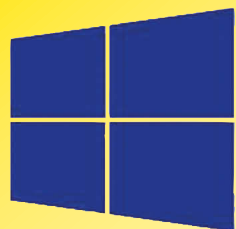


PROgrammez !

Mensuel n°166 - Septembre 2013

www.programmez.com

Bientôt sur vos PC


Windows 8.1

*Une mise à jour majeure, attendue pour octobre,
les nouveautés dévoilées*

Visual Studio 2013

JSON jQuery de A à Z

*Les deux piliers du
développement web*



Je débute avec iOS

Base de données
MariaDB ou MySQL ?

Office 2013
Créer des
apps Office

Matériel
La config idéale
du développeur

© istockphoto-calorftime

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 - 166 - F: 5,95 € - RD



HTML5+jQuery

Toute App, Tout Navigateur, Toute Plateforme, Tout Appareil



IGNITEUITM
INFRAGISTICS JQUERY CONTROLS



Téléchargez Votre Version d'Evaluation Gratuite
www.infragistics.com/telechargez



INFRAGISTICS[®]
DESIGN / DEVELOP / EXPERIENCE

Infragistics Ventes France  0800 667 307 • Infragistics Ventes Europe +44 (0) 800 298 9055

Copyright 1996-2013 Infragistics, Inc. All rights reserved. Infragistics and NetAdvantage are registered trademarks of Infragistics, Inc. The Infragistics logo is a trademark of Infragistics, Inc.

Nicolas Sadirac



« Nous fabriquons les créateurs du monde de demain »

Le numérique a apporté la dématérialisation et une (illusion ?) de gratuité. Pouvait-on rendre gratuite également l'école IT, la fabrique de ceux qui fabriquent la dématérialisation ? Nicolas Sadirac, à la tête de la nouvelle école 42, voulue et financée par Xavier Niel, fondateur de Free, nous a décrit, avec enthousiasme, sa vision iconoclaste de la formation et des métiers de l'IT. Son ambition : donner naissance aux créateurs du monde de demain.

Programmez : Des études gratuites peuvent paraître « au rabais » ?

Nicolas Sadirac : Après une première sélection, nous avons déjà reçu 70 000 candidatures. Après les tests, nous aurons 20 000 candidats « sérieux », nous en recevons en entretien 2500 pour retenir environ 800 inscrits*. Un candidat sur vingt, c'est une sélection délicate. Mais nous aurons ainsi des étudiants plus « forts » que ceux d'Epitech ou Epita.



“ Une formation peer-to-peer ”

sa-demande »...

L'économie se décale : la capacité de fabriquer diminue, on a besoin de gens avec une capacité d'invention. Mais c'est un processus collectif, pas individuel. C'est une transformation du mode de fonctionnement de l'entreprise. Les entreprises à voie hiérarchique traditionnelle vont disparaître.

Programmez : On dit qu'en contrepartie de la gratuité des études, vous faites des économies sur le nombre d'enseignants ?

N.S. : Je n'utiliserais pas le terme d' "enseignant" mais plutôt « encadrant » ou « animateur » ... Ils sont une quarantaine, oui, je les 'économise'. Il n'y a pas de « cours »... nous sommes dans une école moderne ! De notre point de vue, un cours, cela formate, apporte une discipline. Dans un « monde-usine », on a besoin de gens interchangeables, productifs, normalisés. Dans un monde d'idées, il faut des gens tous différents !

On veut aller plus loin dans la « formation peer-to-peer ». L'étudiant apprend en groupe, à travers les autres. Nous dépassons le paradigme de la transmission des connaissances, pour créer une nouvelle logique d'enseignement des connaissances au travers d'une co-créativité, une créativité collective.

On travaille exclusivement par projet. L'étudiant doit être inventif.

On fait en 3 ans le même programme que l'Epitech en 5 ans ! Parce que nos étudiants sont meilleurs : on les a davantage sélectionnés en amont. Dans une école traditionnelle,

60% des ressources pédagogiques s'occupent de la queue du peloton. Nous recherchons des gens brillants, nous ne défendons pas l'égalitarisme, mais un système méritocratique.

Programmez : sans diplômes, vous ne formez pas des ingénieurs mais des techniciens ?

N.S. : Non, ils ne seront pas techniciens, je les appelle des « technoformeurs », ce sont les créateurs du monde de demain, les catalyseurs de transformation. En plus de la technologie, ils savent concevoir des produits, ils sont donc créateurs de monde. Leur valeur ajoutée est cette capacité d'innovation. Comme par exemple concevoir les « technowares » de demain, comme ces lunettes qui permettent de voir les pièces d'une future maison.

Si vous voulez devenir administrateur système d'une grande banque, ne venez pas chez nous. Nous formons ceux qui vont travailler dans les services Innovation, contrairement au modèle de l'informaticien-ingénieur : « je sais-tout-faire, j'écoute-le-client, je-restitue-

De mon point de vue, les gens sans diplôme sont plus courageux. C'est un choix, on nous a proposé de délivrer l'équivalent de Bac+5, un master informatique, en partenariat avec une école informatique ou une université, nous avons refusé. Nous ne croyons pas aux normes !

Programmez : concrètement, sans diplôme, ces étudiants n'auront-ils pas des difficultés à trouver un emploi et un salaire équivalent ?

N.S. : les entreprises recrutent sur la compétence, pas sur les diplômes. Les diplômés seront de moins en moins valorisés. Chez Free, la moitié des informaticiens n'ont pas de diplôme et pourtant ils gagnent 100K €. Nous sortons du paradigme du diplôme pour aller vers celui du « book » : un programmeur montre l'appli qu'il a réalisé : « je sais faire ». Nos étudiants montreront le portfolio des 100 à 120 projets qu'ils auront réalisés !

* L'entretien a été réalisé en juillet.

Propos recueillis par Jean Kaminsky



BUZZWORD

Nicolas Sadirac :
« Nous fabriquons les créateurs
du monde de demain »3

ACTUS

Actu en bref6

RESSOURCES

Événements, agenda10

OUTILS

MySQL vs MariaDB12

DEVELOPPEUR DU MOIS

Benjamin Launay : « il n'y a pas que le code,
il y a aussi le design ! »18

VEILLE TECHNO

Retour sur la conférence //build/



Quoi de neuf pour les développeurs
avec Windows 8.121

Windows 8.1 : du neuf pour le C++26

Windows 8.1 et XAML27

Windows 8.1 : WinJS 2.0 arrive29

Les grandes nouveautés de Visual Studio 2013
et Team Foundation Server 201331

DOSSIER WEBMASTER

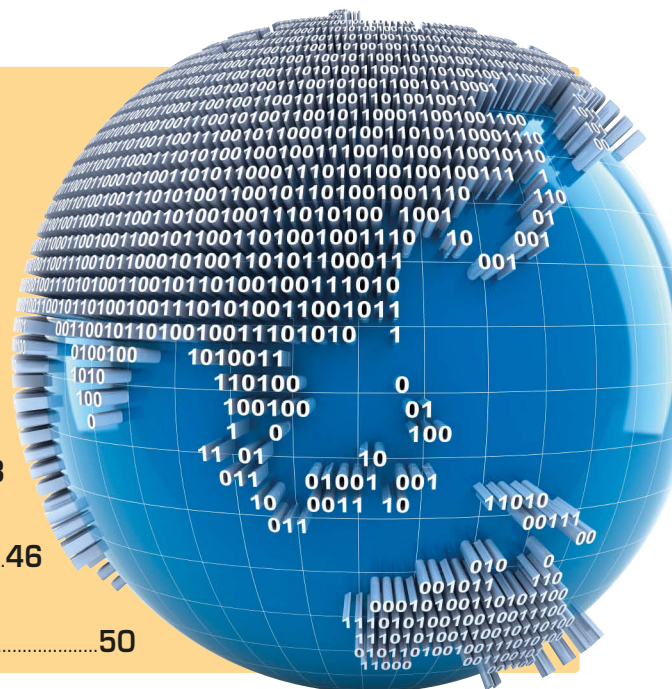
JSON, JQuery : les deux piliers du développement web

jQuery/JSON : simple,
puissant39

JSON, un format
devenu incontournable ?43

Pourquoi JQuery
a-t-il autant de succès ?46

JQuery : 10 bonnes pratiques
à connaître absolument !50



A lire dans le prochain numéro

N°167-octobre 2013, à paraître le 28 septembre 2013

Utilisez et intégrez
Google Maps

Webmaster
HTML 5
par la pratique

Jeux
Unity3D
Le meilleur moteur actuel !

36



MATERIEL

La configuration idéale du développeur36

JE DEBUTE avec...

Introduction au formulaire et persistance avec Symfony 253

CAS D'USAGE

OAuth : exemple d'implémentation en C#56

Office 2013 : comment créer des Apps ?59

PRATIQUE

SQLite : une base de données embarquée64

SPÉCIAL iOS



67

Bien démarrer sur iOS67

Améliorer son environnement de développement iOS71

Xcode : ma première application avec iOS, (1^{re} partie)73

ADMIN

Les expressions régulières au service de l'administrateur Linux79



Je code comme j'écris

Depuis les premières machines, une question revient sans cesse : comment programmer une machine, quel langage créer et utiliser ? Nous pouvons faire remonter le débat à la machine de Pascal (17^e siècle), voire, dès la machine d'Anticythère (antiquité). Car, il existe une constante : pour fonctionner et réaliser les opérations, une machine doit être programmée. Oui, mais avec quoi ?

Des chercheurs du Computer Science and Artificial Intelligence Laboratory du fameux MIT ont récemment publié des articles sur le thème : comment utiliser le langage courant (bref le langage de tous les jours) comme langage de programmation. Le but est que des non-développeurs puissent développer sans maîtriser la technique, sans connaître la « logique » informatique. Un rêve bientôt réalité ? Pas si sûr !

Nous avons donc décidé de lire ces articles dans notre petit jardin, cet été, à l'ombre. Le titre est déjà tout un programme, « From natural language specifications to program input parsers ». Je dis, « chouette, je vais pouvoir écrire des applications comme j'écris... ». Mes petits bytes étaient tout excités. Mais ils allèrent rapidement se remettre à l'assembleur ou au C++.

En réalité, les chercheurs utilisent bien un texte courant en anglais pour créer un code source. Grosso modo, il y a trois étapes :

- 1 Le texte en langage courant décrit le programme et contient des « mots-clés », des expressions particulières, par exemple : intégrer N, 4 nombres réels, etc.
- 2 Ce texte est analysé. Un arbre syntaxique avec les spécifications du texte est généré. Il faut le voir comme une représentation ordonnée et schématisée du programme tel qu'il est décrit dans le texte courant ;
- 3 Le format formel sous la forme d'un « code » est basé sur une grammaire définie qui transcrit l'arbre intermédiaire en « code » ;

Le point 1 est très important à comprendre. Le texte courant n'est pas un texte banal. Pour pouvoir générer un arbre puis le code, le texte de départ doit être structuré et inclure des mots précis, faisant partie d'une grammaire, d'un lexique, reconnus par les parsers... Les chercheurs évoquent un taux de réussite de 72,5 % dans l'interprétation du parser C++. Bref, impossible d'écrire n'importe quoi pour générer une application.

Les chercheurs eux-mêmes avouent les limites de cette approche. Et elles sont nombreuses : coût financier des parsers, erreurs d'interprétation, incompréhension des règles pour celui qui écrit le texte, comment faire du bidirectionnel quand on modifie le code ou l'arbre, etc. D'autre part, le modèle actuel fonctionne uniquement avec des textes anglais, il faudra redévelopper toute la logique, les parsers, les grammaires pour les autres langues. Mais imaginez les erreurs de frappe entre « est » et « et » : le résultat final est totalement faussé, voire, impossible à générer.

A court terme, cette programmation n'a pas d'intérêt, en raison des limites de l'expérimentation. Il existe des langages de haut niveau utilisant des mots clés du langage courant, tels que VLangage ou feu HyperTalk. Mais avec des processeurs toujours plus puissants et des modèles parallèles et capables d'interpréter et de comprendre. Peut-être qu'un jour, cette séduisante idée deviendra réalité. Le développeur n'aura plus qu'à chercher autre chose à faire...

« from natural language specifications to program input parsers » : <http://people.csail.mit.edu/taolei/papers/acl2013.pdf>

Bonne rentrée !

François Tonic

Rédacteur+en+chef (de) Programmez ! [fin de ligne]

Abonnement : Programmez, 17, Route des Boulangers, 78926 Yvelines Cedex 9 - 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.

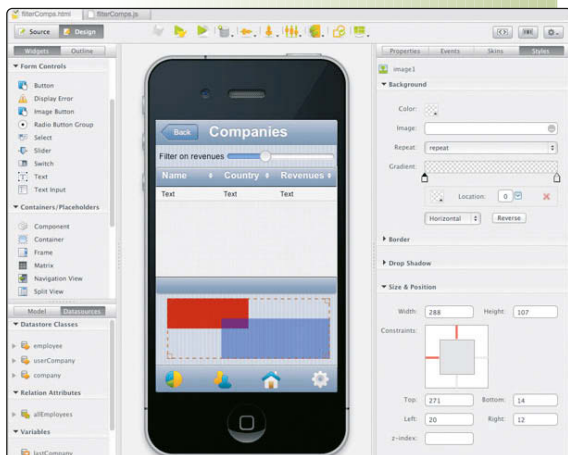
PDF : 30 € (Monde Entier) souscription exclusivement sur www.programmez.com

Editeur : Go-02 sarl, 21 rue de Fécamp 75012 Paris - diff@programmez.com. **Rédaction :** redaction@programmez.com - **Directeur de la Rédaction :** Jean Kaminsky. **Rédacteur en Chef :** François Tonic - ftonic@programmez.com. **Ont collaboré à ce numéro :** F. Mazué, S. Saurel. **Experts :** V. Subramaniam, G. Leborgne, J. Thiriet, A. Fildier, G. Couturier, R. Vidal, S. Hanfi, F. Guillot, S. Khalil, M. Hollebecq, V. Saluzzo, S. Civetta, N. Thenoz, M. Moizard, O. Marguilland, F. Muiseroux, S. Ferquel, J. Antoine, M. Frappat, S. Thach, M. Perfetti. **Illustration couverture :** image du globe : ©iStock/Ymgerman, image du personnage : ©iStockphoto-colorofime - **Publicité :** Régie publicitaire, K-Now sarl. Pour la publicité uniquement : Tél. : 01 41 77 16 03 - diff@programmez.com. **Dépôt légal :** à parution - Commission paritaire : 0717K78366 ISSN : 1627-0908. Imprimeur : S.A. Corelio Nevada Printing, 30 allée de la recherche, 1070 Bruxelles Belgique. **Directeur de la publication :** J-C Vaudecrane. Ce numéro comporte un encart jeté One & One

» 4D a dévoilé Wakanda 5.0 !

Wakanda est une plateforme JavaScript complète et open source. L'éditeur propose dorénavant une large offre de licences et de supports pour son outil. Cette v5 apporte de nombreuses modifications et nouveautés sur le studio, les API, les éditeurs, le serveur. Les équipes travaillent sur les fonctions des versions 6, 7 et 8. Par exemple, dans la version entreprise 8, le module de rapport sera directement accessible depuis le studio. Sur la partie licence, on a le choix entre :

- Wakanda Development Business : permet aux développeurs de développer un nombre illimité d'applications et de ne pas divulguer leur code source au moment du déploiement. Utilisation des versions commerciales du serveur, du studio et du serveur d'applications. 99 € H.T. / an / développeur



- Wakanda Deployment Business est dédiée au déploiement commercial, de telle façon que les développeurs puissent mettre en production leurs solutions sans partager leur code source. 199 € HT par an et par instance de Wakanda Server.
- D'autres licences seront disponibles : Development Enterprise et Deployment Enterprise. Site : <http://developer.android.com/about/dashboards/index.html>

» Il est souvent question de la fragmentation de l'univers Android

Google communique beaucoup sur le sujet pour rassurer car cela reste un sujet de préoccupation pour les développeurs. Depuis juillet dernier, la version Jelly Bean (4.1.x) est la version d'Android la plus utilisée (34 %), juste devant Gingerbread (33 %). Les versions 4.0.x pèsent 22,5 %. La page dashboard indique aussi la version d'OpenGL et les dimensions et la définition des écrans. Des indications précieuses pour les développeurs !

» National Instrument (NI) lance la version 2013 de son outil de

conception graphique LabView. Il est destiné aux ingénieurs et scientifiques pour concevoir et mesurer des environnements embarqués, des machines outils, etc. Cette version supporte les derniers matériels du marché (ARM, Xilinx). LabView évite aux développeurs d'apprendre de nouvelles API ou de maîtriser les derniers capteurs et instruments, l'outil fournit une couche d'abstraction. Parmi les nouveautés : la possibilité de se connecter à l'environnement depuis un terminal mobile, Pour tout savoir sur LabView 2013 : <http://www.ni.com/labview/whatsnew/f/>



» PhoneGap 3.0 a été dévoilé au cœur de l'été

Pour rappel, PhoneGap est un framework de développement open source pour les mobiles.

Il utilise HTML 5, CSS3, JavaScript. Cette version 3 introduit plusieurs améliorations et nouveautés importantes :

- nouvelle architecture pour les plug-ins
 - un outillage amélioré : il est plus facile de basculer entre les différentes plateformes natives même si le SDK n'est pas installé
 - support de iOS 6 (la v7 est en cours), Windows Phone 8 et BlackBerry 10
- site officiel : <http://phonegap.com>

» Microsoft veut simplifier le développement d'application Windows Phone

Pour ce faire, l'éditeur propose une préversion de Windows Phone App Studio. L'outil est accessible depuis un simple navigateur. Le but est très simple : créer une application mobile sans faire de code. Le développeur n'a pas forcément le temps, ni la maîtrise des outils standards.

Pour tester l'application, vous devrez enregistrer un téléphone Windows Phone, avec 2 applications déployables. Les développeurs enregistrés pourront débloquent 3 mobiles et déployer 10 applications (par mobile). Le code source

est disponible et consultable. A noter qu'un certificat devra être installé. Impossible de s'en passer !

App Studio s'appuie sur 4 étapes :

- l'idée
- ajouter du contenu
- choisir un style, un design
- utiliser l'application

Site officiel :

<http://apps.windowsstore.com/default.htm>

» MySQL Workbench 6 facilite la conception

L'outil permet de créer, gérer, modéliser des bases de données aussi bien pour les DBA que les développeurs. Il permet aussi de créer et d'optimiser les requêtes SQL, avec un puissant éditeur dédié.

La principale nouveauté est un module "database migration".

Ce module facilite la migration des bases SQL Server, Sybase ASE, PostgreSQL vers MySQL. L'outil est disponible sur OS X, Windows et Linux. La version communautaire reste gratuite.

» Microsoft met un terme à sa propre technologie de tags : MS Tag

Le service sera totalement arrêté le 19 août 2015. Mais dès le 18 septembre 2018, vous pourrez commencer à migrer vers d'autres services comme Scanbuy qui supporte MS Tag.

Site : <http://tag.microsoft.com/home.aspx>

EN ROUTE POUR LES NOUVEAUX DOMAINES

Vous pouvez maintenant choisir parmi **plus de 700 nouvelles extensions de domaines** pour créer l'adresse originale et facile à retenir qui vous correspond le mieux, comme par exemple **monrestaurant.paris**, **magalerie.art** ou **legrand.shop**. Pré-réservez aussi de nouvelles extensions pour vos domaines existants. Il sera d'autant plus facile de vous trouver !

Avec environ 20 millions de domaines enregistrés, 1&1 est leader du marché de l'enregistrement de noms de domaine en Europe. Grâce à la **fonction de redirection intégrée**, les domaines enregistrés auprès de 1&1 peuvent rapidement et facilement être redirigés vers n'importe quel site Web, quel que soit votre hébergeur.

Plus d'informations sur **1and1.fr**

**NOUVEAU !
PRÉ-RÉSERVEZ
SANS FRAIS
ET SANS ENGAGEMENT***



DOMAINES | MAIL | HÉBERGEMENT | E-COMMERCE | SERVEURS

1and1.fr

* La phase de pré-réservation est gratuite et sans engagement. La disponibilité et le prix des domaines dépendent du registre compétent : 1&1 s'engage à communiquer ces informations dès qu'elles seront disponibles. L'enregistrement du domaine est soumis aux conditions générales de vente de 1&1 et à celles du registre compétent. 1&1 ne peut garantir l'attribution finale des domaines pré-réservés.

Python devient le langage de référence pour les classes préparatoires scientifiques !

Langage très performant et reconnu pour ses qualités, Python sera utilisé à la place des outils propriétaires comme Maple ou Mathematica. Un autre outil open source sera utilisée : SciLab.

PostgreSQL 9.3 sortira avant la mi-septembre

Pour la base de données, il s'agit d'une importante mise à jour, en développement depuis le printemps 2012. Les développeurs préviennent que des incompatibilités pourront avoir lieu entre la 9.3 et les précédentes éditions. Les nouveautés seront nombreuses : support de JSON étendu, amélioration des performances dans l'utilisation de clés étrangères, amélioration des mécanismes de répliqués, fichiers de configurations séparés (via include_dir), tâches en arrière-plan améliorées (et définies par l'utilisateur), écritures sur des tables distantes, fonction de failover plus rapide... Suivre le SGBD sur la communauté française : <http://blog.postgresql.fr>

Nvidia mise tout sur CUDA et OpenGL

CUDA est la technologie permettant d'optimiser l'utilisation du GPU à la place du processeur central, c'est le GPGPU. Son concurrent frontal est OpenCL, ayant pour origine Apple et sous l'égide du Khronos Groupe. Certes, Nvidia propose un SDK et de la documentation technique mais essentiellement en OpenCL 1.1. Le constructeur a dès l'annonce d'OpenGL 4.0 affirmé son soutien et annoncé la livraison de pilotes pour ces cartes mais l'annonce d'OpenCL 2.0 n'a pas eu



Le projet Spark se dévoile ! Initié par Microsoft lors du grand événement E3, Spark sera disponible sur Xbox One et Windows 8. Il s'agit d'un environnement complet de développement de jeux vidéo. Il permet de créer le gameplay, le moteur, le graphisme. Une bêta pour Windows 8 arrivera dans quelques semaines avec des fonctions vocales, de gestes, etc. Tout est fait pour aider le développeur, faciliter le développement, simplifier les tâches. C'est le grand frère d'un autre projet Microsoft, Kodu. Pour en savoir plus : <https://www.facebook.com/joinprojectspark>

Xbox One est désormais accessible aux développeurs indépendants

Jusqu'à présent, Microsoft limitait son accès aux éditeurs. Avec ID@Xbox, cette limitation saute ! Objectif : une fois le développeur certifié, il est auto-éditeur de son jeu. Le processus est simple : inscription, soumission de l'idée du jeu et des informations sur celui-ci, livraison du jeu sur Xbox One. Chaque projet sera examiné. Les développeurs auront accès aux outils et aux SDK. Lien : <http://www.xbox.com/fr-FR/developers/id>



Quel navigateur génère le plus d'erreurs ?

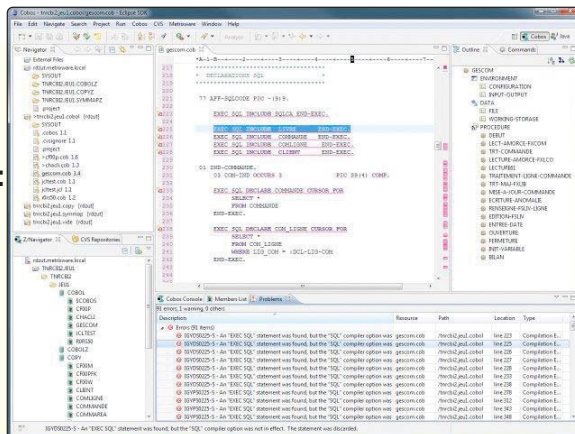
Sauce Labs dévoile son baromètre des navigateurs (basé sur 55 millions de tests). Les résultats sont :

- IE (toutes versions) arrive premier avec 0,25 % d'erreurs
- Safari est 2e avec 0,15 %

Chrome et Firefox sont les meilleurs navigateurs. Si on regarde version par version, IE 6 et 7 sont les pires, avec Chrome 5. Par contre, IE 8 améliore nettement la situation qui devient encore meilleure avec IE 10. En revanche, dans les dernières versions, Safari 6 décroche la 1re place du taux d'erreurs, 0,12 %. Firefox 22 est à 0 ! Etude complète : <http://goo.gl/s8Rs77>

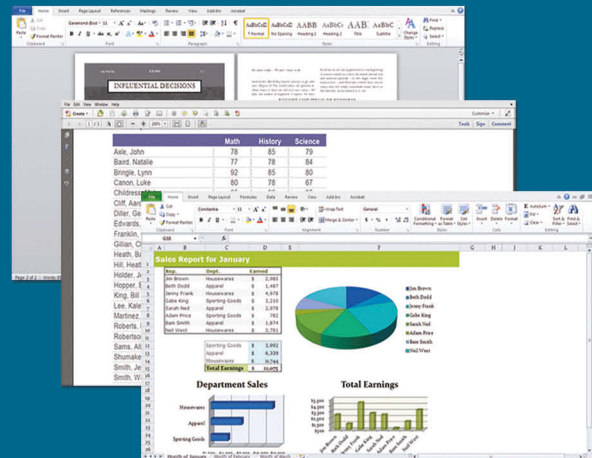
Metrixware dévoile son environnement de développement COBOL sous Eclipse : Cobos 3.0

Cobos permet de coder en langage Cobol directement sous Eclipse. Il est possible de compiler le code Cobol à destination des mainframes. Cette version 3.0 supporte l'interface Endevor (gestionnaire de code et versions pour z/OS, provenant de l'éditeur CA), accès mainframe sans connexion OpenSSH (mode Telnet). Cobos est disponible en open source. Site : <http://metrixware.com/cobos-ide-mainframe-opensource/>



TRAVAILLER AVEC DES FICHIERS?

CONVERTIR
IMPRIMER
CRÉER
COMBINER
& MODIFIER



100% Autonome - Sans Office Automation



+ et beaucoup PLUS
DOC, XLS, PPT, PDF, MSG, VSD, & formats d'images

Obtenez votre version d'évaluation GRATUITE sur www.aspose.com

US Sales: +1 888 277 6734
sales@aspose.com

EU Sales: +44 141 416 1112
sales.europe@aspose.com

 **ASPOSE**
Votre Expert des formats de fichiers

SCANNEZ POUR
REMISE DE -20%





septembre

JDEV 2013 (près de Paris)

Les Journées développement logiciel 2013 se dérouleront du 4 au 6 septembre prochain à l'Ecole Polytechnique (Palaiseau). « Cette manifestation se donne pour objectif d'améliorer la qualité et la pérennité de nos productions logicielles, ainsi que d'ouvrir de nouvelles perspectives avec la présentation des derniers outils et méthodes existants. Les pratiques adaptées au contexte scientifique seront privilégiées », précisent les organisateurs. Inscription et informations : <http://devlog.cnrs.fr/jdev2013>

BootCamp Android (Rennes)

Le GDG Nantes propose le 13 septembre prochain un événement dédié à la programmation Android. Il s'agit de la première conférence de ce groupe de développeur. « Le Bootcamp Android est une introduction accélérée et pratique au développement Android. Vous allez apprendre, vous allez mettre les mains dans le cambouis, vous allez coder, vous allez enfin cocher la case Apprendre à coder sur Android dans votre ToDo-list », indiquent les organisateurs. Attention : places limitées ! Site : <http://www.gdgrennes.org>

agenda**10 OCTOBRE**

Du 03 octobre au 05 octobre 2013, Centre culturel Le Beffroi à Montrouge **OPEN WORLD FORUM 2013**
Cet événement est le premier sommet européen rassemblant politiques, décideurs et experts pour débattre de l'impact technologique, économique et social des technologies libres et ouvertes.
<http://www.openworldforum.org>

JUG Summer Camp 2013 (La Rochelle)

En manque de Java durant l'été ? La communauté Java de Poitou Charentes vous propose le 20 septembre une journée entière dédiée à votre langage favori. Le programme n'était pas encore disponible à l'écriture de cette page. Informations : <http://jugsummercamp.com>

JavaOne 2013 (San Francisco)

Du 22 au 26 septembre, Oracle organise la traditionnelle conférence JavaOne. Comme chaque année, il s'agira de faire le point sur l'univers Java, les technologies, les dernières versions, les futures évolutions. De nombreuses sessions techniques permettront d'approfondir des fonctions et API spécifiques. JavaOne est aussi organisée en Inde, Chine, Russie, Amérique du Sud et au Japon. Site officiel : <http://www.oracle.com/javaone/index.html>

Scrum Gathering Paris 2013 (Paris)

Voici l'événement Scrum de la rentrée, en Europe : la conférence Scrum Gathering ! Elle se déroulera du 23 au 25 septembre à Paris. Elle rassemble les meilleurs spécialistes de la communauté. Preuve de l'importance de l'événement : plus de 260 propositions de sessions ont été déposées à l'organisation ! C'est un lieu unique pour rencontrer, échanger, discuter Scrum ! Informations sur le ScrumAlliance : <http://www.scrumalliance.org>

Rencontres Django (Belfort)

Les 28 et 29 septembre, la communauté Django se rassemblera à Belfort pour parler Django. L'organisation est très simple : « Première demi-journée consacrée aux retours d'expériences sous la forme d'interventions courtes et rythmées : conférences de 15 minutes maximum (plus 5 minutes pour répondre aux questions de l'assistance) et lightning talks de 5 minutes. Vous avez fait quelque chose de génial avec ou autour de Django ? C'est le moment de vous exprimer pour partager votre expérience ! ». Site : <http://rencontres.django-fr.org/2013/>

octobre

Qt Developer Days 2013 (Berlin)

Vous utilisez le framework Qt pour vos interfaces ? Ne ratez pas les Qt Developer Days qui se dérouleront du 7 au 9 octobre à Berlin. Parallèlement se tiendront les Qt Developer Days Starter Edition pour tous les développeurs nouveaux sur la plateforme ! Site officiel : <https://devdays.kdab.com>

Codeurs en Seine (Rouen)

Grande rencontre développeurs le 17 octobre à Rouen : codeurs en Seine ! Événement gratuit, il est organisé par trois communautés de la région normande : Normandie Web Expert, le Normandy Java User Group et le Normandy Agile User Group. « Comme ses grandes sœurs Mix-IT, SoftShake, Breizhcamp, cette conférence se veut un mélange de technologies/pratiques permettant aux développeurs d'élargir leur champ d'horizon et de découvrir ce qui se passe dans les autres communautés », indiquent les organisateurs. L'ambition est de réunir plus de 200 développeurs pour cette première édition. Etant proche de Paris (1h en voiture et 1h15 en train), les organisateurs espèrent réunir entre 200 et 300 développeurs pour cette première. De nombreux orateurs sont d'ores et déjà confirmés tels que Antonio Goncalves ou encore Alexis Moussine-Pouchkine. A ne pas rater ! Site web : <http://www.codeursenseine.com>

**Communauté****Fier d'être développeur : 1^{er} événement !**

Le 28 septembre prochain dans les locaux de l'Epita (près de Paris), l'association Fier d'être développeur organise un grand après-midi de rencontres, de techs et bien entendu de développement ! Le but est de s'éclater, de parler dev et de mieux promouvoir le métier de développeur. Une session spéciale Coding4fun sera animée par le fou-furieux David Catuhe... Si vous êtes fier d'être dev, rendez-vous à 13h30 à l'Epita... pour en savoir plus : <http://fierdetredeveloppeur.org/?p=392>



DANS LA JUNGLE DU CLOUD, NOUS AVONS PRIS UN PEU D'AVANCE...



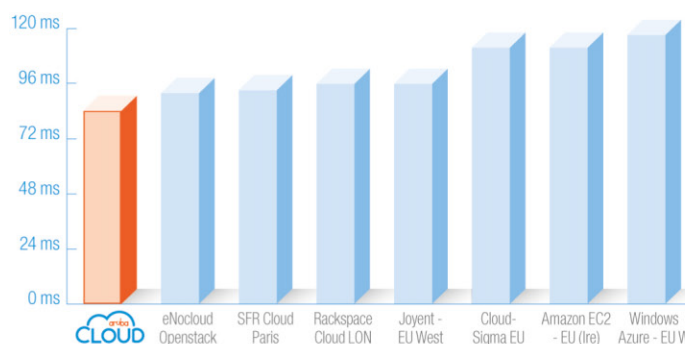
Aruba Cloud, la meilleure offre de Cloud Computing du marché

Le plus performant et le moins cher des Clouds Publics

Fournisseur	Coût mensuel estimé	Performances	Sécurité	Flexibilité
Meilleure proposition			Rackspace	Amazon Web Services
	69 €	★★★★★	★★★★★	★★★★★
Windows Azure	87 €	★★★★★	★★★★★	★★★★★
Amazon Web Services	96 €	★★★★★	★★★★★	★★★★★
CloudSigma	135 €	★★★★★	★★★★★	★★★★★
Rackspace	157 €	★★★★★	★★★★★	★★★★★

Source www.cloudscreener.com
sur les paramètres standards d'évaluation des offres de Cloud Computing.

Classé n°1 en temps de réponse, temps de connexion et disponibilité



Source : Analyse Cedexis réalisée du 5 mai au 5 juin 2013 depuis la France.
www.cedexis.com

**TESTEZ GRATUITEMENT
NOTRE OFFRE**

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



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

MySQL vs MariaDB



MySQL est devenu le Système de Gestion de Base de Données par défaut d'une majorité de projets et le plus utilisé dans les entreprises, à travers le monde. Créé en 1995, il a connu un grand tournant quand MySQL AB, la société propriétaire de MySQL, a été rachetée par Sun Microsystems en janvier 2008, avant que cette dernière ne soit rachetée à son tour par Oracle Corporation en avril 2009.

Ce rachat avait fait craindre la disparition de MySQL, alors le système le plus populaire, au profit d'Oracle, le SGBD d'Oracle Corporation. Mais le futur a été tout autre : Oracle a continué à distribuer MySQL gratuitement, mais la coopération avec le monde du libre n'entraînait plus dans sa stratégie. La méfiance s'est installée et des concurrents libres sont nés et ont commencé à se développer. Un fork s'est particulièrement illustré ces dernières années. Nouveau joujou d'un des développeurs à l'origine de MySQL, MariaDB se veut plus libre, ouvert et communautaire. A l'ETNA, nous nous sommes intéressés à l'éventualité de passer de MySQL à MariaDB, nous avons donc étudié les avantages que présente ce fork.

> Rapide historique

Avant d'entrer dans le vif du sujet, il est nécessaire de comprendre les rivalités et les ressemblances entre MySQL et MariaDB.

MySQL

A l'origine, trois hommes, David Axmark, Allan Larsson et son principal auteur, Michael « Monty » Widenius ont créé une première version en mai 1995. C'est d'ailleurs d'après le nom de la fille de ce dernier, My, que MySQL a pris son nom.

En peu de temps, MySQL acquiert une grande notoriété qui s'explique par plusieurs raisons :

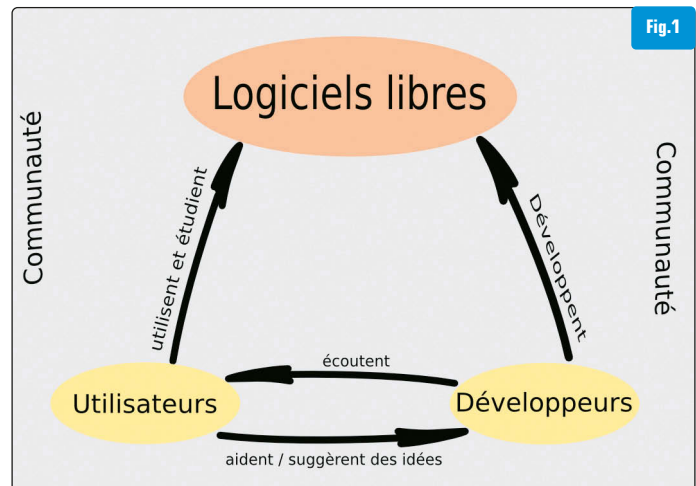
- il est open source
- il est gratuit
- il y avait peu de concurrents gratuits lors de sa création
- il y avait un énorme support de la part des créateurs à ses débuts
- il s'installait très rapidement et avait une prise en main simple

Il est devenu indispensable à la fois pour les particuliers et pour les PME et les multinationales.

Le rachat de MySQL par Sun Microsystems puis par Oracle Corporation a porté un énorme coup au monde du libre. En effet, si beaucoup ont cru qu'Oracle allait relancer certains projets libres qui s'essouffaient par manque de moyens, ce dernier a pris possession du logiciel et a refusé les contributions des utilisateurs. C'est pourquoi une partie de la communauté a estimé devoir forker pour retrouver un projet avec l'esprit libre initial [Fig.1]. Parmi ces personnes, on trouve Monty.

MariaDB

Après le rachat de MySQL, Michael Widenius quitte la société pour lancer sa propre société, Monty Program AB, ainsi que MariaDB (d'après le nom de son autre fille, Maria). Il s'agit d'un fork entièrement compatible avec MySQL. L'un des grands privilèges du monde du libre étant de pouvoir forker un projet, ce qui est impossible avec



Source : Halpanet

les logiciels propriétaires. La première version de MariaDB s'inspire de MySQL 5.1.

Pourquoi Monty s'est-il lancé dans cette nouvelle aventure ? D'une part il a participé depuis le début au code de MySQL et il se méfiait de son futur entre les mains d'Oracle. D'autre part il ne voulait pas perdre toutes les connaissances accumulées durant des années par le groupe de collègues et amis qu'il s'était constitué.

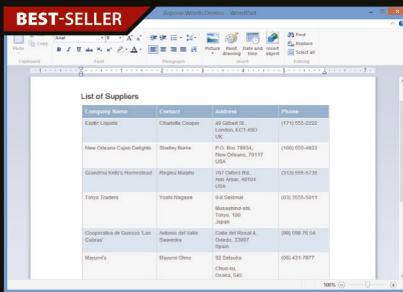
Pour protéger MariaDB, Widenius a fait en sorte que Monty Program AB appartienne à ses employés et donc éviter ainsi les offres agressives de rachat. Parallèlement, il a créé la MariaDB Foundation, qui protège tout ce qui est lié au projet. Le seul moyen que MariaDB disparaisse serait alors qu'il n'y ait plus de développeurs.

> Les avantages

Forcément, Monty a rapidement prévu de nouvelles fonctionnalités et de meilleures performances pour son SGBD. En voici quelques-unes, les principales :

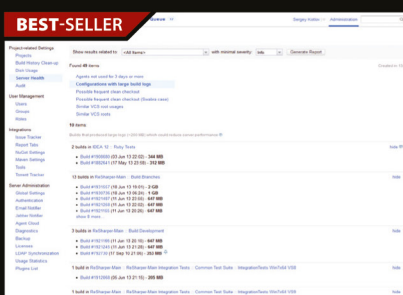
- InnoDB a été remplacé par XtraDB, conçue par la société Percona, en tant que moteur de stockage par défaut
- d'autres moteurs de stockage ont été ajoutés, comme Aria (une version plus performante de MyISAM), SphinxSE ou même Cassandra (pour du NoSQL et notamment utilisé par Facebook et Twitter)...
- les requêtes complexes sont plus rapides, en particulier grâce à Aria
- il y a des optimisations sur les jointures et la gestion des accès concurrentiels

Monty a clairement annoncé son objectif dès le départ avec MariaDB : il a créé une solution de remplacement de MySQL avec de meilleures performances et des nouvelles fonctionnalités. C'est


Aspose.Words for .NET à partir de € 747


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

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


TeamCity à partir de € 1 763


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

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


TX Text Control .NET for Windows Forms/WPF à partir de € 911


Composants complets de traitement de texte pour Windows Forms et WPF.

- Le traitement de texte professionnel pour vos applications
- Chargez, modifiez et enregistrez au format MS Word (DOCX, DOC, RTF)
- WYSIWYG, tableaux imbriqués, cadres, en-têtes, pieds de pages, images, puces, listes numérotées, zoom, sauts de section, etc.
- Exploitez la liaison de données XAML pour lier les barres d'outils et de ruban


DevExpress Universal Suite à partir de € 1 645


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

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

© 1996-2013 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

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

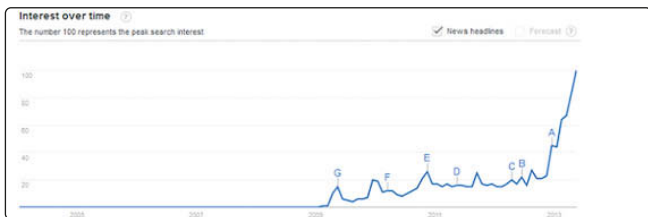
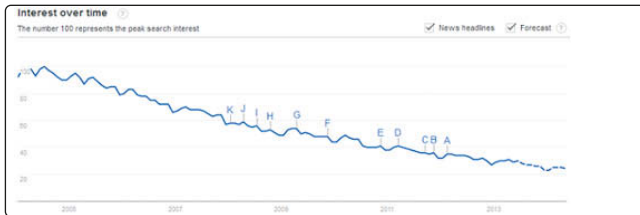
Nous acceptons les bons de commande. Contactez-nous pour demander un compte de crédit.



d'ailleurs pour cette raison que le numéro de version de MariaDB était le même que celui de la version de MySQL sur laquelle il était basé.

> Ils ont fait le pas

La notoriété de MySQL s'est dégradée au fil des ans alors que celle de MariaDB a, au contraire, monté en flèche.



Source : <http://far-fan.com/blog/maria-db>

Les performances de MariaDB ont poussé plusieurs distributions Linux à l'utiliser, citons :

- Arch Linux
- Fedora (depuis Fedora 19)
- OpenSUSE (depuis openSUSE 12.3)
- Red Hat (depuis RHEL 7)

Ce n'est pas disponible sur les dépôts officiels de Debian mais les packageurs sont en discussion pour savoir s'ils vont remplacer MySQL par MariaDB.

D'autres logiciels et grands noms de l'informatique ont aussi fait le grand pas, soit en utilisant exclusivement MariaDB, soit en étant compatibles : phpMyAdmin, Mozilla, Drupal, Zend Framework, Wordpress... Le récent passage de Wikipédia à MariaDB a fait couler pas mal d'encre et beaucoup, maintenant, commencent à s'y intéresser.

Au niveau de la compatibilité entre MySQL et MariaDB, tout a été fait du côté de ce dernier pour que les utilisateurs n'aient pas à se débarrasser de leurs anciens outils et donc n'aient pas à tout recommencer :

- les API et structs s'utilisent de la même manière
- Les noms de fichiers, ports, sockets et binaires sont les mêmes
- Les connecteurs MySQL (PHP, Python, Perl, Ruby...) sont aussi compatibles
- Les paquets mysql-client fonctionnent avec MariaDB et inversement

> Installation et utilisation

Sur les distributions où MariaDB n'est pas installée par défaut, l'installation est assez aisée. Prenons pour exemple Debian (la version 6 Squeeze et la version 7 Wheezy).

Debian Squeeze

Dans le dossier `/etc/apt/sources.list.d`, créez un fichier `maria.db.list` dans lequel vous écrirez les lignes suivantes :

```
# MariaDB 10.0 repository list - created 2013-06-26 14:35 UTC #
http://mariadb.org/mariadb/repositories/ deb
http://ftp.igh.cnrs.fr/pub/mariadb/repo/10.0/debian squeeze main
deb-src
http://ftp.igh.cnrs.fr/pub/mariadb/repo/10.0/debian squeeze main
```

Puis il faut installer la clé signée :

```
sudo apt-key adv --recv-keys --keyserver keyserver.ubuntu.com
0xc9cb082a1bb943db
```

Notez qu'une liste des dépôts officiels est disponible sur le site de MariaDB, vous permettant d'utiliser des versions antérieures ou des miroirs différents. Une fois la clé importée, vous pouvez installer MariaDB :

```
sudo apt-get update
sudo apt-get install mariadb-server
```

Debian Wheezy

Vous pouvez utiliser la même méthode que précédemment en remplaçant « squeeze » par « wheezy ».

Ou alors :

```
sudo apt-get install python-software-properties sudo apt-key
adv --recv-keys --keyserver keyserver.ubuntu.com 0xc9cb082a1bb
943db sudo add-apt-repository 'deb
http://ftp.igh.cnrs.fr/pub/mariadb/repo/10.0/debian wheezy main'
sudo apt-get update
sudo apt-get install mariadb-server
```

```
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MariaDB connection id is 44
Server version: 10.0.3-MariaDB-1''squeeze-log mariadb.org binary distribution
Copyright (c) 2000, 2013, Oracle, Monty Program Ab and others.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
MariaDB [(none)]> _
```

Au premier abord, pas de différences notables. MariaDB utilise également le serveur `mysqld`, donc la commande à utiliser reste « `mysql` ». Le fichier de configuration principal est toujours « `my.cnf` ». La seule différence qui nous saute aux yeux est alors le prompt, changé en « `MariaDB` » au lieu de « `mysql` ».

En utilisation un peu plus avancée, on remarque la présence d'un plus grand nombre de moteurs de base de données (« `show storage engines ;` »). En l'occurrence, la présence de XtraDB, Aria, FederatedX, et SphinxSE qui sont également présents dans MySQL mais qui nécessitent d'être compilés et installés par l'utilisateur.

Les autres changements ne sont pas si visibles mais se situent au niveau des performances.

Tests

Les tests suivants ont été réalisés via DBT-3 (OSDL Database Test 3), un outil de gestion et optimisation de charge de machine et réseau et de tests de performance de systèmes informatiques. Il a été développé par Open Source Development Labs pour Linux et est basé sur TPC-H.

Il est possible de refaire les tests en suivant le lien : <https://kb.askmonty.org/en/dbt3-automation-scripts/>

WINDEV®

ATELIER DE GÉNIE LOGICIEL
PROFESSIONNEL

DÉVELOPPEZ 10 FOIS PLUS VITE



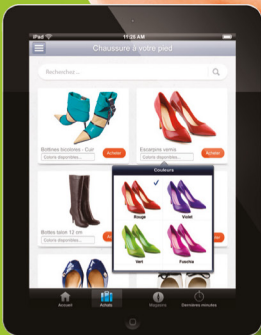
Windows 8, 7, Vista, XP...,
Linux, Mac, Internet, Intranet,
Cloud, Android, iPhone, iPad...

Vos applications sont compatibles

Environnement de développement professionnel, intégralement en français (logiciel, documentations, exemples)

La facilité de développement avec WINDEV est devenue légendaire: vos équipes développent plus vite, la qualité des logiciels est automatiquement élevée, le nombre de fonctionnalités de pointe automatiques est impressionnant.

Vous délivrez plus vite vos logiciels, pour la plus grande satisfaction et rentabilité des utilisateurs.



DEMANDEZ VOTRE DOSSIER GRATUIT

260 pages - 100 témoignages - DVD Tél: 04.67.032.032 info@pcsoft.fr

www.pcsoft.fr



Fournisseur Officiel de la Préparation Olympique

WINDEV®

PLATEFORME INTÉGRÉE
DE DÉVELOPPEMENT

Windows, Mac, Linux, RAD Java, ...

10

développez 10 fois plus vite



**VERSION
EXPRESS
GRATUITE**
Téléchargez-la !

Les tests en question consistent en 22 requêtes complexes utilisées sur MySQL 5.5, MySQL 5.6 et MariaDB 5.5. Les résultats de ces tests sont visibles dans le tableau suivant :

Query	MariaDB-5.3.5		MySQL-5.6.4		MySQL-5.5.21	
	Seconds	Relative	Seconds	Relative	Seconds	Relative
1	289		286	-1.04%	319	+10.38%
2	46		48	+4.35%	490	+965.2%
3	243		1322	444%	534	+119.7%
4	138		141	+2.17%	185	+34.06%
5	187		1232	+558.8%	>7200	-
6	199		191	-4.02%	284	+42.71%
7	861		777	-9.76%	803	-6.74%
8	288		1628	+465.3%	742	+157.6%
9	268		307	+14.55%	>7200	-
10	818		1504	+83.86%	1083	+32.40%
11	13		14	+7.69%	342	2531%
12	213		206	-3.29%	452	+112.2%
13	250		230	-8.00%	1576	+530.4%
14	90		92	+2.22%	>7200	-
15	194		190	-2.06%	401	+106.7%
16	129		149	+15.50%	148	+14.73%
17	800		797	-0.38%	867	+8.38%
18	284		>7200	-	>7200	-
19	62		61	-1.61%	2013	3147%
20	>7200		>7200	-	>7200	-
21	185		182	-1.62%	>7200	-
22	13		14	+7.69%	10	-23.08%

Parmi ces 22 requêtes, MariaDB est plus rapide sur 11 requêtes, suivi de MySQL 5.6 avec 8 et enfin MySQL 5.5 avec une seule. Une des requêtes a mis plus de deux heures à s'exécuter et donc, n'est pas passée avec le timeout à 7200 secondes.

On voit donc que MariaDB 5.5 surpasse la version sur laquelle elle est basée, même si elle se fait rejoindre par la version suivante.

Plus de détails : <http://blog.mariadb.org/mariadb-5-3-optimizer-benchmark/>

Support

Lorsqu'Oracle a racheté Sun, il s'était engagé à ne pas favoriser le développement de la version Entreprise au détriment de la version Community (en licence GPL) et de laisser un libre choix à ses clients au niveau du support. Si cela a bien été respecté, il a tout de même effectué des changements drastiques au niveau du prix. L'offre standard est ainsi passée de 599\$ à 2000\$ par serveur par an tandis

que les versions Entreprise et Cluster sont respectivement passées de 3000\$ et 5000\$ à 5000\$ et 10000\$ par an. Les tarifs de support MySQL sont donc beaucoup moins abordables mais restent toutefois moins chers que ceux d'Oracle SGBD.

Du côté de MariaDB, si le support était fait essentiellement au travers de la communauté que Monty a réussi à créer, un partenariat récent a donné une nouvelle dimension à MariaDB. Le rachat de MySQL et la hausse des tarifs du support ainsi que les marges de manœuvre trop minces pour une bonne évolution de MySQL ont poussé une vingtaine d'anciens de MySQL AB à créer SkySQL en 2010. La nouvelle entreprise entend reprendre le support de MySQL pour des tarifs plus abordables et surtout ne pas laisser MySQL mourir. Le 23 avril 2013, SkySQL et Monty Program ont annoncé que leurs deux sociétés fusionneraient dans les mois à venir. L'entité, qui gardera le nom de SkySQL veut s'imposer comme la référence sur le marché MySQL tout en œuvrant au développement de MariaDB. Le support, pour MySQL et MariaDB se décline en deux offres : 1500\$ par an pour l'offre Standard et 4000\$ par an pour l'offre Advanced.

