PROGRAMMEZ.COM

PROGRAMMEZ!

Mensuel n°179 - Novembre 2014

le magazine du développeur

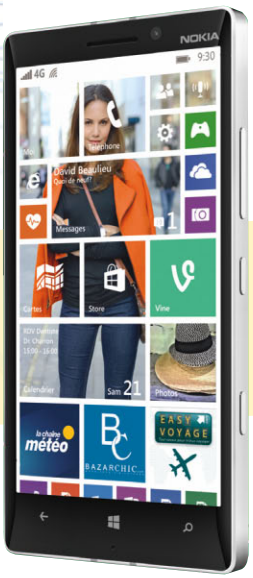
Toute la puissance du

Cloud Computing

Microsoft Azure / Heroku DX
Google Cloud Platform
Amazon Web Services

© nuage : 07-17-13 © frankpetris - intérieur : 07-25-13 © Henrik5000

Web
(re)découvrez
Wordpress



Cahier central

Windows Phone

les apps universelles, les SDK Nokia

Objets connectés / IoT

- Les bornes iBeacon
- Les nouveaux kits Intel Galileo 2 / Starter Kit IoT
- Les réalités alternatives : les nouveaux mondes à votre portée

Antiquité
Les mythiques machines Amstrad

Sécurité Android
Occultez votre code source

Success Story
Magma Mobile

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

M 04319 - 179 - F: 5,95 € - RD

Quelle interopérabilité entre mes différents fournisseurs Cloud ?

Avec Aruba Cloud,

vous avez l'assurance de ne pas être prisonnier d'un fournisseur. Nos services sont intégrés au **driver DeltaCloud** et compatibles **S3**. De plus, vous pouvez utiliser des formats standards d'images de machines virtuelles, **avec VSD et VMDK**, ainsi que des modèles personnalisés provenant éventuellement d'autres sources.

OFFRE SPECIALE 2 *fois plus pour le même prix!***

- **3 hyperviseurs**
- **6 datacenters en Europe**
- **APIs et connecteurs**
- **70+ templates**
- **Contrôle des coûts**

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

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

aruba CLOUD

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

MY COUNTRY. MY CLOUD.*

*MON PAYS, MON CLOUD.

**Pour tout premier versement lors de votre première création de compte entre le 01/10/2014 et le 30/11/2014, sous la forme d'un coupon émis sous quinzaine d'un montant égal au premier montant versé.

Le boss.
ftonic@programmez.com

La checklist

- ☐ *Je code avec NotePad*
- ☐ *J'utilise un vrai OS : sujet idéal pour troller en soirée*
- ☐ *Installer une sonde thermique dans son ultrabook*
- ☐ *J'comprends pas, le compilo refuse de compiler et pourtant j'ai bien fait, avec de jolis noms de variables, de classes et de fonctions qui déchirent. P't'être trop ?*
- ☐ *Paintball ou déjeuner en famille ? Ouhais, paintball.*
- ☐ *Pizza, coca, café, un repas bien équilibré.*
- ☐ *Répondre à un twitt c'est super important même la nuit*
- ☐ *Le comic'con c'est dans 6 mois. Qu'est-ce que je vais mettre ?*
- ☐ *Je suis Néo.*
- ☐ *Dark Vador est mon ami.*
- ☐ *Mettre les mêmes chaussettes et t-shirt une semaine, big check. On est nerd ou on ne l'est pas.*
- ☐ *Se lever à 7h pour aller au boulot, non. Mais se lever à 4h du matin pour faire la queue pour le prochain Hobbit, iPhone, Star wars, Trône de Fer, oui. J'ai des valeurs.*
- ☐ *Quoi le Raspberry Pi ne sert à rien ? Tu rigoles. Mais si, il sert à tout. Bon là je n'ai pas le temps de le faire mais tu verras (note à moi-même : trouver la killer feature sinon je suis bon pour acheter des fleurs ou pire)*
- ☐ *Tu penses à tout pour ta session technique sauf au petit connecteur de m... pour se connecter au vidéoprojecteur ! Big fail !*
- ☐ *Quoi ? Y'a ni WiFi ni 3G, ni 4G, ni Netflix ? Je hais les vacances.*
- ☐ *Suivre le lapin blanc*
- ☐ *Caprica 6 ou Penny ? Sheldon ou Gaïus Baltar ?*
- ☐ *Bah, un écran bleu ? La routine.*
- ☐ *Format C : c'est comme du Beethoven, un classique indémodable*

À LIRE DANS LE PROCHAIN NUMÉRO n°180 en kiosque le 29 novembre 2014

DevOps
Faut-il réellement faire du développement continu ? Comment les outils en Cloud bouleversent notre quotidien ?
Le DevOps à la carte

Facebook Parse
Développez rapidement et en ligne des applications mobiles avec Parse

Langage
Tour d'horizon et prise en main de Swift, le nouveau langage d'Apple

Windows 10, quoi de 9 ?

Les rumeurs les plus insistantes nous on fait miroiter un Windows 9... Cela sera finalement Windows 10 que Microsoft a annoncé le 30 septembre dernier ! Ce saut de numéro de version est volontairement choisi afin de marquer la rupture entre l'expérience utilisateur de Windows 8(.1) et cette nouvelle mouture. D'un point de vue développeur, il n'y a pas de nouveaux SDK mais il est d'ores et déjà intéressant de regarder ce que nous réserve Microsoft !

L'un des objectifs majeurs que Microsoft souhaite mettre en place avec l'arrivée de son nouveau système d'exploitation est l'unification entre les différentes plateformes. La firme de Redmond souhaite que tous les utilisateurs de Windows 10, aient la même expérience que ce soit sur ordinateur, tablette ou smartphone. Avec Windows 8 et Windows Phone 8, Microsoft a déjà fait quelques pas vers cette unification. On retrouve notamment certaines applications modernes communes, des paramètres d'applications et du système d'exploitation partagés entre les deux plateformes.
Selon les premières annonces de Microsoft, Windows 10 devrait donc encore plus renforcer l'unicité entre les supports équipés de cette dernière version. En termes de développement, il sera donc encore plus intéressant de proposer les deux versions dont le code sera mutualisé pour une expérience utilisateur encore plus homogène. Notez que sur les supports non équipés de tactile, la barre des charmes sera supprimée pour assurer une meilleure adéquation avec le clavier et la souris.

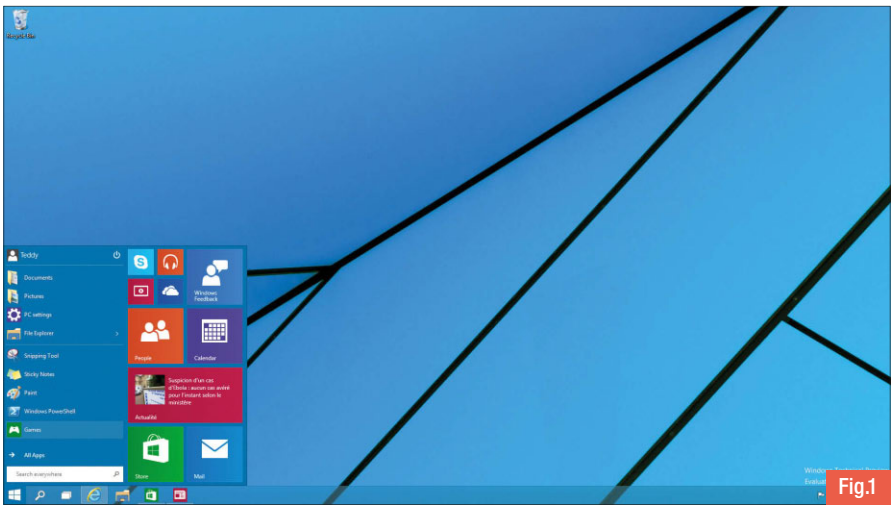


Fig.1

Le retour du menu démarrer

Cette nouveauté, ou plutôt ce retour, du menu démarrer est LA fonctionnalité qu'il manquait à de nombreux utilisateurs de Windows 8. Bien que Microsoft l'ait remplacée par l'écran d'Accueil à la sortie de Windows 8, puis réintégrée, en partie grâce au bouton « démarrer », dans sa version 8.1. C'est bien dans cette nouvelle version qu'il fait son grand retour. Il éclipse donc l'écran d'Accueil (que certaines personnes finalement habituées à utiliser vont peut-être regretter), sans pour autant faire disparaître le clic droit sur le bouton démarrer et propose donc une nouvelle façon d'accéder à nos programmes.
Après quelques minutes d'utilisation, on retrouve assez facilement nos habitudes pour peu d'avoir expérimenté Windows 8 et Windows 7. Ce menu démarrer made in Windows 10, se présente en deux grandes parties. La zone de gauche qui ressemble étrangement au menu démarrer de Windows 7. Elle réunit vos programmes préférés (que l'on a épinglés en haut de la liste) et ceux ouverts récemment. La zone de droite, quant à elle, met en avant sous forme de tuiles, vos applications (bureau ou moderne), liens et dossiers favoris Fig.1.
Depuis l'ancien menu démarrer de Windows 7, Microsoft a fait beaucoup de chemin dans la personnalisation de son environnement. Outre la personnalisation due à la possibilité d'épingler ses programmes, l'utilisateur peut, comme sous Windows 8, modifier la taille des tuiles (petite, moyenne et grande) mais également modifier la hauteur du menu démarrer Fig.2.

La recherche universelle dans le menu démarrer

Les plus attentifs d'entre vous, auront surement remarqué que dans les précédentes images du menu démarrer, le champ de recherche universel y fait également son apparition. Cette recherche universelle est la même que celle qui était présente sous Windows 8 dans la barre des charmes. Elle permet donc d'effectuer des recherches sur des programmes, des paramètres, des documents, et également sur des ressources Internet (sites Internet, images, vidéos) via le moteur de recherche Bing Fig.3.
Les résultats de la recherche effectuée dans le menu démarrer peuvent être visualisés dans une interface proche du moteur de recherche Bing où les résultats y sont plus lisibles et séparés en catégories.

Multi-bureaux virtuels

Les bureaux virtuels font également leur apparition de façon native dans cette nouvelle mouture du système d'exploitation de Microsoft. Cette fonctionnalité permet de ne pas se limiter à un seul bureau et donc ne pas avoir tous vos programmes ouverts au même endroit mais de bien pouvoir séparer son travail en différents espaces. Dans les faits, on pourrait par exemple imaginer l'utilisation de ces bureaux virtuels au travail avec un premier bureau qui contiendrait des programmes professionnels et un second avec des programmes personnels.
L'interface de contrôle des multi-bureaux est accessible via le « Task View », une nouvel icône qui fait son apparition dans la barre des tâches et qui remplace le raccourci Windows + Tab. Dans celle-ci, l'utilisateur peut gérer ses bureaux virtuels (ajouter, supprimer), naviguer de l'un à l'autre, ainsi que visualiser et supprimer les programmes ouverts sur chacun d'entre eux. Cependant, des limitations sont encore

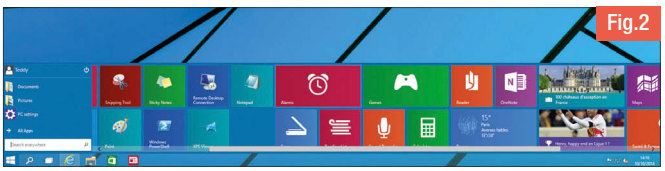


Fig.2


Fig.3

présentes dans cette nouvelle fonctionnalité. Pour le moment, il n'est pas possible d'ouvrir deux fois les mêmes applications Windows Store dans deux bureaux différents ni de déplacer une application d'un bureau à un autre (sauf en la fermant et en la ré-ouvrant dans le bureau souhaité).

L'ancrage (Snap) des fenêtres

Le Snap introduit avec Windows 7, est l'action qui consiste à pouvoir modifier la taille des fenêtres en les déplaçant sur les bords du bureau. Sous Windows 10, ce mode d'affichage a été retravaillé par les équipes de Microsoft et ajoute de nouvelles possibilités. Il est désormais possible d'ancrer jusqu'à quatre fenêtres à la fois, ce qui engendre donc de nouvelles positions (petite fenêtre haut gauche, haut droite, bas gauche, bas droite et demi-fenêtre haut, bas). De plus, lorsque l'on ancre une fenêtre, au lieu de laisser comme auparavant l'autre espace vide, Windows vous propose d'y ancrer un autre programme déjà ouvert. Cela évite de refaire la manipulation d'ancrage si l'on souhaitait mettre côte à côte deux fenêtres.

Floating Modern Apps

Pour prendre en compte certains retours des utilisateurs finaux, Microsoft a repensé également les Modern Apps. Ces applications arrivées avec Windows 8 qui ont pour particularité de s'utiliser en plein écran, peuvent maintenant être utilisées dans les mêmes conditions que des applications classiques. Elles peuvent désormais être utilisées sur le bureau en mode fenêtré et donc être ancrées et redimensionnées (selon certaines limites). Bien que les applications disponibles sur le store et le store lui-même ne soient pas encore optimisées pour Windows 10, certaines comme Sport, Voyage, Cuisine & Vins fonctionnent très bien et permettent de bien concevoir ces nouveaux ajouts. Pour répondre aux développeurs d'applications modernes, pour le moment aucunes bonnes pratiques n'ont été publiées concernant Windows 10. Il vous faudra donc attendre encore un peu avant de pouvoir porter vos applications sur ce nouvel OS. Un dernier effet de bord, dû aux différents ajouts, est celui de la barre des charmes. Pour rappel, ces applications utilisaient notamment la barre des charmes pour rechercher ou paramétrer l'application. Celle-ci n'étant plus disponible sur des PC qui ne sont pas équipés de tactile, Microsoft a corrigé ce problème en ajoutant un menu présent en haut à gauche de chaque application moderne. Il propose les mêmes actions que dans la barre des charmes et l'actionne au besoin.

Comment le tester ?

Vous aussi vous voulez tester Windows 10 ? C'est possible en vous inscrivant directement sur le site https://insider.windows.com/ où vous pourrez télécharger un ISO d'installation. À ce jour il n'existe que les versions anglaise, chinoise et portugaise de Windows. Pour la suite il s'agit d'une installation Windows traditionnelle, rien de nouveau de ce côté-là. Cette version se mettra à jour automatiquement au moyen de Windows Update. Microsoft a d'ailleurs promis un cycle de mise à jour très rapide par ce biais.

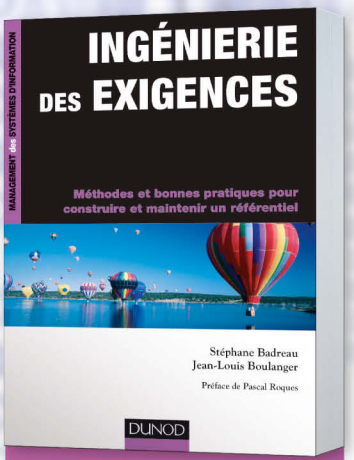
Conclusion

Nous avons présenté les fonctionnalités les plus visibles. Microsoft a attaché une grande importance à l'apport de nouveautés sur l'ensemble de l'OS, même dans la ligne de commande qui supporte, pour notre plus grand bonheur, le copier-coller... Au travers de cette 1ere annonce, Microsoft nous laisse déjà découvrir la nouvelle ergonomie et les écosystèmes applicatifs. De quoi nous laisser envisager de belles applications à venir ! On attend donc avec impatience les prochains SDK et annonces !

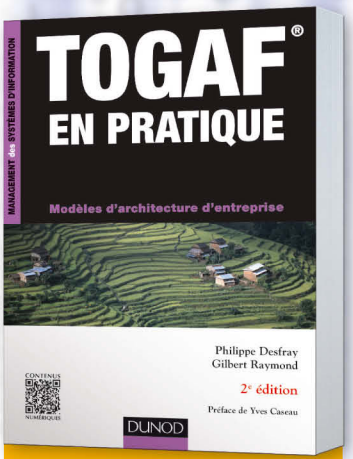
Jonathan **Antoine**,
Teddy **Desmas**,
Maxime **Frappat**,
consultants Infinite Square

INFINITE SQUARE

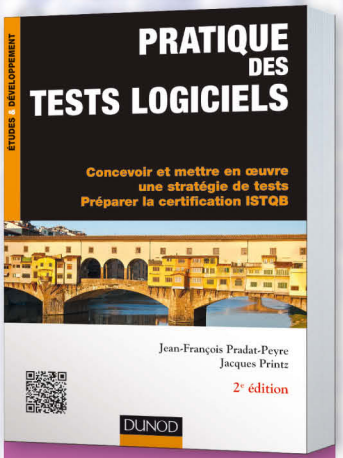
DÉVELOPPEZ VOS COMPÉTENCES



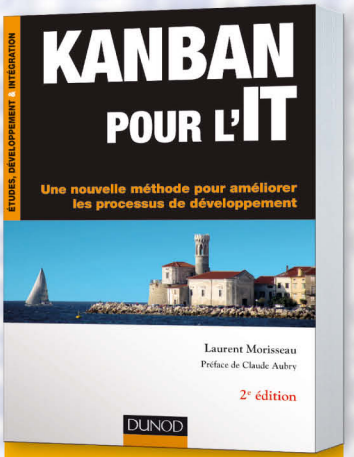
S. BADREAU, J.-L. BOULANGER
9782100706402 • 300 pages • **39,00 €**
Les connaissances de base pour développer des systèmes complexes à forte composante logicielle



P. DESFRAY, G. RAYMOND
9782100712823 • 288 pages • **35,00 €**
Très orienté solutions, ce livre apporte un point de vue de praticiens sur la modélisation d'architectures d'entreprise avec TOGAF®



J.-F. PRADAT-PEYRE, J. PRINTZ
9782100706082 • 240 pages • **29,50 €**
Les bonnes pratiques pour concevoir et mener à bien une stratégie de tests



L. MORISSEAU
9782100710386 • 304 pages • **29,90 €**
Améliorer les processus de développement avec la méthode Kanban

Les actus du savoir

www.dunod.com

DUNOD
ÉDITEUR DE SAVOIRS

Réalisation : VIP

Xamarin Evolve 2014 : les nouveautés à ne pas rater !

Evolve est la conférence organisée par Xamarin au cours de laquelle les développeurs peuvent participer à des trainings, assister à des sessions techniques et, surtout, découvrir les nouveautés sur lesquelles travaillent les équipes de Xamarin pour les prochaines versions ! Cette édition 2014 n'a pas dérogé à la règle et a apporté un ensemble de nouveautés que nous allons développer ici.

Tout d'abord, le nombre de partenaires a augmenté de manière significative depuis l'année dernière. En effet, il y a maintenant 215 partenaires (contre 45 en 2013), chacun agissant dans des domaines d'expertise différents. Ainsi, parmi ces partenaires, on retrouve des sociétés techniques, permettant d'accompagner les clients dans leurs développements (à l'image d'Infinite Square, qui est *Xamarin Authorized Consulting Partner*) ou bien des sociétés connues de longue date pour leur expertise dans la création de contrôles, tel que SyncFusion, Telerik, Infragistics, etc. (la liste exhaustive est disponible à cette adresse : http://blog.xamarin.com/enterprise-component-vendors-join-xamarin.forms-ecosystem/).
La partie Entreprise n'a pas été laissée de côté non plus car un partenariat entre IBM et Xamarin a été annoncé, en direct par Nat Friedman (CEO et co-foundateur de Xamarin) concernant la mise à disposition d'un SDK (pour Xamarin) permettant de se connecter à IBM Worklight (http://www-03.ibm.com/software/products/en/worklight-foundation).

Xamarin Android Player, la nouveauté à ne pas rater

D'un point de vue purement « développeur », de belles nouveautés ont été annoncées. Ainsi, nous avons maintenant à notre disposition le **Xamarin Android Player** ! Fig.1.
Il s'agit d'un émulateur Android, développé en WPF par les équipes de Xamarin, reposant sur une architecture x86 et basé sur VirtualBox. Certes, il s'agit d'une Preview, mais de ce que l'on a été en mesure de voir, les performances semblent vraiment au rendez-vous ! Démarrage en moins de 20 secondes, utilisation d'OpenGL (donc l'exécution de jeux ne pose pas de problème), glisser/déposer de fichiers APK pour installer des applications : tout a été pensé pour faciliter la vie des développeurs !

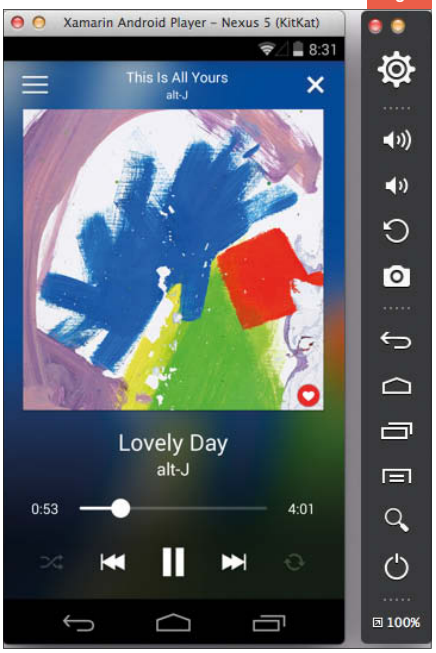
Prototypez vos applications de manière fulgurante avec Sketches

Nous avons également eu droit à la démonstration de **Sketches**, une nouveauté de Xamarin Studio qui permet de prototyper, très rapidement, des applications en utilisant C# ou F#, Fig.2.
Là où Sketches est intéressant, c'est qu'il permet d'avoir un rendu, en temps réel, de l'exécution de son application, et cela quel que soit le Framework ou la plateforme-cible visée. Ainsi, par exemple, vous pouvez prototyper une application Xamarin.Forms et visualiser le résultat directement, pour iOS ou Android.

Xamarin Profiler ou le profiling d'applications iOS/Android

Autre nouveauté purement liée au développement : le **profiler Xamarin** : Fig.3.
Les équipes de Xamarin ont développé un outil, intégré à Xamarin Studio et Visual Studio, qui permet d'analyser tout le côté « managé » (entendre par là .NET) d'une application iOS ou Android. Ainsi, il devient plus simple et pratique de déterminer (et corriger) la source d'éventuelles fuites mémoire au sein de nos différentes applications !

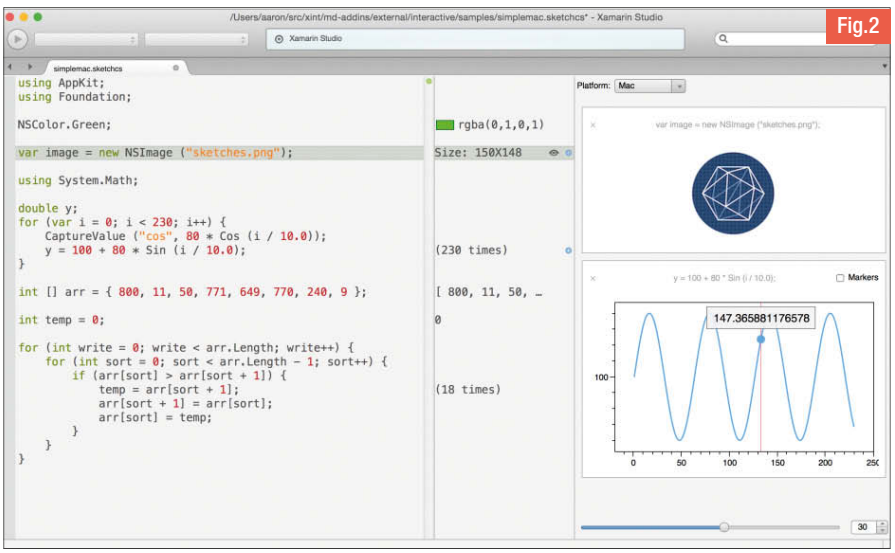
Fig.1



Les améliorations apportées à Test Cloud : Réduction des coûts et des temps d'exécution !

Cette keynote a réellement été riche en contenu car d'autres nouveautés ont également été annoncées. Nous avons notamment appris que Test Cloud, la solution de Xamarin permettant d'automatiser les tests d'applications (en les exécutants, réellement, sur environ 1000 devices iOS/Android) avait connue des améliorations.

Fig.2



ikoula
HÉBERGEUR CLOUD

LE CLOUD GAULOIS, UNE RÉALITÉ ! VENEZ TESTER SA PUISSANCE

EXPRESS HOSTING

Cloud Public
Serveur Virtuel
Serveur Dédié
Nom de domaine
Hébergement Web

sales@ikoula.com
01 84 01 02 66
express.ikoula.com

ENTERPRISE SERVICES

Cloud Privé
Infogérance
PRA/PCA
Haute disponibilité
Datacenter

sales-ies@ikoula.com
01 78 76 35 58
ies.ikoula.com

EX10

Cloud Hybride
Exchange
Lync
Sharepoint
Plateforme Collaborative

sales@ex10.biz
01 84 01 02 53
www.ex10.biz

Parmi les grosses nouveautés, on a découvert qu'il était désormais possible d'enregistrer l'écran des différents devices qui exécuteront l'application dans Test Cloud.
Ainsi, si un test échoue, vous pouvez rejouer la vidéo pour tenter de voir où, comment et pourquoi ce test n'a pas été en mesure d'aboutir.
De plus, il est maintenant possible de paralléliser l'exécution des tests qui sont lancés, via Test Cloud, sur les différents devices. Si cela peut sembler ne pas être important, c'est tout le contraire car l'utilisation de Test Cloud étant facturée au nombre d'heures utilisées, cela permet d'optimiser les temps d'exécution des tests et, par conséquent, de réduire la note.
A titre d'exemple, les équipes de Xamarin.Forms utilisent, en interne, Test Cloud pour exécuter tout un ensemble de tests.
Là où il leur fallait 2h30 pour exécuter ces tests, il ne leur faut plus maintenant que 12 minutes !

Xamarin Insights, ou l'identification des problèmes applicatifs

Enfin, la dernière grosse nouveauté annoncée a été **Xamarin Insights**, un système de monitoring, en temps réel, des applications iOS, Android, Windows, permettant d'identifier et de noter les problèmes qui impactent les utilisateurs : Fig.4.
Xamarin Insights se veut une API extrêmement simple permettant de traquer et identifier les différents bugs pouvant survenir dans l'application. Une fois un bug découvert, des connecteurs sont disponibles pour GitHub, HipChat, Visual Studio Online, etc. afin de permettre aux développeurs, d'être notifié lors de soucis rencontrés.
Xamarin Insights se retrouve ainsi composé de deux éléments :

- Une API, disponible sur Nuget, permettant de traquer les bugs et de les envoyer au serveur,
- Un portail Web, permettant de visualiser et analyser les bugs remontés.

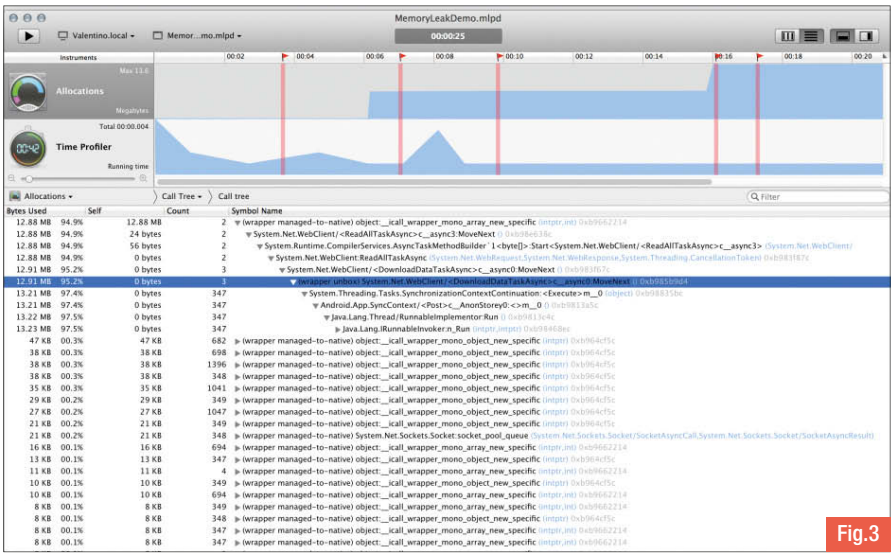


Fig.3

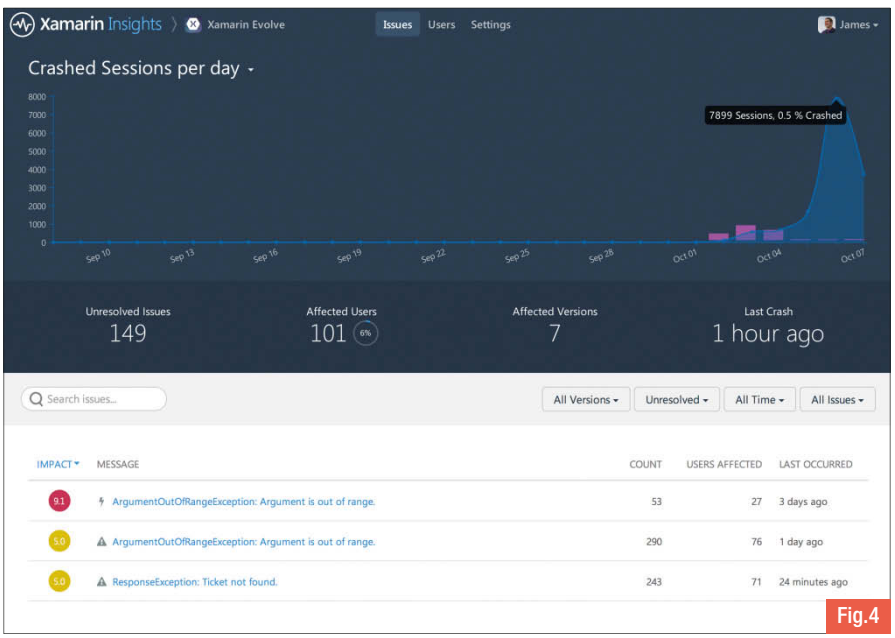


Fig.4

Des annonces pleines d'avenir pour le développement multiplateformes

La Keynote de cette conférence Evolve 2014 a apporté un grand nombre de nouveautés. Certes, beaucoup sont encore en Preview mais cela offre aux développeurs la possibilité d'y voir plus clair sur ce que prépare Xamarin et ce sur quoi ils ont travaillé ces derniers mois. Une chose est sûre, les nouveautés annoncées semblent toutes prometteuses, reste à voir maintenant ce que les équipes de développement nous réservent d'autre pour la suite !

Thomas **LEBRUN**
Consultant Infinite Square
tlebrun@infinitesquare.com
http://blogs.infinitesquare.com/b/tom
http://blog.thomaslebrun.net
@thomas_lebrun

INFINITE SQUARE

TOUT PROGRAMMEZ! SUR VOTRE DESKTOP, TABLETTE ET SMARTPHONE !
Windows 8
Windows Phone 8
Et bientôt sur Android

Montée en charge linéaire et extrêmement performante

NCache™
100% .NET

Pour applications .NET et Java
(supporté sur Windows Azure et Amazon AWS)

TayzGrid™
100% Java

Les données en cache (via NCache), réduisent les accès coûteux en base, et permettent à vos applications de monter en puissance vers “extreme transaction processing” (XTP). TayzGrid est une implémentation native 100% Java de NCache.

Cache distribué en mémoire

- Extrêmement rapide et montée en charge linéaire avec 100% uptime
- Topologie en Miroir, Répliquée, Partitionnée et Cache Client
- NHibernate et Entity Framework cache niveau 2

Optimisation ASP.NET de Web Farms

- ASP.NET Session State cache
- ASP.NET View State cache
- ASP.NET Output Cache provider

Memcached Wrapper Fourni

Partage de données en mode Runtime

- Notifications d'événements puissants pour le partage de pub / sub données

Alachisoft
sales@alachisoft.com
US: +1 (925) 236 3830
Siège de l'entreprise

Télécharger un essai GRATUIT!
www.alachisoft.com

RedFabriQ
info@redfabriq.com
Tel: +33 1 40 16 07 89
Distributeur et intégrateur en France

novembre

DevFest Nantes 2014 : 7 novembre

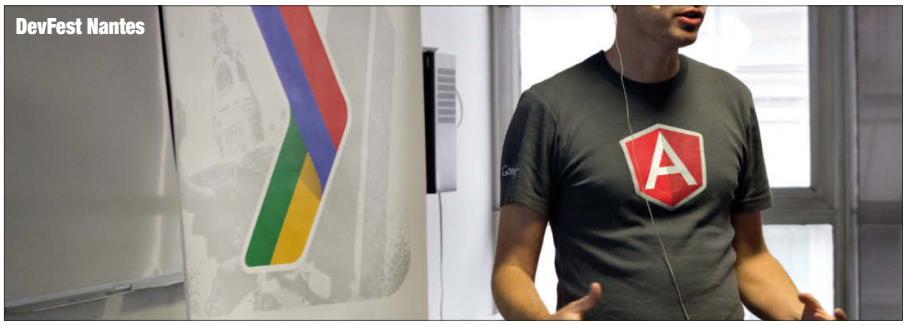
Le Google Developers Group de Nantes organise sa journée développeur et technologies Google. C'est le rendez-vous incontournable de la région. Les DevFests rassemblent des « speakers » nationaux et internationaux, soit des experts Google ou des experts locaux, expliquant leur savoir faire durant des conférences. C'est une occasion pour les participants de côtoyer cet écosystème et de pouvoir discuter en direct avec les acteurs, voire les créateurs de ces technologies. Sont annoncés cette année :

- Brian Dorsey, équipe Cloud chez Google Seattle
- Ludovic Champenois, équipe Cloud chez Google San Francisco
- Cyril Mottier, Google developper expert Android
- Frédéric Harper, évangéliste web chez Mozilla

Le programme de la journée est composé de 24 sessions réparties sur 4 thèmes :

- Mobile : Sujets tournant autour des technologies mobiles (Android Wear, Android, FirefoxOS, ...)
- Web : Sujets tournant autour des technologies web (Angular, Dart, WebComponents, ...)
- Cloud : Sujets tournant autour de la Cloud Platform de Google (AppEngine, Docker, ...)
- Découverte : Sujets visant à offrir une programmation non technique (UX, Diversité, Design, ...)

Pour tout savoir : http://devfest.gdgnantes.com



Green Code Lab Challenge 2014

L'événement de l'éco-informatique aura lieu du 26 au 28 novembre. Comment faire un code et un logiciel écoresponsables ? Toutes les réponses au Green Code Lab ! site : http://www.greencodelab-challenge.org/GCL2014/

Drupagora 2014

L'événement français Drupal se déroulera le 14 novembre à Paris. Ce sera déjà la 4e édition. Site : http://www.drupagora.com/

dotJS + dotCSS

Le 15 novembre, une double journée technique sur JavaScript et les CSS. Les ateliers seront les vedettes de cette journée. Site : http://www.dotjs.eu/workshops

Epitech Innovative Projects

15-16 novembre

Comme chaque année, Epitech expose sur son campus les nombreux projets des étudiants de l'école. Une parfaite occasion pour découvrir les idées et innovations des futurs ingénieurs en informatique. Toutes les informations sur : http://www.epitech.eu

Mobile Dev Day à Mons (Belgique)

Le 27 novembre, venez découvrir le développement mobile et embarqué sur les plateformes Microsoft. Les principaux thèmes seront : mobilité, interface et design, cloud computing et internet des objets. 11 sessions seront jouées durant la journée. Pour en savoir plus : ww.mobiledevday.be

Conférence OW2 les 5 & 6 novembre

Comme chaque année OW2 organise sa grande conférence annuelle. Cette année, OW2Con organise aussi la conférence Open Cloud Day, parallèlement à l'OpenStack Summit. L'événement se déroulera dans les locaux de l'Orange Labs (Issy-les-Moulineaux).
Pour en savoir plus : www.ow2.org

OW2 *The open source community for infrastructure software*

Tour de France PC Soft 2014

PC Soft dévoilera les versions 20 de WinDev, WebDev et WinDev Mobile durant son tour de France annuel. L'événement se déroule l'après midi, à partir de 13h45.
Les dates de novembre : Montpellier (4), Marseille (12), Lyon (13), Toulouse (18), Bordeaux (19), nantes (20), Paris (25), Lille (26), Bruxelles (27).
Les dates de décembre : Strasbourg (2), Genève (3).
Toutes les informations sur http://windev20tdf.fr

Sessions novembre & décembre

Zenika vous propose pour les mois de novembre et de décembre une sélection de sessions et d'événements techniques :

Jeudi 6 novembre à partir de 19h : NightClazz organisée par Zenika sur le thème de Java 8. Inscriptions à venir. Site : http://zenika.com/Evenements/nightclazz.html

Mardi 18 novembre de 9h à 18h : MongoDB Day chez Zenika Paris organisé par MongoDB : http://www.mongodb.com/events/mongodb-paris-2014

Jeudi 4 décembre à partir de 19h : NightClazz Java 8 avancée. Inscriptions à venir. Organisé par Zenika.

Jeudi 11 décembre à partir de 9h : Club(21 organisé par Rupture(21 sur la transformation d'organisations agiles. Inscriptions à venir : http://rupture21.com/club21/

NOUVEAUX SERVEURS BUSINESS LINE

La confiance est un critère important pour le choix d'un serveur. Avec 13 années d'expertise serveur, plus de 6 000 collaborateurs dans 11 pays, 70 000 serveurs hébergés et 7 data centers haute performance qui garantissent la sécurité de vos données, nous sommes votre partenaire de confiance !

NOUVEAU !

Serveurs dédiés Business Line X8i et X10i avec Hardware Dell™ PowerEdge™ R630

- Nouveaux processeurs Intel®Xeon® E5-2600 V3 jusqu'à 10 cœurs HT, 2,3 GHz et 128 Go de RAM DDR4
- Jusqu'à 6 To HDD, Hardware RAID 6 et disque dur optionnel Intel® SSD
- Raccordement 1 Gbit/s avec trafic illimité
- Sécurité maximale grâce à des composants redondants
- Connexion facile avec votre infrastructure de serveur Dell™ existante via Dell™ OpenManage Essentials

Retrouvez notre gamme complète sur 1and1.fr : des serveurs pour démarrer à partir de 19,99 € HT/mois* aux serveurs professionnels pour les plus hautes exigences.

BUSINESS LINE by Dell™

À partir de

199,99 € HT/mois (239,99 € TTC)*

1 MOIS POUR ESSAYER

1 CHOIX VOTRE DURÉE D'ENGAGEMENT

1 APPEL UN EXPERT VOUS RÉPOND

0970 808 911
(appel non surtaxé)

1&1

1and1.fr

* Le serveur dédié X8i est à partir de 199,99 € HT/mois (239,99 € TTC) pour un engagement minimum de 6 mois. Le serveur dédié L2 est à partir de 19,99 € HT/mois (23,99 € TTC) pour un engagement minimum de 6 mois. À l'issue des 6 premiers mois, les prix habituels s'appliquent. Frais de mise en service : 99 € HT (118,80 € TTC) pour le serveur X8i, 29 € HT (34,80 € TTC) pour le serveur L2. Offres également disponibles avec une durée d'engagement minimum de 12 mois ou sans durée d'engagement minimum. Conditions détaillées sur 1and1.fr. Dell, le logo de Dell, l'emblème de Dell et PowerEdge sont des marques protégées appartenant à Dell.

écoconception web
Interface : du bon usage des feuilles de styles

L'utilisation judicieuse des CSS et leur optimisation réduit efficacement l'empreinte ressources. De bonnes pratiques qu'il faut intégrer dès le début du projet et ne surtout pas reléguer au rang d'optimisations finales.

L'intégration, qui consiste à traduire la maquette d'un site web en une interface graphique utilisateur fonctionnelle basée sur HTML5, CSS3 et Javascript, est une étape importante en matière d'écoconception web. Nombre de choix qui seront faits à ce niveau vont avoir un impact déterminant sur la bande passante consommée, sur le poids total des images, sur la consommation CPU du navigateur, etc.
Les bonnes pratiques à mettre en œuvre ne sont pas compliquées. Mais elles sont trop souvent reléguées au rang d'optimisations à réaliser en fin de projet. Or, comme nous le savons tous, ces optimisations sont rarement appliquées car l'urgence en fin de projet est souvent de livrer... pas trop en retard. Il faut donc absolument intégrer les bonnes pratiques que nous vous présentons ci-dessous dès le début de l'intégration, dans un esprit d'intégration continue. Cet article se limite au code xHTML, aux feuilles de styles (CSS) et aux images qui constituent l'interface utilisateur. L'article suivant abordera les bonnes pratiques liées aux interactions, notamment via l'exécution de code Javascript.

Favoriser un design simple, épuré, adapté au web

Trop d'intégrateurs sont écartés de la conception graphique du projet. Or, leur implication en amont, notamment via un dialogue avec les graphistes et ergonomes, est la seule façon de s'assurer que le design du site s'appuie sur le potentiel de HTML5 et CSS3. Malheureusement, trop souvent mis devant le fait accompli, les intégrateurs doivent jouer de ruses et de bidouilles pour fournir une interface utilisateur conforme aux rêves les plus fous des graphistes. Il est donc essentiel de créer ce dialogue le plus tôt possible.

Préférer les CSS aux images

Le transport d'une image depuis le serveur et son affichage dans le navigateur consomment bien plus de ressources que le recours à un style CSS3. Il faut donc privilégier autant que possible les propriétés CSS3 aux images. Par exemple, à la place de :

```
#gauche {
background:#E4EFFF url(images/fond-arrondi.png) no-repeat bottom left;
margin:auto;
max-width:2007px;
}
#droite {
background:#E4EFFF url(images/fond-arrondi.png) no-repeat bottom right;
margin-left:7px;
padding-bottom:20px;
}
#haut {
background:#E4EFFF url(images/fond-arrondi.png) no-repeat top right;
margin-left:-7px;
padding:0;
}
#haut div {
background:#E4EFFF url(images/fond-arrondi.png) no-repeat top left;
height:7px;
width:7px;
}

<div id="gauche">
 <div id="droite">
  <div id="haut">
   <div></div>
  </div><!-- /haut -->
  <p>
  Lorem ipsum dolor sit amet, consectetur adipiscing elit.
  </p>
 </div><!-- /droite -->
</div><!-- /gauche -->
```

Utiliser :

```
#cadre {
  border-radius: 10px;
}

<dvi id="cadre">
  <p>
  Lorem ipsum dolor sit amet, consectetur adipiscing elit.
  </p>
</div>
```

Découper les feuilles de style, mais pas trop #7, #9, #10, #14

Les codes CSS ne doivent pas être embarqués directement dans le code HTML de la page car il est alors impossible de le réutiliser. Pour éviter de transporter inutilement plusieurs fois les mêmes styles entre le serveur et le terminal de l'internaute, il faut donc factoriser les styles au sein d'une ou plusieurs librairie(s). Ces dernières seront stockées dans le cache du navigateur qui ne les téléchargera plus.
Il ne faut cependant pas non plus tomber dans l'excès inverse qui consiste à stocker tous les styles dans une seule librairie. Les styles peuvent être regroupés dans plusieurs librairies, en fonction du contexte. Par exemple, lorsque plusieurs éléments du DOM ont des propriétés CSS communes, il faut les déclarer ensemble dans la feuille de style. Cela aura pour effet de réduire le poids de la CSS, donc d'économiser de la bande passante et de réduire la charge CPU. Certains CMS et frameworks fournissent des solutions automatiques pour effectuer ce type d'optimisation. Il est également possible de s'appuyer sur le serveur HTTP. Le découpage des CSS est tout un art. De manière générale, on peut appliquer un découpage par grande fonctionnalité, dès que l'on sait qu'elle engage l'internaute dans une action spécifique. Par exemple, nous aurions tendance à créer une feuille de style spécifique à la page d'accueil. Or, quelle que soit la façon dont l'internaute arrive sur le site, il est très vraisemblable qu'il passe par la page d'accueil lors de la navigation. On peut donc inclure ces styles dans la CSS principale sans trop de risque. Dans le cadre d'un site e-commerce, il est pertinent de créer une CSS spécifique pour le panier, et éventuellement une seconde pour le processus de « checkout ». Ces deux actions ne seront pas systéma-

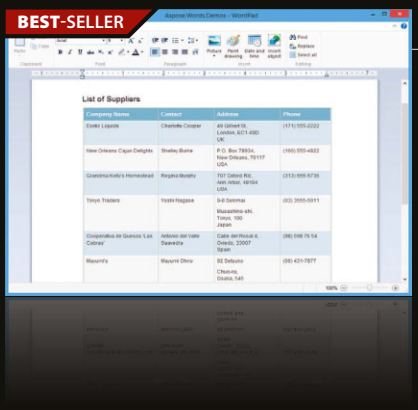
ComponentSource®
The Definitive Source of Software Components
www.componentsource.com

ComponentOne Studio Enterprise 2014 v2 | à partir de € 1 160

ComponentOne
a division of GrapeCity

Outils pour le Développeur Professionnel .NET : Windows, HTML5/Web et XAML.

- Contrôles d'interface utilisateur pour .NET, incluant grilles, graphiques, rapports et planificateurs
- Inclut 40+ widgets interface utilisateur crées avec HTML5, jQuery, CSS3 et SVG
- Nouveau Sparkline pour HTML5 et Web Forms et options pour l'exportation des vues de données
- 40+ composants pour Windows 8.1 & Windows Phone 8.1 (bêta) et support pour Universal Windows
- Supporte toutes les plateformes de Microsoft: Visual Studio 2013, ASP.NET, WinForms, WPF, etc



Aspose.Words for .NET | à partir de € 783

ASPOSE
Your File Format APIs

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

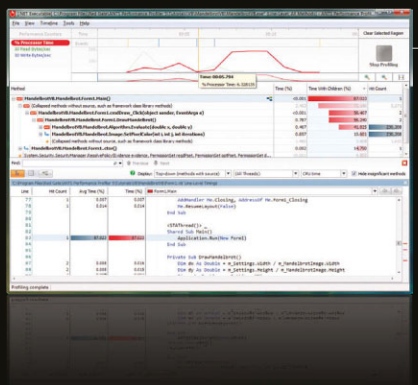
- Création de documents, manipulation du contenu/formatage, puissante capacité de fusion de courrier et exportation en DOC/HTML
- Accès détaillé à tous les éléments d'un document par programmation
- Support les formats de fichiers: DOC, DOCX, WordprocessingML, RTF, HTML, OOXML, OpenDocument, PDF, XPS,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

DevExpress DXperience | à partir de € 1 176

DevExpress

Tous les outils DevExpress ASP.NET, WinForms, Silverlight, WPF et IDE Productivity en un.

- Abonnement de 12 mois pour tous les produits et mises à jour DevExpress et accès aux versions bêta en développement actif
- Modèles et thèmes d'application intégrés exceptionnels
- Support de nouvelle vue Windows 8 UI et panneaux ancrables tactiles
- Support interface codée pour test environnement utilisateur



Red Gate .NET Developer Bundle | à partir de € 760

redgate®

Éradiquez et corrigez le code lent, identifiez le code .NET bogué et comprenez les raisons.

- Bénéficiez d'une vue d'ensemble des performances de vos applications et identifiez les goulots d'étranglement, dans le code ou la base de données
- Trouvez rapidement les fuites de mémoire et optimisez la mémoire de vos codes C# et VB.NET
- Comprenez et déboguez le code de tiers, frameworks, composants et bibliothèques inclus
- Standardisez la gestion des performances de votre équipe de développement

© 1996-2014 ComponentSource. Tous droits réservés. Tous les prix sont corrects au moment de la presse. Prix en ligne mai différentes de celles décrites en raison de fluctuations quotidiennes et remises en ligne.

Siège social en Europe
ComponentSource
30 Greyfriars Road
Reading
Berkshire
RG1 1PE
Royaume-Uni

Siège social aux États-Unis
ComponentSource
650 Claremore Prof Way
Suite 100
Woodstock
GA 30188-5188
États-Unis

Siège social au Japon
ComponentSource
3F Kojimachi Square Bldg
3-3 Kojimachi Chiyoda-ku
Tokyo
Japon
102-0083

Numero vert:
0800 90 92 62
www.componentsource.com

Nous acceptons les bons de commande. Contactez-nous pour demander un compte de crédit.

VISA MasterCard AMERICAN EXPRESS EUROCARD MasterCard

tiquement réalisées par l'internaute. Un découpage comme celui-ci est donc pertinent : `main.css`, `cart.css` et `checkout.css`. Exemple de découpage :

```
/* ------------- */
/* main.css      */
/* ------------- */

/* Common styles */
h1, h2, h3 {
  font-family: serif;
}

p {
  margin: 5px 0 15px 0;
}

.product {
  border: 1px solid #eee
  height: 80px;
  width: 50px;
}

/* Home styles */
.home .product {
  float: left;
}

.product.featured {
  width: 100%;
}

.slider {
  height: 300px;
  width: 940px;
}

/* ------------- */
/* cart.css      */
/* ------------- */

/* Cart styles */
.cart {
  border: 1px solid blue;
}

.cart tr {
  border-bottom: 1px solid #ddd
}

/* ------------- */
/* checkout.css      */
/* ------------- */

/* Cart styles */
.address-fields {
  padding: 3px 5px;
}

.payment-type-selector a {
  float: left;
}

.credit-card-number,
.credit-card-expiration-date,
.credit-card-cvv {
  border: 2px solid orange
}
```

Fournir une CSS print #16.

Eco-concevoir un site ne se limite pas à réduire la consommation de ressources sur le terminal de l'internaute, le réseau et côté serveur. On s'intéresse aussi aux impressions, surtout pour les sites web. Vous avez tous fait cette expérience désagréable qui consiste à cliquer sur le bouton imprimer de votre navigateur. Plutôt que d'obtenir une copie de l'écran sur une seule feuille de papier, ce sont souvent 5, 6 et même jusqu'à 8 pages qui sont imprimées. Vous pouvez éviter ce désagrément en proposant une CSS « `print` » à vos utilisateurs. Cette dernière est la plus dépouillée possible. Elle masque le header, le footer et le menu. Elle supprime toutes les images sauf celle du contenu. Elle masque aussi les contenus "marketing" : sidebar, etc. Lorsque c'est possible, elle utilise aussi une police de caractère économe en encre telle que Century Gothic ou Garamond, Fig.1 et 2.

Optimiser les CSS #12, #13, #15, #17

La façon dont vous écrivez vos feuilles de styles a un impact sur l'efficience finale de votre site web. Il faut notamment écrire des sélecteurs CSS efficaces, basés sur des IDs ou des classes. Ils seront ainsi filtrés plus rapidement, économisant du temps CPU à la machine interprétant les règles. On peut également économiser de la bande passante (et donc du temps de chargement) en travaillant sur le poids des CSS. A la

Fig.1

L'impression d'un écran génère 6 pages à imprimer. Elles sont illisibles.


NOUVELLE VERSION
920
NOUVEAUTÉS
WINDEV
PLATEFORME INTÉGRÉE DE DÉVELOPPEMENT
WINDOWS, .NET, RAD JAVA, ...
20
PCSOFT
DÉVELOPPEZ 10 FOIS PLUS VITE
WINDEV : Logiciel professionnel de développement multi-plateformes
Développements natifs
Java
iOS
www.pcsoft.fr
Des centaines de références sur le site
Version 20 annoncée. Logiciel professionnel. Document non contractuel

Fig.2

7/7/2014 Sites web : à nouveau statiques ?

Publié sur *GreenIT.fr* (http://www.greenit.fr)

Sites web : à nouveau statiques ?

Par *fbordage*
Créé le *25/06/2014 - 07:27*

Si vous avez, comme moi, développé votre premier site web il y a 20 ans, vous serez certainement très intéressés par une tendance qui ne cesse de s'amplifier : le retour aux sites web statiques.

Au commencement du web, les serveurs d'applications - qui permettent de créer dynamiquement (c'est-à-dire à la volée) une page web – n'existaient pas. Les sites étaient développés à la main dans un simple éditeur de texte ou bien générés à l'aide de scripts, parfois à partir d'une base de données. Cette architecture minimaliste garantissait une excellente montée en charge, même pour de très grosses volumétries.

Puis, après avoir commencé à dynamiser le web avec des scripts de type *.cgi, les premiers serveurs d'applications - ASP, Cold Fusion, J2EE, et PHP - sont apparus à la fin des années 90.

Depuis, l'architecture technique des sites web n'a cessé de se complexifier. Aujourd'hui, un simple blog nécessite le plus souvent un serveur web, un serveur d'application, une base de données et un CMS (Drupal, ezPublish, Wordpress, etc.) pour afficher « Hello world ! ».

Retour aux sources, mais nouveaux outils

Face à cette débauche de moyens et aux problèmes techniques rencontrés par des sites web à forte volumétrie, une frange entière de (jeunes) développeurs reviennent aux sources : ils génèrent à nouveaux des sites web statiques.

Si le résultat final est identique, les outils et les mentalités ont évolués. L'état d'esprit est de faire le plus sobre possible. Par exemple, le serveur web se résume souvent à GitHub Pages ou S3 d'Amazon (voir également BitBalloon) lorsque le site est très sollicité. Ce dernier propose de bons temps de réponse pour coût d'hébergement quasi nul.

Le site est généré à partir de moteurs comme Jekyll, Octopress, Middleman, Pelican, etc. Ces derniers génèrent les pages statiques à partir d'un ou plusieurs template(s) écrits en markdown (ou autre). Des éditeurs dédiés tels que Prose.io permettent de générer les templates sans connaissance spécifique.

Les traitements complexes ou la mise à jour dynamique des données sont déportés dans le navigateur (en Javascript) et sur des services en ligne via leur API REST. Par exemple : la mise à jour d'un cours de bourse ne se fera plus en rechargeant une page web crée dynamiquement mais plutôt en appelant une données en REST via un composant javascript statique basé sur une architecture AJAX.

Une approche minimaliste, en phase avec l'écoconception logicielle

http://www.greenit.fr/print/article/logiciels/sites-web-a-nouveau-statiques-5216 1/2

Le recours à une CSS print limite le nombre de pages à 2. Elles sont parfaitement lisibles.

source, vous pouvez par exemple utiliser les notations CSS abrégées pour réduire le poids de la feuille de style. Avant de passer en production, la minification des librairies CSS, avec un outil tel que YUI Compressor, permet de supprimer les espaces et sauts de lignes inutiles. Le module Apache PageSpeed de Google permet aussi d'automatiser cette opération. Mais cette approche est moins intéressante puisqu'elle est répétée autant de fois que la CSS est demandée. Mieux vaut donc privilégier une minification statique. Enfin, si des ressources spécifiques doivent être transmises à certains navigateurs, utilisez les commentaires conditionnels afin que les autres navigateurs ne téléchargent pas inutilement ces ressources.
Exemple de feuille de style non optimisée, puis optimisée, puis minifiée :

```
/* Sélecteur CSS efficaces */

/* Pas optimisée */
div.product a img.teaser {
  padding-left: 4px;
  padding-right: 4px;
  padding-top: 10px;
  padding-bottom: 20px;
}

div.related-prodcut a img.teaser {
  padding-left: 4px;
  padding-right: 4px;
  padding-top: 10px;
  padding-bottom: 20px;
  height: 20px;
  width: 20px;
}

html.ie6 div.related-prodcut a img.teaser {
  width: 12px
}

/* Optimisée */
.product .teaser,
.related-prodcut .teaser {
  padding: 10px 4px 20px 4px;

}

.related-prodcut .teaser {
  height: 20px;
  width: 20px;
}

.ie6 .related-prodcut .teaser {
  width: 12px
}
```

Le sélecteur .ie6 correspond à ce commentaire conditionnel (règle #17)

```
<!--[if lt IE 7]> <html class='ie6'> <![endif]-->
<!--[if IE 7]> <html class='ie7'> <![endif]-->
<!--[if IE 8]> <html class='ie8'> <![endif]-->
<!--[if gt IE 8]><!--> <html> <!--<![endif]-->
```

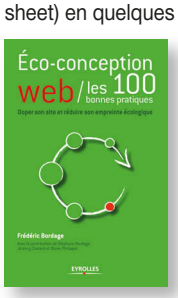
/* Version minifiée de Do, règle #12 */

```
.product .teaser,.related-prodcut .teaser{padding:10px 4px 20px 4px}.related-prodcut .teaser{height:20px;width:20px}.ie6 .related-prodcut .teaser{width:12px}%
```

Au-delà de l'optimisation des ressources elles-mêmes, on peut aussi s'attacher à respecter quelques bonnes pratiques supplémentaires comme l'optimisation du favicon.ico et sa mise en cache (ce fichier est appelé par le navigateur sur toutes les pages, quoi qu'il arrive). Il faut aussi supprimer les balises images dont l'attribut SRC est vide. En effet, si une balise image est présente et que son attribut "src" est vide, le navigateur appelle la page d'index du niveau d'arborescence où il se situe, générant des requêtes HTTP supplémentaires... et inutiles. Et, évidemment, ne pas redimensionner l'image dans le code HTML, mais plutôt une bonne fois pour toutes avant de passer en production. Des évidences qui sont malheureusement souvent oubliées...

Regrouper les images dans un sprite #18

Pour éviter des allers-retours incessants entre le navigateur et le serveur HTTP (qui allongent le temps d'affichage du site tout en chargeant inutilement l'infrastructure serveur), il faut regrouper les images de l'interface dans une seule. Cette dernière est ensuite découpée dynamiquement via une feuille de style. Le surcoût en ressources côté client (mémoire vive et cycles processeurs nécessaires pour découper dynamiquement l'image sprite) est largement compensé par l'économie de bande passante, et le fait de moins solliciter les serveurs. Des dizaines de services en ligne sont capables de générer le sprite et la feuille de styles associés (sprite-sheet) en quelques secondes, gratuitement. Autant en profiter !



