



Developpez

Magazine

Edition de Février-Mars 2009.
Numéro 20.
Magazine en ligne gratuit.
Diffusion de copies conformes à l'original autorisée.
Réalisation : Alexandre Pottiez
Rédaction : la rédaction de Developpez
Contact : magazine@redaction-developpez.com

Index

Java/Eclipse	Page 2
PHP	Page 10
(X)HTML/CSS	Page 18
C/C++/GTK/Qt	Page 24
JavaScript	Page 32
Apache	Page 35
Flex	Page 39
Sybase	Page 41
Mac	Page 44
Sécurité	Page 46
MS Office	Page 49
2D/3D/Jeux	Page 58
Liens	Page 66

Editorial

Ce mois ci, encore plus de rubrique, pour encore plus de plaisir.

Retrouvez une sélection de nos meilleurs articles, critiques de livres, et questions/réponses sur diverses technologies.

Profitez-en bien !

La rédaction

Article Apache

Intégrer la géolocalisation à Apache 2 sous Linux



Connaître la situation géographique des clients et sa prise en compte dans la configuration serveur.

par **Jean-Yves Lacour et julp**
Page 35

Article PHP

Le tutoriel pour grands débutants pressés



Apprendre le PHP en un tour de main.

par **Sylvie Vauthier**
Page 10

Guide pas à pas de l'utilisation de JBoss tools pour le développement JSF

Cet article présente l'utilisation de JBoss tools pour faciliter et industrialiser le développement d'applications web avec JSF

1. Introduction

1.1. Présentation de JBoss Tools

JBoss tools est un ensemble de plugins Eclipse pour le développement J2EE, il inclut les produits suivant :

- **Exadel Studio Product** : plug-in Eclipse pour le développement web, passé sous le giron de JBoss pour enrichir JBoss tools.
- **Hibernate tools** : permet de générer les fichier de mapping hibernate ainsi que les classes DAO en se connectant à la source de données,
- **JBoss jBPM tools** : fournit les outils pour modéliser , exécuter, et monitorer les processus métier. De plus, il permet de faciliter le développement de services web WSDL et des processus métier avec jPDL(jBPM Process Definition Language) et BPEL (Business Process Execution Language)
- **Drools IDE** : pour le développement des règles métier pour le moteur de règles métier *Drools*
- **JBoss Application Server tools** : pour la gestion du serveur d'application JBoss
- **JBoss Seam tools** : pour l'utilisation du framework JBoss Seam

JBoss tools supporte nativement les environnements : JSF-RI, MyFaces, Richfaces, JBoss Seam, Facelets, Shale.

JBoss tools est donc un outil puissant qui fait partie des meilleurs outils open source pour le développement JSF.

1.1.1. Installation de JBoss tools

1.1.1.1. Compatibilités système

- **Java** : Java 5.0 ou plus
- **Système d'exploitation** : Window, Linux.
- **Plug-ins** : JBoss tools contient tous les plugins nécessaires à son utilisation, l'installation vient avec les plugins suivants :
 1. WTP runtime
 2. GEF runtime
 3. JEM runtime
 4. EMF & SDO runtime
 5. XSD runtime

1.1.1.2. Installation

JBoss tools est proposé en téléchargement sur le site suivant ([Lien1](#)) , il est possible d'installer seulement le module de développement web et JSF, ou de prendre tout le package JBoss tools.

Pour installer le plugin, il suffit de télécharger l'archive et de la dézipper dans le dossier d'installation d'Eclipse.

Il est important de vérifier la compatibilité de la version de JBoss

tools avec la version d'Eclipse.

Pour l'installation à l'aide du site de mise à jour d'Eclipse : sélectionnez **Help/Software updates/Add Site**

Ajouter le lien suivant pour la version stable : ([Lien2](#))

Pour la version de développement, utile pour la dernière version d'Eclipse : ([Lien3](#))

Pour vérifier l'installation sélectionnez **Window/Open Perspective/Other/Web development**

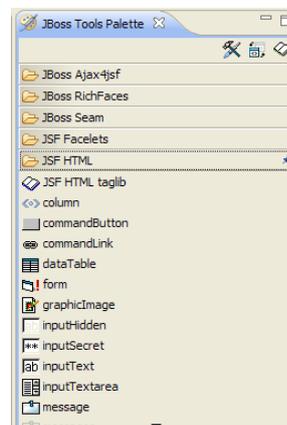
2. Présentation des outils JBoss tools pour JSF

2.1. La perspective Web development

La perspective web development contient le plan de travail pour un projet web. Pour ouvrir cette perspective : **Window/Open perspective/Web development**

2.2. La palette JBoss tools

La palette JBoss tools rassemble les librairies de tags JSF, Richfaces, Facelets et JBoss seam, il est possible d'ajouter d'autres librairies à la palette.



2.3. Editeur Graphique

Un éditeur pour pages JSF de type WYSIWYG est disponible pour aider au développement.

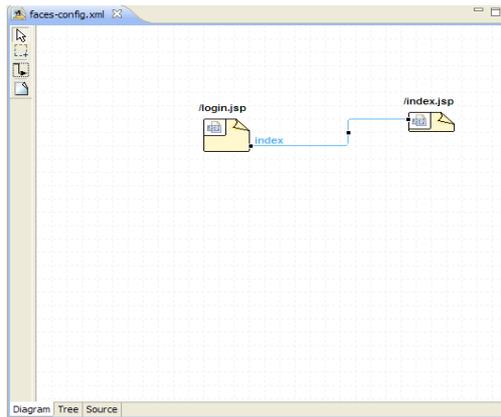


2.4. Vue du fichier de configuration de JSF

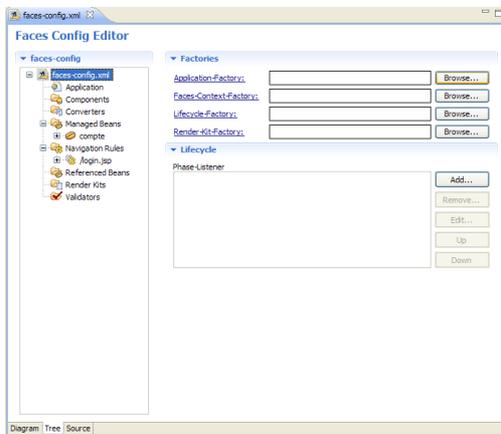
JBoss tools fournit un puissant éditeur de fichier de configuration faces-config.xml, il contient trois modes :

Diagram Tree Source

- **Diagram** Visualise le workflow de navigation



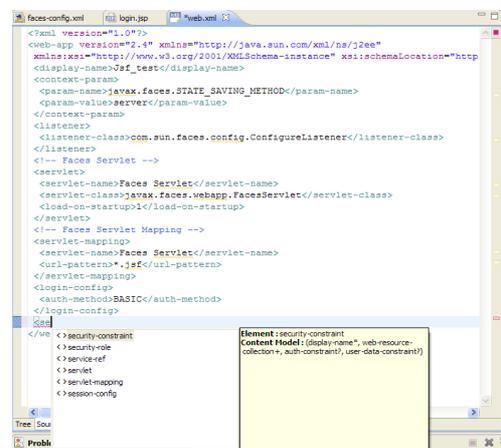
- **Tree** : contient tous les éléments du faces-config avec la possibilité de réaliser les opérations d'ajout/suppression/modification de chaque élément



- **Source** Le code source du faces-config.

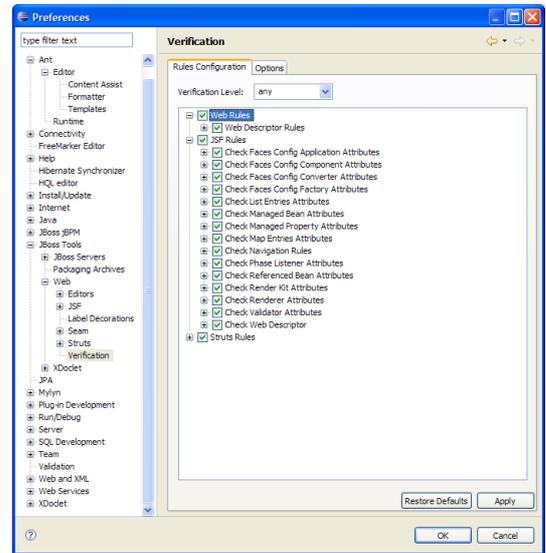
2.5. Assistant du contenu du code source

Dans tous les éditeurs du code source, un assistant est disponible. Il est lancé lorsqu'on commence à écrire, sinon, on peut le solliciter en utilisant CTRL-ESPACE. Voici un exemple dans le code source du web.xml, comme ce qui est proposé en natif dans WTP.

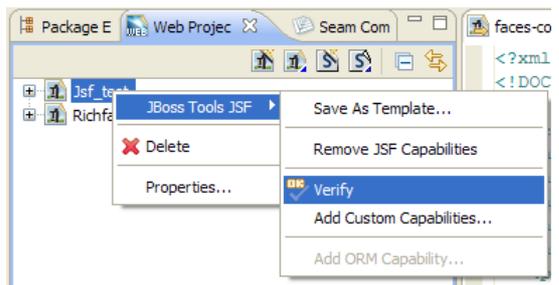


2.6. Vérificateur et reporteur d'erreur

JBoss tools permet de détecter les erreurs dans le faces-config, le web.xml ou les pages JSP. Il permet également de détecter les classes mal définies, une property d'un bean qui n'a pas de getter, ou une erreur de parsing des fichiers XML. Il est possible de définir les niveaux de vérification de JBoss tools, dans **Window/Preferences/JBoss Tools/Web, Verification**.



Il est possible également de faire une vérification explicite en faisant un clic-droit sur un projet JBoss Tools JSF et en sélectionnant ensuite **verify** :



2.7. Autres éditeurs

D'autres éditeurs très intéressants sont fournis avec JBoss Tools comme :

- Editeur XML
- Editeur Graphique de Properties
- Editeur Graphique de TLD
- Editeur Graphique de fichier de description de l'application Web (web.xml)
- Editeur CSS
- Editeur JavaScript
- Visualiseur d'Image
- Editeur XSD
- Editeur WSDL

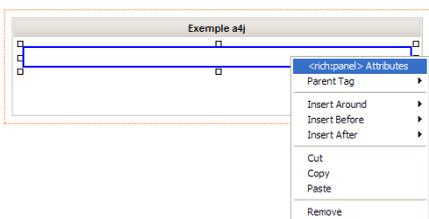
2.8. Présentation des nouveautés de la version 3.0 de JBoss tools

JBoss Tools 3 est disponible depuis le 31 octobre 2008 en version Beta, cette version est la première compatible avec Eclipse 3.4/Ganymede. Pour l'installer, il suffit d'ajouter le site des versions en cours de développement au gestionnaire de mise à jour d'Eclipse, ou la télécharger manuellement à l'adresse suivante ([Lien4](#)).

