Developpez

Le Mag

Édition de aout - septembre 2013.
Numéro 47.
Magazine en ligne gratuit.

Réalisation : Alexandre Pottiez
Rédaction : la rédaction de Developpez
Contact : magazine@redaction-developpez.com

Sommaire

Éditorial

Voici un nouveau numéro de votre magazine préféré incluant les meilleurs ressources des rubriques de Developpez.com

Profitez-en bien !

La rédaction

Article 2D/3D/Jeux

De XNA à MonoGame

Comment passez de XNA à MonoGame.

par **Alexandre Laurent**
Page 13

Article Développement Web

Le webdesign selon les grilles

Cette méthode de travail pourtant répandue dans le domaine de l'architecture et de la typographie est tout simplement mal connue dans le webdesign.

par **Kévin Pinto**
Page 38

Access 2013 et application Web - L'éditeur de macros d'une application Web avec Access 2013

Nous allons aborder l'éditeur de macros Access 2013 dans une application Web.

1. Introduction

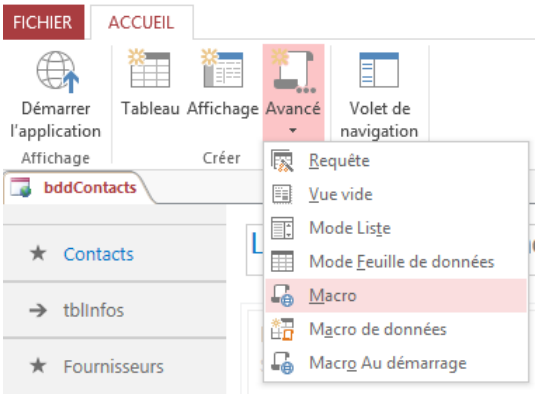
Nous allons dans cet article aborder l'éditeur de macros d'Access 2013 avec une application Web. À la différence d'une application bureau, une application Web d'Access 2013 possède trois types de macros :

- macros Web : celles-ci filtrent les actions et les expressions qui ne peuvent pas être utilisées sur le Web ;
- macros de données : permet d'ajouter, modifier les données des tables (tableaux) ;
- macros au démarrage : permet de configurer l'application en initialisant des variables, valeurs par défaut, l'accès à des vues spécifiques. Celle-ci s'exécute au démarrage de l'application.

Cet article fait suite de celui-ci : Démarrage d'une application Web avec Access 2013 (Lien 01).
Nous allons continuer sur le même exemple que l'article précédent.

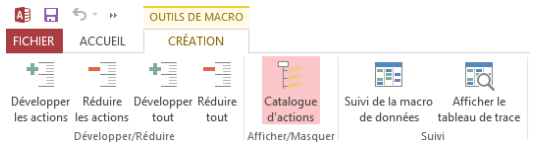
2. Créer une nouvelle macro

Pour créer une nouvelle macro, sélectionnez depuis l'onglet **Accueil** du **Ruban Access** le menu **Avancé** et choisissez le type de macro voulue.



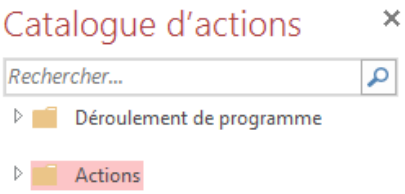
Lors de la création d'une macro, la vue est séparée en trois parties :

Ruban de création :



- Les quatre premières commandes permettent de réduire/développer les actions ou l'ensemble de la page.
- Le catalogue d'actions permet d'afficher/masquer le volet de droite reprenant toutes les actions.
- Les deux dernières commandes permettent de lancer ou non le suivi de la macro de données ou d'afficher dans une table les résultats du suivi.

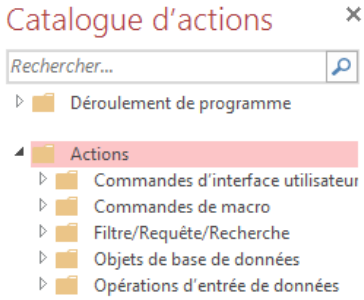
Catalogue d'actions :



Le catalogue d'actions est composé de deux éléments :

- déroulement de programme ;
- actions.

L'élément action est scindé en plusieurs sous-éléments.

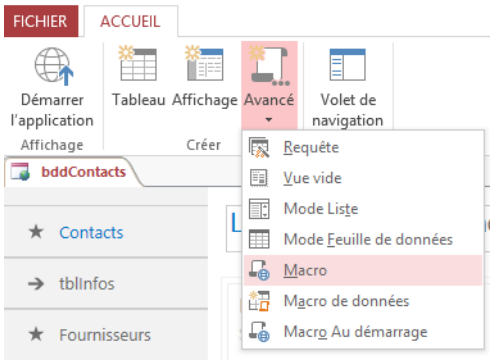


Tous ces sous-éléments regroupent les actions.
Vue de création des macros :

Une liste est disponible pour sélectionner les actions voulues.

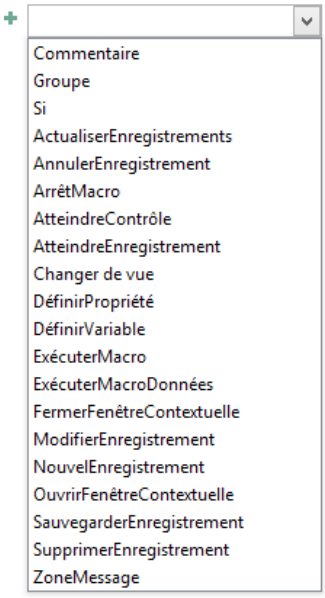
3. Macro

Depuis le menu **Avancé** sélectionnez **Macro**.

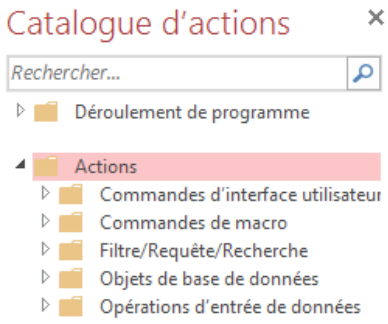


Pour créer une action, deux possibilités s'offrent à vous :

- depuis la liste déroulante qui regroupe l'ensemble des actions disponibles

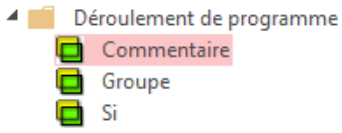


- depuis le catalogue d'actions par un glissé-déposé.



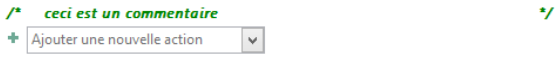
3.1. Déroulement du programme

Ce groupe est commun aux trois types de macros.



3.1.1. Commentaires

Cette action permet d'écrire des informations dans votre programme. Ceci est pratique pour donner des informations sur les actions suivantes.

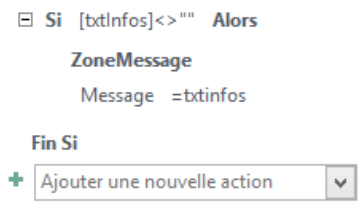


3.1.2. Groupe

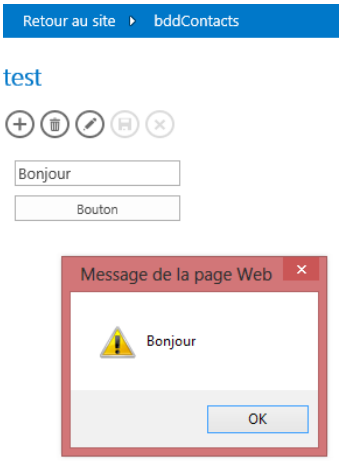
Ceci permet de créer un bloc regroupant plusieurs actions. Ceci permet de donner de la visibilité au programme en permettant de réduire l'ensemble du groupe.
Sur l'image ci-dessous le groupe est réduit.

3.1.3. Si

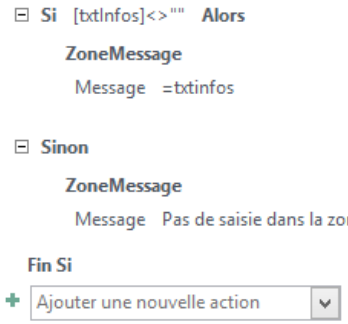
Permet de faire un test logique et oriente l'exécution des actions en fonction. Cette fonction action a son équivalent en VBA : IF ... THEN . ELSE.
Ici le SI permet de faire un test logique (variable, contrôle ou expression). Si le test est vérifié, alors, on exécute les actions qui suivent.
Exemple 1 : **Si** la Zone de Texte comporte des caractères **Alors** on affiche un message avec le contenu de la zone



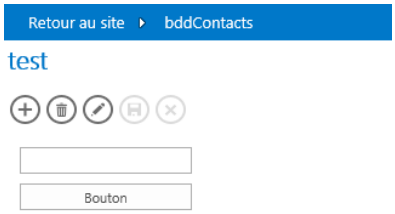
Voici le résultat :

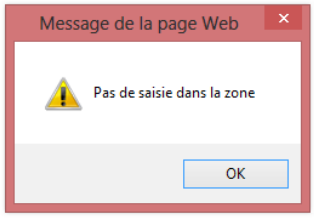


Exemple 2 : **Si** la Zone de Texte comporte des caractères **Alors** on affiche un message avec le contenu de la zone **Sinon** on affiche un message signifiant que rien n'a été saisi dans la zone de texte.



Voici le résultat :





Il est possible de lancer un second test logique si le premier n'est pas vérifié. Pour cela, utiliser Sinon Si.

3.2. Actions

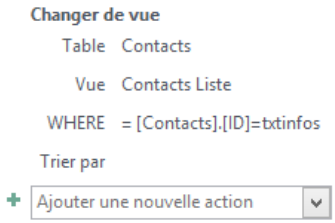
3.2.1. Commandes d'interface utilisateur

3.2.1.1. Changer de vue

Cette action permet de naviguer vers une autre **Vue** avec la possibilité de Filtrer ou non et de Trier ou non.
Vous devez définir :

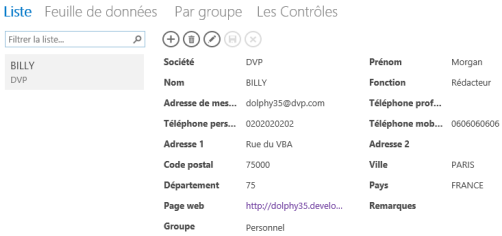
- Table : (champ obligatoire) ceci définit la table de la **Vue ;**
- Vue : (champ obligatoire) nom de la **Vue** à afficher ;
- WHERE : (champ facultatif) clause permettant de filtrer l'affichage de la **Vue** selon un champ ou une chaine SQL ;
- Trier par : (champ facultatif) permet de trier l'affichage selon un champ.

Dans notre exemple, nous allons afficher la **Vue** Contacts Liste basée sur la Table **Contacts** en filtrant par **l'ID** du contact qui sera saisi dans une zone de texte.



Voici la première **Vue** d'où est lancée la macro :

Et le résultat : la **Vue** est filtrée sur **l'ID** 3 :

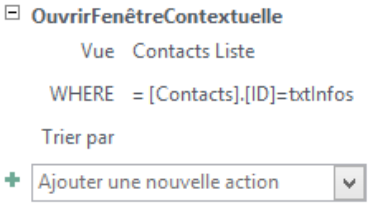


3.2.1.2. FermerFenêtreContextuelle

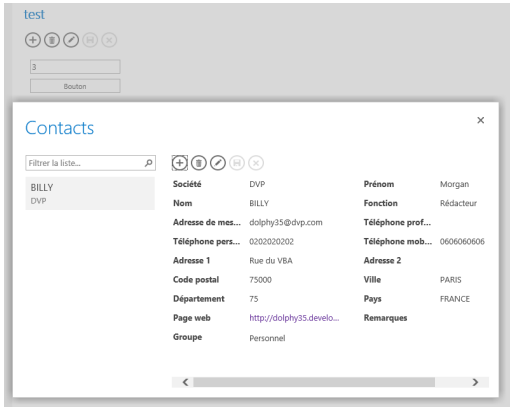
Cette action permet de fermer la fenêtre contextuelle active.

3.2.1.3. OuvrirFenêtreContextuelle

Cette action permet d'ouvrir une **Vue** dans une fenêtre contextuelle. Vous devez renseigner le nom de la **Vue** et la clause **WHERE** si vous voulez filtrer les données.
Dans notre exemple, nous allons reprendre le même filtre que l'on a fait avec l'action **Changer de Vue**, mais ici la **Vue** s'affichera dans une **Vue** contextuelle. Il est recommandé de mettre le nom de la table avec le nom du champ, même si cela fonctionne sans.



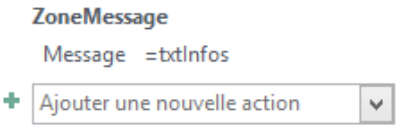
Voici le résultat :



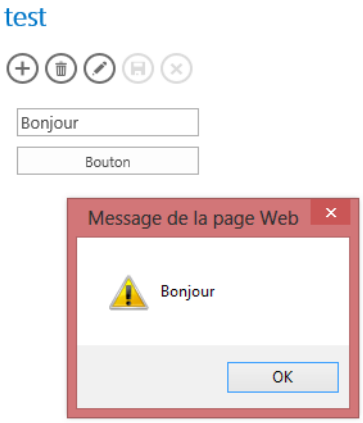
3.2.1.4. ZoneMessage

Cette action permet d'afficher un message d'avertissement

ou d'information.
Ici, nous allons afficher le contenu de la zone de texte :



Voici le résultat :



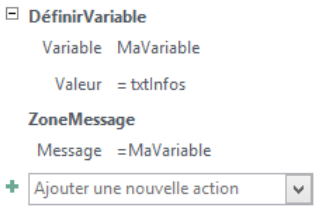
3.2.2. Commandes de macro

3.2.2.1. ArrêtMacro

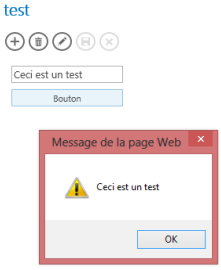
Cette action permet d'arrêter la macro en cours d'exécution.

3.2.2.2. DéfinirVariable

Cette action permet de charger une donnée dans une variable qui peut être réutilisée.
Pour l'exemple, nous allons charger la valeur de la zone de texte dans une variable et nous allons afficher la valeur de la variable dans un message.

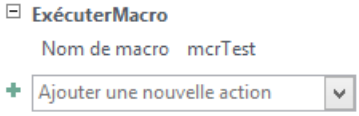


Voici le résultat :



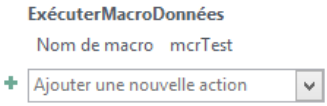
3.2.2.3. ExécuterMacro

Cette action permet de lancer une macro déjà enregistrée.



3.2.2.4. ExécuterMacroDonnées

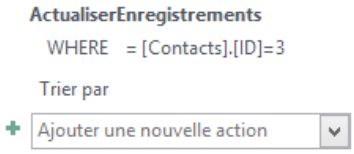
Cette action permet de lancer une macro de données déjà enregistrée.



3.2.3. Filtre/Requête/Recherche

3.2.3.1. ActualiserEnregistrement

Cette action permet l'actualisation de l'enregistrement actuel, il est possible d'ajouter une clause WHERE pour filtrer le jeu.
Dans l'exemple, lors de l'actualisation, nous effectuons un filtre sur l'ID de la table Contacts.



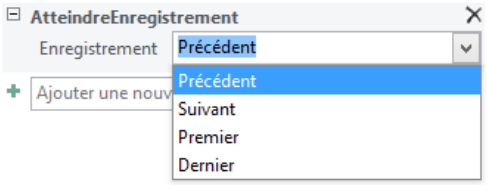
3.2.4. Objets de base de données

3.2.4.1. AtteindreContrôle

Cette action permet de placer le focus sur le contrôle spécifié. Le curseur de la souris se placera sur ce contrôle.
Dans l'exemple suivant, nous activons la liste déroulante lors du clic sur le bouton.

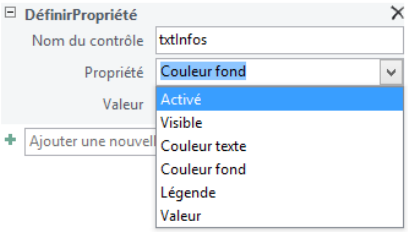
3.2.4.2. AtteindreEnregistrement

Cette action permet de naviguer dans le jeu d'enregistrements, vous pouvez passer à l'enregistrement Précédent/Suivant/Premier/Dernier.



3.2.4.3. DéfinirPropriété

Permet de définir une propriété du contrôle.
Voici les propriétés disponibles :



Dans notre exemple, la couleur de fond de la zone de texte est changée sur le clic du bouton.

Voici le résultat

Pour les propriétés définissant une couleur, la valeur est représentée en hexadécimal. Pour les autres, 0 ou -1.

3.2.5. Opérations d'entrée de données

Ces actions apportent des modifications aux données.

3.2.5.1. AnnulerEnregistrement

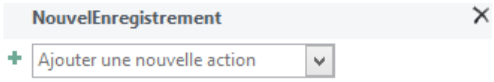
Cette action annule les modifications de l'enregistrement actif.

3.2.5.2. ModifierEnregistrement

Cette action modifie l'enregistrement actif.

3.2.5.3. NouvelEnregistrement

Cette action crée un nouvel enregistrement.

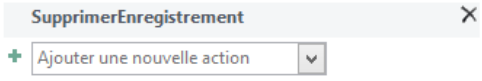


3.2.5.4. SauvegarderEnregistrement

Cette action sauvegarde les modifications de l'enregistrement actif.

3.2.5.5. SupprimerEnregistrement

Cette action supprime l'enregistrement actif.

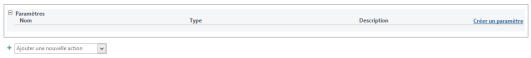


3.2.6. Dans cette base de données

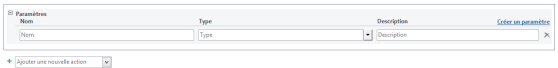
Dans ce groupe, vous retrouvez tous les éléments enregistrés dans l'application.

4. Macro de données

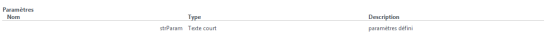
Lors de la création d'une macro de données, une partie supérieure fait son apparition. La partie paramètre. Cette partie permet de définir des paramètres qui seront demandés à l'appel de la macro.



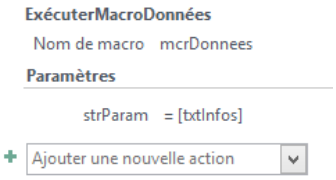
Pour créer un paramètre, cliquez sur le lien **Créer un paramètre** en haut à droite.



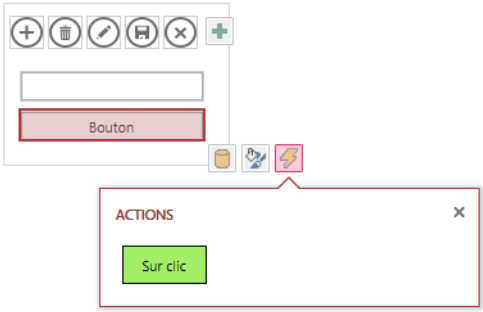
Il ne vous reste plus qu'à définir son **Nom**, le **Type** du paramètre et une description si vous voulez donner des détails.
Ici nous allons créer un paramètre de texte court (String).



Lors de l'appel de la macro de données, il vous est demandé de donner la valeur du paramètre.



Ici nous passons en paramètre de la macro de données la valeur de la zone de saisie. Cette macro est exécutée lors de l'appui d'un bouton dans une vue quelconque.



4.1. Déroulement de programme

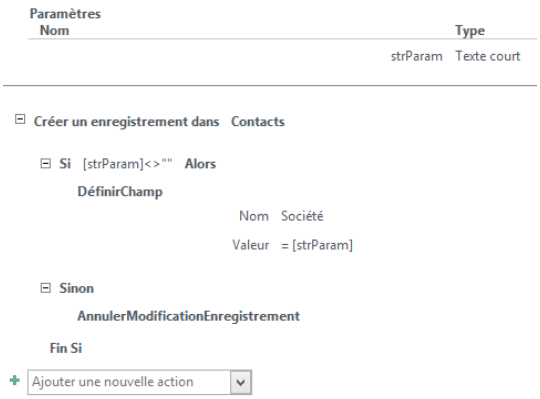
Cette partie est commune à la précédente.

4.2. Blocs de données

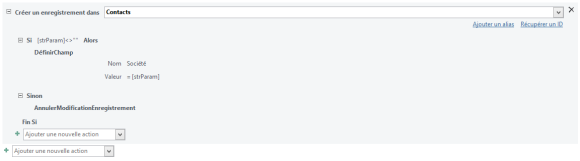
Un bloc de données est un groupement d'actions qui réalisera l'Action du Bloc.
Par exemple, pour la création de données, vous pouvez effectuer un test pour créer un enregistrement.

4.2.1. CréerEnregistrement

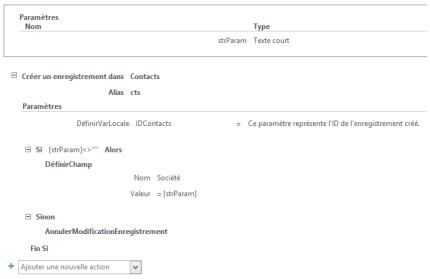
Ce bloc permet de créer un enregistrement dans la table et champ définis.



Ici, le bloc ajoute un enregistrement dans la table Contacts, pour ajouter cet enregistrement un test est réalisé sur le paramètre. Si celui-ci comporte des données, on ajoute la valeur du paramètre dans le champ Société de la table Contacts, sinon on ne fait rien (on annule).
Lorsque vous cliquez sur le nom de la table, vous distinguez deux liens en haut à droite. Ceux-ci permettent de définir :

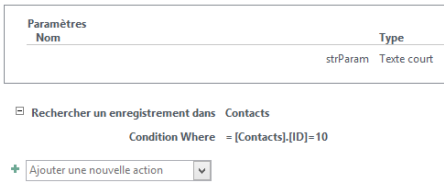


- Alias : nom donné au nouvel enregistrement ;
- Récupération ID : ceci permet de charger dans une variable locale l'ID du nouvel enregistrement.



4.2.2. RechercherEnregistrement

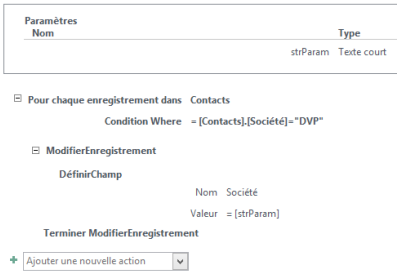
Ce bloc seul ne réalise pas d'action. Il permet de rechercher un enregistrement dans le jeu d'enregistrements.
Attention : si plusieurs enregistrements correspondent, seul le premier sera pris en compte. Si vous voulez modifier l'ensemble des enregistrements, il vous faudra utiliser : **PourChaqueEnregistrement**.
Vous devez définir la table servant de jeu d'enregistrements et la clause **Where** permettant de filtrer sur un enregistrement dans le jeu.



Ici nous sélectionnons l'enregistrement dans la table **Contacts** ayant pour ID 10.

4.2.3. PourChaqueEnregistrement

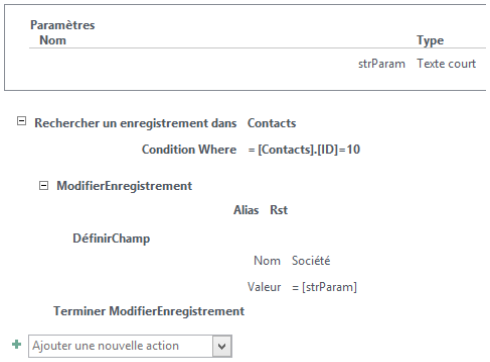
Ce bloc seul ne réalise pas d'action. Il permet de faire une boucle pour chaque enregistrement répondant à la clause **Where** (filtre).
Dans l'exemple, nous allons modifier le nom de la Société pour tous les enregistrements ayant pour nom de Société « DVP » par la saisie de la zone de texte.



Ici aussi vous pouvez définir un alias à l'enregistrement modifié.

4.2.4. ModifierEnregistrement

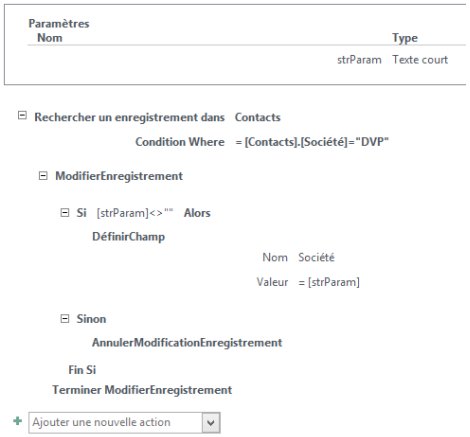
Ce bloc associé au précédent permet de faire une modification dans un enregistrement.
Le bloc **RechercherEnregistrement** permet d'instancier l'enregistrement dans le jeu et le bloc **ModifierEnregistrement** permet de modifier des colonnes de cet enregistrement.
Dans notre exemple nous allons modifier l'enregistrement ayant pour ID 10, la colonne société par le paramètre de la macro de données (à savoir la zone de saisie).



