée d'itérer sur la liste d'objets passés en paramètre pour créer toutes les lignes du document.
- **addRow** – Va itérer sur tous les champs d'une ligne pour créer les cellules et images correspondantes.
- **processCell** – Sera chargée de récupérer la valeur d'une cellule, et, en fonction de son type, d'appeler la bonne méthode de rendu de cellule.
- **setCellValue** – Chargée de rendre une cellule de type texte, date, hyperlien, nombre.
- **processPicture** – Chargée de rendre une cellule type image, en tenant compte d'un nombre maximal d'images. En effet les images sont conservées en mémoire. Afin d'économiser cette dernière nous limitons donc le nombre d'images par document.
- **processPictureUnsecured** – C'est ici que les images seront réellement chargées et ajoutées au document.
- **formatCollectionExportableValue** – Nous extrairons ici les valeurs d'un champ annoté par *@ExportClassCollectionField*. Elle prendra soin de simuler des « sous colonnes », en séparant les champs d'une même ligne par des tirets.
- Les deux méthodes suivantes nous permettront de bien dimensionner chaque ligne en fonction du contenu.
- **generateColumnsHeader** – Première méthode à appeler depuis le *create*, elle va insérer en début de chaque colonne le nom du champ exporté.

Etudions maintenant le code a minima intéressant.

Génération des headers

```
protected void generateColumnsHeader(Row row,
    List<ExcelExportFieldParameters> fields) {
    int i = 0;
    for (ExcelExportFieldParameters descriptor : fields) {
        Cell cell = row.createCell(i++);
        cell.setCellStyle(styleColumnHeader);
        cell.setCellValue(descriptor.getName());
    }
}
```

On parcourt juste la liste de tous les champs à exporter, et on affiche leurs noms, avec un style prédéfini. Si vous voulez permettre la personnalisation des headers de colonne, c'est certainement ici qu'il faudra regarder après avoir modifié la classe *ExcelExportFieldParameters*.

Ajout d'une ligne

```
protected void addRow(Row row, Object rowElement,
    List<ExcelExportFieldParameters> fields) {
```

```

row.setHeightInPoints((short) 10);
int cellsCount = 0;
//On parcourt tous les champs à exporter et on les affiche
for (ExcelExportFieldParameters descriptor : fields) {
    processCell(row, cellsCount++, rowElement, descriptor);
}
row.setHeight((short) -1);
}

```

Rien de bien compliqué, seulement petite subtilité : vous l'avez peut-être remarqué, mais nous fixons d'abord la taille de la ligne à 10 (valeur totalement arbitraire), puis à -1 (taille automatique). Ce petit tour de passe-passe est nécessaire pour bien gérer les images. En effet, insérer une image dans une ligne de hauteur automatique génère une déformation au niveau de la hauteur de celle-ci.

Gestion d'une cellule d'une ligne

```

@SuppressWarnings("rawtypes")
protected void processCell(Row row, int cellPosition, Object rowElement,
    ExcelExportFieldParameters field) {
    if (field.getField().getDescriptor().getType() != ExportField.
        FieldType.IMAGE) {
        Cell cell = row.createCell(cellPosition);
        //Si c'est un champ annoté par @ExportClassCollectionField
        if (field.getField().getDescriptor() instanceof FieldDescriptor
            CollectionExportable) {
            String content = formatCollectionExportableValue(
                (FieldDescriptorCollectionExportable) field.getField()
                    .getDescriptor(), rowElement);
            cell.setCellValue(content);
            CellStyle cellStyle = workbook.createCellStyle();
            cellStyle.setWrapText(true);
            cell.setCellStyle(cellStyle);
        } else {
            try {
                Object value = field.getField().getDescriptor()
                    .getValue(rowElement);
                //Si la valeur récupérée est une liste, il faut formater
                les éléments (1)
                if (value instanceof Collection) {
                    String cellContent = "";
                    for (Object element : (Collection) value) {
                        cellContent += element + "\n";
                    }
                    cell.setCellValue(cellContent);
                    CellStyle cellStyle = workbook.createCellStyle();
                    cellStyle.setWrapText(true);
                    cell.setCellStyle(cellStyle);
                } else {
                    //Sinon c'est sûrement du texte, un nombre, une date, etc...
                    setCellValue(cell, value, field);
                }
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    } else {
        //Le cas des images
        processPicture(row, cellPosition, rowElement, field.getField())
    }
}

```

```

        .getDescriptor());
    }
}

```

Il est à noter que nous ne gérons que les listes sur un niveau. Si jamais vous voulez gérer les listes imbriquées je vous invite à prendre exemple sur le premier code source fourni, dans lequel une fonction récursive extrait les données d'une liste imbriquée et les affiche dans la console.

Gestion des données provenant des annotations ExportClassCollectionField

Voyons comment est géré le formatage des *List<FieldCollectionValues>* présentées plus haut Fig.2.

Il y a autant d'éléments dans la liste que de champs exportables, et les valeurs exportées de la collection sont regroupées. Or, nous voulons les afficher pour pouvoir itérer d'abord sur chaque élément de la collection, puis sur chaque champs de l'instance courante Fig.3.

On remarque que les dimensions des tableaux ont été inversées. Dans les données brutes, la première dimension correspond à une colonne, alors qu'une fois formatée elle correspond à une ligne. Sans plus tarder voici la méthode *formatCollectionExportableValue* qui se charge d'effectuer cette rotation :

```

@SuppressWarnings("rawtypes")
protected String formatCollectionExportableValue(
    FieldDescriptorCollectionExportable descriptor,
    Object data) {
    List<FieldCollectionValues> values;
    List<List<String>> preformatted = new ArrayList<
        List<String>>();
    //On récupère d'abord la valeur du champ
    try {
        values = descriptor.getValue(data);
    } catch (Exception e) {
        e.printStackTrace();
    }
    return «»;
    String retString = "";
    int currentLine = 0;
    boolean first = true;
    for (FieldCollectionValues value : values) {

```

get(0)	Valeur 1-1	Valeur 2-1	Valeur x-1
get(1)	Valeur 1-2	Valeur 2-2	Valeur x-2
get(x)	Valeur 1-*	Valeur 2-*	Valeur x-*

Structure souhaitée pour les listes

Fig.3

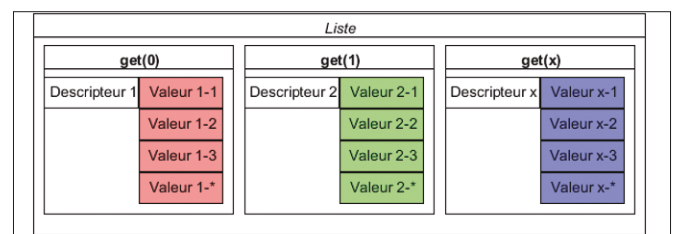


Schéma des *List<FieldCollectionValues>*

Fig.2

```

for (Object colElt : (Collection) value.getValues()) {
    List<String> line;
    if (first) { //Si c'est notre première
        itération, la ligne n'est pas créée
        line = new ArrayList<String>();
        preformatted.add(line);
    } else {
        line = preformatted.get(currentLine);
    }
    //On ajoute à la ligne la valeur du champ
    line.add(colElt.toString());
    ++currentLine;
}
currentLine = 0;
first = false;
}
first = true;
for (List<String> line : preformatted) {
    for (String col : line) {
        if (!first) {
            retString += " - ";
        }
        first = false;
        retString += col;
    }
    retString += "\n";
    first = true;
}
return retString;
}

```

Notons qu'ici, nous ne gérons pas les listes imbriquées. Remarquez qu'il serait simple d'appeler *formatCollectionExportableValue* de manière récursive, afin de bien gérer ces cas-là. Cependant ces données seraient alors difficilement exploitables sous Excel à cause de l'arborescence complexe générée, c'est pourquoi je vous recommande de vous en tenir à cette version.

Génération !

Il ne nous reste maintenant plus qu'à demander à POI de générer la feuille Excel, pour cela rien de plus simple :

```

File xlsxFFile = File.createTempFile("excel_temp", ".xlsx");
// génération des headers et des lignes
try (FileOutputStream out = new FileOutputStream(
    xlsxFFile)) {
    workbook.write(out); } finally {
    ((SXSSFWorkbook) workbook).dispose();
}

```

Nous demandons à notre workbook d'écrire la feuille sur le disque, dans le fichier temporaire créé plus tôt dans la méthode. Cependant, deux subtilités dans cet extrait de code :

- ▀ L'utilisation d'un try-with-resources(7). Une nouveauté de Java 7, fonctionnant sur toutes les classes implémentant l'interface *AutoCloseable*. En fait, dès la fin du bloc *try*, la méthode *.close()* de la variable passée en paramètre (ici *out*) est appelée. Plus besoin du bloc *finally* pour ça.
- ▀ L'appel à *workbook.dispose()*. En effet, c'est l'appel à cette méthode qui va supprimer tous les fichiers temporaires créés par POI.

Et voilà, nous voici maintenant en mesure de :

- ▀ Générer un document XLSX depuis n'importe quelle collection de n'importe quelle classe annotée,
- ▀ Personnaliser les colonnes à afficher,
- ▀ Gérer les date, les textes, les nombres,
- ▀ Gérer les listes d'éléments

Ces fonctionnalités devraient suffire dans la plupart des exports à réaliser.

Généralisation de la génération

La classe de génération proposée présente au moins un défaut : le rendu des listes imbriquées. Il n'a pas été implémenté ici pour la simple et bonne raison que nous n'en avons pas besoin. Il est

cependant simple et possible de rendre de telles données, et nous vous invitons donc une nouvelle fois à vous inspirer du tout premier exemple de code source pour modifier les méthodes de rendu, afin de les rendre récursives.



Ce n'est cependant pas vraiment de la génération XLSX dont je vais vous parler ici, mais plutôt de la possibilité d'implémenter d'autres générateurs. En effet, rien ne nous interdit de rendre nos données sous format de tableau HTML, ou encore de XML, JSON, PDF, etc...

D'ailleurs pour produire un PDF de cette manière assez facilement, rien

ne vous interdit de juste produire votre

tableau HTML et de l'exporter en PDF grâce au logiciel en ligne de commande wkhtmltopdf(8), qui utilise le moteur de rendu WebKit(9), garantissant donc un rendu de qualité.

Enfin, plus généralement, le code produit dans les deux premières parties devrait être facilement transposable dans tout langage proposant des mécanismes de métadonnées et d'introspection, notamment le C#, le PHP, et même en C++ (moyennant quelques modifications) grâce par exemple aux mécanismes d'introspection fournis par Qt.

🔴 Thomas Loubiou,
Etudiant 4ème année à l'EPSE Bordeaux,
Développeur à Systonic pour KeepAlert.
t.loubiou@systonic.fr

🔴 Anthony Don
Directeur Technique Keep Alert / ProDomaines.
a.don@keepalert.com

(1) <http://spring.io/>

(2) <http://code.google.com/p/reflections/>

(3) <http://docs.spring.io/spring/docs/2.5.6/api/org/springframework/context/annotation/ClassPathScanningCandidateComponentProvider.html>

(4) Archive source1.zip

(5) <http://www.hibernate.org/>

(6) <http://poi.apache.org/>

(7) <http://docs.oracle.com/javase/tutorial/essential/exceptions/tryResourceClose.html>

(8) <http://code.google.com/p/wkhtmltopdf/>

(9) Webkit est le moteur de rendu utilisé dans Safari, Chromium, Google Chrome

JSON & jQuery : les incontournables à maîtriser

JSON (JavaScript Object Notation) est un format de données textuel issu initialement de la notation des objets en JavaScript; il est aujourd'hui utilisé dans de très nombreux langages. C'est un concurrent direct d'autres formats de données comme XML ou YAML, dont il se démarque par sa syntaxe peu verbeuse et facile à déchiffrer, et par sa simplicité d'emploi.

Conçu par Douglas Crockford, une des figures les plus emblématiques du langage JavaScript, le JSON est décrit dans la RFC-4627. Il est aujourd'hui un standard bien établi de la communication HTTP.

