Avant-première : BizTalk Server 2009 p.22

PROGRAMMEZ!

Le magazine du développement

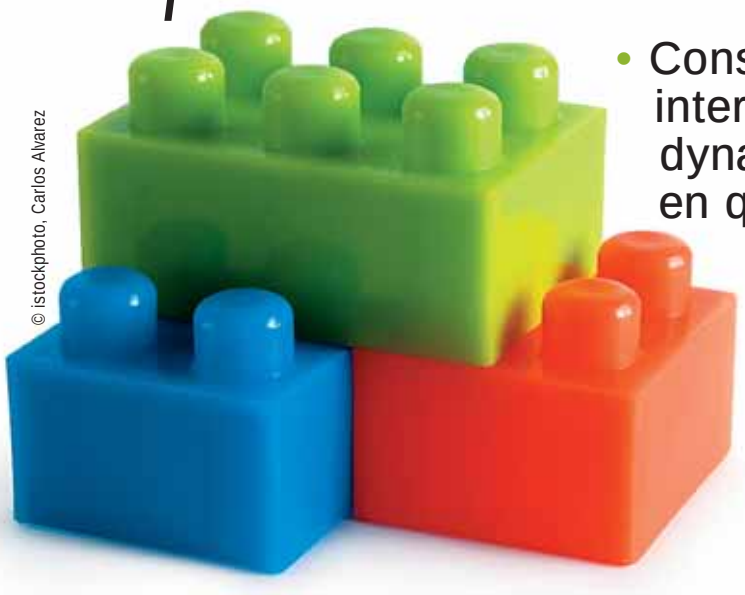
PROgrammez!

www.programmez.com

mensuel n°116 - février 2009

Composants

Simplifiez-vous le code !



© istockphoto, Carlos Alvarez

- Construire des interfaces 3D et dynamiques en quelques clics.
- Les meilleurs composants pour Java, PHP, Flex, .Net !

p.30

Programmation multicore

Découvrez le tout nouveau

Intel Parallel Studio p.58

Java p.14

Coder à la vitesse de la lumière avec

Eclipse !

Python 3.0

Toutes les nouveautés p.19

OpenOffice

victime de la crise ? p.12

Développement web

Devenez un expert **ASP.Net** p.44

Créez votre premier site en quelques clics
Boostez ASP.Net avec jQuery !
Découvrez toute la puissance de ASP.Net Dynamic Data

C++

- **Le nouveau standard C++ arrive**
- **Ecrire des extensions Ruby en C++**

Office

Coder avec le nouveau OpenXML SDK

Web

Mettre en œuvre jQuery

Jeux : **développer son jeu avec XNA Game Studio**

M 04319 - 116 - F: 5,95 €

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

WINDEV WEBDEV WINDEV Mobile

DÉVELOPPEZ 10 FOIS PLUS VITE

Nouvelle version 14

501 NOUVEAUTÉS

Réussissez tous vos projets avec l'outil de développement le plus productif du marché*.

Ouverture
Robustesse
Pérennité
Réduction des coûts
Rentabilité
Adéquation aux besoins
Gestion du changement
Sécurité

UN CODE PORTABLE:
Windows, .Net, Java, PHP, J2EE, XML, Internet, Intranet, Pocket PC, SmartPhone, Client riche ...

VERSION EXPRESS GRATUITE
Téléchargez-la !

14 Nouveau :

- Mashup
- Lien Google
- Lien Salesforce
- HyperFileSQL : full text
- DataBinding
- Nouveaux graphiques
- Nouvelles tables
- Robot de monitoring & surveillance
- Accès Natif PostgreSQL
- Lien Silverlight 2 et Flex
- PHP 5
- 214 Nouveautés fonctionnelles
- 120 Nouvelles fonctions WLangage
- 62 Nouvelles fonctions Java
- 32 Nouvelles fonctions PHP
- 101 Nouvelles fonctions LINUX

PC SOFT 25 ANNÉE

www.pcsoft.fr

DEMANDEZ LE DOSSIER GRATUIT : 244 pages + DVD + **Version Express incluse** + 112 Témoignages.
Tél: **04.67.032.032** ou **01.48.01.48.88** Mail: **info@pcsoft.fr**

Fournisseur Officiel de la Préparation Olympique

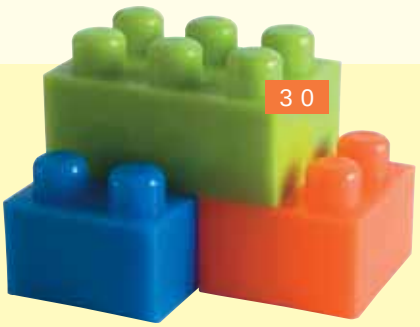
Logiciel professionnel. Document non contractuel. Support technique gratuit : 15 requêtes sur la version en cours de commercialisation.
*. WINDEV a été élu «Langage le plus productif du marché» par les lecteurs de la revue «Programmez!», octobre 2008

PROgrammez! le magazine du développement
numéro 116 - février 2009

sommaire\\

\\ spécial Biztalk Server 2009, l'architecture orientée développeur

Microsoft BizTalk Server 2009



\\ dossier : Composants simplifiez-vous le code !

\\ développement web

\\ carrière

\\ code

\\ temps libre

CD ROM

Intel Parallel Studio *beta*

Parallel Amplifier permet de trouver les goulots d'étranglement et de donner des éléments de réponse à la question cruciale du développement parallèle : est-ce que mon code utilise au mieux les ressources et le potentiel de performance du processeur multi-core ?

Parallel Inspector permet de trouver rapidement et sans failles les erreurs classiques et difficiles à détecter du multi-threading : les « race conditions » et les « deadlocks ». Inspector intègre également un analyseur de mémoire très performant pour détecter les erreurs et les fuites de mémoire.

Parallel Composer comprend un compilateur haute performance C/C++, compatible avec le compilateur Microsoft ainsi que la librairie multimédia Intel Performance Primitives (IPP) et la librairie de multi-threading Threading Building Blocks (TBB).

INTEL PARALLEL STUDIO EST COMPATIBLE AVEC VISUAL STUDIO 2005 ET 2008.
LES 3 MODULES PEUVENT ÊTRE INSTALLÉS ENSEMBLE OU SÉPARÉMENT.

voir article page 58

Enregistrement nécessaire pour obtenir la clé d'activation :
https://registrationcenter.intel.com/BetaCenter/BetaForm.aspx?ProductID=1261&ProgramID=Programmez

éditoll

Quand les erreurs deviennent des fautes

Un rapport explosif rédigé par 30 organisations *, pointe du doigt 25 erreurs de codage. Deux d'entre elles sont cause du piratage de plus d'1,5 million de sites web et de plusieurs millions de PC zombies, chez les utilisateurs de ces sites. Ces simples erreurs sont donc de véritables fautes ! Le rapport conclut en recommandant aux acheteurs de logiciels d'exiger de l'éditeur une garantie de la qualité du code, et de " certifier que le code qu'il fournit ne comporte aucune des 25 erreurs de programmation ". L'éditeur aurait la " responsabilité de corriger ces erreurs et de supporter les dommages qu'elles auraient causés ". Il prévoit l'usage systématique de logiciels de test et l'étude dans les écoles d'informatique de ces erreurs.

Les grands classiques restent pourtant les mêmes : injection SQL ou de code, mauvais contrôle des entrées (par exemple sur un champ), etc. Or pour le développeur, ces oublis peuvent être facilement corrigés sans que cela coûte cher en temps, ou impacte réellement l'ensemble de l'application. Mais encore faut-il y penser !

Et si dans certains cas les erreurs de programmation découlaient de problèmes de compréhension de ce langage technique omniprésent, qui mêle le jargon et l'anglais ? *Array, DSL, pattern, anti pattern, Ajax, classes partielles, cross compilation, cloud computing, SaaS,* compilation déportée, *remote debug*, accessibilité, etc. C'est parfois une langue étrangère pour le développeur non aguerri !
D'ailleurs, nous nous posons la question, à la rédaction de Programmez, en écrivant ou relisant les articles : faut-il traduire, ou décoder ?

Nous avons assisté récemment à une réunion autour de Visual Studio. Et un des points, longuement abordé, fut la traduction des termes de l'environnement de développement en français. Faut-il à tout prix tout traduire, ou bien mixer français et anglais ? Car si de nombreux développeurs avancés privilégient les outils en anglais, la majorité cependant préfère la version française, ce qui est légitime. Mais encore faut-il éviter de donner une mauvaise traduction et surtout de changer la terminologie d'une version à une autre ! C'est pour cela que les éditeurs emploient des terminologues et essaient d'impliquer les communautés dans cette réflexion. Quelques exemples tout bêtes : faut-il traduire " Template " ? Comment traduire " work item " ? Faut-il systématiquement traduire les Menus, au risque quand on scripte son outil, d'être incompatible avec une version non française car n'utilisant pas les mêmes termes de menu ?
Autre point délicat, donner une unique définition pour un terme ou tout du moins la même approche. Ainsi, dans le monde des services en ligne, hébergés (SaaS pour faire court), ou dans l'approche du " nuage informatique ", chaque éditeur y met un peu ce qu'il veut. Il existe là deux modes de lecture : une lecture marketing que l'éditeur veut faire passer, et une lecture technique... quand elle existe ! Bon déchiffrement !

François Tonic et **Jean Kaminsky**

* http://www.programmez.com/actualites.php?titre_actu=Les-25-pires-erreurs-de-programmation-selon-la-NSA&id_actu=4166

http://www.sans.org/top25errors/?utm_source=web&utm_medium=text-ad&utm_content=Announcement_Bar_20090111&utm_campaign=Top25&ref=37029

PROgrammez! Le magazine du développement

Rédaction : redaction@programmez.com
Directeur de la Rédaction : Jean Kaminsky
Rédacteur en Chef :
François Tonic - ftonic@programmez.com
Ont collaboré : F. Mazué, C.Remy, Y. Biet.
Experts : N. Bros, O. Lauzanne, E. Coirier, D. Nolting, R. Baduel, C. Manoliu, G. Harrath, J. Revel, J. Pic, D. Rethana, R. Benedetti, L. Bar, L. Capin, A. Boufous, L.Lefort, R. de Wargny, S.Saurel. J. Chable, F. Pedro, N. Gryman.
Crédits photo : © istockphoto, Carlos Alvarez
© François Cointe " Architectures Orientées Sevices "
Publicité : Régie publicitaire, K-Now sarl
Pour la publicité uniquement :
Tél. : 01 41 77 16 03 - diff@programmez.com
Editeur : Go-02 sarl, 6 rue Bezout
75014 Paris - diff@programmez.com
Dépôt légal : à parution - Commission paritaire : 0712K78366 ISSN : 1627-0908
Imprimeur : ETC - 76198 Yvetot
Directeur de la publication : J-C Vaudecrane
Ce numéro comporte 1 cd rom

Abonnement : Programmez 22, rue René Boulanger, 75472 Paris Cedex 10 - abonnements .programmez@groupe-gli.com Tél. : 01 55 56 70 55 - Fax : 01 55 56 70 20 - 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 : 45 € - Etudiant : 39 € - CEE et Suisse : 51,83 € Algérie, Maroc, Tunisie : 55,95 € Canada : 64,33 € Tom : 79,61 € - Dom : 62,84 € Autres pays : nous consulter. **PDF :** 30 € (Monde Entier) souscription en ligne.

L'INFO PERMANENTE
WWW.PROGRAMMEZ.COM

PROCHAIN NUMÉRO

N°117 mars 2009, parution 28 février

✔ **SGBD**
Sécurité, optimisation, performance, mapping !

✔ **Hacking : haro sur le code !**
Comment renforcer, optimiser le code ?

✔ **Comment choisir son framework Ajax,**

✔ Préparez-vous à **Windows 7**

Votre potentiel, notre passion.™

Microsoft®

■ Avec **Windows 7**, Microsoft veut améliorer la consommation et la gestion de l'énergie sur portable (netbook / notebook). Cela passera par la disponibilité de nouvelles API et une gestion très fine de l'activation ou de la mise en sommeil d'un ou plusieurs cœurs du processeur.

■ **Perforce** améliore encore la gestion des projets et des sources avec la version 2008.2 de Perforce SCM, le système de configuration logicielle. Il est maintenant possible de voir la propagation de modifications entre deux branches, pratique pour se rendre compte de l'impact d'une intégration. Autre nouveauté : la possibilité de définir rapidement une tâche. On notera aussi une fonction intéressante : découverte du réseau. Le tableau de bord a aussi été amélioré. Site : www.perforce.com

■ **DeviceAnywhere** renforce sa présence française avec l'ouverture d'un Datacenter de test, à Paris. La société se spécialisant sur le test applicatif sur terminaux mobiles, plus de 150 modèles sont disponibles.

■ **Symantec** renforce la disponibilité et la reprise d'activité en environnement VMware. Cette annonce fait suite au rachat de Veritas. Veritas Cluster Server de Symantec est une solution complémentaire aux solutions VMware et permet de protéger les applications critiques contre des arrêts non planifiés.

■ **Quest Software** lance PowerGUI 2.0 pour VMware. Il s'agit d'une bibliothèque de scripts PowerShell pour l'administration des environnements virtualisés. Cette version s'intègre au SDK de Vmware, permet la création de rapports ou encore la gestion des sessions. Disponible gratuitement : www.powergui.org

■ Le **cityguide Qype** lance sa nouvelle API permettant d'exploiter les données du site. Elle s'appuie sur http, XML et RESTful. Parmi les fonctions disponibles : recherche dans une ville, afficher des détails (adresse, photos...). Pour y accéder, il est nécessaire de posséder une clé. Site : http://www.qype.fr/developers/api

■ Les utilisateurs **MySQL** ont leur club utilisateur et surtout un site web : lemurg.fr. L'association a pour but de promotionner MySQL, de réunir les utilisateurs, de participer aux développements du SGBD.

Mobile
Palm Pré : la bonne réponse ?

Le constructeur du fameux Palm Pilot est-il sur le point de renaître ? Après plusieurs années très difficiles et une concurrence féroce, Palm a dévoilé son nouveau produit : Palm Pré. Sans vouloir clôner le iPhone d'Apple, le Pré tente au contraire d'être une alternative avec ses propres caractéristiques. Doté d'un design réussi, il fonctionne sur un système Linux, le webOS. Nous ne connaissons pas encore l'ensemble des détails fonctionnels, ni le prix, ni la date exacte de disponibilité. Sur le SDK, à l'heure où nous écrivons ces lignes, peu d'informations circulent. Nous savons que le SDK fonctionne sur Mac et aussi sur Windows. On disposerait d'un runtime d'exécution à l'intérieur d'un navigateur. Nous savons tout de même que le kit de développement est Palm Mojo Application Framework et le Palm Mojo SDK. Les applications du Pré seront surtout des applications web utilisant HTML, CSS et javascript, sans oublier JSON. Le stockage se fera via les capacités de stockage de HTML5. Actuellement ce SDK est en pré-bêta privé.

Web
Google prépare Chrome 2

L'éditeur avance rapidement sur son navigateur intégré, Chrome. C'est désormais la v2 qui est activement en préparation. Pour pouvoir tester Chrome 2, il faut donc mettre à jour son Chrome via le Chrome Channel Changer. Il permet d'accéder aux différents canaux de mise à jour : stables, bêta, préversion technique. Il suffit de sélectionner le canal puis de mettre à jour.
Cette version intègre les dernières évolutions de webkit (moteur de rendu) avec un nouveau mécanisme de zoom et l'autoscroll. Le navigateur inaugure aussi sa propre implémentation http en lieu et place de Winhttp. Autre bonne nouvelle, le passage des tests Acid3. Chrome 2 permet aussi de créer différents profils (permettant de partager la même instance de navigateur entre plusieurs utilisateurs). Il ne reste plus qu'à sortir les versions Mac et Linux !

agenda \\

FEVRIER

• Jeudi 05 février 2009, Paris 8e
Cocktail JAVA EE5, JAVA EE6, GLASSFISH
Assistez à une présentation gratuite animée par Alexis Moussine-Pouchkine Java Web Services Architect Ambassador - Sun Microsystems.
http://www.demos.fr/espace-metier/informatique/actualites/Pages/cocktail-java-glassfish.aspx

• Du 10 au 12 février 2009, Paris 17 e Palais des congrès. **Microsoft TechDays 2009**, le rendez-vous incontournable des professionnels de l'informatique.
http://www.microsoft.com/france/mstechdays/

• Mercredi 11 février 2009, Lille Salle du Gymnase – Place Sébastopol. **Salon Les jeudis**
http://www.lesjeudis.com/salons-informatiques/

• Le Jeudi 12 février 2009, Paris, Porte d'Italie
Symposium vie privée et sécurité sur internet, organisé par l'Epitech. www.epitech.eu

MARS

• Du 31 mars au 2 avril, Paris Expo Porte de Versailles. **RTS Embedded Systems 2009**
http://2009.salon-rts.fr/

• Du 31 Mars au 2 avril, Paris Expo Porte de Versailles, **Solutions Linux Open Source,** 10e édition de l'événement européen de référence sur le marché Linux et du logiciel libre. http://www.solutionslinux.fr/main.php

Méthode

25 erreurs de développement à ne pas faire !

Que font des dizaines d'experts américains et européens quand ils parlent programmation ? Ils établissent la liste des 25 erreurs de programmation les plus dangereuses, à éviter absolument ! La liste des consultants est impressionnante : Symantec, Microsoft, NSA, EMC, etc. L'objectif est de sensibiliser les développeurs à mieux sécuriser leur code et donc les applications. Ces erreurs se répartissent en trois grandes catégories : l'insécurité des interactions entre les composants, les risques de gestion des ressources et la défense poreuse...

Dans la première catégorie, nous retrouvons des failles classiques du développement que l'on rencontre malheureusement encore bien trop souvent : mauvaise validation des entrées, mauvaise sortie, injection SQL, cross site scripting, injection de commande système, transmission en clair d'informations sensibles. Dans la partie ressource, nous retrouvons des erreurs sur la gestion de la mémoire buffer, le contrôle (externe) des noms de fichier ou du chemin d'accès, l'injection de code. Enfin dans la 3e section, nous sommes plus dans l'administration et la sécurité comme un contrôle d'accès non conforme, un algorithme de cryptage cassé mais tout de même employé, exécution avec des privilèges non nécessaires. Pour chaque erreur, les auteurs décrivent le problème, notent la criticité, le coût de correction, le niveau des attaques... Et ils décortiquent point par point ce qu'implique l'erreur au niveau design et architecture du code avant de proposer des solutions et des conseils. Nous vous suggérons vivement de lire et relire cette analyse que tout développeur devrait avoir près de son clavier !

Site : http://www.sans.org/top25errors/

Système

Les kits Azure mis à jour

Microsoft continue de développer sa plate-forme Windows Azure. Aujourd'hui, l'éditeur rend disponible une nouvelle CTP (pré-version technique) : la CTP de janvier 2009. Elle contient les outils, les API nécessaires au développement d'applications pour Azure. Mais pour développer, il faut aussi avoir un IDE. On dispose également d'une nouvelle CTP de Azure Tools for Microsoft Visual Studio January CTP. Cette version comporte : des corrections de bugs, le support du debug pour Silverlight des messages d'erreurs en cas de problèmes sur le Service Configuration et le Service Definition, ainsi qu'une meilleure intégration avec la partie stockage. Pour accéder à Azure et surtout au portail développeur des Azure Services, vous devez disposer d'un compte Live ID. Il permet d'y déployer ses services.

Site : http://www.microsoft.com/azure/windowsazure.mspx

Microsoft TechDays : des nouveautés et du concret

Du 10 au 12 février se tiendront les TechDays de Microsoft à Paris. Comme les années précédentes, la première journée sera très orientée développeur, la seconde plus entreprise et la troisième avec une dominante sécurité. Toutefois, cette répartition sera en grande partie estompée car de nombreuses sessions mêleront développement et approche entreprise. Sur la partie purement sécurité, plus de 20 sessions sont prévues ainsi que sur l'architecture.

Le futur immédiat et le présent

Un des axes du salon sera de présenter et d'expliquer l'ensemble des nouveautés qui vont apparaître d'ici à 12 mois : Windows 7, Windows Azure (et le *cloud* en général), le multitouch et toute la partie .Net 4, Visual Studio 2010. Mais la majorité des sessions développeurs se focaliseront sur les technologies et outils actuels (.Net 3.5 SP1, Visual Studio 2008, SQL Server, Biztalk...). Un des fils rouges retenu concerne l'industrialisation du logiciel, notamment via Team Foundation, les DSL et l'adoption d'UML.

Le développement web sera particulièrement présent avec RIA, Silverlight, Expression Web. Mais si les technologies Microsoft sont bien entendues très présentes, l'open source n'est pas oublié. Comme dans l'édition 2008, des éditeurs libres seront présents sur scène, pour des sessions ou sur les stands. On peut ainsi noter une session PHP sur plate-forme Windows. Parmi les chantres de l'open source, citons la présence de Novell (virtualisation et Mono) et de Sun. Pour gérer en interne l'ensemble des contraintes (salles, disponibilités, intervenants, etc.), un projet spécifique a été développé pour concevoir un moteur de règles, le plus efficace possible. Pour cela, Microsoft France a utilisé le langage F# qui sera expliqué durant ces 3 jours... Côté fréquentation, l'éditeur espère arriver aux mêmes chiffres qu'en 2008, à savoir environ 11 000 visiteurs uniques et 16 000 en tout sur les trois journées. Cette année, les stands communautaires, Microsoft et les partenaires seront mieux regroupés et dispatchés entre les différents villages pour éviter d'isoler les sections comme ce fut le cas en 2008. Les développeurs, et les autres, pourront rencontrer des experts (Microsoft et partenaires) autour des Ask Me Expert.

Une Surface en fer et en " toucher "

Une des attractions attendues du salon est la présence durant la keynote développeur et sur le stand de la société Wygwam d'une *Surface*, la " table " multitouch de Microsoft, sortie officiellement l'an dernier. Wygwam est l'une des trois sociétés retenues par l'éditeur pour travailler sur la technologie Surface en France. Comme nous a confié Gregory Renard, le but est de permettre aux visiteurs de toucher, d'essayer, de voir les réelles capacités de Surface avec des applications pré-installées. Ainsi, on pourra " jouer " avec Virtual Earth. On s'attend aussi à quelques belles surprises durant les différentes keynotes...

CodeFluent 2009 lancé aux TechDays

Le Français Softfluent sortira la version 2009 de sa fabrique logicielle, CodeFluent durant le salon événement Microsoft. Version majeure, elle intègre de nombreuses nouveautés : génération d'interface Sharepoint, services de synchronisation avec Access, support de " Search " dans le langage interne CFQL (recherche multicritère), modélisation dynamique pour implémenter les patterns. Côté base de données, une des grosses nouveautés est le support de SQL Server 2008 et de Linq to SQL. Le retour des utilisateurs a aussi permis à l'éditeur d'affiner l'interface et l'ergonomie de l'outil.
Pour en savoir plus : http://www.softfluent.com/news/codefluent-2009-arrive-.aspx

Au-delà de février

D'autres événements se dérouleront durant le premier semestre 2009. Notons les deux principaux : un tour de France avec les Microsoft Days et sans doute une nouvelle édition de (Re)Mix Paris à une date et à un format non définis pour le moment.

WINDEV WEBDEV WINDEV Mobile

NOUVELLE VERSION 14

PLATEFORME PROFESSIONNELLE DE DÉVELOPPEMENT (AGL)
Internet, Intranet, **WEB 2.0**, RAD **PHP**
Lien à **toutes les bases de données**

WEB 2.0
Ajax en 1 clic
avec WEBDEV 14

PC SOFT 25ème ANNÉE Au service des développeurs

*Sous l'**éditeur de code** de WEBDEV : **un clic** et le traitement programmé devient «**Ajax**»*

WEBDEV 14 permet de développer jusqu'à 10 fois plus vite tous les types de sites et d'applications reliés aux données de votre entreprise. L'activation d'**AJAX** s'effectue naturellement: un simple clic indique que le code à effectuer est de type «**Ajax**». WEBDEV 14 est certainement le seul environnement au monde à proposer autant de souplesse et de puissance.

AGL complet, langage L5G, RAD PHP, débogueur, Web Services, gestionnaire de sources, installateur, base de données SQL intégrée et lien avec toutes les bases du marché, composants, éditeur d'états PDF et code-barres, règles métier, dossier, outils d'administration...: tout est inclus, en français.

Vous aussi réalisez vos sites WEB 2.0 10 fois plus vite... avec WEBDEV 14
(logiciel professionnel, dossier gratuit sur simple demande)

WEBDEV 14 est compatible avec WINDEV 14

***Un des nombreux exemples livrés avec WEBDEV : portail Intranet Honolulu, à télécharger gratuitement** (sur pcsoft.fr).*

www.pcsoft.fr info@pcsoft.fr
Dossier technique et témoignages **sur simple demande**
Tél Province 04.67.032.032 Tél Paris 01.48.01.48.88

UN CODE MULTI-PLATEFORME : WINDOWS, WEB, MOBILE

WEBDEV®

Fournisseur Officiel de la Préparation Olympique

Open Source

La polémique autour d'OpenOffice cache-t-elle une crise de développeurs ?

Photo prise lors de la conférence OpenOffice.org 2007 par Guy Lunardi

Depuis quelques semaines, une polémique secoue une partie des communautés Open Source. La première pierre fut jetée par un développeur très actif sur OpenOffice, **Michael Meeks**, sur son blog. Après avoir vérifié les chiffres et fait des statistiques sur le projet, un constat s'impose : OpenOffice est profondément malade ! Car pour Meeks, il y a péril. Il met en exergue le faible nombre de développeurs " permanents " (développeurs actifs) pour maintenir et développer la suite bureautique. A la mi 2008, 24 développeurs dont la quasi totalité venant de Sun ! A la même époque en 2007, il y avait presque 60 développeurs dont une quarantaine de Sun ! Malgré tout la situation se tenait, tant que les contributeurs extérieurs continuaient à croître, ou à rester stables, ce qui n'est plus le cas. Michael compare leur effectif à celui des contributeurs actifs du noyau Linux, qui reste important : en 2008, les chiffres rassemblés par Meeks indiquent plus de 250 développeurs... soit 10 fois plus que pour la suite bureautique. Même si aux périodes creuses, Linux Kernel se contente d'à peine 160 personnes...

Ce n'est pas la première fois que l'on pointe le problème de la communauté des contributeurs de la suite. Meeks propose comme solutions possibles de changer la gouvernance actuelle et de s'éloigner de Sun, le projet étant toujours très lié à l'éditeur. Une charge contre Sun ? Meeks conclut sa démonstration ainsi : *"Pourquoi mon bug n'est pas corrigé ? Pourquoi l'interface est-elle déplaisante ? Pourquoi les performances sont-elles pauvres ? Pourquoi le projet consomme-t-il plus de mémoire que nécessaire ? Pourquoi est-il lent à démarrer ? "*.

Sun critiqué

Certains commentateurs voulant garder l'anonymat nous ont évoqué une possible manipulation, derrière cette attaque. Novell, qui emploie Meeks, est d'ailleurs critiqué par une partie de la communauté (http://boycottnovell.com/2009/01/09/interview-with-charles-h-schulz/) pour ses positions et développements envers OpenOffice et par rapport au projet Go-OO et le partenariat tissé avec Microsoft, notamment envers OpenXML. Meeks avait en juillet 2008 lancé à nos confrères derStandard.at que si Sun sortait d'OpenOffice.org cela ne pourrait être (entièrement) une chose négative... (http://derstandard.at/druck/?id=1216917892794)

Cependant, Sun a réaffirmé dès mi-novembre 2008 la continuité de son soutien au développement d'OpenOffice. A l'heure où nous mettons sous presse nous n'avions pas encore reçu la position officielle de Sun par rapport aux chiffres avancés par Meeks. Même si cela se révèle un coup d'épée dans l'eau, cette polémique montre les luttes d'influences et les risques de retrait d'un certain nombre de développeurs payés par des entreprises pour collaborer, développer dans des projets open source !

A lire : http://www.gnome.org/~michael/blog/ooo-commit-stats-2008.html

François **Tonic**

Cherche contributeurs désespérément ?

Cyril Lavier (contributeur actif d'Ubuntu) apporte son éclairage sur la situation : *" De mon point de vue, la baisse des contributeurs développeurs est visible, plusieurs projets auxquels je contribuais (principalement au niveau de la traduction française, ndlr) ont fait une pause sur leur développement. Cependant, je n'estime pas que la crise ou le désengagement des entreprises soient les deux seules causes. Il s'avère que beaucoup de ces projets sont gérés par des étudiants ou des jeunes adultes (fraîchement arrivés dans la vie active), donc durant leurs études et le début de leur travail, ces personnes ont encore assez de temps à consacrer à leurs projets libres, dès qu'ils finissent leurs études, ou évoluent dans leur emploi (et dans leur vie personnelle), le temps libre se réduit considérablement. J'ai connu cette situation. À l'heure actuelle, j'ai moins de temps libre pour contribuer qu'en 2006-2007, quand j'étais encore étudiant et ensuite, jeune arrivé dans la vie active. "*

L'Open Source profite de la crise

Tristan Nitot relativise les risques d'une perte de contributeurs: *"Il est certain que la réduction d'effectifs qui touche l'ensemble des industries, y compris le logiciel, aura un impact négatif sur l'emploi des ingénieurs travaillant sur du logiciel libre, même si Mozilla continue d'embaucher. Pourtant, nombreux sont ceux qui recommandent aux informaticiens qui perdent leur emploi de contribuer à un logiciel libre pendant leur recherche d'emploi. En effet, c'est une façon de cultiver ses compétences, d'en rajouter de nouvelles et aussi de gagner en visibilité. A ce titre, je pense que la crise est neutre pour le logiciel libre."*

WINDEV WEBDEV WINDEV Mobile

NOUVELLE VERSION 14

PLATEFORME PROFESSIONNELLE DE DÉVELOPPEMENT (AGL)
Pocket PC, Smartphone, Terminal
Windows CE, Windows Mobile

Mobiles : développez 10 fois plus vite avec WINDEV Mobile 14

VERSION EXPRESS GRATUITE
Téléchargez-la !

*Les **codes-barres** sont gérés en **standard***

WINDEV Mobile 14 permet de développer jusqu'à 10 fois plus vite les applications sur mobile dont votre entreprise et vos clients ont besoin: gestion de stock, force commerciale, géolocalisation, saisies médicales, expertises, relevés de terrain, prise de commande temps réel, réglage de chaîne, ...

La **puissance** et la **facilité** de développement de WINDEV Mobile 14 permettent un développement en quelques journées.

Déploiement **gratuit** sans redevances (base de données incluse), réplication, WiFi, 3G, Internet, lien avec votre S.I., ...

Vous aussi réalisez vos applications mobiles 10 fois plus vite... avec WINDEV Mobile 14

*(Logiciel professionnel, dossier gratuit sur simple demande. Version Express gratuite en téléchargement libre sur **www.pcsoft.fr**)*

WINDEV Mobile 14 est compatible avec WINDEV 14 et WEBDEV 14

Dossier technique et témoignages **sur simple demande**

UN CODE MULTI-PLATEFORME : WINDOWS, WEB, MOBILE

WINDEV® Mobile

www.pcsoft.fr

Coder à la vitesse de la pensée avec Eclipse !

Eclipse 3.4 propose de nombreuses fonctionnalités avancées pour l'écriture de code Java. Nous allons voir dans cet article comment tirer parti de ces astuces pour améliorer la vitesse du développement.

Complétion

Tapez les premières lettres d'un nom de classe, de champ ou de méthode, et invoquez la complétion par la combinaison de touches « Ctrl+Espace »(1). Eclipse propose alors une liste des éléments commençant par le préfixe donné, triés par pertinence. Sélectionnez l'élément souhaité dans cette liste pour l'insérer, et si c'est une classe, rajoutez automatiquement la directive d'import correspondante. La complétion fonctionne aussi à partir des initiales du nom en « *CamelCase* »(2), ou des premières lettres de chaque groupe. Par exemple, invoquer la complétion sur le préfixe « CNFE » ou « ClaNFExc » propose « ClassNotFoundException ». [Fig.1]

Il est même possible de demander la complétion pour les noms de variables dans les déclarations. Par exemple, après « FileWriter », la complétion propose « fileWriter » et « writer » comme noms de variables. Si vous avez des conventions différentes, par exemple si vous préfixez tous vos champs par « f », vous pouvez le préciser dans la page de préférence « Java > Code Style », et Eclipse en tiendra compte. [Fig.2] Pour créer automatiquement le corps d'une classe anonyme avec le squelette de toutes les méthodes à implémenter, positionnez le curseur entre les parenthèses d'un constructeur (par exemple « new ActionListener() »), et invoquez la complétion. Pour créer automatiquement vos *getters* et *setters,* tapez « get » ou « set » et invoquez la complétion.

On dispose aussi de *templates* de code dans Eclipse. Par exemple, tapez « main » suivi de la combinaison de touches « Ctrl+Espace » pour insérer le code suivant :

```
public static void main(String[] args) {

}
```

Il existe des *templates* prédéfinis pour la plupart des formes syntaxiques de Java. Il est aussi possible de définir ses propres *templates :* ouvrir les préférences d'Eclipse, et aller dans la page « Java > Editor > Templates ». Ceci permet de se définir des *templates* spécifiques, qui pourront ensuite être exportés et partagés par les membres d'une équipe.

Quick fixes

Eclipse détecte automatiquement les erreurs pendant la frappe, et propose un mécanisme pour les corriger automatiquement. Ce mécanisme porte le nom de *« quick fixes »*. Pour invoquer un *quick fix,* positionnez le curseur texte à l'endroit de l'erreur, et effectuez la combinaison de touches « Ctrl+1 » (ou « Ctrl+& » sur un clavier AZERTY). Eclipse affiche alors une liste d'actions possibles pour corriger l'erreur.

Par exemple, supposons que vous ayez défini une classe « Dimension », et que vous écriviez dans votre code « dimension.resize(1,2); ». Comme la méthode « resize » n'est pas encore définie, ceci provoque une erreur qu'Eclipse peut corriger par un *quick fix* qui va créer dans la classe « Dimension » une méthode « resize » publique prenant deux entiers en paramètres. [Fig.3]

C'est aussi un excellent outil pour écrire du code rapidement. Pour ceci, il peut être utile d'omettre certaines parties du code, puis d'invoquer les *quick fixes* pour les insérer. Par exemple, tapez une expression, puis utilisez le *quickfix* « Assign statement to new local variable » pour déclarer une variable locale prenant comme valeur l'expression. Eclipse infère alors le type de l'expression et un nom pour la variable, qui se révèle judicieux la plupart du temps. [Fig.4]

Eclipse propose ainsi près de 60 *quick fixes* (3) rien que pour le code Java dans la version 3.4 (Ganymede). Il est aussi possible de définir

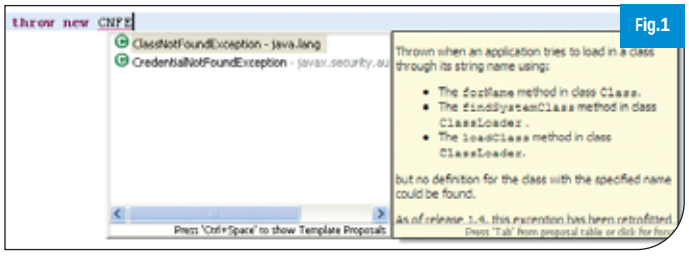


Fig.1

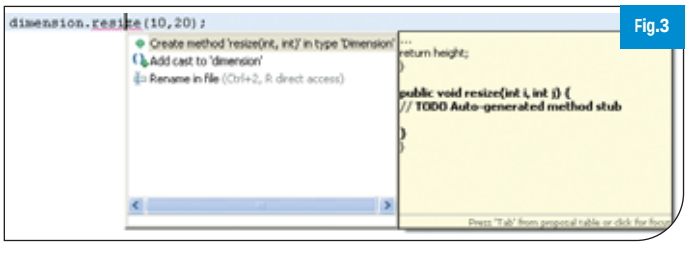


Fig.3

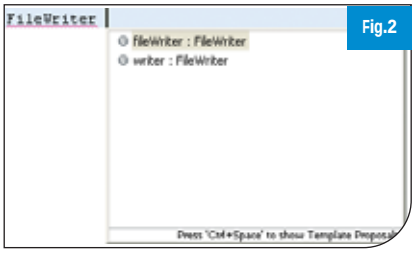


Fig.2

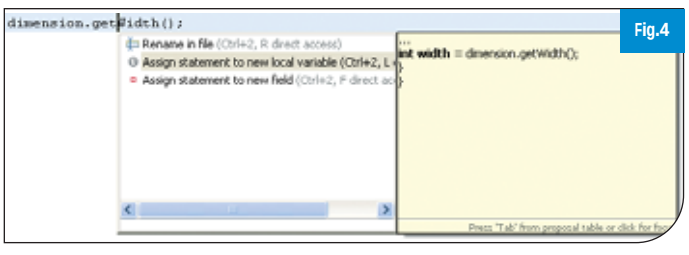


Fig.4

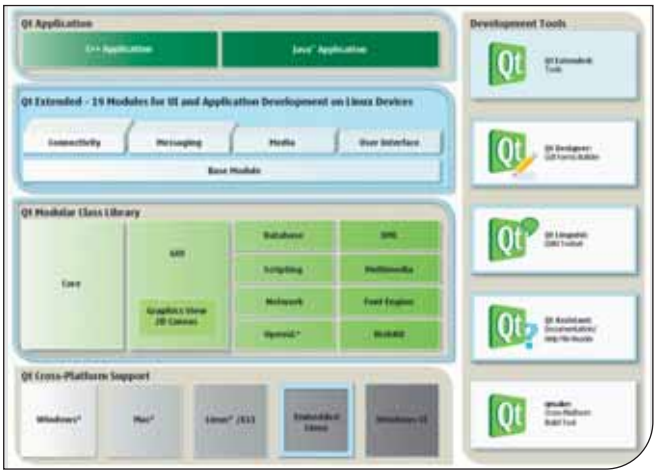
(1) Tous les raccourcis clavier sont redéfinissables dans Eclipse, dans la page de préférence " General > Keys ".

(2) Noms accolés sans espace, chacun commençant par une majuscule.

Suite page 18…

Librairie

Qt 4.5 avance et change de licence



Le rachat de Trolltech par Nokia modifie la stratégie de Qt. Désormais, la librairie C++ pour créer des interfaces graphiques multi plates- formes sera disponible en licence LGPL 2.1 avec la sortie de Qt 4.5 (mars prochain). La licence GPL est gardée. D'autre part, le source de Qt sera publique aussi bien pour la partie desktop qu'embarqué ! L'éditeur souhaite inciter les développeurs à contribuer sur les librairies. Le changement de licence s'explique parfaitement ! D'autre part, une licence commerciale sera aussi disponible, comme c'est le cas actuellement. Pour Nokia, il s'agit outre les contributions, de stimuler une communauté et d'élargir l'usage de Qt. Et sans doute que Nokia souhaite améliorer le support de Qt dans Symbian.

Qt 4.5 apporte des modifications assez profondes avec le support du 64-bit sur le Framework Cocoa. Notons que pour le développeur Mac, il y aura le choix entre Carbon et Cocoa, 32 et 64-bits. WebKit sera aussi mis à jour. Le développeur pourra désormais passer par un debugger livré en standard pour le moteur de Scripting Qt Script. Autre nouveauté intéressante, le support du format OpenDocument et diverses améliorations sur Qt Designer. Site : www.qtsoftware.com

■ **Google Maps** en .Net ? C'est possible avec le framework GMap.Net. Le but est d'interfacer Google Maps avec une application .net. Le tout se présentant comme une dll fournissant les contrôles... site : http://www.codeproject.com/KB/miscctrl/GMap-NET.aspx

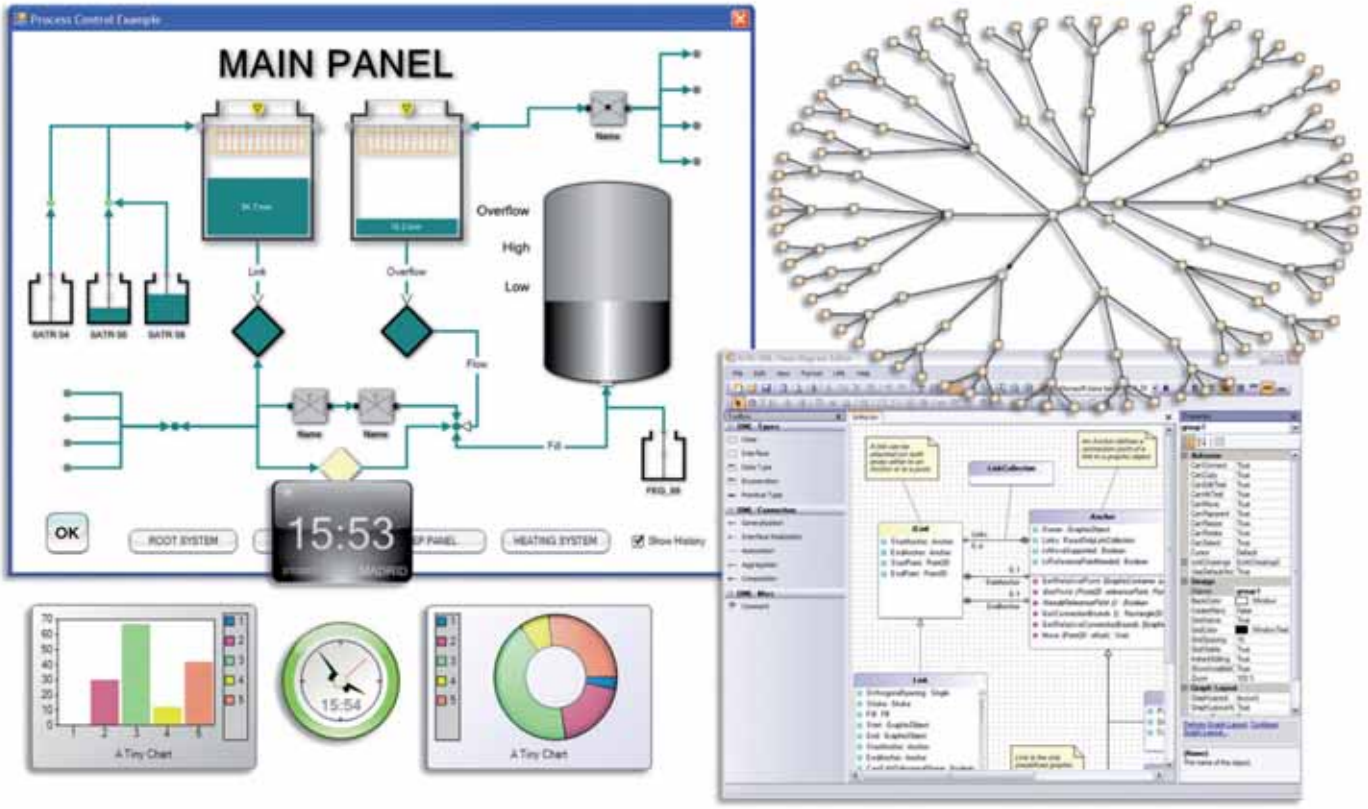
■ Comment valider son modèle **MVC** issu de ASP.Net MVC ? Avec le framework de validation xVal. Il utilise customattributes et Castle. Autorise aussi l'internationalisation des messages. Un projet à suivre : http://xval.codeplex.com/

■ **Google** arrête plusieurs outils et API: Google Catalog Search, Dogdball, Jaiku, Mashup Editor et Google Vidéo. Sur la vidéo, YouTube prend donc toute sa place et c'est un échec de Google Vidéo, mis en place avant le rachat de YouTube !