MariaDB présente de meilleures performances que la version MySQL dont elle est issue, mais il faut attendre des semaines voire des mois après l'apparition d'une nouvelle version MySQL, en fonction des changements dont il faut tenir compte. Ce temps d'attente vaut cependant le coup, au vu des fonctionnalités inédites que présente MariaDB.

Avoir un contrat de support avec Oracle réduit également votre choix : il n'y a pas de support MariaDB proposé par Oracle ! Si toutefois vous comptez avoir du support, sachez que d'autres entreprises (SkySQL par exemple), proposent un support à la fois sur MySQL et sur MariaDB. Si, par contre, vous avez une licence commerciale d'Oracle, vous permettant d'intégrer MySQL dans un logiciel propriétaire, il vaut mieux garder MySQL : les développeurs de MariaDB sont tenus de respecter la licence GPL. Enfin, la présence de MySQL sur les dépôts officiels de toutes les distributions Linux rend les choses plus simples : bien souvent, un simple apt-get suffit à l'installer. Quoiqu'il en soit, la compétition entre MySQL et MariaDB ne peut

apporter que du bon. Après tout, on ne peut pas dire que Firefox n'a pas été bénéfique à Internet Explorer !

Vijai SUBRAMANIAM.

Développeur d'applications Web à l'ETNA.



L'information permanente

- L'actu de Programmez.com : le fil d'info **quotidien**
- La **newsletter hebdo** : la synthèse des informations indispensables.
Abonnez-vous, c'est gratuit !

www.programmez.com

The screenshot shows the Programmez.com website. At the top, there's a navigation bar with links like 'Accueil', 'Actualités', 'Livres', 'Logiciels', 'Services', 'Partenaires', 'Contact'. Below the navigation bar, there are several featured articles and promotions. On the left, there's a section for 'Abonnement' (Subscription) with details about the newsletter. In the center, there are articles about 'Concours Azure/Raytracer', 'Nouveaux flashers sur Programmez', and 'Le Groupe KPI, SSLI spécialiste sur SAP'. On the right, there are promotions for 'Microsoft', 'IrisScan Book 2', and 'Programmez n°149'. The bottom of the page features a 'Tribune Libre' (Free Forum) section and a 'Livres' (Books) section.



VITE, ABONNEZ-VOUS jusqu'à -50%

Profitez de cette offre :
bénéficiez pour 1 euro de plus,
d'un des «livres numériques» au choix,
sélectionnés par la rédaction
de Programmez.

Offre limitée

**Pour 1€ de +
Un «livre numérique» des Editions Eni***



HTML5, CSS3 et JavaScript
Prix Version numérique : 29,26 €



Objective-C
Prix Version numérique : 40,5 €



Android
Prix Version numérique : 29,26 €

**réservée à la France
Métropolitaine**

*Accès html au
contenu intégral de
l'ouvrage sur la
Librairie numérique
des Editions ENI,
d'une valeur de
29,26 € à 40,50 €
selon les titres

Valeur des 11 numéros du magazine : 65,50€ + Valeur du livre : 29,26 € à 40,50 € = Valeur totale : 94,76 € à 106 €
Offre : 50€ seulement (ou 60€ avec accès illimité aux archives)**

Toutes les offres en ligne : www.programmez.com

Abonnez-vous à partir de 3,80 € seulement par mois

(*) Tarifs France métropolitaine

Oui, je m'abonne

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

Attention : l'offre avec livre Eni est limitée et réservée à la France métropolitaine

- ☐ **Abonnement 1 an au magazine + 1 « Livre Numérique ENI »** : 50 € (au lieu de 65,45 €, prix au numéro + valeur du « livre numérique ») ☐ Abonnement seul : 49 €
☐ **Abonnement Intégral : 1 an au magazine + Archives** Internet et PDF + 1 « Livre Numérique ENI » : 60 € ☐ Abonnement Intégral seul : 59 €
☐ **Abonnement 2 ans au magazine + 1 « Livre Numérique ENI »** : 79 € (au lieu de 130,9 €, prix au numéro + valeur du « livre numérique ») ☐ Abonnement 2 ans seul : 78 €
☐ **Abonnement intégral 2 ans au magazine + Archives** Internet et PDF + 1 « Livre Numérique ENI » : 89 € ☐ Abonnement intégral 2 ans seul : 88 €

Livre à Choisir : ☐ Html 5 CSS 3 et JavaScript ☐ Objective-C ☐ Android

Détails sur www.programmez.com/abonnement.php

Tarifs France métropolitaine

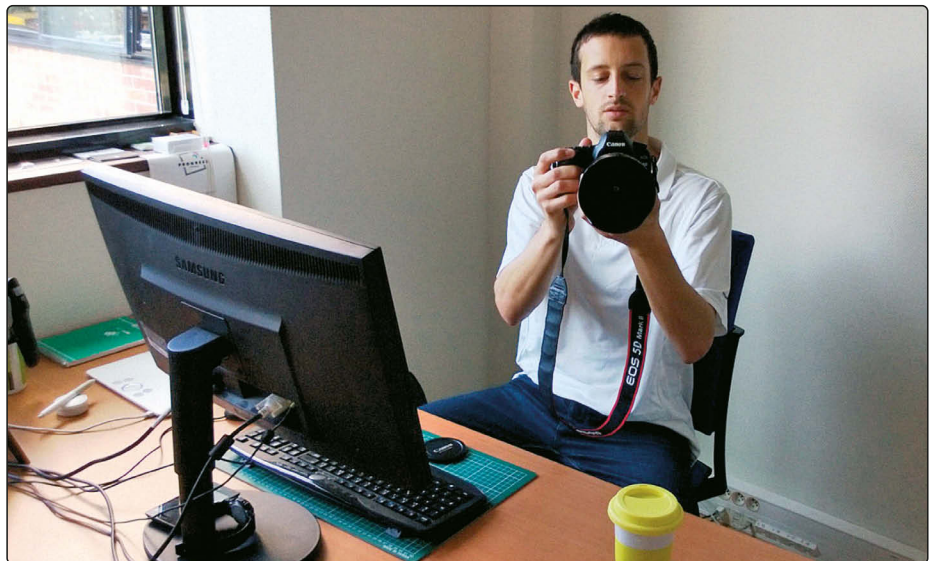
☐ M. ☐ Mme ☐ Mlle Entreprise : _____ Fonction : _____
 Prénom : _____ Nom : _____
 Adresse : _____
 Code postal : _____ Ville : _____
 Tél : _____ **(Attention, e-mail indispensable pour les archives sur internet et offre Eni)**
 E-mail : _____ @ _____

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

☐ Je souhaite régler à réception de facture

Benjamin Launay : « il n'y a pas que le code, il y a aussi le design ! »

« Le développeur du mois » parle des développeurs, voilà une belle lapalissade ! Mais dans un développement, il n'y a pas que le code, il faut aussi du design, de l'ergonomie. Benjamin, designer depuis des années, a répondu à nos questions. Découvrez un « autre monde », passionnant et complémentaire à la programmation...



Comment es-tu tombé dans l'informatique et plus spécialement dans le design ?

Benjamin : Je crois que mon premier pas vers le design s'est fait par le dessin à travers la bande dessinée franco-belge et les mangas. Quand j'étais gosse je prenais une feuille blanche et un crayon et je passais des heures à recopier les dessins que je trouvais dans mes albums. Dans le même temps, j'ai eu mon premier ordinateur, un Atari 1040 STF, sur lequel j'avais Deluxe Paint et une multitude de jeux sur lesquels j'ai passé des heures. Je pense que c'est tout ce temps passé dans ces univers magiques, que ce soit dans la BD, les jeux vidéos et aussi le cinéma qui m'a naturellement amené à m'intéresser à tous les processus de création, qu'ils soient analogiques ou numériques, avec, dans l'idée, de créer mon propre univers. J'ai pratiqué dans mon coin pendant quelques années, surtout le dessin, et puis après le bac je suis parti étudier au Beaux Arts à Caen où j'ai énormément appris. J'ai découvert le travail de la typographie, de la couleur et les règles d'équilibre d'une composition. J'ai également beaucoup pratiqué la photo. C'est à partir de là que j'ai commencé à toucher à tous les outils d'infographie, Photoshop, InDesign, Illustrator, toute la suite Adobe en fait. J'ai découvert la puissance de ces logiciels dans le cadre d'un processus créatif et je n'ai jamais décroché depuis.

Pour toi, qu'est-ce qui fait que l'on aime toujours et encore le design, la technique ?

Benjamin : Ce qui me fait aimer chaque jour mon métier de designer c'est que « j'en apprends » tout les jours. Cela peut sembler banal, mais je pense que si j'arrêtais d'apprendre de nouvelles techniques de nouveaux procédés graphiques, je m'ennuierai assez vite. J'ai commencé ma carrière en faisant quasiment 100% de design print. Puis j'ai continué mes études et je me suis formé sur des projets vidéos, j'ai appris le montage et je me suis fait la main sur des logiciels de post production comme After Effect. Ensuite, j'ai commencé à faire du design pour des projets web et applicatifs. Aujourd'hui je collabore quotidiennement dans mon travail et en dehors avec des développeurs sur des projets où je prends en charge tout l'aspect design UI ainsi que la communication visuelle qui est faite autour des produits. C'est ce qui me plaît dans mon travail, être toujours en mouvement, collaborer avec des corps de métiers différents, monter des projets, devoir apprendre pour les mener à bien et en ressortir avec un nouveau bagage.

Tu as gardé un regard très geek : gadget, veille techno, c'est important pour ton job et ta passion ?

Benjamin : C'est vrai que j'aime me tenir

informé des projets et des sorties en matière de nouvelles technos, particulièrement si cela concerne le design et l'infographie. Mais je ne suis pas très consommateur de gadgets. Je suis particulièrement attentif aux évolutions de Windows en ce moment parce qu'il ont osé une vraie rupture graphique que je trouve très intéressante et que j'expérimente au quotidien dans mon métier. La veille est importante bien sûr dans mon domaine, surtout pour l'inspiration. Je pense que tout graphiste qui se retrouve face à une demande qui ne l'inspire pas se tourne rapidement vers le net, fait ses recherches, pour glaner des infos et amorcer la phase de créa par quelques idées prises ici et là. Personnellement j'ai en favoris une liste longue comme le bras de sites internet de studios et de graphistes qui m'inspirent au quotidien.

Etre designer n'est pas toujours facile : pression, évolution constante, frustration des projets et des "chefs", c'est quoi pour toi d'être designer aujourd'hui ? Le job a-t-il changé depuis tes débuts ?

Benjamin : Ce serait plutôt prétentieux de ma part d'affirmer que le job de graphiste a changé depuis mes débuts étant donné que je suis dans le circuit pro depuis maintenant 4 ans à peine ! En revanche du haut de mes 26 ans je constate déjà de vrais change-

ments de tendance, ou de mode. Un des grands changements qui termine aujourd'hui son cheminement c'est l'avènement de ce qu'on appelle le flat design. Lorsque j'ai commencé à étudier le design graphique pur, le flat design était déjà là, présent dans les écoles d'Art de par les enseignements d'histoire de l'art (le Bauhaus, le style international Suisse, etc.) et les élèves y travaillaient déjà. Pourtant, ce n'est qu'aujourd'hui que ce style prend une vraie place dans notre paysage graphique quotidien, notamment avec les récentes évolutions des interfaces de Windows, Google (un des 1er à avoir fait du flat design), et même d'Apple avec son dernier iOS7 qui quitte progressivement le skeuomorphisme.

Je pense qu'une des difficultés pour un graphiste en entreprise ou en agence c'est de faire valoir de nouvelles tendances à ses supérieurs et de les convaincre qu'il s'agit du design de demain. Si on attend qu'un style graphique soit adopté par les "grands" du circuit, fatalement il est déjà trop tard pour être original et pour se démarquer. La difficulté consiste non seulement à être innovant, mais aussi à franchir le pas et diffuser son innovation. Pour cela il faut croire en son boulot mais aussi avoir la confiance de son équipe.

Et en dehors du boulot, qu'est-ce que tu aimes faire ? Comment trouves-tu l'équilibre entre travail, vie privée, passion, famille ?

Benjamin : J'essaie de travailler le plus souvent possible à des projets personnels, ce qui me permet de toucher à des domaines auxquels je n'ai pas accès au travail. Pour cela j'ai créé le Studio AieAieEye. C'est une entité qui représente pour moi la partie la plus personnelle de mon travail. J'ai créé mon site il y a peu, <http://www.aieaieeye.com/>, j'essaie de l'alimenter régulièrement avec des travaux, des articles et des tutoriels. Je travaille à des projets d'applications avec mon ami développeur Samuel Blanchard qui est MVP Windows Phone. Récemment, je me suis investi dans la refonte graphique du site TechArtGeek.com, refonte qui a été pensée à partir de la charte Modern UI. Je fais également de la photo et de la vidéo. Ce sont des projets que je mène sur les week end et parfois le soir. Cumulé à de grosses journées de travail cela peut parfois faire beaucoup, alors quand je décide de prendre des vraies vacances le mot d'ordre est : "Pas de PC".



Mon bureau

Je travaille sous Windows 7, sur une grosse station graphique dotée d'un processeur Intel Xeon, de 16 go de RAM et d'une carte graphique puissante. Sur cette machine j'ai installé une Master Collection CS6 avec tous les outils qui me sont indispensables. J'ai un écran 24 pouces, une petite tablette graphique Wacom A5 pour la retouche photo. Ma souris est une logitech MX500, rapide, précise et increvable (7 ans de bons et loyaux services). Pour mes travaux photos et vidéos je suis équipé d'un Canon Eos 5D mark2.

Peux-tu nous présenter ton quotidien en quelques mots ?

Benjamin : J'arrive le matin à 9h. Je me fais un petit thé à la menthe et je traite mes mails. S'il y a des demandes urgentes je m'y mets dans la foulée, sinon je reprends mon agenda tel que je l'avais planifié et je me lance. Quand c'est possible je fais les choses les plus répétitives en premier, les déclinaisons de chartes graphiques un peu systématiques, les retouches de photos à la chaîne, comme ça peut arriver, ou les détournages pour préparer des travaux créatifs à venir. J'ai tendance à me garder les tâches les plus créatives pour la fin de matinée et surtout la fin d'après-midi (entre 16h et 19h) C'est là que je suis le plus frais et le plus efficace dans la réflexion purement graphique. Mes combinaisons de couleurs marchent du premier coup, je trouve la bonne typo en quelques clics, mes compositions fonctionnent. Il y a ces moments dans la journée où j'ai l'impression d'avoir plus de "chance" et d'instinct qu'à d'autres.

C'est important quand on fonctionne comme ça de savoir localiser ces moments pour être plus efficace dans le travail et aussi pour passer des journées plus agréables

sans aller à contresens de son propre rythme.

Comment vois-tu ton job évoluer ?

Benjamin : Je n'ai jamais souhaité me spécialiser dans un domaine du design en particulier. Tout m'intéresse et je ne veux pas passer toutes mes journées à faire uniquement de la mise en page ou du montage vidéo, je veux faire les deux et plus encore. Je suis dans le même état d'esprit quand je débute un nouveau projet, je veux pouvoir avoir une vision globale du chantier et être capable en rassemblant toutes mes compétences de définir un objectif graphique précis. Peut-être qu'il s'agit tout simplement du métier de directeur artistique que je suis en train de définir et c'est peut-être ce qui m'attend pour l'avenir.

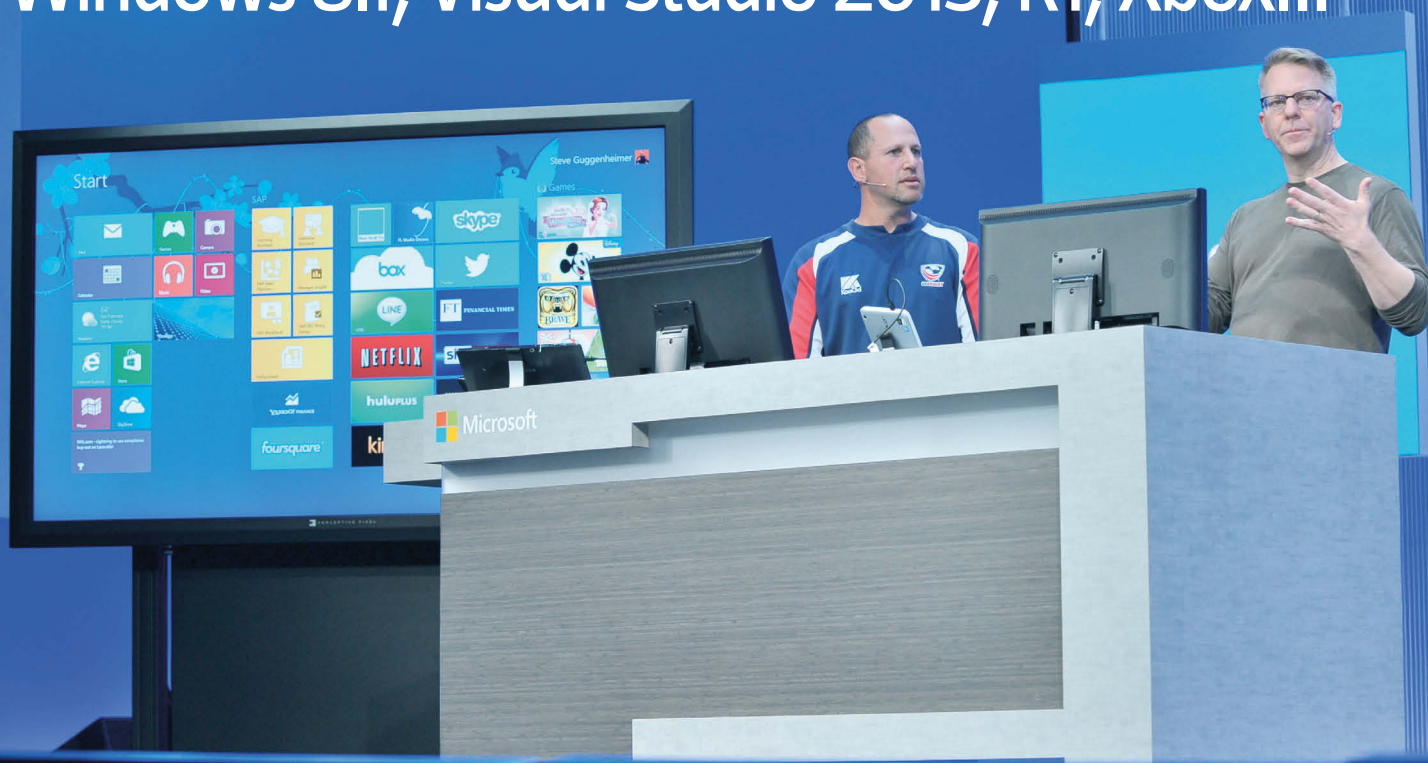
Des conseils aux étudiants et graphistes qui nous lisent ?

Benjamin : Faites ce qui vous plaît, restez actif, toujours en mouvement et montez des projets, même avec zéro moyen, c'est comme ça qu'on apprend le plus.

Mon site : <http://www.aieaieeye.com/>

Retour sur la conférence BUILD

Windows 8.1, Visual Studio 2013, RT, Xbox...



La grande conférence développeur de Microsoft, BUILD, s'est déroulée du 26 au 28 juin dernier. Des milliers de personnes ont fait le déplacement.

Et le programme technique était très chargé : Windows 8.1 et les nouvelles API, Xbox One, la plateforme Bing, Kinect pour Windows (prochaine version), Surface RT, Xbox Music, Babylon.JS, Visual Studio 2013, TFS 2013, les imprimantes 3D dans Windows, les services Windows Azure, le nouveau .Net...

Difficile de s'y ennuyer de 8h30 à 18h.

C'est aussi une parfaite occasion pour rencontrer les équipes techniques, discuter avec les ingénieurs, rencontrer d'autres développeurs.

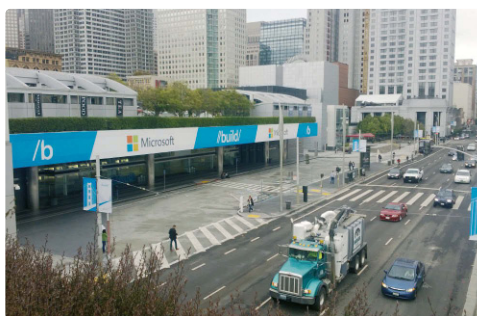
BUILD est un formidable réseau social. Pour les p'tits français qui n'ont pas pu s'y rendre, Microsoft France a organisé une conférence « Back from BUILD » le 5 juillet dernier.

Sur Windows Phone, peu de choses, Microsoft ayant fait l'impasse sur cette plateforme, faute de nouvelles annonces importantes. Il faudra sans doute attendre l'automne.

A noter que Windows 8.1 a été livré aux constructeurs fin août dernier.

Une disponibilité pour tout le monde mi-octobre a été annoncée.

Dans ce dossier, nous revenons sur les principales annonces développeurs.



//build/

François Tonic

(A suivre dans le prochain numéro...)

Quoi de neuf pour les développeurs avec Windows 8.1 ?

Durant la conférence BUILD, Microsoft a levé le voile sur les fonctionnalités de Windows 8.1, une mise à jour majeure de Windows. Cette évolution sera disponible gratuitement à travers les mises à jour Windows.

Outre les améliorations d'ergonomie, cette mise à jour apporte de nombreuses nouveautés pour les développeurs. Nous vous proposons ici d'en faire un petit tour d'horizon.

NOUVEAUTÉS DE LA PLATEFORME

> DirectX

DirectX 11.2 va voir le jour. Cette version apporte des améliorations de performances grâce à une meilleure gestion de la latence et à différentes optimisations des API. Les ressources peuvent également être gérées en tuiles de différentes qualités pour améliorer les performances et la qualité de rendu.

DirectX est, depuis quelque temps déjà, utilisé pour faire le rendu dans les différentes plateformes applicatives (XAML, moteur de rendu de IE, ...). Une nouvelle API basée sur DirectX permet maintenant de faire de la composition sur les images, les animations et les vidéos : l'API DirectComposition. Cette API est au cœur de certaines des optimisations de Windows 8.1 comme, par exemple, la correction du problème des webviews dans les applications XAML, mais aussi les performances d'affichage dans les applications.

> IE11

Avec 8.1 Internet Explorer passe en version 11. En plus des traditionnelles optimisations de performance, IE s'enrichit de WebGL (un wrapper OpenGL) et supporte maintenant MPEG-Dash, un format de vidéo adaptatif basé sur http. La balise vidéo s'enrichit maintenant d'une API de gestion de codec, et d'une autre pour la gestion des DRM. La partie CSS évolue vers moins de préfixes et plus de CSS3 comme les bordures en 9-grid. Le canvas s'enrichit aussi de quelques nouvelles fonctionnalités comme les lignes en pointillé [Fig.1].

Une API de cryptographie fait également son apparition. Cette API permet d'utiliser des algorithmes de hachage ou d'encryption



Scène 3D en OpenGL avec Babylon.js

dans les applications web. Une autre API permet aussi de gérer les interactions avec le presse papier pour mieux gérer les copier-coller. Pour finir sur IE11, les outils pour les développeurs ont été totalement refondus pour s'aligner sur ceux disponibles dans Visual Studio.

> WinRT

Faisons maintenant un petit tour du côté des applications Modern UI et des API WinRT sur lesquelles elles reposent. Certains des blocs existants dans WinRT ont été améliorés, d'autres font leur apparition. Nous ferons un focus sur la partie XAML et sur la partie HTML dans la suite de cet article.

Communications

8.1 permet de communiquer plus facilement avec différents types de périphériques et d'appareils. Les API de proximité (NFC) et de PlayTo (envoi de flux audio ou vidéo sur un périphérique) ont donc été améliorées, avec par exemple le support de Miracast. On trouve également de nouvelles API permettant de gérer les appareils USB, le wifi-direct (connexion point à point en Wifi), le Bluetooth,

et les appareils de lecture (lecteurs de code barre et de cartes magnétique), et des choses plus exotiques comme les API pour les périphériques HID ou les imprimantes 3D.

Interactions

Un autre axe majeur d'amélioration de WinRT concerne les interactions avec Windows et avec les autres applications Modern UI. Les contrats de base ont été enrichis. Le contrat de partage propose maintenant des actions par défaut avec la possibilité de partager une capture d'écran (même si l'application ne sert pas explicitement de source) et un lien vers l'application dans le store. Dans les informations proposées par le partage, le système pré-renseigne également des informations sur votre application (icône et nom), afin que l'application qui reçoit le partage puisse afficher ou garder trace de l'origine de l'information en plus de proposer une description de l'action qui sera effectuée. Le partage permet également de différencier les liens web des liens applicatifs (pour pointer sur un contenu dans une application) [Fig.2].

Le contrat de recherche a également été

modifié. La nouvelle recommandation est de mettre un bloc de recherche dans l'application elle-même (en haut à droite de l'écran) plutôt que de recourir au talisman de recherche. Un contrôle « SearchBox » a été ajouté en XAML et en HTML pour mettre en place cette approche. Il possède les mêmes signatures d'événements que le talisman de recherche pour faciliter la migration des applications existantes. Pour améliorer la recherche, il est aussi possible d'utiliser le moteur d'indexation de Windows pour la recherche dans votre application.

Il est possible également d'interagir avec les contacts et les événements du calendrier. Pour des raisons de sécurité, ces interactions se font à travers Windows et votre application n'accède pas directement aux informations. On peut afficher des fiches contact ou calendrier, ou proposer un nouveau contact ou un nouvel événement. Pour les contacts, il est possible d'enregistrer des actions (téléphone, message, etc.) par vos applications en les déclarant dans le manifeste. L'application Skype par exemple, s'enregistre selon ce principe et est activable dans les fiches de contact [Fig.3].

Windows 8.1 permet maintenant d'afficher plusieurs applications simultanément (4 au maximum par écran selon la résolution). Une application peut également ouvrir des sous-fenêtres qui s'affichent de la même façon qu'une autre application (cette 2e fenêtre est quasi-indépendante et possède son propre thread UI). Le manifeste de l'application permet de définir quelle est la taille minimale d'affichage (320 ou 500px). Il n'existe plus véritablement de mode « snap », charge aux applications de s'adapter à la

place qu'on leur donne. Lorsqu'on ouvre une autre application ou fenêtre, il est possible de déclarer ses préférences quant à la taille que va prendre notre application. L'application ouverte peut aussi exprimer ses préférences d'affichage dans son manifeste.

WinRT propose également une API de synthèse vocale et une API permettant de faire du rendu de PDF et d'arbre de contrôles XAML sous la forme d'images.

Connectivité

Dernier axe d'amélioration de WinRT : la connectivité réseau. La gestion de l'authentification a été améliorée et une API Http vient compléter WinRT. Cette API permet de gérer les appels de services et le cache http. Elle comporte un objet HttpClient similaire à celui de .Net 4.5. La version proposée dans WinRT permet en supplément de contrôler la gestion du cache, et un système de filtres similaires aux HttpHandler de ASPNet pour implémenter des logiques transverses (cache, retry, authentification, ...). Cette API propose également un « ContentPrefetcher ». Il vous laisse définir une liste de ressources réseau qui vont être appelées par Windows pour pré-charger vos données ou vos images. Cette liste peut être définie localement dans l'application, ou sous la forme d'une url d'un site web qui va la retourner sous forme de flux XML. La récupération des données est laissée à l'appréciation du système selon différentes règles (fréquence d'utilisation de l'application, usage des ressources téléchargées, niveau de la batterie et du réseau, ...).

> Webview

Cette nouvelle version du contrôle permet d'intégrer des contenus web à l'intérieur des applications. Que ce soit en XAML ou en HTML, elle se comporte comme une inclusion d'une fenêtre de navigateur. Elle propose davantage d'événements pour tracer ce qui se passe ou intercepter les erreurs. Il existe aussi une API permettant de gérer la navigation (équivalent des boutons Précé-

dent et Suivant des navigateurs). Cette webview offre également la possibilité d'injecter du contenu sous la forme d'un flux de données, en reposant sur une mécanique d'adaptateurs fournis par votre application. Vous pouvez ainsi packager des contenus (formats spécifiques, données cryptées, ...) et les mettre en forme en HTML.

LES NOUVEAUTÉS POUR LES APPS EN HTML / JAVASCRIPT

Toutes les nouveautés et améliorations énoncées précédemment pour IE11 sont applicables aux applications Modern UI en HTML et javascript. On peut donc maintenant faire de la 3D avec WebGL dans nos applications HTML. Par ailleurs, avec la mise à jour vers Windows 8.1, la librairie WinJS passe en version 2. Cette nouvelle mouture apporte de nouveaux contrôles, et les API existantes ont été optimisées. Les contrôles existants, même s'ils ont été améliorés, ne changent pas les signatures de méthode. Le portage d'une application vers Windows 8.1 s'avère donc relativement aisé.

Le modèle de programmation des contrôles a quelque peu évolué pour introduire le design pattern « disposable ». Ce pattern propose un mécanisme standard de libération des ressources, le but final étant d'éviter les fuites mémoires. Les contrôles de la librairie WinJS implémentent maintenant ce pattern, charge à vos contrôles sur mesure de le respecter également. Pour ce faire, rien de plus simple. Déclarez une fonction « dispose » dans laquelle vous devrez libérer vos ressources (par exemple les événements auxquels votre contrôle a souscrit), et appelez cette méthode sur les contrôles contenus à l'intérieur du vôtre.

Le fonctionnement des templates a aussi été revu. La syntaxe reste identique mais les templates sont maintenant compilés en une fonction qui génère le HTML plutôt que d'être interprétés à la volée. Ce fonctionnement permet d'améliorer sensiblement les performances.

Lorsqu'on réalise des applications Modern UI en HTML, une des difficultés est de coordonner le chargement des différents éléments de la page pour avoir un bon ressenti utilisateur. Jusqu'ici, la meilleure solution était d'utiliser des setTimeout ou des setImmediate pour différer l'affichage de certaines portions. Le problème est que si on veut garder une logique dans l'organisation du code, l'ordre dans lequel s'exécutent les

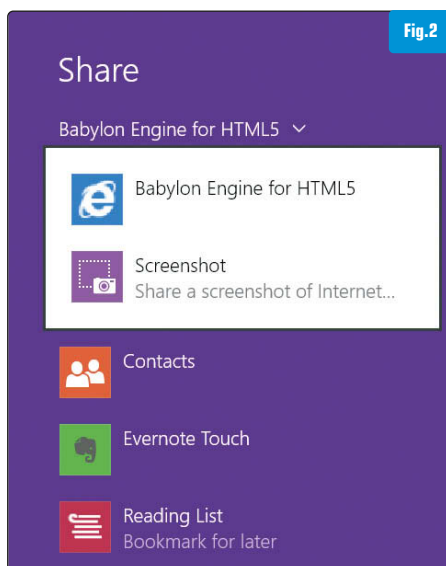


Fig.2

Partage

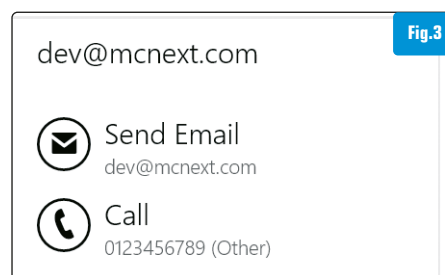


Fig.3

Fiche contact

différentes parties peut être chaotique. WinJS introduit une API qui permet d'orchestrer tout cela grâce à un Scheduler. Cette API fonctionne sensiblement comme un setTimeout, pour lequel on précise un niveau de priorité plutôt qu'un temps. Les éléments d'un même niveau de priorité seront exécutés de façon séquentielle.

> ListView

Le contrôle ListView a fait l'objet d'un grand lifting. Le layout mis en place par le contrôle a été entièrement revu pour permettre d'améliorer les performances et corriger ses petits défauts de jeunesse. Il se base maintenant sur les layout en grille et/ou en flexbox plutôt que de calculer la position des éléments en javascript. Cette nouvelle approche permet de gérer l'affichage en grille et les regroupements, quelle que soit l'orientation de la liste, et autorise le glisser/déposer pour réorganiser les items. Cette opération repose sur les mécanismes standards de HTML, il est donc possible de faire des glisser/déposer depuis ou vers n'importe quel élément HTML.

> Barres de menu

Les barres de menu, ou AppBar, ont été revues pour faciliter leur utilisation. Les barres de menu s'adaptent mieux aux différentes résolutions (en ajustant la taille des éléments selon la résolution), et sont maintenant manipulables au clavier. Par ailleurs, une option sur les commandes des barres de menu permet maintenant d'utiliser n'importe quel élément (zones de saisie, etc.) dans les barres de menu.

> Barres de navigation

Ce nouveau contrôle s'intègre dans la AppBar situé en haut de l'écran pour proposer un menu de navigation, sur le modèle de l'application « actualité » de Windows. Il permet de gérer plusieurs niveaux de menus avec des affichages différents [Fig.4 et 5].

> Les Hub

Le hub est un pattern de conception d'IHM très présent dans Windows 8. Il permet de présenter une succession de rubriques dont la mise en page est parfois très différente. WinJS introduit un contrôle pour faciliter la mise en place de Hubs dans vos applications.

Un nouveau template projet utilisant ce contrôle est également disponible dans les outils de développement [Fig.6].

> Repeater

La ListView est un contrôle très puissant, notamment lorsqu'on souhaite afficher des listes comportant de nombreux éléments. Elle permet de virtualiser les données et/ou l'affichage. Cette puissance s'accompagne d'un surcoût sur les performances. Quand on souhaite n'afficher que quelques éléments (comme c'est souvent le cas dans les hubs), il est préférable de recourir à quelque chose de plus léger. Le contrôle Repeater permet de simplement générer une liste de nœuds HTML à partir d'une source de données.

> ItemContainer

Ce contrôle agit comme une coquille qui apporte la même interactivité que les éléments d'une ListView, à savoir la possibilité de sélectionner ou de cliquer sur un élément. Il sera d'une grande utilité si vous gérez vos listes à la main ou avec le Repeater.

LES NOUVEAUTÉS DE XAML

XAML pour WinRT a aussi eu le droit à un petit lifting grâce à des corrections de bugs, à des optimisations mais aussi avec l'arrivée de nouveaux contrôles. Parmi les autres nouveautés, on notera le support des DPI

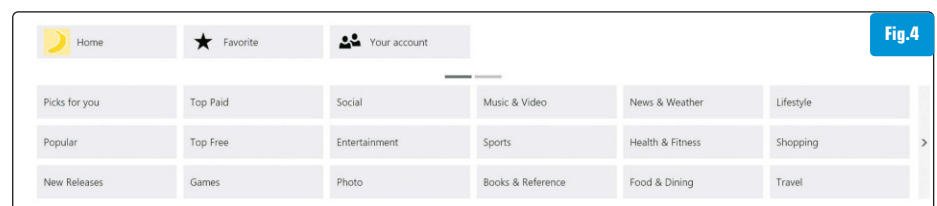
multiples, des hauts contrastes, de l'accessibilité, des améliorations des FlipView. Désormais, on peut aussi contrôler directement la pile de navigation.

> TimePicker et DatePicker

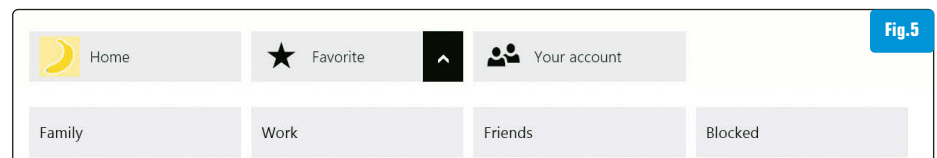
Ces deux contrôles sont attendus depuis longtemps. Bien qu'ils semblent faciles à développer, il faut prendre en compte que chaque pays ou région peut avoir un affichage de date très différent [Fig.7].

> Flyouts

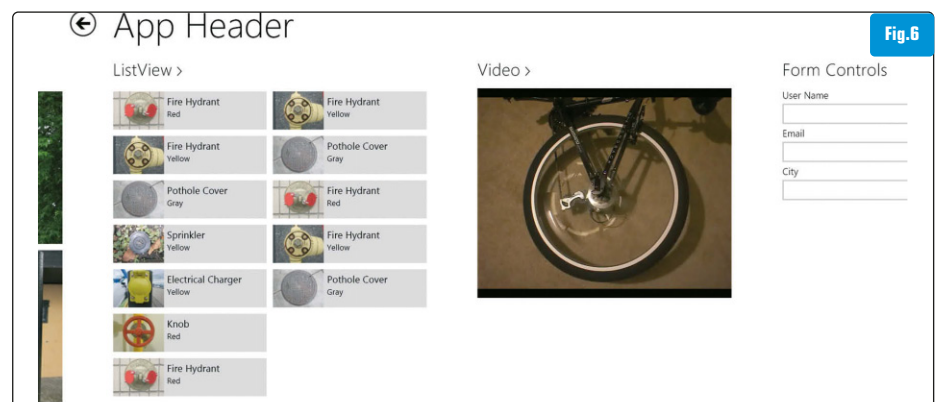
Les Flyouts sont de petites popin qui s'affichent dans vos interfaces Modern UI. Elles servent par exemple à ajouter des options dans une AppBar. Leur utilisation a été améliorée dans Windows 8.1. Tout d'abord on voit l'apparition d'un contrôle Flyout utilisable en XAML via une propriété attachée Flyout. On note aussi la création d'un contrôle MenuFlyout permettant d'afficher des menus supplémentaires, dont l'utilisation se rapproche du ContextMenu en WPF. Le petit dernier est le contrôle SettingFlyout qui permet de gérer un écran de paramètres. Un vrai gain de temps car gérer les différents thèmes, contrastes etc... pouvait être assez fastidieux [Fig.8].



Barre de navigation



Barre de navigation



Contrôle Hub

> Thèmes

Parmi les nouveautés inattendues se trouve l'apparition d'un mécanisme de gestion de thèmes. Une nouvelle propriété `RequestedTheme`, présente dans la plupart des contrôles XAML, permet de spécifier le thème que l'on souhaite utiliser. C'est le genre de choses très utile pour gérer les Flyouts. En conjonction on découvre une nouvelle `MarkupExtension` : `ThemeResource`. Elle s'utilise à la place de `StaticResource` lorsqu'on souhaite utiliser les thèmes par défaut, et permet de rafraîchir en temps réel le thème de l'interface graphique lorsqu'il change.

> Hub

Tout comme WinJS, les contrôles `Hub` et `HubSection` ont été ajoutés pour faciliter la mise en place d'écran de Hub.

> ApplicationBar

Les `AppBar` introduisent deux propriétés, `PrimaryCommands` (commandes globales qui se placeront à droite de l'écran) et `SecondaryCommands` (commandes contextuelles à gauche de l'écran). Elles permettent de mieux gérer l'adaptation des `AppBar` à l'espace disponible pour l'application. Quand la place est suffisante les boutons sont présents avec leurs légendes, dans le cas contraire, les légendes disparaissent, voire même les `SecondaryCommands` disparaissent au profit des `PrimaryCommands`. On note aussi la création de différents types d'icônes pour les boutons des `AppBar`. Cela va grandement simplifier leurs mises en place. `SymbolIcon` permet de spécifier un nom pour une icône et non plus une valeur Unicode de la police Segoe UI Symbol. `FontIcon` permet d'afficher un glyph en spécifiant une police différente comme `Windings`. `BitmapIcon` vous laisse afficher une image, choisir la couleur de fond, comme sur Windows Phone. On peut aussi préciser directement un `Path` géométrique avec la propriété `PathIcon`.

> Media

Jusqu'ici, il était nécessaire de recoder l'interface graphique du lecteur de médias ou d'utiliser des bibliothèques telles que le `PlayerFramework`. Un nouveau contrôle `Media` permet de répondre simplement aux besoins les plus fréquents.

> Graphique

Depuis Windows 8.0 le mélange de DirectX et de XAML reposait sur un contrôle nommé

`SwapChainBackgroundPanel`. Ce contrôle devait être l'élément racine de l'arbre XAML. Grâce à `DirectComposition`, Windows 8.1 introduit le `SwapChainPanel` qui n'a plus cette limitation. Il peut être utilisé n'importe où dans votre IHM !

> RenderTargetBitmap

Derrière ce nom un peu obscur se cache une classe qui nous permet de faire un rendu sous forme de `Bitmap` d'un nœud de l'arbre visuel. A nous les captures d'écran !

> Performance

Diverses améliorations au niveau des performances ont été effectuées. Au démarrage de l'application, les ressources avec clefs (`StaticResource` nommées) et les thèmes (`ThemeResource`) sont chargés en différé. Le XAML est aussi compilé en binaire par défaut, à la manière de ce qui se fait en WPF. Le temps de chargement, la vitesse de défilement et l'empreinte mémoire des `List-`

`View` et des `GridView` ont aussi été améliorées de manière significative.

> Texte et gestion des entrées

Ceux qui développaient sur les autres couches XAML ont pu être décontenancés par l'absence du contrôle `Hyperlink`. Ce problème est maintenant résolu.

De nombreuses améliorations ont été apportées aux contrôles `TextBox` et `TextBlock`, notamment avec l'arrivée de la propriété `PlaceholderText` permettant d'afficher un texte par défaut lorsqu'une `TextBox` est vide.

On note aussi une amélioration des API de manipulation. On peut désormais déclencher des animations de défilement par code (ex : panning du contrôle `ScrollViewer`) ou avoir une prédiction de l'endroit où s'arrêtera l'animation dans le `ScrollViewer`.

Ce dernier est maintenant enrichi par des zones de défilement bloquées, qui offrent les mêmes possibilités que les « `FrozenColumns` » des `DataGrid`.

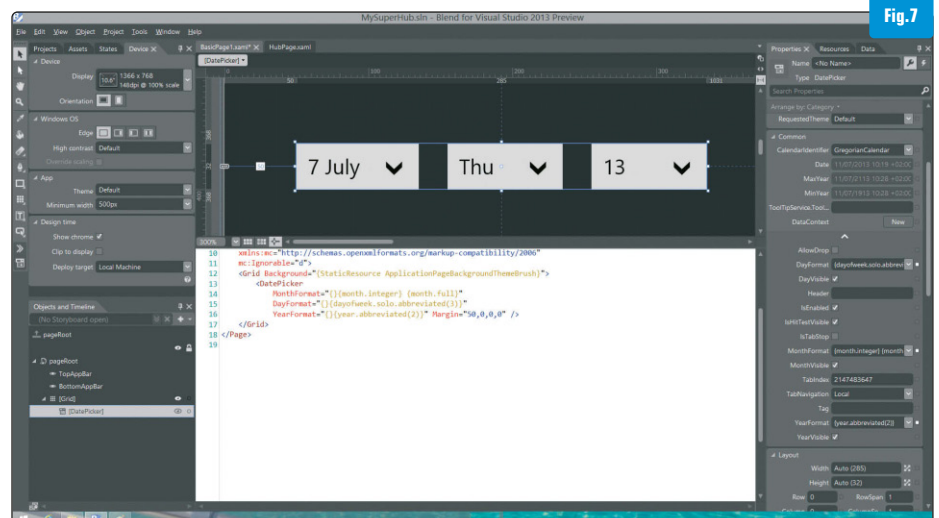


Fig.7

DatePicker

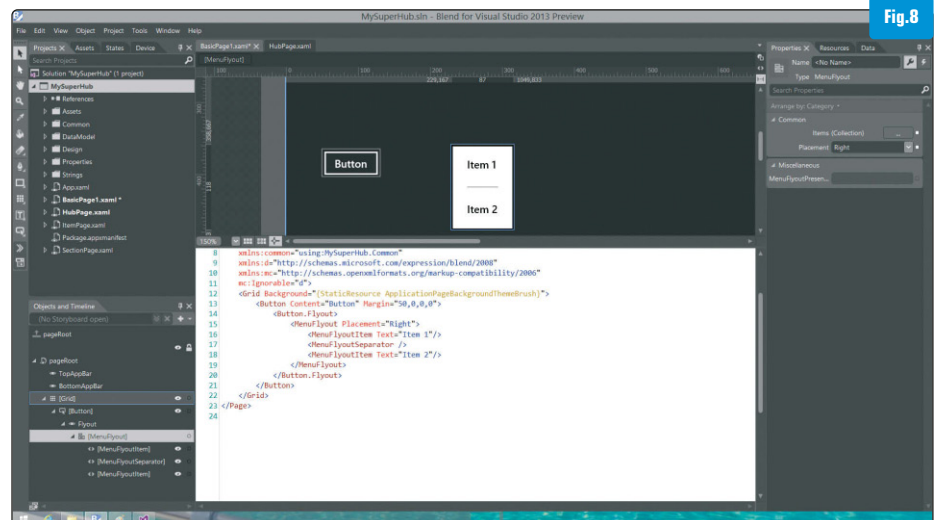


Fig.8

Flyout

> Binding

La performance du moteur de Binding a été améliorée jusqu'à 40%. On apprécie le retour de certaines options telles que TargetNullValue, FallbackValue et UpdateSourceTrigger qui manquaient jusqu'ici dans Windows 8. On peut de nouveau accéder par code à l'expression du Binding avec le rétablissement de la méthode GetBindingExpression. Et enfin, pour les utilisateurs de Blend, les Behaviors sont de retour !

LES OUTILS

Afin de suivre les évolutions précédemment présentés, Visual Studio et Blend ont aussi été mis à jour en version 2013. Les améliorations apportées diffèrent très largement selon votre choix de langage.

> Visual Studio

Outre une rapidité et une robustesse accrue du designer, cette version apporte de nombreuses améliorations dans l'éditeur. Les outils de diagnostic qui ont fait leur apparition au fil des mises à jour ont été améliorés, et intégrés sous un même menu « Performance and Diagnostics », accessible dans le menu « Debug ». Ces outils s'enrichissent également d'un petit nouveau qui permet de suivre la consommation batterie des applications.

> C#

Si vous développez en .NET vous aurez maintenant accès aux valeurs de retour des méthodes et au mode « edit and continue » pour les applications 64 bits.

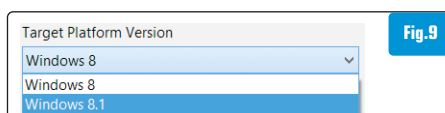


Fig.9

Migration vers Windows 8.1

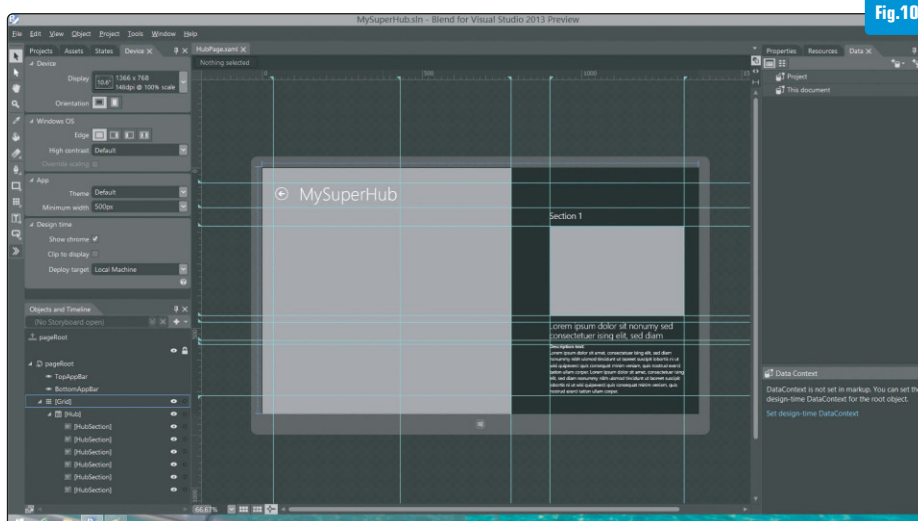


Fig.10

Blend

> C++

Si vous êtes plutôt développeur C++ l'éditeur de code a été amélioré avec, par exemple, le formatage du code ou l'auto-complétion des accolades.

> XAML

Les templates de projets ont été simplifiés et optimisés. Le concept de LayoutAwarePage (XAML) ou d'état visuel a disparu, car dans Windows 8.1 les pages doivent être de taille dynamique. En XAML, le changement de VisualState se fait donc à la main dans votre code en fonction de vos besoins. Le fichier StandardStyles.xaml a disparu au profit de la mécanique de thème. Cela fait donc moins de fichiers XAML à parser et, par extension, améliore les performances de démarrage.

Un « Go to definition » permet de naviguer rapidement vers la déclaration d'une ressource statique, et de nombreuses aides à la saisie ont été ajoutées. Le remplacement des balises ouvertes par des balises auto-fermées (et réciproquement), la modification automatique de la balise, la gestion de snippets de code XAML, l'auto-complétion sur les propriétés des objets liés par Binding, ou bien encore l'amélioration du support des commentaires en XAML (on peut maintenant commenter un groupe de lignes dont certaines sont elles-mêmes des commentaires).

> Blend [Fig.9]

On remarque le retour des Behaviors dans XAML et leur arrivée dans WinJS. Ils sont maintenant développés en natif pour de meilleures performances.

L'ajout des règles et des guides dans l'IHM facilite la mise en forme des interfaces. Elles

peuvent être utilisées sur de multiples pages et s'adaptent aux résolutions d'écrans.

Le panneau « Device » a été mis à jour pour supporter la prévisualisation des différents modes d'affichages de Windows 8.1.

On remarquera aussi le grand retour du panneau « Data » en XAML, qui simplifie la mise en place de sources de données. Un support amélioré de l'édition des barres d'application et des boutons associés est maintenant présent. Les développeurs WinJS sauront apprécier l'ajout d'un éditeur pour les animations à base de KeyFrame à la manière de ce qui existait déjà en XAML.

Et le meilleur pour la fin, avec l'ajout de l'édition des styles et templates en place dans la page. Auparavant dans Blend, lorsque l'on éditait un template de contrôle, on arrivait dans une autre page en perdant le contexte des données. A présent, l'édition se produit et toute modification du template se répercute automatiquement sur les éléments qui l'utilisent. C'est probablement la fonctionnalité qui permettra de gagner le plus de temps lors de la création d'application en XAML.

> Migration d'un projet Windows 8.0 vers Windows 8.1

Migrer un projet Windows 8.0 est très facile. Dans les propriétés du projet et dans l'onglet application il suffit de passer changer la version cible en 8.1 [Figure 8]. Dans la plupart des cas la migration se passera sans souci avec l'assistant de migration. Néanmoins il se peut que quelques corrections soient nécessaires. Avant la migration, Visual Studio vous donnera un lien vers la documentation msdn pour gérer ces problèmes [Fig.10].