JSON : JavaScript Object Notation

Extension : .json

Media type : application/json

Avantages : simple, léger, sérialisable, nombreux parseurs disponibles

Structure et syntaxe

Le JSON s'articule autour de deux structures primaires : le tableau représenté par des crochets «[]», et le dictionnaire (aussi appelé objet ou hash) de paires clef/valeur (aussi appelées propriétés) représentées par des accolades «{}». C'est le mélange de ces deux structures qui permet de construire des graphes complexes de données. Le JSON repose en outre sur quatre types primitifs : les chaînes de caractères, les nombres décimaux, les booléens, et la valeur *null*.

```
[
  true,
  { «Nom»: «Crockford», «Prenom»: «Douglas» },
  [0, 1, 1, 2, 3, 5, 8, 13],
  {
    «NombreJoursSansIncident»: 45,
    «ChaineCommandement»: [
      «Bob», «Roger», null
    ]
  }
]
```

Dans l'exemple précédent, le JSON représente un tableau contenant un booléen, un hash contenant deux propriétés, un autre tableau, et à nouveau un hash dont la valeur d'une des propriétés est, là encore, un tableau. En JavaScript, l'accès aux valeurs d'un tableau JSON se fait par le biais d'un indexeur, comme on le ferait dans la plupart des langages informatiques.

```
var isActive = tableauJSON[0]; // Contient «true»
```

L'accès aux valeurs d'un hash se fait par le biais d'un indexeur nommé, ou en JavaScript par la notation pointée.

```
// Indexeur nommé
var prenom = tableauJSON[1][«prenom»]; // Contient «Douglas»

// Notation pointée
var nom = tableauJSON[1].nom; // Contient «Crockford»
```

L'avantage de l'indexeur nommé est qu'il est possible de récupérer dynamiquement une valeur à partir d'une variable qui contient le nom de la propriété.

```
var nomPropriete = «NombreJoursSansIncident»;
var nombreJours = tableauJSON[3][nomPropriete]; // Contient «45»
```

Le tableau et le dictionnaire se parcourent également de manière différente. Pour le premier on utilisera une boucle «for» (ou la nouvelle méthode *forEach* dans la norme ECMAScript 5), et pour le second une boucle «for in».

```
var tableau = [ «Gryffondor», «Serdaille», «Poufsouffle»,
  «Serpentard» ];
for (var i = 0; i < tableau.length; i++)
  var value = tableau[i];

var objet = { «Black»: «Sirius», «Potter»: «James»,
  «Rogue»: «Severus» };
for (x in objet) {
  var nom = x;
  var prenom = objet[x];
}
```

La différence est d'importance, car en apparence il est possible de parcourir un tableau avec une boucle «for in», mais dans ce cas, vous aurez la surprise de découvrir une valeur inattendue : la propriété «length» du tableau. En effet, la boucle *for in* est conçue pour parcourir toutes les propriétés d'un objet JavaScript, et le tableau constitue un objet comme un autre. Un des inconvénients du JSON est le faible nombre de types supportés, et en particulier l'absence de type *date*/*heure*. Ce manque est généralement compensé par l'utilisation, soit du nombre de millisecondes écoulées depuis le 1er janvier 1970 (le temps POSIX), soit plus récemment en utilisant le format défini par la norme ISO-8601, qui permet d'indiquer avec précision une date et une heure (y compris le fuseau horaire) sous forme de chaîne de caractères. En plus d'être facilement lisible pour un être humain, ce format est également supporté par les navigateurs dont le moteur JavaScript respecte la nouvelle norme ECMAScript 5.

Voici un exemple de date au format ISO-8601 : 2011-12-19T15:28:46.493Z (19 décembre 2011 à 15h28).

Pour être bien formé et donc considéré comme valide, le JSON se doit de respecter quelques règles simples. Les noms des propriétés dans un hash doivent être encadrés par des guillemets doubles, de même que les chaînes de caractères. Pas de guillemets simples ou d'absence de guillemets.

```
{
  «Depart»: «Paris»,
  «Arrivee»: «Shanghai»
}
```

Cette règle est pourtant souvent bafouée car il n'est pas nécessaire de la respecter lorsqu'on utilise le JSON dans le cadre du langage JavaScript. En effet, les moteurs JavaScript sont plus tolérants vis-à-vis de la syntaxe JSON, et le code suivant sera parfaitement exécuté par n'importe quel navigateur.

```
{
  Tournoi: 'Roland Garros',
```

```
'Tour': «Quart de finale»
}
```

En outre, le séparateur décimal des nombres est le point, et l'absence de valeur doit être représentée par le mot-clé *null*.

Sérialisation

Le format JSON se prête facilement à la sérialisation et à la désérialisation en chaîne de caractères. Pour ce faire, la norme ECMAScript 5 a introduit, dans le langage JavaScript, l'objet JSON, qui dispose des méthodes «stringify()» (sérialisation) et «parse()» (désérialisation).

```
var objet = { «Joueur»: «Tony Parker», «Stats»: «12-9-7» };
var objetSerialize = JSON.stringify(objet);
var objetDeserialize = JSON.parse(objetSerialize);
```

Les anciennes versions des navigateurs Internet ne disposant pas de cet objet natif, il est recommandé aux développeurs souhaitant également les cibler d'inclure dans leur projet la librairie json2(1). Conçue par Douglas Crockford lui-même, cette librairie ajoute l'objet JSON, et les deux méthodes citées précédemment si tous les objets ne sont pas présents nativement dans le navigateur. Dans le cas contraire, la librairie se comporte intelligemment et laisse la main à l'objet natif qui sera forcément plus performant.

Il est intéressant de noter également que la librairie bien connue jQuery dispose elle aussi d'une méthode de désérialisation «parseJSON()», là aussi intelligente, mais malheureusement pas d'une méthode de sérialisation. Aujourd'hui la plupart des langages informatiques disposent de librairies permettant de sérialiser/désérialiser, et plus généralement, manipuler du JSON. On peut citer JSON.NET en .NET, GSON en Java, ou encore la librairie dédiée intégrée à PHP depuis la version 5.2.0.

La sérialisation de données JSON répond à une contrainte simple : celles-ci doivent être bien formées selon les règles dont nous avons parlé précédemment. Cela implique qu'il est impossible de sérialiser un objet JavaScript tel qu'une date ou une fonction.

```
// Cet objet, purement JavaScript, ne répond pas à
// la norme JSON et ne pourra donc pas être sérialisé
var objet = {
  «Date»: new Date(2013, 07, 14),
  «Callback»: function (data) { console.log(data); }
};
```

jQuery

On ne présente plus jQuery, la librairie JavaScript utilisée par des millions de sites web à travers le monde. Conçus à l'origine pour réduire les frictions dues aux incompatibilités entre navigateurs et fédérer le développement JavaScript, ses nombreux atouts lui ont permis d'éclipser la concurrence et de devenir une référence du web 2.0. Grâce à jQuery, il est possible d'agir sur la hiérarchie d'une page web avec une redoutable efficacité, ou encore d'effectuer des appels Ajax avec simplicité, tout cela dans quelques dizaines de kilo-octets.

Les deux branches

jQuery a connu deux bouleversements ces derniers mois. Tout d'abord, à partir de sa version 1.9, les méthodes jusqu'ici marquées comme obsolètes ont été purement et simplement retirées. C'est le cas notamment de la méthode «browser()» qui permettait jusqu'alors de détecter la nature et la version du navigateur, pratiquement déconseillée aujourd'hui par l'équipe de jQuery, qui recommande plutôt d'utiliser des outils de détection de fonctionnalités comme Modernizr.

Une page de documentation(2) liste les changements effectués et explique comment en tenir compte pour mettre à jour son code. On peut toutefois se retrouver dans une situation où l'on souhaite bénéficier de jQuery post-version 1.9, pour profiter par exemple des correctifs, tout en conservant les méthodes obsolètes, notamment si l'on utilise des plug-ins qui en ont l'usage. Pour cela, l'équipe de jQuery propose un plug-in spécial nommé «jQuery.migrate», qui permet de rétablir les méthodes qui ont été retirées, et d'afficher des messages de log dans la console de votre débogueur préféré lorsque ces dernières sont utilisées, vous permettant le cas échéant de nettoyer votre code. Il est bien entendu possible de désactiver ces messages.

L'autre grand bouleversement survenu à partir de jQuery 1.9, a été la scission de la librairie en deux branches distinctes : 1 et 2. La différence étant que la branche 2 ne supporte plus le navigateur Internet Explorer en version 6, 7 et 8. La contrepartie est que jQuery s'en retrouve allégé des nombreuses lignes de code qui étaient jusque-là nécessaires pour traiter les cas particuliers levés par ce navigateur. La librairie est aussi plus rapide et plus facilement extensible par l'équipe qui en a la charge.

Pour le moment, il est prévu que ces deux branches continuent d'évoluer de concert.

Le chargement

jQuery est une librairie légère et bien conçue, mais cela ne devrait pas faire oublier qu'il existe des bonnes pratiques pour optimiser son chargement.

La première d'entre elles est de charger conditionnellement la version dite « de développement », conçue pour faciliter le débogage, et la version dite « de production », minifiée et non lisible humainement, mais autrement plus légère.

Autre bonne pratique : l'utilisation d'un CDN (*Content Delivery Network*) pour charger le script. Le principe est que jQuery est chargé depuis un serveur distant optimisé pour délivrer du contenu statique. Les avantages sont nombreux : réduction de la bande passante vers votre propre serveur, mise en cache partagée sur la machine du client (si d'autres sites comme le vôtre utilisent le CDN, il y a une petite chance que jQuery soit déjà chargé) et simplification de la montée en version.

Cette fonctionnalité gratuite est proposée par de nombreux acteurs du web : Google, Microsoft et jQuery lui-même qui vous encourage à charger la librairie directement depuis ses serveurs. Les machines sont généralement disséminées de par le monde afin, là encore, de réduire les temps de chargement. Il y a toutefois un inconvénient majeur à ne pas négliger : rien n'indique que le CDN que vous utilisez ne puisse pas connaître une panne – cela s'est déjà vu – rendant impossible le chargement de jQuery. Ce problème pourrait avoir des conséquences désastreuses sur votre site web selon l'utilisation que vous faites de la librairie. Pour vous en protéger, il est conseillé d'utiliser un système de bascule permettant de charger le script hébergé par vos serveurs en cas de défaillance du CDN.

```
<!-- Ce système de bascule chargera jQuery depuis
votre propre serveur si le CDN (ici celui de
Microsoft) venait à tomber -->
<script src=>http://ajax.aspnetcdn.com/ajax/jquery/
jquery-1.8.2.min.js type=>text/javascript</script>
<script>!window.jQuery && document.write('<script type
=>text/javascript src=>scripts/jquery-1.8.2.min.js
</script>')</script>
```

Un autre point important à ne pas négliger est le chargement des plug-ins. Il est courant aujourd'hui lorsqu'on utilise jQuery d'utiliser également un

(1) <https://github.com/douglascrockford/JSON-js/blob/master/json2.js>

(2) <http://jquery.com/upgrade-guide/1.9>

de ses nombreux plug-ins, parfois plus d'une dizaine pour un seul site web. Il existe là aussi de bonnes pratiques permettant de réduire la bande passante et les temps de chargement. Cela tient à trois points : minifier, concaténer et zipper.

Minifier revient à utiliser un outil spécial – il en existe pour tous les langages – qui va retirer du script tous les caractères inutiles comme les retours à la ligne, voire renommer les variables pour leur donner des noms plus succincts.

Concaténer revient à mettre le code des plug-ins bout à bout dans un même fichier, de manière à réduire le nombre de requêtes HTTP nécessaires à leur chargement.

Zipper revient à activer la compression deflate ou gzip sur son serveur web; une optimisation encore aujourd'hui trop souvent négligée.

Les problèmes de conflits

Étonnamment, plusieurs bibliothèques JavaScript ont décidé d'utiliser le symbole \$ comme raccourci pour accéder à leurs fonctionnalités. On peut citer jQuery, MooTools et Prototype. Cela peut poser problème si vous décidez de charger plusieurs d'entre elles en même temps, cas relativement rare, mais qui peut arriver si vous devez utiliser un plug-in uniquement disponible sur l'une ou l'autre de ces bibliothèques.

jQuery propose sa propre solution pour s'en prémunir : la méthode «noConflict()». Celle-ci permet à jQuery d'abandonner l'usage du symbole \$ pour le rendre aux autres bibliothèques. Il faut alors passer par le mot-clé jQuery pour accéder à ses fonctionnalités. Remarque importante : cette méthode doit être appelée entre le chargement du script de jQuery et celui du script de la bibliothèque tierce.

Une solution alternative et pouvant s'avérer suffisante consiste à englober son code faisant appel à jQuery dans une fonction anonyme auto-exécutée. Cette fonction prendra un paramètre qui ne sera autre que l'objet jQuery, le nom du paramètre en lui-même étant le symbole \$.

```
(function ($) {
    $('#foo').css('color', 'red');
})(jQuery);
```

De cette manière, il n'y a pas de risque de conflit car la fonction limite le scope du symbole \$ à sa seule portée. Notez qu'il est tout à fait acceptable d'englober l'intégralité d'un script dans une telle fonction.