■ La grand' messe **Brainshare** de Novell n'aura pas lieu en mars prochain. C'était un lieu d'échange et de rencontres entre Novell, les partenaires et les clients. La raison de cette annulation est la réduction (très) forte de visiteurs, pour cause de restrictions budgétaires. Un contenu en ligne est cependant prévu.

NOUVEAU

ILOG Diagrammer for .NET



- **Diagrammes et tableaux de bord pour tous les types d'applications graphiques**
- **Algorithmes d'arrangement automatique de graphiques et de diagrammes**
- **Éditeurs graphiques prêts à être intégrés**
- **Intégration totale avec Microsoft Visual Studio**
- **Création d'applications riches ou Web avec Ajax**
- **Tableaux de bord avec animation en temps réel**

http://diagrammerfornet.ilog.com

Microsoft CERTIFIED Partner | ISV Software Solutions

Optimized for Microsoft Visual Studio.net

ILOG An IBM Company

Pourquoi peindre avec les doigts ?

Visualisation Java pour clients riches et Ajax

ILOG JViews 8.5, la dernière version de la suite d'outils graphiques Java d'ILOG, couvre l'ensemble des fonctionnalités de visualisation avancée.

ILOG JViews 8.5 offre :

- Des composants graphiques puissants : diagrammes, courbes, tableaux de bord, cartographie, diagrammes de Gantt
- Des services évolués : agencement automatique de graphes, affichage performant pour des jeux de données volumineux
- Plusieurs techniques de déploiement : clients riches, applications Web interactives Ajax, Eclipse/RCP et portails
- Une expertise prouvée dans les industries les plus exigeantes : Informatique, télécoms, transport, énergie et défense.

Testez un de nos produits Java dès aujourd'hui http://jviews.ilog.com

ILOG
An IBM Company

… suite de la page 14

ses propres *quick fixes* par l'écriture d'un plug-in, mais cela nécessite une connaissance approfondie des mécanismes mis en jeu, ce qui sort du cadre de cet article.

Nettoyage du code

Outre le formatage (Ctrl+Shift+F), l'indentation (Ctrl+I) ou les imports (Ctrl+Shift+O), Eclipse peut aussi corriger automatiquement votre code en faisant appel à l'action de menu « Source > Clean Up… ». Vous pouvez ainsi définir des règles de codage qu'Eclipse se chargera d'appliquer à votre code. Par exemple, vous pouvez forcer de passer par « this » pour accéder aux champs, de mettre tout le temps des accolades autour des blocs, même lorsque ce n'est pas nécessaire, de *parenthéser* toutes les expressions de manière à expliciter l'ordre des calculs, d'utiliser « final » partout où c'est possible, etc. Vous pouvez aussi demander à Eclipse de supprimer les variables, champs ou méthodes non utilisés, d'ajouter les annotations « @Override » manquantes, de supprimer les espaces superflus à la fin des lignes, etc. [Fig.5]

Encore plus pratique : vous pouvez demander à Eclipse de nettoyer automatiquement votre code, suivant les règles que vous aurez établies, à chaque fois que vous sauvegardez. Pour cela, passez par la page de préférence « Java > Editor > Save Actions ». Vous pouvez aussi choisir dans ces préférences de formater votre code à chaque sauvegarde, et de restreindre le formatage aux lignes modifiées si vous le souhaitez (afin de minimiser les différences sur le gestionnaire de versions). Une petite astuce pour le formateur : si vous avez des commentaires que vous souhaitez que le formateur ignore, commencez les par « /*- » au lieu de « /* ».

Exploration du code

Eclipse est un formidable outil de découverte de code Java. Face à un code Java inconnu, vous pouvez découvrir le type et la Javadoc des éléments en les survolant du curseur, et sauter à leur déclaration en cliquant dessus avec la touche « Ctrl » enfoncée (ou en utilisant le raccourci « F3 »). Pour revenir ensuite là où vous étiez, utilisez les flèches dans la barre d'outils, ou les raccourcis « Alt+gauche » pour revenir en arrière, « Alt+droit » pour effectuer l'opération inverse, et « Ctrl+Q » pour revenir à l'endroit de la dernière édition. Si vous arrivez sur une méthode dans une interface, utilisez la « Quick Type Hierarchy » (raccourci Ctrl+T), qui vous donnera la liste des classes implémentant l'interface. Sélectionnez une de ces classes pour sauter à la méthode implémentée dans cette classe. [Fig.6]

Avec l'option « Mark Occurrences » activée, sélectionnez un élément (variable, méthode, etc.) dans l'éditeur pour que toutes ses occurrences soient surlignées dans cet éditeur. Sélectionnez le type de retour d'une méthode pour mettre en valeur tous les endroits où la méthode peut sortir. Sélectionnez une exception déclarée dans la clause « throws » d'une méthode pour voir apparaître tous les endroits où cette exception peut être renvoyée dans la méthode. Et enfin, sélectionnez le nom d'une interface dans la clause « implements » d'une classe pour surligner toutes les méthodes implémentant cette interface dans la classe.

Pour chercher toutes les occurrences d'un élément dans le projet ou le *Workspace,* sélectionnez cet élément et utilisez l'action de menu « Search > References > Project / Workspace » (ou le raccourci Ctrl+Shift+G). Si vous voulez savoir qui appelle une méthode, sélectionnez cette méthode et utilisez l'action « Open Call Hierarchy » (raccourci Ctrl+Alt+H).

Marqueurs de tâches

Tapez « TODO », « FIXME » ou « XXX » à l'intérieur d'un commentaire Java pour créer un marqueur de tâche à cet endroit, qui apparaîtra dans la vue « Tasks » (faire Window > Show View > Tasks pour afficher cette vue). Très pratique pour noter rapidement un point à corriger sans détourner sa concentration de la tâche en cours. [Fig.7]

Raccourcis à connaître

F3	Ouvre la définition du type sélectionné. Fonctionne aussi par Ctrl+Clic sur un élément dans le code source.
Ctrl+Shift+R	Ouvre une boîte de dialogue qui permet d'ouvrir n'importe quel fichier du Workspace par son nom.
Ctrl+Shift+T	Ouvre un type (classe, interface, …) par son nom. Supporte les jokers ('?' et '*') et les noms en CamelCase.
Ctrl+O	Ouvre une vue hiérarchique du code, qui permet d'accéder rapidement à une méthode dans l'éditeur courant, en tapant son nom. Utilisé conjointement avec Ctrl+Shift+T, permet d'atteindre très rapidement n'importe quelle méthode de n'importe quelle classe.
Ctrl+L	Saute à la ligne indiquée par son numéro.
Ctrl+T	Montre l'ensemble des classes dérivées ou héritées (faire Ctrl+T à nouveau) par l'élément sélectionné. Très utile pour trouver rapidement quelle classe concrète implémente une interface donnée, ou dans quelle classe une certaine méthode d'une interface est implémentée.
Ctrl+Shift+O	Organise les imports : ajoute les imports manquants, supprime ceux qui sont inutiles. Sélectionner le projet dans le « package explorer » avant d'utiliser ce raccourci pour organiser tous les imports du projet.
Ctrl+Shift+G	Cherche toutes les occurrences de l'élément sélectionné (classe, méthode, etc.).
Alt+Shift+T	Propose une liste de refactoring applicables à l'élément sélectionné
Ctrl+Shift+F	Formate le code. Les règles de formatage peuvent être définies au niveau du projet, de manière à ce que tous les membres de l'équipe travaillant sur le gestionnaire de version (CVS, SVN, …) aient les mêmes règles.
Ctrl+3	Ouvre n'importe quelle vue, perspective ou éditeur d'Eclipse par son nom. Permet également d'exécuter des commandes.
Ctrl+Shift+L	Le méta-raccourci: il affiche une liste de tous les raccourcis existants dans Eclipse.

■ Nicolas **BROS**
Ingénieur R&D chez Mia-Software

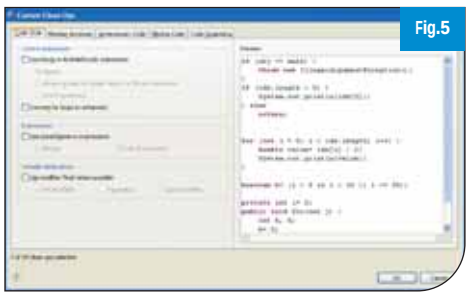


Fig.5


Fig.6

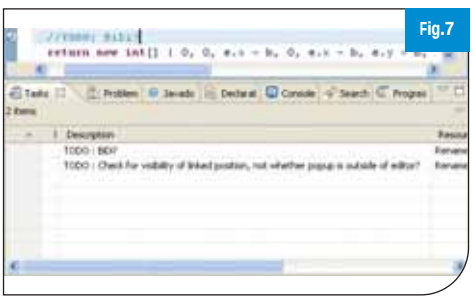


Fig.7

(3) La liste complète est disponible dans l'aide d'Eclipse, dans la section " Java Development User Guide > Reference > Available Quick Fixes ".

Python 3 : évolution en douceur

L'équipe de développement du langage Python vient d'annoncer la sortie de Python 3, dit Python 3000. Cette version ne rompt pas avec la philosophie du langage, il s'agit plus d'une version dédiée au "refactoring" : supprimer les méthodes obsolètes, réorganiser la bibliothèque standard, revoir l'API C... Python prône le fait qu'il ne devrait y avoir qu'une manière naturelle de faire une tâche. Ce grand ménage aide à tendre vers ce but. Mais ne croyez pas qu'elle n'apporte rien de nouveau ! Bien au contraire, Python 3 s'est permis de rompre la compatibilité avec Python 2 pour ajouter son lot de fonctionnalités longuement attendues !

Python est un langage libre dynamique et interprété. Il est dans la même catégorie que Perl, Ruby, PHP et javascript. Ces langages dynamiques sont simples, agréables à programmer et très productifs, mais cela se fait généralement au prix d'une vitesse d'exécution inférieure par rapport aux langages compilés tels que Java et C#.

Comparativement aux autres langages dynamiques, les points forts de Python sont la simplicité et la lisibilité. Le créateur de Python, Guido van Rossum, est parti du constat que le code source était beaucoup plus souvent lu et relu qu'écrit et que donc la lisibilité était à mettre en avant.

Cette lisibilité est l'aspect qui frappera le plus les nouveaux utilisateurs de Python et qui intéresse beaucoup les académiciens : le code est ainsi très proche de l'algorithme "pur".

Mais Python n'est pas que lisible, il s'agit aussi d'un langage très productif. Python dispose d'un éventail de bibliothèques très large. Un bon nombre est déjà inclus avec le langage ("Livré piles incluses") et beaucoup d'autres sont disponibles sur Internet sur le Python Package Index[1]. Ce grand nombre de bibliothèques provient d'une communauté très active. Et c'est grâce à ce large éventail que Python est utilisé pour des tâches très diverses allant de la programmation Web à la génération de code assembleur. Python est un langage de programmation dit "multi-paradigmes" permettant de programmer suivant plusieurs styles (impératif, fonctionnel, etc.), mais qui favorise principalement la programmation orientée objet.

LES NOUVEAUTÉS DE PYTHON 3

Internationalisation et encodages

Python 3 permet une meilleure internationalisation grâce au passage à l'unicode pour les chaînes de caractères (prévu dans la version 6 de PHP). Jusqu'ici, les chaînes de caractères étaient des listes d'octets : chaque caractère correspondant à un octet. Cette façon de faire n'est plus adaptée aux encodages évolués qui n'ont pas une longueur fixe de 8 bits par caractère (ex: utf-8).

Pour cela, Python disposait déjà de chaînes de caractères spécifiques, dites chaînes unicode, qui permettent de s'abstraire de l'encodage de la chaîne et de la manipuler plus facilement, mais au prix d'encodages (d'unicode vers l'encodage) et de décodages (vers unicode) successifs qui peuvent produire des erreurs si l'encodage est mal spécifié.

Par exemple, en Python 2 il fallait écrire u"Jérôme" pour écrire une chaîne unicode et simplement "Jérôme" pour sa contrepartie chaîne simple. L'accent au deuxième caractère s'exprime par des valeurs différentes suivant l'encodage utilisé.

En unicode, la manipulation ne pose pas de problème, mais si c'est une chaîne simple, le développeur doit connaître à l'avance l'encodage utilisé. De même, l'interpréteur doit connaître quel est l'encodage utilisé pour interpréter correctement les caractères non ASCII. En Python 2, on le précise via une balise au début du fichier.

Sans cette balise, l'encodage est ASCII, qui n'autorise aucun caractère spécial (accents, etc.).

En Python 3 plus de soucis, toutes les chaînes de caractères sont par défaut en unicode et l'encodage du fichier est par défaut en utf-8.

Formatage des chaînes de caractères

Python 3 introduit une nouvelle façon de formater les chaînes de caractères. Jusqu'à présent, Python nous offrait de base soit l'opérateur " % " sur les chaînes de caractères, soit la classe " string.Template ", d'un fonctionnement un peu équivalent au nouveau système de formatage, mais un peu complexe à utiliser. L'opérateur " % " est un opérateur binaire : il ne peut prendre que deux paramètres. Pas un de plus. Le premier paramètre étant la chaîne formatée (avec indications rudimentaires sur le format de la valeur), l'ensemble des valeurs à remplacer doit être placé dans le deuxième paramètre, au moyen d'un tuple ou d'un dictionnaire. Il y a donc un choix à faire entre arguments positionnels (avec un tuple) ou arguments nommés (avec un dictionnaire), sans possibilité de mixage.

La classe " string.Template " a été prévue pour être utilisée avec une phase de préparation.

En outre, elle ne permet pas de préciser un formatage propre à chaque

champ, par exemple la précision d'un réel. Python 3 introduit une méthode " format " disponible sur les chaînes de caractères. Cette méthode utilise son instance comme chaîne de formatage et ses arguments comme valeurs à remplacer. On dispose donc de la puissance du passage des paramètres en Python. De plus, ce nouveau mécanisme permet de personnaliser les options de formatage par type de valeur à remplacer.

```
"Bonjour {nom}, il fait {0:.2f} °C dehors.".format(-3.5, nom="Emmanuel")
'Bonjour Emmanuel, il fait -3.50 °C dehors.'
```

Les balises de formatages embarquent tout un petit langage permettant de préciser très finement la façon dont la substitution se fera. C'est un réel progrès par rapport aux mécanismes de formatages précédents.

Print est une fonction

" print " était jusqu'alors un mot clé, instruction du langage. C'est une anomalie, puisque les mots clés servent, dans les langages modernes, à organiser ces instructions. "print" a donc été transformée en fonction classique.

```
# En Python 2
print 'Hello world !'
```

```
# En Python 3
print('Hello world !')
```

Itérateurs

L'utilisation des itérateurs est maintenant généralisée dans Python 3. Par exemple range() qui renvoyait une liste, retourne maintenant un itérateur. D'autres fonctions au comportement équivalent ont également été modifiées. Il en est de même pour les méthodes de la classe "dict" (keys(), values(), items()).
Le passage aux itérateurs permet d'accélérer l'exécution du code puisqu'il est possible de parcourir les éléments d'un itérateur sans avoir besoin d'allouer de la mémoire. Cela rend les mécanismes de cache particulièrement efficaces.

Migration des bibliothèques

En rompant la compatibilité avec les versions 2, Python 3 se coupe temporairement de toutes les bibliothèques tierce partie écrites pour Python 2. Bibliothèques qui font la force de Python. Un outil appelé "2to3"[2], a été développé pour faciliter la migration des bibliothèques de Python 2 à Python 3. Malgré cet outil, porter une bibliothèque reste un travail relativement fastidieux.

Performances

Comme toujours, dans le cadre d'une première release majeure, les fonctionnalités et la stabilité ont été privilégiées par rapport aux performances. Du fait de certaines généralisations, Python 3.0 est, en moyenne, 10% moins performante que Python 2.6. Mais l'on peut s'attendre à un renversement de la balance avec les prochaines versions de Python 3 : la communauté Python étant connue pour être capable d'optimiser son code une fois que celui-ci est stabilisé. Néanmoins, Python 3 reste tout de même plus rapide que Perl, PHP et Ruby.[3]

CONCLUSION

Python 3 est une réalité, le langage a été épuré et amélioré pour lui permettre de relever les défis du futur. L'étendue des nouvelles fonctionnalités est bien plus vaste que ce que nous permet la longueur de cet article mais des ressources Web vous permettront d'aller plus loin [4].

[1] http://pypi.Python.org/pypi
[2] http://docs.Python.org/library/2to3.html
[3] http://shootout.alioth.debian.org/gp4/benchmark.php?test=all&lang=all
[4] http://docs.Python.org/3.0/whatsnew/3.0.html

■ Olivier **Lauzanne**
Ingénieur Python de la société Ingeniweb du groupe Alterway, diplômé de l'Ecole Centrale de Lyon, auteur de la bibliothèque pyquery et membre actif de l'association francophone de Python (www.afpy.org).

■ Emmanuel **Coirier**
Ingénieur et Formateur Python de la société Ingeniweb du groupe Alterway, diplômé de l'Institut Galilée et membre actif de l'association francophone de Python (www.afpy.org).

AVANT-PREMIÈRE

Biztalk Server 2009
l'architecture orientée développeur

1re partie

© François Cointe " Architectures Orientées Sevices "

Aujourd'hui, la sortie de Biztalk Server 2009 permet de mettre le focus sur la solution d'intégration et d'architecture orientée services (SOA) de Microsoft. Outil discret, Biztalk ne démérite pas et sait se rendre utile dans les entreprises. Mais, jusqu'à présent, Biztalk souffrait d'un manque d'intégration .Net et surtout d'une visibilité inexistante côté développeur. Cette version 2009 corrige le tir et mise plus que jamais sur le développeur. En avant-première, nous vous proposons une plongée dans Biztalk Server 2009 avec ce dossier en deux parties.

Le développeur ne s'intéresse pas toujours aux questions d'architecture, surtout s'il ne travaille pas directement pour, ou dans une entreprise utilisant une SOA ou un environnement d'intégration. Or, aujourd'hui, avec le tout service, le développeur doit de plus en plus connaître et comprendre les architectures de services, comment casser les applications monolithiques en différents services. Tout cela préfigure la migration suivante vers le *cloud computing* qui consomme avant tout des services, des web services, de la SOA ! Et Biztalk Server 2009 peut les aider à préparer cette prochaine étape.

Plus prosaïquement, Biztalk Server 2009 apporte une intégration profonde et complète avec .Net et Visual Studio 2008. Ainsi, bien que la conception Biztalk soit très graphique, le développeur garde toujours le contrôle et une vision du code. On accède, en vrac : à un debugger XSLT, aux tests unitaires, à du build automatique. On bénéficie aussi d'une intégration avec Team Foundation pour les projets en équipe et afin de gérer le code source, et aussi la disponibilité d'un bug tracking et du support de MS Project Server.

Cette version 2009 peut se résumer à 4 piliers : la plate-forme, productivité pour le développeur et les équipes, l'intégration B2B et SOA / Web Services. Biztalk se met à jour par rapport aux nouveaux outils serveurs de Microsoft. Il supporte désormais Windows Server 2008 et sait fonctionner en environnement virtualisé avec Hyper-V. Côté référentiel et données, SQL Server 2008 fait son entrée. Cette compatibilité n'est pas innocente sur les performances, notamment en 64-bits ou encore sur les capacités fonctionnelles du BAM.

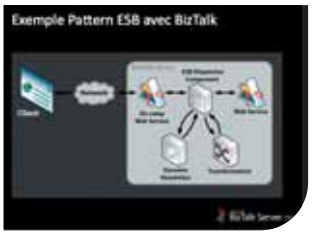
Mais n'oublions pas que Biztalk Server est aussi, et surtout, un centre SOA. Et la principale nouveauté à retenir est l'apparition de UDDI 3.0, l'annuaire de services. La découverte de nouveaux services constitue toujours un défi avec une architecture de services et Biztalk tente ici de proposer une solution standard pour améliorer la découverte et l'utilisation de ces services. Là aussi, il y a du travail pour les développeurs, notamment avec les nouveaux SDK et les adaptateurs. Mais la v2009 sait aussi s'ouvrir aux concurrents, en particulier avec les solutions WebSphere et DB2.

Demain, Biztalk promet encore plus, notamment avec la couche serveur Dublin qui s'annonce comme une révolution dans les services serveurs, la montée en charge dans le monde Windows. Sans oublier Visual Studio 2010, .Net 4 ou encore la nouvelle génération de Business Intelligence : Gemini.

■ François **Tonic**

Les nouveautés de BizTalk 2009

Les apports de BizTalk 2009 se résument à quatre catégories : alignement sur la plate-forme 2008 pour une performance optimale, amélioration de la productivité du développement grâce à une compatibilité Visual 2008 et Team Server, rapprochement du concept SOA et extension de l'offre B2B.



BizTalk 2009 permettra de tirer parti des dernières versions des produits d'infrastructure de Microsoft : Windows Server 2008, SQL Server 2008, SharePoint, et Hyper-V. Pour les applications mission critiques, BizTalk 2009 sera encore plus performant grâce aux améliorations apportées à SQL 2008, et les configurations Hyper V apportent d'énormes avantages pour la gestion des serveurs applicatifs sous x64. Hyper V va simplifier le processus "développement – recettes - mise en production" grâce à un environnent virtuel permettant d'avoir une configuration identique pour ces plates-formes. Important pour la tolérance de panne, le serveur Windows 2008 offre à BizTalk la possibilité d'être mis en cluster à travers des sous-réseaux portant des adresses IP différentes. Jusqu'à maintenant, il fallait mettre en place un VLAN complexe pour avoir ce genre de configuration.

Cycle de vie

BizTalk dans VS 2008 / TFS profite du suivi de projet grâce à l' " Application Lifecycle Management (ALM) support ". Avec cette fonctionnalité, les développeurs peuvent s'appuyer sur les composants VSTS de " bug tracking ", contrôle du code, intégration de " project " serveur, et suivi d'équipe. La structure des projets BizTalk a également changé avec la version 2009. Tout d'abord, ils sont générés en tant que projet C# avec un héritage d'interfaces .Net spécifique. Un projet BizTalk peut donc maintenant contenir d'autres types d'artefacts (classes C#, par exemple). Un fichier assemby.cs est également ajouté au projet afin de rassembler les métadonnées liées à une dll BizTalk. Les développeurs vont apprécier une nouveauté : l'ajout d'un type de projet " test " qui permet de simuler le fonctionnement du flux à l'intérieur de l'environnement VS 2008. Ces projets de test permettent aux développeurs, par exemple, de suivre ligne par ligne l'exécution du code XSLT des maps BizTalk. Les pipelines peuvent également être débogués de la même manière, offrant même la possibilité d'exécuter un flux de bout en bout sans quitter Visual Studio. Les projets BizTalk C# s'intègrent avec MSBUILD afin de synchroniser tout ce qui est BizTalk et artefacts .Net dans un seul processus de recompilation et déploiement. BizTalk 2009 est compatible .Net 3.5., ce qui veut dire que la partie SOA sous WCF permettra d'intégrer les protocoles très modernes - REST, POX, RSS, ATOM, WS-* à l'intérieur de BizTalk.

Le noyau BizTalk se rapproche de plus en plus du concept du SOA. Il étoffe l'intégration grâce à un couplage avec l'annuaire des web services UDDI 3.0, une couche ESB version 2.0, et l'addition de nouveaux adaptateurs WCF. L'intégration UDDI permet de contrôler et de configurer les services SOA indépendamment de BizTalk. BizTalk 2009 fournit deux nouveaux adaptateurs WCF - SQL 2008 et Oracle E-Business Suite en vue d'une intégration complète avec la plate-forme Oracle. De nombreuses anomalies dans le pack adaptateur v1.0 ont été corrigées comme, par exemple, le problème de " timeout " sur une connexion RFC SAP en cas de non réception de l'IDOC. L'architecture ESB intéresse beaucoup d'entreprises pour sa capacité à fédérer l'ensemble de services SOA et à renforcer un grand principe du SOA : " l'agilité ". Les messages traversant un ESB suivent un itinéraire qui détermine l'ensemble des services, orchestrations, et transformations qui agissent sur un message. Un ESB permet d'attribuer des itinéraires dynamiques selon des règles métier. BizTalk est bien adapté à cette architecture et c'est la raison pour laquelle Microsoft a annoncé la sortie de l'ESB 2.0 avec celle de BizTalk 2009. L'ESB 2.0 fournit un langage (Visual Domain Specific Language) pour modéliser des itinéraires. Il consiste en un pack adaptateur " pluggable " qui sert comme point d'entrée / sortie de l'ESB. Un portail consacré à la gestion de l'ESB est également fourni. Finalement, il y a des modifications à la console de l'Administration BizTalk. L'HAT n'est plus disponible mais un administrateur / développeur sera en mesure d'effectuer des requêtes sur les données du " tracking " depuis la console d'Admin. Un deuxième article comprendra les aspects BAM SQL 2008, B2B, RFID, Host Integration Services et modélisation sous BizTalk et Oslo.

Dave **Nolting**
CODit / MS BizTalk Squad
Architect SOA & BPM
dave.nolting@codit.eu

Débuter avec BizTalk Server 2009

L'objectif de cet article est de vous permettre, au travers de la création en pas à pas d'une petite application BizTalk Server 2009, d'avoir un premier aperçu sur les étapes du développement de ce type de projet.

Afin de pouvoir commencer, il est tout d'abord nécessaire de disposer d'un environnement avec tous les outils et bien sûr, BizTalk. Pour cela, trois solutions sont envisageables :

- Configurer de A à Z un environnement (ceci est très formateur mais prend du temps)
- Partir d'un environnement où tous les principaux outils sont déjà présents et ensuite installer BizTalk

Récupérer l'environnement déjà configuré, sur notre site, vous trouverez la VPC (machine virtuelle) contenant le serveur BizTalk. La procédure d'installation y est décrite ainsi que la configuration de l'environnement.

Une fois cet environnement BizTalk configuré,nous sommes fin prêts à démarrer.

Présentation du scénario projet

Notre but sera de construire un service de gestion de réapprovisionnement de stock. Un service reçoit des demandes de réapprovisionnement sous la forme d'un code produit associé à une quantité (pour des raisons de simplification, la réception se fera par fichiers déposés dans un répertoire). Le service traite les demandes de manière différente en fonction de la quantité :

- Les petites quantités (inférieures à 20) sont mises en attente avant d'être traitées et les autres sont traitées immédiatement

Détails d'implémentation :

- L'application gérant les commandes fournisseur prendra la forme d'une base de données SQL Server contenant une table que nous alimenterons avec les demandes.

Implémentation - initialisation du projet BizTalk

La première étape de la réalisation consiste dans la création et la configuration du projet BizTalk dans Visual Studio 2008 : Dans Visual Studio nous choisissons le modèle de projet ***" Empty BizTalk ServerProject "*** puis nous devons configurer les propriétés du projet. La configuration consiste à signer le projet et à rentrer les paramètres de déploiement BizTalk dans les propriétés du projet.

Il nous faut assigner une clé de signature dans la section ***" signing "*** et définir l'application BizTalk dans laquelle sera déployé notre projet au travers de la section ***" Deployment "*** (NB : une application BizTalk est un conteneur logique qui permet de regrouper des schémas, orchestrations, ports ...) : [Fig.1]. A cette étape, notre projet est initialisé et nous pouvons continuer avec la suite du développement.

Implémentation-définition des schémas

La première étape dans un projet BizTalk est systématiquement de définir les formats des messages échangés. Cela se fait par la création de schémas. Dans notre cas nous avons deux types de schémas à définir :

- Le schéma de la demande de réapprovisionnement
- Les schémas permettant d'intégrer cette demande dans la base SQL Server.

Nous allons donc tout d'abord rajouter un premier schéma pour la demande. Ceci s'effectue par l'ajout d'un élément de type schéma dans notre projet (*Clic droit sur le projet – **" Add new Item / Schema File "***) .

Dans l'éditeur de schéma, on peut renommer l'élément racine, puis lui rajouter des éléments enfants pour représenter le code produit et la quantité demandée. (Cela s'effectue dans le menu contextuel de l'élément racine par ***" Insert Schema node / Child Field Element ".*** Il nous est aussi possible de préciser le type de l'élément quantité (xs :int) au travers de la fenêtre de propriété. [Fig.2]

Bien que nous ayons fini de définir la structure de ces schémas, il nous reste une opération de configuration à effectuer. Afin de faciliter l'accès à la quantité depuis l'orchestration, nous allons faire ce que l'on appelle une " distinction " de champ.

Ceci s'effectue en choisissant dans le menu contextuel l'option ***" Promote / Show Promotions... "*** : [Fig.3]

Dans l'écran des champs distingués, il faut sélectionner le champ *quantité* et cliquer sur le bouton ***" Add "*** : [Fig.4]

Notre schéma est maintenant finalisé !

Implémentation-ajout du schéma pour l'accès à la base de données

Nous devons d'abord créer une base de données ***(Stock)*** qui contiendra une table ***(Demandes)*** pour stocker les demandes de réapprovisionne-

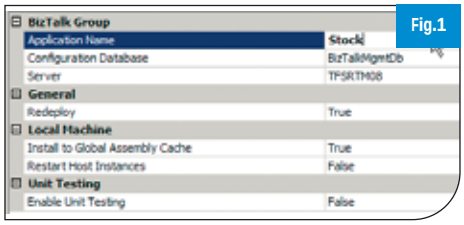


Fig.1

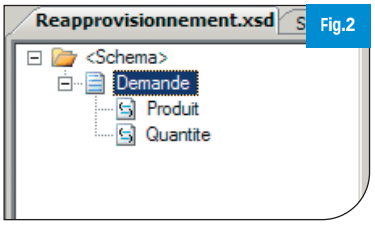


Fig.2

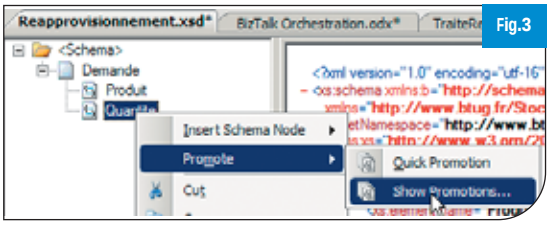
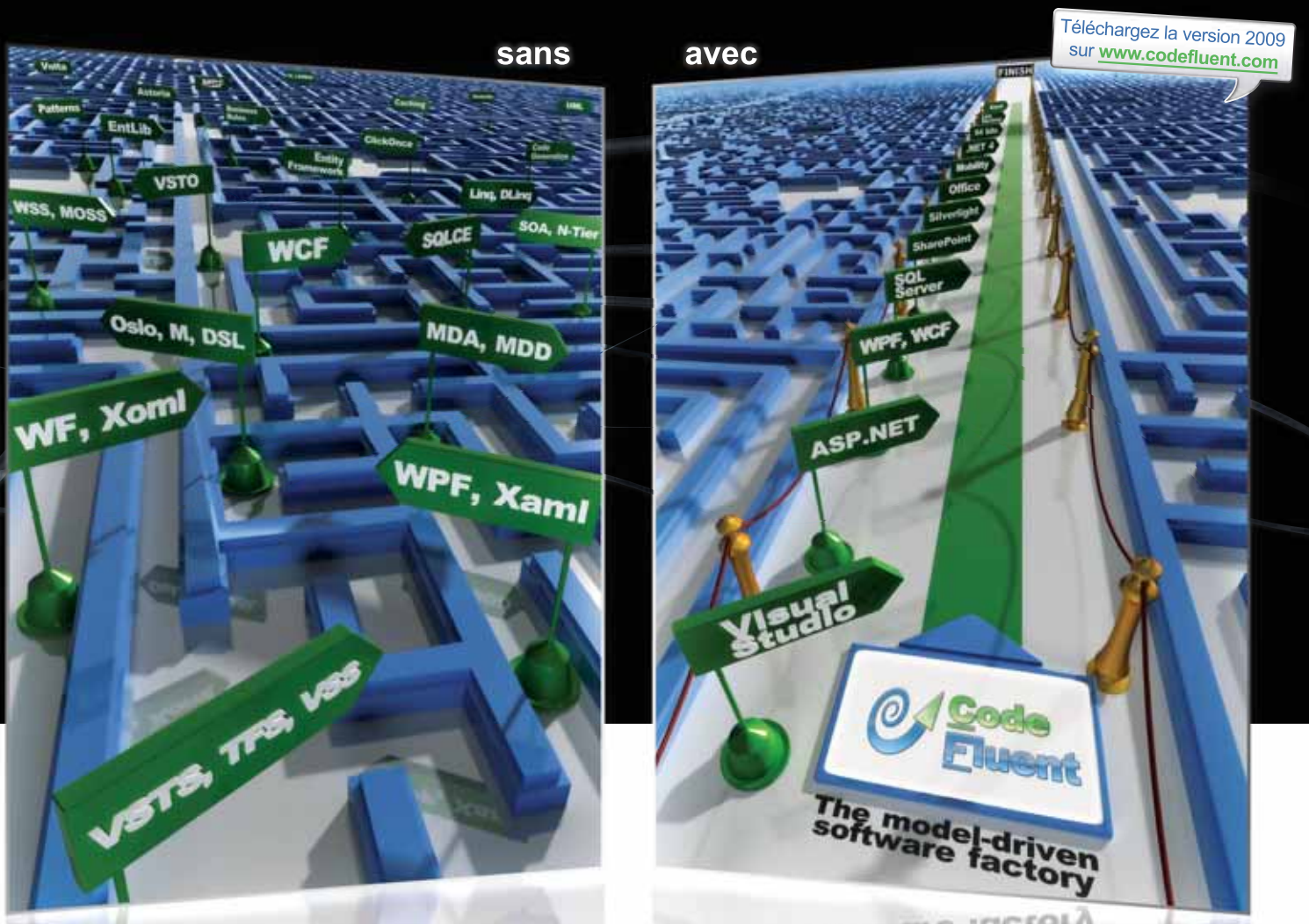


Fig.3

CODEFLUENT

La première fabrique logicielle .NET entièrement pilotée par les modèles



CodeFluent est un produit de génie logiciel qui permet d'industrialiser la fabrication d'applications professionnelles manipulant des données sur la plate-forme .NET en **automatisant la création des composants** à partir d'une modélisation de votre métier.

L'utilisation de CodeFluent vous assure **évolutivité**, **productivité**, **qualité** et **facilité de maintenance** en évitant les risques associés à une mise en œuvre expérimentale des nouvelles technologies.

Le plus court chemin vers les technologies .NET

SoftFluent
46 rue Auguste Blanqui - 94250 Gentilly
01 46 16 05 02 - sales@softfluent.com

ment. Dans SQL Server Management Studio nous ajoutons une base de données Stock, puis nous créons une table ***" Demandes "*** qui contiendra deux champs : un champ produit de type *varchar* et un champ quantité de type *int* : [Fig.5]

Nous allons maintenant, dans Visual Studio, utiliser l'adapteur SQL pour générer un schéma nous permettant d'effectuer des insertions dans la base de données. Au niveau de notre projet BizTalk, il faut choisir, dans le menu contextuel, l'option ***" Add / Add Generated Items "*** : [Fig.6]

Nous choisissons ensuite ***" Add Adapter Metadata "*** et nous sélectionnons l'adapteur ***" WCF-SQL "*** [Fig.7]

Nous arrivons alors dans un écran qui va nous permettre de parcourir les objets de la base de données et de générer des schémas pour appeler une procédure stockée ou encore insérer, modifier, supprimer dans une table ... Pour pouvoir accéder à notre base il faut tout d'abord rentrer l'URI suivante : ***mssql: / / . / / Stock?*** Ensuite on clique sur le bouton ***" Connect "*** et on a accès aux objets : [Fig.8]. On choisit alors la table Demandes, puis les opérations associées apparaissent : On choisit l'opération ***" Insert "*** puis on clique sur ***" Add "*** : [Fig.9]

Après validation, les schémas nécessaires ont été générés.

Implémentation-Création de l'orchestration

Nous allons créer une orchestration, qui aura la charge de recevoir les demandes d'approvisionnement, éventuellement de les mettre en attente, puis de les transmettre à la base de données. Dans notre projet nous rajoutons donc une nouvelle orchestration ***(Add / New Item / BizTalk Orchestration).*** A ce stade, il est nécessaire de faire une pause pour prendre le temps de la réflexion sur l'implémentation de l'orchestration.

Nous désirons maintenant traiter différemment les demandes en fonction de leur quantité et imposer un délai aux plus petites. Notre solution contiendra donc une première réception suivie d'une prise de décision sur l'application d'un délai avant envoi dans la base. Le problème étant posé, voyons l'implémentation.

Dans BizTalk le développement d'une orchestration se fait principalement au travers de deux outils :

- L'éditeur (Designer) visuel d'orchestration dans lequel on construit la logique du processus en assemblant des éléments pris dans la boîte à outils. [Fig.10]
- La fenêtre " Orchestration View " qui nous permet de définir des variables, des messages ... [Fig.11]

En premier, nous allons créer des variables pour contenir les messages reçus et émis par l'orchestration. Cela se fera dans l'orchestration view, au niveau du dossier " Messages " par le menu contextuel ***(" New Message ").*** Il faut ensuite donner un nom au message et sélectionner son type dans les propriétés (il suffit de naviguer vers le bon schéma) : [Fig.12].

On crée ainsi trois messages :

- Un message de demande de réapprovisionnement (msgReappro)
- Un message d'enregistrement dans la base (msgSaveReappro de type Stock.Reappro.TableOp_dbo_Demandes_Insert_InputMessage que l'on trouve dans les types de messages multi-part).
- Et enfin un message pour la réponse à l'insertion dans la base de données (msgReponseDb) dont le type se trouve aussi dans les types " Multi-Part "

Communiquer : création des ports

Pour pouvoir communiquer, notre orchestration a besoin de points d'entrée et de sortie : il s'agit de ce que


Fig.4

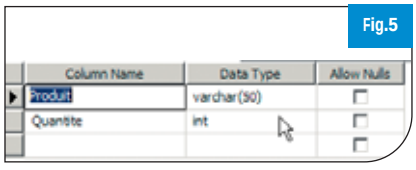


Fig.5

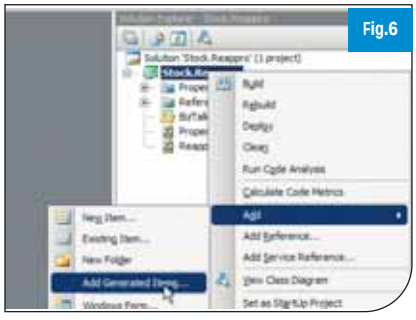


Fig.6

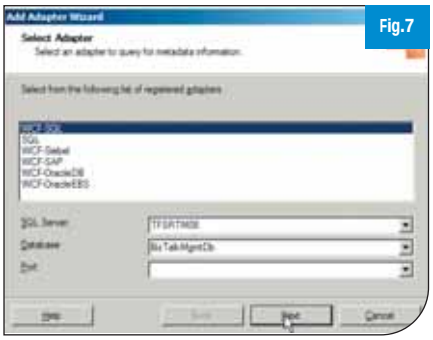


Fig.7

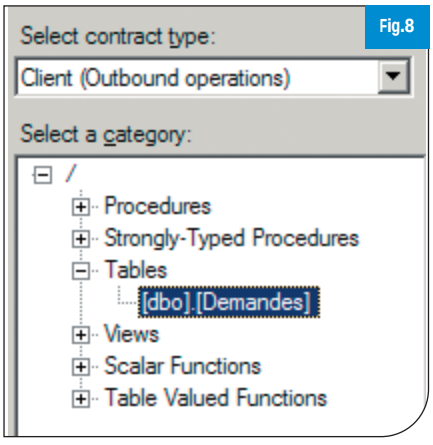


Fig.8

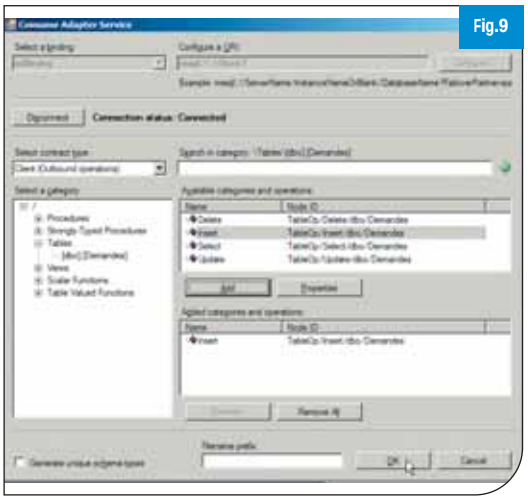


Fig.9


Fig.10

l'on appelle des ports logiques. On peut les créer directement depuis l'orchestration view (Un port de réception et un port d'émission) : [Fig.13].

Les points d'exclamation nous signalent que le type des messages véhiculés par les ports n'est pas précisé : Il faut le sélectionner dans la fenêtre de propriété. Dans le cas du port OutDB nous en avons déjà un, généré précédemment par l'adapteur SQL : [Fig.14].