Ici aussi vous pouvez définir un alias à l'enregistrement modifié.

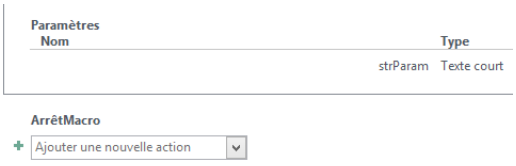
4.2.5. AnnulerModificationEnregistrement

Cette action permet d'annuler le bloc **ModifierEnregistrement** ou **CréerEnregistrement**.
Dans notre exemple, nous allons effectuer un test du paramètre. Si celui-ci présente des données, on modifie l'enregistrement actuel, sinon on annule la modification



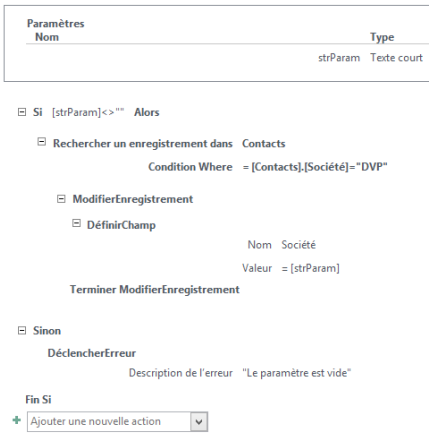
4.5.6. ArrêtMacro

Cette action termine la macro en cours.

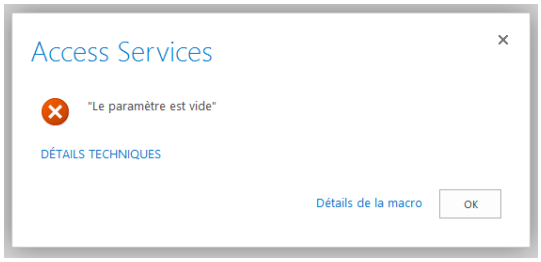


4.2.7. DéclencherErreur

Cette action permet de déclencher un avertissement sous la forme de message d'erreur. Ceci peut vous servir pour avertir de la non-réalisation du bloc.
Dans notre exemple, si le paramètre est vide, une erreur s'affiche.

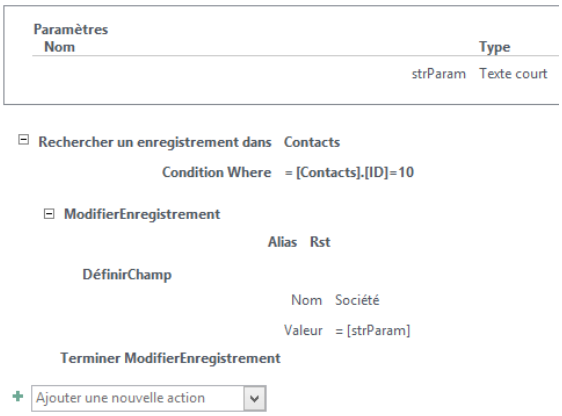


Voici le résultat du déclenchement



4.2.8. DéfinirChamp

Définit la valeur d'un champ sur le résultat d'une expression.
Dans l'exemple suivant, le champ Société est modifié par la valeur de la zone de saisie sur l'enregistrement répondant aux critères de la clause.



Retrouvez la suite de l'article de Morgan Billy en ligne : Lien 02

Compte rendu de la keynote WWDC d'Apple

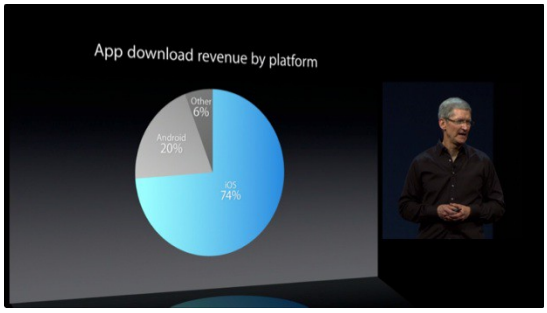
Pour la 24e WWDC, Apple a mis à disposition 100 sessions, 124 laboratoires et 1000 ingénieurs.
Cette keynote commence par l'arrivée de Tim Cook qui présente ses chiffres avant de passer à OS X, aux nouvelles machines et enfin à iOS.

1. Les chiffres

Comme habituellement dans toutes les keynotes Apple, la présentation commence par quelques chiffres, que ce soit sur les ventes de Mac, les Apple Store ou les applications iOS. Voici les quelques chiffres de cette soirée :

1 million de visiteurs par jour dans les Apple Store
407 boutiques dans 14 pays
900 000 applications iOS
5 milliards d'applications téléchargées
575 millions de comptes iTunes Store
10 milliards de dollars pour les développeurs
72 millions de Mac installés
28 millions de Mountain Lion

Les utilisateurs d'iPhone passent en moyenne 50 % plus de temps sur leur smartphone que les utilisateurs d'Android.
60 % du trafic mobile sur le web pour iOS contre 24 % pour Android et 16 % pour les autres
82 % du trafic tablette sur le web pour l'iPad contre 18 % pour les autres.
Il montre des chiffres indiquant que les iPad et iPhone représentent la majeure partie du trafic généré par le shopping mobile avec respectivement 10 % et 8.7 % du trafic contre 5.5 % pour Android durant la plus mauvaise journée de 2012.
Apple revendique le meilleur taux de satisfaction client du marché avec 97 % de taux de satisfaction et 73 % de clients très satisfaits pour iOS contre 53 % pour Windows Phone, 49 % pour Android et 32 % pour BlackBerry.



2. OS X 10.9 alias OS X Mavericks

Présentation de la prochaine version d'OS X, Apple présente **Mavericks** (spot de surf californien) et abandonne ses félins.

Dans cette nouvelle version, le Finder a le droit à l'arrivée de l'option plein écran et des onglets. Fini les nombreuses fenêtres de Finder ouvertes à droite et à gauche.

« *Tags* », un système d'organisation par tags qui permet de tagger des documents afin de les retrouver facilement dans les dossiers intelligents du Finder.

Modification de Mission Control pour permettre la gestion du multiécran.

Amélioration des performances, de l'autonomie, mémoire compressée, l'arrivée de l'**OpenGL 4** et la possibilité de mettre en pause les applications et les plugins.

Safari reçoit une légère modification d'interface pour « *Top Site* » et « *Read List* », amélioration de la ressource mémoire, blocage des plugins. Safari serait **1,4 fois plus rapide** que Chrome pour le JavaScript (SunSpider) et la consommation de CPU baisserait immédiatement au passage en arrière-plan.
On a maintenant la possibilité de synchroniser les mots de passe via iCloud ainsi que les mots de passe Wi-Fi.

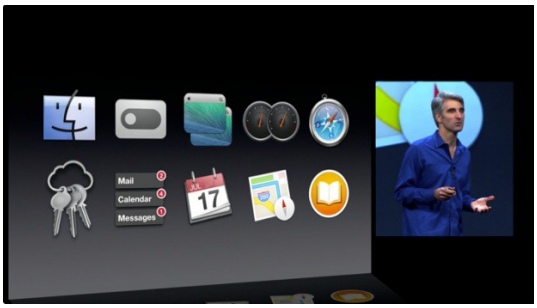
Les notifications OS X n'ont pas de mise à jour graphique, mais on a maintenant la possibilité de répondre directement aux notifications Mail, FaceTime, Messages, mais maintenant, toutes les notifications iOS que vous recevrez sur votre iPhone ou iPad seront aussi sur votre Mac.

L'application Calendrier perd son *habit de cuir* et revient avec une interface classique.

De nouvelles applications font leur apparition, comme **Maps** qui permet de préparer son itinéraire et de les

envoyer sur son iPhone.
iBooks, l'application de lecture, fait son entrée et permet comme sur iOS de tourner les pages et de faire des annotations.

Vous retrouverez tout ça dans la prochaine version d'OS X disponible à partir **d'aujourd'hui** pour les développeurs et **pour tous à l'Automne**.



Page sur OS X Mavericks : Lien 03

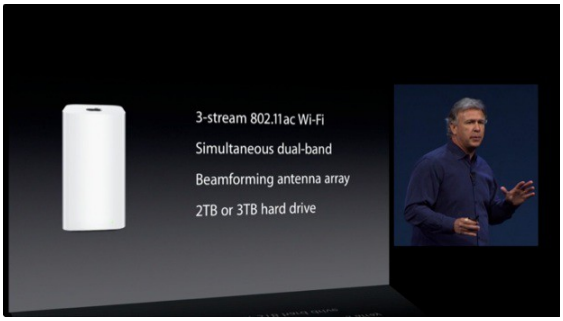
3. Nouveaux MacBook Air

De nouveaux MacBook Air, dotés des nouveaux processeurs Haswell qui permettent d'avoir une autonomie de 9 h pour les versions 11" et de 12 h pour les versions 13", améliorations graphiques de 40 % et Wi-Fi 802.11ac qui est plus rapide.

Nouveau MacBook Air sur l'Apple Store : Lien 04

4. Nouvelle Time Capsule

Une petite mise à jour de Time Capsule avec un nouveau look, Wi-Fi 802.11ac, en version 2 To ou 3 To de disque dur.



Nouvelle Time Capsule : Lien 05

5. Nouveau Mac Pro

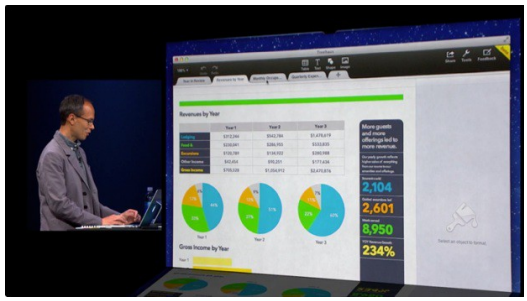
Après de longues années d'attente, Apple met enfin le Mac Pro au gout du jour et nous donne un aperçu de son futur ordinateur de bureau. Un design complètement revu, sombre, en forme de cylindre.

Dedans, une configuration de monstre, avec les nouvelles générations de processeurs Xeon (12 cores), gestion de la mémoire 2 fois plus rapide et 2,5 fois plus rapide pour les SSD.
6 ports Thunderbolt 2, 4 ports USB 3, 2 cartes graphiques (4096 streams processeurs) 2,5 plus rapides que les versions actuelles et peut prendre en compte trois écrans 4K à la fois.

Mac Pro en détail : Lien 06

6. iWork

Une mise à jour d'abord pour iOS de la suite bureautique iWork et une version Cloud (comme Google Docs) et tout ça directement dans son navigateur web et donc aussi sur les autres systèmes d'exploitation, ils ont même fait une démo sous Windows 8…



Tester la bêta d'iWork sur iCloud : Lien 07

7. iOS 7

La présentation commence par une introduction de Tim Cook qui rappelle les chiffres de ventes : 600 millions d'appareils iOS en circulation.

Ensuite Tim Cook présente la répartition des versions des OS Mobiles au sein de la base installée.
D'abord pour iOS : 93 % d'appareils sous iOS 6, 6 % sous iOS 5 et 1 % sous des versions plus anciennes. Puis il fait la comparaison avec la base installée Android : 37 %

d'appareils sous Gingerbread, 33 % sous Jelly Bean, 26 % sous Ice Cream Sandwich, 3 % sous Froyo et 1 % sous d'autres versions. Il en profite pour pointer du doigt la fragmentation côté Android et le désagrément que cela représente pour les développeurs. Il insiste sur le fait que plus d'un tiers des utilisateurs d'Android utilisent une version de l'OS qui date de 2010 et ne profitent pas des dernières innovations en raison de ces 3 années de retard.

En seconde partie vient la présentation de la version 7 d'iOS : le plus important changement d'iOS depuis la création de l'iPhone d'après Apple. Le design de l'interface utilisateur a été entièrement revu. Craig Federighi effectue les démonstrations des nouveautés d'iOS 7.

Il commence par montrer un effet curieux mais amusant où les icônes sur le bureau suivent les mouvements du téléphone par-dessus la photo de fond d'écran. Plutôt joli, bien que pas très utile. Il montre le nouveau design des applications de base : Météo, Contacts, Calendriers, Game Center, Newsstand, Calculatrice, Messages, Mail… toutes ont été entièrement redesignées. Les groupes d'applications peuvent maintenant contenir plusieurs pages d'applications.

Le nouveau design parait plus coloré. Certains éléments rappellent le style de l'interface Modern UI de Microsoft. Peut-être une tentative d'Apple d'attirer des clients de Microsoft ?
Le nouveau centre de notification amélioré est accessible depuis l'accueil sans déverrouiller le téléphone.

Une dizaine de nouvelles fonctionnalités ou applications majeures sont présentées parmi lesquelles : un nouveau centre de contrôle incluant la gestion de la musique en cours, réglage de la luminosité, gestion de la musique en cours de lecture, raccourcis vers des applications, mode avion, verrouillage de l'orientation de l'écran, Wi-Fi et Bluetooth, accès à AirDrop et AirPlay, allumage du flash pour faire lampe torche…

Le multitâche est maintenant disponible pour toutes les applications sans restriction. Ceci est probablement le changement le plus important pour les développeurs. Auparavant, Apple avait volontairement limité le multitâche pour préserver l'autonomie de la batterie. Les restrictions sont levées pour toutes les applications sans perte d'autonomie.

Safari mobile bénéficie lui aussi d'un redesign et d'une nouvelle interface plein écran focalisée sur le contenu. L'UI gaspille moins de place à l'écran. Les nouveautés du Safari de Mavericks comme la lecture d'articles de différents sites rassemblés en un seul flux sur une seule page et la fonctionnalité « liens postés par mes contacts sur les réseaux sociaux » qui compile tous les articles postés par les amis sont également disponibles dans Safari mobile. Les boutons « page précédente » et « page suivante » disparaissent. Voilà sans doute un des changements les plus importants de l'interface d'iOS 7, un maximum de boutons disparaissent au profit de l'utilisation de gestes tactiles, slides, pinch et autres. Apparemment, chez Apple, on préfère caresser les écrans qu'appuyer dessus. Pourquoi pas. Autre modification de Safari mobile, au lieu d'être limité aux 8 onglets habituels, on dispose d'une nouvelle interface animée plutôt élégante pour naviguer dans une infinité de pages ouvertes simultanément. L'intégration du cloud et la communication entre nos différents appareils est poussée à l'extrême dans cette version puisque l'on peut même voir les pages ouvertes sur nos autres appareils ou ordinateurs Apple en temps réel.

AirDrop fait son apparition dans iOS, il permet d'envoyer n'importe quoi à ses contacts alentours en Wi-Fi, peer-to-peer et crypté à condition que ceux-ci disposent également d'un appareil sous iOS 7.

L'application de photographie bénéficie également d'un nouveau design et de l'ajout de filtres pour les photos ainsi que d'un nouvel affichage plus pratique qui évite d'avoir une collection de milliers de photos désorganisées mais permet de les trier simplement par lieu, date, album, etc. Des albums de photos partagés dans le cloud peuvent également être créés.

Siri a été amélioré et intégré dans le tableau de bord des voitures de nombreux constructeurs automobiles sous la forme d'une nouvelle application appelée « iOS in the car ». On peut choisir différentes voix masculines et féminines en Anglais, Français et Allemand. Siri peut contrôler les paramètres du téléphone, ce qui permet par exemple de lui demander de régler la luminosité de l'écran.

Les mises à jour des applications sont maintenant faites automatiquement par l'application App Store. Plus besoin de le faire manuellement après réception d'une notification. Une bonne nouvelle pour les développeurs qui auront moins à se préoccuper du fait que les utilisateurs utilisent des versions obsolètes de leurs applications.

L'application musique n'échappe pas au nouveau design et se voit accompagnée d'une toute nouvelle application appelée iRadio permettant d'écouter des radios thématiques gérées par Apple ou de créer ses propres stations de radio virtuelles à partager avec ses contacts. iRadio va être intégré également à iTunes sur tous les ordinateurs Apple et sur Windows ainsi que sur les Apple TV. Cette nouvelle application sera disponible uniquement aux USA dans un premier temps.

Pour une meilleure protection contre le vol, Find my iPhone rend maintenant impossible de réinitialiser puis d'activer à nouveau un iPhone sans disposer de l'Apple ID et du mot de passe de son ancien propriétaire. De cette manière, un iPhone volé est inutilisable.

Les autres nouveautés qui ne sont pas présentées en détail pendant cette keynote sont rapidement évoquées sur la diapo suivante.

1500 nouvelles API pour les développeurs également évoquées rapidement sur la diapo suivante :

La bêta développeur d'iOS 7 est disponible immédiatement pour iPhone et un peu plus tard pour iPad. Le public, lui, va devoir patienter encore un peu avant d'en profiter.

Commentez la news d'Aurélien Gaymay en ligne : Lien 08

De XNA à MonoGame

Cet article est une republication du magazine Game Developer de mai 2013 expliquant comment vous pouvez passer de XNA à MonoGame.

1. Introduction

Pendant un long moment, l'idée de créer un « vrai » jeu en utilisant des langages managés comme C# était considérée comme de la folie. Mais les choses ont changé et les langages managés prouvent maintenant qu'ils sont tout à fait viables pour créer des jeux, grâce en grande partie à XNA et Unity 3D qui utilisent C# en tant que langage principal ou langage de script. Beaucoup de nouveaux développeurs de jeux ont appris à utiliser XNA afin d'avoir leurs jeux sur Xbox 360 et sur PC sous Windows. Bien que Microsoft ait récemment annoncé que XNA n'évoluera plus (Lien 09) et ne sera pas supporté dans l'interface Modern UI de Microsoft, ces développeurs n'ont pas à tout recommencer à partir de zéro. C'est là qu'arrive MonoGame, une implémentation Open Source de l'API XNA 4.

2. Qu'est-ce que MonoGame ?

Le but de MonoGame était à l'origine de permettre aux développeurs XNA de publier leurs jeux sur iPhone, mais le projet a énormément grandi depuis ses humbles débuts. Il supporte maintenant un certain nombre de plateformes, incluant OS X, Linux, Windows 8, Windows Phone 8, Android et iOS. Pour la plupart, l'utilisation est libre (bien que vous deviez payer la licence pour les frameworks Xamarin iOS et Android) et modifiable, car tout le code est couvert par la licence MS-PL (Microsoft Permissive License).
Maintenant, le but premier de MonoGame est de produire une API compatible XNA et multiplateforme, qui peut être étendue à de nouvelles plateformes au fur et à mesure de leur création. Par exemple, comme MonoGame supporte déjà Android, nous avons été capables d'ajouter le support de la Ouya en quelques jours.
Une fois l'API XNA stable, nous envisageons d'étendre l'API avec de nouvelles fonctionnalités que les gens auraient aimé voir dans XNA et l'étendre pour rendre certaines choses plus simples. Pour la plupart des membres de l'équipe qui travaillent sur MonoGame depuis déjà quelques années maintenant, c'est un projet à long terme.

3. Démarrer avec MonoGame

Premièrement, allez sur la page de téléchargement (Lien 10) et récupérez la dernière version stable. Actuellement, MonoGame supporte Visual Studio et MonoDevelop/Xamarin Studio.
Si vous voulez utiliser Visual Studio : l'installateur Windows va installer les modèles de projet pour toutes les éditions de Visual Studio 2010 et 2012. Des conditions particulières peuvent exister pour votre plateforme cible :

- pour les Windows bureau, vous pouvez utiliser VS 2010 Express ou une édition supérieure ou VS 2012 Express pour bureau ou une édition supérieure sur Windows 7 ou 8 ;
- pour Windows Store, vous avez besoin de VS 2012 Express pour Windows 8 ou une édition supérieure sur Windows 8 ;
- pour Windows Phone 8, vous avez besoin de Windows 8 64-bits et du SDK Windows Phone 8. Celui-ci installera VS 2012 Express pour Windows Phone et pourra aussi fonctionner avec VS 2012 Professionnel ou une édition supérieure. Pour utiliser l'émulateur Windows Phone 8, votre PC doit correspondre à des exigences matérielles spécifiques. Visitez ce site pour plus de détails : Lien 11 ;
- pour Android et/ou iOS, vous avez besoin de VS 2010 ou 2012 Professionnel ou une édition supérieure ainsi que de Xamarin Business ou une édition supérieure sur Windows 7 ou 8.

Pour compiler du contenu au format XNB, vous avez besoin de XNA Game Studio 4.0 ou du SDK Windows Phone 7.11 sur votre machine. C'est une solution temporaire tant que notre remplacement du pipeline de contenu n'est pas complet. Pour installer ces SDK sur Windows 8, vous devez d'abord avoir le client Games for Windows Live.
Une liste complète des conditions et des liens de téléchargement peut être trouvée ici : Lien 12.
Si vous souhaitez utiliser MonoDevelop/Xamarin Studio : MonoDevelop est un EDI gratuit et open source disponible pour Windows, Linux et OS X (Lien 13). MonoGame est disponible sous la forme d'un paquet, qui inclut les routines pour chaque plateforme. Vous pouvez le télécharger sur CodePlex sous la forme d'un fichier .mpack et l'installer à l'aide du « *AddIn Manager* » (gestionnaire de modules) dans MonoDevelop. Récemment, Xamarin a publié Xamarin Studio, une version améliorée de MonoDevelop pour les gens voulant utiliser cet EDI. Il y a aussi un fichier .mpack disponible pour cette version.
Une fois que vous avez installé le paquet requis dans votre EDI, vous êtes prêt à créer un nouveau projet MonoGame. Il y a plusieurs choix possibles, mais pour démarrer, prenez celui conçu pour votre plateforme native : pour Windows, créez une nouvelle application « *MonoGame for Window GL* » (MonoGame pour Window GL) ; pour OS X, créez une application « *MonoGame for MacOS* » (MonoGame pour MacOS) et ainsi de suite. Une fois que vous avez créé un nouveau projet, vous allez voir la classe habituelle « Game » qui effectue le rendu de l'écran bleu bleuet.

Skulls of the Shogun développé par 17-Bit avec MonoGame pour produire une version Windows 8 de leur jeu.

Avant que vous ne commenciez à coder, il y a quelques éléments à savoir. Pour Windows, tous les modèles de projet vont fonctionner directement, mais si vous voulez travailler sur un projet Android, vous aurez besoin de Xamarin.Android provenant de Xamarin. MonoDevelop/Xamarin Studio inclut les modèles de projet pour OS X et iOS, mais ceux-ci ne fonctionneront que sur un OS X à cause des conditions de développement requises. Pour iOS, vous devrez installer Xamarin.iOS et pour OS X vous devrez récupérer MonoMac. De plus, à cause de quelques problèmes avec le module AddIn pour MonoDevelop et Xamarin Studio, vous allez devoir télécharger le code source de MonoGame et ajouter les références dans les projets pour OS X, iOS et Android. (L'équipe de développement travaille sur la correction de ce point.)

4. Récupérer le code

Si vous voulez récupérer le code de MonoGame, vous devez installer *git*. Il existe de jolies interfaces utilisateur pour git, mais la plupart des membres de l'équipe utilisent la ligne de commande. Donc voici la liste des commandes que vous devez utiliser pour récupérer le code :

```
git clone git://github.com/mono/MonoGame.git MonoGame
cd MonoGame
git submodule update -init ThirdParty
```

Ces commandes récupéreront une copie en lecture seule de MonoGame et des bibliothèques tierces dont vous avez besoin. Si vous souhaitez faire des modifications et contribuer au projet, vous devez créer un fork du projet. Vous pouvez trouver les détails ici : Lien 14.

5. Contenu et ressources

MonoGame, tout comme XNA, peut utiliser les fichiers de contenu compilé .xnb. Ces fichiers .xnb sont actuellement créés avec le pipeline de contenu XNA. Au moment de l'écriture de cet article, l'équipe de MonoGame travaille sur une implémentation multiplateforme qui fonctionnera sur Windows, OS X et Linux, mais pour le moment vous aurez besoin d'utiliser le pipeline de contenu de XNA. Il est aussi possible de charger des ressources natives grâce au ContentManager, donc si vous souhaitez charger directement un fichier .png, vous pouvez utiliser le code que vous utilisez habituellement :

```
texture = Content.Load<Texture2D>("character");
```

MonoGame essaiera de charger un fichier .xnb en premier, mais si le fichier n'existe pas, il tentera les types natifs connus pour ce type d'objet. Cette fonctionnalité a été ajoutée à l'origine pour aider les personnes développant pour iOS et OS X et n'ayant pas accès à une machine Windows pour préparer leurs contenus. (Voir la **Figure 1** pour une table des types de fichiers disponibles pour chaque classe.)

Classe	Windows	Linux	OS X	iOS	Android	Windows 8
Texture2D	.xnb .png .jpg .tiff	.xnb .png .jpg .tiff	.xnb .png .jpg .tiff	.xnb* .png .jpg .tiff	.xnb .png .jpg .tiff	.xnb .png .jpg
SoundEffect	.xnb .wav	.xnb .wav	.xnb	.xnb	.wav .mp3 .ogg	.xnb
Song	.xnb .wav	.xnb .wav	.xnb	.xnb	.mp3	.xnb
Model	.xnb	.xnb	.xnb	.xnb	.xnb	.xnb
Effect	.xnb**	.xnb**	.xnb**	.xnb**	.xnb**	.xnb**
SpriteFont	.xnb	.xnb	.xnb	.xnb**	.xnb	.xnb

Figure 1 : formats disponibles pour chaque classe. * - fichier PVRTC compressé seulement, ** - doit être généré par le « Content Pipeline » de MonoGame.