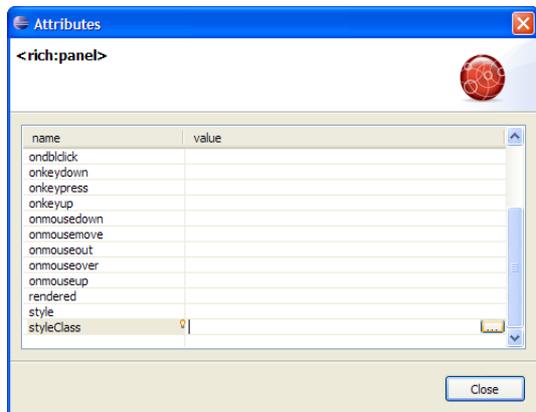
En ce qui concerne le développement JSF, les nouveautés concernent essentiellement l'éditeur visuel. Les problèmes rencontrés avec la version 2 sont pour la plupart corrigés. Parmi les nouvelles caractéristiques de la version 3.0, on peut citer :

- Amélioration de la rapidité de l'éditeur visuel. Il est maintenant possible d'écrire directement et aisément dans l'éditeur visuel.
- **Editeur de classe CSS** : un nouvel assistant de classe CSS est proposé, pour l'utiliser :
 1. Faire un clic droit sur le composant dans l'éditeur visuel et sélectionner ses attributs.

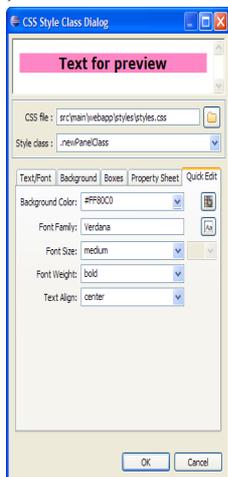
```
<rich:panel binding="#{person.panel}" id="tabs">
</rich:panel>
<br/>
<f:verbatim>
<br/><br/>
</f:verbatim>
</rich:panel>
```



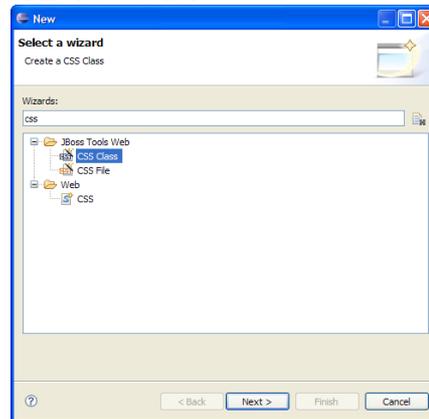
2. Choisir l'attribut styleClass, ensuite, cliquer sur le bouton d'aide à la saisie qui s'affiche sur la droite.



3. Choisir le fichier CSS, une classe de style à modifier, ou créer une nouvelle.



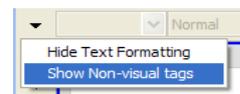
L'assistant CSS peut être sollicité dans l'attribut styleClass d'un composant sélectionné dans la palette ou directement : **File/New/CSS class** ou **File/New/Other**.



- Amélioration de l'assistant de code JSF : les problèmes rencontrés dans l'ancienne version sont en cours de correction, comme, la proposition de composants JSF à l'intérieur de l'attribut **jsfc** de Facelets.

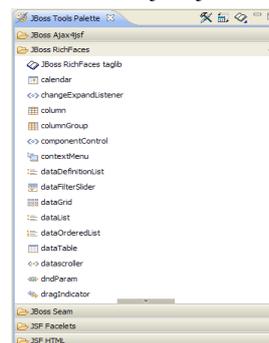


- Affichage des tags non visuels : L'éditeur propose l'affichage de composants non visuels :



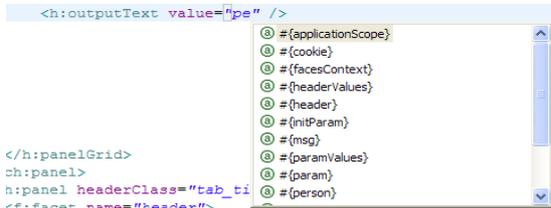
Lorsque cet affichage est choisi, les composants non visuels sont affichés ce qui permet au développeur de les localiser facilement et les sélectionner avec un simple clic.

- Affichage des messages issus du fichier de ressources : les dernières versions de JBoss tools ne permettaient pas de visualiser la valeur des clés de fichiers de ressources, alors que cette possibilité existait dans l'ancienne version du produit tenu par Exadel. Ce problème est corrigé dans la version 3.
- Support de l'ouverture de fichiers inclus à l'aide des tags **a4j:include** et **rich:insert**, comme pour le tag Facelets **ui:include**, l'assistant permet d'accéder directement au fichier inclus à l'aide de l'attribut **src** par l'utilisation de la touche **F3** ou de **CTRL+ clic**.
- Support de la complétion pour davantage de tags JSF notamment les tags JSF core.
- La palette change de look et contient de nouvelles icônes pour la librairie **ajax4jsf** et **Richfaces**.



- Support des nouveaux tags Richfaces par l'éditeur visuel : **rich:hotKey**, **rich:ajaxValidator**, **rich:graphValidator**, **rich:beanValidator**, **rich:extendedDataTable**.
- Support des nouveaux tags JSF 2 : **h:body**, **h:head**, **h:outputscript**, **h:outputstyle**.

On peut regretter néanmoins une régression de l'éditeur de code JSF lors du passage de la version d'Exadel compatible avec Eclipse 3.2 aux nouvelles versions JBoss tools, certains bugs ne sont toujours pas corrigés dans la version 3.0. Par exemple, dans la saisie des tags JSF ou Richfaces, l'éditeur ne filtre pas les valeurs disponibles dans l'expression EL.

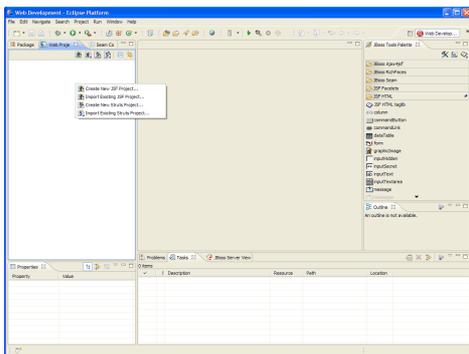


3. Guide de création d'un nouveau projet JSF

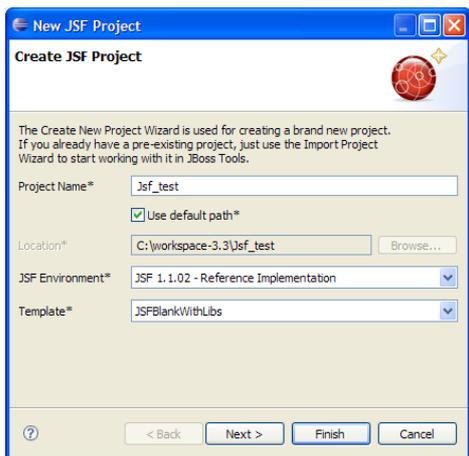
3.1. Création d'un nouveau projet

Nous allons utiliser l'assistant de création d'un nouveau projet JSF :

Clic droit sur la vue **Web project**, puis, sélectionner **Create New JSF project** :



Ajouter le nom du projet, l'environnement JSF et le template. Nous allons choisir l'implémentation JSF 1.1 avec les librairies.



L'assistant nous propose le choix de la version de l'implémentation JSF : JSF 1.1 ou 1.2 ainsi que myfaces 1.1.4.

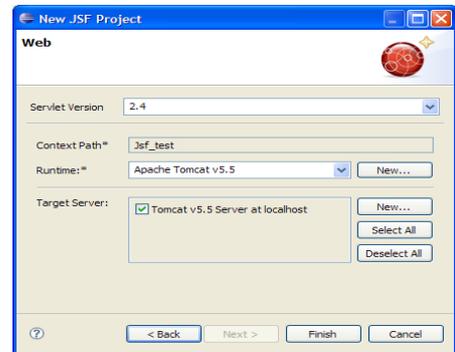
Le template est le type de gabarit du projet créé, les templates proposés sont :

JSFBlank : Création d'un projet web standard avec les capacités JSF

JSFKickStart : Création d'un projet JSF standard avec un exemple prêt à l'emploi.

With/Without Libs : Les deux templates sont disponibles avec une version qui inclut les librairies JSF dans le dossier WEB-INF/libs WithLibs et une autre sans les librairies JSF withoutLibs. Il est néanmoins possible d'ajouter son propre template (voir chapitre 4.1).

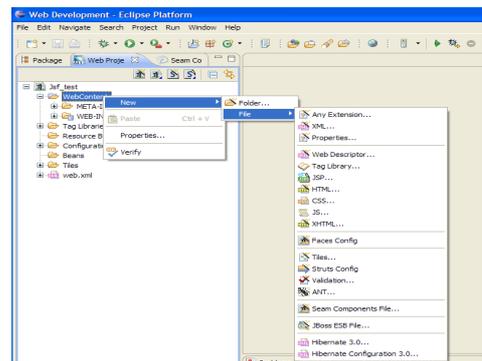
Déclarer la version de l'api des servlet et le serveur de déploiement :



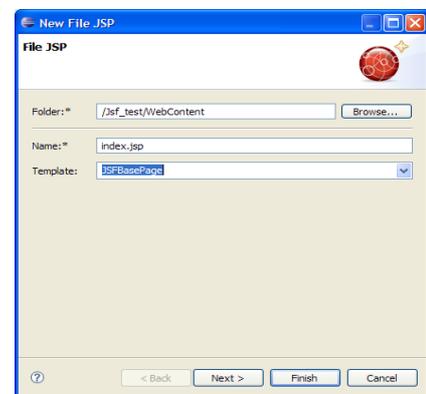
Le projet JSF est créé avec les librairies nécessaires, nous allons maintenant créer une nouvelle page JSF.

3.2. Première page JSF

Création d'un nouveau fichier JSP : on fait un clic droit sur le dossier WebContent, New, File, et on choisit JSP file.