Pour le port *InDemandes*, il nous faut en créer un (option " Create One Way Port Type ") et préciser le type de message (ceci se fait au niveau des propriétés du paramètre Request du port) : [Fig.15].

Au niveau du port de sortie *OutDb* il nous faut également préciser son sens de communication (par défaut les ports sont en réception) au niveau de la propriété ***" Communication Direction "*** de manière à configurer le port en sortie.

A présent, nous pouvons " construire " notre orchestration en déposant des éléments depuis la boîte à outils. On déposera ainsi comme suit : Une réception, une décision avec un délai dans une des branches, un envoi et une réception que l'on prendra soin de renommer : [Fig.16].

Maintenant, la structure est posée il ne reste qu'à configurer chaque élément. Au niveau des réceptions et des envois il faut sélectionner le message qui est reçu ou envoyé. De plus, sur la première réception il faut mettre la propriété ***" activate "*** à ***" true "*** pour indiquer que la réception démarre l'orchestration.

Une fois ceci fait, il est possible de relier les réceptions et les envois à leurs ports respectifs : [Fig.17].

Il nous reste à paramétrer le test sur la quantité dans l'élément ***" Decide ".*** Cela se fait en double-cliquant et en rentrant une expression à évaluer. Dans notre cas: ***msgReappro.Quantite<20.*** Pour le délai, la procédure est identique. Nous rentrons l'expression suivante :

```
new System.TimeSpan(0,3,0);
```

Dernier point : pour pouvoir envoyer dans la base de données, il est nécessaire de fabriquer un message au bon format. Nous allons faire cela grâce à une transformation. Juste avant l'envoi vers la base, on dépose une brique de transformation. Cela crée en fait deux briques qu'il faudra paramétrer (une construction de message et une transformation) : [Fig.18].

Dans la première brique, il faut préciser le message que l'on construit ***(msgSaveReappro)*** dans la deuxième, nous allons préciser en message source la demande et en message destination ***msgSaveReappro.*** [Fig.19].

A la validation, l'éditeur de transformation se lancera automatiquement. Dans celui-ci, il faut alors simplement relier les bons champs : [Fig.20].

Notre développement est fini, nous pouvons déployer le projet dans BizTalk à l'aide de la commande ***" Deploy "*** du menu contextuel da la solution.

Configuration et test

Nous pouvons quitter Visual Studio. Il faut maintenant connecter notre orchestration à des points d'entrée et de sortie " physique " (fichier et base de données). Cette opération se fait dans la console d'administration de BizTalk (accessible dans le menu

Fig.11

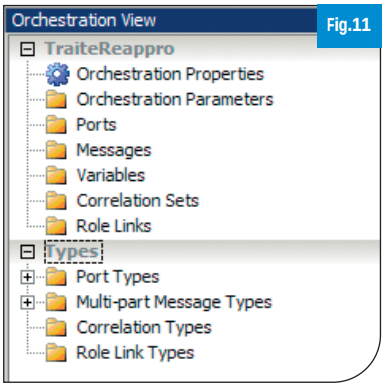


Fig.12

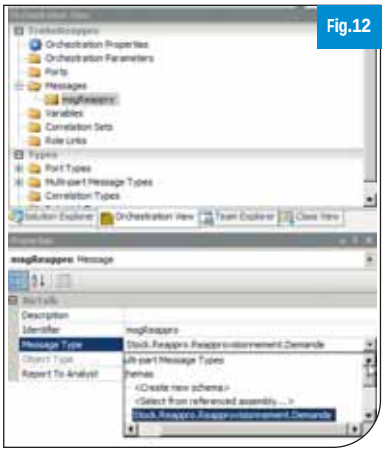


Fig.13

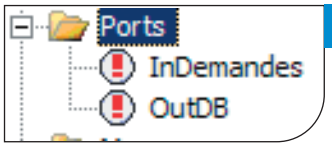


Fig.14

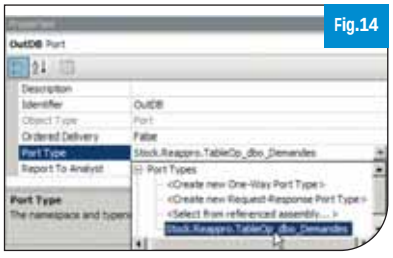


Fig.15

Fig.16

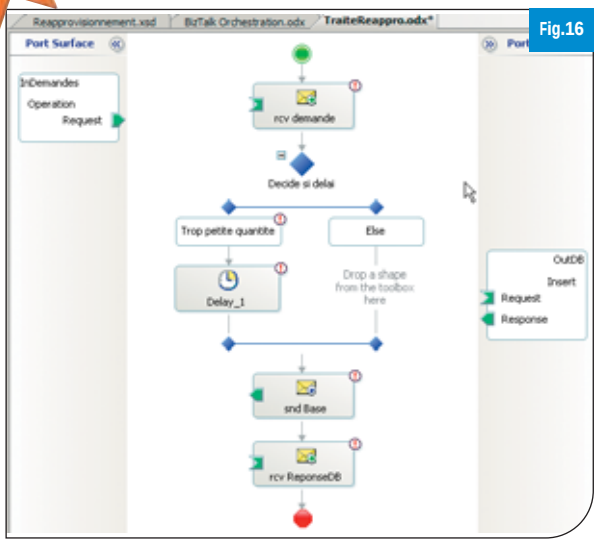
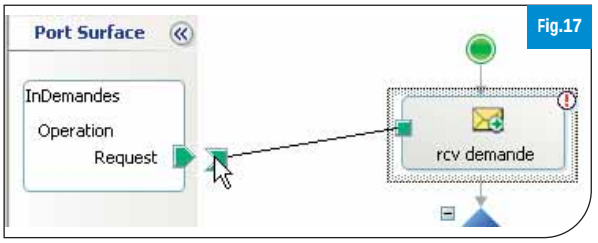


Fig.17



démarrer / Microsoft BizTalk 2009 / Microsoft BizTalk Administration). Quand on déploie l'arborescence de la console, on constate qu'une application ***" Stock "*** apparaît : [Fig.21].

Dans cette application nous allons créer un port de réception (dans le dossier *" Receive Ports "* -> ***Menu contextuel*** -> ***New*** -> ***One Way Receive Port).*** Nous lui donnons un nom, puis nous allons tout de suite sur ***" Receive Location "*** et nous sélectionnons ***" new "*** : [Fig.22].

Nous sélectionnons l'adapteur ***File*** et le pipeline ***XMLReceive*** : [Fig.23].

Puis sélectionner ***" Configure "*** pour rentrer les paramètres de réception (emplacement du répertoire). Après validation, nous avons notre port de réception. Nous devons maintenant créer un port d'envoi. Cette fois ci nous choisissons ***" New -> Static Solicit Response Send Port "***. Il faut ensuite choisir comme type de transport ***" WCF-Custom "*** et comme pipeline de réception ***" XmlReceive".*** Ensuite nous sélectionnons le bouton configure. [Fig.24].

Dans le paramètre URI, il nous faut saisir : **mssql://./Stock.** Et dans Actions : **TableOp/Insert/dbo/Demandes**

Ensuite sur le deuxième onglet (Binding) nous sélectionnons **sqlBinding** : [Fig.25].

Après validation, notre port est créé.

Nous allons maintenant configurer l'orchestration. Pour cela il faut aller dans le dossier ***Orchestration*** et double-cliquer sur celle-ci. On va alors sur ***" bindings "*** et on sélectionne les paramètres : [Fig.26].

Tout est configuré, nous pouvons démarrer notre application en sélectionnant ***" start "*** dans son menu. Pour tester, il suffit de déposer des fichiers de tests dans le répertoire d'entrée et d'observer la table dans la base.

Conclusion

Nous avons vu la réalisation en pas à pas d'une petite application BizTalk. Ce parcours nous aura permis d'apercevoir les schémas, transformations, orchestrations et adapteurs. Le code source du projet est disponible en téléchargement, donc n'hésitez pas à expérimenter.

■ Roch **Baduel** –MVP BizTalk
Responsable pôle Intégration / SOA (MCNEXT)
rbaduel@mcnext.com

Fig.18

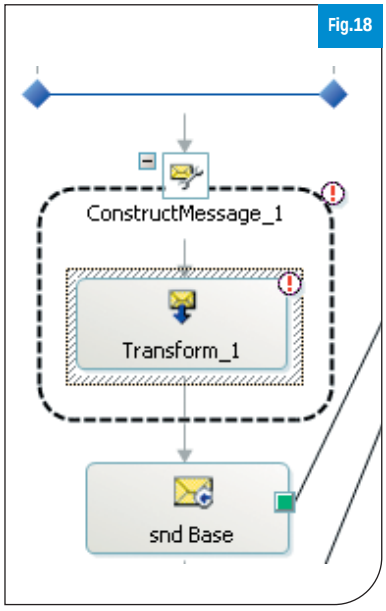


Fig.19

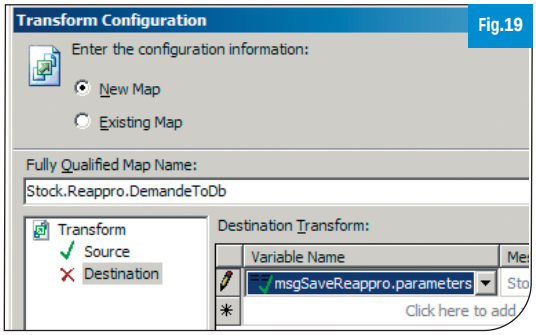


Fig.20

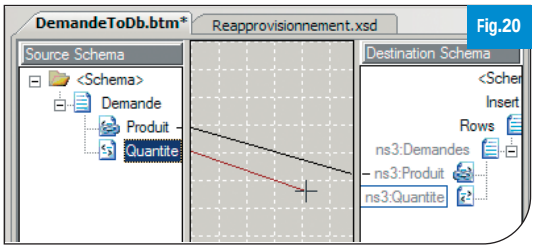


Fig.21

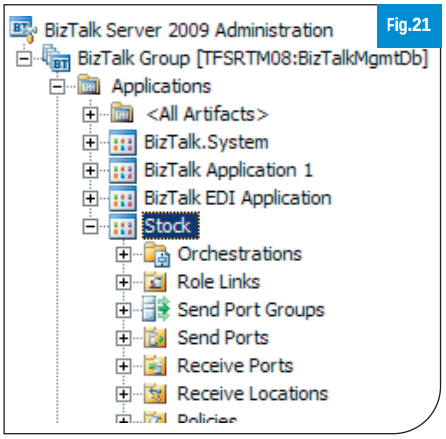


Fig.22

Fig.23

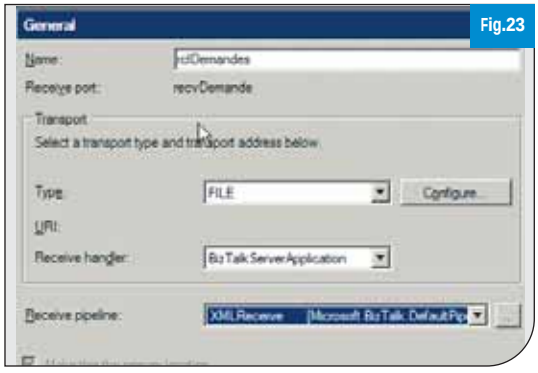


Fig.24

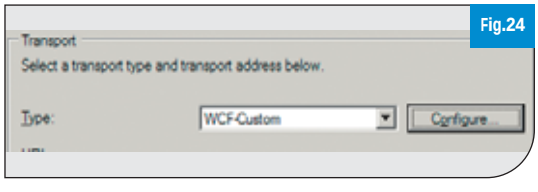


Fig.25

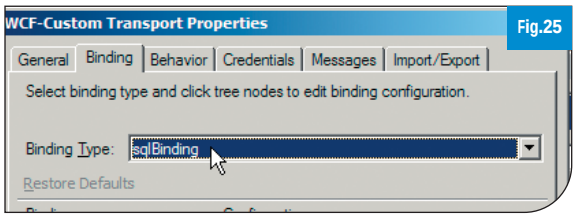
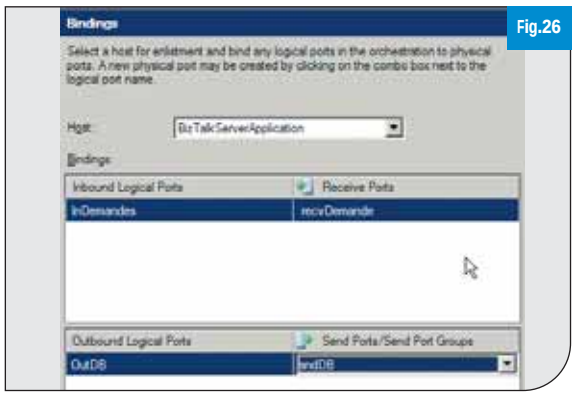


Fig.26



Suite du Spécial BizTalk le mois prochain

LEONARDI

W4-LYRIA : la modélisation au coeur des applications métier

L'ihm

EN TOUTE SIMPLICITÉ !

Pour implémenter des applications riches, accédant à des données hétérogènes (applicatifs métier, BD, et depuis la V4.1, Hibernate, GED et silos documentaires), déployées en Swing, plugin Eclipse ou DHTML/Ajax et s'insérant naturellement dans les processus d'entreprise...

Adoptez la puissance et l'agilité de l'approche Model-Driven

NOUVELLE VERSION LEONARDI V4.2 open source

Concentrez-vous sur votre métier et dotez votre entreprise d'avantages compétitifs durables: amélioration du cycle de vie du logiciel, démarche itérative par prototypage pour coller aux besoins, découplage technologie/métier, évolutivité, agilité et évolutivité, le tout sans expertise technique pointue !

LYRIA

Pour en savoir plus sur la solution Leonardi, rendez-vous sur notre site www.lyria.com ou envoyez-nous un courriel à info@lyria.com

Composants

Simplifiez-vous le code !

Ne pas réinventer la roue et pouvoir réutiliser le code, des propos qui reviennent souvent chez les développeurs. Dans la pratique, le composant, ce petit contrôle que l'on rajoute dans l'environnement de développement, rend bien des services. Pourquoi perdre du temps à coder un grid ou un graph si cela existe déjà et permet d'être plus complet, plus stable, plus pratique à manipuler ? Quand on utilise massivement Ajax, javascript, on est heureux de pouvoir se délester d'une partie du code sur des contrôles prêts à l'emploi !

D'un autre côté, le développeur peut se dire, le composant, *oui, mais je ne fais pas un graph dynamique, ni un agenda tous les jours.* En effet, mais il ne faut pas oublier que les composants proposent une gamme de fonctionnalités, extrêmement large : du contrôle d'interface 3D et dynamique à la génération d'un fichier PDF à la volée avec paramétrage personnalisé. Bref, n'oubliez pas que le composant ne se limite pas aux seules interfaces même si aujourd'hui, ce sont les plus visibles, les plus spectaculaires.

On trouve des composants pour tout, pour tous les langages ou presque. Aujourd'hui, les contrôles clés en main pour Ajax, .Net, Java sont les plus connus, mais les composants PHP, Flex, notamment, se répandent. Autre exemple, dans le développement web, il existe aussi de nombreux composants pour les plates-formes de blogs ou des solutions de forum.

Bien entendu, il ne faut pas choisir un composant les yeux fermés. Le développeur doit déjà savoir ce qu'il veut puis établir une liste de packages composants. Ensuite, les choses sérieuses débutent : les tests. Impossible de choisir un composant sans le tester ! Plusieurs éléments sont à considérer dans le choix qu'on fera: les licences des composants, les critères de redistribution, le nombre de composants dans le package, le langage supporté, la librairie graphique supportée, la disponibilité du code source du composant, sa popularité, l'efficacité du support et de la communauté, sans oublier les mises à jour régulières. Bref, tout ce qui concerne la pérennité du composant doit être votre préoccupation. Car si vous optez pour un composant sans accès au code source et que l'éditeur ferme 3 mois plus tard, que ferez-vous ? Il ne faut pas oublier que le composant se comporte souvent comme une boîte noire et en cas de bugs, comment faire pour les contourner ? Malheureusement, on ne s'en aperçoit que lorsque le développement est déjà avancé. D'autre part, n'oubliez pas la politique de redistribution du composant, surtout si votre application est distribuée à l'extérieur. Faut-il payer une licence ?

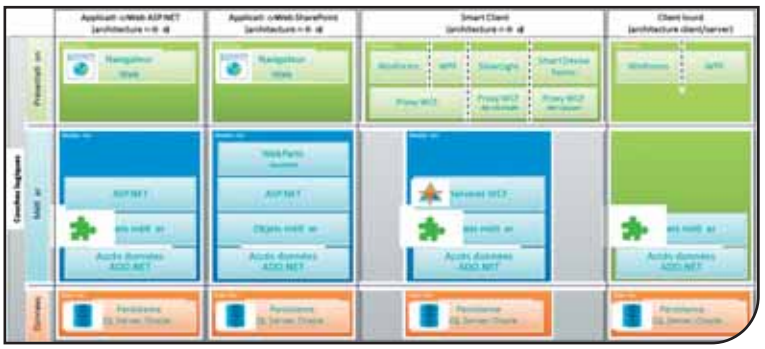
Dans ce dossier nous allons aborder les composants pour PHP, Flex et .Net. Nous verrons aussi les bonnes pratiques à adopter, les précautions à prendre. Nous verrons pourquoi des développeurs les utilisent et ne peuvent plus s'en passer !

■ François **Tonic**

Sources et compléments sur le site www.programmez.com

De l'intérêt d'une approche composant de l'architecture logicielle

La complexité des systèmes d'information a toujours continué de croître et les technologies disponibles ne cessent d'évoluer. De nos jours, sur un projet important, il n'est pas rare que les technologies utilisées évoluent de façon majeure avant même que le projet qui les utilise arrive à la fin.



Pour maîtriser cette complexité, l'approche de développement basé sur des composants est devenue incontournable pour la réussite, la maintenabilité et surtout l'évolutivité d'un projet de développement.

Lors d'un atelier OOPSLA 2000 , Alan Cameron Wills avait utilisé une analogie avec les systèmes HiFi pour imaginer la structure que devrait avoir un logiciel basé sur des composants :

- au plus haut niveau nous avons les appareils avec lesquels le consommateur interagit (lecteur CD, amplificateur, enceintes...) ainsi que les connecteurs et les interfaces standard permettant de les relier ;
- à un niveau plus bas nous trouvons les sous-systèmes dont ces appareils sont composés (alimentation, unité de lecture CD, etc.) et un autre jeu de connecteurs et d'interfaces standardisées ;
- en descendant encore d'un niveau, on trouve les circuits intégrés, les transistors, les diodes laser, etc. et à nouveau des connecteurs et des interfaces standard par le biais desquelles ces éléments sont connectés.

Cette petite illustration montre que la maturité actuelle atteinte dans l'industrie électronique est due à l'utilisation à chaque niveau des composants interconnectés de façon standard. De plus, comme dans beaucoup d'autres domaines matures, les composants utilisés sont produits de manière industrielle par des fabricants spécialisés et sont réutilisés pour assembler plusieurs types de produits. Dans le domaine logiciel, cette maturité est plus difficile à atteindre car les composants du bas évoluent encore très fortement et les sous-systèmes deviennent rapidement obsolètes. Ceux qui conçoivent des applications au plus haut niveau sont donc partagés entre tirer parti d'un niveau intermédiaire structurellement obsolète ou revenir à un niveau plus bas avec l'assurance d'une difficulté à évoluer.

Pour conserver la similitude, l'approche de développement de composants proposée par CodeFluent consiste à construire des sous-systèmes à partir des besoins du haut niveau en les modélisant et assemblant de manière automatisée les éléments du niveau bas. On obtient ainsi une application au plus haut niveau découpée en composants (les sous-systèmes) qui peut évoluer au gré des innovations sur les couches basses.

De manière concrète, CodeFluent se présente comme une véritable fabrique logicielle, de bout en bout, pilotée par le modèle pour l'environnement .NET, avec une approche par composant. Notre fabrique logicielle est matérialisée par l'exécutable ***CodeFluent.Build.exe.*** Il reçoit en argument le nom du fichier " pivot " du modèle. Dans notre exemple, la commande à exécuter en ligne de commande est :

```
CodeFluent.Build.exe
SoftFluent.Contacts.xml
```

Exemple de componisation de l'architecture logicielle

Prenons alors un exemple simple : une application de gestion en ASP.Net avec une base de données SQL Server. L'application devra s'intégrer dans le système informatique actuel de l'entreprise pour lequel le choix d'une architecture orientée services (SOA) a été effectué. Il nous est donc demandé d'exposer nos services métier en tant que services Web. Naturellement, nous allons utiliser WCF pour exposer nos services.

Pour pouvoir utiliser la fabrique logicielle, nous devrons préalablement décrire nos entités ainsi que la configuration des producteurs de différents composants que nous souhaitons utiliser (ces éléments font fonctionner la fabrique). A partir de ces informations, l'outil génèrera le code des composants du projet.

Et le projet se découpera en deux parties : la description des entités métier et la configuration des producteurs de composants. Ces éléments vont alimenter en entrée la fabrique logicielle. Dans cette courte présentation, vous aurez pu constater tout l'intérêt d'une approche par composant de l'application. Et dites vous bien que cette approche n'est pas uniquement liée à .Net, elle est valable dans la plupart des langages utilisés aujourd'hui.

■ Catalin **Manoliu**,
consultant senior chez SoftFluent

Comment les composants améliorent l'interface utilisateur !

L'interface utilisateur d'une application n'est qu'un composant parmi tant d'autres : ce n'est que la partie visible de l'iceberg. Pourtant, pour la majorité des utilisateurs, l'application est confondue avec son interface. Et l'utilisation ou le rejet d'un logiciel est souvent lié à la qualité de son interface.

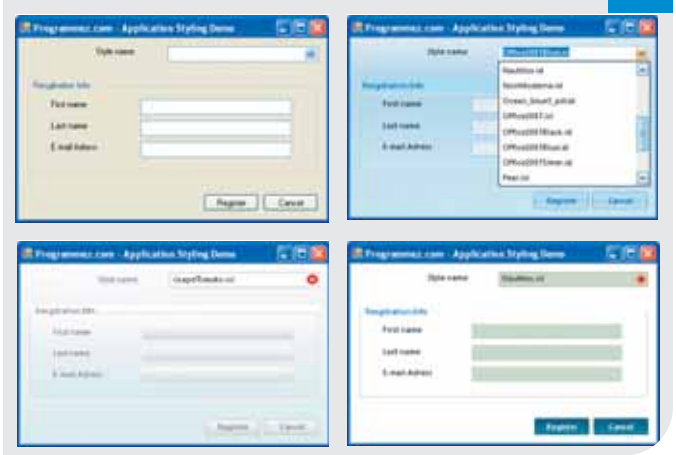
Tout doit être simple pour l'utilisateur, l'interface doit être cohérente et intuitive, attrayante, personnalisable... Autant de challenges à relever pour les concepteurs d'applications, les graphistes et les développeurs. Une des solutions qui s'impose est l'utilisation d'une bibliothèque de contrôles utilisateurs commerciale offrant des solutions puissantes et simples à mettre en œuvre pour la couche présentation de nos applications. J'ai utilisé plusieurs versions de la bibliothèque Infragistics (ma première expérience remonte à 2004) pour réaliser différent types de projets (Web, Client Windows, CAB...) et je dois dire que l'apport d'une telle bibliothèque est considérable si on prend le temps de bien connaître sa logique de fonctionnement et d'explorer sa large gamme d'outils. Une des raisons principales du choix de Infragistics dans mon dernier projet est le 'CAB Extensibility Kit' qui étend les fonctionnalités du framework open source CAB de Microsoft.

La preuve par A+B

A titre d'exemple, je vais introduire un cas simple d'utilisation de la fonctionnalité de gestion d'apparences nommée *ApplicationStyling* de la bibliothèque NetAdvantage for WinForms d'Infragistics. La fonctionnalité ApplicationStyling pour les applications Windows s'apparente à l'utilisation de fichiers de styles CSS pour les applications Web. Cette technologie permet de définir l'apparence de votre application dans des librairies de style et d'offrir plusieurs choix d'apparence pour vos utilisateurs (Il faut noter que ces styles ne s'appliquent qu'aux contrôles de la bibliothèque Infragistics). Voici quelques captures d'écrans de notre application finie : [Fig.1]

La liste déroulante libellée 'Style Name' présente à l'utilisateur les styles qu'il peut appliquer à son application (Rappelons que tous les contrôles utilisateurs –UltraLabel, UltraTextEditor, UltraComboEditor, UltraGroupBox et UltraButton– utilisés dans ce formulaires appartiennent à la suite NetAdvantage for WinForms 2008 v1).

Fig.1



Dans le gestionnaire d'évènement du chargement (Load) de notre unique formulaire 'FormMain' ajoutons l'appel à la méthode 'FillStyleList' qui va charger la liste des styles disponibles :

```
private void FormMain_Load(object sender, EventArgs e)
{
    FillStyleList();
}
```

Voici le corps de la méthode 'FillStyleList' :

```
private void FillStyleList()
{
    cmbStyles.Items.Clear();
    cmbStyles.Items.Add(String.Empty, String.Empty);

    DirectoryInfo stylesFolder
        = new DirectoryInfo(
            Path.Combine(Application.StartupPath, "Styles"));

    FileInfo[] styleFiles = stylesFolder.GetFiles("*.isl");

    foreach (FileInfo styleFile in styleFiles)
    {
        cmbStyles.Items.Add(styleFile, styleFile.Name);
    }
}
```

Cette méthode cherche les fichiers de styles (d'extension '*.isl') dans le sous-dossier 'Styles' du dossier d'exécution de notre application, puis alimente l'UltraComboEditor 'cmbStyles'. Enfin dans le gestionnaire de l'évènement 'ValueChanged' de l'UltraComboEditor 'cmbStyles' il nous suffit de charger le style sélectionné en faisant appel au composant 'StyleManager' responsable de l'habillage de tous nos contrôles Infragistics avec le style demandé :

```
private void cmbStyles_ValueChanged(object sender, EventArgs e)
{
    if (cmbStyles.Value != null)
    {
        FileInfo styleFile = cmbStyles.Value as FileInfo;
        if (styleFile != null)
        {
            StyleManager.Load(styleFile.FullName);
        }
    }
}
```

Enfin il est important de noter que .Net 3.5 SP1 offre en standard des fonctionnalités avancées pour créer des interfaces utilisateur pour tout type d'applications : Windows (Windows Forms et Windows Presentation Foundation), Web (ASP.Net, AJAX) et clients riches (Silverlight) et que Visual Studio 2008 rend cette tâche d'autant plus facile avec des concepteurs de formulaires et des assistants contextuels intelligents.

■ Ghassen **HARRATH**
Expert .NET chez IP-Tech, SSII tunisienne spécialisée dans les développements informatiques pour le marché Français (www.iptech-offshore.com)

Les outils de la Direction Informatique

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

L'INFORMATION SUR MESURE

Actu triée par secteur

Cas clients

Avis d'Experts

Etudes & Statistiques

Infos des SSII

Vidéos

Actus

Evénements

Newsletter

L'INFORMATION EN CONTINU

www.solutions-logiciels.com

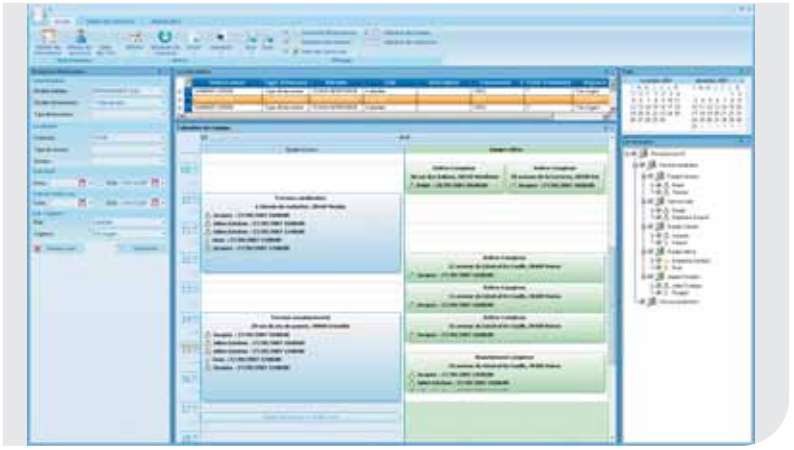
Retour d'expérience :

Yannick Biet, développeur .Net dans la société e-GEE (Meylan)

Je me suis intéressé aux composants Infragistics lors du développement de la couche graphique d'une solution applicative. Alors en phase de refonte technologique, nous cherchions à obtenir une interface moderne et séduisante pour nos clients, et permettant aux développeurs d'afficher des données et des formulaires de saisie complexes aux utilisateurs sans obtenir une interface austère. Nous nous sommes heurtés très tôt à deux problématiques qui nous ont poussés à nous intéresser aux composants logiciels :

- Nous devions pouvoir afficher des plannings et aucun composant standard ne proposait une réponse fonctionnelle à notre besoin,
- Nous devions également disposer de composants performants pour maquetter des solutions en accès web de nos produits.

Il s'agissait donc avant tout, de trouver une solution face à un manque des composants fournis en standard avec .net. Nous avions deux solutions : créer nos composants ou utiliser un composant tiers. Les avantages de la seconde solution compensent largement l'investissement demandé. Certes, il faut acquérir des licences de composants additionnels mais les coûts induits sont dérisoires si on ramène leurs coûts au prix d'une journée d'un ingénieur de développement, sans compter que pour le prix, nous obtenons alors plus de 100 composants, le support technique. De ce fait, l'exploration des autres composants nous a permis d'améliorer grandement notre interface. Nous avons pu en effet implémenter des menus en " ribbons " au look Office 2007, répondant ainsi au besoin de modernité de notre interface, et l'intégration des autres composants de saisie (texte, liste déroulante, ...) disposant eux-aussi de différents styles visuels, dont celui du look Office 2007 nous a permis de présenter une interface cohérente et homogène, sans fausse note.



David Berenger

(Développeur & consultant .Net, Softfluent)

" Eviter de tout refaire, de tout coder, principalement avec Ajax qui est compliqué et pouvoir réaliser une application " user friendly ", en quelques lignes ou via une simple API ! Nous utilisons régulièrement Telerik. Nous utilisons aussi des composants, Aspose, pour générer du PDF ou transformer des fichiers en PDF, pareillement pour générer de l'Excel, avec GemBox. J'utilise des composants quand cela est utile. Je les télécharge, je les teste. On voit rapidement les meilleurs. Le prix n'a jamais été un (réel) problème. Il n'est pas aussi excessif qu'on le pense ", explique David. *L'important est de bien choisir. " La pérennité est un élément à considérer. J'ai déjà rencontré des bugs. Sans code source, j'ai dû remonter dans celui-ci avec Reflector. Le composant est souvent une boîte noire. Quand on rencontre un problème dans un composant JavaScript, cela devient rapidement un cauchemar ",* précise David.

Extensibilité de l'affichage et de la personnalisation des composants

Si les propriétés par défaut des contrôles ne correspondent pas à vos attentes, on peut utiliser des filtres *création filter*, qui permettent de personnaliser le composant, et des filtres *Draw Filter* pour en modifier la présentation. Le développeur peut ainsi développer ses propres filtres, implémentant les interfaces particulières " IUIElementDrawFilter " et " IUIElementCreationFilter ". Ces interfaces contiennent les méthodes appelées par le framework lors de la création ou de l'affichage des composants. En utilisant ces filtres, le développeur a la possibilité d'accéder aux différents sous-éléments constituant le composant, les " UIElement " (élément graphique de base du Framework). Les utilisateurs peuvent retirer certaines de ses briques ou en ajouter de nouvelles pour adapter la présentation du composant au contexte fonctionnel souhaité. Par exemple, dans la capture d'écran ci-dessus, nous avons implémenté plusieurs filtres pour modifier l'aspect visuel des rendez-vous de notre solution de planning. En personnalisant ainsi le composant, nous avons pu afficher plusieurs lignes d'adresses et les ressources affectées au rendez-vous, chacune personnalisée par une icône.

Opter pour le look Office 2007 !

Depuis la version 2007 volume 3, NetAdvantage met à disposition des développeurs le look Office 2007 : Contrôles dont la mise en relief change au survol de la souris, utilisation de dégradés de couleurs et d'animations pour les transitions d'état... mais surtout donnant accès aux nouveaux modes de représentation des menus : Les rubans. On dispose aussi d'un éditeur graphique intégré au designer de fenêtre de Visual Studio, permettant de créer ses propres onglets, menus, boutons, galeries, onglets contextuels... L'ensemble des spécifications et des guides d'implémentation des rubans Microsoft est respectée et bien évidemment les trois looks standard " Office 2007" sont disponibles (Bleu, Noir, Gris Royal). Et si une fenêtre fille de votre application possède ses propres rubans en mode design, ceux-ci seront fusionnés de manière automatique aux rubans de la fenêtre parente lors de son affichage.

Comment créer des interfaces dynamiques

Dans cet article nous allons plus particulièrement voir l'utilisation des composants issus des gammes d'ILOG pour construire une interface graphique sur différentes plates-formes. Notre principal exemple concernera Elixir. Nous aborderons rapidement Java.

Une application Flex : une agrégation de composants

Le code source d'une application Flex est principalement constitué d'un fichier MXML qui décrit l'application. Chaque balise MXML dans ce fichier représente un composant qui constitue une des briques de l'application. La plupart de ces composants sont graphiques, mais il peut aussi s'agir de composants logiques comme ceux permettant l'accès à un service web. Une application Flex contient donc des données (ou alternativement l'accès à ces données à travers un service) et l'interface permettant d'afficher les données. Par exemple, la petite application qui suit affiche, sous forme d'un arbre simple, l'organigramme d'une société, lui-même défini en XML :

```
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml"
                backgroundColor="black">
  <mx:Script>
    <![CDATA[
     [Embed(source="user8.svg")]
      public var user8:Class;

      // ...

       private function pictureFunction(item:Object)
:Object {
        return this[item.@picture];
      }
    ]]>
  </mx:Script>
  <mx:XML id="">
    <employee name="Jon" picture="user8">
      <employee name="Martin" picture="user5">
         <!-- other employees -->
      </employee>
      <employee name="Nandini" picture="user3"/>
    </employee>
  </mx:XML>
   <mx:Label text="Organigramme de la société" color
="white"/>
  <mx:Tree dataProvider="{orgmodel}"
    labelField="@name" iconFunction="pictureFunction"
    width="100%" height="100%"/>
</mx:Application>
```

Le rendu d'une telle application est le suivant : [Fig.1]

On peut noter dans cette application l'utilisation de deux composants ; un de type Label et l'autre de type Tree (arbre). Le composant de type Label est très simple : il affiche juste un texte spécifié en dur par la propriété " text " et d'une couleur définie par la propriété de style " color ". Cette dernière pourrait être déclarée dans un fichier CSS externe pour rendre plus flexible le style de l'application.

Le composant de type arbre est tout simplement connecté à la source donnée en utilisant la propriété standard dans le monde Flex, nommée " dataProvider " (ou fournisseur de données). Les autres propriétés importantes pour la connexion aux données sont :

- l'attribut " labelField " qui permet au composant de savoir où chercher le texte à afficher dans l'arbre parmi les éléments du fournisseur de données, dans notre cas dans l'attribut " name " des nœuds XML ;
- et l'attribut " iconFunction " qui fournit une fonction ActionScript permettant de retrouver l'image de l'employé courant et de l'afficher comme icône dans l'arbre.

On dit que ces attributs permettent de lier le composant au fournisseur de données.

Aller plus loin en utilisant des composants avancés

Tentons maintenant d'améliorer le rendu de notre application. L'avantage de la programmation à base de composants est que si les différents composants respectent bien les mêmes paradigmes de programmation il devient relativement simple de faire évoluer l'application en remplaçant

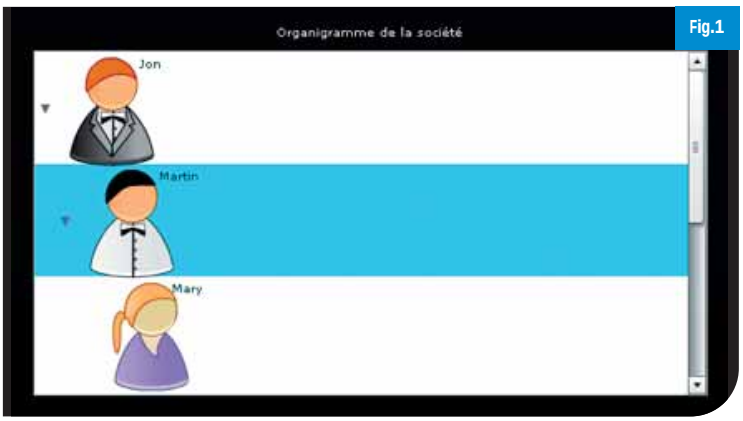


Fig.1

un composant par un autre plus évolué. Par exemple, dans l'extrait de code suivant, l'application conserve le composant Label mais remplace l'arbre simple de la plate-forme par un composant d'organigramme avancé OrgChart fourni par la suite de composant Elixir. Vous pouvez noter que la façon de lier le composant aux données est très similaire au cas précédent et notamment la propriété " dataProvider " est identique :

```
<ilog:OrgChart dataProvider="{orgmodel}"
  width="100%" height="100%">
  <ilog:fields>
    <ilog:OrgChartFields nameField="@name"
                pictureFunction="pictureFunction"/>
  </ilog:fields>
</ilog:OrgChart>
```

Comme vous pouvez le voir ci-dessous, le rendu à déjà beaucoup évolué à travers ce simple remplacement de composant : [Fig.2]

Nous pouvons aller encore plus loin en utilisant un autre paradigme des composants Flex. En effet sur cette plate-forme, les composants affichant des données comme l'arbre ou l'organigramme possèdent une propriété nommée " itemRenderer " qui permet de spécifier un composant délégué qui sera en charge de rendre chaque item, dans notre cas chaque employé. Bien sûr l'organigramme d'Elixir fournit un tel délégué par défaut, mais il est relativement facile de le spécialiser et de le remplacer par votre propre version. Un exemple typique est d'ajouter un bouton d'ouverture / fermeture sur chaque employé permettant à l'utilisateur de choisir de visualiser ou non ses subordonnés. Ceci est relativement aisé en sous-classant le composant délégué par défaut :

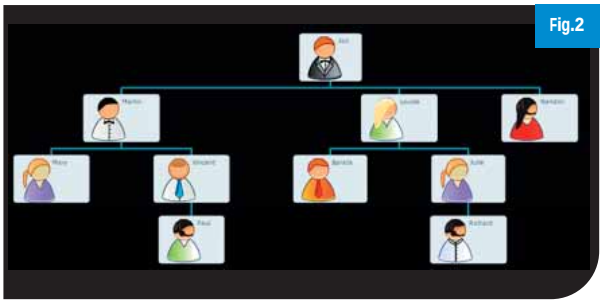


Fig.2

```
package
{
  public class ExpandCollapseItemRenderer extends Org
ChartItemRenderer
  {
    [Embed(source="expand.swf")]
    private static const EXPAND_IMAGE:Class;

    [Embed(source="collapse.swf")]
    private static const COLLAPSE_IMAGE:Class;

    private var _expandCollapse:Image;

// afficher EXPAND_IMAGE ou COLLAPSE_IMAGE en fonction
de l'état
// effectuer une action quand on clique sur l'image
// ...
  }
}
```

Puis en l'utilisant dans l'application à travers la propriété " itemRenderer ":

```
   <ilog:OrgChart id="orgchart" width="100%" height=
"100%"
   dataProvider="{collection}" initialize="collection
.refresh() ">
    <ilog:itemRenderer>
      <mx:Component><local:ExpandCollapseItemRenderer
/></mx:Component>
    </ilog:itemRenderer>
    <ilog:fields>
    <ilog:OrgChartFields pictureFunction="pictureFunction"/>
    </ilog:fields>
  </ilog:OrgChart>
```

Java et les composants

La plate-forme Java définit JFC / Swing comme modèle de composants pour l'interface utilisateur des applications desktop. Elle fournit un ensemble de composants comme JFrame, JTable et JTree permettant de construire rapidement des applications sur le desktop. La suite de visualisation ILOG JViews étend JFC / Swing et ajoute des composants graphiques permettant d'afficher et de manipuler interactivement des données clients.

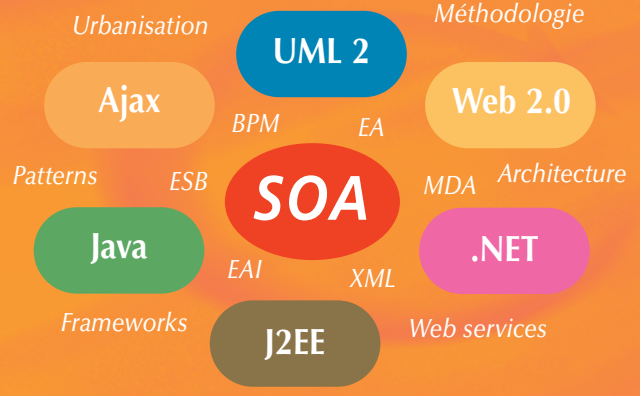
Une des fonctionnalités uniques de JViews est que ses composants graphiques sont également capables de se déployer pour le Web. Pour ce faire, JViews a adopté le modèle de composant JSF (Java Server Faces) défini par Sun. Le composant JSF est alors chargé d'exporter pour le Web l'interface graphique du composant desktop. Le fragment de code suivant montre comment utiliser un composant JSF dans une page Web :

```
<ivdf:diagrammerView id="diagrammer"
  style="width:600px;height:350px"
  data="/data/diagrammer.idpr"/>
```

La particularité du modèle JSF est que toute " l'intelligence " de l'application est située dans un serveur Java, et que l'application déployée consiste uniquement en un ensemble de pages HTML et fichier JavaScript. L'accès aux données s'effectue alors à travers toutes les possibilités fournies par la plate-forme Java, et ses nombreuses extensions.