Comme vous pouvez le voir dans la **Figure 1**, sur Android vous ne pouvez pas utiliser les fichiers .xnb pour les SoundEffect et Song. Vous *devez* utiliser les types natifs pour cette plateforme. Cela est dû aux limitations des classes SoundPool et MediaPlayer sur Android. (Certaines de ces restrictions seront retirées au fil du temps et de la maturation du framework.) De plus, certaines plateformes ne supportent que les fichiers .xnb pour certains types. Cela est dû au fait qu'il est plus efficace d'utiliser les fichiers de contenu compilé et optimisé pour ces types. Par exemple, il est plus performant de prétraiter un Model au moment de la compilation que de charger un fichier .fbx pendant l'exécution et de le décoder sur un périphérique mobile.
Heureusement, le framework MonoGame est fourni avec des outils que nous utilisons pour étendre le pipeline de contenu de XNA afin de produire ce dont nous avons besoin. Toutefois, comme il repose sur le framework XNA, cela ne fonctionnera que sous Windows. Vous pouvez créer un « MonoGame Content Project » (Projet de contenu MonoGame) dans Visual Studio 2010, qui créera un projet au contenu XNA classique *et* un projet XNA qui va, à son tour, construire le contenu du projet. Ces deux projets ont toutes les références nécessaires et les importations de cibles MSBuild pour permettre d'utiliser les processeurs de MonoGame.
Maintenant, lorsque vous ajoutez une nouvelle ressource au contenu du projet et que vous sélectionnez le type de processeur que vous souhaitez utiliser, vous allez voir des processeurs liés à MonoGame : « MonoGame - Effect », « MonoGame - Texture » et ainsi de suite. Ces processeurs effectueront des tâches spécifiques sur les ressources afin

de les optimiser pour la plateforme cible (voir **Figure 2**).

Type	Windows	Linux	OS X	iOS	Android	Windows 8
Texture2D	DXT	DXT	DXT	PVRTC	DXT	DXT
SpriteFont	DXT	DXT	DXT	PVRTC	DXT	DXT
Song	MP3	MP3	MP3	MP3	N/A	MP3
SoundEffect	PCM	PCM	PCM	PCM	N/A	PCM

Figure 2 : exemple de formats internes .xnb optimisés.

Le nouveau projet « Builder » possède plusieurs configurations de construction pour chaque plateforme, donc si vous ciblez l'iPhone, vous pouvez choisir la configuration iOS. Si vous ciblez le Windows Store, choisissez la configuration Windows 8. Les fichiers produits seront placés dans bin\<Configuration>\Release, de sorte que si vous construisez les fichiers pour plusieurs plateformes, vous aurez un répertoire pour chaque plateforme cible. La dernière étape est d'ajouter ces fichiers au répertoire Content de votre projet. Vous pouvez le faire facilement en utilisant la fonctionnalité « Add as Link » (Ajouter comme lien) dans Visual Studio et MonoDevelop. De cette façon, vous pouvez lier un fichier sans avoir à le copier dans le répertoire du projet, ce qui est pratique lorsque vous modifiez le contenu et que vous le compilez avec le « Builder » de contenu, le fichier sera automatiquement récupéré la prochaine fois que vous compilez votre application. Notez que selon la plateforme cible vous devez définir la BuildAction (action de construction) correctement pour vos fichiers ressources afin de les inclure dans le paquet final de l'application.

6. Utiliser des effets personnalisés

Si vous souhaitez utiliser des fichiers Effect d'un projet XNA précédent ou d'un échantillon XNA, vous devez les traiter avec le processeur d'effets de MonoGame afin de les compiler pour la plateforme cible. Certaines plateformes utilisent OpenGL au lieu de DirectX comme bibliothèque graphique, donc le fichier Effect de XNA devra être converti dans le langage de shaders d'OpenGL pour fonctionner.
Plutôt que de demander aux développeurs de réécrire leurs shaders en GLSL, MonoGame installe quelques outils pour convertir automatiquement le HLSL du fichier Effect vers le langage de shaders approprié pour la plateforme cible. Pour les plateformes utilisant DirectX, MonoGame utilise la suite d'outils de DirectX 11 pour compiler vos effets en shaders optimisés. Pour les plateformes OpenGL, le fichier Effect est traité par un outil appelé MojoShader, qui effectue une conversion bas niveau du HLSL vers GLSL permettant à votre effet de fonctionner pour cette plateforme.
Bien sûr, le processus de conversion peut occasionnellement introduire des erreurs ou des fonctionnalités non supportées par la plateforme cible. Par exemple, avec OpenGL Shader Model 3.0, vous ne pouvez pas accéder à la texture dans le vertex shader, donc si votre effet utilise cette fonctionnalité, il ne fonctionnera probablement pas. Prenons un échantillon d'un effet provenant des exemples de XNA disponibles sur le site Xbox Creator Club : l'effet d'extraction de bloom est un bon exemple. Bien que ce ne soit qu'un pixel shader, cela vous donnera une bonne idée du genre de choses dont nous devons nous soucier.

Listing 1 : Extraction de bloom

```
// Le Pixel shader extrait les zones claires de l'image.
// C'est la première étape pour appliquer l'effet de bloom.
sampler TextureSampler : register(s0);
float BloomThreshold;
float4 PixelShaderFunction(float2 texCoord : TEXCOORD0) : COLOR0
{
    // Récupère la couleur originale de l'image.
    float4 c = tex2D(TextureSampler, texCoord);
    // Ajustement pour garder seulement les valeurs plus claires que le seuil fixé.
    return saturate((c - BloomThreshold) / (1 - BloomThreshold));
}

technique BloomExtract
{
    pass Pass1
    {
        PixelShader = compile ps_2_0 PixelShaderFunction();
    }
}
```

Si vous regardez le code de l'effet du **listing 1**, une des premières choses à noter est que cet effet utilise les pixel shaders 2.0 (ps_2_0). Cela est correct pour DirectX 9, mais pour OpenGL et DirectX 11 cela doit être modifié. Avant de changer le code, il est nécessaire de noter que vous pouvez utiliser des définitions conditionnelles dans les fichiers d'effets pour changer le comportement selon le modèle de shaders ciblé. Si vous ciblez le modèle de shaders 4, vous pouvez utiliser :

```
#if SM4
// code
#endif
```

pour gérer ce cas spécial. Ces définitions sont toujours valides lorsque vous utilisez le « Content Pipeline » de MonoGame, donc pour faire en sorte que ce shader supporte toutes les plateformes que nous voulons, nous devons mettre à jour la partie technique pour gérer toutes les plateformes cibles. Dans ce cas, nous devons supporter le modèle de shaders 4 pour DirectX 11 et le modèle de shaders 3 pour OpenGL/GLES. Sachant cela, la passe devient :

```
pass Pass1
{
    #if SM4
        PixelShader = compile ps_4_0_level_9_1 PixelShaderFunction();
    #elif SM3
        PixelShader = compile ps_3_0 PixelShaderFunction();
    #else
        PixelShader = compile ps_2_0 PixelShaderFunction();
    #endif
}
```

Pour chaque type de modèle de shaders, nous définissons

un pixel shader différent. Vous pouvez aussi faire cela avec les vertex shaders, donc une déclaration vs_2_0 deviendra vs_3_0 pour les modèles de shaders 3 et vs_4_0_level_9_1 pour les modèles de shaders 4.
Ensuite, regardons les paramètres passés à la fonction PixelShaderFunction. Dans cet exemple, la fonction ne prend qu'un paramètre, mais afin que cet effet fonctionne correctement avec l'effet SpriteBatch de MonoGame, nous devons ajouter des paramètres supplémentaires. (C'est une limitation de MonoGame qui sera probablement résolue dans le futur.) La bonne nouvelle est que l'ajout de ces paramètres supplémentaires ne casse pas la compatibilité avec XNA, donc, dans ce cas, nous pouvons simplement ajouter les paramètres manquants comme suit :

```
void PixelShaderFunction( float3 position:
POSITION0, float4 color : COLOR0, float2 texCoord
: TEXCOORD0)
```

Dans XNA 3.1, les fonctions PixelShader et VertexShader étaient appelées *PixelShader* et *VertexShader*. Cela n'est plus valide en XNA 4.0 ou dans MonoGame, ce qui explique pourquoi, dans le **listing 1**, nous renommons les fonctions : *VertexShaderFunction* et *PixelShaderFunction*. Bien sûr, vous pouvez les nommer comme vous le souhaitez.

Supergiant Games a utilisé un fork du code de MonoGame pour porter Bastion sur iPad.

Si vous appliquez ces techniques à tous les effets inclus dans l'échantillon de Bloom, le portage du projet pour Windows 8, Android et iOS devient juste une question d'utilisation du constructeur de contenu pour produire les fichiers .xnb pour chaque plateforme cible et ensuite lier ces fichiers à votre projet.

7. MonoGame pour mobiles

Donc, vous avez un bon jeu sur Xbox 360 ou Windows que vous souhaitez porter sur mobile. Quelles sont les choses dont vous devez vous soucier ? Certains points sont évidents : la taille de l'écran, les commandes et la performance. Voici comment gérer ces détails dans MonoGame.

7.1. Résolution d'écran

Sur certaines plateformes, comme Windows Phone 7 et iOS, vous pouvez presque garantir la taille de l'écran des périphériques sur lesquels vous travaillez, mais avec d'autres périphériques et plateformes, ce n'est pas si simple. Windows Phone 7/8 supportent le redimensionnement matériel mais avec Android, Ouya, Windows 8 et les écrans Retina sur iPad et Mac, les développeurs MonoGame doivent prendre en considération de nombreuses résolutions d'écran. Quelques fois, vous allez vous retrouver naviguant dans un champ miné de résolutions d'écran allant de 320x200 jusqu'à la full HD.
Une technique qui a plutôt bien été utilisée dans le passé est le redimensionnement du SpriteBatch. Avec cette méthode, vous pouvez créer votre jeu pour qu'il s'exécute avec une résolution précise, puis, pendant l'exécution, trouver une matrice de redimensionnement qui redimensionnera votre jeu pour correspondre à la taille de l'écran du périphérique sur lequel il s'exécute. Cette matrice est passée au SpriteBatch, qui redimensionne vos graphismes. Dans le code suivant, nous pouvons voir comment calculer la matrice. Notre résolution virtuelle est ici de 800x600.

```
var virtualWidth = 800;
var virtualHeight = 600;
var scale = Matrix.CreateScale(

(float)GraphicsDevice.Viewport.Width /
virtualWidth,

(float)GraphicsDevice.Viewport.Height /
virtualHeight,
                    1f);
```

Une fois la matrice calculée, nous pouvons appeler le SpriteBatch avec le paramètre de la matrice comme suit :

```
spriteBatch.Begin(SpriteSortMode.Immediate, null,
null, null, null, null, scale);
// affichage
spriteBatch.End();
```

Cela appliquera la matrice de redimensionnement pour les éléments affichés par le batch. Il y a d'autres considérations telles que le ratio et les formats que vous devez prendre en compte, mais pour ceux-ci, il y a plein de ressources, originellement écrites pour XNA et pouvant être réutilisées avec MonoGame et appliquées aux nouvelles plateformes. (Pour plus d'informations sur ce sujet, lisez cette discussion : Lien 15.)
À partir d'un certain point, le redimensionnement ne résoudra pas vos problèmes. En rétrécissant de trop vos graphismes, ils deviendront pixelisés et l'agrandissement les rendra flous. C'est là que vous devez repenser vos ressources. Une technique que vous pouvez utiliser avec iOS est d'avoir deux ensembles de ressources : un pour les écrans normaux et l'autre pour les écrans Retina. Vous pouvez le faire en ajoutant un suffixe @2x aux noms des plus grandes ressources et les inclure dans votre paquet de jeu. Le pipeline de contenu de MonoGame a été programmé pour se servir de cette astuce sur iOS, donc si vous avez un fichier de ressources .xnb brut qui possède une version @2x et que vous l'exécutez sur un périphérique avec écran Retina, la ressource ayant une plus grande résolution sera chargée. (Cela augmente malheureusement la taille de votre paquet.)
L'autre option est d'avoir deux versions du jeu, une pour les périphériques normaux et une version HD, chacune utilisant un ensemble de ressources différent. Avec cette méthode, les gens auront le choix de télécharger une version HD du jeu ou la version normale.

7.2. Gestion des commandes

Changer de plateforme signifie que vous devez supporter de nombreux périphériques d'entrée. Sur les plateformes de bureau (Windows, Linux et OS X), MonoGame utilise la bibliothèque Simple Direct Media Library (SDL) pour interfacer les différents joysticks et manettes de jeu pouvant être disponibles. Les entrées de ces périphériques sont redirigées à travers la classe GamePad. La SDL semble avoir des problèmes pour lire la description USB de certains contrôleurs Xbox 360, donc lorsque cela arrivera, il aura du mal à appliquer la configuration adéquate.
Les entrées de la souris et des touchpads sont un point sur lequel MonoGame a dévié par rapport à XNA. Sur Windows Phone 7 et 8, si vous utilisez l'implémentation de XNA fournie par Microsoft, tous vos événements tactiles seront dirigés vers la souris. Cela peut rendre très facile le portage des jeux Windows et Xbox 360 vers ces plateformes. Par contre, avec l'introduction de Windows 8, des périphériques peuvent maintenant avoir une souris et un écran tactile et il est possible que certains jeux les utilisent séparément. Sachant cela, si vous avez un jeu Windows que vous souhaitez porter sur l'application Windows Store, vous devez ajouter le support des écrans tactiles, car votre code gérant la souris ne sera pas utilisé.

7.3. Pousser les performances

Les versions iOS et Android de MonoGame s'exécutent au-dessus de la plateforme Mono. Mono possède un bon compilateur et ces deux plateformes embarquent aussi un éditeur de liens, qui est utilisé pour réduire la taille du paquet de votre application en retirant le code non utilisé. Il possède aussi un très bon garbage collector, mais même s'il y a eu des améliorations majeures sur celui-ci ces dernières années, vous devez garder à l'esprit que vous exécutez votre code sur des mobiles et que ces derniers varient radicalement en termes de performances (surtout avec les périphériques Android). Assurez-vous de regarder vos méthodes Update. Vous pouvez donner du travail au garbage collector en créant de nombreux objets temporaires qui ne seront que jetés. Vous devez aussi réfléchir *à* l'*obligation* de faire des calculs d'une certaine façon : après tout, le code le plus rapide et celui qui n'est pas exécuté. Considérons cette définition de classe :

```
public class Player
{
  public Vector2 Position;
  public Vector2 Scale;
  public void Update(GameTime gameTime) { … }
  public void Draw(GameTime gameTime) { … }
}
```

Dans ce morceau de code, nous définissons un joueur ayant une position et une échelle (scale). Maintenant, supposons que nous utilisions la propriété Matrix du SpriteBatch pour afficher ce joueur en utilisant les données de position et d'échelle, mais aussi pour la détection de collisions. Une méthode simple pour créer la matrice est de le faire dans la méthode Update, comme suit :

```
var matrix = Matrix.CreateTransform(Position) *
Matrix.CreateScale(Scale);
```

Sur les machines de bureau, cela fonctionnera sûrement correctement, mais sur des périphériques mobiles moins puissants cela pourra être un problème. Même si la classe Matrix est une structure et est peu couteuse à créer, il en reste que nous faisons ce calcul à chaque appel à la fonction Update. Si votre jeu s'exécute à 30 images par seconde, rien que ce code provoque de nombreux calculs de matrices que vous pouvez éviter. À la place, vous pouvez déclarer un champ pour la matrice et le mettre à jour dans la méthode Update simplement. Ou, encore mieux, vous pouvez simplement mettre à jour la matrice lorsque les propriétés Position ou Scale ont été modifiées (voir **listing 2**).

Listing 2 : optimisation des appels à Update pour les périphériques mobiles

```
public class Player
{
  Vector2 position;
  Vector2 scale;
  Matrix matrix;

  public Vector2 Position {
    get { return position; }
    set {
      if (position != value) {
        position = value;
        UpdateMatrix();
      }
    }
  }

public Vector2 Scale {
  get { return scale; }
  set {
    if (scale != value) {
      scale = value;
      UpdateMatrix();
    }
  }
}

  public void UpdateMatrix() {
    matrix = Matrix.CreateTransform(Position) *
Matrix.CreateScale(Scale);
  }
  public void Update(GameTime gameTime) { … }
  public void Draw(GameTime gameTime) { … }
}
```

Je sais que c'est un simple exemple, mais si vous pensez aux petites améliorations que vous pouvez faire, alors il a fait son travail. Parmi vous, ceux qui connaissent les mathématiques sauront que l'on peut certainement enlever les propriétés de position et d'échelle de la classe et simplement garder la matrice.
Un autre exemple est la gestion des projectiles tels que les tirs d'une arme ou d'un vaisseau. Une façon de gérer cela est d'avoir une List<Bullet> où vous ajoutez les nouveaux projectiles au fur et à mesure et vous les retirez lorsqu'ils sortent de l'écran ou de la zone de jeu. Lorsque vous ajoutez des éléments à une liste, cela entraine l'expansion de sa capacité, vous créez de nouvelles instances de Bullet à chaque fois. Un système plus efficace serait d'avoir un cache d'instances de Bullet que vous pourriez utiliser pour peupler une liste de projectiles « actifs » en cours d'utilisation. Lorsque le projectile sort du jeu, nous

pouvons juste le replacer dans le cache.
Bien sûr, vous n'avez certainement pas besoin d'une infinité de projectiles. La plupart des jeux limitent le nombre de projectiles ou missiles que l'on peut tirer. Dans ce cas, vous pouvez garder un petit cache afin de ne pas utiliser trop de mémoire et vous pouvez définir une capacité pour la liste des projectiles actifs à l'avance afin que sa taille n'augmente pas en cours de jeu.
Si vous devez créer énormément de variables temporaires pendant le chargement du niveau ou lors de n'importe quelle autre sorte de traitement, vous pouvez essayer d'appeler GC.Collect(0) aussi souvent que vous le pouvez. Cela assurera que le garbage collector collecte les éléments temporaires au lieu d'attendre la prochaine collection automatique. De plus, vous allez souhaiter éviter qu'une collection se produise pendant le jeu. Si le garbage collector doit faire une collection majeure pendant votre boucle de mise à jour, vous allez sans aucun doute voir une pause dans votre jeu. Cela peut être pour seulement quelques secondes, mais cela se remarquera. Notez que contrairement à GC.Collect(0), GC.Collect() effectue une collection complète à travers les ressources allouées sur le tas. Cela peut prendre jusqu'à 100-200 millisecondes (plusieurs trames) donc vous souhaiterez éviter les appels à GC.Collect durant le jeu, sinon, cela introduira une pause ou un lag.
Dernier point, n'utilisez pas trop d'instances de SpriteBatch. Généralement, dans les jeux XNA vous allez vouloir minimiser le nombre d'instances de SpriteBatch. C'est aussi vrai pour MonoGame. L'implémentation de MonoGame possède un cache interne de SpriteBatchItems que nous recyclons pour garder le nombre d'objets alloués au minimum. Par défaut, nous créons 1000 éléments dans notre cache, donc plus vous avez d'instances, moins vous avez de mémoire système et vous devez faire avec.

7.4. Éditeur de liens Android et iOS

Lorsque vous ciblez les plateformes Android et iOS en utilisant la suite d'outils Xamarin, vous devez être conscient de l'éditeur de liens qui est exécuté sur votre code pendant la compilation. Cet éditeur de liens réduit la taille du code de votre paquet en enlevant les parties inutilisées de votre code ou du framework suivant les paramètres de l'outil. Le résultat est que votre paquet final contiendra un framework mono optimisé et non le framework tout entier et les parties inutilisées de votre jeu seront enlevées, elles aussi.
Si vous chargez des objets dynamiquement à travers le pipeline de contenu, vous allez quelques fois vous trouver dans la situation où l'éditeur de liens ne connait pas le type des objets que vous utilisez et les retire du code. Cela mène habituellement à une exception MissingMethodException lorsque vous essayez de construire ou d'appeler une méthode d'un de ces types non liés. La bonne nouvelle est qu'il y a plusieurs méthodes pour informer l'éditeur de liens que vous souhaitez préserver certains objets tels qu'ils sont et qu'il ne les enlève pas. Une méthode consiste à utiliser PreserveAttribute, qui est fourni sur iOS et Android :

```
#if ANDROID

  [Android.Runtime.Preserve(AllMembers=true)]

#elif IOS


[MonoTouch.Foundation.Preserve(AllMembers=true)]

#endif

public class Example {

  public Example ()
  {
  }
}
```

Cela assure que pour les plateformes Android et iOS la classe ne sera pas retirée par l'éditeur de liens.
Il y a d'autres méthodes pour contrôler le comportement de l'éditeur de liens. Si vous souhaitez en apprendre plus, lisez les documentations d'Android et iOS disponibles sur le site de Xamarin :

- documentation pour iOS : Lien 16 ;
- documentation pour Android : Lien 17.

8. Aidez-nous !

Cet article n'a fait qu'effleurer la surface de MonoGame, mais heureusement cela vous motivera pour essayer et peut-être même le rendre meilleur ! Nous avons un certain nombre d'objectifs à l'horizon pour lesquels nous souhaitons être aidés, comme construire un pipeline de contenu multiplateforme, ajouter le support de nouvelles plateformes telles que Windows Phone 8 et Raspberry Pi, utiliser DirectX 11 et étendre XNA. Donc, si vous souhaitez nous aider, rejoignez monogame.net : Lien 18.

Retrouvez l'article de Dean Ellis traduit par Alexandre Laurent en ligne : Lien 19.

Oracle optimise ses solutions dédiées au développement des solutions embarquées avec Java ME SDK 3.3 et Java ME Embedded 3.3

La nouvelle tendance dans le réseau se retrouve dans l'internet des objets. De plus en plus de périphériques de notre quotidien devraient être interconnectés dans les années à venir. Le projet de véhicules intelligents **SIM** (Lien 20) donne déjà un aperçu de ce à quoi devra ressembler le monde des interconnexions de demain.

Pour tirer avantage de cette nouvelle tendance, en plus des solutions matérielles comme les microcontrôleurs (ARM, PIC, etc.) des solutions logicielles doivent être développées. Oracle, à la suite de **Microsoft** (Lien 21), publie ses solutions de développement d'applications pour périphériques embarqués.

Ce sont Oracle Java ME SDK 3.3 (dont le téléchargement inclut des plugins pleinement fonctionnels pour les environnements Eclipse et NetBeans), Oracle Java ME Embedded 3.3 for Raspberry Pi pour ARM, Oracle Java ME Embedded 3.3.1 for KEIL MCBSTM32F200 (basé sur les puces ARM Cortex M3) qui sont disponibles en téléchargement sur son site officiel.

Pour rappel, Oracle Java ME Embedded est un environnement d'exécution Java optimisé pour les microcontrôleurs et autres dispositifs limités en ressources. Il permet aux clients d'étendre la durée de vie, la flexibilité et la valeur des solutions intégrées en offrant des mises à niveau d'applications, sans compromettre l'intégrité et la sécurité du système. Le SDK Java ME fournit un environnement de développement complet pour Oracle Java ME Embedded et Oracle Java Wireless Client.

De plus, via son initiative « Oracle Java Platform Integrator », la société offre la possibilité à des tiers de personnaliser toutes leurs solutions de développement de périphériques embarqués. L'objectif est de permettre à ces derniers de développer des applications pour tout type de périphériques embarqués et d'atteindre ainsi plusieurs segments de marché.

Oracle a également publié une mise à jour pour Java SE Embedded 7 apportant des correctifs de sécurité et une mise à jour mineure pour Oracle Java Embedded Client 1.1.1.

Commentez la news de Christophe Ghokeng en ligne : Lien 22

Oracle abandonne sun.reflect.Reflection.getCallerClass

En tant que développeur Java, nous gardons un oeil avisé sur les méthodes dépréciées du JDK. Ne pas partir du principe que ces méthodes resteront présentes, et toujours annotées de @deprecated, est une grossière erreur. Et nous pouvons le constater aujourd'hui.

Oracle a déprécié (plutôt éliminé) une méthode secrète du package *sun.reflect* appelée Reflection.getCallerClass(int) (Lien 23), et cela est fait pour le jdk7u40. Pour toutes les versions de Java 7, vous pouvez l'utiliser à nouveau en ajoutant la ligne de commande -Djdk.reflect.allowGetCallerClass. Mais la méthode sera complètement supprimé pour Java 8, et les versions suivantes. Une fois supprimée, un appel à la méthode provoquera une *UnsupportedOperationException*.

L'équipe travaillant sur le JDK aimerait savoir comment cette méthode est utilisée dans les applications et si ce code peut être modifié sans aucune dépendance sur les API *sun.**.
Vous pouvez directement faire un retour en rejoignant la mailing list d'OpenJDK core-dev-libs : Lien 24.