CONCLUSION

Cette version reste une préversion, on constate donc encore quelques bugs dans les API et Visual Studio. Elle permet de commencer à prendre la main sur toutes les nouveautés proposées en attendant la version définitive qui arrivera après l'été.

Pour aller plus loin, les exemples de code et la documentation ont été mis à jour sur <http://dev.windows.com>, et les sessions de la Build détaillant ces nouvelles fonctionnalités sont disponibles sur le site <http://www.buildwindows.com>.

Guillaume **Leborgne**, Architecte / Chef de projet MCNEXT

John **Thiriet**, Consultant / Formateur MVP, Client Development MCNEXT

Windows 8.1 : du neuf pour le C++

Avec Windows 8, Microsoft a effectué un important travail afin de faire revenir C++ dans le monde des langages modernes, et faire en sorte que tout ce qui est possible avec les langages .Net le soit aussi en C++.

Les développeurs de Visual C++ et Windows ne se sont pas arrêtés là, et ont ainsi permis aux développeurs C++ d'accéder en plus des API WinRT, à de nombreuses API de plus bas niveau, qui n'ont pas pu être portées en WinRT avec le degré de performance et de simplicité requis. On peut ainsi accéder à des bibliothèques comme Direct3D 11, Xaudio 2, C++ AMP, ainsi qu'à des SDK C++ third party (notamment dans le monde du jeu vidéo) et à des API spécialisées. Certaines API WinRT peuvent même interopérer avec ces API natives pour permettre des scénarios intéressants (interop DirectX + Xaml, interop MediaElement / video html5 + MediaFoundation par exemple).

L'ouverture exclusive à C++ de ces scénarios, ainsi que les efforts d'évangélisation de la part de Microsoft pour valoriser le développement natif montrent une véritable volonté de pousser ce développement, et c'est tout naturellement que Windows 8.1 et Visual Studio 2013 continuent sur cette lancée. Les nouveautés apportées sont de quatre types : implémentation des dernières spécifications du standard C++ (C99, C++ 11, voire un peu de C++ 14), performances du code généré, nouvelles API, et amélioration de l'expérience de développement et de débogage.

> Standard C++11

Depuis Visual Studio 2010, Microsoft a fait évoluer son compilateur pour, petit à petit, implémenter les fonctionnalités du langage et de la bibliothèque standard C++ décrites dans la version 11. Visual C++ 2012 couvrait une très large partie de ces fonctionnalités, et Visual C++ 2013 vient colmater la plupart des brèches encore existantes, tout en initiant le travail d'adoption du futur C++ 14.

Ainsi on trouvera comme nouveautés :

- Arguments par défaut pour les templates de fonctions : Précédemment, seules les classes pouvaient spécifier un type par défaut pour leurs paramètres.
- Délégation de constructeur : Il est désormais possible d'appeler un autre constructeur de la même classe

dans la liste d'initialisation d'un constructeur.

- Liste d'initialisation et initialisation uniforme : permet d'initialiser une instance d'une classe avec un nombre variable de paramètres. Les conteneurs de la STL supportent les listes d'initialisation et il est aussi possible d'écrire nos propres types consommant ces listes. De plus, l'initialisation uniforme (utilisant les accolades plutôt que les parenthèses même pour des constructeurs classiques), permet de résoudre quelques cas obscurs d'ambiguïté de déclaration.
- Raw String Literals : fonctionnalité semblable à l'écriture de strings constant C# utilisant le préfixe '@'. La syntaxe est légèrement différente mais la fonctionnalité est équivalente.
- Variadic Templates : Il s'agit de la possibilité de déclarer des templates de classes ou de fonctions prenant un nombre variable de paramètres. L'implémentation de `std::make_shared` de visual C++ 2013 utilise par exemple cette fonctionnalité.

> Performance

- L'auto-vectorizer (brique du compilateur introduite avec VS 2012 permettant d'exploiter les instructions SSE/AVX des processeurs sans avoir à écrire du code « vectoriel », reconnaît beaucoup plus de pattern de code.
- La compilation pour les plateformes ARM et ATOM a été largement améliorée
- Les Profile Guided Optimizations sont désormais possibles pour les applications Windows Store.
- C++ AMP est désormais capable de détecter les systèmes utilisant de la mémoire partagée entre le CPU et le GPU (chipsets intégrés Intel et AMD, Xenon Phi, Xbox One), et évite donc des recopies de données inutiles quand c'est possible.

> Nouvelles API

- La STL a été mise à jour pour supporter toutes les nouveautés de C++11
- Certaines fonctionnalités de la STL C++ 14 ont aussi été implémentées (functors transparents, `make_unique`, fonctions non

membres de récupération d'itérateurs...)

- Les tâches PPL ont été mises à jour, avec de nouvelles factories, et de nouveaux schedulers.
- Le C++ REST SDK est distribué avec Visual Studio (et a été mis à jour avec la dernière version de PPL)

> Expérience de développement debugging

- Pour les applications Windows 8.1, le debugger capture la stack à chaque déclenchement d'opération asynchrone. Cela permet de revenir facilement et d'accéder au contexte d'initialisation de l'opération lorsqu'on se trouve dans une callback.
- Pour les applications Windows 8.1, les exceptions C++ / CX capturent leur stack trace, et un message personnalisé. Cela simplifie grandement la correction de bugs.
- Le debugger supporte désormais le debugging mixte Javascript/C++.

> Conclusion

Toutes ces nouveautés viennent combler quelques manques importants (le support du debugging asynchrone est par exemple quelque chose de très attendu), et solidifier l'expérience de développement natif. Microsoft poursuit également son effort d'adoption du C++ standard (et est d'ailleurs un des principaux moteurs du comité de standardisation). Cette approche « standard » nous permet non seulement de bénéficier de ce que le langage peut offrir pour la plateforme Windows, mais aussi d'écrire du code multi-plateforme et performant. Avec les nouveaux cycles de développement plus courts adoptés par les équipes Windows et Visual Studio, on peut s'attendre à voir apparaître des mises à jour du compilateur et des bibliothèques, qui feront de Visual Studio un des environnements de développement les plus proche des standards les plus récents.

Simon Ferquel
Infinite Square

<http://blogs.infinitesquare.com/b/simon>



Windows 8.1 et XAML

XAML est maintenant une technologie bien ancrée dans le paysage Microsoft et on pourrait supposer que les évolutions ne sont plus trop au rendez-vous au fur et à mesure que la plateforme devient mature. Nous aurions bien tort et Microsoft nous le prouve avec cette mise à jour du SDK pour les Windows Store Apps en apportant beaucoup de nouveautés : nouveaux contrôles, évolutions des contrôles existants, mais aussi quelques évolutions de la plateforme elle-même. Le développeur XAML qui sommeille en vous ne va pas être déçu.

LES NOUVEAUX CONTRÔLES

Du côté des nouveaux contrôles, Microsoft a bien entendu le message des développeurs et nous propose maintenant des contrôles très utiles apportant la parité avec WinJS. Beaucoup d'entre eux étaient disponibles sous la forme de package Nuget (WinRTXamlToolkit, Babylon Toolkit, etc.) et sont maintenant intégrés au framework.

> DatePicker et TimePicker

Ces deux contrôles présents dans WinJS depuis le début font partie de ceux dont on remarquait le plus l'absence dans Windows 8. Dorénavant inclus dans le framework 8.1, ils permettent à l'utilisateur de renseigner une date ou un horaire. Ces deux contrôles sont optimisés pour un usage à la souris, au clavier et bien sûr au tactile ! Pour en utiliser un, rien de plus simple, il suffit de les déclarer dans votre XAML :

```
<DatePicker x:Name=monDatePicker Header="Date des vacances"/>
<TimePicker x:Name=monTimePicker Header="Heure des vacances"/>
```

Les deux contrôles proposent une propriété Header permettant de renseigner le texte à afficher à l'utilisateur. Vous pouvez retrouver les valeurs sélectionnées par un Binding classique ou via les propriétés Date ou Time dans votre code-behind.

Ils sont génériques, mais vous pouvez les personnaliser selon votre besoin : année minimum et maximum (MinYear et MaxYear), format d'affichage (DayFormat, etc.), affichage de l'année ou non (IsYearVisible) et bien d'autres choses encore.

> SettingsFlyout

Enfin, il est arrivé ! Les panneaux de configuration sont courants et presque toutes les applications en possèdent un. Ils sont maintenant présents dans le framework sous la forme du contrôle SettingsFlyout. Ce contrôle possède une propriété Title pour indiquer le titre, et une propriété IconSource permettant de personnaliser l'icône affichée. C'est ensuite à vous de choisir le contenu à afficher en plaçant vos contrôles en son sein.

```
<SettingsFlyout
    x:Class="InfiniteSquare.MonFlyout"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    IconSource="Assets/Logo.png"
    Title="Mon Settings Flyout"
```

```
Width="346"
HeaderBackground="#FF2B4A80">
<TextBlock Text="Infinite Square"/>
</SettingsFlyout>
```

Pour l'afficher, il faut appeler la méthode « Show » ou « ShowIndependent » d'une instance de Flyout. La première affichera le panneau de settings lors de la fermeture du Flyout, la seconde non.

```
new InfiniteSquare.MonFlyout().ShowIndependent();
```

> Flyout et MenuFlyout

La propriété de dépendance attachée AttachedFlyout vous permet maintenant d'attacher un Flyout à chaque élément dérivant de FrameworkElement. Vous devez fournir un contrôle Flyout contenant l'interface à afficher en son sein. Vous pouvez aussi utiliser la version spécialisée MenuFlyout se comportant de la même façon qu'un menu contextuel.

Charge à vous d'ouvrir le Flyout en fonction de l'interaction utilisateur reçue (Tap, etc.), avec la méthode ShowAttachedFlyout de FlyoutBase (prenant l'élément où est défini le Flyout en paramètre).

Les boutons définissent la propriété « Flyout » permettant de définir un Flyout ouvert automatiquement au Tap de l'utilisateur.

> Hub

Si vous ne l'avez jamais testée, je vous conseille de lancer l'application « Bing Voyage ». Elle affiche une grande image sur sa page d'accueil et divise le reste de la page en différentes sections. Ce type d'affichage est tellement apprécié que le contrôle Hub du framework 8.1 vous permet de le reproduire très facilement.

Un Hub est défini par un Header principal, et plusieurs HubSection enfants. Le header restera visible quelle que soit la position où l'on se trouve dans la page et chaque section peut afficher un contenu spécifique provenant d'une source de données indépendante de ses « frères ». C'est cette différence qui distingue notamment un Hub d'une Gridview groupée.

```
<Hub Header="MonBeauHub">
    <HubSection MinWidth="600" Header="Roi des contrôles">
        <!--Contenu -->
    </HubSection>
    <HubSection MinWidth="600" Header="Que j'aime tes données">
        <!--Contenu -->
    </HubSection>
</Hub>
```

Il s'agit d'un exemple évidemment simpliste car rentrer en détail sur cet élément dépasse le périmètre de cet article, mais il faut retenir que c'est un contrôle très souple, permettant de construire rapidement une interface Windows Store à laquelle est habitué l'utilisateur.

> CommandBar et boutons « AppBar »

Pour mémoire, il fallait sous Windows 8 créer soi-même l'affichage de l'AppBar avec des contrôles, boutons et Panel « classiques ». Maintenant il est possible d'utiliser un Panel de type « CommandBar » qui va se charger pour vous d'afficher les boutons et de se redimensionner avec l'application. Par défaut, les boutons sont affichés dans la partie « général » à droite mais vous pouvez en ajouter à droite en utilisant sa propriété « SecondaryCommands ».

Cette CommandBar prend en charge des éléments de différents types : AppBarButton, AppBarSeparator, et AppBarToggleButton. Ces nouveaux éléments permettent d'afficher des boutons d'appbar avec le cercle entourant le contenu ou encore un séparateur. Chacun de ces éléments possède une propriété IsCompact indiquant si le label doit être caché et les marges réduites. C'est cette propriété qui est utilisée par la CommandBar. Lorsque vous utilisez un de ces éléments en dehors de celle-ci, c'est à vous de choisir le mode approprié (compact ou normal).

Auparavant, les icônes étaient en général fournies au moyen de la police Segoe UI Symbol et il était difficile d'y placer son propre contenu. Dorénavant, vous avez 4 manières de fournir l'icône via la propriété Icon : une icône utilisant une police(FontIcon), une image(BitmapIcon), un Path XAML(PathIcon) ou encore une icône utilisant spécifiquement SegoeUI Symbol (SymbolIcon - une spécialisation de la première).

La création d'une ApplicationBar est alors un jeu d'enfant :

```
<Page.BottomAppBar>
  <CommandBar>
    <AppBarButton Icon="Like" Label="J'aime le code"
    <AppBarSeparator/>
    <AppBarToggleButton Icon="Shuffle" Label="mélanger" Click="Click">
      <AppBarToggleButton.Icon>
        <BitmapIcon UriSource="ms-appx:///Assets/infiniteSquare.png"/>
      </AppBarToggleButton.Icon>
    </AppBarToggleButton>
    <CommandBar.SecondaryCommands>
      <AppBarButton Icon="Dislike" Label="Pas bon" />
    </CommandBar.SecondaryCommands>
  </CommandBar>
</Page.BottomAppBar>
```

> HyperLink

Un ajout tout simple mais bien pratique : il est maintenant possible d'ajouter des liens hypertextes dans les TextBlock et RichTextBlock ! Il suffit de l'utiliser dans un paragraphe comme on peut le faire avec des éléments Run.

LES AUTRES NOUVEAUTÉS

Il n'y a pas que des nouveaux contrôles mais aussi des améliorations de la plateforme XAML plus généralement.

> ScrollView

Le ScrollView expose maintenant des propriétés (TopHeader, LeftHeader, et TopLeftHeader), vous permettant de définir des endroits « gelés » dans celui-ci. Cela fonctionne de la même manière que les lignes « gelées » d'Excel. Il peut donc être très intéressant pour ajouter des informations fixes à l'utilisateur.

> Système de thèmes

Sur Windows 8, vous deviez choisir le thème une fois pour toutes au lancement de l'application. Ce n'est maintenant plus nécessaire, le framework va détecter et appliquer les changements de thèmes.

De plus, chaque objet dérivant de FrameworkElement possède une propriété RequestedTheme permettant de définir le thème du contrôle ainsi que celui de ses enfants. Il est donc tout à fait possible d'imbriquer différents thèmes.

Ce système repose sur l'utilisation de markup extension « ThemeResource » qui se charge de s'abonner au changement de thèmes et de les appliquer. Les contrôles du framework utilisent bien sûr cette balise et il est tout à fait possible de l'utiliser aussi dans vos propres contrôles.

```
<TextBlock Foreground="{ThemeResource InfiniteSquareColor}" />
```

> Amélioration du système de Bindings

On ne peut pas imaginer la plateforme XAML sans le système de binding qui évolue aussi dans cette mouture pour apporter des fonctionnalités déjà connues des développeurs WPF ou Silverlight :

- L'événement DataContextChanged des FrameworkElement vous indique lorsque le contexte de données de votre élément est mis à jour.
- Les options TargetNullValue et FallBackValue des Bindings vous permettent d'indiquer une valeur par défaut si l'objet source du binding est null ou si le binding échoue.
- Il est possible de spécifier l'UpdateSourceTrigger d'un binding pour la valeur PropertyChanged. Cette option est très pratique et manquait cruellement pour les TextBox dont les bindings sur la propriété Text n'étaient mis à jour que sur le changement de Focus.

CONCLUSION

Microsoft a entendu les demandes des développeurs sur XAML qui évolue dans le bon sens.

Nous avons vu que des nouveaux contrôles vont nous permettre de gagner énormément en productivité tout en fournissant une ergonomie et une expérience utilisateur exceptionnelle.

De plus, les anciens composants évoluent aussi pour s'adapter au mieux à l'usage que les développeurs de la plateforme en font. Il ne reste plus qu'à s'emparer de cette nouvelle version !

Jonathan **Antoine** - Consultant

<http://blogs.infinitiesquare.com/b/jonathan>

Maxime **Frappat** - Consultant

<http://blogs.infinitiesquare.com/b/mfrappat>

Prochain numéro
parution 28 septembre 2013

● **Google Maps**

● **Unity3D**

Le meilleur moteur actuel !

● **HTML 5**

par la pratique

Windows 8.1 : WinJS 2.0 arrive

WinJS 1.0, ou la Windows Library pour Javascript, propose des contrôles comme la Listview ou l'AppBar pour Windows 8. Certains développeurs font état de mauvaises performances et d'un manque de personnalisation de certains contrôles... mais Windows 8.1 arrive, et avec lui, WinJS 2.0 ! Cette version apporte sa liste de nouveaux contrôles pour Windows 8.1 mais aussi de nombreuses optimisations sur ceux qui existent déjà, de quoi faire des heureux chez les développeurs et les utilisateurs !

> L'amélioration des contrôles existants

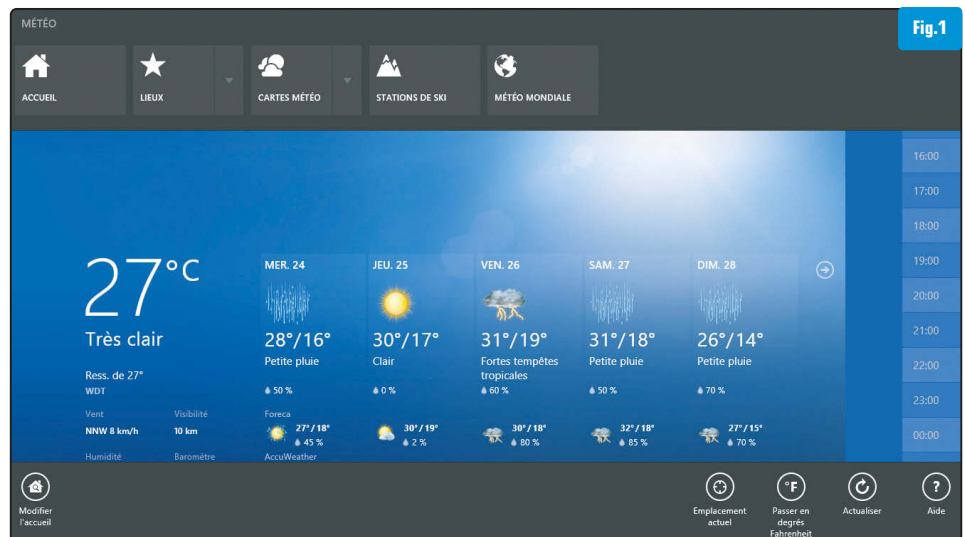
• La Listview :

Contrôle de base dans les applications Windows 8, la listview subit un petit nettoyage et gagne jusqu'à 30% en performance sur Windows 8.1. Jusqu'alors réservé au C#, le **drag-and-drop** est maintenant aussi disponible sur la listview en WinJS et permet d'y déplacer n'importe quel item sur la page et de réordonner les éléments au sein d'une listview. Au niveau des layouts, il est maintenant plus facile de créer des éléments de tailles différentes avec l'ajout du **CellspanningLayout** qui permet de spécifier rapidement la taille d'un item sans avoir à créer un nouveau template ou contrôle personnalisé.

```
<!--Déclaration de la listview -->
<div id="listView" class="win-selectionstylefilled"
  data-win-control="WinJS.UI.ListView"
  data-win-options="{
    itemDataSource: myData.dataSource,
    selectionMode: 'none',
    itemTemplate: smallListItemIconTemplate,
    itemsDraggable: true,
    itemsReorderable: true,
    layout: { type: WinJS.UI.GridLayout }}">
</div>
```

• L'**AppBar** : le contrôle est maintenant entièrement personnalisable. L'appbar peut avoir du contenu personnalisé comme des images ou d'autres contrôles et est devenu accessible avec une navigation au clavier.

```
<div id="AppBar" data-win-control="WinJS.UI.AppBar">
  <button data-win-control="WinJS.UI.AppBarCommand" data-win-
    options="{id:'cmdRemove',label:'Remove',icon:'remove',section
    : 'global',tooltip:'Remove item'}"></button>
  <hr data-win-control="WinJS.UI.AppBarCommand" data-win-
    options="{type:'separator',section:'global'}" />
  <button data-win-control="WinJS.UI.AppBarCommand" data-win-
    options="{id:'cmdDelete',label:'Delete',icon:'delete',section
```



La Navbar sur l'application Météo.

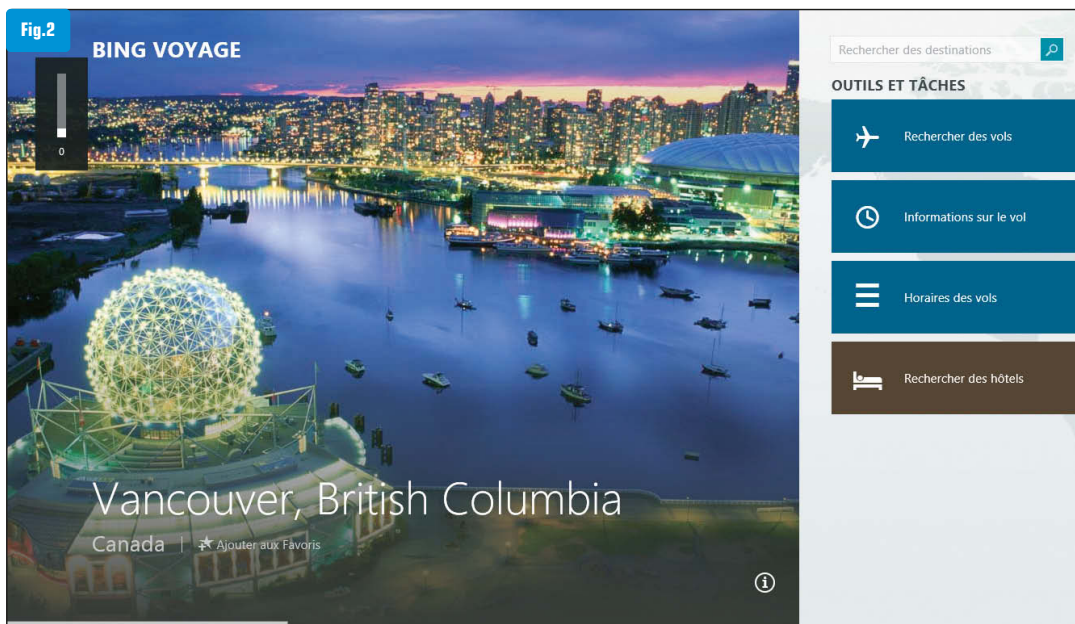
```
: 'global', tooltip: 'Delete item' } }></button>
</div>
```

> Les nouveaux contrôles

• Le **Hub** : grande nouveauté de Windows 8.1, le Hub Control va permettre de rendre vos pages plus dynamiques en organisant plus facilement et plus esthétiquement vos collections de données. Comparable à une Gridview, le Hub supporte le SemanticZoom et offre la possibilité de disposer plusieurs groupes d'éléments (HubSection) sur une même page mais en utilisant des templates de navigation prédéfinis différents. Le Hub est déjà disponible dans l'application Voyage et le Windows Store de Windows 8.1.

```
<div class="hubpage fragment">
  <section class="hub" data-win-control="WinJS.UI.Hub">
    <div class="section1" data-win-control="WinJS.UI.HubSection"
      data-win-res="{ winControl: { 'header': 'Section1' } }"><!-- 1er
      HubSection du Hub --></div>
    <div class="section2" data-win-control="WinJS.UI.HubSection"
      data-win-res="{ winControl: { 'header': 'Section2' } }"><!-- 2ème
      HubSection du Hub --></div>
    </section>
  </div>
```

• La **Navigation Bar** est un nouveau contrôle issu de l'AppBar vous permettant de fournir une liste de navigation organisée pour l'utilisa-



L'application Voyages est basée sur un hub et propose une searchbox directement sur la page d'accueil.

teur. Affichée par un swipe sur le haut de l'écran, elle permet d'afficher une simple liste de liens ou des liens classés par catégories [Fig.1].

- Le **Search Box** : auparavant interdite dans les pages des applications Windows 8, le champ de recherche peut finalement être mis en avant directement dans l'application avec un contrôle dédié [Fig.2].

```
<div data-win-control=»WinJS.UI.SearchBox»></div>
```

- Le **Back Button** est maintenant entièrement intégré dans WinJS 2.0, le contrôle prend en charge automatiquement la navigation sur la page précédente lorsque l'utilisateur clique sur le bouton et se désactive de lui-même s'il n'y a aucune page sur laquelle naviguer.

```
<button data-win-control=»WinJS.UI.BackButton»></button>
```

> L'infrastructure

- Le **Scheduler** est un nouveau moyen pour le développeur de gérer les priorités sur les différentes tâches asynchrones de son application. En effet, sous Windows 8 toutes les tâches sont exécutées immédiatement à leurs appels et il est impossible de définir des priorités d'exécution sur des tâches qui se chevauchent. Par exemple, suite à une interaction sur l'UI, une animation doit pouvoir s'exécuter tout de suite contrairement à une tâche de fond qui n'a pas besoin de retour visuel immédiat. Le Scheduler intervient alors ici pour définir en priorité les tâches à exécuter (animations, retour visuels...) de ce qui peut être exécuté plus tard (tâche de fond, traitement de données...).

```
// Définir la priorité d'une tâche
var task1 = S.schedule(function () {
    window.output(«Tâche 1»);
}, S.Priority.normal);
```

- Le **Dispose model** va vous permettre de libérer vous-même les ressources qui ne sont plus utilisées et ainsi éviter tout débordement de

mémoire. Pour l'utiliser, il suffit d'appeler la méthode `Dispose()` quand le contrôle n'est plus utilisé et ainsi appeler le garbage collector.

> Les autres contrôles

- Le **Repeater** : utilisé pour représenter des listes d'éléments, il permet comme son nom l'indique, de répéter plusieurs fois sur une page un même type d'objets en utilisant un même template pour chacun d'entre eux. Le Repeater est rapidement assimilable à l'`ItemsControl` en C# et comme lui, son usage ressemble à celui d'une `Listview`. Cependant le

Repeater, comme l'`ItemsControl`, est plus léger et ne possède pas toutes les fonctions plus avancées de la `Listview` comme les différentes fonctions de chargement et de sélection d'un élément d'une liste.

- L'**ItemContainer** est un nouveau contrôle totalement flexible permettant de générer une liste d'éléments en dehors d'une `Listview` tout en intégrant des fonctions comme le drag-and-drop ou le swipe.

```
<div id=»monItemContainer»
  data-win-control=»WinJS.UI.ItemContainer»
  data-win-options=»{tapBehavior: 'toggleSelect'}»>
  <div style=»margin: 10px»>
    <div class=»win-type-x-large»>
      MonImage
    </div>
    <img id=»uneImage» src=»/image.png»>
  </div>
</div>
```

- La **Webview** est maintenant disponible et permet d'afficher tout le code HTML5 qu'une `iFrame` ne pouvait pas afficher.

```
<x-ms-webview id=»webview» src=»http://www.programmez.com
  style=»width: 400px; height: 400px;»>
</x-ms-webview>
```

> Conclusion

WinJS 2.0 est une vraie révolution pour Windows 8.1. En plus de l'optimisation des anciens contrôles Windows 8, l'apport d'un grand nombre de nouveaux contrôles permettra aux développeurs et utilisateurs de découvrir et créer de nouveaux templates d'applications qui sauront rendre Windows 8.1 plus attractif que son prédécesseur.

Soriya Thach
<http://blog.soriyathach.fr>



Les nouveautés Visual Studio 2013 et Team Foundation Server 2013

Visual Studio 2013 et Team Foundation Studio 2013 gardent le design de Visual Studio 2012 et Team Foundation Server 2012. Parmi plusieurs thèmes : Framework/OS, Productivité, Agilité, nous avons choisi certaines nouveautés qui font de ces outils des versions indispensables. Tout ce que vous trouverez dans cet article est basé sur ce qui a été présenté pendant la //Build/, c'est-à-dire une version dite « Preview » des outils : tout n'est pas finalisé ou inclus dans ces versions.

FRAMEWORK & OS

> Framework .net 4.5.1

Cette version est, comme le Framework 4.5, une « in place update ». Il n'y a pas, à proprement parler, de nouveautés importantes côté API, mais pas mal de choses ont changé sous le capot afin de nous apporter plus de productivité. En fait, la plupart des améliorations du point de vue de la productivité sont liées à des changements du framework. Au rayon des améliorations purement framework, on peut noter :

- Des améliorations de performances du côté du JIT,
- La compression du LOH (Large Object Heap),
- Moins de dégradation des performances lors du lancement d'applications .Net après la mise à jour du Framework (nouvelles versions ou patch de sécurité).

> Windows 8.1

Décrire toutes les fonctionnalités de Windows 8.1 pourrait faire l'objet d'un magazine

entier et ce n'est pas notre ambition ici. Il est important de savoir que Visual Studio 2013 est requis pour développer pour Windows 8.1. Cependant, pour continuer à développer pour Windows 8, il vous faudra toujours Visual Studio 2012.

> ASP.Net

ASP.Net n'a pas été oublié ! Le développement web est unifié : plus de ASP.Net WebForm d'un côté et MVC d'un autre [Fig.1]. Il n'y a plus qu'un seul modèle [Fig.2]. A vous de choisir votre technologie d'affichage. Au niveau de Visual Studio cela se traduit par un seul projet web et un assistant pour choisir les technologies web qui vous intéressent [Fig.3]. Rien n'est exclusif, vous pouvez choisir par exemple WebForms avec WebAPI ! Avec Visual Studio 2013 arrive ASP.Net MVC 5, ainsi qu'un mode de suspension d'application qui permet de sauvegarder l'état d'une application web plutôt que de la redémarrer en cas de non utilisation. Cette fonctionnalité n'est disponible pour l'instant que sous Windows Server 2012 R2 Preview

et devrait offrir jusqu'à 90% de temps de réponse en moins lors d'une connexion à une application suspendue.

PRODUCTIVITÉ

> Couleurs !

La couleur est de retour dans Visual Studio ! Les équipes de développement ont réintégré petit à petit de la couleur dans l'interface. Juste ce qu'il faut pour la rendre plus lisible [Fig.4]. Bien qu'anecdotique, le thème « blue » est intégré dans Visual Studio 2013.

> Async debug

L'arrivée du framework 4.5 a permis l'utilisation de méthodes asynchrones grâce aux mots clés `async` et `await`. Même si le débogueur de Visual Studio 2012 permet de faire du pas à pas comme en mode synchrone, nous ne disposions pas de la pile d'appels complète, ce qui n'est pas très instructif pour se situer dans le contexte d'exécution. Ce problème est désormais résolu avec l'arrivée de Visual Studio 2013 et Windows 8.1. En effet, on peut maintenant connaître

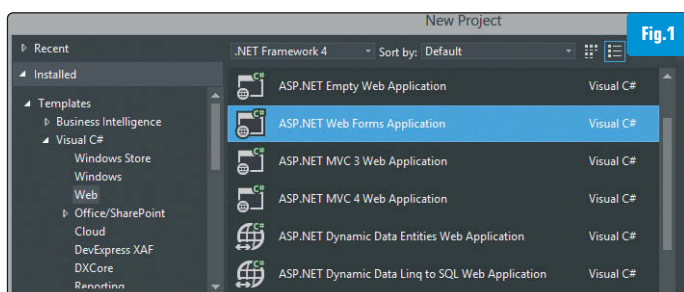


Fig.1

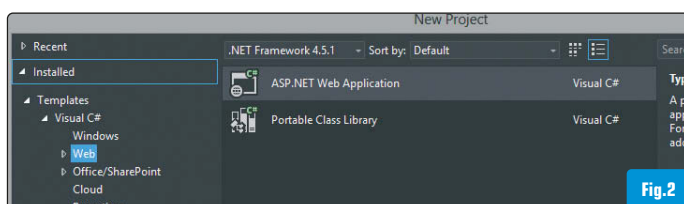


Fig.2

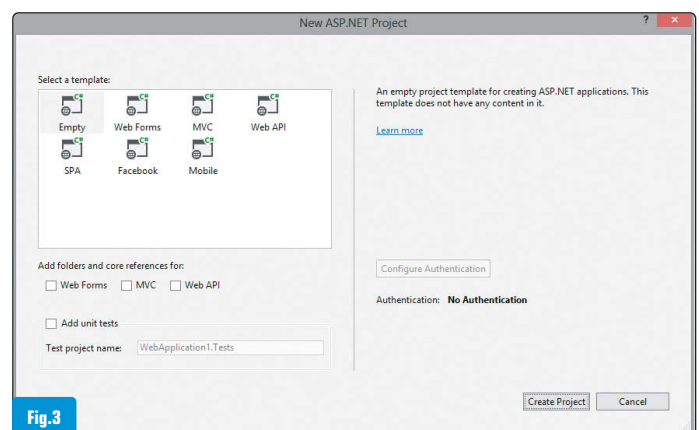


Fig.3

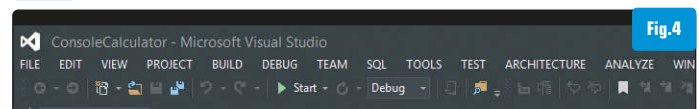


Fig.4

l'origine de l'appel de la méthode asynchrone directement dans la pile des appels (Call Stack) ainsi que dans la fenêtre de suivi des tâches (Task).

> Fenêtre des options

Enfin la fenêtre des options est redimensionnable ! Certes, cela ne va pas augmenter votre productivité, mais cela méritait d'être souligné. Les équipes de Visual Studio sont mobilisées pour que Visual Studio 2013 fonctionne parfaitement quelle que soit la résolution de vos écrans : les propriétaires de Surface Pro apprécieront.

La vraie nouveauté réside dans la possibilité de centraliser les paramètres de Visual Studio 2013 via votre compte Microsoft. Un gain de temps appréciable lorsqu'on change de poste de travail.

> Barre de défilement

La barre de défilement, que l'on pouvait trouver en addin, est intégrée directement à l'IDE.

Pour ceux qui n'en connaissent pas le principe, cette barre permet d'afficher un aperçu du fichier complet. En survolant une zone avec la souris cela affiche le code correspondant. Si vous désirez vous rendre à cet endroit il vous suffit alors de cliquer sur cette zone. Vous constaterez que cela vous permet de naviguer très rapidement au sein d'un fichier [Fig.5].

En revanche, celle-ci n'est pas activée par défaut. Pour cela il suffit de se rendre dans les options à partir du menu « OUTILS OPTIONS... », puis de trouver l'option en question sous : « Editeur de texte/Tous les langages/Barres de défilement ». Pour l'activer il vous suffira de choisir dans la section comportement : « Utiliser le mode mappage pour la barre de défilement verticale » puis de sélectionner la largeur de cette dernière

via la combobox « Vue d'ensemble du code source ».

Vous avez également la possibilité de choisir les éléments à afficher dans la barre :

- Les modifications
- Les marques (points d'arrêt, signets...)
- Les erreurs
- Le signe insertion (votre curseur)

> Indicateurs d'informations de code (Code Lens)

Le principe de Code Lens est très simple : afficher des metadata sur les méthodes directement dans le code source [Fig.6]. Visual Studio 2013 va vous permettre d'accéder à différents indicateurs depuis l'éditeur de texte. Il est important de rappeler que ces données ne sont pas contenues dans le fichier de code source, mais bien gérées par l'IDE. Code Lens vous permet d'afficher jusqu'à cinq indicateurs :

- Etat du test (résultat des tests unitaires),
- Références (appels de cette méthode),
- Testé par (personne ayant effectué le test),
- Auteurs (personnes ayant édité cette méthode),
- Modification (ChangeSet associés).

Vous comprendrez que certains indicateurs nécessitent l'utilisation d'un contrôleur de code source, TFS 2013 en l'occurrence. La sélection de ces indicateurs s'effectue dans les options (OUTILS OPTIONS...) sous l'arborescence « Editeur de texte/Tous les langages/Indicateurs d'informations de code ».

Edit & Continue 64bit

Le mode « Edit & Continue » vous permet d'éditer le code source directement lorsque vous débutez votre application avec Visual Studio. Les modifications sont alors prises en compte sans avoir à relancer l'application. Ce mode vous permet donc de gagner énormément de temps lorsque vous débutez

car vous n'avez pas à recompiler et rejouer votre use case, ce qui peut s'avérer long avec un certain niveau de complexité.

Cette fonctionnalité a été introduite avec la version 2005 de Visual Studio mais seulement pour le mode 32 bit, Visual Studio 2013 vient enfin combler le manque avec le support du mode 64 bit. Le fonctionnement reste identique au mode 32 bit.

Attention, les applications Web nécessitent d'activer cette option dans les propriétés du projet, dans l'onglet « Web » puis l'option « Activer Modifier Continuer ».

> Peek Definition

Nouveauté intéressante dans l'éditeur de texte, le Peek Definition vous permet d'incruster l'affichage d'un fichier au sein d'un autre fichier de code source. Aussi surprenant que cela puisse paraître, vous vous retrouvez avec un onglet d'aperçu au milieu de votre code source. Comme son nom l'indique, on navigue vers la définition de l'élément sélectionné et l'onglet est incrusté à la ligne suivante [Fig.7]. Vous pouvez évidemment promouvoir cet onglet. Vous avez également la possibilité de naviguer de nouveau au sein de l'onglet incrusté, ceci ne créera pas de nouvel onglet d'aperçu mais naviguera directement, tout en conservant votre pile de navigation, ce qui vous permettra de revenir en arrière. On notera toutefois la limite à un seul Peek Definition par fichier.

Le raccourci clavier est « Alt+F12 », mais vous pouvez également vous servir du menu contextuel.

> Carte du code (Code Map)

Ceci n'est pas à proprement parler une nouveauté car elle est apparue avec les mises à jour de Visual Studio 2012, mais elle est restée tellement confidentielle qu'il est important d'en parler. Il s'agit d'un diagram-



Fig.5

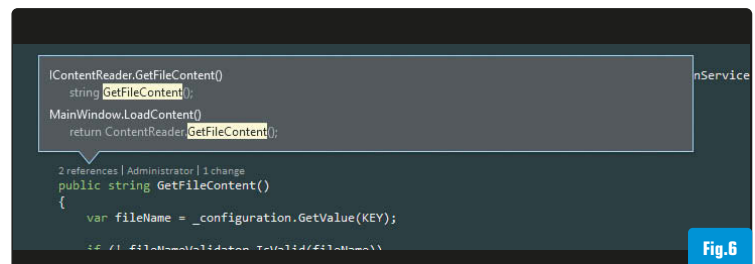


Fig.6

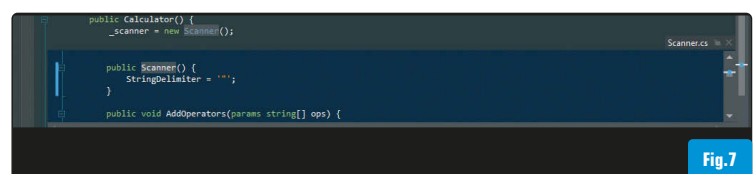


Fig.7

me permettant de cartographier votre code source, vous allez pouvoir visualiser les différentes relations entre les types, méthodes, champs... Le diagramme se construit au fil de votre navigation et non en une seule fois comme le fait le diagramme de classe [Fig.8]. Cet outil graphique est très réussi et très intuitif et a l'énorme avantage d'être intégré à l'IDE. Le résultat est bien plus pratique que d'utiliser la recherche de références habituelles. Ici, vous avez la possibilité de visualiser des références en chaîne, n'ayons pas peur de le dire : en mode plat de spaghetti, entre les fonctions, mais également les champs utilisés, voire partagés par plusieurs fonctions.

Si nous nous arrêtons ici, le bilan serait déjà largement positif. Mais l'équipe de Visual Studio est allée encore plus loin. CodeMap est non seulement accessible en mode développement, mais aussi en mode debug. Cet outil vient compléter très efficacement la suite de debugging qu'est devenu Visual Studio. Grâce à CodeMap vous allez pouvoir

suivre le cheminement de votre session de debug et analyser les références. Contrairement à la fenêtre de pile d'appels, CodeMap ne dépile pas quand vous sortez d'une fonction. Ceci vous permet de garder une trace de votre passage et même d'annoter le diagramme.

Cet outil est parfait pour partir à la découverte du code source si vous venez d'arriver sur un nouveau projet, mais également d'avoir une meilleure compréhension de votre application qui se complexifie au fur et à mesure que le temps passe.

> Naviguer vers... (Navigate to...)

Ceci vous permet de naviguer vers la définition de l'élément sur lequel vous êtes situé dans le code source. Le raccourci par défaut est « Ctrl + , ». Vous vous demandez alors quelle est la différence avec « Atteindre la définition » dont le raccourci est « F12 » ? En fait, le « Naviguer vers... », offre plusieurs possibilités de navigation. Dans Visual Studio

2012 cela vous ouvre une nouvelle fenêtre vous laissant le choix de l'endroit où naviguer. Une fois sélectionné, un nouvel onglet fixe est ouvert [Fig.9]. Dans Visual Studio 2013, la manipulation a été nettement améliorée : l'action n'ouvre plus une nouvelle fenêtre indépendante, mais une fenêtre incrustée tout comme la recherche rapide. De plus un onglet d'aperçu est automatiquement ouvert sur la première réponse trouvée. On peut revenir à l'onglet d'origine très rapidement en appuyant tout simplement sur la touche « Echap ».

> Return Value

Voici encore une amélioration du débogueur de Visual Studio 2013 qui risque de vous simplifier la tâche : le « ReturnValue ». Comme son nom l'indique, il s'agit de fournir la valeur retournée par une fonction lorsque vous débutez votre application.

A l'heure actuelle, pour avoir l'équivalent de cette fonctionnalité vous utilisez sûrement la « Fenêtre Exécution » (*Immediate Windows*) ou même un espion (*Watch*). L'inconvénient de cette méthode est que passer des paramètres devient vite très contraignant. La solution facile est de créer une variable locale pour stocker la valeur de retour uniquement pour des besoins du debug. Pas très pratique, je vous l'accorde. D'où l'utilité de cette nouvelle fonction : Pour pouvoir afficher la valeur retournée par une fonction, vous avez plusieurs possibilités : la méthode la plus simple est d'afficher la fenêtre « Automatique » (*Autos*), le retour d'une fonction est affiché directement sous le nom : « namespace.ReturnValue.Fonction » [Fig.10]. Si vous appelez plusieurs fonctions dans la même instruction, tous les retours seront affichés. Vous pouvez aussi récupérer la

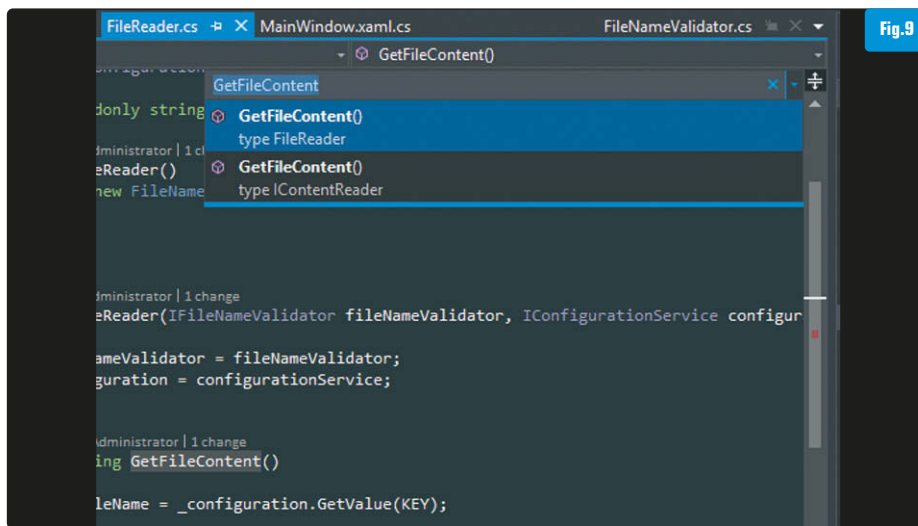


Fig.9

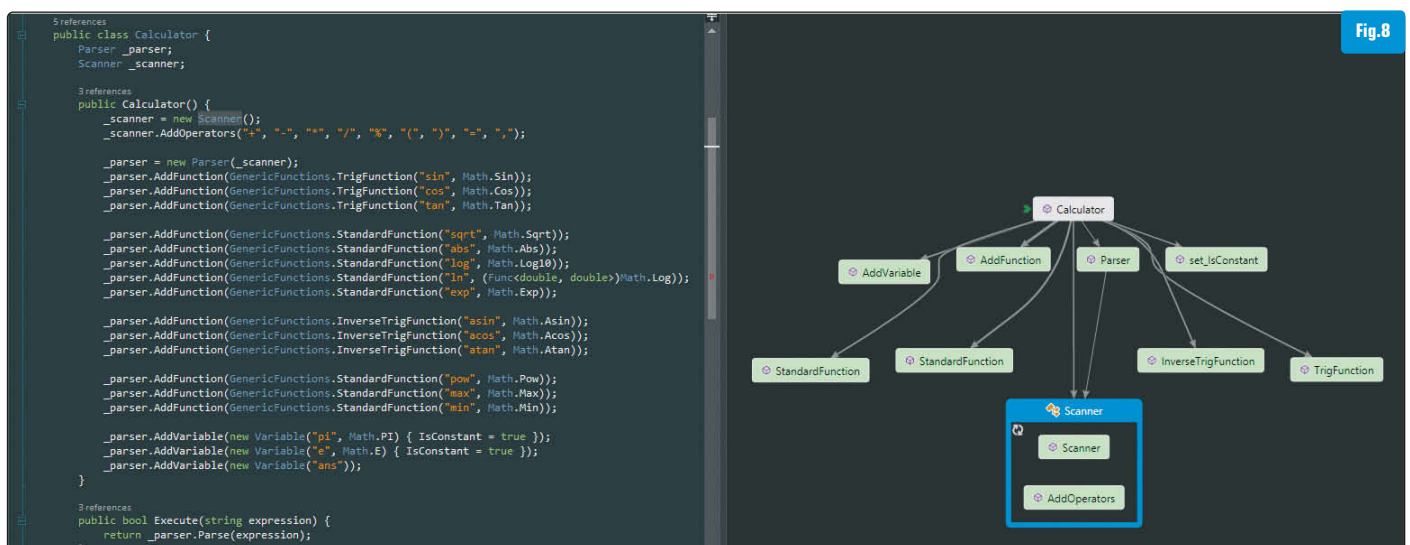


Fig.8

valeur du dernier retour dans un espion grâce à la variable « \$ReturnValue » [Fig.11]. Ceci est également valable pour la « Fenêtre Exécution ».

> Auto complétion

Les utilisateurs de ReSharper connaissent ceci depuis de nombreuses années, en effet il s'agit de la fermeture automatique des parenthèses, accolades, crochets, guillemets et apostrophes. Lorsque vous tapez un de ces caractères ouvrants, l'IDE ajoutera automatiquement le caractère fermant correspondant après votre curseur. Fort simple sur le principe mais très appréciable au quotidien.

AGILITÉ

Beaucoup de choses disponibles dans Team Foundation Service le sont maintenant dans Team Foundation Server 2013. Microsoft aligne, en fonction des rythmes de livraison de ces deux outils, la version cloud et « on premises » de ces outils de développement. Le plus important étant que l'expérience de développement soit la même dans Visual Studio.

> Git

Git a fait son apparition dans Team Foundation Service cet hiver et dans Visual Studio 2012 juste après en tant qu'extension. En tant que DVCS, Git est un énorme changement dans la façon de travailler. Les utilisateurs ont à présent le choix entre Git et TFVC (Team Foundation Version Control) le contrôleur de source historique de TFS pour chaque projet. Il y a plusieurs choses importantes à dire autour de Git :

- Ce n'est pas une version de Git faite par MS, mais une version de Git comme nous pourrions la retrouver sur Github par exemple : Microsoft contribue activement à la communauté Git.
- Git possède la notion de repository, il est donc normal de la retrouver dans TFS : il

est possible d'avoir plusieurs repository pour chaque projet.

- Git dans TFS implémente les mécanismes classiques d'authentification de TFS, mais est aussi accessible de clients non Windows comme Linux ou Mac OS.

Voici à quoi ressemble Git dans Team Foundation Server 2013 (cela est identique dans Team Foundation Service) : nous avons la possibilité de voir les différents repository, de gérer leurs droits respectifs et même de parcourir le code et les branches [Fig.12]. L'expérience utilisateur n'est pas encore aussi complète que sur Github, mais cela est en bonne voie !

Côté Visual Studio 2013 l'intégration est transparente ! Il y a déjà eu un travail phénoménal dans Visual Studio 2012, mais l'intégration dans Visual Studio 2013 est particulièrement réussie. Nous avons la possibilité de voir les différents « repos » de chaque projet, de gérer les branches [Fig.13]... En bref, toutes les fonctions basiques de Git. Visual Studio 2013 étant encore en « preview » au moment d'écrire ces lignes, la ligne de commande de Git est toujours utile.