- Christophe **Jolif** *(Principal Architect),*
- Yunpeng **Zhao** *(Senior Scientist),*
- Eric **Durocher** *(Principal Architect) - ILOG*

Soyez prêts
pour les nouvelles architectures
SOA et Web 2.0 !



SOA est devenu en peu de temps le mot-clé des développements logiciels. SOA est une nouvelle façon de faire qui s'appuie sur un ensemble de technologies existantes : UML, J2EE, .Net, XML, etc. Maîtriser SOA implique de maîtriser ces technologies pour les associer efficacement au sein d'une nouvelle approche.

SOFTEAM Formation, forte de son expérience en Méthodologie, Architecture et Développement, a construit un cursus complet de formation SOA qui vous permet de débuter dès les phases amont, de poursuivre en architecture, et d'aller jusqu'à la réalisation dans le langage de votre choix.

Nouveau catalogue Formation 2009 :

	UML pour la maîtrise d'ouvrage	2 j
	Analyse et conception avec UML	4 j
SOA	Alignement Métier du Système d'Information	2 j
SOA	Architecture d'Entreprise (EA)	2 j
SOA	Méthodologie pour SOA	2 j
SOA	Architecture technique SOA	2 j
SOA	Développement de Web Services en Java	3 j
SOA	Développement de Web Services en C#	3 j
	Architecture distribuée : la synthèse	2 j
	Programmation orientée objet avec Java	4 j
	Développement d'applications JEE 5	5 j
	Développement d'applications JEE 5 Front End	4 j
	Développement d'applications JEE 5 Back End avec EJB 3	3 j
	Maîtrise du framework (Struts / JSF / Spring / Hibernate)	3 j
	Développement d'applications .NET / C#	4 j
	Développement d'applications RIA avec	2 j
Web 2.0	(Ajax / Dojo / GWT / FLEX3)	4 j

Convergence SOA, UML2, BPMN, EA

Modélisation EA, BPMN, SOA avec Objecteering SOA Solution	5 j
Analyse et Conception UML2 avec Objecteering Modeler	5 j
Expression de besoins en UML2 avec Objecteering Requirements	1 j
Architecture MDA avec Objecteering MDA Modeler	2 j
Génération de code Java, .NET C#, C++ avec Objecteering Developer	1 j

SOFTEAM Formation
Calendrier complet et inscriptions en ligne
www.softeam.fr

Paris · Rennes · Nantes · Sophia

Tél. : 01 53 96 84 30 - Fax : 01 53 96 84 01
Paris : 21 avenue Victor Hugo - 75016
Rennes - Nantes - Sophia Antipolis

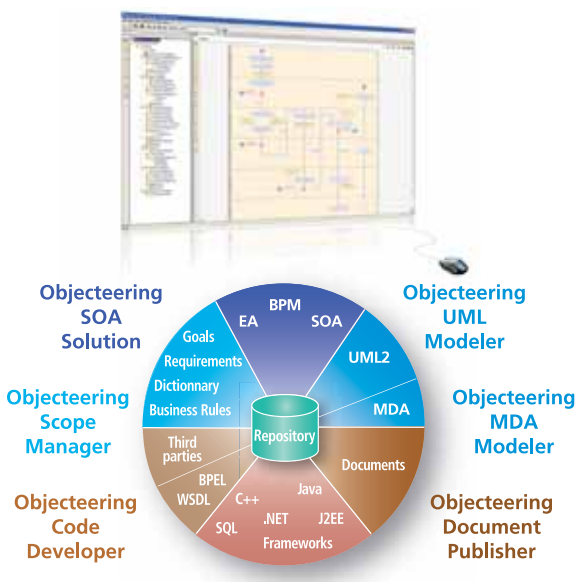
SOFTEAM
Think Object

Objecteering

*Your projects deserve a tool **

La convergence SOA, UML2, BPMN, EA, pour le développement guidé par le modèle

- outil **intégré** assurant la modélisation complète des applications et systèmes,
- **facile** à utiliser : à chaque acteur, à chaque phase, la représentation dédiée,
- **alignement métier** du SI grâce au référentiel partagé et la traçabilité totale entre toutes les vues et phases,
- haute **productivité** grâce à la génération automatique de la documentation et du code.



* *Vos projets méritent un outil.*

www.objecteering.com
Tél. : 01 30 12 16 60 - sales@objecteering.com

Objecteering SOFTWARE
The model-driven development compagny

Des composants MVC pour PHP 5

eZ Components (http://ezcomponents.org) est une bibliothèque de composants dédiée à PHP5. Son développement est assuré et coordonné par Ez Systems. La bibliothèque est Open Source (licence New BSD). Dans cet article nous allons décrire l'utilisation de MvcTools.

MVC découple la couche métier, de la génération du code côté client et de la logique des actions. Ainsi, une modification de l'une de ces trois couches permet un gain de rapidité :

- de développement, grâce à la ré-utilisation de classes existantes,
- de maintenance et évolutivité, puisque les modifications d'une couche ne devraient pas avoir de répercussion sur les autres.

Le composant MvcTools fournit aux développeurs tous les outils pour architecturer une application. Son objectif n'est pas de fournir un framework complet mais seulement différentes briques qui peuvent être utilisées. Ce composant est également directement utilisable sans framework, comme le démontrent les applications **TheWire** : une application du style de **Twitter** pour partager des informations; ainsi que **HelloMvc** : l'exemple qui salue le visiteur dans son langage favori. Les applications sont téléchargeables avec SVN aux adresses suivantes :

TheWire : http://svn.ez.no/svn/ezcomponents/docs/examples/applications/TheWire/

HelloMvc : http://svn.ez.no/svn/ezcomponents/docs/examples/applications/HelloMvc

Classes principales

Le composant MvcTools fournit des classes pouvant être utilisées dans différentes parties d'un framework MVC. C'est-à-dire des fonctionnalités pour le dispatching, lecture de requête, routage, génération de vues, génération de réponse et filtrage.

Parseurs de requête

Le parseur de requête est chargé de lire la requête et de générer un objet ezcMvcRequest correspondant. Au moment ou cet article est écrit, il en existe deux: un pour les requêtes HTTP, et un pour les emails.

Routage

Le routeur analyse les données de la requête et sélectionne le contrôleur adéquat. MvcTools fournit actuellement deux types de routes pour faciliter la programmation du routeur: une pour tester l'url absolue avec une expression régulière et l'autre avec une syntaxe similaire à celle du framework Rails. Il faut étendre la classe ezcMvcRouter et implémenter la méthode abstraite createRoutes() pour qu'elle renvoie la liste de vos routes.

Exemple d'implémentation dans le cas de l'application “Hello” :

```
<?php
/**
 * Le routeur de votre application “hello”.
 */
class helloRouter extends ezcMvcRouter
{
    /**
    * Cette méthode renvoie la liste des routes de l'application.
    *
    * Elle devrait retourner un tableau d'objets qui implémentent
    * l'interface ezcMvcRoute. Par exemple des instances de
    * la classe ezcMvcRegexpRoute.
    *
    * @return array(ezcMvcRoute)
    */
    public function createRoutes()
    {
      return array(
        new ezcMvcRailsRoute( '/:nom', 'helloControler', 'salutations
Personnelles' ),
        new ezcMvcRailsRoute( '/', 'helloControler', 'salutations' ),
      );
    }
}
?>
```

Chaque route définit une expression ('/' ou '/:nom') ainsi qu'un contrôleur (helloControleur) et une action ('salutationsPersonnelles' ou 'salutations'). Le routeur peut se baser sur des urls préfixées, il est ainsi directement utilisable dans un autre projet sans causer de conflit.

Contrôleur

Le contrôleur doit hériter de la classe abstraite ezcMvcController pour être utilisable avec le dispatcheur: c'est le dispatcheur qui l'instancie grâce aux informations renvoyées par le routeur, encapsulées dans un objet de classe ezcMvcRoutingInformation. Le dispatcheur exécute ensuite la méthode *createResult()* du contrôleur, qui doit retourner son résultat encapsulé dans un objet de l'une des classes suivantes: ezcMvcInternalRedirect: dans le cas d'une redirection interne; ou ezcMvcResult: pour encapsuler les variables à envoyer à la vue. MvcTools fournit également une exception à jeter dans le cas où l'action n'a pas été trouvée: ezcMvcActionNotFoundException.

Vue

Les classes de génération de vue actuellement mises à disposition par **MvcTools** utilisent donc l'objet de classe ezcMvcResult renvoyé par le contrôleur. Ainsi, il est possible d'utiliser le même objet de résultat avec n'importe quel type de vue. Avant de commencer à programmer la vue, il faut comprendre le concept des zones. Les zones sont un moyen d'arranger les différentes parties d'une mise en page (“layout” en anglais). Considérez la mise en page suivante : [Fig.1]

Il y a trois zones dans ce petit exemple :

- La zone “menu”, par exemple avec un lien vers la page d'accueil.
- La zone “contenu”, par exemple avec le corps d'un article.

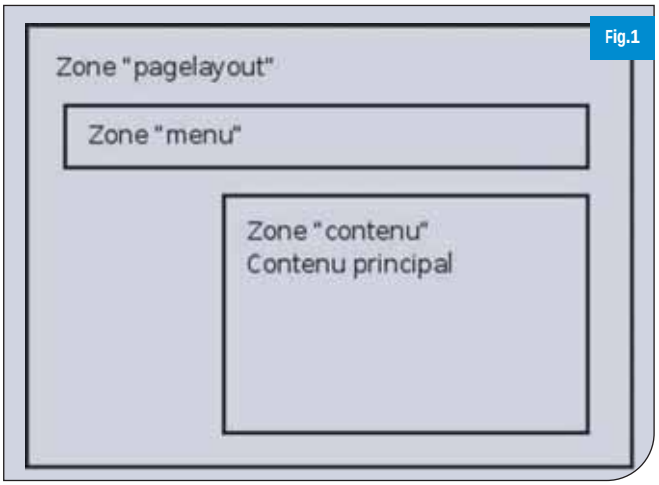


Fig.1

- La zone “pagelayout”, qui encapsule les deux autres, et fournit la mise en page générale, les en-têtes HTML tels que les feuilles de styles.

Chaque vue que vous définissez peut inclure plusieurs zones, chacune a un nom qui sert d'identifiant ainsi qu'un générateur. A l'instar du routeur, il faut étendre la classe de vue abstraite ezcMvcView et implémenter une méthode qui doit retourner la liste des zones de votre application. Toute les zones d'une vue n'ont pas besoin d'utiliser le même générateur. Le paramètre $layout détermine si des zones de type mise en page générale doivent être incluses. Ainsi, il suffit d'appeler la méthode createZones(false) de la vue d'une autre application pour l'inclure, sans inclure sa zone de mise en page, afin d'éviter les conflits. Exemple :

```
<?php
/**
 * La vue de votre application “hello”.
 */
class helloView extends ezcMvcView
{
    function createZones( $layout )
    {
      $zones = array();
      $zones[] = new ezcMvcTemplateViewHandler( 'menu', 'menu.ezt' );
      $zones[] = new ezcMvcPhpViewHandler( 'contenu', '../
templates/salutations_generiques.php' );
      if ( $layout )
      {
        $zones[] = new ezcMvcTemplateViewHandler( 'page_layout',
 'layout.ezt' );
      }
      return $zones;
    }
}
?>
```

Sont actuellement disponibles: ezcMvcPhpViewHandler pour les templates en PHP, ezcMvcJsonViewHandler pour répondre à vos requêtes AJAX par exemple, et ezcMvcTemplateViewHandler qui couple le composant **Template de la bibliothèque eZ Components.** Les vues encapsulent les en-têtes ainsi que le corps de la réponse un objet de classe ezcMvcResponse.

Écriveur de réponse

L'écriveur de réponse a pour responsabilité de renvoyer une réponse utilisable par le client grâce à l'objet de réponse renvoyé par la vue. Un seul écriveur de réponse est actuellement livré avec MvcTools: ezcMvcHttpResponseWriter.

Dispatcheur

Le dispatcheur est responsable du flux complet de la requête. Sans lui, les classes sont lâchement couplées donc facilement réutilisables. Le composant MvcTools livre un dispatcheur basique pour l'instant: ezcMvcConfigurableDispatcher. Il dépend d'un objet de configuration d'interface ezcMvcDispatcherConfiguration, et c'est à ce dernier qu'il délègue les tâches spécifiques tout au long de son exécution :

- instanciation du parseur de requête,
- création de la requête de redirection fatale si nécessaire, donc avec l'url de votre choix,
- filtrage de la requête: ajout, modification, suppression de variables de l'objet requête. Avant et après l'exécution du routeur.
- instanciation du routeur, vous pouvez ainsi importer les routeurs d'autres applications,
- filtrage de l'objet de résultat du contrôleur,
- instanciation de la vue,
- filtrage de l'objet de réponse,
- instanciation de l'écriveur de réponse.

Conclusion

L'utilisation des eZ Components et de MvcTools est une option intéressante pour la réalisation d'application Web. Il s'agit d'une approche unique en langage PHP aujourd'hui, différente de celle proposée par d'autres framework. Cet article a permis de parcourir les principaux éléments d'une telle implémentation. On peut pour l'instant regretter que la documentation ne soit disponible qu'en anglais, celle-ci est néanmoins excellente. L'abstraction de protocole permet à vos contrôleurs d'être nativement utilisables pour créer des applications plus riches, par exemple: avec ou sans AJAX, RSS, Atom, etc. Le préfixage des routes par-routeur permet de ré-utiliser directement les contrôleurs d'autres applications. Le système de zone de la vue permet de ré-utiliser directement les vues d'autres applications.

■ James **Pic**
Développeur indépendant, membre actif du groupe de développement d'eZ Components, il fait partie du pôle de compétence ChocolatPistache.
avec la participation de
■ Derick **Rethans**
Architecte senior chez eZ Systems, responsable du projet eZ Components, il est également un important contributeur du projet PHP.
■ Roland **Benedetti**
Manager d'eZ Systems France
http://ezcomponents.org et http://ez.no.

GLOSSAIRE

MVC : *acronyme de "Modèle-Vue-Contrôleur". MVC est un concept utilisable dans des architectures logicielles d'application Web. MVC est un patron de conception orienté objet.*

Parser / parseur : *analyseur de document XML, il permet d'en extraire les données (on parle de parsing de document) et de vérifier la validité du document.*

Décuplez la puissance de Flex !

Le développement à base de composants logiciels consiste à décomposer un logiciel en plusieurs composants fonctionnels ou logiques, avec des interfaces bien définies utilisées pour la communication entre composants. L'objectif de cette approche est de rendre la création de logiciel proche d'autres activités d'ingénierie comme l'électronique ou la mécanique, dans lesquelles on pioche dans des banques de composants existants que l'on assemble selon les besoins, plutôt que de récréer tous les éléments.

Un **composant logiciel** est un élément d'un système informatique qui propose des services et / ou des événements prédéfinis, et qui est capable de communiquer avec d'autres composants.
D'après Clemens Szyperski et David Messerchmitt, un tel composant devrait vérifier les conditions suivantes :

- Ne pas être spécifique à un contexte donné
- Etre assemblable avec d'autres composants
- Etre encapsulé, c'est-à-dire non accessible autrement que par les interfaces proposées
- Etre une unité indépendante, déployable et versionnable indépendamment des autres éléments du système

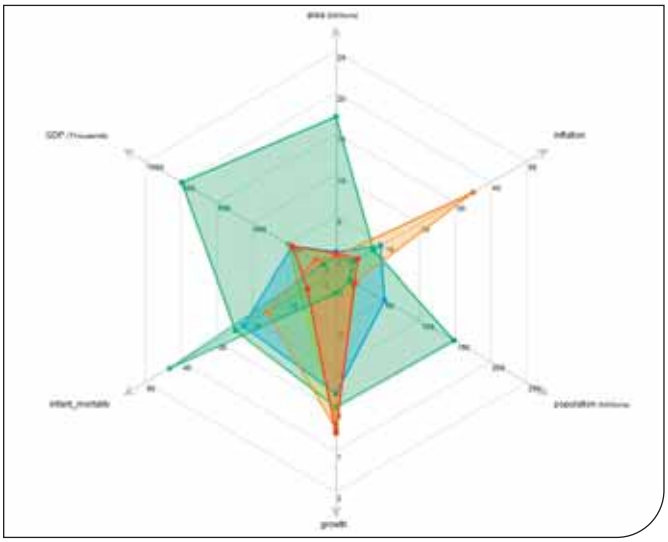
Quelles contraintes ?

Développer un composant qui soit effectivement réutilisable demande un effort important, car le composant requiert:

- Une documentation exhaustive
- Des tests très poussés
- Un contrôle de validité des entrées très robuste
- Un transfert retour des messages d'erreur, là où c'est approprié
- D'être développé avec la conscience qu'il sera utilisé de manière non prévue

Exemple : macro composant en Flex

Nous allons ici présenter un exemple basé sur un des composants Flex du KapLab, choisi pour sa simplicité d'utilisation : le *RadarChart*. Flex nous permet de réaliser très simplement de nouveaux composants facilement réutilisables, dès lors qu'ils sont basés sur des composants existants. La réalisation d'un composant entièrement en AS3, sur la base d'un UIComponent est en revanche nettement plus complexe et ne sera pas abordée ici.



Objectifs du macro-composant

Nous allons fabriquer un macro-composant, formé du composant " DataGrid " du framework Flex, et du composant " RadarChart ", disponible en téléchargement gratuit sur le site du KapLab (http://lab.kapit.fr). Simple d'utilisation, il suffit de lui spécifier un dataProvider pour qu'il affiche tout seul les données de manière correcte. L'objectif est d'obtenir un composant de reporting " prêt à l'emploi ", acceptant un simple fichier XML en entrée pour afficher ensuite un Radar et une liste permettant de sélectionner les séries à afficher. Le composant se chargera d'interpréter le contenu du fichier XML afin d'en extraire la liste des séries et des valeurs à afficher (numériques). Par ailleurs, nous définirons aussi un événement custom afin de détecter les séries sélectionnées.
Le composant devra donc fonctionner tout seul, à partir du moment où on lui aura fourni la source de données XML, ainsi que le nom de l'attribut qui servira de label aux séries.
Le composant développé sera utilisé dans une application de test, avec deux jeux de données XML différents, et une DataGrid permettant d'afficher les données sélectionnées.

Création du composant

Avec Flex, il est possible de créer un nouveau composant avec de l'actionscript pur ou bien avec du code MXML. Nous utiliserons la deuxième approche qui simplifie et raccourcit l'écriture du code. Il suffit donc de créer un nouveau fichier MXML et de choisir la classe de base dont il héritera, par exemple VBox. Le nom de ce fichier MXML représente en fait le nom de la classe du composant créé, qui est immédiatement réutilisable.
Nous allons donc ici créer le fichier " RadarComponent.mxml ", qui sera basé sur une HDividedBox :

```
RadarComponent.mxml
<mx:HDividedBox xmlns:mx="http://www.adobe.com/2006/mxml"
xmlns:radarchart="fr.kapit.radarchart.*">
…
</mx :HDividedBox>
```

Ce fichier MXML sera traduit en code AS3 avant d'être compilé. Ainsi, le fichier RadarComponent.mxml sera-t-il traduit en une

classe du même nom (RadarComponent), qui sera donc réutilisable soit dans du code AS3, soit dans des tags MXML.

Propriétés du composant

Il nous faut définir l'interface publique du composant, notamment les propriétés qui seront accessibles par l'utilisateur. Nous aurons besoin de deux variables :

- labelField : cette propriété indiquera le nom de l'attribut du fichier XML de données, qui servira de titre aux séries. Par défaut, la valeur " name " sera utilisée.
- dataProvider : de type XML, c'est la propriété principale, qui nous permettra de transmettre les données à afficher dans le composant

```
[Bindable] public var labelField : String = "name";
public function set dataProvider(value:XML) : void
```

On remarquera que "labelField" est bindable, ce qui facilite son utilisation, et que la variable dataProvider est définie au travers d'une méthode "setter". Ce dernier point est important, car il nous permettra de contrôler l'affichage du composant au travers de la méthode " commitProperties ", ce qui est la bonne façon de faire avec Flex.

Evénement custom

Nous allons aussi définir une classe d'événement spécifique afin de notifier des changements de sélections sur les séries. Pour cela, commençons par créer une nouvelle classe qui hérite de la classe Event, et dans laquelle nous stockerons la liste des séries sélectionnées :

```
package fr.kapit.demo
{
  import flash.events.Event;

  public class SerieChangeEvent extends Event
  {
…
  }
}
```

Puis nous déclarons cet événement dans notre composant à l'aide des MetaData. Nous verrons dans le code source que cet évènement sera dispatché lors de la sélection de nouvelles séries.

Traitement des données XML

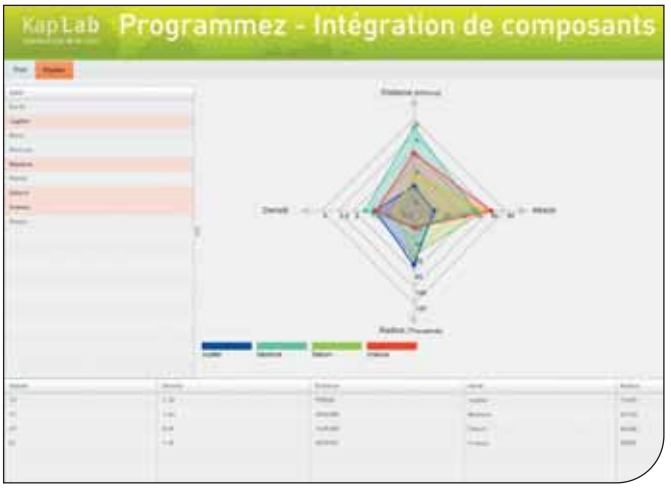
Le setter du dataProvider effectue les opérations suivantes :

- Décodage du XML dans un tableau d'objets dynamiques
- Stockage de ce tableau dans une variable interne privée (_dataProvider)
- Positionnement d'un flag de modification du dataProvider (_dataProviderChanged)
- Invalidation des propriétés (invalidateProperties) pour déclencher l'appel à la fonction *commitProperties*

Affichage du graphe et fonction commitProperties

Dans la fonction commitProperties, on effectue les opérations suivantes :

- Remise à zéro du flag de modification du dataProvider



- Création des colonnes de la DataGrid
- Création des axes du graphique
- Affectation du dataProvider de la DataGrid, ce qui déclenche son affichage
- Affectation du dataProvider du graphe, ce qui déclenche sa mise à jour

Utilisation du composant

Nous allons tout d'abord créer une application principale, ComponentDemo.mxml, dans laquelle nous allons :

- Importer deux jeux de données XML différents, via la directive [Embed] : le fichier " world.xml " qui contient des informations géographiques, et le fichier " planet.xml " qui contient des données sur les planètes du système solaire.
- Déclarer deux instances du composant créé, en spécifiant les données d'entrée.

Le code complet est sur le site www.programmez.com.

On notera les points remarquables suivants :

- Le même composant peut être réutilisé autant de fois qu'on le désire au sein d'une même application
- Pour des raisons de simplicité, notre composant est inclus dans l'application, mais il serait très simple de l'en extraire et de le packager sous la forme d'une librairie SWC réutilisable dans différentes applications (ce qui est effectivement nécessaire pour un composant du monde réel)
- Nous avons pu fournir des jeux de données différents à chaque composant, en utilisant les interfaces d'entrée
- Nous avons pu réagir à l'utilisation d'un composant, au niveau de l'application qui l'utilise, en nous branchant sur les évènements envoyés par le composant (messages)

L'objectif initial est donc atteint, même s'il ne s'agit que d'un exemple très simple. Au travers de celui-ci, nous avons pu cependant vérifier que Flex est le langage de choix pour le développement d'applications à base de composants. Le framework Flex contient lui-même plus de 100 composants pour fabriquer les applications RIA, et la communauté de développeurs en propose encore bien plus.

■ Julien **Revel**

Associé et co-fondateur de Kap IT, il est Directeur Technique et Innovation. Kap IT est une société de conseil, de R&D logiciel spécialisé dans les applications Web 2. http://lab.kapit.fr

Trouver les bons composants

JAVA

Gammes Ilog : l'éditeur récemment racheté par IBM propose plusieurs familles de composants pour les développeurs Java. **JRules,** un ensemble de composants pour les règles métiers, et offre une gestion complète (création, déploiement, monitoring. La famille **JView** se spécialise dans la présentation de données, les graphiques. On a le choix entre **Diagrammer** (librairie pour créer, gérer des graphiques, tableaux de bord...) et **Maps** (cartographie). www.ilog.fr

Infragistics NetAdvantage for JSF : package complet de composants Ajax pour JSF. Propose de nombreux objets d'interface. Régulièrement mis à jour. L'éditeur ne supporte plus la gamme **JSuite.**

Gamme Aspose : éditeur peu connu en France, il propose plusieurs composants Java orientés bureautique, qui permettent par exemple de manipuler, ouvrir, générer des fichiers Excel, Word, PDF sans installer Office ! Supporte aussi OpenXML, OpenDocument, RTF. Pratique et puissant ! L'éditeur propose également un composant pour code à barres (**Aspose.Bar Code**). www.aspose.com

Chilkat : éditeur quasi inconnu en France mais très prolifique, propose de très nombreux composants pour utiliser, implémenter le mail, zip, cryptage, SSH, XML, FTP, etc. Disponible en différents langages dont Java, .Net, Python, C++. http://www.chilkatsoft.com/
Et aussi : **Extreme Component** (http://www.zfq-java.com/), spécialisé dans l'interface, la présentation. **Eltima** (www.eltima.com), propose des composants d'interface pour Swing et les graphiques. **SoftwareFX** avec ChartFX Developer Studio, **PDFlib** (composants pour PDF).

Où les acheter ? Les télécharger ?

Pour les contrôles open source, gratuits, les sites des projets et des éditeurs permettent de les récupérer. Des versions d'essai sont souvent disponibles sur le site de l'éditeur.
Pour acheter, vous pouvez soit passer par le site officiel quand une boutique en ligne existe ou par des distributeurs. En France, on citera : ComponentSource, SOS Developers, Kapitec, STR.

FLEX

Ilog Elixir : composants pour Flex et Air. Propose de nombreux objets d'interface (graph, gauge, organigramme, cartographie). La version 2.0 sera prochainement disponible.

KapIt : éditeur français, il propose toute une gamme de composants pour Flex, allant des diagrammes à la présentation, en passant par la console MVC, le treemap, etc. www.kapit.fr

Efflex : composants graphiques exploitant Flex et Air. On dispose d'un zoom de grid, d'un flip 3D... Pratique et libre. www.efflex.org

FusionCharts : puissant composant animé pour créer des graphiques. Supporte de nombreuses sources de données. Il existe aussi des composants de cartographie, de widget. Pour en savoir plus sur l'offre composants flex : http://flex.org/software/components

.NET

Infragistics : un des éditeurs historiques du marché. L'éditeur décline ses contrôles pour .Net, ASP.Net, winforms, Silverlight, WPF. Cela concerne aussi bien les objets d'interface que les classiques graphiques, diagrammes, radars, etc. Supporte les derniers outils et versions .Net. www.infragistics.com

DevExpress DXperience : package complet de composants d'interface pour Winforms et ASP.Net. www.devexpress.com

Telerik fournit une large gamme de contrôles graphiques pour Winforms, WPF, Silverlight, ASP.Net. Mais l'éditeur propose aussi des composants de mapping relationnel / objet, de reporting. www.telerik.com

Nevron : peu connu en France, Nevron développe des composants haut de gamme pour les graphiques, diagrammes et interfaces utilisateur en général. www.nevron.com

ComponentOne : l'offre de l'éditeur concerne ASP.Net Ajax, WPF, Silverlight et Winforms. Studio Enterprise inclut l'ensemble des contrôles .Net + Actives et la partie mobilité.

L'éditeur possède aussi une gamme de composants pour le reporting. www.componentone.com

ComponentArt : autre spécialiste de l'interface, ComponentArt se spécialise dans l'interface des sites ASP.net avec des contrôles Ajax particulièrement puissants comme un sélecteur de couleur, le créateur du multipage, etc. La partie graphs et diagramme n'est pas oubliée avec Charting pour ASP.net et .Net, ainsi que des contrôles de reporting.
www.componentart.com
Nsoftware : éditeur de composants multi-langages (.Net, ActiveX, java, C, C++, etc.). Fournit de nombreux composants pour le réseau (SSL, mail, SSH), la compression (Zip) ou encore les fonctions business (paypal), Biztalk.
www.nsoftware.com
Xceed : une offre très ouverte sur l'interface (3D, carrousel, diagrammes) aussi bien pour .Net que WPF ou ASP.Net. L'éditeur dispose aussi d'une solide offre sur le stockage, la compression (en Stream, en local, et même en 64bit), sans oublier les fonctions de réseau comme FTP. www.xceed.com

Et aussi : **Iocomp, Dundas, Janus, SoftwareFX, Dart, VantagePoint**

Les autres langages

Impossible de tout lister sur une seule page ! Il existe une offre nombreuse sur les autres langages et plates-formes : Ajax, Javascript, PHP, Delphi, C, C++, Python. Sur PHP, on peut citer l'un des plus connus : ezcomponent. Ou encore les contrôles de JomiTech (JCT). De nombreux éditeurs cités ci-dessus déclinent leurs composants pour différents langages.

Wygwam™

Microsoft Surface™

Wygwam™ : Les techniciens de la Surface™

Stand CUC9

Visible sur le stand de Wygwam™
sur les **TechDays 2009** de Paris

Wygwam
Rue de la Performance 1 – Bâtiment B4 | 59650 Villeneuve d'Ascq | France
Tél. : +33 (0)3 20 82 38 77 | info@wygwam.com | www.wygwam.com

Wygwam BeLux
Rue Victor Corne 64 | B-7700 Mouscron | Belgique
Tél. : +32 (0)56 33 06 60 | info@wygwam.com | www.wygwam.com

En plus de Microsoft Surface™, nous continuons bien entendu à vous guider
sur les technologies Microsoft™ suivantes grace à nos 7 MVPs et notre Regional Director

Microsoft .NET | Live Mesh | Windows Azure | Microsoft HealthVault

Microsoft Office SharePoint Server 2007 | Microsoft Silverlight | MVP Microsoft Most Valuable Professional | Microsoft Certified Partner

Développez votre site web avec ASP.Net !

ASP.NET est un langage web appartenant à la plate-forme .NET de Microsoft, qui vous permet de créer des sites web rapidement et simplement. Nous allons plonger au cœur d'ASP.NET pour mieux l'appréhender. ASP.NET propose un ensemble de technologies qui vont vous simplifier l'écriture d'applications web dynamiques.

ASP.NET a accès à toutes les classes du framework .NET qui est un ensemble de composants pouvant être utilisés dans différents langages de programmation (grâce à la CLR), dont les plus connus sont :

- VB.NET (successeur de VB).
- C#, langage spécialement conçu pour la plate-forme .NET
- IronRuby qui, comme son nom l'indique, est un Ruby sous .Net.
- IronPython qui, de la même manière, est un Python sous .Net.

En résumé, peu importe le langage que vous voulez utiliser, Microsoft met à disposition un ensemble d'outils qui vous faciliteront l'écriture d'une application ou / et d'un site internet. L'ensemble de ces outils, constitue le framework .NET. Nous en sommes aujourd'hui à la version 3.5.

Pré-requis

Combien coûtent .Net et les outils ? En fait, vous pouvez débuter le développement ASP.Net gratuitement ! La gamme Express est suffisante pour démarrer en .Net et dans le développement web ! Vous pouvez dès lors utiliser Visual Web Developer Express pour développer vos sites internet et SQL Server Express pour la partie données.

Vous pouvez retrouver ces deux outils à cette adresse : http://msdn.microsoft.com/fr-fr/express/aa975050.aspx

Si néanmoins vous désirez directement utiliser les outils professionnels, vous pouvez utiliser Visual Studio 2008.

Vous pouvez également télécharger le Framework 3.5 à cette adresse : http://www.microsoft.com/downloads/details.aspx?FamilyId=333325FD-AE52-4E35-B531-508D977D32A6&displaylang=en

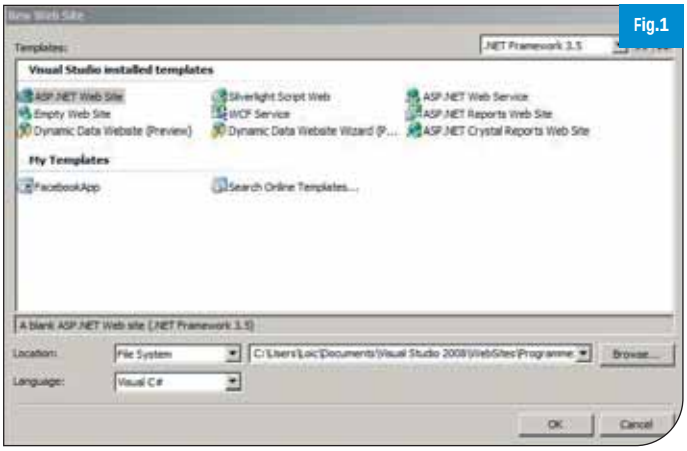
Ce n'est cependant pas obligatoire, le framework 2.0 est installé par défaut et il suffira à développer en ASP.Net. Le framework 3.5 est aujourd'hui le « standard » pour .Net. Ce dernier n'apporte pas beaucoup de nouveautés pour le développement web.

Mon premier projet ASP.Net

Que ce soit avec Visual Web Developer ou Visual Studio, la procédure est la même. Premièrement, vous devez créer un site web. Pour cela, rendez-vous dans le menu File -> New -> Web Site. [Fig.1]
Sélectionnez ensuite « ASP.NET Web Site », donnez-lui un nom et cliquez sur « Ok ». Visual Studio / Web Developer crée pour vous un ensemble de fichiers et de dossiers pour vous aider à démarrer. Vous pouvez voir ces fichiers dans l'explorateur de solution. [Fig.2]
Default.aspx est souvent la page par défaut d'un site ASP.NET. Ceci est configurable dans IIS qui est le programme traitant les demandes sur des pages internet (qui écoute sur le port 80 par exemple). C'est ce que fait Apache si vous êtes habitué à travailler avec PHP, Ruby etc.
Vous remarquez que Default.aspx est un fichier qui lui est attaché. Ce fichier s'appelle Default.aspx.cs. [Fig.3]
C'est dans ce fichier que se trouvera toute la logique métier de notre page. Microsoft a voulu séparer le design et le code, ce qui est une bonne chose. Vous aurez donc tout l'aspect graphique au niveau de Default.aspx et tout l'aspect code au niveau de Default.aspx.cs (on appelle le code contenu dans ce fichier: le code behind). Ceci n'est pas obligatoire, vous pouvez déclarer le design et le code dans un même fichier. Cependant, **nous vous conseillons fortement de toujours séparer le code, de l'interface.** Pour la maintenance, l'évolution, le debug de l'application, c'est bien plus pratique. Le code se trouvera alors entre des balises « script » mais cela nuit à la lisibilité du code et à la compréhension de celui-ci.
Default.aspx ressemble à ceci lorsque vous l'ouvrez :

```
<%@ Page Language="C#" AutoEventWireup="true"  CodeFile="Default.
aspx.cs" Inherits="_Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>

    </div>
    </form>
</body>
</html>
```

Fig.1



Cette page ne fait rien! Si vous voulez voir un aperçu dans le navigateur, vous pouvez effectuer un clic droit sur le nom de la page dans l'explorateur de solution et cliquer sur « View in browser ».
Vous verrez que votre navigateur par défaut se lance et affiche une page blanche. En fait, le contenu de votre page doit être situé entre les balises « form ». Nous reviendrons plus tard sur ces balises.
Si nous regardons la première ligne de notre page, nous remarquons qu'elle indique de nombreuses informations sur la page :

```
<%@ Page Language="C#" AutoEventWireup="true"  CodeFile="Default
.aspx.cs" Inherits="_Default" %>
```

Cette ligne est appelée une « directive ». Toutes les directives d'une page commencent par <%@ et finissent avec %>. Elles permettent de déclarer des informations qui sont nécessaires au compilateur pour pouvoir compiler la page. Par exemple, nous avons besoin de savoir quel langage est utilisé (Language="C#") pour savoir quel compilateur traitera la page, ou encore quel fichier contient le code (CodeFile="Default.aspx.cs"). Le reste de la page devrait vous être familier si vous avez déjà l'habitude de programmer des sites internet. On retrouve une ligne pour le dtd :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

Note : Visual Studio gère très bien ce dtd et vous indique une erreur lorsque vous ne le respectez pas dans votre HTML.

Le reste de la page ne devrait pas vous poser de problème de compréhension, hormis le « form ».

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>

    </div>
    </form>
</body>
</html>
```

Fig.2

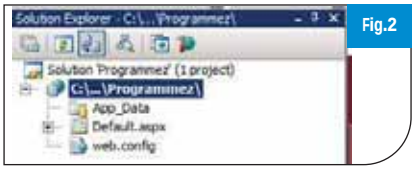
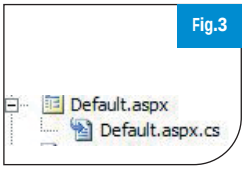


Fig.3



Au niveau du code behind, nous avons ceci :

```
using System;
using System.Configuration;
using System.Data;
using System.Linq;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Xml.Linq;

public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }
}
```

On remarque directement de nombreux « using » qui permettent de faciliter l'utilisation des composants du framework.NET.
Par exemple, sans « using System.IO » vous devriez écrire dans votre code :

```
System.IO.File.Exists("file.txt");
```

Avec le using, vous simplifiez votre code :

```
File.Exists("file.txt");
```

Les « using » qui sont par défaut affichés, dépendent du framework que vous utilisez. Ici, en version 3.5. Ne vous inquiétiez donc pas si vous avez des using différents des miens. Ce qui suit est plus intéressant. On voit la déclaration d'une classe qui hérite de « Page » et qui déclare une méthode Page_Load. Cette méthode est un exemple *d'event handler.* Celle-ci déclenche l'événement Load de notre page. C'est-à-dire que cette méthode sera exécutée à chaque fois que nous nous rechargerons la page. « Load » en anglais signifiant « Chargement ».

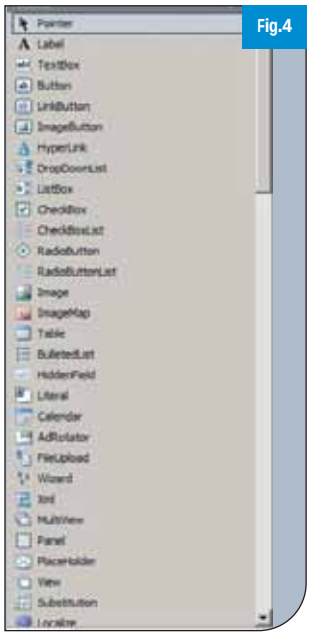


Fig.4

Il existe des événements qui sont déclenchés au chargement de la page que nous verrons un peu plus loin dans cet article (cf. http://www.loicbar.com/post/HTTP-Applications-et-HTTP-Modules.aspx).
Outre toutes les balises HTML que vous connaissez, ASP.NET vous propose des contrôles qui vont simplifier l'écriture de vos applications web. Ces contrôles commencent tous par « <asp : ». On retrouvera parmi les plus utilisé :

- Le label
- Le GridView
- Le Repeater
- Le Literal
- ...

Vous pouvez retrouver la liste entière de ces contrôles dans la « toolbox » de votre Visual Studio / Web Developer. [Fig.4]
Pour insérer l'un de ces contrôles dans votre page, sélectionnez-le dans votre « toolbox » et déplacez-le sur votre page.