Bien utiliser jQuery

Si l'on souhaite utiliser jQuery convenablement, il convient de rappeler quelques-unes des règles sur lesquelles repose la bibliothèque.

En premier lieu, il est important de comprendre comment fonctionne le principe des sélecteurs dans jQuery. Ceux-ci permettent de dresser une liste d'éléments du DOM par le biais de leur id, leur classe CSS, ou encore leur position vis-à-vis de leurs voisins. Un des atouts de jQuery est qu'il est suffisamment intelligent pour laisser la main au navigateur s'il apparaît que celui-ci est capable d'interpréter nativement un sélecteur. C'est le cas des dernières versions de Chrome, Firefox ou Internet Explorer, qui disposent de la nouvelle fonction normalisée «document.querySelector()». Certains sélecteurs que propose jQuery sont néanmoins incompatibles avec cette fonction, mais ces cas particuliers sont suffisamment négligeables pour être ignorés, et seront traités par Sizzle, le moteur interne de parsing du DOM de jQuery.

```
var listeElements = $('#element-id .element-classe');
```

Les éléments du DOM sélectionnés sont regroupés dans un tableau. Pris indépendamment, ce sont effectivement des éléments natifs du DOM, disposant de toutes les propriétés et méthodes disponibles sur le navigateur considéré. Mais le tableau en lui-même est particulier. Il a en effet été

« augmenté » des différentes méthodes de jQuery qui vous permettront d'agir sur les éléments. Et c'est ici que l'on trouve un principe architectural qui a beaucoup contribué au succès de cette bibliothèque en la rendant si souple à utiliser : le chaînage de méthodes.

```
// Il n'y a qu'un seul élément dans le tableau,
// qui est un élément natif du DOM
$('#mon-element')[0].style.color = 'red';
```

Le chaînage de méthodes repose sur le fait que chacune des fonctions de jQuery permettant d'agir sur le tableau d'éléments renvoie ce même tableau d'éléments (ou un sous-ensemble de celui-ci pour les fonctions de filtrage), permettant ainsi d'enchaîner les appels les uns à la suite des autres. Notez néanmoins que certaines rares fonctions renvoyant une valeur peuvent interrompre la chaîne.

```
$('#mon-element')
    .css('color', 'red')
    .width(50)
    .fadeIn()
    .find('.ma-classe')
    .remove();
```

Bien comprendre le principe des sélecteurs et du chaînage de méthodes est primordial pour optimiser son utilisation de jQuery. Par exemple il est vivement déconseillé de rappeler plusieurs fois un sélecteur identique pour effectuer différentes opérations.

```
// Ce code est calamiteux car l'élément est
// re-sélectionné plusieurs fois
$('#mon-element').css('color', 'red');
$('#mon-element').width(50);
$('#mon-element').fadeIn();
```

Par contre il peut être intéressant de placer le tableau d'éléments dans une variable afin d'avoir une sorte de cache permettant d'agir à différents moments sans avoir besoin de répéter la sélection des éléments.

```
var mesElements = $('.ma-classe');

function displayElements() {
    mesElements.show();
}

function animateElements() {
    mesElements.animate({ width: '50px' }, 'fast');
}
```

Les incontournables de jQuery

La puissance du sélecteur de jQuery ne doit pas occulter ses autres API, qui peuvent aussi rendre de grands services au webmaster chevronné.

L'API Ajax

Conçue initialement pour uniformiser l'implémentation de l'objet XMLHttpRequest dans les différents navigateurs, l'API Ajax de jQuery a, depuis, beaucoup évolué. Elle sait aujourd'hui gérer les appels cross-domaine à base de JSONP, la sérialisation des données de formulaires, ou encore la résolution intelligente du type des données renvoyées.

L'API événementielle

L'API événementielle de jQuery ajoute une surcouche à celle du navigateur, permettant non seulement d'uniformiser son fonctionnement, mais aussi de réduire grandement les risques de fuite mémoire. En effet, jQuery est suffisamment intelligent pour désabonner automatiquement vos gestionnaires d'événements lorsque vous supprimez l'élément du DOM auquel ceux-ci

sont branchés. Ce comportement n'est toutefois garanti que si vous passez de manière consistante par jQuery pour vous abonner aux événements et manipuler le DOM. Si vous utilisez le moteur natif du navigateur pour brancher un gestionnaire d'évènement, jQuery sera étranger à ce fait, et ne pourra donc pas nettoyer efficacement derrière vous.

Par ailleurs, l'API événementielle de jQuery permet de contrôler la propagation des événements dans l'arbre du DOM, d'annuler le comportement par défaut des navigateurs (le fait de naviguer vers un lien lorsque l'on clique sur une ancre par exemple) voire de créer vos propres événements arbitraires et de déclencher ceux-ci. Tout cela permet d'avoir un contrôle très fin sur le déroulement des opérations, ce qui s'avère particulièrement utile sur des applications web modernes comme les Single Page Applications.

```
// On intercepte le click sur un lien, on empêche le
// gestionnaire par défaut du navigateur de se déclencher,
// et on annule la propagation aux éléments parents
$(«#monLien»).on(«click», function (e) {
    e.preventDefault();
    e.stopPropagation();
});
```

Autre atout de cette API à ne pas négliger : la délégation d'évènements, qui permet d'utiliser la propagation d'évènements à son avantage pour gérer le déclenchement d'un événement sur plusieurs éléments du DOM avec un seul gestionnaire. C'est très utile dans le cas du rafraîchissement d'une page en Ajax, afin de ne pas avoir à systématiquement réabonner les gestionnaires d'évènements, ou encore pour réduire la taille de l'empreinte mémoire, un seul gestionnaire pouvant gérer de très nombreux éléments du DOM.

```
$(«table»).on(«click», «td», function (e) {
    // Ce gestionnaire sera appelé à chaque click
    // sur une cellule du tableau
});
```

L'API Data

HTML5 a introduit un nouvel attribut «data» permettant de stocker arbitrairement des données sur un nœud du DOM. C'est un ajout très intéressant, car auparavant les développements créaient souvent leurs propres attributs non normalisés. Maintenant ils disposent d'une véritable interface, construite autour de la propriété «dataset» qui permet d'accéder à ces valeurs en JavaScript.

Pourtant les vieux navigateurs, et notamment Internet Explorer jusqu'à la version 11, ne disposent pas de cette propriété «dataset». L'API de jQuery et sa méthode «data» viennent pallier à ce souci en proposant un accès uniformisé et fiable, aux valeurs stockées sur les nœuds du DOM. C'est un outil simple à utiliser et très pratique lorsque l'on a besoin d'associer un élément à des métadonnées non supportées par la norme HTML.

```
$(«#maDiv»).data(«businessId», «92310B92-54A2-4350-
B000-A4F45BDEFB8A»);
```

L'API d'animation

Bien qu'elle ait été rendue quelque peu obsolète par l'arrivée de nouvelles propriétés CSS3, l'API d'animation de jQuery peut s'avérer encore utile lorsqu'il s'agit de travailler avec les plus anciens navigateurs, mais aussi pour la facilité avec laquelle elle permet d'enchaîner de manière fluide différentes opérations sur les éléments du DOM.

Notez que le plug-in jQuery UI, parent de jQuery, permet d'ajouter de nouvelles options d'animation. Le plug-in jQuery Easing permet quant à lui d'utiliser de nouvelles fonctions d'assouplissement.

```
$(«#maDiv»)
```

```
.fadeIn(«fast»)
.animate({ «width» : 50 }, 600)
.delay(300)
.slideUp();
```

L'API Deferred

L'API Deferred, encore assez jeune dans l'écosystème jQuery, est aujourd'hui mal connue, mais elle devient vite indispensable pour quiconque souhaite concevoir des applications web vivantes et dynamiques.

Cette API permet en effet de gérer avec souplesse les appels asynchrones, et ceux-ci ne manquent pas dans les sites modernes. Animations, appels Ajax et navigation dans les *Single Page Applications* sont autant de sources potentielles d'exécution de code décalées dans le temps.

Par exemple, vous pouvez vouloir rafraîchir les données qu'affiche votre page en effectuant un appel Ajax, mais par définition, vous ne savez pas quand celui-ci se terminera. Par ailleurs, une fois les données récupérées, vous souhaitez animer l'élément du DOM qui va les afficher, et finalement avertir l'utilisateur que l'opération est terminée.

Tout cela était déjà possible auparavant, mais nécessitait d'enchevêtrer de nombreux «callbacks» (c'est ainsi que l'on nomme les méthodes chargées de s'exécuter à la suite d'un appel asynchrone). La conséquence était que votre code JavaScript pouvait vite prendre l'allure d'un plat de spaghetti.

L'API Deferred offre des mécanismes élégants pour pallier ces problèmes architecturaux, avec notamment la possibilité d'enchaîner les callbacks avec le chaînage de méthode, ou de rassembler les appels asynchrones pour n'effectuer une opération que lorsqu'ils seront tous réalisés.

```
$.when(appelAsynchrone1, appelSynchrone2).then(function () {
    // Les deux appels asynchrones se sont réalisés
});
```

L'API Deferred gère par ailleurs les notions de réussite, d'échec et de progression des appels asynchrones, et est compatible avec les autres API de jQuery, en particulier l'API Ajax et l'API d'animation.

```
$.ajax({
    «url»: «http://www.mon-backend.fr»,
    «dataType»: 'json',
    «type»: 'GET'
}).done(function (data) {
    // ici vous pourrez par exemple afficher les
    // données «data» renvoyées par l'appel Ajax
});
```

Tout un pan de l'API Deferred est aussi dédié à la création et à la manipulation de vos propres appels asynchrones, ce qui s'avère là encore très utile si vous faites un usage immodéré des fonctions «setTimeout()» et «setInterval()» du langage JavaScript.

```
var dfd = $.Deferred();

dfd.done(function (text) {
    console.log(text);
});

dfd.resolve(«C'est fini»);
```

🔴 François Guillot

Consultant .NET – MCNEXT

Blog : <http://www.mcnextpost.com> – Site Internet : www.mcnext.com

Android Annotations : Mettez votre code Android au régime !

Le développement d'applications Android connaît un franc succès et de plus en plus de développeurs se laissent tenter par les promesses de la plateforme. Ce succès ne saurait cependant masquer un de ses points faibles, à savoir, la quantité astronomique de code d'infrastructure à écrire pour réaliser une application. Non content de réduire la productivité, ce problème induit un code moins lisible et donc moins maintenable. Fort de ce constat, Android Annotations a vu le jour dans le but de permettre au développeur de se concentrer pleinement sur le code métier en générant le code boilerplate pour son usage. Tour d'horizon d'une solution qui promet d'améliorer la qualité des applications Android.

À écouter les développeurs Android, la critique qui revient le plus souvent sur la plateforme concerne les innombrables lignes de code d'infrastructure à écrire lors de la réalisation d'une application. En effet, travailler avec Android nécessite la manipulation de nombreux éléments provenant de ressources XML, contraignant de fait le développeur à abuser d'appels à `findViewById` pour récupérer ses vues en effectuant des casts souvent infondés, ou bien encore à multiplier les appels à `getString` pour ses libellés. Pire encore, le recours à de simples threads pour des tâches asynchrones nécessite la création d'`AsyncTask` au code des plus verbeux. Que dire encore de la gestion des événements pour laquelle la création de classes anonymes est devenue la norme ! Cette liste est loin d'être exhaustive et l'on se retrouve bien souvent avec des classes Activity au code boilerplate monstrueux.

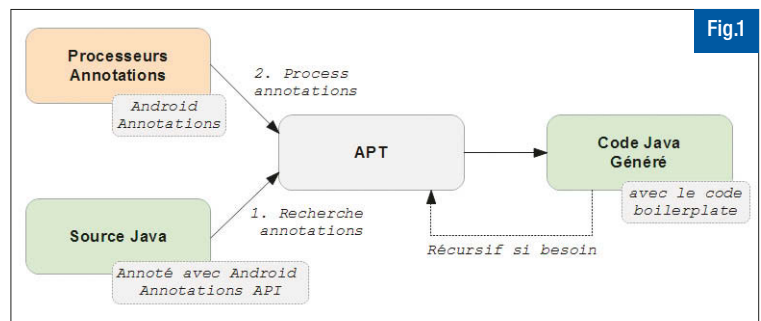
Au final, le développeur Android effectue un nombre incalculable de fois les mêmes opérations, ce qui réduit grandement sa productivité. En sus, le code produit n'est pas toujours très lisible, avec un niveau de verbosité trop élevé. Malheureusement, il en résulte des applications moins maintenables avec un code métier manquant de clarté. Afin de remédier à cette problématique, un développeur français a lancé le projet Android Annotations fin 2010. Si plusieurs projets ont vu le jour dans l'optique d'améliorer la productivité sur la plateforme en proposant différents concepts connus des développeurs Java, Android Annotations se distingue par une approche radicalement différente.