Frédéric **Bordage**, *expert écoconception logicielle, GreenIT.fr,* @greenit
Jérémy **Chatard**, *directeur technique de Breek,* @breekfr

Retrouver ces bonnes pratiques dans le livre : Eco-conception

Bienvenue dans l'univers du code

PROGRAMMEZ!
le magazine du développeur

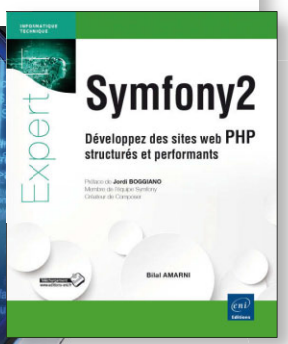
Offre spéciale(1)

Programmez! et les éditions ENI vous plongent dans l'univers du PHP

pour 1€ de plus

(valeur du livre : 29,26 €
livre numérique,
email indispensable)

(1) Valable uniquement pour la France métropolitaine. Quantité limitée. L'abonnement doit être envoyé avant le 29 novembre



© nuage : 07-17-13 © frankpeters - intérieur : 07-25-13 © Henrik5000

Spécial étudiant 39€(*)
1 an 11 numéros

1 an 11 numéros 49€
seulement (*)

2 ans 22 numéros 79€
seulement (*)

(*) Tarifs France métropolitaine

Toutes nos offres sur www.programmez.com

Oui, je m'abonne

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

☐ ***Abonnement 1 an au magazine + livre numérique ENI :*** 50 € (au lieu de 65,45 € / prix au n° + 29,26 €) / abonnement seul : 49 €

☐ ***Abonnement 2 ans au magazine + livre numérique ENI :*** 80 € (au lieu de 130,90 € / prix au n° + 29,26 €) / abonnement seul : 79 €

☐ ***Abonnement spécial étudiant 1 an au magazine + livre numérique ENI :*** 40 € (au lieu de 65,45 € / prix au n° + 29,26 €) / abonnement seul : 39 €
Photocopie de la carte d'étudiant à joindre

☐ M. ☐ Mme ☐ Mlle Entreprise : Fonction :

Prénom : Nom :

Adresse :

Code postal : Ville :

Tél : ***(Attention, e-mail indispensable pour les archives sur internet)***

E-mail : @

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

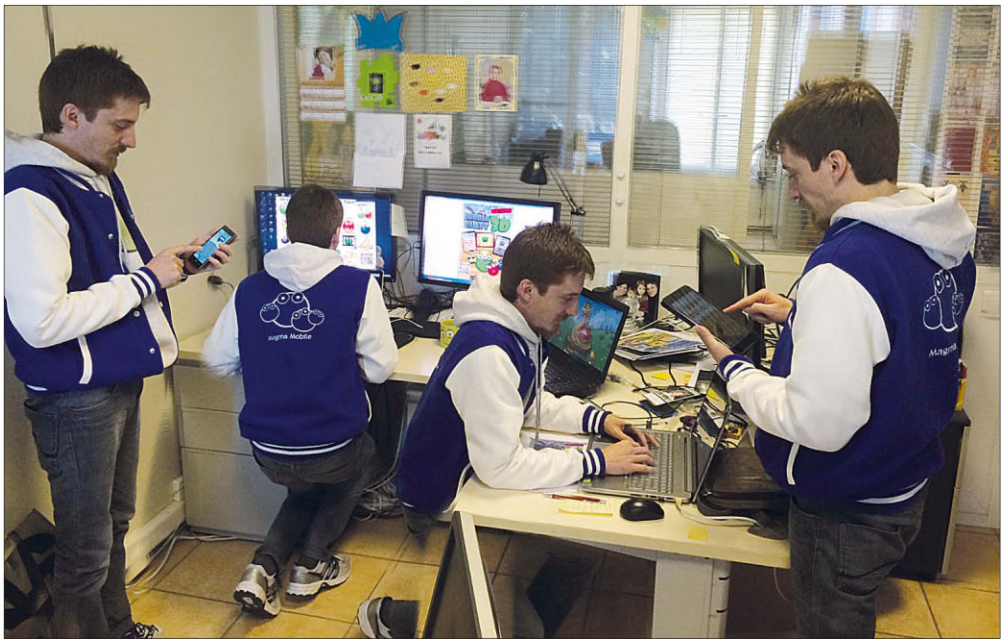
☐ Je souhaite régler à réception de facture

** Tarifs France métropolitaine*

PROG 179
***Offre limitée**, valable jusqu'au 29 novembre 2014*

Magma Mobile : « être là au bon moment »

Magma Mobile est une belle réussite française dans le jeu vidéo sur smartphone (Android, iOS, Windows Phone) : plus de 130 jeux disponibles, plus de 380 millions de téléchargements, des joueurs dans le monde entier. Nicolas Sorel revient pour Programmez ! sur cette aventure débutée il y a plus de 20 ans.



Nicolas Sorel, alias Nix dans les communautés de développeur, s'est fait connaître avec le site Codes-Sources. « J'ai débuté l'informatique et la programmation dès mes 9 ans avec un Amstrad 6128. J'aimais les jeux. Et je me suis lancé très vite dans le code. Je n'ai pas fait d'études. J'ai fait de nombreux jobs ici et là, notamment vendeur de sandwichs. Puis j'ai trouvé un bac pro technicien informatique » raconte Nix. Mais durant ces années, la programmation ne l'a jamais lâché ! Il a travaillé plus de deux ans dans une entreprise pour y gérer l'informatique et le parc. Notre « serial entrepreneur » va créer sa première société dès l'an 2000, « pour vendre des DVD, mais le marché n'était pas encore mûr. » poursuit Nicolas. Puis, il va créer plusieurs autres sociétés dans le développement de sites Web. « Dès 1999, je crée un nouveau site Web qui deviendra Codes-Sources. J'ai appris tout seul à développer et j'en avais assez de toujours trouver des ressources, des codes en Anglais. Au départ, j'avais créé Codes-Sources pour mes besoins... » se rappelle Nix. Codes-Sources allait grossir peu à peu et regrouper de plus en plus de langages et d'outils, avant de devenir une des références des développeurs. Mais en 2005, il fallait faire un choix. Ne pouvant mener deux fronts en parallèle, la société de développement et Codes-Sources, le choix se porta sur la communauté. « Je n'imaginais pas en faire un métier. » se rappelle Nix.

« Voir les opportunités »

À partir de 2007, le smartphone allait devenir la révolution que nous vivons aujourd'hui. L'iPhone venait tout juste de sortir et le succès était encore très timide. Google travaillait à son propre système mobile : Android. « J'ai eu le déclic en octobre 2008 en voyant une démo. J'ai été immédiatement séduit ! Très rapidement, nous avons regardé le SDK d'Android. J'avais à cette époque des terminaux Windows Mobile avec lesquels je ne faisais rien. » poursuit Nicolas. Des opportunités ont rapidement été vues par Nicolas et Emmanuel, son partenaire depuis plusieurs années. En quelques mois, les développeurs enregistrent des vidéos de formation Android, sortent un livre aux éditions Eyrolles et surtout créent un site inédit sur les apps Android : androlib. Nous sommes en septembre 2009... Le Rubicon est franchi ! Magma Mobile est créé toute fin 2009. Les premières embauches se font dans la foulée. Rapidement, la société compte cinq personnes. Mais Magma Mobile n'est pas le Magma Mobile d'aujourd'hui. Au départ, il s'agissait de développer pour des clients des applications Android. « Nous étions sans doute les seuls en France à faire ce genre de chose. Quand nous avons testé les terminaux G1, j'ai trouvé cela tellement génial, que Manu a codé très rapidement un jeu de flipper en 2D ! Résultat : le jeu existe toujours et il a eu plus de 30 millions de téléchargements ! » s'enthousiasme Nix.

Osons !

C'est début 2011 que Magma Mobile va devenir Magma Mobile ! « Aujourd'hui, ce serait impossible de refaire ce que nous avons fait en 2011. À l'époque, il n'y avait que 6000 apps, très peu d'éditeurs. C'était le bon timing pour faire quelque chose, bref, être là au bon moment. » recadre Nix. Et le bon moment était de proposer des jeux sur Android alors que tout le monde regardait vers iOS. Et les « tu es fou » n'ont pas tardé. Et pourtant, c'est la bonne idée. Rapidement, Magma Mobile s'installe dans de nouveaux locaux, double les effectifs pour atteindre rapidement une dizaine de personnes (aujourd'hui, Magma tourne à environ 15 personnes). Et les jeux commencent à sortir. Le succès ne tarde pas !

Miser sur le gratuit pour gagner de l'argent

« Notre business mobile est très simple : nos jeux sont gratuits. Ils intègrent des publicités, mais elles sont faites pour ne pas être agressives ou trop intrusives. C'est un point important. Je fais très attention à ce détail. » poursuit Nix. Actuellement, Magma Mobile propose 90 jeux sur Android, 50 sur iOS et 13 sur Windows Phone. Le rythme de sortie est assez soutenu : 1 à 2 jeux par mois. Et les équipes travaillent toujours sur 3-4 jeux simultanément. Tous les jeux sont traduits dans une vingtaine de langues, ce qui n'est pas toujours le cas chez les autres éditeurs, car la traduction prend du temps, nécessite des compétences et un certain budget. Magma Mobile cartonne sur Android avec plus de 350 millions de téléchargements contre 20-30 millions sur iOS. Ce différentiel s'explique par un lancement plus tardif sur la plate-forme Apple. « Mais nous ne voulons pas rentrer dans la bataille » précise Nix. Sur Windows Phone, le bilan n'est pas si mal : + 1 million de téléchargements par mois ! Mais sur cette plate-forme, la rentabilité est moindre, car le niveau de la publicité est moindre que sur les autres plates-

formes. Et les tarifs publicitaires sont là aussi moins élevés. La répartition des joueurs est assez atypique pour un éditeur français : 4 % en France ! Le gros des joueurs vivent en Russie, Brésil et États-Unis, la traduction systématique des jeux y est pour quelque chose, et les bons classements de ceux-ci dans les stores.

« L'important est de faire les choses bien et pas de l'approximatif »

Magma Mobile ne veut pas faire une surenchère technologique. Mais les développeurs testent régulièrement de nouvelles choses. « Nous avons développé une course de voiture en 3D. Il s'agit d'une belle vitrine, mais le retour sur investissement est très incertain. Nous testons aussi l'intégration de Facebook pour créer des défis dans nos jeux, jouer entre amis, etc. » recadre Nicolas. Mais des technologies telles que Facebook nécessitent une connexion constante au réseau, or, l'essence même des jeux de Magma est de pouvoir jouer de n'importe où, n'importe quand, 10 minutes, 30 minutes. Nicolas a d'ailleurs une belle expression : « jeux de salles d'attente ».
« Je n'ai pas une stratégie toute tracée. Le but est de faire ce que j'aime. Si on ne dévie pas, il y aura, un jour, un retour. Mais il faut être aussi réaliste, surtout dans le jeu. Des développeurs vont croire que tout est possible en sortant un jeu et qu'il sera un énorme succès alors qu'il sera téléchargé une centaine de fois. Le développeur abandonne, mais le jeu reste disponible ce qui pollue le marché finalement. Il y a beaucoup de jeux et beaucoup d'acteurs. Il ne faut pas croire qu'un jeu comme Candie Crush est le fruit du hasard, mais d'une machine parfaitement huilée. » martèle Nicolas. Mais il faut aussi de la rigueur et de l'organisation, car la sortie d'un nouveau système est toujours une source de tension. Le nouvel iOS a obligé à être très réactif et à vérifier que tout fonctionne correctement. Et XCode 6 a cassé du code qu'il a fallu corriger. Bien entendu, la veille technologique est importante même si finalement, tout le monde à Magma en fait inconsciemment. « Il faut tester les choses réellement pour voir, se faire une idée, voire, les opportunités éventuelles. » analyse Nicolas. Au début de Magma, un moteur graphique a été spécialement développé en interne pour Android. Mais avec l'arrivée des autres plateformes, il fallait redévelopper toute une partie du jeu pour le porter. Pour faciliter ces portages, il a été décidé d'utiliser la plate-forme Unity. « Nous attendons avec impatience Unity 5 qui intègre WebGL. On peut espérer de porter très rapidement des jeux sur le Web. Nous allons regarder cela de très près » précise Nix.

Un esprit Magma ?

« J'ai besoin de coder même si j'ai moins de temps. Mais je suis aussi un garde-fou en regardant par exemple des détails dans les jeux » sourit notre entrepreneur. Parfois les développeurs peuvent aller trop loin dans la technique et oublier le joueur... « Si un jour quelqu'un me remplace pour le suivi au quotidien des équipes, il faudra qu'il me ressemble, qu'il code encore régulièrement... » conclut-il.

François **Tonic**

INTERGRAPH®
http://www.intergraph.com

DEVELOPPEURS / INGENIEURS LOGICIEL H/F

Intergraph Corporation est une société américaine d'Ingénierie dédiée à la conception et à la vente de logiciels informatiques dans le domaine de la CAO. Pour sa division **Process Power & Marine**, Intergraph développe une ligne de produits CAO 3D, basés sur l'architecture Objet de Microsoft. Les marchés ciblés sont : la conception d'usines, la construction navale, la construction d'installations on shore et offshore.

MISSIONS

- Affecté à une équipe du Centre de Compétence d'Intergraph France à Rungis, vous participerez aux actions de recherche-développement de l'équipe en prenant en charge, en collaboration avec les ingénieurs-architectes, l'analyse et le développement de composants d'applicatifs pour la réalisation des logiciels. Utilisation de la méthodologie agile Scrum.
- Vous prendrez progressivement la responsabilité de la mise en œuvre du développement et la maintenance d'un ou plusieurs composants. Vous aurez à utiliser les technologies du marché (Microsoft .NET, VS2013, TFS ; Oracle 11g, MSQL server 2010 ; Spatial ACIS R21, HP Quick Test Pro ; Coverity ; Version One).

Vous aurez les objectifs suivants :

- Définition des besoins des clients.
- Développement et intégration de nouvelles fonctionnalités et tests automatisés pour garantir la qualité du logiciel.
- Gestion et optimisation de l'usage de la mémoire des logiciels.
- Amélioration des performances de calcul des produits.
- Validation de l'intégration du code par un processus de certification.

VOS ATOUTS

- Formation informatique de haut niveau, diplôme d'ingénieur option informatique.
- Maîtrise d'un ou plusieurs langages de programmation, bonne connaissance des outils de bases de données.
- Maîtrise de l'anglais oral et écrit.

REMUNERATION

Entre 35 et 45K bruts par an, selon expérience.

LIEU DE TRAVAIL

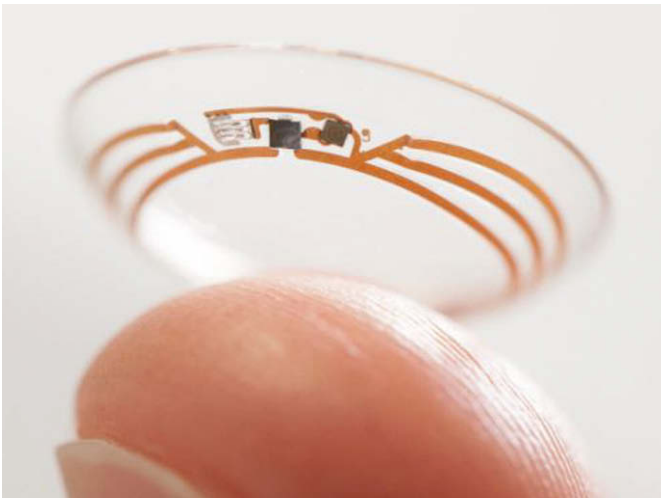
Rungis (94).

COMMENT POSTULER

Merci d'adresser CV + lettre de motivation à Mme Cordonnier à l'adresse e-mail : karen.cordonnier@intergraph.com.

Réalité augmentée, réalité immersive, nouvelles ergonomies : nous créons de véritables mondes parallèles

Le meetup du 24 septembre dernier de Paris Réalité Alternative, à l'initiative de Greg Madison, a été l'occasion de faire un point sur les nombreuses initiatives actuelles, et de découvrir quelques usages. Incontestablement, plusieurs mondes parallèles émergent peu à peu, à côté du monde physique et réel.



Si vous êtes passionné de SF ou d'anticipation, vous vous souvenez sans doute de la notion de projection des Cylons (Battlestar Galactica), du monde virtuel de Caprica, du holodeck de Star Trek Next Generation, de Tron, etc. Les exemples sont nombreux. Mais une constante est à garder à l'esprit : l'ordinateur n'est plus au centre du monde numérique. C'est déjà un peu le cas aujourd'hui, mais demain, notamment avec les objets connectés qui se multiplieront, ce sera encore plus vrai. Il est intéressant de constater qu'il existe plusieurs approches dans ces réalités dites alternatives :

- La réalité augmentée, typiquement Moverio BT-200, Meta Pro
- La réalité virtuelle/immersive, typiquement avec des casques/lunettes de type Occulus.

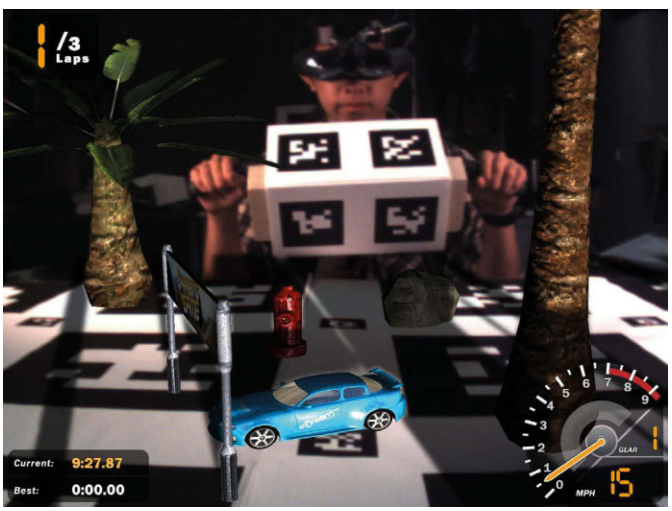
À cela se rajoute toute une effervescence sur les nouvelles interfaces, l'ergonomie naturelle. Cette réinvention de l'interface est nécessaire dans les réalités alternatives, les métaphores de nos interfaces n'ont plus lieu d'être. Car par définition, tout objet, tout élément du monde peut devenir une interface, un point d'accès vers « autre chose ». Nous pouvons faire la même remarque sur la souris qui est aujourd'hui concurrencée par le doigt (tactile), mais demain (et c'est déjà partiellement le cas aujourd'hui), ce sera la main, le doigt, les ondes cérébrales, l'œil, ou toute autre partie de son corps.

Réalité virtuelle, réalité immersive : Occulus Rift les offre au grand public !

Comme le meetup l'a parfaitement rappelé, le casque de réalité virtuelle Occulus Rift (racheté par Facebook) n'est pas le premier exemple. Cela fait plus de 30 ans que des casques de ce type sont construits, notamment pour la recherche et la simulation numérique (DARPA, NASA...). Nintendo avait échoué dans une première tentative grand public il y a 20 ans, mais sans doute qu'à l'époque, le concept était trop radical même si nous connaissions déjà Tron et d'autres succès de science-fiction.

Ce type d'appareillage permet d'être immergé (d'où le terme réalité immersive) dans un univers virtuel. C'était l'idée de départ. On navigue, on se déplace dedans. Une vision stéréoscopique est proposée pour construire. Nous ne voyons que ce que les écrans du casque nous proposent. L'expérience est souvent impressionnante. Et plus la qualité du monde virtuel sera poussée, plus on s'y prendra au jeu ! Aujourd'hui, Occulus Rift n'est qu'un des nombreux projets de casques existants. Malheureusement, aucune date de commercialisation n'est aucune annoncée fermement (2015, 2016 ?). Par contre, une certitude : ces casques révolutionnent déjà le prix de ces matériels. Mais, ils offrent aussi de nouvelles possibilités pour les jeux. Revers de la médaille, ces casques ne sont pas autonomes, ils nécessitent un ordinateur pour la puissance de calculs. Par contre, plusieurs casques alternatifs fonctionnent avec des smartphones. Un gros travail sur l'ergonomie générale est toujours en cours et afin d'améliorer et d'optimiser la taille du casque qui demeure encombrant et lourd.

D'autre part, on peut aussi observer que ces casques s'ouvrent désormais vers l'expérience, c'est à dire un monde réel ou semi-réel afin de mixer les deux réalités (physique – réel et virtuel/augmentée), notamment par des caméras. Et ces caméras ou toute autre extension permettent d'étendre les interactions et de reconnaître l'ensemble des mouvements du corps. La réalité augmentée fait désormais partie d'Occulus notamment dans la médecine. Antoine Rigitano a fait un point sur Occulus et sur les usages. Le jeu reste aujourd'hui le premier usage. Mais, d'autres usages se développent : cinéma, visite de musée ou d'un monument, visite immobilière, les simulations industrielles, classes virtuelles, la pornographie, les réseaux sociaux (dans des mondes virtuels),



l'entraînement sportif, usages militaires... Il n'y a pas de véritables limites.

Réalités alternatives : l'illusion devient réelle, le réel devient illusion

Greg l'a parfaitement rappelé durant sa présentation, l'informatique telle qu'on la connaît est en train de disparaître, ou plus exactement d'être de plus en plus invisible et de se transformer. Informatique est synonyme aujourd'hui de PC, de Mac, d'ordinateurs portables, de serveurs.

Trois mouvements bouleversent totalement notre rapport physique avec l'informatique, ou plus exactement le matériel informatique :

- Le cloud computing
- Le smartphone, la tablette
- Le wearable : l'informatique sur soi, à porter

Tout cela contribue à dématérialiser l'informatique, à réduire sa présence ou tout le moins à la transformer en autre chose. Et demain, tout objet sera potentiellement technologique, informatique. Ce sont les fameux objets connectés, l'Internet des objets. Votre tasse de café pourra être connectée à Internet pour fournir des informations sur votre activité et l'activité de l'objet, mais aussi il pourra devenir, de facto, un support numérique. Comme tout devient potentiellement numérique, surface numérique, on peut alors créer des mondes numériques qui vont se superposer au monde réel (et donc aussi au monde physique, notre univers palpable). Nous sommes alors dans une réalité mixte : réelle et augmentée. L'interface s'affiche sur les objets réels et s'adapte à chaque contexte, chaque situation. Cela nécessite des capteurs, des périphériques pour les afficher comme des lunettes (on pense à Google Glass), des capteurs sur les doigts, la tête, etc. Greg avait imaginé il y a quelques années un projet assez fou : 7th Sense. C'est un concept, mais qui montre parfaitement vers quoi on peut aboutir à terme. Impressionnant ! Mais ces mondes peuvent aussi exister et interagir avec des outils comme Kinect, Leap Motion ou des caméras de projection qui projettent l'interface, l'interaction sur les objets. Comme le dit si bien Greg, il s'agit de détourner les objets pour en faire autre chose, les utiliser autrement. Pour Greg, avec pertinence, les gens ne porteront pas des casques de réalités alternatives et n'iront pas dans la rue avec. Sans doute, faut-il chercher dans les notions de wearables, des périphériques moins intrusifs tels que les lunettes, des gants, etc. Et plusieurs projets cherchent à créer des lentilles intelligentes (intégrant des capteurs, voire, une

caméra). Là, nous nous rapprochons de l'Homme augmenté soit pour améliorer nos performances, soit pour redonner des facultés (marcher, voir, entendre) à une personne.

« Le paradigme de l'égosystème »

Greg Madison parle d'égosystème. Pour faire simple, tous les capteurs, les données, les objets connectés, les réseaux sociaux, etc. créent, construisent un monde numérique parallèle au monde réel. Bref, nous créons de facto des univers à côté de notre monde réel. Ces mondes parallèles existent déjà partiellement, il suffit de voir les reconstructions 3D de villes entières (Maps, Plan...), la modélisation 3D des logements par tout le monde (projet Tango de Google), etc. Ces matrices pourront ensuite servir à poser des interactions, des données que l'on pourra voir, utiliser. Savoir à tout moment ce qui se passe près de moi, voir si des amis sont disponibles, etc. Autre concept très intéressant, les Smart Object du MIT, les objets intelligents possédant des tags. Et quand on scanne ce tag, on accède en réalité augmentée aux fonctions de l'objet et on interagit avec lui. L'objet doit bien entendu comporter une intelligence et être relié au réseau numérique.

Metavers : un Internet habité

Autre intervention remarquée durant ce meetup, Nicolas Barrial qui évoque le metavers ou comment l'Internet peut être habité. Dans ce monde, on se déplace, on vit sous forme d'avatar, comme ce fut le cas avec Second Life. C'est un « monde » en devenir et qui se cherche un peu dans sa définition (lire notamment « ready player one » ou revoir la série Caprica, Matrix, Tron, Dark City). Nicolas donne une « définition » du metavers : simulation totale (du monde, de la vie), une matrice de réalités numériques, un ensemble de mondes virtuels et il est toujours en ligne et il est mémoriel (en principe). Il est possible de créer un metavers depuis un metavers. Le metavers repose (ra) sur un protocole. Mais comme le précise Nicolas, le metavers reste un monde humain avec les avatars de soi. Mais ce monde ne doit pas être fermé ni figé. Il faut donner la possibilité de l'étendre, de construire directement dans le metavers. Il doit aussi être capable de révolutionner tout le monde, d'évoluer. Il est intéressant de constater que le metavers sera influencé par le comportement humain et que ce ne sera pas une informatique sans âme, sans émotion. Passionnant ou flippant ?

Pour découvrir la conférence : https://www.youtube.com/watch?v=GHVA4t92so8

Des questions existentielles

Que l'on parle de réalité augmentée, de mondes numériques parallèles, d'Internet des objets, de metavers, il ne faut pas omettre la dimension morale et philosophique. Est-ce moral et éthique d'augmenter l'Homme ? Le metavers fait-il partie de l'Humanité ? Qui contrôle nos données ? Qui contrôle les technologies et les protocoles ? Va-t-on vers un monde tel que décrit par Blade Runner ? Autant de questions qui méritent des débats et une réflexion la plus neutre possible.

Sur le wearable et l'Homme augmenté, et même sur les réalités alternatives en général, quels impacts sur la santé ? Actuellement, nous n'avons aucun recul pour mesurer les conséquences à long terme d'une lentille augmentée ou d'un accessoire wearable. Sans vouloir être réactionnaire, il faudra se poser la question et surtout y répondre. Depuis 20 ans, il y a aussi le fait que les jeux vidéo généreraient des comportements asociaux, voire, violents. Là, encore, il faudra se poser la question. Google Glass a provoqué de nombreux débats. Ainsi, de nombreuses interdictions ont été promulguées soit par des autorités légales soit par des sociétés (cinéma, restaurant, etc.). Le législateur réagit souvent a posteriori. Mais surtout, il doit réagir sur quelque chose qu'il ne connaît pas et qui n'est pas forcément palpable. L'utilisateur doit-il laisser le contrôle de ses informations, de ses données à ses mondes numériques, voire, à des sociétés exploitant ensuite ses informations ? Un juste milieu devra être trouvé. Mais nous ne sommes pas encore à ce stade.

Pour en savoir plus

Paris Réalités Alternatives : http://www.gregmadison.fr

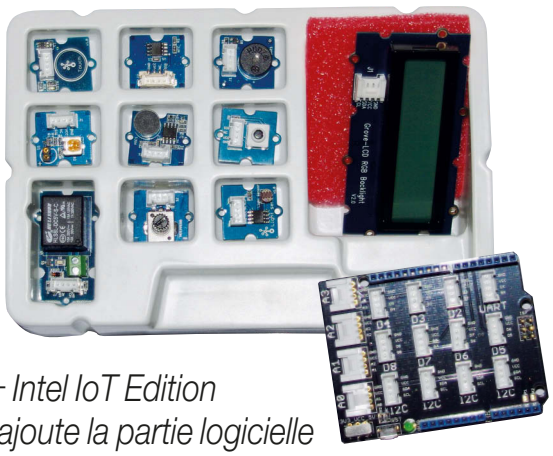
Softkinetic http://www.gregmadison.fr/2014/06/softkinetic-ou-matiere-virtuelle.html

Argus : capteur pour « voir » http://www.2-sight.eu/ee/home/10?black_white=true

worldbuilder; https://www.youtube.com/watch?v=VzFpg271sm8

François **Tonic**

IoT : les kits Intel pour créer ses propres objets connectés



Intel propose désormais aux bidouilleurs et aux développeurs plusieurs kits très complets : Galileo 2, Grove Starter Kit Plus – Intel IoT Edition (par Seed Studio) et enfin Edison. À cette partie matérielle se rajoute la partie logicielle pour les développeurs (SDK, librairies, outils).

Intel se montre très offensive sur les objets connectés, l'Internet des Objets (IoT) et les cartes intégrées de type Arduino, Raspberry Pi, etc. L'éditeur veut aller plus loin avec la nouvelle carte, Edison.

Galileo 2 : une belle carte

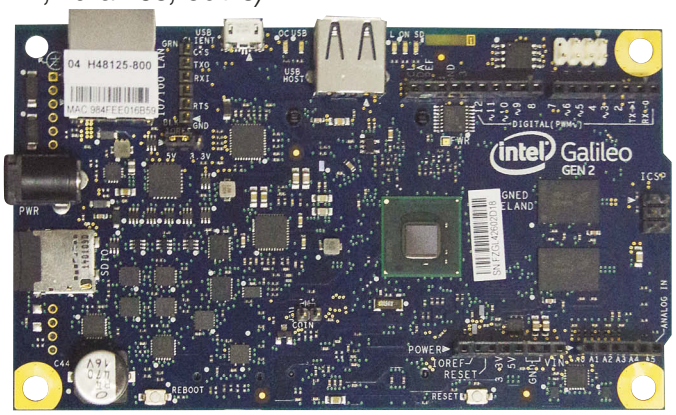
Intel a sorti il y a quelque temps une version 2 de sa carte Galileo. Cette carte est relativement chère : presque 90 € selon les magasins, la v1 s'achète aux alentours de 65 €, contre 20-25 € pour une carte Arduino Uno. Pour autant, la Galileo ne cherche pas à concurrencer frontalement les Arduino et autres Raspberry Pi, mais plutôt à construire sa propre communauté. Oui, Galileo est compatible avec les Arduino sur de nombreux éléments (Arduino IDE, de nombreux shields), mais non, ce n'est pas une « Arduino like ».

Si vous passez d'une génération 1 à la génération 2 (gén 2), vérifiez la compatibilité des codes et du matériel. La gén 2 introduit plusieurs modifications sur la carte, par exemple, un module USB TTL UART 6 broches 3,3 V remplace le port RS-332 prise jack pour le débogage Linux. Le régulateur de puissance accepte des alimentations de 7 à 15 V.

Cette gén 2 est toujours très bien finie. La qualité est au rendez-vous. Le package est plutôt agréable. Le contenu est très spartiate :

- La board (carte) gén 2
- Le bloc d'alimentation secteur

C'est tout. À vous d'avoir le reste à portée de main. Toute la documentation est accessible en ligne sur les sites Intel dédiés à Galileo et IoT. On aurait pu avoir



une carte micro SD et le câble micro USB.

Grove Starter Kit Plus – Intel IoT Edition : là on commence à s'amuser

C'est le complément indispensable à votre Galileo. Ce kit repose un shield à installer sur la board. L'intérêt du kit est de proposer différents capteurs (bouton, tactile, moteur, relais, leds, buzzer, capteur de luminosité, capteur de température, écran LED...). Il inclut tous les câbles nécessaires (mappes pour les capteurs, micro-USB, etc.). Le kit est très soigné dans le package et la qualité des capteurs. La documentation en ligne permet de rapidement créer de petits scénarios permettant maîtriser chaque capteur et le fonctionnement du shield, si on n'est pas habitués. Ce kit est proposé à 79 $ (attention à choisir la version adaptée à sa board).

Avertissement : Toute modification et soudure doivent se faire dans un environnement adapté et en respectant les consignes de sécurité. La modification d'un matériel est sous votre seule responsabilité et la garantie constructeur risque de ne plus être valable.

Edison : l'informatique à porter et à embarquer

Ce qui frappe avec Edison, c'est la dimension ridiculement petite de la carte. Galileo est un « géant » à côté. Edison utilise un SoC Quark. Le processeur embarqué est un Atom 2 cœurs. Il intègre la mémoire vive, le réseau et des interfaces GPIO (General Purpose Input/Output).

Edison est taillé pour le wearable (l'informatique sur soi comme les montres, les vêtements intelligents, etc.) et l'IoT, et tous les objets connectés. Côté langages de développement, nous retrouvons les classiques : C, C++, Node.JS, Python. Et Intel fait évoluer régulièrement ces supports. Edison peut être utilisé avec d'autres terminaux, se connecter à des services Cloud. Il est compatible Arduino et Galileo. Il supporte Arduino Sketch et Linux (Yocto). Côté outillage, rien que du classique : Arduino IDE, Eclipse, Intel XDK. Pour pouvoir développer et créer des scénarii d'usage, Edison ne suffit pas. Il est nécessaire de l'installer sur une carte Arduino, ou compatible.

« Qu'est-ce que tu vas en faire ? », question d'une Geek

La question revient souvent. Bon OK, c'est petit, c'est joli, mais qu'est-ce qu'on en fait de ces cartes quand on est développeur ou bidouilleur ? Mieux vaut trouver des réponses sinon, on aura droit à « tu vois bien, ton truc prend la poussière et c'est tout ». J'ai connu ça avec mon premier Raspberry Pi, hormis le monter en serveur media. Seule, la Galileo ne sert pas à grand-chose et son usage sera très vite limité. Comme nous le verrons, cette board prend son sens quand on la couple avec des shields et des capteurs. Edison prend son sens dans un contexte embarqué. Sa taille permet des scénarios plus indépendants et plus variés qu'avec Galileo. Galileo pourra par exemple servir dans un contexte domotique : chauffage, alerte de température, lumières, arrosage, une alarme/surveillance, surveiller l'alimentation de son chien, etc. La connectivité avec les réseaux est très simple. Et vous pouvez mettre en place des notifications et des traitements complexes. Vous pouvez imaginer, avec un peu d'électronique et des capteurs supplémentaires connectés une machine à café, vérifier l'état d'un frigo, le taux d'humidité dans une pièce, etc.

François **Tonic**

Grove Starter Kit Plus – Intel IoT Edition + Galileo gen 2 = x^n possibilités !

Passons maintenant à l'utilisation de nos kits IoT. Nous allons les premiers pas en IoT et les différents outils de développement que l'on peut utiliser. Programmez ! reviendra très régulièrement sur l'IoT et les objets connectés en général. Enjoy !

Un montage très simple à réaliser, un thermomètre. Il suffit de brancher en A0 le capteur de température puis d'utiliser le servomoteur et de créer un cadrant gradué. Le code est très simple : on déclare les variables et les capteurs puis selon la température, le servomoteur change la position de l'aiguille.

Pour bien démarrer

- Installer la dernière version de l'image système (à installer sur une micro-SD),
- Mettre à jour les pilotes et le firmware,
- Utiliser uniquement les SDK, pilotes... pour la board gen 2,
- Les outils de développement sont disponibles sur Linux, OS X, Windows,
- Vous pouvez utiliser une connexion USB pour relier votre PC et la board mais le mieux est de passer par le port Ethernet (attention : éviter de la brancher directement sur votre PC).

Pour démarrer, nous vous conseillons vivement Arduino IDE. Il s'agit d'un IDE basic mais suffisant pour bien démarrer la programmation Arduino + Starter Kit Plus. Le langage par défaut est le C. D'autres langages sont disponibles mais dans ce cas, vous utilisez d'autres outils tels que Intel XDK IoT Edition. Le Starter Kit Plus permet de démarrer rapidement mais vous serez vite limité. Des dizaines de capteurs sont proposés par le constructeur du kit (Seeed). Les prix varient de quelques dollars à plus de 30 $.

Trouver un scénario d'usage

Le plus difficile avec un IoT / objet connecté est de trouver et de définir un usage. Le Starter Kit Plus avec sa douzaine de capteurs offre un terrain de jeux intéressant. Vous pouvez utiliser plusieurs kits si besoin. Vous apprendrez rapidement les différentes zones de branchements du shield. Le shield possède :

- 4 connecteurs pour les entrées analogiques : A0 à A4, par exemple pour connecter le capteur de température
- 8 connecteurs numériques pour y connecter les leds, les moteurs, etc.
- 4 connecteurs I2C : bus basses vitesses pour les transporteurs de données.