```
<%@ Page Language="C#" AutoEventWireup="true"  CodeFile="Default.
aspx.cs" Inherits="_Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
       <asp:Label ID="Label1" runat="server" Text=" Mon premier
 label"></asp:Label>
    </div>
    </form>
</body>
</html>
```

Nous avons par exemple ajouté un label à notre page. Vous remarquez la propriété runat="server" qui permet d'indiquer que ce contrôle devra être évalué au niveau du serveur.
Si vous lancez la page de votre explorateur, vous verrez que notre *asp :label)* a été transformé en une balise *span* :

```
<span id="Label1">Mon premier label</span>
```

Que s'est- il passé ? Le compilateur à vu un contrôle qui devait être interprété par le serveur, celui-ci l'a traité pour le remplacer par du HTML, compréhensible par le navigateur. Dans ce cas-ci, l'utilisation d'un contrôle ne s'avère pas utile. L'utilité vient lorsque vous traitez ce contrôle dans le code behind. Vous avez en effet la possibilité d'éditer les propriétés du label en code behind (Text, etc.) :

```
protected void Page_Load(object sender, EventArgs e)
{
    Label1.Text = "Ce label a été traité dans le code behind";
}
```

Ce qui donnera le résultat suivant :

```
<span id="Label1">Ce label a été traité dans le code behind</span>
```

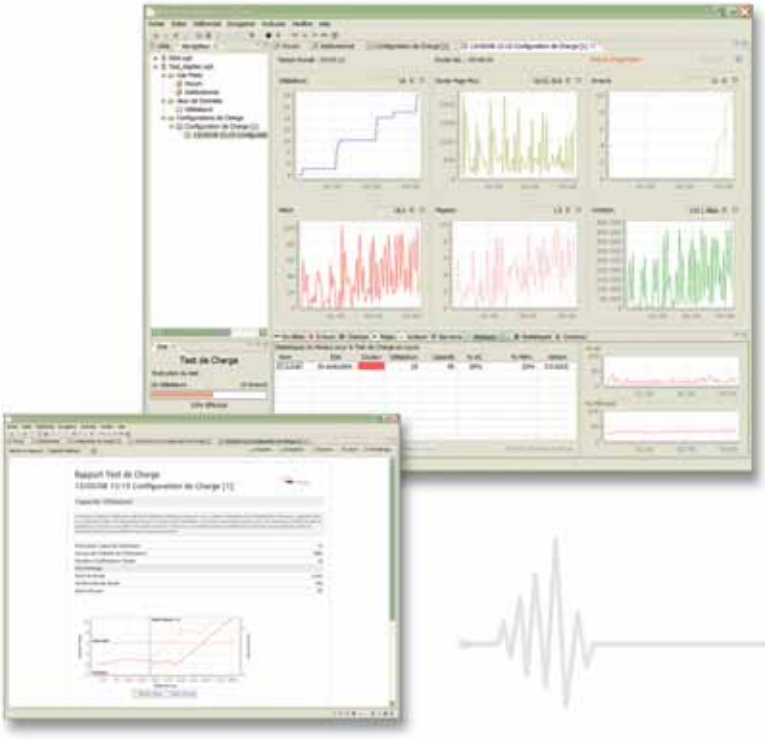
Note : ce que vous écrivez dans le code behind pour la propriété « Text » prend le dessus sur ce que vous avez écrit dans le design de la page, car l'événement Load de la page passe après le traitement du « render » de la page. La propriété de votre label est, dans un premier temps, égale à « Mon premier label » mais est ensuite réécrite au moment du Load.

Le web.config

Le Web.config est un fichier que vous retrouvez à la racine de votre application et qui contient toutes les informations sur la configuration de votre application. Si vous ouvrez le Web.config de votre application web, vous trouverez de nombreuses informations. Détaillons les plus importantes à prendre en compte.

webperformance SUITE

Le Meilleur du Test de Performance Web en toute simplicité et au meilleur rapport qualité-prix



- **Développement automatisé de Cas-Tests et Test de Charge**
- **Rapports d'analyse des Tests de performance et de charge**
- **Analyse des Serveurs et de l'impact sur les performances**
- **nouveau Interface utilisateur et Rapports en Français**

Découvrez comment Web Performance Suite peut faire gagner vos applications Web en Qualité, Fiabilité et Performance

Evaluation gratuite de Web Performance Suite sur http://www.kapitec.com/Pub/WP?id=116

Bénéficiez de notre assistance technique pendant les 15 jours d'évaluation.

kapitec Software

Power Your Web Projects

Tél.: 05 34 27 90 03
sales@kapitec.com

La déclaration des éléments qui se trouveront dans le Web.config :

```
<configSections>
  <sectionGroup name="system.web.extensions" type="System.Web.Configuration.SystemWebExtensionsSectionGroup, System.Web.Extensions, Version=3.5.0.0, Culture=neutral, PublicKeyToken=31BF3856AD364E35">
    <sectionGroup name="scripting" type="System.Web.Configuration.ScriptingSectionGroup, System.Web.Extensions, Version=3.5.0.0, Culture=neutral, PublicKeyToken=31BF3856AD364E35">
      <section name="scriptResourceHandler" type="System.Web.Configuration.ScriptingScriptResourceHandlerSection, System.Web.Extensions, Version=3.5.0.0, Culture=neutral, PublicKeyToken=31BF3856AD364E35" requirePermission="false" allowDefinition="MachineToApplication"/>
      ...
      </sectionGroup>
    </sectionGroup>
  </sectionGroup>
</configSections>
```

Les différentes libraires dont a besoin notre application :

```
<assemblies>
  <add assembly="System.Core, Version=3.5.0.0, Culture=neutral, PublicKeyToken=B77A5C561934E089"/>
  <add assembly="System.Web.Extensions, Version=3.5.0.0, Culture=neutral, PublicKeyToken=31BF3856AD364E35"/>
  <add assembly="System.Data.DataSetExtensions, Version=3.5.0.0, Culture=neutral, PublicKeyToken=B77A5C561934E089"/>
</assemblies>
```

La déclaration des contrôles ASP.NET :

```
<pages>
        <controls>
          <add tagPrefix="asp" namespace="System.Web.UI" assembly="System.Web.Extensions, Version=3.5.0.0, Culture=neutral, PublicKeyToken=31BF3856AD364E35"/>
          <add tagPrefix="asp" namespace="System.Web.UI.WebControls" assembly="System.Web.Extensions, Version=3.5.0.0, Culture=neutral, PublicKeyToken=31BF3856AD364E35"/>
        </controls>
      </pages>
```

C'est dans cette section que nous ajouterons nos propres contrôles.

La déclaration de nos chaînes de connexion vers une base de données :

```
<connectionStrings>
  <add name="LimbourgBDConnectionString" connectionString="Data Source=.\SQLEXPRESS;AttachDbFilename=|DataDirectory|\ProgrammezBD.mdf;Integrated Security=True;User Instance=True" providerName="System.Data.SqlClient"/>
</connectionStrings>
```

Les contrôles ASP.NET

Microsoft fournit de nombreux contrôles que vous pouvez utiliser pour enrichir vos applications web. On peut répertorier ces contrôles en plusieurs catégories.

Les contrôles standard

Ces contrôles vous permettent d'afficher des éléments comme des boutons, des labels, des champs de texte etc. On pourra par exemple facilement créer un formulaire simple à l'aide de ces contrôles :

```
<asp:Label ID="lbNom" runat="server" Text="Nom"></asp:Label> :
        <asp:TextBox ID="tbNom" runat="server"></asp:TextBox> <br />
       <asp:Label ID="lbPrenom" runat="server" Text="Prenom"></asp:Label> :
       <asp:TextBox ID="tbPrenom" runat="server"></asp:TextBox> <br />
        <asp:Label ID="lbEmail" runat="server" Text="Email"></asp:Label> :
        <asp:TextBox ID="tbEmail" runat="server"></asp:TextBox> <br />
        <asp:Label ID="lb" runat="server" Text="J'ai plus de 18ans"></asp:Label> :
       <asp:CheckBox ID="cbPLusDe18ans" Text="" runat="server" /><br />
         <asp:Button ID="btGo" runat="server" Text="Envoyer" />
```

Ce qui affichera ceci sur votre navigateur internet : [Fig.5]

On peut regarder ce que cela gère au niveau du code source :

```
<span id="lbNom">Nom</span> :
<input name="tbNom" type="text" id="tbNom" /> <br />
<span id="lbPrenom">Prenom</span> :
<input name="tbPrenom" type="text" id="tbPrenom" /> <br />
<span id="lbEmail">Email</span> :
<input name="tbEmail" type="text" id="tbEmail" /> <br />
<span id="lb">J'ai plus de 18 ans</span> :
<input id="cbPLusDe18ans" type="checkbox" name="cbPLusDe18ans" /><br />
<input type="submit" name="btGo" value="Envoyer" id="btGo" />
```

Les contrôles de validation

ASP.NET vous fournit un ensemble de contrôles qui vous permettront d'effectuer des vérifications sur les données entrées par les utilisateurs. On peut donc améliorer notre formulaire ci-dessus en ajoutant un contrôle pour vérifier si l'email n'est pas nul.

```
<asp:Label ID="lbNom" runat="server" Text="Nom"></asp:Label> :
<asp:TextBox ID="tbNom" runat="server"></asp:TextBox> <br />
<asp:Label ID="lbPrenom" runat="server" Text="Prenom">
</asp:Label> :
<asp:TextBox ID="tbPrenom" runat="server"></asp:TextBox> <br />
<asp:Label ID="lbEmail" runat="server" Text="Email">
</asp:Label> :
<asp:TextBox ID="tbEmail" runat="server"></asp:TextBox>
<asp:RequiredFieldValidator
    ID="ValEmail" ControlToValidate="tbEmail" runat="server"
ErrorMessage="Email Requis"></asp:RequiredFieldValidator> <br />
<asp:Label ID="lb" runat="server" Text="J'ai plus de 18 ans">
</asp:Label> :
<asp:CheckBox ID="cbPLusDe18ans" Text="" runat="server" /><br />
<asp:Button ID="btGo" runat="server" Text="Envoyer" />
```

Il existe plusieurs contrôles de ce type :

- RequiredFieldValidator
- RangeValidator
- RegularExpressionValidator

• CompareValidator
• CustomValidator
Vous pouvez également créer vos propres contrôles de validation. Pour plus d'information, consultez les liens ci-dessous :
• http://www.loicbar.com/post/Cr%c3%a9er-son-propre-contr%c3%b4le-de-Validation.aspx
• http://www.loicbar.com/post/Cr%c3%a9er-un-AjaxValidator.aspx

Les contrôles riches

ASP.NET vous propose quelques contrôles avec une réelle plus value. Par exemple, créer un calendrier dynamique en ASP.NET est vraiment très simple. Il existe en fait un contrôle asp :calendar qui vous permet d'afficher un calendrier dynamique. Dans ce type de contrôles, on retrouve aussi la possibilité de créer des assistants très facilement (ex : création de formulaires en plusieurs étapes). Vous pouvez aussi installer et utiliser des composants externes pour améliorer votre interface, (voir le dossier *Composants* dans ce numéro !

Les contrôles de données

Les contrôles de données permettent de très facilement aller rechercher des données dans une base de données ou encore un fichier XML. Vous utiliserez ces contrôles pour tout ce qui concerne la visualisation, l'ajout, la modification ou la suppression des données. Vous pourrez par exemple très facilement afficher les données dans un tableau à l'aide d'un GridView et un DataSource (ObjectDataSource, SQLDataSource ou XMLDataSource en fonction de la source des données).

Note : Le framework 3.5 apporte de nombreuses nouveautés avec l'ajout du ListView, du DataPager et du LINQDataSource. Plus d'information à ce sujet sur mon ancien blog : http://loicbar.com/post/ListView-et-DataPager.aspx

Ci-dessous un exemple d'utilisation du contrôle « repeater » et SQL-DataSource :

```
<asp:Repeater ID="Repeater3" runat="server" DataSourceID="DSCC">
    <ItemTemplate>
        <h1><%# Eval("Prenom") %> <%# Eval("Nom") %></h1>
       <p><%# Eval("Titre") %> - <%# Eval("Biographie")%></p>
    </ItemTemplate>
</asp:Repeater>
<asp:SqlDataSource ID="DSCC" runat="server"
   ConnectionString="<%$ ConnectionStrings:LimbourgBDConnectionString %>"
    SelectCommand="SELECT TOP 3 [PersonneId], [Nom], [Prenom],
[Biographie], [Titre] FROM [Personne] WHERE ([PersonneId] = @PersonneId)">
        <SelectParameters>
        <asp:QueryStringParameter DefaultValue="1" Name="PersonneId"
                QueryStringField="id" Type="Int32" />
        </SelectParameters>
</asp:SqlDataSource>
```

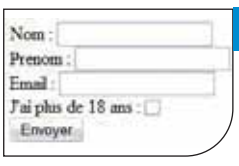


Fig.5

Fig.6

Fig.8

Fig.9

Ces quelques lignes vont permettre d'afficher une liste de personnes. « Eval » permet de mapper un élément avec un champ de la base de données. Le repeater est un peu comme une boucle qui parcourt tous les éléments de la source de données passée comme valeur de la propriété « DataSourceID ». Ici la source de données contient les résultats de la requête SQL passée comme valeur à la propriété « SelectCommand » :
SELECT TOP 3 [PersonneId], [Nom], [Prenom], [Biographie], [Titre] FROM [Personne] WHERE ([PersonneId] = @PersonneId)
@PersonneId désigne un paramètre que nous définissons juste après :

```
<SelectParameters>
    <asp:QueryStringParameter DefaultValue="1" Name="PersonneId"
        QueryStringField="id" Type="Int32" />
</SelectParameters>
```

On indique ici que le paramètre doit être trouvé dans l'url de la requête. On va en fait chercher le paramètre « id » de l'url. S'il n'existe pas, on prend par défaut la valeur « 1 ».
Pour vous simplifier l'écriture de ce code, Visual Studio vous fournit un assistant de paramétrage d'un DataSource. Pour l'utiliser, rendez-vous en mode « design » et cliquez sur le petit icône en haut à droite de votre SQLDataSource. [Fig.6]
Cliquez ensuite sur « Configure DataSource ». Une nouvelle fenêtre apparaît où vous avez la possibilité de définir votre chaîne de connexion à votre base de données. Celle-ci peut être définie en cliquant sur « New Connection » et en remplissant les différents champs de connexion (nom du serveur, login, mot de passe, base de données). En cliquant sur suivant, vous avez là un utilitaire qui vous permet de créer votre requête. [Fig.7]
Cliquez ensuite sur suivant, puis sur terminer. Votre « DataSource » est maintenant configuré. Pour en savoir plus sur SQLDataSource, consultez le lien suivant : http://msdn.microsoft.com/fr-fr/library/system.web.ui.webcontrols.sqldatasource.aspx

Les contrôles de navigation

Ces contrôles vous permettent d'afficher des éléments de navigation standard comme des menus, des arbres d'éléments. On peut en effet, très facilement afficher un arbre d'éléments avec le contrôle *asp :TreeView :*

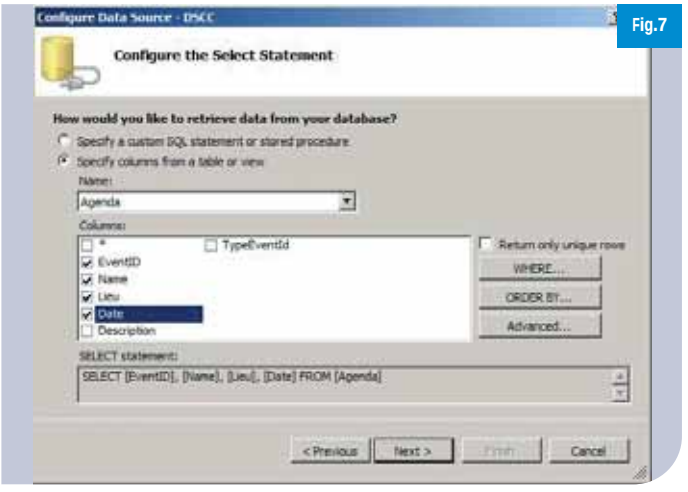


Fig.7

```
<asp:TreeView ID="TreeView1" runat="server">
<Nodes>
    <asp:TreeNode Text="Ordinateur">
       <asp:TreeNode Text="Favory">
         <asp:TreeNode Text="News">
         <asp:TreeNode Text="MSN" NavigateUrl="http://www.msn.com"/>
           <asp:TreeNode Text="MSNBC News" NavigateUrl="http:
//www.msnbc.msn.com"/>
          </asp:TreeNode>
        </asp:TreeNode>
    </asp:TreeNode>
</Nodes>
</asp:TreeView>
```

Pour donner ceci : [Fig.8]

Les contrôles de Login

Ces contrôles sont très souvent utilisés pour gérer tout ce qui est « gestion utilisateur ». Vous pouvez très facilement gérer des comptes utilisateurs avec ASP.NET. Celui-ci vous fournit les contrôles pour les actions suivantes :

- Login d'un utilisateur
- Affichage du nom de l'utilisateur actuellement connecté
- Enregistrement d'un utilisateur
- Perte de mots de passe
- Changement de mot de passe
- Affichage d'éléments en fonction du statut de l'utilisateur (connecté, non connecté, dans un groupe, ou pas)
- Gestion de rôles d'utilisateur (administrateur, membre, etc.).

Pour permettre l'utilisation de ces composants, vous devez les régler à l'aide de l'administration centrale de votre site ASP.NET : [Fig.8]

Dans l'onglet sécurité, vous pouvez choisir d'utiliser *l'ASP.NET MemberShip Provider* en cliquant sur le lien « Sélectionnez le type d'authentification ». Sélectionnez alors « Par internet ». Pour plus d'information sur les composants login, consulter le lien suivant : http://quickstarts.asp.net/QuickStartv20/aspnet/doc/ctrlref/login/default.aspx

Pour pouvoir configurer correctement votre base de données SQL avec SQL MemberShip Provider : http://www.loicbar.com/post/QUIZZ!-Comment-utiliser-SQL-Server-pour-les-composants-login.aspx

Les contrôles WebParts

Les WebParts d'ASP.NET vont nous permettre de créer des applications web de type portails. Vous surfez sur ce type de site très souvent, par exemple en vous rendant sur Yahoo!, My Msn et d'autres… Si vous venez de l'univers Java, vous étiez habitués au portlet. Plus d'information :

- http://www.loicbar.com/post/Introduction-aux-WebParts-ASPNET-(1).aspx
- http://www.loicbar.com/post/Introduction-aux-WebParts-(2).aspx
- http://www.loicbar.com/post/Introduction-aux-WebParts-(3).aspx
- http://www.loicbar.com/post/Introduction-aux-WebParts-(4).aspx

Les contrôles HTML

Vous pouvez utiliser les contrôles HTML dont vous avez l'habitude et interagir avec eux dans le code behind en leur ajoutant la propriété runat=server : Entrez votre nom :

```
<input id="Name" type=text size=40 runat=server>
```

Vous pouvez alors modifier les propriétés de ces contrôles dans le code behind : Name.Value = "ici";

Lorsqu'ASP.NET voit un tag HTML avec la propriété runat, il sait qu'il a affaire à un contrôle ASP.NET. Par exemple, ici, il sait que l'input est un HtmlInputText.

Plus d'informations :

- http://quickstarts.asp.net/QuickStartv20/aspnet/doc/ctrlref/html/default.aspx

PostBack et ViewState

Une petite introduction à ce qui caractérise le modèle de programmation ASP.NET. Revenons sur notre exemple de formulaire. Lorsque nous voulons envoyer les informations de notre formulaire, nous devons gérer un événement sur le clic du bouton. Pour cela, nous devons lui déclarer un handler :

```
<asp:Button ID="btGo" runat="server" Text="Envoyer" onclick=
"btGo_Click" />
```

Lorsque vous cliquez sur le bouton, il y a alors un Postback ou encore une requête « POST » sur la page en cours. Vous devez gérer cet événement dans le code behind sur la méthode « btGo_Click » :

```
protected void btGo_Click(object sender, EventArgs e)
{
    // Traitement ici
}
```

En HTML c'est l'élément form qui permet d'envoyer une requête POST (ou GET). Si on regarde le code source de notre page on remarque que notre « form runat=server » s'est bien transformé en formulaire :

```
<form name="form1" method="post" action="Default.aspx" onsubmit
="javascript:return WebForm_OnSubmit();" id="form1">
    ….
</form>
```

On remarque que l'action du formulaire est bien la page actuelle. Le JavaScript est là car il y a des choses à vérifier après l'envoi du formulaire (à cause de notre validator).

On remarque également un *input* appelé *ViewState* :

```
<input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE" value
="/wEPDwUKLTcwNjkwMjI1M2QYAQUeX19Db250cm9sc1JlcXVpcmVQb3N0QmF
ja0tleV9fFgEFDWNiUEx1c0RlMThhbnOWq/eV0Z6qaaIUowq3lP+t4xSruA==" />
```

Ce champ *hidden* est géré automatiquement par ASP.NET pour sauvegarder l'état des différents contrôles de la page. C'est grâce à cela que, lorsque vous effectuez un PostBack, vos informations sont toujours visibles dans le formulaire. Vous n'avez donc pas besoin de sauver vous-même l'état des différents éléments de votre page en ASP.NET.

Nous espérons vous avoir donné ou redonné envie de plonger dans le développement web avec ASP.Net et les futures évolutions du langage, la possibilité d'utiliser Ajax ou encore ASP.Net MVC. Rien que du bon pour notre plate-forme préférée !

■ Loïc **Bar** – *MVP et CEO de Wipus* - http://blogs.developpeur.org/loicbar/default.aspx

jQuery & ASP.NET : un jeu d'enfant !

Les outils de manipulation du DOM (Document Object Model) ne manquent pas mais aucun n'atteint l'efficacité de jQuery. J'en veux pour preuve son extrême popularité. Celle-ci n'a pas échappé à Microsoft qui l'intègre désormais dans Visual Studio. Mais voyons ce que cette bibliothèque javascript apporte à ASP.Net.

Asp.Net souffre de manques laissés intentionnellement par Microsoft. Les composants natifs étant en effet basiques, on peut tout faire avec (l'essentiel est là, il suffit de l'étendre) mais à l'usage, pour les customiser, on doit entrer dans du code et des optimisations pas toujours évidents. Des éditeurs se sont engouffrés dans la brèche, mais leurs composants sont propriétaires et parfois chers. Avec jQuery, il est simple de trouver ou de développer des composants légers et efficaces. Ils sont avant tout en html / javascript et il est simple de les incorporer dans des composants ASP.Net. De plus, la communauté est nombreuse et active ! jQuery est une pure librairie javascript pour l'exécution sur navigateur web et elle tient en un fichier : jQuery-1.2.6.js (dernier en date). Rien d'autre n'est nécessaire pour commencer. On peut par ailleurs travailler sous Visual Studio le fichier html et les scripts javascript qu'il contient. Si on aime Visual Studio c'est aussi pour l'intellisense et le Ctrl+Espace. Qu'à cela ne tienne : Microsoft a prévu un patch qui permet d'obtenir l'intellisense sur les scripts javascript, dont la librairie jQuery. [Fig.1]

Mon premier composant Jquery

En html

Voici un composant qui pourrait s'avérer utile dans une RIA de gestion. Un tableau extensible, dont certaines cellules sont éditables, afin d'éviter l'éternel écran " maître / détails ". [Fig.2]

Maintenant, pour ceux qui connaissent javascript, imaginez le temps qu'il faudrait pour le réaliser en pur javascript et notamment avec des " document.getElementByld() " ! Eh bien avec l'aide de jQuery, comptez... une petite heure.

Première étape, le téléchargement du fichier javascript de la librairie, afin de le référencer dans le html :

```
<script type="text/javascript" src="./script/jquery-1.2.6.js"></script>
```

Le fichier doit être visible par l'explorateur au même titre que la page elle-même : [Fig.3]

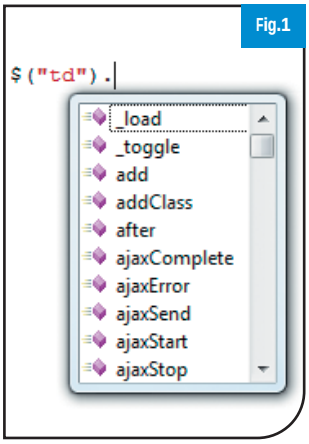


Fig.1

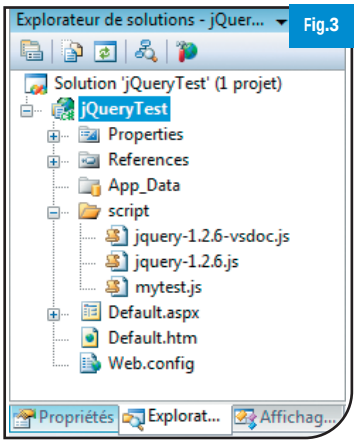


Fig.3

De même la présence du fichier secondaire jQuery...-vsdoc.js assure l'intellisense dans le designer. Seconde étape : l'écriture coordonnée du html et de la feuille de style.

```
<style type="text/css">
    td.edited { background-color:#999; }
    td.editable { color:#66F; }
    .../...
    tr.subTitle { font-weight:bolder; }
    tr.details { empty-cells:show; }
</style>
<body>
.../...
<table class="interactiveTable">
    <tr class="title"><td colspan="4">>> Group 1</td></tr>
    <tr class="subTitle">
        <td>Elements</td>
        <td>Editable details</td>
        <td>Editable details 2</td>
    </tr>
    <tr class="details">
        <td>Element 1:1</td>
        <td class='editable'>Details on 1:1</td>
        <td class='editable'>Details on 1:1</td>
```

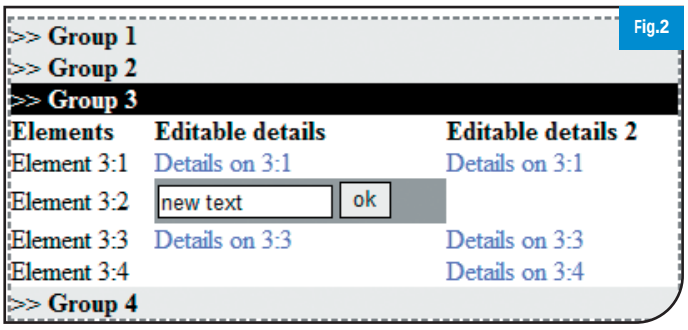
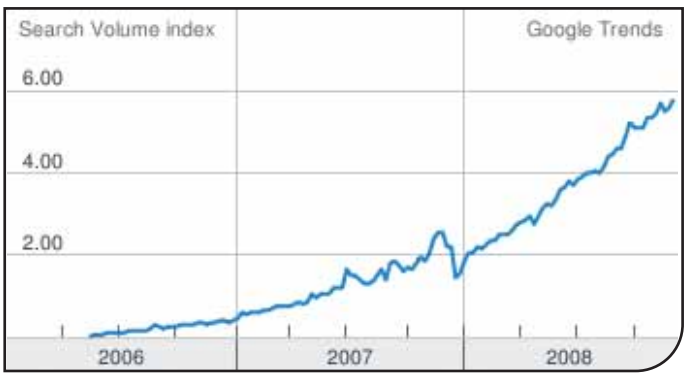


Fig.2



```
    </tr>
.../...
</table>
</body>
```

L'important dans le code html est que s'il est bien écrit à la base, il doit suffire, sans réécriture spécifique. Si le tableau est bien structuré, le script n'a plus qu'à cacher certaines lignes, rendre certaines cellules éditables, gérer en somme toute la partie dynamique. Pas de tag " magique " en plein milieu du html, juste du html. jQuery est non intrusif. Troisième et dernière étape : le script :

```
<head>
.../...
<script language="javascript" type="text/javascript">
// <!CDATA[
    $(document).ready(
    function() {
    .../...
    });
// ]]>
</script>
</head>
```

Là où des notions de javascript ne sont pas inutiles, c'est pour bien assimiler la nécessité d'imbriquer les fonctions les unes dans les autres pour le bon fonctionnement de l'ensemble. Ici par exemple l'ensemble des fonctions faisant appel à jQuery sont écrites dans la déclaration du gestionnaire d'événement " ready ". L'événement correspondant à la fin de chargement de la page, on s'assure ainsi que les scripts ne tentent pas d'interagir avec des parties non encore chargées. De la même manière, si l'on veut gérer le fait qu'un clic sur une cellule la transforme en zone de texte avec bouton et que le clic sur le bouton met à jour la cellule avec la nouvelle valeur, il va falloir inclure la déclaration du gestionnaire d'événement clic du bouton dans celle du clic de la cellule, afin que les boutons créés à la volée soient pris en compte. C'est une petite gymnastique à intégrer, qui devient vite assez exaltante. Attention en revanche à la lisibilité du code. Autre notion très pratique et particulière à jQuery, l'ensemble des fonctions de manipulation renvoie l'élément manipulé, ce qui permet le chaînage des expressions :

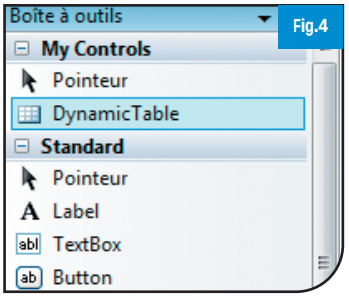


Fig.4

```
$("tr[@class=titleSelected]").removeClass("titleSelected").ad
dClass("title");
```

Ici par exemple on change la classe css d'un ensemble de lignes du tableau.

En ASP.net

Admettons à présent que l'on veuille intégrer tout cela dans un userControl Asp.Net afin d'en disposer en mode design de façon intégrée. De cette manière on pourra lier le tableau à une source de données, web-service, ADO.Net que sais-je encore. Rien de plus simple. On a deux possibilités, soit on reprend l'ensemble du code html du tableau dans le userControl, soit plus élégamment, on remplace le tableau par un gridView ou autre composant Asp.Net. Dans ce dernier cas, on reprend simplement les classes css.

La démarche inverse peut-être adoptée, à savoir que l'on peut aussi bien partir d'une page Asp.Net intégrant une somme de composants et lui ajouter une couche de vernis jQuery, de la même manière que l'on ajuste les feuilles de style après coup.

Entre les deux, un bon plan peut-être de reprendre quelques-uns des composants jQuery UI disponibles, de les adapter aux composants Asp.Net du type grid, Calendar, TreeView etc. et ainsi de monter à moindres frais une bibliothèque de composants réellement originaux. Par ailleurs, jQuery s'assemble très bien avec le framework Asp.Net MVC. Dans tous les cas, le marché est ouvert, on devrait voir apparaître très bientôt de nombreux composants mariant les deux mondes. [Fig.4]

CONCLUSION

Finalement quand on aborde jQuery, c'est la simplicité d'utilisation qui vient d'abord à l'esprit. Mais sa force est dans son caractère autonome et non intrusif. Si l'on ajoute son orientation composants et l'adhésion incroyable qu'il génère, il devient clairement incontournable. Et il peut devenir un vrai plus pour vos développements Asp.Net. En revanche, pour la plupart des composants disponibles, le simple copier / coller ne fonctionne pas tel-quel. Il y a souvent un peu de code à retoucher mais c'est la plupart du temps du javascript et rarement très complexe. Alors, si le javascript ne vous fait pas peur, que vous ne craignez pas de remettre à l'occasion les mains dans le cambouis, jQuery est à découvrir. Et pour cela, une seule adresse : jQuery.com.

■ Laurent **Capin** – *Consultant SFEIR*
■ Aboubaker **Boufous** – *Ingénieur d'études SFEIR*

Déployer son projet ASP.Net

Il existe plusieurs manières pour déployer un site ASP.Net. La plus rapide est de passer directement par l'environnement de développement (Visual Studio ou édition Visual Web Developer). Il supporte plusieurs protocoles tels que webdav, ftp.

Quand on déploie de l'ASP.NET sur un serveur, celui-ci doit respecter des pré-requis techniques. C'est notamment sensible dans l'utilisation de serveurs mutualisés chez un hébergeur sur lequel le développeur n'a pas de " remote access ". Pour cela, on peut utiliser l'outil Web Platform Installer pour vérifier le serveur. D'autre part, avec IIS 7 (le serveur web de Microsoft), la configuration se veut plus simple avec un simple web.config. Il n'y a plus besoin de fichiers MSI, ni de description. Il est bien sûr possible d'aller plus loin dans l'automatisation du build et du déploiement en passant par un outil de type MS Build. Autre point important à vérifier : le niveau de sécurité du serveur. Si celui-ci est en mode medium trust, il faut que le développement soit sur le même mode ; dans le cas contraire, des problèmes de sécurité apparaîtront ! Enfin, si vous utilisez des extensions comme MVC ou Ajax, il n'y a pas d'élément spécifique à rajouter sur le serveur, les DLL seront présentes lors du déploiement. Sur l'hébergement, pour les professionnels, l'offre se diversifie à un prix abordable comme chez ASP Serveur, Amen.

■ F.T.

Dynamic Data : dynamiser ASP.Net

Dynamic Data est une nouvelle technologie présente dans le service pack 1 de .Net 3.5. Elle fournit une architecture permettant de construire des applications orientées données. Il est donc possible d'afficher dynamiquement des pages basées sur le schéma d'une base de données.

Configuration utilisée pour l'article :
- Visual Studio 2008
- SQL Server 2005
- Framework .Net 3.5 SP1
- La base de données Northwind

Création d'un premier projet Dynamic Data

Ouvrez Visual Studio et cliquez sur : **file > new project,** dans la partie Web choisissez Dynamic Data Web Application et nommez votre projet : DemoDynamicData. Cliquez sur OK. A partir ce cet instant, la structure de votre projet est créée.

Ajoutez à présent une connexion vers la base de données **Northwind** se trouvant sur votre serveur SQL dans le Server Explorer de Visual Studio : [Fig.1].

La prochaine étape consiste à lier votre application à la base de données, pour cela vous allez utiliser la technologie **LINQ.** Ajoutez à la racine de votre projet un composant de type **LINQ To SQL Classes** et appelez-le Northwind.dbml : [Fig.2].

Glissez-déposez ensuite toutes les tables de la base de données depuis le Server Explorer vers votre nouveau fichier [Fig.3].

Une dernière modification dans le fichier **Global.asax** du projet est nécessaire avant de pouvoir exécuter l'application. Ajoutez cette ligne au début de la méthode **RegisterRoutes :**

```
model.RegisterContext(typeof(NorthwindDataContext), new Context
Configuration() { ScaffoldAllTables = true });
```

Compilez et démarrez votre application. Vous venez de créer votre premier projet Dynamic Data en quelques minutes.

Les fonctions fondamentales

Voyons à présent ensemble les fonctionnalités principales qu'offre Dynamic Data, de base, sans nécessité d'aucun développement. La première page affiche la liste des tables présentes dans votre fichier Northwind.dbml. Chaque nom de table est en réalité un lien pointant vers une liste des enregistrements présents dans la table [Fig.4]. Au début de chaque ligne se trouvent trois liens : *Edit, Delete* et *Details*. Le premier permet l'édition de l'enregistrement, cliquez dessus pour modifier l'élément : [Fig.5].

Le deuxième permet la suppression de l'enregistrement, alors que le troisième affiche son détail dans une nouvelle page. Comme vous pouvez le remarquer l'interface est relativement intuitive.

Les relations entre les tables sont également utilisables. Prenez par exemple le cas de la table **Customers,** il est possible d'obtenir toutes les commandes liées à un client précis grâce au lien **Orders**. En mode édition, les colonnes de type " clés étrangères " se transforment en liste déroulante afin de permettre la sélection d'une nouvelle valeur.

Dynamic Data fournit également une notion de filtre. Pour illustrer la manière dont l'application permet de filtrer les données, nous allons utiliser la table **Product.** Les listes déroulantes situées en haut de la liste permettent de filtrer les enregistrements. Par défaut, les tables peuvent être filtrées sur les champs de type booléen et sur les clefs étrangères. Autre point intéressant : l'application vérifie les contraintes de la base de données et avertit l'utilisateur si les données entrées sont incorrectes.

La notion de routage

Dynamic Data est basé sur le modèle MVC. Cela signifie qu'il intègre la notion de routage. Le routage permet de conserver des Url facilement lisibles et compréhensibles par l'utilisateur. Dans le site que nous avons créé précédemment, toutes les tables présentes dans la base de données sont affichées en utilisant le même template de page. Cependant leur Url est différente. Prenez par exemple la page listant tout les produits, son Url est la suivante : http://xxxxx/Products/List.aspx

Concentrez-vous sur la fin de l'Url : Products / List.aspx, deux éléments sont spécifiés : le nom de la table et l'action demandée.

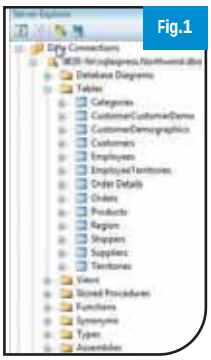


Fig.1


Fig.2

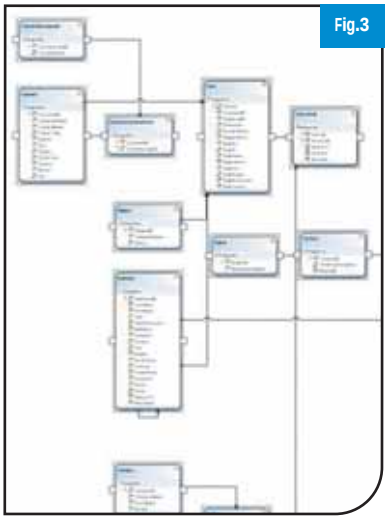


Fig.3


Fig.5


Fig.4

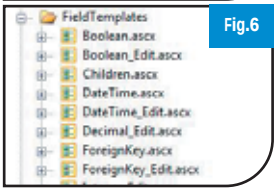


Fig.6

Demandez à présent l'édition d'un élément et regardez la fin de l'Url : /Products/Edit.aspx?ProductID=1
De nouveau la table est spécifiée ainsi que l'action demandée, plus dans ce cas l'ID de l'élément à modifier. Ce format d'url est défini dans le fichier **Global.asax** de votre application :

```
routes.Add(new DynamicDataRoute("{table}/{action}.aspx")
{
      Constraints = new RouteValueDictionary(new { action = "List
|Details|Edit|Insert" }),
       Model = model
 });
```

Ce routage par défaut permet de définir que toutes les actions (List,Detail,Edit et Insert) seront redirigées vers une url de type {table}/{action}. Dans cet exemple, nous allons inverser {table} et {action} :

```
routes.Add(new DynamicDataRoute("{action}/{table}.aspx")
 {
      Constraints = new RouteValueDictionary(new { action = "List
|Details|Edit|Insert" }),
         Model = model
});
```

Affichez à nouveau la liste des produits et remarquez la nouvelle Url : http://xxxxx/List/Products.aspx
Cette fois l'action est placée avant la table. Vous venez de voir les bases du routage, ce concept est beaucoup plus puissant, il est possible par exemple de définir des règles particulières pour certaines tables ou actions. Cet article étant consacré à Dynamic Data et non à MVC, nous allons en rester là pour ce sujet.

Personnalisation du template de page

Nous avons donc vu jusqu'à présent que Dynamic Data générait automatiquement les pages sur la base d'un template de page, essayons par exemple de modifier la page affichant la liste des éléments d'une table, il s'agit de la page **List.aspx.** Elle se trouve à cet endroit : / DynamicData / PageTemplates / List.aspx
Le but de l'opération est d'ajouter un logo sur le haut de la page, commencez par charger votre fichier image dans l'application. Ouvrez ensuite le fichier List.aspx dans Visual Studio et ajoutez cette ligne dans le haut de la page :

```
<asp:Image runat="server" ImageUrl="~/logo.jpg" />
```

Exécutez à nouveau l'application et constatez que toutes les pages de liste ont été modifiées et affichent maintenant le logo.

Personnalisation des templates de champs

Comme pour les pages, les champs peuvent également être modifiés par l'intermédiaire de templates. Les templates de champs se trouvent dans ce répertoire : / DynamicDate / FieldTemplates. Il contient une série de contrôles utilisateurs représentant les différents types de données utilisables. Il y a deux contrôles par type de champ : un premier pour la lecture et un second pour l'édition. Par exemple Text.ascx et Text_Edit.ascx. [Fig.6].

Les validations personnalisées

Il est également possible d'en définir de nouvelles au niveau du site. L'objectif de cet exemple est d'obliger l'utilisateur à entrer un nom de produit d'au moins 5 caractères, dans le cas contraire un message d'erreur devra être affiché. Commencez par ajouter une nouvelle classe à votre projet et nommez la *Product.cs,* cette classe viendra compléter la classe Product générée par le système, n'oubliez donc pas le mot clé partial :

```
public partial class Product{
…
}
```

Nous allons travailler avec l'événement déclenché lors de la mise à jour du nom du produit et renvoyer une exception sur le nom comptant moins de 5 caractères.