PRINCIPE DE FONCTIONNEMENT

Au lieu de travailler avec la réflexion au runtime, le framework propose une approche basée sur de la génération de code à la compilation. En effet, la réflexion reste un domaine dans lequel Dalvik ne fournit pas de performances suffisantes pour un univers mobile comme Android où gestion mémoire et temps d'exécution sont prépondérants. A contrario, la génération de code à la compilation offre l'avantage de ne pas impacter les performances d'une application au runtime.

Pour ce faire, Android Annotations rajoute une étape à la compilation en s'appuyant sur l'Annotation Processing Tool (APT) de Java 6 [Fig.1]. De fait, la génération de code est transparente pour le développeur puisque réalisée à la compilation.

Techniquement parlant, le framework génère à la compilation une sous-classe de chaque activité annotée. Suffixée par un underscore, celle-ci contient le code d'infrastructure correspondant aux annotations utilisées. Pour une activité `MyActivity`, l'annotation `@EActivity` indique à APT qu'un processeur d'annotations d'Android Annotations est à utiliser :



Fonctionnement d'Android Annotations

```
@EActivity
public class MyActivity extends Activity {
    // ...
}
```

Une sous-classe `MyActivity_` est alors générée dans le même package au sein d'un dossier source différent comme suit :

```
public final class MyActivity_ extends MyActivity {
    // ...
}
```

Contenu au sein de cette classe, le code boilerplate est de fait facilement débuggable. Les comportements générés par le framework fonctionnent principalement en surchargeant des méthodes de l'activité parente telles que `onCreate`. L'activité contenant le code étant `MyActivity_`, il est nécessaire de le préciser au sein du Manifest :

```
<activity android:name=".MyActivity_" />
```

La même précaution est à prendre lors du démarrage d'une activité annotée. Pour les développeurs du monde Java, ce fonctionnement n'est pas sans rappeler celui de Lombok. Si le but visé reste identique, la mise en pratique est cependant différente. En effet, Lombok modifie directement le code source à la compilation en s'appuyant sur des bibliothèques internes au JDK ou au compilateur d'Eclipse, ce qui le rend dépendant de ces derniers. Afin d'être parfaitement portable entre compilateurs, Android Annotations adopte une approche standard avec APT en générant le code puisqu'APT n'autorise pas la modification directe du code à la compilation.

INSTALLATION

Actuellement en version 2.7, Android Annotations se compose de 2 jars récupérables en standalone ou en tant que dépendances Maven. Només androidannotations.jar et androidannotations-api.jar, ils contiennent respectivement les processeurs d'annotations utilisés à la compilation ainsi que l'API au sein de laquelle on retrouve l'ensemble des annotations du framework. Son installation au sein d'un IDE comme Eclipse est relativement aisée. En effet, au sein d'un projet Android, il suffit d'activer APT et d'ajouter le jar androidannotations en tant que factory tout en prenant soin de positionner le compilateur en Java 1.6 [Fig.2]. Enfin, il ne reste plus qu'à ajouter le second jar aux dépendances du projet. Très léger (tout juste 60Ko), ce dernier sera intégré à l'APK de l'application.

EN PRATIQUE

Android Annotations se révèle compétent dans de nombreux domaines. Cela inclut la gestion des vues et ressources, la gestion des événements, les threads, ainsi que l'injection de dépendances sans oublier une gestion des préférences typesafe. Cette liste n'est pas exhaustive et le projet demeure très actif, ce qui laisse présager de nouvelles améliorations dans un futur proche.

Plutôt que de détailler ces domaines au travers d'exemples indépendants, il semble plus pertinent de se baser sur une application réelle déjà publiée sur le Play Store et de montrer avec quelle simplicité le framework permet d'améliorer sa lisibilité en effectuant un refactoring. Pour ce, nous nous appuyons sur l'application FastMeteo développée dans le cadre d'un article de Programmez ! N°145 et disponible ici : <https://play.google.com/store/apps/details?id=com.app.fastmeteo>. En ce sens, nous nous intéressons à l'activité ListCitiesActivity qui affiche une liste de villes et leur météo [Fig.3].

Cette activité présente l'avantage d'utiliser un large panel des fonctionnalités proposées par Android Annotations avec la présence d'une AsyncTask pour le chargement des données météo, la récupération de vues et de ressources, la gestion d'événements sur la liste des villes, la récupération des préférences utilisateur pour la couleur de fond d'écran ou encore l'utilisation de services avec des méthodes statiques. En somme, il s'agit du candidat parfait pour un refactoring à l'aide du framework :



Activité ListCitiesActivity

```
public class ListCitiesActivity extends ListActivity {
    // ...
    ProgressBar progressBar;
    View mainListCities;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.list_cities);
        progressBar = (ProgressBar) findViewById(R.id.progressBar);
        mainListCities = findViewById(R.id.mainListCities);
    }
}
```

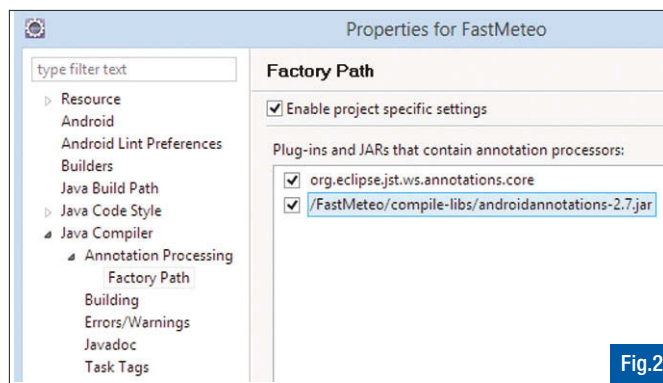


Fig.2

Configuration d'APT avec Android Annotations

```
getListView().setOnItemLongClickListener(new OnItemLongClickListener() {
    @Override
    public boolean onItemLongClick(AdapterView<?> av, View view,
int pos, long id) {
        // ...
        showDialog(DIALOG_REMOVE_CITY_ID, args);
        return false;
    }
});

getListView().setOnItemClickListener(new OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> av, View view, int
pos, long id) {
        Meteo meteo = meteos.get(pos);
        // ... affiche la météo courante
    }
});

@Override
protected void onResume() {
    super.onResume();
    mapColors = Util.getColorsFromPreferences(getApplication
Context());
    mainListCities.setBackgroundColor(Color.parseColor(mapColors
.get(BACKGROUND_COLOR_KEY)));

    if (copyMeteos == null || copyMeteos.size() == 0) {
        loadMeteoForListCities();
    } else {
        hideProgressBar();
        meteos = new ArrayList<Meteo>(copyMeteos);
        getMeteoAdapter().notifyDataSetChanged();
    }
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.list_cities_menu, menu);
    return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
```



```

case R.id.refreshCities:
    loadMeteoForListCities();
    return true;
case R.id.addMenu:
    showDialog(DIALOG_ADD_CITY_ID);
    return true;
default:
    return super.onOptionsItemSelected(item);
}
}

private void loadMeteoForListCities() {
    ArrayList<String> cities = Util.getCitiesInList(getApplicationContext());

    if (cities.size() > 0) {
        if (Util.isLastUpdateOutOfDate(lastUpdate)) {
            copyMeteos.clear();
            new CitiesMeteoLoader(this).execute(cities.toArray(new String[0]));
            lastUpdate = new Date();
        }
    } else {
        hideProgressBar();
    }
}

private void loadMeteoForCity(String city) {
    if (city != null) {
        new CitiesMeteoLoader(this).execute(city);
    }
}

public void showProgressBar() {
    progressBar.setVisibility(View.VISIBLE);
}

public void hideProgressBar() {
    progressBar.setVisibility(View.GONE);
}

public void publishMeteo(Meteo... publishedMeteos) {
    for (Meteo meteo : publishedMeteos) {
        if (meteo.isCityInError()) {
            Toast.makeText(this,
                getResources().getText(R.string.precise_name_city).toString().replace("#city#", meteo.getCity()),
                Toast.LENGTH_SHORT).show();
        } else {
            Util.addCityInList(getApplicationContext(), meteo.getCity());
        }
        copyMeteos.add(meteo);
    }
}

meteos = new ArrayList<Meteo>(copyMeteos);
getMeteoAdapter().notifyDataSetChanged();
}

private void removeCityInList(String city, int position) {

```

```

Util.removeCityInList(getApplicationContext(), city);
// ...
}

@Override
protected void onRestoreInstanceState(Bundle state) {
    super.onRestoreInstanceState(state);
    copyMeteos = (ArrayList<Meteo>) state.getSerializable(LIST_CITIES_PARAM);
}

@Override
protected void onSaveInstanceState(Bundle outState) {
    outState.putSerializable(LIST_CITIES_PARAM, copyMeteos);
    super.onSaveInstanceState(outState);
}

// ...
}

```

COMPOSANTS

La première étape consiste à préciser que l'on souhaite que notre activité soit reconnue par Android Annotations et que ce dernier génère son code boilerplate. A ces fins, on l'annote avec `@EActivity` en précisant en entrée l'id de son content view afin qu'il soit appliqué dans la méthode `onCreate` de la classe fille générée. On remarque ensuite que l'activité utilise des méthodes statiques de la classe `Util` pour récupérer la liste des villes ou sauvegarder une ville nouvellement ajoutée. Il serait ainsi plus propre d'avoir recours à des services pour réaliser ces opérations. L'avantage du framework dans ce cas va être de proposer l'annotation `@EBean` qui, une fois appliquée sur une classe, permettra d'injecter celle-ci de manière transparente au sein d'une classe annotée. Nous verrons plus loin comment mettre en place cette injection de dépendances. Pour l'heure, nous créons donc les services remplaçant les appels aux méthodes statiques de la classe `Util` sur le même plan que celui-ci :

```

@EBean
public class CitiesService {
    @RootContext
    Context context;

    public ArrayList<String> findAll() {
        // ...
    }

    public add(String city) {
        // ...
    }
}

```

Outre ces 2 annotations, le framework en met également à disposition plusieurs autres, rendant possible la génération de code pour divers composants du SDK Android, parmi lesquels on retrouve `@EApplication`, `@EFragment`, `@EProvider`, `@EReceiver` ou encore `@EService` dont les noms indiquent clairement leur rôle.

INJECTION

Un composant pris en charge par Android Annotations bénéficie de fait de l'injection de dépendances. Ainsi, nos services créés précédemment et annotés avec `@EBean` pourront être injectés entre eux ou bien au sein de

l'activité à l'aide de l'annotation `@Bean` au niveau de la propriété, à la contrainte près que cette dernière ne devra pas être privée. En outre, le framework permet également l'injection de vues via l'annotation `@ViewById` positionnée sur un champ d'une activité. Basé sur le principe de convention over configuration, il n'impose de préciser l'id de la vue correspondante côté XML que si celle-ci a un nom différent de la propriété du champ. On notera avec bonheur la possibilité d'accéder au contexte d'une application directement au sein de classes améliorées avec Android Annotations via `@RootContext`, ce qui est mis à profit au sein de nos services. Enfin, l'injection de ressources est accessible sur le même principe via des annotations de type `@StringRes`, `@StringArrayRes`, etc ... Tout ceci appliqué à nos besoins donne un code désormais à l'allure suivante :

```
@EActivity(R.layout.list_cities)
public class ListCitiesActivity extends ListActivity {
    @ViewById
    View mainListCities;
    @ViewById
    ProgressBar progressBar;
    @StringArrayRes
    String[] worldCities;
    @StringRes
    String add, cancel, remove;
    @Bean
    CitiesService citiesService;
    @Bean
    CacheService cacheService;
    @Bean
    MeteoService meteoService;
    @Bean
    LocationService locationService;
    // ...
}
```

Ce simple code annoté apporte déjà une grande simplicité au développeur puisqu'il permet de dire adieu aux appels à `findViewById` ou à `getString`. Un gain de temps et de lisibilité plus qu'important déjà.