Choix de template :



L'assistant nous propose de définir le template de page, les templates proposés sont :

Blank : une page JSP vide

JSFBasePage : page JSF avec la saisie des librairies de tags à utiliser :

JSPRedirect : une page de redirection.

Il est possible de créer sa propre page de template (voir chapitre 4.3).

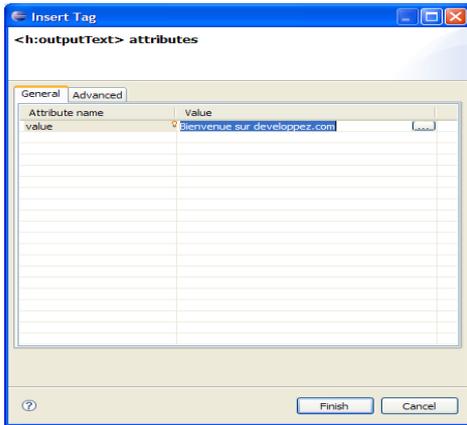
Nous allons choisir **JSFBasePage**, et nous sélectionnerons ensuite les tags utilisés dans la page, JSF core et JSF HTML. Notre page contient donc :

JSFBasePage

```
<%@ taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<%@ taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<html>
  <head>
    <title></title>
  </head>
  <body>
    <f:view>
      </f:view>
    </body>
  </html>
```

Nous souhaitons maintenant afficher un message sur la page, nous pouvons procéder de deux manières :

En utilisant la palette : utiliser la palette Jbos Tools, on sélectionne JSF HTML dans la palette, et on fait un clic sur `outputText`.



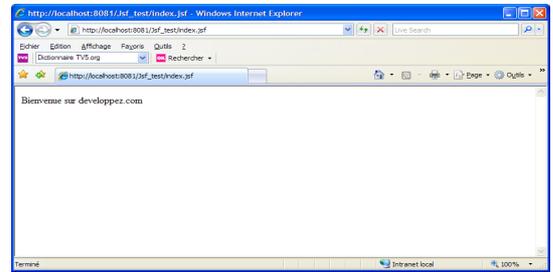
En utilisant l'éditeur : il suffit de commencer à écrire 'h:out' et l'éditeur propose la liste des composants qui correspondent aux premières lettres tapées.



Déployer l'application : pour déployer notre application, on l'ajoute dans les modules web dans la perspective J2EE/Server, ou on clique droit sur le projet et on fait **Run on Server**. Le démarrage et l'arrêt du serveur peuvent se faire à l'aide de :



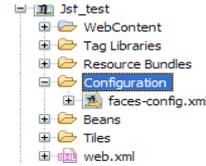
Le résultat :



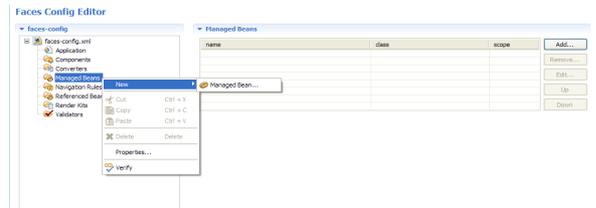
3.3. Ajouter un managedBean

Nous allons utiliser l'éditeur du fichier de configuration de JSF pour ajouter un nouveau managedBean.

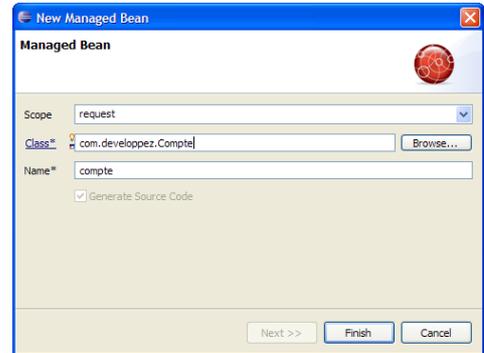
Ouvrir le fichier `faces-config.xml` qui se trouve dans : le dossier configuration



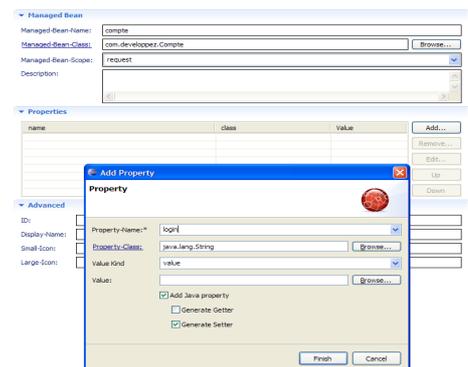
Aller dans l'onglet **Tree**, et clic droit sur `ManagedBeans` et faire **New ManagedBean**.



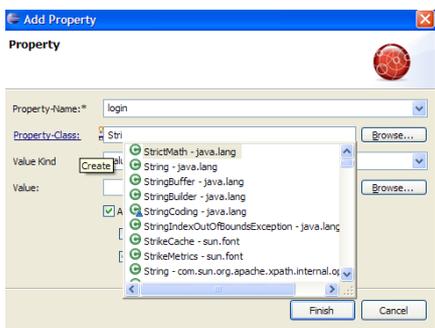
Saisir les paramètres du managedBean : l'assistant de création du managedBean nous propose de saisir le scope, la classe et le nom du managedBean, on coche `generate Source Code`, la classe sera générée dans le package défini.



Ajouter des propriétés : nous allons maintenant ajouter les propriétés du bean, en cliquant sur le bouton **Add** dans l'éditeur du managedBean.



Si on ne connaît pas le package exact de la classe, utiliser l'assistant en cliquant sur **CTRL-ESPACE**.

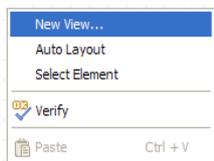


3.4. Ajouter une règle de navigation

Nous allons créer la navigation entre la page login.jsp et index.jsp. Passons maintenant sur l'onglet **Diagram** de l'éditeur du faces-config.



Clic droit dans l'éditeur, **New View**.



On ajoute une page JSFBasedPage login.jsp.

Dans le



diagramme, on réalise un clic sur l'icône create New connection

On lie index.jsp à login.jsp.

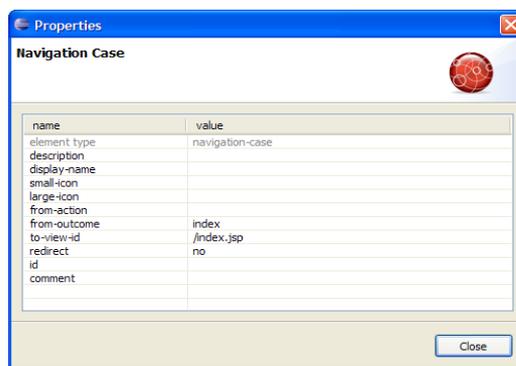


Les livres Java

Hardcore Java

Java has quickly become one of the most important languages in programming, particularly for professional and enterprise-level projects. From its infancy as a language primarily used for web applets to its maturity through servlets, Enterprise JavaBeans, and database access, Java has become a complex and robust tool for today's developer. Hardcore Java takes this language and breaks it apart, piece by piece, revealing the important secrets and tricks that will take you from a junior-level programmer to a seasoned and expert developer. You'll fly through the fundamentals and quickly find yourself learning about advanced memory management techniques, optimization and bytecode-level enhancements, and the techniques required to build lightning-fast GUIs. Throughout the book, you'll also master the art of writing and maintaining bulletproof and error-proof code, all while

Il est possible de modifier le cas de navigation en double cliquant sur la connexion.



Ajouter une action dans la page login.jsp.

```
<h:form>
    <h:panelGrid columns="2" cellspacing="5"
        cellpadding="5">
        <h:outputText value="Login" />
        <h:inputText value="#{compte.login}" />
    >
        <h:outputText value="Mot de passe" />
        <h:inputSecret
            value="#{compte.password}" />
        </h:panelGrid>
        <h:commandLink
            value="#{compte.connect}"></h:commandLink>
</h:form>
</f:view>
```

Nous pouvons accéder directement à la propriété du bean en cliquant sur **CTRL** + lien sur la property.

Ajouter la fonction de contrôle, dans la classe Compte en ajoutant la fonction connect :

```
public String connect(){
    return "index";
}
```

Retrouvez la suite de l'article de Lotfi Mellouk en ligne : [Lien5](#)

grasping the intricacies of the Java language. Hardcore Java covers:

- Use of the final keyword to optimize and protect your Java classes.
- Complete and thorough coverage of all types of nested classes, including how to optimize anonymous and inner classes.
- Detailed discussion of immutable objects, including unique tips on when to use them (and when not to).
- Elimination of bugs through exception-handling management.
- In-depth studies of constants, including their impact on the Java memory model.
- The most thorough discussion of reflection in print, moving far beyond other books' "Hello World" coverage.
- Construction and use of dynamic proxies, in both Java

Standard and Enterprise editions.

- Expansive coverage of weak references, including usage patterns and their role in garbage collection and memory management.

Hardcore Java is an invaluable addition to every programmer's library, and even the most advanced developers will find themselves moving beyond their own conceptions into truly advanced applications of the language. Thousands of lines of code, heavily commented and easily runnable, illustrate each concept in the book.

Critique du livre par Baptiste Wicht

Ce livre présente plusieurs particularités du langage Java ainsi que beaucoup de trucs et astuces permettant d'utiliser au mieux ce langage.

Le premier chapitre passe en revue les particularités du langage Java. Il met en avant quelques spécialités du langage et démontre quelques erreurs qui sont communes aux programmeurs Java. Le chapitre suivant est complètement dédié au mot clé final. Il parle de l'utilité du mot clé final pour les champs, les méthodes et les classes. Ce chapitre traite aussi de la compilation conditionnelle. Le chapitre suivant est dédiée aux objets immuables, de leur conception à leur utilité.

Le quatrième chapitre est entièrement dédié aux collections du langage Java. Il présente les différents types de Collection que propose l'API standard et les différents concepts qu'il est utile de connaître pour choisir une collection. Ce chapitre ne se contente pas de lister les différentes collections, il va décrire en détails les mécanismes de chacune de ces dernières. Il va également traiter de la performance des collections et des arguments pouvant conduire au choix de l'une ou de l'autre collection.

Le chapitre 5 décrit le mécanisme des exceptions dans Java. Il présente également les bonnes manières de gérer les exceptions dans un programme Java. Le chapitre suivant décrit en profondeur le système des classes internes. Il va décrire tous les types de classes internes, leur utilité et leur utilisation. Il donne aussi un résumé de ce que peut faire et ne peut pas faire chaque type de classe interne.