```
public partial class Product
    {
        partial void OnProductNameChanging(string value)
        {
            if (value.Length < 5)
                throw new ValidationException("Le nom doit contenir
 au moins 5 caractères");
        }
    }
```

Exécutez l'application et essayez de sauver un nom de produit contenant moins de 5 caractères.
Une autre possibilité existe pour valider les données sauvées, ce sont les attributs, ils sont par exemple très utiles pour valider des données de type numérique. Voyons comment ajouter une validation vérifiant si la quantité en stock est comprise entre 0 et 500, dans le cas contraire nous devons afficher un message d'erreur.
Reprenez le fichier Product.cs créé à l'étape précédente et ajoutez y une nouvelle classe, il s'agit d'une classe de " MetaData "

```
    public class ProductMetaData
    {
        [Range(0,500,ErrorMessage="La quantité doit être comprise
 entre 0 et 500")]
        public Object UnitsInStock;
    }
```

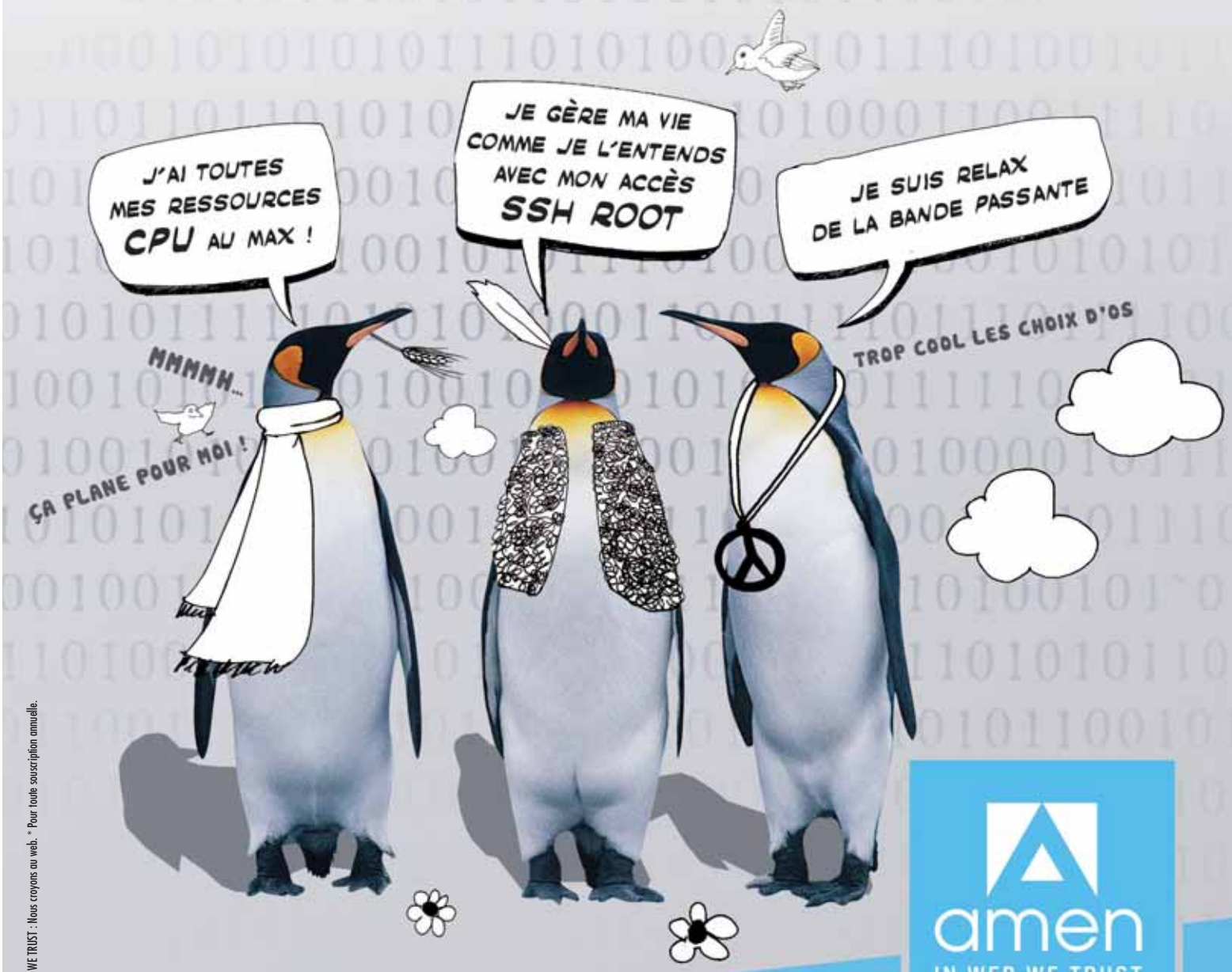
Cette classe contient la définition du champ UnitsInStock, il s'agit du champ que nous désirons valider. Ce champ contient un attribut de type Range, il permet de définir une valeur minimum et maximum pour un champ numérique. Il permet également de définir le message d'erreur si le champ ne peut pas être validé.
Il reste ensuite à lier cette classe avec la classe créée à l'étape précédente, cela se fait également à l'aide d'un attribut :

```
[MetadataType(typeof(ProductMetaData))]
public partial class Product{
…
}
```

Vous pouvez maintenant relancer l'application et tester cette validation. Bonne programmation " dynamique " !

■ Ludovic **LEFORT**
SharePoint Technical Consultant MCP, MCTS & MVP SharePoint
lefortludovic@gmail.com - http://lefortludovic.blogspot.com

NOUVEAU VDS+ d'AMEN :
le bonheur est dans le serveur !



amen
IN WEB WE TRUST

À PARTIR DE
5 € HT /MOIS
soit 5,98 € TTC/MOIS*

SERVEURS PRIVÉS AMEN : BÉNÉFICIEZ DE RESSOURCES GARANTIES QUI VOUS SONT PROPRES (PROCESSEUR, MÉMOIRE, DISQUE DUR...) TOUT EN PROFITANT D'UNE PLATEFORME INFOGÉRÉE 24H/24 - 7J/7.

- Hébergement multi-sites/multi-domaines
- Interface d'administration : Plesk 8.6
- Systèmes d'exploitation : Fedora Core 8, Suse 10.3, Debian 4.0, Ubuntu 8.04 ou CentOS 5
- Part CPU minimum : de 1 à 6
- Mémoire garantie : de 256 Mo à 1 Go
- Espace disque : de 5 Go à 30 Go
- Bases de données : illimitées
- 1 adresse IP fixe
- Accès Root

PARTENAIRE INDUSTRIEL
entrepreneurs, faites le choix de l'économie numérique
www.economienumerique.pme.gouv.fr

Pour plus de renseignements : 0892 55 66 77 (0.34 €/mn) ou www.amen.fr
NOMS DE DOMAINE - EMAIL - HÉBERGEMENT - CRÉATION DE SITE - E-COMMERCE - RÉFÉRENCEMENT

Travailler en freelance

2e partie

Les informaticiens ont la possibilité de travailler en freelance. Parfois subi, plus souvent choisi, ce statut n'est pas toujours bien connu. Nous poursuivons notre enquête dans cette deuxième partie sur cette manière d'exercer le métier d'informaticien.

Samuel Breton, prestataire de création de sites internet, qui a créé sa société, Lexik

Pour qui travaillent les freelances ?

" Il y a deux profils de clients, ceux qui font un appel d'offres et qui cherchent un prestataire pour des projets 'one shot', et que cela ne dérange pas de changer de prestataire pour chaque projet. Et les autres, qui souhaitent être plus accompagnés et rester avec un partenaire de confiance ", observe **Samuel Breton.**

Les freelances s'imposent généralement à leur client par leur niveau de technicité et leur expérience, le côté indépendant n'est pas nécessairement pris en compte par les donneurs d'ordre. L'idéal est de travailler directement avec ses clients, mais en réalité des SSII servent souvent d'intermédiaire entre le freelance et le client. Surtout lorsqu'il s'agit de grandes entreprises qui ont des fournisseurs référencés. *" Je travaille directement avec certains clients depuis plus de 10 ans, avec un pool de jours par an. D'autres fois, je travaille avec une SSII qui répond à un appel d'offres nécessitant une technologie pointue : j'aide la SSII à gagner le projet ",* explique **Sami Jaber.**

Sami Jaber, consultant et architecte spécialisé en .net, qui a créé sa société, DNG Consulting (" Design New Guides " en anglais ou " Développer de Nouveaux Guides ")

Les petites entreprises, voire les particuliers, font volontiers appel aux indépendants. C'est le cas de Nicolas Payelle qui donne des cours d'informatique pour des retraités et conçoit des sites internet pour des entreprises.

Statut et dépendances

Il est possible d'être indépendant et salarié à la fois, en passant par une structure intermédiaire, appelée " portage salarial ". *" Le portage salarial est la meilleure porte d'entrée pour le travail indépendant : pas de déclaration d'entreprise, pas de paperasses, un soutien amical, on s'occupe directement de son premier client - et des suivants. Et quand l'affaire est lancée, on a le choix de rester ou de se mettre à son compte, l'esprit libre ! ",* estime Dany Le Du, auteur de l'ouvrage *Le portage salarial* (cf. Bibliographie).

Toutefois, selon Michel Paysant, *" le portage est marginal pour les informaticiens, car assez cher : une fois retranchées les charges sociales et patronales, il ne reste que 35 à 40% du montant facturé au client, alors qu'avec le statut de travailleur indépendant, il reste environ 50%. "* C'est pourquoi la grande majorité des informaticiens freelance sont à leur compte. Ils ont le statut de travailleur indépendant ou de créateur d'entreprise (Sarl). Ce que confirme Sébastien Develay : *" Les charges plus importantes et les contraintes administratives plus élevées m'ont conduit à opter pour le statut d'indépendant. De plus, la création d'un compte indépendant ne nécessite pas de dépôt de capital. "*

Avantages et inconvénients des réseaux ou sites fédérateurs

Les places de marché, réseaux, annuaires de développeurs et autres sites fédérateurs constituent une opportunité incontournable pour obtenir des clients rapidement et de qualité, une source de contacts et de contrats, fiable, rapide et efficace. Leur intérêt est évident pour de nombreux freelances : *" Je peux enfin me consacrer exclusivement à mon métier : finies les interminables prospections commerciales ",* déclare un développeur, utilisateur de la plate-forme Codeur.com.

Le site fédérateur est surtout utile au début, le temps de se faire une notoriété. En effet, les donneurs d'ordre souhaitent souvent des garanties bien plus élevées avec un indépendant qu'avec une SSII, estimant que le risque est plus grand. Une place de marché permet de réduire ce risque, pris aussi bien par les porteurs de

Serge Roukine, cofondateur de Codeur

Codeur : mettre en relation les donneurs d'ordre et les prestataires

Codeur est une place de marché sur internet, réunissant donneurs d'ordres et prestataires indépendants. Créée à la fin 2006, elle compte environ 4000 porteurs de projets inscrits et 7000 prestataires. Ces derniers sont soit des freelances, soit des entreprises individuelles, soit encore des petites SSII ou des web agencies. Depuis le début, près de 5000 projets ont été déposés sur cette plate-forme. La fréquentation est en constante augmentation, du fait de la notoriété croissante : *" Nous sommes passés d'une quinzaine de projets par jour avant l'été à 25 par jour actuellement ",* indique **Serge Roukine,** cofondateur de Codeur avec Sébastien Peltey. Site : www.codeur.com

projets quant aux compétences et au sérieux des freelances, que les prestataires freelances eux-mêmes, qui n'ont pas toujours la certitude d'être payés. Avantage que certains anciens utilisateurs de ces sites mettent en doute : *" Les sites internet d'appel d'offres représentent souvent des abonnements relativement chers pour des projets/prospects qui ne sont pas toujours suffisamment bien qualifiés. "* Quant aux retours d'annuaires, *" ils ne sont pas en rapport avec nos attentes ",* estime Samuel Breton.

■ Claire **Rémy**

Pour en savoir plus

Bibliographie :

- *S'installer à son compte* (2e édition, 2006), par Michel Paysant, Editions d'Organisation, collection Les Guides du freelance
- *Le portage salarial* (2008), par Dany Le Du, Editions d'Organisation, collection Les Guides du freelance

Sites utiles :

www.freelance-europe.com : Freelance en Europe est un lieu de rencontre et d'initiatives pour la défense et la promotion des freelances.

www.unasa.org : le site de l'UNASA (Union nationale des associations agréées) fournit des données sur l'emploi des TPE et professions libérales en France.

www.auto-entrepreneur.fr : le statut d'auto-entrepreneur (entreprise individuelle) doit entrer en vigueur le 1er janvier 2009 (cf. Loi de modernisation de l'économie n° 2008-776 du 4 août 2008, Titre 1 chapitre I). Ce statut s'adresse en particulier aux personnes qui ne veulent pas nécessairement créer une société commerciale pour exercer leur nouvelle activité et souhaitent pouvoir débuter ou arrêter facilement leur activité indépendante, y compris en étant déjà salarié ou retraité.

David Négrier, associé, directeur technique de The Coding Machine

The Coding Machine : une SSII qui fait travailler des freelances

The Coding Machine (TCM) prend des projets relativement importants, que la société décompose en petits composants, lesquels sont " distribués " entre différents développeurs. Il s'agit la plupart du temps de freelances. Situés sur tous les continents, ils travaillent de chez eux. L'origine du projet remonte à décembre 2003. Alors que les deux membres fondateurs de TCM travaillent sur un projet de création d'entreprise portant sur le conseil Open Source, ils ont l'idée d'adapter à l'entreprise les méthodes utilisées par les communautés du Libre. Parallèlement, ils approfondissent la définition de la méthodologie Open Source pour réaliser des prestations de développement. Aujourd'hui, TCM représente une communauté de 400 développeurs inscrits, dont une quarantaine avec lesquels la société travaille régulièrement. *" L'avantage de travailler avec des freelances, par rapport à une SSII, c'est la possibilité de montée en charge très forte en mettant beaucoup de développeurs en parallèle ",* explique **David Négrier,** associé, directeur technique de TCM. Cela exige toutefois une manière particulière de travailler, pour éviter les surprises : *" Les composants distribués aux freelances doivent être petits (représentant au maximum 10 jours de travail) ; il faut avoir une bonne connaissance de la communauté freelance. Lorsque nous lançons un projet, nous reprenons 3/4 de développeurs connus et 1/4 de nouveaux. "*

Site : www.thecodingmachine.com

Freelance.com : une communauté d'indépendants

Freelance.com propose aux entreprises les services de freelances en informatique. La spécificité de ce réseau est d'offrir à ses clients l'assistance d'un " Manager Conseil ", lui-même indépendant, qui assure l'adéquation entre l'offre et la demande, ainsi que le suivi de la mission. Freelance.com facture directement les clients, qui n'ont donc qu'un interlocuteur. Pour travailler avec ce réseau, le freelance doit impérativement être enregistré en profession libérale ou société ; il doit posséder une solide expérience professionnelle et faire la preuve de sa fiabilité.

www.freelance.com

Enregistrement nécessaire pour obtenir la clé d'activation :
https://registrationcenter.intel.com/BetaCenter/BetaForm.aspx?ProductID=1261&ProgramID=Programmez

Avant-première exclusive !

Intel Parallel Studio : simplifier le développement parallèle !

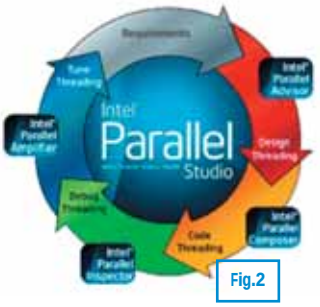
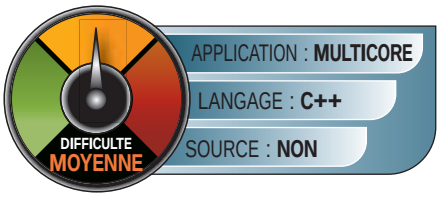


Fig.2

Alors que les processeurs multi-cores (multi-cœurs) prolifèrent, améliorer les performances logicielles, et mieux utiliser la puissance processeur, demeure un enjeu important pour rester compétitif, surtout pour les applications développées en C ou C++. Mais jusqu'à présent le développement parallèle restait complexe à aborder pour le développeur. Programmez ! vous invite à découvrir en avant-première la nouvelle solution Intel, avec un tout nouveau plug-in Visual Studio, qui va radicalement changer votre vision du développement parallèle.



L'année 2009 sera non seulement l'année de la prolifération en masse des processeurs à 4 cœurs (Quad-Core) et le début des processeurs à 8 cœurs (Octo-Core), mais aussi l'année ou le développeur sera de plus en plus confronté aux performances des applications, de son code source sur ces processeurs. Bref, comment tirer parti des performances pour accélérer les applications Windows ? Les techniques de programmation dite parallèle et le " multi-threading " sont difficiles à maîtriser. Elles requièrent souvent un travail intensif d'apprentissage et de structuration et généralement une refonte partielle ou totale du code. Les outils d'aide à l'implémentation du multi-threading disponibles sur le marché sont rares et encore souvent complexes : Intel propose depuis quelques années des outils et des librairies multi-plates-formes pour la programmation sous C et C++. Jusqu'à présent, ces outils étaient surtout réservés aux développeurs de calculs hautes-performances ou à l'imagerie numérique.

PARALLEL STUDIO : LE PREMIER PAS !

Annoncée en août 2008, la collaboration entre Intel et Microsoft va aboutir cette année au lancement d'une nouvelle série d'outils appelée Intel Parallel Studio (prévue au second trimestre). Celle-ci supportera également la future Concurrency Runtime de Microsoft qui permettra finalement d'avoir une infrastructure commune de gestion des ressources cœurs de processeur pour toutes librairies et langages, que ce soit .net ou natif.

Intel Parallel Studio se présente sous la forme de plug-in comprenant 4 modules distincts (et utilisables séparément) qui permettront d'aborder la programmation parallèle directement sous l'environnement Visual Studio. Aujourd'hui, 3 des 4 modules sont disponibles et prêts à être installés en version bêta sur le CD :

1. Parallel Composer
2. Parallel Inspector
3. Parallel Amplifier
4. Parallel Advisor (disponible ultérieurement)

UNE INSTALLATION SIMPLE ET UNE PARFAITE INTÉGRATION

Les 3 modules sont simples à installer à partir d'un installateur commun et s'intègrent parfaitement dans l'environnement Visual Studio 2005 ou 2008. Chaque module peut-être installé séparément. Après installation, les modules sont disponibles soit directement en barre d'outils [Fig.1]soit par le menu "outils " (" tools ").

Partant d'une méthodologie éprouvée de développement parallèle en 4 phases [Fig.2], chaque module s'imbrique dans une de ces phases :

1. Phase **" Design "** (Parallel Advisor), répondant aux questions par où commencer ? quels segments de code bénéficieront le plus du multi-threading ?
2. Phase **" Codage "** (Parallel Composer)
3. Phase **" Debug "** (Parallel Inspector)
4. Phase **" Tuning "** (Parallel Amplifier)

Tous les modules sont utilisables et compatibles avec le compilateur Microsoft Visual C++ (et aussi avec le compilateur Intel C++).

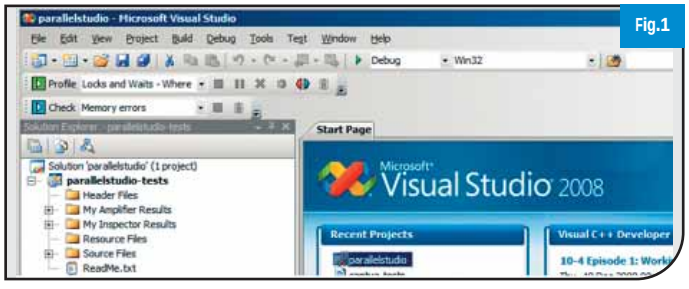


Fig.1

Les 2 barres d'outils pour Inspector (" Check ") et Amplifier (" Profile ")

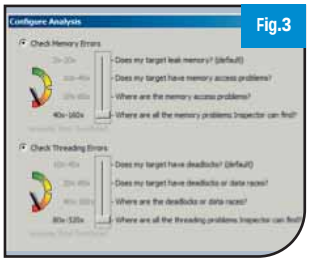


Fig.3

Configuration des analyses du module Inspector


Fig.4

Analyse du niveau de parallélisme ("concurrency analysis")

Parallel Composer complémente Visual Studio avec de nouvelles fonctionnalités et méthodes de multi-threading au niveau applicatif : il comprend un compilateur haute performance C/C++, compatible avec le compilateur Microsoft ainsi que la librairie multimédia Intel Performance Primitives (IPP) et la librairie de multi-threading Threading Building Blocks (TBB). Le compilateur supporte dès à présent le standard OpenMP 3.0, les fonctions *lambda,* ainsi que les fonctions : *autovectorization, auto-parallelization* et *spawn.* La librairie TBB est un élément clé du Composer. Elle offre une nouvelle approche relativement simple du multi-threading basée sur des templates C++, ce qui rend le code aisément portable et très performant.

Parallel Inspector est l'outil le plus innovant du trio (en attendant le quatrième outil Parallel Advisor). En principe, il s'agit d'un debugger spécifiquement adapté aux méthodes de développement parallèle. Le but étant de donner confiance au développeur et que son code ne va pas générer des bugs imprévisibles et destructifs.

Inspector permet de trouver rapidement et sans failles les erreurs classiques et difficiles à détecter du multi-threading : les " race conditions " et les " deadlocks ". Inspector intègre également un analyseur de mémoire très performant pour détecter les erreurs et les fuites de mémoire. L'outil, très flexible, peut être facilement configuré pour varier l'intensité et la durée des analyses, et permettre ainsi de faire des tests rapides ou bien des tests exhaustifs [Fig.3].

Parallel Amplifier est l'outil d'analyse de la performance et se place dans la 4e phase de la méthodologie (phase de " tuning "). L'outil permet de trouver les goulots d'étranglement et de donner des éléments de réponse à la question cruciale du développement parallèle : est-ce que mon code utilise au mieux les ressources et le potentiel de performance du processeur multi-core ?

Les fonctionnalités de Parallel Amplifier sont multiples et se basent principalement sur des analyses de performance en profondeur au niveau du processeur :

1. Analyse “Hotspot” des segments de code gros consommateur de ressources processeurs.
2. Analyse du niveau de parallélisme (“concurrency analysis”) permettant d'une part de repérer les segments de code qui sont parallèles et ceux qui ne le sont pas, et d'autre part d'identifier les segments de code qui sont exécutés de façon sérielle et limitent ainsi la montée en charge (" scalability "), voir [Fig.4].
3. Analyse de “Locks and Wait” pour détecter les objets de synchronisation et de blocage qui consomment le plus de temps d'attente.

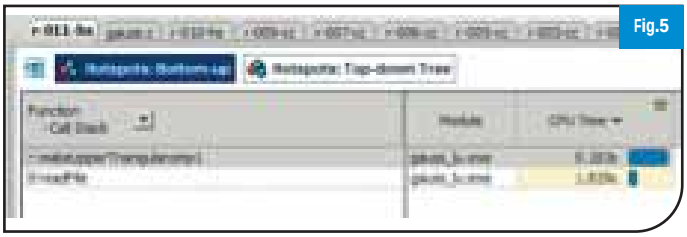


Fig.5

La première fonction crée un " hotspot "

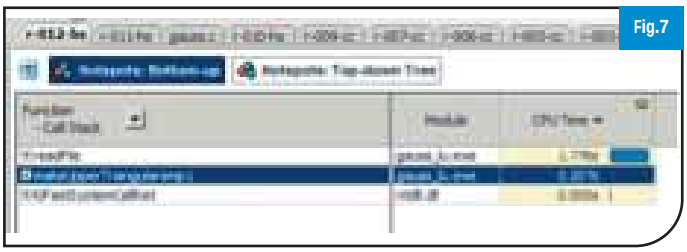


Fig.7

UNE MISE EN ŒUVRE PERMANENTE

L'intérêt principal de Parallel Studio réside dans la simplicité d'utilisation et l'intégration dans Visual Studio, ce qui permet une mise en œuvre permanente tout au long du processus de développement.

EXEMPLE SIMPLE

Prenons l'exemple d'un algorithme de résolution de matrice creuse (en anglais, " sparse matrix solver") utilisant la méthode par élimination de Gauss (méthode LU); un algorithme classique pour résoudre des problèmes d'éléments finis en mécanique.

Après compilation du code en C++, nous utilisons Amplifier pour trouver un " hotspot " éventuel. L'écran de résultats nous montre une utilisation du processeur excessive pour une des fonctions [Fig.5].

Un double-click sur la fonction nous envoie sur les lignes code responsables du " hotspot " : [Fig.6]. Après analyses des matrices, nous changeons le code de la façon suivante :

```
#pragma omp parallel for private (i,j,k, term, temp)

      for (i = 0;i < dimension - 1; i++){
            temp = A[i][i];
            colindex = (i >> 1) << 1;
            for (j = i + 1 ; j < dimension; j++){
                  term = A[j][i] / temp;
                  //for (k = i; k < dimension; k++){
                 for (k = colindex; k < colindex + 2; k++){
                        A[j][k] = A[j][k] - term * A[i][k];
                  }
                  b[j] = b[j] - term * b[i];
            }
      }
```

Le résultat est immédiat : [Fig.7]. Un des grands avantages de Parallel Studio est de permettre des analyses comparatives côte-à-côte [Fig.8]. Cet exemple relativement simple montre la facilité d'utilisation de Parallel Studio dans le quotidien de l'utilisateur de Visual Studio. Le parallélisme va devenir une impérative constante et normale dans le processus de développement logiciel.

Dans les articles prochains nous aborderons les autres outils plus en détail avec des exemples plus complexes.

En attendant, nous vous recommandons d'installer Parallel Studio (ou bien le ou les modules qui vous intéressent) et de commencer immédiatement avec de premières expériences. Plus d'informations sont aussi disponibles en anglais sur le site http://www.go-parallel.com/.


Fig.6

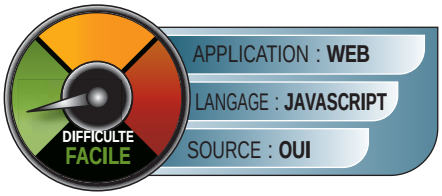
■ Ralph de **Wargny**

Directeur de la division outils de développement d'Intel pour la région Europe centrale et du sud. ralph.wargny@intel.com


Fig.8

jQuery : l'enfant terrible du web 2 !

Avec l'avènement du Web 2.0, le Javascript a fait un retour en force sur le devant de la scène ces dernières années. Puissant, mais original car basé sur la programmation orientée objet par prototype, son utilisation en contexte industriel a rendu nécessaire la création de frameworks regroupant un certain nombre de fonctionnalités utilisées couramment dans ces applications Web, qu'elles soient visuelles (drag and drop, sliders, …) ou techniques (manipulation du DOM, requêtes AJAX, …).



jQuery se présente sous la forme d'un fichier javascript unique pesant moins de 20Ko dans sa version compressée pour environnement de production, ce qui en fait une bibliothèque ultra légère. Elle est compatible avec les principaux navigateurs du marché et permet ainsi de profiter du Javascript sans avoir à se soucier du navigateur client cible. Son utilisation se révèle pertinente dans les cas suivants :

- Manipulation du DOM (Document Object Model)
- Gestion des évènements
- Simplification des requêtes AJAX
- Interfaces graphiques et effets visuels

Elle est en sus assez simple, puisque basée sur un seul objet Javascript ! Il s'agit de l'objet jQuery utilisable via l'alias $(). Enfin, son architecture lui permet d'être facilement étendue via l'emploi de plug-in, présents à foison sur internet, qui viennent parfaitement se greffer sur le noyau de JQuery.

PREMIERS PAS

L'installation de la bibliothèque est triviale puisqu'il suffit de télécharger le fichier javascript de la version que l'on souhaite utiliser et ensuite de l'importer dans son code javascript comme on le fait avec n'importe quel fichier source Javascript habituellement. Notre premier exemple va consister à modifier le DOM d'une page HTML afin d'ajouter un message d'alerte sur l'ensemble des liens de cette dernière. La mise en place de notre code doit se faire une fois que le DOM de notre page est totalement chargé. Pour cela, nous mettons en place un évènement document chargé comme suit :

```
$(document).ready(function(){
     // Traitement effectué lorsque le DOM est chargé
}) ;
```

Le traitement que nous souhaitons réaliser va utiliser un sélecteur JQuery pour récupérer l'ensemble des éléments liens de notre page HTML et placer ensuite une fonction en *callback* pour l'évènement *click* de ces éléments :

```
<html>
<head>
 <script type="text/javascript" src="jQuery.js"></script>
 <script type="text/javascript">
     $(document).ready(function(){
          $("a").click(function(){
               alert($(this).text());
          });
     });
</script>
 </head>
 <body>
     <a href="">Premier lien</a><br />
     <a href="">Deuxième lien</a>
</body>
 </html>
```

Une fois le chargement de cette page réalisé, chaque lien se verra appliqué une fonction *callback* appelée lors de l'évènement *click*. Cette fonction se contente ici d'afficher un message d'alerte reprenant le contenu textuel du lien concerné. Ce traitement n'a bien entendu rien d'extraordinaire et aurait pu être réalisé d'autres façons. Par exemple, via un parcours direct du DOM en utilisant la méthode getElementsByTagName. Cette solution s'avère plus longue que celle proposée ci-dessus. Une autre solution consisterait à placer directement un attribut *onclick* sur les balises lien de notre page HTML ! Ce qui deviendrait vite fastidieux à développer mais également à maintenir, notamment en cas de changement du contenu du message d'alerte …

MANIPULATION AVANCÉE DU DOM

Le paragraphe précédent nous aura permis de mettre en avant le terrain de chasse privilégié de jQuery : la manipulation du DOM. Longtemps considéré comme le domaine de prédilection du framework Prototype, force est de constater que jQuery le surpasse sur ce point désormais ! En effet, la syntaxe utilisée par jQuery se révèle incroyablement puissante puisque proche des standards que sont CSS (2.1 voire même en version 3) et XPath. En outre, la sélection d'éléments du DOM peut se faire en combinant les 2 syntaxes et en utilisant un certain nombre de filtres dont certains sont spécifiques au framework. Ainsi, pour sélectionner les sept premiers paragraphes du quatrième *div* du *div* d'id *container,* on pourra utiliser le sélecteur suivant : $("#container div:eq(4) p:lt(8)"). Cette sélection pouvant s'effectuer via l'utilisation de la fonction *find* de l'objet jQuery pour un résultat identique : $("#container").find("div:eq(4)").find("p:lt(8)"). On peut ainsi constater que chaque méthode de l'objet jQuery retourne elle-même un objet jQuery, ce qui permet de chaîner les appels à ses méthodes. En outre, jQuery propose également les fonctions *filter* et *not* sur cet objet. Alors que *filter* réduit la sélection aux éléments correspondants à l'expression du filtre, la fonction *not* permet le contraire en enlevant tous les éléments correspondants à l'expression passée en entrée. Une fois un ensemble d'éléments du DOM sélectionnés, jQuery offre un certain nombre de possibilités

L'agenda 2009 de cet **informaticien** est déjà bien rempli

Salon RECRUTEMENT Lesjeudis.com

INFORMATIQUE : DEVELOPPEMENT - AMOA - FINANCE

12 février - LILLE

Salle du Gymnase - de 11h à 19h

- 5 mars - NANTES
- 26 mars - GENEVE
- 2 avril - AIX-EN-PROVENCE
- 9 avril - PARIS
- 11 juin - TOULOUSE
- 14 mai - BORDEAUX
- 24 septembre - PARIS
- Octobre - RENNES
- Novembre - LYON

Renseignements et pré-inscription sur **www.lesjeudis.com**

d'interaction. La plupart des possibilités offertes par le DOM en Javascript sont couvertes : cela va de l'ajout de classe ou de propriétés CSS en passant par l'ajout de nœuds pour arriver à l'ajout d'évènements. Pour chaque évènement existant sur une page HTML, comme *onclick, onchange, onsubmit ...*, il existe un équivalent sur l'objet jQuery. L'exemple suivant détaille l'utilisation de certaines de ces fonctionnalités :

```
<html>
<head>
<script type="text/javascript" src="lib/jQuery.js"></script>
<script type="text/javascript">
     $(function(){
          // écriture en rouge du lien contenant programmez
          $("a[href*=programmez]").css("color", "red");
         // parcours des div d'id différent de test et ajout de
 la classe yellow
          $("div[id!=test]").each(function(){
               $(this).addClass("yellow");
          });
         // sélection des éléments de  la liste d'id liste et
 mise en place
          // d'une fonction callback sur l'évènement hover
          $("#liste li").hover(function(){
                $(this).addClass("red");
          }, function(){
                $(this).removeClass("red");
         }).each(function(i){ // modification du contenu des
éléments de la sélection
                $(this).html(i + ' ' + $(this).html());
          });
     });
</script>
 <style>
     #container, #container2{
     width : 50%;
     border : 1px solid black;
     padding : 5px;
     margin-bottom : 20px;
     }
     .yellow{
     background : yellow;
     }
     li.red{
     background : red;
     width : 30%;
     }
 </style>
 </head>
 <body>
     <div id="container"><a href="http://www.programmez.com">
Programmez.com</a><br />
     Programmez ! est le magazine de tous les langages</div>
    <div id="container2"><a href="http://www.developpez.com1"
>Developpez.com</a><br />
     Developpez.com le site de tous les langages</div>
     <div id="test">
          <ul id="liste">
               <li>First</li>
               <li>Second</li>
               <li>Third</li>
               <li>Fourth</li>
               <li>Fifth</li>
          </ul>
     </div>
</body>
</html>
```

La page HTML de notre exemple possède 3 éléments *div* dont le dernier contient une liste de valeurs. Une fois le DOM de la page chargé, on va mettre en place une fonction que l'on passe en entrée de l'alias $(). Cette syntaxe est équivalente à celle du premier exemple qui, elle, utilisait la méthode *ready* sur le sélecteur document. Notre fonction va tout d'abord sélectionner tous les liens de la page dont l'attribut *href* contient la chaîne de caractères "programmez" via une requête XPath et appliquer ensuite sur cet élément une nouvelle propriété CSS pour modifier sa couleur d'écriture. Ensuite, on sélectionne l'ensemble des éléments *div* ayant un *id* différent de test et on utilise la méthode *each* sur l'objet jQuery retourné. Cette dernière va permettre d'appliquer une fonction sur chacun des éléments de l'objet sur lequel on l'applique. La fonction appliquée se charge d'ajouter la classe yellow aux éléments parcourus, ce qui aura pour effet pratique la mise en jaune de l'arrière-plan des *div* sélectionnés. Enfin, le dernier traitement de la fonction s'intéresse aux éléments de la liste en ajoutant 2 fonctions callback sur l'évènement *hover* de ses éléments. Ces dernières sont appelées respectivement lorsque la souris de l'utilisateur passe au-dessus de l'un d'entre eux et lorsque cette dernière quitte la zone d'un élément de la liste. Le but ici étant de mettre en rouge l'arrière-plan de l'élément de la liste survolé par la souris. L'affichage de cette page au sein d'un navigateur est présenté à la figure 1.

REQUÊTES AJAX

Outre ses possibilités en matière de manipulation du DOM, jQuery amène également sont lot de fonctionnalités pour faciliter l'utilisation de requêtes AJAX. La méthode principale à utiliser se nomme *ajax* et doit être appelée sur l'alias $ comme suit : $.ajax(). Son utilisation évite de se soucier des problèmes de compatibilité inter-navigateurs et est configurable rapidement via quelques paramètres simples. Ainsi, le chargement d'un fichier test.htm se fait de la manière suivante :

```
$.ajax({
     type : "GET",
     url : "test.htm",
     error : function(msg){
          alert("Erreur au chargement  : " + msg);
     },
     success : function(data){
          // Traitement des données représentées par data
     }
});
```

Ici, la requête est de type GET et on met en place 2 fonctions *callback* qui seront appelées en cas d'erreur / succès de la requête. En cas de succès, la fonction *callback* permet de traiter via la variable *data* les données récupérées suite à la requête AJAX. Les paramètres de la méthode ajax sont nombreux et couvrent la plupart des besoins du développeur dans la mise en place de requêtes AJAX.

D'autre part, l'objet jQuery met à votre disposition d'autres méthodes couvrant des besoins plus spécifiques. Leur emploi évite de configurer de manière hasardeuse ses requêtes AJAX et offre un gain de temps intéressant. Citons ainsi les méthodes *post* et *get* qui réalisent respectivement de simples requêtes POST et GET ainsi que les fonctions getJSON et getScript qui permettent respectivement de récupérer du contenu depuis un fichier au format JSON et depuis un fichier Javascript. Les méthodes liées à AJAX au sein de JQuery ne s'arrêtent pas à ces quelques exemples, bien au contraire. Le lecteur intéressé se rapportera à la documentation de la bibliothèque afin de découvrir les immenses possibilités de celles-ci. Enfin, voici un exemple de tableaux dont les données sont chargées via une requête AJAX depuis un fichier au format JSON :

```
<html>
<head>
<link rel="stylesheet" href="css/style.css" type="text/css" id
="" media="print, projection, screen" />
<script type="text/javascript" src="lib/jQuery.js"></script>
<script  type="text/javascript"  src="lib/jQuery.tablesorter.
js"></script>
<script type="text/javascript">
     var refreshTable = function(table){
          table.trigger("udpate");
          table.trigger("appendCache");
     } ;

     var addRowToTable = function(table, data){
        $("tbody", table).append("<tr><td>" + data.Id +"</td>
<td>" + data.Title + "</td><td>" +
           data.Author + "</td><td>" + data.Publish + "</td>
<td>" + data.Price + "</td></tr>");
     } ;
     var loadContent = function(table, file){
          $.getJSON(file, function(json){
               for(i = 0;i < json.data.length; i++){
                    addRowToTable(table, json.data[i]);
               }
               refreshTable(table);
          });
     } ;
     $(function(){
          $("#sortable").tablesorter({widgets: ['zebra']});
          $("input").click(function(){
              loadContent($("#sortable"), "data/content.json");
          });
     });
</script>
 <style>
     #container{
          margin-bottom : 20px;
     }
</style>
</head>
<body>
     <div id="container">
          <table id="sortable">
               <thead>
                    <tr>
                         <th>Id</th>
                         <th>Title</th>
                         <th>Author</th>
                         <th>Publish</th>
                         <th>Price</th>
                    </tr>
               </thead>
               <tbody></tbody>
          </table>
     </div>
     <input type="button" value="Charger les données" />
</body>
</html>
```

La figure 2 présente la page à son chargement et la figure 3 quant à elle montre la page une fois les données chargées, via un clic sur le bouton prévu à cet effet.

INTERFACE GRAPHIQUE

Au niveau interface graphique, jQuery n'est pas en reste également puisque la bibliothèque propose un grand nombre d'effets visuels, que ce soit dans son noyau ou via le grand nombre de plug-in disponibles sur internet. Parmi eux, on citera le projet jQuery UI qui est développé autour de Paul Bakaus et qui est associé officiellement au projet principal jQuery. De ce fait, la réalisation d'effets de type *fade in / out*, *slide up / down* ou *hide / show* sur des éléments du DOM devient un jeu d'enfant, puisqu'il suffit d'appeler la méthode jQuery correspondante sur l'élément sélectionné. Ainsi, pour mettre en place un effet visuel *slide* sur un *div* d'*id* container, on procèdera comme suit :

```
function testSlide(){
     $("#container").slideUp("normal");
     $("#container").slideDown("normal");
}
```

Fig.1

Rendu de l'exemple au sein d'un navigateur

Fig.2

Rendu du tableau vide

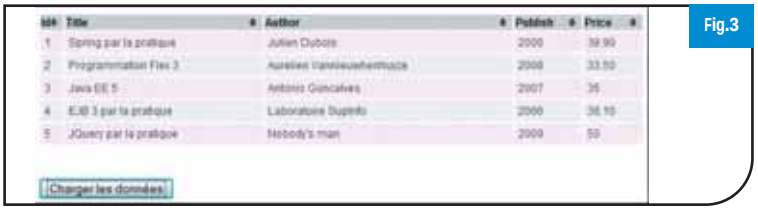


Fig.3

Rendu du tableau une fois les données chargées

Les méthodes liées à ces effets s'accompagnent de nombreuses options qui vont permettre leur configuration exacte, afin de préciser par exemple la vitesse souhaitée pour la réalisation de l'effet ...
Les apports du plug-in jQuery UI concernent des effets graphiques plus poussés tels que le *drag and drop*, le redimensionnement ou la sélection de calques à la souris. L'utilisation de ces fonctionnalités s'avère réellement très simple :

```
$(function(){
     $("#container").draggable({helper: 'clone'}).resizable();
      $("#container2").droppable({
            accept : "#container",
            activeClass : 'droppable-active',
            hoverClass : 'droppable-hover',
            drop : function(ev, ui){
                  $(this).append("<br />Dropped !");
            }
      }).resizable();
      $("#container3").draggable().resizable();
});
```

En considérant une page HTML contenant 3 éléments *div* d'*id* respectifs container, container2 et container3, on va pouvoir définir lesquels sont *droppables* ou *draggables*, tout en spécifiant les comportements à effectuer en cas de *drop* sur l'un d'entre eux. Les paramètres nous offrent également la possibilité de préciser quels éléments un élément peut accepter en action *drop*. Pour terminer, on va utiliser la méthode *resizable* pour autoriser le redimensionnement des *divs* container et container3. Le rendu de cet exemple est présenté aux figures 4 et 5.
Sur la figure 5, on constate que le *div* de couleur bleue a été redimensionné et que le *div* de couleur rouge est en train d'être droppé dans le *div* de couleur verte qui autorise cette action.
Enfin, le plug-in jQuery UI ne se contente pas d'amener des fonctionnalités en termes d'interactions mais il apporte également des composants utilisés fréquemment dans les applications web et dont le développement s'avère fastidieux. Parmi ces composants, on citera notamment :

- Un *datepicker* qui permet de sélectionner une date depuis un calendrier
- Un système "d'accordéons" pour la gestion de menus
- Des boîtes de dialogue modales
- Un *slider*

PLUG-IN

L'architecture résolument simple du framework jQuery n'est pas due au hasard. En effet, elle est le résultat d'une volonté de rendre le framework facilement extensible autour de son noyau. Le nombre de plug-in existants et la réactivité de la communauté autour de jQuery suffisent à prouver que le développement de plug-in est chose aisée. De ce fait, tout manque de fonctionnalités peut être rapidement contourné, soit en trouvant son bonheur dans la liste des plug-in existants, soit en développant soi-même son propre plug-in répondant à ses besoins.
Pour illustrer cette facilité de création, nous allons mettre au point un plug-in somme toute banal, permettant de modifier la couleur d'arrière-plan de l'élément sur lequel il est appelé. Ce plug-in aura pour nom *backgroundcolor* et sera réalisé au sein du fichier jquery.backgroundcolor.js pour respecter les règles de nommage des plug-in jQuery. Voici le contenu de ce fichier :

```
(function($){
      $.fn.backgroundcolor = function(options){
            var defaults = {color : "white"};
            var options = $.extend(defaults, options);
            $(this).css("background", options.color);
      };
})(jQuery);
```

La création au sein de function($) permet d'éviter des collisions de l'alias $ avec d'autres bibliothèques éventuellement utilisées et qui se baseraient également sur cet alias. Notre plug-in développé nous allons maintenant l'utiliser comme suit :

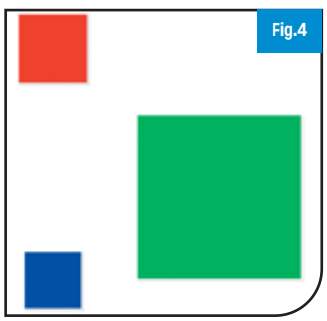
```
<script type="text/javascript" src="lib/jquery.js"></script>
<script type="text/javascript" src="lib/jquery.backgroundcolor.
js"></script>
<script type="text/javascript">
      $(function(){
            $("#container").backgroundcolor({color : "green"});
      });
</script>
```

Ici, il est important de remarquer l'ordre des inclusions des fichiers Javascript. On commence toujours par inclure la bibliothèque jQuery puis les différents plug-in que l'on souhaite utiliser. Ensuite, on sélectionne un élément du DOM et on applique notre méthode avec en entrée le paramètre color positionné à "green". Ce plug-in reste bien entendu basique et devrait être étendu pour pouvoir être utilisé convenablement avec jQuery, avec notamment la prise en compte de l'appel en chaînes des méthodes ...

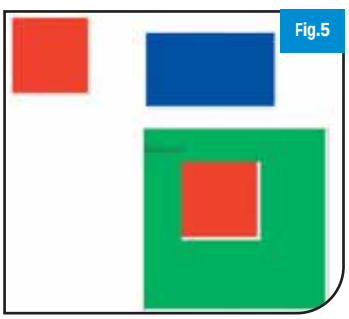
CONCLUSION

jQuery a également ses défauts avec un manque relatif de performance sur la manipulation du DOM de par sa syntaxe hyper puissante de sélection. En outre, il conviendra de choisir attentivement ses plug-in et de prendre garde aux méthodes du framework puisqu'elles peuvent être abandonnées, bien que fonctionnelles, si le choix est fait par la communauté en charge du projet. C'est en quelque sorte le petit revers de l'utilisation de projets open-source. Ainsi, il peut être intéressant de se tourner vers d'autres frameworks Javascript du même style, tels que Prototype, Ext Js, mootools ... et j'en passe, afin de se faire sa propre opinion et de choisir la bibliothèque la mieux adaptée à ses besoins.

■ Sylvain **Saurel** – *Ingénieur Etudes et Développement Java / JEE*
sylvain.saurel@gmail.com



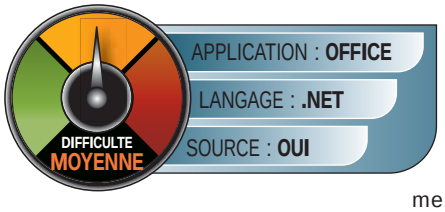
Divs au chargement de la page



Divs après diverses opérations

Manipuler les documents Office 2007 sans Office avec le SDK Open XML v2

Après de nombreuses années où Microsoft ne proposait la manipulation de documents Office qu'au travers d'Office Automation, la sortie du format Open XML – récemment devenu une norme ISO – a bousculé cette contrainte.



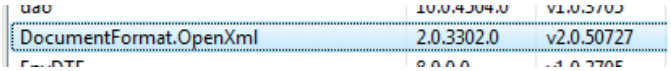
En effet, dans l'idée de toujours rendre plus productifs les développeurs, Microsoft n'a cessé de fournir des outils pour créer ou consommer des documents Office ; et cela sans nécessiter aucune instance d'Office sur la machine hôte. C'était déjà une réalité avec la sortie de .NET 3 et l'espace de nom System.IO.Packaging, puis la sortie de la première version du SDK Open XML ; aujourd'hui, Microsoft frappe fort et confirme son engagement envers cette norme de documents bureautique avec la sortie du SDK Open XML v2.

La v2 encapsule la partie structurelle du fichier – organisé selon la spécification Open Packaging Convention - et la structure XML des différentes parties d'un document dans un modèle objet simple et flexible à manipuler. Ceci dit, comment cela se traduit-il en pratique ? Un exemple étant plus parlant qu'un long discours, et pour vous faire une idée rapidement, voici le code nécessaire à un développeur pour créer un document Word entièrement fonctionnel avec le SDK v2 :