Oracle a toujours prévenu les développeurs de ne pas utiliser les méthodes des packages sun. Vous pouvez en lire les raisons ici : Lien 25.

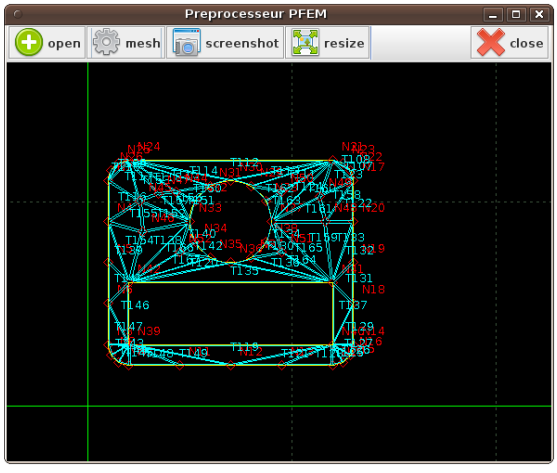
Commentez la news d'Olivier Pitton en ligne : Lien 26

PFEM2D - Définition de l'interface utilisateur et gestion des éléments graphiques en Java2D

Cet article présente la mise en place en Java d'un panneau graphique personnalisé hérité du composant Swing JPanel.

1. Introduction

Nombre de questions reviennent sur les forums afin de savoir comment dessiner un graphe en Java, quelle API utiliser… Ce tutoriel permet de poser les bases de conception d'un composant graphique de type JPanel personnalisé. Personnalisé au sens où tout l'affichage est à définir par le programmeur afin d'avoir le rendu désiré par l'utilisateur : couleurs, éléments à afficher, interaction utilisateur…
Le composant graphique sera tout d'abord créé par héritage de la classe Swing JPanel. Un gestionnaire d'éléments graphiques sera ensuite défini afin de gérer l'affichage des différents éléments. Les éléments graphiques seront alors définis dans différentes classes (modèle, point, ligne…). Un exemple d'application final permettra d'observer le comportement de notre composant graphique en situation d'utilisation.
Ce tutoriel s'intègre dans une série d'articles basés sur le développement d'un logiciel de modélisation d'éléments finis 2D nommé PFEM2D. Les différentes classes porteront donc ce préfixe dans la suite de l'article.
Dans le cadre de PFEM2D, l'interface visuelle nécessite l'affichage du modèle en 2D, avec un quadrillage vertical et horizontal afin de positionner les différents éléments. Une interaction avec la souris est également nécessaire afin de déplacer et zoomer l'affichage.
L'image suivante montre un rendu du composant graphique personnalisé, intégré dans l'application PFEM2D :



Dans la suite de cet article, le code source est présenté au plus proche de la description technique des diverses fonctionnalités du code final. Le code affiché peut donc n'être qu'un extrait partiel d'une classe particulière. Le code source complet de l'article est disponible en téléchargement au chapitre .

2. Définition du panneau graphique

2.1. Surcharge de JPanel : la classe PFEM2DGuiPanel

La méthode de base pour créer un composant graphique est de s'appuyer sur celui qui s'en rapproche le plus dans la bibliothèques de composants Swing et de surcharger sa méthode **paintComponents**.
Dans notre cas, il s'agit du composant **JPanel**. Le composant graphique créé portera le nom de .
La méthode **paintComponents** est surchargée afin de dessiner :

- le fond en noir ;
- les axes X et Y du repère 2D en trait plein vert ;
- le quadrillage en pointillés verts ;
- les éléments graphiques, avec leur affichage personnalisé.

L'interaction avec la souris est ajoutée au composant PFEM2DGuiPanel en ajoutant des listeners **MouseListener** et **MouseMotionListener**.
Afin d'améliorer l'expérience utilisateur, deux méthodes seront ajoutées :

- une méthode permettant d'effectuer une capture d'écran de l'affichage du composant **PFEM2DGuiPanel** (pratique pour créer les images de cet article) ;
- une méthode permettant de centrer l'affichage sur les éléments à afficher.

La trame de base de la classe **PFEM2DGuiPanel** est présentée ci-après :

Classe PFEM2DGuiPanel

```
package plegatfem2d;

import java.awt.BasicStroke;
import java.awt.Color;
import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.event.MouseEvent;
import java.awt.event.MouseListener;
import java.awt.event.MouseMotionListener;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;
import javax.imageio.ImageIO;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;
import javax.swing.JPanel;

/**
 * Classe définissant un JPanel personnalisé.
 * Ce JPanel permet l'affichage d'une grille 2D
(axes X et Y, quadrillage)
 * ainsi que d'objets graphiques 2D définis par
le programmeur.
```

```
 * L'affichage est interactif via la souris,
l'utilisateur peut ainsi déplacer la vue, ainsi
que zoomer
 * sur une zone particulière.
 *
 * @author Jean-Michel BORLOT
 */
public class PFEM2DGuiPanel extends JPanel
implements MouseListener, MouseMotionListener {

    /**
     * Constructeur.
     */
    public PFEM2DGuiPanel() {
        super();

        this.mode = VIEW;

        this.offsetX = 0;
        this.offsetY = 0;

        this.echelle = ECHELLE_BASE;

        this.addMouseListener(this);
        this.addMouseMotionListener(this);
    }

    private PFEM2DObjectManager pom;
// gestionnaire d'objets graphiques
    private int xold, yold;
// coordonnées du point local précédent
    private int xstart, ystart;
// coordonnées du point local initial
    private int mode;
// mode d'interaction : VIEW, DRAG ou SCALE
    private double offsetX, offsetY;
// décalage de la vue
    private double echelle;
// échelle de la vue
    private static int VIEW = 0;
    private static int DRAG = 1;
    private static int SCALE = 2;
    private static double ECHELLE_BASE = 100.;
// échelle de base
    private static double stepX=100, stepY=100;
// pas du quadrillage, sens X et sens Y

    /**
     * Renvoie le gestionnaire d'objets
graphiques
     * @return le gestionnaire d'objets
graphiques
     */
    public PFEM2DObjectManager getPom() {
        return pom;
    }

    /**
     * Définit le gestionnaire d'objets
graphiques
     * @param pom le gestionnaire d'objets
graphiques
     */
    public void setPom(PFEM2DObjectManager pom) {
        this.pom = pom;
    }
}
```

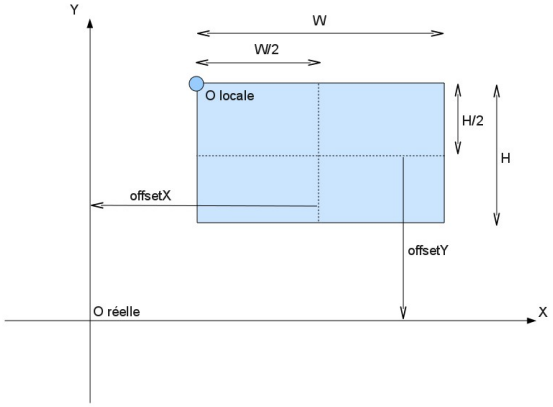
Dans ce code, les propriétés **ECHELLE_BASE**, **stepX** et **stepY** sont définies « en dur » et ne font pas l'objet de modifications ultérieures. Dans le cas où l'utilisateur doit avoir la possibilité de modifier ces valeurs, il conviendra d'intégrer les getters et setters nécessaires.

2.1.1. Conversion coordonnées réelles-coordonnées locales

La fonction primordiale de la classe **PFEM2DGuiPanel** est d'afficher à l'écran les objets que l'on souhaite dessiner. Pour cela, il est nécessaire de faire une conversion entre les coordonnées réelles des objets vers les coordonnées de la zone d'affichage sur l'écran (ou « locales ») et réciproquement. Quatre méthodes permettent de faire ces conversions :

- **getLocalCoordX**, pour la conversion d'une abscisse réelle vers l'abscisse locale ;
- **getLocalCoordY**, pour la conversion d'une ordonnée réelle vers l'ordonnée locale ;
- **getRealCoordX**, pour la conversion d'une abscisse locale vers l'abscisse réelle ;
- **getRealCoordY**, pour la conversion d'une ordonnée locale vers l'ordonnée réelle.

Afin de mieux comprendre le processus de conversion, observons le schéma suivant :



Le rectangle bleu représente la zone qui est affichée à l'écran. Son origine est localisée au coin haut/gauche, l'axe X local positif est orienté vers la droite, l'axe Y local positif vers le bas.
Son centre est décalé par rapport à l'origine du repère réel de **offsetX** suivant l'axe X, et de **offsetY** suivant l'axe Y.

Le signe des valeurs **offsetX** et **offsetY** est défini relativement au repère local, c'est à dire qu'une valeur positive décale la zone bleue respectivement vers la gauche et vers le bas.

La procédure de conversion « repère réel vers repère local » est très simple :

- on part de la valeur de coordonnée dans le repère réel ;
- on multiplie par un facteur échelle/ECHELLE_BASE ;
- on multiplie par un facteur -1 si on calcule une valeur Y, afin de réorienter l'axe vers le bas ;
- on ajoute respectivement W/2 ou H/2 suivant que l'on calcule une valeur X ou Y, afin de positionner l'origine au centre de la zone d'affichage ;
- on ajoute respectivement le décalage **offsetX** ou **offsetY** suivant que l'on calcule avec une valeur

X ou Y afin de décaler l'origine à sa position réelle dans le repère local.

La procédure de conversion inverse, « repère local vers repère réel », reprend la procédure ci-dessus, mais en sens inverse.
Les quatre méthodes permettant de faire ces conversions sont présentées ci-après :

Conversion de coordonnées

```
    /**
     * Convertit une abscisse réelle en abscisse
locale (repère JPanel)
     * @param x l'abscisse réelle
     * @return l'abscisse locale
     */
    public int getLocalCoordX(double x) {

        return (int) Math.round(this.getWidth() /
2 + this.offsetX + x * this.echelle /
ECHELLE_BASE);

    }

    /**
     * Convertit une ordonnée réelle en ordonnée
locale (repère JPanel)
     * @param y l'ordonnée réelle
     * @return l'ordonnée locale
     */
    public int getLocalCoordY(double y) {

        return (int) Math.round(this.getHeight()
/ 2 + this.offsetY - y * this.echelle /
ECHELLE_BASE);

    }

    /**
     * Convertit une abscisse locale (repère
JPanel) en abscisse réelle
     * @param x l'abscisse locale
     * @return l'abscisse réelle
     */
    public double getRealCoordX(double x) {

        return (x - (this.getWidth() / 2 +
this.offsetX)) * ECHELLE_BASE / this.echelle;

    }

    /**
     * Convertit une ordonnée locale (repère
JPanel) en ordonnée réelle
     * @param y l'ordonnée locale
     * @return l'ordonnée réelle
     */
    public double getRealCoordY(double y) {

        return (y - (this.getHeight() / 2 +
this.offsetY)) * -ECHELLE_BASE / this.echelle;

    }
```

2.1.2. Surcharge de paintComponents

La méthode à modifier pour personnaliser l'affichage de notre **JPanel** est la méthode **paintComponents**. Mais avant de la modifier, il faut savoir ce que nous voulons lui faire dessiner !

Dans notre cas, l'affichage se compose :

- d'un fond noir ;
- des axes X et Y en traits pleins verts ;
- des axes de quadrillage sens X et sens Y, en traits pointillés vert foncé (codage RGB=60/85/60), avec un pas **stepX** en sens X et **stepY** en sens Y ;
- des éléments graphiques qui seront dessinés par le gestionnaire d'éléments graphiques.

Ceci se traduit dans le code Java par les blocs suivants :

- dessin avec **fillRect** d'un rectangle plein noir occupant toute la zone d'affichage ;
- changement de couleur de dessin avec **setColor**, et dessin avec **drawLine** d'une ligne horizontale (à l'ordonnée y=0 locale) et d'une ligne verticale (à l'abscisse x=0 locale) ;
- définition du motif des pointillés avec **setStroke** et **BasicStroke**, puis dessin du quadrillage ;
- appel de la méthode de dessin du gestionnaire d'éléments graphiques.

Le code complet de la méthode **paintComponents** est présenté ci-après :

Méthode paintComponents

```
    // surcharge de la méthode paintComponent
afin de dessiner notre vue personnalisée
    @Override
    public void paintComponent(Graphics g) {

        Graphics2D g2 = (Graphics2D) g;

        int h = this.getHeight();
        int w = this.getWidth();

        g.setColor(Color.BLACK);
        g.fillRect(0, 0, w, h);

        // tracé des axes et du quadrillage

        int roundOffsetX = (int)
Math.round(this.offsetX);
        int roundOffsetY = (int)
Math.round(this.offsetY);
        int localStepX = (int)
Math.round(this.echelle / ECHELLE_BASE * stepX);
        int localStepY = (int)
Math.round(this.echelle / ECHELLE_BASE * stepY);

        g.setColor(Color.GREEN);
        g.drawLine(w / 2 + roundOffsetX, 0, w / 2
+ roundOffsetX, h);
        g.drawLine(0, h / 2 + roundOffsetY, w, h
/ 2 + roundOffsetY);

        int offsetStrokeX = 7 - roundOffsetX % 7;
        int offsetStrokeY = 7 - roundOffsetY % 7;

        float[] dash = {2, 5};
        BasicStroke bsX = new BasicStroke(1,
BasicStroke.CAP_SQUARE, BasicStroke.JOIN_ROUND,
10, dash, offsetStrokeX);
        BasicStroke bsY = new BasicStroke(1,
BasicStroke.CAP_SQUARE, BasicStroke.JOIN_ROUND,
10, dash, offsetStrokeY);

        g.setColor(new Color(60, 85, 60));
```

```
        g2.setStroke(bsY);

        int nbxp = (w - (w / 2 + roundOffsetX)) /
localStepX + 1;
        int nbxm = (w / 2 + roundOffsetX) /
localStepX + 1;

        for (int i = 1; i < nbxp; i++) {
            int xline = w / 2 + roundOffsetX +
(int) Math.round(i * stepX * this.echelle /
ECHELLE_BASE);
            g.drawLine(xline, 0, xline, h);

        }
        for (int i = 1; i < nbxm; i++) {
            int xline = w / 2 + roundOffsetX -
(int) Math.round(i * stepX * this.echelle /
ECHELLE_BASE);
            g.drawLine(xline, 0, xline, h);
        }

        g2.setStroke(bsX);

        int nbym = (h - (h / 2 + roundOffsetY)) /
localStepY + 1;
        int nbyp = (h / 2 + roundOffsetY) /
localStepY + 1;

        for (int i = 1; i < nbyp; i++) {
            int yline = h / 2 + roundOffsetY -
(int) Math.round(i * stepY * this.echelle /
ECHELLE_BASE);
            g.drawLine(0, yline, w, yline);
        }
        for (int i = 1; i < nbym; i++) {
            int yline = h / 2 + roundOffsetY +
(int) Math.round(i * stepY * this.echelle /
ECHELLE_BASE);
            g.drawLine(0, yline, w, yline);
        }

        g2.setStroke(new BasicStroke());

        if (this.pom != null) {
            this.pom.draw(g, this);
        }
    }
```

2.1.3. Interaction avec la souris

La classe **PFEM2DGuiPanel** étendant les interfaces **MouseListener** et **MouseMotionListener**, les événements suivants sont gérés par la classe, avec les actions associées ci-dessous :

- mouseDragged : suivant le mode courant, et en fonction du déplacement de la souris :
  - en mode « view » : pas d'effet,
  - en mode « drag » : déplace la zone affichée,
  - en mode « scale » : modifie le facteur de zoom ;
- mouseMoved ;
- mouseClicked : pas d'action associée ;
- mousePressed : changement de mode suivant le bouton de la souris pressé :
  - bouton gauche : bascule en mode "drag" (déplacement),
  - bouton milieu : bascule en mode « scale » (zoom) ;
- mouseReleased : passage en mode « view » (visualisation) ;
- mouseEntered : pas d'action associée ;
- mouseExited : pas d'action associée.

La méthode **mousePressed** est relativement simple, elle modifie l'état de la propriété **mode** suivant le bouton souris pressé et initialise les propriétés **xOld** et **yOld** (ainsi que les propriétés **xstart** et **ystart** en cas de mode « scale » activé) avec les coordonnées de la souris.
La méthode **mouseDragged** est un peu plus complexe, étant donné qu'elle gère l'influence du déplacement de la souris pour les deux modes principaux d'interaction :

- En mode « drag » (déplacement de la zone d'affichage), le déplacement de la souris provoque un déplacement similaire de la zone d'affichage. Par exemple, si la souris se déplace de 10 pixels vers la droite, la zone d'affichage se déplace également de 10 pixels vers la droite. Pour cela, on ajoute aux propriétés **offsetX** et **offsetY** (qui gèrent le décalage de la zone d'affichage) l'écart de position entre la position actuelle de la souris et la position précédente, gardée en mémoire dans les propriétés **xOld** et **yOld**. Les valeurs de **xOld** et **yOld** sont finalement modifiées pour prendre celles de la position actuelle de la souris.
- En mode « scale » (zoom de la zone d'affichage), le déplacement de la souris provoque une modification du facteur d'échelle de la zone d'affichage. Afin d'avoir une interaction « intuitive », le centre du zoom est positionné à l'endroit où l'utilisateur presse le bouton milieu de la souris (événement géré par la méthode **mousePressed** qui place les coordonnées de ce point dans les propriétés **xstart** et **ystart**). Cette fonctionnalité complique un peu le calcul dans la mesure où une modification de l'échelle seule va zoomer la zone d'affichage en prenant l'origine du repère réel comme centre de zoom.
  Afin de positionner le centre de zoom là où l'utilisateur a initialement cliqué, il nous faut modifier simultanément les valeurs de l'échelle et des décalages **offsetX** et **offsetY**. Ces modifications sont présentées dans le code complet des méthodes de gestion des événements en lignes 171 à 173.
  De la même manière qu'en mode « drag », les valeurs de **xOld** et **yOld** sont finalement modifiées pour prendre celles de la position actuelle de la souris.

Les formules de calcul des décalages pour le zoom s'appuient sur le fait que le centre de zoom doit garder les mêmes coordonnées en repère local quelque soit la valeur de l'échelle.

Le code complet des méthodes de gestion des événements souris est présenté ci-après :

Interaction avec la souris

```
    // surcharge de la méthode mouseDragged.
Gestion du déplacement souris avec bouton appuyé
    @Override
    public void mouseDragged(MouseEvent e) {
```

```
        if (this.mode == DRAG) {
            int x = e.getX();
            int y = e.getY();

            this.offsetX = this.offsetX + (x -
this.xold);
            this.offsetY = this.offsetY + (y -
this.yold);

            this.xold = x;
            this.yold = y;

            this.repaint();
        } else if (this.mode == SCALE) {
            int x = e.getX();
            int y = e.getY();

            int dx = x - this.xold;
            int dy = y - this.yold;

            int delta;
            if (Math.abs(dx) > Math.abs(dy)) {
                delta = dx;
            } else {
                delta = dy;
            }

            double newEchelle = Math.max(1,
this.echelle * Math.pow(1.01, delta));
            double newOffsetX = this.offsetX +
this.getRealCoordX(this.xstart) / ECHELLE_BASE *
(this.echelle - newEchelle);
            double newOffsetY = this.offsetY -
this.getRealCoordY(this.ystart) / ECHELLE_BASE *
(this.echelle - newEchelle);

            this.offsetX = newOffsetX;
            this.offsetY = newOffsetY;
            this.echelle = newEchelle;

            this.xold = x;
            this.yold = y;

            this.repaint();
        }
    }

    @Override
    public void mouseMoved(MouseEvent e) {
    }

    @Override
    public void mouseClicked(MouseEvent e) {
    }

    // surcharge de la méthode mousePressed.
Gestion du clic souris.
    @Override
    public void mousePressed(MouseEvent e) {
        this.xold = e.getX();
        this.yold = e.getY();

        if (e.getButton() == MouseEvent.BUTTON1)
{
            this.mode = DRAG;
        } else if (e.getButton() ==
MouseEvent.BUTTON2) {
            this.mode = SCALE;
            this.xstart = e.getX();
            this.ystart = e.getY();
        } else {
            this.mode = VIEW;
        }
    }

    // surcharge de la méthode mouseReleased.
Gestion du laché de clic souris.
    @Override
    public void mouseReleased(MouseEvent e) {
        this.mode = VIEW;
    }

    @Override
    public void mouseEntered(MouseEvent e) {
    }

    @Override
    public void mouseExited(MouseEvent e) {
    }
```

2.2. Capture d'écran

La première méthode additionnelle effectue une capture d'écran de la zone d'affichage du composant.
Un **JfFileChooser** permet de sélectionner le fichier de sortie dans lequel sera enregistrée l'image. Seuls deux formats sont autorisés dans cette méthode, le png et le jpg. Après vérification de l'extension du fichier, l'image est dessinée dans une **BufferedImage**, puis enregistrée dans le fichier choisi.

Méthode takeScreenshot

```
     /**
     * Crée une capture d'écran de l'affichage
courant.
     * Le fichier de sortie est défini par
sélection du fichier via un JFileChooser.
     * Deux formats possibles en sortie : png ou
jpg
     */
    public void takeScreenshot() {

        JFileChooser fc = new JFileChooser();

        int returnVal = fc.showOpenDialog(this);

        if (returnVal ==
JFileChooser.APPROVE_OPTION) {

            File fichier = fc.getSelectedFile();
            String nomFichier =
fichier.getName().toLowerCase();

            if ((nomFichier.endsWith("png")) ||
(nomFichier.endsWith("jpg"))) {
                try {
                    BufferedImage bufImage = new
BufferedImage(this.getSize().width,
this.getSize().height,
BufferedImage.TYPE_INT_RGB);

this.paint(bufImage.createGraphics());

                    String extension = "png";
                    if
(nomFichier.endsWith("jpg")) {
                        extension = "jpg";
                    }

                    ImageIO.write(bufImage,
```

```
extension, fichier);
                } catch (IOException ex) {
                    ex.printStackTrace();
                }
            } else {

JOptionPane.showMessageDialog(this, "Veuillez
préciser l'extension du fichier image (png ou
jpg).",
                        "Extension fichier
manquante", JOptionPane.ERROR_MESSAGE);
            }
        }
    }
```

2.3. Recadrage de l'affichage

La seconde méthode additionnelle recentre l'affichage sur les éléments graphiques à afficher :

Méthode centreView

```
    /**
     * Centre la vue sur les éléments à afficher.
     * @param boundingBox tableau de 4 double :
Xmin, Xmax, Ymin,Ymax, en coordonnées réelles
     */
    public void centreView(double[] boundingBox)
{

        if ((boundingBox[1] - boundingBox[0] !=
Double.NEGATIVE_INFINITY) && (boundingBox[3] -
boundingBox[2] != Double.NEGATIVE_INFINITY)) {

            double echelleX = (this.getWidth() -
50) * ECHELLE_BASE / (boundingBox[1] -
boundingBox[0]);
            double echelleY = (this.getHeight() -
50) * ECHELLE_BASE / (boundingBox[3] -
boundingBox[2]);

            this.echelle = Math.min(echelleX,
echelleY);

            this.offsetX = -
(this.getLocalCoordX((boundingBox[1] +
boundingBox[0]) / 2) - this.getLocalCoordX(0));
            this.offsetY = -
(this.getLocalCoordY((boundingBox[3] +
boundingBox[2]) / 2) - this.getLocalCoordY(0));

            this.repaint();
        }
    }
```

Cette méthode nécessite le passage en paramètre des coordonnées de la « bounding box », ou boite englobante en français, afin de pouvoir recadrer au mieux la vue sur les éléments à afficher.
Une vérification sur la taille de la « bounding box » est effectuée, de manière à ne pas recentrer la vue s'il n'y a rien à afficher !
Une marge de 50 pixels est ajoutée autour de la zone d'affichage afin d'améliorer le rendu.
Les échelles sens X et Y sont calculées, l'échelle finale étant le mini de ces deux échelles.
Les décalages sens X et Y sont calculés comme étant les distances du centre de la « bounding box » au centre de l'affichage (point (0,0) local).
Au final, un rafraichissement de l'affichage est appelé.

Retrouvez l'article de Jean-Michel Borlot en ligne : Lien 27

Eclipse

Les dernières news

La fondation Eclipse publie Eclipse Kepler 4.3

Une nouvelle version d'Eclipse est disponible. Elle porte le nom de Kepler. Cette version marque la fin officielle du support de la branche 3.x d'Eclipse par la fondation. Elle continue donc sur la lancée de Juno.

Des informations supplémentaires sur les nouveautés de cette version sont disponibles à cette adresse : notes pour la version 4.3 (Lien 28).

Le projet Kepler se compose de **72 projets** (114 en comptant les sous-projets), pour un total d'environ **58 millions de lignes de code** par **428 committers**. **Cinq projets** ont rejoint le « simulatenous release train » : EMF Diff/merge, Sphinx, Stardust, Hudson et Maven integration pour WTP (Web Tools Platform).
Des améliorations ont été faites au niveau de l'interface, comme la possibilité de combiner plusieurs vues dans une même fenêtre « détachée », particulièrement intéressante pour les développeurs travaillant sur plusieurs écrans, le repositionnement des barres d'outils ou la possibilité de choisir d'emblée l'éditeur pour l'ouverture d'un fichier directement dans la fenêtre « Open Resource ».