> Agile Portfolio

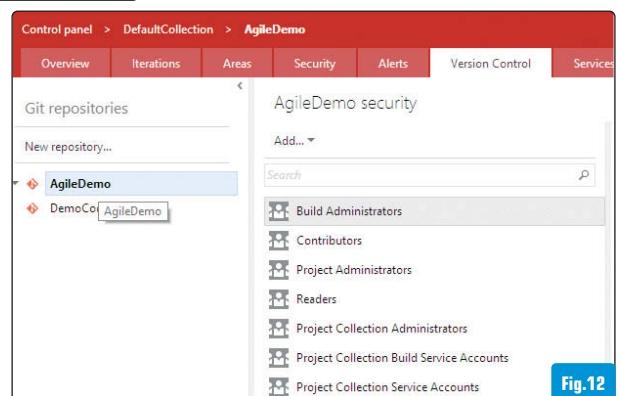
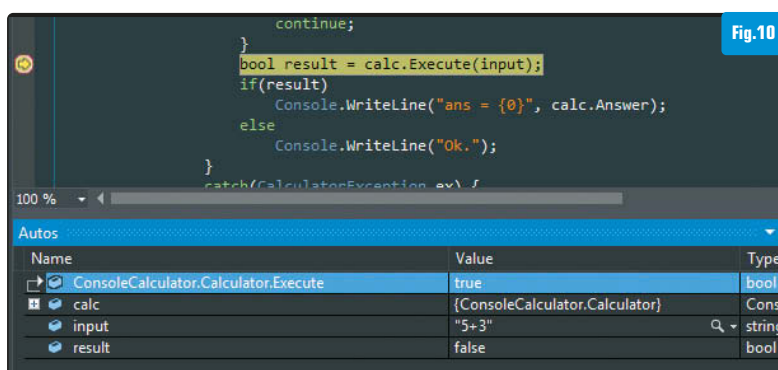
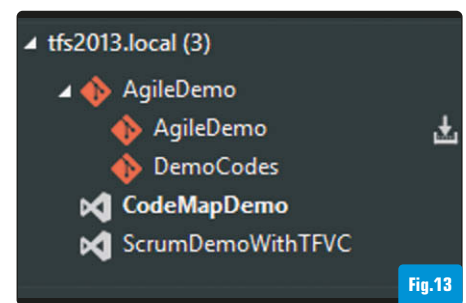
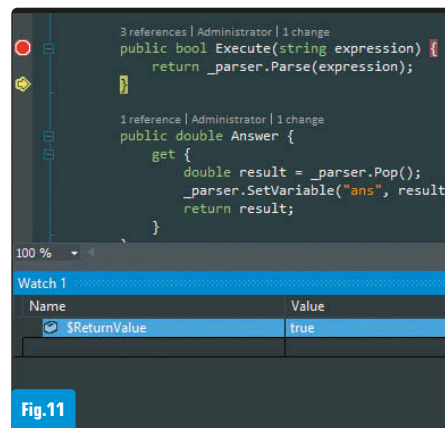
C'est sûrement la fonctionnalité la plus attendue pour la gestion de projet agile ! Elle répond à une problématique très simple :

Avec le portail de TFS 2012 on pouvait aisément gérer le projet d'une équipe, mais lorsque l'on avait besoin d'augmenter le nombre de contributeurs au projet, les équipes étaient obligées de partager le même backlog. Il n'y avait pas beaucoup d'alternatives : soit utiliser les « area » pour identifier les équipes du backlog, soit utiliser plusieurs « team project ». Cette dernière option bien qu'alléchante, posait plein de problèmes. Généralement, le code source était partagé entre les équipes, il y avait un « team project » pour le code et « un team project » par équipe, et cela interférait avec le fonctionnement de « My work » du Team Explorer. Et surtout, il fallait tout le temps synchroniser les itérations, et les requêtes multi-projets étaient complexes, souvent fausses et donc source de problèmes.

Le problème est résolu dans TFS 2013 : 2 nouveaux artefacts apparaissent dans un « team project » :

- La notion de Feature
- La notion d'équipe

La « Feature » s'ajoute dans la hiérarchie des workitems comme suit : Feature > Product Backlog Item/Bug > Task [Fig.14]. Le but de la « Feature » est d'être cross équipe. La notion d'équipe permet de segmenter les utilisateurs d'un « Team Project » en sous-ensembles. Chaque ensemble est associé à une « area » et cette « area » est utilisée dans les user stories pour identifier les équipes. La grande différence est qu'il est à présent possible de sélectionner une équipe dans l'interface web et Visual Studio pour



automatiquement filtrer les work items. Il y a donc vraiment plusieurs équipes dans un même « Team Project » [Fig.15].

Tout se configure via l'interface web où les utilisateurs peuvent ajouter des équipes.

Maintenant nous pouvons affecter des éléments de travail pour chaque équipe en les liant aux « Features ».

Ensuite il suffit de filtrer dans l'interface web [Fig.16] ou dans Visual Studio [Fig.17].

> Team Room

Team Room est une autre fonctionnalité qui est déjà présente dans Team Foundation Service. C'est basiquement un système de chat (les plus âgés d'entre nous diraient IRC !), mais il permet également d'agrèger des événements serveur ou même des liens vers des workitems. Côté événements il y a :

- Les opérations sur le contrôleur de source,
- Le statut des builds,
- Le changement d'état des workitems.

Cela peut sembler inutile dans un premier temps, car les personnes sont censées se parler plutôt que discuter via une page web, mais le fait qu'en plus nous pouvons suivre les événements de TFS et qu'il y a un historique rend les choses intéressantes. Prenons le cas par exemple d'une Team Room dédiée à une branche de production. Nous

pouvons avoir les informations sur les derniers checkins et dernières builds. Si, en plus, les équipes de l'infra y ont accès, cela fait un point de contact en temps réel assez puissant [Fig.18].

> InRelease

InRelease n'est pas apparu dans la preview mais un peu après. Au moment où nous écrivons ces lignes, un schéma s'esquisse :

- La partie Edition d'InRelease sera intégrée dans Visual Studio 2013,
- La partie serveur sera bien sûr intégrée dans Team Foundation Server 2013,
- Les agents de déploiement auront des licences séparées.

Cette intégration montre la volonté de Microsoft d'avoir un outil DevOps complet. Une version preview est aussi en téléchargement libre : <http://www.incyclesoftware.com/inrelease/inrelease-2013-preview/>

> Commentaires en ligne

Il est maintenant possible d'écrire des commentaires dans le code directement depuis l'interface web. C'est un énorme gain de productivité car, la plupart du temps, nous n'avons pas besoin d'ouvrir un projet dans Visual Studio pour nous rendre compte que le code contient des éléments à

corriger : un simple éditeur de texte suffit. Dans notre cas nous avons l'éditeur en ligne et la possibilité pour chaque changeset d'afficher le code qui a changé et donc celui dont la personne qui a fait le changeset est responsable [Fig.19].

CONCLUSION

Cette version, même si elle n'est pas révolutionnaire, contribue clairement à l'amélioration des performances de l'équipe. Si on ajoute à cela des mises à jour tous les trois mois, nous avons maintenant un outil de plus en plus productif et qui évolue plusieurs fois par an.

Alexandre Demoulin

Consultant chez Cellenza

Cellenza - Software Development Done Right

Blog: <http://blog.cellenza.com>

Michel Perfetti

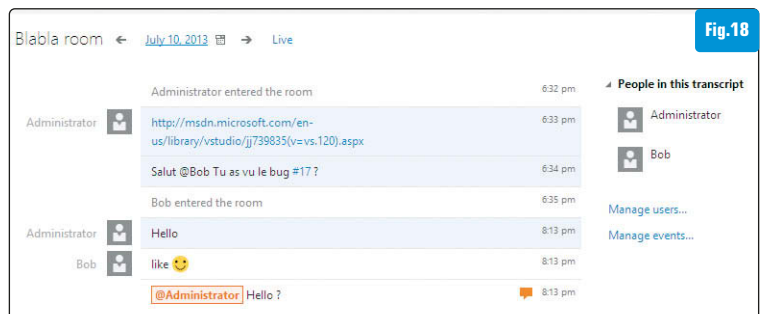
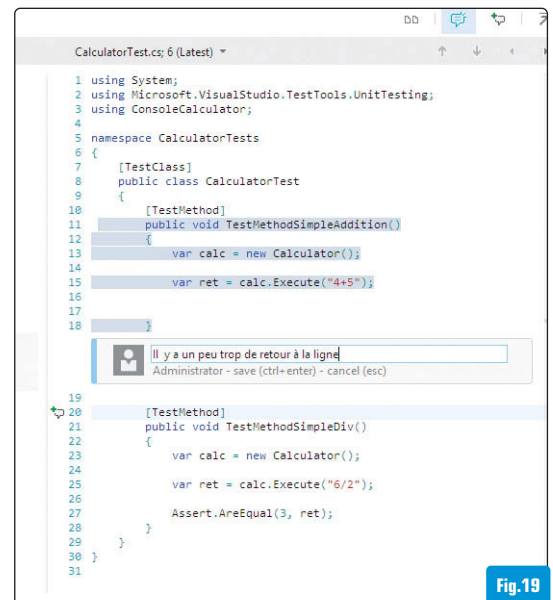
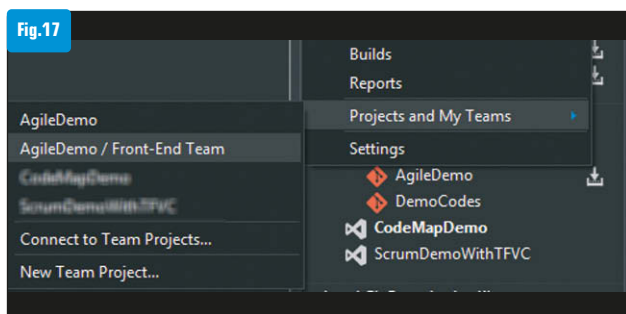
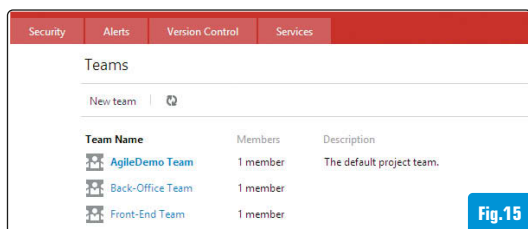
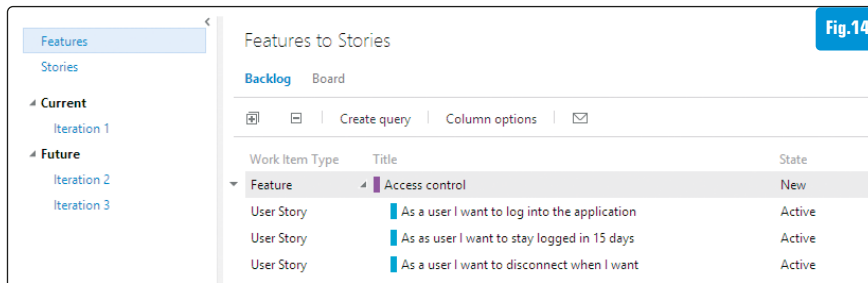
Manager / MVP Visual Studio ALM

chez Cellenza

Cellenza - Software Development Done Right

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

Twitter : @miiitch



La configuration idéale du développeur

Quel que soit le domaine d'activité du développeur aujourd'hui, les impératifs matériels sont relativement similaires. Mais une très grande majorité des configurations proposées par les constructeurs ne répond pas pleinement à ses besoins. Dans tous les cas, pour qui veut configurer sa machine, quelques fondamentaux doivent être respectés avant d'envisager l'achat des composants hardware.

> Comment procéder ?

Trois éléments sont à privilégier dans une machine pour un développeur : le disque, la mémoire vive et le processeur.

- La rapidité de disque : un développeur utilise des machines virtuelles pour simuler les serveurs d'application ou web et ainsi constater le plus rapidement possible l'impact des modifications directement effectuées dans le code. Nous conseillons donc d'utiliser un SSD performant dédié à cette



partie-là dans l'idéal, afin de maximiser les performances et la réactivité de ces « VM »

- La mémoire RAM : L'exécution simultanée de nombreuses commandes et d'applications chargent bien sûr cette partie des ressources de la machine. Il est conseillé bien évidemment de développer sur du 64 bits pour pouvoir utiliser une grande quantité de mémoire.
- Enfin le processeur, qui se place étonnamment 3e dans ce classement de priorités matérielles. Evidemment, plus le processeur sera capable d'effectuer de calculs sur une même période, plus les compilations seront rapides.

Forts de ce constat, nous pouvons donc conclure avec certitude qu'il est impossible de trouver un PC naturellement destiné aux développeurs parmi les PC complets proposés par les constructeurs. Le choix d'un intégrateur devient plus logique dans ces métiers spécifiques.

Les différentes nouvelles technologies sur le marché comme les derniers processeurs Intel Haswell (socket 1150), ou les « solid-state drive » (SSD) sont des technologies qui apporteront un réel plus à l'activité et au rendement du développeur. Mais toujours en privilégiant l'ordre de priorité des dépenses dans cet ordre : Disque à Mémoire à Processeur à Carte graphique.

Le choix de la carte graphique sera fonction aussi du type de développement. Un développeur qui se destine à des créations graphiques 2D ou 3D aura besoin de s'orienter vers un support de calcul offert par un GPU. Avec des fonctionnalités permettant l'utilisation du calcul en parallèle directement via le chipset graphique. Il suffit dans ce cas de remplacer la carte graphique en fonction de son activité et partir au besoin sur des cartes graphiques professionnelles pour l'OpenGL comme les cartes NVIDIA TESLA



ou Quadro, ou encore les cartes FirePro du concurrent AMD. Mais ce choix de carte reste spécifique aux besoins de calculs graphiques ou d'utilisation de plateformes de traitement parallèle comme CUDA.

> Conseils de configuration

Donc, selon les critères expliqués ci-dessus, nous pouvons imaginer des machines sur une base technique suivante :

1 boîtier + alimentation (500 W 80Plus bronze au minimum)

Exemple : Cooler Master Silencio 550W

1 Disque système pour les performances générales du système d'exploitation (SSD)

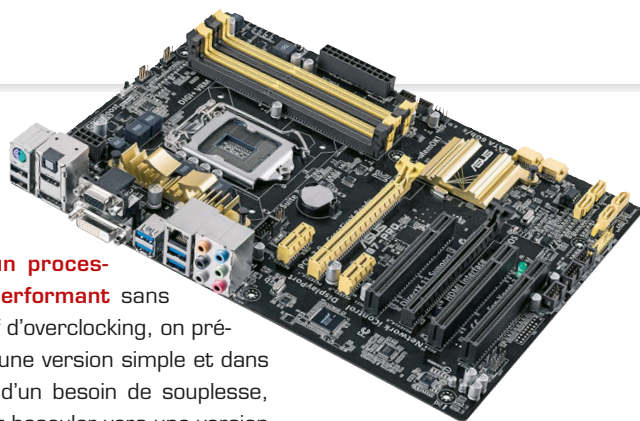
Exemple : Samsung SSD 840 PRO 128 Go
(nous préférons le Crucial M4 ou Intel SSD série 520, NDLR)

1 Disque dur ou **disque SSD** pour l'ensemble des environnements de travail à différencier du système et des données.

Exemple : Samsung SSD 840 PRO 256 Go

De la mémoire suffisante en quantité et aux performances moyennement élevées (éviter l'entrée de gamme et les fréquences trop basses des barrettes)

Exemple : G.Skill XM Series RipJaws X Series 16 Go (kit 2 x 8 Go) DDR3 1600 MHz CL9
(bien entendu cela va aussi dépendre de la mémoire acceptée par la machine / carte mère, NDLR)



Pour un proces-
seur performant sans
objectif d'overclocking, on pré-
fèrera une version simple et dans
le cas d'un besoin de souplesse,
on peut basculer vers une version
"K" mais en modifiant la carte
mère sélectionnée ci-dessous
pour profiter pleinement des fonc-
tions d'overclocking du chipset.

Exemple : Intel Core i5-4670 (3.4 GHz)

La carte mère cohérente avec
l'ensemble des pièces sélection-
nées ci-dessus, en gamme et en
évolutivité.

Exemple : ASUS H87-PRO

La carte graphique peut rester
très simple mais avec 2 sorties
pour le multi écran. Là, les dévelo-
ppeurs avec des besoins spécifiques
sur le calcul graphique s'orienteront
sur une pièce bien plus performante.
Mais pour les autres, le code ne
nécessite pas un gros investissement
on restera sur une carte en
dessous de 100 €.

Exemple : ASUS GT640-2GD3 2 GB
(ou une carte Gigabyte)

Pour le reste, c'est du classique,
un HDD supplémentaire pour
ceux qui ont un besoin de stocka-
ge de données interne 2To, toute
marque en économie d'énergie.

Exemple : WD Green Desktop 2 To
SATA 6Gb/s (pour des exigences de
stabilité et performances prenez un
disque certifiée serveur comme les
WD Caviar, NDLR)

Un lecteur Optique standard
comme un graveur de DVD, on
passera au format Blue en fonc-
tion des besoins personnels ou
une utilisation polyvalente de la
machine (backup froid, archivage,
installation).

Exemple : LG GH24NS95 SATA Noir

Pour le système d'exploitation, le
64 bits sera donc de rigueur pour
la pleine exploitation des fonction-
nalités du matériel (attention :

certains logiciels ne peuvent fonc-
tionner en 64-bit et pour de la
compilation 32, ayez toujours un
système 32 sous la main, NDLR).
Un intégrateur comme www.LDLC.com
propose la possibilité ou non de
monter la machine, si l'utilisateur
se sent l'âme d'un technicien et
désire connaître sa machine sur
le bout des doigts. La machine
d'exemple n'est ni la plus chère, ni
la moins chère. Comptez environ
1 200 €.

> Les plateformes d'achat

Au-delà des plateformes LDLC.com
ou Matériel.net, les sites construc-
teurs offrent souvent de meilleures
précisions sur les caractéristiques
du composant convoité, avec par-
fois une tarification plus avantageuse.
Notez les sites nvidia.fr et
asus.com qui permettent en plus
un comparatif des tarifs sur diffé-
rents sites d'e-commerce. Pour
Aymeric Fouchault, Directeur d'Epi-
tech Nantes, gros acheteur de
composants, notamment à l'occa-
sion du projet NG Drone (AR drone
+ Leap Motion) : « les sites chinois
Alibaba.com et dx.com sont très
bien, peu chers, avec des services
clients de qualité et des produits
livrés dans la semaine qui suit
l'ordre de commande. Tout est
réuni pour me satisfaire ». Enfin,
deux sites à l'initiative de l'AFUL
(Association Francophone des utili-
sateurs de Logiciels Libres), [bons-
vendeurs-ordinateurs.info](http://bons-vendeurs-ordinateurs.info) et [bons-
constructeurs-ordinateurs.info](http://bons-constructeurs-ordinateurs.info) mé-
ritent le détour pour y voir plus clair !

Merci à Stéphan Euthine (LDLC.com)
pour ses conseils avisés.

Adrien Fildier
Guillaume Couturier
Epitech Toulouse

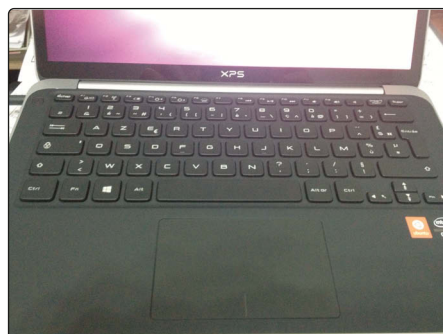
Dell XPS édition développeur : un ultrabook performant

Dell avait annoncé une édition développeur de son modèle
XPS, un ultrabook plutôt réussi aussi bien du côté perfor-
mances que design général de la machine. Cette dernière est
livrée avec la distribution Ubuntu 12.04 LTS. Nous avons été
agréablement sur-
pris par la qualité de
fabrication du XPS, la
qualité homogène de
l'écran et les perfor-
mances.

Il faut dire que la
configuration testée
était équipée d'un
Intel Core i7, de 8 Go
de ram et d'un
disque SSD de 256
Go. Et cela se vérifie
dès le premier dé-
marrage. Le systè-
me est opérationnel
en quelques secon-
des, et très réactif.
Nous apprécions
beaucoup la simplicité
de la logithèque
Ubuntu et de Ubuntu
One.

Dell a privilégié la partie matérielle. La partie développeur est,
elle, réduite au minimum. A vous d'installer vos environne-
ments préférés. Espérons que pour de futures éditions, Dell
puisse faire des partenariats avec des éditeurs de logiciels de
développement ou pré installer tous les principaux environne-
ments open source. Si vous cherchez une machine de travail et
mobile, le XPS est un excellent choix.

Le tarif est un peu élevé, même par rapport aux excellents Mac-
Book Air et MacBook Pro : à partir de 1 199 € TTC.



LES +

Les performances
La qualité de finition
La configuration
Ecran 13" full HD

LES -

Le clavier petit et les touches
Le trackpad non tactile
Une offre logicielle développeur trop minimaliste
Pas de connecteur magnétique à la MagSafe
Tarif un peu élevé
Difficulté pour installer Windows

JSON, jQuery : les deux piliers du développement web

On met facilement en avant HTML 5, CSS 3, JavaScript, mais on oublie trop vite deux éléments incontournables dans la majorité des développements web actuels, et futurs : JSON et JQuery.

Pour faire simple, JSON est un format d'échange de données JavaScript. Il signifie JavaScript Object Notation. Le format est très léger et d'une structure souple et utilisable avec tous les langages. JSON repose sur un format texte totalement indépendant, ce qui facilite grandement son usage. Son succès est venu avec l'explosion d'Ajax. On le compare parfois à XML mais XML est plus lourd à utiliser (même si les outils pour le manipuler sont pléthoriques) et surtout très verbeux. Aujourd'hui, la simplicité de JSON est son atout, même s'il est moins riche qu'un format XML...

JQuery est l'autre vedette du web. Ce framework JavaScript est populaire par son approche tout-en-un et évite de tout coder en JavaScript. Il aide le développeur à aller plus vite. Il est léger, compatible CSS 3 et multi-plate-forme. Mais finalement, JQuery

n'est qu'un framework JS parmi d'autres. Il a rapidement été adopté grâce à une communauté active, aux extensions, aux déclinaisons mobiles.

Pour ce numéro de rentrée, Programmez ! vous propose de mieux comprendre JSON et JQuery : découverte, comment implémenter et les bonnes pratiques à acquérir immédiatement.

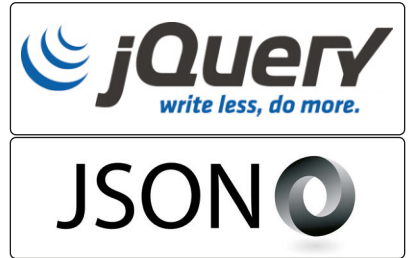
François Tonic





jQuery/JSON : simple, puissant

Début 2006, le très prometteur framework jQuery fait son apparition. Ce dernier permet enfin de manipuler le DOM simplement et offre un moyen pratique de rendre les pages web plus dynamiques. Quelques mois plus tard, le JSON est standardisé et offre un nouveau formatage des informations plus épuré, contrairement au XML.



Depuis 2009, Node.js permet de programmer en JavaScript côté serveur. Node.js offre d'énormes possibilités et permet de créer des Web Application Full-JS. Grâce au JSON et à l'évolution de jQuery avec, entre autres, l'intégration de l'AJAX, le WEB subit une nouvelle mutation qui permet l'apparition de Web Applications. Propulsées par le développement du nouveau standard de HTML5 et de CSS3, les Web Applications s'enrichissent de fonctionnalités qui les rendent presque aussi puissantes qu'une application native (développée en C, C++, etc.).

L'évolution du Web est loin d'être terminée, en atteste le support de nombreuses nouvelles fonctionnalités du HTML5, et l'adoption de ce standard par des géants de l'informatique tels que :

- Google avec gmail
- Microsoft avec Office 365
- Apple avec l'iCloud
- ...

Dans ce contexte, le JSON devient un format de fichiers très utilisé. Nombreuses sont les entreprises qui revoient leurs API pour prendre en compte le JSON avec le XML. Et le futur du JSON reste prometteur puisqu'il fait partie des outils devenus quasiment incontournables aujourd'hui. En ce qui concerne jQuery, son futur peut paraître plus incertain. Son plus grand concurrent devient paradoxalement le navigateur lui-même et ses API qui permettent également de manipuler le DOM plus facilement. Par exemple, avec l'ajout de « `querySelector` » qui correspond à un des multiples atouts qui rendaient jusqu'alors jQuery presque indispensable.

Nombre de développeurs, à l'image de Maxime Thirouin (@MoOX) ne sollicitent jQuery qu'à de très rares occasions. Ils préfèrent désormais utiliser le « `querySelector` » du navigateur qui est intégré dans la majorité de ces derniers. Cependant, des bibliothèques (*library*) telle que Backbone.js reposent sur le framework jQuery, ou équivalent (Zepto). Car ce framework n'est pas qu'un « `querySelector` ». C'est aussi, et avant tout, un outil qui offre de nombreuses fonctionnalités telles que :

- l'implémentation d'un système événementiel
- des plugins de contrôle du design
- un système d'intégration de plugins externes simples
- ...

Entre autres, c'est aussi une encapsulation de l'AJAX, des événements du DOM, etc. En clair, jQuery est un très bon framework qui se défend encore d'un point de vue fonctionnalités et encapsulations. Enfin, jQuery est un framework qui continue d'évoluer. En plus de 7 ans d'existence, jQuery se développe avec le WEB et ses fonctionnalités se diversifient. Désormais dans sa version 2.0, ce framework s'est adapté afin de répondre à de nouvelles demandes, telles que jQuery UI (user interface) ou jQuery Mobile.

Porté par une forte communauté de développeurs, massivement utilisé (Wordpress, Wikipédia...), jQuery est depuis 2012 soutenu par une fondation garantissant certainement sa pérennité future.

> Point forts de jQuery

Pourquoi jQuery s'est-il retrouvé être l'un des framework web incontournable ? La raison en est simple : sa facilité d'utilisation mise en exergue sur le site de la jQuery Foundation : « *write less, do more* ». En effet, jQuery est simple d'accès.

> Manipuler le DOM

Lors de sa sortie en 2006, manipuler le DOM était lourd et fastidieux. Pour sélectionner un élément il fallait utiliser différents appels du DOM.

Exemple (JS / DOM)

```
var firstId = document.getElementById("firstId") ; // Sélectionne un élément avec l'ID « firstId »
var firstClass = firstId.getElementsByClassName("firstClass") ;
// Sélectionne les éléments ayant la Class « firstClass » (pas implémenté dans tous les navigateurs).
var contentOfFirstId = firstId.innerText ; // Affiche le contenu sous forme de texte
var contentOfFirstClass = firstClass[0].innerText // Si on veut lire tout le tableau de class, il faut le parcourir
```

Exemple (JS / jQuery)

```
var firstId = $("#firstId") ; // le # signifie ID en CSS
var firstClass = $(".firstClass") ; // le . signifie Class en CSS

var contentFirstId = firstId.text() ; // récupère la valeur sous forme de texte
var contentOfFirstClasses = firstClass.text() ; // récupère la valeur de toute les class sous forme de texte
var contentOfFirstClass = $(".firstClass :first").text();
var contentOfFirstClassLast = $(".firstClass :last").text();
```

Le code est bien plus simple et bien plus maintenable dans le second exemple. On peut voir qu'il est aisé de sélectionner un ou plusieurs éléments. De plus, jQuery offre même la possibilité de chercher simplement un élément dans un autre.

> Comment jQuery utilise les bonnes pratiques du JavaScript ?

jQuery utilise les « bonnes pratiques » du JavaScript en faisant en sorte que les arguments soient des « hash » (du JSON) plutôt que d'envoyer tous les arguments les uns après les autres.



Exemple :

```
var $elm = $("#id") ;
var firstClass = $(".firstClass", $elm) ;
var anotherClass = $(".anotherClass", $elm) ;
```

Qu'est-ce qu'un hash ? C'est simplement du JSON, mais quand on l'utilise en JavaScript on appelle ça un hash.

Exemple (JS / JSON) :

```
var hash = {
  "name" : "dupont",
  "age" : 21,
  "isAlive" : true
  "address" : "192.168.19.1"
};
```

Une des bonnes pratiques du JavaScript dans jQuery est d'utiliser un hash contenant les variables que l'on souhaite modifier. L'exemple le plus simple serait celui de l'utilisation de l'AJAX. Grâce au hash, jQuery peut compléter le hash envoyé par ses options personnelles.

Exemple (JS/jQuery) :

```
// Version simple
$.ajax({
  url : "http://www.google.com"
});

// Version avancée
$.ajax({
  url : "http://www.google.com" ,
  type : "GET",
  dataType : "html"
  // etc...
});
```

Les deux requêtes précédemment écrites sont identiques pour jQuery car il auto-complète le hash avec ses options par défaut.

> Quelques bonnes pratiques avec jQuery

Il est possible de programmer de différentes manières et d'obtenir des résultats identiques. Ainsi, il est très facile de programmer d'une façon qui n'est pas optimisée avec jQuery. Pour cela, voici quelques « best practice » avec jQuery, proposées par Jean-Christophe Queval-Bourgeois (@jc_queval) étudiant de 2e année à Epitech Toulouse.

Exemple (JS/jQuery) :

```
// Sélection d'élément avec des actions multiples
// Méthode peu optimisée
$("#mon_id").addClass("red");
$("#mon_id").click(function () {
  $("#mon_id").remove("red");
  $("#mon_id").add("blue");
});
```

On peut voir assez souvent ces lignes de code sur le web. Dès lors, regardons de plus près ce qui n'est pas optimisé dans ce cas. Quand on utilise `$("#mon_id")`, on fait faire un traitement de recherche dans le DOM à chaque fois. Cela prend un temps considérable quand on répète l'opération à de multiples reprises. L'une des meilleures techniques est de stocker cette référence dans une variable, et comme on utilise les bonnes pratiques, il faut rajouter un

« \$ » devant sa variable pour signaler que cette dernière est en réalité un objet jQuery. Dans la continuité des bonnes pratiques, jQuery a été conçu de façon à pouvoir faire du « chaining », ce qui n'est absolument pas exploité dans l'exemple ci-dessus.

Exemple (JS/jQuery) :

```
// Sélection d'élément avec des actions multiples
// Méthode avec les bonnes pratiques
var $elm = $("#mon_id");
$elm.addClass("red").click(function () {
  $elm.removeClass("red").addClass("blue");
});
```

De cette façon, on optimise considérablement le traitement. jQuery est stocké dans une variable afin d'éviter de se déplacer dans le DOM à chaque fois. Le « chaining » permet de rester dans la continuité de l'action en cours. jQuery est un excellent moyen de chercher dans le DOM. En outre, il offre différents outils pour pourvoir à cette requête. Cependant, ces outils ne peuvent être utilisés dans tous les cas.

Exemple (JS/jQuery) :

```
// Faire des recherches
// 1 - Bad
$("#mon_id").find("p").find(".block").html("Mon nouveau texte");
// 1 - Good
$("#mon_id p .block").html("Mon nouveau text");

// 2 - Bad
var login = «»;
var password = «»;
$("#mon_id").find("input").each(function () {
  if ($(this).attr("name") == "login") {
    login = $(this).val();
  }
  else if ($(this).attr("name") == "password") {
    password = $(this).val();
  }
});
// 2 - Good
var $elm = $("#mon_id");
var login = $('input[name=login]', $elm).val(),
password = $('input[name=password]', $elm).val();
```

Le but des exemples ci-dessus n'est pas de montrer tous les moyens de recherche possible, mais il énumère quelques cas où le code est parfois surchargé pour rien alors que jQuery gère très bien la demande. Il arrive dans certains cas que nous ayons besoin d'affecter des événements à un élément que nous avons effacé puis recréé. Il ne fait pas de mal, dans certain cas, de s'assurer que le navigateur a bien pris en compte que c'est un nouvel élément. Pour cela un simple « `.off(«*»)` » suffit.

Exemple (JS/jQuery) :

```
// Reinitialiser de façon sûre un element
$(«.button»).off(«*»).click(function () {
  alert(«Click !»);
});
```

Ce genre de pratique est fait en général dans le cas où la page est rechargée de façon partielle, par des requêtes AJAX par exemple.

Les outils des Décideurs Informatiques

*Vous avez besoin d'info
sur des sujets
d'administration,
de sécurité, de progiciel,
de projets ?
Accédez directement
à l'information ciblée.*

Cas clients
Actu triée par secteur | **Avis d'Experts**



Actus / **Evénements** | **Newsletter** | **Vidéos**



www.solutions-logiciels.com

☐ **OUI, je m'abonne** (écrire en lettres capitales)

Envoyer par la poste à : Solutions Logiciels, service Diffusion, GLIE - 17 chemin des Boulangers 78926 Yvelines cedex 9 - ou par fax : 01 55 56 70 20
1 an : 50€ au lieu de 60€, prix au numéro (Tarif France métropolitaine) - Autres destinations : CEE et Suisse : 60€ - Algérie, Maroc, Tunisie : 65€ , Canada : 80€ - Dom : 75€ Tom : 100€
10 numéros par an.

☐ M. ☐ Mme ☐ Mlle Société

Titre : Fonction : ☐ Directeur informatique ☐ Responsable informatique ☐ Chef de projet ☐ Admin ☐ Autre

NOM Prénom

N° rue

Complément

Code postal : Ville

Adresse mail

☐ Je joins mon règlement par chèque à l'ordre de SOLUTIONS LOGICIELS ☐ Je souhaite régler à réception de facture



> Mettre à profit le JSON et JavaScript

Le JSON et JavaScript, étant issus de la norme ECMAScript, offrent des possibilités très intéressantes, comme de créer un objet que l'on peut « compléter » au fil du code. Backbone.js, conçu pour créer des Web Applications en fournissant une base solide, utilise des hash pour pouvoir créer des objets descendant des « classes » de base.

Exemple (JS / JSON)

```
// Créer un objet en JS en utilisant la structure du JSON
var obj = {
  say: function () {
    console.log("Hello World!");
  }
};

obj.sayGB = function () {
  console.log("Good Bye !");
};

obj.say();
obj.sayGB();
```

La création d'objets en JavaScript utilisant le principe du JSON, est très facile. Cependant, il y a quand même des différences avec une class et un hash, comme par exemple, l'instanciation.

Exemple (JS/JSON)

```
function newObjByHash() {
  return ({});
};

var objByClass = new objByClass();
var objByHash = newObjByHash();
```

Pour créer un objet « instanciable » avec un hash, il faut établir une fonction qui va créer un nouveau contexte et qui va retourner un objet venant d'un contexte qui sera détruit. Cela provoquera alors la création d'un nouvel objet ne dépendant pas d'un contexte précis. Les hash sont très flexibles et offrent, en partie, les mêmes possibilités qu'une classe. Il est possible d'utiliser « this » afin d'accéder aux attributs ou méthodes du hash.

Exemple (JS/JSON)

```
var hash = {
  str: "Hello World",
  say: function () {
    console.log(this.str);
  }
};

hash.say(); // "Hello World"
```

Utiliser du JSON reçu par AJAX

```
var monObj = {
  data: {}
};

function getAjaxContent(data) {
  monObj.data = JSON.parse(data);
  console.log(monObj.data); // affichera le JSON reçu par la requête
}
```

> Scénario d'utilisation de jQuery / JSON

L'utilisation de jQuery et du JSON, pour réaliser des actions communes telles qu'une simple connexion à un panneau d'administration, peut se faire en JSON.

Exemple

```
function connectMe (e) {
  e.preventDefault(); // on empêche le signal d'envoi du formulaire
  $.ajax({
    url: "/api/account",
    data: {
      login: $('input[name="login"]').val(),
      password: $('input[name="password"]').val()
    }
  })
  .done(function (res) {
    /*
    ** Si la connexion à réussi
    ** res = {status: "OK", url: "/url/"};
    ** Sinon
    ** res = {status: "BAD", message: "la cause"};
    */
    try
    {
      res = JSON.parse(res);
      if (res['status'] == "BAD") {
        $('#error_msg').fadeIn(500).html(res.message);
        return;
      }
      // Nous partons du principe que le serveur retourne l'URL
      // du panneau d'administration
      window.location.href = res.url;
    }
    catch (e) {
      alert("Mauvais type de donnée renvoyé");
    }
  });
};

// on ajoute l'événement «submit» au formulaire ayant l'ID «connectMe»
// Quand l'événement sera realise, jQuery appellera la «callback»
connectMe.
$('#connectMe').submit(connectMe);
```

> Liens utiles

Communauté française des utilisateurs de JavaScript :

<http://francejs.org>

jQuery Foundation :

<https://jquery.org/>

Formations JavaScript :

<http://www.js-attitude.fr/>

Remerciements à Jean-Christophe Queval-Bourgeois (Développeur HTML5 / CSS 3 / JavaScript, spécialisé dans les Web Applications. https://twitter.com/jc_queval - <http://www.ariesdiluna.com>) et Sébastien Chopin [CTO Pantera Commerce, Créateur de animateur ToulouseJS, <http://www.toulousejs.org/> @Atinux]

Adrien Fildier (Epitech Promo 2017)

& Guillaume Couturier (Epitech Promo 2017)



JSON, un format devenu incontournable ?

Au cours des 8 dernières années l'utilisation du JavaScript n'a cessé d'augmenter. En effet en 2005, le web a fortement évolué avec l'émergence du paradigme AJAX et Web 2.0. Cette évolution a permis l'enrichissement des interfaces utilisateurs et la démocratisation du langage JavaScript. Plus récemment, l'essor du JS (JavaScript) a popularisé un format d'échange : le JSON. Sa croissance est telle qu'aujourd'hui le JSON est devenu le format incontournable d'échange de données sur le web.

1 JAVASCRIPT : LA RÉVOLUTION EST EN MARCHÉ

> La (re)naissance

Considéré à l'origine comme un langage trop basique et intrusif pour l'utilisateur, le JavaScript a longtemps traîné une mauvaise réputation auprès des développeurs web. Cette réputation est en train d'évoluer et ce, pour plusieurs raisons.

La première de ces raisons est l'**important écosystème qui gravite autour de ce langage** : il existe des milliers de bibliothèques et de frameworks open source disponibles. Ainsi, le JS est le langage le plus représenté sur la plate-forme GitHub (<https://github.com/languages/>).

C'est notamment grâce à des bibliothèques et des frameworks tels que jQuery, MooTools, Prototype, qu'il est devenu plus simple de développer des interfaces clients enrichies avec des composants graphiques, des animations et des effets visuels, améliorant l'expérience des utilisateurs.

Une seconde raison expliquant l'essor du JS est liée à sa **facilité d'apprentissage**. En effet, plusieurs bibliothèques permettent de cadrer et masquer la grande permissivité du langage, et facilitent la création d'applications riches (RIA) en AJAX (*Asynchronous JavaScript and XML*) pour les développeurs.

Dire que l'essor du JS n'est pas lié à l'AJAX serait mentir. En effet, l'asynchronisation avec les requêtes AJAX a permis d'obtenir des sites et des applications web fluides, sans rechargement de page, ressemblant ainsi à de véritables applications de bureau. AJAX a permis de réaliser notamment des **Single Page Application** (comme Gmail). L'enrichissement des interfaces a fait croître le volume de code JS. C'est pour cela que de nouveaux frameworks JS MVC (Model View Controller) sont apparus afin d'organiser et architecturer le code JS. Ces frameworks connaissent une forte croissance et permettent de profiter de la puissance des moteurs des nouveaux navigateurs web. Ceci permet notamment de déporter davantage de traitements côté client et ainsi réduire les calculs sur les serveurs.

La dernière - et à nos yeux la principale - raison concerne le domaine de l'affichage sur différents périphériques. A l'heure actuelle, l'utilisation combinée de JavaScript, HTML5 et CSS3 constitue **la meilleure solution pour développer des applications web cross plateformes**, qui réagissent différemment et qui s'adaptent sur les principales plateformes (mobile, bureau, tablette, et TV).

> L'utilisation avec AJAX

AJAX (*Asynchronous JavaScript and XML*) est un standard W3C permettant l'échange de données entre client et serveur ainsi que la mise à jour du contenu de la page web sans avoir à la rafraîchir.

Plus précisément, AJAX est une architecture permettant d'envoyer des requêtes XHR (*XMLHttpRequest*) pour commander des traite-

ments sur le serveur depuis le client et mettre à jour le DOM (*Document Object Model*) de la page web sans la recharger. Les requêtes XHR sont banalisées et simplifiées par l'utilisation des frameworks qui offrent des méthodes et des services permettant de les gérer en quelques lignes de code. Par exemple, jQuery offre une palette de services qui permettent d'interroger des serveurs et de manipuler la donnée reçue en réponse (sans se soucier du navigateur).

```
$.ajax({
  url: «http://localhost/rest/service/clients»
}).done(function ( data ) {
  console.log(data.item[0].name);
});
```

L'utilisation de l'AJAX permet d'asynchroniser les actions. En d'autres termes, il n'est plus nécessaire d'attendre la réponse du serveur(1) avant de pouvoir effectuer une autre action. Il suffit de faire la demande au serveur et de continuer à exécuter des traitements côté client (exemple : présenter la page suivante, et l'informer plus tard avec un message lors de la réponse valide du serveur). Quand le serveur renvoie la réponse il suffit de traiter les données reçues et finaliser l'action. La réponse à une requête AJAX est moins volumineuse qu'une réponse de requête HTTP normale, car elle ne contient que la donnée minimum pour modifier l'affichage de la page. C'est le JS qui va s'occuper de modifier le contenu de la page préalablement chargée et d'y insérer la donnée reçue.

De ce fait le flux sur la bande passante est considérablement réduit puisque les sources HTML, les feuilles de styles CSS et les scripts JS ne sont demandés et chargés qu'une seule fois.

> Le design pattern MVC pour limiter "l'effet spaghetti"

JavaScript combiné à l'utilisation de l'AJAX a toutefois été victime de son succès. En effet, de plus en plus utilisé au sein des applications, AJAX a nuí énormément à la qualité du code de ces dernières. Les développeurs se sont rendus compte que les applications étaient composées essentiellement de code spaghetti impossible à maintenir.

Ce constat s'est accompagné d'une réflexion des développeurs et architectes afin de repenser l'architecture des applications. Un focus particulier a été mis sur l'utilisation des design patterns au niveau de la couche front office, ces derniers ayant démontré leur succès dans les langages de développement lourds tel que le .Net et Java. Le design pattern utilisé pour modulariser les scripts JS est le

(1) Une réponse qui dans une architecture standard client serveur contient la page HTML qui embarque les données demandées à afficher au client.



pattern MVC. Désormais, avec l'utilisation de ce pattern nous ne parlons plus de code JS mais directement d'application JS. Cette architecture autour du MVC permet de faciliter la maintenance du code, de réduire le code dupliqué en déclarant par exemple les structures des objets dans les modèles, de faciliter le développement des tests, etc. Différents frameworks JS respectant le pattern MVC ainsi que des variantes sont actuellement disponibles comme **Backbone** pour le pattern MVP (Model View Presenter), Knockout pour le pattern MVVM (Model View View Model) ou **Angular** pour le pattern MVW (Model View Whatever).

Ce design pattern a permis aux applications JS de s'adapter à l'évolution des architectures côté serveur. En effet, beaucoup d'applications reposaient sur une architecture SOAP (Simple Object Access Protocol) couplée au format d'échange XML. Les web services étaient alors utilisés pour faire communiquer des applications hétérogènes entre elles. En revanche, notamment avec les nouveaux usages de la mobilité, ils sont davantage utilisés aujourd'hui dans une logique client /serveur classique reposant sur une architecture REST plus simple à mettre en place. Ce changement d'utilisation des web services mais également d'architecture a entraîné la modification du standard du format d'échange. C'est grâce à cela que le JSON s'est très vite popularisé au détriment du XML.

2 LES FORMATS D'ÉCHANGES

Les données échangées entre navigateur client et site/application web doivent suivre un standard afin d'être interprétables facilement et rapidement. Les principaux formats d'échange actuels sont : le plain text, le INI, le XML et le JSON. Le format plain text est simplement du texte brut échangé entre le navigateur et le serveur.

Le format INI, initialement utilisé pour créer des fichiers de configuration, comporte des sections qui contiennent un ensemble de clés valeurs et peut donc représenter des objets.

Exemple :

```
[Student]
name=John Do
class=Terminal
```

Le XML (eXtensible Markup Language) est un format dit de balisage, chaque donnée devant être encadrée par des balises. Le principe initial du XML est de pouvoir faciliter les échanges de données complexes telles que des arbres, des listes, etc. De plus, ce format est extensible car il permet de définir ses propres espaces de noms, c'est-à-dire un vocabulaire spécifique qui décrit la structure et le contenu du fichier XML à travers un schéma XSD (XML Schema Document).

Exemple :

```
<?xml version="1.0" encoding="UTF-8"?>
<student>
  <name value="John Do"/>
  <class value="Terminal"/>
</student>
```

Le JSON (JavaScript Object Notation) est un format qui permet d'échanger et de stocker des informations au format texte. Il supporte deux types d'entité : les tableaux représentés par «[]», et les objets représentés par «{}». JSON respecte la norme suivante : les valeurs contenues par les tableaux ou par les propriétés des objets, sont des chaînes de caractères, des nombres entiers ou décimaux,

des booléens, la valeur spéciale *null*, ou bien entendu, des sous-tableaux ou des sous-objets. Bien que le JSON soit un format natif au langage JavaScript, il est interdit d'affecter des fonctions ou des dates car les parseurs du JSON des autres langages (jackson pour Java, Php, ...) risquent de ne pas accepter la chaîne au format JSON si elle ne respecte pas cette norme.

Exemple :

```
{student :
 {name: "John Do", class: "Terminal" }
}
```

Le format plain text est adapté pour échanger des messages simples mais ne l'est pas pour des applications qui vont utiliser des données structurées. Le format INI est très rapidement interprétable mais il n'existe pas de parseurs par défaut au sein des navigateurs et il reste limité en termes de représentation d'objets avec une arborescence complexe. Entant donné que le plain text et le INI sont des formats ne permettant pas de décrire des objets complexes, la question qui se pose alors et qui est reprise dans de nombreux débats sur internet est : quel format faut-il choisir entre le JSON et le XML ? La réponse à cette question dépendra d'abord des performances et de l'architecture côté serveur. Ainsi, le XML restera très pertinent, principalement dans le cas où il faut pouvoir valider les données en fonction d'un DTD (Document Type Definition) ou d'un schéma XSD, ce qui est souvent le cas dans des applications inter-entreprises. Dans tous les autres cas le JSON est le format à utiliser pour plusieurs raisons :

- il est moins volumineux que le XML et donc plus performant au niveau du transport réseau,
- il est natif et donc directement interprétable par le JavaScript sans faire appel à une librairie,
- il est plus rapide en lecture et en écriture que le XML,
- il gère les tableaux,
- il n'a pas de mots réservés,
- il est généralement plus lisible que le XML sur des petites structures.

Pour confirmer nos propos sur la performance du JSON face au XML nous avons réalisé un test simple en lisant en JavaScript la même donnée formatée en JSON et en XML. Nous avons exécuté ce test 100 fois pour chaque cas et obtenu les résultats moyens suivants :

Le fichier JSON :

```
{«name»: «John Do», «class»: «Terminal»}
```

Le fichier XML :

```
<?xml version="1.0"?>
<data>
  <name>John Do</name>
  <class>Terminal</class>
</data>
```

Puis nous avons comparé les résultats :

-> 0.048 millisecondes pour le fichier JSON

-> 0.308 millisecondes pour le fichier XML

D'après ce test, la lecture du JSON est environ 6 fois plus rapide que pour le XML. Aussi, il faut garder à l'esprit que cette différence va s'accroître davantage en fonction de la taille et de la complexité de la donnée à lire. Ce petit test a mis en évidence que le JSON est le format le plus performant parmi les principaux formats d'échange, mais cela ne veut pas dire qu'il est le plus performant de tous les for-



mats. En effet, il existe des formats plus performants que le JSON. C'est le cas par exemple de MessagePack(<http://msgpack.org/>). Ce format qui a pour slogan "C'est comme du JSON mais plus rapide et plus léger" se base sur une sérialisation binaire pour gagner en performance. Par exemple, les entiers courts vont être encodés sur un seul octet au lieu de deux, plusieurs améliorations de ce type vont permettre de diminuer le volume d'octets à transférer sur le réseau, et donc d'être plus performant que le JSON. Ce format n'étant pas natif au javascript, il est nécessaire d'importer la librairie avant de pouvoir l'utiliser. La librairie est disponible pour de nombreux langages tels que le JavaScript, Java, Php, Python, C/C*, Perl, Objective-C, C#, Ruby, etc...

> Le cross domain et JSONP

L'utilisation de l'AJAX et du JSON a facilité le chargement de données depuis des serveurs afin d'enrichir les pages affichées sur le navigateur. Cependant, un problème se pose en termes de sécurité car l'accès aux services web n'est pas sécurisé : tout le monde peut avoir accès aux données du service !

Pour sécuriser l'accès à ces services, la majorité des navigateurs web actuels implémentent la spécification de sécurité CORS (Cross Origin Resource Sharing) qui se base sur le nom de domaine à l'origine de la page qui demande l'accès au service.

Pour des requêtes de type GET ou POST avec certains MIME Types, le navigateur va ajouter un header *Origin* à la requête contenant le nom de domaine à l'origine de la requête XHR.

Si l'appelant est autorisé à accéder aux ressources, le serveur renvoie la réponse à la requête en ajoutant le header *Access-Control-Allow-Origin* contenant la liste des noms de domaine autorisés. Dans le cas où le nom de domaine de l'appelant n'est pas autorisé, le serveur renvoie une erreur. Pour les autres types de requêtes, le navigateur va préalablement envoyer une requête *OPTION* appelée requête *Preflight* afin de vérifier si le nom de domaine à l'origine de la requête peut accéder à la ressource. Si le serveur valide l'accès, il envoie la réponse contenant le header *Access-Control-Allow-Origin*, puis la requête initiale est envoyée au serveur qui y répond.

La politique CORS ne s'applique pas à toutes les requêtes HTTP, par exemple les requêtes générées par des balises `<script>` n'ont pas de limitation au niveau de nom de domaine.

```
<script type="text/javascript" src="http://www.sitedistant.com/service?param=value">
</script>
```

L'exemple ci-dessus envoie une requête GET au web service afin de charger du JSON. Néanmoins, cet appel ne peut fonctionner en l'état. En effet, le serveur va bien répondre à la requête mais il sera impossible pour le client de récupérer ce flux : la réponse n'étant affectée à aucun objet JS, le client ne pourra pas exploiter la réponse du serveur. JSONP (JSON with Padding) constitue la solution à ce problème. Il s'agit de messages JSON formatés spécifiquement et délivrés par le serveur afin qu'ils puissent être interprétés et utilisés par le client. Le principe réside dans le fait de passer un paramètre de callback au serveur. Ce paramètre de callback indique au serveur le nom d'une fonction JS qui traitera le résultat de l'appel. La fonction réceptrice est ajoutée et recevra les données du serveur distant.

```
<script type="text/javascript" src="http://www.sitedistant.com/service?param=value&callback=myCallBackFunction">
```

```
function myCallBackFunction(response) {
    console.log(response.data);
}
</script>
```

Réponse JSONP du serveur :

```
myCallBackFunction({«Name»: «Jhon Do», «classe»: «Terminal»});
```

Dans l'exemple ci-dessus les données seront reçues encapsulées dans l'appel de la fonction `myCallBackFunction()`. Cette fonction sera appelée avec l'objet JSON en argument lors de l'interprétation du code par le navigateur. Il est important de noter que seules les requêtes HTTP de type GET sont supportées avec le JSONP.