Le chapitre 7 traite des constantes. Il parle des constantes de substitution, des objets constants et des options constantes. Il traite également des "bit fields" qui sont des variables combinant plusieurs états vrais-faux. Avec le chapitre suivant, on passe dans la modélisation des données. Ce chapitre indique tous les points auxquels il faut faire attention quand on modélise un modèle de données. Il traite des contraintes sur le modèle, du langage de modélisation naturel et du choix entre une base de données orientée objet ou relationnelle.

Le chapitre suivant traite de la réflexion et son utilité dans un programme Java. Un premier exemple utilise la réflexion pour faciliter le logging des classes et un autre pour les tests unitaires. Le chapitre 10 décrit les proxies. Il va décrire le proxy statique et le proxy dynamique généré directement par Java. L'avant-dernier chapitre décrit les références et présente les différentes utilités de chacun des types de référence présente dans le langage Java. Enfin le dernier chapitre repasse sur les nouveautés de Tiger. Ce chapitre n'est utile qu'aux personnes ne connaissant pas encore ces nouveautés.

En conclusion, ce livre est très bon pour compléter ses connaissances sur le Java. En effet, ce livre traite de sujets qui ne sont habituellement pas traités dans les autres livres que j'ai pu lire

sur Java. Néanmoins, j'ai trouvé que certains chapitres n'allaient pas assez en profondeur dans leur sujet. Cependant, ce livre m'a permis d'appréhender des sujets encore quasi-inconnus et d'en approfondir d'autres. Je conseille ce livre à tous les utilisateurs du langage Java qui n'ont pas forcément encore poussé à fond dans le langage.

Wicket in action

Wicket bridges the mismatch between the web's stateless protocol and Java's OO model. The component-based Wicket framework shields you from the HTTP under a web app so you can concentrate on business problems instead of the plumbing code. In Wicket, you use logic-free HTML templates for layout and standard Java for an application's behavior. The result? Coding a web app with Wicket feels more like regular Java programming.

Wicket in Action is a comprehensive guide for Java developers building Wicket-based web applications. It introduces Wicket's structure and components, and moves quickly into examples of Wicket at work. Written by core committers, this book shows you the "how-to" and the "why" of Wicket. You'll learn to use and customize Wicket components, to interact with Spring and Hibernate, and to implement rich Ajax-driven features.

Critique du livre par Thomas Queste

Le livre en lui-même est bien présenté et agréable à lire. Les chapitres sont courts et bien découpés, ce qui facilite son usage comme référence pour plus tard.

Il a été écrit par deux des principaux Core Developers de Wicket depuis qu'il est OpenSource. On peut donc leur faire confiance quand ils donnent des conseils ;-). Les parties de code sont assez claires même si j'ai eu un peu de mal à comprendre où rattacher quoi à certains moments.

Les auteurs détaillent le framework progressivement même si l'application est déjà bien lancée au chapitre 3. Chaque partie est expliquée en détail mais pas trop. Je ne me lancerai quand-même pas dans la publication d'un composant sans y regarder à deux fois mais les principes et les bases sont évoqués.

Le chapitre sur Spring est bien présenté, on voit différentes façons de l'intégrer, y compris par les annotations. La partie sur « Aller en production » est vraiment intéressante car on y apprend que Wicket peut exposer des Mbeans pour se reconfigurer à chaud. Il est aussi expliqué comment tester ses composants et ses pages.

J'aurais aimé un chapitre sur les projets annexes que l'on retrouve côté Wicket Stuff ([Lien6](#)).

Par exemple, dans le chapitre sur la sécurité, Acegi n'est pas évoqué. Certes c'est un livre sur Wicket, mais quelques paragraphes pour montrer les extensions annexes auraient été les bienvenus.

De plus, le livre ne revient pas sur les changements qui ont eu lieu au fil des versions. Pourquoi ne pas parler de Wicket 2 qui est mort-né ? Et Wicket 1.4, quelles seront les nouveautés ? Allez, je demandais juste un bout d'introduction. Cette version est à peine évoquée au fil des pages.

Le livre a répondu à beaucoup des questions que je me posais. Son spectre n'est pas très large (pas de Wicket Stuff) mais largement suffisant pour démarrer rapidement. Je le recommande à ceux qui veulent tester Wicket et se lancer dans leur premier

projet.

Mais d'ailleurs, vous devriez déjà être sur Amazon ([Lien7](#)) pour commander votre exemplaire.

Retrouvez la critique originale (et complète) ici ([Lien8](#))

Critique du livre par Frédéric Morin

« Wicket In Action » ou comment mélanger la théorie et la pratique de façon agréable. Si vous souhaitez aborder le développement Web sous un nouvel angle, Wicket est la solution. Ce framework permet une réelle séparation de la partie HTML et de la partie JAVA. Cependant, l'approche étant fondamentalement différente, sur quel livre se baser pour réaliser l'apprentissage de ce framework ?

« Wicket in Action » livre publié chez Manning, sorti récemment, présente le framework avec beaucoup de pertinence, en ne sautant aucune étape, et en laissant encore un peu de travail au lecteur pour l'acquisition de certains composants.

Après une brève présentation du pourquoi, du comment, le livre s'attache à nous faire comprendre le fonctionnement intrinsèque du framework via une petite application Web (« Boutique de vente de Fromage ») mettant en œuvre les concepts fondamentaux. Cette approche très pragmatique du sujet permet au lecteur de comprendre en douceur le raisonnement qu'il devra acquérir pour réaliser lui-même des services. La présentation pas à pas des différents composants de l'application, de la méthode de mise en œuvre, et de la manière de séparer visualisation pour un graphiste et intégration pour un développeur, donne indéniablement envie d'en savoir plus.

Les chapitres suivants permettront d'approfondir les sujets, parfois abordés rapidement au cours de l'exemple (comme la notion de modèles), via une judicieuse montée en puissance, pour finir par l'intégration de composants Ajax dans les pages. Tous les composants de Wicket ne sont pas détaillés laissant au lecteur le plaisir de chercher pour voir plus loin.

Ce livre est écrit en outre par Martjin Dashorst, un collègue de Jonathan Locke (créateur de Wicket), qui connaît très bien le sujet pour avoir travaillé sur les toutes premières versions du framework, et qui a participé aux orientations sur les concepts et développements.

Maven: The Definitive Guide

For too long, developers have worked on disorganized application projects, where every part seemed to have its own build system, and no common repository existed for information about the state of the project. Now there's help. The long-awaited official documentation to Maven is here. Written by Maven creator Jason Van Zyl and his team at Sonatype, Maven: The Definitive Guide clearly explains how this tool can bring order to your software development projects. Maven is largely replacing Ant as the build tool of choice for large open source Java projects because, unlike Ant, Maven is also a project management tool that can run reports, generate a project website, and facilitate communication among members of a working team. To use Maven, everything you need to know is in this guide. The first part demonstrates the tool's capabilities through the development, from ideation to deployment, of several sample applications -- a simple software development project, a simple web application, a multi-module project, and a multi-module enterprise project. The second part offers a complete reference guide that includes: The POM and

Project Relationships The Build Lifecycle Plugins Project website generation Advanced site generation Reporting Properties Build Profiles The Maven Repository Team Collaboration Writing Plugins IDEs such as Eclipse, IntelliJ, and NetBeans Using and creating assemblies Developing with Maven Archetypes

Several sources for Maven have appeared online for some time, but nothing served as an introduction and comprehensive reference guide to this tool -- until now. Maven: The Definitive Guide is the ideal book to help you manage development projects for software, webapplications, and enterprise applications. And it comes straight from the source.

Critique du livre par Eric Siber

Ce livre porte bien son nom. Non seulement il est destiné à tout public, mais en plus il constitue vraiment un guide complet sur Maven 2.

En effet, il aborde tous les aspects de Maven 2 (cycle de vie, gestion de dépendances, outils, extensions, etc.) en fournissant au lecteur suffisamment d'éléments pour que celui-ci puisse approfondir s'il le souhaite.

Le découpage de l'ouvrage est bien fait et permettra à ceux qui ont déjà un peu d'expérience avec Maven de se focaliser sur l'un ou l'autre chapitre de la partie *Maven by Example* avant d'aller parcourir la partie *Maven Reference* qui se propose d'approfondir les connaissances de base.

Les premiers chapitres sont vraiment écrits dans une approche progressive et illustrent les possibilités de Maven 2 via des applications exemples représentatives.

J'ai été impressionné par le nombre d'astuces et de conseils prodigués, permettant d'utiliser Maven 2 de manière intelligente et non en suivant une simple mécanique de déclaration pure et dure de dépendances.

Si j'ai trouvé le chapitre sur les assemblages un peu plus indigeste que les autres, celui sur m2eclipse devrait ravir ceux qui sont lassés par l'édition de fichier XML et ne se sentent pas assez productifs dans leur utilisation de Maven 2 (on regrettera juste que les illustrations soient parfois à 1 ou 2 pages de la description, la faute au nombre important de visuels nécessaires à ce chapitre). Les chapitres plus avancés sur Nexus et la création de plugins sont également intéressants, même si sans doute moins utiles à la majorité des lecteurs. On a notamment droit à un éventail complet de l'utilisation et configuration de Nexus, excellent point de départ pour vous aider à initier un proxy/repository d'entreprise.

Enfin, on appréciera la présence des 2 annexes permettant respectivement d'avoir un aperçu des éléments de configuration pouvant être déclarés dans le fichier de configuration global (*settings.xml*), ainsi que les artefacts du serveur Geronimo qui vous permettront de pallier à l'indisponibilité (via les repositories) de ceux correspondants à la spécification Java EE (vous évitant de devoir installer ces derniers à la main).

En conclusion, *Maven: The Definitive Guide* est un livre bien écrit, assez complet, et dans lequel tout lecteur devrait s'y retrouver.

Ce qui est vraiment appréciable, c'est que les auteurs ne masquent en aucun cas les défauts et problématiques courantes survenant avec Maven 2, mais surtout qu'ils fournissent au lecteur les outils et pratiques pour traiter ces derniers.

Un livre à recommander.

Retrouvez ces critiques de livre sur la page livres Java : [Lien9](#)

Les derniers tutoriels et articles

PHP : Le tutoriel pour grands débutants pressés

Ce tutoriel, comme son nom l'indique, est intégralement conçu pour des grands débutants pressés.

Appelons grand débutant celui qui n'a jamais programmé ni entamé aucune initiation sur les notions générales de la programmation. Nous ambitionnons donc de satisfaire ce public, s'il est prêt à travailler régulièrement et à y consacrer une semaine intensive, par exemple...