GESTION DES ÉVÈNEMENTS

Autre contrainte, la gestion des événements des vues condamne souvent le développeur à multiplier la création de classes anonymes de type listener, ce qui est peu lisible. Pour éviter cet écueil, Android Annotations fournit des annotations permettant le binding entre une méthode d'une activité et une vue du layout associé. Là encore, le principe de convention over configuration s'applique et les annotations proposées sont nombreuses allant de `@Click` pour le click sur une vue à `@ItemClick` et `@ItemLongClick` qui vont nous intéresser ici. Ces dernières permettent en effet de réagir aux événements de click et de click long sur un `AdapterView` tel qu'utilisé par la liste de villes. Au niveau de notre activité, ces annotations signifient l'adieu aux classes anonymes implémentant les interfaces `OnItemClickListener` et `OnItemLongClickListener` :

```
@ItemLongClick(android.R.id.list)
public void listItemLongClick(int position) {
    // ...
    showDialog(DIALOG_REMOVE_CITY_ID, args);
}

@ItemClick(android.R.id.list)
public void listItemClick(Meteo meteo) {
```

```
// ... affiche la météo courante
}
```

THREADS

Au niveau des threads, le framework propose les annotations `@Background` et `@UiThread` qui vont permettre de réaliser des opérations en tâches de fond sans avoir recours à la classe `AsyncTask`. Une plus grande simplicité qui amène également une meilleure compréhension du code. Ainsi, la méthode devant s'exécuter dans un thread différent est annotée avec `@Background`, et, les méthodes appelées en son sein devant s'exécuter au sein du thread principal sont annotées avec `@UiThread`. Pour notre activité, cela revient à supprimer l'`AsyncTask` `CitiesMeteoLoader` et à rapatrier le code de chargement de la météo des villes au sein d'une méthode `loadMeteoForCity` annotée avec `@Background`. La méthode `publishMeteo`, appelée par cette dernière, s'est, quant à elle, annotée avec `@UiThread` puisqu'elle agit sur les éléments visuels de l'activité et doit donc être exécutée dans le thread principal :

```
@UiThread
public void showProgressBar() {
    // ...
}

@UiThread
public void hideProgressBar() {
    // ...
}

@Background
public void loadMeteoForCity(String... params) {
    showProgressBar();
    // ... load
    hideProgressBar();
}

@UiThread
public void publishMeteo(Meteo... meteos) {
    // ... maj liste
}
```

CONCLUSION

Android Annotations est déjà une solution relativement mature dans un univers des applications Android où tout bouge très rapidement.

Utilisé par de nombreuses applications publiées sur le Play Store, il ne cesse de faire ses preuves et devient rapidement indispensable à tout développeur l'ayant essayé.

Outre ses qualités initiales permettant une réduction importante des lignes de code (près de 40 % sur notre exemple) accompagnée d'une plus grande lisibilité et d'une maintenance simplifiée, il offre avant tout au développeur Android des possibilités intéressantes en termes d'architecture de code pour pallier les manquements du SDK.

Le tout sans impact sur les performances puisque tout est généré à la compilation.

Les esprits chagrins pourront lui reprocher une solution reposant sur de la génération de code mais en considérant les limitations de l'architecture d'Android et les gains de productivité apportés par Android Annotations, le choix sera vite fait !

 Sylvain Saurel – Ingénieur d'Etudes Java / JEE
sylvain.saurel@gmail.com

La solution Power BI de Microsoft

2^e partie

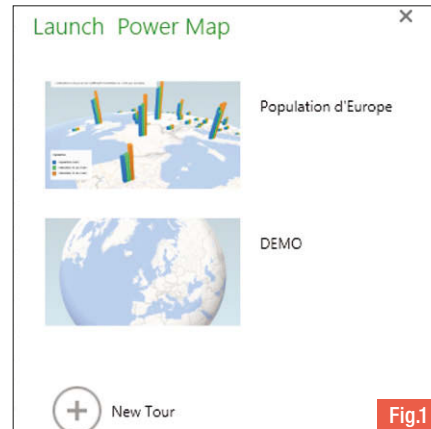
Le volume des données ne cessant d'augmenter, la Business Intelligence est devenue essentielle à la prise de décision en entreprise. L'an dernier, Microsoft commercialisait une nouvelle version de sa solution de Business Intelligence d'entreprise à travers SQL Server 2012. Power BI pour Office 365 vient compléter cette offre avec un ensemble de nouvelles fonctionnalités et services qui permettent de visualiser les données et de les partager de façon intuitive à travers des outils familiers comme Excel et SharePoint. Poursuivons notre exploration dans cette 2^e partie.

POWER MAP

Power Map permet d'afficher des données géographiques en 3D sur un globe terrestre directement depuis Excel. Chaque rapport peut contenir une ou plusieurs scènes. Une scène peut être vue comme une diapositive dans PowerPoint. L'outil a été pensé sous forme d'une cinématique parcourant les scènes les unes après les autres comme une vidéo. En cliquant sur le bouton de lancement, l'animation démarre en plein écran. La caméra reste une durée paramétrable sur chaque scène et passe de l'une à l'autre de manière dégradée et fluide, comme s'il s'agissait d'une navigation manuelle sur une carte Bing. Il est également possible d'animer les scènes en intégrant une donnée temporelle. Une fois la conception terminée, il est possible de générer une vidéo de la cinématique au format MP4 (en basse qualité 360p pour Mobile, en bonne qualité 720p pour ordinateur et tablette et en haute définition 1080p). Comme tous les outils de reporting, Power Map se lance depuis l'onglet « INSERTION ». Contrairement à Power View, les rapports Power Map ne sont pas stockés dans des feuilles Excel. Un premier écran permet donc de sélectionner une cinématique existante ou d'en créer une **Fig.1**. L'écran de conception comporte 4 éléments **Fig.2** :

- ▶ [1] La barre d'outils en haut ;
- ▶ [2] La liste des scènes à gauche ;
- ▶ [3] La fenêtre de paramétrage à droite ;
- ▶ [4] La scène au centre.

Power Map utilise le modèle de données Power Pivot comme source de données. La création



d'une scène commence par la sélection des données géographiques (latitude, longitude, pays, ville, etc.). Lorsque cela est possible, il est recommandé de saisir plusieurs types différents de données pour aider Power Map à trouver la bonne localisation. En effet, ajouter uniquement la ville peut potentiellement causer des erreurs de localisation car certains noms de ville sont utilisés dans plusieurs pays différents.

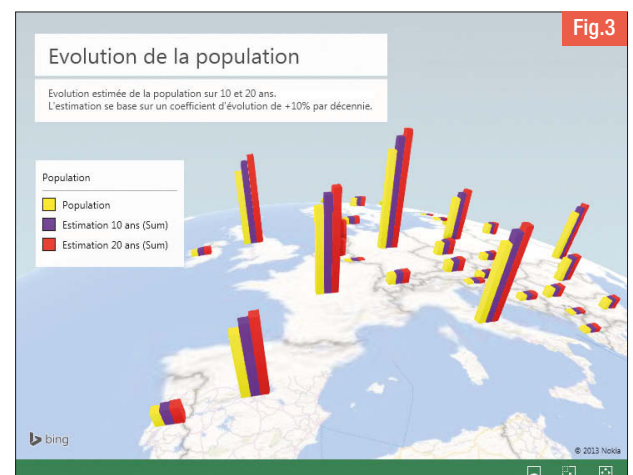
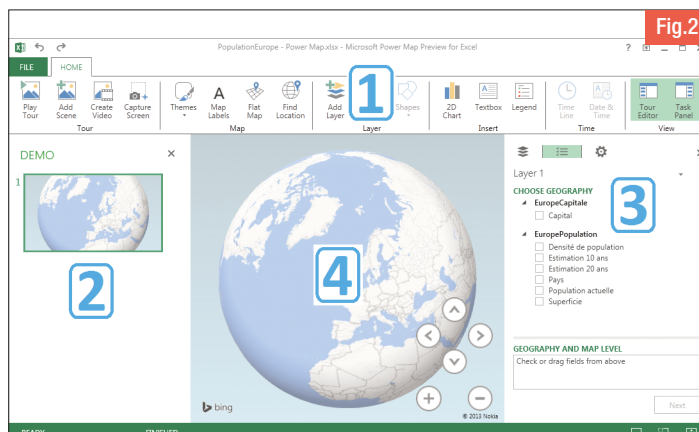
L'étape suivante consiste à créer une ou plusieurs couches de représentation graphique. En effet, il est possible d'afficher dans une même scène plusieurs graphiques différents avec des positions différentes. Pour créer un graphique, il suffit de choisir le type de représentation graphique et la donnée sur laquelle faire la variation. La taille et la couleur des éléments graphiques sont personnalisables depuis la fenêtre de paramétrage. Des zones de texte et des graphiques en 2D peuvent être ajoutés à la scène depuis la barre d'outils. Il existe quatre

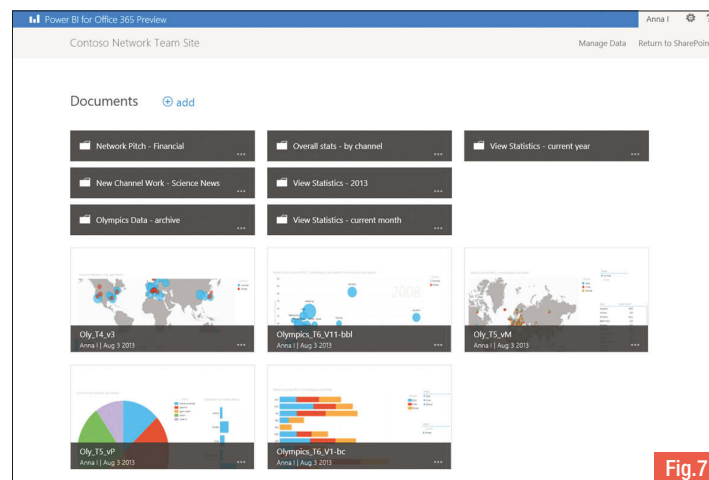
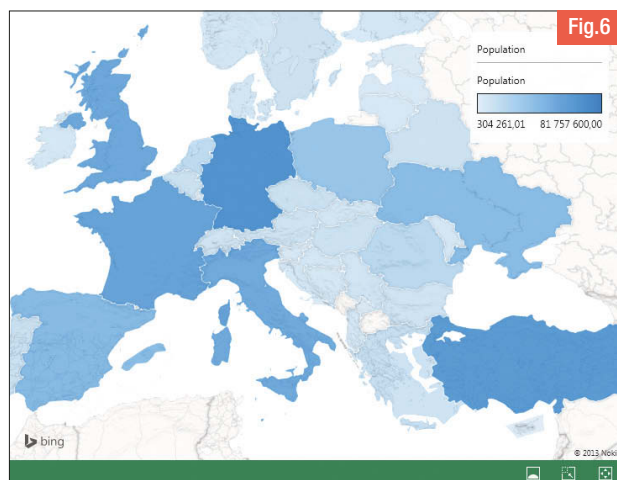
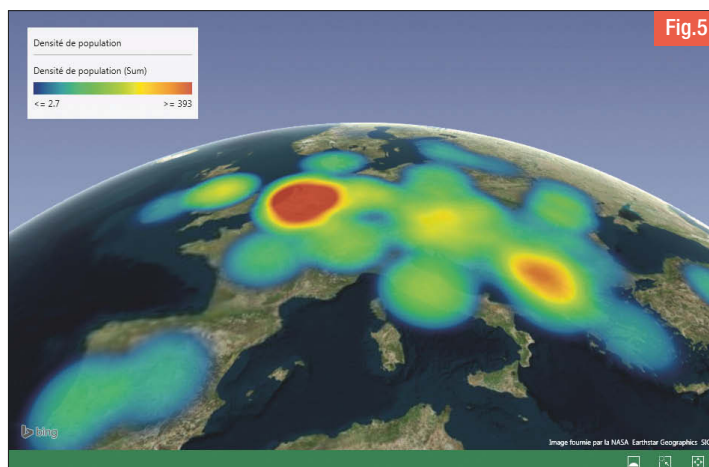
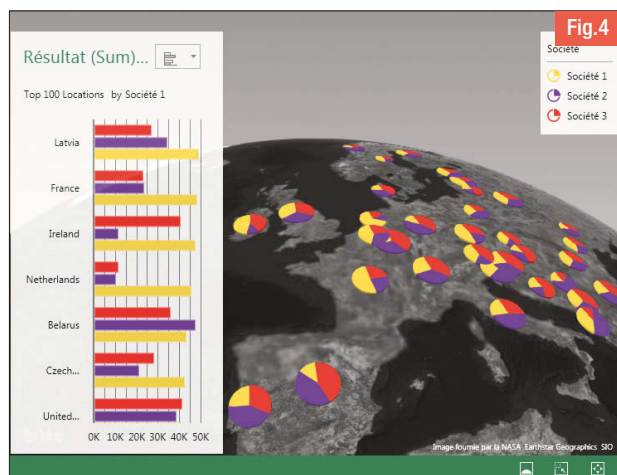
types de représentations graphiques : Colonne **Fig.3**, Bulle **Fig.4**, Zone de chaleur **Fig.5**, Région **Fig.6**. Malgré une option pour définir la finesse d'affichage, Power Map demande beaucoup de ressources. Il est préférable d'avoir Excel en version 64 bits même s'il fonctionne en version 32 bits. Une carte graphique de bonne qualité permettra d'obtenir un affichage plus fin et des animations plus fluides. Enfin, une bonne connexion Internet sera nécessaire pour un chargement fluide des cartes lors du zoom sur une zone.