Vous pouvez très rapidement créer une mini base météo, afficher la température du frigo ou créer de petits scénarios domotiques. Là encore, votre imagination sera la seule limite, à condition bien entendu de posséder les bons capteurs. Les codes proposés sur le Arduino IDE (les Sketch) permettent très facilement de se faire la main. Les codes sont très compréhensibles et en quelques heures, si vous n'êtes pas trop habitué au C, vous serez capable de combiner plusieurs capteurs et de créer des scénarios simples. Une erreur fréquente, quand on débute, est de se tromper dans les numéros de ports des différents capteurs. Le code se structure ainsi :

```
// les include nécessaires
# include < >
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

On compile puis on transfert à la board... Et le code s'exécute.
La connexion peut se faire en filaire ou en sans-fil. Des modules wifi sont disponibles vous permettant de créer de vrais objets IoT. Ainsi, votre objet deviendra indépendant, mais attention à l'énergie. Vous pouvez opter pour une alimentation secteur ou sur batterie (pile, solaire, batterie).
Vous constaterez vite que parfois la board répond mal ou pas du tout. Vous allez régulièrement jouer du bouton Reboot / Reset pour retrouver une board fonctionnel. Quand vous testez de multiples combinaisons de capteurs, cela deviendra un réflexe.
Pensez à installer les librairies et les Sketch Grove. Certains capteurs nécessitent des librairies non présentes par défaut sur l'outil Arduino.

D'autres IDE

Si Arduino IDE est l'outil le plus rapide pour prendre en main les kits, Intel et d'autres éditeurs proposent plusieurs autres outils :
Intel XDK IoT Edition : environnement complet pour coder en JavaScript. Nous avons eu quelques crashs sous OS X et des problèmes d'exécution. Disponible aussi Wyliodrin : un environnement en ligne. La construction des codes est très visuelle. La reconnaissance de la board est parfois délicate. Payant.
Eclipse édition Intel IoT : l'environnement Eclipse packagé pour Galileo et Edison. Support de C et C++. Gratuit.

Nous reviendrons très prochainement sur la mini-board Edison.

François **Tonic**

Les +	Les –
Le tarif des kits	Les limites du Starter Kit Plus
La qualité des capteurs	Les reboot de la board
Prise en main	Les instabilités constatées sur OS X
Le côté sans limite	L'intégration IDE – matérielle
La bidouille des années 1980 - 1990	

WORDPRESS 4.0 : redécouvrez la joie de WordPress

1ere partie : installation & configuration

Pour certains, ce n'est qu'un simple outil de gestion de blog permettant à son administrateur et aux divers contributeurs une mise à jour rapide et facile de son contenu chaud et froid grâce à son interface intuitive et élégante, pour d'autres il permettra de développer un vrai site vitrine d'entreprise ou d'évènement avec un design facilement manipulable et interchangeable.

Certains l'utiliseront même comme solution (légère) pour développer un site de e-commerce ou encore comme simple Portfolio. Une chose est sûre, depuis sa création en 2003, ce simple gestionnaire de blog qu'est à l'origine WORPRESS a su faire son chemin et développer une communauté très active à travers le monde pour devenir aujourd'hui un des CMS PHP les plus utilisés.
Il vous est proposé, au travers de ce premier dossier WORDPRESS, le survol des grandes lignes de l'installation / la gestion de contenu et l'administration simple de WORDPRESS.

Installation sous Windows

Vous utiliserez ici un environnement Windows avec WAMP (qui intègre une base MySql, Serveur Apache et rapidement paramétrable). Pour une prise en mail rapide, téléchargez « wordpress 4.0 » sur le site officiel : « https://fr.wordpress.org/ », c'est un simple fichier « zip » de presque 7Mo !
Décompressez-le dans le dossier racine de gestion de vos sites web. (Par défaut sur web, c'est « www » dans le dossier d'installation de WAMP). Avant toute chose, il est important dans MYSQL de créer une base de données qui accueillera les tables de WORDPRESS (exemple de nom de base : wordpress), avec WAMP passez directement via l'interface phpmyadmin (accessible directement dans le menu). Connectez-vous via le navigateur à votre site WORDPRESS. (Si par exemple vous avez laissé le dossier WORDPRESS tel quel dans le dossier « www » alors en local avec WAMP l'adresse sera : http://localhost/wordpress)
Avant de démarrer, il est important de vous munir des informations suivantes :

- Le nom de la base de données.
- Nom d'utilisateur MysSql
- Mot de passe de cet utilisateur
- Et l'adresse de base de données (par défaut, c'est localhost).

Une fois ces éléments en votre possession, il ne vous reste plus qu'à cliquer sur « c'est parti ! », Fig.1. Les informations récupérées devront être placées une à une dans les « textbox » adéquates. Préfixe des tables, c'est un texte que WORDPRESS utilisera pour préfixer chaque table lors de leur création dans la base de données. (Vous validez avec « envoyer »). Après validation des données saisies, WORDPRESS vous indique que l'installation peut démarrer. (Vous validez avec « lancer l'installation ») Fig.2. Vous voilà sur le paramétrage de base de notre site, 5 informations vous seront demandées :

Fig.1

- Le titre de votre site (blog)
- Un identifiant de connexion ainsi que son mot de passe, qui correspond ici à l'utilisateur qui aura un rôle d'administrateur, nous y reviendrons.
- L'adresse de messagerie de cet utilisateur.
- Et une checkbox, afin de déterminer si vous voulez que votre site soit référencé.

Une fois terminée, une page de validation vous indique le succès de cette installation. Dès la fin de l'installation, WORDPRESS vous propose de vous authentifier avec un login/mot de passe. Il s'agit de celui renseigné à la création du site. (Notez l'url de votre page de connexion (http://<url du site>/wp-login)).
Sans authentification, votre site est accessible sur l'url de base avec une vue publique sans la barre de menu supérieur WORDRESS adapté à votre rôle dans WORDPRESS.

Prise en main de WORDPRESS

Avant de commencer, retenez que l'accès au tableau de bord (administration) est à l'adresse : <http://<url du site>/wp-admin>
Lors de l'authentification, vous vous retrouvez « propulsé » sur votre « tableau de bord» : Fig.3. Ce « tableau de bord » vous donne accès à des options en fonction de vos droits :

- Les différentes activités du site,
- Editer un brouillon rapidement,
- Créer un widget,
- Les news de la communauté.

Sur la gauche, dans le menu latéral, l'ensemble des actions disponibles en fonction de votre rôle (cet article vous indiquera quels sont les droits des administrateurs). Vous pouvez séparer ce menu en 2 sous catégories :

- **La gestion de contenu** (articles, médias, pages, commentaires) : qui concerne toutes les sections de données que vous allez insérer dans votre site,
- **Paramétrage de WORDPRESS** (Apparence, Extensions, Utilisateurs, outils, réglages) : qui concerne ici tous les paramètres d'apparence, d'administration et de fonctionnement de votre site.

Les publications et la gestion de contenu

Vous avez deux grands types de publication qui caractérisent vos contenus dans WORDPRESS.

- **Les articles** : ce qu'on appelle le contenu chaud d'actualité
- **Les pages** : appelé aussi contenu froid non lié à l'activité qui est plutôt statique.

La gestion de ces deux types de contenu se passe dans le menu latéral à leur nom respectif.

Les pages :

La création.

La création d'une « page » passe par « menu latéral gauche » « pages > ajouter ». L'édition de celle-ci ce fait à l'aide *d'un éditeur de texte.* Grâce à l'éditeur de texte en mode visuel, il est facile de mettre en page son texte sans aucune connaissance en développement. Il est

possible de rajouter des médias via le bouton «ajouter un média » au-dessus de l'éditeur sans se soucier du lieu de stockage et du lien à créer. Vous le verrez un peu plus tard Fig.4.
Sur la partie droite vous avez deux encarts :

- **Publier** : ou l'on paramètre l'état (si c'est un brouillon / publié), la visibilité (par tous / privé, authentification obligatoire / protégé par un mot de passe), et la date de publication.
- **Attributs de la page** : dans cet encart vous pouvez déterminer si la page prend sa place dans une hiérarchie précise (utile pour les menus), s'il utilise un modèle, cette section n'est pas toujours visible en fonction du thème, et son ordre (dans le menu par rapport aux autres...)

La gestion. Fig.5
Dans le menu « Pages > Toutes les pages », vous pouvez gérer simplement les différentes pages de votre site. Au-dessus de la liste de vos pages, se trouve une barre de filtre qui vous permet de choisir les pages que vous voulez afficher dans votre liste, par exemple : les publier, celles à la corbeille ou toutes.
En passant le curseur de la souris dessus, vous avez accès à plusieurs fonctionnalités :

- Modifier : permet de revenir sur la page de modification / création de la page,
- Modification rapide : permet d'effectuer des paramétrages rapides de la page, ex : l'état, autorisation des commentaires, l'ordre, le titre, et si c'est une page protégée avec un mot de passe voire privée,
- Corbeille : déplacer la page dans la corbeille (elle n'est plus accessible sur le site. Si vous souhaitez la supprimer définitivement, il faut l'indiquer explicitement),
- Afficher : permet de voir la page.

Les articles :

Création
La création d'un article se passe dans le « menu latéral gauche » « articles > créer ».
Vous retrouvez quasiment la même interface que pour les pages :

- Le titre,
- L'éditeur de texte,
- L'encart « Publier »,
- L'ajout de médias.

Certains encarts importants font leur apparition comme : **« catégories »** et **« mots clefs »** qui, une fois associés à un article, permettent aux visiteurs de se repérer plus facilement et simplifier grandement sa navigation de celui-ci.
Les catégories peuvent être administrées dans le « menu latéral gauche» « articles > catégories ». Une « catégorie » peut aussi être intégrée dans une hiérarchie de « catégories », il comprend :

- Un identifiant (version normalisée du nom),
- Un parent (s'il est hiérarchisé),
- Une description.

Une page liée à la catégorie est virtuellement créée et permet de lister l'ensemble des articles associés. URL par défaut de la page de la catégorie : http://<url **de votre site>/ ?cat=<id categorie>**
Les « mots clefs » peuvent être administrés dans le « menu latéral » « articles > mots-clés »
Les mots clefs permettent une identification plus précise de l'article. Il est commun aujourd'hui de voir sur un blog ou site de base vitrine, une catégorie pour un article suivi de plusieurs mots-clefs liés à celui-ci.
A l'instar des catégories, une page est virtuellement créée pour répertorier l'ensemble des articles partageant le même mot clé.
http://<url **de votre site>/ ?tag=<id mots clés >**

Gestion des articles
Comme pour des pages, vous retrouverez les mêmes pouvoirs de modifications sur vos articles, avec les différents menus disponibles. Toutefois, lors de la modification rapide d'un article, des options supplémentaires apparaissent :

- Modifications des catégories,
- Mots clés,
- La checkbox « mettre le contenu en avant ».

Gestion des menus :

C'est le cœur même de la navigation sur votre site (ou blog). Par défaut, un menu est généré par WORDPRESS (avec le thème par défaut : il correspond aux liens vers la page d'accueil et d'autres pages que vous avez créées). La création du menu se passe dans « l'administration » dans la « barre latérale gauche » « Apparence > Menus » Fig.6. Pour la création du menu, il vous suffit de lui donner

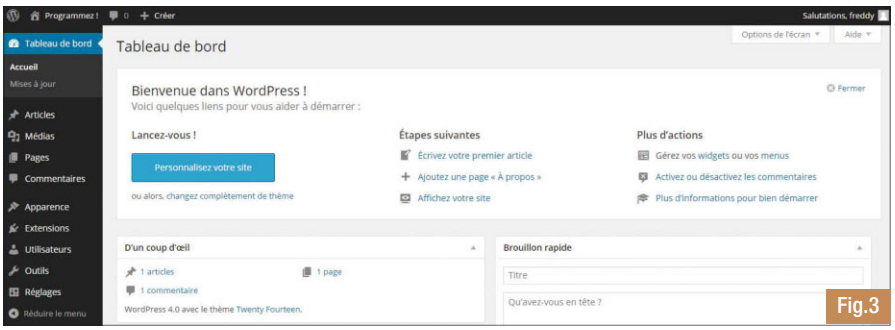


Fig.3

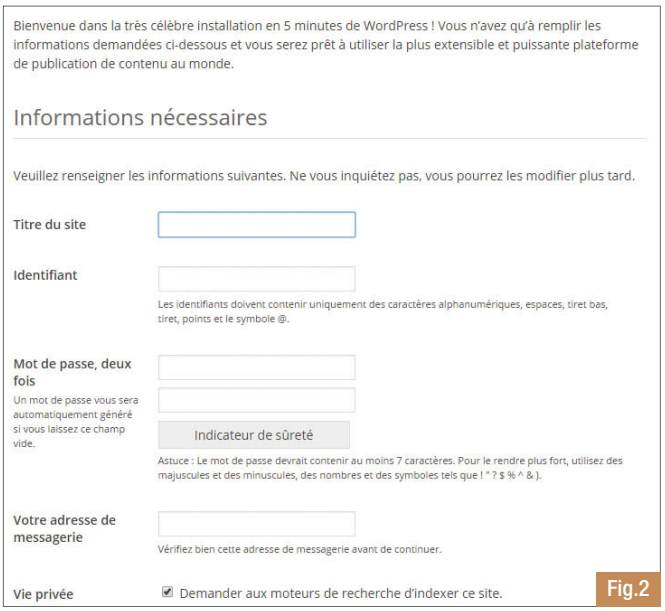
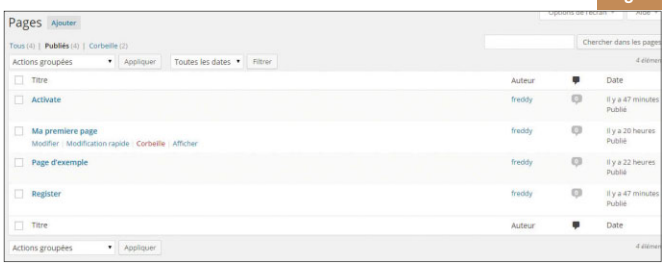


Fig.2

Fig.4 *Une page nécessite l'ajout d'un titre, d'un texte à l'aide de l'éditeur (en mode brut/html ou en mode visuel)*

Fig.5



un « titre » et de cliquer sur « créer le menu », une fois l'action effectuée, votre menu est créé. Vous pouvez, à ce moment-là, ajouter 3 types de contenus à votre menu (se trouvant à gauche dans le cadre principal de la section):

- Des pages,
- Des liens (vers d'autres sites, par exemple),
- Des catégories (sections que nous avons vues avec les articles).

Dans la partie « structure du menu », vous pouvez à tout moment déplacer par « drag and drop » les différents menus que vous avez ajoutés :

- Pour changer l'ordre d'apparition,
- Pour les décaler et faire d'un lien/page/catégorie un sous élément d'une autre entité du menu.

(Exemple sur la figure 7, la page « Page d'exemple » est un sous élément de le page « Ma première page », cela se traduit par une hiérarchisation du menu).

Les médias :

Il existe un 3eme type de contenu (en plus des pages et des articles) que vous allez pouvoir ajouter à votre site (blog) : ce sont les « médias ». Ils sont stockés dans une bibliothèque (accessible via le menu médias). Cela représente tout type de fichier que vous voulez importer dans WORDPRESS (mp3, pdf, images, etc...). Il existe 2 grandes manières de rajouter un média :

- Directement à la création d'un article / page (au-dessus de la page d'édition que nous avons vue précédemment avec le bouton « ajouter un média »),
- Par le menu « médias » se trouvant dans la barre latéral gauche.

L'insertion d'un nouveau « media » dans un article ou dans une page l'intègrera directement dans la bibliothèque des médias de WORDPRESS. Vous pouvez aussi directement faire référence à un media déjà contenu dans la bibliothèque de médias de votre WORDPRESS. L'affichage de votre média est géré automatiquement par le moteur WORPRESS ; par exemple, si vous intégrez un morceau de musique un lecteur apparaîtra sur votre page. Astuce : une chose intéressante à noter est la possibilité de créer une « galerie » lors de l'ajout de media directement dans un article ou une page.

Les commentaires :

Suivant le type de site que vous créez, il est possible de donner la possibilité à vos visiteurs de déposer des commentaires (par défaut, c'est activé). Le paramétrage général des commentaires se fait via le menu réglages > discussion du menu latéral. Différentes options sur cette page vous sont proposées, pour ne citer que les plus pertinentes dans notre cas :

- Autoriser les visiteurs à publier des commentaires sur les derniers articles,
- L'auteur d'un commentaire doit renseigner son nom et son adresse de messagerie,
- Les notifications à la publication de commentaire ou en attente de modération,
- Les mots clefs interdits dans les commentaires que l'on doit modérer,
- L'affichage d'avatar,
- Classement (en fonction de l'âge, un peu comme un PEGI).

Le paramétrage des commentaires peut se faire directement sur un article ou page en particulier dans la section articles ou pages en cliquant sur « modification rapide » en bas d'un élément dans votre liste d'articles/pages, une case à cocher « autoriser les commentaires » vous est accessible.

L'administration

Modération des commentaires

Un commentaire peut avoir 3 types de statut : Approuvé (il est alors sur le site), en Attente (seul l'administrateur du site peut le voir), et indésirable. Lorsqu'un commentaire vient d'être posté sur votre site, sur le menu latéral de l'administration, un chiffre s'incrémente à côté du menu « commentaires » pour indiquer combien de commentaires sont en attente de validation. Pour la gestion des commentaires, il vous suffit de cliquer sur ce menu.

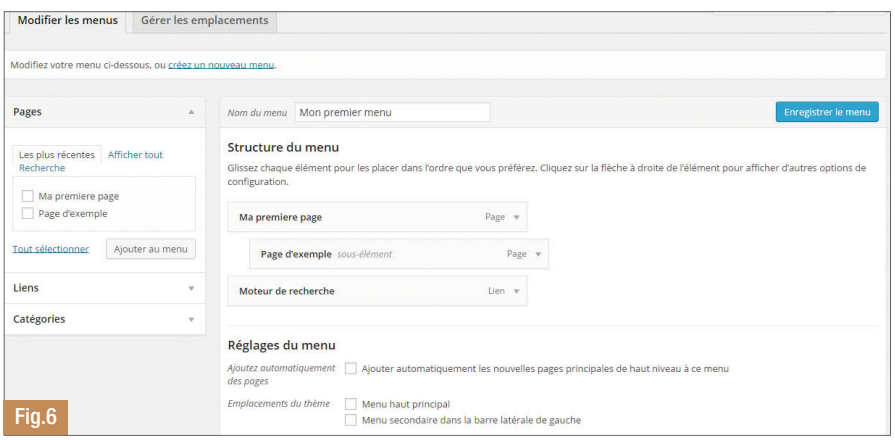


Fig.6

La liste de l'ensemble des commentaires du site est accessible via le menu du haut de la liste ; vous pouvez déterminer le filtre sur les commentaires que vous désirez voir, dans ce cas précis, vous souhaitez voir que ceux qui sont dans un état « En attente ». En passant la souris sur un commentaire, un menu apparait vous proposant : d'approuver, le rendre indésirable, le placer à la corbeille, faire une réponse rapide ou le modifier. Rendre un commentaire indésirable n'est pas une action irréversible et peut être modifiée.

Administration des Utilisateurs et des rôles

Ces paramétrages s'effectuent dans le menu « utilisateurs » du menu latéral. L'accès à cette zone vous permet de voir l'ensemble des utilisateurs de votre site et d'éditer votre profil. Le profil correspond aux informations du compte avec lesquelles vous êtes authentifié : nom, prénom, etc ... (modifiable), mais aussi la préférence d'affichage de l'administration comme les couleurs. A la création d'un « utilisateur », plusieurs champs sont obligatoires :

- Identifiant,
- E-mail,
- Mot de passe,
- Le rôle.

Les rôles

Les rôles sont au nombre de 5, voici la liste par ordre de pouvoir :

- Administrateur : vous avez tous les droits !
- Editeur : il publie et édite l'ensemble des articles contenu dans WORDPRESS (même s'il n'est pas l'auteur),
- Auteur : même qu'Editeur sauf que son pouvoir se limite au contenu dont il est l'auteur.
- Contributeur : il crée et édite des articles et pages dont il est l'auteur mais n'a pas de pouvoir de publication,
- Abonné : il a accès au tableau de bord et à son profil.

Vous pouvez donner l'autorisation aux visiteurs de s'inscrire eux-mêmes sur votre site en cochant la CheckBox : « tout le monde peut s'enregistrer » dans le menu Réglages > Général du menu latéral.

A noter : Vous pouvez aussi indiquer dans le menu « réglages > Général» le rôle par défaut de tout nouvel utilisateur, il vous est conseillé de le laisser à abonner dans le cas où vous permettez l'inscription aux visiteurs sans action de l'administrateur.

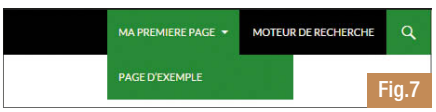


Fig.7

Les thèmes

Grâce aux outils qui vous sont proposés par le site, vous allez ainsi aborder, dans cette section, la gestion de l'apparence de votre site. Ensuite, vous verrez dans un prochain dossier comment en développer un. Par défaut WORDPRESS, vous impose un thème, mais cela peut être modifié. Tout se passe dans le menu « Apparence > Thèmes » dans le menu latéral gauche. Par défaut, 3 thèmes vous sont proposés, mais rien ne vous empêche d'en récupérer d'autres. La communauté WORDPRESS est très active et déborde d'imagination.

Pour ajouter 1 thème à WORDPRESS, il existe 2 méthodes :

- En récupérer un que vous déposez directement dans le répertoire « wp-content/themes » de votre site, à la main. En général, le thème que vous installez est un simple dossier avec un ensemble de fichiers qui le définissent (vous y reviendrez dans un prochain dossier sur le développement d'un thème). Une fois le dossier déposé dans ce répertoire, il est alors accessible via l'interface des thèmes de notre WORDPRESS à côté des 3 autres par défaut.
- Ajoutez directement depuis l'interface en cliquant sur le bouton « ajouter » et, en sélectionnant un thème proposé, cliquez sur « installer », l'ensemble des thèmes présents sont les thèmes officiels de la communauté. L'avantage d'utiliser ce protocole, c'est que lorsqu'il y a une mise à jour sur le thème, l'interface d'administration de WORDPRESS vous l'indique et vous pouvez automatiquement le mettre à jour en 1 clic Fig.8.

L'application d'un thème est simple, une fois installé et accessible dans le menu « apparence > thèmes » il vous suffit simplement de cliquer sur le bouton « activer » sur la vignette de ce thème.

Les Widgets :

Un widget est un composant graphique capable de fournir une fonctionnalité sur votre site : un jeu, un mini moteur de recherche etc... L'avantage d'utiliser un widget à ce moment-là permet d'ajouter des éléments dynamiques sans avoir à développer une seule ligne; ous verrez comment en développer un dans un prochain dossier. La gestion des widget se passe dans le menu « apparence> widgets », à gauche de cette page l'ensemble des widgets disponibles et sur la droite les différentes zones pouvant contenir des widgets.

Ces zones sont dépendantes du thème utilisé ! Des zones présentes sur un thème ne le sont pas forcément sur un autre, attention donc au changement de thème. WORDPRESS propose des widgets par défaut. Pour en rajouter d'autres, il faudra faire l'acquisition de composants type « plugins » proposant un affichage de widgets. L'insertion d'un widget est simple, on « drag and drop » le widget dans la zone voulue au niveau de l'interface de gestion des widgets.

Les plugins

WORDPRESS vous donne la possibilité d'intégrer des nouvelles fonctionnalités à votre site, l'ajout d'une fonctionnalité de gestion d'évènements, des boutons pour les réseaux sociaux, gestion d'un forum. La gestion de plugin se passe dans le menu « extensions » du menu latéral gauche. Il existe là aussi 2 manières d'installer un plugin :

- Installation à la main en récupérant un fichier compressé sur Internet ou que vous avez développé (qui représente le dossier contenant les fichiers constituant le plugin) et vous placez le dossier dans le répertoire « wp-content > plugins » du répertoire d'installation de WORDPRESS.
- Soit en utilisant directement l'interface d'administration en cliquant sur « ajouter » à côté du titre « extensions » ; à l'instar des thèmes, vous pouvez sélectionner et installer le plugin développé par un membre de la communauté sur votre WORDPRESS en 1 clic.

Il existe plus de 25 000 plugins. Une fois installé, il vous suffit de simplement l'activer dans l'interface du gestionnaire des plugins de votre site. Attention : suivant le plugin installé, certains peuvent demander des paramétrages supplémentaires. Certains plugins contiennent des widgets : une fois votre plugin installé et activé, allez faire un tour du côté des widgets pour les disposer comme nous l'avons vu précédemment sur votre page. Quand un plugin de la communauté est mis à jour, l'interface de gestion de votre site vous l'indique simplement par l'incrémentation d'un chiffre à droite d'Extensions, ce chiffre indique le nombre de plugin en attente de mise à jour. Il vous suffira alors de cliquer sur « mettre à jour automatiquement ». Il est possible de mettre un jour un plugin directement à la main en remplaçant les fichiers sources par les nouveaux dans le répertoire que nous avons vu tout à l'heure lors de l'installation d'un plugin.

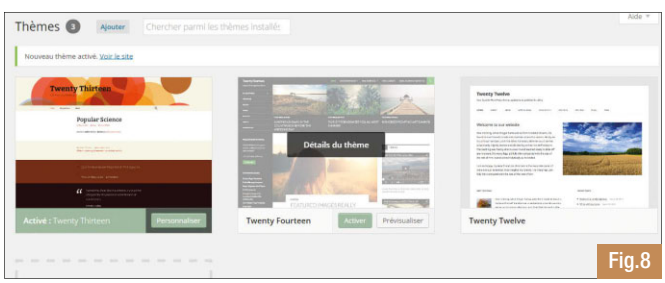


Fig.8

Personnalisation

Cette section va vous permettre de peaufiner les derniers éléments de notre affichage de votre site.... Dans le tableau de bord, dans la section « apparence > personnalisé » du menu latéral gauche, vous avez la possibilité de modifier certains éléments affichés sur votre site comme :

- Titre du site,
- Le slogan (description),
- Les couleurs (font, et background),
- Des éléments du thème, comme par exemple l'image de fond de l'entête du site pour certains thèmes,
- La navigation, ou placer les menus que vous avez créés,
- Contenu de la page d'accueil : page statique que vous avez créée ou liste d'articles dynamiques.

Vous pouvez aussi à ce niveau-là gérer le placement des widgets dans les zones appropriées. L'avantage de la manipulation des widgets à ce niveau vous permet de voir directement le résultat sur votre page.

Autres sections du tableau de bord :

Pour clôturer, vous pouvez retrouver sur votre tableau de bord d'autres options de paramétrage. Dans le menu « Réglages > Général », vous trouverez l'ensemble des paramètres liés aux :

- Titre,
- Slogan,
- Adresse du site,
- L'autorisation d'inscription d'un visiteur,
- Fuseau horaire, Format de date,
- Langue,
- Etc.

Dans le menu « Réglages > Ecriture », vous trouverez :

- Mise en forme automatique des émoticônes,
- Les corrections automatiques des balises XHTML,
- La catégorie par défaut d'un article,
- Paramétrage du serveur de messagerie,
- Paramétrage du serveur de mise à jour.

Dans le menu « réglages > permaliens », vous pourrez paramétrer la forme de vos permaliens de votre site par défaut, par exemple : http://localhost/programmez/2014/10/09/exemple-article/ au lieu de : http://localhost/programmez/?p=123

Freddy Hebrard
Responsable SI Centre de Ressources Régional Lorraine
MCTS/MCPD Microsoft
freddyhebrard@hotmail.com
@freddyhebrard

Cloud & développeur : bienvenue dans la matrice

Le Cloud Computing est partout, ou presque. On en parle partout : de l'utilisateur de smartphone à la direction générale d'une entreprise. Bien entendu, nous ne sommes pas au même niveau de Cloud et sous ce terme générique, se cache une multitude de réalités et d'offres.

Pour simplifier, le Cloud s'articule autour de 3 éléments : SaaS (logiciels / services à la demande, ex. : Office 365, Salesforce...), PaaS (plate-forme, ex. : Cloudfoundry, Heroku, Azure, Google App Engine) et le IaaS (infrastructure, ex. : Google Compute Engine, IBM SmartCloud, HP Cloud, Azure VM). En Cloud, ces services sont tout ou partie managés, c'est à dire que ce sont les fournisseurs qui gèrent les services. Ainsi quand vous utilisez une base de données en mode Cloud, vous n'avez pas à la déployer, à administrer la partie infrastructure. Le but est de simplifier votre travail et de vous concentrer sur celui-ci. Vous ne vous occupez que de votre développement / projet. Vous n'êtes pas administrateur réseau, ni DBA, ni responsable d'exploitation.

Même si vous avez besoin de services, de postes de développeurs pré-configurés avec tels ou tels outils, vous pouvez en quelques minutes provisionner des workloads complets (machines virtuelles + piles logicielles). C'est notamment très utile quand vous exécutez des batteries de tests (= création des bancs de tests). La notion de « dev / test » était un des arguments envers les entreprises et les développeurs.
La question n'est plus de savoir si vous allez utiliser un jour un service Cloud mais plutôt : où, quand, comment ? Et quel fournisseur utiliser...
Plusieurs usages sont possibles et ils sont très réels et concrets :

- Déploiement de sites web directement en Cloud : on ne s'occupe plus de connaître la puissance du serveur et l'administration est la plus simple et automatisée. En moins de 5 minutes vous pouvez activer un serveur et déployer un site web, capable de monter en charge automatiquement. En moins de 30 secondes, vous pouvez éteindre le site web si vous n'en avez plus besoin.
- Utiliser des ressources Cloud pour étendre les capacités de son application : stockage, traitement long de données, backup / recovery...
- Backend applicatif : les services Cloud Cloudcomme backend d'applications, typiquement les applications mobiles.
- Utiliser des services dédiés de développement avec vos outils : Build dans le nuage, environnement de tests, gestion des sources et des projets directement en mode Cloud

Utiliser des services Cloud dans ses développements implique bien entendu d'apprendre de nouveaux SDK, des API et de nouvelles bonnes pratiques. Techniquement, un développement avec des services Cloud ne change pas fondamentalement son travail au quotidien. Si vous créez des applications déployées sur le Cloud, vous aurez à maîtriser les architectures et notamment les notions de multitenants pour mutualiser au maximum les ressources.
Une des problématiques sera de savoir optimiser son code, l'architecture et le fonctionnement général de l'application pour minimiser au maximum l'usage des ressources. Car basiquement, la moindre ressource Cloud se paie à la fin du mois ! Et vous comprendrez très vite qu'il faut traquer le code en trop, les requêtes mal optimisées et les transferts inutiles de données... C'est la véritable « informatique à la demande ». Revers de la médaille, l'informatique devient impalpable. Et plus que jamais, vous aurez à subir « y'a qu'à faire ça » ! Or, ce n'est pas parce que nous sommes en mode Cloud que tout devient plus simple, plus rapide.

Oui, le Cloud est un formidable outil pour le développeur mais non, il ne fait pas de véritables miracles car de nouvelles contraintes apparaissent...

Mais soyons clair. Le 100 % Cloud n'est pas pour tout de suite, du moins, pour beaucoup d'applications. L'usage hybride (déploiement local + services Cloud) va largement se répandre, notamment pour les apps. Tout n'est pas transposable sur le Cloud. Faut-il absolument redéployer une application existante en mode Cloud ? Migrer un existant est toujours périlleux et le coût (budgets, planning) peut rapidement déraper. Le Cloud peut être pertinent sur de nouveaux projets, mais là encore, il faut se demander pourquoi utiliser un service Cloud, lequel choisir, comment l'utiliser dans son développement.

Dans ce dossier, nous allons revenir sur les dernières nouveautés d'Amazon Web Services, de Google et d'Azure. Nous parlerons aussi outils, architectures, usages réels.

François Tonic

LE CLOUD ET LE DÉVELOPPEUR : 10 MOIS PLUS TARD, BILANS ET PERSPECTIVES

Dans le numéro 170 de Programmez !, nous avions publié un long dossier « Le Cloud & le développeur ». Plusieurs thèmes étaient abordés : comment développer sur le cloud, comment et pourquoi le cloud révolutionne le développement, les différentes offres, la sécurité, le cloud et les applications, le cloud et les apps mobiles, l'architecture multitenant.

Dix mois plus tard, un constat : les services managés se sont multipliés, de nouveaux acteurs sont arrivés, d'autres ont arrêté des services ou ont fermé. Par contre, les fondamentaux demeurent : architecture multitenant, optimisation pour payer moins, approche hybride (notamment fusion entre PaaS et IaaS), utilisation des conteneurs de type Docker (voir l'excellent article du n°178 sur le sujet). Le choix est toujours aussi sensible.

Sur les chantiers de l'interopérabilité, des standards, le travail n'a pas forcément beaucoup avancé. Nous voyons que les workloads (environnement complet de production : applications, piles techniques, serveurs) ont encore du mal à aller d'un cloud à un cloud, l'approche Docker fait enfin sauter ce délicat problème même si nous en sommes qu'au début et que son implémentation par les fournisseurs cloud n'est pas homogène, ni toujours présent. Le backend mobile est plus que jamais d'actualité et Amazon et Google courent après Azure qui a su lancé ses SDK avant tout le monde ! Pour le site web, le cloud est un excellent service. Pour du code existant, cela demeure du cas par cas et selon le langage, la technologie. Tout n'est pas transposable ou migrable sur le Cloud.

On peut aussi se demander si le développeur développe pour et sur le cloud. Nous avons mené un petit sondage express sur programmez.com. Le résultat est le suivant :

- 50% utilisent des services cloud
- 50% n'en n'utilisent pas.

© 07-25-13 © Henrik5000

Il faut penser à tout !

Quand on développe en mode Cloud, de nombreux éléments sont à prendre en compte. Ne pas y aller tête baissée, surtout si c'est votre premier projet Cloud. Comme vous le verrez dans ce dossier, nous aborderons plusieurs plateformes et différentes problématiques. Finalement, vous retrouverez grosso modo les mêmes types de services, les mêmes API, les mêmes pratiques, même si chaque fournisseur / plateforme aura ses propres spécificités.

Le SaaS ne reflète en rien la complexité du Cloud au niveau PaaS (plateforme) et IaaS (infrastructure).

Choisir

Choisir, voilà la première chose que vous devrez faire. Vous aurez le choix entre du PaaS et du IaaS. L'avantage du PaaS est de masquer toute l'infrastructure et de vous concentrer sur le code, le projet. La plateforme est managée par le fournisseur. Depuis le PaaS, vous aurez accès à de nombreux services (base de données, annuaire, identité, load balancing, CDN, réplication, services backend mobile, etc.). Selon le PaaS, la richesse fonctionnelle variera.

Si vous travaillez essentiellement avec les technologies et outils Microsoft, Azure sera sans aucun doute le plus indiqué.

Vous pouvez aussi monter des workloads (environnement complet avec les piles techniques nécessaires) sur des IaaS supportant Windows Server. Dans le monde Java, le choix est aujourd'hui relativement large : Oracle, Azure, CloudFoundry, Google AppEngine, Heroku. Le choix pourra se faire sur les services annexes, le support JDK / JEE, les tarifs.

Une question se posera aussi sur la viabilité à long terme de certaines offres Cloud.

Par exemple, Cloudbees a officiellement arrêté son offre PaaS. Avec des fournisseurs importants, les risques sont minimisés, même si une mauvaise surprise n'est jamais impossible. Ensuite, les rachats de petits fournisseurs et éditeurs du Cloud se multiplient, ce qui peut modifier ou compromettre vos applications en production utilisant ces services.

Le fait d'utiliser des technologies standard et ouvertes (notion de Cloud ouvert ou open Cloud) vous évitera de nombreux problèmes et facilitera le rapatriement ainsi que le redéploiement.

Beaucoup de projets, notamment les applications Web et sites Web peuvent fonctionner sur du Cloud fonctionnant sur du Windows ou du Linux. Cela importe peu. L'important est surtout les services et API exposés.

Quelques critères :

- Mes besoins réels : que ce soit sur un nouveau projet ou un projet existant. Faut-il déployer un workload dédié ?
- Le type de Cloud : IaaS, PaaS, les deux,
- Simuler le coût mensuel et annuel du ou des services cloud choisis,
- Quelles sont les API, les SDK disponibles,
- Modèle de données si j'en ai besoin,
- Compatibilité avec les outils de développement et de déploiement.

Notons que les outils open source sont très nombreux sur le Cloud : frameworks, API, outils. N'hésitez jamais à les utiliser quand cela est possible.

« Le Cloud c'est vachement mieux avec du réseau ! » : oups, le mode déconnecté

Aucun Cloud public ne garantit un SLA (niveau de qualité) à 100 % c'est à dire sans aucune panne ou interruption de service. Les taux varient de 99,5 à 99,99999 %. Lisez très attentivement les clauses des contrats SLA pour savoir ce qui est inclus ou exclus, si le SLA est mensuel ou annuel. En cas de panne (et prise en charge), le fournisseur a-t-il prévu une compensation ?

Au-delà, pour le développeur se pose une question : le mode déconnecté. Quand le réseau n'est pas accessible (quelle qu'en soit la raison), cela peut obliger le développeur à implémenter le mode déconnecté. Les principaux Cloud fournissent des SDK ou mécanismes pour assurer un mode déconnecté et resynchroniser après la

FAITES CHAUFFER LA CARTE BLEUE

Soyons clairs : oui le Cloud c'est pratique et très souple, non, le Cloud n'est pas gratuit et peut même vous coûter très cher ! L'avantage du Cloud est de ne pas créer une immobilisation matérielle, ainsi vous n'avez pas à acheter de nouveaux serveurs pour étendre les capacités et à attendre parfois plusieurs semaines entre la commande et la mise en production. Sur un PaaS ou un IaaS, il suffit de quelques secondes pour souscrire un nouveau service, pour basculer d'une instance mutualisée à une instance réservée ou pour provisionner 10 VM supplémentaires. Voilà la souplesse du Cloud : vous ne gérez plus de parcs machines, plus de maintenances, plus de gestion de cycle de vie. Vous provisionnez et "déprovisionnez" les ressources nécessaires.

Toute ressource utilisée / provisionnée / consommée sera facturée soit dans une offre de facturation à la demande (je paie ce que j'utilise), soit la ressource sera décomptée d'un quota de ressources (ex. : dans des forfaits incluant x ressources). La facturation est parfois complexe à maîtriser car selon le service, la tarification change : tarif à la seconde / minute / heure, selon le nombre de transactions, selon les Gigaoctets consommés, selon la zone géographique, etc. Soyez très vigilant.

N'hésitez pas à simuler le budget mensuel et à bien comprendre la tarification et les conditions d'utilisation de celle-ci. En cas de dépassement d'un quota défini dans un forfait, le dépassement peut être douloureux à la fin du mois.

Sur les instances de machines virtuelles, les prix baissent assez fortement depuis 12 mois, même s'il existe toujours des différences entre les fournisseurs. Les différences se font sur les options (ex. : le load balacing inclus ou non) et sur tous les services (données, CDN, backup, mobile, etc.). Vous allez vite apprendre, souvent trop tard, combien le Cloud peut se révéler cher, très cher. Vous devez optimiser, optimiser, optimiser : le code, l'architecture, tous les flux de données, les transactions, les synchronisations et la réplication des données, etc. Plus votre projet Cloud sera optimisé, moins élevée sera la facture. Par exemple, faut-il absolument synchroniser une base entière à chaque connexion d'un utilisateur ou juste en faire un delta. Oui vous allez perdre quelques secondes, mais votre budget sera optimisé...

récupération d'une connexion réseau. Le mode déconnexion concerne essentiellement les données. On fait une distinction entre les applications (sessions applicatives) et les données. Les deux n'ont pas les mêmes besoins de resynchronisation. Sur un mobile, quand l'application Facebook n'a pas de connexion, il l'affiche clairement. Au retour d'une connexion, il lance une nouvelle synchro de données.

Par exemple, Google App Engine propose pour les services mobiles des mécanismes pour synchroniser les données et reprendre l'activité après une déconnexion (offline mode). Dans ce cas, toutes les données sont alors (re)synchronisées en tâche de fond.

Architecture & bonnes pratiques

Reportez-vous à l'article dédié dans ce dossier.

Tester, encore et toujours

Le Cloud évolue chaque mois, pour ne pas dire, chaque semaine. Très régulièrement, les API et les SDK sont mis à jour par les fournisseurs. La mise à jour n'est pas toujours obligatoire, pas immédiatement du moins, pour les SDK. Installer sur une machine (qui ne soit pas votre poste de développement), les nouveaux SDK et API pour les évoluer et voir les comportements avec votre code. Si les tests sont positifs, et éventuellement après des modifications dans le projet, vous pourrez déployer une nouvelle version des applications, si cela se justifie. Là encore, ne vous jetez pas sur les dernières API pour les déployer immédiatement. C'est le meilleur moyen de casser la compatibilité et d'introduire des bugs et dysfonctionnements. Dans la même idée, il peut arriver qu'un nouveau SDK ou une nouvelle version de son IDE ne coopère pas bien. Testez dans une VM ou une partition séparée avant de déployer sur votre environnement de développement. N'hésitez jamais à déployer en pré-production. Le Cloud est aussi un formidable outil pour monter et démonter des environnements de tests en quelques minutes, c'est ce que l'on appelle le Dev / Test. Aujourd'hui, il est très facile d'en créer.

Sécurité : respectez-la !

Re-soyons clairs ! Les fournisseurs de Cloud fournissent plusieurs niveaux de sécurité : tout d'abord sur les datacenters, et dans les services et les connexions vers eux. Sans oublier le cryptage des données par défaut (pas toujours). Que se soit chez Google ou Microsoft, ou Amazon, les mécanismes de sécurité sont longuement décrits dans les documentations techniques.

Un seul mot d'ordre pour le développeur :

- Respecter à la lettre les pratiques de sécurité,
- Implémenter les mécanismes officiels,
- Chiffrer et crypter toutes les connexions et transactions inter-cloud, hybrides, mobiles,
- Utiliser les mécanismes d'authentification et de gestion d'identité.

Aujourd'hui, le Cloud n'est pas moins sécurisé qu'une application Web. Mais le développeur, et l'administration Cloud, ont leur rôle à jouer. A vous d'être vigilant. Les récentes affaires de photos démontrent les conséquences d'un hack ou d'une faiblesse sécuritaire.

François **Tonic**
Adepte du Cloud Computing depuis 2008

Interview d'Alexis Muller, CEO de GenMyModel

En quoi le Cloud a-t-il modifié, changé, le développement et le quotidien des développeurs ?

Nous sommes une petite équipe (huit personnes) dans une start-up technologique et développons un outil de modélisation en mode SaaS (GenMyModel). Ce que le Cloud a changé pour nous se résume en trois idées simples : travail en équipe favorisé, mises en production simplifiées et gestion plus fine des infrastructures. Sur ce dernier point, par exemple, c'est un vrai changement pour une start-up de pouvoir choisir dès le démarrage, des prestations d'hébergement de très bons niveaux et évolutives à un prix parfaitement accessible. Pour nous, en interne, le Cloud a profondément changé les méthodes de travail grâce à des outils nouvellement disponibles comme Cloudbees (intégration continue). Si on compare avec les outils de 2010, les sprints sont plus courts, les mises en production sont plus simples, plus rapides. Les outils Cloud comme GitHub changent également notre façon de travailler avec les tiers. Lorsqu'on collabore avec des développeurs basés sur un autre site, les équipes peuvent disposer sans délais des mêmes outils, sans que d'autres questions parasites viennent perturber l'essentiel : la question des outils et des compatibilités de versions d'un même outil ne se pose plus. Tout est conçu pour collaborer simplement, nativement. De notre point de vue, c'est ça qui change tout par rapport aux outils classiques.

Quels sont les avantages et les inconvénients du Cloud pour le développeur ?

Du côté des points négatifs, on doit évoquer le périmètre fonctionnel qui n'est pas toujours équivalent aux outils classiques. Nous pensons ici aux éditeurs de code Java par exemple, qui ne sont pas encore adoptés en interne parce qu'ils n'offrent pas encore exactement le même niveau de performance que les outils actuellement utilisés.

Dans la colonne "avantages", on peut citer l'absence d'installation / de configuration / de déploiement pour les environnements de test et de production, le prix de départ et les offres évolutives, les fonctionnalités collaboratives souvent avancées et la sécurité des données. De notre point de vue, en effet, les données sont mieux protégées quand elles le sont par des professionnels dont c'est le métier que lorsqu'elles le sont par les entreprises elles-mêmes. En bref, moins de temps passé sur la configuration, plus de temps passé sur les développements.

Comment choisir les services ?

Dans notre cas, de façon naturelle, les services sont d'abord testés par des membres de l'équipe sur des projets personnels ou non critiques. La décision d'adoption de l'outil est ensuite prise selon les réponses aux questions suivantes : ce service permet-il de délivrer plus vite ou d'une meilleure façon ? Quel problème précis ce service permet-il de régler / de contribuer à régler ? L'unité d'œuvre choisie par l'éditeur est-elle cohérente par rapport à notre façon de travailler ? Quels sont les paliers de facturation ? Comment pourra-t-on évaluer l'efficacité du service ?

Architecture(s) Cloud : les bonnes pratiques

Depuis le début du Cloud Computing, l'accent a été mis principalement sur la migration d'applications dans le nuage. Avec l'idée au départ qu'il n'y avait aucune différence fonctionnelle entre le nuage et le datacenter on-premise, juste une différence de coût ! Nous savons à présent, après de nombreux retours d'expériences, que ce n'est pas vrai ! La plus grande source de revenus pour les fournisseurs de plateformes Cloud ce sont les applications Cloud construites spécifiquement pour cette technologie.

Tout d'abord en tant qu'architecte Cloud, il est indispensable de comprendre les bénéfices apportés par le Cloud Computing tant d'un point de vue business que technique par rapport à votre projet. En voici une liste non-exhaustive :

- Scalabilité (montée en charge),
- Haute disponibilité,
- Zéro investissement en infrastructure,
- Automatisation,

Le Cloud, en soi, ne peut pas être considéré comme une révolution de l'IT, mais une simple évolution avec son lot de nouveaux cas d'usage, une certaine maturité et l'industrialisation de l'infrastructure. Le Cloud met en avant certains concepts « anciens » pour construire des architectures Web hautement redimensionnables et introduit de nouveaux concepts qui changent la façon de construire et de déployer des applications. Lorsque l'on passe de la conception à la mise en œuvre d'une application, on a le sentiment que « tout a changé, mais rien n'est différent ». Le Cloud change les processus, les patterns, une certaine philosophie et fait la part belle aux principes des architectures orientées services.

Contraintes du Cloud

Lorsque vous décidez de migrer tout ou partie d'une application dans le Cloud, vous aurez tendance à faire un mapping entre vos spécifications système et celles disponibles dans la plateforme Cloud retenue. Vous allez vous apercevoir rapidement que vous ne trouverez pas exactement les ressources répondant à votre cahier des charges soit en termes de RAM, soit en termes de CPU ou d'IOPS. Il faut bien comprendre que le Cloud fournit un catalogue de modèles de machines virtuelles prêtes à l'emploi, c'est la différence entre le sur-mesure et le prêt à porter dans le domaine de la mode. Vous devez également comprendre que le Cloud fournit des ressources que l'on peut qualifier d'abstraites, et le Cloud prend toute sa puissance en les combinant avec un modèle de provisioning à la demande. Ne soyez donc pas inquiet si vous ne disposez pas exactement de l'équivalent de vos besoins, vous aurez toujours la possibilité de compenser votre

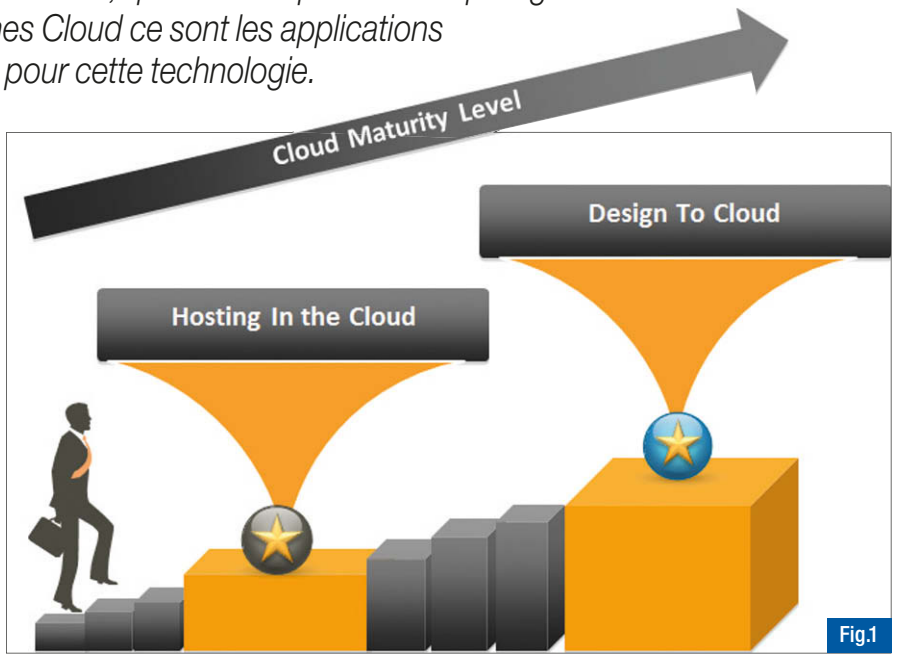


Fig.1

besoin par encore plus de ressources dans le Cloud. D'ailleurs, n'est-il pas plus intéressant d'avoir deux machines virtuelles à 2 cœurs qu'une seule machine à 4 cœurs ? On en rediscutera un peu plus loin dans cet article. En fait, les contraintes du Cloud peuvent devenir un avantage pour votre application en vous apportant plus de scalabilité et de performance.

Migration

Lors d'une migration, il faut bien se rendre à l'évidence: si vous migrez une application de mauvaise qualité dans le Cloud, elle restera de mauvaise qualité ! Rien n'est magique. En d'autres termes, si votre application n'est pas redimensionnable on-premise, elle ne le sera pas non plus dans le Cloud, de même pour la performance ! Pour les architectes logiciels, la migration est un enjeu majeur. On distingue deux types de migrations selon son niveau de maturité et les éventuelles contraintes techniques (figure 1) :

- *Hosting in the Cloud* : il s'agit de migrer tel quel l'application dans le Cloud en utilisant principalement les ressources de type IaaS,
- *Design to Cloud* : on va cette fois-ci modifier ou adapter l'architecture pour tirer profit pleinement des bénéfices du Cloud, on mixera alors des éléments de type IaaS et PaaS.

On constate également parmi les tendances du Cloud que l'IaaS (Infrastructure-as-a-Service) n'est pas suffisant pour profiter pleinement du potentiel du Cloud. Héberger simplement des services dans le Cloud ne crée pas suffisamment de bénéfices pour le client et pas suffisamment de profit pour le fournisseur. Amazon commence déjà à étendre son offre IaaS en offrant de nouveaux services que l'on peut qualifier de services PaaS. Ces fonctionnalités couplées aux services IaaS forment la meilleure combinaison pour apporter de la valeur aux clients.

Les plateformes Cloud de demain seront, selon moi, les plateformes de type PaaS. Microsoft avait d'ailleurs lancé initialement sa plateforme avec uniquement des composants PaaS. Ils avaient peut-être été trop vite sur ce marché. Toujours selon moi, la démarche intellectuelle pour passer d'une application classique au PaaS est beaucoup plus complexe que le passage d'une application classique vers le IaaS. D'ailleurs, les populations concernées ne sont pas les mêmes, la première s'adresse plutôt aux développeurs et la seconde aux équipes d'infrastructure. La réalité n'est pas aussi tranchée. De plus, l'offre initiale de Microsoft était trop contraignante techniquement pour supporter les différentes topologies d'application. Microsoft a su faire « machine arrière » . C'est pourquoi la première étape de migration (ou de développement d'une nouvelle application) consiste à bien étudier l'ensemble des services proposés par le fournisseur afin d'adapter l'architecture. Il est inutile de redévelopper ou de migrer des fonctionnalités offertes par le fournisseur car :

- Un service en mode PaaS est provisionné en quelques clics,
- Ce service est fourni avec un SLA en termes de disponibilité,
- L'infrastructure sous-jacente est gérée par le fournisseur.

Prenons un exemple : vous développez une application Web qui a besoin d'un scheduler qui exécute des batchs journaliers. Avec une démarche « Hosting in the Cloud », vous allez provisionner des machines virtuelles, au minimum deux pour avoir une certaine disponibilité, et développer ce mécanisme de batch avec la mise en place d'un mécanisme de « lock » pour ne pas traiter les données en double. Or, certains fournisseurs Cloud vous proposent un service de type Scheduler en mode PaaS, le développeur se concentre alors uniquement sur le côté business du batch et non plus sur l'architecture technique à mettre en place. On tend donc ainsi vers une application « Design To Cloud ». Les exemples sont nombreux : service de queue, de notification, ... Se pose alors la question de la réversibilité et de la dépendance avec le fournisseur. Si ces deux points sont une contrainte forte, alors il faut prévoir dès le début du projet un plan de réversibilité et mesurer les efforts et impacts pour le mettre en place. Ceci est tout à fait gérable ! Mais lorsque l'on développe une application avec .Net, se pose-t-on la question de la réversibilité vers Java ? Je vous laisse méditer ce point. Il est fort à parier que les principaux fournisseurs apporteront des services assez comparables, on le voit déjà avec des solutions de type cache et gestion d'identité fournis par Amazon et Microsoft à ce jour.

Cloud et développeur

L'impact pour le développeur est donc non négligeable. Nous allons à présent vous présenter quelques patterns pour une architecture Cloud.

Découpler les composants

Le Cloud, comme indiqué un peu plus haut dans l'article, tire profit de concepts issus de la SOA notamment le couplage lâche. La clé consiste à développer des composants qui ne sont pas fortement dépendants entre eux. En effet, si un composant est en erreur, long à répondre ou fortement occupé pour des raisons variées, les autres composants du système doivent continuer à travailler normalement comme si aucun incident ne s'était produit. Bienvenue dans le monde de l'asynchronisme et des mécanismes de Queuing ! Pour ce faire, on va, par exemple, mettre en place des serveurs de queue afin que deux composants puissent dialoguer entre eux comme le montre la figure 2. Le composant A n'est donc plus dépendant de la disponibilité du composant B et chaque composant travaille à son rythme. De plus, si la taille augmente à cause d'une hausse de volumétrie, on va pouvoir « scaler » le composant B uniquement en rajoutant des instances de ce composant et dès qu'il a géré les messages en attente, on effectuera l'opération inverse et on réduira le nombre d'instances. Dans une architecture globale Cloud, on ne va pas gérer le scaling de chaque composant indépendamment mais en fonction de leurs liens logiques. On parle dans ce cas de définir des **unités de scaling** afin de gérer les règles de scaling entre les différents composants. En fonction du nombre de visiteurs d'un site, on va déterminer le nombre de ressources à allouer, le stockage nécessaire, etc... et ainsi le coût d'hébergement dans le Cloud devient déterministe (voir figure 3) et est potentiellement fonction d'un besoin métier et pas uniquement d'un besoin technique.

Elasticité

L'élasticité permet d'optimiser les ressources en collant au plus près du besoin réel évitant ainsi d'avoir une infrastructure sous utilisée ou sur utilisée. Cette élasticité peut être gérée selon 3 modes :

- **Manuel :** le client décide manuellement d'ajouter ou de retirer des ressources. Cela se fait par anticipation à un pic d'activité comme le lancement d'une campagne marketing,
- **Programmé :** pour absorber des pics prévisibles comme des soldes ou autres événements comme une clôture comptable, on peut programmer l'élasticité à l'avance,
- **Automatique :** l'élasticité va se baser sur des métriques de supervision afin de réagir généralement à des pics d'activité non prévus. On se base sur des compteurs comme le CPU, la mémoire ou le nombre de messages dans une queue selon la typologie de votre application.

Mettre en place le mécanisme d'élasticité oblige dans un premier temps à automatiser l'ensemble du processus du déploiement et de définir des workflows pour orchestrer l'ordre de déploiement de l'ensemble des composants. Le système peut alors « scaler » sans intervention humaine, évitant notamment d'éventuelles erreurs de manipulation et d'avoir des dossiers d'installation complexes non mis à jour. Dans un environnement Cloud nous avons à disposition des API qui vont justement permettre l'automatisation du processus de déploiement. Il est recommandé de prendre le temps nécessaire pour mettre en place l'automatisation dès le début du projet de migration et de le mettre à jour au fur et à mesure. Créer un déploiement automatisé et répétable réduira fortement les erreurs et facilitera l'élasticité et l'efficacité de votre solution. Pour automatiser un déploiement, il faut :

- Créer une bibliothèque de scripts pour l'installation et la configuration (certains appellent cela un livre de recettes),
- Gérer les configurations et déploiements via l'utilisation d'un agent,
- « Bootstrapper » les ressources pour les paramétrer.

On parle également « d'Infrastructure as a Code », une approche très séduisante qui permet notamment de faire dialoguer les développeurs et les opérations. Reste à définir qui prend en charge cette écriture de scripts. Parmi les outils du marché, on peut parler de Puppet, Chef, Powershell DSC pour ne citer qu'eux. En fait, au démarrage des machines virtuelles, ces dernières récupèrent les scripts en fonction de leur rôle respectif comme « serveur Web », « serveur applicatif », « serveur base de données » et se l'appliquent. En plus de l'élasticité classique, on peut utiliser cette approche pour créer des environnements projets en quelques clics et un minimum d'effort (Dev, Recette, Production).

Les données

Concernant la partie « Data » d'un projet Cloud, de nouveaux patterns vont permettre d'accroitre la performance et la scabilité de votre couche de données. Lorsque vous utilisez une base de données en mode PaaS, il est judicieux de découper votre modèle de données en plusieurs sous-modèles. Chaque sous-modèle est alors hébergé sur un serveur

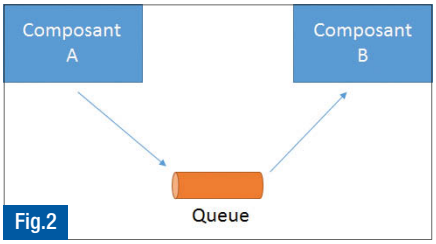


Fig.2

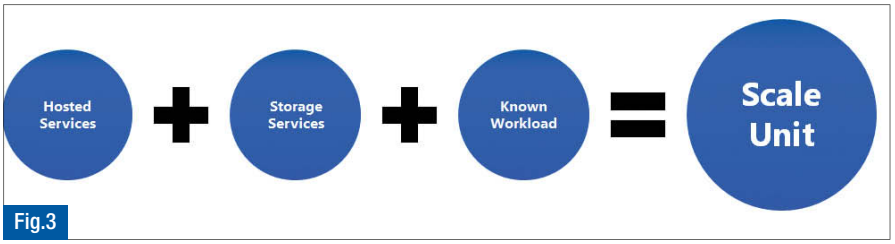


Fig.3

de données différent allouant par la même occasion plus de ressources matérielles pour un coût équivalent. Typiquement pour un site e-commerce, on peut créer un sous-modèle Client / Commande / Produits. Un autre pattern intéressant consiste à séparer les données d'historique des données live. Prenons toujours comme exemple un site e-commerce, on isole sur un serveur l'historique des commandes et sur un autre serveur les commandes du jour. On peut encore aller plus vite en s'autorisant la dénormalisation de l'historique pour avoir des requêtes de lecture plus efficientes. Ainsi les requêtes d'insertion, d'update ne sont pas impactées par les requêtes de lecture plus gourmandes en termes de CPU et de traitement. Pour être complet, il reste à mettre en place un mécanisme pour transférer les commandes du jour vers la base historique à partir de batchs ou d'événements. Ce pattern s'appelle **CQRS – Command and Query Responsability Segregation** ! Le développeur doit également prendre ces évolutions dans son approche du modèle de données pour profiter pleinement du Cloud !

Multi-tenant ou single tenant

Pour illustrer le concept de multi-tenant, comparons deux sociétés bien connues : SAP et Salesforce. Faisons l'hypothèse que SAP a 100 000 clients dans le monde. Chacune de ces entreprises dispose d'une version différente, soit 100 000 instances différentes du même logiciel. Chaque entreprise met à jour sa version de SAP quand elle le souhaite, le plus souvent quand elle le peut ! On estime qu'environ 15 % des clients de SAP utilisent la dernière version. Chaque migration de version est un projet à part entière relativement couteux pour les entreprises. Imaginons que plus de 80 000 entreprises soient clientes de Salesforce. Ces 80 000 entreprises utilisent la même instance du logiciel ; dans le jargon des professionnels, cela se nomme «multi-tenant» ou multi-locataires en français. C'est **LA caractéristique clef des solutions SaaS** : une seule instance du code est partagée par des milliers ou des millions d'utilisateurs du logiciel. Ces 80 000 entreprises utilisent toutes la dernière version du logiciel. Quand Salesforce met en œuvre, tous les trimestres, une nouvelle version, tous les clients, disposent, immédiatement, de cette nouvelle version en toute transparence. On considère que la Révolution Industrielle Informatique (R2I) est déclenchée par l'arrivée des véritables solutions SaaS.

J'ai choisi les leaders de chacun des deux marchés, SAP et Salesforce. J'aurais pu remplacer SAP par Oracle, Sage ou Microsoft Dynamics ; j'aurais pu remplacer Salesforce par TalentSoft. Difficile d'imaginer une *innovation de rupture plus forte* dans le monde de l'informatique professionnelle !

Mais le passage vers le multi-tenant (fig 4) n'est pas sans impact sur votre architecture logicielle et les personnalisations sont limitées car vous partagez un socle commun avec l'ensemble des clients. Le passage vers le multi-tenant aura un coût supplémentaire par fonctionnalités pendant les phases de développement mais le coût d'arrivée d'un nouveau client sur votre plateforme est moindre en mode multi-tenant. Il est donc important de calculer le nombre de clients nécessaires pour atteindre votre point de rentabilité comme l'illustre la fig 5. La migration d'une solution existante peut s'avérer complexe car il faut revoir en profondeur l'architecture logicielle pour bien isoler le client connecté. Bien souvent, le plus simple est de le prévoir dès la conception d'une nouvelle application. Concernant l'architecture des données, on distingue 3 modes : le mode isolé où chaque client a une base séparée, le mode partagé : une base pour tous les clients et un schéma de données partagée, et, entre ces deux modes, le schéma séparé au sein d'une même base. Le choix entre ces 2 modes n'est pas binaire, on parle de continuum avec variations possibles entre les 2 modes extrêmes selon votre contexte et vos contraintes. Le mode multi-tenant a également ses contraintes. Certes cela va réduire les coûts pour le fournisseur de l'application, va permettre une meilleure montée en charge mais Scale Fast, Fail fast : un bug ou un incident va se répercuter immédiatement à l'ensemble des clients.

Mesurer !

Un autre point important pour le développeur est la télémétrie ! Dans la méthodologie SCRUM, on définit généralement le Dod – Definition of Done - pour décrire quand on considère qu'un développement est terminé : cela inclut généralement les tests unitaires, un pourcentage de couverture de code, un build automatisé, un packaging automatisé...

Avec le Cloud, le Dod doit évoluer et inclure qu'un développement est terminé s'il est « mesurable ». Afin d'éviter l'effet boîte noire des ressources Cloud et de donner de la visibilité, le monitoring est un facteur clé de succès pour comprendre ce qui se passe sur la plateforme, estimer pro-activement les besoins. Il n'est pas concevable de mettre en production une application dans le Cloud sans avoir réfléchi au moyen de la superviser efficacement tant au niveau des ressources utilisées, de l'usage de l'application que sur la performance de l'application. Des outils comme des APM – Application Performance Management – sont très pratiques et assez rapides à mettre en œuvre. D'ailleurs de nombreuses solutions en mode SaaS sont disponibles comme NewRelic.

Conclusion

Cet article a pour objectif de vous sensibiliser sur la migration d'une application et la mise en place de patterns d'architecture. La liste des patterns présentée ici n'est pas exhaustive. Je vous recommande la lecture de cette page web : http://msdn.microsoft.com/en-us/library/dn568099.aspx. Un point également important à retenir est que ces patterns peuvent également être mis en place dans une architecture on-premise avec certainement un peu plus de difficultés que dans un environnement Cloud.

Michel **Hubert**
MVP Azure
Manager Cloud & Integration
Cellenza

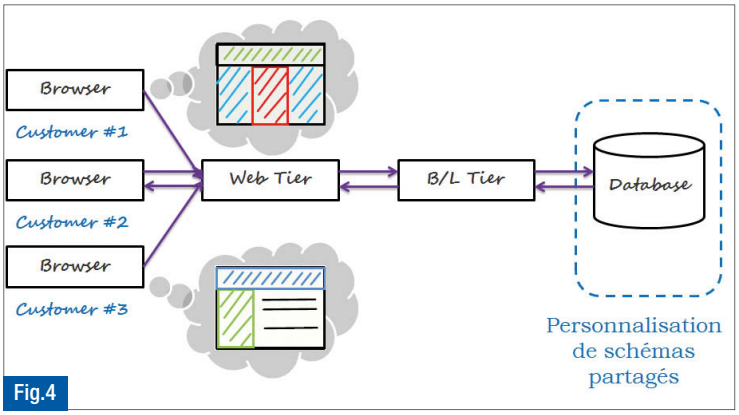


Fig.4

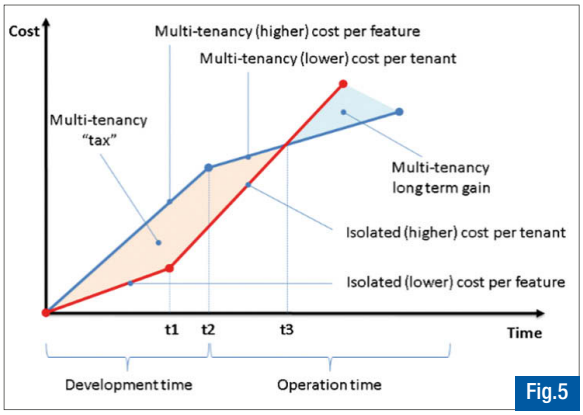


Fig.5

MariaDB dans le Cloud

La base de données MariaDB (fork de MySQL) offre de nouvelles perspectives avec un minimum d'apprentissage. Elle a su s'adapter avec les technologies comme le Cloud. Lorsque l'on parle de base de données sur Internet, MySQL est souvent le premier nom prononcé. Depuis plusieurs années on a vu arriver de nouvelles bases relationnelles et non relationnelles (NoSQL) qui ont pour but de répondre aux attentes du Web actuel et du Cloud. Chaque base a des avantages et des inconvénients. Nous allons aborder trois fonctions de MariaDB.

Installation

Pour installer la base de données MariaDB, vous devez stopper les services MySQL :

```
$ apt-get install mariadb-server mariadb-client
```

Au niveau de la configuration, vous retrouvez le fichier de configuration /etc/mysql/my.cnf qui reconnaît vos différentes bases de données et leurs emplacements Fig.1.

Pour l'utiliser, les outils comme PhpMyAdmin, Heidi... sont compatibles; en ligne de commande vous pouvez utiliser aussi bien mysql -u ou mariadb -u pour vous connecter à la plateforme. MariaDB se comporte comme MySQL, que vous connaissez, c'est à dire :

- Au niveau de l'architecture, la différence sera le prix et les services.
- Au niveau du code il est transparent

MaxScale

MaxScale est un proxy sous la forme d'un plugin. Il a pour but de proposer un équilibrage de charge de connexion appelé 'Connection Load Balancing' (CLB) et une déclaration Load Balancing appelée 'Statement Load Balancing' (SLB).

MaxScale : https://mariadb.com/products/mariadb-maxscale

MariaDB Galera Cluster

MariaDB Galera Cluster est une approche très intéressante car elle assure la réplication multi-maître synchrone. Ainsi, vos données sont plus sûres par rapport à de la réplication classique (Maître / Esclave) car elles sont répliquées immédiatement sans aucun retard. Par ailleurs, vous pouvez lire et écrire n'importe quel nœud ce qui facilite la vie lors de vos développements d'applications.

Severalnines Cluster Control

Parce qu'un cluster de base de données peut être difficile à mettre en place et à maintenir, Severalnines Cluster Control permet de contrôler votre cluster de base de données. En effet, ici, le Cluster Control vous donne le pouvoir de déployer, gérer, surveiller différents clusters. Il prend en charge une variété de SQL et NoSQL pour faciliter l'utilisation du Load balancing via Haproxy. Au niveau des bases de données relationnelles (SQL), vous utilisez les clusters de bases MySQL comme Galera Cluster pour MySQL, MariaDB Galera Cluster, Standalone MariaDB/MySQL, etc. Au niveau des bases de données non relationnelles (NoSQL), vous avez à disposition MongoDB sharded Cluster, TokuMX, etc... Lien : http://www.severalnines.com/clustercontrol

```
mysql -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 181
Server version: 5.2-MariaDB-mariadb102~lucid-log (MariaDB -
http://mariadb.com/)

This software comes with ABSOLUTELY NO WARRANTY. This is free
software,
and you are welcome to modify and redistribute it under the
GPL v2 license

Type 'help;' or '\h' for help. Type '\c' to clear the current
input statement.

MariaDB [(none)]>
```

Fig.1

Cas pratique

L'article va associer les technologies présentées un peu plus haut avec MariaDB dans un environnement Cloud.

Installer sur un Cloud privé

Tout d'abord, vous devez souscrire un service Cloud. Nous utiliserons un service Amazon Web Services (AWS) et le scénario 2 : 'VPC publiques et privées sous-réseaux' pour obtenir un Cloud privé.

Pour effectuer l'opération, vous pouvez vous reporter à la documentation officielle de Amazon VPC (Amazon Virtual Private Cloud) pour créer votre espace :

http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_Scenario2.html

Ce choix est recommandé si vous voulez exécuter une application web destinée au public, tout en maintenant les serveurs back-end qui, eux, ne sont pas accessibles au public, ce qui correspondant au mode Hybride. Maintenant que vos 2 environnements Cloud (public et privé) ont été créés, vous devez installer un nœud de contrôle sur le cluster S9 et aller lui attribuer une IP élastique qui nous servira d'exemple pour notre cas pratique Fig.2.

Dans le réseau privé, vous trouverez les nœuds MariaDB Galera

Mise en place du VPC

Pour installer notre espace virtuel Cloud privé, vous devez suivre un certain nombre d'étapes (inspirées de la documentation officielle) :

- Posséder un compte AWS http://aws.amazon.com/fr/ ,
- Lancer la console Amazon VPC qui se trouve à l'adresse suivante : https://console.aws.amazon.com/vpc/ ,
- Lors du premier lancement, vous créez un VPC à partir de l'assistant,
- Notre choix se portera sur la deuxième option pour obtenir notre espace avec des sous-réseaux publics et privés,
- Enfin pour terminer la procédure et permettre de lancer le NAT dans le sous-réseau public, vous renseignez les différents champs demandés pour permettre la création des tables, sous-réseaux et la passerelle Internet.

Groupe Maxscale-nat

L'étape 2 va créer un autre groupe sécurité SSH dans le but de ne pas stocker le tout au même endroit que vous appellerez MAXSCALE-nat. Pour créer celui-ci, vous repassez par la console Amazon VPC. Ensuite vous choisissez 'Groupe de sécurité' que vous créez. Enfin, vous renseignez les différents champs comme :

- Le nom du groupe de sécurité : Maxscale-nat
- Une description
- Sélectionnez l'ID de votre VPC dans la liste proposée

Après la création de ce groupe, nous ne devez pas négliger les règles de sécurité car elles sont importantes et s'effectuent de la manière suivante : Vous choisissez le groupe qui vient d'être créé ('Maxscale-

nat'). Vous vous retrouvez sur un nouvel écran composé de différents onglets. Tout d'abord, vous devez modifier les règles pour le trafic entrant comme le montre l'image Fig.3. Ensuite, configurer les règles sortantes s'effectue comme le montre l'image Fig.4.

Réglage du routage NAT

Enfin, il est nécessaire de faire quelques réglages au niveau du routage NAT. Pour cela, nous vous conseillons de consulter le lien et d'en suivre la procédure http: //docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_Scenario2.html . La procédure est identique, présentée un peu plus haut (voir l'assistant de démarrage VPC).

Onglet Route

Cet onglet va retrouver vos différents espaces. Ainsi, la première ligne présentée, pilote les différentes instances de la VPC qui par conséquent communiquent entre elles. La deuxième ligne décrit l'entrée qui envoie tout autre trafic de sous-reseau à l'instance NAT (ex : la destination i-e1bdf1a3).

MariaDB Galera cluster

Maintenant que l'environnement est fonctionnel, vous devez préparer les différents nœuds dont vous avez besoin pour utiliser MariaDB Galera Cluster. Notre cas nécessite 3 nœuds que vous créez. Vous créez une instance EC2, de la manière classique en choisissant EC2-VPC au sous réseau 10.0.0.0/24 et le groupe de sécurité MAXSCALE-nat. L'attribution des nœuds va définir :

- Des tranches d'IP (ex : 10.0.0.10, 10.0.0.11, 10.0.0.12),
- Définir une image AMI pour un utilisateur sans clé SSH et de les déployer 2 autres sur la même base.

Accéder aux nœuds VPC

Après avoir créé les routes et pour les utiliser, vous utilisez l'instance NAT à partir d'Elastic IP associée par la boîte de dialogue EC2 Elastic IP. Grâce à cette opération, vous serez en mesure d'accéder à l'instance NAT à partir d'une IP publique via SSH. Cependant, les risques de conflits avec les nœuds sont présents car vous devez configurer l'interface NAT pour accéder à Internet pour accéder aux mises à jour d'AWS. En général, vous avez besoin de changer les itinéraires pour les rendre permanent. Exécutez les lignes de commandes suivantes :

```
cat /etc/rc.local
ip route del default via 10.0.0.1 dev eth0
ip route add default via 10.0.0.83 dev eth0
```

Comme 10.0.0.83 est l'IP du VPC dans l'instance du NAT, maintenant vous vous connectez en mettant à jour l'application

```
$ yum update -y
```

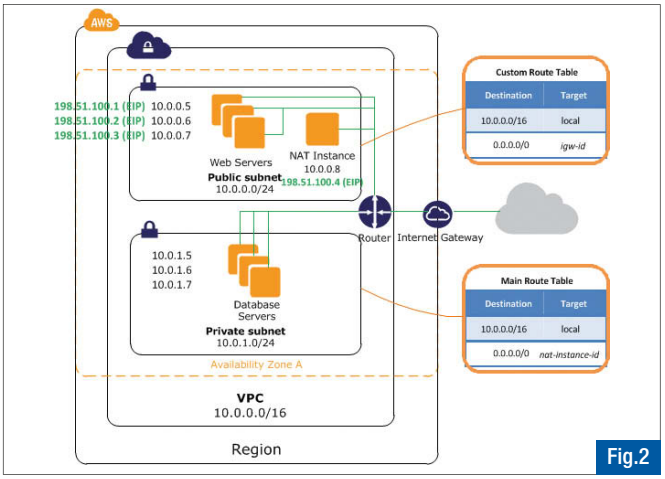


Fig.2

Configuration Cluster Severalnines

Après avoir préparé le cluster, vous configurez MariaDB Galera dans AWS VPS, en utilisant Cluster Severalnines. Pour cela, vous utilisez le script d'installation de contrôle à partir de l'URL suivante : http://www.severalnines.com/New-Galera-Configurator/index.html

La configuration s'effectue à travers une interface comme le montre l'image. A la fin de la configuration, un script de téléchargement (appelé S9) est disponible, avec toutes les instructions.

Installation

L'installation de MariaDB Galera Cluster va utiliser 'Severalnines cluster control' que vous venez de préparer. Pour cela, vous vous connectez en SUDO SSH sur la machine 10.0.0.83 et effectuez ceci :

```
$ cd s9s.installa.script
$ tar xvf s9s-galera-mariadb-2.8.0-rpm.tar.gz
$ cd s9s-galera-mariadb-2.8.0-rpm
$ cd mysql/scripts/install/
$ bash ./deploy.sh 2>&1 |tee cc.log
```

L'installation est assez longue. A la fin, vous obtiendrez le jeton de l'API ClusterControl et l'URL de clustercontrol avec les instructions de connexion.

Installation du proxy MaxScale

La dernière étape concerne l'installation du proxy MaxScale, disponible à l'adresse Internet suivante : https://github.com/skysql/MaxScale

La configuration et l'installation s'effectuent facilement, car toutes les informations sont disponibles dans le fichier readme de la source GIT.

Conclusion

En résumé, MariaDB en mode Cloud ne nécessite pas d'installation particulière car elle est considérée comme un composant à part entière, comme tout autre composant qui vous permet de définir votre configuration dans le Cloud. C'est un avantage de souplesse. Par ailleurs, l'article vous a montré la possibilité de bénéficier facilement de la haute disponibilité du Cloud, et d'un paramétrage configurable au niveau de l'infrastructure.

Plus besoin de vous poser la question de l'architecture : cette souplesse s'appuie sur le principe de nœud et vous pouvez à tout moment ajouter et enlever un nœud en un seul clic. Le seul coût que vous avez à prendre en considération est l'espace Cloud (ici AWS) et la consommation de la bande passante que vous utilisez. De plus, le support de ces modules passe par une prestation payante pour garantir la qualité du service de la base de données.

Christophe **Villeneuve**

Consultant IT pour Neuros, auteur du livre "Drupal avancé" aux éditions Eyrolles et auteur aux Editions ENI, Rédacteur pour WebRIVER, membre des Teams DrupalFR, AFUP, LeMug.fr (MySQL/MariaDB User Group FR), Drupagora, PHPTV..

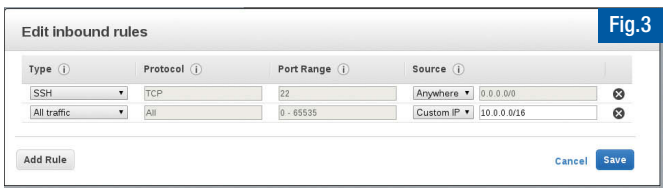


Fig.3

Fig.4

DzMob

www.dzmob.net

Développez vos applications mobiles et tablettes

Nous réalisons vos projets iOS, Android et Windows Phone.

Notre engagement est de réaliser des applications de qualité grâce à nos atouts : Compétences techniques, élégance d'un design bien pensé, rigueur dans la gestion de projets et agilité.

Nous avons réalisé

AudiSpotted
Sur iPhone & Android

TradingSat
Sur iPhone & Android

High Quality Mobile Development

19, rue des Champs, 92600 Asnières sur Seine — contact@dzmob.net — +33 9 83 90 20 74

Azure Websites : vos sites Web sous stéroïdes !

Avec Azure Websites, le développeur dispose d'un service d'hébergement facilitant la mise en ligne rapide d'applications Web en fournissant des options simples et flexibles pour leur déploiement et leur gestion. Dans cet article nous allons rappeler les différents modes d'hébergement d'Azure, présenter l'architecture interne et les spécificités du service et détailler plusieurs fonctionnalités.

HÉBERGEMENT D'UN SITE WEB DANS AZURE

Les différents modes d'hébergement

La plateforme Azure propose plusieurs options pour déployer et assurer l'exécution d'applications web. Il est bien entendu possible d'héberger son site en s'appuyant sur le service IaaS (Infrastructure as a Service) d'Azure : Azure Virtual Machines. Celui-ci permet de créer, déployer, et gérer des machines virtuelles Windows ou Linux (au format .vhd). La virtualisation de l'infrastructure physique, le stockage et le réseau sont directement gérés par Azure, mais la gestion de la machine virtuelle, l'installation et la configuration du serveur Web, et la gestion de l'application doivent être gérées explicitement. Azure est aussi une plateforme Cloud de type PaaS. Le PaaS propose trois approches pour le développement, le déploiement et l'exécution de sites et de services Web :

- Les « Web / Worker Roles », aussi appelés « Cloud Services » permettent au développeur de bâtir son application en s'appuyant sur l'utilisation de modèles d'environnement prédéfinis. Charge au développeur de créer son application de manière compatible à ces modèles, de créer un package à l'aide de Visual Studio ou d'Eclipse puis de le déployer dans Azure,
- Les « Azure Mobile Services » ciblent exclusivement la mise en place de backends pour applications mobiles. Ils permettent en effet d'accélérer la mise en place de l'authentification des utilisateurs, des services de notification, du stockage et de l'exposition des données nécessaires pour une application mobile connectée. Différents SDKs clients existent pour faciliter la connexion de ce service avec les applications des différentes plateformes du marché (Windows 8, Windows Phone 8, HTML5 Client, iPhone/iPad, Android, Xamarin ou Apache Cordova),
- Enfin, Azure Websites que nous allons détailler.

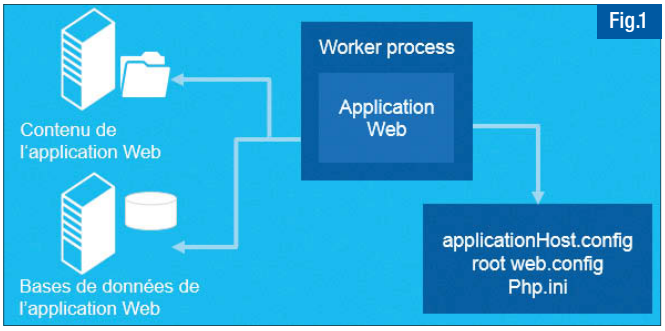
Quel mode d'hébergement choisir ?

Si vous ne ciblez pas une application mobile, trois options sont à considérer pour le développement : Machines Virtuelles, Cloud Service et Websites. Pour de nombreux usages et scénarios, Websites est le service d'hébergement le plus simple à mettre en place et à utiliser, pour les applications existantes et nouvelles. Le diagramme suivant illustre le positionnement relatif de chacune de ces options.

POURQUOI UTILISER WEBSITES POUR UNE APPLICATION WEB ?

Environnement de serveur web traditionnel

En hébergement traditionnel, des contraintes vont apparaître lorsque votre application Web doit être massivement déployée sur une ferme de serveurs (typiquement en mode cluster). Une application Web typique est composée du contenu de l'application stocké dans un répertoire de fichiers, associé à une ou plusieurs bases de données d'application, et complété par les méta-données de configuration requis par le serveur Web. Ce modèle convient parfaitement pour un nombre relativement restreint d'applications Web. Les développeurs et les administrateurs peuvent aisément gérer quelques applications et s'assurer que les diverses ressources externes et les données de configuration restent



synchronisées. Ce modèle devient complexe quand vous multipliez les applications Web sur une même infrastructure ou que l'application nécessite de nouvelles ressources. Plus il y a d'applications et de serveurs, plus l'administration devient complexe. Ces approches traditionnelles de gestion d'applications Web deviennent peu à peu inadaptées et ne permettent pas de garantir la montée en charge et la maintenabilité nécessaire.

Le modèle Azure Websites

Dans un modèle Websites, le niveau de granularité proposé par la plateforme est le processus de travail (« worker process »), dont le but est d'exécuter l'application cible, et non la machine virtuelle. Du point de vue de l'application, rien ne change : les API standards que les développeurs utilisent pour accéder aux ressources externes telles que des fichiers et bases de données continuent à fonctionner comme dans le précédent modèle. Le processus de travail responsable de l'exécution de l'application Web est fourni avec les fichiers de configuration requis par les applications Web. Ni l'application Web, ni le processus de travail ne sont « conscients » du fait qu'ils s'exécutent dans Azure Websites (et non sur un serveur web IIS « traditionnel »), ce qui permet de ne pas avoir à développer ou à packager son application spécifiquement pour ce modèle Fig.1. Azure gère, administre, configure, met à jour, ... Vous vous concentrez sur vos développements et l'application.

Architecture interne

Le service s'appuie sur le serveur Web IIS (Internet Information Server), Windows, et son extension ARR (Application Request Routing) pour gérer la répartition de charge. L'architecture interne de ce service est décrite dans le schéma suivant : Fig.2. Chaque requête HTTP/S passe par une couche de load balancers ARR, puis est redirigée vers l'un des serveurs virtuels où l'application recherchée est instanciée. Lors de la première requête, ou si l'un des serveurs virtuels hébergeant le processus de travail de l'application n'est plus disponible pour une raison ou pour une autre, le système va automatiquement, et à la volée, instancier l'application sur un nouveau serveur virtuel, en récupérant le code source et la configuration de l'application stockés dans un système de stockage persistant (Azure Storage).

QUELQUES FONCTIONNALITÉS

Création et galerie de sites

Websites permet d'héberger une application Web existante mais permet

également de créer un nouveau site Web à partir d'une galerie de modèles de sites. Les plus récentes versions des frameworks open source sont disponibles (WordPress, Drupal, Umbraco, DNN, etc.). Le portail sera votre outil de configuration pour accéder aux paramètres, à la configuration, aux tableaux de bords, aux bases de données (Azure SQL Database, MySQL, SQL Server en mode IaaS, etc...) Fig.3.

WebJobs : profitez des traitements en tâche de fond

Depuis quelques mois, le développeur peut utiliser les WebJobs. Ils permettent de déployer et exécuter du code en tâche de fond sur les machines hébergeant vos sites. Le code s'exécute alors comme un batch planifié (exécution unique ou répétitive), peut être déclenché manuellement (via l'API REST, l'utilisation d'un SDK Azure ou le portail) ou comme un service (exécution en continu surveillée par la plateforme et redémarrée si nécessaire : le service s'exécutera tant que le site est considéré comme actif, état qui peut être forcé via l'option « AlwaysOn »). Les WebJobs héritent de l'ensemble des caractéristiques de la plateforme Websites (montée en charge, remote debug, load balancing, autoscalaing, supervision, etc.). Les WebJobs sont très utiles pour mettre en place des mécanismes asynchrones (l'application Web envoie un message dans une file d'attente, et ce message est dépilé et traité par un Azure WebJob s'exécutant régulièrement de manière asynchrone). Un SDK dédié peut être utilisé pour créer ces WebJobs, par exemple pour créer des tâches qui s'interfacent naturellement avec les files d'attente ou le stockage Azure et bénéficient ainsi d'un outillage complémentaire pour la supervision et le debugging. Exemple d'un message lu dans une file avec déclenchement de l'écriture d'un fichier binaire dans Azure Storage :