Mais bien entendu, qui peut le plus peut le moins !

Aussi ce tutoriel ne devrait pas rebuter l'informaticien qui connaît déjà un ou plusieurs autres langages, mais souhaiterait se mettre à PHP, langage devenu incontournable pour la programmation web. Ce lecteur-là pourra parcourir à son rythme le tuto qui suit et y trouver réponses à ses questions...

Dans tous les cas, bonne lecture !

1. PHP ETAPE 1 : INTRODUCTION

1.1. Les pré-requis

1.1.1. Ce qu'il vous faut connaître

Pour profiter de ce tutoriel PHP pour grand débutant pressé, il faut répondre aux critères suivants :

a) **Connaître le langage d'affichage HTML** dans les grands principes...

Si ça n'est pas le cas, commencez par là. (Voir liste tutos recommandés sur la question dans le chapitre conclusion de ce cours)

b) **Posséder un ordinateur et une connexion internet**

1.1.2. Ce qu'il vous faut installer

1.1.2.1. Si vous êtes sous Windows

a) **Wamp server 2** : un environnement (gratuit bien sûr) qui vous permettra de programmer PHP en local... ([Lien10](#))

b) **Notepad++**

Un éditeur de texte (gratuit aussi) qui colore automatiquement le code que l'on entre en fonction de la logique de votre langage informatique, ce qui vous permettra une lecture facile... Cette aide indispensable s'appelle la coloration syntaxique. ([Lien11](#))

1.1.2.2. Si vous êtes sur Mac

Equivalent **Wamp server** ([Lien12](#))

Equivalent **Notepad++** ([Lien13](#))

1.1.2.3. Si vous êtes sous Linux

Equivalent **Wamp server** ([Lien14](#))

Equivalent **Notepad++** ([Lien15](#))

1.1.3. Et c'est tout pour ce tutoriel !

Pour le reste, on va essayer d'avancer, même si vous n'avez jamais programmé.

Ce tuto ne prétend pas être un topo savant et encore moins exhaustif : c'est une initiation, une première couche de peinture, indispensable pour faire vos premiers pas dans PHP, voire dans la programmation sans trop de souffrance, et pour que les couches suivantes tiennent.

Il se découpe en 8 étapes, avec des travaux pratiques. Il est bien évident que si vous êtes un grand débutant et que vous ne faites pas les travaux pratiques vous-mêmes, tout ce que vous lirez vous sortira par la tête aussi vite que c'est entré.

Il est donc recommandé, pour les grands débutants bien évidemment, de travailler un jour par étape, et de faire les travaux pratiques, pas simplement de se précipiter sur les réponses.

Amusez-vous bien durant votre semaine d'apprentissage !

1.2. Que fait PHP ?

Vous avez de bonnes notions de **HTML et CSS**, vous savez donc que ces deux langages ne sont pas des langages de programmation, mais des langages de **simple affichage statique**.

Facile d'afficher pour vous une page web qui dit : "bonjour, on est lundi." Le seul souci, c'est que l'on sera toujours lundi sur votre page...

Pas très dynamique tout ça. C'est là qu'intervient PHP qui est un langage de programmation web.

Il produit du code HTML.

En quoi est-il utile ?

Parce que le **code HTML que produit PHP change en fonction des circonstances que vous avez programmées**. On dira qu'il **introduit du dynamisme dans la page web**.

1.3. Quand PHP intervient-il ?

Le code PHP que vous avez inséré dans vos pages agit à chaque chargement (et donc rafraîchissement) de page web. Ceci pour relativiser son "dynamisme".

L'interactivité avec l'utilisateur se limite à certaines actions de l'utilisateur.

Un rafraîchissement de page, c'est par exemple le clic de l'utilisateur sur le bouton 'submit' inclus dans une balise form, ou bien une première arrivée sur une page web.

1.4. Un zeste d'algorithmique

Il faut enfin aborder dans cette introduction quatre notions clés de la programmation.

1.4.1. Tout d'abord le concept de variable

Un exemple suffira. On a tous eu entre les mains une publicité faussement personnalisée du type :

"Bonjour Madame Michu

Vous avez gagné notre canapé extra-cuir lors du tirage du 20/12/2008."

Il est clair que l'entreprise qui envoie cette pub possède une base de données avec des tas de noms. Et que c'est un programme automatique qui stocke les valeurs de Madame, de Michu, et la date dans des variables. Le reste est du texte figé. On va dire que **civilité, nom et date sont des variables...**

1.4.2. La programmation

Autre chose, programmer, c'est donner une suite d'instructions à la machine. Exactement comme l'on construirait une recette de cuisine. Sauf que l'on écrit la recette à l'usage d'une machine qui ne comprend que le binaire et qu'une instruction à la fois. **On avance, on avance, instruction par instruction, de façon linéaire.**

Dans la structure du programme, deux types de construction peuvent s'éloigner de cette linéarité :

1.4.3. L'instruction en boucle

On a une action affreusement répétitive. Du genre, découpe les trois plaques de chocolat en carrés individuels. En programmation, on ne va pas dire 150 fois, détache le carré de chocolat, détache le carré de chocolat, détache le carré de chocolat...

Alors on va faire ce que l'on nomme une boucle (il y en a de plusieurs sortes), du genre :

Boucle tant que

```
tant qu'il y a une plaquette de chocolat{
    détache le carré de chocolat
}
```

Une boucle parce que le programme lit la condition, entre dans la boucle après la première accolade (si la condition est vraie) et **"boucle" tant que la condition est vraie...**

1.4.4. L'instruction en condition

On a une condition. Restons sur notre recette de mousse au chocolat.

Si j'ai 6 gourmands, je casse 8 oeufs, si j'ai 6 personnes au

régime, je casse 4 oeufs, mais par défaut, une bonne mousse, c'est 6 oeufs pour 6...

Conditions qui s'excluent l'une l'autre

```
Sors le plat.
si gourmand{
    casse 8 oeufs
}
sinon si régime{
    casse 4 oeufs
}
sinon{
    casse 6 oeufs
}
Bien mélanger...
```

Cette fois, vous comprendrez que le cheminement logique linéaire se sépare en trois voies parallèles, un peu comme des rails, puis se regroupe à nouveau à la fin.

Le programme ne doit passer que par une et une seule des trois voies.

1.4.5. Pour conclure sur la programmation

Programmer, c'est finalement ramener toute problématique à l'une des trois possibilités suivantes :

- une instruction qui s'exécute automatiquement,
- une instruction qui s'exécute autant de fois que nécessaire dans une boucle,
- une instruction qui s'exécute dans une condition.

Ajouté à tout cela un certain nombre de variables que l'on manipule joyeusement...

Et vous avez les grandes lignes de la programmation.

1.5. TPI

Selon la méthode que vous venez de lire, pourriez-vous tracer la structure générale du programme qui gère la **distribution de carburant d'une pompe à essence automatisée qui ne distribuerait que ces trois carburants : Diesel, Super, et GPL ?**

Attachez-vous à la logique suivante : **telle action de l'utilisateur : telle partie du programme s'enclenche...**

Disons que ça commencerait ainsi :

```
//commentaire : affichage permanent
affiche 'Merci d'avoir choisi Vroum-Vroum !
Choisissez votre carburant :
Gazole, SuperSP ou GPL ?';

Si réponse{
recueille carburant choisi
//etc. etc.
}
```

1.6. Correction du TPI

Retenez déjà que dans toutes les propositions de correction, il y a **toujours plusieurs possibilités**, celle-ci n'est qu'un exemple.

Vous pourrez également vous imprégner de cette idée : ça n'est pas simple de penser à tout à la place d'une machine !

Ca exige des trésors de rigueur et de patience ! Mais bon, ça devient vite passionnant.

Distributeur automatique de carburant

```
//affichage permanent
affiche 'Merci d'avoir choisi Vroum-Vroum !
Choisissez votre carburant :
Gazole, SuperSP ou GPL ?';

//réaction quand il y a utilisateur
Si réponse {
    recueille carburant_choisi;
    affiche 'Entrez votre carte bleue';
    tantque info carte bleue pas entrée{
        affiche 'Entrez votre carte bleue';
    }
Sinon si info carte bleue entrée {
    affiche 'Entrez votre code';
    tant que code pas entré ou pas bon{
        affiche 'Entrez votre code';
    }
    Sinon si code bon{
        affiche 'Vous pouvez vous servir en ';
        affiche carburant_choisi;
    }

    //Pour la suite, initialisation de ma variable
nb_litres
    nb_litres=0;

    //Réaction quand utilisateur décroche le
pistolet
    Si user a décroché robinet{

        si réponsechoisie="Gazole"{
            prix_au_litre=1.12;
        }
        sinon si réponsechoisie="SuperSP"{
            prix_au_litre=1.21;
        }
        //réponsechoisie="GPL", le dernier cas par
défaut
        sinon{
            prix_au_litre=0.69;
        }
        tantque robinet coule{
            //le compteur tourne
            nb_litres=nb_litres+1;

            prix_temporaire=nb_litre *
prix_au_litre;
            affiche prix_temporaire;
        }
        //quand on arrive à cette ligne du programme, user
a raccroché le pistolet
        prix_total=prix_temporaire;
        débite le prix total de son compte;
        affiche 'Merci, et à la prochaine fois chez Vroum-
Vroum !';
    }
}
```

2. PHP ETAPE 2

2.1. Comment ça marche ?

L'ossature de toute page HTML

```
<html>
  <head>
    <title>Ma page de test</title>
  </head>
  <body>
    <h1>Bienvenue sur le site de toto </h1>
    <p>Le blabla de ma page...</p>
```

```
</body>
</html>
```

Voici une page HTML bien traditionnelle comme on les aime.
Copiez-collez en le contenu dans votre éditeur de texte notepad++.
Enregistrez et nommez-la test.html

Vous remarquerez que le code de la page se **colore de façon syntaxique**, ce qui le rend bien plus compréhensible. C'était l'intérêt de vous faire télécharger cet éditeur de texte. Vous apprécierez beaucoup dès que l'affaire se corsera.

Pour l'instant, enregistrez ce fichier sur votre bureau, fermez notepad++, et **double-cliquez maintenant directement sur votre fichier test.html, cela vous ouvrira une page web comme si vous étiez sur internet... Pourtant, vous n'y êtes pas, sur internet. Vous êtes "en local"**, bref, sur votre ordi, et vous ne faites que voir votre fichier **à la façon d'une page web**.