Pour télécharger cette nouvelle version, rendez-vous sur la page de téléchargement d'Eclipse : Lien 29

Dans les nouveautés, nous pouvons citer :

- la mise en place d'un outil de remédiation dans p2 pour les problèmes d'installation de plugins qui propose automatiquement une solution pour continuer l'installation ;
- le déploiement d'Orion en version 3.0, plateforme Web pour le développement d'applications Web ;
- une amélioration globale des performances au niveau de l'interface ;
- le déploiement d'Eclipse RAP en version 2.0 ;
- la prise en charge de Java EE 7 par le projet WTP (Web Tools Platform) ;
- un outil de recherche d'icônes directement intégré pour rendre plus facile leur utilisation dans le développement de plugins.

La liste des nouveautés se trouve ici :

- pour le workbench : Lien 30
- pour JDT : Lien 31
- pour Eclipse RCP : Lien 32

Et vous ?

- Que pensez-vous de cette nouvelle version ? Et de ces nouvelles fonctionnalités ?
- Avez-vous déjà essayé Eclipse 4.3 Kepler ?

Commentez la news de Mickael Baron en ligne : Lien 33

Les derniers tutoriels et articles

Les raccourcis d'Eclipse

Ce document présente les raccourcis d'Eclipse les plus utiles. Il vous offre également un mémento à imprimer vous-même.

1. Introduction

Téléchargez le mémento ici : memento-eclipse.pdf sur Developpez.com (Lien 34) ou memento-eclipse.pdf sur Icauda (Lien 35).
Les IDE tels qu'Eclipse sont de vraies mines d'or. Ils proposent de très nombreuses fonctionnalités. Comprendre son IDE est indispensable pour en tirer le meilleur et être le plus productif possible. Pour cela, il faut notamment connaitre (et utiliser) les raccourcis clavier. Ces derniers ne vous seront pas forcément tous utiles, mais la plupart vous feront réellement gagner du temps. Reste à s'en souvenir. Pour ma part, je dois dire que je les oublie sans arrêt.
Ce document vous propose de découvrir les principaux raccourcis, dont certains se ressemblent tant dans les combinaisons de touches utilisées que dans leurs effets.
Je vous propose aussi de télécharger gratuitement un mémento papier que vous pourrez laisser sur un coin de votre bureau et relire de temps en temps.

1.1. Versions

19 juin 2013 : création du document.

2. Le mémento

En plus de la version Web (en HTML) de ce document, que vous êtes en train de lire, je vous propose de télécharger gratuitement un mémento à imprimer vous-même en recto-verso.
Téléchargez le mémento ici : memento-eclipse.pdf sur Developpez.com (Lien 34) ou memento-eclipse.pdf sur Icauda (Lien 35)

La licence *Creative Commons* vous demande de respecter trois points : attribution, pas d'utilisation commerciale et pas d'œuvre dérivée. Si vous souhaitez modifier ce mémento, par exemple pour l'utiliser dans une de vos plaquettes ou pour ajouter votre logo, il vous suffit de me contacter et j'aurai le plaisir de vous accompagner et de vous offrir (gratuitement) une version utilisable et plus pratique.

3. Les raccourcis

Eclipse propose de nombreux raccourcis dont certains utilisent les mêmes combinaisons de touches. C'est le contexte qui détermine alors l'action appropriée. Par exemple, la touche F2 affiche la description d'un élément quand il est sélectionné. Quand c'est un objet qui est sélectionné, la touche F2 permet de le renommer.

3.1. Édition

Raccourcis	Action
Ctrl+Espace	Assister au contenu (dont autocomplétion)
Alt+/	Complétion du mot
Ctrl+Maj+Espace	Informations de contexte
Ctrl+Maj+Y	Mettre en minuscules
Ctrl+Maj+X	Mettre en majuscules
Alt+Maj+O	Marquer/démarquer les occurrences
Ctrl+C / Ctrl+Insert	Copier
Ctrl+X / Maj+Suppr	Couper
Ctrl+V / Maj+Insert	Coller
Ctrl+Alt+Haut / Ctrl+Alt+Bas	Dupliquer la ligne en cours ou des lignes sélectionnées (en haut ou en bas)
Ctrl+D	Effacer la ligne
Ctrl+Suppr	Effacer le mot suivant
Ctrl+Arrière	Effacer le mot précédent
Ctrl+Maj+Suppr	Effacer jusqu'à la fin de la ligne
Ctrl+Maj+Entrée	Insérer une ligne au-dessus
Maj+Entrée	Insérer une ligne en dessous
Home	Aller en début de ligne
Fin	Aller en fin de ligne
Alt+Bas	Descendre de N lignes
Alt+Haut	Monter de N lignes
Ctrl+Droite	Aller au mot suivant
Ctrl+Gauche	Aller au mot précédent
Ctrl+Bas	Scroller vers le bas
Ctrl+Bas	Scroller vers le haut
Ctrl+Home	Aller en haut du fichier
Ctrl+Fin	Aller en bas du fichier
Ctrl+Maj+Q	Basculer en diff rapide
Ctrl+1	Proposer une solution rapide
Ctrl+Z	Défaire
Ctrl+Y	Refaire
F2	Afficher la description
Ctrl+Maj+Insert	Basculer en mode insertion

3.2. Sélection

Raccourcis	Action
Alt+Maj+Bas	Restaurer la dernière sélection
Ctrl+A	Sélectionner tout
Alt+Maj+Haut	Sélectionner élément englobant
Alt+Maj+Droite	Sélectionner l'élément suivant
Alt+Maj+Gauche	Sélectionner l'élément précédent
Ctrl+Maj+Droite	Sélectionner le mot suivant
Ctrl+Maj+Gauche	Sélectionner le mot précédent
Maj+Fin	Sélectionner jusqu'à la fin de la ligne
Maj+Home	Sélectionner jusqu'au

3.3. Recherche

Raccourcis	Action
Ctrl+K	Rechercher le suivant
Ctrl+Maj+K	Rechercher le précédent
Ctrl+F	Rechercher/remplacer
Ctrl+J	Rechercher incrémentalement
Ctrl+Maj+J	Rechercher anti-incrémentalement (inversée)
Ctrl+G	Rechercher la déclaration dans le « workspace »
Ctrl+Alt+G	Rechercher le texte dans le « workspace »
Ctrl+H	Ouvrir la boite de recherche
Ctrl+Maj+G	Rechercher les références dans le « workspace »
Ctrl+Maj+U	Rechercher les occurrences

3.4. Refactoring

Raccourcis	Action
Alt+Maj+C	Modifier la signature d'une méthode
Alt+Maj+L	Extraire une variable locale
Alt+Maj+M	Extraire une méthode
Alt+Maj+R	Renommer
Alt+Maj+T	Ouvrir le menu de refactoring

3.5. Source

Raccourcis	Action
Ctrl+Maj+/	Commenter/décommenter le bloc
Ctrl+Maj+M	Importer l'élément
Ctrl+Maj+J	Initier la Javadoc
Ctrl+Maj+F	Formater
Ctrl+I	Indenter le bloc
Ctrl+Maj+O	Organiser les imports
Alt+Maj+S	Ouvrir le menu rapide de source

Alt+Maj+Z	Ouvrir le menu rapide d'entourage (pour entourer avec if, try, for, etc.)

3.6. Debug

Raccourcis	Action
Alt+Maj+D	Ouvrir le menu de lancement de debug
F11	Relancer le dernier debug
F11	Relancer le dernier debug
F8	Continuer
Ctrl+F2	Terminer
F5	Aller dans l'instruction (entrer)
F6	Aller à l'instruction suivante
F7	Sortir
Ctrl+Maj+B	Ajouter/supprimer un point d'arrêt

3.7. Run

Raccourcis	Action
Alt+Maj+X	Ouvrir le menu de lancement
Ctrl+F11	Relancer la dernière exécution

3.8. Fichier

Raccourcis	Action
Ctrl+F4 / Ctrl+W	Fermer
Ctrl+Maj+F4 / Ctrl+Maj+W	Fermer tout
Ctrl+N	Créer nouveau (via boite d'assistance)
Alt+Maj+N	Créer nouveau (via menu)
Ctrl+P	Imprimer
Alt+Entrée	Afficher les propriétés
F5	Rafraichir
F2	Renommer
Ctrl+S	Sauver
Ctrl+Maj+S	Sauver tout

3.9. Navigation

Raccourcis	Action
Alt+Gauche	Retourne au précédent (dans l'historique)
Alt+Droite	Retourne au suivant (dans l'historique)
Ctrl+L	Aller à une ligne (en indiquant son numéro)
Ctrl+Maj+P	Aller à l'accolade correspondante (de l'ouvrante à la fermante, et réciproquement)
Ctrl+Maj+Bas	Aller au membre suivant (pour passer d'une méthode à la suivante par exemple)
Ctrl+Maj+Haut	Aller au membre précédent
Ctrl+Q	Aller au dernier emplacement modifié
Ctrl+Alt+H	Ouvrir la fenêtre d'appel hiérarchique
F3 / Ctrl+Clic	Aller à la déclaration d'un élément
Maj+F2	Voir la Javadoc externe
Ctrl+Maj+R	Ouvrir une ressource (fichier/classe/etc.)
Ctrl+Maj+T	Ouvrir un type (classe)
Ctrl+F3	Voir la structure (d'une classe par exemple)
F4	Voir la hiérarchie du type cible
Ctrl+Maj+H	Voir la hiérarchie d'un type
Ctrl+T	Voir la hiérarchie rapide du type cible
Ctrl+O	Voir le « outline » rapide du type cible
Alt+Maj+W	Voir menu pour le type cible
Ctrl+F10	Afficher le menu de la vue
Ctrl+M	Maximiser/rétablir la vue/perspective
Ctrl+F6	Aller à l'éditeur suivant
Ctrl+Maj+F6	Aller à l'éditeur précédent
Ctrl+F7	Aller à la vue suivante
Ctrl+Maj+F7	Aller à la vue précédente
Ctrl+F8	Aller à la perspective suivante
Ctrl+Maj+F8	Aller à la perspective précédente
Ctrl+E	Switcher rapidement d'éditeur
Ctrl+Maj+E	Switcher d'éditeur
Ctrl+Maj+L	Voir la liste des raccourcis

4. Conclusions

Voilà donc une centaine de raccourcis. Il en existe d'autres, mais il est peu probable que vous les utilisiez donc je vous les épargne. Avec l'habitude, vous en utiliserez certains au quotidien (comme Ctrl+Maj+F ou le basique Ctrl+C) et vous oublierez sans doute les autres. Pensez à regarder cet article, et surtout le mémento papier, de temps en temps car les raccourcis peuvent vraiment vous faire gagner en productivité dans le cadre de votre utilisation d'Eclipse.

Retrouvez l'article de Thierry Leriche-Dessirier en ligne : Lien 37

NetBeans 7.4 : support du JDK 8, nouvelles fonctionnalités HTML5, optimisation des performances et du « refactoring », la bêta de l'EDI open source sort

NetBeans franchit un nouveau cap pour offrir une meilleure expérience aux développeurs pour la création d'applications Java, C, C++ et Web (PHP, HTML, JavaScript, CSS).

Oracle vient de publier la bêta de la version 7.4 de l'environnement de développement open source, qui lève le voile sur un lot de nouveautés pour le développement HTML5, Java EE et mobile pour Android et iOS.

NetBeans 7.4 bêta apporte des nouvelles fonctionnalités HTML5 pour la prise en charge du développement d'applications Web mobiles hybrides en utilisant le framework Web PhoneGap, le support des navigateurs iOS et Android, la prise en charge de l'édition des feuilles de styles SASS et LESS, et l'enregistrement des modifications de « Chrome Developer Tools ».

Les fonctionnalités HTML5 de l'EDI peuvent être utilisées dans les projets Java EE et les applications PHP.

Les développeurs PHP vont apprécier la prise en charge de Nette Framework 2 (avec les Template Latte), de Zend Framework 2, du framework de test Atoum, de l'analyse statique du code et des améliorations de l'éditeur et du « refactoring ».

Toujours côté développement Web, JavaScript bénéficie d'une meilleure prise en charge des frameworks AngularJS, Knockout et ExtJS. Les utilisateurs du langage de script pourront tirer parti des améliorations de la précision de la complétion de code et de JSON (JavaScript Object Notation).

En ce qui concerne Java, de nouvelles caractéristiques de la préversion de JDK 8 ont été intégrées, notamment les Profiles et les expressions Lambdas. La complétion de code, les conseils et le Refactoring passent à une étape supérieure. L'EDI est livré avec les mises à jour de certains outils, notamment Ant 1.9.0 et Maven 3.0.5. Le « Native Packaging » fait également son apparition.

Pour JavaFX, la plateforme de développement d'applications Web riches (RIA), une nouvelle boite de dialogue pour les fichiers FXML est disponible pour les projets Maven, des améliorations des options de déploiement des projets et une meilleure intégration avec les projets JavaSE sont au rendez-vous.

Les développeurs C/C++ n'ont pas été oubliés. NetBeans 7.4 apporte pour ceux-ci une optimisation du débogage et de l'exécution des applications, des améliorations des fonctions de recherche et le formatage de style C/C++ par projet.

Globalement, NetBeans 7.4 bêta optimise les tâches de « profiling », améliore la prise en charge des outils de contrôle de versions (Subversion, Git et Mercurial) et des bases de données.

Commentez la news de Hinault Romaric en ligne : Lien 38

80 % des appareils mobiles vendus au second trimestre utilisent Android, l'OS de Google s'impose plus que jamais

Avec désormais 79,3 % des parts soit environ 187 millions d'appareils vendus de par le monde pour le deuxième trimestre de cette année, Android règne sans partage sur le marché du mobile. Un vaste et solide partenariat que Google a établi avec des constructeurs comme Samsung, LG, Motorola Huawei et bien d'autres. En un mot comme en cent, le succès incontesté de ce système d'exploitation ne saurait être attribué à un seul constructeur, mais bel et bien à l'ensemble, même si Samsung y contribue beaucoup plus largement que les autres.

Des chiffres révélés par l'IDC qui donnent le tourni : iOS, son dauphin, ne représente que 13,2 % des parts avec environ 31 millions d'appareils vendus. Quant au Windows Phone, il se cantonne à la troisième place avec un honorable 3,7 % représentant environ 9 millions d'appareils vendus.

Top Smartphone Operating Systems, Shipments, and Market Share, 2013 Q3 (Units in Millions)

Operating System	2Q13 Unit Shipments	2Q13 Market Share	2Q12 Unit Shipments	2Q12 Market Share	Year-over-Year Change
Android	187.4	79.3%	108	69.1%	73.5%
iOS	31.2	13.2%	26	16.6%	20.0%
Windows Phone	8.7	3.7%	4.9	3.1%	77.6%
BlackBerry OS	6.8	2.9%	7.7	4.9%	-11.7%
Linux	1.8	0.8%	2.8	1.8%	-35.7%
Symbian	0.5	0.2%	6.5	4.2%	-92.3%
Others	N/A	0.0%	0.3	0.2%	-100.0%
Total	**236.4**	**100.0%**	**156.2**	**100.0%**	**51.3%**

Les trois premiers enregistrent respectivement une croissance de vente annuelle de 73,5 %, 20 % et 77,6 % tandis que les autres connaissent des baisses dans leurs ventes. Ramon Llamas, responsable recherche de l'équipe Mobile Phone de l'IDC, attribue le déclin d'iOS à la cyclicité de l'iPhone. « *Sans un nouveau lancement de produit depuis l'iPhone 5 depuis près d'un an déjà, les parts de marché d'Apple étaient vulnérables face au lancement de nouveaux produits chez la concurrence. Mais avec un nouvel iPhone et un iOS retravaillé, Apple est bien parti pour regagner du terrain.* »

En général, le marché du mobile connait une croissance de 51,3 % et passe de 156 millions d'appareils vendus l'année dernière à la même période à 236 millions d'appareils vendus.

Commentez la news de Stéphane Le Calme en ligne : Lien 39

Google officialise Android 4.3 Jelly Bean, introduit OpenGL SE 3.0, Bluetooth Smart Ready et une meilleure gestion du multiutilisateur

Après une longue attente, Google a finalement dévoilé Android 4.3 lors d'une conférence de presse organisée par la société hier.

Cette nouvelle itération de la branche 4, Jelly Bean, n'apporte pas de grandes évolutions pour les utilisateurs. La présentation de l'écran d'accueil reste inchangée, ainsi que la recherche et les notifications.

La nouveauté intéressante d'Android 4.3 est la fonctionnalité « Multi-User Restricted Profiles », permettant de disposer d'un meilleur contrôle sur l'utilisation des applications et autres contenus pour un utilisateur. Le multiutilisateur est apparu dans l'OS avec Android 4.2.

Les développeurs vont apprécier la prise en charge d'OpenGL ES 3.0. OpenGL est une interface que les développeurs peuvent utiliser pour dessiner des objets 3D et contrôler les effets visuels dans les jeux. Le support de la dernière version de la plateforme fournira aux applications de meilleures performances graphiques.

Le nouveau Bluetooth Smart Ready facilitera la détection des dispositifs compatibles à proximité. Google a optimisé la gestion de la consommation d'énergie.

Cette mouture introduit également l'écoute silencieuse des ondes Wi-Fi pour faciliter la géolocalisation. À première vue, cette fonctionnalité serait ressentie comme gourmande en énergie. Cependant, elle est une alternative sérieuse au GPS, qui est davantage gourmand en énergie pour effectuer des opérations de localisation.

D'autres modifications mineures ont été apportées à l'OS, comme de nouveaux emojis dont le thème est basé sur la nature, une nouvelle fonte de Roboto et l'ajout de plusieurs widgets.

Le déploiement d'Android 4.3 a débuté hier en OTA pour les appareils Galaxy Nexus, les Nexus 4, 7 et 10.

Commentez la news de Hinault Romaric en ligne : Lien 40

Les derniers tutoriels et articles

Gestion des exceptions - En Java pour Android (et Eclipse)

Les exceptions sont très importantes en Java et leur gestion correcte permet non seulement un développement plus simple, et moins sujet aux « bogues », mais surtout permet de fournir des indications précieuses à l'utilisateur.

1. Les exceptions

Il existe sur Developpez.com un très bon cours Java dont un chapitre traite exclusivement des exceptions : Chapitre Exceptions (Lien 41).
Un tutoriel donne aussi quelques bonnes pratiques sur la gestion des exceptions : Tutoriel (Lien 42).
Vous y trouverez les généralités nécessaires pour la bonne compréhension des exceptions en Java, et des bonnes pratiques pour les gérer. Nous allons voir ici quelques aspects « philosophiques » sur les exceptions, et comment véritablement les gérer dans un environnement Java Android.

1.1. Pourquoi les exceptions ?

1.1.1. Un outil de gestion des erreurs

Comme son nom l'indique, une exception signifie que quelque chose d'inattendu s'est produit pendant l'exécution du code, et que celle-ci n'a pu arriver à son terme.
Ce peut être dû à l'environnement ou à une mauvaise utilisation du code, ou simplement encore, à une « erreur simple ».
Il existe en effet deux écoles en programmation. L'une suggère de réserver les exceptions à des événements inattendus, l'autre utilise les exceptions pour signaler les erreurs « normales ».
Prenons l'exemple d'un paiement par carte de crédit. Des exceptions signaleront forcément les cas où le service n'est pas joignable, ou une erreur dans les données transmises. Mais quid des événements du style « pas assez d'argent à la banque » ? La première école utilisera un champ d'erreur dans l'objet retourné par la fonction, l'autre utilisera une exception.
La « bataille » est cependant moins « forte » dans le monde Java, car le langage distingue deux types d'exceptions. Les exceptions « classiques » (nécessitant d'être déclarées par la fonction), et les exceptions « runtime ». Ainsi, Java propose un mécanisme de « déclaration » des erreurs possibles, et à mon avis, autant l'utiliser.

1.1.2. Un outil de débogage et de développement

Les exceptions ne sont pas de simples objets sans contenu, une exception maintient une foule d'informations très utiles au débogage (entre autres) :

- une « stacktrace » : c'est-à-dire l'état de la pile des appels de fonctions au moment où l'exception a été construite (« new »). Cette trace inclut les noms de fichiers et lignes, mais aussi les noms de classes et de fonctions. Une information inestimable ;
- l'exception peut contenir une « cause ». Par exemple, une exception « HTTP » pourrait avoir comme cause une exception « réseau » (un problème de communication) ou une exception « serveur » (le serveur a refusé la demande) ;
- le « logging » devient alors primordial, et utiliser les fonctions Android prévues à cet effet : Log.e(TAG, « un message », exception) permet d'obtenir toutes les informations de l'exception dans le « logcat », absolument indispensable pour le débogage.

Voici un exemple de trace produit par une exception dans le logcat :

Stack Trace
```
06-05 09:24:19.939: E/AndroidRuntime(1554): FATAL EXCEPTION: main
06-05 09:24:19.939: E/AndroidRuntime(1554): android.content.res.Resources$NotFoundException: Resource ID #0x0
06-05 09:24:19.939: E/AndroidRuntime(1554): at android.content.res.Resources.getValue(Resources.java:1013)
06-05 09:24:19.939: E/AndroidRuntime(1554): at
```

```
android.content.res.Resources.loadXmlResourcePars
er(Resources.java:2098)
06-05 09:24:19.939: E/AndroidRuntime(1554):
at
android.content.res.Resources.getLayout(Resources
.java:852)
….
06-05 09:24:19.939: E/AndroidRuntime(1554):
at
android.os.Handler.dispatchMessage(Handler.java:9
2)
06-05 09:24:19.939: E/AndroidRuntime(1554):
at android.os.Looper.loop(Looper.java:137)
06-05 09:24:19.939: E/AndroidRuntime(1554):
at
android.app.ActivityThread.main(ActivityThread.ja
va:4745)
06-05 09:24:19.939: E/AndroidRuntime(1554):
at java.lang.reflect.Method.invokeNative(Native
Method)
06-05 09:24:19.939: E/AndroidRuntime(1554):
at
java.lang.reflect.Method.invoke(Method.java:511)
06-05 09:24:19.939: E/AndroidRuntime(1554):
at
com.android.internal.os.ZygoteInit$MethodAndArgsC
aller.run(ZygoteInit.java:786)
06-05 09:24:19.939: E/AndroidRuntime(1554):
at
com.android.internal.os.ZygoteInit.main(ZygoteIni
t.java:553)
06-05 09:24:19.939: E/AndroidRuntime(1554):
at dalvik.system.NativeStart.main(Native Method)
```

Comme on le voit, chaque ligne décrit un des appels de fonctions ayant conduit à l'exception. Par exemple, au moment de l'exception, le programme était dans la fonction run() de la classe MethodAndArgsCaller, déclarée dans la classe com.android.internal.os.ZygoteInit (fichier « ZygoteInit.java », ligne 786). La toute dernière ligne est toujours : dalvik.system.NativeStart.main(Native Method) puisque c'est le point d'entrée de tous les programmes Java.

1.2. Rappels de syntaxe

1.2.1. Levée d'une exception

En Java, une exception est simplement « levée » par l'utilisation du mot-clé « throw ». L'objet passé à la commande throw doit hériter de « Throwable » (« Exception » hérite de « Throwable »).

1.2.2. Déclaration d'exception

Toute fonction qui lève une exception non « runtime » (donc qui n'hérite pas de « RuntimeException »), doit impérativement le déclarer. Ceci est fait en rajoutant le mot-clé « throws » après la déclaration de la fonction, suivi par les classes d'exception que la fonction est susceptible de lancer.
Par exemple :

```
public void myFunction(int param) throws
IOException, AnotherException;
```

1.2.3. Try/Catch/Finally

À tout moment, il est possible d'intercepter les exceptions levées par une séquence de code.
Un bloc « try » est créé, dans lequel on mettra la séquence à protéger, suivie au moins d'un bloc « catch » ou « finally ».
Un bloc « catch » sera exécuté si une exception du type « catchée » est levée dans le code protégé.
Le bloc « finally » est toujours invoqué, qu'une exception ait été levée ou non, catchée ou non.
Sans bloc « catch » ou « finally », le bloc « try » devient alors inutile et provoquera une erreur de compilation.
Par exemple :

```
try {
   …
   code pouvant envoyer les exceptions
IOException, et NumberFormatException
   …
} catch (IllegalArguementException pe) {
   // si une exception de type
NumberFormatException a été levée, on passera ici
   // parce que NumberFormatException hérite de
IllegalArguementException
} finally {
   // on passera toujours ici :
   // à la fin du bloc try si aucune exception
n'a été levée,
   // après le bloc catch si une exception de
type "IllegalArguementException" a été levée
   // ou dès la levée de toute autre exception
}
```

Il est possible de rajouter autant de blocs « catch » que nécessaire… Ceux-ci sont vérifiés dans l'ordre de leur déclaration… Ainsi si une exception A hérite de B, alors le bloc « catch » de B devra apparaitre après le bloc « catch » de A.
Il n'est pas nécessaire d'avoir un bloc « catch » si on veut s'assurer qu'une séquence de code soit toujours exécutée (par exemple pour libérer des ressources), un bloc « finally » est suffisant.