```
public static async Task BlobNameFromQueueMessage(
    [QueueTrigger("persons")] Person persons,
    string Name,
    [Blob("persons/{Name}BlobNameFromQueueMessage", FileAccess.Write)]
Stream output)
{
  byte[] messageBytes = Encoding.UTF8.GetBytes("Hello " + Name);

  await output.WriteAsync(messageBytes, 0, messageBytes.Length);
}
```

Autoscaling : une fonction tellement pratique

Une des fonctions phares de Websites est l'autoscaling. Azure s'occupe pour vous de gérer la montée en charge d'une application. Directement dans le portail Azure, vous accédez à l'autoscaling et vous l'activez et le désactivez. Dans cet exemple, le développeur configure l'autoscale selon les critères suivants : déployer votre application automatiquement sur de 3 à 5 instances, en fonction de la charge CPU moyenne sur les dernières minutes. Ainsi, vous payez 5 instances seulement quand vous en avez besoin, mais vous gardez tout de même au minimum 3 instances tout le temps afin d'assurer le meilleur temps de réponse possible pour vos visiteurs.

AZURE WEBSITES POUR LES DEVELOPPEURS

Langages et Framework

Websites est un service très ouvert et vous pouvez ainsi héberger une très grande variété de langages et de frameworks. Cette ouverture offre une grande souplesse en termes de reprise d'un existant. Même ASP, qui n'apparaît pas dans l'interface de configuration, peut être utilisé : comme il s'agit d'un langage de script interprété inclus dans Windows, il peut s'exécuter sans problème.

	Websites	Cloud Services	Virtual Machines
ASP.NET	■	■	■
Classic ASP	■	■	■
Java / Tomcat	■	■	■
Java / Jetty	■	■	■
PHP	■	■	■
Node.js	■	■	■
Python	■	■	■
Ruby		■	■

■ Installé par défaut ■ Installable

Chaque migration vers Azure Websites est différente. Cela peut être un simple redéploiement du code, comme cela peut nécessiter de passer sur une version de Framework supporté ou de réécrire telle ou telle partie de l'application pour être conforme aux prérequis d'Azure Web Sites ou aux bonnes pratiques d'une architecture Cloud (externaliser les logs, rendre le site Stateless, ne plus utiliser de composants systèmes, utilisation d'un autre système de cache, ...).

Outils de développement

Autre avantage, il n'impose aucun outil particulier aux développeurs. N'importe quel outil de développement Web, sur l'ensemble des plateformes Windows, OSX, ou Linux peut être utilisé pour développer un site ayant pour vocation d'être déployé via l'un des multiples protocoles de déploiement supportés par la plateforme : Git, FTP, WebDeploy, TFS,... Microsoft propose une intégration d'Azure Websites dans ses outils Visual Studio, Web Matrix et dans l'éditeur de code de Visual Studio Online (nom de code « Monaco »), accessible directement depuis son navigateur et dédié au développement d'applications Web. Ce dernier peut être utilisé pour éditer un site web déjà

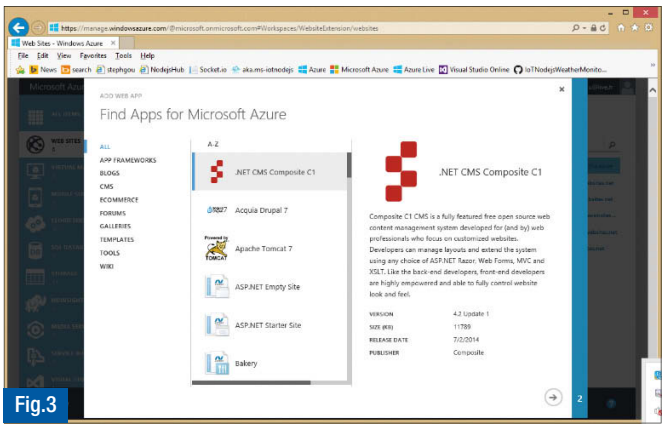


Fig.3

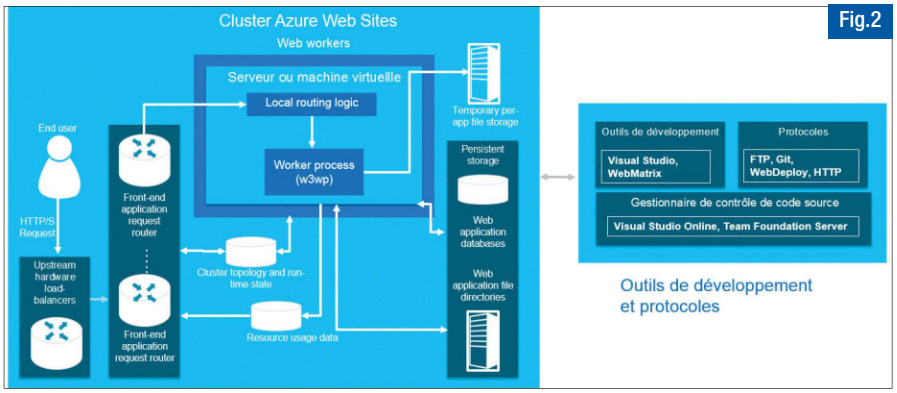


Fig.2

déployé et permet d'établir des connexions aux gestionnaires de source comme Visual Studio Online ou un référentiel Git accessible en ligne.

Contrôleur de code source

Le service propose une intégration native avec différents contrôleurs de code source ou outils de déploiement : Visual Studio Online, GitHub, Git Local, CodePlex, Bitbucket ou encore Dropbox, ce qui permet d'implémenter très simplement un processus d'intégration continue. Par exemple, il est très rapide de configurer un déploiement automatique de la dernière version de l'application Web dans un environnement Azure Websites de test/pré-production à chaque fois qu'un build est réussi.

Architecture Cloud

Un autre impact d'Azure Websites pour les développeurs est la nécessité de mettre en place des architectures pensées pour le Cloud. Contrairement à l'utilisation de machines physiques ou de machines virtuelles dédiées où le degré de liberté est très fort, l'utilisation de services de type PaaS impose une séparation claire entre les différents rôles d'une application. Il n'est ainsi pas envisageable de stocker des données sur les frontaux Web par exemple. Par contre, des services comme Azure SQL Database ou ClearDB MySQL peuvent être utilisés pour vos bases de données transactionnelles, Azure Storage peut être utilisé pour le stockage de fichiers, Azure Search peut être utilisé pour mettre en place un mécanisme de recherche, Azure Cache peut vous aider à mettre en place un cache basé sur Redis, ... Il est nécessaire de considérer Azure comme une boîte comportant différents services adaptés à des besoins spécifiques, dans laquelle on peut piocher pour mettre en place les différentes briques de son infrastructure.

AZURE WEBSITES POUR LES OPÉRATIONS

Tests de charge

Les tests de charge dans Azure peuvent être implémentés grâce à Visual Studio et Visual Studio Online en s'appuyant sur des injecteurs directement montés dans Azure. Grâce à ce service, vous pouvez vous concentrer sur la définition de vos tests, Azure se charge de toute l'infrastructure nécessaire à leur exécution. L'application déployée aura donc fait l'objet de mesures permettant de garantir son bon comportement en charge avant même d'être mise en production.

Supervision

Le portail permet de superviser les sites hébergés en proposant des métriques sur les performances et l'utilisation de l'application, et en offrant un accès rapide aux fichiers de logs des requêtes et aux diagnostics de panne.... Avec Visual Studio Online Application Insights (ou l'un des autres systèmes de monitoring applicatif comme AppDynamics ou New Relic), il est également possible d'observer du système d'un point de vue applicatif, afin de constater un éventuel dysfonctionnement ou une perte de disponibilité et même de faire le lien avec la version du Build correspondante ou la ligne de code concernée grâce à Visual Studio Online.

Provisioning et Configuration

Il est possible de provisionner son application par la mise en œuvre des différents SDK Azure, des Cmdlets Azure PowerShell, de l'API REST de management, ou du langage de scripting Azure Cross-Platform Command-Line Interface disponible sur des environnements Mac ou Linux. Il est également possible d'utiliser la nouvelle notion « Azure Resource Manager » qui permet de déclarer une entité logique dans laquelle sont regroupés différents types de ressources, parmi lesquelles Azure Websites, Azure Storage, Azure SQL Database... Le principe est de pouvoir considérer l'ensemble de ces services comme un tout (son application), et de les gérer ensemble plutôt qu'unitairement. La solution proposée pour le déploiement et la configuration d'un groupe de ressources est déclarative afin de faciliter la configuration des ressources, de leurs dépendances, de leurs interconnexions, de leurs droits d'accès et même de leurs éléments de facturation. Cette notion se fonde sur l'utilisation d'un modèle baptisé « Azure Template » qui va pouvoir garantir l'idempotence, simplifier l'orchestration, la gestion du cycle de déploiement ou encore le retour sur une version antérieure en cas de problème. Ces templates sont implémentés en .json et peuvent donc être gérés dans un contrôleur de code source et ainsi évoluer en parallèle du code de l'application.

MODÈLES TARIFAIRES ET DE MONTÉE EN CHARGE

Azure Websites propose plusieurs options de montée en charge, en « scale out » : Gratuit, Partagé, Basic et Standard. Pour les tiers Basic et Standard, il est même possible de choisir entre 3 tailles de machines, pour une montée en charge en « scale up ». A noter que vous pouvez regrouper vos sites dans un conteneur logique appelé « Web hosting plan », afin que ces sites partagent les mêmes ressources physiques. Il est ainsi possible d'avoir des instances dédiées à votre usage, tout en mutualisant ces instances pour plusieurs sites, qui restent cependant gérés indépendamment. C'est donc Azure qui se charge pour vous de la complexité liée à cette mutualisation.

	Free	Partagé (Preview)	Basic	Standard
Mutualisation	Partagé	Partagé	Dédié	Dédié
Nb. max. de sites	10	100	Illimité	Illimité
Montée en charge (nb. max. d'instances)		6 instances partagées par site	3 instances dédiées	10 instances dédiées voir plus via le support
Paiement	-	Par site et par instance	Par instance	Par instance
Stockage	1 Go	1 Go	10 Go	50 Go
Transfert de données sortantes	Jusqu'à 165 Mo par jour	Illimité	Illimité	Illimité
Sous-domaine azurewebsites.net avec FTP/S and SSL	X	X	X	X
Support de noms de domaine personnalisés		X	X	X
Support de SSL sur les noms de domaine personnalisés			En supplément	5 connexions SNI SSL and 1 connexion IP SSL incluses
Load balancer intégré		X	X	X
Toujours actif			X	X
Web Sockets			350 par site	Illimités
Sauvegarde				X
Mise à l'échelle automatique				X
WebJobs	X	X	X	X
Azure Scheduler				X
Différents environnements de publication				X
SLA de disponibilité	-	-	99.9%	99.9%

Testez dès aujourd'hui

Pour vous lancer, rendez-vous sur www.azure.com, créez un compte gratuitement via la version d'évaluation, testez par vous-même et n'hésitez pas à nous dire sur le Twitter @Azure_France ce que vous en pensez !

Stéphane **Goudeau** et Benjamin **Talmard**
Microsoft Azure France

Google Cloud Platform : Tour d'horizon

Ces 2 dernières années, Google a multiplié les annonces autour de sa plateforme Cloud. Toutefois, celle-ci existe depuis bien plus longtemps puisque la plateforme App Engine est proposée depuis mi 2008. Quels ont été les changements entrepris en 2 ans pour que la plateforme Cloud de Google devienne incontournable? Qu'apporte-t-elle aux développeurs? C'est ce que nous allons détailler dans cet article.

L'offre de Google Cloud s'articule en 3 axes:

- Le compute. Il regroupe App Engine et Compute Engine. Il permet de faire tourner votre code,
- Le storage. Il regroupe des bases de données comme DataStore, Cloud SQL et Cloud Storage pour stocker les données et les fichiers,
- Les services. Développés par Google, ils apportent de nouvelles fonctionnalités sans ajouter de développement : ce sont des services "clefs en main". Cela va de l'exposition d'une API REST facilitée via Cloud Endpoint à la gestion de DNS avec Cloud DNS.

Compute

Historiquement, l'offre compute de Google a commencé en 2008 avec **App Engine**, représentant son offre PAAS (Platform As A Service). App Engine vous permet d'écrire votre code en Java/PHP/Go/Python et en une commande de le déployer en production. Votre application est alors totalement managée par les ingénieurs Google pour garantir sa disponibilité et sa stabilité. Cependant, App Engine renvoie une mauvaise image avec son Lockin qui impose l'utilisation de certaines classes uniquement: il faut adapter votre code à la plateforme. Cette limite tend à disparaître avec les Managed VMs. Nous reviendrons sur cette évolution.

L'autre pan du compute est **Compute Engine** (GCE), la plateforme IAAS (Infrastructure As A Service) de Google, annoncée il y a 2 ans. Compute Engine permet de créer une infrastructure complète pour vos traitements avec gestion du réseau, firewalls, load balancing et storage. On peut aussi instancier des VMs pour y mettre ce que l'on veut. Compute Engine est encore jeune mais au fil des semaines propose de plus en plus de fonctionnalités aussi bien pour les opérationnels que pour les développeurs.

Storage

La plateforme Google propose 3 solutions de stockages :

DataStore : une base de données NoSQL. Egalement disponible sur Compute Engine, c'est un service indépendant permettant de stocker vos données de manière transactionnelle, sans schéma. Google se charge de la réplication et de la scalabilité pour vous.

Cloud SQL : un MySQL dans le Cloud ! Vous pouvez vous servir de cette base comme de n'importe quelle base MySQL. Les drivers, scripts et autres outils que vous avez déjà mis en place avec une base de données MySQL fonctionnent également. Comme pour le DataStore, Google se charge de votre réplication et de la scalabilité.

Cloud Storage : un système de fichiers dans le Cloud. Vous pouvez stocker vos fichiers directement dans le Cloud. Le système gère le versionnage des fichiers. Comme pour les autres services, Google assure la réplication et la disponibilité des données.

Services

La liste des services proposés par la plateforme s'élargit au jour le jour.
Les principaux sont :

BigQuery : service de stockage et de requêtage interactif orienté Big Data. Impressionnant par ses performances, le service vous permet de requêter des tera octets de données en quelques secondes grâce à un langage de requêtage ressemblant fortement à SQL.

Cloud Endpoint : service disponible dans AppEngine qui permet d'exposer une API REST de manière simplifiée. Il permet de générer des snippets de code aussi bien web, que Java ou mobile.

Cloud DNS : service de DNS configurable via une API hébergée directement dans le réseau Google.
Vous profiterez alors des performances du réseau Google.

Le Google Cloud, quel intérêt pour les développements?

LES DÉVELOPPEMENTS ? MAIS DANS QUEL LANGAGE ?

Les intéractions avec le Google Cloud peuvent se faire de 2 manières: via la console web ou l'API REST. Pour vous simplifier la tâche, Google a créé une multitude de clients de l'API REST en Java, Python, Go, PHP, JavaScript, etc. Nous nous focaliserons sur l'approche Java pour App Engine et ligne de commande via l'outil Google Cloud SDK (écrit en Python) pour Compute Engine.

App Engine

Comme évoqué ci-dessus, App Engine représente l'offre PAAS de Google Cloud. Pour déployer une application sur cette plateforme, il suffit d'envoyer l'application packagée (.war pour une application Java). La suite est automatisée: déploiement, scaling, monitoring, etc. Il existe malgré tout quelques règles à respecter.

Les runtimes

Les runtimes de App Engine sont l'équivalent d'un serveur d'application. Ils viennent avec leur lot de services disponibles et il suffit de coder les

intéractions avec ceux-ci. Il existe 4 runtimes : Java, Python, PHP (preview) et Go (experimental). NodeJS serait le 5eme runtime disponible prochainement.
En Java, vous pouvez créer puis déployer une application très rapidement. Par exemple, avec Maven (3.1 minimum) vous créez une nouvelle application avec la commande suivante:

```
mvn archetype:generate -DarchetypeGroupId=com.google.appengine.
archetypes -DarchetypeArtifactId=skeleton-archetype -DarchetypeVersion=1.7.5
```

Une fois l'application créée, actualisez-la avec la dernière version du runtime Java de Google App Engine (GAE) dans le pom.xml à l'aide de la propriété appengine.target.version (par exemple 1.9.7).
Ensuite, créez une servlet :

```
public class HelloWorldServlet extends HttpServlet {
    @Override
    protected void doGet(HttpServletRequest req, HttpServlet
Response resp) throws ServletException, IOException {
        resp.getWriter().print("HelloWorld");
    }
}
```

Mettez à jour le fichier web.xml :

```
    <servlet>
        <servlet-name>HelloWorld</servlet-name>
        <servlet-class>fr.xebia.programmez.HelloWorldServlet
</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>HelloWorld</servlet-name>
        <url-pattern>/helloworld</url-pattern>
    </servlet-mapping>
```

Enfin, pour tester en local, il suffit de lancer la commande mvn appengine:devserver, puis d'aller sur http://localhost:8080/helloworld pour voir le helloworld s'afficher.
Pour le déploiement de votre helloworld sur les serveurs de Google, il faut tout d'abord créer un projet dans la console https://console.developers.google.com/project.
Une fois créé, notez le "Project ID" que Google vous a attribué et copiez-le dans le fichier appengine-web.xml dans le tag <application>. Il ne reste plus qu'à lancer la commande mvn appengine:update pour déployer l'application. Une fois terminé, allez sur http://<Project ID>.appspot.com/helloworld pour afficher votre beau helloworld.

Les services et les outils

Pour le stockage des données de votre application, GAE propose d'utiliser sa solution nommée "DataStore", une base de données NoSQL utilisant Bigtable. Vous pourrez également utiliser une base de données relationnelle (MySQL) si nécessaire. Toutefois, l'utilisation de cette dernière ne fait pas partie du quota gratuit quotidien.
Dans le framework, GAE fournit un ensemble de services :

- Un système de *queue* permettant de déporter les tâches longues dans des processus séparés,
- Un service de cache (memcache),
- Un service *capabilities* vous permettant d'interroger l'état en temps réel des services GAE afin de prévoir une alternative si un problème ou une maintenance sont identifiés,
- Un service *channel* pour faire du push vers des clients JavaScript,
- Un service d'indexation/recherche de documents avec la possiblité d'indexer des points géolocalisés, en plus des classiques date, texte, etc.
- Le service "Endpoint" qui vous permet de développer rapidement une API accessible grâce à des clients générés pour iOS, Android ou JavaScript,
- La gestion OAuth des utilisateurs Google.

Concernant les outils de développement, Google met à disposition un plugin Eclipse pour le runtime Java. Pour le développement d'une API avec le service Endpoint, vous pourrez utiliser l'IDE Android Studio s'appuyant sur IntelliJ. De manière plus générale, vous pourrez utiliser un plugin Maven permettant d'être agnostique de l'IDE utilisé.
Dans les fonctionnalités à venir, Cloud Debugger est un outil pouvant se révéler puissant pour un PAAS comme GAE. Il permettra de poser des break points dans votre code via la console de GAE, de vérifier les valeurs, de débugger votre application en direct et cela, même si celle-ci est répartie sur plusieurs instances. Cloud Debugger est pour le moment accessible en tests après acceptation de Google .

Push to deploy

La fonctionnalité "push to deploy" ou "release pipeline" initialement accessible pour Python et PHP est depuis peu disponible pour Java. Pour fonctionner, Google va créer une instance Jenkins dans la partie Compute de votre projet. Après chaque push vers votre repository GitHub (ou un repository GIT spécifiquement hébergé par Google), il lancera automatiquement le build, les tests et le déploiement vers App Engine. Vous l'aurez compris, pour les projets Java, il vous faudra autoriser la facturation pour financer l'instance Jenkins. A noter que pour les plus petits projets, il est possible de désactiver ponctuellement la fonctionnalité.
Concernant l'utilisation de cette fonctionnalité, Google doit être autorisé à accéder à votre code en créant soit un repository GIT hébergé par Google, via la console Google Cloud, soit en autorisant l'accès à un repository GitHub. Vous aurez alors accès au code source à partir de la console web Fig.1.
Vous pouvez maintenant éditer directement votre code dans la console et valider les modifications dans le repository GIT.
En activant l'option "Use Maven to build test and deploy", le serveur Jenkins lancera à chaque commit le build, les tests, et finalement le déploiement sur App Engine.

Le Lockin et le sandbox

Un runtime GAE représente un langage et un framework qui permettent de développer des applications scalables. Cependant, avec la dépendance de ce framework, vous ne pourrez plus déployer votre

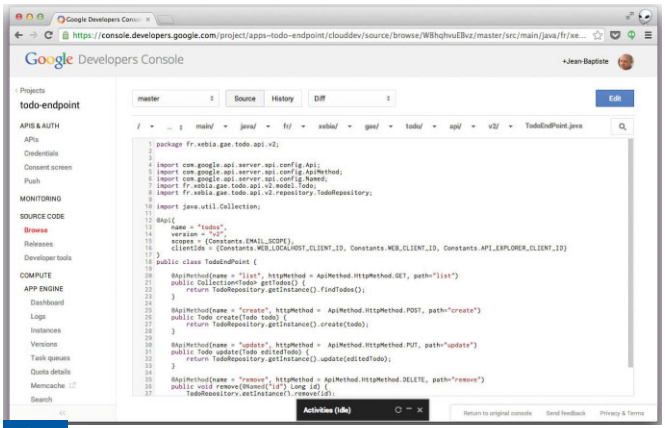


Fig.1 *Edition de code dans la console web de Google Cloud*

application ailleurs que dans les serveurs GAE. Ce problème de Lockin dénoncé par de nombreux développeurs a fait réagir Google qui a répondu avec un TCK (Technology Compatibility Kit) pour le framework GAE. Celui-ci permet alors d'implémenter sa propre version alternative et compatible: déployer ailleurs que chez Google est maintenant possible, le projet CapeDwarf de Red Hat en est un exemple.
En plus du Lockin, les runtimes s'exécutent dans un "sandbox" appliquant des limitations afin d'assurer la sécurité et la scalabilité des applications développées (pas d'accès au systeme de fichier, manipulation des threads limités, etc.).

Les Managed VMs Fig.2.

Pour les raisons citées précédemment dans "Le Lockin et le sandbox", Google propose une alternative qui ambitionne de combiner l'aspect PaaS de Google App Engine (GAE) et la souplesse de Google Compute Engine (GCE) : les Managed VMs. Grâce aux Managed VMs, on peut maintenant avoir accès à un runtime standard tout en profitant des avantages d'App Engine (monitoring, scalabilité, services...).
Reprenez votre helloworld et modifiez-le pour afficher le résultat de la commande "uname -a" qui permet de présenter les informations de l'OS où se trouve la JVM :

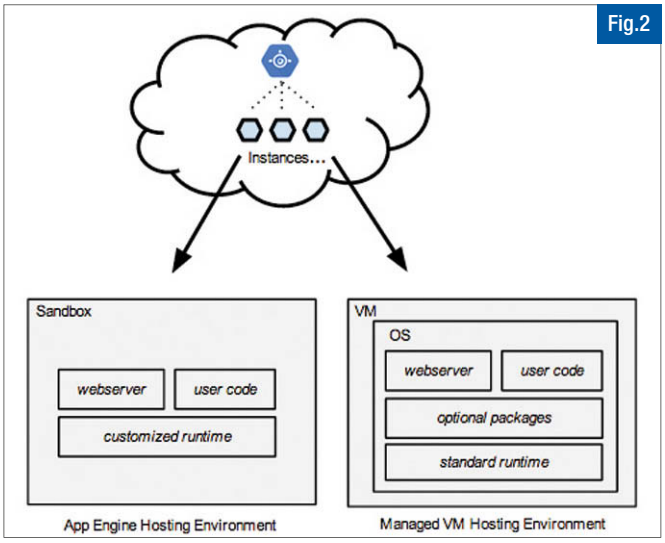
```
protected void doGet(HttpServletRequest req, HttpServletResponse
 resp) throws ServletException, IOException {
       Process p = Runtime.getRuntime().exec("uname -a");
      BufferedReader b = new BufferedReader(new InputStream
Reader(p.getInputStream()));
       resp.getWriter().print(b.readLine());
   }
```

Lorsque que l'on éxécute ce code dans App Engine, on obtient une erreur :

HTTP ERROR 500

```
Problem accessing /helloworld. Reason:
  access denied ("java.io.FilePermission" "<<ALL FILES>>"
"execute")
Caused by:
java.security.AccessControlException: access denied ("java.
io.FilePermission" "<<ALL FILES>>" "execute")
 at java.security.AccessControlContext.checkPermission(Access
ControlContext.java:457)
 at   java.security.AccessController.checkPermission(Access
Controller.java:884)
  at java.lang.SecurityManager.checkPermission(SecurityManager.
java:549)
[...]
```

Fig.2



Le Lockin nous empêche d'accéder à la classe FilePermission. Avec les Managed VMs, cette limite est levée. Pour activer la création d'une instance App Engine en mode Managed VMs, il vous suffit de rajouter dans le fichier appengine-web.xml la ligne :

```
<vm>true</vm>
```

Maintenant, lorsque l'on exécute le code précédent, on a bien le résultat de la commande "uname -a". Ceci n'est qu'un exemple simpliste mais l'accès à un vrai runtime rend App Engine beaucoup plus souple. Il donne la possibilité de rendre des frameworks, tel que Grails, utilisables dans AppEngine et vous donne accès à des machines avec plus de CPU et de mémoire. Enfin dans un avenir proche, App Engine gèrera les runtimes custom donnant une totale liberté tout en conservant les avantages d'App Engine !
Liens utiles :
https://developers.google.com/appengine/ : documentation de App Engine
https://github.com/GoogleCloudPlatform/appengine-java-vm-hello : exemple helloworld

Compute Engine

Compute est la partie IaaS du Google Cloud. Vous allez pouvoir créer votre infrastructure complète avec la création de réseaux, firewalls, load balancers, et machines virtuelles. Pour la réaliser, vous pouvez passer par la console web, l'outil Google Cloud SDK, ou encore, REST.

Google Cloud SDK: l'outil couteau suisse

Pour faciliter la vie du développeur, Google propose l'outil Google Cloud SDK. C'est un utilitaire en ligne de commande permettant d'interagir très facilement avec Compute Engine, App Engine et les autres services Google Cloud. Cet outil se télécharge à l'adresse suivante: https://developers.google.com/cloud/sdk/.

S'authentifier une fois pour toute

Une authentification est nécessaire pour accéder aux APIs Google Cloud. Celle-ci se fait en exécutant la commande :

```
gcloud auth login
```

Cette commande va permettre à l'outil de se brancher à votre compte Google et de vous authentifier de manière transparente pour toutes vos opérations avec le Google Cloud.
Il peut également être pertinent de persister le nom de projet pour éviter de le saisir à nouveau à chaque commande :

```
gcloud config set project MY_PROJECT_NAME
```

Scripter la création de VM Compute Engine

Le Google Cloud SDK vous permet en une simple ligne de commande de créer une instance Compute Engine:

```
gcutil addinstance my-first-instance --machine_type=n1-
standard-2 --image=debian-7 --zone=us-central1-a --wait_
until_running --auto_delete_boot_disk
```

Dans le détail:
- `gcutil`: est l'outil pour interagir avec Compute Engine,
- `addinstance`: indique à Compute Engine que l'on veut créer une instance,

- `--machine_type`: définit le type d'instance souhaitée. Il existe actuellement 4 grandes familles d'instances :
  - Instances standards (préfixées par n1-standard) : machines non spécialisées à utiliser pour des traitements n'ayant pas de besoins particuliers,
  - Instances high memory (préfixées par n1-highmem) : machines spécialisées pour les traitements gourmands en mémoire,
  - Instances high CPU (préfixées par n1-highcpu): machines faites pour les traitements très gourmands en CPU,
  - Instances avec partage de CPU : machines faites pour les petits traitements. Très utiles pour les tests en début de projet car très peu onéreuses.

Pour avoir la liste complète des types d'instances, vous pouvez utiliser la commande :

```
gcutil listmachinetypes
```

- `--image`: affiche le type d'OS utilisé par l'instance. Actuellement 4 OS. sont supportés : Debian 7, CentOS 6, Red Hat 6 et SuSE Linux 11. Pour avoir la liste des différentes images: `gcutil listimages`. Il est possible de créer soit même ses images.
- `--zone`: définit la zone dans laquelle votre instance sera. La position géographique de la zone est importante car plus elle sera proche, moins la latence réseau sera importante. Il existe des régions en Europe, USA, et Asie.

`gcutil listzones`: permet de lister les zones disponibles.

- `--wait_until_running`: précise que l'on souhaite attendre que l'instance soit bien créée avant de rendre la main.
- `--auto_delete_boot_disk`: permet de supprimer le disque dur de l'instance dès que celle-ci est eteinte.

Accéder en SSH à une instance Compute Engine

Pour accéder à une instance Compute Engine, Gcutil nous fournit une commande:

```
gcutil ssh my-first-instance
```

Cette commande n'est que l'alias d'une commande SSH standard mais elle permet d'accéder directement à la machine souhaitée. Mieux encore, si votre environnement n'est pas encore créé, la commande va vous générer une clef SSH et va la répliquer sur le Google Cloud pour accéder à toute nouvelle machine.

Liens utiles :

https://developers.google.com/cloud/sdk/ : Homepage Cloud SDK

http://googlecloudplatform.blogspot.fr/2014/03/tips-and-tricks-command-line-access-to.html : tips & tricks Cloud SDK

Deployment Manager: automatiser l'installation de votre architecture

Lorsqu'on se connecte sur une instance Compute Engine, il n'y a que l'OS. Pas de logiciel tiers installé, encore moins vos applicatifs. Pour pouvoir automatiser l'installation d'une machine, vous pouvez faire un snapshot d'une instance déjà installée et la dupliquer. Cependant, la maintenance est généralement assez compliquée (mise à jour d'un applicatif, changement de paramètres, etc.). D'autres problématiques telles que la création de vos règles de load balancing, firewall et la gestion du scaling peuvent se présenter. Pour y répondre, le Deployment Manager a été créé.

Le Deployment Manager est un service qui permet, à partir d'un fichier de configuration JSON ou YAML, de créer une ressource. Il existe 5 types de ressources :

- LOAD_BALANCING (lbModule): crée des règles de load balancing,
- HEALTH_CHECK (healthCheckModule): crée des règles pour vérifier l'intégrité de vos applicatifs,
- FIREWALL (firewallModule): gère les règles d'accès réseau à vos machines,
- REPLICA_POOL (replicaPoolModule): crée une batterie d'instances de même type à partir d'une seule configuration,
- AUTOSCALING (autoscalingModule): gère des règles pour démarrer ou arrêter des instances sans intervention.

Pour mieux comprendre, prenons un fichier de description YAML mettant en place une architecture comprenant les éléments suivants:

- Des serveurs NodeJS deployés sur des instances Compute Engine,
- Un load balancing entre ces serveurs,
- Un health check vérifiant que les serveurs sont disponibles,
- Une règle firewall pour ouvrir le port 8080,
- Une règle d'auto-scaling créant une nouvelle instance dès que celle-ci arrive à 80% de CPU.

Le fichier de description YAML commence par un header donnant un nom à notre déploiement:

```
name: my-first-deployment
description: Mon premier déploiement
modules:
 #Tous mes modules ici
```

Déclarer un replica pool

Voici la déclaration en elle-même:

```
nodejs:
type: REPLICA_POOL
replicaPoolModule:
 numReplicas: 2
 replicaPoolParams:
 v1beta1:
  machineType: n1-standard-1
  zone: us-central1-a
  baseInstanceName: nodejs
  disksToCreate:
   - boot: true
    initializeParams:
     sourceImage: #url vers image
     diskSizeGb: 100
  initAction: install
  networkInterfaces:
   - network: default
    accessConfigs:
     - name: External NAT
      type: ONE_TO_ONE_NAT
 envVariables:
 PORT:
  value: 8080
```

- `nodejs` : le nom donné au module.
- `type` : le type de module. Ici, nous définissons un REPLICA_POOL. Ce module crée des instances identiques.
- `replicaPoolModule` : début de déclaration du module.
- `numReplicas` : le nombre d'instances à créer.
- `v1beta1` : la version du module utilisée.
- `machineType` : le type d'instances Compute Engine à démarrer.
- `baseInstanceName` : le préfixe des noms d'instance. Le nom final d'une instance sera quelque chose comme nodejs-<chaine de caractères aléatoire>.

- `disksToCreate` : Définition du disque dur à utiliser.
- `boot` : indication sur le disque dur souhaité est bootable.
- `initializeParams` : les paramètres pour initialiser le disque.
- `sourceImage` : l'url vers l'image OS. Si on veut une debian par exemple, l'url est https://www.googleapis.com/compute/v1/projects/debian-cloud/global/images/debian-7-wheezy-v20131120.
- `diskSizeGb` : la taille du disque en Go.
- `initAction` : référence vers une action à lancer une fois l'instance démarrée.
- `networkInterfaces` : définition de la partie réseau.
- `network` : le nom du réseau à utiliser.
- `accessConfigs` : définition d'un accès. Les valeurs ici sont toujours les mêmes dans cette version. S'il n'y a pas d'`accessConfigs` défini, l'instance n'aura pas d'accès Internet.
- `envVariables` : définition des variables d'environnement à la machine. Ici, on définit la valeur du port de NodeJS.

Dans la définition de ce module, on référence l'action install. Une action est un ensemble de commandes linux qui est lancé après le démarrage de votre instance. La définition d'une action se place après les modules dans notre fichier.

```
actions:
 install:
   commands: [
    "wget -O nodejs.tar.gz http://nodejs.org/dist/v0.10.29/
node-v0.10.29-linux-x64.tar.gz",
     "tar -xzf nodejs.tar.gz",
     "cp node-v0.10.29-linux-x64/bin/* /usr/local/bin/",
     "cp -r node-v0.10.29-linux-x64/lib/* /usr/local/lib/",
 "%file:setup-node.sh",
     "/usr/local/bin/node /srv/www/hello.js"
   ]
```

Dans cette définition, il n'y a que des commandes Linux standard à l'exception de "`%file:setup-node.sh`" qui permet de référencer un fichier externe. Ici un script SH va générer un fichier JS pour démarrer votre node:

```
DIR=/srv/www
FILE=hello.js
mkdir -p $DIR
cat <<EOF >$DIR/$FILE
var http = require('http');
var server = http.createServer(function (request, response) {
 response.writeHead(200, {"Content-Type": "text/html"});
 response.end("<html><body>Hello World</body></html>");
});

server.listen($PORT);
EOF

chmod a+r $DIR/$FILE
```

Déclarer un load balancing

```
load_balancer:
 type: LOAD_BALANCING
 lbModule:
   ipAddress: 1.2.3.106
   ipProtocol: TCP
   portRange: 200-65535
   targetModules: [
    nodejs
   ]
```

- `load_balancer` : le nom du module de load balancing.
- `type` : le type de module.
- `lbModule` : debut de déclaration du module.
- `ipAddress` : l'IP du load balancer. Si on ne définit pas ce paramètre, l'IP sera automatiquement assignée.
- `ipProtocol` : on peut faire un load balancing sur différents types de protocole. Par défaut, ce sera TCP.
- `portRange` : la plage de port que l'on veut load balancer. Si on ne définit pas ce paramètre, tout le trafic sera load balancé.
- `targetModules` : où pointe le load balancing. Dans notre cas, ce sera sur vos instances NodeJS.

Déclarer un health check

```
health_check:
 type: HEALTH_CHECK
 healthCheckModule:
host: 1.2.3.106
path: /
port: 8080
checkIntervalSec: 5
timeoutSec: 5
unhealthyThreshold: 2
healthyThreshold: 2
```

- `health_check` : nom du module,
- `type` : type du module,
- `healthCheckModule` : définition du healthCheckModule,
- `host` : url ou IP que le health check doit appeler (ici, on définit l'IP de notre load balancer),
- `path` : le chemin à tester,
- `checkIntervalSec` : fréquence en seconde d'appel,
- `timeoutSec` : temps en seconde au bout duquel l'appel est considéré en timeout,
- `healthyThreshold` & `unhealthyThreshold` : nombre de fois où le test est ok (ou ko) avant de déclarer une instance up (ou down).

Déclarer un firewall

```
firewall:
 type: FIREWALL
 firewallModule:
   network: default
   sourceRanges: [ 0.0.0.0/0 ]
   allowed: [ {
IPProtocol: tcp,
    ports: [ 8080 ]
   } ]
```

- `firewall` : nom du module,
- `type` : type du module,
- `firewallModule` : définition du firewallModule,
- `network` : le network concerné par notre règle,
- `sourceRanges` : plages d'IP depuis lesquelles on accepte du trafic,
- `allowed` : définition du trafic que l'on accepte.

Déclarer de l'auto scaling

```
webserver_autoscale:
```

```
type: AUTOSCALING
autoscalingModule:
  minNumReplicas: 2
  maxNumReplicas: 6
  targetUtilization: 0.8
  signalType: AVG_CPU
  targetModule: nodejs
```

- `webserver_autoscale` : nom du module d'auto scaling,
- `type` : type du module,
- `autoscalingModule` : définition du autoscalingModule,
- `minNumReplicas` : nombre minimum de replica de l'architecture,
- `maxNumReplicas` : nombre maximum à attendre (il s'agit d'une pratique pour plafonner, car plus on aura d'instance plus on payera),
- `targetUtilization` : de 0 à 1, permet de mettre un niveau d'utilisation du CPU (ici 80%),
- `signalType` : actuellement, seulement AVG_CPU est supporté, qui correspond à la charge CPU,
- `targetModule` : définition du module qui doit être répliqué.

Lancer le déploiement

Vous pouvez maintenant lancer le déploiement. Il s'effectue en 2 étapes:

Etape 1

Enregistrer son template dans le Deployment Manager en effectuant la commande suivante :

```
gcloud preview deployment-manager templates create --template
-file my-first-deployment.yaml mynewtemplate
```

On enregistre avec cette commande notre fichier YAML my-first-deployment.yaml sous le nom mynewtemplate au sein du Deployment Manager. Avec cette commande, il y a aussi une validation syntaxique de votre fichier.

Etape 2

Lancer le déploiement avec cette commande :

```
gcloud preview deployment-manager deployments --region us-
central1 create --template mynewtemplate nodejs_deployment
```

Vos composants vont être créés au sein de Compute Engine. Votre architecture est maintenant prête!

Lien utile : https://developers.google.com/deployment-manager/ : Home Deployment Manager

Combien ça coûte ?

La problématique du coût est récurrente lorsque l'on parle de Cloud. Pour une meilleure compréhension du coût de Compute Engine, Google a mis en place un price calculator. Vous pouvez le trouver ici : https://cloud.google.com/products/calculator/

Donnons l'exemple d'une infrastructure de base dans l'auto scaling. Même si l'auto scaling n'est pas pris en compte, celà permet d'avoir une bonne idée du prix de l'infrastructure, Fig.3.

DOCKER, KUBERNETES ET GOOGLE CLOUD

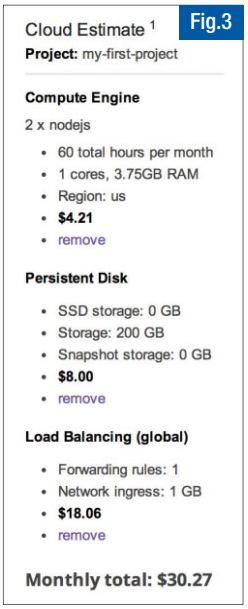
Association de Docker et Google Compute Engine (GCE)

Docker est une révolution dans le monde de la virtualisation, Google fait partie de ceux qui placent beaucoup d'espoirs dans cette technologie. Exploitant les possibilités offertes par Linux, en particulier LXC et CGroup, Docker permet de déployer des *containers*: machines virtuelles n'embarquant pas de système d'exploitation, mais exploitant les ressources du système hôte directement. Les avantages retirés sont l'allégement et l'accélération du déploiement de *container*.

Il est donc possible de déployer sur GCE une VM avec un Docker prêt à l'emploi et un agent permettant de monitorer les *containers*.

La commande suivante permet de déployer ladite VM :

Fig.3




```
gcloud compute instances create docker-instance --image container
-vm-v20140710 --image-project google-containers --zone europe
-west1-a --machine-type f1-micro
```

Une fois l'instance démarrée, connectez-vous dessus à l'aide de la commande Gcutil SSH, vous pourrez vérifier que Docker est bien présent :

```
$ gcutil ssh docker-instance
docker-instance$ sudo docker info
```

```
Gerome@docker-instance:~$ sudo docker info
Containers: 0
Images: 0
Storage Driver: aufs
 Root Dir: /var/lib/docker/aufs
 Dirs: 0
Execution Driver: native-0.2
Kernel Version: 3.14.0-0.bpo.1-amd64
WARNING: No swap limit support
```

Cette instance est prête à héberger des containers Docker. On parle cependant d'une seule instance. Plutôt que de lancer vos instances puis vos containers sur chacune d'entre elles, il serait préférable de travailler avec un cluster d'instances GCE.

Kubernetes

Google a une certaine expérience avec les containers puisque toutes ses applications sont packagées dans ces derniers: le service de recherche, Gmail, etc. Pour déployer et assurer la scalabilité de ses services, Google utilise son propre système: Omega. Omega est néanmoins un système trop complexe et surtout trop critique pour être mis à disposition du grand public. Google a donc décidé de développer un nouvel outil, Kubernetes, qui permet de gérer un cluster de containers. Toujours en version bêta, Kubernetes se veut très ouvert afin d'être enrichi par la communauté et de s'adapter à d'autres plateformes que le Cloud de Google. Ce système est capable de travailler avec Vagrant, mais aussi avec le Cloud de Microsoft, Azure. Dans cette aventure, Google a réussi à entrainer avec lui Microsoft, IBM, Red Hat et, bien sûr, Docker. L'atout de Kubernetes est de fournir des mécanismes de redémarrage automatique, d'ordonnanceur et de réplication faisant de lui plus qu'un simple orchestrateur.

Cette version bêta, comme tout nouveau projet, manque encore de documentation. On trouve néanmoins sur le GitHub du projet des exemples à dérouler qui nous donnent un aperçu des possibilités de

Kubernetes, en particulier l'exemple Guestbook. Après le démarrage d'un cluster d'instances composé d'un master et quatres "minions", Kubernetes se montre capable de déployer plusieurs containers dans le cluster. Le choix de la répartition entre les instances est décidé par Kubernetes. D'après Google, cet algorithme de répartition sera amélioré à terme afin de prendre en compte la charge des machines notamment.
En supposant que vous ayez déjà un compte GCE configuré, il vous suffit de clôner le projet GitHub de Kubnernetes et de lancer la commande suivante :

```
$ cluster/kube-up.sh
```

Cela va créer un cluster sur GCE avec la configuration suivante : Fig.4.
Un master est alors créé ainsi que quatre minions.
Vous pouvez alors interagir avec le cluster via l'outil en ligne de commande `cluster/kubecfg.sh`.
Pour le détruire :

```
$ cluster/kube-down.sh
```

Fonctionnement de Kubernetes

Master instance

Le master est composé d'une API permettant de gérer trois ressources principales qui sont :

- Pods,
- Replication controllers,
- Services.

Minion instance

Sur chaque minion, Docker est disponible et permet d'exécuter des containers. Ces derniers, s'ils sont fortement liés entre eux (i.e. des containers qui doivent interagir ensemble, scaler ensemble) sont rassemblés au sein d'une unité nommée **Pod**. Les minions vont alors contenir un ou plusieurs pods, chacun constitué d'un ou plusieurs containers Docker.
Un autre composant est également installé sur chaque minion: **Kubelet**. Cet agent a pour rôle de s'assurer que le Pod est bien bien configuré, démarré et stable dans sa configuration lorsqu'il fonctionne. La configuration en question est décrite à l'aide de fichiers manifest décrits au format YAML (https://developers.google.com/compute/docs/containers/container_vms#container_manifest).

```
Gerome:~ $ gcutil listinstances
+---------------------+---------------+---------+---------------+---------------+
| name                | zone          | status  | network-ip    | external-ip   |
+---------------------+---------------+---------+---------------+---------------+
| kubernetes-master   | europe-west1-a | RUNNING | 10.240.142.32  | 23.251.137.182 |
+---------------------+---------------+---------+---------------+---------------+
| kubernetes-minion-1 | europe-west1-a | RUNNING | 10.240.69.157  | 130.211.86.94  |
+---------------------+---------------+---------+---------------+---------------+
| kubernetes-minion-2 | europe-west1-a | RUNNING | 10.240.121.126 | 130.211.90.64  |
+---------------------+---------------+---------+---------------+---------------+
| kubernetes-minion-3 | europe-west1-a | RUNNING | 10.240.0.229   | 146.148.25.38  |
+---------------------+---------------+---------+---------------+---------------+
| kubernetes-minion-4 | europe-west1-a | RUNNING | 10.240.181.46  | 130.211.80.247 |
+---------------------+---------------+---------+---------------+---------------+
```

Fig.4 *Architecture du cluster Kubernetes*

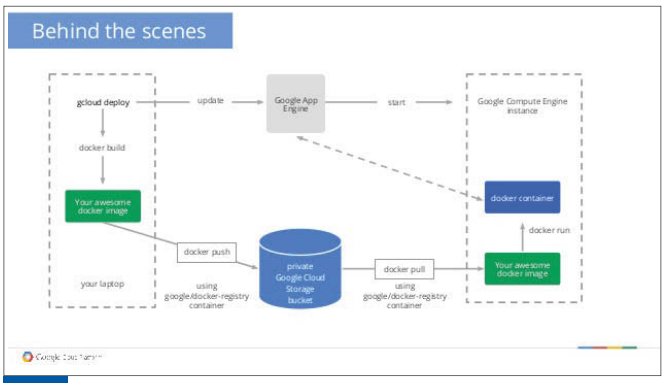


Fig.5 *Intégration de Docker dans App Engine*

Label

Il est possible d'affecter des labels aux Pods afin de faciliter la configuration des services ou des replication controllers. Ces derniers disposent dans leur configuration d'une propriété label selector qui permet d'affecter des Pods selon leurs labels.

Replication controller

Lorsqu'un replication controller est créé, on lui associe un template et un nombre de réplicas souhaités. Le controller s'assure du nombre minimum de réplicas souhaités dans le cluster. S'il détecte une erreur, il démarrera de lui même un nouveau replica.

Kubernete Service

Le load-balancer est un service proposé dans Kubernete. Celui-ci va gérer le trafic d'un ou plusieurs Pods. Lors de la création d'un service, on associe un port aux Pods sélectionnés, en se basant sur leurs labels. Une fois le service créé, tous les Pods du cluster peuvent y accéder par le port indiqué dans la configuration.

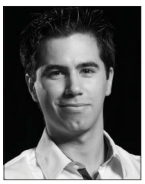
Docker et App Engine Fig.5.

Comme expliqué dans la partie sur les Managed VMs d'App Engine, le PaaS du Google Cloud devient de plus en plus souple. La prochaine étape est de déployer un conteneur docker directement dans App Engine Comme pour une Managed VM, il faut spécifier l'option VM à true et préciser que le runtime voulu est custom. Ce n'est pas plus compliqué que ça! Cette fonctionnalité, non disponible à l'heure actuelle, ne devrait plus tarder.
Liens utiles :
https://github.com/GoogleCloudPlatform/kubernetes : GitHub de Kubernetes
http://www.slideshare.net/dotCloud/google-app-engine-at-dockercon-14 : présentation de Google sur App Engine et Docker

Conclusion

Le Google Cloud propose un vaste panel de services aux développeurs et essaie de faciliter les interactions avec ceux-ci en proposant un nombre impressionnant de clients et langages différents. De plus, avec l'arrivée des Managed VMs, Google tente d'effacer l'image négative du Lockin en proposant un PAAS qui a la souplesse d'un IAAS tout en gardant les facilités de déploiement, de monitoring et de scaling automatique. En misant sur Docker, Google veut clairement être le plus standard possible.
Malgré son jeune âge, Compute Engine est déjà très attractif. On sent la longue expérience de Google en matière de gestion d'infrastructure. La simplicité avec laquelle on peut monter une infrastructure en est la preuve et on a hâte de voir ce que va nous proposer Google dans un futur proche. Rendez-vous le 4 novembre prochain pour la 2ème édition du **Cloud Platform Day**, où, nous en sommes sûrs, Google aura encore beaucoup d'annonces à nous faire!


E. Briand


G. Egron


J.B. Claramonte

Xebia

Amazon AppStream : ou comment exécuter des applications 3D sur tous les terminaux

Lorsque nous développons une application qui requiert des capacités graphiques (jeux, modélisation, rendu 3D temps réel, etc.) nous nous posons toujours la question de la capacité des terminaux à l'exécuter ou à l'afficher. Vient alors le choix difficile du sous-ensemble des terminaux qui, techniquement, sont aptes à exécuter notre application sans à-coups et dans les meilleures conditions visuelles pour nos utilisateurs.

En effet, si je choisis tel ou tel terminal parmi les PC, Mac, iOS, Android, serai-je à même de toucher la totalité de mes utilisateurs ? Suis-je limité à une petite frange de la population si mon application (ou mon jeu) est gourmande en ressources graphiques ? De même, si je choisis plusieurs technologies de terminaux, devrai-je maintenir plusieurs versions de mon application ? Ces questions peuvent être abordées plus sereinement grâce au service AppStream qui permet de diffuser (streamer) vos applications depuis le cloud vers des périphériques exécutant FireOS, Android, iOS, OS X et Windows. Vous allez ainsi pouvoir adapter vos applications existantes pour les streamer depuis le cloud d'Amazon en utilisant le kit de développement logiciel (SDK) qui est fourni. Les cas d'usage de cette technologie sont très nombreux, parmi lesquels on peut retrouver : La CAO, la 3D ainsi que les outils de simulation. Ces applications, très gourmandes en ressources graphiques, pourront être exécutées côté serveur en déportant leur affichage sur des périphériques légers et plus maniables que de coûteuses stations graphiques. La mise en œuvre souvent lourde (par le téléchargement de fichiers volumineux) de ce genre d'applications se réduit donc à l'installation d'un client de streaming AppStream. Grâce au SDK AppStream vous pouvez déporter une partie des scènes de vos jeux sur AppStream et rendre l'expérience utilisateur « Instant On » en leur évitant de patienter pendant le chargement des assets du jeu. Nous vous invitons à découvrir le témoignage de l'éditeur islandais CCP, créateur du jeu EVE Online, ici : http://youtu.be/V0lg206whll. Dans cet article nous allons parcourir les concepts de base du service AppStream ainsi que l'utilisation de son SDK pour développer une application streamée vers des terminaux mobiles.

AppStream : concepts de base (Fig.1)

AppStream est un système qui permet de déployer vos applications dans le cloud d'Amazon, de les rendre scalables et d'assurer leur streaming vers vos utilisateurs, où qu'ils soient, quel que soit leur terminal et quel que soit leur nombre.

Vous pouvez non seulement adapter vos applications existantes pour les streamer depuis AppStream, mais surtout imaginer une nouvelle génération d'applications « hybrides » où l'action se déroule sur le terminal de l'utilisateur et où les lourds calculs d'affichage se font par exemple côté serveur. Vous pourrez ainsi, en une seule opération, vous affranchir de la fragmentation des terminaux mobiles, bénéficier d'une sécurité améliorée ainsi que de cycles de déploiement et de mise à jour accélérés ! Regardons maintenant de quoi est composée une application AppStream.

Les composants d'une application AppStream (Fig.2)

Une application AppStream est composée de plusieurs éléments qui collaborent entre eux pour fournir un service de streaming de votre application. Ces éléments sont les suivants :

- ***Les serveurs AppStream*** : ce sont des instances Amazon EC2 à très forte capacité graphique (Instances de la famille g2 hébergeant des cartes graphiques) qui vont exécuter votre application et assurer son élasticité,
- ***L'application*** à proprement parler : il s'agit ici de votre code applicatif. Ce peut être une application que vous aurez adaptée pour AppStream ou une toute nouvelle application. Dans les deux cas, votre application appellera les fonctions de l'API AppStream `XStxServerAPI` pour envoyer les flux audio et vidéo vers les terminaux de vos utilisateurs et sera appelée par les fonctions de call back de la même API pour répondre aux événements générés par les utilisateurs (saisie utilisateur, click, swipe, etc.). Nous verrons le SDK en détail dans les paragraphes suivants.
- ***Les Clients*** : ce sont les applications clientes installées sur les postes de travail et les terminaux mobiles de vos utilisateurs. Elles ont pour rôle d'encoder et décoder les flux audio et vidéo de votre application. Ces applications utilisent, elles aussi, le SDK Amazon AppStream et instancient l'API `XStxClientAPI` pour recevoir les flux audio/video et envoyer les entrées utilisateurs vers ***l'Application***.
- ***Le service d'autorisation*** : toute application publique se doit d'avoir un service pour y contrôler l'accès. C'est le rôle du ***service d'autorisation*** (entitlement service). Il s'agit d'un service Web que vous développez et qui assure l'authentification des utilisateurs (en utilisant un service tiers comme Login With Amazon, Google+ ou Facebook, par exemple) et leur autorise l'accès à votre application. C'est aussi l'endroit où vous ajouterez vos règles de facturation, si vous souhaitez monétiser votre application.

Fig.1

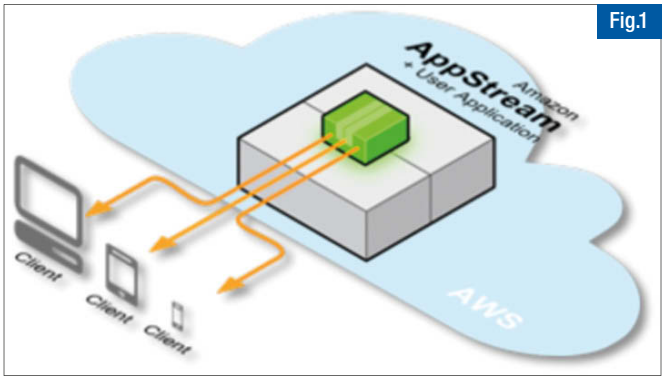
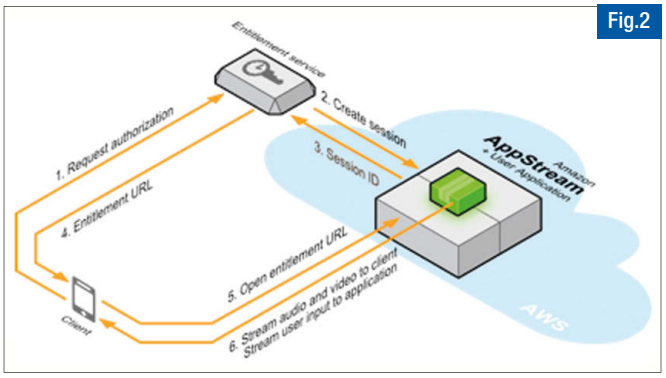


Fig.2



La cinématique de l'application sera alors la suivante :

1 – Votre utilisateur utilise une ***application cliente*** pour s'authentifier auprès du ***service d'autorisation***.

2 – Le service d'autorisation valide l'utilisateur et fait une demande de création auprès des ***serveurs AppStream***.

3 – Un identifiant de session unique est généré par les ***serveurs AppStream*** et retourné au ***service d'Autorisation***.

4 – Le service d'Autorisation construit une URL signée, sécurisée qui est retournée à ***l'application Cliente***.

5 – L'application Cliente utilise cette URL pour contacter les Serveurs de manière sécurisée.

6 – Les ***serveurs*** reconnaissent l'utilisateur, l'application et l'ID de session et commencent à streamer les flux Audio et Video vers ***l'application cliente***.

L'application de cette cinématique a de nombreux avantages dont les deux principaux suivants :

- Séparation de la partie authentification du code applicatif,
- Délégation de la partie « scalabilité » de votre application aux services Amazon AppStream : à chaque demande d'ouverture de session, Amazon AppStream gère le parc de serveurs nécessaires à l'absorption de la charge générée par vos utilisateurs.

Construire une application AppStream

L'application streamée est le cœur d'AppStream. Afin d'être streamée, votre application doit :

- Faire des appels d'initialisation au service AppStream,
- Indiquer à AppStream qu'elle est prête à ouvrir des sessions utilisateur,
- Implémenter préalablement les interfaces nécessaires pour que AppStream puisse la notifier de l'arrivée de nouveaux utilisateurs et commencer le streaming des flux audio et vidéo.

Tout ceci se fait par l'intermédiaire des APIs disponibles dans SDK AppStream Fig.3.

Comme on peut le voir sur ce schéma, pour être streamée, votre application doit référencer un certain nombre d'objets de l'API et se conformer à un certain nombre d'interfaces de la même API.

Le tableau suivant donne la description des objets qui doivent être référencés par votre application :

Une fois que vous avez implémenté les interfaces nécessaires et référencé les objets du SDK, vous allez pouvoir gérer les événements AppStream et commencer à streamer vos frames vidéo.

Les paragraphes suivants donnent des extraits de code illustrant les différentes étapes à suivre pour implémenter une application streamée.

Initialisation d'une Application Streamée

Avant toute chose, votre application doit construire l'objet de plus haut niveau du SDK AppStream ; celui qui vous permettra d'initialiser votre application et d'obtenir les objets de gestion de sessions.

L'exemple ci-dessous montre l'initialisation d'une application :

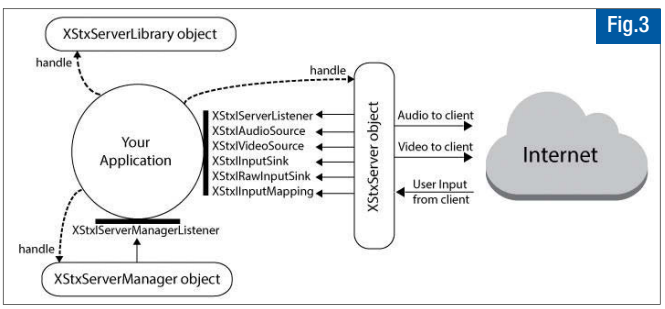


Fig.3

Objet	Description
`XStxServer`	C'est l'objet que votre application utilise pour recevoir et envoyer les flux video/audio vers les applications clientes
`XStxServerManager`	Il s'agit de l'objet qui gère les sessions client et les événements qui s'y rapportent. C'est par exemple le callback `XStxIServerManagerListener FcnServerInitialize` qui vous notifie de l'ouverture d'une session et qui vous fournit une instance de l'objet `XStxServer` pour streamer vos flux audio/video.
`XStxServerLibrary`	Objet de plus haut niveau, c'est par son intermédiaire que vous obtenez une instance de l'objet `XStxServerManager`

Les interfaces minimales à implémenter sont les suivantes (d'autres interfaces sont disponibles mais ne sont pas requises pour le fonctionnement minimal d'une application streamée) :

Interface	Description
`XStxIInputSink`	Reçoit les données utilisateur fournies par `XStxServer` (clavier, souris, touch)
`XStxIServerListener`	Reçoit les événements de `XStxServer` lorsqu'il établit ou lorsqu'il perd la connexion avec des applications clientes
`XStxIServerManagerListener`	Reçoit les événements de `XStxServerManager` lorsqu'il ouvre ou ferme des sessions client.
`XStxIVideoSource`	Appelée par `XStxServer`, cette interface permet d'obtenir les frames vidéo à envoyer aux applications clientes

```
int runAsAppStreamGame(int argc, const char* argv[]) {
...
     /** Initialize the XStxServer library */
     XStxResult result = XStxServerLibraryCreate(
          XSTX_SERVER_API_VERSION_MAJOR,
          XSTX_SERVER_API_VERSION_MINOR,
          &serverLibraryHandle);
     if (result != XSTX_RESULT_OK)
     {
          goto exit;
     }

  ...
}
```

Streaming de frames vidéo vers les applications clientes

Pour envoyer les frames vers les applications clientes, quelques opérations sont à réaliser au préalable :

- Définir la source de vidéo : il s'agit ici d'indiquer à `XStxServer` où aller « demander » les frames vidéo à pousser vers les applications clientes,
- Choisir un mode de streaming : on indique à `XStxServer` s'il doit venir chercher les frames vidéo, ou bien si c'est l'application qui fournira les frames vidéo à `XStxServer`. Ceci permet, entre autres, de mieux maitriser le taux de rafraîchissement des images poussées vers les applications clientes,
- Choisir une gamme de couleurs : AppStream vous donne la possibilité de choisir votre espace colorimétrique afin de vous offrir le meilleur rendu sur les terminaux de vos clients. Vous pourrez ainsi choisir entre YUV444 ou YUV420 pour trouver le meilleur équilibre entre espace

colorimétrique et volumétrie des images poussées vers les applications clientes.

Les chapitres suivants donnent des exemples d'implémentation de ces opérations :

Choix du mode de streaming

```
static XStxResult getVideoMode(void* context, XStxVideoMode* mode)
{
// We push frames to AppStream when we want to.
// AppStream will adapt by dropping frames if we're too fast
    *mode = XSTX_VIDEO_MODE_PUSH_IMMEDIATE;
    return XSTX_RESULT_OK;
}
```

Dans l'exemple ci-dessus l'application fixe le mode de streaming à `PUSH_IMMEDIATE` : vous fournissez les frames de votre vidéo à votre vitesse et AppStream gèrera le frame rate optimal vers les applications clientes.

Choix de la gamme de couleurs

Pour ce faire, votre application doit appeler la fonction `XStxServerAddChromaSamplingOption()` pour notifier AppStream de votre choix de gamme colorimétrique. Lorsqu'une application cliente ouvre une session, Amazon AppStream vérifie les options colorimétriques disponibles pour l'application cliente et pour l'application streamée : si plusieurs options sont disponibles, AppStream choisit toujours le profil colorimétrique le plus riche et notifie l'application streamée du choix de profil an appelant le callback `XStxIServerListener2FcnServerConfigurationSettingsReceived`, ce qui vous permet de vous adapter dynamiquement à chaque type d'application cliente.

L'extrait de code suivant décrit la fonction `XStxServerAddChromaSamplingOption` :

```
/**
 * Inform the server of server application's chroma sampling capability.
 * You can call this as many times as you want for each chroma sampling
 * listed in XStxChromaSampling. If this method is never called, then
 * XSTX_CHROMA_SAMPLING_YUV420 will be used by default chroma sampling.
 * Register a callback function at XStxIVideoSourceFcnSetChromaSampling
 * so that STX server can notify server application which chroma sampling
 * will be used for streaming.
 * @param[in] serverHandle The handle of the server.
 * @param[in] chromaSampling The chroma sampling scheme. Look at XStxAPI.h
 * @return This function will return one of these values.
 * Return code                | Description
 * -------------------------- | --------------------------
 * XSTX_RESULT_OK             | The operation is successful.
 * XSTX_RESULT_INVALID_HANDLE | serverHandle is invalid.
 * XSTX_RESULT_INVALID_ARGUMENTS| chromaSampling is not recognized.
 */
XSTX_API_EXTERN XStxResult XSTX_API XStxServerAddChromaSamplingOption(
    XStxServerHandle serverHandle,
    XStxChromaSampling chromaSampling);
```

Envoyer des frames vidéo aux applications clientes

Chaque image générée par votre application doit être envoyée vers les applications clientes. Pour cela vous pouvez utiliser soit la fonction XStxServerPushVideoFrame à chaque image créée, soit être appelé directement par XStxServer à chaque fois qu'il doit pousser une image/frame vers les applications clientes.

L'extrait de code suivant indique, comment modifier une application existante pour ajouter une conversion d'espace colorimétrique et l'envoi de chaque image créée par l'application :

```
...