La seule contrainte regrettable est la même que pour les cartes dans Power View, elle réside en la connectivité à Bing. En effet, sans connexion Internet, le lancement de Power Map n'est pas possible. Néanmoins, si la perte de connexion se produit une fois Power Map démarré (en pleine présentation par exemple), l'absence de connectivité à Bing ne causera pas d'erreur. Les cartes utilisées depuis le lancement restent accessibles dans le cache, mais le zoom sur une zone ne déclenchera plus le chargement d'une carte plus fine.

POWER BI POUR OFFICE 365

Comme vu précédemment, les outils Power BI dans Excel permettent de créer des présentations interactives. Cependant, leur utilité est limitée si elles ne sont pas partagées. Pour répondre à ce besoin, la solution intègre Power BI Sites. C'est une fonctionnalité accessible à travers Office 365 et portée par SharePoint Online. Elle offre la possibilité de créer un site Web





d'entreprise spécialisé dans le partage et la visualisation de rapports de Business Intelligence. Le site Power BI est accessible directement depuis le menu du portail d'Office 365.

Les classeurs Excel chargés sur le site sont consultables directement depuis le navigateur à l'aide d'Excel Web App (la version online d'Excel) Fig.7. Une petite limitation tout de même, les cinématiques Power Map ne sont pas consultables depuis le navigateur. Un des avantages d'utiliser un site Power BI est la possibilité de charger et consulter en ligne des classeurs faisant jusqu'à 250 Mb, ce qui est déjà volumineux pour ce type de fichier. <http://go.microsoft.com/fwlink/?LinkID=317723&clcid=0x409>

Lorsqu'un classeur Excel est publié, toutes les feuilles sont consultables. Mais dans certains cas, il peut être intéressant d'en masquer certaines. Pour cela, il suffit d'aller dans l'onglet Fichier d'Excel et de sélectionner « Options d'affichage du navigateur ». Il sera alors possible de masquer des feuilles pour les consultations en ligne. Tous les utilisateurs ayant les droits d'accès au site pourront voir l'ensemble des classeurs publiés, sauf si des autorisations sont définies sur un classeur ou un dossier particulier. Une requête créée dans Power Query peut également être partagée sur ce site. Celui-ci dispose d'une interface appelée Data Management pour gérer les requêtes, réaliser des statistiques d'utilisation et sécuriser les sources de données.

Une autre fonctionnalité intéressante est le requêteur intelligent Power Q&A. Des requêtes peuvent être lancées sur une source de données, de manière littérale sous forme de question. Des rapports sous forme de tableaux ou de graphiques sont générés à la volée. Prenons, par exemple, une source contenant des ventes par pays et par société. On pose la question suivante : "Which Brand has the highest Sales by Country". L'outil générera alors automatiquement un graphique à barre représentant les ventes par pays et par société trié par ordre décroissant des ventes.

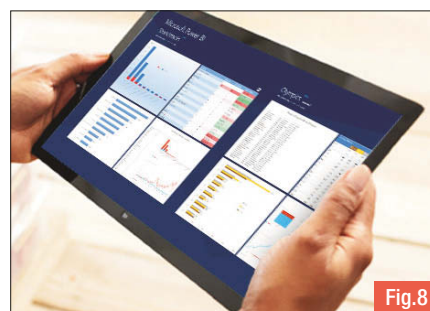
POWER BI APP

Microsoft Power BI App est une application Windows adaptée aux supports mobiles donnant accès aux informations stockées sur le site Power BI d'Office 365. Son interface de nouvelle génération permet de visualiser des rapports en plein écran et avec la même interactivité que dans Excel. Cette application fonctionne sur toutes les versions de Windows 8 et Windows RT pour PC et tablette Fig.8.

<http://office.microsoft.com/en-us/excel/redirect/HA104103581.aspx>

CONCLUSION

La solution Power BI offre un ensemble d'outils et de services qui rendent la BI intuitive et collaborative. C'est l'association bien pensée d'Excel, un outil reconnu et familier, avec la puissance de la BI Microsoft et l'expérience col-



laborative d'Office 365. On retiendra notamment la création d'animation décisionnelle en 3D dans Power Map qui offre une expérience immersive unique.

Il faudra voir sur le long terme l'utilisation qui en sera faite en parallèle des outils de Business Intelligence d'entreprise fournis par la solution SQL Server. En effet, rendre la BI self-service à ce point accessible à tous pourrait engendrer une prolifération de rapports et de sources de données auparavant centralisés dans les outils d'SQL Server.

Cette solution est actuellement en version Preview, d'autres outils pourront donc encore voir le jour d'ici la version finale prévue durant l'année 2014. Le site de présentation de la solution Power BI est accessible à cette adresse : <http://powerbi.com>.

 **Adrien GUET** - Consultant, Sogeti FRANCE
adrien.guet@sogeti.com
<http://www.azone.fr>

Les Apps SharePoint 2013 : Authentification et autorisation

2^e partie

Désormais à sa 5^{ème} version, SharePoint est depuis longtemps une plateforme de développement polyvalente pour la création d'applications. Elle a fréquemment vu évoluer les modèles de développement proposés : fichier WebPart, solutions de ferme, solutions Sandbox. Microsoft apporte désormais une nouvelle solution aux problématiques de déploiement de code pour SharePoint avec les Apps.

La gestion des jetons, cas pratique

Nous avons vu précédemment que l'authentification des Apps auprès de SharePoint se faisait par le biais du protocole OAuth. La cinématique est la suivante (Fig. 3.) :

- L'utilisateur accède par URL à une page SharePoint où est installée l'application MonApp.com.
- SharePoint détecte l'application MonApp.com dans la page. Il demande au service ACS de créer et signer un Context Token contenant des informations contextuelles (utilisateur actuel, site web en cours, ...).
- ACS renvoie le Context Token à SharePoint. Il est signé par le biais d'un secret partagé entre l'application et ACS (client_secret dans le web.config de l'application). Ce Context Token contient un Refresh Token.
- SharePoint effectue un rendu d'une page incluant une IFrame vers l'application dans laquelle il transmet le Context Token.
- L'IFrame amène le navigateur à demander le rendu de la page applicative. Le Context Token est inclus dans cette requête.
- L'application dispose du Context Token. Elle valide la signature du token à l'aide du secret partagé avec ACS, pour s'assurer que la demande émane bien du serveur SharePoint.
- L'application extrait le Refresh Token du Context Token et le communique à ACS pour obtenir un Access Token.
- ACS renvoie un Access Token à l'application qui peut le mettre en cache (l'opération de demande à ACS d'un Access Token est coûteuse).
- L'application peut ensuite exploiter cet Access Token pour effectuer des appels REST/Web services ou C# Client Object Model à des ressources SharePoint.
- SharePoint répond avec les informations demandées.
- L'application effectue le rendu de la page (dans l'IFrame).

Concrètement, comment cela se traduit-il dans le code ?

Lorsque vous créez une App, Visual Studio génère automatiquement dans le projet la classe TokenHelper.cs qui est une classe utilitaire pour manipuler les jetons OAuth. Par défaut il nous génère également une page Default.aspx comportant le code suivant :

```
var contextToken = TokenHelper.GetContextTokenFromRequest(
    Page.Request);
var hostWeb = Page.Request[«SPHostUrl»];

using (var clientContext = TokenHelper.GetClientContext
    WithContextToken(hostWeb, contextToken, Request.Url.
    Authority))
{
    clientContext.Load(clientContext.Web, web => web.Title);
    clientContext.ExecuteQuery();
}
```

La 1^{ère} ligne commande de récupérer le Context Token (étape 6 de la cinématique d'authentification). Si vous regardez le détail de la fonction `GetContextTokenFromRequest` dans le code du helper, vous constaterez qu'elle n'effectue aucun appel :

```
public static string GetContextTokenFromRequest (Http
Request request)
{
    string[] paramNames = { «AppContext», «AppContextToken»,
«AccessToken», «SPAppToken» };
    foreach (string paramName in paramNames)
    {
        if (!string.IsNullOrEmpty(request.Form[paramName])) return
request.Form[paramName];
        if (!string.IsNullOrEmpty(request.QueryString[paramName])) return
request.QueryString[paramName];
    }
    return null;
}
```

Le code se contente de rechercher le Context Token dans les variables de POST et de la QueryString. Lorsque vous cliquez sur une App dans SharePoint, celui-ci interroge ACS (étape 2) et POST le paramètre `SPAppToken` à l'IFrame hébergeant l'APP (étape 4). La méthode `GetClientContextWithContextToken` sert ensuite à interroger ACS pour récupérer un Access Token et générer un contexte client pour interroger SharePoint. Cette méthode cache cependant quelques étapes. On comprend mieux le traitement effectué en le décomposant ainsi :

```
var appWeb = new Uri(Request.QueryString[«SPAppWebUrl»]);
var hostWeb = new Uri(Request.QueryString[«SPHostUrl»]);
// Récupérer le Context Token de la requête
var contextToken = TokenHelper.GetContextTokenFromRequest(
    Page.Request);
// Valider que le token provient bien de SharePoint
SharePointContextToken tokenContent = TokenHelper.ReadAnd
ValidateContextToken(contextToken, Request.Url.Authority);
// Récupérer le Access token de l'utilisateur
var userAccessToken = TokenHelper.GetAccessToken(token
Content.RefreshToken, tokenContent.TargetPrincipalName,
hostWeb.Authority, tokenContent.Realm);
// Créer un contexte SharePoint à partir du Access token
de l'utilisateur
var clientContext = TokenHelper.GetClientContextWithAccess
Token(appWeb.ToString(), userAccessToken.AccessToken)
```

Si vous préférez faire des appels aux API REST au lieu d'utiliser le C# Object Model (CSOM), il est nécessaire d'ajouter le jeton d'accès aux en-têtes d'authentification.

Exemple :

```
HttpWebRequest endpointRequest = (HttpWebRequest)Http
WebRequest.Create(requestUri);
endpointRequest.Method = «GET»;
```

```
endpointRequest.Accept = «application/json;odata=verbose»;
endpointRequest.Headers.Add(«Authorization», «Bearer « + user
AccessToken.AccessToken);
HttpWebResponse endpointResponse = (HttpWebResponse)
endpointRequest.GetResponse();
```

Dans cet exemple, requestUri correspond à l'endpoint REST que vous interrogez.

Élévation de privilège

L'Access Token de l'utilisateur permet de créer un contexte client qui agira sur SharePoint avec les droits de l'utilisateur courant. Quelques étapes supplémentaires sont nécessaires pour créer un contexte qui exécutera les requêtes avec les permissions de l'App. Tout d'abord, il faut positionner l'attribut AllowAppOnlyPolicy à « True » dans le AppManifest.xml ce qui correspond dans Visual Studio à l'option « Allow the App to make App-only calls to SharePoint » (lorsque vous déployez l'application, vous constaterez désormais qu'un nouveau message apparaît sur l'écran de confirmation : « L'autoriser à partager ses autorisations avec d'autres utilisateurs. »). Ensuite il faut récupérer non plus l'Access Token de l'utilisateur, mais celui de l'application. Heureusement la classe TokenHelper contient pour cela la méthode GetAppOnlyAccessToken.

```
var appOnlyAccessToken = TokenHelper.GetAppOnlyAccessToken(
tokenContent.TargetPrincipalName, appWeb.Authority,
tokenContent.Realm);
```

Vous pouvez ensuite créer un contexte client ou réaliser un appel REST avec jeton :

```
var clientContext = TokenHelper.GetClientContextWith
AccessToken(appWeb.ToString(), appOnlyAccessToken.
AccessToken);
```

Performance et mise en cache

La méthode TokenHelper.GetAccessToken communique avec ACS ce qui peut être coûteux si les appels sont fréquents. Aucun mécanisme de cache n'est prévu par défaut et il est du ressort de l'application de mettre en cache les différents jetons afin d'éviter ces aller-retour superflus. Le paramètre « CacheKey » du ContextToken, unique pour la combinaison : **identifiant utilisateur / identifiant du provider d'identité / identifiant de l'application / royaume** (le royaume est l'équivalent de la WebApplication traditionnelle SharePoint), facilite la mise en œuvre d'un cache. Les étapes sont alors :

- ▶ A la 1ère requête récupérer le Context Token,
- ▶ Récupérer l'Access Token de l'utilisateur,
- ▶ Créer un cache serveur avec pour clé CacheKey et y stocker l'Access Token,
- ▶ Renvoyer un cookie avec un nom connu au navigateur comportant ce CacheKey,
- ▶ Pour une nouvelle requête du navigateur, lire le cookie et récupérer la valeur de CacheKey,
- ▶ Charger l'Access token du cache.