> Bonnes pratiques

Le JSON en lui-même ne nécessite pas l'utilisation de bonnes pratiques, il faut simplement respecter sa norme. En revanche, il y a quelques astuces à retenir :

- Faire transiter sur le réseau seulement la donnée nécessaire. Il n'est pas utile de renvoyer l'ensemble des attributs d'un objet mais seulement les attributs souhaités. Typiquement, dans une application 5 tiers, il est préférable de ne pas envoyer les objets entités directement mais d'utiliser des DTO (Data Transfer Object).
- Le principe de pagination se fait côté serveur et pas au niveau du client, il n'y a aucun intérêt à envoyer tout un ensemble d'éléments alors qu'au niveau du client nous souhaitons afficher seulement les 10 premiers éléments.
- A moins de développer une API publique, pour des raisons de sécurité, il faut éviter d'utiliser le format JSONP.
- Ne pas donner l'accès à tous les domaines au web service avec "access-control-allow-origin: *" à l'exception du développement d'une API publique.
- Pour éviter toute erreur d'un parseur JSON, il est déconseillé d'utiliser les caractères espace, tabulation, et retour à la ligne lors de la construction de l'objet JSON.
- Dans une application REST pensez toujours à retourner les codes de statuts HTTP et développez des tests sur le code réponse pour que l'utilisateur de l'application ne soit pas bloqué lors d'une perte de session. Il est toujours plus simple de tester un code retour que de remplir son code de déclaration conditionnelle comme : `if (typeof(maVariable) != «undefined»)`
- Pour sécuriser les données d'une application REST qui transitent sur le réseau il faut utiliser le protocole HTTPS
- Penser à se protéger des attaques CSRF (Cross Site Request Forgery).

CONCLUSION

Il est indiscutable que le JavaScript prend de plus en plus de place au sein des applications web, avec les nouveaux frameworks. L'architecture des applications RIA basée de plus en plus sur REST ainsi que le nouvel usage qui est fait des web services a rapidement popularisé le format JSON. Néanmoins nous avons vu que de nouveaux formats d'échange apparaissent tel que MessagePack qui s'avère plus efficace en termes de performance que le JSON. Nous pouvons alors nous poser la question de l'avenir du JSON, va-t-il laisser sa place comme XML, va-t-il disparaître ou bien va-t-il évoluer vers une nouvelle norme ?

Ramuntxo Vidal & Souhail Hanfi - Consultants chez Neoxia

Pourquoi jQuery a-t-il autant de succès ?

jQuery est une bibliothèque JavaScript publiée pour la première fois en janvier 2006 par John Resig, aujourd'hui une des voix les plus écoutées dans le microcosme de ce langage. Cette époque coïncide avec l'essor de ce qu'on a appelé le « web 2.0 », c'est-à-dire des sites plus dynamiques rendant l'expérience utilisateur plus vivante. C'est aussi l'avènement de l'Ajax, un système permettant de mettre à jour à la volée le contenu d'une page web, sans avoir à recharger celle-ci.

JavaScript, jusqu'alors un langage peu usité, va soudain connaître un vif regain d'intérêt, qui va aboutir à l'apparition en l'espace de quelques mois de nombreuses bibliothèques visant à en simplifier l'usage. Car un gros problème se pose : les navigateurs de ce temps-là, qu'ils soient Internet Explorer, Firefox ou Safari, ont chacun leurs propres particularités, que le développeur web se retrouve à devoir traiter individuellement. Par exemple, la création d'un appel Ajax peut varier en fonction du navigateur qu'utilise le client, voire de la version de celui-ci.

```
// Création d'un objet XMLHttpRequest «à l'ancienne»
var request, ua = navigator.userAgent.toLowerCase();
if (!window.ActiveXObject)
    request = new XMLHttpRequest();
else if (ua.indexOf('msie 5') == -1)
    request = new ActiveXObject("Msxml2.XMLHTTP");
else
    request = new ActiveXObject("Microsoft.XMLHTTP");
```

C'est pour normaliser ces contraintes de développement qu'apparaissent des bibliothèques comme Prototype, MooTools, et donc jQuery. Leur objectif est identique : servir de tampon entre le développeur web et le navigateur du client, de manière à épargner au premier de devoir s'adapter au second. À ce jeu, jQuery va rapidement sortir du lot, et s'imposer comme incontournable.

Production ready

L'utilisation de jQuery est totalement gratuite quel qu'en soit l'usage, et la bibliothèque est publiée sous licence MIT, ce qui la rend adéquate même pour des projets commerciaux. Par ailleurs, sa taille est limitée (seulement 32 Ko une fois minifiée et gzippée), elle est compatible avec l'intégralité des navigateurs du marché (y compris les versions les plus vieillissantes comme Internet Explorer 6), et son code source est intégralement consultable sur github [1].

Des mises à jour, correctives ou évolutives, sont aussi publiées régulièrement, en faisant un projet *opensource* très actif. En outre, jQuery jouit d'une vaste communauté qui fournit aussi bien des patches que de très nombreux et très variés plugins, permettant d'étendre ses fonctionnalités bien au-delà de ses possibilités initiales. Le site officiel dispose aussi d'un forum très actif où chacun peut venir poser ses questions, exposer ses problèmes, ou aider les autres membres de la communauté.

jQuery dispose aussi d'une excellente documentation, complète, bien organisée, avec de nombreux exemples très explicites permettant de comprendre très facilement à quoi servent et comment utiliser ses nombreuses fonctionnalités.

Tout ceci contribue à faire de jQuery une bibliothèque très populaire utilisée par 54,7% des 10 000 sites les plus visités au monde [2], et sponsorisée par de gros poids lourds de l'industrie tels que Microsoft (qui l'inclut en standard dans ses templates d'applications web) ou Google, qui l'hébergent par ailleurs sur leur CDN respectif [3].

Un sélecteur performant

La manipulation du DOM avec le langage JavaScript repose initialement sur la sélection grâce à celui-ci d'éléments dans l'arbre que représente la page web. A l'origine, il n'existait que deux méthodes permettant de sélectionner ces éléments : *document.getElementById* qui renvoyait un élément par son Id, et *document.getElementsByTagName* qui renvoyait tous les éléments correspondant à un certain type de balise. Certains navigateurs proposaient également une méthode *document.getElementsByClassName* très pratique, mais malheureusement caduque car non supportée par toutes les plateformes.

La première mission de jQuery a donc consisté à normaliser la sélection des éléments du DOM en JavaScript. Pour cela, elle intègre une sous-bibliothèque nommée Sizzle, qui consiste en un moteur de parsing visant à parcourir l'arbre de la page aussi vite et aussi efficacement que possible.

Sizzle a deux atouts dans sa poche. D'abord, il utilise les mêmes sélecteurs que la norme CSS3, ce qui permet aux développeurs de capitaliser sur leur apprentissage, puisque celle-ci est appelée à prendre le pas au fur et à mesure que les utilisateurs, et surtout les entreprises, migreront vers des navigateurs plus modernes supportant justement CSS3 et plus généralement HTML5.

Ensuite, Sizzle est un moteur de parsing « intelligent ». Aussi optimisé soit-il, ce moteur ne saurait remplacer la puissance brute des fonctionnalités natives du navigateur. C'est pourquoi Sizzle n'hésite pas à laisser la main à ce dernier lorsqu'il sait qu'il saura se montrer plus efficace. C'est le cas notamment pour les navigateurs modernes qui disposent des nouvelles méthodes *document.querySelector* et *document.querySelectorAll*, qui reposent, elles aussi, sur les sélecteurs de la norme CSS3. Deux exceptions à ce principe toutefois : ces deux méthodes connaissent pour l'instant des bugs d'implémentations connus de Sizzle, et par ailleurs certains sélecteurs un peu particuliers de Sizzle ne sont pas compatibles avec elles. Dans ces deux cas, Sizzle n'utilisera donc pas les fonctionnalités natives du navigateur, et préférera s'en remettre à son propre moteur.

[1] <https://github.com/jquery/jquery>

[2] <http://royal.pingdom.com/2012/06/20/jquery-numbers/>

[3] <http://ajax.googleapis.com/ajax/libs/jquery/2.0.0/jquery.js>
<http://ajax.aspnetcdn.com/ajax/jquery/jquery-2.0.0.js>



La « magie » des sélecteurs dans jQuery ne doit toutefois pas faire perdre de vue au développeur qu'il existe des bonnes pratiques à ne pas négliger. Typiquement, elles consistent à capitaliser sur les éléments disposant d'un Id (et qui sont donc uniques), à réduire le champ de la recherche à une portion de la page, et à mettre en cache les résultats.

```
// On sélectionne d'abord un élément par son Id avec # (rapide car Sizzle utilise document.getElementById)
// Puis on sélectionne les sous-enfants de cet élément ayant la classe «sous-elements», Sizzle ne parse donc qu'une partie du DOM
// On place le résultat en mémoire dans une variable «result»
var result = jQuery(«#element-avec-id .sous-elements»);
result.hide();
// Plus tard dans le code, on agit directement sur les éléments sans avoir à les re-sélectionner
result.show();
```

Si l'on dispose déjà d'un élément en mémoire à l'intérieur duquel on souhaite trouver des éléments enfants, jQuery propose un principe de « contexte », passant par un second argument, permettant là encore de réduire le champ de recherche à une portion du DOM.

```
// On a sélectionné l'élément ayant l'Id «comments»
var comments = jQuery(«#comments»);
// On cherche dans le contexte de cet élément les éléments enfants ayant la classe «notes»
var notes = jQuery(«.notes», comments);
```

Un inconvénient de ces sélecteurs est qu'ils reposent sur la programmation impérative, et nécessitent de manipuler des chaînes de caractères si l'on souhaite les créer dynamiquement. Fort heureusement, jQuery propose alternativement d'utiliser le potentiel de la programmation déclarative.

> Une architecture innovante

La construction « `jQuery(sélecteur)` » que l'on a vue jusqu'à maintenant est le point de départ des manipulations avec jQuery. Elle permet de rassembler les résultats dans un tableau JavaScript spécial, puisque augmenté par les différentes méthodes de jQuery. Ce tableau dispose d'une propriété « `length` » permettant de savoir combien d'éléments l'on manipule, chacun de ces éléments pris indépendamment étant un nœud du DOM tout ce qu'il y a de plus classique. A partir de ce tableau d'éléments, jQuery applique un schéma architectural qui remporte un certain succès de nos jours : le chaînage de méthodes (auss appelé *fluent interface*). Le principe est que les méthodes que l'on souhaite appliquer aux éléments viennent se « chaîner » les unes aux autres, et qu'après avoir effectué l'opération qui leur est dévolue, chacune d'entre elles renvoie l'ensemble initial ou un nouveau sous-ensemble d'éléments du DOM.

```
// On sélectionne tous les éléments ayant la classe «list», on leur applique du rose comme couleur de fond, puis on sélectionne leurs sous-éléments de type «li», on écrit «hello» à l'intérieur, on les fait disparaître rapidement selon un effet de fondu, on attend 500 millisecondes, puis on les affiche à nouveau
jQuery(«.list»)
  .css(«backgroundColor», «pink»)
  .find(«li»)
  .text(«hello»)
  .fadeOut(«fast»)
  .delay(500)
  .show();
```

Ce principe permet d'enchaîner rapidement les modifications, en travaillant de concert avec la puissance des sélecteurs, de manière à optimiser au maximum les performances dans le navigateur du client. Il permet aussi de traverser l'arbre du DOM de manière cette fois déclarative.

```
// Cette fonction va rechercher dans l'élément ayant l'Id «mon-tableau» les sous-éléments respectant le sélecteur passé en paramètre
function findInTable(searchSelector) {
  return jQuery(«#mon-tableau»).find(searchSelector);
}
```

Les seules exceptions sont les méthodes renvoyant une valeur, qui ne sont pas chaînables.

```
// Il n'est pas possible de chaîner des méthodes après «val()» car celle-ci retourne la valeur du champ input
var value = jQuery(«input#textbox»).val();
```

Notez que si vous avez du mal à retenir toutes les méthodes de l'API jQuery, il existe de nombreuses « cheat sheet » plus ou moins à jour sur Internet. L'architecture de jQuery permet aussi de l'étendre facilement par prototypage.

```
// Cette fonction permet de cacher ou afficher les éléments en fonction du résultat retourné par une autre fonction passée en paramètre
// Notez bien qu'elle respecte le chaînage de méthodes en retournant l'ensemble sur lequel elle agit
jQuery.fn.showOrHide = function (func) {
  return this.each(function() {
    var $this = $(this);

    if (func())
      $this.show();
    else
      $this.hide();
  });
};
```

Ce principe d'extension est d'ailleurs le point de départ de la création des plugins. Il est par ailleurs possible de créer ses propres sélecteurs.

```
// Créer un nouveau sélecteur «:inline» permettant de réduire la liste d'éléments à ceux affichés en mode «inline»
jQuery.expr[':'].inline = function(element, i, match) {
  return jQuery(element).css('display') === «inline»;
};

var inlineElements = jQuery(«.mes-elements:inline»);
```




> Le module Ajax

jQuery propose un module Ajax intelligemment encapsulé, puisqu'il cache efficacement la différence d'implémentation de l'objet XMLHttpRequest entre navigateurs, mais expose tout le nécessaire pour effectuer des requêtes asynchrones avec le maximum de souplesse.

```
// Un appel Ajax classique avec jQuery
jQuery.ajax('/produits', {
  «type»: «post»,
  «dataType»: «html»,
  «data»: { «id»: 12 },
  «success»: function (result) {
    jQuery(«#container»).html(result);
  },
  «error»: function () {
    alert(«Oops!»);
  }
});
```

Ce module supporte l'utilisation des appels Ajax synchrones, du jsonp, et est chaînable avec l'API de gestion des callbacks dont nous allons bientôt parler.

> La gestion des événements

Depuis l'avènement du « web 2.0 » et des pages web dynamiques, et encore plus aujourd'hui avec celui de HTML5 et des *single page applications*, la gestion des événements en JavaScript nécessite un contrôle très fin. jQuery propose une API extrêmement riche sur ce sujet, permettant de s'abonner à un événement, d'en déclencher un, voire de créer ses propres événements arbitraires.

```
// Abonnement à l'évènement «click» d'un bouton
jQuery(«button#mon-bouton»).on(«click», function (e) {
  // Action à effectuer
});

// Déclenchement du click sur un bouton
jQuery(«button#mon-bouton»).trigger(«click»);

// Désabonnement à l'évènement «click» d'un bouton
jQuery(«button#mon-bouton»).off(«click»);

// Abonnement à un évènement totalement arbitraire
jQuery(«#un-element»).on(«mon-evenement-arbitraire», function (e) {
  // Action à effectuer
});

// Déclenchement de l'évènement arbitraire
jQuery(«#un-element»).trigger(«mon-evenement-arbitraire»);

// Simulation de l'appui sur une touche du clavier
var e = jQuery.Event(«keyup»);
e.which = 13; // Touche entrée
jQuery(«input#textbox»).trigger(e);
```

Dans la version 1.7 de jQuery, l'API de gestion des événements a été largement remaniée pour ne garder que deux points d'entrée : *on* et *off*. Ceux-ci remplacent et permettent d'effectuer les mêmes opéra-

tions que les trois paires *bind/unbind*, *delegate/undelegate* et *live/die* qui sont d'ores et déjà dépréciées et seront amenées à disparaître de la bibliothèque.

Bind permet à un élément de s'abonner à un événement, mais *delegate* et *live* sont plus intéressants. Ils permettent de déléguer la capture d'un événement sur un élément parent de l'élément réellement visé. Par exemple, si l'on souhaite abonner toutes les cellules d'un tableau à l'évènement *click*, on peut avantageusement le faire en branchant un délégué à la table elle-même. Ainsi, au lieu de créer potentiellement une myriade de handlers dans la mémoire du navigateur (un par cellule), on en crée un seul. Ce système repose sur le principe de bulle des événements, qui lorsqu'ils sont déclenchés remontent le long de l'arbre du DOM et peuvent être captés dans les branches supérieures. *Delegate* permet de se brancher au niveau que l'on veut tandis que *live* nous branche directement sur l'élément body, captant ainsi tous les événements remontants jusqu'en haut de l'arbre (body en étant la racine).

```
// Le deuxième argument permet d'indiquer qu'il s'agit d'un
délégué, et que ce sont les cellules (th ou td) du tableau
qui doivent déclencher l'évènement
jQuery(«#mon-tableau»).on(«click», «th, td», function (e) {
  // e.target désigne la cellule exacte qui a été cliquée
  var clickedElement = jQuery(e.target);
});
```

L'API permet aussi facilement de stopper la propagation d'un événement, ou d'empêcher le navigateur d'effectuer son action par défaut.

```
jQuery(«input[type='radio']»).on(«click», function (e) {
  // Le navigateur ne cochera pas le bouton radio comme il
  devrait le faire
  e.preventDefault();

  // L'évènement ne bullera pas le long de l'arbre du DOM
  e.stopPropagation();
});
```

Bien entendu, le fait d'empêcher un événement de remonter empêchera d'éventuels délégués de le capter au passage. Autre élément important : le simple fait de passer par jQuery pour s'abonner à des événements permet aussi de s'assurer du bon nettoyage de ceux-ci dans la mémoire lorsque des éléments sont retirés du DOM. Cette règle est aussi valable lorsque l'on remplace des pans entiers de la page (par le biais d'un appel Ajax par exemple) puisque jQuery parcourt récursivement l'arbre du DOM pour désabonner les événements, mais aussi supprimer toutes données arbitraires (notamment les siennes) qui pourraient subsister en mémoire.

```
// Suppression du bouton du DOM
// jQuery nettoie de la mémoire le handler associé à
l'évènement «click» sur le bouton
bouton.remove();
```

> La gestion des callbacks

Introduit dans jQuery 1.5 et amélioré à plusieurs reprises depuis, la gestion des callbacks est un pan de la bibliothèque encore mal connu mais qui s'avère extrêmement utile pour construire les *single page*



applications. Celles-ci sont en effet de grosses consommatrices de requêtes Ajax et d'animations pour faire vivre la page, et ces deux types d'opérations, asynchrones par nature (on ne sait pas quand elles se termineront), nécessitent l'utilisation de callbacks.

Fort de ce constat, l'équipe derrière jQuery a décidé de fournir cette API qui offre de très nombreuses facilités pour manipuler et chaîner les callbacks, mais aussi pour créer ses propres fonctions asynchrones.

Expliquer en détail comment fonctionne cette API nécessiterait un article entier [4]. Pour faire simple, sachez que le principe repose sur l'objet `jQuery.Deferred`, qui contrôle le cycle de vie d'une opération asynchrone, et d'une *promise*, issue de cet objet, et qui lorsqu'on la communique à un autre objet consommateur va permettre à ce dernier de se brancher et de suivre l'évolution de l'opération.

```
// Ces deux opérations sont asynchrones et renvoient une
// promise (ajax la renvoie directement dans l'objet de retour)
var animation = jQuery("#mon-element").fadeOut().promise();
var ajaxCall = jQuery.ajax("/list-of-procucts");

// On chaîne les deux opérations asynchrones, et on affiche
// un message lorsque celles-ci se terminent avec succès
jQuery.when(animation, ajaxCall).done(function
(animParameters, ajaxParameters) {
    alert("Opérations réussies.");
});
```

Cette API très souple s'avère d'une grande aide dès lors qu'il s'agit de gérer des grappes d'appels asynchrones. Elle l'est aussi pour qui voudrait concevoir son propre système asynchrone.

```
function asynchronousCall() {
    var deferred = jQuery.Deferred();

    // Cette opération encadrée par setTimeout se déroulera
    // dans 5 secondes, soit bien après la sortie de la fonction
    setTimeout(function () {
        // N'importe quelle opération puis...
        deferred.resolve("foobar");
    }, 5000);

    return deferred.promise();
}
```

```
}
```

```
// On appelle la fonction asynchrone et on attend qu'elle ait
// fini d'effectuer son opération avant d'afficher le résultat
asynchronousCall().done(function (result) {
    alert("L'opération est terminée et le résultat est : « +
result);
});
```

Et pour l'avenir ?

Deux versions de jQuery cohabitent aujourd'hui : la branche 1.9 et la branche 2.0. Elles ont en commun d'avoir définitivement abandonné certaines méthodes jugées obsolètes pour alléger le poids de la bibliothèque. C'est le cas notamment de toute la partie dédiée à la détection des navigateurs, considérée aujourd'hui comme une mauvaise pratique. Ces méthodes s'avèrent pourtant parfois indispensables, notamment dans le cadre de l'utilisation de certains plugins, c'est pourquoi existe la bibliothèque jQuery Migrate [5], qui en plus de rétablir l'ancien fonctionnement de jQuery permet de diagnostiquer quel morceau de code utilise quelle méthode obsolète en affichant des messages dans la console.

La différence importante entre ces deux versions est que la 2.0 ne supporte plus Internet Explorer 6, 7 et 8. L'objectif ici est encore une fois de réduire le poids de la bibliothèque, mais aussi d'optimiser les performances en se débarrassant des cas particuliers de traitement rendus nécessaires par le comportement et les nombreux bugs des anciennes versions du navigateur. La simplification de la base de code de jQuery qui en découle devrait également permettre de faciliter l'introduction de nouvelles améliorations (notamment sur l'API de gestion des callbacks). Ces deux branches devraient cohabiter pendant encore quelque temps, avec des versions 1.10 et 2.1 d'ores et déjà prévues sur la feuille de route de jQuery [6].

François Guillot

Consultant .NET – MCNEXT

Blog : <http://www.mcnextpost.com> – Site Internet : www.mcnext.com

[4] <http://fguillot.developpez.com/tutoriels/javascript/manipulation-callbacks-avec-jquery/>

[5] <http://blog.jquery.com/2013/05/01/jquery-migrate-1-2-0-released/>

[6] <http://bugs.jquery.com/roadmap>



Abonnez-vous jusqu'à -50%
et pour **1€ de +**
recevez un "livre numérique"
des Éditions ENI



Coupon d'abonnement page 17 et sur www.programmez.com/abonnement.php



jQuery : 10 bonnes pratiques à connaître absolument !

Facile d'accès, rapide et puissant, jQuery s'est imposé comme la bibliothèque de référence pour simplifier les interactions avec les fonctionnalités les plus communes du langage Javascript et les différents navigateurs du marché. Cette apparente facilité en ferait presque oublier un certain nombre de bonnes pratiques essentielles qui, non appliquées, produisent bien souvent des effets désastreux en termes de performances. Afin d'éviter ces écueils, cet article propose de passer en revue 10 bonnes pratiques essentielles que tout développeur utilisant jQuery se doit de connaître.

1 GÉRER CORRECTEMENT L'INCLUSION DE JQUERY

Ce point semble trivial mais il est important de partir du bon pied avec jQuery et cela commence par son inclusion au sein d'une page. Une gestion correcte de celle-ci passe par les éléments suivants :

> Utiliser la dernière version de jQuery

Chaque nouvelle version de jQuery amène ses améliorations en termes de performances mais également son lot de correctifs. La dernière version stable, la 2.0.0, vient de sortir et la mettre en place tout de suite ne peut qu'être bénéfique.

> Utiliser une version minifiée de jQuery

Les chargements effectués par les navigateurs pour récupérer les fichiers Javascript utilisés au sein d'une page sont toujours consommateurs et pénalisent le temps de chargement. De fait, réduire la taille de ces fichiers tout en les combinant est une pratique essentielle pour les fichiers Javascript et CSS. Appliqué à jQuery, cela revient à utiliser une version minifiée de la bibliothèque en production.

> Utiliser une copie de jQuery servie par un CDN

Via son service de Content Delivery Network (CDN), Google met à disposition une copie des bibliothèques Javascript open-source les plus populaires sur ses serveurs, ce qui inclut jQuery. De nombreux sites utilisant cette solution, les utilisateurs de votre site auront sûrement une version cachée de jQuery sur leur navigateur, ce qui évitera des requêtes de téléchargement inutiles lors de l'affichage de vos pages. L'inclusion de jQuery via le CDN de Google se résume à la ligne suivante :

```
<script src="//ajax.googleapis.com/ajax/libs/jquery/2.0.0/jquery.min.js"></script>
```

> Prévoir une solution alternative au CDN

Même si le CDN de Google est fiable, il est toujours bon de prévoir une solution alternative lors de l'inclusion de contenus externes à son serveur. Dans le cas présent, il faut donc tester si le chargement de jQuery depuis le CDN a fonctionné et dans le cas contraire proposer une version de jQuery en local comme suit :

```
<script src="//ajax.googleapis.com/ajax/libs/jquery/2.0.0/jquery.min.js"></script>
<script>window.jQuery || document.write('<script src="js/libs/jquery-2.0.0.min.js"></script>')</script>
```

2 UTILISER LES SÉLECTEURS EFFICACEMENT

Fonctionnalité centrale de jQuery, les sélecteurs constituent une notion essentielle qu'il est primordial de bien comprendre.

> Utiliser en priorité les sélecteurs d'ID

Employé avec un ID en entrée, le sélecteur `$()` est alors mappé sur la fonction native `getElementById()`. Les fonctions natives étant toujours plus rapides que celles définies par une bibliothèque, les sélecteurs d'ID sont donc la solution la plus rapide pour récupérer un élément au sein d'une page :

```
$('#elt');
```

> Utiliser les sélecteurs de tags

Mappé sur la fonction native `getElementsByTagName()`, le sélecteur de tag est le second sélecteur le plus rapide. Cependant, il faut éviter de l'utiliser sur des tags présents en grand nombre dans une page. Ainsi, `$('head')` est généralement plus rapide qu'un `$('input')` puisque l'exécution de ce dernier reste dépendante du nombre d'inputs dans la page.

> Utiliser les sélecteurs de classes avec un contexte

Nécessitant un parcours complet du DOM, les sélecteurs de classes sont les sélecteurs jQuery les plus lents. Autant que faire se peut, il sont donc à proscrire. Une solution intermédiaire consiste à toujours définir des ID sur les éléments d'une page HTML de sorte que l'on puisse utiliser le sélecteur de classe avec en entrée un contexte restreint au sein duquel la recherche doit être réalisée. Ainsi, pour appliquer une couleur aux éléments d'une page marqués par la classe `myClass`, la mauvaise solution est la suivante :

```
$('.myClass').css('color', '#ACACAC');
```

La bonne solution consiste à spécifier un conteneur englobant ces éléments qui servira de contexte pour la recherche :

```
$('.myClass', '#myContainer').css('color', '#ACACAC');
```

Enfin, il faut prendre garde à l'utilisation des sélecteurs commençant par un tag puis ciblant ensuite un ID. Ainsi, l'exemple ci-dessous est à proscrire :

```
$('div#content');
```

En effet, ce sélecteur est très lent puisque nécessitant le parcours



de l'ensemble des éléments div à la recherche de l'ID *content* en leur sein.

3 METTRE EN CACHE LES OBJETS JQUERY

S'il ne devait rester qu'une seule bonne pratique au sein de cette liste, ce serait sûrement celle consistant à mettre en cache les objets jQuery récupérés via des sélecteurs. Compte tenu du temps induit par l'exécution des sélecteurs, la mise en cache des objets sélectionnés s'impose lorsque ceux-ci doivent être utilisés plus d'une fois. La mise en cache consistant simplement à garder une référence du résultat d'un sélecteur au sein d'une variable pour un emploi futur :

```
// sans cache
$('#myDiv').css('background-color', 'red');
$('#myDiv').bind('click', function() { ... });
$('#myDiv').show();

// avec cache
var myDiv = $('#myDiv');
myDiv.css('background-color', 'red');
myDiv.bind('click', function() { ... });
myDiv.show();
```

Dans le même ordre d'idée, il est également intéressant de tirer parti de la fonctionnalité d'appels chaînés de jQuery, ce qui raccourcit sensiblement le code précédent :

```
$('#myDiv').css('background-color', 'red').bind('click', function() { ... }).show();
```

4 LIMITER LES MANIPULATIONS DIRECTES DU DOM

Manipuler le DOM est une opération coûteuse. De fait, il faut réduire au maximum les manipulations visant à le modifier. Ce principe s'applique plus généralement au Javascript. En considérant l'insertion d'éléments au sein d'une liste, la solution basique consiste à ajouter les éléments un par un au sein de la liste, via la fonction `append()` de jQuery :

```
var listContent = [...];
var myList = $('#myList'); // sélecteur de l'élément ul

for (var i = 0; i < listContent.length; i++) {
    myList.append('<li>' + listContent[i] + '</li>');
}
```

Effectuant une manipulation directe du DOM pour chaque élément de la liste, ce code est très lent. Il est préférable de stocker l'ensemble des insertions au sein d'une chaîne de caractères avant d'effectuer un seul ajout via la fonction `html()` :

```
var content = '';

for (var i = 0; i < listContent.length; i++) {
    content += '<li>' + listContent[i] + '</li>';
}
```

```
myList.html(content);
```

Une solution encore plus rapide existe consistant à remplacer le nœud DOM de la liste initiale avec le nouveau contenu de la liste :

```
var content = '<ul id="myList">';
for (var i = 0; i < listContent.length; i++) {
    content += '<li>' + listContent[i] + '</li>';
}

content += '</ul>';
myList.replaceWith(content);
```

5 PRIVILÉGIER LES FONCTIONS JAVASCRIPT NATIVES

De par la puissance de sa syntaxe, jQuery améliore grandement la productivité des développeurs, ce qui peut pousser parfois à ne plus utiliser des fonctions Javascript de base en recourant systématiquement aux fonctions de la bibliothèque. Abuser de jQuery conduit quelquefois à une réduction des performances. Ainsi, pour itérer sur les éléments d'un tableau, la solution jQuery est basée sur sa fonction `each` :

```
var myArray = [...];

$.each(myArray, function(i) {
    myArray[i] = ... ;
});
```

L'emploi de cette fonction se révèle extrêmement lent et il est en réalité préférable de recourir à l'emploi d'une simple boucle `for` :

```
for (var i = 0; i < myArray.length; i++) {
    myArray[i] = ... ;
}
```

Toujours dans le même domaine, un autre exemple est également parlant. Il concerne la surutilisation du sélecteur `$(this)` qui est faite au sein des fonctions de callback définies avec jQuery. Prenons l'exemple suivant :

```
$('#myLink').click(function() {
    alert($(this).attr('id'));
});
```

Si comme ici, la seule raison du recours à `$(this)` est de récupérer l'ID de l'objet, cela s'avère consommateur sans aucune raison. Un simple recours à `this.id` est suffisant :

```
$('#myLink').click(function() {
    alert(this.id);
});
```

6 EVITER DE MIXER CSS ET JQUERY

Via sa fonction `css()`, jQuery permet de modifier directement les propriétés CSS d'un élément. Néanmoins, son utilisation peut être sour-



ce de problèmes. La plupart du temps il est préférable de manipuler les propriétés CSS via l'ajout ou la suppression de classes CSS. Cela se réalise à l'aide des fonctions `addClass()` et `removeClass()`. De fait, pour changer dynamiquement la couleur du texte d'un paragraphe, la meilleure solution consiste à créer des classes CSS spécifiques définissant les couleurs d'écriture souhaitées puis ensuite de les changer à l'aide des fonctions susnommées :

```
.red {
    color : red;
}
.blue {
    color : blue;
}

// code jQuery
var myParagraph = $('#myParagraph');
myParagraph.addClass('blue');
myParagraph.removeClass('blue').addClass('red');
```

7 BIEN EMPLOYER LES FONCTIONS AJAX

Le support Ajax fourni par jQuery est fantastique puisque la bibliothèque affranchit le développeur de problématiques de compatibilité entre navigateurs. Cependant, la présence de 4 fonctions pour les appels Ajax est source de questionnement pour le développeur qui ne sait pas toujours par où démarrer. En effet, que choisir entre `get`, `getJSON`, `post` ou encore `ajax` ? La réponse est simple puisque les 3 premières fonctions ne sont que des wrappers s'appuyant sur `ajax`. Utiliser directement cette dernière pour les appels Ajax est donc légèrement plus efficace. En outre, cette fonction propose différents callbacks pour interagir avec les appels Ajax. Dans l'idéal, un appel Ajax pourra donc proposer un callback pour `beforeSend` indiquant à l'utilisateur qu'une requête est en cours, un pour `success` et enfin un pour `error` en cas d'échec de la requête :

```
$.ajax({
    type : 'POST',
    url : 'http://MY_URL',
    data : {data1 : 'data1', ...},
    beforeSend : function() {
    },
    success : function(data) {
    },
    error : function(xhr, error) {
    }
});
```

D'autre part, un problème peut survenir lors d'appels Ajax réalisés depuis la méthode callback d'un click sur un lien. Pour ces derniers, on se retrouve avec une requête Ajax annulée. Pour s'en prémunir, il est nécessaire d'appeler `preventDefault()` sur l'objet `event` une fois l'appel Ajax effectué :

```
$('#myLink').bind('click', function(event) {
    $.ajax( ... );
    event.preventDefault();
});
```

8 UTILISER DES OBJETS LITTÉRAUX EN PARAMÈTRES

La plupart des fonctions jQuery acceptent des objets littéraux en paramètre, ce qui s'avère pratique pour proposer un code plus concis. Comme pour l'objet littéral passé en entrée de la fonction `ajax()`, il est possible de définir un objet littéral pour passer des données en paramètre d'une fonction. Un tour sur la documentation du projet (<http://api.jquery.com/>) permettra de vérifier si une telle possibilité est offerte par une fonction tout en ayant accès à un grand nombre d'exemples.

9 PRENDRE GARDE AUX PLUGINS

Du fait de sa popularité et de son extensibilité, jQuery regorge de plugins étendant ses fonctionnalités core. Devant l'offre pléthorique en plugins, la qualité est hétérogène et il est essentiel d'effectuer certaines vérifications de base avant d'utiliser un plugin :

- Taille : si un simple plugin est plus lourd que jQuery par exemple, il est bon de se poser la question de son utilisation.
- Performance : essayer le plugin sur son projet en le profilant avec un outil tel que Firebug.
- Compatibilité : tester si le plugin ne brise pas la compatibilité inter-navigateurs amenée par jQuery.
- Mobile : ne pas oublier que le mobile est une cible de plus en plus importante pour les sites et vérifier à minima qu'un plugin ne vient pas empêcher l'utilisation d'un site sur les navigateurs mobiles.

10 NE PAS UTILISER JQUERY COMME REMÈDE MIRACLE

La dernière bonne pratique peut sembler étonnante mais elle est gage d'un certain pragmatisme. jQuery étant la bibliothèque Javascript la plus utilisée, nombreux sont ceux qui y recourent systématiquement pour des tâches pouvant être réalisées directement en CSS avec un bon support inter-navigateurs. De même, pour des sites ayant très peu de code Javascript, il n'est pas nécessaire d'inclure jQuery si seule sa syntaxe concise en est la raison. Enfin, bien que les fonctionnalités apportées par jQuery et ses nombreux plugins soient vastes, il ne faut pas se fourvoyer en déplaçant certaines tâches serveur sur la partie cliente. Ainsi, si un plugin de validation existe, il ne doit pas faire perdre de vue que la validation côté serveur demeure un point de passage inamovible.

CONCLUSION

jQuery propose un ensemble de fonctionnalités visant à simplifier la vie du développeur en l'abstrayant des tâches Javascript récurrentes tout en assurant une compatibilité optimale entre navigateurs. La puissance et la facilité d'utilisation de l'outil ne doivent pas faire perdre de vue au développeur un certain nombre de bonnes pratiques qui, non appliquées, conduisent bien souvent à de sérieux problèmes de performances ou encore de stabilité. La liste proposée dans cet article n'est bien entendu pas exhaustive mais elle constitue néanmoins une base solide recensant les écueils les plus souvent rencontrés sur des projets jQuery et les solutions adéquates pour s'en prémunir. Avec ces bonnes pratiques, vous pourrez désormais partir du bon pied avec jQuery.

Sylvain Saurel – Ingénieur d'Etudes Java / Java EE
sylvain.saurel@gmail.com

Introduction au formulaire et persistance avec **Symfony 2**

2^e partie

Dans cette 2^e partie, nous allons créer notre première entité et la persister en base de données en passant par un formulaire. Je vous invite à reprendre la base d'application installée en suivant l'article du n°165 du magazine.

FORMULAIRE ET PERSISTANCE DE VOTRE PROPRE MODÈLE

> Création d'une entité

Prenons un exemple simple d'une Todo list. Une tâche est constituée d'un **identifiant unique** et d'une **description**. Cet exemple se repose sur l'utilisation de l'ORM Doctrine (site officiel : doctrine-project.org). Générons une nouvelle entité avec la commande suivante : `app/console doctrine:generate:entity`.

Un nouveau fichier « Task.php » dans le dossier `src/Acme/DemoBundle`, (Code généré sur le site www.programmez.com).

Il nous faut maintenant créer la base de données. Pour cela, lancez la ligne de commande `app/console doctrine:database:create`.

Il faut ensuite lancer la ligne de commande `app/console doctrine:schema:update --force` afin de mettre à jour le schéma de votre base de données conformément aux annotations @ORM de la classe d'entité. Notre table est maintenant créée en base de données.

NB : Assurez-vous de bien avoir renseigné les paramètres de connexion à votre base de données dans le fichier `parameters.yml` situé dans le dossier `app/config`.

FORMULAIRE D'ENREGISTREMENT

> Classe de formulaire

Créons la classe de formulaire `TaskType` : dans le dossier `src/Acme/DemoBundle/Form`, créez le fichier « `TaskType.php` » et saisissez le code suivant :

```
<?php

namespace Acme\DemoBundle\Form;

use Symfony\Component\Form\AbstractType;
use Symfony\Component\Form\FormBuilderInterface;

class TaskType extends AbstractType
{
    public function buildForm(FormBuilderInterface $builder,
        array $options)
    {
        $builder->add('description', null, array('label'=>'
Description de la nouvelle tâche : '));
        $builder->add('Envoyer', 'submit');
    }

    public function getName()
    {
        return 'taskForm';
    }
}
```

```
}
}
```

Dans cette classe, deux méthodes importantes pour construire notre formulaire. Dans la méthode « `buildForm` », nous ajoutons les champs à afficher dans le formulaire :

- Le champ `description` ;
- et le bouton de soumission de formulaire (de type `submit`) ayant pour label « Envoyer ».

La seconde méthode « `getName` » doit être implémentée et permettre d'indiquer le nom du tableau pour chacun des champs du formulaire. Dans notre exemple, le champ de `description` est donc généré ainsi en HTML :

```
<textarea id="taskForm_description" required="required" name="taskForm[description]"></textarea>
```

Afin de modifier le label d'un champ du formulaire, il vous faut passer le tableau associatif d'option(s) avec pour clé 'label' et en valeur ce que vous souhaitez voir s'afficher dans votre formulaire (pour en savoir plus : <http://symfony.com/doc/current/book/forms.html#field-type-options>).

> Contrôleur

Il vous faut ensuite créer la classe de contrôleur contenant l'action permettant de gérer notre formulaire. Créez le fichier `TaskController.php` dans le dossier `src/Acme/DemoBundle/Controller` du bundle et saisissez le code suivant :

```
<?php

namespace Acme\DemoBundle\Controller;

use Symfony\Bundle\FrameworkBundle\Controller\Controller;
use Symfony\Component\HttpFoundation\Request;
use Acme\DemoBundle\Form\TaskType;
use Acme\DemoBundle\Entity\Task;

use Sensio\Bundle\FrameworkExtraBundle\Configuration\Route;

class TaskController extends Controller
{
    /**
     * @Route("/create", name="create_task")
     */
    public function indexAction(Request $request)
    {
        $task = new Task();
        $form = $this->createForm(new TaskType(), $task);
    }
}
```



```

if ($request->isMethod('POST')) {
    $form->submit($request);
    if ($form->isValid()) {
        $em = $this->getDoctrine()->getManager();
        $em->persist($task);
        $em->flush();
        return $this->redirect($this->generateUrl('create_
task'));
    }
}

return $this->render('AcmeDemoBundle:Task:index.html.twig',
array(
    'form' => $form->createView(),
));
}
}

```

Deux éléments sont importants : notre objet *task* et l'objet *form*. Pour la création du formulaire, on passe une nouvelle instance de la classe « *TaskType* » et l'objet *task*. Si la méthode HTTP est POST, on met à jour l'objet *task* avec les informations postées (*\$request*) dans le formulaire grâce à la méthode « *submit* ». Enfin, nous récupérons le gestionnaire d'entité afin de convertir l'objet *\$task* en enregistrement dans la base de données.

> Routing

Enfin, il faut indiquer dans le fichier « *routing.yml* », situé dans le dossier *src/Acme/DemoBundle/Resources/config* le code suivant :

```

task:
    resource: «@AcmeDemoBundle/Controller/TaskController.php»
    type:      annotation
    prefix:    /task

```

Ces lignes permettent d'indiquer que les routes écrites en annotation dans la classe de contrôleur « *TaskController* » sont à prendre en compte. Le mot clé « *prefix* » permet d'indiquer que toutes les routes du contrôleur « *TaskController* » commencent par le motif d'url « */task* ». Ainsi, pour accéder au formulaire, il faudra indiquer l'url « */task/create* ».

> Vue

Et enfin, créons notre template twig pour afficher le formulaire : dans le dossier *Resources/views/Task* du bundle, saisissez le code suivant dans le fichier *index.html.twig* :

```

{% extends «AcmeDemoBundle::layout.html.twig» %}

{% block content %}

    {{ form_start(form) }}
    {{ form_errors(form) }}

    {{ form_row(form.description) }}

    {{ form_end(form) }}
{% endblock %}

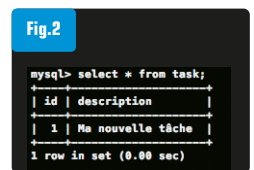
```



Formulaire d'ajout

Affichez maintenant votre page */task/create* : [Fig.1].

Remplissez et envoyez le formulaire. Et voilà ! Une nouvelle tâche est apparue en base de données [Fig.2].



Entrée en base de données

> Validation

Nous n'avons pas parlé de la ligne suivante dans le contrôleur : *if (\$form->isValid())*. En effet, il est possible d'implémenter une validation sur chacun des champs d'un formulaire.

Prenons un cas concret : à la soumission du formulaire, nous souhaitons que le champ *description* ne soit pas vide.

Pour cela, il vous faut ajouter l'annotation suivante dans le fichier *Task.php*, dans le dossier *src/Acme/DemoBundle* :

```

/**
 * @var string
 *
 * @ORM\Column(name="description", type="text")
 * @Assert\NotBlank()
 */
private $description;

```

Il faut également ajouter la ligne suivante :

```
use Symfony\Component\Validator\Constraints as Assert;
```

Rafraîchissez maintenant la page et tentez de soumettre le formulaire en laissant vide le champ *description*.

Puis, nous vérifions que les informations sont valides (pour en savoir plus sur la validation de formulaire <http://symfony.com/doc/current/book/forms.html#form-validation>).

CRUD

> La génération

Avec Doctrine, il est possible de générer ce que l'on appelle un CRUD :

- « *Create* » - La création ;
- « *Read* » - La consultation ;
- « *Update* » - La mise à jour ;
- « *Delete* » - La suppression ;

Il arrive très régulièrement d'avoir à faire ce type d'action sur les données.

Doctrine offre donc la possibilité de générer un CRUD à l'aide de la commande suivante : *app/console generate:doctrine:crud*. Avant toute chose, effacez les fichiers que nous avons créés précédemment, afin d'éviter tous conflits lors de la génération :

- la classe de formulaire « *TaskType.php* » ;



- la classe de contrôleur « TaskController.php » ;
- et le template « index.html.twig ».

Générons maintenant le CRUD sur l'entité **Task** à l'aide de la ligne de commande suivante : `app/console generate:doctrine:crud --entity=AcmeDemoBundle:Task --format=annotation --with-write --no-interaction`

Passons en revue les différents arguments présentés ci-dessus :

- `--entity` : indique le chemin logique vers l'entité (ici, l'entité **Task** située dans le bundle **AcmeDemoBundle**) ;
- `--format` : indiquer le type de configuration que vous souhaitez (ici, les annotations) ;
- `--with-write` : déclencher la génération des méthodes de contrôleur permettant l'écriture (*create* et *edit*) ;
- `--no-interaction` : empêcher les interactions avec l'utilisateur.

Huit méthodes de contrôleur ont été générées :

- `indexAction` ;
- `createAction` ;
- `newAction` ;
- `showAction` ;
- `editAction` ;
- `updateAction` ;
- `deleteAction` ;
- `createDeleteForm`.

Par ailleurs, les templates associés ont été générés dans le dossier `src/Acme/DemoBundle/Resources/views/Tasks`.

Et enfin, la classe de formulaire « `TaskType.php` » dans le dossier `src/Acme/DemoBundle/Form`.

NB : Le code généré vous permet d'avoir une base, le but est de le modifier pour répondre à vos besoins.

> Vue d'ensemble de la classe de contrôleur générée

Jetons un coup d'œil à la classe de contrôleur : dans un premier temps, on peut remarquer l'annotation `@Route` sur la classe indiquant le préfixe de chacune des routes.

```
/**
 * Task controller.
 *
 * @Route("/task")
 */
class TaskController extends Controller
{
    //
}
```

NB : Pensez à retirer la ligne `prefix: /task` dans le fichier `routing.yml` dans le dossier `src/Acme/DemoBundle/Resources/config` afin de ne pas avoir de conflit de préfix.

> Liste des enregistrements

Rendez-vous sur l'url « `/task/` » afin d'afficher la liste des entrées en base de données.

La méthode de contrôleur appelée est `indexAction` : On y retrouve la tâche que nous avons enregistrée grâce à notre premier formulaire. Le template correspondant est le fichier `index.html.twig` situé dans le dossier `src/Acme/DemoBundle/Resources/views/Task`.

> Ajout d'un enregistrement

Pour ajouter une nouvelle tâche, il vous faut cliquer sur le lien « Create a new entry ».

La méthode de contrôleur `newAction()` est appelée pour l'affichage du formulaire. Le template associé est nommé `new.html.twig`. Dans le template, on peut voir à cette ligne que le formulaire est soumis sur la route ayant pour nom « `task_create` » :

```
<form action="{{ path('task_create') }}" method="post" {{ form
  _enctype(form) }}>
```

La route ayant pour nom « `task_create` » correspond à la méthode `createAction()` se chargeant d'effectuer la persistance de l'objet `task`.

NB : Réduire au maximum ses méthodes de contrôleur est une bonne pratique (vingt-cinq lignes en moyenne) afin de bien découper son code selon le modèle MVC et surtout garder une méthode lisible et maintenable.

IMPORTANT : La méthode « `bind` » est maintenant dépréciée en version 2.3 de Symfony. Il est fortement conseillé d'utiliser la méthode « `submit` » dans votre code.

Dans son contrôleur, il est important de toujours rediriger l'utilisateur après le traitement de son formulaire : ici, l'utilisateur est redirigé vers la route « `task_show` », soit la méthode de contrôleur `showAction()`.

> Edition d'un enregistrement

En cliquant sur le lien « Edit », l'utilisateur est redirigé vers la route « `task_edit` », soit la méthode `editAction()`.

Dans le même principe que pour l'ajout, le formulaire est soumis sur la route « `task_update` », soit la méthode de contrôleur `updateAction()`.

> Suppression d'un enregistrement

La suppression se fait via le bouton « Delete » redirigeant vers la route « `task_delete` », soit la méthode de contrôleur `deleteAction()`. La méthode se charge simplement de récupérer l'enregistrement en base de données et de le supprimer via l'entity manager.

> Vue d'ensemble de la classe de formulaire générée

Le code que l'on peut retrouver dans la classe n'est pas très différent de celui que nous avons écrit ensemble.

Ceci dit, la méthode `setDefaultOptions` configure certaines options de votre formulaire : ici, l'option « `data_class` » indique l'entité sur laquelle on associe le formulaire. Le sujet des formulaires est très large. Pour aller plus loin, plusieurs possibilités :

le book *symfony2* - <http://symfony.com/doc/current/book> ;

le code du composant formulaire (dans le dossier `vendor/symfony/symfony/src/Symfony/Component/Form`) ;

les cookbooks - <http://symfony.com/doc/current/cookbook/form/index.html> .

A bientôt pour la dernière partie d'initiation au framework *Symfony2*.

Sarah Khalil - Chef de Projet technique

SensioLabs

sarah.khalil@sensiolabs.com

OAuth : exemple d'implémentation en C#

Cet article a pour but de vous montrer pas à pas l'implémentation de OAuth 1.0 en C# dans une portable class library, la rendant utilisable dans n'importe quel type de projet. Nous nous baserons sur HttpClient, sorti il y a peu en version stable et disponible sur NuGet. Pour une explication théorique de « pourquoi OAuth et comment ça marche », je vous invite à consulter un précédent article : <http://onefor4.wordpress.com/2013/05/26/oauth-1-0-en-bref/>

PRÉREQUIS

Pour pouvoir utiliser HttpClient, votre portable class library doit supporter au minimum Windows Phone 7.5 et le .NET Framework 4.5. Mais pour tout nouveau projet, cette limitation est peu contraignante. Les différentes classes externes dont vous allez avoir besoin sont uniquement officielles et se trouvent sur NuGet, cherchez donc « Microsoft http Client Libraries » pour l'HttpClient ainsi que « Microsoft.Bcl.Async » pour l'utilisation de l'assembly Microsoft.Threading.Task avec les mots clés async et await.