Rouvrez maintenant l'éditeur de texte et copiez/collez le bout de code suivant dans lequel on a introduit des balises PHP.

test.php contient des balises PHP

```
<html>
  <head>
    <title>Ma page d'accueil </title>
  </head>
  <body>
    <h1>Bienvenue sur le site de toto </h1>
    <p> Toto fait de l'anglais :</p>
    <?php
      echo '<p>Hello ! What is the day today ? It is
.date("l").' !</p>';
    ?>
  </body>
</html>
```

Quand vous l'enregistrez, sur votre bureau, **renommez-le test.php**, (changez l'extension de .html en .php). Ainsi, **le serveur php est avisé que cette page contient du code PHP** (que le navigateur ne sait pas afficher en ce qui le concerne).

Fermez votre éditeur. Double-cliquez directement sur votre page test.php : Quelle déception ! Cela ne vous ouvre pas une page web mais... le code que vous venez de quitter, sous un éditeur de texte qui est probablement bloc-notes sans coloration syntaxique. (Au passage, profitez-en pour paramétrer votre ordi pour ouvrir notepad++ par défaut pour les fichiers PHP)...

C'est ici que le fait d'avoir installé WampServer sur votre machine va vous être utile. En effet, il vous faut un **interpréteur PHP installé en local**.

Ouvrez votre serveur wamp. Une fois démarré, cliquez sur son icône dans la barre d'état à droite, une sorte de demi-sphère jaune, un peu comme un demi-pamplemousse.
Choisissez www directory.



2.2. Les variables

\$ devant une chaîne de caractères signifie que nous parlons d'une variable, on la nomme comme on veut ensuite, mais jamais d'accent ni d'espace dans un nom de variable.

Une variable est typée : elle peut prendre 4 types (dans l'ordre ci-dessous) :

Les types des variables

- string (chaîne de caractères)
- integer (nombre entier)
- bool (booléen qui signifie logique binaire genre vrai ou faux)
- float (nombre réel)

-syntaxe des variables

```
<?php
$nom='Mickaël';
$age=17;
$gars=true;
$taille=1.75;
?>
```

Étudiez bien chaque détail

- Toutes les instructions (toutes) **se terminent par un point virgule.**
- **Seule la variable string exige les guillemets** (simples ou doubles, préférez les apostrophes pour l'instant, nous verrons pourquoi un peu plus loin)...
- La booléenne prend la valeur **true, mais sans guillemets**, elle a pour contraire... **false**.
- **Le nombre réel prend un point à l'anglo-saxonne et non une virgule...**

N'hésitez pas à personnaliser ces exemples et à les triturer en **changeant les données pour les assimiler.**

2.3. echo

Bien maintenant que nous avons défini le type des variables en même temps que nous leur avons affecté une valeur, **affichons les grâce à echo, dans un environnement HTML :**

-syntaxe de la commande echo

```
<?php
echo'<p>Bonjour à tous.<br/>
Mon vrai nom n\'est pas Toto.<br/>
Mon vrai nom est ',$nom.'<br/>
J\'ai ',$age.' ans et je mesure ',$taille.'m.<br/>
Et comme mon nom l\'indique, je suis ' ;
if ($gars==true){
    echo 'un garçon.</p>';
}
else{
    echo 'une fille. </p>';
}
?>
```

Commentaires sur cette instruction echo :

Ici je n'ai fait que deux instructions 'affiche' : Une automatique, et une conditionnelle.

Vous remarquerez que je suis allée à la ligne **au sein de l'affichage grâce à la balise br/**. Vous remarquerez que toutes les apostrophes qui sont seulement du texte ont été **neutralisées en plaçant \ devant chacune d'elle** pour qu'elles ne soient pas considérées comme la fin de la chaîne de caractères.

Vous remarquerez que l'affichage du contenu de la variable se fait

Dans la fenêtre qui s'ouvre, créez un dossier PHP et glissez-y votre fichier test.php.

Vous passerez toujours par cette icône "pamplemousse" pour accéder à vos fichiers.

Cliquez maintenant sur cette icône, puis sur localhost. Dans la fenêtre qui s'ouvre, rubrique vos projets, ouvrez le dossier PHP... Cliquez sur votre fichier test.php et admirez le travail... **Le jour affiché en anglais, date qui sera toujours à jour...**

Cliquez affichage/source sur votre navigateur : Vous pouvez constater qu'il n'y a **plus trace de PHP**, il ne reste que du code HTML...

Voici rapidement ce qui se passe à chaque rafraichissement :

le serveur PHP va droit aux fichiers qui ont une extension PHP, puis il va droit aux balises PHP de ce fichier, et interprète ce qu'il y a dedans :

ici, il affiche (echo) tout ce qui est entre parenthèses, puis concatène ou colle si vous voulez (le point) le résultat de la fonction date qui renvoie le jour en anglais...

Il a donc interprété du PHP pour le traduire en HTML.

Ensuite, il repasse le fichier au navigateur, qui lui ne sait afficher QUE du HTML (ou du CSS)... et bien ça tombe bien, il ne voit que du HTML dorénavant...

Mais si vous revenez demain sur cette page, la fonction date affichera un autre jour dans le HTML. Voici pour l'aspect technique.

Pour la prise en main de Wamp, ne retenez que ceci pour l'instant :

Prise en main de Wamp

1. Quand vous voulez retrouver votre code : icône puis wwwdirectory et PHP/votre fichier...
2. Quand vous voulez voir le produit de votre travail, bref, tester votre code : icône puis localhost, répertoire PHP et votre fichier...

Ca deviendra vite un réflexe...

Maintenant que tout tourne, notre première étape portera sur **les variables** et sur la structure **echo** qui signifie donc affiche et que l'on utilise en permanence...

Voici quelques exemples... que vous testerez en les rajoutant sur votre fichier **au sein des balises PHP** bien entendu.

automatiquement, en mettant la variable nue dans le code.

Vous remarquerez que le point sert à concaténer (ou coller du texte bout à bout).

Il vous faut gérer les espaces à l'affichage au sein des guillemets. Ici la coloration syntaxique devient indispensable.

Enfin vous remarquerez que ma condition (si c'est un garçon ou une fille), bien qu'écrite à la ligne dans mon code et ouvrant une seconde instruction echo, s'affiche tout de même dans la continuité de mon précédent echo...

Autrement dit, c'est toujours au niveau du code HTML que se gèrent les retours à la ligne de l'affichage (balises br/, balises p ou autres).

Les retours à la ligne que vous faites spontanément dans votre code PHP après chaque instruction ne sont que des retours à la ligne à destination du programmeur pour une meilleure lisibilité de son code.

Dernière remarque : Prenez bien l'habitude de savoir à quel niveau vous êtes : au niveau du méta-langage PHP ou bien au niveau du code HTML qui aboutit à un affichage.

2.4. Guillemets ou Apostrophes ?

2.4.1. Ce que je ne conseille pas

Si vous êtes un peu flemmard, vous trouverez bien pratique de savoir que les guillemets "" (contrairement aux guillemets simples ou apostrophes) permettent ceci en PHP:

Le guillemet double permet ce code raccourci

```
<?php
$age=18;
//Ceci affichera directement
//J'ai 18 ans.
echo"J'ai $age ans.";
?>
```

Alors vous allez peut-être trouver cela un peu tordu d'utiliser les guillemets simples pour PHP du coup, puisque l'exemple ci-dessus règle en un seul coup de cuiller à pot le problème de l'interprétation directe de la variable et le problème de l'apostrophe du texte "J'ai"...

Pourtant, je pense que vous devrez procéder de façon plus rigoureuse si vous êtes amené à continuer avec PHP, aussi je vous prie d'adopter la convention suivante, pour ce tuto du moins :

2.4.2. Ce que je conseille

A l'avenir, vous allez fréquemment produire du HTML avec votre code PHP, puisque c'est sa fonction essentielle. Ainsi en adoptant la convention suivante, vous saurez toujours si vous êtes au niveau de PHP ou bien de HTML.

Gestion des apostrophes et des guillemets recommandée

```
<?php
echo'<div class="contenu">blablabla</div>';
?>
```

Quand vous utilisez un ' : on est au niveau de PHP.

Quand vous utilisez un " : on est au niveau du HTML.

Pour la clarté de vos idées, je conseille donc ceci :

Un code logique

```
<?php
$age=18;
echo'J'ai '.$age.' ans.';
?>
```

Bien sûr, il vous restera du coup à avoir le réflexe :

Attention, dans le contenu de mon HTML, entre deux balises, si j'ai une apostrophe dans mon texte (ce qui revient somme toute assez rarement) :

il faut que je pense au caractère d'échappement.

Mais ça ne devrait pas poser de problème majeur, toutes les phrases ne contenant pas autant d'apostrophes que dans l'exemple ci-dessous...

Une apostrophe ne fait pas le printemps

```
<?php
echo'<div class="contenu">L'hirondelle vole à tire
d'ailes vers l'Europe du Nord : L'hiver s'éteint.</
div>';
?>
```

2.5. TP2

Pour vos premiers travaux pratiques, vous programmerez ceci : A partir d'une page vierge de notepad++ et sans copier-coller, vous ferez la page emploi.php qui devra afficher ceci :

BONJOUR

Le salaire auquel j'aspire pour bien vivre : 2000€
La branche dans laquelle je travaille ou souhaiterais travailler : L'informatique
Pour préciser : L'informatique est la branche dans laquelle je travaille.
La note moyenne que j'ai obtenue au bac : 11.5

Bien entendu les données suivantes : 2000, L'informatique, et 11.5 ainsi que mon option (dans laquelle je travaille ou souhaite travailler) sont des variables car on peut les modifier à souhait.

Vous prendrez soin de les afficher en gras, ces variables...

Bon courage... Et interdiction de courir à la solution au premier échec.

Lisez bien les messages d'erreur, ils indiquent la ligne à modifier (ou parfois la ligne du dessus).

Il est inévitable de faire beaucoup d'erreurs de ce type au début... Donc patience, relisez le cours, mais ne venez à la solution que quand votre code produit quelque chose de satisfaisant et sans copier/coller.