2. Au niveau de la fonction

Pour voir comment gérer les exceptions dans votre programme, commençons au niveau le plus bas : la fonction.

2.1. Généralités

Toute fonction a des préconditions (conditions devant être remplies pour que la fonction puisse s'exécuter) et des postconditions (conditions remplies à l'issue de l'exécution de la fonction).
La règle générale est que si la fonction est incapable de fournir ses postconditions, elle doit lever une exception, sans avoir aucun autre effet de bord (comme si elle n'avait jamais été appelée).
Concernant les préconditions, c'est un peu comme vous le sentez (après tout c'est votre programme). Ou vous partez du principe que la fonction sera toujours invoquée correctement (pas de précondition à vérifier dans ce cas), ou vous préférez être tranquille et donc protéger les appels contre une mauvaise utilisation (et souvent contre vous-même).

2.1.1. Gérer les préconditions

À l'entrée dans la fonction, il est temps de vérifier que les paramètres passés suivent toutes les préconditions.
Si ce n'est pas le cas, une exception de type

« RuntimeException » (qui n'a donc pas besoin d'être déclarée par la fonction) devrait être levée. L'utilisation d'une exception de type « runtime » évite de polluer les blocs try/catch des fonctions appelantes (un défaut de préconditions étant une erreur du programmeur).
L'exception la plus utilisée dans ce cas de figure est sans doute « IllegalArgumentException ».
Attention, il existe un paramètre caché dans tous les appels de fonctions membres : « this » ! En effet, il est possible qu'on doive aussi vérifier l'objet sur lequel est appelée la fonction. Dans ce cas on utilise souvent « IllegalStateException » pour signaler que l'objet n'est pas dans un état correct. Encore une fois, c'est une « RuntimeException ».
Pour finir, il se peut que la machine virtuelle Java ne remplisse pas les préconditions (par exemple ne supporte pas un certain charset, ou une langue précise…), là encore nous pourrons utiliser une des innombrables RuntimeException (voire créer nos propres RuntimeException si besoin).
Dans tous les cas, le message de l'exception se doit d'être parfaitement compréhensible par tous les intervenants du projet : ce sont eux qui sont visés par ces exceptions, et en aucun cas, l'utilisateur de l'application.

2.1.2. Gérer les postconditions

2.1.2.1. Erreurs d'état

La fonction peut très bien ne pas fonctionner si un fichier est en lecture seule… ou si une heure donnée est dépassée, ou n'importe quoi d'autre. Si une telle erreur « d'état » est rencontrée et empêche de satisfaire les postconditions, une exception doit être levée.
Comme ce n'est pas une erreur de programmation et que cet état peut intéresser l'utilisateur (pas de réseau par exemple), une exception normale sera levée.
Pour que l'appelant de la fonction puisse distinguer les erreurs entre elles (et éventuellement agir différemment selon le type d'erreur), il faut surtout faire attention à ne pas mélanger les exceptions entre elles. La classe d'une exception signifie quelque chose, et en aucun cas, par exemple, on n'utilisera IOException pour signifier une erreur de contenu de données !
Du coup, la fonction devra déclarer l'envoi de ces exceptions, afin que, justement, l'appelant puisse les gérer au mieux.

2.1.2.2. Erreurs des sous-fonctions

Pendant l'exécution de la fonction, celle-ci va certainement appeler d'autres fonctions qui elles aussi peuvent envoyer des exceptions. Souvent des problèmes d'entrée/sortie (les exceptions les plus fréquentes), mais aussi des erreurs de données (parsing, format…).
Si la fonction est capable de gérer l'erreur, l'appel doit être protégé par un bloc try/catch, bien faire attention de ne « catcher » que les exceptions vraiment récupérables. Il est toutefois utile d'utiliser une méthode de log pour informer un programmeur que l'exception a eu lieu (et pourquoi). Par exemple :

```
try {
    // algorithme 1
    appel de fonction qui risque d'envoyer
ParseException
} catch (ParseException ex) {
    Log.i("MAFONCTION", "Erreur de parsing,
utilisation de l'algorithme 2",ex);
    // algorithme 2
}
```

Si la fonction est incapable de gérer l'erreur, elle doit arrêter son traitement, et, point sans doute le plus essentiel, revenir à l'état initial (comme si la fonction n'avait jamais été appelée). Par exemple une fonction « move » (sur un objet graphique par exemple) qui lève une exception, doit s'assurer qu'aucune coordonnée n'a été modifiée avant de lancer l'exception.
Ces exceptions sont en général gérées de manière globale à la fonction (un bloc try/catch/finally englobant la fonction en entier)… Les parties « catch » se chargeront de faire revenir la machine virtuelle à l'état initial et la partie « finally » à fermer l'ensemble des ressources potentiellement ouvertes :

```
InputStream is = null;
try {
   is = open(...);
   // algorithme
} catch (Exception1 error1) {
   // retour à l'état initial
} catch (Exception2 error2) {
   // retour à l'état initial
} finally {
   if (is != null) try { is.close(); } catch
(Exception ex) { Log.w("MAFONCTION", "Couldn't
close the stream!",ex); }
}
```

Si l'exception n'a pas besoin d'être traduite, et qu'il n'y a pas besoin de revenir à un état précédent (fonction sans effet de bord), il n'est bien sûr pas nécessaire de la « catcher », il suffit de déclarer l'exception comme étant lancée par la fonction.
Si l'exception n'a pas besoin d'être traduite, mais que l'on doit faire des choses pour « revenir » à l'état initial, elle doit être « catchée » (pour revenir à l'état initial) puis immédiatement renvoyée (voir dans le chapitre suivant une autre méthode).
Si l'exception a besoin d'être traduite en une autre exception, non seulement elle doit être « catchée », mais surtout passée en tant que « cause » à la nouvelle exception (après retour en arrière de l'état de la fonction bien entendu).
En aucun cas, la fonction ne doit s'arrêter sans lever d'exception pour signaler justement à l'appelant que tout ne s'est pas passé correctement.

2.1.2.3. Le paradigme de transaction

Plutôt que de devoir catcher toutes les exceptions et les relancer pour revenir à un état « précédent », il existe une manière de gérer en une fois toutes les exceptions. C'est le principe « transactionnel » utilisé dans les bases de données :

```
boolean success = false;
int storedValue = this.myValue;
try {
    // algorithme (qui risque d'envoyer des
exceptions)...
```

```
    // tout à la fin:
    success = true;
} finally {
   // ..
   // fermeture des ressources
   // ..
   if (!success) {
       // restauration de l'objet
       this.myValue = storedValue;
   }
}
```

En cas d'exception pendant l'algorithme, le code ne passera jamais sur « success = true », donc le bloc finally sera exécuté (il l'est toujours) avec « success » valant « false », la fermeture des ressources sera effectuée, et l'objet sera remis à son état initial.
Par contre, si aucune exception n'a eu lieu, success est bien passé à « true », et le bloc finally (toujours exécuté) va donc ne rien faire du tout (à part fermer les ressources).
Attention, si l'objet peut être accessible par plusieurs « threads » en même temps, il est indispensable d'implémenter un système de « lock » pendant la « transaction » (simplement déclarer la fonction comme étant « synchronized » peut suffire). Sinon un thread peut voir un objet dans un état « temporaire » qui ne sera pas forcément appliqué à la fin de la transaction !
Il est à noter que l'approche inverse peut-être implémentée : stocker tout le long de l'algorithme des valeurs « temporaires » et, durant le bloc « finally », appliquer (en cas de succès) les valeurs temporaires à l'objet. Cette approche, si elle permet de s'affranchir des problématiques de lock entre threads (voir ci-dessus), a l'inconvénient d'empêcher l'appel d'autres méthodes de l'objet durant la transaction (en effet celles-ci verront toujours l'état de l'objet avant la transaction, et pas son état temporaire actuel).

2.2. La fonction par l'exemple

Prenons un cas simple : une fonction « String login(String user, String password) ». La fonction doit recevoir deux éléments non vides (login et password) et fournir une « clé » de validation. Elle n'a aucun effet de bord. La postcondition est donc « fournir une clé d'authentification »… Si a un moment cette fonction est incapable de le faire, elle doit lever une exception. Retourner « null » n'est bien entendu pas une option.
La fonction fera hélas appel à un (mauvais) web service, qui renvoie une réponse HTTP « OK » mais le contenu « ERROR » en cas d'erreur de login, ou un token valide si tout s'est bien passé. Un bon web service renverrait une erreur HTTP pour signaler une erreur de login.
Commençons par la fonction « simple » sans aucune gestion des exceptions :

```
    /**
     * @param username
     * @param password
     * @return The authentication token.
     */
    public String login(String username, String
password)
    {
        HttpClient  client = new
DefaultHttpClient();

        String url =
String.format("https://my.webservice.com/login.ph
p?user=%0&pwd=%1",username,password); //$NON-NLS-
1$

        HttpGet getRequest = new HttpGet(url);
        HttpResponse response =
client.execute(getRequest);
        if
(response.getStatusLine().getStatusCode()>=400)
            return null;
        HttpEntity entity = response.getEntity();
        String token = entity.toString();
        if ("ERROR".equalsIgnoreCase(token))
            return null;

        return token;
    }
```

2.2.1. Gestion des exceptions « existantes »

Déjà, la fonction ne compilera pas… Si vous utilisez Eclipse, celui-ci vous dira immédiatement que l'appel à client.execute() risque d'envoyer les exceptions « ClientProtocolException » et « IOException » qui ne sont pas gérées…
Malheureusement Eclipse va proposer de « générer » un code de gestion de ces exceptions par un simple try/catch autour des appels problématiques. Le résultat étant désastreux puisque par défaut les exceptions seront simplement passées sous silence. Ainsi, sur un « ClientProtocolException », l'appel à « execute » ne va rien renvoyer, et nous allons appeler response.getEntity() sur un objet null (NullPointerException et le programme va se fermer).
De plus, toutes les exceptions ne seront pas gérées de la même façon. Voyons cela en détail.

2.2.1.1. Renvoi des exceptions « utiles » à l'appelant

Que signifient les exceptions levées par le code ?
IOException signifie qu'une exception d'entrée/sortie (le service n'est pas joignable, il s'est produit une erreur TCP/IP, etc.) a eu lieu… Il est évident qu'une telle exception est utile à l'appelant, et devrait donc lui être passée directement.
On va donc simplement rajouter « throws IOException » à la déclaration de la fonction :

```
    /**
     * @param username
     * @param password
     * @return The authentication token.
     * @throws IOException if an error occured
during the transfer
     */
    public String login(String username, String
password) throws IOException
    {
        HttpClient  client = new
DefaultHttpClient();

        String url =
String.format("https://my.webservice.com/login.ph
p?user=%0&pwd=%1",username,password); //$NON-NLS-
1$

        HttpGet getRequest = new HttpGet(url);
```

```
        HttpResponse response =
client.execute(getRequest);
        if
(response.getStatusLine().getStatusCode()>=400)
            return null;
        HttpEntity entity = response.getEntity();
        String token = entity.toString();
        if ("ERROR".equalsIgnoreCase(token))
            return null;

        return token;
    }
```

2.2.1.2. Disparition des exceptions « inutiles »

Dans le cas de ClientProtocolException, nous savons pertinemment qu'elle ne se produira jamais : la fonction maitrise la construction de l'URL, elle ne peut pas renvoyer cette exception… Il va donc falloir la gérer.
Si l'URL avait été passée en paramètre, il aurait fallu bien entendu traduire cette exception (ou la déclarer comme possible) !
Première chose : c'est bien entendu l'ensemble de la fonction qui échoue… n'ayons pas peur, et protégeons l'ensemble de la fonction.

```
    /**
     * @param username
     * @param password
     * @return The authentication token.
     * @throws IOException if an error occured
during the transfer
     */
    public String login(String username, String
password) throws IOException
    {
        try {
            HttpClient  client = new
DefaultHttpClient();

            String url =
String.format("https://my.webservice.com/login.ph
p?user=%0&pwd=%1",username,password); //$NON-NLS-
1$

            HttpGet getRequest = new
HttpGet(url);
            HttpResponse response =
client.execute(getRequest);

            if
(response.getStatusLine().getStatusCode()>=400)
                return null;

            HttpEntity entity =
response.getEntity();
            String token = entity.toString();
            if ("ERROR".equalsIgnoreCase(token))
                return null;

            return token;
        } catch (ClientProtocolException ex) {
            Log.e("LOGIN","HTTPS non
supporté !",ex);
            return null;
        }
    }
```

Voilà, à partir de maintenant le code compile correctement. Mais nous n'en avons pas fini pour autant. Il faut maintenant vérifier les préconditions et postconditions.

2.2.2. Rajout des exceptions manquantes

2.2.2.1. Préconditions

Parmi les préconditions de la fonction, il y a bien sûr le fait de recevoir un « username » et un « password »… nous allons donc simplement le vérifier en utilisant une « RuntimeException ». Les exceptions « runtime » n'ont pas besoin d'être déclarées par la méthode et expriment souvent un défaut dans le code. La plus adaptée dans le cas est bien sûr IllegalArgumentException :

```
    /**
     * @param username
     * @param password
     * @return The authentication token.
     * @throws IOException if an error occured
during the transfer
     */
    public String login(String username, String
password) throws IOException
    {
        if (username == null) throw new
IllegalArgumentException("Username cannot be
null");
        if (password == null) throw new
IllegalArgumentException("Password cannot be
null");

        try {
            HttpClient  client = new
DefaultHttpClient();

            String url =
String.format("https://my.webservice.com/login.ph
p?user=%0&pwd=%1",username,password); //$NON-NLS-
1$

            HttpGet getRequest = new
HttpGet(url);
            HttpResponse response =
client.execute(getRequest);

            if
(response.getStatusLine().getStatusCode()>=400)
                return null;

            HttpEntity entity =
response.getEntity();
            String token = entity.toString();
            if ("ERROR".equalsIgnoreCase(token))
                return null;

            return token;
        } catch (ClientProtocolException ex) {
            Log.wtf("LOGIN","HTTPS non
supporté !",ex);
            return null;
        }
    }
```

Attention, la gestion des préconditions n'est pas obligatoire, mais vous évitera pas mal d'heures de débogage dans le cas de gros projets.

2.2.2.2. Postconditions

C'est là où les exceptions « normales » vont entrer en jeu.

Quand la fonction est incapable de livrer un état correspondant à la postcondition (ici un token d'authentification correct), elle doit lancer une exception.
Si nous lisons la fonction, il y a trois cas où elle sera incapable de délivrer un token : si le status-code de l'appel HTTP est >= 400 (erreur HTTP), si ClientProtocolException a été lancé, ou simplement si le web service a retourné « ERROR ».
Dans le premier cas, cela veut dire qu'une erreur de communication s'est produite… nous traduirons donc ce cas par un lancement de IOException. Le message étant contenu dans le statut HTTP, nous l'utiliserons directement.
Le second cas est une erreur de configuration et sera traduite par une erreur « runtime » IllegalStateException (sans oublier de passer l'exception initiale en « cause » afin de ne rien perdre). À noter que dans ce cas, le log peut être supprimé (puisque l'exception sera traduite et passée à l'appelant, rien ne sera « perdu »).

```
    /**
     * @param username
     * @param password
     * @return The authentication token.
     * @throws IOException if an error occured
during the transfer
     */
    public String login(String username, String
password) throws IOException
    {
        if (username == null) throw new
IllegalArgumentException("Username cannot be
null");
        if (password == null) throw new
IllegalArgumentException("Password cannot be
null");

        try {
            HttpClient  client = new
DefaultHttpClient();

            String url =
String.format("https://my.webservice.com/login.ph
p?user=%0&pwd=%1",username,password); //$NON-NLS-
1$

            HttpGet getRequest = new
HttpGet(url);
            HttpResponse response =
client.execute(getRequest);

            if
(response.getStatusLine().getStatusCode()>=400)
                throw new
IOException(response.getStatusLine().getReasonPhr
ase());

            HttpEntity entity =
response.getEntity();
            String token = entity.toString();
            if ("ERROR".equalsIgnoreCase(token))
                return null;

            return token;
        } catch (ClientProtocolException ex) {
            throw new
IllegalStateException("HTTPS non supporté !",ex);
        }
    }
```

Il ne reste donc plus que le cas du Login failure… Pour ce cas précis, il nous faut un moyen d'indiquer à l'appelant qu'il n'a pas eu son token à cause d'une erreur d'authentification (et non environnementale ou de développement)… Pour cette raison, nous allons créer une exception spécifique : AuthenticationException, déclarée comme pouvant être lancée par la fonction, et que nous n'enverrons que dans ce cas précis :

```
    /**
     * Exception thrown when invalid credentials
were given.
     */
    public static class AuthenticationException
extends Exception {
        /**
         * @param msg
         */
        public AuthenticationException(String
msg) {
            super(msg);
        }
    }

    /**
     * @param username
     * @param password
     * @return The authentication token.
     * @throws IOException
     * @throws AuthenticationException
     */
    public String login(String username, String
password) throws IOException,
AuthenticationException
    {
        try {
            HttpClient  client = new
DefaultHttpClient();

            String url =
String.format("https://my.webservice.com/login.ph
p?user=%0&pwd=%1",username,password); //$NON-NLS-
1$

            HttpGet getRequest = new
HttpGet(url);
            HttpResponse response =
client.execute(getRequest);
            if
(response.getStatusLine().getStatusCode()>=400)
                throw new
IOException(response.getStatusLine().getReasonPhr
ase());
            HttpEntity entity =
response.getEntity();
            String token = entity.toString();
            if ("ERROR".equalsIgnoreCase(token))
                throw new
AuthenticationException("Invalid
username/password");
            return token;
        } catch (ClientProtocolException ex) {
            throw new
IllegalStateException("HTTPS non supporté !",
ex);
        }
    }
```

Et voilà, notre fonction est maintenant finie, protégée contre les mauvais appels, les erreurs éventuelles, et surtout sans jamais aucune perte d'information.

Il reste encore du travail, notamment quand le passage des paramètres est erroné (un mot de passe avec « & » ou « ? » risque de faire planter la requête), mais en tout cas aucune exception ne sera oubliée.

3. Au niveau du « framework »

L'appelant d'une fonction utilisant les mêmes principes, les exceptions vont donc « naturellement » remonter jusqu'à un niveau ou vous ne pourrez plus rien changer… parce que la fonction hérite d'une fonction prédéfinie (au pire le « main » du programme Java).
C'est le cas en particulier quand vous programmez dans un « framework » (comme Android). Et pour cause, que signifierait que « onCreate » n'a pas fonctionné ? Que l'on doit quitter l'application ?… et que « onClick » n'a pas fonctionné ? Que l'on doit faire comme si l'utilisateur n'avait pas cliqué ?
Ces fonctions ne peuvent donc pas renvoyer d'exception. Et si c'est le cas, l'application sera immédiatement interrompue (application xxxx has stopped…).
Mais ce ne sont pas que les seuls exemples, nous trouverons de telles restrictions dans les tâches de background, dans les threads… Regardons les cas possibles :

3.1. Le thread UI

Dans les cas de l'UI, rien n'est changé quant à la gestion des sous-fonctions… un bloc try/catch englobant sera souvent utilisé… Par contre, impossible ici de renvoyer l'exception (ou une autre), il va nous falloir simplement « interrompre » l'utilisateur, et lui dire pourquoi.
Il y a plusieurs moyens d'y parvenir : utilisation d'un « Toast », d'une « Boite de dialogue », d'une TextView…
Libre à vous de choisir la meilleure implémentation, mais dans tous les cas, ce devra être fait. Il est très ennuyeux de ne pas réaliser l'action demandée par l'utilisateur et de ne pas lui dire pourquoi.
Bien entendu, le message à l'utilisateur doit être simple, concis, explicite, sans obligatoirement fournir tous les détails. C'est pour cette raison qu'il est impératif de « loguer » l'exception entière !
Ainsi notre fonction login (hormis le fait qu'elle enverrait un « NetworkOnMainThreadException » dans ce cas-là) pourrait être utilisée comme suit :

```
public void onClick(View v)
{
    String username =
usernameEdit.getText().toString();
    String password =
passwordEdit.getText().toString();
    try {
        this.authToken =
login(username,password);
    } catch (IOException ex) {
        Log.w("LOGIN","Login network
error !",ex);

Toast.makeToast(this,R.string.networkError,Toast.
TOAST_LONG).show();
    } catch (AuthenticationException ex) {
        Log.i("LOGIN","Login failure !",ex);

Toast.makeToast(this,R.string.invalidAuthenticati
on,Toast.TOAST_LONG).show();
    }
}
```

Comme on le voit, les deux exceptions ne produiront pas le même message (localisé) pour l'utilisateur, et dans les deux cas un « log » sera produit, avec l'exception entière, une simple information pour l'authentification non valide (erreur « normale »), et un warning dans l'autre (erreur « anormale » n'empêchant pas toutefois le programme de marcher).

3.2. L'AsyncTask

Le cas de l'AsyncTask est relativement simple, puisqu'il fournit déjà le framework pour « repasser » dans le mode UI (et donc afficher un texte à l'utilisateur).
On verra donc un code dans le style :

```
class LoginTask extends
AsyncTask<LoginData,Void,String>
{
    private Exception error;
    protected String doInBackground(LoginData …
data) {
        try {
            return
login(data[0].getUserName(),data[0].getPassword()
);
        } catch (Exception ex) {
            Log.w("LOGINTASK","Failed to
login",ex);
            this.error = ex;
            return null;
        }
    }

    public void onPostExecute(String token) {
        if (this.error == null) {
            // authentication success !
        } else {
            // authentication failed ! (error in
this.error)
        }
    }
}
```

3.3. Les threads & Runnable

Tous les « Runnable » doivent voir leur fonction « run() » entièrement protégée. Un manquement à cette règle provoquera une mort prématurée du thread, sans que quiconque soit au courant.
Comme dans le cas des AsyncTask, il vous faudra certainement stocker l'erreur en tant qu'état du thread, afin de pouvoir récupérer l'erreur et la propager par la suite (à l'utilisateur peut-être).
Par contre le passage de cette information au gestionnaire du Runnable risque d'être plus ardu (messages par Handlers, etc.).

Retrouvez la suite de l'article de Nicolas Romantzoff en ligne : Lien 43

Le webdesign selon les grilles

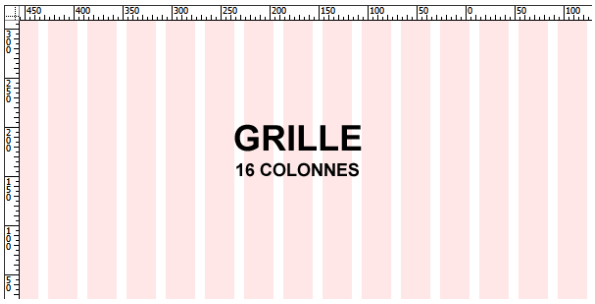
Tout bon webdesigner sait que l'esprit créatif ne suffit pas à rendre son site professionnel. L'une des bonnes pratiques de base du webdesign est la conception d'un template structuré et celle qui s'apparente le mieux à cela est la grille. Cette méthode de travail pourtant répandue dans le domaine de l'architecture et de la typographie est tout simplement mal connue dans le webdesign.

Le principal souci est que la plupart des webdesigners pensent que la conception d'une grille est une tâche ennuyeuse réservée aux mises en page à largeur fixe. Elle peut certainement l'être dans certains cas. Cela est particulièrement vrai si vous l'utilisez comme un formulaire où chaque case doit être complétée avec des données…

1. Qu'est-ce qu'une grille ?

Une grille est par définition un « quadrillage » composé de repères :

- les colonnes, véritable armature, servent à **positionner les éléments à l'horizontale** ;
- les gouttières représentent les **espaces entre chaque colonne** ;
- les marges sont aussi appelées « blanc tournant ». Surtout utilisées en PAO dans le domaine du prépresse, les marges servent à **aérer le texte**, il en est de même pour les marges d'une grille ;
- les lignes (axes horizontaux), moins récurrentes mais tout aussi importantes, servent à **positionner les éléments à la verticale**.

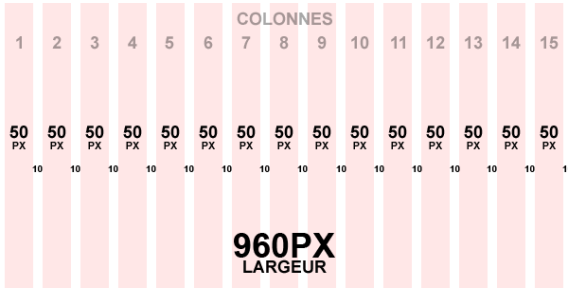


2. Pourquoi utiliser une grille ?

Une grille a pour fonction de **structurer un site Internet**. Elle représente l'armature générale sur laquelle s'organise votre page Web et le positionnement de votre contenu est logiquement influencé par cette pratique favorisant ainsi sérieusement votre intégration CSS en la rendant plus rapide. Elle permet entre autres d'y voir plus clair lors de la conception d'une maquette.
Les éléments qui la composent sont parfaitement alignés et ce de façon proportionnelle. Tout est plus harmonieux et l'œil humain aime ça.
Pour finir, la grille se veut également utile dans l'adaptation de la structure d'un site Internet sur smartphone et tablette *via* les **points de rupture** qui améliorent sa lisibilité. Peu importe l'appareil utilisé, vous n'aurez aucun mal à anticiper le comportement de ces paliers d'adaptation vous garantissant ainsi un affichage optimal sur la grande majorité des résolutions (320px, 640px, 768px, 1280px…).