IMPLÉMENTATION

> Paramètres OAuth

Nous commençons tout d'abord par l'initialisation de paramètres propres à OAuth qui sont au nombre de 5 ou 6 (selon que l'on doit ou non utiliser l'*oauth token*). Pour le *nonce* nous nous contentons de générer un nombre aléatoire entre zéro et cent millions. Pour le timestamp, il nous faut récupérer le nombre de secondes écoulées depuis le 1er janvier 1970. Le timestamp et le nonce peuvent servir au serveur à limiter le nombre de requêtes pour un même couple nonce/timestamp. Pour la version de oauth, nous utilisons ici la version 1.0. Pour la méthode de signature, nous utilisons la méthode HMAC-SHA1 utilisée, entre autres, par Twitter et Tumblr. Et quant à la *consumer key* et l'*oauth token* (pour ce dernier, si applicable), ceux-ci sont propres à l'application cliente et à l'utilisateur final.

```
private static Dictionary<string, string> InitOAuthParameters
(string oauthConsumerKey, string accessToken)
{
    TimeSpan ts = DateTime.UtcNow - new DateTime(1970, 1, 1, 0, 0, 0);
    string oauthtimestamp = Math.Round(ts.TotalSeconds).ToString();
    string oauthnonce = new Random().Next(100000000).ToString();

    Dictionary<string, string> parameters = new Dictionary
<string, string>();
    //oauth parameters
    parameters.Add(«oauth_nonce», oauthnonce);
    parameters.Add(«oauth_timestamp», oauthtimestamp);
    parameters.Add(«oauth_version», «1.0»);
    parameters.Add(«oauth_signature_method», «HMAC-SHA1»);
    parameters.Add(«oauth_consumer_key», oauthConsumerKey);

    if (!string.IsNullOrEmpty(accessToken))
    {
        parameters.Add(«oauth_token», accessToken);
    }
    return parameters;
}
```

> Générer la signature

Signer une requête se fait en trois étapes : tout d'abord la génération de la *signature base string*, son hachage avec la ou les clés privées et son ajout au *Authorization Header*. Pour générer la chaîne de caractères à hacher, nous avons besoin de l'url de la requête sans les paramètres, la méthode HTTP utilisée (GET, POST, PUT, DELETE, PATCH, ...), les paramètres OAuth générés précédemment, les paramètres propres à la requête ainsi que la clé *consumer secret* et, si applicable, la clé *access token secret*. La *signature base string* est composée, dans l'ordre, de la méthode HTTP en majuscule, un caractère '&', l'url encodée, un autre caractère '&' puis les paramètres, séparés eux aussi par des '&'. Vérifiez bien l'ordre de chaque paramètre (ordre alphabétique), la case, ainsi que l'encodage qui est fait deux fois pour les paramètres ! Ici, nous tombons sur une limite des portable class library : l'espace de nom «System.Security.Cryptography» n'a pas été porté, mais il vous suffit d'assigner la méthode «ComputeHmacSha1» lors de l'utilisation de cette PCL par le biais d'une classe abstraite ou par delegate (pour cela, reportez-vous à l'article de Loïc Rebours sur l'inférence de type dans un numéro précédent)

```
private static string GenerateSignature(string methodType,
string url, Dictionary<string, string> authParameters, Dictionary
<string, string> contentParameters, string consumerSecret,
string accessTokenSecret = null)
{
    string baseString = string.Empty;
    string hash = string.Empty;

    string key = EscapeOAuthString(consumerSecret) + «&»;

    if (!string.IsNullOrEmpty(accessTokenSecret))
    {
        key += EscapeOAuthString(accessTokenSecret);
    }

    byte[] keyBytes = Encoding.UTF8.GetBytes(key);

    List<KeyValuePair<string, string>> sorted = new List<Key
ValuePair<string, string>>();

    foreach (KeyValuePair<string, string> pair in authParameters)
    { sorted.Add(pair); }
    if (contentParameters != null)
        foreach (KeyValuePair<string, string> pair in content
Parameters) { sorted.Add(pair); }

    StringBuilder sb = new StringBuilder();

    sorted = sorted.OrderBy(p => p.Key).ToList();

    foreach (KeyValuePair<string, string> pair in sorted)
    {
        //même si la chaîne finale est encodée, nous devons l'encoder
une première fois
        sb.Append(EscapeOAuthString(pair.Key));
    }
}
```



```

sb.Append(«=»);
//même si la chaîne finale est encodée, nous devons l'encoder
une première fois
sb.Append(EscapeOAuthString(pair.Value));
sb.Append(«&»);
}
sb.Remove(sb.Length - 1, 1);

baseString = methodType + «&» + EscapeOAuthString(url) +
«&» + EscapeOAuthString(sb.ToString());

byte[] hashBytes = ComputeHmacSha1(Encoding.UTF8.GetBytes
(baseString), keyBytes);

hash = Convert.ToBase64String(hashBytes);

return hash;
}

```

Pour la méthode permettant d'encoder les différentes parties, il suffit de remplacer les caractères non alphanumériques et ne faisant pas partie de l'ensemble «-._~» par un caractère pourcent [%] suivi de l'équivalent du caractère en hexadécimal [le caractère « deux points » ':' devient ainsi «%3A»].

```

private static readonly string unreservedChars = «ABCDEFGHIJKL
MNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789-._~»;

private static string EscapeOAuthString(string toEscape)
{
    StringBuilder builder = new StringBuilder();

    foreach (char c in toEscape)
    {
        if (unreservedChars.Contains((c.ToString())))
            builder.Append(c);
        else
            builder.Append('%' + string.Format(«{0:X2}», (int)c));
    }

    return builder.ToString();
}

```

> Header d'autorisation

Le header d'autorisation est ce qui va permettre au serveur de vérifier la signature de la requête, à vous donc de le composer des paramètres OAuth précédents en y ajoutant la signature, avec comme clé «oauth_signature». Le schéma du header est ici tout simplement «OAuth».

```

private static string CalculateOAuthHeader(Dictionary<string,
string> oauthParams)
{
    StringBuilder sb = new StringBuilder(«OAuth »);

    foreach (var pair in oauthParams.OrderBy(kvp => kvp.Key))
    {
        sb.Append(pair.Key + «=\» + EscapeOAuthString(pair.Value) + «\»,»);
    }
}

```

```

return sb.Remove(sb.Length - 1, 1).ToString();
}

```

> Construire la requête avec HttpClient

A présent, il ne nous manque plus qu'à construire la requête en s'aidant des différentes méthodes présentées précédemment. Le code suivant peut s'appliquer pour envoyer une requête avec oauth vers n'importe quelle url, avec n'importe quels paramètres et méthode HTTP.

```

public static async Task<HttpResponseMessage> SignRequest(Http
Method method, string requestUrl, string consumerKey, string
consumerSecret, Dictionary<string, string> contentParams,
string oauthToken, string accessTokenSecret)
{
    HttpClient client = new HttpClient();

    var oauthParams = InitOAuthParameters(consumerKey, oauthToken);

    string signatureBaseString;
    var signature = GenerateSignature(method.Method, requestUrl,
oauthParams, contentParams, consumerSecret, out signatureBase
String, accessTokenSecret);

    if (method.Method == HttpMethod.Get.Method && contentParams
!= null && contentParams.Count > 0)
    {
        requestUrl += «?»;
        foreach (var item in contentParams.OrderBy(kvp => kvp.Key))
        {
            requestUrl += EscapeOAuthString(item.Key) + «=» + Escape
OAuthString(item.Value) + «&»;
        }
        requestUrl = requestUrl.Remove(requestUrl.Length - 1, 1);
    }
    oauthParams.Add(«oauth_signature», signature);

    string authHeader = CalculateOAuthHeader(oauthParams);
    HttpRequestMessage request = new HttpRequestMessage(method,
requestUrl);
    request.Headers.Add(«Authorization», authHeader);

    if (method.Method != HttpMethod.Get.Method && contentParams
!= null)
    {
        request.Content = new FormUrlEncodedContent(contentParams);
    }

    HttpResponseMessage response = await client.SendAsync(request);
    return response;
}

```

EXEMPLE DE TWITTER

Cette implémentation de OAuth est pleinement utilisable pour s'authentifier à Twitter. Comme prérequis, allez simplement sur <http://dev.twitter.com/>, créez votre application et récupérez la *consumer key* et *consumer secret* associés. L'exemple qui suit est fait

dans une application console toute simple pour vous montrer le fonctionnement de la PinAuth de Twitter.

> Récupération des token client temporaires

La première étape va être de demander à Twitter une paire de token temporaire lui permettant par la suite de signer la requête de demande d'*access token* de l'utilisateur. Cette demande se fait en POST à l'url https://api.twitter.com/oauth/request_token et la requête est signée simplement avec la *consumer secret*.

```
//Récupération de client token temporaire
string requestTokenUrl = «https://api.twitter.com/oauth/request_token»;
Dictionary<string,string> contentParams = new Dictionary<string, string>();

var requestTokenResponse = await Requester.SignRequest(
    HttpMethod.Post,
    requestTokenUrl,
    consumerKey,
    consumerSecret,
    contentParams,
    null,
    null);

string requestTokenResponseJson = await requestTokenResponse.Content.ReadAsStringAsync();
string[] splitResponse = requestTokenResponseJson.Split('&');
string oauthToken = «»;
string oauthTokenSecret = «»;
foreach (string param in splitResponse)
{
    string[] paramSplit = param.Split('=');
    if (paramSplit[0]==»oauth_token»)
    {
        oauthToken = paramSplit[1];
    }
    else if (paramSplit[0]==»oauth_token_secret»)
    {
        oauthTokenSecret = paramSplit[1];
    }
}
```

> Créer et utiliser l'url d'autorisation

Ensuite, nous dirigeons l'utilisateur vers une page l'invitant à s'identifier auprès de Twitter puis à taper dans notre application le code PIN qui lui est affiché. L'url (requête GET dans le navigateur de l'utilisateur) est <https://api.twitter.com/oauth/authorize> auquel nous ajoutons le paramètre «*oauth_token*» ayant la valeur que nous venons de récupérer.

```
//montre à l'utilisateur la page de connexion de Twitter
string authorizeUrl = «https://api.twitter.com/oauth/authorize»;
string userAuthorizeUrl = authorizeUrl + «?oauth_token=» + oauthToken;
Process.Start(«iexplore.exe», userAuthorizeUrl);
```

> Récupérer l'access token de l'utilisateur

L'étape finale consiste à demander l'*access token* de l'utilisateur. La requête est en POST sur https://api.twitter.com/oauth/access_token en ajoutant comme paramètre «*oauth_verifier*» ayant pour valeur le

code PIN entré par l'utilisateur. La requête est signée par le couple *consumer secret*/*oauth token secret* (celui, temporaire, récupéré lors de la première requête).

```
//Récupération des user access token
Console.WriteLine(«Tapez le code PIN»);
var pin = Console.ReadLine();
Dictionary<string, string> contentParameters = new Dictionary<string, string>();
contentParameters.Add(«oauth_verifier», pin);
string accessTokenUrl = «https://api.twitter.com/oauth/access_token»;
var accessTokenResponse = await Requester.SignRequest(
    HttpMethod.Post,
    accessTokenUrl,
    consumerKey,
    consumerSecret,
    contentParameters,
    oauthToken,
    oauthTokenSecret);

string accessTokenResponseJson = await accessTokenResponse.Content.ReadAsStringAsync();
splitResponse = accessTokenResponseJson.Split('&');
foreach (string param in splitResponse)
{
    string[] paramSplit = param.Split('=');
    if (paramSplit[0] == «oauth_token»)
    {
        oauthToken = paramSplit[1];
    }
    else if (paramSplit[0] == «oauth_token_secret»)
    {
        oauthTokenSecret = paramSplit[1];
    }
}
```

> Requêtes authentifiées

Maintenant, l'*oauth token* et l'*oauth token secret* récupérés sont ceux propres à l'utilisateur, vous pouvez donc à présent les utiliser pour émettre des requêtes authentifiées. De plus, contrairement au couple de token client émis au début (sur l'url «*request_token*»), ceux-ci ne sont pas temporaires, vous pouvez donc les stocker dans le système de fichier ou sur une base de données pour les réutiliser plus tard sans redemander à l'utilisateur de s'authentifier.

Essayez donc par exemple, en utilisant ce couple de token de faire une requête GET sur https://api.twitter.com/1.1/statuses/home_timeline.json pour récupérer la timeline de l'utilisateur.

GÉNÉRALISATION

Il est possible que cette implémentation-là ne prenne pas en compte les spécificités de TOUS les services en ligne utilisant OAuth 1.0, notamment la nécessité de renseigner un Content-Type ou un Accept à la requête, mais ce code est facilement adaptable à des petites modifications de ce genre.

Mathieu Hollebecq - Développeur Windows Phone et Windows 8 à ClevLab - Microsoft Student Partner - <http://onefor4.wordpress.com>

Office 2013 : comment créer des Apps ?

Avec la sortie de la dernière version de la suite Office dans sa mouture 2013, Microsoft a décidé d'ouvrir les horizons de ce produit incontournable au développement d'Apps. Les « Apps for Office » marquent un tournant stratégique dans son développement. Mais concrètement, une App pour Office, ça fait quoi ? Ça fonctionne comment ? Et ce qui nous intéresse le plus en tant que développeur : comment développe-t-on une App pour Office ?

Cet article répondra à toutes ces questions en vous présentant tout d'abord les tenants et aboutissants des Apps pour Office, puis plus particulièrement le développement de celles-ci par le biais notamment d'un petit atelier dans lequel nous créerons notre première App. Nous terminerons enfin par la présentation de l'Office Store et les opportunités offertes par celui-ci.

LES APPS OFFICE, EN DÉTAIL

> Qu'est-ce qu'une App Office ?

Une App Office n'est autre qu'une page web intégrée dans les applications de la suite Office ! En effet, aussi étonnant que cela puisse paraître, les Apps Office sont bâties essentiellement sur les technologies web telles que HTML5, CSS3, Javascript et les API REST.

Par le biais d'une API relativement riche, que nous détaillerons dans la suite de cet article, une App Office va pouvoir interagir avec l'application cliente dans laquelle elle est hébergée afin d'en étendre les fonctionnalités et répondre ainsi à des besoins spécifiques.

> Les différents types d'Apps

Il existe plusieurs types d'Apps pour Office dépendant à la fois du logiciel client mais aussi de la portée de l'App.

En fonction de son type, elle sera affichée à différents endroits au sein de l'application cliente.

Nous allons détailler les différents types d'Apps pour Office qui existent et que nous pouvons créer.

Tout d'abord, les « TaskPane Apps », s'intégrant dans un panneau adjacent au contenu du logiciel client (c'est le cas de l'application

Currency Converter par exemple). Ce type d'application est disponible dans Word, Excel, PowerPoint et Project, dans le client lourd ou dans l'application web (Office web App). Ensuite, les « Content Apps », s'affichant au cœur du document en cours d'édition. Celles-ci sont uniquement disponibles dans Excel dans sa version client lourd ou web application. Enfin, les « Mail Apps », s'ajoutant comme actions disponibles sur un mail ou une tâche reçue dans outlook (client lourd et web application) [Fig.2].

COMMENT FONCTIONNE UNE APP OFFICE

> Hébergement et déclaration d'une App Office

Une App Office, pour être accessible aux utilisateurs, doit être hébergée quelque part au même titre que n'importe quelle application web. Pour cela, nous pouvons très bien opter pour un hébergement sur un serveur privé nous appartenant ou celui d'un hébergeur, mais nous pouvons aussi choisir une solution Cloud telle qu'Azure. Cette dernière a le gros avantage de la scalabilité aisée, ainsi que la continuité et qualité de service que nous lui connaissons. Par ailleurs, la tâche d'hébergement est alors fortement simplifiée

lorsque l'on dispose d'un abonnement Office 365. En effet, dans ce cas l'App est hébergée directement dans notre instance, sans avoir à se soucier des problématiques de disponibilité [Fig.3].

Peu importe le mode d'hébergement retenu, chaque App déclarera dans un fichier XML (le Manifest) différentes informations concernant son identité et ses fonctionnalités. Il contiendra l'adresse de la page web de l'App, son type (TaskPane app, Content app, Mail app), les applications de la suite Office avec lesquelles elle est compatible ainsi que les permissions qu'elle requiert. Nous verrons lors de la présentation des outils que Visual Studio 2012 permet de facilement générer ce fichier lors de la création du projet.

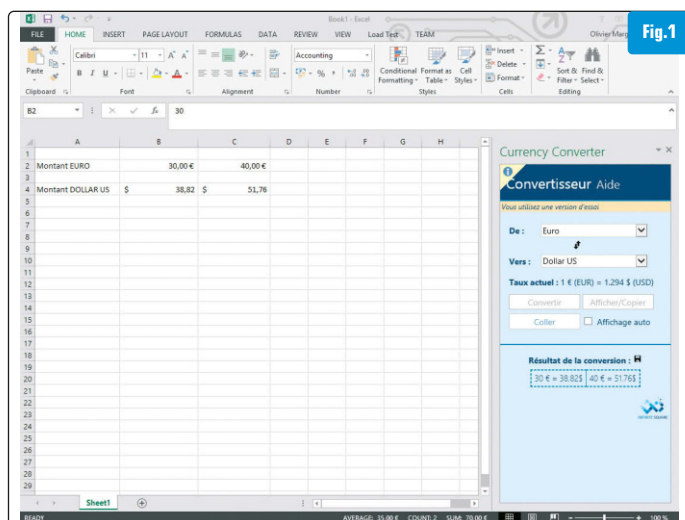


Fig.1

Un petit exemple d'application, avec Currency Converter pour Excel, permettant de réaliser des conversions monétaires dans différentes devises. Réalisation: Gaëtan BOUVERET - Infinite Square [Fig.1].



Fig.2

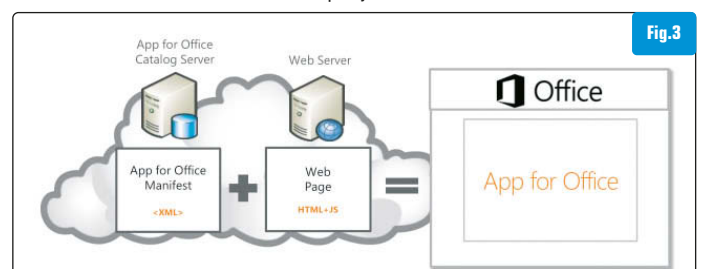


Fig.3

> Un tout nouveau framework et une API Javascript

Les Apps Office reposent sur un tout nouveau framework dénommé **WEF** (Web Extensibility Framework). Celui-ci permet à une page web d'être intégrée dans une application Office et d'exécuter du code pouvant interagir avec des documents Office et des objets Exchange, selon certaines contraintes.

Afin d'interagir avec les applications Office clientes, les Apps disposent d'une API en javascript. Celle-ci est présente dans l'espace de nom **Microsoft.Office.WebExtensions**. Il est important de noter que peu importe l'application cliente pour laquelle vous voulez créer une app, vous utiliserez toujours cette même API. Le premier point d'entrée de l'API Office est **Office.context** qui contient les informations sur le contexte d'exécution de l'App telle que la langue d'affichage de l'application cliente par exemple, mais aussi et surtout trois objets qui nous intéressent tout particulièrement :

Office.context.document permettant de lire et écrire du contenu dans un document Office ainsi que mettre en place des events handlers sur des changements dans ce document.

Office.context.roamingSettings permettant de lire et écrire des informations qui seront stockées dans le document.

Office.context.mailbox permettant d'accéder aux objets Exchange dans le cas d'une Mail App [Fig.4].

Dans un document Office, une App peut lire et écrire des données pouvant être de trois types différents. Il peut s'agir de texte, correspondant à un paragraphe dans un document Word par exemple ou encore une cellule d'une feuille Excel, d'un tableau à une dimension, correspondant à une colonne dans un document Excel par exemple, ou encore une matrice, correspondant à un tableau à deux dimensions. Il est aussi possible de lire et écrire du contenu de type Office Open XML. Les principales interactions avec un document Office se font lors de la sélection (ou du changement de sélection) d'un contenu grâce aux events handlers. Mais l'API Office offre aussi un mécanisme de binding permettant d'agir directement au niveau d'une section d'un document. Celui-ci permet par exemple d'effectuer une action lorsqu'une plage de données est modifiée. Le binding entre l'App et le document peut être effectué sur les trois types de données décrites précédemment (texte, tableau et matrice).

DES OUTILS À NOTRE DISPOSITION

Des outils adaptés et facilitant la tâche

Microsoft a rendu la création d'Apps facile dans Visual Studio 2012 grâce à un modèle de projet totalement dédié à cela. Si ce n'est déjà

fait, il faudra par contre installer les outils dédiés, intitulés «Microsoft Office Developer Tools for Visual Studio 2012» et disponibles en version finale à cette adresse <http://aka.ms/OfficeDevToolsForVS2012>.

Nous reviendrons plus en détail sur le développement d'Apps Office dans Visual Studio 2012 dans la suite de cet article, et notamment lorsque nous créerons notre première App. Nous allons nous intéresser maintenant à un outil totalement nouveau introduit par Microsoft et dénommé «Napa» !

«Napa» est un éditeur de code disponible en tant qu'App, que vous pouvez installer sur une instance Office 365, dans un site SharePoint de type «developer site». C'est donc un outil 100% Cloud qui ne requiert aucune installation sur votre machine. Un abonnement Office 365 étant nécessaire, vous pouvez souscrire à une version d'essai, accessible pendant 30 jours, à cette adresse <http://gg.gg/O365DEVTRIAL> [Fig.5]

Une fois «Napa» installé sur votre site développeur, vous pouvez créer tous les types d'Apps Office mais aussi SharePoint.

Même si l'éditeur semble assez sommaire, il est très bien conçu et dispose même de la saisie assistée sur le code Javascript [Fig.6].

Bien entendu, l'objectif de «Napa» n'est pas de remplacer Visual Studio, mais il permet une première approche du développement d'Apps. Il est d'ailleurs possible d'exporter très simplement la solution générée par celui-ci afin de poursuivre le développement dans Visual Studio 2012 par exemple.

PASSONS À LA PRATIQUE AVEC NOTRE PREMIÈRE APP OFFICE !

Afin de s'initier au développement d'App Office, nous allons créer notre première App pour Excel. Ce sera une App de type «Task pane» s'intégrant dans une feuille Excel. Elle permettra tout d'abord de lire le contenu des cellules sélectionnées, mais aussi d'écrire le

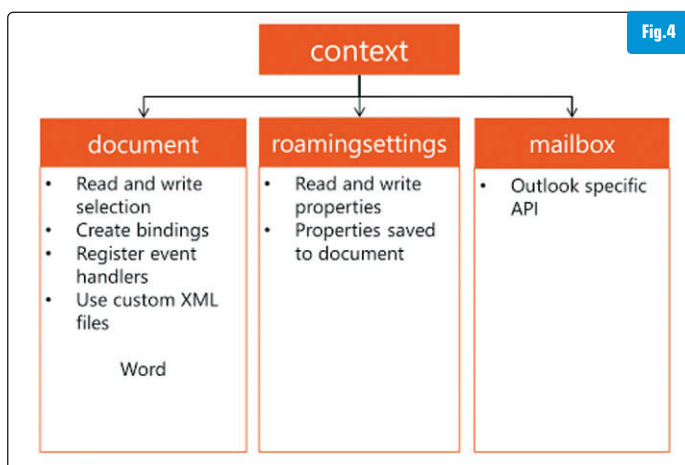


Fig.4

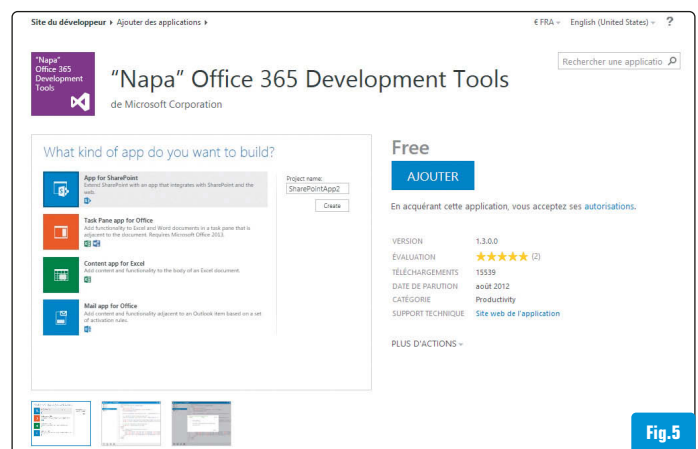


Fig.5

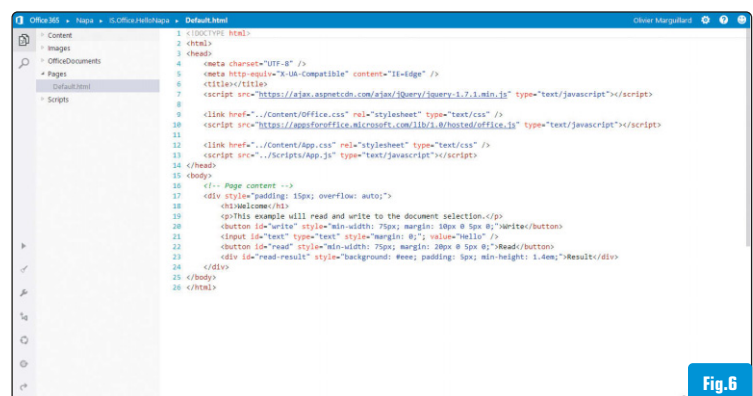


Fig.6

fameux «Hello World!» dans le document. Pour cela, nous utiliserons Visual Studio 2012 avec les «Microsoft Office Developer Tools for Visual Studio 2012», comme évoqué dans la partie précédente.

> Première étape : création du projet

Nous allons donc créer le projet dans Visual Studio, en nous basant sur le modèle «App for Office 2013». Donnons-lui un nom bien évocateur et le tour est joué, nous pouvons passer à l'écran suivant [Fig.7].

Une fois cette étape passée, un assistant Visual Studio nous demande de préciser le type d'App que nous souhaitons créer. C'est en fait ici que le fichier XML de déploiement de l'App est généré [Fig.8].

> Surprise, le projet de base contient déjà pas mal de choses !

Une fois le projet créé, nous nous rendons compte que nous avons directement une App prête à l'emploi. En effet, si nous lançons la solution, Excel s'ouvre et affiche notre App qui sait déjà faire la moitié de ce que nous voulions! Nous allons donc regarder ce qui s'est passé pour que nous en arrivions là ...

Tout d'abord, détaillons les projets et fichiers présents dans notre solution. Le premier projet est de type «App for Office 2013» et contient uniquement le fameux fichier XML de définition de l'App. Le second projet est un projet Web contenant les fichiers de style CSS

permettant de personnaliser l'interface utilisateur, notre page HTML qui contient la structure de l'App, les fichiers de code Javascript avec la logique métier de notre App, ainsi que les librairies telles que JQuery et les images embarquées dans notre App [Fig.9].

Nous allons maintenant nous intéresser au fichier *Home.js* contenant la logique métier de notre App. Lors de l'initialisation de notre page, on attache à l'événement click du bouton ayant l'ID «get-data-from-selection» la méthode «getDataFromSelection()». Celle-ci se sert alors de l'objet *Office.context.document* pour

récupérer le contenu de la sélection, par le biais de la méthode asynchrone «getSelectedDataAsync» prenant en paramètres le type de données retourné et la fonction de callback correspondante qui affichera le résultat dans une zone de notification.

```
(function () {
    «use strict»;

    // The initialize function must be run each time a new page is loaded
    Office.initialize = function (reason) {
        $(document).ready(function () {
            app.initialize();

            $('#get-data-from-selection').click(getDataFromSelection);
        });
    };

    // Reads data from current document selection and displays a notification
    function getDataFromSelection() {
        Office.context.document.getSelectedDataAsync(Office.Coercion
        Type.Text,
        function (result) {
            if (result.status === Office.AsyncResultStatus.Succeeded) {
                app.showNotification('The selected text is:', '«' +
                result.value + '»');
            } else {
                app.showNotification('Error:', result.error.message);
            }
        }
    );
    }
})();
```

Ajoutons notre «Hello world!»

Il va falloir rajouter un bouton d'action qui, lors du clic, inscrira dans la cellule sélectionnée, le fameux «Hello World!».

Pour cela, nous rajoutons dans la structure HTML de notre page, l'élément en question.

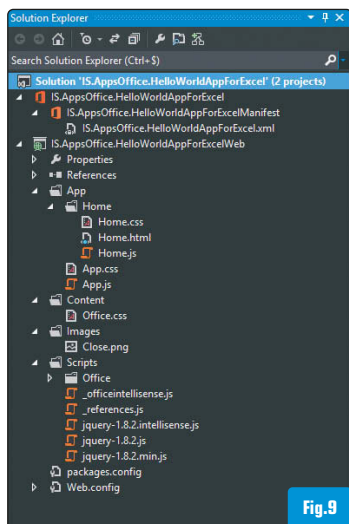


Fig.9

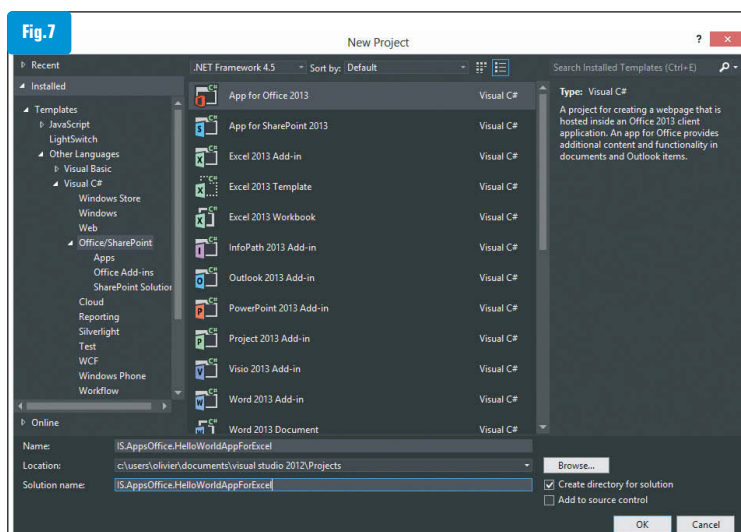


Fig.7

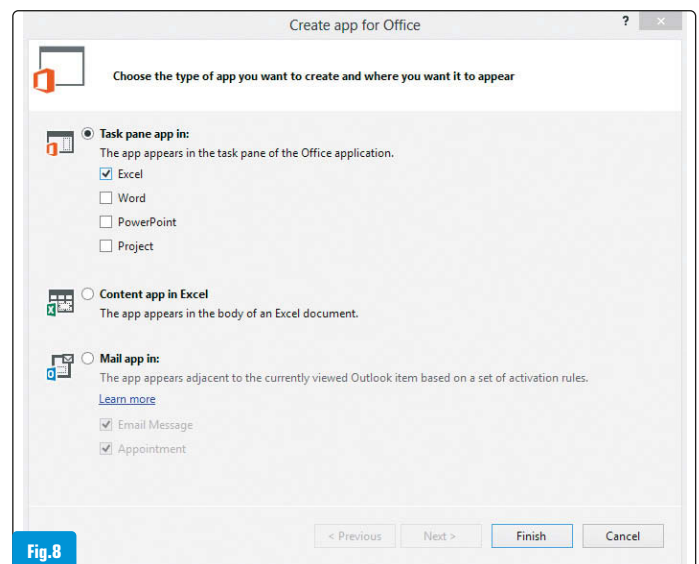


Fig.8

```

<!-- Page content -->

<div id="content-header">
  <div class="padding">
    <h1>Welcome</h1>
  </div>
</div>
<div id="content-main">
  <div class="padding">
    <p><strong>Add home screen content here.</strong></p>
    <p>For example:</p>
    <button id="get-data-from-selection">Get data from selection
</button>

    <br />
    <p>«Hello World!» : </p>
    <button id="hello-world-to-selection"></button>

    <p style="margin-top: 50px;">
      <a target="_blank" href="http://go.microsoft.com/fwlink/
?LinkId=276812">Find more samples online...</a>
    </p>
  </div>
</div>

```

Nous allons ensuite modifier le fichier de script *Home.js* afin de rajouter le code nécessaire pour effectuer cette action. Tout d'abord, nous attachons à l'événement clic sur le bouton nouvellement créé, la méthode «setHelloWorldToSelection()».

Cette dernière va alors se servir de l'objet *Office.context.document* pour remplacer le contenu de la cellule sélectionnée, par la phrase désirée.

Pour cela, on utilise la méthode asynchrone «setSelectedDataAsync» qui prend en paramètre le contenu à insérer ainsi que le type de celui-ci (texte, tableau ou matrice), puis la méthode de callback appelée en retour de cette méthode.

```

// The initialize function must be run each time a new page
is loaded
Office.initialize = function (reason) {
  $(document).ready(function () {
    app.initialize();

    $('#get-data-from-selection').click(getDataFromSelection);
    $('#hello-world-to-selection').click(setHelloWorldToSelection);

  });
};

// Set Hello World to the current document selection and display
the result
function setHelloWorldToSelection() {
  Office.context.document.setSelectedDataAsync('Hello World !',
{ coercionType : Office.CoercionType.Text}, function (result) {
  if (result.status === Office.AsyncResultStatus.Succeeded) {
    app.showNotification('Hello world successfull !');
  } else {
    app.showNotification('Error:', result.error.message);
  }
}

```

```

});
}

```

> Un peu de personnalisation ne fera pas de mal...

Afin de rendre les choses un peu (mais vraiment un tout petit peu!) plus jolies, nous allons personnaliser le design de notre App par le biais des feuilles de styles. Pour cela, nous modifions d'abord la structure HTML pour lui ajouter un conteneur à destination du logo de notre magazine favori, puis nous supprimons le texte du bouton affichant le «Hello World» qui sera remplacé par un logo (d'une boîte géniale!).

```

<!-- Page content -->
<div id="content-header">
  <div class="padding">
    <h1>Hello World for Programmez!</h1>
  </div>
</div>
<div id="content-main">
  <div class="padding">
    <div id="programmez-logo-container"></div>

    <p>For example:</p>
    <button id="get-data-from-selection">Get data from selection
</button>

    <br />
    <p>«Hello World!» : </p>
    <button id="hello-world-to-selection"></button>

    <p style="margin-top: 50px;">
      <a target="_blank" href="http://go.microsoft.com/fwlink/
?LinkId=276812">Find more samples online...</a>
    </p>
  </div>
</div>

```

Il est nécessaire d'ajouter au dossier *Images* de notre solution les deux logos en question. Puis nous modifions en conséquence le fichier *Home.css*.

```

#hello-world-to-selection
{

  background-image:url(«../../Images/InfiniteSquare-logo.png»);
  background-repeat:no-repeat;
  width:110px;
  height:80px;
  color:transparent;
  border:none;
}

#programmez-logo-container
{
  background-color:white;
  background-image:url(«../../Images/Programmez-logo.PNG»);
  background-repeat:no-repeat;

```



```
height:45px;
width:100%;
}
```

Et voilà, grâce à la simplicité des feuilles de styles CSS, nous venons de personnaliser rapidement le design de notre App. Cela reste certes très succinct, mais libre à vous d'imaginer des interfaces d'App agréables et un peu plus poussées [Fig.10].

LE MARCHÉ DES APPS POUR OFFICE

Des opportunités intéressantes

Bien que lancé depuis peu de temps, le Store Office semble très prometteur. En effet, la suite Office de Microsoft étant le produit le plus distribué, que ce soit en entreprise ou chez les particuliers, il n'est pas difficile d'imaginer les opportunités qu'offre ce nouveau marché des Apps. Cependant, le frein majeur à l'envolée des Apps Office est sans nul doute le fait qu'il soit nécessaire d'avoir la version 2013 d'Office, mais cela n'est qu'une question de temps avant que celle-ci soit de plus en plus répandue. De plus, le fait que les Apps soient compatibles avec les versions web (Office web App) des applications Office et même en consultation sur mobile, offre déjà beaucoup de possibilités.

> Quels canaux de distribution pour mes Apps ?

La distribution des Apps peut se faire de deux façons. Soit l'App est publiée sur l'Office Store directement, au tarif que vous aurez fixé, soit elle est publiée sur un catalogue d'entreprise. Pour les utilisateurs finaux, il y aura peut de différence entre ajouter une App provenant de l'Office store ou du catalogue d'Apps de l'entreprise [Fig.11].

> Gagner de l'argent avec les Apps, c'est possible !

Il est évident que l'aspect financier peut être un côté séduisant du développement d'Apps Office. Il existe par ailleurs trois moyens de gagner de l'argent avec ses Apps : soit en rendant l'application payante et alors vous décidez de son prix d'achat sur lequel vous partagerez 20% des revenus avec Microsoft, soit vous faites de l'in-app purchase et vous rendez par exemple certains modules de vos applications payants, ou encore, vous pouvez placer de la publicité sur votre App. Bien-sûr ces modes de rémunération ne sont pas exclusifs et vous pouvez sans problèmes les mélanger.

> Un système de licences flexible

Lors de l'achat d'une App, une licence est accordée à l'utilisateur. Celui-ci est identifié avec un compte Microsoft et la licence applicative est donc rattachée à ce compte. Cependant, lorsque nous ajoutons une App sur un document Word par exemple, celle-ci est

directement intégrée dans le document en question, et elle «voyagera» avec lui. Il est ainsi possible de transmettre ce document à une personne n'ayant pas acheté l'App sans pour autant qu'elle lui soit inaccessible. Il est donc du ressort du développeur de l'App de s'assurer que l'utilisateur courant ait une licence valide pour celle-ci. Pour cela, une API de licensing est disponible afin de gérer ces aspects. De plus, il est possible de rendre une App uniquement disponible individuellement (une licence = un utilisateur) ou bien de donner la possibilité d'acquérir des licences pour l'ensemble d'une société à un tarif plus attractif.

> Une validation obligatoire et un store mondial avec des limites

Afin d'être disponibles sur l'Office Store, les Apps passent par une étape de validation du côté de Microsoft, au même titre qu'une application Windows 8 avant d'être publiée sur le Windows Store. Une fois cette étape passée, le «Seller dashboard» nous offre la possibilité de consulter le volume des ventes et de gérer nos Apps publiées [Fig.12].

Bien que disponible depuis n'importe où dans le monde et pour n'importe qui, l'Office Store est pourtant uniquement présent aux Etats-Unis pour le moment. D'autres stores ont ouvert un peu partout dans le monde, permettant ainsi d'adapter nos Apps aux différents marchés.

> A vous maintenant !

J'espère que cette petite introduction aux Apps pour Office vous a donné l'envie de créer les vôtres. Pour aller plus loin, je ne peux que vous recommander le site dev.office.com, véritable mine d'or d'informations. De plus, Microsoft a mis à disposition quelques exemples d'Apps que vous pourrez trouver à cette adresse: <http://gg.gg/APPS-SAMPLE>. Par ailleurs, n'hésitez pas à me contacter pour échanger sur le sujet, ce sera avec plaisir !

Olivier Marguillard
Consultant SharePoint
Infinite Square
omarguillard@infinitesquare.com

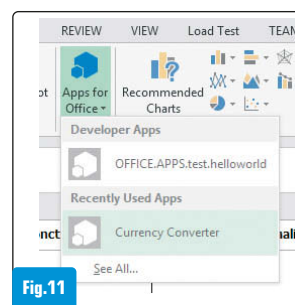


Fig.11

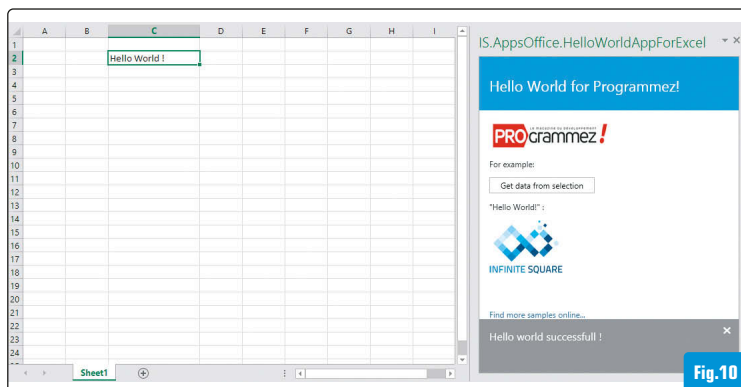


Fig.10

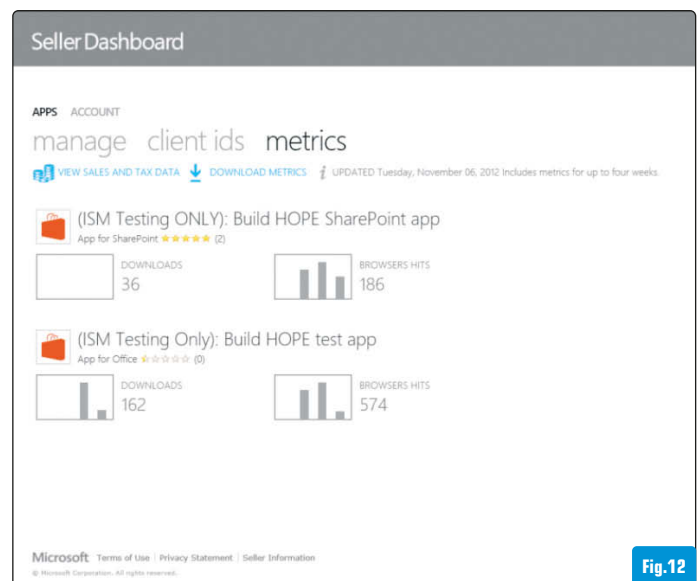


Fig.12

SQLite : une base de données embarquée

SQLite est le système de base de données embarqué permettant d'écrire ou de lire des données directement sur le système de fichiers. SQLite offre toutes les fonctionnalités principales d'une base de données avec des Tables, Index, Vues et Triggers. Cette librairie est très utilisée pour embarquer des données au sein de votre application cependant elle ne remplacera jamais des bases de données telles que MYSQL, ORACLE ou SQL SERVER pour le stockage de données côté serveur.

1 INSTALLATION

Pour Windows RT, SQLite est disponible sous la forme d'une extension pour Visual studio. Pour l'installer cliquer sur Tools > Extensions and Updates [Fig.1].

Rechercher « Sqli » et installer SQLite for Windows Runtime [Fig.2].

2 CRÉATION D'UN PROJET UTILISANT SQLITE

Sous Visual Studio, créer un nouveau projet d'application pour le Windows Store. Choisir le langage C# et partir d'un projet vide nommé VideoStore. Une fois le projet créé, nous allons ajouter la librairie SQLite et ses dépendances à notre projet. Pour cela, ajoutez une référence à votre projet [Fig.3]. Dans la section Windows > Extension, cocher « SQLite for Windows Runtime » et « Microsoft Visual C++ Runtime » [Fig.4]. Parce que SQLite est une librairie écrite en C++ notre projet ne peut plus être compilé en « Any CPU », il faudra donc spécifier une architecture (x84, x64 ou ARM) afin que le compilateur sache quelle version d'architecture utiliser. Cela aura une conséquence lors de la publication de notre application car nous

ne pourrons plus publier une application ciblant toute architecture, mais une version d'application pour chaque architecture existante, soit 3 versions. Pour modifier l'architecture rendez-vous dans « Configuration Manager. » [Fig.5]. Puis Modifier la plateforme afin de correspondre à l'architecture de votre machine [Fig.6].

Afin de lire ou écrire des données plus facilement avec SQLite en C#, nous allons installer un paquet NUGGET nommé « sqli-net » qui va nous permettre d'effectuer facilement des opérations de lecture et d'écriture en utilisant uniquement des objets. Pour l'utiliser cliquer sur « manage nugget package ... », puis rechercher et installer « sqli-net » [Fig.7 et 8].

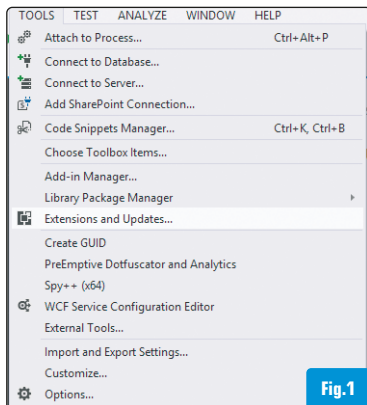


Fig.1

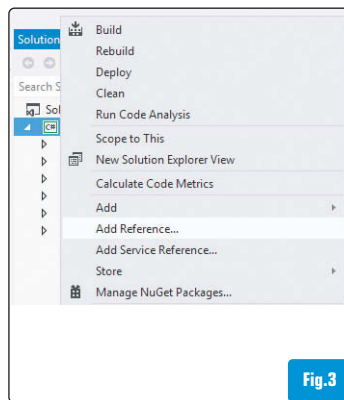


Fig.3

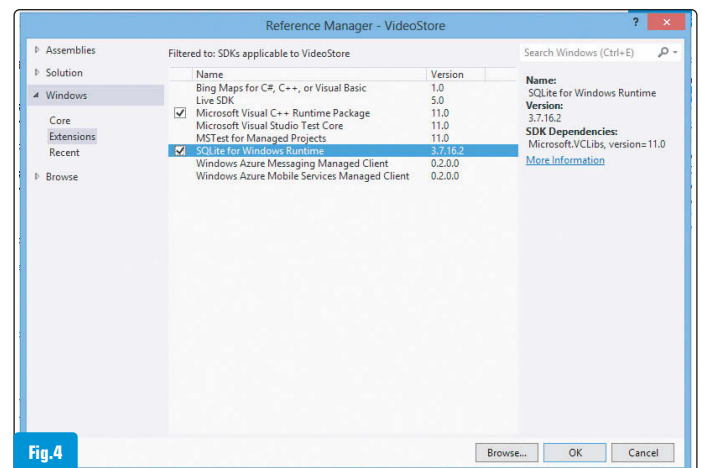


Fig.4

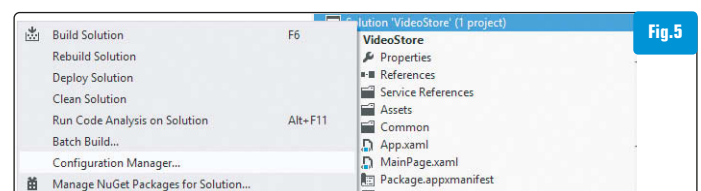


Fig.5

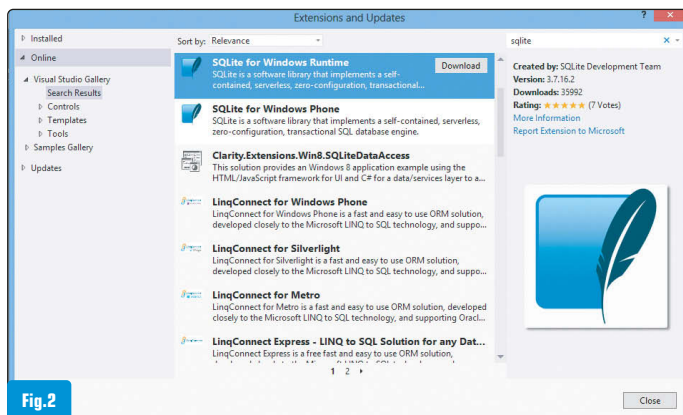


Fig.2

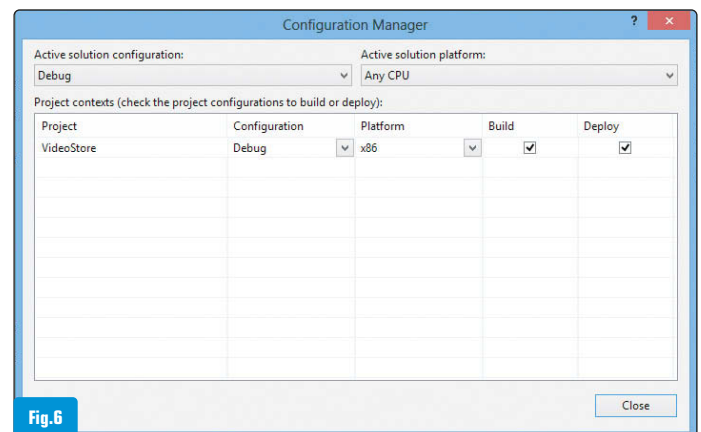


Fig.6

Ce paquet ajoute deux fichiers à notre solution :

- **SQLite.cs** : Ce fichier contient des méthodes synchrones d'accès à SQLite
- **SQLiteAsync.cs** : Ce fichier contient les méthodes asynchrones d'accès à SQLite

Afin d'être en harmonie avec la philosophie de Windows 8 nous n'utiliserons que les méthodes de « SQLiteAsync.cs »

3 CRÉATION D'UNE BASE DE DONNÉES LOCALE

Tout d'abord, nous allons définir les types d'informations à stocker en base. Dans notre exemple, nous allons utiliser des Catégories afin de pouvoir classer chaque type de vidéo. Nous aurons également des vidéos qui ont pour attributs un nom et une durée :

```
public class Video
{
    [PrimaryKey(), AutoIncrement()]
    public int ID { get; set; }

    public String Name { get; set; }

    public double Duration { get; set; }

    [Indexed()]
    public int GenreId { get; set; }
}

public class Genre
{
    [PrimaryKey(), AutoIncrement()]
    public int ID { get; set; }

    public String Title { get; set; }
}
```

Ajouter une classe nommée « StoreDataBase » à votre projet, cette classe va nous servir de point d'accès pour l'utilisation de la base de données. Ajouter ensuite une méthode nommée « Initialize » qui va nous permettre d'initialiser la base de données.

```
public static String DBPath { get; set; }

public static async Task Initialize()
{
    DBPath = Path.Combine(ApplicationData.Current.LocalFolder.Path,
        «VideoStore.db»);
    //création des tables de manière asynchrone si elle n'existe pas
    SQLite.SQLiteAsyncConnection db = new SQLiteAsyncConnection
        (DBPath);
    await db.CreateTableAsync<Video>();
    await db.CreateTableAsync<Genre>();
}
```

La variable nommée « DBPath » va déterminer l'emplacement de notre base de données dans le système de fichiers.

La méthode `CreateTableAsync<T>` permet de vérifier si la table correspondant à l'objet « T » existe, et de la créer si nécessaire.

Nous allons ensuite nous assurer que notre base de données contient les données nécessaires au fonctionnement initial de notre application grâce à une méthode nommée « InitDB » :

```
public static async Task InitDb()
{
    SQLite.SQLiteAsyncConnection db = new SQLiteAsyncConnection
        (DBPath);
    var genres = await db.Table<Genre>().ToListAsync();

    //Si aucun genre n'est enregistré alors on enregistre des
    genres par défaut
    if (genres.Count == 0)
    {
        var genreTab = new[] {
            «Action»,
            «Comedie»,
            «Aventure»,
            «Drame»
        };
        foreach (var title in genreTab)
        {
            await db.InsertAsync(new Genre() { Title = title });
        }
    }
}
```

Cette méthode vérifie que nous avons des genres disponibles pour catégoriser nos vidéos et ajoute des genres par défaut si aucun n'est disponible.

4 AJOUT ET RÉCUPÉRATION DES DONNÉES

Toujours dans la classe « StoreDataBase » nous allons créer une méthode afin de récupérer une liste de vidéos en fonction des critères souhaités.

```
public static async Task<List<Video>> GetVideos(Predicate<Video>
    predicate = null)
{
    SQLite.SQLiteAsyncConnection db = new SQLiteAsyncConnection
        (DBPath);
    var videos = db.Table<Video>();
    if (predicate != null)
    {
        //si un filtre est spécifié alors on filtre les vidéos
        videos = videos.Where(v => predicate(v));
    }
    //retourne la liste de vidéos
    return await videos.ToListAsync();
}
```

L'avantage de « sqlite-net » c'est qu'il nous évite d'écrire des requêtes SQL en ne manipulant que des objets C# et surtout nous avons la possibilité d'effectuer des requêtes « LINQ » afin de pouvoir affiner à notre guise le résultat de chaque requête.

Ensuite la récupération de contenu s'effectue grâce au code suivant :


```
var videos = await StoreDataBase.GetVideos(video => video.
GenreId == actionGenreId);
```

Ici on ne récupère que les vidéos appartenant au genre « Action ». Il est important de noter que « sqlite-net » nous permet d'effectuer toutes nos opérations de manière asynchrone, ce qui est vital dans une application Windows 8, car cela permet de ne pas bloquer l'affichage de l'interface utilisateur.