Exemple (en utilisant la session ASP.Net) :

```
var contextToken = TokenHelper.GetContextTokenFromRequest
(Page.Request);
if (contextToken != null) // on est sur un postback
{
    var appWeb = new Uri(Request.QueryString[«SPAppWebUrl»]);
    var hostWeb = new Uri(Request.QueryString[«SPHostUrl»]);
    SharePointContextToken tokenContent = TokenHelper.
ReadAndValidateContextToken(contextToken, Request.Url.
Authority);
```

```
var cacheKey = tokenContent.CacheKey;
// Recuperer le Access token de l'utilisateur
var userAccessToken = TokenHelper.GetAccessToken(token
Content.RefreshToken, tokenContent.TargetPrincipalName,
hostWeb.Authority, tokenContent.Realm);
Session.Add(cacheKey + «_userAccessToken», userAccess
Token.AccessToken);
Response.Cookies.Add(new HttpCookie(«CacheKey», cacheKey));
}
else
{
    var contextCookie = Request.Cookies[«CacheKey»];
    var cacheKey = contextCookie.Value;
    var userAccessToken = Session[cacheKey + «_userAccessToken»];
}
```

C'est mieux mais ce n'est pas encore idéal. Plusieurs problèmes se posent encore :

- ▶ S'il s'agit d'une nouvelle session, le cache est perdu alors que le cookie était peut-être encore valide.
- ▶ La durée de vie de l'Access Token est d'environ 12h. Si la session utilisateur dure plus longtemps, le jeton ne sera plus valide.
- ▶ Si l'utilisateur a supprimé le cookie, l'application plante lors de l'accès.
- ▶ L'Access Token n'est pas encrypté. Cela peut potentiellement constituer une faille de sécurité s'il est possible pour un tiers de consulter les données de session.

Les deux premiers points peuvent être traités en exploitant une autre implémentation de cache mieux adaptée que la session, par exemple le Cache ASP.Net. L'Access Token et le Refresh Token comportent tous deux une propriété **exp** indiquant la durée de vie du jeton. Il suffit alors de mettre le jeton en cache pour une durée légèrement inférieure à cette durée d'expiration. Enfin, il est nécessaire d'obtenir un nouveau Context Token quand celui-ci est perdu (point 3) ou quand l'Access Token a expiré. La classe TokenHelper propose la méthode GetAppContextTokenRequestUrl qui fournit une URL permettant à l'utilisateur de récupérer un Context Token :

```
Response.Redirect(TokenHelper.GetAppContextTokenReque
stUrl(«https://cellenza.sharepoint.com», Server.Url
Encode(Request.Url.ToString())));
```

Conclusion

L'une des ambitions de SharePoint 2013 est de remplacer le développement de solutions Sandbox par les Apps, permettant ainsi de déporter du code qui influait souvent sur la stabilité de la plateforme SharePoint.

Nous avons vu que le modèle Apps permet de s'affranchir de certaines limitations des solutions SandBox : il est désormais possible d'élever les privilèges d'accès aux ressources. De plus, le modèle définit des permissions d'accès granulaires aux ressources contrairement aux solutions WSP classiques. Ce nouveau modèle de développement n'est néanmoins pas sans représenter de nouveaux challenges pour l'authentification et l'autorisation d'accès aux ressources SharePoint. Il contraint à autogérer les jetons d'accès au sein de son application. Heureusement, la communauté SharePoint est active sur ce sujet et de nouvelles solutions sont apportées chaque jour (telles que SharePointContext Helper) afin de simplifier la gestion des jetons dans vos Apps. Il y a fort à parier que Microsoft intégrera à l'avenir ce type de solutions dans une mise à jour.

🔴 Antoine DONGOIS

Manager chez Cellenza - Cellenza - Software Development Done Right
Blog: <http://blog.cellenza.com>

Le quotidien du designer

La pratique du design industriel est ancienne et ancrée dans les process, mais la pratique du design d'interaction, bien plus récente, est peu intégrée. Ces dernières années les entreprises ont tout juste commencé à percevoir la différence entre le graphisme et le design d'interaction (ou design d'expériences numériques). L'utilité du designer lors de la phase de lancement des projets, sa présence en tant qu'organisateur d'atelier (client/utilisateurs), et l'importance des phases de storyboard et wireframe sont de moins en moins discutées. La place du graphiste commence elle aussi à être plus claire.

Le design est un métier qui se veut international. Le designer d'interaction conçoit pour l'Homme, qui peut être de n'importe quelle nationalité, de n'importe quelle religion, et avec n'importe quelles connaissances et aisance avec les nouvelles technologies.

SAVOIR

Son rôle est de concevoir en gardant à l'esprit la connaissance des utilisateurs. Il doit savoir écouter, comprendre, analyser, afin de répondre aux besoins des utilisateurs dans le budget fixé. Il doit garder à l'esprit l'objectif des clients.

Sa connaissance des tendances lui permet de guider son client vers des choix qui, soit donneront à son projet une longueur d'avance, soit lui permettront de rester dans la course.

Comme pour le design industriel, le designer doit connaître les propriétés intrinsèques d'un matériau pour savoir comment il va l'utiliser et quel objet il va pouvoir concevoir. Une compréhension des aspects techniques et une communication constante avec les équipes de développement lui permettront de sortir des Wireframe d'applications pertinentes et réalisables.

Le design d'interaction est essentiel. Un mauvais design peut entraîner de lourdes problématiques. En mai 1999 au Royaume-Uni le temps d'attente pour obtenir un passeport est passé de 2 semaines à une plage de 7 à 10 semaines. La raison principale de cette attente était en fait le changement des applications permettant la création des passeports.

L'objectif du designer est de rendre palpable et visuel ce que pense le client, ou ce dont les utilisateurs ont besoin.

Pour transformer les idées en applications, le designer d'interaction va commencer par écouter, analyser, puis dessiner. Les croquis à la main, quelques mots clés, sont les premiers éléments qui vont être réalisés par le designer. Le design d'interaction est une discipline principalement créative. Elle est analytique dans sa première phase, créative dans sa seconde phase et devient technique et précise dans une troisième phase.

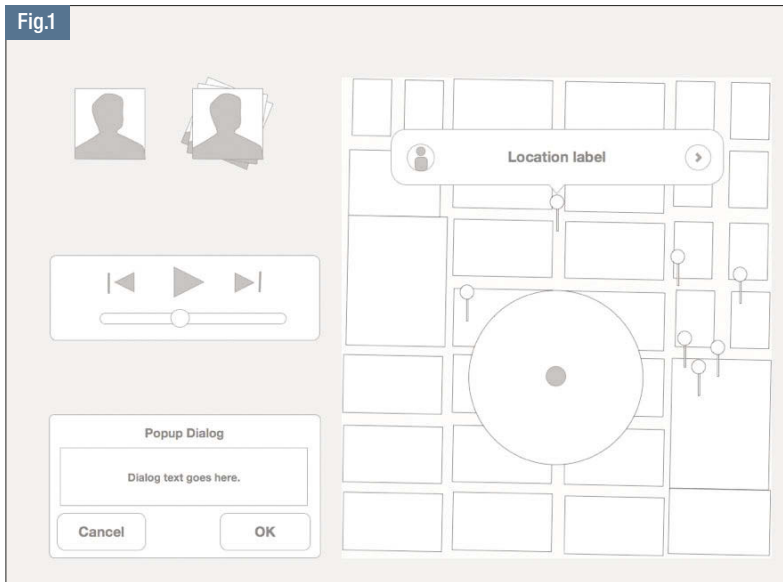
ÉQUIPE ET PROCESSUS

Le designer travaille avant tout en équipe, avec des graphistes, des développeurs et chacun d'entre eux a ses propres contraintes. Chaque métier a son domaine d'intervention et les contraintes qui y sont liées. On connaît les contraintes pour les développeurs, mais moins celles du designer qui doit concevoir une application intuitive, simple, utile, utilisable, pertinente et cohérente. En permanence il doit garder en tête les contraintes de budget. Il va devoir trouver le bon équilibre entre l'expérience utilisateur, les contraintes techniques liées au développement, la complexité graphique. Il doit répondre aux besoins du client tout en respectant ses contraintes de marketing et de communication.

Le processus de design se fait en plusieurs étapes et la première d'entre elles est l'animation d'ateliers avec le client et les utilisateurs.

L'objectif est la compréhension du besoin du client et des besoins et attentes des utilisateurs. Cette étape est clairement la plus importante du projet. C'est à ce moment-là que le designer va récupérer l'ensemble des informations nécessaires au design du projet.

Fig.1



Ces informations sont la base sur laquelle va reposer l'ensemble du projet, il est donc nécessaire que cette base soit solide, encore plus quand le client ne fournit pas de cahier des charges.

Le designer devra récupérer les informations de contexte du projet. Il devra aussi récupérer des informations sur les valeurs et l'image de la société. Quelques questions clés permettent de tirer un maximum d'informations des ateliers, mais nous y reviendrons juste un peu plus bas.

Le designer en ressortira avec une liste de fonctionnalités, de contraintes, de règles, de besoins et d'envies.

Pour les ateliers, un des fonctionnements les plus efficaces est d'en faire un premier avec le client, puis plusieurs avec les utilisateurs clés.

Un utilisateur clé est une personne travaillant sur le terrain, qui connaît bien le métier, ou qui peut permettre une meilleure acceptation de l'outil au sein de l'équipe s'il est impliqué dans le projet.

LES QUESTIONS À SE POSER

Voici quelques questions à poser qui permettent au designer de guider le client dans la bonne direction pendant l'atelier. On pourrait presque imprimer cette liste et la poser systématiquement : qui sont les utilisateurs ?

Afin de savoir à quelle tranche d'âge on s'adresse (actifs ou retraités), si les utilisateurs sont à l'aise avec les nouvelles technologies ou pas, si on s'adresse à des femmes, hommes, enfants, etc. Quel est leur univers ? L'étude de l'univers des utilisateurs permettra de créer une ambiance d'application adaptée aux utilisateurs. On parle ici principalement de mots clés utiles pour le

designer comme pour le graphiste. Il arrive que pour répondre à cette question les créatifs soumettent aux clients une planche d'ambiance qu'ils valideront.

Quel est le nombre d'utilisateurs ? Le nombre d'utilisateurs permet de déterminer la largeur du panel ciblé et le nombre à sélectionner pour que les tests utilisateurs soient efficaces.

Pour certaines applications le nombre d'utilisateurs simultanés permettra de déterminer le choix du support. Un à deux, une borne tactile ou une application Kinect. Trois, quatre, cinq, une table tactiles, etc.

Où l'application va-t-elle être utilisée ? Le lieu va avoir un impact sur le choix matériel.

Car un périphérique est conçu pour des conditions particulières. Cette question permettra d'orienter le choix vers un matériel durci ou non, une technologie tactile ou une autre.

Un changement de lieu pourra entraîner à la fois une problématique de fonctionnement du périphérique ainsi qu'une problématique dans la conception de l'interface.

L'utilisateur sera-t-il seul ou accompagné ?

Y a-t-il une phase d'apprentissage ou de formation de prévue afin de déterminer le niveau d'initiation à intégrer dans la solution ?

Une application dans un salon en libre-service devra être ultra simple alors que dans une application métier, la phase d'apprentissage permet un certain niveau de complexité.

Combien de temps, de manière globale (ou par jour), sera utilisée l'application ?

En combien de temps faut-il pour que l'utilisateur arrive à tel ou tel niveau d'information/d'action ?

Les réponses à ces questions permettront d'aider à déterminer la hiérarchie et les gestuelles qui permettent d'accéder à une information comme à une autre, selon leur importance.

La quantité d'information et la façon d'y accéder seront impactées par les réponses à ces questions.

Quelles sont les problématiques connues à résoudre ? Et quel est l'objectif de l'application ? Vendre plus, accroître la productivité, concevoir un outil accepté par les utilisateurs, faire du buzz, etc. tous ces objectifs doivent être connus par le designer, car selon les objectifs, le design d'un même type d'application sera complètement différent.

Ensuite vient la phase d'analyse. Le designer va étudier 3 points tels que :

- Les informations clés énoncées par le client et les utilisateurs lors des ateliers en ce qui concerne le projet
- Les valeurs de la société à retranscrire dans l'application.
- L'analyse des outils actuels et outils concurrents.