2.6. Correction du TP2

emploi.php

```
<html>
  <head><title>Ma page d'accueil </title></head>
  <body>
    <h1>BONJOUR </h1>
    <?php
      //Initialisation des variables
      $travail='L'informatique';
      $salaire=2000;
      $etude=false;
      $bac=11.5;
      //affichage
      echo'Le salaire auquel j'aspire pour bien
vivre : <b>'.$salaire.'</b><br/>
```

```

    La branche dans laquelle je travaille ou
souhaiterais travailler : <b>'. $travail.'</b><br/>
    Pour préciser : <b>'. $travail.'</b>;
    if ($etude==true){
        echo' <b>est la branche dans laquelle je
souhaiterais travailler.</b><br/>';
    }
    else{
        echo' <b>est la branche dans laquelle je
travaille.</b><br/>';
    }
    echo'La note moyenne que j'ai obtenue au bac :
<b>'. $bac.'</b>';
    ?>
</body>
</html>

```

3. PHP ETAPE 3

Pour l'instant, on a manipulé des variables que l'on entrain "à la main" soi-même.

Il nous faut maintenant voir comment manipuler véritablement des variables, entrées par l'utilisateur.

3.1. Les formulaires

Vous le savez, un formulaire en HTML, c'est la suite de balises suivante :

un formulaire typique sur la page saisie.php

```

<form name="inscription" method="post"
action="saisie.php">
    Entrez votre pseudo : <input type="text"
name="pseudo"/> <br/>
    Entrez votre ville : <input type="text" name="ville"/
><br/>
    <input type="submit" name="valider" value="OK"/>
</form>

```

ci, ce formulaire présente une zone de saisie pour entrer son pseudo, va à la ligne, une zone de saisie pour entrer sa ville, va à la ligne, et enfin un bouton pour valider sur lequel sera écrit 'OK'...

Le but, vous l'aurez compris, c'est de récupérer, via PHP, les infos entrées par n'importe quel usager.

Voici comment les choses vont se découper :

Les attributs dans la balise form précisent le nom du formulaire, puis précisent que les variables contenues dans ce formulaire **seront envoyées par la méthode POST** (au moment où l'utilisateur cliquera sur le bouton "submit") **à la page saisie.php** (disons que c'est notre page de départ, oui, celle où il y a le formulaire)...

Quelles variables y aura-t-il, que comporteront-elles et surtout, quels noms porteront ces variables ?

Il y aura la variable `$_POST['pseudo']`, qui constitue le texte entré dans la zone pseudo par l'utilisateur avant d'avoir cliqué sur submit...

Et on continue ainsi, **selon la formule immuable `$_POST['name']` pour chaque input**, car les variables sont automatiquement nommées ainsi.

Pareil pour `$_POST['ville']`...

Enfin `$_POST['valider']` sera la variable qui dira, si elle existe bien entendu, qu'il y a eu clic sur la validation, et si elle n'existe pas, qu'il n'y a pas eu clic...

Information très importante, nous y reviendrons souvent...

Les choses sont claires ?

Comment allons-nous donc récupérer maintenant cette affaire, sur le fichier qui se nomme donc `saisie.php` et comporte ceci ?

saisie.php

```

<html>
  <head><title>Ma page d'accueil</title></head>
  <body>
    <h1>Bienvenue sur le site de toto </h1>
    <h2>Commencez-donc par vous inscrire :</h2>
    <form name="inscription" method="post"
action="saisie.php">
      Entrez votre pseudo : <input type="text"
name="pseudo"/> <br/>
      Entrez votre ville : <input type="text"
name="ville"/><br/>
      <input type="submit" name="valider"
value="OK"/>
    </form>
  </body>
</html>

```

En PHP, il faut toujours commencer par classer vos idées ainsi :

Quelle est la condition pour que mon code s'exécute ?

Autrement dit, quelle action de l'utilisateur va déclencher mon code...

Ah, tout ça sera donc dans une condition, voyons donc immédiatement la syntaxe du si :

3.2. La syntaxe de la condition if

Structure d'une condition if basique

```

if(user a cliqué sur valider){
    récupère la variable pseudo;
    récupère la variable ville;
    écris 'Salut (son pseudo) de (sa ville). Bienvenue
sur mon site !';
}

```

Voilà donc la structure d'une condition...

On entoure la condition entre des parenthèses, on encadre toutes les instructions dans des accolades.

Et encore une fois bien sûr, chaque instruction se termine toujours par un point-virgule.

Oubliez un détail de ce genre, et tout plante... Alors la syntaxe d'un truc aussi fréquent, il vaut mieux la répéter jusqu'à la savoir par coeur...

Tout en restant près de notre formulaire, imaginons que nous compliquons un peu notre condition :

Dans le premier si (il a cliqué), nous aimerions dire qu'en plus, si sa ville est Paris, nous lui proposons de le rencontrer...

Ce sera un 'si' imbriqué.

Même principe, mais cette fois, pensez à indenter votre code dès son élaboration, pour ne pas vous prendre les pieds dans le tapis.

Indenter, c'est décaler d'un cran le code pour raisonner par niveau d'imbrication...

Une condition imbriquée

```

if(user a cliqué sur valider){
    récupère la variable pseudo;
    récupère la variable ville;
}

```

```

    écris 'Salut (son pseudo) de (sa ville). Bienvenue
sur mon site !';
    if(sa ville)=='Paris'{
        écris 'On est plusieurs de Paris sur le site.
Si tu veux qu'on se voie, contacte-nous !';
    }
}

```

Remarquez une chose essentielle :

if(\$ville=='Paris') comporte deux signes =

C'est indispensable pour une comparaison.

La variable ne reçoit pas 'Paris' (signe = d'affectation), elle est comparée (==) à Paris.

Vous ferez forcément cette erreur au début, d'autant plus difficile à repérer qu'elle ne provoquera pas de message d'erreur :

Le programme comprendra => Si la variable reçoit Paris, et ce sera toujours vrai.

Autrement dit, votre programme ne fera pas ce que vous voulez, mais le langage ne pourra pas vous préciser votre bug...

Donc bien penser à ça en premier quand une condition ne tourne pas comme on veut.

(Ai-je bien mis un double égal dans la comparaison pour la condition ?)

Une dernière chose pour les si, **s'il y a un si, il faut parfois un ou plusieurs sinonsi, puis un sinon pour conclure** en tout dernier...

Restons toujours dans notre thématique formulaire :

Mettons que j'ai un formulaire qui m'a renvoyé l'âge entré par l'utilisateur :

```

<?php
$age=$_POST['age'];
?>

```

La syntaxe donnera ceci :

Conditions successives qui s'excluent l'une l'autre

```

<?php
if($age<5){
    $verdict='Ouh le bébé !';
}
elseif($age<13){
    $verdict='Vous êtes un enfant !';
}
elseif($age<18){
    $verdict='Vous êtes un(e) ado !';
}
else{
    $verdict='Ah ! enfin un(e) adulte !';
}
echo $verdict;
?>

```

Ici, pas d'imbrication, puisque ce sont des conditions successives et non imbriquées l'une dans l'autre.

La dernière (le else), c'est bien sûr 'sinon', le truc par défaut...

Ici, que signifie-t-il concrètement ?

Que user a 18 ans ou plus, n'importe quel entier > ou = à 18...

Dans tous les cas, il est affecté une valeur à \$verdict au moment où il parvient à l'instruction 'affiche moi le verdict'...

L'instruction n'est pas dans une condition, elle s'exécute à tous les coups.

Mais la variable \$verdict elle, n'est pas initialisée de la même façon selon que c'est l'une ou l'autre des 4 conditions de ce système de conditions qui l'a initialisée.

En d'autres termes, **dès l'instant où le programme tombe sur une condition qui se réalise, il cesse de tester la suite et passe à l'exécution de l' instruction...**

3.3. Les formulaires, suite

Revenons maintenant sur le traitement de notre formulaire de départ, avec une toute petite condition bien classique :

```

if(user a cliqué sur valider){
    récupère la variable pseudo;
    récupère la variable ville;
    écris 'Salut (son pseudo) de (sa ville). Bienvenue
sur mon site !';
}

```

Comment allons-nous dire en PHP la condition ou les instructions ? De la façon suivante, que je commenterai ensuite :

la syntaxe pour récupérer une variable POST

```

<?php
if(isset($_POST['valider'])){
    $pseudo=$_POST['pseudo'];
    $ville=$_POST['ville'];
    echo 'Salut '. $pseudo.'de '.
$ville.'<br/>Bienvenue sur mon site !';
}
?>

```

Traduction :

s'il existe une **variable POST qui a pour nom 'valider'** et que donc user a validé,

\$pseudo reçoit la variable POST qui a pour nom 'pseudo'

\$ville reçoit la variable POST qui a pour nom 'ville'

affiche... bon la fonction **echo** n'a plus de secret pour vous...

Remarque :

On peut très bien se passer de ranger la variable de type \$_POST['name'] dans une variable au nom simplifié de type \$name, et la manipuler directement !

Mais si on doit la manipuler sans arrêt, c'est plus agréable (**et plus stable**) de la renommer...

Autre remarque :

Puisque c'est l'attribut **name** dans la balise d'un objet type formulaire qui va déterminer le nom de la variable POST, **pensez-bien à ne pas y mettre d'accent, ni d'espace...**

Le name ne sert pas à l'affichage, le user ne le verra pas ! C'est pour le programmeur, pour vous...

3.4. Les formulaires, fin

On remet tout ça bout à bout dans le code maintenant !

Fichier saisie.php : un formulaire et son traitement

```

<html>
    <head><title>Ma page d'accueil</title></head>
    <body>
        <h1>Bienvenue sur le site de toto </h1>
        <h2>Commencez-donc par vous inscrire :</h2>
        <form name="inscription" method="post"
action="saisie.php">

```

```

        Entrez votre pseudo : <input type="text"
name="pseudo"/> <br/>
        Entrez votre ville : <input type="text"
name="ville"/><br/>
        <input type="submit" name="valider"
value="OK"/>
    </form>
    <?php
    if(isset($_POST['valider'])) {
        $pseudo=$_POST['pseudo'];
        $ville=$_POST['ville'];
        echo 'Salut '. $pseudo.'de '.
$ville.'<br/>Bienvenue sur mon site !';
    }
    ?>
    </body>
</html>

```

Vous expérimentez tout ça en local bien entendu...

Changez des détails, triturez, appropriez-vous la méthode et la syntaxe...

D'autant que le TP3 est un véritable travail de programmation cette fois.

Prenez bien votre temps...

3.5. TP3

Sans copier, ni copier/coller le cours, concevez une page qui s'appelle imc.php, et propose un formulaire sous la forme suivante :

```

Entrez votre prénom :
Entrez votre taille (sous la forme 1.70) :
Entrez votre poids (en kilos) :
OK

```

Ensuite vous traiterez le formulaire en PHP pour que quand la personne clique, elle voit apparaître :

```

Bonjour (son prénom)
Votre IMC (indice de masse corporelle) est exactement :
(son imc)
Vous avez (une corpulence normale ou bien Vous êtes en
surpoids, ou bien Vous êtes obèse etc...)