Nous allons ensuite écrire la méthode nous permettant de sauvegarder des vidéos en base. Puisque sqlite-net ne permet pas l'utilisation de clé externe, nous allons vérifier avant chaque ajout que la catégorie de la vidéo existe bien en base de données.

```
public async Task<Boolean> SaveVideo(Video video)
{
    SQLite.SQLiteAsyncConnection db = new SQLiteAsyncConnection
(DBPath);
    var genre = await db.Table<Genre>().Where(g => g.ID == video.
GenreId).FirstAsync();
    if (genre == null)
        return false; //le genre de la vidéo n'existe pas en base
de données

    //vérifie si la vidéo existe déjà en base.
    var dbVideo = await db.Table<Video>().Where(v => v.ID == video.
ID).FirstOrDefaultAsync();
    if (dbVideo == null)
    {
```

```
//La vidéo n'existe pas alors on l'insère en base de données
await db.InsertAsync(video);
}
else
{
    //La vidéo existe alors on met à jour l'entrée en base de données
    await db.UpdateAsync(video);
}
return true;
}
```

5 PUBLICATION SUR LE WINDOWS STORE

Comme vu précédemment, l'utilisation de la librairie SQLite exclut la possibilité de créer une application ciblant tout type de CPU, il sera donc nécessaire de générer un paquet pour chaque architecture.

Pour cela, allez sur votre projet puis cliquez sur store > create app package, une fenêtre apparaît à l'écran et vous demande si vos paquets seront publiés sur le Windows Store, choisissez votre option et validez-la [Fig.9 et 10]. Vous arrivez ensuite sur la page de création de paquet. Dans la configuration du paquet désélectionner Any CPU et choisir les 3 architectures disponibles (x86, x64 et ARM). Le résultat de cette opération créera 3 versions de votre application qu'il faudra fournir lors de la publication afin que votre application soit disponible sur toutes les plateformes [Fig.11 et 12].

Lors de l'installation de l'application à partir du Windows store, la version installée sur la machine du client sera la version qui satisfera au mieux sa configuration. Ainsi les Tablettes ARM utiliseront la version ARM, les versions 32 bits de Windows utiliseront la x86 et les versions 64 bits utiliseront de préférence la version x64, mais peuvent utiliser x86 si nécessaire.

6 CONCLUSION

SQLite est un outil rapide et performant qui va nous permettre de stocker les données importantes de notre application. Il est important de noter qu'ici les fichiers sont sauvegardés en clair dans le système de fichiers de l'utilisateur et que son contenu est facilement modifiable. Il est donc important d'ajouter une notion de sécurité et de chiffrement si l'on désire sauvegarder des informations sensibles.

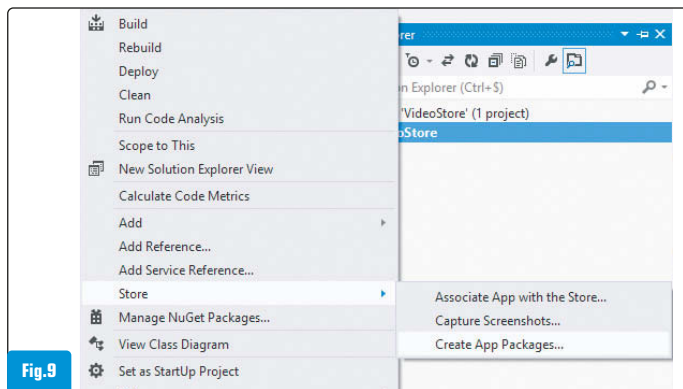


Fig.9

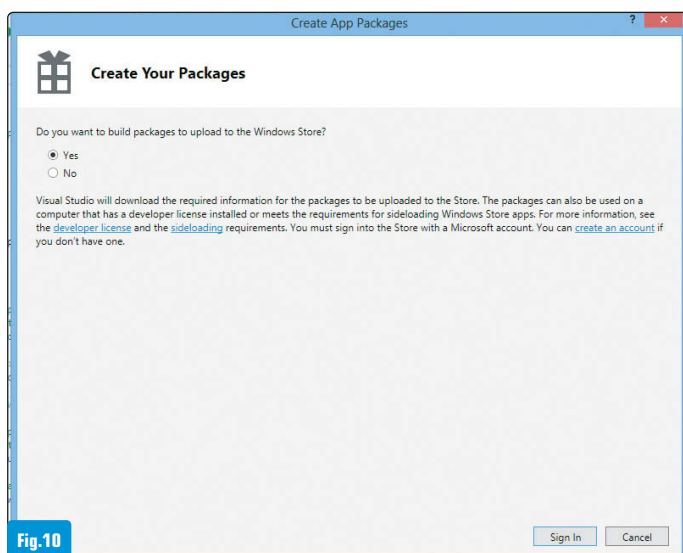


Fig.10

Fitzgerald Muiseroux
Spécialiste Windows 8, Itelios

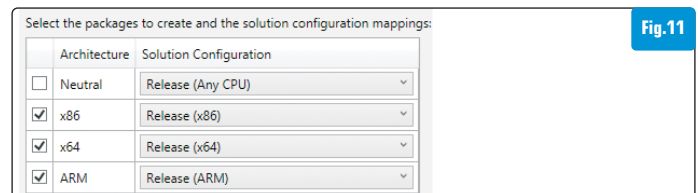


Fig.11

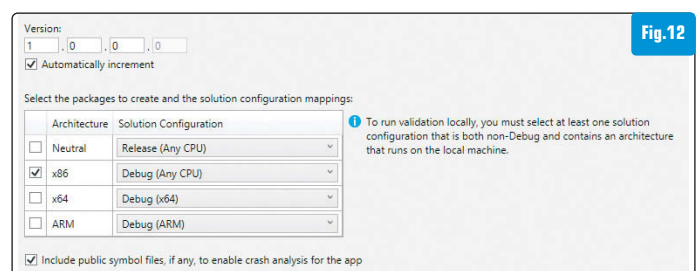


Fig.12

Bien démarrer sur iOS

Le développement d'application pour iOS (et pour les systèmes Apple en général) est un pas à franchir pour chaque développeur. Quasiment rien de ce que vous avez utilisé sur un autre système n'est réutilisable (ni le langage de développement, ni le SDK, ...)!

Dans ce genre de situation, il est important de bien maîtriser les bases dès le début pour ne pas se retrouver dans l'embarras plus tard. Nous vous proposons un dossier pour bien débuter sur iOS, axé sur deux thématiques : le langage Objective-C et les frameworks iOS à connaître.

LE LANGAGE DE DÉVELOPPEMENT

Objective-C est un langage particulier qui peut sembler horrible pour un novice. Le système de message, l'héritage du C, la gestion mémoire... Tous ces concepts peuvent paraître compliqués, mais en partant avec de bonnes explications, cela devient un jeu d'enfant ! Bien entendu nous ne pourrions pas tout voir en un seul article...

> Les bases à connaître

Dans un langage objet, il y a deux points à connaître pour commencer un développement : la syntaxe et les différents objets de base.

> Les objets de base

L'Objective-C étant un superset du langage C, il permet d'utiliser les variables de type C : int, char, short, float, etc. Cependant, Objective-C définit un ensemble d'objets de base permettant de se passer des variables de type C et ainsi permettre une écriture plus propre.

On peut classer les objets de base en deux catégories : les collections et les autres. Les collections permettent de stocker d'autres objets, sous forme de listes, de dictionnaires, d'ensembles, etc.

Il y a trois types de collections à connaître en Objective-C :

• NSMutableSet

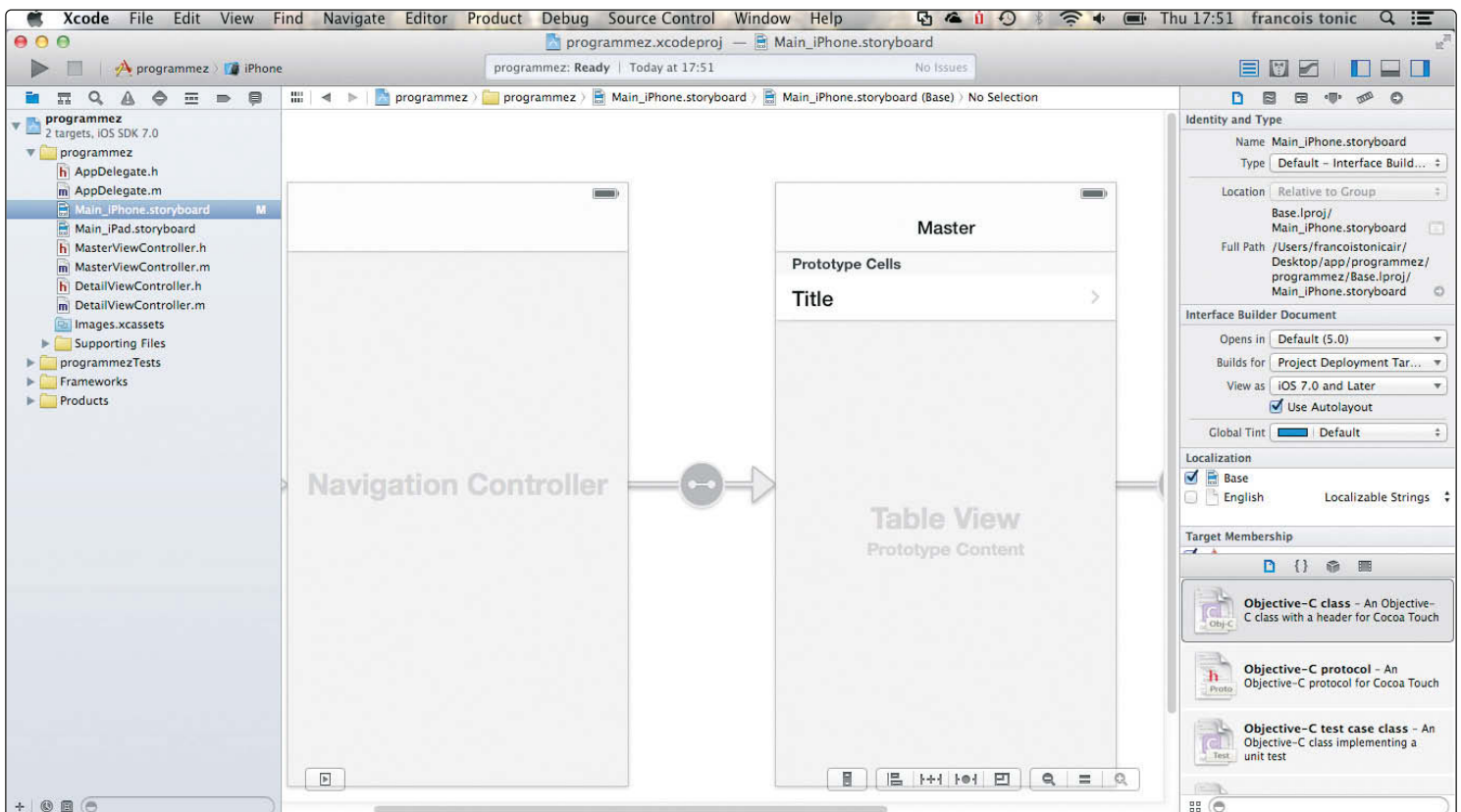
Permet de grouper un ensemble d'objets non-ordonnés : ni clé, ni index. Celui-ci est plus rapide d'accès qu'une liste ordonnée, mais ne garantit aucun ordre lors de l'insertion ou de la redistribution des objets.

• NSMutableArray

Permet de créer des listes d'objets. Ces listes, contrairement à la classe NSMutableSet sont classées par un index (numéroté).

• NSMutableDictionary

Permet de créer des dictionnaires de données. Cela s'apparente à une liste, non pas basée sur un index numéroté, mais sous forme de clé-valeur. Ces collections ne permettent pas de stocker autre chose que des objets Objective-C. Comme indiqué plus haut, il existe aussi d'autres types de base à connaître :



Environnement de développement Xcode, dernière version

• NSString

La classe NSString est la classe permettant la manipulation de chaîne de caractères en Objective-C

• NSValue

La classe NSValue est une classe assez particulière. Elle est destinée à faciliter le stockage de n'importe quelle structure en Objective-C : variable de type C, pointeur, objet, etc.

• NSNumber

La classe NSNumber hérite de la classe NSValue mais n'est destinée qu'au stockage de valeur numérique (int, float, etc.)

• NSData

La classe NSData permet le stockage et la gestion de données brutes (binaires)

Notez que n'importe quel objet Objective-C doit hériter d'une classe : NSObject. Sans cela votre objet ne sera pas considéré comme tel.

Immutable vs Mutable

En Objective-C, il y a une notion assez forte des objets «mutables» contre les objets «non-mutables».

Un objet qui ne spécifie pas s'il est «mutable» est par défaut «non-mutable» dans les conventions d'Apple. La plupart des objets vus précédemment ont tous leur pendant «mutable» : NSMutableString, NSMutableArray, etc.

> La syntaxe objet Objective-C

Par convention, la création d'objets en Objective-C se fait dans deux fichiers distincts :

- MaClasse.h où seront définies les méthodes et les propriétés accessibles de la classe
- MaClasse.m où le code sera fourni pour implémenter les différentes méthodes définies dans le header.

La définition d'une classe se fait par l'utilisation du mot-clé @interface :

```
@interface MaClasse : NSObject

@end
```

Vous noterez ici l'utilisation du mot-clé @end indiquant la fermeture de l'interface. Plusieurs éléments peuvent être définis dans une classe :

Propriétés

Les propriétés sont les attributs contenus dans l'objet, celles-ci sont visibles depuis l'extérieur.

Autrement dit, n'importe quelle portion de code créant un objet d'un type donné, peut accéder aux propriétés de celui-ci. Il n'y a pas de notion de «privé» ou «public» dans ce langage. Celle-ci se définissent avec le mot-clé @property : @property (nonatomic, strong) NSString* title.

Notons ici que nous ajoutons deux autres mots-clés : nonatomic et strong. Ceux-ci permettent d'indiquer au compilateur que l'accès à la propriété n'est pas réellement «Thread-Safe» et que le gestionnaire mémoire doit maintenir une référence sur l'objet. Il existe d'autres mots-clés, en voici quelques-uns :

- assign - Ne doit être utilisé que pour des propriétés de type C (et non pas sur des objets Objective-C)
- weak - Indique au compilateur qu'il ne doit pas ajouter de référence forte sur cet objet (à l'inverse de strong)
- retain - Identique à strong

- readonly - Indique que cette propriété n'est pas assignable (elle ne générera pas d'accessor en écriture)

Les propriétés d'une classe sont accessibles avec le . (e.g. myObject.myProperty).

Méthode

Les méthodes sont définies sans mot-clé particulier en Objective-C :

```
-(void) suffixWithTitle:(NSString*)str;
```

La syntaxe est assez particulière, elle est liée au système de messages en Objective-C. Le symbole au départ indique le type de méthode : méthode de classe (+), aussi nommée méthode statique, ou méthode d'instance (-) et le mot-clé entre parenthèses définit le type de retour (void, variable C ou objet).

La suite définit le nom de la méthode avec ses paramètres. Un message en Objective-C correspond à l'appel d'une méthode dans un autre langage. Ceux-ci sont cependant plus verbeux et plus proches d'une syntaxe humaine :

```
-(void) sendMessageWithTitle:(NSString*)title content:(NSString*)
content toEmail:(NSString*)email;
```

Cette méthode définit trois paramètres : title, content et email. Les méthodes sont appelées avec des crochets en Objective-C :

```
[myObject sendMessageWithTitle:@»Article» content:@»L'article
est prêt à être publié !» toEmail:@»redaction@mail.com»];
```

Variable d'Instance

Les variables d'instance sont propres à l'objet instancié et ne sont pas accessibles directement par un autre objet. Elles s'apparentent un peu aux attributs privés d'une classe en Java. Celles-ci se définissent juste après la déclaration de l'interface avec des accolades :

```
@interface MaClass : NSObject {
    NSString* maVariableDinstance1;
    NSString* maVariableDinstance2;
}
```

Elles sont directement accessibles dans les implémentations de la classe (sans utilisation du mot clé 'self').

Les protocoles

Un protocole en Objective-C s'apparente à l'interface en Java. Ils définissent uniquement des prototypes de méthodes et de propriétés. Toute classe qui implémente un protocole devra implémenter ces méthodes. Ils se définissent avec le mot-clé @protocol

```
@protocol MaClassDelegate

@end
```

Les protocoles sont très souvent utilisés en Objective-C pour établir le design pattern «Delegate», très répandu sur iOS.

> Les mots-clés à retenir :

- @interface - Définition d'une classe
- @implementation - Implémentation d'une classe

- @protocol - Définition d'un protocole (e.g. une interface en Java)
- self - Accès à l'objet courant (e.g. this en Java)
- super - Accès à l'objet parent de l'objet courant
- retain, release et autorelease - Non utilisable dans un projet ARC, ces mots-clés permettent la gestion mémoire de l'application.

> Embellir son code source

Objective-C étant un langage très verbeux, il est important que celui-ci soit propre et facile à comprendre afin de faciliter sa réutilisation.

> Catégorie VS Héritage

La plupart des langages objet possèdent le concept d'héritage. Celui-ci permet qu'un objet B hérite des méthodes et des propriétés d'un objet A.

Cela dit, bien que l'héritage soit très simple d'utilisation et simplifie le développement si celui-ci est bien pensé, cette notion n'est pas toujours très simple à maintenir.

Prenons l'exemple d'un projet iOS où tous les ViewControllers héritent de l'objet principal UIViewController.

Après quelques mois d'utilisation, on vous demande d'ajouter une fonctionnalité à toutes les vues ! Seulement, vous vous retrouvez avec une centaine de ViewControllers à mettre à jour.

Deux solutions :

- Créer une nouvelle classe héritant de la classe UIViewController, ajouter la fonctionnalité et modifier toutes les classes existantes pour hériter de cette nouvelle classe. Très simple à faire, mais un peu barbant... («*Developers are Lazy*»)
- Créer une catégorie de la classe UIViewController, ajouter la fonctionnalité et ajouter cette catégorie dans le Global Prefix Header du projet.

La première solution implique un changement (même minime) sur tous les ViewControllers du projet alors que la seconde non. Celle-ci est visiblement plus simple, plus propre et plus maintenable.

Cependant, une catégorie est limitée dans son utilisation, celle-ci ne permet que d'ajouter des nouvelles fonctionnalités, mais pas de modifier des fonctionnalités existantes ni d'ajouter de nouvelles propriétés. (Je vous invite à consulter l'article précédent dans Programmez n°165 : «Objective-C Expert» où il est question de «Method Swizzling» et de «Associated Object», deux concepts qui résolvent cette problématique)

> Le «Syntax Subscripting»

Le «Syntax Subscripting» est une fonctionnalité nouvelle apparue avec LLVM Compiler 4.0, le compilateur Objective-C utilisé par Apple. Celle-ci permet de réduire la taille du code en ajoutant deux styles d'écriture : dictionary-style ou index-style.

Elle est utilisée par Apple pour simplifier l'utilisation des objets NSArray ou NSDictionary. Prenons un exemple, le morceau de code suivant permet la création d'un dictionnaire et d'une liste de données sans «Syntax Subscripting» :

```
NSDictionary* dict = [NSDictionary dictionaryWithObjectsAndKeys:
    @»value», @»key», nil];
NSString* value = [dict objectForKey:@»key»];

NSArray* array = [NSArray arrayWithObjects:@»str1», @»str2», nil];
NSString* str = [array objectAtIndex:0];
```

Prenons le même code, mais avec «Syntax Subscripting» :

```
NSDictionary* dict = @{@"key» : @»value»};
NSString* value = dict[@"key»];

NSArray* array = @[«str1», «str2»];
NSString* str = array[0];
```

Le code est bien plus lisible. Ces deux nouveaux styles d'écriture sont possibles pour n'importe quelle classe. Pour cela, il suffit d'utiliser les méthodes :

- - (id)objectAtIndexedSubscript:(NSUInteger)idx;
- - (void)setObject:(id)obj atIndexedSubscript:(NSUInteger)idx;
- - (id)objectForKeyedSubscript:(id <NSCopying>)key;
- - (void)setObject:(id)obj forKeyedSubscript:(id <NSCopying>)key;

Elles permettent de définir l'action à réaliser lorsqu'on utilise les deux styles précédemment vus sur des objets.

> La gestion mémoire (Automatic Reference Counting)

La gestion mémoire en Objective-C était quelque chose de «barbare» il y a quelques années. Celle-ci s'apparentait à un Garbage Collector semi-automatique.

Celui-ci fonctionnait en tâche de fond pour supprimer les objets qui n'était plus référencés dans le code (release et retain). Apple a revu complètement cette stratégie et plutôt que de maintenir un Garbage Collector comme dans d'autres langages tels que Java, ils ont choisi de favoriser la gestion mémoire classique.

Cependant, afin de réduire les problèmes liés au développement, la gestion mémoire se fera durant la compilation et non plus durant le développement.

Là est toute la magie de ARC, lors de la compilation, il analyse le code source, et rajoute les méthodes de suppression d'objets lorsqu'il voit qu'un objet n'est plus référencé nulle part.

LES FRAMEWORKS À CONNAÎTRE

> Le SDK iOS

Parmi les frameworks natifs fournis par Apple, il y a les principaux et ceux qui ne servent que dans certains cas particuliers. Il y a aussi un framework nécessaire à n'importe quel développement en Objective-C: Foundation.framework, c'est lui qui apporte les bases du langage et ses concepts dans un projet.

> Les principaux frameworks

Le premier à connaître est celui qui est nécessaire à la réalisation d'une application iOS : **Cocoa**. Ou plus exactement *Cocoa Touch*, qui est le petit frère de Cocoa, le framework principal d'OS X. Cocoa fournit toute la couche applicative nécessaire au développement d'une application : Accès aux systèmes de fichiers, au réseau, la gestion des UI, le multi-threading, etc.

Celui-ci se nomme UIKit.framework dans Xcode. Apple fournit aussi un ensemble de frameworks pour simplifier l'utilisation de fonctionnalités du device, en voici quelques-uns à connaître absolument :

- QuartzCore.framework - Framework permettant l'animation 2D et la manipulation d'image et de vidéo.
- CoreGraphics.framework - Framework permettant la gestion graphique d'une vue ou d'une fenêtre ainsi que le dessin en deux dimensions.

- `CFNetwork.framework` - Framework d'accès à la couche réseau (socket BSD, TLS, SSL, etc.).
- `MapKit.framework` - Framework permettant l'utilisation de l'API Maps d'Apple (précédemment l'API Google Maps).
- `CoreLocation.framework` - Framework d'accès aux informations de géolocalisation du device.
- `CoreData.framework` - Framework de gestion des données persistées (SQLite, Fichiers binaires, etc.).
- `MediaPlayer.framework` - Framework permettant la lecture de fichier multimédia (vidéo/son).

Il y en a beaucoup d'autres que je vous invite à découvrir dans la documentation d'Apple, mais ceux-là vous permettront de faire 95% de vos développements iOS.

Bien que les frameworks fournis par Apple permettent d'effectuer toutes les tâches de notre développement. Il existe aujourd'hui un grand nombre de frameworks tiers simplifiant l'utilisation des frameworks natifs.

> Pour La gestion réseau

> AFNetworking

AFNetworking est LE framework réseau à connaître absolument. Développé principalement par @Mattt Thompson, il est aujourd'hui un framework de référence pour la réalisation des couches d'accès aux web services. Il se distingue par une écriture simple, utilisant massivement les blocks Objective-C et par une encapsulation de la gestion de serialisation/désérialisation des différents formats existants : JSON, XML, etc.

> SocketRocket

SocketRocket est un framework permettant la création d'un client WebSocket sous iOS. Très simple d'utilisation, il est basé sur le framework `CFNetwork.framework` d'Apple, ce qui lui permet d'utiliser les fonctionnalités intégrées de communication sécurisée TLS (via `CFStream`).

> RestKit

RestKit est un framework basé sur le framework AFNetworking, dédié à l'interconnexion avec un web service de type RESTful. Celui-ci propose, en plus d'AFNetworking, un ORM («Object Relational Mapping») intégré, compatible avec le framework `CoreData.framework` d'Apple qui est la référence en matière de gestion des données sous iOS.

> Pour faciliter le débogage et la distribution

La plupart des frameworks permettent de faciliter la tâche du développeur dans son travail quotidien. Cependant il existe d'autres types de frameworks, notamment pour simplifier le débogage ou la distribution des applications.

> TestFlight

TestFlight n'est pas juste un framework mais un service complet de distribution Over-the-Air d'applications. Celui-ci se base sur le mode de fonctionnement Ad-Hoc d'Apple. Il permet de restreindre l'accès à l'application pour que seules les personnes autorisées par le développeur puissent l'utiliser. TestFlight fournit une application OSX qui va détecter la génération d'un nouvel IPA pour l'envoyer automatiquement aux testeurs, facilitant ainsi la mise en ligne d'une nouvelle version. Il permet aussi d'avoir certains services en plus : Remote

Logging, Analyseur de Crash, Commentaires, etc. De plus, TestFlight est en train d'adapter son service pour fonctionner avec Android.

> CocoaLumberjack

CocoaLumberjack, ou Lumberjack, est un framework de logs. Il permet de gérer plusieurs logs à la fois avec différents niveaux de priorité. Il se dit plus rapide que la solution de log d'iOS (NSLog) et permet l'écriture des logs sur différents supports à la fois : base de données, fichier texte, xml, etc.

> Reveal

Reveal est un framework assez récent et encore en bêta. Il permet d'analyser en détail l'état courant du téléphone via le simulateur iOS. Cela permet de comprendre certains bugs.

Par exemple, il arrive parfois qu'une vue mal gérée soit cachée derrière d'autres vues, mais il nous est impossible de déterminer simplement que c'est celle-ci qui fait boguer notre application.

Reveal va afficher une vue en trois dimensions représentant l'état courant du téléphone où chaque composant présent sur l'écran sera détaillé.

> Chercher d'autres frameworks ?

Il existe bien d'autres frameworks pour iOS. La communauté, très active, implique qu'il y a beaucoup de développeurs qui enrichissent des frameworks existants ou en créent de nouveaux.

Les deux principales sources à connaître sont :

> GitHub

C'est un gestionnaire de codes sources publics, très à la mode, que les développeurs iOS utilisent très souvent pour héberger leurs frameworks.

Celui-ci propose un moteur de recherches efficace permettant une recherche par mots-clés et par langage de développement.

De fait, si vous saisissez «Network» ou «Grid View» et que vous spécifiez «Objective-C», vous avez de fortes chances d'obtenir des résultats probants (AFNetworking pour «Network»...)

GitHub — <http://github.com>

> CocoaControls

C'est un site internet servant d'annuaire des différents frameworks tiers disponibles sur la plateforme. Il se veut complet et impartial car ce sont les développeurs eux-mêmes qui contribuent en informant celui-ci directement. Ce site est devenu une référence en la matière. La plupart des développeurs n'hésitent pas à mettre en avant leur contenu sur le site dès la première publication sur le gestionnaire de codes sources publics. De fait, il arrive de trouver des composants qui ne sont pas encore terminés ou encore un peu bogués.

Depuis quelque temps, le site propose aussi l'achat de frameworks payants, cependant, contrairement à d'autres sites, les frameworks payants restent très minoritaires.

CocoaControls - <http://www.cocoacontrols.com>



Vincent Saluzzo

Ingénieur informatique

Expert Technique iOS

<http://www.vincentsaluzzo.com>

itelios

Améliorer son environnement de développement iOS

Xcode, l'environnement de développement pour iOS, est très agréable à l'utilisation. Avec les outils qu'il fournit (Interface Builder, Instruments, etc.), la prise en main est rapide pour les développements et les tests. Il existe cependant d'autres outils pour faciliter le développement au quotidien, je vous propose d'en découvrir quelques-uns.

POUR AMÉLIORER XCODE

Xcode permet l'ajout de fonctionnalités via une gestion de plug-ins. Bien que la documentation d'Apple à ce sujet soit assez mince, des développeurs ont créé divers plug-ins pour réduire notre temps de développement. Depuis quelque temps, un gestionnaire de paquets a été développé pour les plug-ins Xcode. Celui-ci s'apparente à un plug-in, dédié à la gestion d'autres plug-ins. Il se nomme Alcatraz. Vous pouvez le télécharger sur la page **GitHub** du projet : <http://mneorr.github.io/Alcatraz/>. Il est totalement gratuit et Open Source. Une fois téléchargé et installé, Alcatraz fournit une interface graphique dédiée au téléchargement de plug-ins, templates et thèmes pour Xcode [Fig.1]. Les plug-ins intégrés dans le repository sont une contribution des développeurs. Certains actuellement présents méritent d'être connus. Les voici :

> DLAutoresizingMaskPlugin

Ce plug-in permet de rendre la lecture/écriture des contraintes AutoresizingMask plus intuitive dans le code source [Fig.2]. Bien que cette fonctionnalité soit présente dans Interface Builder, il peut arriver que le développeur souhaite modifier le comportement

de l'auto-redimensionnement dans le code source, auquel cas aucun outil visuel n'est là pour l'aider, sauf s'il a ce plug-in.

> KSLImageNamed

Ce plug-in simplifie l'instanciation de la classe UIImage. La plupart du temps, une UIImage est instanciée avec le nom d'une image enregistrée dans Xcode avec la méthode

```
[UIImage imageNamed:@"myImage"];
```

Cependant, il est impossible de savoir à quoi correspond cette image depuis le code source. Ce plug-in résout ce problème-là en affichant l'image choisie par l'utilisateur sous forme de pop-in. De plus, il facilite la recherche d'une image à la volée depuis le code [Fig.3].

> Lin

Qui n'a jamais détesté devoir internationaliser une application en allant chercher dans le fichier de ressources quelle clé peut correspondre à telle chaîne de caractère... Lin simplifie ce travail en proposant les clés-valeurs correspondant au texte écrit directement depuis le code source, lorsque l'on commence à utiliser la fonction d'accès aux ressources localisées (NSBundleLocalizedString...).

De plus, il permet de parcourir les chaînes de caractères localisées directement depuis le code, Lin permet aussi de les éditer dans chaque langue disponible.

> OMColorSense

Ce plug-in permet de rendre plus "lisibles" les objets UIColor (ou NSColor) dans le code source en affichant une boîte de dialogue représentant la couleur instanciée.

Il existe d'autres plug-ins pour Xcode, dont certains ne sont pas référencés dans Alcatraz, n'hésitez pas à fouiller un peu sur le web.

LES OUTILS TIERS

Commençons par un logiciel, ou plutôt un plug-in pour OS X lui-même.

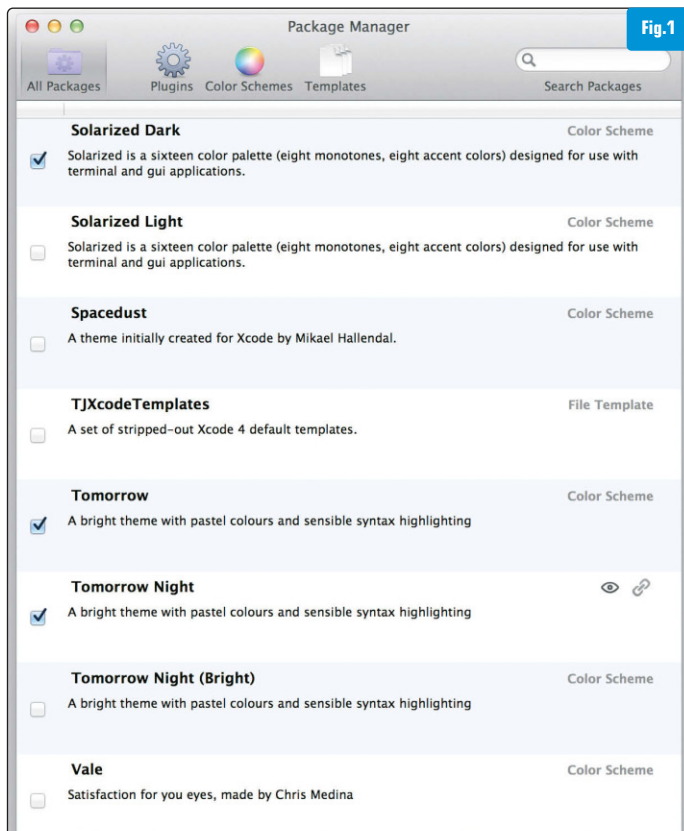


Fig.1

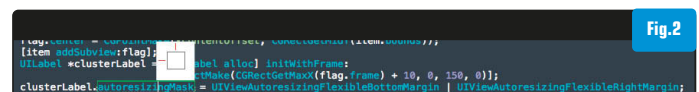


Fig.2

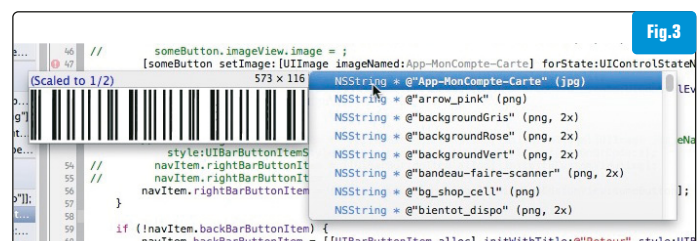


Fig.3

Il s'agit de Hex Color Picker disponible à l'adresse <http://wafflesoftware.net/hexpicker/>.

Comme son nom l'indique, ce plug-in ajoute un onglet au color picker de OS X, permettant la sélection d'une couleur par la simple saisie de son code hexadécimal ou de son nom de couleur.

Il sera très apprécié par les développeurs web mais aussi par les développeurs iOS qui ont écrit leur propre méthode d'instanciation de la classe UIColor basée sur un code couleur hexadécimal par exemple.

Site Web : <http://wafflesoftware.net/hexpicker/> - **Gratuit**

> CocoaRestClient

La plupart des applications iOS nécessitent une connexion internet. Elles sont, en général, consommatrices de Web Services de type RESTful. Cependant, développer un client RESTful sans tester le web service avant, *c'est un peu comme déployer une application sur l'AppStore sans l'avoir testée avant*, c'est mal !

Je vous présente donc un logiciel très simple d'utilisation, uniquement dédié aux possesseurs d'OSX (des versions similaires existent sur toutes les plateformes) : **CocoaRestClient**.

Disponible en téléchargement à l'adresse suivante <https://github.com/mmattozzi/cocoa-rest-client>, celui-ci est gratuit et Open Source, donc libre à vous de le modifier à votre guise.

(C'est ce que j'ai fait !). Il possède une fonctionnalité de sauvegarde des différentes méthodes, gère le formatage JSON et XML, supporte le SSL, etc.

Site Web : <https://code.google.com/p/cocoa-rest-client/> - **Gratuit**

> Resizer

Resizer fait partie de la famille de ces petits logiciels qui ne payent pas de mine mais qui nous facilitent le travail au quotidien. Sa tâche est simple : redimensionner automatiquement les images au format *retina* dans un format *non-retina*.

Pas plus, pas moins. Son interface fonctionne en drag&drop et les images redimensionnées sont plutôt de bonne qualité.

Cela permet de ne pas se prendre la tête à redimensionner chaque image créée dans le format non-retina. N'oublions pas... "Developers are Lazy" !

Site Web : <http://bonobolabs.com/> - **Gratuit (Disponible sur Mac AppStore)**

> ImageOptim

ImageOptim est un logiciel de la même catégorie que *Resizer*, simple, facile, il fait ce qu'on lui demande sans en faire plus.

Son but est de réduire l'empreinte mémoire des images au format PNG/JPEG. Pour cela, il utilise des bibliothèques existantes qui implémentent divers algorithmes de compression : AdvPNG, OptiPNG, Pngcrush, JpegOptim, etc.

Comme Resizer, l'interface fonctionne en drag&drop et les images perdent parfois plus de 50% de leur poids initial.

Site Web : <http://imageoptim.com/> - **Gratuit**

> xScope

xScope n'est pas un logiciel simple et mono-fonctionnel. Il serait plutôt le couteau suisse du développeur.

Les fonctionnalités présentes sont les suivantes : Mirroring, capture de dimensions, affichage de règles on-screen pour faire des mesures détaillées, loupe avec capture de couleur, guides visuels, etc. Celui-ci, une fois lancé, reste en tâche de fond dans la barre des

tâches d'OS X, ce qui permet d'accéder très facilement aux divers outils qu'il propose.

Site Web : <http://xscopeapp.com/> - **26,99€ (Disponible sur Mac AppStore)**

> iExplorer

iExplorer est un logiciel payant de la société **Macroplant** qui permet d'accéder aux détails d'un iDevice. Cela est très pratique lorsqu'on manipule le système de fichiers dans une application car il peut s'avérer que l'on obtienne des erreurs avec la classe NSFileManager du SDK sans réellement comprendre pourquoi. Cependant, il ne se limite pas qu'à cela. Vous pourrez aussi exporter vos musiques, messages, photos, ... de votre iDevice, faire un backup, etc.

Site Web : <http://www.macroplant.com/iexplorer/> - **35\$**

> StringsEditor

Ce logiciel permet de manipuler les fichiers de ressources localisés de Xcode, les fichiers .strings. Bien que Xcode nous permette de le faire nativement sous forme de fichier texte, l'utilisation d'un tableur est souvent plus rapide et plus intuitive pour les personnes qui ne développent pas. StringsEditor propose ce type d'interface et permet d'avoir accès à l'ensemble des traductions possibles d'un même fichier.

Site Web : <http://tickplant.com/stringseditor/> - **4,49€ (Disponible sur Mac AppStore)**

> SourceTree

SourceTree est un logiciel gratuit à posséder absolument pour tous ceux qui utilisent Git comme gestionnaire de contrôle de code source. Xcode propose une solution intégrée mais celle-ci entraîne parfois certains bugs désagréables en utilisation courante.

Ce logiciel permet donc de ne pas utiliser Xcode et de ne pas avoir à manipuler l'invite de commande, ce qui rend parfois les choses plus faciles. De plus, SourceTree intègre nativement le processus de gestion **Git-Flow** permettant ainsi de faciliter le travail en équipe sur un même projet, très important en entreprise.

Site Web : <http://www.sourcetreeapp.com/> - **Gratuit**

> Screentaker

Ce petit logiciel permet de générer des screenshots dans le style Apple : avec une main tenant un iPhone ou deux iPhones côte à côte. Il est aussi possible d'ajouter nos propres modèles de screenshots via une API de développement de plug-in.

Site Web : <http://fabian-kreiser.com/screentaker/> - **4,49€ (Disponible sur Mac AppStore)**

> Liya

Liya est un logiciel de parcours de base de données. Compatible avec trois systèmes de gestion de base de données : MySQL, PostgreSQL et SQLite3, il s'avère très utile lorsque l'on utilise CoreData et SQLite pour la gestion des données persistantes dans une application iOS/OSX.

Il propose une interface simple, permettant de voir les tables présentes, de les modifier et d'effectuer des requêtes SQL.

Site Web : <http://cutedgesystems.com> - **Gratuit (Disponible sur Mac AppStore)**

Vincent Saluzzo

Ingénieur informatique - Expert Technique iOS



Xcode : ma première application iOS 1^{re} partie

Développer une application pour iPhone vous fait rêver, mais vous ne vous êtes pas encore décidé à mettre la main à la pâte ? Cet article est fait pour vous !

Vous allez découvrir ici tous les aspects du développement d'une application iOS au cours d'une série d'articles. Nous avons choisi pour vous guider de créer une application présentant des données de films, récupérées depuis le site RottenTomatoes.

Dans ce premier article, vous allez apprendre à créer votre projet, commencer l'interface graphique et récupérer la liste des films actuellement au cinéma. À vos marques, prêts ? Codez !

Attention : code complet à télécharger sur www.programmez.com

CRÉATION DU PROJET SOUS XCODE

Afin de développer notre première application iPhone, nous avons besoin d'un ordinateur Mac et la dernière version de Xcode, l'environnement de développement d'Apple, qui permet la construction d'applications à la fois pour iOS et pour OS X.

Xcode - dont la dernière version stable, au moment de l'écriture de cet article est la 4.6.3 - est librement téléchargeable depuis le Mac App Store. Il est livré avec des outils de développement nécessaires à l'instrumentation de nos applications et iOS Simulator, un environnement qui permet l'exécution d'application iOS (soit iPhone ou iPad) sur notre ordinateur. Pour installer et tester nos applications maison sur notre iPhone ou iPad nous avons besoin d'un compte développeur qui - au moment de la rédaction de cet article - coûte 79 € par an et permet de connecter jusqu'à 100 périphériques de test. L'exécution d'une application sur l'iOS Simulator, cependant, ne nécessite pas de compte payant et fournit une expérience cohérente avec celle d'un vrai terminal avec peu ou pas de différence par rapport à un périphérique réel, surtout lorsque nous utilisons les API les plus courantes. Une fois que nous avons installé Xcode sur notre ordinateur, nous allons enfin pouvoir le lancer. Dans la fenêtre de dialogue de démarrage, nous serons en mesure de créer notre premier projet Xcode en sélectionnant «Create a new Xcode project» [Fig.1].

L'écran suivant va nous permettre de choisir un modèle de projet. Chaque modèle offre des classes de commodité et du code standard. Afin d'apprendre les éléments fondateurs d'une application iOS, dans ce tutoriel, nous allons opter pour le modèle «Empty application» qui ne fournit aucune classe autre que AppDelegate, le point d'entrée de notre application [Fig.2].

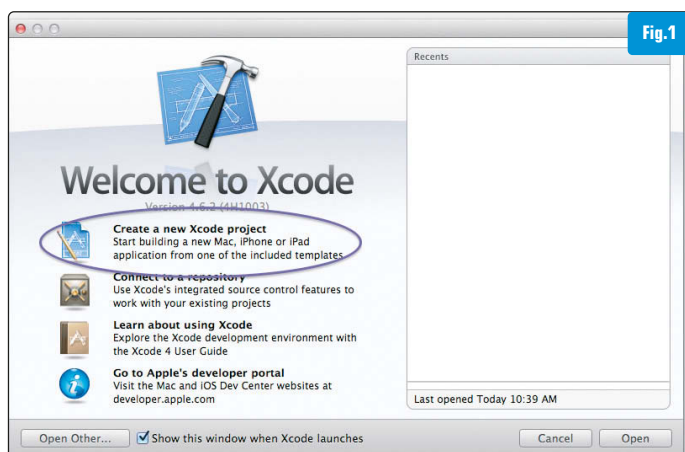


Fig.1

Après avoir sélectionné «Empty application», nous allons cliquer sur le bouton «Next», qui nous amènera à un écran où nous pourrions saisir les informations de base de notre application :

- «Product Name» : le nom de l'application,
- «Organization Name» : le nom de notre société,
- «Company Identifier» : une chaîne de caractères qui doit identifier nos applications et doit être unique pour chaque entreprise ou développeur qui soumet une application sur l'App Store,
- «Class Prefix» (optionnel) : les lettres qui précéderont les noms de toutes les classes que nous allons créer,
- «Devices» : les types de dispositifs que nous aimerions cibler et trois cases à cocher [Fig.3].

En ce qui concerne «Product Name» et «Organization Name» nous pouvons évidemment choisir n'importe quelle valeur. «Company Identifier» devra être une chaîne de caractères similaire à une url inversée, de type «fr.nom_societe». Dans la liste déroulante «Devices», nous pouvons choisir entre «iPhone», «iPad» ou «Universal». Ce dernier choix consiste en une application ciblant les deux types de terminaux. Nous n'avons pas besoin ici de sélectionner «Use Core Data», car Core Data est un gestionnaire de base de données qui permet

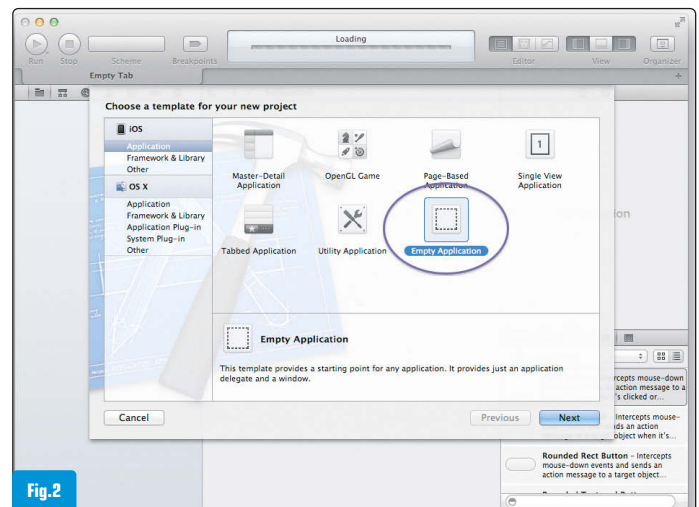


Fig.2

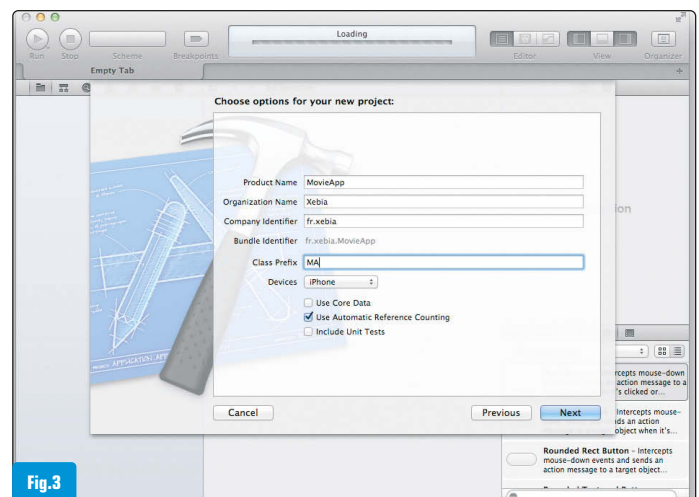


Fig.3

de stocker et d'interroger des valeurs au sein de notre application. Dans notre cas, les données se trouvent sur le serveur de RottenTomatoes, donc nous n'avons pas besoin de les garder dans l'application. Nous sélectionnons «Use Automatic Reference Counting», un mécanisme qui simplifie grandement la gestion de la mémoire dans notre code, et «Include Unit Tests». Lorsque nous aurons terminé avec les options, en cliquant sur «Next» nous pourrions choisir le dossier qui stockera nos fichiers de projets. C'est la dernière étape de la configuration du projet : nous pouvons maintenant commencer à développer notre Movie App. Finalement !

CRÉATION DU SPLASH SCREEN ET DE L'ICÔNE

> Le splash screen

Quand le système lance une application sur n'importe quel type d'appareil (iPhone, iPad, iPod), une image statique est temporairement affichée à l'écran. C'est l'application qui est lancée qui se doit d'embarquer cette image statique. Le but de cette image est de pouvoir afficher quelque chose à l'écran, immédiatement après que l'utilisateur ait décidé de lancer l'application. Cela permet aussi à l'application de disposer d'un petit temps afin qu'elle puisse s'initialiser et préparer les différentes vues et données à afficher. Une fois que l'application a terminé la phase d'initialisation, le système retire automatiquement le splash screen et affiche les vues de l'application.

Chaque application iOS doit fournir au moins une image pour le splash screen. L'application que nous souhaitons mettre en place est conçue pour l'iPhone et de ce fait, nous avons besoin de trois images pour le splash screen : Default.png, pour les iPhone non retina (par exemple, l'iPhone 3GS) et Default@2x.png pour les iPhone retina, hormis l'iPhone 5 (par exemple, l'iPhone 4S) et Default-568h@2x.png pour l'iPhone 5. Les images du splash screen doivent obligatoirement être des fichiers PNG.

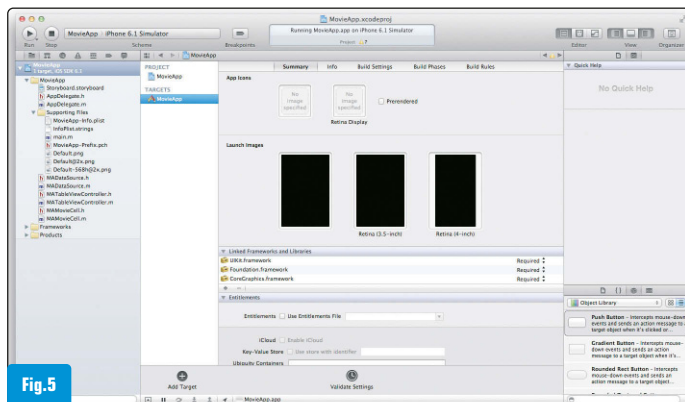


Fig.5

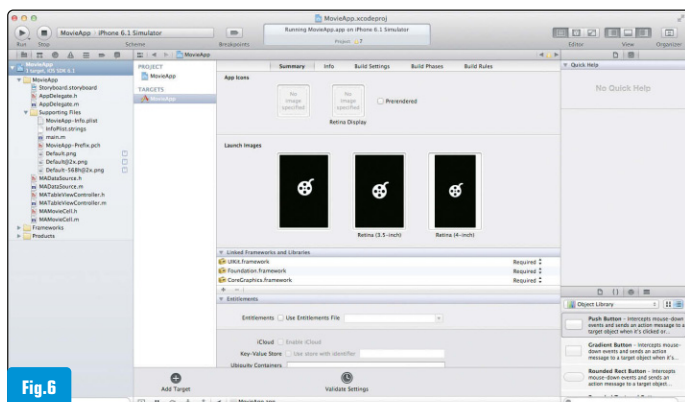


Fig.6

Voici les formats à respecter :

Default.png : 320x480px

Default@2x.png : 640x960px

Default-568h@2x.png : 640x1136px

Lors de la création de notre projet iOS, des images noires ont été générées par défaut et sont présentes dans l'arborescence du projet.

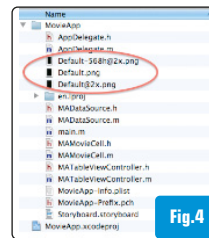


Fig.4

[Fig.4].

Afin de personnaliser notre application, il suffit de créer trois images respectant les dimensions listées ci-dessus et de remplacer les images générées. Une fois cette opération faite, nous pouvons vérifier la bonne prise en compte de ces nouvelles images par Xcode en allant dans les paramètres de notre application :

Avant la configuration des images splash screen [Fig.5]

Après la configuration des images splash screen [Fig.6]

> L'icône

Chaque application iOS a besoin d'une icône qui sera affichée sur l'écran d'accueil de l'appareil mais aussi sur l'App Store. En réalité, une application peut spécifier une multitude d'icônes qui seront utilisées dans différentes situations. Par exemple, une application peut fournir une petite icône qui sera affichée et utilisée dans les résultats de spotlight et elle peut fournir une icône beaucoup plus grosse et en haute résolution qui sera utilisée pour les appareils supportant l'affichage retina. Pour une application exclusivement pour l'iPhone, il faut embarquer à minima deux formats d'icônes :

- Icon.png : 57x57px
- Icon@2x.png : 114x114px (pour les écrans retina)

Afin d'indiquer à Xcode les icônes que l'on souhaite utiliser pour notre application, il suffit de se rendre dans l'écran des paramètres de notre application (voir la capture d'écran précédente) et de faire du glisser/déposer au niveau de la zone «App Icons» : l'image Icon.png doit être déposée dans le carré de gauche, et l'image Icon@2x.png, correspondant à l'icône pour les écrans retina, doit aller dans le carré de droite : [Fig.7].