EnterCriticalSection(&g_frameCriticalSection); // Don't want
to be interrupted now

// Copy back buffer data. Can also use D3DXLoadSurfaceFrom
Surface if we need to resize/change pixel format
g_D3DDevice->GetRenderTargetData(g_backBuffer, g_memBuffer);
D3DLOCKED_RECT lockedRect;
g_memBuffer->LockRect(&lockedRect, NULL, D3DLOCK_READONLY);
// Convert to YUV so we can supply it to AppStream
switch (g_chromaSamplingType)
{
    case XSTX_CHROMA_SAMPLING_YUV420:
        convertToYUV420((unsigned char*)lockedRect.pBits,
WINDOW_WIDTH, WINDOW_HEIGHT, 2, 1, 0, 4, lockedRect.Pitch,
g_videoFrame.mPlanes);
        break;
    case XSTX_CHROMA_SAMPLING_YUV444:
        convertToYUV444((unsigned char*)lockedRect.pBits,
WINDOW_WIDTH, WINDOW_HEIGHT, 2, 1, 0, 4, lockedRect.Pitch,
g_videoFrame.mPlanes);
        break;
    default:
       assert(!"Unknown chroma sampling type"); // Make sure
 we don't get an unknown chroma sampling type
}
g_memBuffer->UnlockRect();

XStxServerPushVideoFrame(g_serverHandle, &g_videoFrame); //
Push the video frame

LeaveCriticalSection(&g_frameCriticalSection);

...
```

Ainsi, chaque image générée par l'application est collectée par AppStream et réencodée en flux vidéo vers les applications clientes : vous ne modifiez pas votre application pour y ajouter des méthodes d'encodage/décodage, c'est le rôle d'AppStream.

Conclusion

Nous avons vu comment modifier une application existante pour y ajouter les fonctionnalités qui assureront son fonctionnement et son streaming dans AppStream. Il est, bien sûr, possible de créer des applications de toutes pièces pour les distribuer...

Un exemple complet d'application est compris dans la documentation du SDK AppStream et est disponible ici : http://docs.aws.amazon.com/AppStream/latest/developerguide/AppStream-downloads.html#AppStream-downloads-sdk

Vous pouvez le télécharger gratuitement pour tester l'application exemple pour la streamer sur des terminaux.

Pierre **Gilot**,
Solutions Architecte AWS

Azure DocumentDB : une base NoSQL

Cet été, Microsoft a annoncé la disponibilité en version préliminaire de DocumentDB. Il s'agit d'une base de données noSQL permettant de stocker des documents JSON. Dans cet article, je vous propose de voir ensemble le positionnement de DocumentDB et comment on peut l'utiliser. Nous verrons également des exemples de code.

Au sein des moteurs noSQL, il s'agit d'une des offres de bases de données orientées documents. Comme MongoDB par exemple, DocumentDB stocke du JSON, et permet l'exécution de code côté serveur, écrit en JavaScript. Certaines des spécificités décrites ci-après sont en revanche des différences par rapport à MongoDB.

On notera que MongoDB, tout comme d'autres bases noSQL, est aussi disponible en tant que service dans Azure, via des offres partenaires qu'on retrouve sur http://azure.microsoft.com/fr-fr/gallery/store/.

Pour la base MongoDB, il y a les offres des partenaires MongoDB et MongoLab. En revanche, Microsoft ne propose pas directement de MongoDB as a service.

Au sein du cloud Azure, on peut comparer DocumentDB par rapport à SQL Server dans des VM, SQL Databases, les tables et les blobs du stockage Azure.

Bien que ce soit une base de données non relationnelle, DocumentDB gère des transactions et des requêtes riches (avec des WHERE, OR, AND par exemple). DocumentDB ne stocke pas, contrairement aux blobs ou tables, que du JSON. Comme SQL Databases, tables et blobs, c'est un service PaaS géré par Microsoft, accessible en HTTP REST(1).

Dans les choix effectués lors de la conception de DocumentDB, deux éléments sont importants : il n'y a pas besoin de préciser un schéma JSON à l'avance, et le développeur peut lui-même faire son choix entre performance et cohérence.

Pour le fonctionnement sans schéma, par défaut, il suffit de stocker un document JSON pour pouvoir requêter sur n'importe lequel de ses attributs; ceci puisque la base indexe tout le document grâce à un moteur spécialement étudié pour cela, en particulier en termes de performances.

(1) SQL Database est accessible en HTTP REST pour le management. L'accès aux données se fait en binaire sur des sockets.

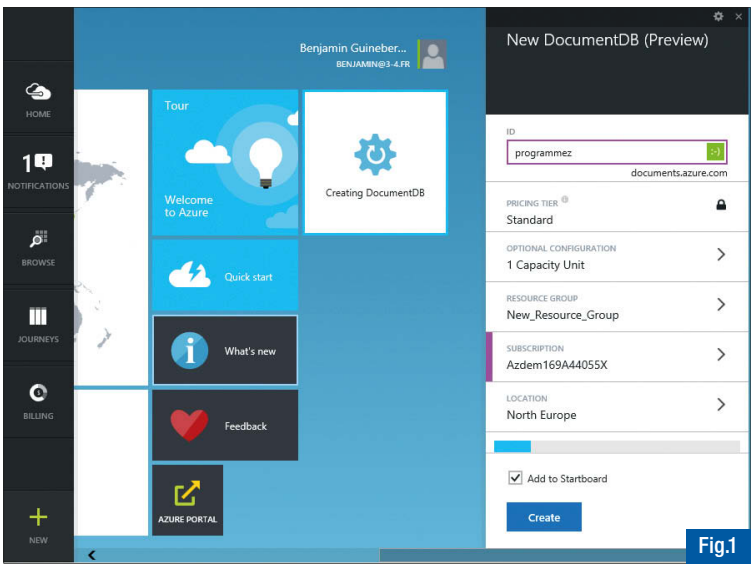


Fig.1

Pour les niveaux de cohérence, il en existe quatre :

- « strong » qui fait des écritures synchrones sur une majorité de nœuds et demande à une majorité de nœuds pour la lecture,
- « bounded » qui garantit l'ordre des écritures avec réplication asynchrone et demande à une majorité pour la lecture,
- « session » où ce qu'un client lit est cohérent avec ce qu'il a lui-même écrit,
- « eventual » où la cohérence entre écritures et lecture finit par arriver. Evidemment, moins on est exigeant sur la cohérence, plus le système est rapide.

Parmi les autres caractéristiques de DocumentDB, on trouvera la possibilité de stocker des pièces attachées à un document (elles sont alors stockées dans le service de blobs Azure), et la disponibilité d'un langage SQL de requêtage, comme beaucoup de bases noSQL du marché (ex : Cassandra) pour des raisons que je ne développerai pas ici !

Comment démarrer

Pour démarrer, il faut disposer d'un abonnement Azure. Si vous n'en n'avez pas, vous pouvez souscrire à une offre d'essai gratuite qui comprend 150 € de ressources pendant 1 mois. Il suffit pour cela de suivre le lien en haut à droite de la page http://azure.com.

La création d'un compte DocumentDB se fait depuis le nouveau portail à https://portal.azure.com. Il suffit de choisir « New », « DocumentDB », puis de remplir les valeurs demandées. Par exemple : Fig.1.

Oui, mais combien ça coûte ? Le prix est, dans la version préliminaire qui inclut 50% de remise, de 0,55 € par jour (16,76 € par mois) par unité de capacité (10 Go, 2000 unités de demande / seconde). Les détails et le prix le plus à jour sont disponibles à http://azure.microsoft.com/fr-fr/pricing/details/documentdb/.

Une fois le compte créé, le portail fournit des informations, dont les clefs d'accès primaire et secondaire Fig.2. Il y a également la vignette « Quick Start » qui donne tout un ensemble de liens pour démarrer rapidement : documentation, SDK en .NET, Node.js, JavaScript et Python, forum, prix, exemples de code, ... et d'ailleurs, il est temps de passer au code.

Exemples de code

L'exemple de code .NET qu'on trouve dans « Quick Start » montre différents aspects : Fig.3. Pour exécuter ce code sur votre compte DocumentDB, il suffit de copier les valeurs URI, Primary Key dans le fichier appSettings.config et d'exécuter. Le compte permet de créer des bases de données :

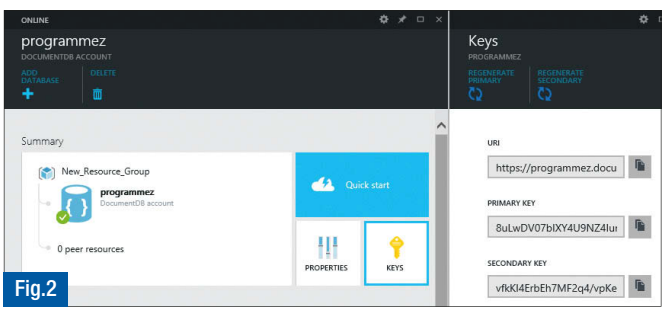


Fig.2

```
database = await client.CreateDatabaseAsync(new Database { Id = id });
```

Dans une base, on peut créer des collections (l'équivalent des tables d'une base relationnelle) :

```
collection = await client.CreateDocumentCollectionAsync(dbLink,
 new DocumentCollection { Id = id });
```

Dans une collection, on peut insérer / mettre à jour des documents, comme dans l'exemple ci-dessous :

```
//Create a dynamic object,
dynamic dynamicOrder = new
{
    id = "DYN01",
    purchaseOrderNumber = "PO18009186470",
    orderDate = DateTime.UtcNow,
    total = 5.95,
};

Document createdDocument = await client.CreateDocumentAsync(colSelfLink,
dynamicOrder);

//get a dynamic object
dynamic readDynOrder = (await client.ReadDocumentAsync(created
Document.SelfLink)).Resource;

//update a dynamic object
readDynOrder.ShippedDate = DateTime.UtcNow;
await client.ReplaceDocumentAsync(readDynOrder);
```

On peut requêter en LINQ :

```
query = client.CreateDocumentQuery<Family>(colSelfLink)
  .Where(f => f.Id == "AndersenFamily" || f.Address.City == "NY")
  .Select(f => new { Name = f.LastName, City = f.Address.City });
```

Ou en SQL:

```
var q = client.CreateDocumentQuery(colSelfLink,
"SELECT f.LastName AS Name, f.Address.City AS City " +
       "FROM Families f " +
       "WHERE f.id='AndersenFamily' OR f.Address.City='NY'");
```

Evidemment, ce n'est pas parce qu'on a du SQL que tout est permis. Par exemple, les jointures ne sont possibles qu'au sein d'un même document. Interdire les jointures entre documents différents permet de garder la souplesse d'une montée en charge horizontale, typique dans les bases noSQL.

Il y a bien d'autres exemples de code dans cette solution, et je préfère vous laisser les découvrir et vous montrer un peu de code en Python, ce qui sera plus lisible que du REST dans Fiddler.

Là encore, je repars du portail et suis le lien d'installation du SDK Python : **Fig.4**.

L'exemple de code suivant :

```
import pydocumentdb.documents as documents
import pydocumentdb.document_client as document_client
import pydocumentdb.errors as errors
import pydocumentdb.http_constants as http_constants


# localhost
masterKey = '8uLwDV07bIXY4U9NZ4IunA7iOjLMjllyO5oZzdXpQcHhhbY
fO1aprD6PiXu3ZtzxELRf0s83nac7OnAJ5rp67A=='
host = 'https://programmez.documents.azure.com:443'

with document_client.DocumentClient(host, {'masterKey': masterKey})
as client:
  # create database
  db = client.CreateDatabase({ 'id': 'sample database' })
  # create collection
  collection = client.CreateCollection(
    db['_self'],
    { 'id': 'sample collection' })
  # create document
  document = client.CreateDocument(collection['_self'],
{   'id': 'sample document',
'foo': 'bar',
      'key': 'value' })
```

Il génère la base, la collection et le document visibles via l'outil disponible à https://github.com/mingaliu/DocumentDBStudio. **Fig.5**.

Je vous engage à tester par vous-même, regarder les exemples de code, découvrir les triggers, les procédures stockées, les pièces attachées, les UDFs, les utilisateurs et permissions, ...

Benjamin **Guinebertière** - *Evangéliste technique Azure*.
J'accompagne techniquement des startups et autres entreprises de tailles différentes qui vont vers le cloud Microsoft Azure, avec ou sans Big Data ou Machine Learning. Egalement, je parle (conférences), écris (blogs, ...), et prends les feedbacks des utilisateurs.
Vous pouvez me retrouver à http://3-4.fr.

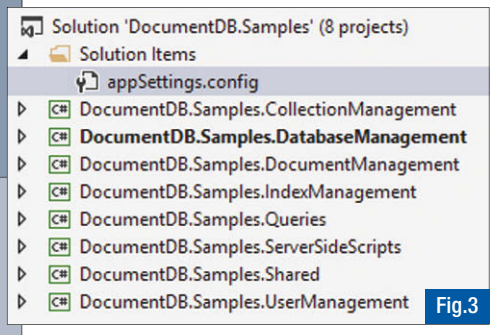


Fig.3

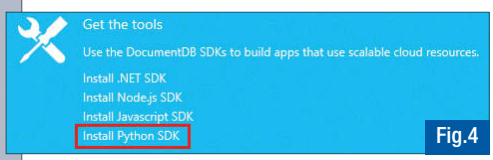


Fig.4

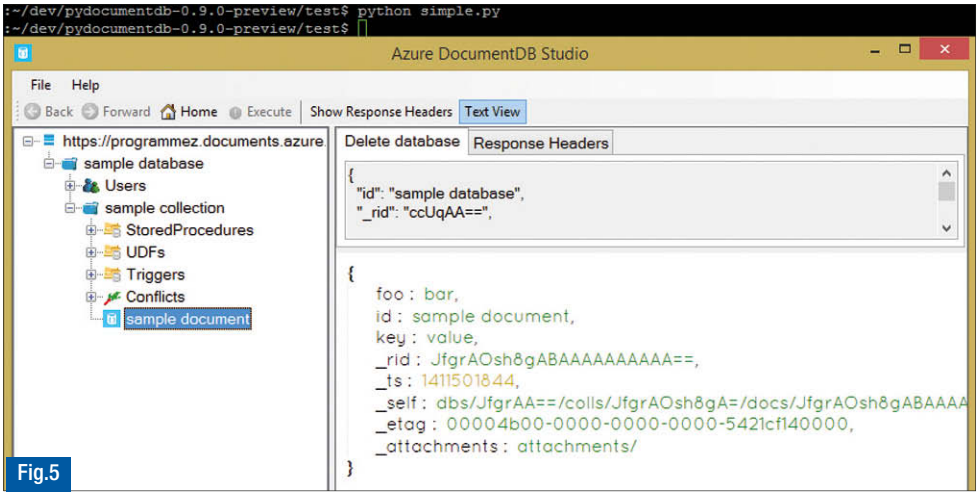
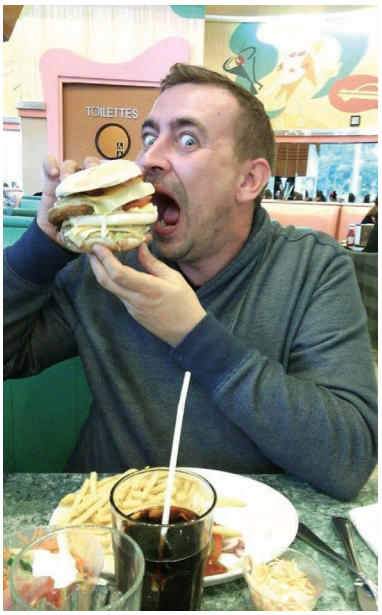


Fig.5

Témoignage

Programmation Cloud



Nombre de mes clients ne veulent plus investir en serveurs, licences, sauvegarde, maintenance, ... C'est pourquoi les solutions que je développe sont maintenant orientées Cloud : juste de la location de services, pas d'investissement en matériel, licence ou autre gestion IT, extensibilité, sécurité, ... Pour ce faire, j'ai choisi Azure pour plusieurs raisons : rapide à mettre en route, intégration avec Visual Studio, usage de TFS Online, gratuit (j'ai la chance d'avoir un BizSpark et que l'on peut l'utiliser en production grâce au crédit offert de 115€ par mois), très fiable et le nom de Microsoft apporte une confiance en l'infrastructure plutôt qu'un hébergeur plus « local ».

La technique que j'utilise en général est très simple : je crée une base de données SQL Azure via le portail (modèle DB First), un service Web utilisant les WEB Api que je connecte à ma base de données et que je publie sur un site Web Azure. Sans écrire une seule ligne de code, j'ai donc un service Web REST (REpresentational State Transfer) qui me permet de réaliser du CRUD (Create, Read, Update, Delete) sur ma base de données SQL Azure, le tout, en XML ou en JSON (selon la méthode employée par l'application cliente pour accéder au service).

Ensuite vient l'étape de création de l'application elle-même. Les choix sont vastes ! Pour ma part, Universal App mais avec usage des PCL (Portable Class Library) afin de pouvoir également réutiliser mon code dans des applications Win32.

Ceci pour les cas spécifiques ne pouvant pas être gérés en WinRT comme l'accès direct à des périphériques non-reconnus ou des appels à des APIs qui ne sont pas encore portées en WinRT. Ce peut aussi être pour un portage futur sur Xbox.

Je place donc le maximum de code dans ma PCL (Models, ViewModels) en utilisant le pattern MVVM et son super paquet Nugget : MVVM Light ! Attention à bien utiliser la version dédiée aux PCL et ce, dans chaque projet de la solution.

Un petit mot sur les modèles. Comme l'import de votre base de données crée automatiquement les modèles de base, il serait idiot de les recréer dans votre application, d'autant que nous autres programmeurs, avons la (mauvaise) réputation d'être des fainéants et de récupérer tout ce qu'on peut pour écrire le moins possible. Pour cela, j'utilise la méthode des fichiers liés afin d'importer dans ma PCL les modèles générés pour le service REST. Je crée alors des nouvelles classes héritant de celles-ci afin de les spécialiser métier (y compris certaines propriétés spécifiques pour leur affichage telles que la visibilité, les couleurs, ... que j'utiliserai dans des converters dans mes vues). Cela me permet d'avoir toujours la même structure, de la base de données à mon application cliente.

Vous allez certainement me dire, et c'est TOUJOURS le cas quand vous proposez quelque chose de nouveau ou d'innovant, j'ai nommé le fameux « Oui mais... » : « Oui mais, quand l'utilisateur n'a pas de connexion data ? ». C'est évidemment l'un des premiers arguments avancé par mes clients. La solution est simple à mettre en place : le mode déconnecté.

Pour ce faire, il vous suffit, une fois vos données lues, d'en conserver une copie en local en sérialisant vos collections de données. Si au lancement suivant vous n'avez pas accès au service REST (problème de connexion, site en maintenance, autre erreur, ...), il vous suffit de charger votre copie locale.

Afin de (dé)sérialiser les collections, j'utilise JSON.Net de NewtonSoft (http://james.newtonking.com/json) téléchargeable via Nugget, évidemment.

Pour sérialiser une instance de la classe « Maclasse » nommée « moninstance » :

```
String s = Newtonsoft.Json.JsonConvert.SerializeObject(moninstance)
```

Pour désérialiser une chaîne « s » en classe « Maclasse » :

```
Maclasse moninstance ;
Moninstance = Newtonsoft.Json.JsonConvert.DeserializeObject<Maclasse>(s)
```

Vous n'avez plus qu'à lire et écrire sur le disque cette sérialisation pour bénéficier d'un système de cache rapide et simple à mettre en place ! N'hésitez pas à créer une petite librairie pour gérer votre gestion du cache afin, évidemment, de réutiliser cette technique au travers de vos différentes applications.

Avec cet ensemble de techniques, vous pouvez réaliser une application Windows (Phone) 8.1 en 2 petites heures facilement, en partant de rien.

Stéphane **VIDOUSE**
CEO Cassio sprl, Mons, Belgique - MCP
Microsoft Extended Expert Team Accredited Member
(http://www.microsoft.com/belux/meet/)

Votre Abonnement **PDF**

pour seulement **30 € par an**

(soit 2,73 € le numéro)

www.programmez.com

Heroku DX : une nouvelle expérience développeur

Pour aider les développeurs au quotidien, Heroku (filiale de Salesforce) vient de lancer Heroku DX. Il s'agit d'un ensemble de services pour simplifier et accélérer le développement d'applications. Les développeurs peuvent ainsi, plus que jamais, se concentrer sur la création d'applications. Guillaume Roques (en charge de la relation développeurs chez Salesforce en Europe, au Moyen-Orient et en Afrique) revient sur l'outil et les conséquences pour le développeur.

Heroku vient de lancer Heroku DX. De quoi s'agit-il ?

Heroku DX est un ensemble de services destinés à faciliter la vie des développeurs en simplifiant le processus de développement, d'exécution et de montée en charge de leurs applications. Heroku DX s'inscrit dans une démarche d'amélioration de l'expérience des développeurs qui a toujours été dans l'ADN d'Heroku depuis sa création.

Comment fonctionne-t-il ?

Heroku DX repose sur 3 services.

Premièrement, nous avons **Heroku Button**. Heroku a toujours été reconnu comme une plateforme permettant aux développeurs de très rapidement démarrer sur une nouvelle application. A tel point que beaucoup de sites proposant des tutoriaux, utilisent Heroku comme plateforme de déploiement par défaut, car, en seulement 2 étapes vous pouvez très simplement mettre votre application en ligne. Avec Heroku Button, eh bien nous avons voulu simplifier encore plus ce processus en supprimant l'étape 2. Ce service permet de déployer une nouvelle application en un seul clic. *Il n'y a plus d'étape 2 !* Le développeur n'a plus besoin de configurer la plateforme, il lui suffit de cliquer sur un bouton et Heroku Button se charge de configurer l'environnement d'exécution, les add-ons requis et de réaliser le 1er déploiement de l'application. Tout projet GitHub peut très simplement ajouter ce service par l'intermédiaire d'un fichier app.json avec quelques metadonnées. À ce jour, plus de 400 boutons sont déjà disponibles dans la galerie d'applications disponible sur http://buttons.heroku.com. On peut citer Dropbox, Twilio.org Rapid Response Kit et Uber API Python/Flask. Deuxièmement, nous avons **Heroku Dashboard + Metrics**. Il s'agit d'un tableau de bord permettant de contrôler l'activité de vos applications déployées sur Heroku. Heroku Dashboard + Metrics permet de suivre la performance de ces applications, grâce à la mesure de certaines informations : ressources consommées, délais de réponse, erreurs, mémoire... Le tableau ne se contente pas d'afficher ces métriques mais aussi de recommander des changements à apporter à vos applications pour en améliorer la performance. Le nouveau design permet d'afficher ces informations de manière intuitive, visuelle et interactive. C'est idéal pour aider les développeurs à repérer et à résoudre d'éventuels problèmes !

Pour celles et ceux qui souhaitent connaître les dessous de l'interface de ce nouveau tableau de bord, sachez qu'il a été développé avec ember.js.

Enfin, nous trouvons **Heroku Postgres DbX**. De nombreuses applications utilisent Heroku Postgres afin de gérer leurs données. Heroku Postgres DbX vient d'abord avec une nouvelle série de plans de licences qui améliorent d'un facteur 2 à 3 la performance des plans précédents, le tout pour un coût identique. Par ailleurs, les plans Premium et au-delà, ajoutent l'encryption des données stockées.

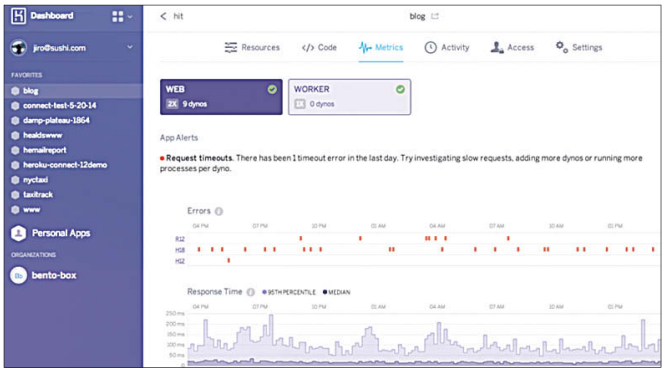
Heroku Postgres DbX vient aussi avec un nouveau tableau de bord qui permet de mieux comprendre le comportement des bases de données relationnelles, et d'en faciliter la gestion. Présentées sous forme graphique et textuelle, elles s'accompagnent d'outils analytiques pour que les développeurs puissent identifier les requêtes ayant le plus d'impact sur leur base de données et optimiser ainsi la performance de leurs applications.

Heroku DX facilite la vie du développeur mais pourquoi parlez-vous autant d'entreprise ?

Les objets connectés se développent aujourd'hui à un rythme effréné. Il devrait y en avoir plus de 75 milliards en 2020. Sans compter les 5 milliards de smartphones attendus pour 2017. Tous ces objets communiquent, échangent et modifient profondément la manière dont les entreprises se positionnent dans leur usage des technologies. Aujourd'hui, toutes les entreprises deviennent des éditeurs de logiciels et se doivent de construire de nouvelles applications le plus rapidement possible qui tirent profit de ces objets et des données qu'ils génèrent : plus question de prendre 6 mois pour penser un concept, 6 mois pour le développer et 6 mois pour le lancer. Il est critique de pouvoir développer des applications de la manière la plus simple, la plus rapide et la plus sécurisée possible, et c'est exactement ce que Heroku DX favorise.

Salesforce propose aussi Force.com au sein de sa plateforme. Quels liens entre Heroku DX et Force.com ?

Force.com est dédié au développement d'applications métiers, internes à l'entreprise, tandis qu'Heroku est plus orienté pour les applications externes pour des clients. Mais les 2 font partie de la même plateforme, à savoir Salesforce1 Platform. À ce titre, tout ce qui – comme Heroku DX – contribue à améliorer l'expérience développeur au service des clients s'inscrit parfaitement dans notre stratégie. Car Heroku DX répond à une demande croissante au sein de nos utilisateurs et partenaires : la mise à disposition d'outils adaptés aux développeurs.



Adoption du Cloud : aider les entreprises à prendre le virage du numérique grâce au Système d'Information Hybride

Le marché du Cloud Computing a déjà séduit de très nombreuses entreprises. Pourtant, plusieurs études indépendantes montrent que l'usage de l'informatique dématérialisée dans les entreprises est loin d'être aussi répandu qu'on pourrait le penser.

Actuellement, un tiers environ des entreprises françaises exploitent une infrastructure, des plateformes ou des logiciels disponibles sous forme de services de *Cloud(1).* Les petites entreprises, elles, sont nombreuses à ne pas être suffisamment informées sur les avantages apportés par l'informatique en nuage.

Si l'adoption du *Cloud Computing* tarde à se matérialiser dans le monde de l'entreprise, ce n'est pas nécessairement pour des raisons techniques. Au contraire, on constate que les freins à l'adoption peuvent être à la fois culturels et organisationnels. Ils peuvent également être liés aux craintes que suscitent les récentes affaires d'espionnage, les intrusions dans les plateformes de stockage en ligne ou encore les failles découvertes dans les systèmes informatiques utilisés sur le nuage, qui ont créé un véritable climat de défiance vis-à-vis du *Cloud Computing*.

Nous pensons qu'il ne faut pas considérer le *Cloud* comme un SI qui viendrait se substituer au système d'information classique. Au contraire, les deux approches, complémentaires l'une de l'autre, peuvent être combinées pour former un SI Hybride qui peut apporter de nombreux bénéfices aux entreprises.

Le système d'information hybride = extension de l'entreprise

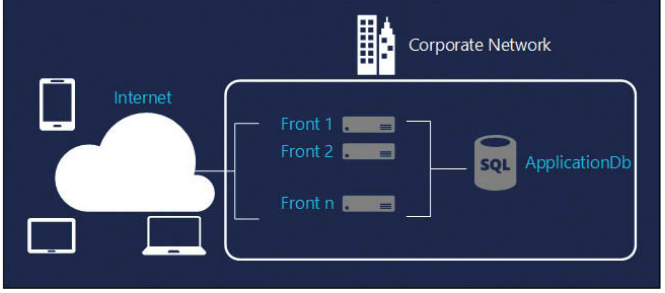
Le SI hybride est le système d'information constitué de l'ensemble du périmètre classique, situé *à demeure*, étendu à l'infrastructure, aux plateformes ou aux logiciels disponibles sous forme de services de *Cloud Computing* exploités par l'entreprise.

Il permet de tirer parti des avantages d'infrastructure apportés par le *Cloud* – Élasticité, capacité de calculs massifs, etc. – et de concilier les impératifs de sécurité et de confidentialité en conservant le stockage de données critiques à l'intérieur de l'entreprise.

Il permet également de mettre en œuvre de nouveaux scénarios innovants via l'utilisation des services en ligne. Au-delà de la messagerie d'entreprise, les exemples d'applications dont les entreprises peuvent tirer parti sont très nombreux : de la gestion de la relation client (CRM) à l'informatique décisionnelle en passant par la génération et l'édition de formulaires en ligne, les exemples sont nombreux et les domaines d'applications variés.

Les entreprises peuvent ainsi commencer à exploiter le *Cloud Computing* de manière progressive et tactique en ouvrant leur périmètre informatique au SI hybride. Il ne leur est plus nécessaire de procéder à la migration en big-bang de leur informatique sur le nuage. Au contraire, une migration par étapes est privilégiée, sur un périmètre restreint des logiciels et des données impliqués.

Fig.1



Architecture classique

Scénario de mise en œuvre : exposition de données sur le nuage

Un premier scénario innovant, que l'on peut envisager avec ce type de solution, concerne la migration d'une application Web stratégique vers le nuage (Figure 1), pour tirer parti de l'élasticité du *Cloud* ainsi que de la réduction du coût des infrastructures à demeure. La plateforme Azure étant extrêmement proche d'un environnement Windows Server, il est souvent possible de faire fonctionner une telle application sans aucun changement dans le code source. Malheureusement, la migration nécessite de l'accompagner d'un transfert des données également vers le nuage. Cette décision pouvant remettre en cause, à elle seule, le projet Fig.1.

La mise en place d'un scénario hybride doit nous permettre de créer une infrastructure Front sur le Cloud tout en gardant la partie Hébergement de données sensibles à l'intérieur du SI.

Microsoft a ajouté à sa liste de services Cloud une offre dite « Hybrid Connection » permettant de mettre en place une liaison sécurisée entre votre système d'information à demeure et des ressources Cloud de type Azure Web Sites ou Azure Mobile Service. Cette connexion est initiée à partir d'un client léger appelé « *Hybrid Connection Manager* » à installer sur son SI à demeure, qui se sert des ports TCP/HTTP disponibles, à choisir parmi une plage restreinte, permettant d'avoir une empreinte minimale et un bon contrôle de la part des équipes d'exploitation.

Les ressources à demeure que l'on souhaite exposer sur le Cloud – parmi lesquelles sont actuellement supportées les bases de données SQL Server, MySQL et les services Web HTTP – doivent être mises à disposition via des ports statiques au client de communication.

Configuration hybride, aucun changement pour le développeur

La mise en place va donc consister à migrer la partie application Front Web sur le service de type *Azure Web Sites* et à configurer la connexion hybride. Les sites Web Azure sont en mesure d'accueillir de nombreuses technologies tel que Python, Java, .Net, NodeJS, etc. L'infrastructure de communication permettant de faire dialoguer le connecteur à demeure et les services Azure est hébergée dans la

(1) Source : 3ème Édition de l'étude PAC CloudIndex, http://www.cloudindex.fr/.

rubrique « BizTalk Services » sur Azure. La première étape consiste donc à créer un nouvel espace de nommage sur « BizTalk Services ». Nous y renseignons le nom de l'hôte de la ressource (par exemple, le nom du Serveur SQL), ainsi que le port de communication qui sera utilisé (1433, dans notre cas).
La connexion Hybride utilise une autorisation de type Shared Access Signature pour sécuriser la connexion entre le SI Interne et la plateforme Cloud. Les étapes de configuration sont ensuite simples :

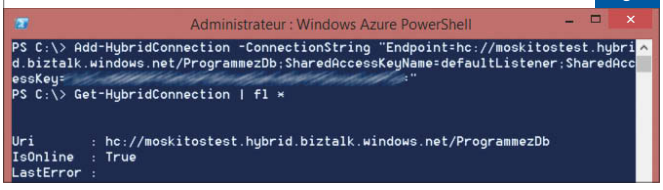
- Téléchargement de l'Hybrid Connection Manager à partir du portail de configuration,
- Récupération de la chaîne de connexion liée à notre ressource, de type hc://namespace/nom;sharedAccess;Token
- Utilisation de Powershell pour ajouter les connexions à notre Manager, les CmdLets sont Add-HybridConnection et Get-HybridConnection pour connaître les états Fig.2.

Une fois la liaison établie, la connexion hybride est visible dans le panneau de configuration des ressources de type *Web Sites* ou *Mobile Services* Fig3.
Désormais, il ne reste plus qu'à configurer la chaîne de connexion à la base de données pour lui indiquer de pointer vers la ressource hybride. En effet, les ressources sont accessibles de la même manière que si elles étaient situées dans le nuage :

```
Server=MoskitosJeremie,1433;Database=ProgrammezOnPremises;
User ID={userName};Password={Password}
```

qui est une chaîne de connexion classique à un serveur SQL. Il n'est donc pas nécessaire de mettre en place une architecture compliquée de type proxy avec des autorisations spéciales ni d'écrire du code supplémentaire Fig.4.

Fig.2



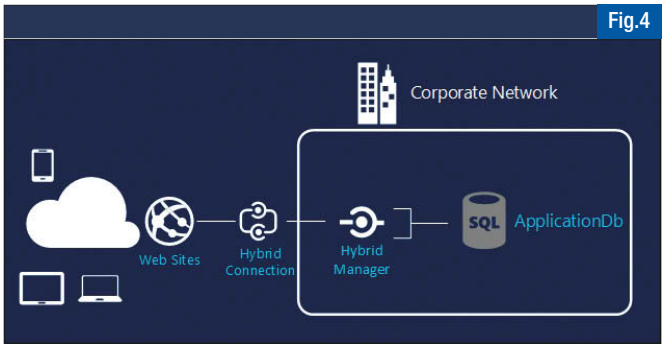
Configuration Powershell

L'utilisation des connexions hybrides permet, dans ce cas, d'exposer les données d'origine – toujours situées à l'intérieur du SI – à l'application Web. Le seul changement portant sur la chaîne de connexion qui relève, la plupart du temps, de la configuration de l'application.
On le comprend, l'utilisation d'une telle solution permet d'exposer facilement des données à l'extérieur de l'entreprise, sans refonte importante de l'infrastructure réseau, et sans nécessité de changer le code des applications.
Cependant, nous pouvons voir que ce genre de scénario est, pour l'instant, limité à des ressources de type *Base de Données* ou *Web Service*. Pour élargir ce scénario à un ensemble plus vaste de ressources, par exemple pour des applications métiers de type *Line Of Business* ou *Legacy*, il peut être nécessaire de se tourner vers des plateformes d'Intégration en mode SaaS, que nous appelons IPaaS (pour Integration Plateform as a Service).

Jérémie **DEVILLARD**
Directeur Conseil @Moskitos
http://jeremiedevillard.wordpress.com
Twitter : @jeremiedev

Maxime **LABELLE**
Directeur R&D @Moskitos
http://maximelabelle.wordpress.com
Twitter : @SpringComp

Fig.4



Infrastructure Hybride

Fig.3

Mise en place d'une connexion hybride

Restez connecté(e) à l'actualité !

- L'**actu** de Programmez.com : le fil d'info **quotidien**
- La **newsletter hebdo** : la synthèse des informations indispensables.
- **Agenda** : Tous les salons, barcamp et conférences.

Abonnez-vous, c'est gratuit !

www.programmez.com

Les Frameworks Front-End Web

1ere partie

Utiliser un Framework Front End lors d'un développement web permet de gagner un temps considérable sur la mise en place d'une interface utilisateur. Une foule de composants réutilisables sont la plupart du temps déjà disponibles.

Pour ne citer que ceux-ci : Grille responsive, Formulaire, Bouton, Police d'icône, Popup, Accordéon, Onglet, Tooltip, Notification.
Ces Frameworks s'adressent à la fois aux développeurs et intégrateurs mais aussi aux designers dans la réalisation de maquettes.

Le premier à aborder s'intitule Bootstrap, ici dans sa version 3. Le projet a débuté mi-2010 dans les locaux du réseau social Twitter sous le nom de « Twitter Blueprint » grâce à la collaboration de Mark Otto et Jacob Thornton. Probablement l'un des plus connus et utilisés dans sa catégorie, il est présent sur une majeure partie des sites web de présentation de composants de développement open source mis en ligne ces dernières années. On pourra citer par exemple le site officiel de AngularJS (http://angularjs.org) ou encore Intercooler (http://intercoolerjs.org/). C'est aussi, depuis Visual Studio 2013, la bibliothèque graphique utilisée par défaut pour les projets web ASP.NET. Celle-ci vient remplacer l'illustre mais vieillissant jQuery UI. Nous présenterons ensuite Foundation (version 5). Ce produit trouve ses origines dans la charte de design interne à Zurb dans les années 2008. Moins répandu que le premier, celui-ci se distingue par sa ligne éditoriale. Bootstrap reste relativement simple à personnaliser en surface, via un fichier « variable.less » ou encore l'interface de personnalisation, mais montre des complexités lorsqu'on souhaite aller plus loin. Foundation se veut moins fourni à ce niveau afin de pousser l'intégration d'une charte dès les premières phases de conception.

Enfin le troisième, Ink version 3, n'a pas communiqué précisément sur son historique mais semble avoir fait surface fin 2012 dans le giron du moteur de recherche portugais Sapo. Bien moins répandu que ses deux concurrents, il se distingue par une particularité peu commune : il n'est pas basé sur la bibliothèque jQuery. Il est fourni avec beaucoup plus de composants que ses concurrents. Au contraire des deux autres, qui sont conçus de manière plus agnostique, il est pensé pour s'interfacer plus facilement avec Node.js.

Afin de concentrer le comparatif sur un exemple concret et synthétique, nous allons étudier l'utilisation de ces trois bibliothèques afin de concevoir une horloge numérique. Celle-ci sera composée d'un formulaire de recherche pour la ville, d'une grille représentant les températures, indicateur météo, date et heure, et d'un bouton ouvrant une pop-up « A propos ».
Nous détaillerons les cas d'utilisation selon différentes étapes :

- Le Template de code minimaliste,
- Le bloc de recherche,
- La grille de l'horloge,
- Le bouton déclenchant l'ouverture de la pop-up

Bootstrap dit « le Champion »

Pour commencer, voici le Template HTML minimaliste qu'il faut mettre en œuvre pour utiliser Bootstrap :

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>Horloge m&eacute;t&eacute;o</title>
  <link href="css/bootstrap.min.css" rel="stylesheet">
</head>
<body>
  <div class="container">
    <!-- SUITE DU CONTENU A INSERER ICI -->
  </div>
  <script type="text/javascript" src="js/jquery.min.js"></script>
  <script type="text/javascript" src="js/bootstrap.min.js"></script>
</body>
</html>
```

On remarquera la présence de la balise « div » portant la classe « container », dans laquelle nous allons insérer notre horloge afin de la faire interagir avec le système de grille responsive. Une feuille de style CSS et deux fichiers JavaScript sont utilisés pour seules dépendances. Pour ajouter le bloc de recherche, on ajoute une balise « div » ayant la classe « row ». Dans celle-ci on trouve une nouvelle « div » ayant pour classes « col-lg-12 », représentant une colonne de 12 unités, ainsi que « form-inline » pour permettre d'affecter un style spécifique au formulaire de recherche :

```
<div class="row first">
  <div class="form-inline col-lg-12" role="form">
    <div class="input-group">
      <input type="text" class="form-control" value="Paris">
      <span class="input-group-btn">
        <button class="btn btn-default" type="button"><i class=
"glyphicon glyphicon-search"></i></button>
      </span>
    </div>
  </div>
</div>
<br/>
```

Voici le résultat visuel de ce bloc de code, un formulaire en ligne avec un bouton adjoint au champ de saisie :

Pour ajouter l'horloge, nous allons maintenant créer deux nouvelles lignes de deux colonnes chacune via la classe « col-xs-6 », imbriquées dans une « div » de classe « well » afin de faire ressortir le bloc :

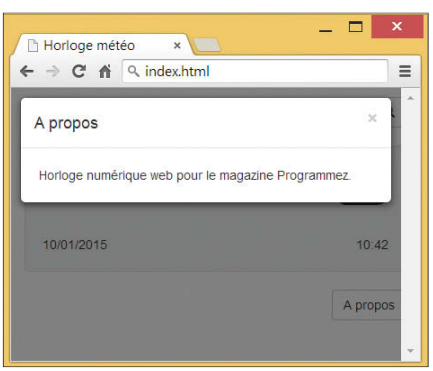
```
<div class="row">
  <div class="col-lg-12">
    <div class="well">
      <div class="row">
        <div class="col-xs-6 degree">
          12&deg;
        </div>
        <div class="col-xs-6 text-right weather">
          <i class="glyphicon glyphicon-cloud"></i>
        </div>
      </div>
```

```
      <div class="row">
        <div class="col-xs-6">
          10/01/2015
        </div>
        <div class="col-xs-6 text-right">
          10:42
        </div>
      </div>
    </div>
  </div>
</div>
```

Concernant la police d'icône présente au sein du Framework, elle en comporte 200. Ce n'est pas la plus exhaustive mais c'est amplement suffisant pour la plupart des projets. Enfin, pour ajouter un bouton qui ouvre une pop-up, nous allons ajouter une nouvelle ligne contenant une unique colonne alignée à droite via la classe « text-right » :

```
<div class="row">
  <div class="col-lg-12 text-right">
    <a class="btn btn-default" data-toggle="modal" data-target=
"#apropos">A propos</a>
  </div>
</div>
<div class="modal fade" id="apropos" tabindex="-1" role="dialog"
 aria-labelledby="aproposLabel" aria-hidden="true">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <button type="button" class="close" data-dismiss="modal"
 aria-hidden="true">&times;</button>
        <h4 class="modal-title" id="aproposLabel">A propos</h4>
      </div>
      <div class="modal-body">
        Horloge num&eacute;rique web pour le magazine Programmez.
      </div>
    </div>
  </div>
</div>
```

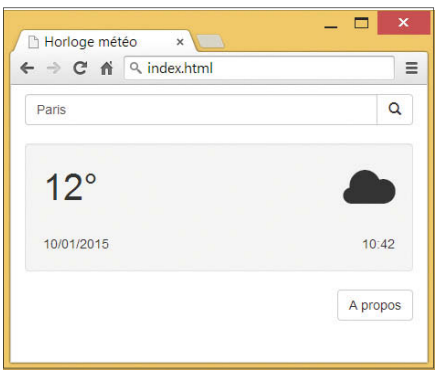
La pop-up la plus minimaliste est composée d'un en-tête, permettant d'afficher un titre et un bouton de fermeture, et d'un corps, comportant le message associé. On regrettera le fait que la pop-up ait une largeur limitée :



Pour finir, quelques styles pour personnaliser le rendu, à ajouter dans la balise « head ». Une classe « first » permettant d'initialiser la première ligne afin qu'elle ne soit pas collée au haut de la page, et deux autres « degree » et « weather » permettant de dimensionner la police des indicateurs de température et météo :

```
<style type="text/css">
  .first {
    margin-top: 10px;
  }
  .degree {
    font-size: 40px;
  }
  .weather {
    font-size: 56px;
  }
</style>
```

Et voilà le résultat :



Foundation surnommé « le Challenger »

Pour Foundation, le Template minimaliste est celui-ci :

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-
scale=1.0" />
  <title>Horloge m&eacute;t&eacute;o</title>
  <link rel="stylesheet" href="css/foundation.css" />
  <link rel="stylesheet" href="foundation-icons/foundation-icons
.css" />
  <script type="text/javascript" src="js/vendor/modernizr.js"></script>
</head>
<body>
  <!-- SUITE DU CONTENU A INSERER ICI -->
  <script type="text/javascript" src="js/vendor/jquery.js"></script>
  <script type="text/javascript" src="js/foundation.min.js"></script>
  <script type="text/javascript">
    $(function () {
      $(document).foundation();
    });
  </script>
</body>
</html>
```

On notera la présence de deux feuilles de style ainsi que trois fichiers JavaScript. Un point important à prendre en considération est l'absence de police d'icône incluse dans le Framework. Selon le point de vue, cela peut

être un point positif ou négatif. Positif si vous souhaitez déjà utiliser une police spécifique et que vous souhaitez également éviter les dépendances inutiles. Négatif si vous n'avez pas le temps de chercher la police adaptée à vos besoin et que vous auriez souhaité un package complet, à l'image de Bootstrap. Quoi qu'il en soit, une police d'icône a été mise à disposition par le même éditeur et c'est celle-ci que nous allons utiliser pour l'exemple. Continuons notre exemple avec l'ajout du bloc de recherche :

```
<div class="row first">
  <div class="large-12 columns">
    <div class="row collapse">
      <div class="small-11 columns">
        <input type="text" id="right-label" value="Paris">
      </div>
      <div class="small-1 columns">
        <a href="#" class="button postfix"><i class="fi-magnifying
-glass"></i></a>
      </div>
    </div>
  </div>
</div>
```

Comme pour Bootstrap, une « div » de classe « row » comportant un système de colonne via les classes « columns » et « large-12 ». À l'intérieur de celle-ci, deux colonnes permettent d'afficher un champ de formulaire et son bouton de soumission.

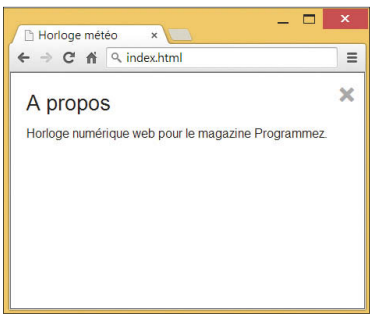
Pour ajouter l'horloge, nous allons comme pour Bootstrap créer une grille de deux lignes par deux colonnes, le tout encapsulé dans une « div » de classe « panel » afin de faire ressortir cet encart :

```
<div class="row">
  <div class="large-12 columns">
    <div class="panel">
      <div class="row">
        <div class="small-6 columns degree">
          12&deg;
        </div>
        <div class="small-6 columns text-right weather">
          <i class="fi-cloud"></i>
        </div>
      </div>
      <div class="row">
        <div class="small-12 columns">
           
        </div>
      </div>
      <div class="row">
        <div class="small-6 columns">
          10/01/2015
        </div>
        <div class="small-6 columns text-right">
          10:42
        </div>
      </div>
    </div>
  </div>
</div>
```

Enfin, la pop-up se décline de manière bien plus sobre que son concurrent précédent. Elle est composée d'un titre « h2 », d'un paragraphe et d'un bouton de fermeture :

```
<div class="row">
  <div class="large-12 columns text-right">
    <a href="#" class="button small" data-reveal-id="apropos">
A propos</a>
  </div>
  <div id="apropos" class="reveal-modal" data-reveal>
    <h2>A propos</h2>
    <p>Horloge num&eacute;rique web pour le magazine Programmez.</p>
    <a class="close-reveal-modal">&#215;</a>
  </div>
</div>
```

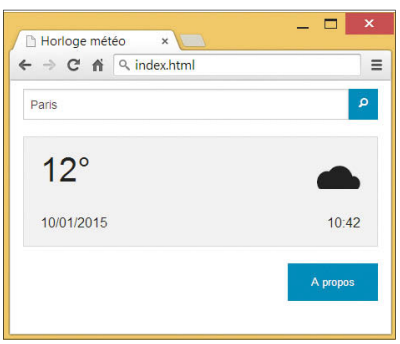
Remarque intéressante : la pop-up passe en mode plein écran dans le cas d'un affichage au format mobile. C'est un parti pris original et tout à fait fondé. Le format mobile ne permet pas de gâchis d'espace par une double marge, horizontale et verticale, et le rendu « responsive » en tient compte. Foundation tire son épingle du jeu avec un choix astucieux.



Encore une fois quelques styles personnalisés permettant d'aérer le contenu et de régler la taille des polices :