```
using (WordprocessingDocument package = WordprocessingDocument
.Create(docName, WordprocessingDocumentType.Document))
{
  // Ajouter une partie de contenu au document
  package.AddMainDocumentPart();

  // Créer le contenu de votre document
  package.MainDocumentPart.Document =
    new Document(
      new Body(
        new Paragraph(
          new Run(
            new Text("Hello World !")))));

  // Enregistrer le contenu dans le document
  package.MainDocumentPart.Document.Save();
}
```

La création de document bureautique n'a jamais été aussi simple ! La v2 gérera lui-même pour vous toutes les opérations de génération ou de manipulation de la structure XML et OPC. Pour résumer, vous pouvez manipuler entièrement un document grâce au modèle objet sans avoir jamais à créer vous-même une quelconque structure XML. Par rapport aux précédents SDK ou librairies disponibles, cette approche simplifie à l'extrême la vie des développeurs.

Deux manipulations sont nécessaires de votre part pour utiliser le SDK Open XML v2 : télécharger la CTP du SDK Open XML et ajouter la référence aux DLLs *WindowsBase* et *DocumentFormat.OpenXml* à votre projet.



Notez que le SDK supporte très bien la programmation avec Linq To XML – comme nous le verrons par la suite - qui permet de requêter ou de construire une structure XML très simplement et rapidement. Côté performance, il vous faudra moins de 3 secondes pour générer 1000 documents avec le code précédent sur une machine standard.

LE SDK EN DÉTAIL

L'espace de nom *System.IO.Packaging* autorise la manipulation du document au niveau de sa **structure logique** Open Packaging Convention (OPC offre une structure d'accueil flexible pour le stockage des **parties** dans le fichier compressé final). La première version du SDK Open XML se contentait de permettre la manipulation des parties de façon typée (chaque partie possédait ainsi des propriétés et méthodes adéquates à sa fonction dans le document) ; la seconde version du SDK fournit quant à elle des classes typées pour le **contenu** des parties (paragraphes, cellules, etc.).

Le kit permettra aussi de valider des documents Open XML existants ou ceux que vous avez composés. Néanmoins, soyez conscients que le modèle objet proposé ici n'est pas un remplaçant du modèle objet d'Office tel qu'on le retrouve dans VSTO, et que vous devez néanmoins connaître la structure générale des fichiers Open XML. Dans la même logique, ce SDK a été conçu uniquement pour manipuler le format Open XML, par conséquent n'espérez pas y trouver de convertisseurs (vers HTML, XPS, doc, etc.), ni de validateur si vous manipulez directement le XML sous-jacent sans passer par l'API, ou encore d'outils fonctionnels (recalcul de feuille Excel, pagination d'un document Word, etc.). Ce SDK traite uniquement le format Open XML et il le fait bien, malgré sa jeunesse et son statut de CTP. Car en effet, cette version est encore en Community Technology Preview et ne pourra être déployée dans des solutions d'entreprise avant le « Go Live » de l'éditeur.

LES OUTILS LIVRÉS AVEC LE SDK

Un SDK livré sans outil complémentaire n'en étant pas vraiment un, sachez que cette version du SDK s'accompagne d'une documentation et d'une multitude d'outils. Leur seul but est d'épauler les développeurs dans leur apprentissage et l'utilisation au jour le jour du SDK. Voici un détail de chaque outil apporté par la version disponible lors de la rédaction de cet article.

OpenXml Diff

Ce premier outil vous permettra de faire la différenciation entre le contenu de deux documents Open XML. Cela a son intérêt lorsque vous générez des documents, si ceux-ci ne s'ouvrent pas ou n'affichent pas ce que vous espériez. Il vous suffira alors de les comparer à une version qui s'ouvre, par exemple générée par Office, pour savoir ce que vous devrez modifier dans votre solution de génération de documents Open XML. [Fig.1]

Le Class Explorer

Cet outil est une documentation à part entière dans laquelle vous pourrez chercher les classes et les énumérations proposées par le SDK. En face de chaque classe, vous trouverez la section de la spécification Open XML de l'ECMA ainsi que la documentation MSDN de la classe. [Fig.2]

Le Document Reflector

Cet outil est la Rolls des outils de ce SDK puisqu'il permettra aussi bien aux débutants qu'aux experts de pouvoir générer le code de génération d'un document complet, que ce soit un document Word, Excel ou PowerPoint. En effet, il vous suffira d'ouvrir un document avec cet outil, et il générera pour vous l'intégralité du code C# utilisant le SDK Open XML v2 pour générer à l'identique le document ainsi ouvert. En plus de proposer cette fonctionnalité qui devrait augmenter la productivité des développeurs de façon conséquente, cet outil permet aussi de prendre connaissance du XML de chaque partie ou des éléments du document (paragraphe, cellule, diapositive, etc.). [Fig.3]

UTILISER LE SDK OPEN XML PAR LA PRATIQUE

Les scénarios d'utilisation d'un tel SDK sont assez nombreux, cela peut aller de la simple génération de factures ou de rapports à l'extraction de données pour exportation dans une base de données. N'oublions pas les scénarios de conversion ou de manipulation de documents à la volée (formatage d'entreprise, suppression des données personnelles, etc.).

Dans cet article nous allons voir ensemble les deux scénarios de base, à savoir la génération et la consommation de documents. Dans le premier cas, il s'agira d'un document Word, dans le second cas d'un document Excel.

Génération d'un document Word

La génération d'un document Word passe par l'utilisation du modèle objet du SDK qui se révèle être efficace et rapide pour cette tâche. Pour générer le document du code suivant, nous créons simplement deux parties : la partie principale (celle qui contient les paragraphes du document) et la partie de style (contenant l'ensemble des styles du document).

```
using (WordprocessingDocument wordDoc =
  WordprocessingDocument.Create("SampleDemo.docx",
  DocumentFormat.OpenXml.WordprocessingDocumentType.Document))
{
  // Création d'une nouvelle partie de contenu
  MainDocumentPart mainPart = wordDoc.AddMainDocumentPart();
  // Création d'un nouveau paragraphe
  Document mainDoc = new Document(
    new Body(
    new Paragraph(
      new ParagraphProperties(
        new ParagraphStyleId() { Val = "Title" }),
      new Run(
        new RunProperties(
          new Italic()),
          new Text("Bienvenue au SDK Open XML V2 !")))));
  // Enregistrement de la partie
  mainDoc.Save(mainPart);
  // Création d'une nouvelle partie de style
  StyleDefinitionsPart stylePart = mainPart.AddNewPart<StyleDefinitions
Part>();
  Styles styles = new Styles(
    new Style(
    new Name() { Val = "Title" },
    new BasedOn() { Val = "Normal" },
    new NextParagraphStyle() { Val = "Normal" },
    new LinkedStyle() { Val = "TitleChar" },
    new UIPriority() { Val = 10 },
    new PrimaryStyle(),
    new Rsid() { Val = "002C2DBE" },
    new ParagraphProperties(
      new ParagraphBorders(
        new BottomBorder() { Val = BorderValues.Single, Color
 = "auto", Size = (UInt64)4UL, Space = (UInt64)1UL }),
      new SpacingBetweenLines() { Line = 240, LineRule = Line
SpacingRuleValues.Auto },
      new ContextualSpacing()),
    new RunProperties(
      new RunFonts() { AsciiTheme = ThemeFontValues.MajorHigh
Ansi, HighAnsiTheme = ThemeFontValues.MajorHighAnsi, EastAsia
Theme = ThemeFontValues.MajorEastAsia, ComplexScriptTheme =
ThemeFontValues.MajorBidi },
      new Spacing() { Val = 5 },
      new FontSize() { Val = (UInt64)52UL },
      new FontSizeComplexScript() { Val = (UInt64)52UL })
    ) { Type = StyleValues.Paragraph, StyleId = "Title" });
  // Enregistrement des styles
  styles.Save(stylePart);
}
```

Comme énoncé précédemment, l'utilisation de cette v2 nécessite une connaissance du format. Chaque classe utilisée dans le code précédent

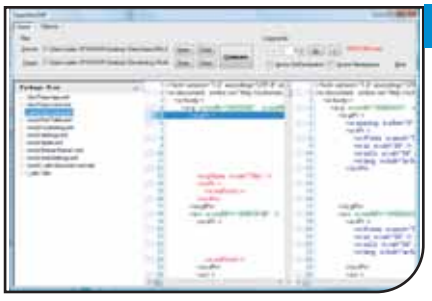


Fig.1

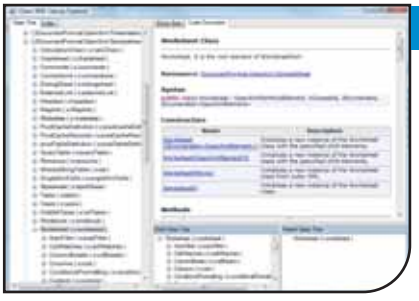


Fig.2

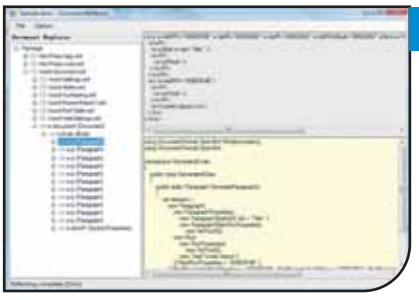


Fig.3

représente un élément ou un attribut XML qu'il vous faudra connaître pour exploiter le format à son maximum. L'exécution de l'application permettra de générer un document que vous ouvrirez sans aucun problème dans Office 2007 ou **Open Office 3.0.** [Fig.4]
Utilisé parallèlement avec Linq et les nouvelles fioritures de la nouvelle syntaxe de C# 3, la productivité des développeurs atteindra son apogée dans ce genre de projet.

CONSOMMER UN DOCUMENT EXCEL

Attaquons maintenant à la discipline inverse de la précédente partie, c'est-à-dire, la consommation de données d'un document Open XML. L'exemple qui suit procède à l'extraction des données d'un document Excel en utilisant le SDK et un peu de Linq.

```
static WorksheetPart ObtenirPartieFeuille(WorkbookPart classeur,
 string nomFeuille)
{
 try {
   Sheet feuille = classeur.Workbook.Descendants<Sheet>()
     .Where(s => nomFeuille.Equals(s.Name)).First();
   return (WorksheetPart)classeur.GetPartById(feuille.Id);
 }
 catch (Exception) {
   return null;
 }
}

static Cell ObtenirCellule(WorksheetPart feuille, int index
Ligne, int indexColonne)
{
 try {
   Row row = feuille.Worksheet.Descendants<Row>()
     .Where(r => indexLigne == r.RowIndex).First();
   return (Cell)row.ChildElements.ElementAt(indexColonne);
 }
 catch (Exception) {
   return null;
 }
}

static string ObtenirValeurCellule(Cell cellule, SharedString
TablePart sharedStringTablePart)
{
 if (cellule.ChildElements.Count == 0)
   return null;

 string value = cellule.CellValue.InnerText;
 if ((cellule.DataType != null) && (cellule.DataType
== CellValues.SharedString))
   value = sharedStringTablePart.SharedStringTable.Child
Elements[Int32.Parse(value)].InnerText;
 return value;
}

static void Main(string[] args)
{
 using (SpreadsheetDocument excelDoc = SpreadsheetDocument.
Open("Sample.xlsx", false))
 {
   WorksheetPart feuille1 = ObtenirPartieFeuille(excelDoc.
WorkbookPart, "Sheet1");

   Cell cA1 = ObtenirCellule(feuille1, 1, 0); // Ligne 1, première
 colonne (base 0)
   Cell cA2 = ObtenirCellule(feuille1, 2, 0); // Ligne 2, première
 colonne (base 0)
   Console.WriteLine("{0}  {1}", ObtenirValeurCellule(cA1,
 excelDoc.WorkbookPart.SharedStringTablePart), ObtenirValeur
Cellule(cA2, excelDoc.WorkbookPart.SharedStringTablePart));
 }
}
```

Le SDK rend la consommation d'un document Open XML aussi aisée que la génération. Les classes typées du SDK répondent à la logique de Linq to XML et elles vous permettront d'écrire du code de façon productive et intuitive sans vous égarer dans les méandres techniques de la sérialisation / désérialisation XML.
A l'instar de l'exemple sur la génération du document Word, il vous faudra connaître les bases de la structure d'un document pour être à même de récupérer les données voulues. Par exemple, dans le cas d'un document Excel, vous devez savoir que les valeurs de type chaînes sont contenues dans une partie indépendante qui indexe toutes les chaînes de votre classeur afin d'en optimiser le stockage. [Fig.5] et [Fig.6]

CONCLUSION

La création de documents Office n'a jamais été aussi simple avant l'arrivée du SDK v2. Cette version devrait apporter aux développeurs de la productivité et de la simplicité dans la manipulation de documents Office, et cela, quel que soit le format utilisé (Word, Excel ou PowerPoint). Les nombreux outils annexés avec cette version du SDK assureront également deux objectifs : le premier de formation et le second, d'accroissement de votre productivité. Cette version du SDK, bien qu'encore en CTP, est relativement stable et performante et mérite que l'on y prête attention pour les projets à venir. Néanmoins, son statut de CTP vous empêchera de la déployer dans vos projets clients pour le moment. Il faudra sûrement attendre la sortie du SP2 d'Office 2007 ou de Office « 14 » pour savourer la fraîcheur de cette librairie qui n'augure que de bonnes choses pour les documents Office dans les années à venir. Egalement, n'oubliez pas de rejoindre la communauté Connect sur le site associé au SDK (https://connect.microsoft.com/site/sitehome. aspx?SiteID=589) afin de déposer vos feedbacks, vos bugs, etc. et de contribuer à son amélioration.

■ Julien **Chable** – *Wygwam France*
Expert Open XML, développeur .Net

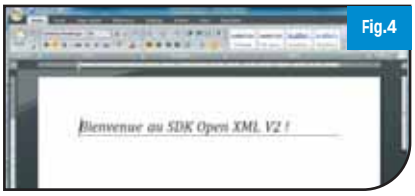


Fig.4

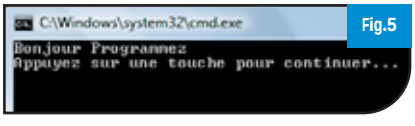


Fig.5

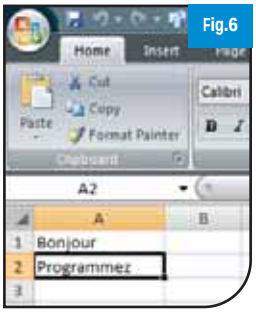
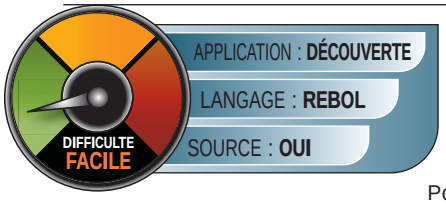


Fig.6

A la découverte de REBOL

2e partie

Nous continuons notre découverte de ce langage agréable, anti-conformiste, et par bien des aspects *"rébolutionnaire"*.



Le mois dernier, nous nous sommes familiarisés avec REBOL et ses fonctionnalités de base. Aujourd'hui nous irons un petit peu plus loin. Pour cela nous allons bricoler une micro base de données, type répertoire téléphonique, et faire circuler les données sur le réseau entre un serveur et un client graphique. Tout cela en un seul article peut sembler ambitieux. C'est pourtant un jeu d'enfant car REBOL est expressif, puissant et concis.

1 REBOL ET LES OBJETS

Carl Sassenrath, le créateur de REBOL a été pendant longtemps un inconditionnel de la programmation objet, puis après huit années de pratique, est revenu peu à peu sur son opinion, jugeant finalement la programmation objet "académique" telle qu'on la rencontre en Java ou C++, peu productive. Pour le coup le paradigme objet de REBOL est plutôt libre et finalement très souple. En REBOL un objet est essentiellement un agrégat de valeurs regroupées dans un contexte commun. Les valeurs peuvent être des scalaires, des séries, des fonctions, etc. Bref les objets sont des structures de données complexes. En REBOL un objet peut être prototypé, ce qui ressemble à la déclaration de classe des langages académiques, puis instanciés. Nous voulons modéliser une micro base de données qui décrit les membres du personnel de Programmez!. Chaque enregistrement conservera les nom, prénom, salaire et rôle de chaque membre. Pour pouvoir instancier les objets en une ligne de code, nous allons aussi nous doter d'une fonction make-membre. Nous plaçons le tout dans le fichier modele-membre.r que vous trouverez sur notre site.

```
REBOL [
  Title: "Modele membre"
  Date: 7-Aug-2008
  File: %modele-membre.r
  Author: "Fred"
  Version: 1.0
  Purpose: {definit un modèle d'objet et une fonction de creation }
]

membre: make object! [
  nom:
  prenom:
  salaire:
  role: none
]

make-membre: func [
  "Cree un objet de type membre"
  p-nom [string!] "Nom"
  p-prenom [string!] "Prenom"
  p-salaire [integer!] "Salaire"
  p-role [string!] "Role"
][
  make membre [
    nom: p-nom
    prenom: p-prenom
    salaire: p-salaire
    role: p-role
  ]
]
```

Petite remarque en ce qui concerne le modèle de données: contrairement aux apparences tous les membres se voient attribuer *none* par défaut et non pas le membre *rôle* seulement. En effet REBOL supporte les affectations multiples. La disposition adoptée dans notre code à pour but la clarté et la lisibilité de la définition de l'objet, mais

```
nom: prenom: salaire: role: none
```

aurait été équivalent et aurait mis en évidence l'affectation multiple.

2 REBOL ET LA RÉUTILISATION DU CODE

Nous pouvons maintenant créer notre base de données. Voici le code, fichier create-db.r, qui se charge de l'opération.

```
REBOL [
    Title: "Create DB"
    Date: 7-Aug-2008
    File: %create-db.r
    Author: "Fred"
    Version: 1.0
    Purpose: {Cree une micro base de donnees }
]

; Charger le modele pour les enregistrements
do %modele-membre.r

db: []

insert db make-membre "Kaminsky" "Jean" 1000000 "Big Boss Man"
insert db make-membre "Tonic" "Francois" 10000 "Redac'Chef"
insert db make-membre "Develay" "Sebastien" 100 "Webmaster"
insert db make-membre "Mazue" "Frederic" 1 "Grouillot"

save/all %db.txt db
```

Par ce script nous pouvons commencer à apprécier la puissance de REBOL, appuyée sur les types de haut niveau et sur les raffinements de fonction. Nous appelons la fonction *do,* capable d'exécuter aussi bien un bloc de code qu'un script REBOL comme ici. Cet appel à *do* avec comme paramètre le fichier modele-membre a pour effet d'incorporer au dictionnaire de REBOL le modèle d'enregistrement de notre base de données ainsi que la fonction d'instanciation. Ensuite,

nous définissons notre base de données comme une série vide. Il ne nous reste plus qu'à insérer les enregistrements et à sauvegarder le tout. Nous sauvegardons dans un fichier. Le point intéressant est que nous utilisons le raffinement *all* de la fonction *save*. Ceci à pour effet de sauvegarder notre série sous forme sérialisée. Il nous suffira de la recharger avec la fonction symétrique *load* et son raffinement *all* pour retrouver nos données en mémoire, telles qu'elles y étaient au moment de la sauvegarde. Voici à quoi ressemblent nos données sérialisées :

```
#[object! [
  nom: "Mazue"
  prenom: "Frederic"
  salaire: 1
  role: "Grouillot"
]] #[object! [
  nom: "Develay"
  prenom: "Sebastien"
  salaire: 100
  role: "Webmaster"
]] #[object! [
  nom: "Tonic"
  prenom: "Francois"
  salaire: 10000
  role: "Redac'Chef"
]] #[object! [
  nom: "Kaminsky"
  prenom: "Jean"
  salaire: 1000000
  role: "Big Boss Man"
]]
```

3 CHARGER, TRIER ET FORMATER LES DONNÉES

Le chargement est une formalité, comme évoqué plus haut. Nous voyons que les enregistrements ont été sauvegardés dans l'ordre inverse de leur insertion dans la série, ce qui n'est en rien surprenant. Pour notre propos, il est naturel de vouloir être en mesure de trier les enregistrements, par exemple en les classant par ordre alphabétique du membre nom. Notre script read-db.r s'acquitte de cette tâche.

```
REBOL [
    Title: "Read DB"
    Date: 7-Aug-2008
    File: %read-db.r
    Author: "Fred"
    Version: 1.0
    Purpose: {lit une micro base de données }
]

; Charger le modele pour les enregistrements
do %modele-membre.r

db: load/all %db.txt
db: sort/compare db func [a b] [a/nom < b/nom]

text-db: ""
t: text-db

foreach item db [
  append t item/role append t " "
  append t "Nom: " append t item/nom append t " "
  append t "Prenom: " append t item/prenom append t " "
  append t "salaire: " append t item/salaire append t " "
  append t "|^/"
]

unset [t]
```

Pour trier la base de données chargée sous forme d'une série et rangée dans la variable *db,* nous utilisons une fonction fort pertinemment nommée *sort*. Et puisque nous commençons à bien connaître REBOL, nous ne sommes pas surpris que cette fonction possède de nombreux raffinements. Nous utilisons ici le raffinement *compare.* Ce raffinement est nécessaire quand REBOL ne sait pas a priori comment trier des valeurs. Si nous avions une série de types intégrés, par exemple des entiers

```
s: [1 3 2]
```

REBOL saurait la trier en ordre croissant

```
>> t: sort s
== [1 2 3]
```

Ou en ordre décroissant via le raffinement *reverse*

```
>> r: sort/reverse s
== [3 2 1]
```

Pour un type utilisateur, nous utilisons comme nous l'avons dit le raffinement *compare,* ce qui demande de donner une fonction de tri en argument supplémentaire à *sort :*

```
func [a b] [a/nom < b/nom]
```

Cette fonction a l'allure des fonctions anonymes que l'on rencontre avec des langages comme par exemple Python. Notre fonction est générique en ce sens qu'elle va savoir trier n'importe quel objet, pourvu que celui-ci comporte un membre nom. Ensuite nous transformons notre base de données en une chaîne de caractères. Les enregistrements sont séparés par le caractère |. Nous faisons ceci pour montrer un exemple de travail avec les chaînes en REBOL et parce que nous avons choisi une communication en mode ligne à travers un socket, ainsi que nous le verrons plus loin. Mais il y aurait bien d'autres (et meilleures) manières de procéder. REBOL est un langage plein de ressources sur le réseau. Nous rangeons la chaîne de caractères dans la variable *text-db.* Mais pour la constitution, nous travaillons avec une variable t, pour gagner du temps à l'écriture. Cela nous donne aussi l'occasion de découvrir deux particularités de REBOL. *t* est définie ainsi

```
t: text-db
```

Si le contenu de *text-db* se retrouve bien dans t, cette dernière n'est toutefois pas une "autre" variable. C'est seulement un synonyme de *text-db*. Autrement dit, toute opération affectant le contenu de *t* se répercutera sur le contenu de *text-db,* et réciproquement. Pour prendre une analogie avec d'autres langages, tout se passe comme si *text-db* et t se comportaient comme des références. Cette carac-

téristique de REBOL peut surprendre l'utilisateur non prévenu. A la fin de notre script nous faisons :

```
unset [t]
```

Après quoi *t* n'existe plus, mais *text-db* elle, en revanche, continue à exister :) Une dernière remarque. Si notre script *read-db* était invoqué par un autre script et que ce dernier contienne une variable *t*, celle-ci serait affectée par notre petit tour de passe-passe qui doit donc être employé avec précaution.

4 REBOL, LES PORTS DE COMMUNICATION ET LE RÉSEAU

REBOL se veut un langage particulièrement à l'aise sur le réseau, et il l'est effectivement. Il embarque nativement une ribambelle de protocoles réseau, dont, naturellement, TCP que nous allons utiliser pour écrire un serveur rudimentaire. Mais avant cela, découvrons une fonctionnalité d'entrées / sorties propre à REBOL: les ports. Un port REBOL regroupe deux concepts. D'abord il généralise la notion de fichiers, car il permet de travailler, outre les fichiers disques, avec le réseau, les consoles, les bases de données, etc. et ceci avec un jeu commun de fonctions. Ensuite, toutes les données devant transiter à travers des ports sont placées dans des séries. Pour travailler avec les ports, on n'utilisera donc pas des fonctions de lecture / écriture dans des fichiers comme *read* ou *write,* mais des fonctions d'insertion / extraction de données dans des séries. Pour prendre un élément de comparaison, on peut dire que le port REBOL se rapproche des flux Java. Illustrons ceci par un tout petit exemple qui pourrait être saisi directement dans la console REBOL :

```
close insert open/new %mon-fichier.txt "Programmez!"
```

Par cette simple ligne de code, nous ouvrons un port ayant comme support physique un fichier et nous insérons la chaîne Programmez! dans le port. Comme nous l'avons dit plus haut, *insert* est une fonction qui travaille avec les séries. Nous pouvons relire le contenu du fichier de manière similaire :

```
p: open %mon-fichier.txt
print copy p
close p
```

Ici la fonction *copy,* comme son nom le suggère, retourne une copie de la série de données du port. Tout se passe comme si le contenu du port étant lu entièrement en une fois. Mais si l'on reprend la comparaison avec les flux Java, nous trouvons ici une différence notable: le port n'est pas vide de données à l'issue de la lecture. Ainsi on pourrait très bien appeler *copy* plusieurs fois consécutivement.

5 UN SERVEUR RUDIMENTAIRE

Les ports REBOL sont un vaste sujet qui mérite d'être approfondi, mais le peu que nous en savons nous permet déjà d'écrire un serveur rudimentaire :

```
REBOL [
 Title: "DB Server"
 Date: 7-Aug-2008
 File: %db-server.r
 Author: "Fred"
 Version: 1.0
 Purpose: {sert une micro base de données via TCP }
]

do %read-db.r

server-port: open/lines/no-wait tcp://:6969
forever [
 connection-port: first server-port
 wait connection-port
 print copy connection-port
 insert connection-port text-db
 wait connection-port
 close connection-port
]
```

Nous commençons par appliquer la technique vue plus haut: nous exécutons le script *read-db.r.* Ainsi notre base de données et sa représentation textuelle sont chargées en mémoire. (On part de l'hypothèse simplificatrice que la base de données n'est pas modifiée tant que le serveur tourne ;) Ensuite nous ouvrons notre port de communication. Le simple fait de mentionner le protocole TCP dans le nom du port enclenche le travail avec les sockets. Nous donnons 6969 comme valeur de port du socket. Le terme de port ici ne doit pas prêter à confusion: Les ports REBOL sont différents des ports de socket. Ensuite dans une boucle perpétuelle, nous nous mettons à l'écoute du port REBOL. Pour ouvrir ce port nous avons utilisé le raffinement *lines,* donc la communication s'effectuera ligne par ligne. Nous attendons une requête du client, c'est-à-dire une ligne de texte quelconque émise par celui-ci. Nous imprimons cette ligne en écho sur la console côté serveur. On remarque que c'est bien *copy,* fonction de manipulation de série, qui est utilisée pour extraire les données. Puis en réponse à la requête nous écrivons dans le port, au moyen de la fonction de série *insert,* la ligne représentant les données de notre base. Le code de notre serveur est vraiment très simple. Mais remarquons quand même qu'il fonctionne en mode synchrone, ce qui est acceptable seulement pour une communication en peer to peer à la maison ou dans le cadre de cet article. Un vrai serveur doit fonctionner en mode asynchrone pour accepter des connexions simultanées, ce qui est un peu plus compliqué, même en REBOL, et sort du cadre de cet article d'initiation.

6 LE CLIENT

Le code du client est lui aussi extrêmement simple :

```
REBOL [
 Title: "DB Client"
 Date: 7-Aug-2008
 File: %db-client.r
 Author: "Fred"
 Version: 1.0
 Purpose: {client de db-server.r }
]

p: open/lines/no-wait tcp://localhost:6969
insert p "requete"
wait p
```

```
print copy p
close p
```

7 VID LE DIALECTE DES INTERFACES GRAPHIQUES

Les RIA étant dans l'air du temps, nous n'allons pas nous contenter d'un client en mode texte, mais écrire un petit client graphique, ce qui en REBOL est un jeu d'enfant. Nous utilisons pour cela REBOL / view (exécutable rebview ou rebview.exe selon votre plate-forme), une extension de REBOL / Core. Cette extension est un dialecte REBOL destiné à la création d'interfaces graphiques. REBOL a cette caractéristique unique d'avoir été conçu pour être enrichi de dialectes, ce que l'on nomme dans d'autres contextes des DSL, ou Domain Specific Languages, c'est à dire des langages dédiés à une tâche bien particulière. Ici le dialecte s'appelle VID et il est dédié à la création d'interfaces graphiques. La définition des dialectes REBOL est basée sur la fonction couteau suisse *parse* dont nous avons parlé le mois dernier. Un des atouts majeurs d'un dialecte ou DSL, est évidemment une redoutable efficacité dans son domaine ;) Ainsi en va-t-il du dialecte VID qui permet d'écrire un code remarquablement concis. Voici à titre d'exemple un jeu de pousse-pousse codé en REBOL

```
Rebol [Title: "Sliding Tile Game"]
view center-face gui: layout [
 origin 0x0 space 0x0 across
 style p button 60x60 [
   if not find [0x60 60x0 0x-60 -60x0]
     face/offset - empty/offset [exit]
   temp: face/offset face/offset: empty/offset
     empty/offset: temp
    ]
 p "A" p "B" p "C" p "D" return p "E" p "F" p "G" p "H" return
 p "I" p "J" p "K" p "L" return p "M" p "N" p "O"
 empty: p 200.200.200 edge [size: 0]
]
```

Ce code a été emprunté sur le site http://musiclessonz.com/rebol_tutorial.html et vous en voyez le résultat ci-dessous [Fig.1]. Construire une interface graphique en REBOL est ludique. Nous commençons ainsi:

```
view layout []
```

Fig.1

Deux lignes de code pour une interface totalement fonctionnelle.

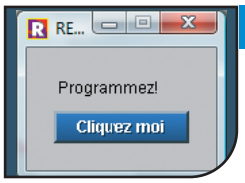
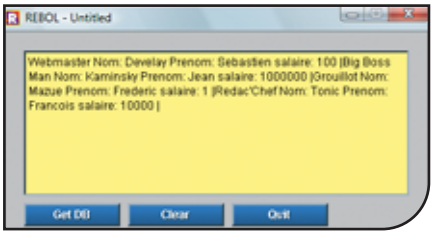


Fig.2

Un jeu de pousse-pousse, codé avec seulement quelques lignes de REBOL



Notre client graphique en action

view est une fonction qui attend un layout en argument et le fait apparaître sur l'écran. *layout* est la fonction qui construit ce *layout,* c'est-à-dire un assemblage de contrôles à partir des arguments qu'elle reçoit. Parmi ceux-ci figure toujours un bloc dans lequel sont définis ces contrôles, appelés *faces* en REBOL. Si notre première ligne de code ouvre une fenêtre minuscule, nous pouvons améliorer très facilement les choses :

```
view layout [text "Programmez!"]
```

Notre fenêtre contient maintenant un contrôle affichant du texte. Voulons-nous y ajouter encore un bouton capable de réagir à un clic de souris ? [Fig.2].

```
view layout [text "Programmez!"
  button "Cliquez moi" [alert "Abonnez vous! :)"]
]
```

Dans le bloc de code définissant le layout nous avons ajouté un bouton. Et à ce bouton est attribué un bloc de code qui sera exécuté lors du clic par l'utilisateur. Le lecteur qui voudra aller plus loin se familiarisera très facilement avec VID à partir de sa documentation. Voici maintenant le code de notre client graphique :

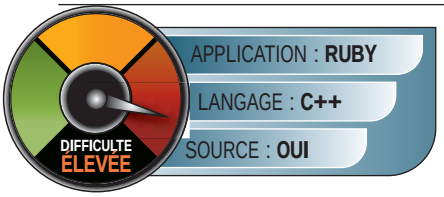
```
REBOL [
    Title: "DB Client GUI"
    Date: 7-Aug-2008
    File: %db-client-gui.r
    Author: "Fred"
    Version: 1.0
    Purpose: {client graphique de db-server.r }
]

get-data: func [
][
 p: open/lines/no-wait tcp://localhost:6969
 insert p "requete"
 wait p
 result: copy p
 close p
 result
]

view layout [my_area: area wrap
 across
 button "Get DB" [ my_area/text: get-data
   system/view/caret: head my_area/text
   system/view/focal-face: my_area
   show my_area]
 button "Clear" [ my_area/text: none show my_area]
 button "Quit" [quit]
]
```

On remarquera dans ce code très concis comment nous définissons le focus sur le contrôle de texte et comment nous y positionnons le curseur. Enfin, point important, bien remarquer l'emploi de la fonction show pour rafraîchir l'affichage quand un élément de l'interface est modifié.

Frédéric **Mazué** - *fmazue@programmez.com*

Ecrire des extensions au langage Ruby avec C++

Ruby, comme tous les langages de script, n'est pas un foudre de guerre. Ce n'est pas ce qu'on lui demande. Mais parfois il est besoin d'accélérer un script ou encore d'accéder à une librairie native. On écrit alors une extension en code natif. Voyons comment procéder avec C++.



Les langages de script apportent l'avantage de la concision du code, ainsi que la facilité et la rapidité de l'écriture de celui-ci, parce que ce sont des langages de haut niveau. Le revers de la médaille avec les langages de scripts, en raison même de leur nature, se situe au niveau de la vitesse d'exécution. Dans l'immense majorité des cas d'utilisation des scripts cette lenteur n'est pas pénalisante. Mais parfois le besoin de s'affranchir d'un goulet d'étranglement, ou encore le besoin d'invoquer une API native nécessite l'écriture d'une extension. Ruby, le langage de script qui nous intéresse aujourd'hui prévoit cette situation, et fournit un jeu d'API et de macros pour s'interfacer avec du code C. Nous allons étudier comment manier ce jeu d'API et écrire un peu de code C et surtout beaucoup de C++, langage que votre serviteur préfère à C.

1 EMBARQUER DU CODE C OU C++ DANS DU CODE RUBY

Mais avant d'entrer dans le vif du sujet, commençons par découvrir une curiosité qui peut se révéler utile dans certaines occasions. Il existe une gemme Ruby, du nom de *RubyInLine* qui permet d'embarquer du code C ou C++ directement dans du code Ruby. Une fois cette gemme installée, voici un exemple de ce qu'il est possible de faire (attention, respectez la disposition du code) :

```
require 'inline'

class Programmez
 inline do |builder|
   builder.c '
     void hello()
     {
       puts("Programmez!");
       puts("Abonnez vous! :-)");
     }'
 end
end

Programmez.new.hello
```

Pour que cela fonctionne, la gemme *RubyInline* fait le travail que nous ferons ensemble plus loin, elle génère à la volée une extension en C (ou C++) puis la compile pour aboutir à la création d'une librairie. Les esprits curieux peuvent consulter le répertoire .ruby_inline pour étudier le code généré. Le procédé est amusant mais malheureusement *RubyInLine* ne connaît qu'un sous-ensemble réduit de C et C++. C'est pourquoi nous mettons maintenant les mains dans le cambouis.

2 UNE EXTENSION EN C

Ruby dispose d'un mécanisme de génération de makefile, à partir d'un petit script. Voici un exemple basique pour la compilation d'une extension du nom de Programmez :

```
# fichier extconf.fb

require 'mkmf'

extension_name = 'Programmez'
dir_config(extension_name)
create_makefile(extension_name)
```

Exécuter ce script par

```
ruby extcon.rb
```

aboutit à la génération d'un makefile. Ensuite la commande:

```
make
```

construira l'extension à partir de tous les fichiers C et ou C++ qu'elle trouvera dans le répertoire courant. A quoi ressemblent ces fichiers C / C++ ? Voici un exemple de base, en C, de type Hello World!

```
#include "ruby.h"
#include <stdio.h>


VALUE ModuleProgrammez = Qnil;

VALUE hello(VALUE self)
{
 puts("Programmez!");
 puts("Abonnez vous! :)");

 return Qnil;
}

void Init_Programmez()
{
 ModuleProgrammez = rb_define_module("Programmez");
 rb_define_method(ModuleProgrammez, "hello", hello, 0);
}
```

Nous y remarquons d'abord la présence d'une fonction d'initialisation de l'extension. Le préfixe Init_ est requis et la casse du nom de la fonction doit correspondre à celui de l'extension. Dans son inter-

face avec C, Ruby manipule toutes les valeurs sous le type générique VALUE qui peut être vu comme une sorte de référence. Une VALUE nulle a la valeur *Qnil*. Le reste du code parle de lui-même. Nous renvoyons le lecteur à la documentation pour le détail des API utilisées. Une fois l'extension compilée, ce qui ne pose aucun problème sous Linux, on peut la tester dans l'interpréteur interactif, comme ceci :

```
irb
irb(main):001:0> require 'Programmez'
=> true
irb(main):002:0> include Programmez
=> Object
irb(main):003:0> hello
Programmez!
Abonnez vous! :)
=> nil
irb(main):004:0>
```

3 LE MÊME EN C++

Travailler avec C c'est bien, mais de l'avis de votre serviteur travailler avec C++ c'est mieux car ce langage est plus expressif et facilite la gestion des erreurs. Par contre il y a quelques petites subtilités, même avec un simple exemple de type Hello World. Transposons en C++ l'exemple précédent.