En effectuant cette opération, Xcode copie automatiquement les images dans le répertoire du projet et les ajoute dans son arborescence. À cette étape, lorsque vous lancez l'application iPhone (sur simulateur par exemple), vous verrez le splash screen au démarrage ainsi qu'une jolie icône sur l'écran d'accueil de l'iPhone.

MISE EN PLACE DU STORYBOARD

Les storyboards ont été présentés avec l'arrivée d'iOS 5 et permettent d'économiser beaucoup de temps lors de la conception d'appli-

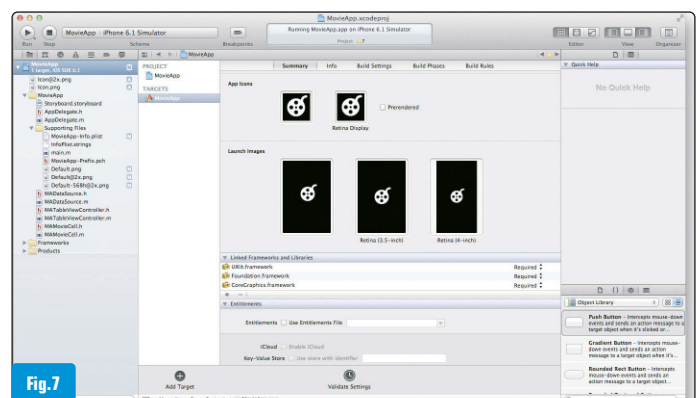


Fig.7

cations. Un storyboard représente les écrans de l'application ainsi que les transitions entre ces derniers. Notre application ne contient que quelques écrans mais une application bien plus complexe peut très bien avoir plusieurs storyboards. Voici un exemple de storyboard : [Fig.8].

Sans même connaître le périmètre fonctionnel de cette application, on peut d'ores et déjà savoir que l'application dispose de deux écrans et qu'il est possible de naviguer entre ces deux écrans.

Afin de créer le storyboard de notre application, rendez-vous dans Xcode puis dans le menu File -> New -> File, sélectionnez «User Interface» dans la colonne de gauche puis «Storyboard». Appuyez sur «Next» et laissez les paramètres par défaut [Fig.9].

Une fois cette étape effectuée, le nouveau storyboard apparaît dans l'arborescence du projet. Après l'avoir sélectionné, nous pouvons voir qu'il est pour l'instant totalement vide, ce qui est normal. L'objectif ici est de le remplir pour indiquer à l'application quel écran afficher au démarrage.

L'éditeur de storyboard est très pratique : il est possible d'y glisser/déposer de nouveaux éléments à partir d'une bibliothèque d'objets (qui se trouve en bas de la colonne «Utilities» à droite). Mettons en œuvre cette opération en faisant glisser au milieu du storyboard un View Controller depuis la bibliothèque d'éléments : [Fig.10].

Vous remarquerez qu'une flèche pointe vers ce viewController. Cela indique que cet élément est le viewController initial, c'est-à-dire que c'est le viewController à afficher au démarrage de l'application. Afin de manipuler un peu plus le storyboard, je vous invite à glisser dans le viewController, fraîchement créé, un label (UILabel) et de changer son texte, écrivons par exemple «Ecran d'accueil» : [Fig.11].

Une application, pour être affichée à l'écran, dispose d'une window. Cette window se doit d'avoir un viewController par défaut à afficher au démarrage. Si l'on se rend dans le .m de l'AppDelegate de notre

application, on y verra la méthode - (BOOL)application:(UIApplication *) application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions. Cette méthode est automatiquement appelée par le système lors du démarrage de l'application. C'est ici que la window de l'application est instanciée et que l'on doit y associer le viewController par défaut à afficher.

Pour cela, il suffit de créer une référence à notre storyboard, et grâce à cette référence il sera ensuite possible de demander au storyboard une référence sur le viewController initial. Une fois celui-ci obtenu, il ne reste plus qu'à l'associer à la window. Voici à quoi ressemble la méthode avec ces informations :

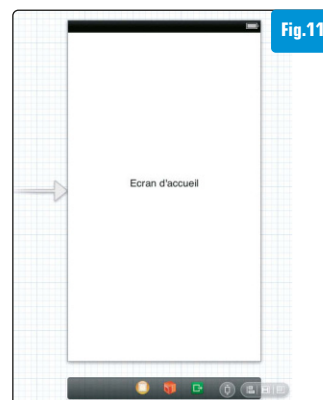
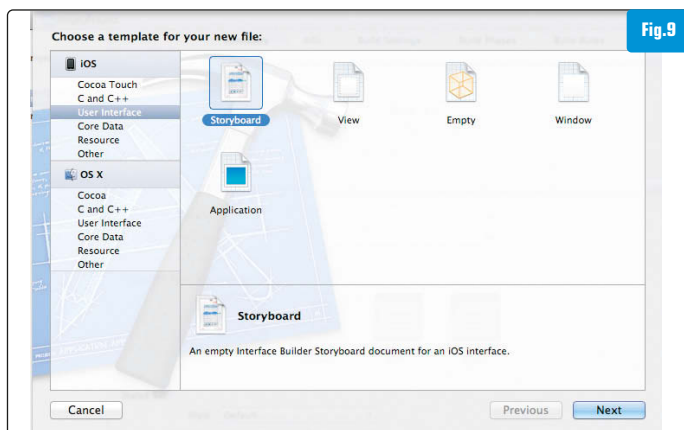
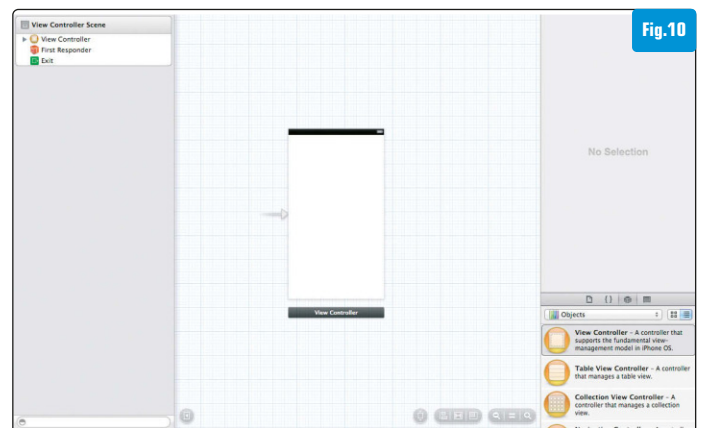
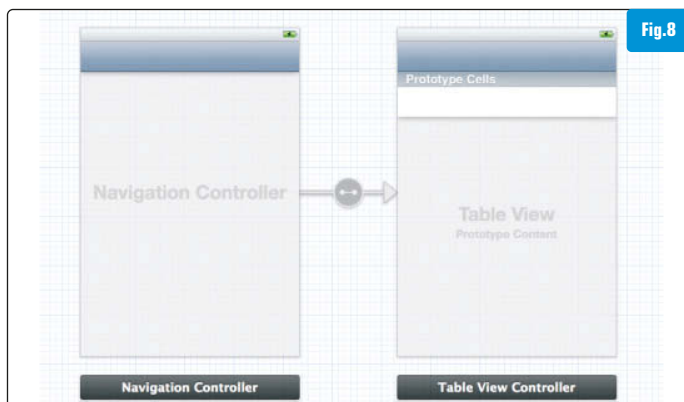
AppDelegate.m

Voir code complet sur www.programmez.com [Code 1]

A cette étape, si vous compilez l'application et la lancez, vous verrez l'écran d'accueil, tel qu'il est spécifié sur le storyboard : [Fig.12].

MISE EN PLACE D'UN UITABLEVIEWCONTROLLER

Dans le cadre de cet article, nous souhaitons mettre en place une application qui affiche une liste de films. Et qui dit liste sous iOS dit UITableView. Une UITableView est un composant qui est en définitive un «tableau» présentant une série de données sous la forme d'une liste. Cette liste peut être personnalisée à souhait. Ici, nous allons nous contenter de personnaliser la UITableView (la liste) avec des cellules (des lignes) qui contiennent la photo, le titre et l'année des films. Afin de mettre en place notre UITableView, nous allons utiliser un UITableViewController. Cet objet est un controller très pratique car il encapsule directement une UITableView. Commençons par créer une classe qui hérite de UITableViewController. Pour cela, allez dans le menu File ? New ? File puis choisissez Cocoa Touch dans la liste de gauche et enfin Objective-C class. Dans l'étape suivante ren-



seigneur 'UITableViewController' comme sous-classe et 'MUITableView-Controller' comme nom de classe puis validez.

A cette étape, notre classe héritant de UITableViewController est créée mais il reste encore à l'instancier au sein de l'application. Pour cela, nous allons remplacer le UIViewController qui contient le label «Ecran d'accueil» par notre «MUITableViewController». Dans le storyboard de l'application, commencez par supprimer le viewController existant. Maintenant, faites un glisser/déposer de l'objet «Table View Controller» depuis la collection d'objets vers le storyboard. Vous noterez que le précédent UIViewController a une apparence différente de notre UITableViewController. En effet, ce dernier dispose d'une tableView qui affiche les cellules. Maintenant, il nous faut associer ce nouveau UITableViewController à notre classe MUITableView-Controller précédemment créée. Pour cela, sélectionnez le Table View Controller dans la scène du storyboard : [Fig.13].



Fig.13

Une fois cela effectué, allez dans l'inspecteur de l'élément et changez la classe du Table View Controller de UITableViewController (classe par défaut) à MUITableView-Controller. Vous aurez aussi

noté la flèche qui pointe vers notre MUITableViewController. Cela indique que c'est notre viewController initial et donc qu'il sera affiché au lancement de l'application : [Fig.14].

A ce stade, l'application affiche une liste vide. Voyons maintenant comment la remplir en récupérant des données !

PRÉSENTATION DE L'API ROTTENTOMATOES

Nous avons choisi de récupérer les données cinématographiques depuis le site RottenTomatoes (<http://www.rottentomatoes.com>) qui possède une API publique. Le principe est simple : à l'aide de requêtes spécifiques, nous allons interroger le site et celui-ci va nous renvoyer les informations demandées au format JSON, c'est-à-dire sous forme d'un ensemble de clés et de valeurs qui représentent un objet ou une liste d'objets.

C'est un service web RESTful destiné à faciliter l'accès aux données du site. Tous les appels se feront depuis l'URI de base <http://api.rottentomatoes.com/api/public/v1.0>, en s'authentifiant grâce à une API_KEY. Le schéma d'appel est donc très simple : **Base_URI / Path_To_Resource ? apikey=YOUR_API_KEY & key1=value1 & key2=value...** avec key1 et key2 des paramètres et value1 et value2 leurs valeurs respectives.

Voici par exemple le lien pour récupérer la deuxième page de la liste des films qui vont bientôt sortir, chaque page ayant 20 films :

http://api.rottentomatoes.com/api/public/v1.0/lists/movies/upcoming.json?apikey={your_api_key}&page_limit=20&page=2

Pour plus d'informations sur l'utilisation de cette API, nous vous invitons à lire la documentation complète qui est disponible à cette adresse : <http://developer.rottentomatoes.com/docs>

Afin de pouvoir réaliser notre projet, il vous faut tout d'abord obtenir votre API_KEY, en créant un compte sur le site à l'adresse <http://developer.rottentomatoes.com/member/register>. Remplissez le formulaire et dans la section « Register Your New Application », donnez un nom à votre projet, une url et le type « Mobile Application ».

Formulaire de création de compte sur RottenTomatoes [Fig.15].

Section «Register Your Application» du formulaire, qui permet d'obtenir la clé d'API [Fig.16].

Vous recevrez alors un email de vérification pour finaliser la création de votre compte, qui contient un lien vers une page où vous pourrez trouver votre clé.

RÉCUPÉRATION DES DONNÉES

Maintenant que vous avez votre clé d'API, nous allons pouvoir commencer à connecter l'application. Afin de bien séparer les fonctionnalités, nous allons créer un nouvel objet héritant de la classe NSObject qui gèrera tous les appels à l'API RottenTomatoes. Dans XCode, faites Fichier -> Nouveau Fichier, sélectionnez «Cocoa Touch» dans la liste à gauche puis «Objective-C class». Vérifiez que la classe hérite bien de NSObject et nommez-la MADataSource.

Fig.15

Fig.16

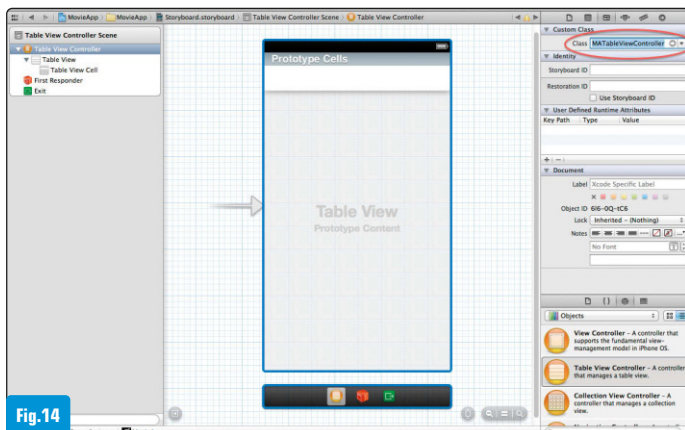


Fig.14

Dans un premier temps, nous allons récupérer la liste des films du box office. Dans le fichier `MADDataSource.h`, nous allons rajouter une **property** `dataArray` qui contiendra la liste des résultats et une méthode `loadBoxOfficeData`. Sous iOS, une **property** est définie par la syntaxe suivante :

Voir code complet sur www.programmez.com (Code 2)

Nous ne nous attarderons pas sur les attributs des **property**, car cela pourrait être l'objet d'une exploration plus approfondie mais nous dirons simplement que, pour chaque **property** qui représente un élément visuel géré par le Storyboard, les attributs corrects doivent être «nonatomic, weak», pour les autres **properties** les attributs seront (dans la plupart des cas) «nonatomic, strong».

MADDataSource.h

Voir code complet sur www.programmez.com (Code 3)

Dans le fichier `MADDataSource.m`, nous allons ajouter deux `NSString`s statiques pour définir notre clé d'API et l'URI de base

MADDataSource.m

Voir code complet sur www.programmez.com (Code 4)

L'avantage de cette solution est de rassembler les différentes variables de configuration au même endroit et de pouvoir éviter les fautes de frappe lorsque nous allons les utiliser.

Passons maintenant aux choses sérieuses : la récupération des films du box office. D'après la documentation de l'API, la liste peut être obtenue à l'adresse `lists/movies/box_office.json` et deux paramètres peuvent être ajoutés : «limit», qui définit le nombre de films renvoyés et «country» qui permet de définir le pays qui nous intéresse. Le principe que nous allons utiliser est simple : récupérer les données grâce à la fonction `+[id]dataWithContentsOfURL:(NSURL*)aURL` de la classe `NSData` puis les transformer en objets du kit de développement (`NSArray` ou `NSDictionary`) pour pouvoir les manipuler facilement grâce à la classe `NSJSONSerialization` prévue à cet effet. La première méthode que nous allons créer sera notre constructeur d'URL. Elle prend en paramètre le chemin relatif pour la fonction à appeler ainsi qu'un dictionnaire de paramètres. Cela va nous permettre de créer les URL pour nos différents appels en y ajoutant des paramètres. Elle pourra donc être réutilisée pour n'importe lequel de nos appels.

MADDataSource.m

Voir code complet sur www.programmez.com (Code 5)

Maintenant que nous pouvons construire l'URL où sont les données qui nous intéressent, il faut les récupérer.

Ajoutez une nouvelle `NSString` constante au début du fichier `MADDataSource.m` pour définir le chemin relatif de la requête «Box Office»

MADDataSource.m

Voir code complet sur www.programmez.com (Code 6)

Nous allons maintenant charger les données depuis le site et enregistrer la réponse dans notre liste `dataArray`. Nous allons récupérer le code pays de l'utilisateur puis former l'URL grâce à notre méthode. Notre URL sera donc : `http://api.rottentomatoes.com/api/public/v1.0/lists/movies/box_office.json?apikey={your_api_key}&country={user_country_code}&limit=20`

MADDataSource.m

Voir code complet sur www.programmez.com (Code 7)

Regardons un peu le contenu de la réponse du serveur :

Réponse du serveur : **Voir code complet sur** www.programmez.com (Code 8)

La liste des films est enregistrée dans la variable `movies`. C'est donc cette valeur qui nous intéresse.

Il ne nous reste plus qu'à préparer l'affichage des données. En effet, chaque film possède plusieurs propriétés, identifiées par leur clé, qui contiennent les infos. Afin une fois de plus de bien séparer les fonctionnalités de l'application, le `viewController` n'est pas censé connaître ces clés. Nous allons donc créer une méthode qui va renvoyer un dictionnaire avec 3 clés : `infoTitle`, `infoSubtitle`, `infoImageURL` dont les valeurs seront celles d'un des éléments de notre liste.

MADDataSource.m

Voir code complet sur www.programmez.com (Code 9)

Afin de pouvoir utiliser cette méthode dans le `viewController`, il ne nous reste plus qu'à la déclarer dans `MADDataSource.h`, sans oublier d'ajouter nos clés !

MADDataSource.h

Voir code complet sur www.programmez.com (Code 10)

Maintenant que notre `dataSource` est créé et fonctionnel, rajoutons-le au `viewController` afin que ce dernier puisse accéder aux données des films :

MATableViewController.h

Voir code complet sur www.programmez.com (Code 11)

Il ne nous reste plus qu'à récupérer les données une fois que le `viewController` se charge. Pour cela, il suffit d'allouer la `dataSource` et de lancer le téléchargement des données dans le `viewDidLoad` du `viewController` :

MATableViewController.m

Voir code complet sur www.programmez.com (Code 12)

AFFICHAGE DES DONNÉES DANS LE TABLEAU

Une fois toutes les données téléchargées, nous sommes maintenant en mesure de les présenter dans notre `UITableViewController`. Le `tableView` contiendra une liste de films où chaque item présentera

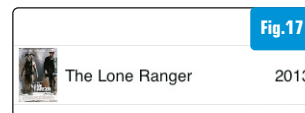


Fig.17

l'affiche du film, le titre et l'année d'émission, comme le montre l'image [Fig.17].

Sous iOS, chaque élément d'une `tableView` s'appelle une cellule et doit être de type `UITableViewCell`. Grâce au storyboard précédemment créé, nous pouvons configurer l'apparence de la cellule directement dans l'éditeur d'interface sans que notre code ait à gérer le positionnement des éléments.

Tout d'abord, nous devons créer la classe de notre cellule. Dans Xcode, Cliquons sur **File -> New -> File -> Objective-C class** et remplissons les valeurs suivantes :

• **Class:** `MAMovieCell`

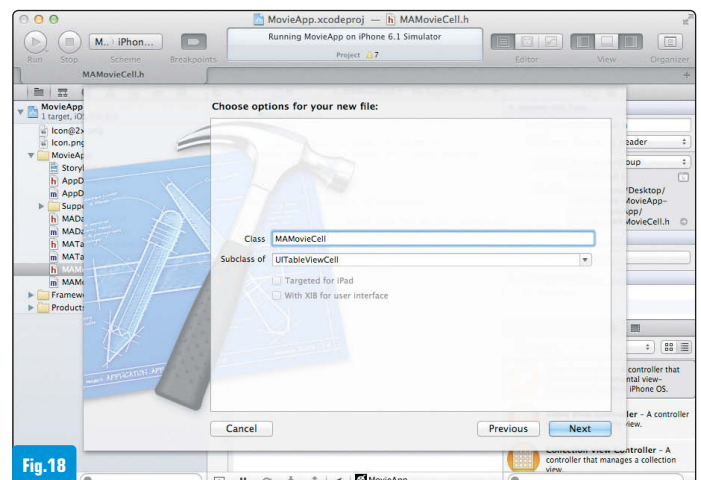


Fig.18

• Subclass of: UITableViewCell [Fig.18].

Nous pouvons maintenant cliquer sur **Next** et enregistrer. Xcode a créé deux nouveaux fichiers : **MAMovieCell.h** et **MAMovieCell.m** qui contiennent respectivement la déclaration et les implémentations de notre objet cellule.

Dans le fichier **MAMovieCell.h**, ajoutez trois éléments : deux labels (titre et année) et une imageView (pour l'affiche du film).

MAMovieCell.h

Voir code complet sur www.programmez.com (Code 13)

L'attribut **IBOutlet** indique que la propriété sera gérée par l'éditeur d'interface (Interface Builder, d'où le préfixe «IB»).

Maintenant que nous avons créé les propriétés de l'objet, nous pouvons passer à la définition de leur aspect et de positionnement. Ouvrons le Storyboard une fois de plus. À l'intérieur du tableViewController, nous allons cliquer sur la région vide juste en dessous de «**Prototype cells**» : cette région représente notre cellule, mais elle est encore vide.

Pour la configurer, changeons la classe de la cellule en saisissant **MAMovieCell**, comme montré dans l'image [Fig.19].

Maintenant il faudra cliquer sur le quatrième onglet de la gauche et mettre la valeur «**Identifiant**» à «**MovieCell**». Cette information nous sera très utile un peu plus loin [Fig.20].

Nous pouvons maintenant ajouter les éléments visuels de notre cellule : dans la bibliothèque d'objets, recherchez imageView et faites-la glisser sur la cellule vide. Une fois l'imageView ajoutée à la cellule, nous pouvons la redimensionner et la déplacer comme souhaité.

Une autre étape fondamentale est de dire à Xcode que l'imageView

que nous avons ajoutée est celle définie à travers la property dans notre code.

Pour ce faire, il faut cliquer sur «**Movie Cell**» dans la liste à gauche de Xcode et sélectionner le dernier onglet dans le menu de droite.

Xcode va maintenant montrer les propriétés de sortie de notre cellule (vous vous souvenez de l'attribut IBOutlet que nous avons ajouté ci-dessus ?) et vous devriez être en mesure de voir «thumbnail», «subtitleLabel» et «titleLabel».

Nous pouvons maintenant connecter la property «thumbnail» à l'imageView : cliquez sur le petit cercle à droite de la propriété «thumbnail» et, avec le curseur, tracez une ligne depuis ce cercle vers l'imageView.

Un IBOutlet connecté aura un rectangle arrondi autour du nom de la propriété. Faisons de même pour les deux libellés, en ajoutant deux UILabels sur la cellule et reliant leur IBOutlet à l'élément respectif.

L'image montre à quoi le storyboard devrait ressembler.

Notre travail sur le storyboard est maintenant terminé. Revenons donc au «vrai» codage ! Ouvrons le MATableViewController et importons le .h de la MAMovieCell :

MATableViewController.m

Voir code complet sur www.programmez.com (Code 14)

Maintenant, ajoutons (ou remplaçons) les trois méthodes qui gèrent ensemble dataSource, tableViewController et cellules.

Voir code complet sur www.programmez.com (Code 15)

Cette méthode informe la tableView du nombre de sections qui doivent être affichées. Puisque nos éléments seront tous affichés sur une seule section, nous renvoyons «1» simplement.

Voir code complet sur www.programmez.com (Code 16)

La méthode ci-dessus communiquera à la tableView le nombre de cellules qui doivent être affichées. Les informations sont extraites à partir de notre source de données : on renvoie ainsi le nombre d'éléments contenus dans notre «dataArray».

Voir code complet sur www.programmez.com (Code 17)

Enfin, le code ci-dessus permet de configurer la cellule. Voici quelques détails supplémentaires à ce sujet :

Voir code complet sur www.programmez.com (Code 18)

Cette ligne instancie une MAMovieCell en utilisant un «Identifiant» spécifié qui, dans notre cas est «MovieCell». Souvenez-vous, dans le Storyboard, nous avons mis cette valeur juste après avoir saisi le nom de la classe de la cellule et c'est bien grâce à ce paramètre que le système sera en mesure d'instancier la classe pour nous.

Voir code complet sur www.programmez.com (Code 19)

Ici, nous avons stocké dans la variable infoDict l'information contenue dans notre dataSource.

Voir code complet sur www.programmez.com (Code 20)

Et, enfin, ici nous avons attribué les propriétés «texte» de nos libellés ainsi que l'«image» de l'imageView. Les valeurs sont de nouveau extraites du dataSource et, plus précisément, de la variable infoDict que nous venons de créer.

CONCLUSION

Dans le prochain article, nous verrons comment afficher les détails d'un film et réagir à différentes interactions de l'utilisateur. D'ici là, bon code et à la prochaine fois !

Simone Civetta, Nicolas Thenoz, Martin Moizard, consultants chez Xebia

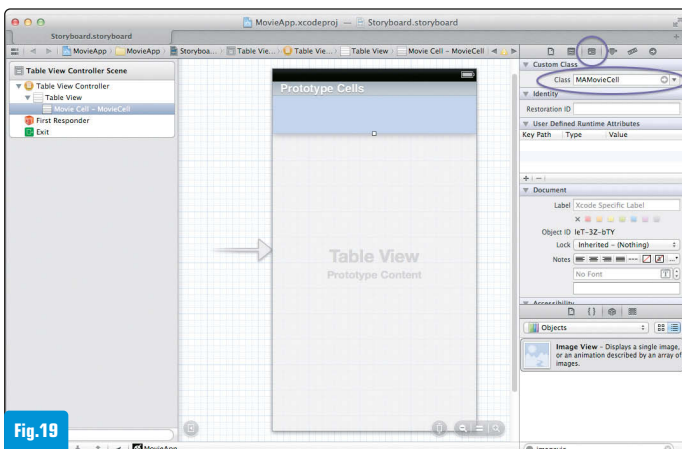


Fig.19

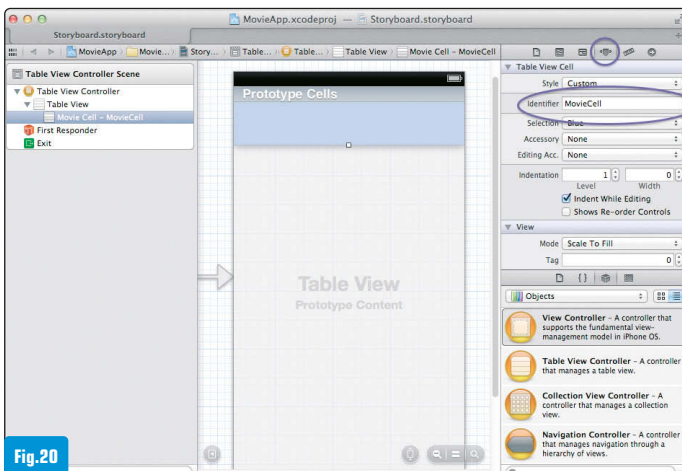


Fig.20

Les expressions régulières au service de l'administrateur Linux

Les expressions régulières font souvent figure d'épouvantail. Pourtant, utilisées avec les outils du Shell, elles peuvent vous rendre d'immenses services, et même parfois vous sauver la mise.

La philosophie d'UNIX, donc de Linux, est de mettre à disposition de l'administrateur système de très nombreux petits outils très spécialisés, mais très efficaces dans leurs domaines respectifs d'application. De nos jours, les administrateurs systèmes tendent à délaisser ces outils traditionnels, aux abords certes peu séduisants, au profit d'interfaces d'administration graphique, en général dans un navigateur. Plesk, DirectAdmin, Webmin sont des exemples d'outils d'administration bien connus. Ils ont indéniablement leur valeur, dans beaucoup de situations, et surtout quand tout va bien. Mais finalement, rien ne remplace les bons vieux outils dans les situations courantes et plus encore dans les situations difficiles. Et lorsque tout va mal et que la situation est critique, il n'y a finalement plus qu'eux pour vous tirer d'embarras. Nous allons voir ensemble quelques exemples pour illustrer cela, en nous appuyant en outre sur les expressions régulières, de si mauvaise réputation, mais qui sont leurs compléments indispensables.

1 LES EXPRESSIONS RÉGULIÈRES

Le terme expression régulière est la transcription de l'anglais *regular expression* qui se traduit en fait par expression rationnelle. Mais en pratique, l'immense majorité des administrateurs systèmes utilisent le terme expression régulière, et je ferai de même dans cet article, si vous le voulez bien. Les expressions régulières sont issues des théories mathématiques des langages formels des années 40. Leur principe est qu'une chaîne de caractères, dite motif, et formée de signes cabalistiques parfaitement illisibles :) permet de décrire des ensembles de chaînes de caractères qui correspondent aux critères du motif. Leur application toute trouvée en informatique est la recherche de chaînes de caractères dans un fichier ou dans un nom de fichier.

```
fred@mars: ~/Developpement
ubercart/uc_store/uc_store.admin.inc: $result = db_query("SELECT * FROM {uc_cou
ubercart/uc_store/uc_store.admin.inc: $result = db_query("SELECT * FROM {uc_cou
ubercart/uc_store/uc_store.admin.inc: db_query("UPDATE {uc_countries} SET v
ubercart/uc_store/uc_store.admin.inc: $result = db_query("SELECT * FROM {uc_cou
ubercart/uc_store/uc_store.admin.inc: db_query("UPDATE {uc_countries} SET v
ubercart/uc_store/uc_store.admin.inc: $country = t(db_result(db_query("SELECT c
ubercart/uc_store/uc_store.admin.inc: $result = db_query("SELECT * FROM {uc_cou
ubercart/uc_store/uc_store.admin.inc: db_query("DELETE FROM {uc_countries} WHER
ubercart/uc_store/uc_store.admin.inc: $result = db_query("SELECT * FROM {uc_cou
ubercart/uc_store/uc_store.admin.inc: db_query("UPDATE {uc_countries} SET ver
ubercart/uc_store/uc_store.admin.inc: $result = db_query("SELECT * FROM {uc_cou
ubercart/uc_multiprice/uc_multiprice.module: // load uc_countries
ubercart/uc_multiprice/uc_multiprice.module: $result = db_query("SELECT c
ubercart/uc_multiprice/uc_multiprice.module: $result = db_query("SELECT country
ubercart/uc_multiprice/uc_multiprice.module: $country = db_fetch_object(db_qu
ubercart/uc_multiprice/uc_multiprice.module: $result = db_query("SELECT * FROM
ubercart/uc_views/views/uc_views.views.inc: 'base' => 'uc_countries',
ubercart/uc_views/views/uc_views.views.inc: 'table' => 'uc_countries',
ubercart/uc_views/views/uc_views_handler_filter_country.inc: $result = db_q
R BY country_name");
fred@mars: ~/Developpement$
```

Bien pour un début, mais à affiner :-)

2 OPÉRATEURS ET QUANTIFICATEURS

Il existe de nombreuses familles d'expressions régulières, chacune avec leurs spécificités. Les simples, les étendues, celles qui sont particulières à un langage (Python SQL, Perl, TCL, etc.) Mais au final, elles ont toutes un tronc commun qui se résume à quelques opérateurs et quantificateurs, dont le maniement, à la base rébarbatif, est finalement plutôt simple.

Voici d'abord 3 opérateurs :

Opérateur	Signification
^	Le motif qui suit cet opérateur (ou prédicat) doit se situer en début de ligne.
\$	Le motif qui précède cet opérateur doit se trouver en fin de ligne
	Cet opérateur permet de proposer un choix de motifs

Et maintenant trois quantificateurs :

Quantificateur	Signification
?	Le caractère ou groupe de caractères qui le précède doit être présent zéro ou une fois
+	Le caractère ou groupe de caractères qui le précède doit être présent une ou plusieurs fois
*	Le caractère ou groupe de caractères qui le précède doit être présent zéro, une ou plusieurs fois

Et c'est tout ! Le plus important avec les expressions régulières étant de les pratiquer pour s'y habituer et les maîtriser.

3 POUR SE FAIRE LA MAIN

Très souvent, le développeur a besoin de retrouver une ligne particulière dans une montagne de code qu'il a écrit, ou, pire, dans une montagne de code écrit par quelqu'un d'autre. Ainsi l'autre jour, j'avais besoin de trouver dans quel fichier résident des lignes de code dans Ubercart (un module de e-commerce pour Drupal), travaillant avec la table uc_countries en base de données. Pour cela, l'utilitaire shell grep est imbattable :

```
cd le_repertoire_sous_ubercart
grep -R uc_countries ubercart
```

Le commutateur -R demande une recherche récursive dans les dossiers et sous-dossiers. uc_countries est le motif à chercher. L'expression régulière est tellement simple que cela en est frustrant :) En une fraction de seconde, les résultats de la recherche apparaissent, mais ils sont pléthoriques et il faut affiner la recherche. En fait je me souviens que la ligne de code PHP que je recherche commence par \$country =. J'affine donc ma recherche avec, cette fois, une "vraie" expression régulière. Je me souviens en outre qu'il y a

db_result quelque part dans cette ligne de code. Je demande donc tel quel à grep :

```
grep -R '^ *\$country =.*db_result.*uc_countries.*\$' ubercart
```

qui en une fraction de seconde me dit dans quel fichier se situe la ligne que je cherche :

```
ubercart/uc_store/uc_store.admin.inc: $country = t(db_result(db_query("SELECT country_name FROM {uc_countries} WHERE country_id = %d", $country_id)));
```

Notre premier exemple d'expressions régulières, quoique très simple nous en apprend toutefois beaucoup. D'abord nous remarquons que l'expression régulière est donnée entre quote et non entre guillemets. Ceci est important pour éviter le traquenard classique qui est que le shell interprète \$ comme indiquant un nom de variable dans une expression entre guillemets. Lorsqu'une expression est donnée entre quote, le shell ne s'y intéresse pas. On dit qu'elle est protégée. Voici l'expression régulière proprement dite maintenant :

```
^ *\$country =.*db_result.*uc_countries.*\$
```

Nous cherchons dans du code PHP Drupal, donc très probablement il est indenté selon les conventions de codage Drupal et donc il commence par un ou plusieurs espaces. Ou s'il est mal écrit, il n'est pas indenté et il n'y a pas d'espaces :) Ceci s'exprime tout simplement par ^ *. Avec ^ pour début, l'espace pour espace et * pour zéro ou plusieurs espaces en début de chaîne. Puis vient \\$country que l'on cherche, avec un \$ échappé puisque \$ est un opérateur. Puis vient un certain nombre de caractères. Le point est le caractère générique dans les expressions régulières. Donc nous mettons .* pour exprimer 'un certain nombre de caractères'. Puis db_result, puis quelque chose encore avec .*, puis uc_countries, puis quelque chose enfin pour terminer .*\$. Finalement c'est tout simple :-)

4 VÉRIFIER L'ÉTAT DES DISQUES

Une des tâches de l'administrateur système est de s'assurer constamment du bon fonctionnement des disques des serveurs dont il a la charge, pour prévenir la perte de données. Par exemple il peut demander à ses serveurs l'envoi régulier, dans des mails, des résultats de la commande :

```
cat /proc/mdstat
```

exécutée dans un script lancé par cron, pour connaître l'état de disques montés en RAID 1. (Pour plus d'informations sur ce sujet, lire Administrer le RAID d'un serveur Linux dans Programmez! 155). La commande va renvoyer quelque chose comme ceci :

```
md1 : active raid1 sda1[0] sdb1[1]
10238912 blocks [2/2] [UU]

md2 : active raid1 sda2[0] sdb2[1]
721806272 blocks [2/2] [UU]
```

Si un disque est cassé vous verrez, au lieu de [UU], [_U], ou [U_], ou même [_]. Sachant cela, il peut être intéressant, si vous administrez beaucoup de serveurs, de ne recevoir un mail qu'en cas de pro-

blème avéré. Nous sommes précisément dans un cas où l'opérateur l fait merveille. Ecrivons un petit script :

```
#!/bin/bash

check=$(cat /proc/mdstat | grep "_U|U_|__")
#ou
#check=$(cat /proc/mdstat | egrep "_U|U_|__")

if [ -n "$check" ]
then
    echo $check | mail -s "[ALERTE] RAID $(hostname) endommagé" mon.adresse@mon-domaine.com
fi
```

Dans cet exemple, l'expression régulière passée à grep n'est qu'un choix de trois motifs. On remarque qu'avec grep, l'opérateur l doit être échappé. On peut s'en dispenser avec egrep en remplacement de grep, comme montré dans la ligne en commentaire dans le script. egrep apporte la facilité de travailler avec les expressions régulières étendues, ce qui allège souvent l'écriture des expressions régulières. En revanche, egrep a le défaut de ne pas être parfaitement portable dans le monde d'UNIX.

Toujours pour s'assurer du bon fonctionnement de ses serveurs, un administrateur voudra vérifier, outre l'état du RAID, l'état des disques eux-mêmes avec les outils de monitoring S.M.A.R.T. Ainsi il lancera à intervalles réguliers, la commande (par exemple) :

```
smartctl -d ata -a /dev/sda
```

Le problème étant que la sortie de cette commande est fort volumineuse, et l'éplucher pour chacun des disques de chacun de serveurs est non seulement trop fastidieux, mais encore source d'erreur due à l'inattention engendrée par la masse d'information. Une première possibilité pour alléger le travail est de filtrer les paramètres dont l'évolution est importante dans le sens où elle peut annoncer une panne. Par exemple un script contenant ceci :

```
ERRORS="Raw_Read_Error_Rate\|Reallocated_Sector_Ct\|Reallocated_Event_Count\|Current_Pending_Sector\|Offline_Uncorrectable\|Spin_Retry_Count"
```

```
/usr/sbin/smartctl -a -d ata /dev/sda | grep -i $ERRORS
```

Produira une sortie beaucoup plus concise et lisible :

1	Raw_Read_Error_Rate	0x000b	100	100	016	Pre-fail	Always	-	0
5	Reallocated_Sector_Ct	0x0033	100	100	005	Pre-fail	Always	-	0
10	Spin_Retry_Count	0x0013	100	100	060	Pre-fail	Always	-	0
196	Reallocated_Event_Count	0x0032	100	100	000	Old_age	Always	-	0
197	Current_Pending_Sector	0x0022	100	100	000	Old_age	Always	-	0
198	Offline_Uncorrectable	0x0008	100	100	000	Old_age	Offline	-	0

Mais là encore, si les expressions régulières nous ont déjà bien aidés, on peut encore faire mieux, grâce à elles. En effet dans la sortie ci-dessus, le plus important est la dernière colonne. Si une valeur non nulle apparaît, on peut commencer, sinon à s'inquiéter, au moins à se dire qu'il faut garder un oeil sur le disque. Testons donc cette valeur et demandons au système de nous envoyer un mail dans ce cas. Voici un exemple de script qui surveillerait ainsi l'état des deux disques du RAID d'un serveur.


```
#!/bin/bash

ERRORS="Raw_Read_Error_Rate\|Reallocated_Sector_Ct\|Reallocated_Event_Count\|Current_Pending_Sector\|Offline_Uncorrectable\|Spin_Retry_Count"

disques="/dev/sda /dev/sdb"

function alerte
{
    if [ $# -eq 2 ]
    then
        echo "alerte mail le disque $2 se degrade"
    fi
}

for disque in $disques
do
    check=$(/usr/sbin/smartctl -a -d ata $disque | grep -i $ERRORS | rev | cut -d" " -f1 | grep [1-9])
    alerte $check $disque
done
```

Outre les expressions régulières, ce script montre que les commandes chaînées dans un tube sont un des secrets de l'administration système Linux :) Dans la boucle qui teste chacun des disques, la sortie de la commande *smartctl* est d'abord filtrée, comme indiqué précédemment, au moyen d'une première expression régulière. Il se trouve que la valeur qui nous intéresse dans chacune des lignes filtrées se situe en fin de ligne. Il est fort pratique de la passer en début de ligne en retournant chacune des chaînes grâce à la commande *rev*. Ensuite, nous isolons ce premier élément grâce à la commande *cut*, commande très souvent méconnue, mais si pratique. Enfin une deuxième expression régulière teste si la valeur ainsi isolée contient un chiffre autre que 0, c'est-à-dire un nombre dans la plage de 1 à 9, ce qu'exprime `[1-9]`. Si c'est le cas, *check* contient une valeur non nulle. Dans ce cas, notre fonction *alerte* recevra deux arguments et elle saura qu'elle doit poster un mail à l'administrateur.

5 SED

Sed est un outil qui est de plus en plus méconnu, quand il n'est pas considéré comme un archaïsme. En réalité, il est un outil irremplaçable. Sed signifie Stream editor.

Ce qui pour faire simple, signifie qu'il peut travailler sur une ligne à la fois dans un fichier, comme au bon vieux temps où les éditeurs de texte ne pouvaient éditer qu'une seule ligne à la fois, c'est-à-dire dans les années 70 :) Sed s'utilise ainsi :

```
sed -e 'instructions' < fichier_a_traiter
```

Les instructions que connaît Sed sont très limitées, mais pourtant l'outil est puissant. L'une d'elles est l'instruction de substitution. C'est là que l'on fournira à Sed une expression régulière et, lorsqu'il y aura correspondance de motif, Sed modifiera en conséquence la ligne concernée dans le fichier traité. Cela semble parfaitement malcommode et en apparence ça l'est. Cependant Sed permet des commandes d'une efficacité redoutable.

6 MODIFIER LE CONTENU D'UN LOT DE FICHIERS DE CONFIGURATION

Supposons que nous voulions déménager disons, la centaine de sites hébergés sur un serveur vers un autre serveur. Ce cas de figure est très fréquent et votre serveur a eu à le traiter tout récemment. Au cours de l'opération, il faudra, par exemple, modifier tous les fichiers de configuration de nom de domaine, c'est-à-dire les fichiers **.hosts* sous le répertoire */etc/bind/pri* Un fichier *.hosts* contient des lignes comme celles-ci

	IN	A	1.2.3.4
www	IN	A	1.2.3.4
mail	IN	A	1.2.3.4
smtp	IN	A	1.2.3.4
pop	IN	A	1.2.3.4
pop3	IN	A	1.2.3.4
imap	IN	A	1.2.3.4
sql	IN	A	1.2.3.4
mysql	IN	A	1.2.3.4

où 1.2.3.4 est bien entendu une adresse IPv4 de serveur. Il faudra donc modifier cette adresse, ce qui peut représenter au total des milliers de lignes à corriger, ce qui est parfaitement rébarbatif et générateur d'erreurs de frappe. C'est là que Sed intervient :

```
sed -e 's/1\2\3\4/4\3\2\1/g' < fichier > nouveau fichier
```

fera le travail. Ici l'instruction est *s* pour substitution. Vient ensuite le séparateur */* puis l'adresse IP de départ, qui fera office d'expression régulière, et dont les *.* sont échappés, puis un autre séparateur */* puis l'adresse ip d'arrivée.

Enfin derrière un dernier séparateur vient l'option *g*, pour global, qui indique à Sed de traiter toutes les lignes au lieu de s'arrêter à la première ligne du fichier qui présente une correspondance de motif. Mettons cela dans un script qui transformera d'un coup tous les fichiers **.hosts*, tout en les plaçant dans un sous-répertoire cible.

```
#!/bin/bash

fichiers=$(ls *.hosts)

for fichier in $fichiers
do
    sed -e 's/1\2\3\4/4\3\2\1/g' $fichier > ./cible/$fichier
done
```

Sed vient déjà de démontrer son utilité incontestable, mais il est des cas où l'on pense rarement à l'utiliser, comme par exemple le renommage de fichiers en masse, alors qu'il fait merveille.

7 TRAVAILLER AVEC DES NOMS DE FICHIERS

Voici un autre exemple tiré d'une situation réelle. Dans un répertoire se trouvent, parmi d'autres fichiers, des fichiers PDF qu'il faut déplacer dans un répertoire cible en les renommant de façon à ce qu'il ne soit pas possible de deviner le nom des fichiers déplacés.

Le choix est fait d'ajouter un GUID dans le nom du fichier. Par exemple :

```
fichier-1.pdf
```

doit devenir :

```
cible/fichier-1-81fe4796-ab67-11e2-b1c3-0800275cf909.pdf
```

Là encore, s'il y a une poignée de fichiers à traiter, on le fera à la main. S'il y en a quelques milliers, les expressions régulières et Sed vont nous sauver la mise. Commençons par filtrer les fichiers PDF au moyen de la commande *find* :

```
find . -maxdepth 1 -type f -regex '.*\.pdf$'
```

Pour le fun, nous montrons ici que *find* comprend lui aussi les expressions régulières, mais un simple :

```
find . -maxdepth 1 -type f -name '*.pdf'
```

fonctionnait aussi :) En revanche le commutateur *maxdepth* est nécessaire pour empêcher *find* de parcourir récursivement l'arborescence de répertoires. On remarque que *find* renvoie des noms de fichiers incluant le répertoire :

```
./fichier-1.pdf
./fichier-2.pdf
```

etc.

Maintenant nous ajoutons la commande Sed magique :

```
find . -maxdepth 1 -type f -regex '.*\.pdf$' | sed -r 's/^\.
/(.+)\.(.+)/mv & .\cible\1-\$(uuid).\2/'
```

Mais qui mérite quand même quelques petites explications :) Ainsi

```
^\.\/(.+)\.(.+)
```

filtre les noms commençant par *./* sachant que *.* doit être échappé ainsi que nous l'avons déjà vu et que */* doit être échappé pour ne pas être confondu avec le séparateur d'instructions de Sed, puis on accepte tout, mais en le séparant en 2 groupes de caractères délimités par des parenthèses et séparés par un point échappé. Ce décorticage du nom du fichier permet de s'appuyer ensuite sur une fonctionnalité unique de Sed, les références arrière. Ainsi les noms de fichiers sont remplacés en sortie de cette façon :

```
mv & .\cible\1-\$(uuid).\2/
```

ou *\1* et *\2* sont les deux groupes de caractères, précédemment isolés dans les parenthèses de l'expression régulière. Bien remarquer aussi la présence de l'esperluette, qui représente la correspondance à l'expression régulière dans la ligne d'entrée. A ce stade, essayez la commande dans un répertoire qui contient des fichiers PDF. Elle vous produira une sortie ressemblant à ceci :

```
mv ./fichier-2.pdf ./cible/fichier-2-\$(uuid).pdf
mv ./fichier-1.pdf ./cible/fichier-1-\$(uuid).pdf
```

Nous avons donc, comme nous le voulons, des commandes Shell qui déplacent des fichiers, en leur ajoutant un GUID grâce à la commande *uuid*. Il ne reste plus qu'à exécuter tout cela, en l'injectant dans un Shell :

```
find . -maxdepth 1 -type f -regex '.*\.pdf$' | sed -r 's/^\.
/(.+)\.(.+)/mv & .\cible\1-\$(uuid).\2/' | bash
```

8 TRAVAILLER AVEC LES BASES DE DONNÉES

Autre cas de figure tiré d'une situation réelle. Une table MySQL a été effacée par maladresse dans phpMyAdmin. Bien sûr une sauvegarde existe, mais l'administrateur système a fait un dump global de toutes les bases de données présentes sur le serveur, ce qui donne un fichier de plusieurs giga-octets. Bref, un fichier gigantesque qu'il est absolument impossible de manipuler avec un éditeur, aussi moderne soit-il. Fort heureusement, Sed est encore là, qui sait sortir d'un fichier tout ce qu'il y a entre deux expressions régulières séparées par une virgule. Supposons que la table à récupérer s'appelle 'languages'. Une seule ligne de Sed vous sauvera la vie en récupérant tout ce qu'il y a dans le sandwich des instructions SQL *CREATE TABLE* du fichier de sauvegarde :

```
sed -n -e '/CREATE TABLE.*languages/,/CREATE TABLE/p' fichier-
gigantesque.sql > languages.sql
```

Bien remarquer la présence du commutateur *-n* conjointe à l'option */p* (pour print) qui demande à Sed de sauter toutes les lignes, sauf celles qui sont dans le sandwich, tranche de pains incluses :-)

Frédéric Mazué - fmazue@programmez.com

ABONNEMENT

PDF

30 € par an

soit 2,73 € le numéro

www.programmez.com



Abonnement INTÉGRAL

Pour un supplément de 10 € an
accès illimité aux archives

Cette option est réservée aux abonnés pour 1 an au magazine, quel que soit le type d'abonnement (Standard, Numérique, Etudiant). Le prix de leur abonnement normal est majoré de 10 € (prix identique pour toutes zones géographiques). Pendant la durée de leur abonnement, ils ont ainsi accès, en supplément, à tous les anciens numéros et articles/ dossiers parus.

**PRIX
OCIRP**

**Acteurs
Économiques
& Handicap**

**6^e
ÉDITION**

Atalan

Lauréat
du Prix "Innovation et
nouvelles
technologies"



L'OCIRP, les organismes de prévoyance qui le composent et ses partenaires sont fiers de mettre en lumière ce projet particulièrement exemplaire. AcceDe Web est un projet collaboratif initié par Atalan, en partenariat avec 10 grandes entreprises, des associations, des écoles et institutions. L'idée ? Créer des outils méthodologiques

facilitant la mise en œuvre des référentiels accessibilité numérique, jugés trop complexes par les équipes métiers. Ces notices AcceDe Web, diffusées gratuitement, s'adressent aux différents acteurs du web et leur expliquent simplement ce qu'ils doivent faire pour concevoir des sites web accessibles. En savoir plus sur www.exempleasivre.fr ou sur www.ocirp.fr

LES PARTENAIRES FONDATEURS



LES PARTENAIRES MÉDIA



handicap.fr



OCIRP

unis par excellence

WINDEV®

ATELIER DE GÉNIE LOGICIEL
PROFESSIONNEL

DÉVELOPPEZ 10 FOIS PLUS VITE



Windows 8, 7, Vista, XP...,
Linux, Mac, Internet, Intranet,
Cloud, Android, iPhone, iPad...

Vos applications sont compatibles

Environnement de développement professionnel, intégralement en français (logiciel, documentations, exemples)

La facilité de développement avec WINDEV est devenue légendaire: vos équipes développent plus vite, la qualité des logiciels est automatiquement élevée, le nombre de fonctionnalités de pointe automatiques est impressionnant. Vous délivrez plus vite vos logiciels, pour la plus grande satisfaction et rentabilité des utilisateurs.



DEMANDEZ VOTRE DOSSIER GRATUIT

260 pages - 100 témoignages - DVD Tél: **04.67.032.032** info@pcsoft.fr

www.pcsoft.fr



Fournisseur Officiel de la Préparation Olympique



**VERSION
EXPRESS
GRATUITE**
Téléchargez-la !