```
<style type="text/css">
  .first {
    padding-top: 10px;
  }
  .degree {
    font-size: 40px;
  }
  .weather {
    font-size: 56px;
  }
</style>
```

Et voici le résultat final de l'horloge :



Hugo Carnicelli - *Consultant chez SoftFluent*

Connectez vos objets Arduino, Spark.IO, .NET Microframework, Galileo à Microsoft Azure 1ère partie

L'Internet des objets (Internet of Things, ou IoT) est en plein boom. Entre les coûts des processeurs et les coûts des composants électroniques vraiment très faibles, plus une faible consommation électrique, il est devenu possible de créer de nombreux objets plus ou moins intelligents. Ajoutez à cela des connexions devenues possibles grâce à du Bluetooth Low Energy (BLE), des composants Wifi ou GPRS à prix réduits, ces nouveaux objets peuvent se connecter à Internet ou entre eux.

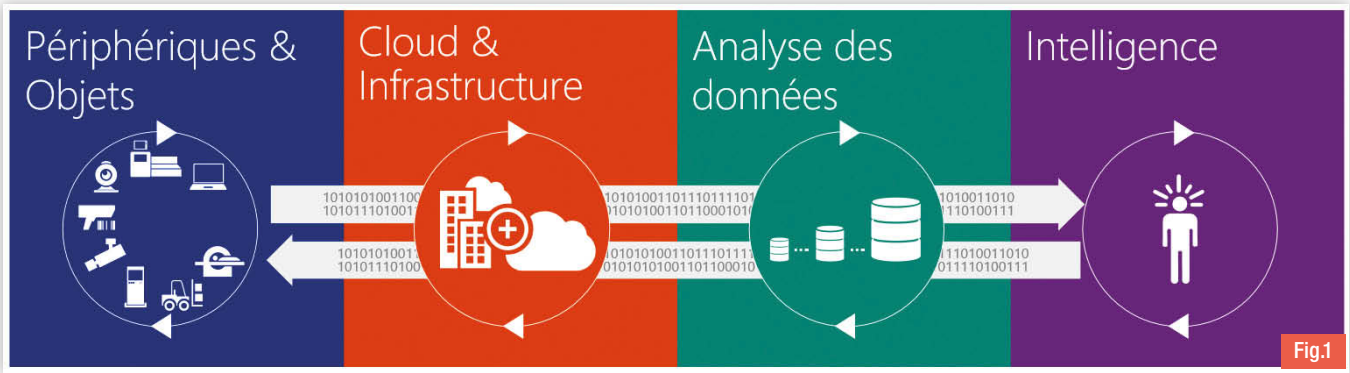


Fig.1

Etre connecté c'est bien, mais ça ne sert fondamentalement à rien si les données générées ne sont pas utilisées ou si l'objet n'effectue aucune action. Avec les infrastructures Cloud, il existe le moyen de stocker des données depuis ces objets, mais aussi d'ordonner à ces objets d'effectuer des tâches en fonction de l'analyse de ces données.

L'IoT est en réalité assez complexe et il faut répondre à de nombreuses questions : créer des objets connectés ou connecter des objets existants ? Comment gérer ces objets connectés à distance ? Comment sécuriser l'accès à ces objets et sécuriser les communications de ces objets ? Comment gérer les données générées ? Comment les analyser ? Comment renvoyer les informations aux objets pour qu'ils changent leur comportement et s'adaptent ? Quels outils de développement ? Fig.1.

Il est essentiel de répondre à ces questions pour tout projet IoT. Les réponses peuvent être différentes en fonction des projets, y compris pour les choix des technologies à utiliser. Au-delà de l'effet de mode il y a un vrai intérêt dans l'IoT. Le schéma ci-dessus représente le cycle complet d'un projet IoT. Il commence par les objets qui remontent des données. Il y a ensuite l'infrastructure de stockage puis l'analyse et de l'intelligence autour des analyses. La remontée et le traitement d'information doit apporter un bénéfice et une modification des objets ou de leur environnement. Ainsi il faut ensuite redescendre la chaîne jusqu'aux objets pour qu'ils puissent intervenir sur leur environnement.

C'est cet ensemble complet, qui part des objets et qui revient aux objets, qu'il est nécessaire de maîtriser dans un projet IoT.

Périphériques et objets

Mon besoin est relativement simple : automatiser au maximum mon arrosage automatique en fonction des éléments comme la température, l'humidité du sol, l'humidité de l'air, la pluviométrie.

Mon existant : j'ai des capteurs de température et d'humidité Oregon Scientific. J'ai développé il y a quelques années un système d'arrosage pilotable à travers une interface web où je peux programmer des temps d'arrosage comme je le souhaite. Ce premier projet utilise la technologie .NET Microframework et un Netduino Plus qui est un board embarqué, utilisant un processeur ATMEL AT91SAM7X512. .NET. Microframework est une implémentation Open Source de .NET qui permet de bénéficier de cette technologie directement sur ces chips, sans aucun OS. Plus de détails sur ce projet et les sources sur : (http://blog.msdn.com/laurelle).

Ce que j'ai décidé de faire : je me suis dit qu'à la fois, il serait bien que je puisse rendre communiquant mes capteurs existants, et développer de nouveaux capteurs complémentaires. Pour cela, j'ai décidé d'utiliser des Arduino et des Spark.IO. Les deux sont à base de processeur ATMEL ATmega328 et implémentent le framework Wiring.org. Ces technologies sont très utilisées par les passionnés pour créer tous types de projets. Ce framework open source permet de facilement réaliser des projets IoT. Il est possible de le faire tourner directement sur le chip, c'est le cas sur les ATMEL ATmega328 ou au dessus d'un OS.

C'est le cas par exemple des boards de type Arduino YÚN ou Galileo (http://www.arduino.cc). Les boards à base de Galileo peuvent faire tourner Linux ou Windows en tant qu'OS sur lequel la couche Wiring est ajoutée. Les développements se font en C/C++. Microsoft a récemment annoncé le support de Windows sur ce type de processeur, le tout en Open Source (plus d'info sur http://dev.windows.com/en-us/featured/Windows-Developer-Program-for-IoT et les sources disponibles sur https://github.com/ms-iot).

L'ajout de classes spécifiques à l'OS permettent de bénéficier de ce que ces OS proposent comme un accès à de l'encryption, de la gestion de sécurité, de l'authentification.

Connecter des objets existants

J'ai créé un premier objet à base de Spark.IO qui me permet de décoder le protocole de mes sondes existantes. Après quelques recherches sur Internet, j'ai trouvé les excellents articles d'Olivier Lebrun qui m'ont permis de me mettre le pied à l'étrier (cf http://connectingstuff.net/blog/decodage-protocole-oregon-arduino-1/).

Les sondes communiquent avec une base à travers des émissions de message sur la fréquence 433MHz. Cette fréquence est assez fréquemment utilisée pour des communications entre des éléments tels que des capteurs de températures, l'envoie de commandes de chauffage. Dans le

cas des sondes Oregon, elles communiquent dans un protocole assez simple et non crypté. Il est donc relativement facile de récupérer ces informations. Et aussi relativement facile de pouvoir créer des capteurs compatibles ou utilisant le même protocole Fig.2.

L'envoi d'un message se fait en modulant un message dans un émetteur de fréquence 433MHz qui sera démodulé par un récepteur. La plupart des protocoles simples fonctionnant dans l'air utilisent un codage dit Manchester. L'envoi d'un 0 se fait par l'envoi d'un état bas, puis, l'information est ensuite envoyée en inversé, c'est-à-dire haut puis bas. L'envoi d'un 1 se fait en inverse, c'est-à-dire d'abord par un état haut puis bas et de nouveau bas puis haut. Le décryptage se fait donc en analysant les changements d'états successifs en fonction de la période de ces changements. Je ne vais pas rentrer dans le détail du décryptage ici.

En général dans la plupart des projets mettant en jeux de nombreux objets communicants, seuls quelques-uns sont reliés à Internet, les autres communiquent entre eux de proche en proche, voire même ne communiquent qu'en émission. Ces architectures ne sont pas nouvelles et sont utilisées depuis très longtemps dans le domaine des alarmes par exemple. Il y a une centrale qui est reliée à Internet soit à travers le réseau GSM, soit en filaire soit en IP à travers une box. La centrale est associée à des capteurs très simples d'ouverture de porte qui envoient un message quand la porte est ouverte et un message de synchro pour dire qu'ils sont toujours vivant, donner l'état de leur batterie. Leur communication n'est que dans un sens. Le capteur n'a pas besoin d'être sophistiqué.

Certains éléments d'alarme peuvent être plus intelligents comme des caméras de surveillance avec la capacité de pouvoir parler avec les personnes se trouvant éventuellement dans la pièce. Ils vont donc communiquer dans les deux sens.

La centrale doit pouvoir prévenir si un élément ne fonctionne pas correctement, si un élément lève une alerte ou renvoie vers le capteur bidirectionnel les informations venant de l'extérieur.

C'est donc une architecture équivalente que je vais utiliser dans le cadre de mon projet Fig.3.

Les capteurs existants comme les nouveaux capteurs vont donc utiliser le même protocole en 433MHz. Attention tout de même car l'utilisation des fréquences et l'usage est très réglementé à travers le monde. Les messages envoyés doivent être courts et non continus. J'ai donc créé un capteur permettant de mesurer la vitesse du vent, la direction et la pluviométrie ainsi qu'un autre mesurant la température, humidité de l'air, humidité du sol, luminosité.

Ces deux capteurs sont à base d'ATmega 328. L'intérêt est leur faible consommation énergétique et la possibilité de les endormir pour limiter encore la consommation. Mes capteurs doivent pouvoir être autonomes et se recharger uniquement en solaire à l'aide d'une batterie pour le fonctionnement nocturne.

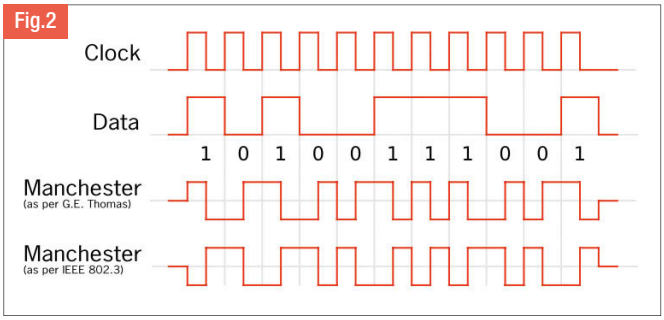
Côté outil de développement, je recommande, pour tout développement avec Arduino, d'utiliser Visual Studio avec l'addin gratuit Visual Micro (http://www.visualmicro.com/). Il est possible d'utiliser toute version de Visual Studio, y compris les versions gratuites Express (il faut utiliser dans ce cas la version Desktop pour le support du C++).

Il offre par exemple la possibilité de faire du débogage, mais aussi facilite l'écriture du code avec la correction syntaxique automatique, l'ajout de fichiers headers à base de template .

De façon très synthétique et sans rentrer dans le détail de l'implémentation du protocole de communication, voici le code minimum nécessaire côté Arduino pour mesurer la vitesse du vent et de façon simplifiée, le code pour l'envoi d'un message.

La communauté Arduino étant importante, il est assez facile de trouver des éléments de code autour de capteurs spécifiques. Ces codes fonctionnent d'ailleurs moyennant une simple adaptation sur tous les chips

Fig.2



Source Wikipedia http://en.wikipedia.org/wiki/Manchester_encoding

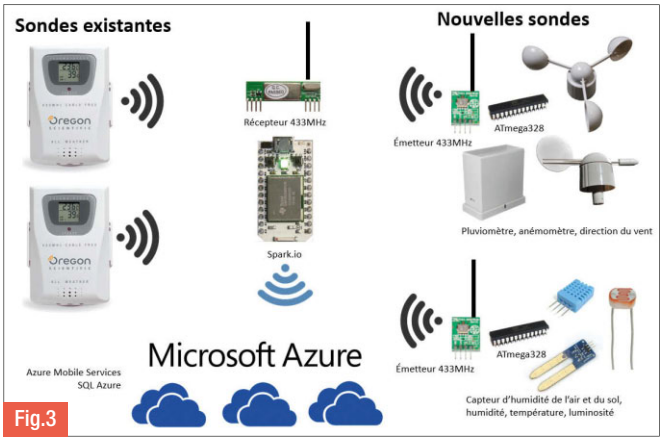


Fig.3

implémentant Wiring.org. Code complet sur www.programmez.com.

Côté Spark.IO pour le décodage, je ne rentrerai pas non plus dans le détail. Le décodage ressemble également à cela :

- Avant toute chose, il est nécessaire de désactiver le cloud Spark.IO à l'aide de Spark.disconnect(); dans la fonction void setup(). Son utilisation étant non contrôlée et étant gourmand en ressources, le décodage ne fonctionne pas quand il est activé,
- Une durée est calculée entre 2 changements d'état du récepteur,
- Cette durée est soit courte si comprise entre 200 et 700 millisecondes, soit longue si entre 700 et 1200. Les autres sont non valides,
- Tant que les impulsions sont valides :
  - Le décodage Manchester est opéré à chaque nouvelle impulsion,
  - Le signal est décodé au fur et à mesure,
  - Dès que la synchro est trouvée (32 impulsions 1 soit 0xFF 0xFF envoyé), vérifier le type de sonde, continuer à décrypter les impulsions.
- Si à un moment les données ne sont pas valides, réinitialiser le décodeur,
- Une fois le décodage terminé, il est possible de récupérer le message et de le décoder.

Le Spark.IO a une architecture intéressante car il possède une connexion Wifi. Il est « over the air » (OTA, donc à distance). L'outil de développement est dans ce cas un outil en ligne, relativement simple mais assez efficace. Il est également possible d'utiliser Visual Studio comme pour l'Arduino pour le programmer mais on perd la possibilité de le programmer OTA.

Le Spark.IO offre également un accès à des données que l'on peut publier sur le cloud de Spark.IO. Cela a l'air attrayant comme cela mais se révèle très rapidement vraiment limité.

D'une part, il n'est pas possible de savoir si les informations sont publiées et quand, d'autre part, il n'est pas possible de stocker un historique des données. Et enfin, l'objet doit être connecté même pour accéder à la dernière donnée publiée.

Laurent **Ellerbach** - *Audience Marketing Microsoft EMEA - DX*

A la découverte du iBeacon

Depuis quelques années, Apple embarque dans ses iPhones et iPads un nouveau module Bluetooth, de type Low Energy (LE). Ce module, développé pour améliorer la fiabilité et réduire la consommation énergétique, prend aussi en charge un nouveau protocole, le Bluetooth 4.0.

Ces évolutions ont permis à la firme de Cupertino d'introduire un système de géolocalisation indoor baptisé sous le nom marketing iBeacon, depuis iOS 7. Effectivement, nous parlons ici de "nom marketing" car les composants hardware sur lesquels iBeacon repose sont totalement standards et reproductibles par n'importe quel constructeur.
Un cas d'usage, parmi les plus fréquents, est d'utiliser les émetteurs pour localiser un utilisateur à l'intérieur d'une boutique et lui proposer des notifications locales l'informant des produits qui se trouvent à proximité. Depuis quelques mois, Apple – parmi d'autres – implémente donc cette fonctionnalité afin de proposer de l'aide et des promotions personnalisées selon les différents rayons visités.

Mais plus précisément, qu'est-ce qu'un iBeacon ?

Un iBeacon est un petit appareil Bluetooth LE qui émet un signal d'une portée moyenne de 50 mètres. Ce dernier est visible par un terminal implémentant le protocole Bluetooth 4.0 comme les iPhones (à partir du 4S), les iPads (à partir de la 3ème génération) ou les plus récents appareils Android. Cette borne émettrice (beacon en anglais) est identifiée par la combinaison de trois valeurs : un UUID (souvent l'identifiant univoque du producteur), un numéro majeur et un numéro mineur. C'est grâce à des APIs haut niveau mises à disposition depuis iOS 7, qu'une application iOS peut facilement détecter les iBeacons émettant à proximité et déclenchant des actions développées préalablement.

Fonctionnement en arrière-plan

Concernant iOS 7, les ingénieurs de Cupertino ont décidé d'intégrer largement les iBeacons à l'intérieur du système d'exploitation. Par exemple, les principales interactions entre les émetteurs et l'application peuvent se produire même lorsque celle-ci est inactive voire fermée manuellement par l'utilisateur. Dans ce cas, à proximité d'une borne iBeacon, le système ré-instanciera automatiquement l'application afin de lui permettre d'exécuter des actions implémentées préalablement par le développeur. Apple est toujours en train d'optimiser l'intégration avec le Bluetooth 4.0 pour réduire la consommation énergétique et accélérer la détection des émetteurs.

Les APIs

Ainsi, un nouveau set d'APIs a été introduit à partir d'iOS 7 pour implémenter les fonctionnalités de géolocalisation indoor. Ces APIs se basent principalement sur les classes CLLocationManager et CLRegion.

CLLocationManager

La classe CLLocationManager s'occupe de la gestion des événements liés à la géolocalisation et à la direction. Il est désormais possible de récupérer les informations de positionnement de l'utilisateur à l'aide des technologies telles que le GPS, la boussole ou, bien sûr, les iBeacons. La position géographique étant une information sensible pour l'utilisateur, il a la possibilité de refuser l'accès aux données du service de localisation d'une application. C'est à son démarrage que le framework Core Location invitera l'utilisateur à autoriser le service de

localisation. Si l'utilisateur refuse, l'objet CLLocationManager signale une erreur appropriée à son delegate. Il est également possible de vérifier l'état de l'autorisation explicite de l'application en utilisant la méthode authorizationStatus.
Afin de configurer et d'utiliser un objet CLLocationManager, il est nécessaire d'affecter un objet personnalisé à la propriété delegate. Cet objet doit être conforme au protocole de CLLocationManagerDelegate.

CLRegion

Une CLRegion est une classe abstraite. Elle identifie une région de l'espace géographique dans laquelle on peut programmer des actions lorsque certains événements ont lieu. Par exemple, nous pouvons déclencher des actions lorsque nous rentrons ou sortons de l'espace identifié par la CLRegion.

CLBeaconRegion

CLBeaconRegion est une sous-classe de CLRegion représentant une borne iBeacon. Comme pour CLRegion, nous pouvons programmer des actions lorsque certains événements se manifestent grâce au location manager.
Il est possible d'identifier une région en utilisant les combinaisons suivantes :

- ProximityUUID (initWithProximityUUID:identifier:) : tous les beacons ayant le même UUID feront partie de la même region,
- ProximityUUID et majeur (initWithProximityUUID:major:identifier:) : tous les beacons ayant le même UUID et l'attribut "major" constitueront la même region,
- ProximityUUID, majeur et mineur (initWithProximityUUID:major:minor:identifier:) : le beacon qui fournit à la fois ces trois valeurs identifiera cette region.

Le paramètre "Identifier", inclus dans les trois constructeurs ci-dessus, représente une chaîne de caractères qui constitue le nom de la CLBeaconRegion. Nous pouvons ainsi repérer facilement la région à l'intérieur de notre code.
Après avoir instancié une région, il est temps de la monitorer, c'est à dire, d'observer les événements de transition. Cette opération est activée grâce à la méthode startMonitoringForRegion: CLLocationManagerDelegate, nous permettant notamment d'être notifié lorsque nous entrons ou sortons d'une région.
Comme expliqué ci-dessus, le monitoring est opéré automatiquement par le système d'exploitation, même quand l'application est en tâche de fond ou totalement terminée. Par conséquent, afin d'éviter que le

système ait à prendre en charge un nombre excessif de monitoring, Apple impose une limitation de 20 régions gérées par une application. Il est donc possible de se désinscrire d'une région et d'annuler le monitoring avec la méthode stopMonitoringForRegion: .
Les changements d'état, les transitions ainsi que les erreurs seront envoyés au delegate de notre CLLocationManager. Celui-ci implémentera le protocole CLLocationManagerDelegate et plus particulièrement des méthodes telles que :

- locationManager:didEnterRegion:
- locationManager:didExitRegion:
- locationManager:didDetermineState:forRegion:
- locationManager:monitoringDidFailForRegion:withError:
- locationManager:didStartMonitoringForRegion:

CLBeacon

La classe CLBeacon représente un beacon qui a été rencontré lors du monitoring d'une CLRegion. Ce n'est pas à l'utilisateur de créer directement les instances de cette classe. L'identité d'un beacon est définie par son proximityUUID et par les propriétés majeur et mineur qui font parties des valeurs de configuration d'un beacon.

Observation des beacons dans une région (ranging)

En rentrant dans une région, il est possible de monitorer certaines valeurs propres aux différents beacons qui la composent.
Les méthodes startRangingBeaconsInRegion: et stopRangingBeaconsInRegion: nous permettent d'opérer ce monitoring de géolocalisation.
De plus, il est possible de connaître notre distance par rapport au beacon considéré. Cette information peut prendre une des valeurs suivantes : CLProximityUnknown, CLProximityImmediate, CLProximityNear, CLProximityFar.
Pour calculer avec plus de précision notre distance – en mètres – nous pouvons utiliser la propriété accuracy de l'objet CLBeacon.
Malheureusement, de nombreux facteurs (comme les interférences radio) peuvent réduire la puissance reçue et se répercuter négativement sur la précision de la valeur obtenue.

LET'S CODE !

Maintenant que nous connaissons la théorie, il est temps de construire une application mobile nous permettant de récupérer les informations des beacons à proximité.

Les bases

Débutons par un nouveau projet de type single view application.
Pour cela, créons une nouvelle classe de type UITableViewController que nous allons nommer XBBeaconViewController. Cette classe contiendra la liste des beacons visibles à proximité. Dans le storyboard, remplaçons le UIViewController par un UITableViewController. La classe de ce dernier doit prendre la valeur de XBBeaconViewController. À l'intérieur de la tableView, ajoutons deux cellules :

- Une cellule de type Basic, contenant le texte "Aucun beacon visible", ayant "NoVisibleBeaconCell" en tant que "Identifier".
- Une cellule vide, de type Subtitle, ayant "BeaconCell" en tant que "Identifier".

L'interface

Dans le fichier XBBeaconViewController.m, nous y ajoutons une extension et déclarons un objet "locationManagerObserver" de type "id". Grâce aux NSNotification, nous serons informés de l'état de nos beacons.

```
@interface XBBeaconViewController()
@property (nonatomic, strong) id locationManagerObserver;
@end
```

Ensuite, nous fournissons l'implémentation des méthodes principales du tableViewController.

```
@implementation XBBeaconViewController

- (void)viewDidLoad {
    [super viewDidLoad];
    [self.tableView reloadData];
    [self initObservers];
}

- (void)dealloc {
    [[NSNotificationCenter defaultCenter] removeObserver:self.observer];
}

- (void)initObservers {
    // TODO
}

#pragma mark - Table View

- (UITableViewCell *)tableView:(UITableView *)tableView cell
ForRowAtIndexPath:(NSIndexPath *)indexPath {
   static NSString *NoVisibleBeaconCellIdentifier = @"NoVisible
BeaconCell";
    static NSString *BeaconCellIdentifier = @"BeaconCell";
    // TODO
}

- (NSInteger)tableView:(UITableView *)tableView numberOfRows
InSection:(NSInteger)section {
    // TODO
}

@end
```

Nous sommes maintenant en mesure d'afficher à l'écran une liste (vide, pour l'instant) de beacons.

Le Gestionnaire iBeacon

Continuons en créant une nouvelle classe de type NSObject, nommée XBBeaconLocationManager.

Entête

Le contenu de l'entête XBBeaconLocationManager.h est le suivant :

```
#import <Foundation/Foundation.h>
#import <CoreLocation/CoreLocation.h>
extern NSString * const XBBeaconLocationManagerValueChanged
Notification;
@interface XBBeaconLocationManager : NSObject <CLLocationManager
Delegate>
+ (XBBeaconLocationManager *)sharedManager;
@property (nonatomic, readonly) NSArray *allBeacons;
@end
```

XBBeaconLocationManagerValueChangedNotification est le nom de la notification que nous allons envoyer lorsque des nouveaux beacons seront disponibles. Comme mentionné précédemment, pour dialoguer avec le CLLocationManager, notre nouvelle classe est conforme au protocole CLLocationManagerDelegate.
La méthode de classe sharedManager nous permettra d'accéder à une instance singleton de type XBBeaconLocationManager.
Pour terminer avec l'en-tête, la propriété allBeacons nous servira à récupérer la liste des beacons visibles.

Implémentation du gestionnaire iBeacon

Nous allons maintenant développer la partie la plus importante de l'application : celle qui s'occupera de récupérer et de renvoyer la liste des bornes iBeacons visibles. Ouvrons le fichier XBBeaconLocationManager.m et ajoutons les lignes suivantes :

```
NSString * const XBBeaconLocationManagerValueChangedNotification =
@"XBBeaconLocationManagerValueChangedNotification";
@interface XBBeaconLocationManager () <UIAlertViewDelegate>
@property (nonatomic, strong) CLLocationManager *location
Manager;
@property (nonatomic, strong) NSArray *allRegions;
@property (nonatomic, strong) NSMutableArray *currentBeacons;
@end
```

Le code que nous avons écrit permet de nommer la constante XBBeaconLocationManagerValueChangedNotification et de déclarer les propriétés qui contiendront le CLLocationManager et la liste des beacons actuels. Le tableau allRegions fournit alors la liste des régions que nous allons monitorer. Ensuite, ajoutons le code suivant :

```
@implementation XBBeaconLocationManager
```

Puis, créons le constructeur de l'objet singleton :

```
+ (XBBeaconLocationManager *)sharedManager {
    static XBBeaconLocationManager *sharedManager = nil;
    static dispatch_once_t onceToken;
    dispatch_once(&onceToken, ^{
        sharedManager = [[self alloc] init];
    });
    return sharedManager;
}
```

Ensuite, intégrons le constructeur de la classe :

```
- (instancetype)init {
if (self = [super init]) {
  [self initAccordingToAuthorizationStatus];
}
return self;
}
```

Accédons aux données de géolocalisation

La méthode suivante demande à l'utilisateur la permission d'accéder aux données de géolocalisation, grâce à la fonction de authorisationStatus décrite précédemment.

```
- (void)initAccordingToAuthorizationStatus {
    if ([CLLocationManager authorizationStatus] != kCLAuthorization
StatusAuthorized) {
        UIAlertView *alertView = [[UIAlertView alloc] init
WithTitle:@"Activer iBeacon ?" message:@"Authoriser l'accès
à vos donnés de localisation ?" delegate:self cancelButton
Title:@"Non" otherButtonTitles:@"Oui, continuer", nil];
        [alertView show];
    } else if ([CLLocationManager authorizationStatus] != kCL
AuthorizationStatusDenied
          && [CLLocationManager authorizationStatus] != kCL
AuthorizationStatusRestricted) {
        [self initialize];
        return;
    }
}
```

Instancions le CLLocationManager et les régions iBeacon.

```
- (void)initialize {
   [self initializeLocationManager];
   [self initBeaconRegions];
}
```

À l'aide de la méthode initWithUUIDString décrite dans l'introduction, nous allons créer un nouvel objet de type CLBeaconRegion en spécifiant l'identifiant unique de la région. Dans notre exemple, l'UUID utilisé est celui qui est imposé par défaut dans les bornes du fabriquant Estimote.

Monitorons les régions

Nous pouvons maintenant commencer à monitorer les régions avec startMonitoringForRegion.
La méthode requestStateForRegion: forcera la détection de la région, même si elle avait déjà été repérée.

```
- (void)initBeaconRegions {
 CLBeaconRegion *beacon = [[CLBeaconRegion alloc] initWith
ProximityUUID:
   [[NSUUID alloc] initWithUUIDString:@"B9407F30-F5F8-466E-
AFF9-25556B57FE6D"]
                        identifier:@"Xebia Beacon Region"];

   // Ajouter plus de beacons si nécessaire
   self.allRegions = @[beacon];

   for (CLBeaconRegion *region in self.allRegions) {
       [self.locationManager startMonitoringForRegion:region];
       [self.locationManager requestStateForRegion:region];
   }
}
```

Enfin, créons le CLLocationManager

```
- (void)initializeLocationManager {
 // Initialise un nouveau Location Manager
 self.locationManager = [[CLLocationManager alloc] init];
 self.locationManager.delegate = self;
}
```

Prenons en charge le protocole CLLocationManagerDelegate

La partie que nous allons maintenant écrire nous est fournie par l'implémentation du protocole CLLocationManagerDelegate.

La méthode didDetermineState est appelée par le CLLocationManager lorsqu'une région est traversée en même temps que locationManager:didEnterRegion: et locationManager:didExitRegion:. Également, le location manager appelle cette méthode en réponse à l'invocation de requestStateForRegion: que nous avons invoquée ci-dessus. Lorsqu'un changement d'état intervient, nous lançons ou arrêtons la détection précise des iBeacons à l'aide des méthodes startBeaconRanginInRegion: et stopBeaconRangingInRegion:.

```
#pragma mark - CLLocationManagerDelegate

- (void)locationManager:(CLLocationManager *)manager didDetermine
State:(CLRegionState)state forRegion:(CLRegion *)region {

 if (manager != self.locationManager) {
   return;
 }

 if (state == CLRegionStateInside) {
   [self startBeaconRangingInRegion:region];
 }

 else if (state == CLRegionStateOutside) {
   [self stopBeaconRangingInRegion:region];
 }
}


- (void)locationManager:(CLLocationManager *)manager didEnter
Region:(CLRegion *)region {
 if (manager != self.locationManager) {
   return;
 }

 [self postValueChangedNotification];
 [self startBeaconRangingInRegion:region];
}

- (void)locationManager:(CLLocationManager *)manager didExit
Region:(CLRegion *)region {
 if (manager != self.locationManager) {
   return;
 }
 [self postValueChangedNotification];
 [self stopBeaconRangingInRegion:region];
}
```

Détectons les beacons

Une fois l'implémentation du monitoring terminée, il est temps de s'occuper du ranging, c'est à dire la détection précise des beacons visibles. Lorsque ces derniers sont repérés, ajoutons-les au NSMutableArray currentBeacons et lançons une notification, implémentée ci-dessous.

```
- (void)locationManager:(CLLocationManager *)manager didRange
Beacons:(NSArray *)beacons inRegion:(CLBeaconRegion *)region {
 self.currentBeacons = [NSMutableArray array];

   // Trouve le beacon le plus proche
 if ([beacons count] > 0) {
   CLBeacon *closestBeacon = beacons[0];
   if (closestBeacon.proximity == CLProximityImmediate) {
     [self.currentBeacons addObject:closestBeacon];
     [self postValueChangedNotification];
   }

   // Trouve les autres beacons
   else if (closestBeacon.proximity == CLProximityNear) {
     [self.currentBeacons addObject:closestBeacon];
     [self postValueChangedNotification];
   }

   else {
     [self.currentBeacons addObject:closestBeacon];
     [self postValueChangedNotification];
   }
 }
   // Aucun beacon n'a été trouvé - le signal a été perdu
 else {
     [self postValueChangedNotification];
     return;
 }
}
```

Des implémentations ultérieures du protocole CLLocationManagerDelegate nous permettent de récupérer une des erreurs intervenues dans la détection des beacons.

```
- (void)locationManager:(CLLocationManager *)manager ranging
BeaconsDidFailForRegion:(CLBeaconRegion *)region withError:
(NSError *)error {
 [self postValueChangedNotification];

 if (error.description) {
   NSLog(@"Location error while ranging. Description: %@",
 error.description);
 }
}
- (void)locationManager:(CLLocationManager *)manager monitoring
DidFailForRegion:(CLRegion *)region withError:(NSError *)error {
 [self postValueChangedNotification];
 if (error.description) {
   NSLog(@"Location error while monitoring. Description: %@",
 error.description);
 }
}
```

Opérons sur l'état de ranging

Les deux méthodes suivantes permettent le lancement et l'arrêt de l'opération de ranging. Par conséquent, une fois évoqué stopRangingBeaconsInRegion, nous ne recevrons plus d'appels à locationManager:manager didRangeBeacons:beacons inRegion:region

```
- (void)startBeaconRangingInRegion:(CLRegion *)region {
 [self.locationManager startRangingBeaconsInRegion:(CLBeacon
Region *)region];
}

- (void)stopBeaconRangingInRegion:(CLRegion *)region {
 [self.locationManager stopRangingBeaconsInRegion:(CLBeacon
```

```
Region *)region];
}
```

Implémentons maintenant les méthodes postValueChangedNotifications et allBeacons. La première envoie à l'aide du NSNotificationCenter une nouvelle notification de nom postNotificationName:XBBeaconLocation ManagerValueChangedNotification, tandis que la seconde fournit un accès externe à la liste des beacons calculée précédemment.

```
- (void)postValueChangedNotification {
  [[NSNotificationCenter defaultCenter] postNotificationName:
XBBeaconLocationManagerValueChangedNotification object:nil user
Info:nil];
}

- (NSArray *)allBeacons {
   return _currentBeacons;
}
```

Pour conclure avec XBBeaconLocationManager.m, récupérons les changements d'état des autorisations d'accès à la géolocalisation, à l'aide de CLLocationManagerDelegate.

```
- (void)locationManager:(CLLocationManager *)manager didChange
AuthorizationStatus:(CLAuthorizationStatus)status {
  if (status == kCLAuthorizationStatusAuthorized) {
    [self initialize];
  } else if (status == kCLAuthorizationStatusDenied) {
    [[[UIAlertView alloc] initWithTitle:nil
                                message:NSLocalizedString(@"Merci
d'activer les services de localisation.", nil)
                                delegate:nil
                      cancelButtonTitle:nil
                      otherButtonTitles:@"OK", nil] show];
  }
}
@end
```

AppDelegate

Afin de créer une instance singleton de XBBeaconLocationManager, nous allons modifier l'AppDelegate de notre application.

```
- (BOOL)application:(UIApplication *)application didFinish
LaunchingWithOptions:(NSDictionary *)launchOptions {
   [self initBeaconTracking];
   return YES;
}
- (void)initBeaconTracking {
   [XBBeaconLocationManager sharedManager];
}
```

En appelant la méthode sharedManager, l'instance shared (partagée) est chargée en mémoire et commence à détecter les transitions sur les CLBeaconRegion.

Implémentation finale de l'interface

Il ne nous reste plus qu'à afficher les éléments dans le tableau. Revenons au XBBeaconViewController en remplaçant les méthodes suivantes :
Dans initObservers, nous allons écouter les notifications nommées XBBeaconLocationManagerValueChangedNotification envoyées par le singleton XBBeaconLocationManager. Lorsqu'une nouvelle notification est reçue, nous déchargeons les données à l'aide de la méthode reloadData.

```
- (void)initObservers {
   __weak typeof(self) weakSelf = self;
     self.locationManagerObserver = [[NSNotificationCenter
defaultCenter] addObserverForName:XBBeaconLocationManager
ValueChangedNotification object:nil queue:nil usingBlock:
^(NSNotification *note) {
        [weakSelf.tableView reloadData];
   }];
}
```

Pour terminer, implémentons cellForRowAtIndexPath et numberOfRowsInSection. Le premier s'occupe de récupérer les beacons détectés par le XBBeaconLocationManager et de les afficher à l'écran. Si aucun beacon n'est détecté, la cellule vide ("Aucun résultat") sera affichée à l'écran.

```
- (void)locationManager:(CLLocationManager *)manager didChange
AuthorizationStatus:(CLAuthorizationStatus)status {
  if (status == kCLAuthorizationStatusAuthorized) {
    [self initialize];
  } else if (status == kCLAuthorizationStatusDenied) {
    [[[UIAlertView alloc] initWithTitle:nil
                                message:@"Merci d'activer les
services de localisation."
                                delegate:nil
                      cancelButtonTitle:nil
                      otherButtonTitles:@"OK", nil] show];
  }
}
```

Le second, numberOfRowsInSection, nous renvoie le nombre de beacons détectés ou 1 (afin d'afficher la cellule "Aucun résultat") à l'écran.

```
- (NSInteger)tableView:(UITableView *)tableView numberOfRows
InSection:(NSInteger)section {
   NSInteger allBeacons = [[XBBeaconLocationManager shared
Manager].allBeacons count];
   return allBeacons ? allBeacons : 1;
}
```

Conclusion

Avec un exemple simple, nous avons montré comment utiliser la technologie iBeacon pour détecter les émetteurs sur nos iDevices. Mais comment faire si nous n'avons pas d'émetteurs ? Il est possible de simuler un iBeacon à l'aide du projet Open Source BeaconOSX, disponible à l'adresse : https://github.com/mttrb/BeaconOSX. Il s'agit d'une application avec une interface graphique simple que l'on peut paramétrer depuis son Mac.
Pour terminer, si vous désirez plus de renseignements ou souhaitez

télécharger la totalité du code source de notre application, rendez-vous ici : https://github.com/xebia-france/ibeacon-tutorial-ios.

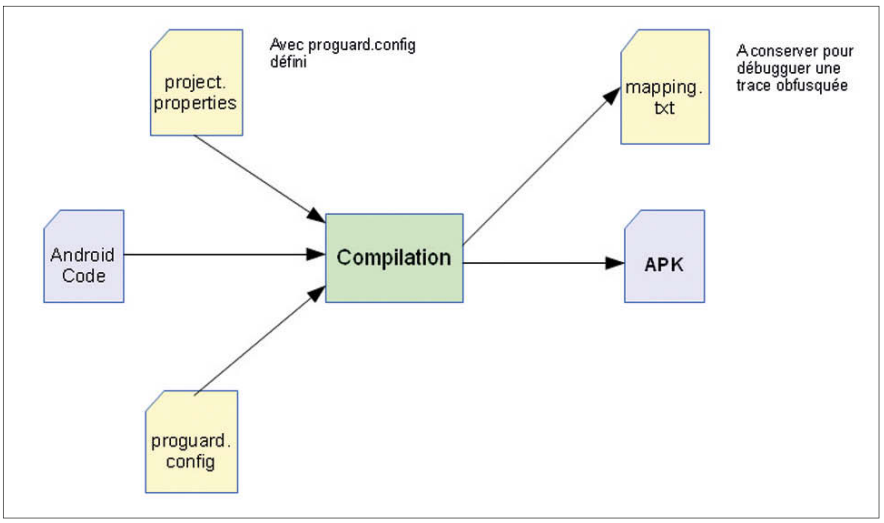
Simone **CIVETTA**
Xebia

Obfusquez votre code Android avec ProGuard

Fourni en standard avec le SDK Android, l'outil ProGuard est une solution d'obfuscation puissante et fortement configurable mais qui reste malheureusement méconnue des développeurs Android. Les avantages de sa mise en place lors de la génération d'un APK sont pourtant nombreux, comme nous allons le voir dans la suite de cet article.

L'obfuscation est une technique utilisée en informatique pour modifier du code compilé afin de le rendre incompréhensible et le protéger d'éventuelles opérations de décompilation. Ainsi, le but premier de l'obfuscation est de protéger le code source d'une application, et donc la propriété intellectuelle d'un développement. Pour mettre en place une telle technique, les développeurs Android s'avèrent chanceux puisque le SDK s'amène en standard avec une solution puissante et complète nommée ProGuard.

Disponible au sein du répertoire tools du SDK, ProGuard est un outil permettant d'obfusquer du code Android mais également de le réduire et de l'optimiser. Pour ce faire, il va supprimer le code inutilisé et renommer les classes, champs et méthodes avec des noms à la sémantique plus obscure. Le résultat de ces transformations est un APK plus léger et plus difficilement utilisable par des tiers recourant à des techniques de reverse engineering. ProGuard se révèle donc indispensable pour les applications utilisant des fonctionnalités sensibles que l'on souhaite protéger.



Processus de compilation avec proguard activé sous Android.

Activer ProGuard

ProGuard est pleinement intégré au système de build sous Android que l'on soit sous Eclipse, avec le plugin ADT, ou que l'on soit sous Android Studio, avec le système de build Gradle. Utiliser ProGuard n'est en revanche pas une obligation, et c'est ce qui fait qu'il est relativement méconnu des développeurs Android, mais sa mise en oeuvre est fortement recommandée. A la création d'un projet Android, un fichier proguard.cfg est automatiquement généré à sa racine. Il s'agit du fichier de paramétrage définissant comment l'outil va optimiser et obfusquer le code du projet. Le fichier généré correspond à un paramétrage basique pas forcément adapté à toutes les applications. Il est donc fort à parier qu'il vous faudra le paramétrer pour l'adapter aux besoins spécifiques de votre application.

Pour activer ProGuard, il faut définir la propriété proguard.config au sein du fichier project.properties se trouvant à la racine d'un projet Android. Cette propriété définit le chemin vers le fichier de configuration ProGuard utilisé pour le projet. Côté Gradle, la configuration se fait directement au sein du fichier de build.

Exécuter ProGuard

L'exécution de ProGuard est ensuite automatique lors du build d'application. Ce dernier pouvant être lancé via un Ant release, via l'export d'APK d'Eclipse ou par un build Gradle. Bien entendu, ProGuard n'est jamais exécuté en mode debug puisque cela reviendrait à compliquer le debug durant la phase de développement. Suite à son exécution, l'outil génère 4 fichiers en sortie dans le répertoire proguard à la racine du projet sous Eclipse ou bien dans /bin/proguard si Ant a été employé :

- dump.txt décrivant la structure interne de toutes les classes de l'APK généré,
- mapping.txt listant le mapping entre la version originale des classes et celle obfusquée. Ce fichier est primordial pour débugguer des stack traces d'applications obfusquées comme nous le verrons par la suite,
- seeds.txt listant les classes et membres non obfusqués,
- usage.txt détaillant le code retiré de l'APK.

Vous l'aurez compris, garder ces fichiers est essentiel lorsque l'on publie une application obfusquée.

Réduire la taille de son code

Au gré des différentes versions d'une application, la base de code associée ne cesse de grossir. De fait, la taille de l'APK grossit en conséquence induisant des problématiques importantes. En effet, plus une application est grosse, plus elle a de chances de ne pas être téléchargée par des utilisateurs dont les réseaux présentent de faibles débits ou possèdent des périphériques anciens aux mémoires limitées. Réduire la taille de son APK doit donc être un souci constant, et s'avère extrêmement avantageux. Dans tous les cas, même si le gain en taille n'est pas une de vos préoccupations première, configurer ProGuard pour détecter le code inutilisé et le supprimer est une bonne pratique qui peut être réalisée en activant les options suivantes dans le proguard.cfg :

```
-dontusemixedcaseclassnames
-dontskipnonpubliclibraryclasses
-verbose
```

Obfusquer son code

Le sujet premier de cet article étant l'obfuscation de code sous Android, il serait temps de mettre en oeuvre ProGuard à ces fins. Comme dit précédemment, obfusquer son code Android est une pratique indispensable du fait des nombreux outils existants pour extraire le contenu d'un APK. Par défaut, ProGuard obfusque tout le contenu de votre projet. Prenons le code original suivant comme exemple :

```
public class Data {
  private final int mId;
  private final int mResult;
  private final String mMessage;

  public Data(int id, int result, String message) {
    mId = id;
    mResult = result;
    mMessage = message;
  }

  public int getId() {
    return mId;
  }

  public int getResult() {
    return mResult;
  }

  public String getMessage() {
    return mMessage;
  }
}
```

Une fois obfusqué par ProGuard, ce code aura l'allure suivante :

```
public class a {
  private final int a;
  private final int b;
  private final String c;

  public a(int paramInt1, int paramInt2, String paramString) {
    this.a = paramInt1;
    this.b = paramInt2;
    this.c = paramString;
  }

  public int a() {
    return this.a;
  }

  public int b() {
    return this.b;
  }

  public String c() {
    return this.c;
  }
}
```

Le fichier mapping.txt généré par ProGuard permet ensuite de faire le lien entre la version obfusquée d'un code et la version originale. Il est donc indispensable pour déboguer des stack traces d'applications publiées. Inutile de préciser qu'il doit donc être conservé et versionné.

Bien que la reflection soit fortement déconseillée sous Android pour des questions de performance notamment, certains développeurs y ont recours dans leurs projets pour diverses raisons. Dans le cas de l'activation de ProGuard et donc de l'obfuscation, l'application ne fonctionnera plus lors d'appels à des méthodes ou classes par reflection et lancera un ClassNotFoundException ou un MethodNotFoundException.

Optimiser son code

Les travaux d'optimisation sur les classes compilées vont permettre d'implémenter de petites optimisations basées sur la version de Java ciblée. Celles-ci vont permettre un gain de performance sur certaines opérations du langage. Néanmoins, des incompatibilités sont connues avec certaines versions de Dalvik (à voir avec le moteur d'exécution ARM maintenant). Si ces options d'optimisation sont activées, il faudra donc procéder à un travail de test en profondeur sur l'APK produit, pour vérifier que l'application ne souffre pas de ces incompatibilités. Voici un exemple d'options d'optimisation à activer :

```
-optimizations !code/simplification/arithmetic,!code/simplification
/cast,!field/*,!class/merging/*
-optimizationpasses 5
-allowaccessmodification
-dontpreverify
```

Préserver du code

L'outil ProGuard traite par défaut tout le contenu d'un projet, ce qui implique donc qu'il va traiter les bibliothèques ajoutées en dépendances. S'il s'agit de bibliothèques open source par exemple, il n'y a aucune raison de les obfusquer puisque leur contenu est déjà public.

Dans ces cas-là, pour améliorer le temps de build notamment, il est préférable de les exclure des traitements réalisés par ProGuard.

Considérons un projet utilisant la bibliothèque populaire ActionBarSherlock, et la version 4 de la bibliothèque de support Android. Pour préserver le code de ces classes, les lignes suivantes seront à ajouter au fichier de configuration :

```
-keep class android.support.v4.app.** { *; }
-keep interface android.support.v4.app.** { *; }
-keep class com.actionbarsherlock.** { *; }
-keep interface com.actionbarsherlock.** { *; }
```

Décoder une stack trace obfusquée

Lorsqu'un bug se produit sur une application obfusquée publiée, la stack trace produite contient les noms des classes et méthodes en version obfusquée fort logiquement. Difficile donc pour le développeur de trouver l'origine du problème. C'est là qu'intervient le fichier mapping.txt généré lors de la phase d'obfuscation par ProGuard. En conservant ce fichier correspondant à une version obfusquée donnée, il est possible de reconstituer une stack trace compréhensible via l'outil retrace fourni par ProGuard :

```
retrace.sh -verbose mapping.txt obfuscated_trace.txt
```

Conclusion

Technique primordiale pour préserver le code source d'une application du reverse engineering, l'obfuscation reste souvent délaissée par les développeurs Android. En effet, très peu d'entre eux la mettent en oeuvre lors de la génération de l'APK d'une application alors qu'avec l'intégration de la solution ProGuard au process de build, la plateforme leur a simplifié le travail au maximum. Puissant et configurable à souhait, ProGuard gagne à être connu et profite à tous les types d'applications.

L'outil open source propose un nombre important d'options de configuration qu'il conviendra d'aller étudier sur le site du projet pour tirer partie au mieux de son utilisation.

Sylvain **Saurel**
Ingénieur d'Etudes Java / JEE
sylvain.saurel@gmail.com

Découverte des Chrome Apps Mobiles

2e partie

En Janvier 2014, la mise à disposition d'une chaîne de développement permettant de créer ses applications Chrome "natives" sur les plateformes mobiles a été annoncée par Google. Après avoir fait un premier tour d'horizon sur les Chrome Apps « Desktop », nous allons maintenant faire un tour des Chrome Apps sur Mobile.

Il est tentant de penser que les Chrome Apps mobile s'exécutent en s'appuyant sur l'application Chrome installée sur votre device (que ce soit sur Android ou sur IOS). Il s'agit en réalité d'une **chaîne de compilation** - actuellement en mode Developer Preview (c'est-à-dire, pas encore stable) – qui s'appuie sur Apache Cordova pour créer des applications natives à partir du code d'applications Chrome. Nous allons faire un tour d'horizon de ce toolkit et les particularités des applications Chrome pour mobile.

Cordova permet de créer des applications natives pour plateformes mobiles (entre autres pour Android, IOS, BlackBerry, etc.) à partir d'application Web codées en HTML et JavaScript, tout en offrant un ensemble d'API plus puissantes permettant de tirer parti des fonctionnalités offertes par le matériel.

Quand Adobe a relâché PhoneGap à la communauté Open Source, il a été appelé Cordova et reversé à la fondation Apache http://cordova.apache.org/. Google contribue activement. L'idée est de proposer les API « clés » de Chrome pour les utiliser au sein des plateformes Android et IOS. Cordova va créer une application « coquille » native autour de notre application Chrome codée en HTML et JavaScript. N'hésitez pas à jeter un œil sur l'article « Créer des applications hybrides avec PhoneGap » publié dans Programmez n°164.

La Chrome App s'exécute au sein des Webviews d'applications natives créées avec Cordova. Sur IOS, c'est le UIWebControl. Sur Android, seul KitKat (Android 4.4) propose un WebView s'appuyant sur un moteur Chromium (l'implémentation Open Source de Chrome) qui inclut donc les avancées du moteur Javascript V8 ainsi qu'un meilleur support des standards Web. Cependant, cette version de Chrome est pour l'instant limitée à la version 30 et vit indépendamment de la version « application » installée. Du coup, l'implémentation de Chrome dans les Webviews comporte encore quelques limitations. Pour les versions antérieures d'Android, on s'appuie sur le moteur de rendu d'Android Browser (qui, lui, est basé sur WebKit). Plus de détails sur : https://developer.chrome.com/multidevice/webview/overview

Limitations

Il y a plusieurs limitations à l'heure actuelle, car les outils sont encore en cours de développement, notamment, certaines fonctionnalités Web sont désactivées ; heureusement elles sont portées par Cordova. Ainsi une application Chrome peut ne pas fonctionner telle quelle sur Mobile. La plupart des problèmes ergonomiques sont également recensés, car une application Desktop ne réagira pas de la même façon qu'une plateforme Mobile :

- Problèmes de layout sur des petits écrans, notamment en orientation portrait. Utiliser les media-queries CSS permet d'optimiser l'affichage du contenu pour les plus petits écrans.
- La taille des fenêtres indiquées via `chrome.app.window` sont ignorées et prennent la dimension de l'écran du mobile. Il est préférable de supprimer les dimensions codées en dur et de concevoir son application avec différentes tailles à l'esprit.
- etc.

Beaucoup d'API sont disponibles, parmi lesquelles : identity (OAuth2), Payments, pushMessaging, sockets, notifications (seulement Android), storage, syncFileSystem, alarms, idle, power. Cependant, toutes les API JavaScript de Chrome ne sont pas implémentées et toutes les fonctionnalités des applications Chrome « Desktop » ne sont pas disponibles : notamment, la balise webview, indexedDB, getUserMedia(), le code natif NaCL. Le tableau ci-contre recense les fonctionnalités portées sur mobile par plateforme.

Installation

Dans cette partie nous allons nous concentrer sur la création d'un nouveau projet pour la plateforme Android. L'installation et la création de projet pour IOS sont similaires en adaptant à l'IDE XCode.

Pour les deux plateformes

Il faut installer Node.js et le plugin Chrome-Cordova-Application (CCA).

Node.js est nécessaire pour les installations suivantes et la gestion de dépendances via NPM. Il suffit de récupérer l'installeur sur le site officiel : http://nodejs.org/ et de lancer l'installation standard.

Chrome-Cordova-Application intègre Cordova ainsi que les plugins nécessaires à la constitution de l'application Chrome mobile. L'installation se fait via npm en lançant la commande `npm install -g cca`.

Pour Android

Il faut commencer par installer Java JDK et le SDK Android (http://developer.android.com/sdk) sous `C:\Development\android-sdk-windows` par exemple. La variable d'environnement PATH doit contenir le chemin vers les `tools` et `platform-tools` du SDK :

```
C:\Development\android-sdk-windows\platform-tools
C:\Development\android-sdk-windows\tools
```

Il faut également s'assurer de l'ajout du SDK cible (depuis le Android SDK Tools) ainsi que du driver du device. A cette étape, la commande `Android` et tous les outils associés doivent être accessibles en ligne de commande et le device doit être reconnu par adb.

Pour iOS

Pour le développement IOS, il est nécessaire d'utiliser OS X, installer XCode 5. Nous utilisons également **ios-deploy** (https://github.com/phonegap/ios-deploy) pour installer et déployer des applications depuis la ligne de commande. Ios-deploy s'installe en lançant la commande `npm install -g ios-deploy`, ainsi que **ios-sim** (https://github.com/phonegap/ios-sim) pour nous permettre de lancer l'application sur le simulateur en ligne de commande. **ios-sim** s'installe depuis npm en lançant la commande `npm install -g ios-sim`.

Vérification

Vérifier si tout est bien installé en lançant la commande `cca checkenv`:

```
$ cca checkenv
cca v0.0.9
## Checking that tools are installed
Android SDK detected.
```

	Android	IOS
Alarmes	✔	✔ (1)
Système de Fichier	✔	✔
Internationalisation	✔ (2)	✔ (2)
Identité	✔	✔
Veille (idle)	✔	✔
Notifications	✔	û
Power	✔	✔
Push Messaging	✔ (3)	✔ (3)
Socket TCP/UDP	✔	✔
Stockage	✔ (4)	✔ (4)
API syncFileSystem	✔ (3)	✔ (3)
Paiements	✔ (5)	✔ (5)
Bluetooth	✘	✘
Commandes	✘	✘
Menus contextuels	✘	✘
Galerie médias	✘	✘
Permissions	✘	✘
Port Série	✘	✘
Info Système	✘	✘
Synthèse vocale	✘	✘
types	✘	✘
USB	✘	✘
Webstore	✘	✘
Balise <webview>	✘	✘
Client Natif (NaCl)	✘	✘

(1) Les alarmes ne se déclenchent que si l'application est au premier plan
(2) Les méthodes JS (chrome.i18n) sont fonctionnelles mais pas les instructions CSS.
(3) Qualité Bêta
(4) Local storage only. Le sync storage fonctionne en mode dégradé (comme storage.local)
(5) Qualité Alpha

Cette liste va évoluer au fur et à mesure de l'avancement, les détails sont disponibles ici : https://github.com/MobileChromeApps/mobile-chrome-apps/blob/master/docs/APIStatus.md

A noter que l'étape de **checkenv** sera effectuée à chaque fois que la commande cca est lancée. Pour utiliser Cordova avec une application Chrome existante, on utilisera l'outil cca (Cordova Chrome App) en ligne de commande.

Création d'un nouveau projet

Il est possible de créer un projet vide ou de créer un projet à partir des sources d'une application Chrome existante.
Pour la création d'un projet vide :

```
$ cca create MonApp
```

`MonApp` représente le nom de l'application.
Si les sources de l'application Chrome existent déjà, il est possible de les importer lors de la création du projet en utilisant la commande :

```
$ cca create YourApp --copy-from=path/to/manifest.json
```

Les ressources seront alors copiées dans le répertoire `www`.

Spécificité des développements Chrome App mobile

Cycle de vie et événements

Dans le modèle Chrome, les applications peuvent être notifiées par plusieurs évènements de cycle de vie applicatif. Ils sont notamment lancés lorsque l'application se lance et s'arrête, ou en réponse à certaines actions utilisateur, comme l'installation de mise à jour. Comprendre ceci est crucial pour développer une application qui peut sauver et restaurer son état effectivement sur un desktop et mobile Fig.1. On compte parmi les évènements : *onRestarted*, *onInstalled*, *onLaunched*, *onSuspend*, *onSuspendCancelled*. Malgré le fait que ces événements soient disponibles sur les différentes plateformes, ils ne sont pas déclenchés au même moment et ils ont une sémantique différente entre l'environnement Desktop et l'environnement mobile dont il faut avoir conscience pour le développement de son application Chrome App Mobile.

Sur Desktop :

- Au lancement, la page background est lancée,
- *onRestarted* peut être lancé si Chrome s'est arrêté inopinément,
- *onInstalled* peut être lancé si l'application vient juste d'être installée ou mise à jour,
- Quand l'application est lancée : *onLaunched* est déclenché,
- Quand toutes les fenêtres de l'application sont mises en arrière-plan, l'application peut être suspendue. Dans ce cas, onSuspend est déclenché.
- Si l'application est remise au premier plan, avant d'être terminée, *onSuspendCanceled* est déclenché.

Sur Mobile :
En général, les utilisateurs ne ferment pas les fenêtres ni les applications directement. C'est l'OS qui s'occupe de fermer les applications (elles sont suspendues et peuvent être terminées à n'importe quel moment), mais l'utilisateur n'en a pas conscience, car les applications ne sont pas visibles en même temps. La plupart des applications ne démarrent pas sans l'intention d'être lancées et ne tournent pas en arrière-plan (dès qu'elles sont cachées, elles sont suspendues). Ainsi l'évènementiel du cycle de vie est différent, trois évènements peuvent apparaître lors du démarrage :

- *onRestarted* : seulement si l'application s'est arrêtée prématurément. Cela signifie qu'un évènement précédant *onSuspend* n'a pas été exécuté jusqu'au bout.
- *onInstalled* - quand la version indiquée dans le `manifest.json` a changé (et donc également lors de la première installation).
- *onLaunched* - dans tous les cas au lancement de l'application étant donné que les applications sont immédiatement lancées au démarrage sur mobile.

Ces évènements seront lancés après que les scripts background aient été exécutés, il faut donc attacher des listeners pour ceux-ci. Si l'application n'est pas éteinte, et est seulement remise au premier plan, seul *onSuspendCanceled* sera déclenché Fig.2.
Lors d'une reprise :

- Chaque fois que l'utilisateur switche d'application, l'application Chrome devient suspendue. *onSuspend* est déclenché, parce que l'OS peut détruire l'application à n'importe quel moment.
- Si l'utilisateur revient sur l'application avant que l'OS ne l'ait détruite, *onSuspendCanceled* est déclenché.
- C'est la responsabilité de l'application de sauver son état quand *onSuspend* est appelé.
- Si l'application est tuée par l'OS pendant que l'application est proprement suspendue, c'est un arrêt "propre". Lors du prochain démarrage, *onLaunched* sera donc déclenché.

Si l'application est tuée par l'OS alors que la suspension est incomplète, alors, *onSuspendCanceled* ne sera déclenché, mais à la place *onRestarted*.

Manifest Mobile

Le fichier manifest.mobile.json est un complément au manifest.json qui permet d'indiquer les paramétrages spécifiques aux applications mobiles. Par exemple, les spécifications de version de Android

(versionCode) ou IOS (CFBundleVersion), et qui n'apparaissent donc pas dans le Manifest générique.

```
{
  "packageId": "com.your.company.HelloWorld",
  "versionCode": 1,
  "CFBundleVersion": "1.1.1"
}
```

La documentation des API indique si des propriétés doivent être ajoutées spécifiquement au fichier manifest.mobile.json. La commande `cca prepare` va ensuite constituer les fichiers natifs en récupérant les paramétrages issus des manifest.json et manifest.mobile.json. Par exemple, le Manifest Android (Manifest.xml nécessaire à la constitution de l'APK) : le champ `android:versionName` est déduit de la version de l'application renseignée dans le manifest.json et le champ `android:versionCode` est déduit du fichier manifest.mobile.json. La commande `cca prepare` est automatiquement ré-exécutée lors des commandes `cca build`, `cca run` ou `cca emulate`. Par contre, lors du développement depuis l'IDE, elle ne sera pas exécutée. Il faudra donc lancer la commande `cca prepare` manuellement pour s'assurer que les modifications des manifest Chrome soient correctement prises en compte.

Déploiement

Deux options sont envisageables pour compiler, packager et tester l'application : utiliser l'outil en ligne de commande, ou alors les lancer depuis l'IDE. Dans les deux cas, vérifiez que le mobile est connecté en USB et reconnu par Eclipse ADT ou Xcode.

Lors de l'étape d'empaquetage, l'application native est créée (sous la forme d'un APK pour Android et d'un IPA pour IOS) et les ressources Web de l'application Chrome sont intégrées à l'application. A cette étape, il n'y a rien de spécifique à Chrome, ni à CCA. A noter que comme l'application native est créée, il faudra disposer des certificats (et donc d'un compte développeur pour IOS). Pour plus de détails sur ce point et sur le lancement d'application depuis l'IDE, n'hésitez pas à aller voir à cette adresse pour plus de détails https://github.com/MobileChromeApps/mobile-chrome-apps/blob/master/docs/Develop.md. Le déploiement peut se faire comme une application native classique, à savoir sur un émulateur en lançant les commandes `cca emulate android` ou `cca emulate ios` respectivement pour les cibles Android et IOS, ou alors directement sur le device en lançant les commandes `cca run android` ou `cca run ios`.

Debugger avec Chrome Apps Developper Tools (Android only)

Le Chrome Apps Developer Tool (Chrome ADT) est une application Android qui permet de télécharger et exécuter une application Chrome sans utiliser l'outil cca. L'application est encore en cours de développement (pre-alpha) mais il est déjà disponible en version pré-packagée APK à cette adresse : https://github.com/MobileChromeApps/harness/releases/ (prendre la dernière version construite). Elle pourra être installée sur le mobile de tests. Cette application Chrome ADT offre deux facilités pour le développeur, pour exécuter des applications Chrome sans passer par la constitution d'une application native de la commande `cca build`.

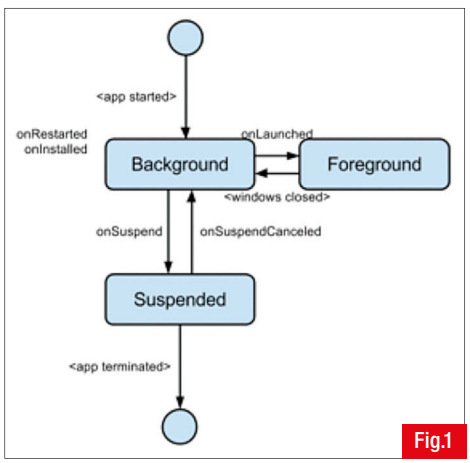


Fig.1

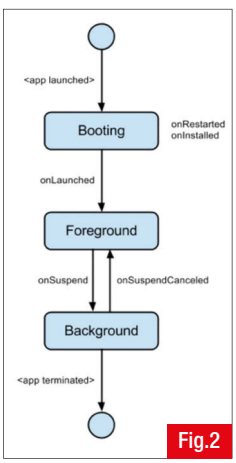


Fig.2

Test à partir de CRX

Cette fonctionnalité permet d'avoir un aperçu rapide d'une application Chrome déjà packagée en CRX (issu d'une application empaquetée pour Chrome Desktop) sur le device Android, sans passer par une phase de packaging CCA.

Test à partir des sources statiques

Ici, on utilise seulement CCA pour « servir » les ressources web (HTML, CSS, JS, etc.) que l'application Chrome ADT pourra exécuter au sein d'un conteneur de test. Pour cette dernière option, on utilise la commande « cca serve » qui va créer un serveur web de ressources statiques (par défaut sur http://localhost:8000).