```
#include <iostream>

using namespace std;

#include "ruby.h"


VALUE ModuleProgrammez = Qnil;

VALUE hello(VALUE self)
{
  cout << "Programmez! C++ " << endl;
  cout << "Abonnez vous! :)" << endl;
  return Qnil;
}

extern "C" void Init_Programmez()
{
  ModuleProgrammez = rb_define_module("Programmez");
  rb_define_method(ModuleProgrammez, "hello",
    reinterpret_cast<VALUE (*)(...)>(hello), 0);
}
```

D'abord il est pertinent d'inclure les en-têtes C++ standard en premier. Sans cela, un compilateur (celui de Visual Studio 2008 par exemple) pourrait dérailler à cause des déclarations contenues dans l'en-tête ruby.h. Ensuite, les extensions étant des librairies chargées dynamiquement par Ruby, celui-ci doit trouver la fonction point d'entrée (ici Init_Programmez). Le nom de celle-ci ne doit pas être décoré par le compilateur, c'est pourquoi nous déclarons la fonction extern "C". Enfin il faut remarquer le transtypage de la fonction hello. Notre fonction hello ne reçoit qu'un seul paramètre, mais rb_define_method s'attend à recevoir un pointeur sur fonction variadique, c'est-à-dire recevant un nombre indéfini de paramètres. Un compilateur C laisse passer cela. Un compilateur C++, plus chatouilleux ne l'accepte pas, d'où le transtypage. Enfin, il est nécessaire que l'édition de liens incorpore la librairie standard C++. Pour cela, et sous Linux il convient d'ajouter une ligne dans le fichier extconf.rb :

```
# avec un espace devant -lstdc++
$LIBS = $LIBS + ' -lstdc++'
```

4 COMPILER SOUS WINDOWS

Sous Windows, les choses peuvent s'avérer moins directes en raison des disparités entre compilateur et versions de compilateurs. Pour travailler à l'aise, la meilleure solution, de l'humble avis de votre serviteur, est de compiler vous-même votre Ruby à partir des sources. L'opération est sans histoire en suivant les indications qui viennent avec la distribution. Ensuite si l'on utilise le compilateur Microsoft, c'est une bonne idée d'utiliser Visual Studio, version Express (et gratuite) ou supérieure. On créera alors un projet de type *makefile.* On ajoutera, en plus des fichiers C++, le fichier extconf.rb au projet, puis on configurera les commandes de construction ainsi : [Fig.1]

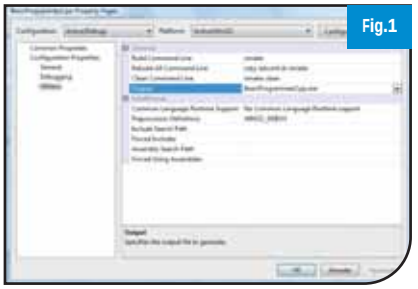


Fig.1

Configuration d'un projet de type Makefile pour compilation d'une extension à Ruby sous Visual Studio.

Commandes	Valeur
Build Command Line	nmake
Rebuild All Command Line	ruby extconf.db
	nmake
Clean Command Line	nmake clean

Etant bien entendu que l'exécutable Ruby est pointé par le PATH de votre système. Enfin, sous Visual C++ et pour assurer un bon comportement du code en cas de levée d'exception, on imposera une option de compilation en ajoutant une ligne dans le fichier extconf.rb:

```
$CFLAGS= $CFLAGS + '/EHsc'
```

5 UNE EXTENSION DE CLASSE

Augmenter les performances peut être une raison qui préside à l'écriture d'une extension à Ruby. Utiliser les ressources d'une librairie native peut être une autre raison. Nous allons maintenant écrire une extension plus élaborée, illustrant principalement ce dernier point. Ce sera aussi l'occasion d'aborder certains aspects pas toujours très bien documentés de l'interface Ruby / C. Nous voulons que notre extension soit vue par Ruby comme une classe Ruby ordinaire. Cette classe sera un conteneur de caractères. Les caractères sont rangés de manière contiguë dans une zone de mémoire dont il faudra assurer la gestion. La classe disposera d'un accesseur permettant d'obtenir un caractère en fonction de son rang. Un mauvais indice lèvera une exception Ruby. Si l'indice est correct, l'accesseur retournera une chaîne constituée de cet unique caractère. La classe disposera d'une méthode pour redéfinir le contenu de la zone mémoire. Pour le programmeur Ruby qui utiliserait la classe, cette zone mémoire et les données qu'elle contient sont totalement

opaques. La seule ressource native que nous utilisons est la gestion mémoire. Notre exemple reste volontairement simple pour être portable sur n'importe quel système disposant de Ruby et d'un compilateur C++. Toutefois cet exemple pourra être adapté sans difficulté à la gestion d'un handle de fichier, d'un sémaphore, ou de n'importe quoi d'autre, le principe étant toujours le même. Enfin notre code doit être robuste aux exceptions C++, afin de ne pas perturber le fonctionnement de l'interpréteur Ruby. Voici le code qui réalise tout cela et que nous allons analyser en détail:

```
#include <new>
#include <iostream>
#include <cstdlib>


#include "ruby.h"

static const char* default_text = "Programmez!";
static VALUE ModuleProgrammez = Qnil;
static VALUE CharHolder = Qnil;

struct CharHolderData
{
 char* p;
 int length;
};

static void CharHolderMark(struct CharHolderData* chd)
{
}

static void CharHolderFree(struct CharHolderData* chd)
{
 delete[] chd->p;
 delete chd;
 std::cout << "Memoire de CharHolder liberee" << std::endl;
}


static VALUE CharHolderAllocate(VALUE klass)
{
 struct CharHolderData* chd;

 try
 {
   // Allocation d'une CharHolderData sur le tas,
   // c'est obligatoire...
   chd = new CharHolderData;
   int size = strlen(default_text);
   chd->length = size;
   size++; // 0 final ;)
   // Les deux lignes ci-dessous
   // pour test manque de mémoire
   // std::bad_alloc ba;
   // throw ba;
   chd->p = new char[size];
   strcpy(chd->p, default_text);
   std::cout << "Allocation memoire pour" << std::endl;
   std::cout << "la classe CharHolder effectuee" << std::endl;
 }
 // Ne pas laisser passer une exception C++
 catch(std::bad_alloc)
 {
   std::cerr << "Impossible allouer memoire pour CharHolder"
     << std::endl;
   // et à la place, lever une exception Ruby
   rb_raise(rb_eRuntimeError, "Exception Ruby, manque de memoire");
 }
 return Data_Wrap_Struct(klass, CharHolderMark, CharHolderFree,
 chd);
}

static VALUE GetAt(VALUE object, VALUE index)
{
 struct CharHolderData* chd;
 int at;
 char buffer[2];

 // Avant toute chose,
 // Vérification du type reçu
 Check_Type(index, T_FIXNUM);

 // récupérer les données maintenues dans la classe
 Data_Get_Struct(object, struct CharHolderData, chd);

 at = NUM2INT(index);
 if(at < 0 || at > chd->length-1)
   rb_raise(rb_eRuntimeError, "Exception Ruby, indice incorrect");

 // Fabriquer une chaine Ruby à partir d'un caractère
 buffer[0] = chd->p[at];
 buffer[1] = 0;
 return rb_str_new2(buffer);
}

static void Set(VALUE object, VALUE text)
{
 char* newtext;
 int newlength;
 struct CharHolderData* chd;

 // Avant toute chose,
 // Vérification du type reçu
 Check_Type(text, T_STRING);

 newtext = rb_string_value_cstr(&text);
 newlength = strlen(newtext);
 // récupérer les données maintenues dans la classe
 Data_Get_Struct(object, struct CharHolderData, chd);
 try
 {
   // Les deux lignes ci-dessous
   // pour test manque de mémoire
   // std::bad_alloc ba;
   // throw ba;

   char* t = new char[newlength+1];
   strcpy(t, newtext);
```

```
    // Si tout est en ordre valider
    // les résultats
    delete[](chd->p);
    chd->p = t;
    chd->length = newlength;
    std::cout << "Reallocation memoire CharHolder effectuee"
      << std::endl;
  }
  // Ne pas laisser passer une exception C++
  catch(std::bad_alloc)
  {
    std::cerr << "Impossible de reallouer memoire pour CharHolder"
      << std::endl;
    // et à la place, lever une exception Ruby
    rb_raise(rb_eRuntimeError,
      "Exception Ruby, manque de memoire");
  }
}

extern "C" void Init_Programmez()
{
  // Création du module
  ModuleProgrammez = rb_define_module("Programmez");

  // Création de la classe Ruby
  CharHolder = rb_define_class_under(
    ModuleProgrammez,
    "CharHolder", rb_cObject);
  // Définition de la méthode d'allocation
  rb_define_alloc_func(CharHolder, CharHolderAllocate);
  // Définition de la méthode GetAt
  rb_define_method(CharHolder, "GetAt",
    reinterpret_cast<VALUE (*)(...)>(GetAt), 1);
  // Définition de la méthode Set
  rb_define_method(CharHolder, "Set",
    reinterpret_cast<VALUE (*)(...)>(Set), 1);
}
```

Commençons par le commencement, à savoir le point d'entrée, qui est la fonction *Init_Programmez.* On y trouve d'abord la création du module. Rien de nouveau sur ce point par rapport aux exemples précédents. Vient ensuite la déclaration de classe de nom *CharHolder* au moyen de la fonction *rb_define_class_under.* L'appel retourne une VALUE, type fourre-tout de l'interpréteur RUBY comme nous l'avons mentionné plus haut. Le suffixe *under* du nom de fonction signifie en bon français "sous". Toute classe Ruby est sous-classe d'une autre classe, d'où la présence de ce suffixe. Ici il s'agit de la classe *cosmique Object* connue en permanence de l'interpréteur sous le nom de *rb_object.* Ensuite, et grâce à l'API *rb_define_alloc_func,* on attribue à la classe une fonction baptisée *CharHolderAllocate* et gérant la mémoire utilisée par notre code au sein de la classe. Enfin on définit les deux accesseurs de la classe.

6 GESTION DE MÉMOIRE UTILISATEUR SOUS INTERPRÉTEUR RUBY

Regardons maintenant attentivement le code de CharAllocate. D'abord, nous veillons à ne pas laisser en sortir une exception C++ en cas d'insuffisance mémoire. Si cette éventualité se produisait, cela mettrait fin à Ruby, alors que celui-ci continuera peut-être à fonctionner malgré tout. Donc toute exception C++ est capturée, puis on lève le cas échéant une exception Ruby en lieu et place. Ensuite nous allouons sur le tas, c'est évidemment obligatoire, une structure devant contenir les données que nous initialisons. Cette initialisation entraîne une autre allocation mémoire qui elle non plus ne doit pas déboucher sur une exception C++. Remarquons bien la toute dernière ligne de notre fonction. Grâce à un appel à l'API *Data_Wrap_Struct* nous associons notre structure de données à une fonction de marquage et une fonction de libération de ressources. Notre fonction de marquage ne fait rien. Mais, dans un autre contexte, nous devrions éventuellement y marquer comme utilisée une classe qui serait englobée par la nôtre. La fonction de libération de ressources est triviale. En interne Ruby fait un comptage de référence sur les objets qu'il gère. Quand le compteur tombe à zéro, l'interpréteur libère les ressources occupées par les objets, non sans avoir auparavant appelé les fonctions de libération de ressources détenues par les objets. C'est à ce moment là que notre fonction *CharHolderFree* sera appelée. L'accesseur *GetAt* illustre 4 points intéressants. D'abord, Ruby étant un langage dynamiquement typé, nous devons nous assurer, via la macro *Check_Type,* que l'utilisateur invoque notre méthode en lui passant un nombre entier propre sur lui. Ceci fait, nous récupérons un pointeur sur notre structure contenant nos données natives. Pour cela, nous employons l'API *Data_Get_Struct* qui est la réciproque de l'API *Data_Wrap_Struct* vue plus haut. Ensuite nous voyons comment convertir, au moyen de la macro NUM2INT, un nombre Ruby en un entier natif. Naturellement, nous testons maintenant de façon triviale la validité de l'indice et levons une exception Ruby si quelque chose cloche. Enfin nous voyons comment retourner notre caractère, ce qui revient à construire une chaîne Ruby de un caractère, au moyen de l'API *rb_str_new2.* Avec ce que nous savons, le code de notre accesseur *Set* est simple: vérification de type, récupération de valeur et de structure comme vu plus haut. Là encore, toutes les éventuelles exceptions C++ sont muselées. Enfin nous avons pris garde de ne bien réaffecter les valeurs dans notre structure, la longueur du texte notamment, qu'une fois la réallocation du tampon mémoire effectuée. Ceci afin d'être sûr que notre classe conserve un état cohérent même si la réallocation mémoire lève une exception C++. Sur le site, le lecteur trouvera un script baptisé *test.rb* qui teste les fonctionnalités de notre classe, ainsi que son comportement en cas d'erreur de typage, d'indice, etc. [Fig.2] Dans le source C++ DemoClasse.cpp, le lecteur trouvera à divers endroits deux lignes de code mises en commentaires :

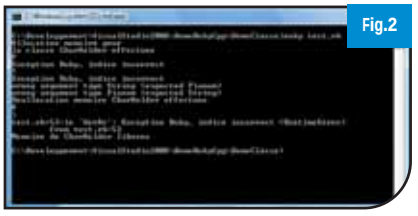


Fig.2

Comportement de notre classe sous le script test.rb.

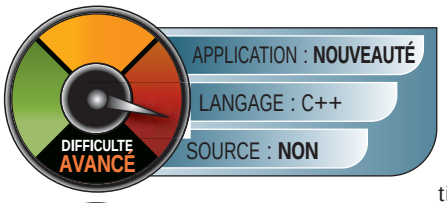
```
// std::bad_alloc ba;
// throw ba;
```

Ces lignes simulent un manque de mémoire. Retirez les commentaires, recompilez, et vous pourrez visualiser ce qui se passerait en cas de manque de mémoire dans l'interpréteur Ruby lors de l'initialisation ou de l'utilisation de notre classe.

■ Frédéric **Mazué**
fmazue@programmez.com

C++0x, une révolution dans le monde de C++

C++0x est le nom du prochain standard C++, devant sortir officiellement dans le courant de cette année 2009. Cette évolution est si riche qu'elle s'apparente plutôt à une révolution.



Au fil du temps C++ est devenu très riche, alliant la programmation de bas niveau et des possibilités incroyables avec des fonctionnalités telles que les templates. Le standard C++0x à venir, va encore considérablement enrichir le langage. Deviendra-t-il encore plus complexe, difficile et " piégeux " pour autant ? Il semble que non, et c'est encore un de ces paradoxes dont ce langage a décidément le secret. De nombreux nouveaux mots-clés, une syntaxe enrichie, de nouvelles fonctionnalités, et pourtant il devrait être plus confortable à l'utilisation et même plus performant à l'exécution, grâce, notamment, à l'introduction de la sémantique de déplacement (move semantic) en alternative à la sémantique de copie, seule supportée jusqu'à présent. Passer, même brièvement, en revue toutes les fonctionnalités nouvelles de C++0x est naturellement impossible ici. Nous préférons nous concentrer sur quelques-unes, particulièrement intéressantes.

LES CONCEPTS

Une des grandes forces de C++ est la programmation générique au moyen de templates. Une des grandes difficultés en C++ est ... l'écriture et l'utilisation des templates. Tout codeur C++ a un jour été confronté à cette situation: une erreur dans le code à template, qui ne compile donc pas, et voilà que le compilateur émet un flot intarissable de messages d'erreur absolument incompréhensibles. Une mésaventure qui a dégoûté bien des débutants et fatigue même les programmeurs expérimentés. Examinons avec un cas concret, une fonction générique retournant le plus petit des deux arguments reçus.

```
template<typename T>
  const T& min(const T& x, const T& y) {
  return x < y ? x : y;
}
```

Ce code très simple recèle le problème. Rien n'annonce à l'utilisateur de cette fonction que dans son corps l'opérateur < est employé. (Nous supposons pour l'exemple que cette fonction est complexe et fait partie d'une librairie dont nous avons la documentation, mais que le code est absolument illisible, comme tout bon code C++ à base de template :). Si l'utilisateur passe un type n'ayant pas d'opérateur <, le compilateur va se fâcher tout rouge au moment de l'instanciation du template. Ici l'exemple est certes simple, mais dans le cas général, l'absence de l'opérateur se manifeste dans les tréfonds de l'implémentation, d'où un flot intarissable de messages incompréhensibles. Le concept remédie à cela en définissant un contrat sur les types manipulés par la fonction générique, ou, exprimé autrement, une contrainte sur les types. Le créateur de la fonction est obligé de respecter le contrat, l'utilisateur également. Voici ce que cela donne pour notre exemple, et avec les deux nouveaux mots-clés *concept* et *requires* :

```
concept LessThanComparable<typename T> {
  bool operator<(const T& x, const T& y);
}

template<typename T>
requires LessThanComparable<T>
const T& min(const T& x, const T& y) {
  return x < y? x : y;
}
```

Le concept ci-dessus déclare qu'un type qui veut s'y conformer doit disposer de l'opérateur <. Ensuite nous revenons à notre fonction générique déclarée comme requérant que tout type T manipulé soit conforme au concept déclaré en amont. Passons du côté client. Il est naturel de vouloir invoquer notre fonction ainsi.

```
min(1, 2);
```

Cet appel qui compilait avec l'ancienne version de la fonction *min* ne compile plus, mais sans émettre des messages incompréhensibles. Le compilateur dira simplement qu'il ne sait pas que int répond au concept *LessThanComparable.* Nous devons l'en informer explicitement en déclarant une *concept map :*

```
concept_map LessThanComparable<int> { }
```

Le type intégré int disposant de l'opérateur < tout va bien. Par contre si nous donnons une *concept map* erronée :

```
concept_map LessThanComparable<std::complex<int> > {
}
```

Immédiatement le compilateur verra l'absence de l'opérateur, avant l'instanciation du template, donc sans entrer dans les abysses de l'implémentation, et pourra donc émettre un message clair. Bien entendu déclarer une *concept map* pour des cas aussi simples est lourd. C'est pourquoi une automatisation est prévue. On écrira :

```
auto concept LessThanComparable<typename T> {
  bool operator<(const T& x, const T& y);
}
```

Avec cet auto concept, c'est le compilateur qui générera les *concept maps* selon les besoins, soulageant ainsi l'utilisateur d'une besogne

fastidieuse. Revenons un instant du côté concepteur. Le concept garantit que le code n'utilise pas un opérateur imprévu à l'insu de notre plein gré. Ainsi si min est écrite ainsi :

```
template<typename T>
requires LessThanComparable<T>
const T& min(const T& x, const T& y) {
  return x > y ? y : x;
}
```

Le compilateur se plaindra immédiatement de l'usage de l'opérateur > qui ne figure pas dans la liste des requis. A nouveau le message d'erreur émis sera clair. On voit que ce mécanisme force autant le concepteur que l'utilisateur à respecter des contraintes sur un type, et ainsi une instanciation de templates ne peut plus échouer. Il est d'ores et déjà possible de se faire la main sur les concepts de C++ avec le compilateur expérimental ConceptGCC (http://www.generic-programming.org/software/ConceptGCC/)

LE MULTITHREADING

Avec la venue de ce nouveau standard, C++ commence à pallier une lacune: une bibliothèque standard trop éloignée des besoins industriels. Les détracteurs de C++ et partisans de Java ou C# mettent souvent en avant, non leur langage préféré, mais plutôt les bibliothèques qui l'accompagnent. C++ commence donc à enrichir ses bibliothèques. Bon nombre de ces enrichissements sont des emprunts à la fameuse librairie Boost (www.boost.org). Ainsi une extension pour le travail avec les expressions régulières, mais surtout une extension pour le travail avec les threads, ce qui à l'heure des processeurs multicoeurs était absolument indispensable. Cette extension permettra d'écrire du code portable au niveau source. Gros soulagement pour le développeur donc, qui ne devra plus se plonger dans les particularités des API des systèmes hôtes. Ceux qui sont familiarisés avec Boost vont s'y reconnaitre :

```
void worker();
std::thread t(worker);
```

Une fonction de travail 'worker' est ainsi exécutée dans un thread. Nous sommes en C++, donc la librairie de thread est écrite avec des templates et donc est générique. Plutôt qu'une fonction nous pouvons passer un objet fonction c'est-à-dire une instance d'une classe ayant un opérateur () :

```
class worker
{
  public:
  void operator()();
};
C++Ox: The Dawning of a New Standard
worker my_worker;
std::thread t(my_worker);
```

Nous sommes en C++, donc l'instance de my_worker est copiée lors de l'instanciation du thread. Il est possible de l'éviter ainsi :

```
std::thread t(std::ref(my_worker));
```

Encore plus intéressant, une fonction (ou un constructeur de classe) de thread peut recevoir des arguments :

```
void with_params(double d i,std::string s,std::vector<int> v);
std::thread t(with_params, 1.0,
  "Programmez!",std::vector<int>(1, 2 ,3));
```

Et bien entendu, cette extension pour la programmation multithread vient avec toute la panoplie de fonctionnalités de synchronisation requises, et que nous ne décrivons pas ici.

LES TEMPLATES VARIADIQUES

Cela n'a pas échappé à votre œil de lynx d'expert en C++, dans l'exemple ci-dessus le constructeur de la classe std::thread peut recevoir un nombre variable d'arguments. Exactement de la même manière que la fonction *printf* du C peut recevoir un nombre variable d'arguments. *printf* est une fonction variadique. Désormais, les entités génériques de C++ pourront être instanciées grâce à une nouvelle fonctionnalité pertinemment baptisée les templates variadiques :

```
template<class ... Types> class MaClasse { };

template<class ... Types> void f(Types ... args);
```

De quoi repousser encore plus loin les possibilités de C++ et donner du grain à moudre à des générations de développeurs et de sérieux maux de têtes aux implémenteurs :) Il reste énormément à dire sur C++0x, nous y reviendrons très certainement. Souhaitons que ce nouveau standard soir rapidement finalisé, et surtout rapidement pris en charge par les compilateurs.

■ Frédéric **Mazué** - fmazue@programmez.com

1re partie

Créer un jeu vidéo de A à Z avec XNA

Cette série d'articles a pour objectif de vous faire découvrir les bases de la programmation d'un jeu vidéo. Si l'univers est plus sympa qu'une application classique, cela demande toutefois beaucoup de travail et d'organisation. Pour un débutant, il vaut mieux être réaliste sur ses objectifs et boucler chaque projet de A à Z. Dans notre cas, nous partagerons notre expérience, autour d'un jeu de gestion de type Sim City / Civilisation. Nous développerons en C# et XNA.

Nous nous pencherons sur la programmation *gameplay* (c'est-à-dire la programmation qui touche directement la logique du jeu en elle-même). Nous donnerons dans ce premier article peu d'exemples de code (disponible sur le site www.programmez.com) afin de mieux pouvoir se concentrer sur les explications des concepts. C'est parti !

Premiers Objectifs

- Charger la carte de la France avec ses villes
- Se déplacer et zoomer sur la carte
- Construire quelques bâtiments avec des touches du clavier
- Sélectionner les villes et bâtiments

Le Game Design

Première chose à faire : avoir une idée ! Et consignée par écrit s'il vous plaît. Ce cahier des charges, ou encore « spécifications fonctionnelles », se nomme un *Game Design Document* dans le jargon. C'est un élément central du développement. Il faut par contre noter que celui-ci est très rarement fixe et évolue (très) régulièrement au cours du développement. Il est cependant préférable que les « principes » de base restent immuables. Une bonne pratique à adopter est de regrouper l'avis de personnes externes tout au long du projet : de l'idée sur le papier à sa finition. C'est ce qu'on appelle des « playtests ». Ils sont notamment très utiles pour valider ce que l'on a déjà fait et surtout prendre les mesures nécessaires pour corriger les défauts de conception le plus tôt possible.

Voici le concept de notre jeu résumé en quelques mots : le joueur doit satisfaire les besoins de plus en plus nombreux de la population par la construction de bâtiments adéquats tout en respectant l'environnement. Il devra également investir dans la recherche pour améliorer le rendement de ses infrastructures au cours du jeu.

Quelles différences par rapport à une application classique ?

Avant de commencer à décrire la programmation *gameplay,* un peu de théorie s'impose. Un fossé sépare la programmation classique (en général événementielle) de celle d'un jeu (en général séquentielle).

Les ressources

Un jeu est une application multimédia qui nécessite des images, des modèles 3D, des scripts, des sons et d'autres éléments extérieurs au programme. Les ressources représentent ces éléments chargés en mémoire par le jeu afin de les utiliser. Elles sont en général très gourmandes et nécessitent d'être bien gérées.

La boucle de jeu

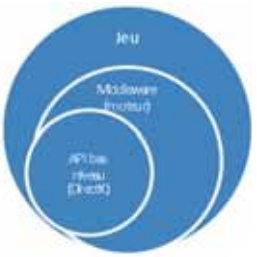
Un jeu, c'est simplement une grande boucle qui se répète jusqu'à la fin du programme. Ni plus, ni moins. De manière générale, elle exécute toujours la même chose dans un ordre bien précis :

- Mise à jour des entrées utilisateurs : clavier, souris, joystick, etc.
- Mise à jour des entités du jeu : création / destruction, déplacement, etc.
- Rendu à l'écran

Une machine à états

Tout d'abord qu'est-ce qu'un état de jeu? Et bien c'est une partie bien spécifique de l'application : l'introduction, le menu principal, les sous-menus, le jeu en lui-même, etc. Chaque état a une manière de fonctionner bien distincte mais exécute globalement toujours les mêmes fonctions (Draw, Update, etc.) décrites dans la boucle de jeu. De plus, chaque état nécessite de charger des ressources qui lui sont propres. Alors comment gérer ces cas-là ? La manière la plus simple de procéder est d'utiliser le pattern Etat.

Les différentes couches d'un jeu



De manière générale un jeu moyen se divise en plusieurs « couches ». Un schéma ci-contre vaut mieux que toute explication.

API bas niveau : ce sont les bibliothèques qui permettent en général d'effectuer le rendu, de lire des sons, de gérer les entrées utilisateurs ou bien même la physique. Elles sont généralement conçues pour une plate-forme bien précise. Par exemple pour le rendu, Direct X est utilisé pour Windows alors qu' Open GL est la référence sous Mac OS X et Linux.

Middleware : Communément appelé moteur, ce sont des bibliothèques de plus haut niveau qui ajoutent toutes les fonctionnalités dont un jeu peut avoir besoin, comme la gestion des entités de jeu,

DÉVELOPPEZ VOTRE SAVOIR-FAIRE

Programmez ! *est le magazine du développement*

Langage et code, développement web, carrières et métier : Programmez !, c'est votre outil de veille technologique.

Pour votre développement personnel et professionnel, abonnez-vous à Programmez !

ATTENTION ! Tarifs en hausse le mois prochain !

11 numéros par an : **45 €***

soit **3 Numéros GRATUITS**

**Tarif France métropolitaine*

Choisissez votre formule

- **Abonnement 1 an au magazine : 45 €**
  (au lieu de 65,45 € tarif au numéro) *Tarif France métropolitaine*
- **Abonnement Intégral : 1 an au magazine + archives sur Internet et PDF : 57 €** *Tarif France métropolitaine*
- **Abonnement PDF / 1 an : 30 €** *- Tarif unique*
  Inscription et paiement exclusivement en ligne
  www.programmez.com
- **Abonnement Etudiant : 1 an au magazine : 39 €**
  (au lieu de 65,45 € tarif au numéro) *Offre France métropolitaine*

+ Abonnement INTÉGRAL

ACCÈS ILLIMITÉ aux ARCHIVES du MAGAZINE pour 1€ par mois !

Cette option est réservée aux abonnés pour 1 an au magazine, quel que soit le type d'abonnement (Éco, Numérique, Etudiant). Le prix de leur abonnement normal est majoré de 12 € (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.

OUI, *je m'abonne*

Vous pouvez aussi vous abonner en ligne et trouver tous les tarifs www.programmez.com

PROGRAMMEZ

- ☐ **Abonnement 1 an au magazine : 45 €** (au lieu de 65,45 € tarif au numéro) *Tarif France métropolitaine*
- ☐ **Abonnement Intégral : 1 an au magazine + archives sur Internet et PDF : 57 €** *Tarif France métropolitaine*
- ☐ **Abonnement Etudiant : 1 an au magazine : 39 €** (au lieu de 65,45 € tarif au numéro) *Offre France métropolitaine*

☐ M. ☐ Mme ☐ Mlle Entreprise : ..... Fonction : .....

Nom : ..... Prénom : .....

Adresse : .....

Code postal : ..... Ville : .....

Tél : ..... E-mail : .....

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

A remplir et retourner sous enveloppe affranchie à :
Programmez ! - Service Abonnements - 22 rue René Boulanger - 75472 Paris Cedex 10.

abonnements.programmez@groupe-gli.com

PROgrammez ! Le magazine du développement

Offre limitée, valable jusqu'au 28 février 2009

Le renvoi du présent bulletin implique pour le souscripteur l'acceptation pleine et entière de toutes les conditions de vente de cette offre.

Conformément à la loi Informatique et Libertés du 05/01/78, vous disposez d'un droit d'accès et de rectification aux données vous concernant.
Par notre intermédiaire, vous pouvez être amené à recevoir des propositions d'autres sociétés ou associations.
Si vous ne le souhaitez pas, il vous suffit de nous écrire en nous précisant toutes vos coordonnées.

PROG 116

la caméra, la gestion des ressources, etc. Un middleware peut également ajouter un niveau d'abstraction supplémentaire qui permet de rendre le code du jeu portable sur différentes plates-formes sans se soucier des API bas niveau. *Ogre 3D* en est l'exemple parfait.
Jeu : ce sont les bibliothèques et l'exécutable qui contiennent le code du jeu à proprement parler. Ce code utilise de manière générale les fonctionnalités offertes par le middleware. Si celles-ci ne suffisent pas, on peut également définir ses propres outils bien spécifiques.

XNA dans tout ça ?

XNA englobe à la fois la couche *API bas niveau* puisqu'il repose directement sur Direct X et une partie du *middleware.* En effet, d'une part il étend les fonctionnalités de DirectX et en simplifie l'utilisation, d'autre part il propose un panel de classes de plus haut niveau qui permet par exemple de gérer facilement les ressources.
Cette plate-forme offre de multiples avantages :
- Gratuit avec Visual Studio C# Express
- Très Simple d'accès. On ne se concentre que sur le code du jeu
- Productif
- Cross Platform, c'est-à-dire qu'elle permet de développer à la fois sur PC et XBOX 360

XNA est un bon middleware pour débuter ou pour un projet de petite taille. Mais il a également ses limites dès lors que la taille du projet augmente. Il faut alors développer son propre moteur pour étendre ses fonctionnalités ou en utiliser un tiers déjà réalisé (comme le très connu Torque X par exemple).

Notre petit moteur maison : ECOT Engine

Pour notre projet nous avons choisi de développer un petit moteur qui est venu compléter les quelques lacunes de XNA dans certains domaines. Nous y avons ainsi rajouté par exemple une GUI (Graphic User Interface), une gestion des sprites (animation 2D), des états de jeu, un système de particules, un système de statistiques et d'autres outils bien utiles. Tout ceci est accessible via la classe Kernel. Par simplification, nous l'avons volontairement « lighter » mais il s'adaptera à la progression des articles.

A l'attaque !

Attaquons nous tout d'abord au développement du gameplay à proprement parler, c'est-à-dire des classes qui représentent directement des éléments concrets dans le jeu.
La carte (ou map) représente le terrain où les villes et bâtiments seront construits. Dans notre jeu, ce terrain est à l'échelle d'un pays et peut couvrir des régions aussi diverses que des plaines, des déserts, des montagnes ou bien des forêts. Il faut donc pouvoir modéliser cela de manière simple et générique. Pour cela nous utilisons ce que l'on appelle des *tiles*.
Une tile représente un « morceau » de terrain à afficher. Cela peut être de l'herbe, du sable, de l'eau mais aussi des arbres, des rochers ou bien même des éléments décoratifs. Ce sont les « briques » que nous allons assembler les unes à côté des autres dans une grille afin de construire notre map. Les tiles sont regroupés dans des fichiers nommés *tilesets.* C'est notamment une astuce très pratique pour la gestion de la mémoire puisqu'une fois la tile déjà chargée, son utilisation est peu coûteuse. Ce procédé a notamment été très utile sur les premières consoles.
Afin de pouvoir modéliser également la faune et la flore, nous allons utiliser deux couches de tiles pour le terrain :
- L'arrière-plan : herbe, sable, terre, eau
- Les éléments : montagnes, forêts, fleurs

Les éléments sont affichés par-dessus l'arrière plan. Passons maintenant au code. La classe *Tile* renseigne sur la texture associée (le tileset), la position du tile dans la texture et la famille du tile (son type) :

```
public class Tile
{
  public Texture2D Texture;
  public Point UVPosition;
  public TileFamily Family;

  public static float Size = 64;
}
```

Notre map sera modélisée par deux tableaux à deux dimensions contenant des tiles, un pour l'arrière-plan, l'autre pour les éléments :

```
private Tile[,] tiles;
private Tile[,] elements;
```

L'affichage

Pour afficher notre map à l'écran nous utilisons la méthode **Render** de la classe **Map.** XNA nous permet de dessiner facilement un élément 2D grâce à la classe **SpriteBatch.**
La map est en général plus grande que ce que nous pouvons afficher à l'écran, nous dessinons donc uniquement les tiles visibles pour l'optimisation.
Le procédé est également utilisé pour la 3D. Grâce à la position de la caméra obtenue par **Kernel.Instance.Origin,** au zoom obtenu par **Kernel.Instance.Camera.Zoom** et à la taille de l'écran obtenue par **Kernel.Instance.Width** et **Kernel.Instance.Height,** nous sommes capables de sélectionner les bons tiles.

```
public void Render(GameTime gameTime)
{
  SpriteBatch spriteBatch = Kernel.Instance.SpriteBatch;

  int x = (int)(Kernel.Instance.Origin.X / Tile.Size);
  int xLimit = Math.Min(x + (int)Math.Ceiling(Kernel.Instance.Width
 / Tile.Size / Kernel.Instance.Camera.Zoom) + 1, this.size.Width);
  int yBeg = (int)(Kernel.Instance.Origin.Y / Tile.Size);
  int yLimit = Math.Min(yBeg + (int)Math.Ceiling(Kernel.Instance.Height
 / Tile.Size / Kernel.Instance.Camera.Zoom) + 1, this.size.Height);

  spriteBatch.Begin(SpriteBlendMode.AlphaBlend, SpriteSortMode.
Immediate, SaveStateMode.None, Kernel.Instance.Camera.TransformMatrix);

  // Dessine les tiles uniquement visibles
  for ( ; x < xLimit; x++)
    for (int y = yBeg ; y < yLimit; y++)
      if (this.DrawTile(MapLayer.Background, x, y, 1))
        this.DrawTile(MapLayer.Elements, x, y, 1);

  spriteBatch.End();
}
```

Intégration

Ok, nous avons à présent les éléments de base de notre gameplay c'est-à-dire pour le moment notre map. Il faut maintenant évidement l'intégrer à notre fameuse boucle de jeu, rendre tout cela interactif et gérer l'affichage.

Des classes pratiques

Dans un jeu, nous avons souvent besoin d'accéder rapidement à tout un panel d'autres classes. Nous ne pouvons pas toutes les passer en paramètre, cela polluerait le code. Nous utilisons donc, de manière raisonnable le pattern *Singleton* pour les classes importantes. Le code du jeu en lui-même n'est donc pas écrit directement dans l'état de jeu. Nous avons créé la classe singleton **GameController** pour la logique et **GameRenderer** pour le rendu.

Le GameController

C'est une des classes principale du jeu, c'est elle qui fait l'interface entre toutes les autres classes du jeu. Lorsqu'on l'initialise, le **GameController** s'occupe grossièrement de charger la map et de placer la caméra :

```
public void Initialize()
{
  // Charge la map
  map = new Map(string.Format(@"Maps\{0}.etm", mapName));

  // Centre la caméra sur la map, assigne le zoom et les limites
  Kernel.Instance.Camera.Position = new Vector2(map.Size.Width
  * Tile.Size / 2, map.Size.Height * Tile.Size / 2);
  Kernel.Instance.Camera.Zoom = 0.8f;
  Kernel.Instance.Camera.MinLimit = new Vector2(0.0f, 0.0f);
  Kernel.Instance.Camera.MaxLimit = new Vector2(
    map.Size.Width * Tile.Size,
    map.Size.Height * Tile.Size);

  Kernel.Instance.Cursor.Enable = true;

initialized = true;
}
```

Les états de jeu

Les états de jeu sont gérés par notre classe **StateManager** qui permet de passer d'un état à l'autre. Cette classe hérite directement de la classe **DrawableGameComponent** fournie par XNA. Contentons nous de savoir qu'un **DrawableGameComponent** est mis à jour et affiché automatiquement par XNA. Dans notre petit jeu, nous allons pour l'instant créer deux états :

- L'état Intro, que nous ne décrirons pas ici
- L'état gameplay

Nous allons donc créer la classe **GameplayState** qui hérite de la classe **GameState** de notre moteur. C'est dans cette classe que toutes les mises à jour et le rendu du jeu « principal » auront lieu. Elle possède 5 méthodes importantes :

- LoadContent : qui permet de charger les ressources en mémoire et d'initialiser l'état.
- UnloadContent : qui fait l'inverse.
- HandleInput : qui permet de récupérer les entrées utilisateur.
- Update : qui permet de mettre à jour la logique.
- Draw : qui permet d'afficher les éléments à l'écran.

Nous surchargeons les méthodes **LoadContent** pour charger notre map en mémoire :

```
public override void LoadContent(bool loadAllContent)
{
  // Crée le controller
  new GameController("France").Initialize();

  // Crée le renderer
  new GameRenderer();

  base.LoadContent(loadAllContent);
}
```

La méthode **Update** se contente d'appeler celle de notre **GameController,** de même que la méthode **Draw** appelle celle de notre **GameRenderer.**

La classe Game

Grâce à XNA, il n'y a plus de boucle de jeu à gérer manuellement. Il suffit de dériver la classe Game qui propose des méthodes de base, que nous avons juste à remplir, un peu comme notre classe **GameState.**

```
protected override void BeginRun()
{
// Initialise le moteur
engine.Initialize("Bootstrap", "ECOT.Game.Text");

// Crée les états de jeu qui s'enregistrent automatiquement
new EcothinkState();
new GameplayState();

// Définit l'état de jeu par défaut
engine.StateManager.SwitchTo("Ecothink", null);

base.BeginRun();
}
```

Et c'est tout ! La mise à jour et le rendu se feront automatiquement du fait que notre classe **StateManager** est un composant de jeu affichable.
Je vous invite à regarder le code qui se trouve sur le site www.programmez.com afin de mieux comprendre tous les mécanismes que nous ne pouvons expliquer ici. J'espère que cet article vous aura donné un avant- goût de ce qui vous attend dans les prochains articles.

LIENS UTILES

UML : http://smeric.developpez.com/java/uml/etat/
PlayTest : http://www.bruno-urbain.com/html/10tipsforplaytest.htm
Tutoriaux : http://www.riemers.net/
Creator Club : http://creators.xna.com

Lauréat de l'Imagine CUP dans la catégorie jeu vidéo et soutenue par Microsoft, ECOThink développe des jeux vidéo innovants grâce à la technologie XNA. contact@ecological-tycoon.com

■ Frédéric **PEDRO** & Nicolas **GRYMAN**

Google Adwords, analytics et AdSense

*Difficulté : ****
Editeur : Eni éditions
Auteur : J-N Anderruthy
Prix : 20,95 €

Google, outre le moteur de recherche, fournit de très puissants outils de marketing en ligne pour créer des campagnes de publicité en ligne, avoir de la publicité et pour finement analyser le trafic de son site. Le but est de générer du trafic et du business. Le livre se destine à l'ensemble des développeurs web, du novice aux experts sans oublier le webmaster. Nous verrons tout d'abord comment ne pas perdre d'argent lors de la mise en place d'une campagne réussie avec Google AdWords et comment optimiser les coûts et les conversions afin d'en maximiser la rentabilité. Le lecteur comprendra comment utiliser l'optimiseur de campagnes et comment régler au mieux chaque fonction pouvant influencer la campagne. La troisième partie du livre est consacrée à AdSense et à la création d'un système publicitaire faisant la promotion de vos propres produits ou de ceux d'un annonceur tiers. A lire et à relire !

Programmation Flex 3

livre du mois

*Difficulté : *** - Editeur : Eyrolles*
Auteur : Aurélien Vannieuwenhuyze - Prix : 35 €

Avec Flex 3, Adobe propose une plate-forme technologique complète alliant avec Air, la partie offline, et l'animation avec Flash. Doté de puissants services serveurs, Flex est sans doute la plate-forme internet riche la plus complète, la plus mature ! Mais la maîtriser nécessite une solide expérience et une connaissance approfondie de Flex Builder et d'ActionScript. L'ambition de l'auteur est de brasser l'ensemble de la technologie : le SDK, le builder. Ainsi on prend connaissance des contrôles, de l'accès aux données, des conteneurs, de l'interface, des skins, etc. Air est bien entendu abordé. La bonne idée du livre est de présenter des cas concrets de mise en œuvre pour mieux comprendre l'utilité de Flex et de certaines fonctions. L'auteur aborde aussi le futur Flex 4, une excellente idée ! Au final, un bon ouvrage que tout développeur Flex doit lire et posséder.

Programmer objet avec Oracle

*Difficulté : ****
Editeur : Vuibert
Auteur : Christian Soutou
Prix : N.C.

On a tendance à oublier que Oracle n'est pas seulement un SGBD, c'est aussi une plate-forme de développement comme peut l'être SQL Server avec Net. L'auteur, spécialiste reconnu, revient sur les larges possibilités objet d'Oracle, de la v9 à la v11g., et force est de reconnaître que l'objet est fortement présent dans Oracle. On saura même stocker des objets, réaliser rapidement un mapping avec JDBC ou utiliser Oracle XML DB. Une référence !

Le web : 15 ans déjà et après ?

*Difficulté : **
Editeur : Dunod
Auteur : J-P Corniou
Prix : 19,90 €

Né un peu par " hasard ", il est aujourd'hui incontournable, même s'il est fragile dans ses fondations... Souvent qualifié de révolution virtuelle, le web a pourtant un impact réel sur l'économie, l'emploi. Dans un style clair et vivant l'auteur explique comment le " réseau " s'est construit et quels ont été les boosters de ce formidable décollage : moteurs de recherche, messagerie électronique, e-commerce...Il nous conduit à réfléchir aux mutations que le web a introduites dans nos vies et dans la société.

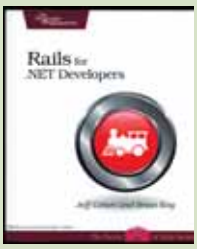
Sharepoint 2007

*Difficulté : ****
Editeur : éditions Eni
Auteur : collectif
Prix : 39 €

Sharepoint est un serveur puissant mais souvent méconnu. Ce livre vous plonge au cœur de sharepoint, du développement .Net et des composants pour personnaliser Office SharePoint Server. On comprendra mieux webparts, les flux, les formulaires, etc. La partie déploiement dévoile les réflexes à avoir, les bonnes pratiques pour réussir. Les deux derniers chapitres permettent au lecteur d'envisager l'utilisation des dernières technologies (.NET 3.5) et des techniques de développement plus avancées.

en anglais

Rails for .Net Developers



*Difficulté : ****
Editeur : O'Reilly
Auteur : collectif
Prix : 28 €

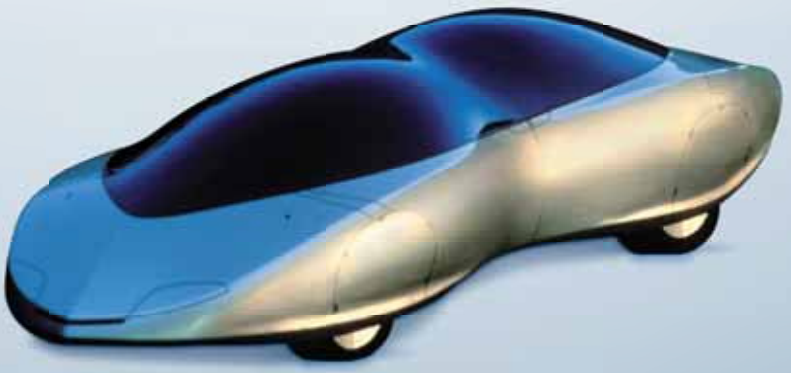
.Net est la nouvelle terre de prédilection de Ruby, notamment avec le projet IronRuby permettant d'intégrer Ruby à la plate-forme. Cet ouvrage vise à comprendre comment utiliser Ruby dans et avec .Net, ainsi que Rails. Les auteurs abordent bien entendu les aspects MVC et Crud (ActiveRecord), sans oublier les formulaires, les tests et l'interaction avec Ajax.

Developing FaceBook Platform Applications with Rails

*Difficulté : ****
Editeur : O'Reilly
Auteur : M. J. Mangino
Prix : 26 €

Facebook est l'un des réseaux sociaux les plus en vogue actuellement. Et grâce aux API de la plate-forme, on peut rajouter ses propres applications. Dans cet ouvrage, l'auteur explique comment coder pour Facebook en Rails ! On revient tout d'abord sur ce qu'est la plate-forme sociale et les applications qui la forment. Puis on attaque directement avec une première application. Au-delà de l'exemple assez simple de départ, il convient de mieux intégrer les fonctions sociales de Facebook comme la notification, les commentaires, les invitations, etc. Rapide à lire et agrémenté de nombreux codes, le développeur web y trouvera une bonne base de départ.

EVOLUTION
REVOLUTION



FAITES EVOLUER VOTRE CODE.

L'innovation en développement parallèle.

Analysez, compilez, déboguez, contrôlez et optimisez votre code pour le multicore avec Intel® Parallel Studio. Fait pour les applications sérielles d'aujourd'hui et les innovations parallèles de demain.

Pour en savoir plus et tester la version beta: **www.intel.com/go/parallel**

intel
Software
Products