3. Les dimensions d'une grille ?

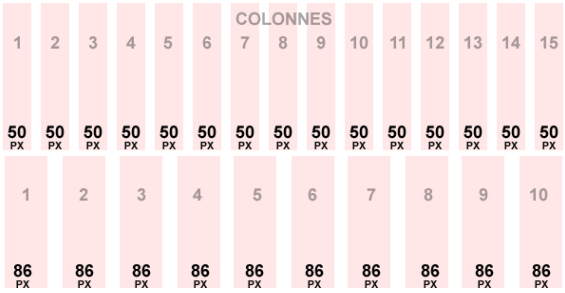
Certains standards existent, avec par exemple la valeur de 960px de largeur qui revient couramment pour s'adapter aux petites résolutions. N'étant pas une règle absolue, il est possible de prendre comme exemple le phénomène « *responsive* » qui s'accroit de jour en jour dû aux supports numériques dont le nombre augmente de façon fulgurante. Certaines estimations placent à 40 % la part du marché mobile en 2016. Ce fait, certains sites l'ont pris très au sérieux, ils utilisent donc des grilles adaptées.
Le nombre de colonnes qui composent une grille est souvent de 12, 16 ou encore 24, mais encore une fois, rien ne vous oblige à suivre cette tendance. Certains de vos projets forts en contenu nécessiteront sûrement plus de 24 colonnes avec des dimensions personnalisées, ceci s'applique également aux lignes, aux gouttières et aux marges.



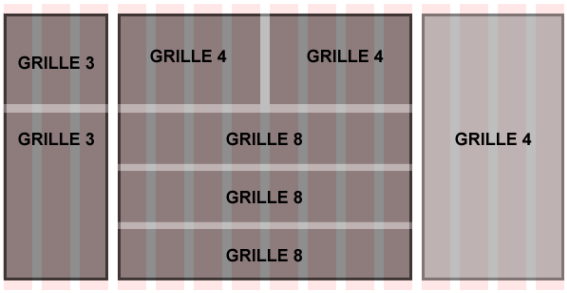
4. Quelle grille choisir ?

Le monde d'aujourd'hui étant source de disparités technologiques, il en résulte une diversité d'affichages écran allant du simple au triple… Vous comprendrez qu'il est difficile dans un cas comme celui-là de vous conseiller une grille en particulier. Il vous faudra par conséquent adapter cette dernière par rapport à votre projet.
Nous pouvons penser de la manière suivante : plus le site a de contenu textuel et imagé, plus le nombre de colonnes est important. À l'inverse, moins il y a de contenu, moins il y a de colonnes. C'est un principe de base.
Notez également que le nombre de colonnes impacte

directement leur largeur respective. En effet, pour une grille composée de 12 colonnes, leurs largeurs sont bien supérieures à celles d'une grille composée de 24 colonnes (mathématiquement deux fois plus large sauf modification éventuelle des gouttières).



Dans le cas où vous avez une idée concrète de l'architecture qu'aura votre site, construisez ou téléchargez votre **grille sur *Photoshop*** et amusez-vous à placer vos premiers blocs comme vous l'imaginez en tenant compte du quadrillage.
Si à l'inverse vous semblez être victime du syndrome de la « page blanche », amusez-vous à placer toutes sortes de grilles. Quelquefois, le simple fait d'avoir sa grille positionnée devant soi suffit à stimuler votre créativité, vous commencerez à placer un bloc, puis un autre… Au bout du compte, vous trouverez votre armature plus facilement qu'avec un document vierge sur fond blanc.



5. Des outils sur le Web pour générer vos grilles

Pour les plus fainéants, il est possible de **générer des grilles en ligne** très facilement. Plusieurs sites proposent cet outil, nous en retiendrons cependant quatre qui sont selon moi les meilleurs.

5.1. ModularGrid.org

Vous entrez les valeurs qui vous chantent et vous téléchargez votre grille sur *Photoshop*. Trois choix de grilles vous sont proposés :

- PNG ;
- motif ;
- masque.

Vous choisissez celle qui convient au mieux, selon vos habitudes.

Vous pouvez également l'utiliser en tant qu'extension *Photoshop* : Lien 44.

Pour tester l'outil, rendez-vous sur ModularGrid : Lien 45.

5.2. Guideguide.me

Puisque nous parlons de *Photoshop*, il existe une extension appelée **GuideGuide** (compatible CS5 & CS6) permettant de générer des grilles à l'aide des repères (CTRL + : pour les afficher). Depuis le panneau *GuideGuide*, vous pouvez contrôler gouttières, lignes, colonnes, marges… Vous y trouverez d'autres fonctionnalités intéressantes que nous vous laissons découvrir sur le site officiel : Lien 46.

5.3. 960.gs

Un site complet qui vous propose un *framework* CSS simple mais sacrément efficace. Vous générez simplement votre grille avec les valeurs de votre choix et exportez votre code en CSS. Des grilles par défaut sont également téléchargeables où chaque fichier CSS contient une grille.
Chaque colonne est définie *via* une et chaque div est en float. Les gouttières quant à elles sont définies par un simple margin-right, annulables par une surcharge avec les classes .alpha et .omega.
Pour tester l'outil, rendez-vous sur 960.gs : Lien 47

5.4. 978.gs

Il s'agit là d'une alternative au site précédent.
Concrètement qu'est-ce qui change ?
Déjà ce site est plus intuitif (Lien 48), plus convivial… Mais la véritable différence réside dans le fait que la grille **978.gs** est légèrement plus large que celle proposée par 960.gs (18 pixels de différence).
Pourquoi choisir la grille 978px plutôt que la grille 960px ?
Cette nouvelle grille est issue d'un constat mettant en cause la largeur des gouttières de la grille 960 qui rendrait le montage des blocs difficile. Ce nouveau « standard » aérerait davantage votre structure en intégrant des gouttières plus larges.
Pour tester l'outil, rendez-vous sur 978.gs : Lien 49.

6. Quelques exemples

Pour que vous perceviez mieux l'intérêt de cette **technique de webdesign**, voici quelques exemples de sites Internet dont les structures se basent sur des grilles :

Le site de Netmacom utilise le principe de grille. Il suffit d'observer la structure globale pour s'en rendre compte.

Le célèbre site marchand de produits High-tech, Materiel.net, connu pour la qualité de ses services, utilise également un système de grille sur son site Internet. La quantité de texte est telle qu'il aurait été très délicat d'outrepasser cette règle de bonne pratique.
La page d'accueil de 960.gs expose aussi différents exemples de sites Internet basés sur des grilles plus ou moins différentes : Lien 47.

7. Conclusion

Outre ses vertus bienfaisantes pour vos petits yeux, la grille vous permet finalement de gagner un temps précieux et n'est en aucun cas un obstacle à votre créativité. Bien au contraire, elle vous ouvre la porte à une **analyse de l'espace** plus précise et incontestablement plus facile ! De cette manière vous placez correctement vos blocs même si au départ vous ne saviez pas vers quelle structure vous tourner.
Souvenez-vous qu'une grille n'est pas qu'un moyen de remplir l'espace dont vous disposez, elle représente l'armature qui va vous aider à **organiser** cet espace.
Une fois que vous aurez saisi tout l'intérêt d'appliquer une telle méthodologie et que vous comprendrez son fonctionnement, vous pourrez alors vous tourner vers des constructions de grilles moins rigides pour créer des sites Internet plus élégants qui ne laisseront pas transparaitre au premier coup d'œil la méthode avec laquelle ils ont été conçus.
Un des critères importants sur lequel on évalue la qualité d'un site Internet se concentre sur cette méthode de travail. Autrement dit, c'est ce qui peut faire la différence entre un **site professionnel** et un site amateur.

Retrouvez l'article de Kévin Pinto en ligne : Lien 50

Introduction à RubyMotion

RubyMotion est un framework permettant d'écrire des applications iOS et OS X natives en Ruby. Le code Ruby est compilé exactement de la même façon que le serait du code Objective-C. Toutes les API Cocoa sont donc disponibles nativement. Le code généré est conforme aux règles de l'Apple Store et n'a pas besoin d'embarquer un interpréteur.

RubyMotion est une évolution du projet MacRuby, initié par Laurent Sansonetti. MacRuby est un projet open source qui a été sponsorisé par Apple et qui a pour but d'écrire des applications natives pour OS X en Ruby. Laurent, via sa société HipByte, a fait évoluer ce projet dans un premier temps pour permettre le développement d'applications iOS natives en Ruby. Depuis sa version 2.0, il permet également d'écrire des applications OS X.

RubyMotion n'est toutefois pas un projet open source, il vous faudra une licence pour pouvoir obtenir le kit de développement, mais également avoir accès à un support ultraréactif. Il est à noter que les mises à jour sont gratuites.

Les autres intérêts majeurs de RubyMotion sont la gestion automatisée de la mémoire et surtout le REPL qui est en fait une console interactive IRB enrichie permettant de déboguer en live une application lancée dans le simulateur. On pourra par exemple cliquer sur un élément pour analyser ses propriétés, les modifier ou encore ajouter des vues à la volée. On a donc un framework nous permettant d'utiliser Ruby tout en conservant des performances dignes des applications écrites en Objective-C avec en prime un environnement de développement léger et puissant.

Cet article va tenter de vous présenter les bases de RubyMotion. Il est le premier d'une série qui sera dédiée à l'écriture d'applications iOS en Ruby.

1. Création d'une application iOS

RubyMotion est livré avec un ensemble d'outils simplifiant la gestion d'un projet, sa compilation, son déploiement sur un *iDevice* ou encore son débogage. On a donc une commande motion qui permet notamment de créer le squelette d'un projet, de mettre à jour RubyMotion ou

encore de créer un ticket de support.
Au sein du projet, un Rakefile est à notre disposition nous permettant de :

- compiler le projet pour le simulateur ou un appareil ;
- créer une archive .ipa pour distribution ;
- obtenir les variables de configuration du projet ;
- générer les ctags pour faciliter la navigation dans l'éditeur de texte ;
- lancer le simulateur ;
- lancer les batteries de tests ;
- compiler le code sous forme de bibliothèque réutilisable.

Créons notre premier projet de test :

```
$ motion create HelloWorld
Create HelloWorld
Create HelloWorld/app/app_delegate.rb
Create HelloWorld/Rakefile
Create HelloWorld/resources/Default-568h@2x.png
Create HelloWorld/spec/main_spec.rb
```

Nous pouvons donc aller dans le répertoire nouvellement créé et vérifier la configuration par défaut

```
$ cd HelloWorld
rake config
background_modes       : []
build_dir              : "./build"
codesign_certificate   : "iPhone Developer:
Nicolas Cavigneaux (8KBAAAKF4)"
delegate_class         : "AppDelegate"
deployment_target      : "6.1"
device_family          : :iphone
entitlements           : {}
files                  :
["./app/app_delegate.rb"]
fonts                  : []
framework_search_paths : []
frameworks             : ["UIKit", "Foundation",
"CoreGraphics"]
icons                  : []
identifier             :
"com.yourcompany.HelloWorld"
interface_orientations : [:portrait,
:landscape_left, :landscape_right]
libs                   : []
motiondir              : "/Library/RubyMotion"
name                   : "HelloWorld"
prerendered_icon       : false
provisioning_profile   :
"/Users/nico/Library/MobileDevice/Provisioning
Profiles/338B4DF5-111F-498F-BD4A-
AAAAAAAAAB.mobileprovision"
resources_dirs         : ["./resources"]
sdk_version            : "6.1"
seed_id                : "8ZBABAA36Y"
short_version          : "1"
specs_dir              : "./spec"
status_bar_style       : :default
version                : "1.0"
weak_frameworks        : []
xcode_dir              :
"/Applications/Xcode.app/Contents/Developer"
```

On voit donc le certificat de signature (nécessaire au déploiement sur un *iDevice*) utilisé, le type de cible (ici « iphone » et iOS 6.1 minimum), les *frameworks* Cocoa chargés, l'icône qui sera utilisée pour présenter l'application, l'identifiant de l'application, les orientations supportées, le nom de l'application, le répertoire de ressources (images, sons…) ou encore le numéro de version.

1.1. Structure d'un projet

On peut maintenant ouvrir le projet dans un éditeur pour en apprendre plus. Commençons par le Rakefile :

```
# -*- coding: utf-8 -*-
$:.unshift("/Library/RubyMotion/lib")
require 'motion/project/template/ios'

Motion::Project::App.setup do |app|
# Use `rake config' to see complete project
settings.
app.name = 'HelloWorld'
end
```

C'est à cet endroit qu'est défini le type de *template* à utiliser, ici « ios » qui détermine le *framework* et la cible du projet. On voit également que le nom de l'application est défini dans le bloc setup. On peut donc y définir des valeurs personnalisées pour chaque variable de configuration disponible.
Il y a ensuite le fichier app/app_delegate.rb qui est le point d'entrée de l'application :

```
class AppDelegate
  def application(application,
didFinishLaunchingWithOptions:launchOptions)
    true
  end
end
```

Par défaut, l'application ne fera qu'afficher un écran noir. C'est dans la méthode application que nous pouvons instancier notre fenêtre principale soit via du code Ruby, soit en chargeant un fichier InterfaceBuilder (.xib).
Vous pouvez donc compiler et lancer l'application dans le simulateur :

```
$ rake
   Build ./build/iPhoneSimulator-6.1-Development
 Compile ./app/app_delegate.rb
  Create ./build/iPhoneSimulator-6.1-
Development/HelloWorld.app
    Link ./build/iPhoneSimulator-6.1-
Development/HelloWorld.app/HelloWorld
  Create ./build/iPhoneSimulator-6.1-
Development/HelloWorld.app/Info.plist
  Create ./build/iPhoneSimulator-6.1-
Development/HelloWorld.app/PkgInfo
    Copy ./resources/Default-568h@2x.png
  Create ./build/iPhoneSimulator-6.1-
Development/HelloWorld.dSYM
Simulate ./build/iPhoneSimulator-6.1-
Development/HelloWorld.app
(main)>
```

Un binaire est généré dans le répertoire build puis le simulateur lance l'application. On obtient notre fameux écran noir dans le simulateur.
Le répertoire resources quant à lui ne contient qu'une image noire par défaut. Le répertoire spec contient lui un

fichier de test d'exemple :

```
describe "Application 'HelloWorld'" do
  before do
    @app = UIApplication.sharedApplication
  end

  it "has one window" do
    @app.windows.size.should == 1
  end
end
```

Les tests utilisent la bibliothèque Bacon (Lien 51) qui est un clone allégé de RSpec. Ils permettent de tester facilement, comme d'habitude en Ruby, les différents aspects de l'application que ce soit les vues, les contrôleurs ou encore les modèles.
Vous pouvez lancer les tests via rake spec, l'unique test par défaut ne passera pas. Il vérifie que l'application a bien une fenêtre (UIWindow) ce qui n'est pas le cas de notre application actuelle. Nous verrons plus en détail l'écriture des tests dans un prochain article.
Notez que vous pouvez créer de nouveaux répertoires dans app/ pour organiser votre code. Il sera donc courant dans les projets d'avoir les répertoires :

- app/views : vues personnalisées (ex. : un TableView amélioré par vos soins) ;
- app/models : classes définissant des objets métier ;
- app/controllers : comme en Rails, ce sont les aiguilleurs qui permettent aux vues de communiquer avec les modèles ou les services externes.

2. REPL : Console interactive

Vous l'aurez peut-être remarqué en lançant la tâche rake, une fois l'application lancée dans le simulateur, le terminal nous rend la main dans une console interactive IRB qui nous permet d'inspecter et de manipuler l'application en cours d'exécution :

```
(main)> p self
main
=> main
(main)> alert = UIAlertView.new
=> #<UIAlertView:0x96735f0>
(main)> alert.message = "Hello World!"
=> "Hello World!"
(main)> alert.show
=> #<UIAlertView:0x96735f0>
```

Nous venons, depuis la console interactive, de créer et d'instancier une modale contenant le texte « Hello World » puis nous l'avons affichée. Elle est donc visible dans le simulateur !
Un outil dont vous ne pourrez plus vous séparer tellement il est pratique. Il n'existe aucun équivalent en Objective-C ou dans XCode, cet outil est une merveille.
Si vous faites ⌘-clic sur la modale, vous verrez que self change pour devenir #<UITextEffectsWindow:0x9437340>. On peut de cette manière inspecter n'importe quel élément très facilement.

3. Création d'une fenêtre

Nous allons maintenant ajouter à notre application une fenêtre principale pour y afficher notre « Hello World ». Pour cela nous allons retourner dans le fichier app/app_delegate.rb pour instancier la fenêtre et l'associer à un contrôleur :

app/app_delegate.rb
```
class AppDelegate

  def application(application,
didFinishLaunchingWithOptions:launchOptions)
    @window =
UIWindow.alloc.initWithFrame(UIScreen.mainScreen.
bounds)
    @window.rootViewController =
MainViewController.alloc.init
    @window.makeKeyAndVisible
    true
  end

end
```

Tout d'abord, nous créons une UIWindow en initialisant sa taille à celle de l'écran.
Une fois cette fenêtre créée, nous définissons le contrôleur qui lui est associé et qui servira à régir son comportement. Nous allons ici utiliser le contrôleur MainViewController que nous allons écrire par la suite.
Une fois la fenêtre créée, nous la marquons comme étant la fenêtre principale qui sera donc chargée automatiquement au lancement de l'application.
Pour finir, nous retournons true, ce qui est nécessaire au lancement de l'application.
Voyons maintenant le contrôleur !

app/controllers/main_view_controller.rb
```
class MainViewController < UIViewController

  def viewDidLoad
    view.backgroundColor =
UIColor.scrollViewTexturedBackgroundColor
  end

end
```

Nous redéfinissons ici la méthode viewDidLoad qui est appelée automatiquement dès que la vue associée au contrôleur est chargée. Ce que nous faisons dans cette méthode est tout simple, en effet on ne fait que changer le fond de la fenêtre. Par défaut celle-ci est noire, elle utilisera une texture grise disponible par défaut sous iOS.

UIWindow avec texture de fond

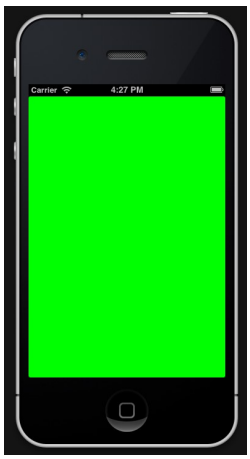
Nous aurions pu utiliser une couleur pleine :

```
class MainViewController < UIViewController

  def viewDidLoad
    view.backgroundColor = UIColor.greenColor
  end

end
```

Nous aurions dans ce cas un fond vert :



UIWindow avec couleur de fond

3.1. Utiliser une image de fond

Il est possible d'utiliser une image en background. Il faut tout d'abord mettre à disposition cette image dans le répertoire resources. Dans notre exemple, ce fichier s'appelle customBackground.jpg :

```
class MainViewController < UIViewController

  def viewDidLoad
    background_view =
UIImageView.alloc.initWithFrame(UIScreen.mainScre
en.bounds)
    background_view.image =
UIImage.imageNamed("customBackground.jpg")
    self.view.addSubview(background_view)
  end

end
```

UIWindow avec image de fond personnalisée

Nous commençons donc par instancier une vue pour l'image via une UIImageView que l'on cale sur la taille de l'écran. On instancie ensuite l'image en elle-même (UIImage) en utilisant son nom de fichier. Il ne nous reste finalement plus qu'à ajouter cette nouvelle vue à la vue principale.
Nous aurions pu utiliser loadView plutôt que viewDidLoad pour que l'image soit chargée avant même que la vue n'apparaisse, nous verrons cela dans l'exemple suivant.

4. Ajout d'un label

Nous allons maintenant ajouter deux labels qui permettront d'afficher « Hello World » ainsi que l'heure au chargement de la vue. Commençons par le plus simple à savoir le label « Hello World ».

4.1. Label « Hello World »

```
class MainViewController < UIViewController

  def loadView
    self.view = UIView.alloc.init

    background_view =
UIImageView.alloc.initWithFrame(UIScreen.mainScre
en.bounds)
    background_view.image =
UIImage.imageNamed("customBackground.jpg")
    self.view.addSubview(background_view)

    helloLabel = UILabel.alloc.initWithFrame
[[110, 30], [200, 50]]
    helloLabel.backgroundColor =
UIColor.clearColor
    helloLabel.textColor = UIColor.whiteColor
    helloLabel.text = "Hello World!"

    self.view.addSubview(helloLabel)
  end

end
```

Nous utilisons loadView pour préparer le style et les éléments de la vue, nous devons donc instancier nous-mêmes la vue associée au contrôleur. loadView se charge normalement de ça, mais puisque nous écrasons la méthode par défaut, il faut penser à le faire manuellement.
On définit le background comme précédemment. Vient ensuite la création du label, on commence donc par instancier le label (UILabel) en définissant sa taille. On utilise d'ailleurs ici un sucre syntaxique de RubyMotion. Notre [[110, 30], [200, 50]] correspond en fait à un CGMakeRect(110, 30, 200, 50). Le premier nombre correspond à la position en X, le deuxième en Y, le troisième à la largeur du label et le dernier à sa hauteur.
On s'assure que le fond du label soit transparent et que le texte soit en blanc. Finalement on définit le texte du label, puis on l'ajoute à la vue principale.

UIWindow avec label

4.2. Label heure courante

```
class MainViewController < UIViewController

  def loadView
    self.view = UIView.alloc.init

    background_view =
UIImageView.alloc.initWithFrame(UIScreen.mainScre
en.bounds)
    background_view.image =
UIImage.imageNamed("customBackground.jpg")
    self.view.addSubview(background_view)

    helloLabel = UILabel.alloc.initWithFrame
[[110, 20], [200, 50]]
    helloLabel.backgroundColor =
UIColor.clearColor
    helloLabel.textColor = UIColor.whiteColor
    helloLabel.text = "Hello World!"

    self.view.addSubview(helloLabel)

    timeLabel = UILabel.alloc.initWithFrame [[90,
60], [200, 50]]
    timeLabel.backgroundColor =
UIColor.clearColor
    timeLabel.textColor = UIColor.whiteColor

    # Initialisation d'un calendrier à la date /
heure actuelles
    calendar =
NSCalendar.alloc.initWithCalendarIdentifier(NSGre
gorianCalendar)
    date = NSDate.date
    calendar.components(NSMinuteCalendarUnit,
fromDate: date)

    # Utilisation de la locale fr_FR pour
formater les dates
    fr_FR =
NSLocale.alloc.initWithLocaleIdentifier "fr_FR"

    format = NSDateFormatter.alloc.init
    format.locale = fr_FR
    format.setDateFormat("dd MMM yyyy - HH:mm")

    # Conversion de la date en chaine
    dateString = format.stringFromDate(date)

    timeLabel.text = dateString

    self.view.addSubview(timeLabel)
  end

end
```

UIWindow avec date

La première partie est identique à l'exemple précédent.
Nous initialisons ensuite un calendrier (au format grégorien) puis nous récupérons la date courante. On définit la précision souhaitée pour la décomposition de la date puis on force la locale française pour obtenir des dates en français.
Une fois cela fait on peut définir le format de date souhaité en sortie. Il ne nous reste plus qu'à générer la chaine puis à l'utiliser dans le label. Finalement le label est ajouté à la vue principale.

5. Conclusion

En quelques lignes de code, nous avons une application fonctionnelle. Certes elle n'est pas très utile, mais démontre la facilité d'écriture d'une application iOS (UI incluse) en RubyMotion. Nous aurions pu construire l'UI dans l'interface builder et l'importer dans le code ce qui nous aurait encore épargné des lignes de code mais nous verrons cela dans un article à venir.
Dans le prochain article sur RubyMotion, nous verrons comment gérer un formulaire basique, récupérer et traiter les entrées utilisateur.
Vous trouverez le code de cet article sur GitHub : Lien 52

Retrouvez l'article de Synbioz en ligne : Lien 53

Écrire mon premier article avec WordPress

Vous débutez avec WordPress ? Voici un article parfait pour vous. Comment créer un article avec WordPress ?

Nous parlons ici d'une fonctionnalité de base de WordPress, celle d'ajouter du contenu à votre site internet, autant maitriser la question tout de suite.

Écrire son premier article n'étant pas juste écrire des mots dans une fenêtre, nous verrons en détail dans ce guide du débutant WordPress tout ce qu'il faut savoir pour bien faire les choses dès la première fois.

Le mot de guide n'étant pas utilisé au hasard, le contenu de cette page est complet. N'hésitez pas à mettre cet article dans vos favoris pour y revenir ultérieurement.

Nous traiterons dans ce guide d'utilisation la différence entre pages et articles, ce qu'il faut installer avant d'écrire son premier article et comment l'écrire, ce fameux premier article avec WordPress.