Ce dernier point permettra au designer de proposer une solution innovante qui apportera de vraies valeurs ajoutées par rapport aux solutions existantes.

Ensuite, il faut s'atteler aux véritables problématiques qu'il va falloir régler. Ces problématiques sont souvent liées à la productivité dans les applications métier et d'attractivité dans les applications grand public.

C'est pendant les ateliers avec les utilisateurs et pendant la phase d'analyse de l'outil que va émerger cette liste de problématiques à régler. Une problématique très actuelle : « *transformer les applications métiers à la souris en applications tactiles, afin de permettre aux utilisateurs d'être plus efficaces tout en leur apportant de la souplesse avec un outil adapté à leur quotidien* ». Le designer va ensuite pouvoir entrer dans sa phase de créativité.

La phase créative est une partie très importante dans les projets, qu'il s'agisse de la phase créative du designer, ou de la phase créative du graphiste. C'est grâce à ces phases que la solution développée sera unique et pertinente par rapport au contexte de chaque client et de chaque projet.

L'objectif est donc de rechercher des idées innovantes, de nouveaux types d'interactions ou de gestuelles. Par exemple, trouver une navigation qui tire parti du support sur lequel on conçoit.

Il faut trouver des idées qui permettent de résoudre les problématiques tout en répondant à l'ensemble des attentes des utilisateurs. L'objectif ici est de laisser libre court à son imagination sans (trop) penser aux contraintes techniques. Va émerger de ce travail un ensemble de croquis, des photos, des mises en situation et des story-boards.

Après cette phase, le designer va reprendre toutes ses idées et les replacer dans un wireframe. Le wireframe permet de mettre à plat l'ensemble des pages et donc l'ensemble des interactions. Le designer va donc repasser sur l'ensemble de ses idées, en écarter certaines pour en sélectionner d'autres, les simplifier et les homogénéiser afin qu'elles répondent à l'ensemble des contraintes.

L'objectif est de clarifier le fonctionnement et la composition de l'application afin de pouvoir présenter le wireframe aux équipes techniques. Cette phase avec les équipes techniques permet de lister ce qui est possible ou non dans le budget et le temps imparti.

Les designers conçoivent chacun avec l'outil qui leur correspond le mieux. Mon outil dans une première phase est toujours le papier crayon. En voici les 6 raisons ; avant tout la rapidité d'exécution est inégalable, la rapidité de modification l'est aussi.

Cette technique permet une grande réactivité pendant les ateliers, car les corrections se font en live et le designer n'est pas caché derrière son écran. Le designer va garder la main à tout moment contrairement à un outil de type Balsamiq Mockups qui rend le wireframe collaboratif. Souvent le client se prend pour un designer et effectue des modifications en ligne en pensant que le fait de savoir manipuler un outil de Moku lui donne automatiquement les compétences de designer.

Il devient plus complexe ensuite d'argumenter et de retirer ces modifications. La cinquième raison est que ce type de rendu ne bridera pas la créativité des graphistes qui interviennent juste après. Cela peut paraître anecdotique, mais c'est une vraie problématique dans les équipes pluridisciplinaires qui regroupent des graphistes, des designers d'interaction et des ergonomes.

Et la dernière raison est que le client et les utilisateurs ne cherchent pas à se projeter sur un résultat graphique, mais restent focalisés sur l'essentiel à cette étape : le contenu, le fonctionnement, les interactions, les fonctionnalités.


L'étape d'après est la réalisation d'un wireframe « au propre » avec les spécifications détaillées de chaque écran, ce qui permettra d'avoir une base solide pour écrire les cahiers de tests et pour les développeurs de manière générale.

Après quelques corrections suite aux entretiens avec les équipes techniques, le designer va pouvoir livrer le wireframe d'une solution, adaptée au besoin du client, aux attentes des utilisateurs et aux contraintes de budget.

EN RÉSUMÉ

Le design consiste en la conception de solutions cohérentes et adaptées aux tendances, budget, image de marque, besoins et envies. Il intègre les contraintes techniques, ergonomiques, fonctionnelles, symboliques, esthétiques et structurelles. Pour réaliser une solution intuitive, ergonomique, simple, utile, efficace, pertinente et cohérente.

Le designer doit pouvoir adapter sa réflexion indépendamment du domaine, de la complexité et du support (physique ou virtuel). C'est cette souplesse qui permet aux designers de se projeter dans de nouveaux sujets et d'absorber de nouvelles contraintes quotidiennes tout en restant créatifs et innovants. Le monde est en perpétuelle évolution. Le design l'est aussi.

 Johanna Rowe Calvi
Wygwam

@johanna_rowe

www.johannarowe.com

Page Facebook : Design in Progress

Timeline : 1988-1997



Objet : Newton

Dans l'univers des assistants numériques, on se souvient facilement du Palm Pilot. Un des projets les plus ambitieux fut le Newton d'Apple, décliné sous la forme d'un assistant, MessagePAD, et d'un « portable », le eMate (dédié à l'éducation).

L'assistant numérique fut une des grandes idées de John Sculley (le même qui vira Steve Jobs) dès 1987. L'idée était de pouvoir utiliser un appareil, avec un stylet, pour des tâches quotidiennes, avec la capacité à écrire directement sur l'écran et que cette écriture soit reconnue (reconnaissance de l'écriture). Le développement du projet fut tout sauf simple : budget, situation économique d'Apple, conflits divers et variés avec la direction et les autres équipes, manque d'une vision claire.

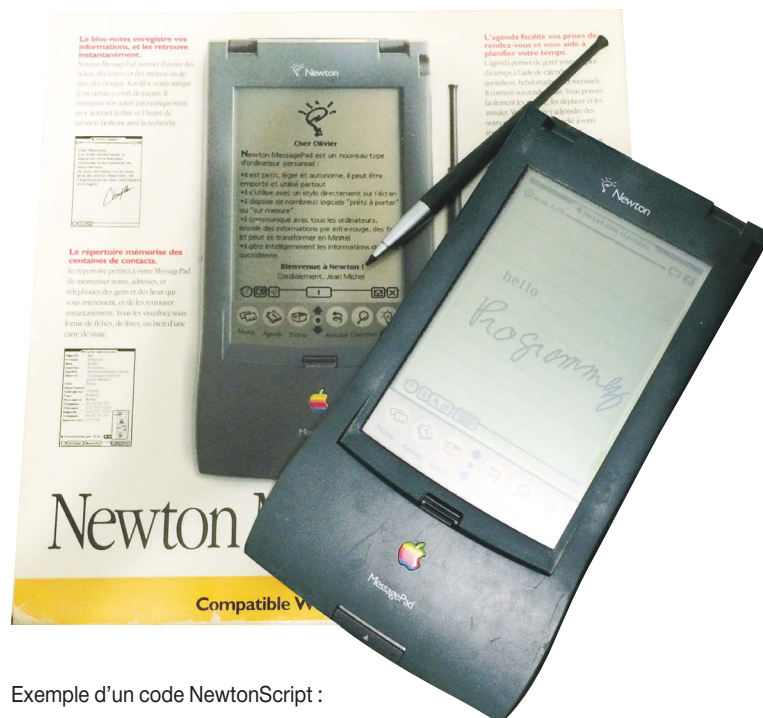
Un matériel trop limité et un système trop lent

Le projet est présenté officiellement en mai 1992 et sera réellement disponible à la vente en été 1993. Avec tous les défauts de cette version. Apple n'avait pas eu le temps ou les ressources techniques pour optimiser les applications, améliorer le matériel. Le MessagePad était lent, très lent, la reconnaissance d'écriture aléatoire, surtout en Français.

Le système NewtonOS reposait sur un micro-noyau. Au-dessus, nous avons les couches de communication, le moteur de reconnaissance et l'environnement NewtonScript. NewtonScript était un langage orienté objet, il s'inspirait du langage Smalltalk, de Lisp et de Self. Au niveau système, C++ restait le maître, alors que les applications et l'interface étaient écrites en NewtonScript. Le matériel n'était pas assez puissant et trop limité : processeur 20 Mhz, 4 Mo de ROM, maximum 640 Ko de mémoire, écran noir et blanc. L'écran sera réellement amélioré avec l'eMate et MessagePad 2000. Ce dernier améliore de nombreux éléments matériels : processeur, écran, prise en main, système. Il est capable de recevoir et d'émettre des e-mails, possède un navigateur Internet. Les premiers écrans, non rétro-éclairés, jouaient souvent à l'effet miroir, très désagréable à l'usage. L'interface ressemblait à ce que l'on pouvait trouver sur HyperCard et le système Macintosh de cette époque.

La technologie de reconnaissance d'écriture sera améliorée avec un nouveau moteur : CaliGrapher. Ce moteur était développé par la société soviétique ParaGraph. Apple n'arrivant pas à améliorer suffisamment son propre moteur.

Sur NewtonScript, des passionnés ont développé des émulations et des outils pour continuer à l'utiliser : <http://newtonscript.org>



Exemple d'un code NewtonScript :

```
aFrame:= {foo: 10,
bar: func(x)
begin
    if foo then Print ("hello");
    if x > 0 then
begin
    local foo; //local variable to function
    foo:= 42;
end
return foo;
end;
}
```

La fin

Apple n'a jamais su et pu développer un écosystème performant. Les ventes furent médiocres, la situation d'Apple très délicate. La concurrence était très vive. Le Palm Pilot était plus petit, plus rapide et surtout moins cher. Apple était au bord de la faillite en 1997. Durant les premiers mois, Steve Jobs donna quelques espoirs aux équipes Newton mais rapidement, il décida de liquider le projet.

Pour en savoir plus

Site : <http://www.aventure-apple.com/flops/newton.html>

Time Machine est aussi votre rubrique. Envoyez-nous vos chroniques temporelles : redaction@programmez.com

Abonnement : Programmez, 17, Route des Boulangers, 78926 Yvelines Cedex 9 - Tél. : 01 55 56 70 55 - abonnements.programmez@groupe-gfi.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 exclusivement 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 : N. Saurel
Secrétaire de rédaction : Olivier Pavie
Experts : C. Villeuneve, Greg, Julien Dollon, Jason de Oliveira, B. Masson, Maxime Labelle, J. Devillard, J. Guittard, F. Sznadgerman, H. Bouhanef, A. Bouadma, FM, M. Taslimanka Sylla, L. Ellerbach, K. Alinjres, X. Millot, S. Sound, T. Loubiou, A. Don, F. Guillot, S. Saurel, A. Guet, A. Dongois, J. Rowe-Calvi, G. Rouchon, F. Bellahcene,

M. Perfetti, P-H. Gache, J. Landon, P. Puschmann

Crédits couverture : 09-05-13 © jauhani / iStock

Maquette : Pierre Sandré

Publicité : Régie publicitaire, K-Now sarl. Pour la publicité : Tél. : 01 41 77 16 03 - diff@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 : 0967320934

Contacts

Rédacteur en chef : ftonic@programmez.com

Rédaction : redaction@programmez.com

Webmaster : webmaster@programmez.com

Publicité : diff@programmez.com

Evenements / agenda :
redaction@programmez.com

Dépôt légal : à parution - Commission paritaire : 1215 K 78366 - ISSN : 1627-0908

© NEFER-IT / Programmez, janvier 2014
Toute reproduction intégrale ou partielle est interdite sans accord des auteurs et du directeur de la publication.

Développeurs,
Professionnels de l'informatique,
Secteur Public,
Décideurs métiers et IT,
Inscrivez-vous sur **www.mstechdays.fr**
Suivez-nous sur **#mstechdays**

Microsoft
tech·days
11,12,13 février 2014

DIGI TAL IS BUSI NESS

LE RENDEZ-VOUS INCONTOURNABLE
DE L'INNOVATION NUMÉRIQUE
AU PALAIS DES CONGRÈS DE PARIS

3 Keynotes
6 zones expérientielles
140 exposants
300 sessions
18 000 visiteurs
Une Techdays TV

Le saviez-vous ?



De Janvier à Avril 2014,

WINDEV est parrain de l'émission CAPITAL sur M6



Environnement de
développement
professionnel
cross-plateformes



*Fournisseur Officiel de la
Préparation Olympique*

Siège 3 rue de Puech Villa - 34197 MONTPELLIER Cedex 05 **Tél. 04 67 032 032** Fax 04 67 03 07 87 Tél. Support Technique: 04 67 03 17 17

Agence 142 Avenue des Champs Elysées 75008 PARIS Tél. 01 48 01 48 88

PC SOFT Informatique - SAS au capital de 2 297 548 Euros - RCS 330 318 270 - Code APE 5829B

www.pcsoft.fr