```
D:\projets\MonApp>cca serve
cca v0.0.9
…
Static file server running on port 8000 (i.e. http://local
host:8000)
CTRL + C to shut down
```

Depuis, l'application Chrome ADT, on indique l'URL sur laquelle cette application est servie, à savoir : `http://<adresse_ip>:8000`

Le gros avantage de l'outil est de supprimer les étapes d'empaquetage natif, qui peuvent être assez longues et répétitives, surtout pendant la phase de développement. Le cca serve offre en plus la possibilité de servir les ressources à chaud (dès qu'elles sont modifiées, elles sont disponibles), côté Chrome ADT, un simple « Update » permet d'avoir à disposition la dernière version de l'application. Un gain de temps considérable !

Distribution

Les applications Chrome Desktop sont distribuées sur le **Chrome Web Store** (http://chrome.google.com/webstore), alors que CCA encapsule les applications Chrome mobiles dans une application native et sont donc, quant à elles, distribuées sur les Store mobile dédiés, à savoir **Play Store** pour **Android**, et **l'App Store** pour **IOS**.

Conclusion

Pour les développeurs qui cherchent un moyen rapide de développer une application sur du multi-device, les applications Chrome Cordova Mobile fournissent une des solutions les plus faciles à mettre en place. L'outillage basé sur Cordova permet d'avoir une base commune ce qui permet aux développeurs d'avoir une application native et réellement portable sur tous les supports sans avoir à entrer trop en détails sur les spécificités de Android et IOS. Pour l'instant les API disponibles sont encore limitées, mais gageons que les équipes de développement de Chrome sauront enrichir leur outil pour permettre l'accès à toutes les API et offrir des fonctionnalités équivalentes entre Desktop et Mobile.

Florent **Dupont** (fdupont@sqli.com)
Expert Web & Mobilité chez SQLI - http://florent-dupont.blogspot.fr/

A la découverte des Xamarin.Forms

Annoncé le 28 Mai 2014, en même temps que la nouvelle version de Xamarin, Xamarin.Forms est un nouveau Framework, s'appuyant sur Xamarin. Il permet de pousser encore plus loin la mutualisation de code en offrant aux développeurs d'écrire le code de leur interface graphique une seule fois, le Framework se chargeant de faire le rendu correspondant en fonction du device et de l'OS cible.

L'objectif de Xamarin.Forms est simple : permettre le développement d'interfaces graphiques, utilisant des contrôles natifs en alliant la même base de code. On écrit donc son code une fois et c'est le moteur qui va se charger, lors de l'exécution de l'application, de faire le rendu des contrôles natifs, comme on peut le voir sur l'image 1.

Afin de permettre cette mutualisation de code, il a été nécessaire aux équipes Xamarin, de mettre à disposition un nouveau jeu d'APIs qui sont « converties » dans leur équivalent natif lors de l'exécution de l'application. Dans ces nouvelles APIs, on retrouve différents éléments ayant chacun des fonctionnalités dédiées. Ainsi, on va disposer de contrôles qui gèrent différents modèles de pages (ContentPage, MasterDetailPage, TabbedPage, etc.), des contrôles qui gérent la mise en page (StackLayout, RelativeLayout, GridLayout, etc.) et enfin des contrôles qui proposent des fonctionnalités, du plus basique (Bouton, Slider) à des contrôles plus évolués (ListView, Picker, etc.).

La création d'un projet Xamarin.Forms peut être faite de 2 façons différentes :

- En utilisant les « Portable Libraries »,
- En utilisant le concept de « Shared Project », introduit par Microsoft lors de la Build 2014 et permettant de mutualiser du code entre Windows Phone et Windows 8 : Fig.2.

Le choix de l'une ou l'autre des possibilités est à votre charge, sachant qu'il n'y a pas de règles précises : gardez juste en tête que le contenu des projets « Shared » est répliqué, à la compilation, sur les projets qui le référencent, alors que, dans le cas d'une « Portable Library », la compilation produit une DLL qui est ajoutée en référence aux projets.

Pour les développeurs que nous sommes, l'utilisation de Xamarin.Forms est extrêmement simple puisqu'au final, il ne s'agit que d'apprendre les nouvelles APIs. De plus, vous avez 2 possibilités pour créer vos interfaces graphiques avec Xamarin.Forms :

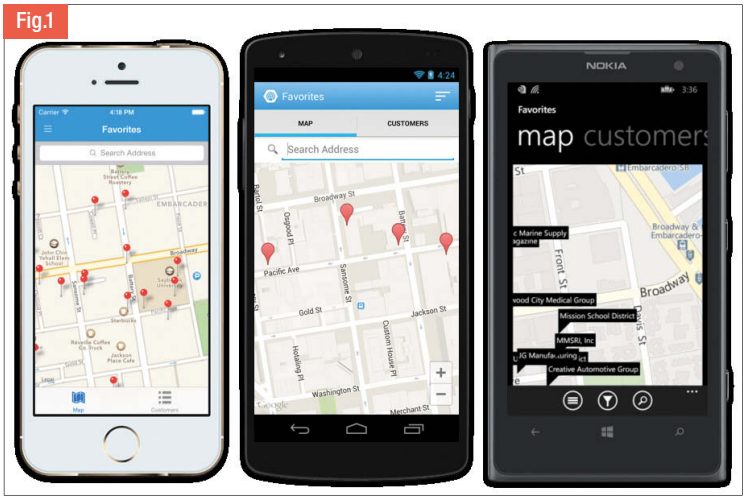
- Utiliser XAML : qui peut être très pratique si vous êtes développeur WPF/Windows Phone/Windows 8 car vous maitrisez déjà la syntaxe de XAML (donc cette solution peut vous convenir). Attention cependant, dans sa version actuelle, le plugin Xamarin.Forms intégré à Visual Studio ne propose pas l'IntelliSense, il vous faudra donc connaître l'API sur le bout des doigts (ou alors, vous devrez utiliser Xamarin Studio).
- Utiliser C# : qui peut s'avérer plus long à mettre en œuvre (générer une interface graphique à partir de code n'est pas le plus rapide) mais on dispose de tout l'écosystème de Visual Studio pour nous aider dans le développement (IntelliSense, plugins tels que Resharper, etc.).

Ainsi, l'exemple de code suivant vous permet de visualiser le code nécessaire pour la création d'une page de type « TabbedPage », soit une page avec plusieurs onglets (ou PivotItem, en fonction de la plateforme qui exécutera le code), dans laquelle 2 items ont été ajoutés :

```
public class MainPage : TabbedPage
{
  private readonly OrganizerView _organizerView;
  private readonly AttendeeView _attendeeView;

  public MainPage()
  {
```

Fig.1



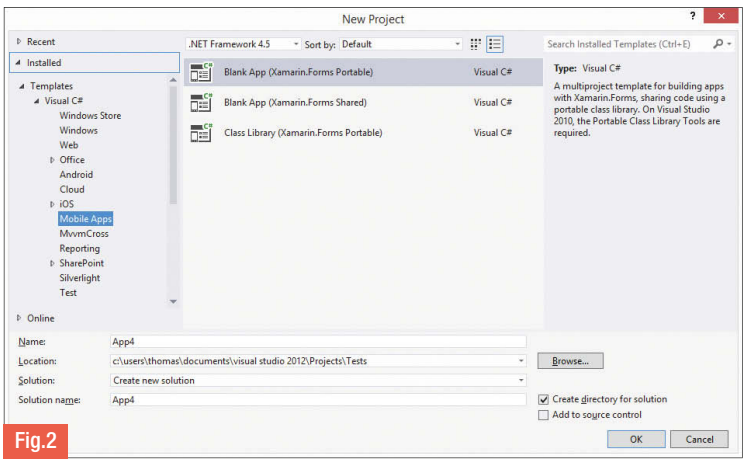


Fig.2

```
    this.Title = "Meeting Organizer";

    this._organizerView = new OrganizerView();
    this._attendeeView = new AttendeeView();

    this.Children.Add(this._organizerView);
    this.Children.Add(this._attendeeView);
  }
}
```

Pour gérer tout ce qui est affichage des données, interaction avec l'utilisateur, etc., Xamarin.Forms s'appuie énormément sur le principe de *databinding*, qui consiste à définir un objet qui servira de source de données, et à lier les composants de l'interface graphique aux propriétés de cet objet.

Si vous êtes familier du pattern MVVM, ce genre de notion ne vous est pas inconnu, et c'est une bonne chose car cela vous permettra de bien appréhender le système de binding utilisé dans Xamarin.Forms.

En effet, le code présenté ci-dessous permet, via la méthode **SetBinding**, de lier la propriété ItemsSource de la ListView à une

propriété « Meeting » de l'objet défini dans le BindingContext (autrement dit dans la source de données utilisée pour le binding) :

```
private readonly AttendeeViewModel _viewModel = new Attendee
ViewModel();

public AttendeeView()
{
  this.BindingContext = this._viewModel;

  this.Title = "Participant";

  this.CreateView();
}

private void CreateView()
{
  var listView = new ListView();
  listView.SetBinding(ListView.ItemsSourceProperty, "Meetings");

  var viewLayout = new StackLayout
  {
    VerticalOptions = LayoutOptions.FillAndExpand,
    Children = { listView }
  };

  Content = viewLayout;
}
```

Cependant, si vous exécutez le code précédent (et cela, quel que soit l'OS ou le device), vous remarquerez que rien n'est affiché directement dans l'interface graphique. Cela est dû au fait qu'il est nécessaire de définir la manière dont chacun des éléments de la liste va s'afficher. Pour cela, il faut définir la propriété ItemTemplate de la ListView et lui affecter un DataTemplate spécifique, comme on peut le voir ci-dessous :

```
public class MeetingCell : ViewCell
{
  public MeetingCell()
  {
    this.CreateView();
  }

  private void CreateView()
  {
    var dateLabel = new Label();
    dateLabel.SetBinding(Label.TextProperty, new Binding(path:
"Date", stringFormat: "{0:dd/MM/yyyy}"));

    var subjectLabel = new Label { Font = Font.SystemFontOfSize(35) };
    subjectLabel.SetBinding(Label.TextProperty, "Subject");

    var viewLayout = new StackLayout { Children = { dateLabel,
 subjectLabel }, Padding = new Thickness(0, 0, 0, 25)};

    View = viewLayout;
  }
}
```

La classe hérite de ViewCell, indiquant ainsi que l'on veut représenter une cellule de notre ListView. Ensuite, le code est similaire à ce qui a été vu précédemment : on utilise la méthode SetBinding des contrôles pour lier (éventuellement en utilisant des convertisseurs) leurs propriétés graphiques aux propriétés de l'objet utilisé comme source de données.
Il est à noter que, contrairement au code d'avant, la propriété BindingContext n'a pas été définie (tout du moins pas de manière explicite). En effet, ce modèle de données est utilisé pour chacun des éléments de la liste et, d'un autre côté, nous avons lié la source des éléments de la liste à une propriété de notre ViewModel (bien souvent une liste d'éléments, implémentant IList<T> ou IEnumerable<T>). Ainsi, lors de l'exécution de l'application, le moteur de binding sait que chacune des lignes de la ListView est en fait automatiquement lié à l'élément correspondant dans la liste utilisée comme source de données, ce qui explique que le contexte de données n'a pas besoin d'être défini explicitement. Enfin, il faut savoir que Xamarin.Forms met à la disposition des développeurs un mini-container IoC qui, outre le fait qu'il soit très rapide (Xamarin annonce des temps de démarrage inférieurs à 10ms), permet d'avoir du code qui reste spécifique à la plateforme. En effet, certains composants (tels que la partie authentification de Windows Azure Mobile Services) doivent être implémentés différemment en fonction de la plateforme sur laquelle ils sont exécutés. Pour cela, il est possible d'utiliser ce fameux container IoC, de la manière la plus simple possible. On commence par déclarer une interface, dans son projet Shared (ou sa Portable Library) puis on implémente l'interface, de manière spécifique, sur chacune des plateformes :

```
[assembly: Dependency(typeof(FacebookAuthenticationService))]
namespace MeetingOrganizer.Droid.Services
{
  public class FacebookAuthenticationService : IAuthentication
Service
  {
    public async Task AuthenticateAsync(MobileServiceClientProxy
 mobileServiceClientProxy)
    {
      await mobileServiceClientProxy.MobileService.LoginAsync
(Forms.Context, MobileServiceAuthenticationProvider.Facebook);
    }
  }
}
```

Notez l'utilisation de l'attribut **Dependency** qui permet, justement, d'enregistrer la classe indiquée dans le container. Enfin, pour utiliser la version correspondante à la plateforme sur laquelle l'application est en train de s'exécuter, il suffit, dans le ViewModel, d'utiliser la classe **DependencyService** :

```
await DependencyService.Get<IAuthenticationService>().Authenticate
Async(MobileServiceClientProxy.Current);
```

Au cours de cet article, nous avons été en mesure de constater que l'utilisation de Xamarin.Forms est relativement simple au final. Certes, nous ne sommes pas rentrés dans tous les détails techniques (il s'agissait ici d'un article de présentation) mais cela vous permet d'appréhender cette nouvelle technologie (qui ne cesse d'évoluer !) et qui, nous avons pu le voir, semble très prometteuse. Certes, il y a des choses à changer/optimiser mais il faut garder en tête que ce n'est qu'un début.

Thomas **LEBRUN** – *Consultant Infinite Square*
tlebrun@infinitesquare.com

ZF2 Discovery et le modèle MVC

2ère partie

Nous poursuivons notre découverte technique de ZF2 Discovery et le modèle MVC.

Il faut s'hydrater

En premier lieu, nous devons paramétrer notre ResultSet pour lui demander d'utiliser notre classe Status afin de représenter un objet de résultat, mais également comment en transformer les informations. Cette méthode de transformation (extraction/injection) d'informations s'appelle l'hydratation, "hydrator" en anglais et fut introduite au sein de Zend Framework 2. Chaque "hydrator" a besoin de 2 méthodes : extract($object) et hydrate(array $data, $object).

Le principe est simple : transformer un objet en un tableau de données (extract) ou l'inverse, alimenter un objet avec un tableau de données (hydrate). ZF2 fournit des hydrators prédéfinis dans le namespace Zend\Stdlib\Hydrator : ClassMethods, ObjectProperty, Reflection, ArraySerializable. Il est possible de définir son propre hydrator avec la classe abstraite AbstractHydrator. Il existe dès lors une implémentation de cette méthode de transformation pour générer les jeux de résultats au sein de ZF2. Rappelez-vous, le type de retour par défaut en utilisant *Zend\Db\TableGateway\AbstractTableGateway* est *Zend\Db\ResultSet\ResultSet*.

Nous allons, nous, utiliser la classe *Zend\Db\ResultSet\HydratatingResultSet*. Pour ce faire, créez une classe ResultSetFactory dans le namespace Application\Model implémentant FactoryInterface. Après examen du constructeur de *Zend\Db\ResultSet\HydratatingResultSet*, voici comment paramétrer cette dernière :

```
[…]
use Zend\Stdlib\Hydrator\ClassMethods as ClassMethodsHydrator;
use Zend\Db\ResultSet\HydratingResultSet;

class ResultSetFactory implements FactoryInterface
{
 public function createService(ServiceLocatorInterface $service
Locator)
 {
   $rowObject = new \Application\Entity\Status();
   return new HydratingResultSet(new ClassMethodsHydrator(),
 $rowObject);
 }
}
```

Déclarez maintenant cette classe comme factory du service Application\Model\ResultSet dans le fichier de configuration du module. Enfin, nous devons préciser à notre service Application\Model\StatusTable d'utiliser ce service ResultSet comme jeu de résultats lors des requêtes, comme deuxième paramètre de son constructeur :

```
[…]
class StatusTableFactory implements FactoryInterface
{
 public function createService(ServiceLocatorInterface $service
Locator)
 {
   $dbAdapter = $serviceLocator->get('Zend\Db\Adapter\Adapter');
   $resultSet   =   $serviceLocator->get('Application\Model\
ResultSet');
   return new StatusTable($dbAdapter, $resultSet);
 }
}
```

Ce service appelé depuis le contrôleur nous renvoie donc un tableau d'objets Application\Entity\Status :

```
public function indexAction()
{
 $service   =   $this->getServiceLocator()->get('Application\
Model\StatusTable');
 $statuses = $service->fetchAll();
 return array('statuses' => $statuses);
}
```

Itérons maintenant sur ce tableau dans notre vue application/status/index.phtml :

```
<table class="table table-striped »>
[…]
<tbody>
 <?php foreach ($this->statuses as $status): ?>
 <tr>
   <td><?php echo $status->getId(); ?></td>
   <td><?php echo $status->getUser(); ?></td>
   <td><?php echo $status->getMessage(); ?></td>
   <td><?php echo (new \DateTime($status->getCreatedAt()))-
>format('d/m/Y H:i'); ?></td>
   <td>
     <a href="<?php echo $this->url('status', array('id' =>
 $status->getId(), 'action' => 'update')); ?>" class="btn
 btn-default"><i class="glyphicon glyphicon-edit"></i></a>
     <a href="<?php echo $this->url('status', array('id' =>
 $status->getId(), 'action' => 'delete')); ?>" class="btn
 btn-danger"><i class="glyphicon glyphicon-trash"></i></a>
   </td>
 </tr>
 <?php endforeach; ?>
</tbody>
</table>
```

Veuillez remplir le formulaire

Maintenant que nous pouvons lister nos statuts, nous devons pouvoir en créer et les modifier. Nous allons pour cela créer des formulaires avec ZF2. Pourquoi ne pas simplement écrire des éléments HTML et traiter la requête POST ? Vous allez découvrir les avantages d'un objet de formulaire (structure, évolutions, validation) et ses relations avec le(s) objet(s) qu'il gère.

Première étape, créer ce formulaire, en ajoutant une classe « Status » dans le dossier src/Application/Form et héritant de Zend\Form\Form. Dans son constructeur, nous allons définir quelques propriétés au formulaire et lui ajouter des éléments :

```
public function __construct()
{
 parent::__construct('status');
 $this->setAttribute('role', 'form');
```

```
  $this->add(array(
    'name' => 'id',
    type' => 'hidden',
  ));

  $this->add(array(
    'name' => 'user',
    'type' => 'text',
    'options' => array(
      'label' => 'User',
      'label_attributes' => array('class' => 'control-label col-sm-2'),
    ),
    'attributes' => array(
      'class' => 'form-control',
      'placeholder' => 'User pseudo',
    ),
  ));
}
```

Comme vous le voyez, la méthode "add" de la class Zend\Form\Form accepte différents types d'arguments : un tableau associatif de configuration, ou bien une instance d'élément déjà défini.
Dans le premier cas, c'est Zend\Form\Factory qui fabrique l'élément à partir des paramètres.

- Le "type" est une valeur littérale d'un Zend\Form\Element,
- Le "name" servira à remplir l'attribut "name" de l'élément et servira de clé pour l'élément dans les valeurs du formulaire,
- Les options, "label" mis à part, sont propres au type d'élément,
- Les attributs servent à définir les attributs HTML de l'élément.

Ajoutez un élément pour le message, un champ de type CSRF (pour s'assurer que la soumission du formulaire vient bien de la page qui l'a créé), ainsi qu'un champ de type "submit". Notre formulaire créé, il faut le déclarer comme service dans le manager "qui va bien", soit le FormElementManager. Dans le fichier module.config.php, ajoutez une entrée dans le tableau de configuration :

```
'form_elements' => array(
  'invokables' => array(
    'form.status' => 'Application\Form\Status',
  ),
),
```

A ce niveau, utiliser le formulaire dans un contrôleur ne devrait plus être un secret pour vous ;)

```
public function indexAction()
{
  $statuses = $this->getStatusTable()->fetchAll();
  $form = $this->getServiceLocator()->get('FormElementManager')
->get('form.status');

  return compact('form', 'statuses');
}
```

Notez le "lazy-loading" de notre objet StatusTable à la première ligne. Voici la méthode appelée :

```
/**
 *
 * @var Application\Model\StatusTable
 */
protected $statusTable;

protected function getStatusTable()
{
  if (null === $this->statusTable) {
    $this->statusTable   =   $this->getServiceLocator()->get
('Application\Model\StatusTable');
  }

  return $this->statusTable;
}
```

Une autre approche aurait été d'injecter cet objet StatusTable dans notre contrôleur. Voyez cet article du blog de Rob Allen à ce sujet : http://bit.ly/1u6XQ36
Nous allons insérer le formulaire au sein de notre tableau de statuts (application/status/index.phtml), en dernière ligne.
Première étape, "préparer" le formulaire.

```
<?php
$form = $this->form;
$form->prepare();
```

Disposant de notre "objet de formulaire", nous utiliserons les aides de vue relatives aux formulaires avant d'en afficher ses éléments :

```
<?php echo $this->form()->openTag($form); ?>
<?php echo $this->formhidden($form->get('id')); ?>
<table class="table table-striped">
[...]
    <tr>
      <td> </td>
      <td><?php echo $this->formtext($form->get('user')); ?></td>
      <td><?php echo $this->formtext($form->get('message')); ?></td>
      <td><?php echo $this->formhidden($form->get('csrf')); ?></td>
      <td><?php echo $this->formsubmit($form->get('submit'));
 ?></td>
    </tr>
  </tbody>
</table>
<?php echo $this->form()->closeTag($form); ?>
```

Chaque « type » de composant possède son aide de vue dont la liste appartient au namespace Zend\Form\View\Helper. La soumission du formulaire (par défaut en POST) se fera (encore par défaut) sur le même couple contrôleur/action que celui qui l'a invoqué, en l'occurrence Application\StatusController::indexAction. Afin de gérer cette soumission, nous n'allons pas comme à l'accoutumée simplement tester si la requête est de type "POST". Car rien de plus agaçant qu'un navigateur qui nous "impose" la re-soumission d'un formulaire sur un refresh de page ou sur un retour sur la page précédente... Nous allons plutôt implémenter le pattern "Post Redirect Get", aka "PRG". Le principe est le suivant (source Wikipedia) : "au lieu de directement retourner une page, la requête POST retourne une en-tête (header) de redirection. Une implémentation correcte des spécifications HTTP 1.1 exige que l'application web retourne une réponse HTTP 303 dans cette situation, assurant ainsi que le navigateur puisse effectuer un rafraîchissement de la page sans répéter la soumission de la requête POST". Une implémentation de ce motif dans notre contrôleur, en utilisant le plugin FlashMessenger :

```
public function indexAction()
{
  if (($prg = $this->prg()) instanceof Response ) {
    return $prg;
```

```
  }

  $statuses = $this->getStatusTable()->fetchAll();
  $form = $this->getServiceLocator()->get('FormElementManager')
->get('form.status');

  if ( false !== $prg ) {
    $form->setData($prg);
    $this->getStatusTable()->saveStatus($form->getData());
    $this->flashMessenger()->addSuccessMessage('User\'s status
 successfully created');
    $this->redirect()->toRoute('status')->setStatusCode(303);
  }

  return compact('form', 'statuses');
}
```

Le premier bloc "if" assure de renvoyer la réponse directement si la page provient d'une autre soumission POST. Puis nous configurons les éléments qui doivent être affichés sur la page. Enfin, le 2ème bloc "if" teste si l'accès à ce contrôleur est une requête POST. Notez, après traitement de cette requête, la redirection via le plugin Redirect, pour laquelle nous devons spécifier le code 303 ("See Other"). Ce code est par défaut 302 ("Moved Temporarily") dans le code du plugin.

Copie à corriger

Maintenant que notre formulaire nous permet de créer des statuts, nous devons nous assurer que les données envoyées sont à la fois sécurisées et intègres. Sécurisées en évitant les injections SQL et autres vilains bouts de code. Intègres en vérifiant que ces données respectent nos règles métiers. ZF2 met à disposition une multitude de classes de filtres et de validation, ainsi qu'un mécanisme pour filtrer et valider un jeu de données avec ces classes qu'est l' "input filter", bien connu des développeurs ZF1. Je sens des dents grincer... En effet, ce système était complexe à configurer, mettre en œuvre et réutiliser au sein de ZF1. C'est pourquoi son modus operandi a été complètement revu et optimisé pour ZF2. Commençons donc par ajouter des filtres et validateurs à notre formulaire. Nous allons (encore, et oui...) utiliser une fabrique pour les déclarer et configurer. Pour que cette fabrique soit prise en compte, il faut que notre classe de formulaire signe l'interface InputFilterProviderInterface. Son nom parle de lui-même... Cette interface déclare la méthode getInputFilterSpecification(), implémentée comme suit :

```
public function getInputFilterSpecification()
{
  return array(
    'user' => array(
      'required' => true,
      'validators' => array(
        array(
          'name' => 'InArray',
          'options' => array(
            'haystack' => array('jguittard', 'jdoe'),
          ),
        ),
      ),
    ),
    'message' => array(
      'required' => true,
      'filters' => array(
                array(
          'name' => 'StringTrim'
        ),
                array(
          'name' => 'StripTags'
        ),
              ),
      'validators' => array(
        array(
          'name' => 'StringLength',
          'options' => array(
            'min' => 3,
            'max' => 140,
          ),
        ),
      ),
    ),
  );
}
```

Nous déclarons 2 entrées pour 2 champs de notre formulaire. "user" ne pourra avoir que pour valeur "jguittard" ou "jdoe". "message" sera une valeur littérale de 3 à 140 caractères, dont nous enlevons les premiers et derniers espaces vides, ainsi que tout tag qui pourrait contenir du code malveillant. A savoir que les valeurs validées sont d'abord filtrées ! N'hésitez pas à tester en changeant ces règles ou en en ajoutant de nouvelles. Vous trouverez filtres et validateurs respectivement dans Zend\Filter et Zend\Validator.
Maintenant que notre formulaire contient des règles de validation, il peut donc être validé. Ajoutons un bloc conditionnel dans notre contrôleur :

```
$form->setData($prg);
if ($form->isValid()) {
  $this->getStatusTable()->saveStatus($form->getData());
  $this->flashMessenger()->addSuccessMessage('User\'s status
successfully created');
  $this->redirect()->toRoute('status')->setStatusCode(303);
}
```

Deux choses importantes à savoir : Il n'est pas possible d'obtenir les données d'un formulaire contenant un inputfilter tant que celui-ci n'a pas été validé. Si le formulaire n'est pas valide, les erreurs commises seront disponibles dans Zend\Form::getMessages() avec les messages par défaut des validateurs et/ou ceux que vous auriez vous-même définis.

Next !

Vous retrouverez l'application entièrement fonctionnelle sur Github à l'adresse : https://github.com/jguittard/zf2-discovery-app.git. J'ai ajouté quelques décorations dans les vues. Toutes vos remarques et questions sont les bienvenues sur https://github.com/jguittard/zf2-discovery-app/issues
Pour continuer votre découverte de ZF2, vous avez à disposition la documentation en ligne sur http://bit.ly/1rDjXdb ou bien, et ce au 1er octobre, mon site http://youzend.com
La communauté est très active sur ce framework open-source qui se démarque des autres par sa robustesse, son respect des standards et sa scalabilité. Avec un minimum de pré-requis en POO, il n'y aura pas de limite à vos développements. De très nombreux modules sont disponibles sur Github et depuis peu sur http://modules.zendframework.com.
Regardez, forkez, modifiez, that's the spirit ! Prochainement : la gestion des évènements dans ZF2, ou comment orchestrer votre application comme un maestro ;-)

Julien **Guittard** - *Architecte / Formateur PHP/Zend* - @julienguittard

Java Virtual Machine : optimisation et réactivité de la JVM 3e partie

Parmi les grands axes d'optimisation d'une application Java, il y a la bonne gestion de sa mémoire et du garbage collector. Pour rappel, lorsque celui-ci s'exécute, toute l'application peut être figée pendant quelques millisecondes (ou plus), le temps de nettoyer les références inutiles. Le second axe d'amélioration est d'optimiser les accès au monde extérieur à la JVM (notamment dans le cadre d'accès à la base de données et des requêtes SQL / HSQL ou autres entrées/sorties).

Test sur l'influence de la taille des méthodes appelantes et appelées sur l'inlining

Voici un autre test pour vérifier la réaction de l'inlining lors de fusion entre de grandes méthodes appelées ou appelantes avec des petites méthodes appelantes ou appelées.

```
public static void step1_grandCallerNotInline1165opcode() {
  double[] tmp = new double[] {
    Double.valueOf("445"), Double.valueOf("445"),
    Double.valueOf("445"), Double.valueOf("445"),
     …. Encore beaucoup …..
    };
    double[] testBidon = Arrays.copyOf(tmp , 100 );
    for (int i = tmp.length; i < 100; i++) {
      testBidon[i] = Double.valueOf("445");
    }
    step1_petitCalleeNotInline52opcode(testBidon);
}

public static void step1_petitCalleeNotInline52opcode(double
[] testBidon) {
  double sum = 0.0d;
  final int taille = 100;
  for (int i = 0; i < 10000; i++) {
    for (int idx = 0; idx < taille; idx++) {
      sum += testBidon[idx];
      sum /= 1.00001d;
    }
  }
}
```

Bilan sur le test de vitesse d'exécution selon la taille des méthodes appelantes et appelées :

Indépendamment des versions de JDK, si la taille de la méthode appelée ou appelante devient trop grande, la vitesse est moindre.

Sur l'ensemble des tests sur les JVM, le résultat est plus homogène lorsqu'on utilise une méthode appelante plus petite que la méthode appelée. Celle-ci peut être plus grande ou de taille quasi équivalente à la première.

Même avec plusieurs essais et une séance de warm up, les résultats sont parfois surprenants, il est donc peu évident de tirer des conclusions définitives. En général, les JDK 32 bits (6 et 7) en mode « -client » sont incapables d'accélérer les petites méthodes appelés ou appelantes.

Test sur l'influence de méthode courte et du full inlining

Le dernier test consiste à vérifier le comportement de la JVM lorsque les méthodes sont suffisamment courtes (environ moins de 6 lignes de code). En effet, les méthodes ne dépassant pas 35 octets de bytecode de code-sont éligibles à un « Full Inlining » qui est une fusion directe des méthodes appelées et appelantes. L'idée du test est de vérifier le comportement de la JVM et sa capacité à fusionner plusieurs petites méthodes et l'influence des paramètres lors des appels de méthodes

```
private static double[] step1_fullInlinableCreate2Values
_Test1() {
   return new double[] {
       Double.valueOf("445"), Double.valueOf("445")};
}

private static void step1_fullInlinedFillArrayInLoop_Test1
(final int nbElementsCrees,
    double[] testBidon, int i) {
  double[] tmpArray = step1_fullInlinableCreate2Values_Test1();
  for (int idx=0; idx < nbElementsCrees; idx++) {
    testBidon[(i*nbElementsCrees)+idx] = tmpArray[idx];
  }
}

/**
 * Test inling avec le corps de la méthode dans la boucle
+ plusieurs paramètres en E/S....
 */
public static void step1_methodeAvecTresPetitInlining_Test1() {
  final int nbElementsCrees = step1_fullInlinableCreate2
Values_Test1().length;
  final int maxElements = 4200; // supérieur à 8000
  double[] testBidon = new double[maxElements*nbElements
Crees];
  for (int i = 0; i < maxElements; i++) {
    step1_fullInlinedFillArrayInLoop_Test1(nbElementsCrees,
testBidon, i);
  }

  double sum = 0.0d;
  final int taille = 100;
  for (int i = 0; i < 10000; i++) {
    for (int idx = 0; idx < taille; idx++) {
      sum += testBidon[idx];
      sum /= 1.00001d;
    }
  }
}
```

Bilan sur le test de vitesse d'exécution selon la taille des méthodes appelantes et appelées :

- De manière générale, à trop découper les méthodes dans lesquelles on a des « for » imbriqués, on a des contre-performances. Exemple : 1

méthode de 30 lignes découpées en 5 méthodes de 6 lignes. On arrive aux limites de l'inlining des JVM actuelles et c'est finalement au moins 20 fois plus lent que des méthodes de taille intermédiaire d'environ 20 à 30 lignes de code.

- Plus on prend une version récente du JDK, plus c'est optimisé pour les petites tailles de méthodes.
- Il y a peu de différences dans l'appel des méthodes si la boucle for est en dehors ou à l'intérieur de la méthode appelée.
- Il y a peu de différences si l'appel d'une méthode utilise le passage par plusieurs paramètres ou le passage d'une référence de type databean. Sauf pour le JDK 7 et 8 en 64 bit où, après warm up, l'utilisation d'un databean ou d'attributs privés semble être meilleurs que l'envoi de tableau en paramètre.

Bilan sur l'ensemble des 3 tests :

- Les options Oracle qui marchent (mais à ne pas désactiver) : désactivation de l'OSR (-XX:-UseOnStackReplacement), désactivation d'une partie de l'inlining (-XX:-Inline), désactivation de l'Escape Analysis (-XX :-DoEscapeAnalysis)
- Sur la JVM Oracle, l'option -XX:+TieredCompilation ne réagit pas pareil lorsqu'on force -client ou -server. Dans la majorité des tests, les résultats sont meilleurs avec –server
- La compilation AoT (via -Xcomp ou IBM -Xjit:count=0) n'est pas la meilleure idée pour de bonnes performances car toutes les méthodes sont compilées à la 1ère exécution. Il faut un temps de warm up beaucoup plus long pour des applications lancées avec cette option. L'option par défaut « -Xmixed » reste le meilleur compromis.
- Si vous travaillez sur une JVM en OS virtualisé, vérifiez bien que le mode SSE4 et 4.2 du CPU soient activés, le risque est de perdre en temps de réponse lors de calculs intensifs.
- Le paramètre pour changer le seuil de 1Ko pour l'inlining semble ne pas fonctionner ou être inefficace car il n'y a aucun changement notable quand on le change.
- **La JVM a besoin d'un "warm up" pour tirer parti de sa quintessence.** C'est à prendre en compte pour des tests de performances Web ou batch et également pour des micro-benchmarks.

Conclusion générale sur l'optimisation de la JVM

Le sujet est vaste et finalement très peu connu et étudié. En effet, c'est d'abord le paramétrage du Garbage Collector et de la mémoire qui seront les premiers facteurs d'accélération d'une application fonctionnant sur la JVM. Dans la majorité des cas, une réduction de la taille de la méthode en la descendant jusqu'à 20 à 30 lignes de code va apporter un gain de vitesse notable sur Java 7 et supérieur. Cependant, pour que l'application soit elle-même plus rapide, il faut que la méthode « refactorée » soit réellement appelée souvent et soit un goulet d'étranglement pour le reste de l'application. Après avoir modifié le code source de quelques projets Open Source (base de données H2 et Derby ainsi que d'autres projets) en ayant découpé quelques grosses méthodes en blocs plus fins, des gains ont été perçus mais pas assez significatifs.

Etait-ce aussi que les tests sur des bases de données mémoires étaient quand même fortement liés aux entrées/sorties ? Je compte poursuivre d'autres tests sur des projets moins dépendants des I/O. Toujours est-il qu'il faut se méfier du code produit par des générateurs de code Java ou de bytecode. Autant le compilateur scalac crée des petites classes et des petites méthodes, autant les jrubyc et le compilateur jsc de Rhino créent sans option une grosse méthode comportant l'ensemble du programme à exécuter : la méthode principale peut vite dépasser le seuil de 2Ko de bytecode et se retrouver beaucoup plus lente que si le code généré était mieux découpé.

Ce constat est le même pour les JSP qui peuvent vite se retrouver avec une méthode « _jspService() » très grosse. Cependant les JSP ne produisent en général qu'un ajout de chaînes de caractères dans un buffer et même en interprété cela reste encore rapide. Selon, les tests effectués, la JVM utilisait moins de CPU sur des JSP dont la taille était « optimisée » pour une compilation assembleur mais les temps de réponse globaux étaient les mêmes qu'avec des JSP plus grosses. Dans l'article « Au cœur des optimisations de code de la JMV », j'ai proposé les premières briques d'un outil capable de calculer automatiquement la taille des méthodes des classes.

Dans une prochaine série d'articles, nous verrons comment tirer parti du "JITspector" : la version améliorée de l'outil qui analyse la taille des méthodes lors de l'intégration continue. Cette 2ème série aura également pour but de montrer la démarche de refactoring de méthodes trop grosses pour ne se concentrer que sur celles les plus sollicitées afin de les optimiser de manière adéquate.

Olivier **Guilloux** - *SQLi*

Tout Programmez!
sur une clé USB

Tous les numéros depuis le n°100.

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

29,90 €*

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

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

Timeline : 1984

Objet : Amstrad invente l'ordinateur plug'n'play

La machine qui a démocratisé l'informatique en Europe, et plus particulièrement en France, est l'Amstrad CPC, un ordinateur conçu pour être peu cher.

A la fin des années 70, les entreprises britanniques spécialisées dans les équipements électroniques grand public adhèrent immédiatement à la nouvelle mode des micro-ordinateurs lancée aux USA. À raison : Sinclair, qui avait inventé la première télévision de poche en 66, lance en 1980 le ZX80, puis le ZX81 et le ZX Spectrum. En 1982, ses bénéfices grimpent à 8,55 millions de livres. La même année, Acorn, société créée par des transfuges de Sinclair, lance le BBC. Comme pour le ZX Spectrum, la demande excède rapidement les possibilités de production. A Brentwood, au Nord-est de Londres, Alan Michael Sugar sent qu'il y a là un nouveau terrain de jeu pour son concept d'appareils tout-en-un. Depuis 15 ans, son entreprise Amstrad connaît un certain succès en vendant des chaînes Hi-fi monoblocs : elles ont l'apparence d'éléments séparés, mais il n'y a bien qu'un seul boîtier réunissant tourne-disque, platine cassette et tuner, avec une seule alimentation. C'est bien plus facile à installer et bien moins cher à fabriquer que les éléments séparés de la concurrence. Sachant, que tous les micros ordinateurs sont alors des appareils à assembler qui ne s'adressent qu'à un public d'afficionados, Alan Sugar a l'idée d'une machine pour la famille, dotée d'une seule prise électrique et livrée avec un écran, pour ne pas solliciter la télévision du salon.

Roland Perry, le créateur des CPC.

Alan Michael Sugar, le patron d'Amstrad.

Des bonnes idées à petit prix

Amstrad n'a pas vraiment le profil d'une entreprise d'innovation, il s'agit plutôt d'un négociant de produits électroniques à très bas prix. Il n'y a pas ici de conviction technique : le processeur Z80 sera choisi parce que c'est celui qui fait tourner le Basic écrit par une entreprise du coin, Locomotive Software, bien moins chère que l'américain Microsoft. La machine est construite sur la base de composants au rabais. Heureusement pour elle, l'ingénieur en chef Roland Perry, embauché pour l'occasion, est un as des bonnes idées à petit prix.

Pour éviter d'investir dans le design d'une carte mère trop compliquée, Roland Perry demande à son copain de fac Mark-Eric Jones de créer une puce simple, la GateArray, qui s'occupe juste de basculer les composants de la machine dans un état ou l'autre. Par exemple, alors que le Z80 ne peut normalement adresser qu'un nombre limité de composants alentour, la Gate Array commute son bus mémoire pour remplacer à la volée les ressources avec lesquelles il échange des données. Effet immédiat : la carte mère est débarrassée de tout un tas de relais électroniques qui auraient coûté une fortune. Bien plus tard, des afficionados se serviront de cette fonction pour faire grimper la mémoire de la machine à 4 Mo (fragmentée en blocs de 16Ko), alors que le Z80 ne peut normalement adresser que 64Ko.

Parmi les composants vidéo les moins chers du marché, Roland Perry choisit le CRTC 6845 parce qu'il produit à l'écran les couleurs les plus éclatantes. La qualité des images est justement ce qui pèche sur les machines de l'époque. Le C64 a 5 gris et 11 nuances fadasses. Le micro d'Amstrad aura donc une palette de 27 couleurs avec des rouges vifs, des verts qui brillent et des bleus profonds. Alan Sugar trouve l'idée excellente : son ordinateur s'appellera donc le CPC - Colour Personal Computer - pour bien montrer qu'il est meilleur que les autres dans ce domaine. Poussant le concept jusque dans ses derniers retranchements, Alan Sugar a l'idée douteuse de doter le premier CPC d'un clavier aux couleurs criardes.

Un affichage plus versatile que la concurrence

A noter que le CRTC 6845 est tellement cheap qu'il ne sait produire que des pixels. Il n'y a ni sprites hardwares pour faire des animations rapides façon C64, ni mode texte pour économiser de la mémoire comme sur Apple II. En revanche, en interagissant, la GateArray et le CRTC donnent des affichages plus versatiles que la concurrence. La machine peut afficher tout à la fois un mode haute résolution (640x200 pixels) en 2 couleurs pour les utilitaires, et un autre en 16 couleurs (160x200 pixels) pour les jeux. Et il n'y a pas ces contraintes de proximité qui obligent les C64 et ZX Spectrum à parsemer leurs écrans d'affreux blocs de pixels plus ou moins monochromes.

Le CPC 464 avec son clavier coloré pour montrer qu'il était le roi de la couleur.

Les éditeurs de jeux et, plus tard, les démomakers, mettront à profit d'autres combinaisons entre la Gate Array et le CRTC : tantôt un écran vertical comme dans les bornes d'arcade (128x256 pixels, dans Arkanoid, notamment), tantôt une zone d'affichage réduite pour accélérer les animations (128x200 pixels), tantôt des modes plein écran (de 192x272 à 768x272 pixels, selon le nombre de couleurs).

Amstrad France, roi du monde

L'Amstrad CPC 464 sort en septembre 1984. En quelques semaines, c'est un succès sans précédent, notamment grâce à la France, où il se vendra les deux tiers des 3 millions de machines écoulées en Europe. Outre l'argument du premier micro tout-en-un qui fait mouche, le prix du CPC complet, 2700 F, est le même que celui du C64... sans écran, ni lecteur de cassettes. Alan Sugar devra aussi la carrière

de ses ordinateurs à Marion Vannier, la dirigeante de sa filiale française. Débauchée de l'importateur Cogel en 1982, Marion Vannier connaît les réseaux de la grande distribution par cœur. Là où les importateurs concurrents adressent des niches, elle convainc La Redoute, puis les hypermarchés. Quand ses homologues européens brûlent les premiers bénéfices dans des BMW de fonction, elle investit dans de la publicité. L'invention de la mascotte crocodile, pour dire qu'Amstrad allait croquer le marché, c'est elle.

Trois CPC successifs en seulement un an

Amstrad fait très rapidement évoluer sa machine. Huit mois après le 464, arrive le CPC 664, avec un lecteur de disquettes à la place de celui à cassettes. Étonnamment, le format choisi est le 3 pouces, dont on sait déjà qu'il est condamné au profit du nouveau standard 3,5 pouces de plus grande capacité. Oui, mais Justement : du fait du déclin des disquettes et des lecteurs 3 pouces, Alan Sugar a pu négocier un très bon prix sur toute la production encore en cours. La presse de l'époque crie à l'erreur stratégique ! En vérité pas du tout : ce choix n'aura aucune incidence sur le succès de la machine, puisqu'il ne sera à aucun moment nécessaire dans la carrière du CPC d'échanger des fichiers avec d'autres plateformes. Et pour Amstrad, ce format à part est même une aubaine commerciale de plus : c'est lui qui vend l'essentiel des disquettes 3 pouces vierges, celles que le public s'arrache pour stocker les copies pirates des jeux. Techniquement, le système de gestion des disquettes, AMSDos, est écrit une fois encore par Locomotive Software et reprend le principe de MS-Dos (noms en 8+3, jokers * et ?, etc.). La puce qui contrôle le lecteur est exactement la même que celle que qui gérait les lecteurs 5,25 pouces dans les premiers IBM PC. C'est la moins chère du marché.

Encore trois mois plus tard, en août 85, Amstrad lance le CPC 6128, un 664 avec un design plus épuré et, surtout, 128Ko de RAM au lieu de 64. En fait, la mémoire supplémentaire ne sert à rien d'autre qu'au marketing : les éditeurs ne

Le CPC 6128, plus sobre, se veut plus professionnel ; il s'agit juste d'entretenir la nouveauté.

En 1990, Amstrad tente d'imposer face à l'Amiga et l'Atari ST un 464+... à cassettes.

l'utilisent jamais pour ne pas être incompatibles avec le parc installé des précédents modèles. Il n'empêche, c'est de nouveau un carton plein : Amstrad France vend 20.000 CPC par mois, 90.000 à Noël. Amstrad Magazine, le mensuel dédié au CPC, écoule 120.000 exemplaires par mois, soit plus que le leader de la presse informatique de l'époque, SVM. Comme Steve Jobs l'a fait aux USA, Alan Sugar vient de démocratiser le micro-ordinateur en Europe.

Mort à cause du désintérêt d'Amstrad

Et puis... plus rien. Le CPC parti sur sa lancée, Alan Sugar passe à autre chose. Surfant sur sa crédibilité fulgurante dans l'informatique, il met sur le marché fin 1985 un ordinateur spécialisé dans la bureautique : le PCW, en version 8256 (256 Ko de RAM) ou 8512 (512 Ko de RAM). Il s'agit en fait d'un CPC dont la carte mère est intégrée dans un moniteur monochrome vert et qu'Amstrad livre avec une imprimante. Le traitement de texte et le tableur sont, comme toujours, écrits par Locomotive Software. Sans la ROM originale, aucun jeu du CPC ne tourne

Le CPC 664, distribué en Allemagne sous la marque Schneider, est juste un 464 avec lecteur de disquettes.

Amstrad choisit le format de disquettes 3 pouces parce que, sur le point de disparaître, il est moins cher.

dessus. Seuls les vieux utilitaires CP/M sur les éternelles disquettes 3 pouces fonctionnent. Il n'empêche, avec un prix quatre fois inférieur à celui du Macintosh et une seule prise à brancher, c'est encore un succès. En 1986, Amstrad se met cette fois-ci au compatible PC. Son PC1512 est deux fois moins cher que les concurrents. Il est le seul compatible vendu en grande surface grâce à Marion Vannier. Et hop, re-carton commercial !

Amstrad ne s'intéresse de nouveau à ses CPC qu'en 1990, lorsqu'il peine à se réinventer encore sur le marché des PC. Mais c'est un intérêt tout relatif : ses 464+ et 6128+, désormais dotés d'une palette de 4096 couleurs et de sprites hardwares grâce à une puce ASIC peu chère, opposent toujours leurs vieilles caractéristiques de 8 bits aux nouveaux Amiga et Atari ST 16 bits. C'est un bide. Au point qu'Amstrad se réoriente ensuite dans les antennes satellites.

Yann **Serra**

Abonnement : Service Abonnements PROGRAMMEZ, 4 Rue de Mouchy, 60438 Noailles Cedex. - Tél. : 01 55 56 70 55 - *abonnements.programmez @groupe-gli.com* - Fax : 01 55 56 70 91 - du lundi au jeudi de 9h30 à 12h30 et de 13h30 à 17h00, le vendredi de 9h00 à 12h00 et de 14h00 à 16h30. **Tarifs** abonnement (magazine seul) : 1 an - 11 numéros France métropolitaine : 49 € - Etudiant : 39 € CEE et Suisse : 55,82 € - Algérie, Maroc, Tunisie : 59,89 € Canada : 68,36 € - Tom : 83,65 € - Dom : 66,82 € Autres pays : nous consulter.
PDF : 30 € (Monde Entier) souscription 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 : S. Saurel, Yann Serra, F. Bordage, J. Chatard

Secrétaire de rédaction : Olivier Pavie
Experts : H. Carnicelli, L. Ellerbach, S. Civetta, F. Dupont, T. Lebrun, J. Guittard, O. Guilloux, J. Antoine, T. Desmas, M. Frappat, F. Hebrard, M. Hubert, C. Villeneuve, S. Goudeau, B. Talmard, E. Briand, G. Egron, J-B Claramonte, P. Gilot, S. Vidouse, J. Devillard, M. Labelle

Crédits couverture : © 07-17-13 © frankpeters © 07-25-13 © Henrik5000
Maquette : Pierre Sandré

Publicité : PC Presse, Tél.: 01 74 70 16 30, Fax : 01 41 38 29 75 pub@programmez.com

Imprimeur : S.A. Corelio Nevada Printing, 30 allée de la recherche, 1070 Bruxelles, Belgique.
Marketing et promotion des ventes : Agence BOCONSEIL - Analyse Media Etude
Directeur : Otto BORSCHA oborscha@boconseilame.fr
Responsable titre : Terry MATTARD
Téléphone : 0967320934

Ce numéro comporte :
1 encart jeté Component Source sur une partie du tirage
1 encart pré-piqué Windows Phone sur la totalité du tirage

Contacts
Rédacteur en chef : ftonic@programmez.com
Rédaction : redaction@programmez.com
Webmaster : webmaster@programmez.com
Publicité : pub@programmez.com
Evenements / agenda : redaction@programmez.com

Dépôt légal : à parution - Commission paritaire : 1215 K 78366 - ISSN : 1627-0908

© NEFER-IT / Programmez, octobre 2014
Toute reproduction intégrale ou partielle est interdite sans accord des auteurs et du directeur de la publication.

“Quand un collègue me refile un projet dont il est content de se débarrasser”

Chaque semaine, de nouvelles aventures ! www.commitstrip.com

Unity : questions – réponses avec David Helgason, le grand patron

Tout d'abord, pouvez-vous nous donner plus de détails sur comment Unity se positionne vis-à-vis des développeurs et des designers ?

David : Unity veut continuer à démocratiser le développement de jeux et réduire les délais et les coûts de développement de nouveaux jeux. La communauté Unity représente plus de 650 000 développeurs actifs (par mois, NDLR) qui utilisent nos services et outils. Nous continuons à ouvrir de nouveaux « chemins » à notre communauté. Nous donnons aux développeurs la possibilité avec peu d'effort de porter leurs jeux sur une multitude de plateformes incluant Windows, OS X, le Web, Android, iOS, Windows Phone 8, Linux. Notre travail, et les partenariats, avec les principaux constructeurs de consoles nous aide à ouvrir là aussi de nouvelles opportunités sur les consoles à la communauté qui avait bien plus de difficultés sur ces terminaux, par le passé. Par exemple, notre collaboration avec Sony permet de développer des outils pour la PS4, la PS Vita, la PSM et le service de cloud gaming du constructeur. Autre partenariat, celui avec Microsoft pour bâtir des outils de développement pour Windows Store et Windows Phone 8. Nous avons aussi dévoilé des outils dédiés à la Xbox One (début août, NDLR). Ces outils sont certifiés Microsoft Games Studio et ID@Xbox. Nous avons aussi le support de la Nintendo Wii U.

Comment un développeur utilise-t-il vos outils ?

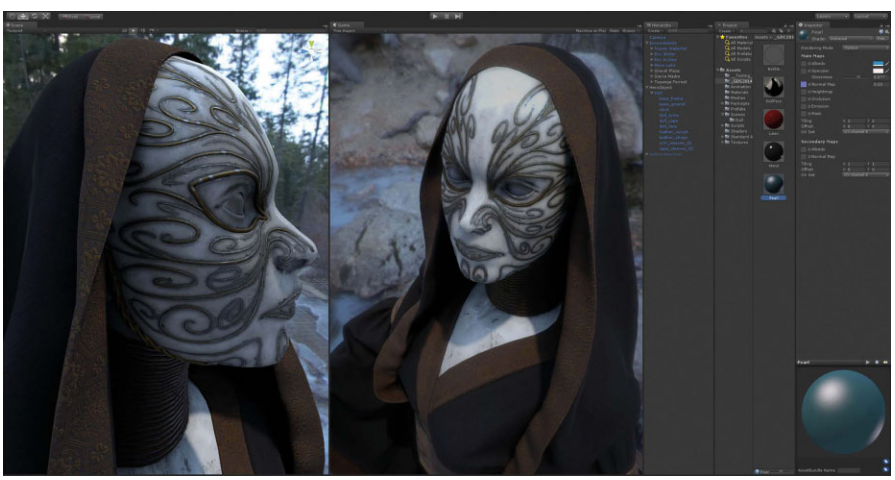
David : rappelons que Unity est un environnement de développement intégré. L'éditeur permet aux développeurs d'intégrer et d'utiliser les différents composants d'un jeu. Au risque d'être simpliste, le développeur (ou le designer) place les différents éléments de création pour créer les niveaux, créer les différentes animations durant l'ensemble du gameplay, créer le système du jeu, l'ergonomie, l'interface et ajouter des scripts pour les contrôles, l'intelligence artificielle, la logique du jeu, etc. L'éditeur est extensible. Il est très simple de développer son extension ou d'en installer. Les extensions peuvent se trouver sur Unity Asset Store. Une fois en place, les développeurs choisissent la plateforme pour générer le projet et générer pour les plateformes cibles. Encore une fois, notre force est la facilité avec laquelle nous construisons le processus. Nous en sommes fiers. Il y a un travail d'optimisation pour supporter les contraintes matérielles de chaque plateforme mais il faut se dire que le développeur peut cibler n'importe quelle plateforme, avec un minimum d'effort technique derrière. Très récemment, nous avons annoncé Unity Cloud Build. Ce nouveau service permet de générer les projets pour les tester. Ainsi, très rapidement, le développeur peut découvrir les problèmes, les bugs et rapidement les corriger.

WebGL, WebCL, OpenCL, Cuda, 64-bit, multi-plateforme... Comment faites-vous pour supporter autant de technologies ?

David : tout d'abord, nous avons la conviction que le développeur doit être libre de ses choix et être capable de cibler telle ou telle plateforme à tout moment. Nous ne savons pas à l'avance quelle technologie ou plateforme le développeur aura besoin. Nous devons donc supporter un très large éventail de technologies. C'est vrai qu'il s'agit d'un défi pour nous, et nos équipes travaillent constamment sur ces supports. Nous évoluons, testons de nombreuses technologies, et nous décidons ensuite de l'opportunité et la nécessité de supporter telle ou telle technologie. Bien entendu, nous scrutons les besoins et les demandes de la communauté.

Unity 5 arrivera dans les prochains mois. Quelles surprises nous préparez-vous ?

David : beaucoup ! Nous avons beaucoup ouvert notre cycle de développement. Nous avons régulièrement posté des articles sur notre blog pour expliquer et présenter les futures fonctions, les nouveaux outils. Il y aura donc un peu moins de surprises que par le passé.



Sur abonnement ou en kiosque

Le magazine des pros de l'IT

Mais aussi sur le web

Ou encore sur votre tablette

L'INFORMATICIEN
www.linformaticien.com

JUSQU'AU 14 DÉCEMBRE 2014

UNE TÉLÉ POUR 1 EURO DE PLUS

INCURVÉ

Télé SAMSUNG

Pour 1 Euro de plus, choisissez :

Télé Samsung écran **Incurvé 121 cm** HD Réf: UE48H6800

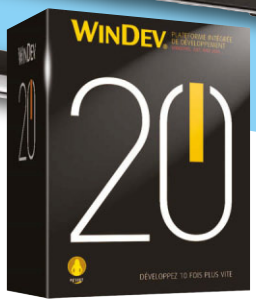
ou

Télé Samsung Affichage **4K ultra HD 127 cm, LED**
Définition d'image fabuleuse: 3.840 x 2.160
Réf: UE50HU6900

ou

Télé Samsung 140 cm 2D/3D HD
Réf: UE55H6400

COMMANDEZ WINDEV 20 (OU WEBDEV OU WINDEV MOBILE) CHEZ PC SOFT ET RECEVEZ UN SUPERBE MATÉRIEL AU CHOIX POUR «1 EURO DE PLUS»



WINDEV®

Et vous pouvez également choisir :

(x2) Nouveau smartphone Samsung Galaxy S5

Configurations détaillées sur www.pcsoft.fr

(x2) Nouvelle tablette Samsung Galaxy Tab S 10,5p

Configurations sur www.pcsoft.fr

Le tout nouveau Samsung Galaxy Alpha

Configurations détaillées sur www.pcsoft.fr

Samsung Galaxy Gear 2 Lite + Smartphone Galaxy Note 4

Configurations sur www.pcsoft.fr

Ou encore un PC portable DEL ou un PC de bureau DELL ou une station de travail DELL

Pour bénéficier de cette offre exceptionnelle, il suffit de commander WINDEV Mobile 20 (ou WINDEV 20, ou WEBDEV 20) chez PC SOFT au tarif catalogue avant le 14 décembre 2014: pour 1 Euro de plus, vous recevrez alors le ou les magnifiques matériels que vous aurez choisis. Offre réservée aux sociétés, administrations, mairies, GIE et professions libérales, en France métropolitaine. **L'offre s'applique sur le tarif catalogue uniquement.** Voir tous les détails et des vidéos sur : **www.pcsoft.fr** ou **appelez-nous**.
Le Logiciel et le matériel peuvent être acquis séparément. Tarif du logiciel au prix catalogue de 1.650 Euros HT (1.973,40 TTC). Merci de vous connecter au site www.pcsoft.fr pour consulter la liste des prix des matériels et les dates de disponibilité. Tarifs modifiables sans préavis.

Descriptif technique complet des matériels sur www.pcsoft.fr

Document non contractuel. Logiciel professionnel. Version 20 annoncée

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

Fournisseur Officiel de la Préparation Olympique

www.pcsoft.fr

Microsoft

Développez des applications universelles. L'expérience ultime.

Lumia 930
DAS(1): 0,60 W/kg

(1) Le DAS (débit d'absorption spécifique) des téléphones mobiles quantifie le niveau d'exposition maximal de l'utilisateur aux ondes électromagnétiques, pour une utilisation à l'oreille. La réglementation française impose que le DAS ne dépasse pas 2 W/kg. Lumia est une marque déposée de Microsoft Corporation ou de ses sociétés affiliées. Microsoft®, Windows® et le logo Windows sont des marques déposées des sociétés du groupe Microsoft Corporation. Les autres noms de produits et de sociétés mentionnés sont des marques déposées ou des noms commerciaux appartenant à leurs propriétaires respectifs. Visuels non contractuels. © 2014 Microsoft Corporation. Tous droits réservés. © 2014 Nokia. Nokia France, une filiale de Microsoft Mobile Oy. Tous droits réservés. R.C.S. Nanterre B 493271522.

Découvrir le Nokia IMAGING SDK en C#

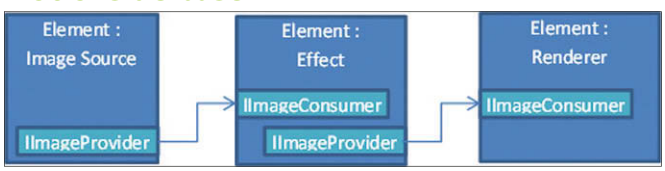
Au fils des années, Nokia a constamment innové pour améliorer la qualité de la caméra de ses smartphones. L'une de ces innovations est l'utilisation du sur-échantillonnage : le device capture une image haute-résolution pour créer une image de 5 mégapixels. Pour cela chaque pixel de l'image finale est calculé à partir de plusieurs pixels de l'image haute résolution. Ceci permet, entre-autres, de diminuer le bruit de capture et des artefacts liés à l'aliasing.

Pour permettre au développeur de tirer profit de cette innovation, Nokia a développé le SDK Imaging. Le but de ce SDK est de fournir un outil pour gérer les images haute-résolution et de traitement d'image et d'effets (plus de 50 filtres disponibles). Tout en minimisant l'utilisation mémoire et le temps de calcul. Avec l'arrivée des applications universelles, le SDK Imaging est maintenant compatible Windows Phone 8 et Windows 8.

Intégration du sdk avec NuGet

Pour utiliser le SDK Imaging dans votre projet, il suffit d'utiliser l'utilitaire Nuget, de chercher le package "Nokia Imaging SDK" et de cliquer sur installer. Le SDK est constitué d'un composant WinRT développé en C++/CX et supporte uniquement les cibles x86 et ARM. Une fois le SDK ajouté à votre projet, vous devez supprimer la compilation pour les cibles x64 et "Any CPU". La cible x86 sert au développement sur l'émulateur Windows Phone et Windows 8 x86. La cible ARM sert au développement sur un téléphone Windows Phone et Windows 8 ARM.
Vous trouverez plus d'informations sur le site de Nokia developper : http://nokia.ly/1uiU8Tr

Notions de base



Le SDK est architecturé sous forme d'une chaîne de traitements. Pour cela, il définit deux interfaces. La première est IImageProvider qui signifie que la classe est une source de pixels. La seconde est IImageConsumer qui signifie que la classe manipule les pixels provenant d'un IImageProvider. Une classe utilisée en début de chaîne implémente uniquement l'interface IImageProvider. Elle est nommée Image Source. Son rôle est de fournir les pixels à traiter par la chaîne de traitement. Pour utiliser les pixels de la caméra, vous pouvez utiliser la classe CameraPreviewImageSource. Pour les pixels d'un fichier image vous pouvez utiliser (en fonction de l'origine du fichier) RandomAccessStreamImageSource, StorageFileImageSource, StreamImageSource ou BufferImageSource. Une classe utilisée en fin de chaîne implémente uniquement l'interface IImageConsumer. Elle est nommée Renderer. Cette classe implémente une méthode RenderAsync qui exécute la chaîne de traitement de manière asynchrone et génère un résultat. Le SDK exploite les composants asynchrones de WinRT et il est préférable de maîtriser les notions await/async. Le WriteableBitmapRenderer sert à récupérer le résultat dans un WriteableBitmap. Le JpegRenderer sert à compresser le résultat dans un fichier jpeg. Entre une Image Source et le Renderer, il est possible d'ajouter des classes implémentant à la fois l'interface IImageConsumer et IImageProvider nommée Effect. Ces classes servent à appliquer des traitements sur les pixels. Comme un Effect implémente les deux interfaces, un Effect peut utiliser un autre Effect en entrée. Le principal Effect du SDK est le FilterEffect qui va appliquer des filtres sur les pixels. Les Filtres du SDK sont des classes implémentant l'interface IFilter. Pour appliquer un filtre, utiliser l'effet FilterEffect en lui fournissant un tableau de Filtres. Lors de l'exécution de la chaîne de traitement, le FilterEffect applique les filtres dans l'ordre du tableau. Voici un code typique d'utilisation du SDK

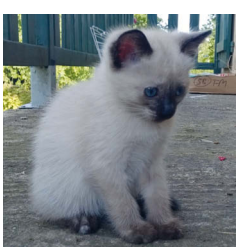
```
async Task<IBuffer> appliquerFiltre(Stream stream)
{
 using (var source = new StreamImageSource(stream))
 using (var filterEffect = new FilterEffect(source) { Filters
 = filters })
 {
   filterEffect.Filters = new IFilter[]
   {
     new AntiqueFilter(),
     new RotationFilter(35.0)
   };
   using (var renderer = new JpegRenderer(filterEffect))
   {
     return await renderer.RenderAsync();
   }
 }
}
```

Ici nous avons codé la chaîne de traitement « source -> filterEffect -> renderer ». Lors de l'appel à RenderAsync, le SDK utilise l'Image Source pour décoder les pixels du fichier images, puis le FilterEffect applique les filtres du tableau. Le JpegRenderer récupère les pixels résultats et les compresse dans un fichier JPEG. Comme le SDK est développé en C++/CX, il utilise des ressources non managées qui ne sont pas gérées par le Garbage Collector. Pour permettre leurs libérations, la plupart des classes du SDK implémentent l'interface IDispose. Une fois l'utilisation d'une classe du SDK finie, il est important de la disposer pour éviter de l'utilisation mémoire inutile. Le langage C# fournit le mot clef using qui s'occupera d'appeler le dispose pour vous.

Recadrer une image

Lors de l'initialisation de la caméra, on commence par demander les résolutions de capture possible. Sur les smartphones Lumia utilisant

Recadrage image origine



Recadrage apres recadrage

un capteur haute-résolution, vous constaterez que l'on ne vous proposera pas la résolution du capteur mais au maximum une image de 5 mégapixels. Ceci est le choix fait par Nokia et Microsoft pour éviter aux applications photo de devoir gérer les images haute-résolution. Toutefois, si vous capturez une image de 5 mégapixels, l'utilisation du sur-échantillonnage est exploitée au niveau du capteur pour fournir une image optimale.
Pour utiliser toute la résolution du capteur, il faut connaître sa taille. Pour cela, vous pouvez utiliser la propriété Microsoft.Phone.Info.DeviceStatus.DeviceName et chercher s'il correspond à un device Lumia particulier. La capture haute résolution permet de prendre une image 16:9 (exploite toute la largeur du capteur) ou 4:3 (exploite toute la hauteur du capteur). Voici une méthode qui vous donnera la résolution à utiliser pour le Lumia 1020, Lumia 1520, Lumia Icon, Lumia 930 et Lumia 830 :