1. Articles, pages, quelle différence ?

Lorsqu'on parle de contenu avec WordPress on pense tout de suite à un billet de blog, mais WordPress vous propose aussi de créer des pages. Mais quelle est la différence ?
Aucune, tous les deux sont une manière de créer du contenu pour votre site. Mais s'il y a deux types, c'est qu'au fond, il doit bien y avoir une différence.
La différence se situe plus au niveau du traitement par **WordPress** que de la manière dont vous allez devoir créer les choses. Un article s'insérera dans votre fil de blog, alors qu'une page est… une page. Cette dernière sera alors détachée de la chronologie qu'impose un blog. Couramment, on crée une page pour tous les éléments fixes du site comme :

- une page de contact ;
- une page *À propos* ;
- une page d'histoire de votre entreprise ;
- une page d'accueil…

Mais pas que ! Parfois, il est intéressant de se poser la question. Vais-je faire un *billet* ou un *guide* qui sera une page pivot de mon site ? Un exemple : notre page de curation de la newsletter laredoute.fr (Lien 54). Vous le voyez, la limite est finalement plutôt mince.
Avec l'arrivée des *Customs Posts Types*, on pourrait même être tenté de dire les pages et articles ne sont que des *Customs Posts Types*. Peut-être que maintenant vous vous dites : mais les *Customs Posts Types* c'est quoi.
Disons que c'est une manière d'agencer du contenu en fonction d'un type. Sur notre site, nous utilisons les *Customs Posts Types* pour présenter les livres : Lien 55. Cela nous permet de gérer une note, un titre, un ISBN, des auteurs… Un article classique pourrait faire l'affaire, mais avec un *Custom Post Types*, c'est une manière plus fine et dédiée de gérer les choses.
Cet article, car nous sommes ici sur la partie blog, se concentrera donc sur la partie création d'articles, mais pourra facilement servir de base pour la création d'une page.

2. Avant de créer un article avec WordPress…

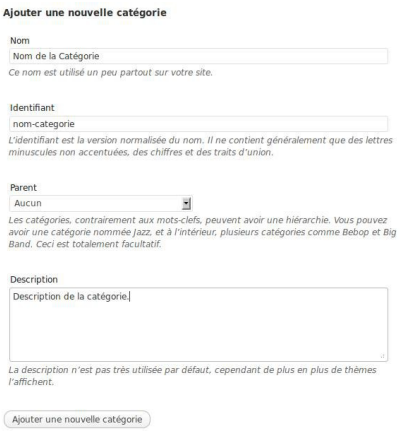
C'est la toute première fois que vous écrivez un article sous WordPress, alors la première des choses à faire est de créer… au moins une catégorie.

2.1. … créer une catégorie avec WordPress …

Si vous ne voulez pas que la catégorie utilisée par défaut soit *Non Classé*, alors il va déjà falloir en créer une autre.
Pour ça, rendez-vous dans votre interface d'administration *-> Articles -> Catégorie*.

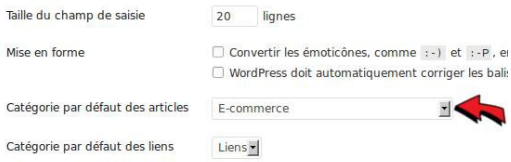
Pour créer une catégorie, il faut :

- lui donner un nom. Pensez à vos mots-clés, sans faire trop long ;
- un identifiant. Cette partie apparaitra dans l'URL. Pas d'accent ni d'espace… bien qu'on puisse faire des URL très originales avec WordPress ;
- une catégorie parente. Il ne s'agit pas d'une obligation, votre première catégorie n'ayant aucun parent par définition ;
- une description. Celle-ci pourra être utilisée ou non par votre thème. Il ne s'agit pas de la balise meta description.



Une fois toutes les infos renseignées, il vous suffit de cliquer sur *Ajouter une nouvelle catégorie* et le tour est joué.
Vous souhaitez que la catégorie que vous venez de créer soit celle utilisée par défaut lors de la création d'un article. Cela peut être très utile aussi bien pour les distraits que pour ceux utilisant très souvent la même catégorie.
Voici la démarche à suivre. Toujours à partir de votre interface d'administration WordPress, allez dans *Réglages -> Écriture*.

Dans *Catégorie par défaut des articles*, choisissez votre catégorie par défaut.



2.2. … installer un plugin pour le référencement …

Le **référencement WordPress** vous importe ? (Lien 56)
Alors installez le *plugin* **SEO WordPress by Yoast** (Lien 57), un excellent **plugin WordPress** (Lien 58). Dans cet article, nous partirons du principe que le référencement, c'est important, alors vous aurez installé ce *plugin*. Vous vous demandez **comment installer un plugin WordPress ?** (Lien 59)

2.3. … installer TinyMCE Advanced …

L'éditeur de base de WordPress, TinyMCE est un éditeur de type WYSIWYG. Il est bien fait, mais nous lui préférons sa version dopée, TinyMCE Advanced (Lien 60) qui vous permettra d'avoir accès à des options avancées comme les ancres, les tableaux, la taille de police de caractères…

2.4. … configurer l'écran de saisie.

L'écran qui vous est proposé par WordPress dépend de deux éléments :

- des *plugins* que vous aurez installés. Tous n'auront pas une incidence sur cette partie, mais certains en auront une énorme, comme WordPress SEO ;
- des *options d'écran* que vous aurez paramétrées.

Les *options d'écran*, ça ne vous dit rien ? N'avez-vous jamais vu cet élément en haut à droite de votre interface de création WordPress ?

Pour en savoir plus sur les options des écrans WordPress, lisez l'article **7 options cachées dans les pages et articles** WordPress : Lien 61.

2.4.1. Zone de titre

La première boite de dialogue ne devrait poser aucun souci. Il s'agit ici de donner un titre à votre article WordPress. Il n'y a pas de règle pour écrire un titre. Cependant, je vous conseille de ne pas faire trop long, d'être clair pour vos lecteurs - ou peut-être un peu énigmatique - et de ne pas hésiter à travailler un peu vos mots-clés. Pour ces derniers, allez-y doucement. Un ou deux groupes de mots-clés suffisent largement. Après, cela n'aurait plus d'effet positif.
Notez qu'il sera possible de modifier l'URL seulement quand l'article sera sauvegardé une première fois.

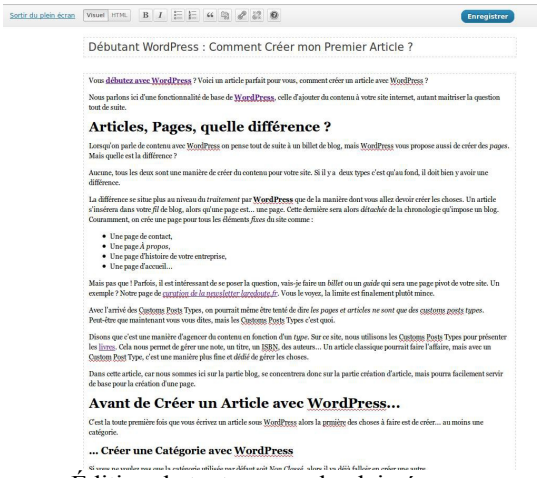
2.4.2. Zone de contenu

La *zone de contenu* est le champ qui vous permet d'insérer… du contenu dans vos articles WordPress. Il s'agit du champ ayant le plus de propriétés à gérer. Son utilisation est plutôt intuitive, vous écrivez du contenu. Mais un petit tour ne serait pas de trop pour cerner cette zone.
La première chose que vous voyez, ce sont les éléments de mise en page à votre disposition. L'éditeur de texte étant WYSIWYG, vous verrez des éléments qui vous rappelleront quelque peu un éditeur comme Microsoft Word. En bien plus simple.
Si vous ne comprenez pas à quoi servent ces icônes, testez-les, vous ne casserez rien.

Une petite icône mérite notre attention, car il s'agit d'une option nouvellement introduite, l'écriture *zen*. L'éditeur *zen* est un éditeur pleine page, sans fioritures, pour vous permettre de rester concentré sur l'écriture de votre contenu. La mise en page interviendra plus tard.
Pour activer ce mode, cliquez sur cette icône :

Passer en mode 'zen' plein écran

Cette vue se chargera :

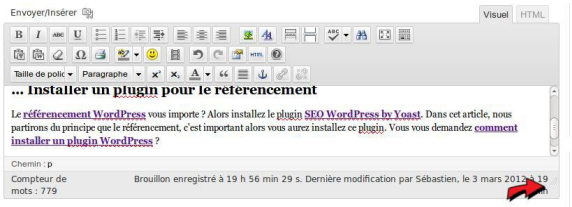


Édition de texte en mode plein écran

Pour sortir de ce mode, il suffit de cliquer sur *Sortir du plein écran :*

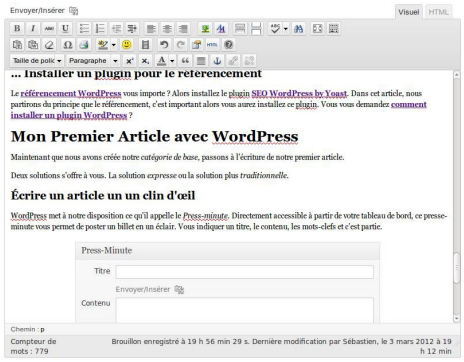
Sortir du mode plein écran

Vous préférez le mode *classique* pour écrire du contenu ? Saviez-vous que vous pouviez adapter cette zone à vos envies ? Pour l'agrandir, il suffit de saisir la poignée présente en bas à droite de l'éditeur de texte et de la monter ou descendre en fonction du résultat recherché.



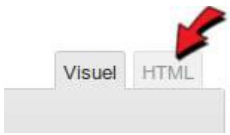
Agrandir la zone de saisie de texte

Nous, nous venons d'agrandir la fenêtre de rédaction du contenu.



Jolie mise en abime non ? :)

Le WYSIWIG c'est pour les *noobs* ? Vous préférez utiliser directement le HTML ? Pas de problème, WordPress ne vous oblige pas à utiliser l'éditeur WYSIWIG, l'onglet HTML vous permettant simplement de passer de l'un à l'autre. Notez que cette option est également disponible en écriture plein écran.



Basculer du mode visuel au mode HTML

L'éditeur HTML ressemblera à cela :

Éditeur HTML

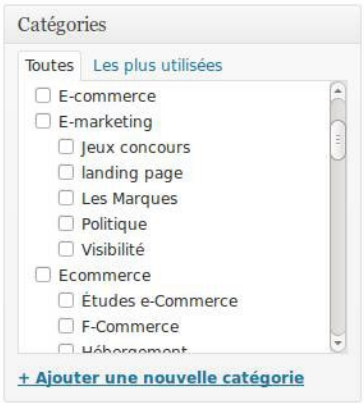
En plus d'écrire du texte, vous souhaitez insérer une image ? Il vous suffit de cliquer sur cette icône présente en haut à gauche de l'éditeur.

Insérer des images avec WordPress

2.4.3. La catégorie de l'article

Vous avez créé plusieurs catégories pour vos articles ? C'est maintenant le temps de choisir celle dans laquelle votre article s'insérera. Choisir plusieurs catégories pour un article n'est pas un crime, vous avez bien le droit de répartir votre contenu entre plusieurs catégories, mais attention toutefois à ce que cela ne soit pas une habitude. Si c'est le cas, alors repensez vos catégories. Deux risques se présentent à vous :

- ce que l'on nomme le *duplicate content*. Votre article sera accessible à partir de plusieurs URL différentes, les moteurs les indexeront toutes, éparpillant par là même votre présence ;
- vos lecteurs risquent de ne pas être très contents d'avoir l'impression que deux catégories se ressemblent comme deux gouttes d'eau.



Sélectionner la catégorie de l'article

2.4.4. Les mots-clés

Cette partie pourrait prêter quelque peu à confusion. Je m'explique. Les mots-clés de l'article, comme son nom l'indique, doivent servir à lister les mots-clés les plus pertinents de votre article. Mais cela ne s'arrête pas là. En fait, quand vous ajoutez un mot-clé, vous créez une nouvelle catégorie… troublant ?
Pas lorsque l'on a compris *pourquoi* l'ajout d'un mot-clé revient à créer une nouvelle catégorie. En fait, WordPress parlera de *taxonomie*. La taxonomie est l'art de ranger les choses dans des boites avec des étiquettes dessus en leur donnant un joli nom. Ici, il s'agit de relier entre eux des articles. Vous me direz, oui les catégories sont là pour trier le contenu. Je vous répondrai, ça dépend.
Une catégorie est bien là pour classifier le contenu, mais comment faire pour relier entre eux deux articles dans deux catégories différentes ? En passant par les mots-clés bien sûr !
Mais attention, vos mots-clés ne doivent pas s'appeler comme une catégorie. Déjà parce que WordPress vous interdira de le faire – quand on vous dit que c'est intimement lié – et parce que cela serait une erreur d'un point de vue référencement.

Choisissez vos mots-clés avec soin

2.4.5. Image à la une

L'image à la une est un élément utilisé par votre thème. Il se pourrait que celui-ci ne les supporte pas… mais ça devient tellement courant que ça m'étonnerait beaucoup.
Pour illustrer cet exemple, nous prendrons l'article **Top 100 Plugins WordPress** : Lien 62.
Pour ajouter une image à la une, cliquer sur *Mettre une image à la une*.

Choisir une image à mettre en une

Une fois l'image sélectionnée, elle apparaitra de cette manière dans votre interface d'administration :

Image à la une sélectionnée dans l'interface d'administration

Et de différentes manières sur votre site, en fonction de votre thème. Voici ce que cela donne pour notre site :

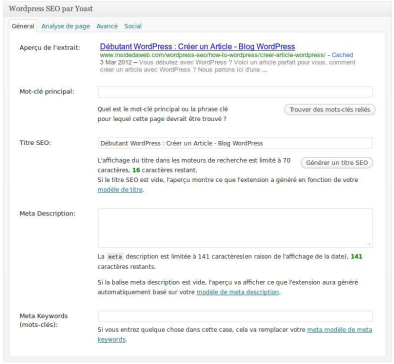
Image à la une sur la page d'accueil

Image à la une sur la page catégorie

Image à la une dans un article/page

2.4.6. Une pincée de référencement

Si vous écrivez pour le Web, c'est certainement dans le but d'être lu ou vu. Sinon, autant tenir son blog sur un rouleau de papier toilette non ?
Alors, ne faisons pas les choses à moitié, pensons référencement tout de suite. Bien entendu, vous avez suivi nos conseils et installé SEO WordPress by Yoast, c'est bien. ;)
Voici à quoi ressemblera l'interface avec ce *plugin* :



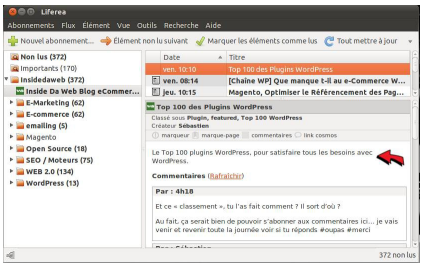
Je vous conseille la lecture de l'article **WordPress SEO by Yoast** si vous souhaitez comprendre son fonctionnement :

Lien 63.

2.4.7. Gérer un extrait de son article

Un *extrait d'article* est une sorte de résumé pour votre article qui apparaitra dans votre flux RSS et, en fonction de votre thème, à différents endroits de votre site.
La première des choses, écrire votre extrait dans la partie consacrée.

Ce résumé sera repris par votre thème, ou pas.



2.4.8. Publier son premier article

Maintenant que nous avons fait du bon boulot en écrivant comme il se doit notre premier article avec WordPress, il va falloir le publier.
En fait, WordPress ne vous offre pas que deux états pour vos articles qui sont *brouillon* ou *publié*. WordPress gère aussi un état *d'attente de relecture*. Très pratique pour ceux faisant beaucoup de fautes ou encore lorsque l'on travaille avec un client qui doit valider le contenu avant publication. Pour cela, il faut cliquer sur le lien *Modifier* à droite de *État*.

Ce n'est pas tout, WordPress vous permet aussi de gérer plusieurs niveaux de visibilité d'un article. Ce dernier peut être *public* – par défaut – *protégé par mot de passe* ou encore *privé*.

Lorsque vous cliquerez sur *Publier*, vous allez publier tout de suite votre article. Mais il est possible de planifier la sortie de ses articles avec WordPress. Un jeu d'enfant.

Un petit détail toutefois, assurez-vous de l'heure et de la date de votre serveur pour être certain que votre premier billet apparaitra en temps et en heure.

3. Mon premier article avec WordPress

Maintenant que nous avons créé notre *catégorie de base*, passons à l'écriture de notre premier article.
Deux solutions s'offrent à vous. La solution *expresse* ou la solution plus *traditionnelle*.

4. Écrire un article en un clin d'œil

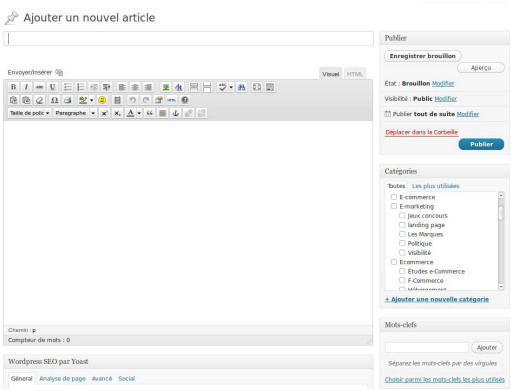
WordPress met à notre disposition ce qu'il appelle le *Press-minute*. Directement accessible à partir de votre tableau de bord, ce *Press-minute* vous permet de poster un billet en un éclair. Vous indiquez un titre, le contenu, les mots-clés et c'est parti.

Ici, il ne vous sera ni possible de choisir votre catégorie – d'où la création d'une catégorie de base, ni possible d'optimiser votre titre. En outre, notre *plugin* pour optimiser le référencement ne servira pas à grand-chose. Par contre, vous serez en mesure de publier un article en un rien de temps.

5. Écrire un article comme un grand

Je dois bien le dire, nous n'utilisons jamais le *Press-Minute*, nous passons toujours par l'éditeur principal.

Voici à peu près ce que vous aurez comme écran.



Détaillons un peu ce qui se présente à l'écran, dans l'ordre d'utilisation à savoir :

- le titre ;
- le contenu et l'éditeur de contenu ;
- le choix d'une catégorie pour l'article ;
- les mots-clés de l'article ;
- l'image d'illustration de votre billet ;
- les balises pour le référencement ;
- l'extrait de l'article ;
- la publication de l'article.

6. Conclusion

Cet article avait pour but de vous guider dans l'ensemble des tâches à faire pour bien publier son premier article. J'espère qu'il vous aura été utile. Si vous connaissez des astuces sur la création de son premier billet avec WordPress, partagez votre expérience dans les commentaires.

Retrouvez l'article d'Inside da Web en ligne : Lien 64

Liens

Lien 01 : http://dolphy35.developpez.com/article/access2013/web/
Lien 02 : http://dolphy35.developpez.com/article/access2013/macrosweb/
Lien 03 : http://www.apple.com/osx/preview/
Lien 04 : http://www.apple.com/macbook-air/
Lien 05 : http://www.apple.com/airport-time-capsule/
Lien 06 : http://www.apple.com/mac-pro/
Lien 07 : https://beta.icloud.com/
Lien 08 : http://www.developpez.net/forums/d1352642/systemes/mac/compte-rendu-keynote-wwdc-apple/
Lien 09 : http://www.gamasutra.com/view/news/185894/Its_official_XNA_is_dead.php
Lien 10 : http://www.monogame.net/downloads
Lien 11 : http://www.microsoft.com/en-GB/download/details.aspx?id=35471
Lien 12 : https://github.com/mono/MonoGame/wiki/Required-SDKs
Lien 13 : http://www.monodevelop.com/
Lien 14 : https://github.com/mono/MonoGame/wiki/GIT%27ing-Started-With-Git
Lien 15 : http://www.david-amador.com/2010/03/xna-2d-independent-resolution-rendering/
Lien 16 : http://docs.xamarin.com/guides/ios/advanced_topics/linker
Lien 17 : http://docs.xamarin.com/guides/android/advanced_topics/linking
Lien 18 : http://www.monogame.net/
Lien 19 : http://jeux.developpez.com/tutoriels/MonoGame/Transition-XNA/
Lien 20 : http://www.developpez.net/forums/d1357745/club-professionnels-en-informatique/actualites/intelligence-reseau-voitures-au-service-securite-automobilistes/
Lien 21 : http://www.developpez.com/actu/58780/-Lab-of-Things-Microsoft-ouvre-la-beta-de-sa-plateforme-dediee-a-l-Internet-des-Objets-le-SDK-est-disponible-pour-les-universitaires/
Lien 22 : http://www.developpez.net/forums/d1365928/bases-donnees/oracle/oracle-optimise-ses-solutions-dediees-au-developpement-solutions-embarquees/
Lien 23 : http://www.docjar.com/docs/api/sun/reflect/Reflection.html
Lien 24 : http://mail.openjdk.java.net/mailman/listinfo/core-libs-dev
Lien 25 : http://www.oracle.com/technetwork/java/faq-sun-packages-142232.html
Lien 26 : http://www.developpez.net/forums/d1360605/club-professionnels-en-informatique/actualites/oracle-abandonne-sunreflectreflectiongetcallerclass/
Lien 27 : http://plegat.developpez.com/tutoriels/java/pfem2d_guipanel/
Lien 28 : http://download.eclipse.org/eclipse/downloads/drops4/R-4.3-201306052000/news/
Lien 29 : http://www.eclipse.org/downloads/
Lien 30 : http://help.eclipse.org/kepler/index.jsp?topic=%2Forg.eclipse.platform.doc.user%2FwhatsNew%2Fplatform_whatsnew.html&cp=0_6
Lien 31 : http://help.eclipse.org/kepler/topic/org.eclipse.jdt.doc.user/whatsNew/jdt_whatsnew.html?cp=1_6
Lien 32 : http://help.eclipse.org/kepler/topic/org.eclipse.pde.doc.user/whatsNew/pde_whatsnew.html?cp=4_5
Lien 33 : http://www.developpez.net/forums/d1357721/environnements-developpement/eclipse/fondation-eclipse-publie-eclipse-kepler-43/
Lien 34 : http://thierry-leriche-dessirier.developpez.com/tutoriels/eclipse/raccourcis/fichiers/memento-eclipse.pdf
Lien 35 : http://www.icauda.com/files/memento-eclipse.pdf
Lien 36 : http://creativecommons.org/licenses/by-nc-sa/3.0/fr/
Lien 37 : http://thierry-leriche-dessirier.developpez.com/tutoriels/eclipse/raccourcis/
Lien 38 : http://www.developpez.net/forums/d1365938/java/edi-outils-pour-java/netbeans/netbeans-74-support-du-jdk-8-nouvelles-fonctionnalites-html-5-optimisation-performances/
Lien 39 : http://www.developpez.net/forums/d1369587/systemes/autres-systemes/mobiles/80-appareils-mobiles-vendus-au-second-trimestre-utilisent-android/
Lien 40 : http://www.developpez.net/forums/d1366207/java/general-java/java-mobiles/android/google-officialise-android-43/
Lien 41 : http://jmdoudoux.developpez.com/cours/developpons/java/chap-exceptions.php
Lien 42 : http://anisfrikha.developpez.com/tutoriel/java/exceptions/#LVI
Lien 43 : http://nicroman.developpez.com/tutoriels/android/exceptions/
Lien 44 : http://modulargrid.org/#panel
Lien 45 : http://modulargrid.org/#app
Lien 46 : http://guideguide.me/
Lien 47 : http://960.gs/
Lien 48 : http://978.gs/demo/
Lien 49 : http://978.gs/
Lien 50 : http://kpinto.developpez.com/tutoriels/css/webdesign-selon-grilles/
Lien 51 : https://github.com/chneukirchen/bacon/
Lien 52 : https://github.com/synbioz/rubymotion_part1
Lien 53 : http://synbioz.developpez.com/tutoriels/ruby/introduction-rubymotion/
Lien 54 : http://www.insidedaweb.com/curation-newsletter-laredoute-fr/
Lien 55 : http://www.insidedaweb.com/livre-web-informatique/
Lien 56 : http://www.insidedaweb.com/wordpress-seo/referencement-wordpress-seo/guide-complet-pour-optimiser-votre-referencement-sous-wordpress/
Lien 57 : http://www.insidedaweb.com/wordpress-seo/plugin/wordpress-referencement-wordpress-seo-by-yoast/
Lien 58 : http://www.insidedaweb.com/category/wordpress-seo/plugins-wordpress/
Lien 59 : http://www.insidedaweb.com/wordpress-seo/plugin/wordpress-debutant-comment-ajouter-plugin-widget-wordpress/
Lien 60 : http://wordpress.org/extend/plugins/tinymce-advanced/
Lien 61 : http://www.insidedaweb.com/wordpress-seo/debuter-avec-wordpress/7-options-cachees-pages-articles-wordpress/
Lien 62 : http://www.insidedaweb.com/wordpress-seo/plugins-wordpress/wordpress-top-100-plugins/
Lien 63 : http://www.insidedaweb.com/wordpress-seo/plugins-wordpress/wordpress-referencement-wordpress-seo-by-yoast/
Lien 64 : http://insidedaweb.developpez.com/tutoriels/php/wordpress/premier-article/