```
Windows.Foundation.Size GetCameraResolution()
{
  var nom = DeviceStatus.DeviceName;
  //lumia 1020
  if (nom.Contains("RM-875") || nom.Contains("RM-876") || nom.
Contains("RM-877"))
  {
    return new Windows.Foundation.Size(7712, 4352); // 16:9 ratio
    //return new Windows.Foundation.Size(7136, 5360); // 4:3 ratio
  }
  else if (
  //Lumia 1520
  nom.Contains("RM-937") || nom.Contains("RM-938") || nom.
Contains("RM-939")
  //lumia 930
  || nom.Contains("RM-1045")
  //lumia icon
  || nom.Contains("RM-927)
  {
    return  new Windows.Foundation.Size(5376, 3024); // 16:9 ratio
    //return  new Windows.Foundation.Size(4992, 3744); // 4:3 ratio
  }
  //Lumia 830
  else if (nom.Contains("RM-983") || nom.Contains("RM-984") || nom.
Contains("RM-985"))
  {
    return  new Windows.Foundation.Size(3840, 2160); // 16:9 ratio
    //return  new Windows.Foundation.Size(3520, 2640); // 4:3 ratio
  }
  return   PhotoCaptureDevice.GetAvailableCaptureResolutions
(SENSOR_LOCATION).First();
}
```

Une fois une image haute-résolution capturée, il ne faut pas la sauvegarder dans la galerie image. Ce type d'image est mal supporté par l'OS. Il est conseillé de générer une image de 5 mégapixels de résolution 3072x1728 pour un facteur 16:9 ou 1936x2592 pour un facteur 4:3. Lorsque vous générez l'image de 5 mégapixels, le SDK exploite le sur-échantillonnage pour créer une image optimale presque identique à une image capturée à cette résolution. La vraie raison de capturer une image haute résolution dans votre application est le recadrage de l'image. Le recadrage permet de modifier le cadrage de la photo ou de zoomer dans l'image. Comme lors de l'Edition d'une image avec l'application Lumia Camera. De plus grâce au sur-échantillonnage, il est possible de simuler un zoom en conservant une très bonne qualité. Par exemple, le Lumia 1020 permet de simuler un zoom allant jusqu'à un facteur 3 avec une faible perte de qualité. Pour recadrer une image avec le SDK il faut utiliser le filtre ReframinfFilter. Le cadre est principalement défini par une zone de l'image (ReframinfFilter.ReframingArea) et son orientation (ReframinfFilter.Angle) :


Chaine de traitement image origine


Chaine de traitement après rotation


Chaine de traitement après filtre antique

```
async Task<IBuffer>
recadrerImage( Stream stream, Size imageResolution, Rectangle
 cadre, double angle)
{
 using (var source = new StreamImageSource(stream))
 using (var filters = new FilterEffect(source))
 {
   filters.Filters = new IFilter[] { new ReframingFilter
(cadre, orientation) };
   using (var renderer = new JpegRenderer(filters))
   {
     renderer.Size = imageResolution;
     return await renderer.RenderAsync();
   }
 }
}
```

L'article « Memory-efficient Navigation in Very High Resolution Images on Windows Phone » (http://nokia.ly/1w6Q1KW) exploite la capacité de recadrage du SDK pour naviguer dans des images de très haute résolution allant jusqu'à 709 Megapixels.

Développer votre propre filtre ou effet

Les développements d'un effet et d'un filtre sont très proches et se résument par la surcharge de la méthode void OnProcess (PixelRegion sourcePixelRegion, PixelRegion targetPixelRegion). La classe PixelRegion en paramètres fournit les informations de l'image source et de l'image de sortie à manipuler :

- ImagePixels : buffer de pixels,
- ImageSize : taille de l'image dans le buffer,
- Pitch : distance entre deux lignes d'image dans le buffer,
- StartIndex : position du premier pixel de l'image dans le buffer.

Le buffer d'une image est un tableau 1D où les pixels sont rangés ligne par ligne les uns après les autres. A l'aide des informations du PixelRegion, on peut convertir la coordonnée 2D {x, y} d'un pixel en index du buffer :

```
int getIndex( PixelRegion pixelRegion, int x, int y)
{
 return pixelRegion.StartIndex + y * pixelRegion.Pitch + x;
}
```

Pour parcourir les pixels de l'image, il faut utiliser une double boucle for :

```
for(int y = 0; y < sourcePixelRegion.ImageSize.Height;++y)
 for(int x = 0; x < sourcePixelRegion.ImageSize.width; ++x)
 {
```

```
    int indexSource = getIndex(sourcePixelRegion, x, y);
    int indexTarget = getIndex(targetPixelRegion, x, y);
    var pixel = sourcePixelRegion.ImagesPixels[indexSource];
    targetPixelRegion.ImagesPixels[indexTarget ] = pixel;
  }
}
```

Sauf cas particulier, le pixel est au format ARGB représenté par un entier de 4 octets. Pour convertir la valeur d'un pixel en valeurs ARGB et inversement il suffit d'utiliser des opérateurs binaires (voir le code de la classe DoubleEffect). Pour implémenter un Effect, il faut hériter de la classe CustomEffectBase et donc réimplémenter la méthode OnProcess. Pour l'utiliser dans la chaîne de traitement, il faut l'ajouter comme un Effect du SDK.

```
public class DoubleEffect : CustomEffectBase
{
  public DoubleEffect(IImageProvider source) : base(source)
  {
  }
  protected override
  void OnProcess(PixelRegion sourcePixelRegion, PixelRegion target
PixelRegion)
  {
    for(int y = 0; y < sourcePixelRegion.ImageSize.Height;++y)
      for(int x = 0; x < sourcePixelRegion.ImageSize.width; ++x)
      {
        int indexSource = getIndex(sourcePixelRegion, x, y);
        int indexTarget = getIndex(targetPixelRegion, x, y);
        uint pixel= sourcePixelRegion.ImagePixels[indexSource];
        //converison de la valeur pixel en valeurs ARGB
        var a = (byte)((pixel >> 24) & 255);
        var r = (byte)((pixel >> 16) & 255);
        var g = (byte)((pixel >> 8) & 255);
        var b = (byte)((pixel) & 255);
        //multiplication des valeurs rgb par 2
        r = (byte)Math.Min(255, r * 2);
        g = (byte)Math.Min(255, g * 2);
        b = (byte)Math.Min(255, b * 2);
        //conversion des valeurs ARGB en valeur pixel
        var newPixel = (uint)(b | (g << 8) | (r << 16) | (a << 24));
        targetPixelRegion.ImagePixels[indexTarget] = newPixel ;
      }
  }
}
```

Dans le cas d'un Effect, lorsque la méthode OnProcess est appelée, le PixelRegion contient tous les pixels de l'image et aucune optimisation mémoire ne sera possible par le SDK. Ceci peut devenir problématique si vous utilisez des images haute-résolution.
Pour implémenter un Filter, il faut hériter de la classe CustomFiltertBase et réimplémenter la méthode OnProcess. Le Filter s'ajoute dans le tableau de filtre d'un FilterEffect. Contrairement à l'Effect, le SDK va pouvoir optimiser l'utilisation mémoire en découpant l'image en tuiles. Pour chaque appel de OnProcess, le PixelRegion contiendra uniquement les pixels d'une tuile. Si pour votre traitement vous avez besoin d'accéder à plusieurs pixels pour calculer le pixel final, vous pouvez spécifier une marge de pixels autour de la tuile dont vous avez besoin. Ces pixels seront accessibles par le buffer du sourcePixelRegion.

Conclusion

Cet article n'a pu présenter qu'une petite partie des possibilités du SDK qui propose, entre-autre du blending, de la segmentation d'image, de l'incrustation d'image, ...
Pour compléter l'article et aller plus loin sur l'utilisation du SDK Nokia Imaging, je vous conseille de regarder les ressources (http://nokia.ly/1vdPu7o), les articles du wiki (http://nokia.ly/1rb9vwb) et le forum d'entraide (http://nokia.ly/1yrqCgW) du site de « Nokia developper ». En complément j'ai commencé à regrouper différents exemples d'utilisation du SDK sur github (http://bit.ly/ZdMWLR).
Si vous deviez choisir un SDK simple de traitement d'image sur Windows et Windows Phone, n'hésitez pas à utiliser le SDK Imaging.

■ Yan **Verdavaine**
Ingénieur expert R&D et Développeur Mobile
http://yan.verdavaine.free.fr/

Création d'un GIF

Les SDK fournissent des outils très intéressants. L'un d'eux est le GifRenderer. Cette classe permet de convertir une liste d'images vers un fichier Gif animé. Ce format est le plus utilisé sur internet pour afficher de petites animations. N'hésitez pas à faire un tour sur http://lesjoiesducode.fr/ pour de superbes exemples. L'utilisation typique du GifRenderer est le suivant :

```
public async Task<IBuffer>
RenderToGifAsync(List<IImageProvider> imageSources)
{
    using (var renderer = new GifRenderer())
    {
        renderer.Sources = imageSources;
        renderer.Duration = 100;
        return await renderer.RenderAsync();
    }
}
```

La liste d'images est donnée sous forme d'une liste de IImageProvider. Elle est donc constituée d' « ImageSource » et de chaîne de traitement terminée par un « Effect ». L'appel à la fonction RenderAsync s'occupera de décoder les « imageSource » et d'exécuter les chaînes de traitements pour les assembler dans l'ordre des IImageProvider de la liste vers une animation Gif. Trois autres propriétés permettent de paramétrer le résultat :

- Duration : le temps en milliseconde que sera affichée chaque image,
- NumberOfAnimationLoops : nombre de fois que l'animation doit boucler,
- Size : taille de l'animation. Le Gif n'est pas adapté pour afficher des images avec une trop grande définition. Il est donc conseillé de limiter la taille de la largeur ou de la hauteur à 600 pixels.

Le code « Image Sequencer » (http://nokia.ly/1yAYjNb) fournit un très bon exemple d'utilisation.

Introduction au SensorCore SDK

Les applications et gadgets santé/fitness, genre Fitbit, ont le vent en poupe. Maintenant, les smartphones s'y mettent aussi. Certains téléphones Android ont des podomètres intégrés. Il y a même sur certains modèles un lecteur de rythme cardiaque. Apple a le HealthKit. Et Microsoft (Nokia) a sorti fin juin une première béta du SensorCore SDK. Cet article va détailler, avec des exemples, les fonctionnalités du SDK. Au moment de la rédaction, la version du SDK était la 0.9.1.3.

SensorCore a 4 fonctionnalités :

- Step Counter : Podomètre, compte les pas de l'utilisateur.
- Activity Monitor : Suivi d'activité. Donne des informations sur l'activité actuelle de l'utilisateur (Marche, course, immobile,...)
- Place Monitor : Suivi de lieu. Identifie les lieux ou l'utilisateur a passé un certain temps. Sa maison, son lieu de travail, son café préféré,....
- Track point monitor : Suivi d'itinéraire. Enregistre des points de suivi le long de l'itinéraire emprunté par l'utilisateur.

Quelques points intéressants sur la technologie SensorCore elle-même :

- Optimisation de l'utilisation de la mémoire et de la batterie: SensorCore tourne en tâche de fond sans utiliser beaucoup de ressources, même si il est activé en permanence.
- Sécurité : Les utilisateurs peuvent choisir à tout moment quelles données peuvent être enregistrées. Elles sont stockées uniquement sur le téléphone, et pas sur un service Cloud. Elles peuvent aussi être effacées.
- Accès immédiat à l'historique : Les applications ont un accès immédiat à 10 jours maximum d'historique des données. Lors de l'installation d'une application celle-ci peut immédiatement utiliser les données déjà disponibles.

Avant de détailler les fonctionnalités commençons par le commencement : l'installation du SDK

Installation

Tout d'abord un point important : SensorCore n'est disponible (au moment de la rédaction) que sur certains téléphones : les Lumia 1520, Icon, 930, 630 et 635. Il faut aussi être en Windows Phone 8.1 ET avoir la mise à jour Cyan. Si vous ne possédez pas un de ces téléphones, vous pouvez toujours utiliser l'émulateur Windows Phone, avec le package de test fourni par Nokia. Commencez par créer un projet Windows Phone 8.1 (le SDK est compatible avec les applications Windows Phone 8.1 Silverlight ou Windows Phone 8.1 XAML). Le SDK est distribué via NuGet : « Lumia SensorCore SDK » (Celui-ci étant actuellement en béta, n'oubliez pas de dire à Nuget d'inclure les prerelease aussi). Le SDK étant en natif, il ne peut pas être compilé en « AnyCPU » et nous devons donc enlever cette configuration. Vous devez maintenant choisir le type de plate-forme à la compilation : x86 si c'est pour être utilisé avec l'émulateur, ARM si c'est pour être déployé sur un téléphone. Voilà, le SDK est installé et prêt à être utilisé. Les classes SensorCore se trouvent dans le namespace Lumia.Sensor. Pour simplifier le développement, Nokia fournit des exemples de helpers pour l'appel des méthodes SensorCore :

```
public static async Task<bool> CallSensorcoreApiAsync(Func
<Task> action)
{
  Exception failure = null;
  try
  {
    await action();
  }
  catch (Exception e)
  {
    failure = e;
  }
  // VÈrifie s'il y a eu une erreur
  if (failure != null)
  {
    MessageDialog dialog;
    switch (SenseHelper.GetSenseError(failure.HResult))
    {
      case SenseError.LocationDisabled:
        // Location est dÈsactivÈ, demandons ‡ l'utilisateur
si il veut l'activer
        dialog = new MessageDialog("Location has been disabled.
Do you want to open Location settings now?", "Information");
        dialog.Commands.Add(new UICommand("Yes", async cmd
=> await SenseHelper.LaunchLocationSettingsAsync()));
        dialog.Commands.Add(new UICommand("No"));
        await dialog.ShowAsync();
        new System.Threading.ManualResetEvent(false).WaitOne(500);
        return false;

      case SenseError.SenseDisabled:
    // Motion est dÈsactivÈ, demandons ‡ l'utilisateur si il veut l'activer
        dialog = new MessageDialog("Motion data has been disabled.
Do you want to open Motion data settings now?", "Information");
        dialog.Commands.Add(new UICommand("Yes", async cmd
=> await SenseHelper.LaunchSenseSettingsAsync()));
        dialog.Commands.Add(new UICommand("No"));

        await dialog.ShowAsync();
        new System.Threading.ManualResetEvent(false).WaitOne(500);
        return false;

      default:
        // Autre erreur
        dialog = new MessageDialog("Failure: " + SenseHelper.
GetSenseError(failure.HResult), "");
        await dialog.ShowAsync();
        return false;
    }
  }
  return true;
}

private static async Task CreateMessageDialog(string message,
string title, string label, UICommandInvokedHandler command, bool no)
{
  var dialog = new MessageDialog(message, title);
  dialog.Commands.Add(new UICommand(label, command));
  if (no) dialog.Commands.Add(new UICommand("No"));
  await dialog.ShowAsync();
}
```

Vous devez lui passer une fonction qui retourne une Task (nécessaire si on veut utiliser le pattern Async/Await). Le helper va s'occuper d'appeler cette fonction et de prévenir l'utilisateur s'il y a eu une erreur ou si les capteurs sont désactivés. Voici un exemple, ou l'on teste si le Step Counter est disponible :

```
await MySensorCoreHelper.CallSensorcoreApiAsync(async () =>
       {
         if (await StepCounter.IsSupportedAsync())
         {
           // StepCounter est disponible
         }
         else
         {
           // StepCounter n'est pas disponible
         }
       });
```

Il est conseillé de désactiver SensorCore lorsque l'on quitte (que ce soit à cause d'une navigation, mise en arrière-plan de l'application,...) la page qui l'utilise. Voilà un exemple ici, pour une application Windows Phone XAML. L'évènement VisibilityChanged est utilisé pour décider si on active ou on désactive SensorCore suivant la visibilité de la page. Pour les applications Windows Phone Silverlight, il faut utiliser OnNavigatedTo et OnNavigatedFrom

```
public sealed partial class MainPage
{
  private IPlaceMonitor _placeMonitor;

  public MainPage()
  {
    InitializeComponent();
    NavigationCacheMode = NavigationCacheMode.Required;
    Window.Current.VisibilityChanged += Current_VisibilityChanged;
  }

  async void Current_VisibilityChanged(object sender, Windows.
UI.Core.VisibilityChangedEventArgs e)
  {
    if (!e.Visible)
    {
      if (_placeMonitor != null) await MySensorCoreHelper.Call
SensorcoreApiAsync(async () => await _placeMonitor.DeactivateAsync());
    }
    else
    {
      if (_placeMonitor != null) await MySensorCoreHelper.Call
SensorcoreApiAsync(async () => await _placeMonitor.ActivateAsync());
    }
  }
}
```

Activity Monitor

Commençons par l'Activity Monitor qui, comme le nom l'indique, suit et enregistre les activités de l'utilisateur. Par activité on parle des mouvements, n'attendez pas que votre Smartphone enregistre que vous avez pris un café à 10 :12 (Grosse déception pour Big Brother). SensorCore détecte 5 types d'activités :

- Walking : L'utilisateur marche.
- Running : L'utilisateur court. Non seulement la vitesse, mais aussi l'intensité du mouvement est utilisée pour déterminer si l'utilisateur marche ou court. Même si la vitesse est semblable, il peut faire la différence entre une marche rapide et une course.
- Stationary : L'utilisateur est stationnaire. Il y a certains mouvements, mais trop petits pour être considérés comme marche ou course (comme lorsque qu'on a une conversation devant la machine à café)
- Idle : L'utilisateur est au repos total. Le téléphone ne détecte aucun mouvement, comme s'il est posé sur une table, par exemple.
- Moving : L'utilisateur est en mouvement, mais SensorCore n'arrive pas à déterminer l'activité exacte.

Il y a deux façons d'interroger Activity Monitor : soit explicitement, soit en souscrivant à un évènement (ReadingChanged) qui se déclenche lors d'un changement d'activité. Pour les lectures explicites, on peut avoir l'activité actuelle (GetCurrentReadingAsync), l'activité à un moment donné (GetActivityAtAsync) ou avoir l'historique des activités pour une période donnée (GetActivityHistoryAsync). L'historique est limité aux 10 derniers jours maximum. Les deux premières méthodes renvoient un objet de type ActivityMonitorReading. Pour l'historique, une liste classée chronologiquement d'ActivityMonitorReading sera renvoyée. Le premier enregistrement est, si disponible, le dernier enregistrement avant le début de la période demandée. Ce qui permet de voir quelle transition d'activité à eu lieu au début de la période. Le type ActivityMonitorReading a deux propriétés : Mode (l'activité) et TimeStamp (la date et l'heure de l'enregistrement). Un exemple de lecture explicite :

```
await MySensorCoreHelper.CallSensorcoreApiAsync(async () =>
{
  if (await ActivityMonitor.IsSupportedAsync())
  {
    _activityMonitor = await ActivityMonitor.GetDefaultAsync();
    var actuelle = await _activityMonitor.GetCurrentReadingAsync();
    Debug.WriteLine("Actuellement : {0} , heure : {1}", actuelle.
Mode, actuelle.Timestamp);

    var semainepassÈe = await _activityMonitor.GetActivityHistory
Async(DateTimeOffset.Now.AddDays(-7), TimeSpan.FromDays(7));
  // Toute la semaine passÈe
    Debug.WriteLine("ActivitÈ semaine passÈe : ");
    foreach (var reading in semainepassÈe)
    {
      Debug.WriteLine("ActivitÈ : {0} , heure : {1}", reading.
Mode, reading.Timestamp);
    }
  }
});
```

Un autre qui utilise ReadingChange :

```
await MySensorCoreHelper.CallSensorcoreApiAsync(async () =>
{
  if (await ActivityMonitor.IsSupportedAsync())
  {
    _activityMonitor = await ActivityMonitor.GetDefaultAsync();
    _activityMonitor.ReadingChanged+= (source, value) => Debug.
WriteLine("Hier : {0} , heure : {1}", value.Mode, value.Timestamp);
  }
});
```

Step Counter

Le podomètre à deux compteurs : un pour la marche, l'autre pour la

course. Le total des pas est donc la somme de ces deux compteurs. Quelques remarques :

- Il y a moyen de réinitialiser ou de désactiver le podomètre. Le nombre de pas renvoyé est donc le nombre depuis la dernière réinitialisation / activation, et pas le nombre absolu (depuis la création du téléphone)
- L'historique est de 10 jours maximum, avec une granularité de 5 minutes.
- Lors de l'initialisation de la lecture, SensorCore a un délai de 6-7 secondes avant d'envoyer les premières lectures (Ceci afin d'éviter les faux positifs). Mais, à moins de problème d'initialisation, les pas lors de ces 6-7 secondes sont comptés. Après ce délai, c'est (quasi) en temps réel.
- La précision du compteur dépend aussi de la façon dont l'utilisateur utilise son téléphone (façon de le tenir, endroit où il met son téléphone) ainsi que de l'environnement (il peut y avoir des faux positifs si l'utilisateur est en vélo, ou passager d'une voiture qui est secouée,...)

Il y a moyen de lire les valeurs actuelles (GetCurrentReadingAsync), celles à un moment donné (GetStepCountAtAsync), ou la liste des valeurs dans un intervalle donné (GetStepCountHistoryAsync). Les deux premières méthodes renvoient un objet de type StepCounterReadin et la dernière une liste de StepCounterReading (Le premier enregistrement est, si disponible, le nombre de pas avant le début de la période, afin de savoir où on en était au début de cette période). Les propriétés du type StepCounterReading sont : WalkingStepCount (le nombre de pas en marchant), WalkTime (le temps de marche), RunningStepCount (le nombre de pas en courant), RunTime (le temps de course) et TimeStamp (la date et l'heure de l'enregistrement). Un exemple :

```
await MySensorCoreHelper.CallSensorcoreApiAsync(async () =>
{
  if (await StepCounter.IsSupportedAsync())
  {
    _stepCounter = await StepCounter.GetDefaultAsync();
    var hier = await _stepCounter.GetStepCountAtAsync(Date
TimeOffset.Now.AddDays(-1)); // Hier a la meme heure
    Debug.WriteLine("Hier : pas course {0} , durÈe course {1},
pas marche {2} , durÈe marche {3}, heure : {4}", hier.Running
StepCount, hier.RunTime, hier.WalkTime, hier.WalkingStepCount,
hier.Timestamp);
  }
});
```

Place Monitor

Le Place Monitor identifie les lieux où l'utilisateur passe un certain temps (au moins 10 minutes). Un lieu sort de la liste s'il n'est plus visité durant 10 jours.. SensorCore travaille en tâche de fond et utilise des méthodes « passives » comme le WiFi ou les antennes-relais pour déterminer la position, ceci afin de réduire l'utilisation des ressources. La précision dépend donc de l'environnement. Le positionnement dans un centre urbain, avec des cellules plus petites et plus de Wifi, sera plus précis qu'en pleine campagne. Si une application utilise le GPS, SensorCore va alors utiliser ses données de manière transparentes (donc sans impact sur la batterie), pour augmenter la précision. Par défaut, un lieu a comme surface un cercle de 200 mètres de rayon, qui peut grandir suivant le comportement de l'utilisateur. Il y a une distance minimale de 500 mètres entre deux lieux. Si moins, SensorCore va (sauf cas exceptionnels) essayer de fusionner les deux. Il y a deux lieux spéciaux : Home (maison) et Work (bureau). SensorCore essaye de trouver ces deux lieux suivant certains critères :

- Si le téléphone est en chargement.
- L'activité de l'utilisateur.
- La façon dont le téléphone est utilisé.
- Si l'utilisateur est à cet endroit durant certaines périodes (la nuit, l'utilisateur est probablement à la maison)

A noter que cette classification prend 2-3 jours et n'est pas fiable à 100% (Si par exemple l'utilisateur a au travail un comportement qui est considéré comme celui de la maison) et peut changer avec le temps (Si l'utilisateur déménage par exemple, mais la détection peut prendre plus de 10 jours). Place Monitor a 3 méthodes pour récupérer les lieux : une pour Work (GetWorkAsync), une pour Home(GetHomeAsync) et une pour tous les lieux connus (GetKnownPlacesAsync). Les deux premières renvoient un objet de type Place et la dernière renvoie une liste de Places classées par fréquentation. Les propriétés du type Place sont : ID (un identifiant), Kind (le type de lieu : Work, Home, KnownPlace), Position (les coordonnées GPS), Radius (le rayon du cercle qui représente ce lieu). Un exemple :

```
await MySensorCoreHelper.CallSensorcoreApiAsync(async () =>
{
  if (await PlaceMonitor.IsSupportedAsync())
  {
    _placeMonitor = await PlaceMonitor.GetDefaultAsync();

    var maison = await _placeMonitor.GetHomeAsync();
    Debug.WriteLine("Maison : position {0} , rayon {1}", maison.
Position, maison.Radius);

    var lieuxconnus = await _placeMonitor.GetKnownPlacesAsync();
    Debug.WriteLine("Lieux connus : ");

    foreach (var reading in lieuxconnus)
    {
      Debug.WriteLine("Type {0}, position {1} , rayon {2}", reading.
Kind, reading.Position, reading.Radius);
    }
  }
});
```

Track Point Monitor

Le Track Point Monitor a pas mal de points communs avec le Place Monitor. La grande différence est qu'ici il n'y a pas de notion de lieux connus. Il s'agit juste d'un suivi des déplacements. SensorCore enregistre la position de l'utilisateur toutes les 5 minutes si celui-ci a bougé d'au moins 500m. La distance entre deux points peut donc beaucoup varier, selon que l'on soit à pied ou en voiture sur l'autoroute, par exemple. Track Point Monitor utilise les mêmes techniques de Place Monitor pour collecter les données (antennes-relais, Wifi, et GPS si celui-ci est utilisé par une application). Ceci et les 5 minutes entre enregistrements font que Track Point n'est pas assez précis pour une application de course à pied par exemple. Cette illustration de Nokia qui montre un suivi par GPS (en rouge) et par Track Point Monitor (en bleu) est parlante : Fig.A. Il y a moyen de lire la position à un moment précis (GetPointAtAsync) ou la liste des positions dans un intervalle de temps (GetTrackPointsAsync). L'historique est de 10 jours. Les objets retournés sont du type TrackPoint pour les deux premières méthodes et une liste de

TrackPoint pour la dernière. Les propriétés du type TrackPoint sont : Position (La position), Radius (le rayon du cercle autour de la position qui représente l'erreur de calcul de la position. Le rayon dépend de la méthode utilisée par SensorCore pour déterminer la position), LengthOfStay (le temps passé par l'utilisateur à cette position), Timestamp (Date et heure de la mesure). Un exemple :

```
await MySensorCoreHelper.CallSensorcoreApiAsync(async () =>
{
 if (await TrackPointMonitor.IsSupportedAsync())
 {
   _pointTracker = await TrackPointMonitor.GetDefaultAsync();
   var hier = await _pointTracker.GetTrackPointsAsync(Date
TimeOffset.Now.Date.AddDays(-1), TimeSpan.FromDays(1));
   Debug.WriteLine("Lieux visitÈs hier : ");
   foreach (var reading in hier)

   {

     Debug.WriteLine("Position {0} , rayon {1}, durÈe {2},
heure {3}", reading.Position, reading.Radius, reading.LengthOf
Stay, reading.Timestamp);
   }
 }
});
```

Simulation

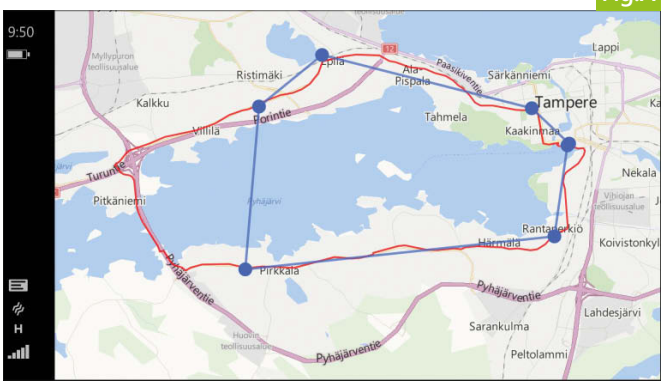
Tester SensorCore n'est pas facile. Il faut un téléphone compatible (l'émulateur Windows Phone ne l'est pas). De plus, il est difficile de demander aux développeurs de courir 2km à chaque test !
Heureusement, Nokia fournit une version simulation de SensorCore. C'est aussi un package Nuget : SensorCore SDK Testing Tools.
Au lieu d'utiliser la classe StepCounter par exemple, vous allez utiliser StepCounterSimulator (dans le namespace Lumia.Sense.Testing). Ces deux classes implémentent l'interface IStepCounter, donc vous pouvez utiliser le même code que ce soit l'original ou la simulation, c'est juste l'instanciation qui va différer. Le principe est le même pour les autres fonctionnalités (Vous avez peut-être remarqué que, dans mon exemple d'activation/désactivation de SensorCore avec VisibilityChanged, les champs privés sont de type IStepCounter, IActivityMonitor,... et non pas StepCounter, ActivityMonitor). Par exemple :

```
await MySensorCoreHelper.CallSensorcoreApiAsync(async () => {

#if DEBUG
        _placeMonitor = await PlaceMonitorSimulator.GetDefaultAsync();
#else
        if (await TrackPointMonitor.IsSupportedAsync())
        {
          _placeMonitor = await PlaceMonitor.GetDefaultAsync();
        }
        else
          return; // en situation rÈelle, prÈvenir l'utilisateur#endif
        var maison = await _placeMonitor.GetHomeAsync();
        Debug.WriteLine("Maison : position {0} , rayon {1}",
maison.Position, maison.Radius);

        var boulot = await _placeMonitor.GetWorkAsync();
        Debug.WriteLine("Boulot : position {0} , rayon {1}",
boulot.Position, boulot.Radius);
```

Fig.A




```
        var lieuxconnus = await _placeMonitor.GetKnownPlacesAsync();
        Debug.WriteLine("Lieux connus : ");
        foreach (var reading in lieuxconnus)
        {
          Debug.WriteLine("Type {0}, position {1} , rayon {2}",
 reading.Kind, reading.Position, reading.Radius);
        }

});
```

Dans cet exemple, si je suis en débug j'instancie le Place Monitor. Sinon, j'utilise le simulateur. Le reste du code ne change pas.
Les simulateurs contiennent des données de test mais vous pouvez aussi charger vos données via des surcharges de GetDefaultAsync. Les données chargées sont tout de suite disponibles et répétées en boucle. Si vous voulez un historique de 4 jours par exemple, vous pouvez dire au simulateur de le charger avec une date antérieure à celle actuelle :

```
var donnees = await SenseRecording.LoadFromFileAsync("donnÈes.
json");
_placeMonitor = await PlaceMonitorSimulator.GetDefaultAsync
(donnÈes,DateTimeOffset.Now.AddDays(-4));
```

Le simulateur va charger les données comme si elles commençaient il y a 4 jours, créant ainsi un historique. Avec un téléphone compatible, vous pouvez enregistrer vos propres données avec SenseRecorder :

```
_placeMonitor = await PlaceMonitor.GetDefaultAsync();
var enregistreur = new SenseRecorder(_placeMonitor);
await enregistreur.StartAsync();

// enregistre les donnees

await enregistreur.StopAsync();
await enregistreur.GetRecording().SaveAsync();
```

SenseRecorder va sauver les données dans le répertoire Documents. Pour les récupérer, connectez votre téléphone à votre PC et naviguer dans le répertoire Phone/Documents

Conclusion

Nokia fournit un petit SDK bien sympathique. Pour le moment seuls les nouveaux téléphones sont compatibles, ce qui fait que les applications SencorCore ont pour le moment un public limité. Mais le succès des Lumia étant grandissant, ce ne sera bientôt plus un problème. Le site Nokia Developer pour SencorCore est ici : http://developer.nokia.com/resources/library/Lumia/sensorcore-sdk.html

Bon amusement avec SensorCore !

Olivier **Matis** - *Mobile Development Engineer chez Arcana Studio* (www.arcanastudio.net) *Microsoft MVP Windows Platform*
@Guruumeditation - www.guruumeditation.net

Les applications universelles

Lors de la Microsoft Build// 2014, une annonce a retenu l'attention des développeurs : les applications universelles (Universal App). La promesse : un seul code va pouvoir créer des applications pour plusieurs plateformes à la fois (Windows, Windows Phone et … Xbox).

L'outil permettant cette prouesse est bien sûr Visual Studio avec une nouvelle version 2013 update 3 apportant de nouveaux templates de projets « universels ». Au niveau des choix techniques, la même latitude que pour une application Windows existe : C#/XAML, JS/HTML, C++/XAML, Direct X ou une composante des trois.
En parallèle, une nouvelle version du Framework Silverlight 8.1 pour Windows Phone a été annoncée. Il va donc y avoir deux types d'applications différentes sur le Store Windows Phone :

- Les applications XAML Silverlight 8.1 (Windows Phone Silverlight dans Visual Studio)
- Les applications Windows Phone Store XAML (Windows Phone dans Visual Studio)

Il va sans dire que Microsoft a unifié la plateforme entre Windows (8.1) et Windows Phone : le même code pourra être exécuté sans modification. Pour résumer et faire simple, vous pouvez dorénavant faire fonctionner une application Windows sur Windows Phone avec quelques adaptations sur les écrans.

Que faire des applications existantes ?

Les applications existantes continuent de fonctionner et d'être supportées par Microsoft à la fois sur Windows et sur Windows Phone. Si vous souhaitez migrer vers une nouvelle version du Framework pour profiter des nouveautés, plusieurs scénarios sont possibles. Il n'y a rien à faire pour une application Windows 8.1 car les applications Windows universelles sont en réalité des applications Windows 8.1. Pour une application Windows Phone 8.0, il y a 2 choix possibles : migrer vers Silverlight 8.1 (assistant automatisé dans Visual Studio) ou migrer votre application vers le Framework universel. Cette migration est manuelle mais reste abordable pour des projets de taille modérée. Bien sûr, il est recommandé de commencer tous les nouveaux développements en utilisant le nouveau modèle d'application universelle.

Comment cela marche ?

Le point de départ est l'assistant de création d'un nouveau projet de Visual Studio qui vous propose de créer une nouvelle application universelle vide ou d'autres projets « universels » avec quelques exemples. Ce template est constitué de 3 projets : MyUniversalApp.WindowsPhone, MyUniversalApp.Windows et MyUniversalApp.Shared, Fig.1. Le projet « Shared » est le plus important des trois. Tout ce qui sera placé à l'intérieur sera automatiquement disponible dans les 2 autres projets. Même l'arborescence de dossiers sera respectée. Par exemple, le fichier App.xaml qu'il contient est le même, exactement le même, pour l'application Windows et Windows Phone. Il va sans dire que dorénavant la bonne pratique consiste donc à mettre le maximum de choses dans ce projet Shared car cela est partagé entre toutes les plateformes. Voici quelques exemples de ressources que vous n'aurez plus à dupliquer dans une solution universelle :

- Les différents dictionnaires de ressources : tous vos pinceaux, les polices, certains styles de contrôles, etc. sont généralement communs au travers des applications,

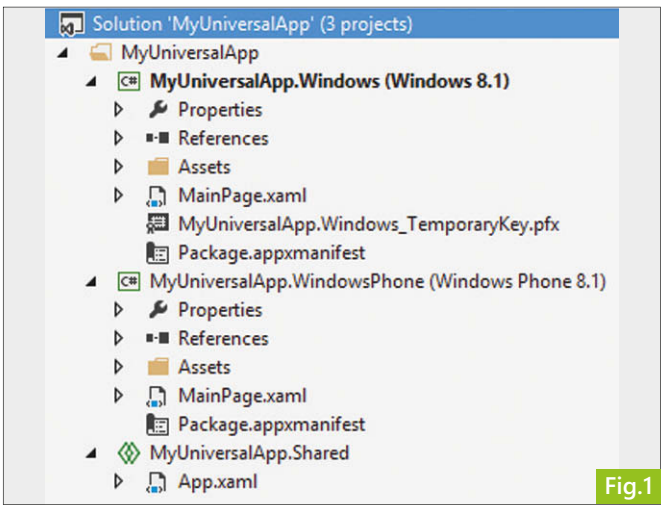


Fig.1

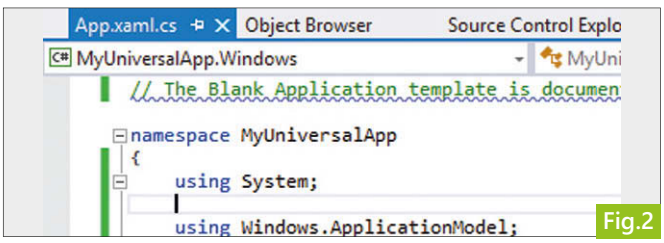


Fig.2

- Les différentes traductions de l'application (fichiers .Resw) : plus qu'une seule traduction à faire !
- Les différentes images,
- Vos différents services applicatifs (stockage de paramètres, dialogues avec l'utilisateur, etc.).

De plus, si vous utilisez le pattern MVVM (ce que je vous recommande chaudement), toute votre logique applicative, vos différentes règles métiers, etc. sont facilement partageables en plaçant tous les ViewModels dans le dossier Shared.

Comment s'y retrouver et différencier les plateformes ?

Lorsque l'on cible plusieurs plateformes avec le même code source il est facile de se perdre et de ne pas savoir dans quel « contexte » nous nous trouvons. Microsoft a résolu cela de façon élégante en ajoutant un context switcher dans Visual Studio. Ce petit menu vous permet à tout instant de savoir sur quelle plateforme vous êtes et de passer d'une plateforme à l'autre en utilisant la liste déroulante. Cela reste d'ailleurs valable dans le designer XAML qui vous affichera la surface de dessin correspondant à la plateforme choisie, Fig.2.
Il reste alors à pouvoir différencier dans le code un comportement d'une plateforme à l'autre. Pour cela, Microsoft propose d'utiliser les constantes et directives de pré-compilation suivantes :

- WINDOWS_PHONE_APP : pour votre application Windows Phone.
- WINDOWS_APP : pour votre application Windows.

Vous trouverez un exemple concret dans la partie "Gestion du bouton back ".

Mon lecteur de RSS universel

Place au code ! Nous allons maintenant réaliser un petit lecteur RSS utilisant le pattern MVVM. La première étape est de créer le projet en utilisant le template « Blank App » de la catégorie « Application

universelle ». Je le nomme « MyUniversalApp » et les 3 projets précédemment discutés sont créés par Visual Studio.

Création du ViewModel principal

Le ViewModel principal va contenir toute la logique de récupération des articles et informations du flux RSS. Il n'y a aucune différence de comportement entre une plateforme et l'autre et je vais donc tout naturellement créer ma classe HomeViewModel dans un dossier « Pages/Home/ » du projet Shared. Le mécanisme de Binding reposant toujours sur l'interface INotifyPropertyChanged, j'ai aussi créé une classe BindableBase pour simplifier son utilisation. Vous trouverez son implémentation dans le code joint à l'article. L'application utilisera les ViewModels en suivant le paradigme ViewFirst et l'initialisation du ViewModel va donc pouvoir être lancée directement depuis le constructeur. Le travail à réaliser est très simple dans notre cas :

- J'indique que je suis en train de travailler en mettant IsLoading à true.
- Je déclenche la récupération du flux en utilisant les APIs du namespace Windows.Web.Syndication.
- Une fois les informations reçues, je remplis quelques propriétés d'affichage.

```
// Lancement de l'initialisation
public HomeViewModel() { InitAsync(); }
private async Task InitAsync()
{
  IsLoading = true;

  try
  {
    PageTitle = string.Empty;

    //Récupération du flux RSS
    var syndicClient = new SyndicationClient();
    var feed = await syndicClient.RetrieveFeedAsync(
      new Uri("http://blogs.infinitesquare.com/api/rss"));

    FeedItems = feed.Items.ToList();

    //Affichage des informations sur le flux
    PageTitle = string.Format("{0} : {1} derniers éléments.",
           feed.Title.Text, FeedItems.Count);
    LastUpdate = feed.LastUpdatedTime.ToString("F");
  }
  catch (Exception exception)
  {
    //TODO meilleure gestion des erreurs
    new MessageDialog("Une erreur s'est produite... :("
        + Environment.NewLine + exception).ShowAsync();
  }
  finally
  {
    IsLoading = false;
  }
}
```

Partage de ressources

Avant de construire la vue principale, je vais tout d'abord créer un fichier de ressources communes à mes deux applications. Pour cela, j'ajoute un fichier « SharedResources » dans le dossier « Assets/Resources » du projet Shared. J'y ajoute alors 2 éléments à titre d'exemples : une couleur d'accentuation et un convertisseur utilisé pour afficher les auteurs d'un article de blog.

```
<ResourceDictionary
  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">
  <SolidColorBrush x:Key="MainColorBrush"
           Color="#7171FF" />
  <common:AuthorsToStringConverter x:Key="AuthorsToStringConverter"/>
</ResourceDictionary>
```

Il ne me reste plus alors qu'à le référencer dans mon fichier App.xaml pour rendre ses ressources disponibles partout.

```
<Application.Resources>
  <ResourceDictionary>
    <ResourceDictionary.MergedDictionaries>
      <ResourceDictionary
        Source="/Assets/Resources/SharedResources.xaml" />
    </ResourceDictionary.MergedDictionaries>
  </ResourceDictionary>
</Application.Resources>
```

Création de la vue principale

Construisons maintenant la vue principale qui va afficher simplement la liste des éléments du flux en utilisant les propriétés du ViewModel. Pour cela j'ajoute un nouvel élément de type « page blanche » « HomePage.xaml » dans le dossier « Pages/Home ». Dans cette vue je vais placer une grille avec deux lignes. La première ligne affichera le titre de la page et la seconde contiendra un élément GridView. Ici, la magie du Framework d'application universelle va opérer car on se trouve dans le projet Shared. Sur Windows, la GridView disposera les éléments les uns à côté des autres (en utilisant un panel ItemWrapGrid) alors que sur Windows Phone elle disposera les éléments les uns au dessus des autres (en utilisant un VirtualizingStackPanel) convenant mieux à un affichage portrait. Automatiquement, l'affichage sera donc adapté au device où la GridView est utilisée ! De la même façon, j'ajoute une AppBar avec un bouton rafraichir à la page et son design sera adapté à la plateforme !

```
<Page.DataContext>
  <home:HomeViewModel />
</Page.DataContext>
<Grid>
  <Grid.RowDefinitions>
    <RowDefinition Height="Auto" />
    <RowDefinition Height="*" />
  </Grid.RowDefinitions>
  <TextBlock Text="{Binding PageTitle}"
         Style="{StaticResource PageTitleStyle}" />

  <GridView Grid.Row="1"
        Style="{StaticResource GridViewStyle}"
        ItemsSource="{Binding FeedItems}">
    <GridView.ItemTemplate>
      <DataTemplate>
        <!--Contenu du dataTemplate-->
      </DataTemplate>
    </GridView.ItemTemplate>
  </GridView>
</Grid>
<Page.BottomAppBar>
  <CommandBar >
```

```
    <CommandBar.PrimaryCommands>
      <AppBarButton Icon="Refresh" Click="OnRefreshClick"/>
    </CommandBar.PrimaryCommands>
  </CommandBar>
</Page.BottomAppBar>
```

Adaptation aux différents devices

Finalement, il reste quelques problématiques : les marges et les différentes tailles de polices ne sont pas forcément adaptées. Pour résoudre ce problème nous allons définir les styles « `PageTitleStyle`» et « `GridViewStyle` » dans des fichiers spécifiques à chaque plateforme. La façon la plus simple et rapide de faire cela est de créer un fichier « PlatformSpecificResources.xaml » dans le dossier « Assets/Resources » de chacun de mes projets spécifiques. Dans ce fichier de ressources je vais pouvoir personnaliser l'affichage que je souhaite pour chaque élément en fonction de la plateforme. Par exemple, sur Windows je vais ajouter une marge gauche de 120px sur les titres de pages.

```
<Style x:Key="PageTitleStyle"
       TargetType="TextBlock"
       BasedOn="{StaticResource HeaderTextBlockStyle}">
  <Setter Property="Foreground"
       Value="{StaticResource MainColorBrush}" />
  <Setter Property="Margin"
       Value="120,40,0,40" />
</Style>
```

Ensuite, je vais indiquer à l'application de charger ce fichier de ressources à son lancement en le déclarant dans le fichier App.xaml. L'application sélectionnera alors automatiquement le bon fichier à utiliser en fonction de la plateforme. Comme j'utilise les mêmes clefs pour définir les styles sur chacune des plateformes, mon affichage sera personnalisé !

```
<Application.Resources>
  <ResourceDictionary>
    <ResourceDictionary.MergedDictionaries>
      <ResourceDictionary  Source="/Assets/Resources/Shared
Resources.xaml" />
      <ResourceDictionary Source="/Assets/Resources/Platform
SpecificResources.xaml" />
    </ResourceDictionary.MergedDictionaries>
  </ResourceDictionary>
</Application.Resources>
```

Affichage du contenu d'un article

Pour afficher le contenu d'un article, je vais créer une nouvelle page « ArticleDisplayPage.xaml » dans le dossier « Pages/ArticleDisplay » de mon projet Shared. Cette page va uniquement contenir un contrôle WebView pour afficher l'article. Je peux alors m'abonner au clic sur un élément de la GridView pour déclencher une navigation. Etant donné que la page est connue sur les deux plateformes et que le système de navigation est unifié, cela est un jeu d'enfant :

```
void ListViewBase_OnItemClick(object sender, ItemClickEventArgs e)
{
  string itemUrl = ((SyndicationItem)e.ClickedItem).Id;
  App.RootFrame.Navigate(typeof(ArticleDisplayPage), itemUrl);
}
```

Ensuite, au chargement de ma page « ArticleDisplayPage », je n'ai qu'à déclencher le chargement de la WebView :

```
protected override void OnNavigatedTo(NavigationEventArgs e)
{
  base.OnNavigatedTo(e);
  if (e.NavigationMode == NavigationMode.New)
  {
    string uri = e.Parameter.ToString();
    WebView.Navigate(new Uri(uri, UriKind.Absolute));
  }
}
```

Gestion du bouton back

Le bouton back existe physiquement sur Windows Phone mais pas sur Windows. Pour répondre à cette problématique, j'utilise le même système de style que précédemment pour masquer le bouton sur Windows Phone en utilisant la propriété Visibility. De plus, le bouton back physique n'est pas pris en compte par défaut dans l'application et a donc comme comportement de quitter celle-ci.
Pour éviter ce comportement pour le moins gênant, il suffit de s'abonner dans le constructeur de la page à l'événement `HardwareButtons.BackPressed` et de faire ce traitement :

```
    public ArticleDisplayPage()
    {
      this.InitializeComponent();
#if WINDOWS_PHONE_APP
      HardwareButtons.BackPressed += (_, b) =>
        {
          GoBack();
          b.Handled = true;
        };
#endif
  }
```

Travail terminé

Et voilà ! En quelques lignes de code nous avons un lecteur de flux RSS qui fonctionne sur Windows mais aussi sur Windows Phone.

D'autres améliorations sur le Store

Une des nouveautés non liée au code des applications universelles est le partage d'identité (facultatif) entre les applications des différentes plateformes. En activant cette fonctionnalité lors de la préparation du package pour la soumission sur le Store, vous profiterez de ces avantages :

- Les settings en mode « roaming/synchronisés » seront communs entre toutes vos applications.
- Une application achetée sur un Store l'est aussi sur les autres.
- Les produits d'achats « in-app » sont partagés entre les plateformes.

Pour aller plus loin

Nous avons fait un bon tour d'horizon sur les applications universelles et afin d'aller plus loin je vous conseille ces quelques ressources :

- L'extension de Visual Studio « Shared Project Reference Manager" vous permet de créer un projet "Shared" et de l'utiliser pour n'importe quel type de projet (même traditionnel).
- Le site code.msdn.com qui contient des dizaines d'exemples d'applications universelles.
- Le livre « MVVM de la découverte à la maitrise » afin d'approfondir le pattern MVVM.

Jonathan ANTOINE - *Infinite Square*

Programme accélérateur Windows

Microsoft France a mis en place un programme d'accompagnement gratuit pour aider les développeurs d'applications Windows et Windows Phone : l'Accélérateur Windows.

Avec plus de 1 000 développeurs français inscrits depuis sa création, le programme Accélérateur Windows permet de bénéficier d'une aide personnalisée entièrement gratuite avec pour seule condition d'avoir un projet d'application à destination des stores Windows et/ou Windows Phone.
Il permet donc d'avoir un suivi de son projet d'application via un contact chez Microsoft France, de recevoir du support technique gratuit, et de bénéficier de soutien en termes de visibilité une fois l'application publiée sur le Store.

Des ressources

Le programme vous offre l'inscription au Windows Dev Center permettant de publier son appli sur les stores Windows et Windows Phone, vous met à disposition une tablette ou un Windows Phone pour tester votre application. Il vous permet également de retrouver des tutoriaux adaptés, d'être mis au courant régulièrement des derniers événements/concours et enfin d'assister tous les mois à une formation à distance sur un sujet de développement spécifique (ex : monétisation, gaming, design, Azure Mobile Services, etc.).

Du coaching technique gratuit

Vous disposez d'un support technique à distance gratuit permettant d'avoir des réponses rapides et précises sur un sujet ou un bug. L'Accélérateur Windows permet également d'avoir une entrevue personnalisée avec un expert technique Microsoft France pour échanger sur son application, ou encore d'une revue à distance pour recevoir des axes d'amélioration et des conseils sur son application.

De la visibilité pour l'application

Enfin, l'Accélérateur Windows offre aux développeurs la possibilité d'une mise en avant sur les Stores Windows et Windows Phone (sous réserve de validation de l'équipe du store), une mise en avant en appli du jour sur Presse-Citron, sur les blogs et réseaux sociaux Microsoft.

Témoignages de développeurs

Christophe Delouche, développeur du jeu Fruity Mix sur Windows Phone, décrit son expérience au sein du programme : *« Je fais partie de l' « Accélérateur Windows » qui est un accompagnement mis en place par Microsoft. Au travers de ce programme, j'ai rencontré des personnes formidables, réactives et très compétentes. J'ai donc été soutenu pendant toute la durée du projet, aussi bien au niveau du coaching, qu'au niveau technique, et même pour la visibilité. »*
De même, Vincent Monteil, étudiant et développeur du jeu Hundredious sur Windows Phone explique que *« Le programme Accélérateur Microsoft est une excellente opportunité pour les personnes qui hésitent à franchir le pas, avec un accès gratuit aux ressources nécessaires à la publication d'une application, et un support très personnalisé et de qualité qui permettra de se faire à la logique de codage sous Windows. »*

Plus d'infos et inscriptions sur
www.accelerateurwindows.com

■ Anthony **Virapin** – *Responsable relations développeurs mobiles*
Microsoft France

SOURCE DE « DEVELOPER HEROES »

Ayant pour objectif d'optimiser la qualité des applications sur le store Windows, l'Accélérateur Windows met donc à disposition tout ce qui peut contribuer à aider au succès du développeur sur la plateforme Windows. Ainsi, de nombreux développeurs ayant bénéficié de l'aide du programme ont pu être mis en avant lors de l'opération « Developer heroes » (appheroes.windows.fr) lancée par Microsoft France et contribuant à renforcer leur visibilité. En continuité avec cette initiative, Microsoft France vient également de lancer un programme d'aide pour tous les projets cloud : la Pépinière Microsoft Azure.

Supplément Windows Phone de Programmez! 179. Octobre 2014. Coordination : Olivier Lovisa (Microsoft France). Maquette : Pierre Sandré.
Directeur de la publication : François Tonic - Editeur : NEFER-IT / Brétigny sur Orge. Photos et illustrations tous droits réservés - Imprimé en Belgique.