```

Pour la petite opération, l'IMC, c'est le poids divisé par la taille divisé par la taille...

En voici la syntaxe :

\$poids/(\$taille*\$taille)

(Nous reviendrons plus tard sur les opérateurs)...

Pour les conditions qui déterminent les tranches de l'IMC, consultez wikipedia par exemple, rubrique : interprétation de l'IMC, classification de l'OMS.

N'omettre aucune tranche bien entendu !

[\(Lien16\)](#)

3.6. Correction du TP3

```

imc.php
<html>
    <head><title>Votre IMC</title></head>
    <body>
        <h1>Déterminez votre IMC et sachez quelle est
votre corpulence d'un point de vue médical</h1>
        <h2>Entrez les données suivantes </h2>
        <form name="formulaire" method="post"
action="imc.php">
            Entrez votre prénom : <input type="text"
name="prenom"/> <br/>
            Entrez votre taille (sous la forme 1.70) :
<input type="text" name="taille"/> <br/>
            Entrez votre poids (en kilos) : <input
type="text" name="poids"/> <br/>
            <input type="submit" name="valider"
value="OK"/>
        </form>
        <?php
        if(isset($_POST['valider'])) {
            $prenom=$_POST['prenom'];
            $taille=$_POST['taille'];
            $poids=$_POST['poids'];

            $imc=$poids/($taille*$taille);

            echo 'Bonjour '.$prenom.'<br/>
Votre IMC (indice de masse corporelle) est
exactement : '.$imc.'<br/>';

            if ($imc<16.5){
                $verdict='Vous êtes en dénutrition.';
            }
            elseif ($imc<18.5){
                $verdict='Vous êtes maigre.';
            }
            elseif ($imc<25){
                $verdict='Vous avez une corpulence
normale.';
            }
            elseif ($imc<30){
                $verdict='Vous êtes en surpoids.';
            }
            elseif ($imc<35){
                $verdict='Vous êtes en état d\'obésité
modérée.';
            }
            elseif ($imc<40){
                $verdict='Vous êtes en état d\'obésité
sévère.';
            }
            else{
                $verdict='Vous êtes en état d\'obésité
massive.';
            }

            echo $verdict;
        }
        ?>
    </body>
</html>

```

Retrouvez la suite de l'article de Sylvie Vauthier en ligne : [Lien17](#)

Les derniers tutoriels et articles

Des feuilles de style CSS dynamiques avec PHP

Les feuilles de style CSS sont très pratiques pour séparer le fond et la forme mais il leur manque la possibilité de déclarer des variables que l'on pourrait réutiliser tout au long d'un projet web. Heureusement pour nous PHP est là et nous allons voir comment l'utiliser pour dynamiser nos CSS.

1. Introduction

Les bonnes pratiques de développement web font leur chemin et l'intégrateur web prend l'habitude de séparer le fond (code HTML) de la forme (feuille de style CSS et Javascript) dans des fichiers distincts pour une saine séparation des pouvoirs et une maintenance aisée. Malgré tout, quand je regarde mes feuilles de style, je trouve une palanquée de valeurs redondantes ou dépendantes d'autres valeurs. En attendant qu'une prochaine génération ([Lien18](#)) de CSS gère les variables, voici plusieurs manières d'ajouter du code PHP dans vos fichiers CSS.

2. Trois techniques pour ajouter du PHP dans vos CSS

Pour inclure des variables ou du code PHP dans une feuille de style, il n'y a pas 36 solutions : il faut s'arranger pour que le serveur interprète le code PHP. Pour cela, je vous propose trois méthodes :

2.1. Inclure un fichier PHP avec la fonction "include()"

La première et la plus simple est tout simplement d'inclure un fichier PHP contenant vos déclarations CSS qui pourront contenir tout le code PHP que vous voudrez à l'intérieur d'une balise style :

Méthode avec la fonction include()

```
<style type="text/css">
  <?php include('/styles.php'); ?>
</style>
```

2.2. Servir le contenu d'un fichier PHP en tant que "text/css"

La deuxième solution utilise toujours un fichier *styles.php*, mais cette fois, on le place à l'intérieur de la balise `<link>` :

Balise link

```
<link rel="stylesheet" href="/styles.php"
type="text/css" />
```

Dans le fichier *styles.php*, il suffit d'ajouter une ligne pour indiquer au navigateur que le contenu doit être considéré comme du "text/css" :

Header PHP de content-type

```
<?php
  header('content-type: text/css');
?>
```

Comme le fait remarquer Xavier Borderie dans Des CSS dynamiques via PHP ([Lien19](#)), la différence avec un "vrai" fichier CSS, c'est que le fichier *styles.php* ne sera pas placé dans le cache du navigateur et sera chargé systématiquement à chaque requête. Pour remédier à cela, il propose d'ajouter la ligne :

Header PHP du code HTTP 304

```
<?php
  header('HTTP/1.0 304 Not Modified');
?>
```

Dans le même esprit, il semble possible d'utiliser les fonctions de *Cache-Control* comme *max-age* et *must-revalidate* :

Header PHP de cache-control

```
<?php
  header('Cache-Control: max-age=3600, must-
revalidate');
?>
```

Ceci indique le nombre de secondes durant lesquelles la page devra être conservée dans le cache du navigateur, à partir du moment où elle a été demandée.

2.3. Utiliser une règle de réécriture pour transformer les .css en .php

La troisième solution est identique à la deuxième, si ce n'est que le fichier CSS se termine par l'extension *.css* : un fichier *htaccess* se charge de de la réécriture des fichiers se terminant par *.css* en *.php*.

- Dans le fichier HTML :

Balise link

```
<link rel="stylesheet" href="/styles.css"
type="text/css" />
```

- Dans le fichier *htaccess* :

Code de réécriture

```
RewriteEngine on
RewriteRule (.*)\.css /$1.php
```

Ou alors, on peut aussi demander au serveur d'interpréter les fichiers *.css* comme étant des fichiers PHP en rajoutant cette ligne dans un fichier *htaccess* :

Parsing des .css par PHP

```
AddType application/x-httpd-php .css
```

3. Mettre des variables ou des fonctions PHP dans notre fichier servi comme "text/css"

Pour commencer, voici un récapitulatif des éléments abordés plus haut avec quelques exemples de variables PHP placées dans des tableaux `$_SESSION` afin qu'elles soient disponibles tout au long de la navigation.

- Le fichier *index.php*

Exemple d'utilisation : le fichier *index.php*

```
<?php
session_start();
$_SESSION['bg']['body'] = '#00f';
$_SESSION['bg']['container'] = '#0f0';
$_SESSION['bg']['footer'] = '#f00';
$_SESSION['txt']['police'] = '#333';
$_SESSION['txt']['border'] = '#000';
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
xml:lang="fr" lang="fr">
<head>
<title>Calendrier PHP</title>
<meta http-equiv="Content-Type" content="text/html;
charset=utf-8" />
<link rel="stylesheet" href="style.css" media="all"
type="text/css" />
</head>
<body>
<div id="container"></div>
</body>
</html>
```

- Le fichier *styles.php* (ou *styles.css* avec le fichier *htaccess*)

Exemple d'utilisation : le fichier *style.php*

```
<?php
header( 'content-type: text/css' );
session_start();
?>
body {
background: <?php echo $_SESSION['bg']['body']; ?>
}
#container {
background: <?php echo $_SESSION['bg']
['container']; ?>;
color: <?php echo $_SESSION['txt']['police']; ?>;
border: 1px solid <?php echo $_SESSION['txt']
['border']; ?>
}
```

4. Pour conclure

Si l'on excepte la première solution qui semble un peu "cheap", les deux suivantes semblent allier la souplesse du PHP avec le respect des standards du web et la conformité aux bonnes pratiques. Sans oublier les validateurs qui ne devraient rien trouver à redire. Au point que je me demande si l'utilisation des variables en CSS n'est pas une pratique plus répandue que je ne le croyais.

Retrouvez l'article de Bruno Bichet en ligne : [Lien20](#)

Réalisez un histogramme avec CSS

La visualisation de données est réalisée le plus souvent par des applications flash ou avec l'aide de certains langages de programmation. Ces solutions sont-elles le seul moyen de présenter, disons un simple graphique de données ? Et si on essayait rien qu'avec du bon vieux CSS ?

1. Approche

Dans cet exemple, je n'utilise pas de JavaScript, ni aucune application en arrière-plan. Tout ce sur quoi je compte est un balisage bien formé et du CSS.

Donc, l'objectif est de présenter des données sous forme graphique. Nous pourrions dire que la graphique est un objet en deux dimensions. Donc, le meilleur choix de structure et de sémantique est la liste de définitions. Pourquoi ? Eh bien, pour commencer, il s'agit d'une liste de points. Bien que la liste soit linéaire, on pourrait interpréter la définition des titres (dt éléments) sur les points d'axe x et la définition des descriptions (éléments dd) en tant que valeurs sur l'axe y.

Si cela n'a pas de sens en raison de mes piètres capacités d'explication :) disons que ce que je vais faire est transformer ceci :

```
Day 1
36
Day 2
45
Day 3
51
Day 4
48
Day 5
55
```

en cela :



uniquement avec du css.

2. La balisage

Dans mon exemple, j'ai utilisé une période des 12 derniers jours et présenté mon niveau d'énergie au travail en pourcentage, 100% étant l'énergie la plus élevée que j'ai jamais déployée dans un travail. Comme vous pouvez le voir, le 6ème jour est plus important, parce que j'ai reçu un gros chèque et que ça m'a boosté :)

Quoi qu'il en soit, pour la structure de ce type de données, j'ai choisi la liste de définitions.

La définition du titre contient le titre du jour :

```
<dt>Day 1</dt>
```

Et la description des définitions contient la valeur

```
<dd>36</dd>
```

À l'intérieur de l'élément de description des définitions je vais ajouter un span et un élément em imbriqués. Non pas dans un but structurel, mais pour me donner suffisamment d'éléments avec lesquels travailler. Je vais aussi attribuer des classes que je pourrais utiliser plus tard.

Alors la description de définition ressemble maintenant à ceci:

```
<dd><span class="type2 p80"><em>80</em></span></dd>
```

3. Style

La liste de définition contient les actes comme un tableau, je vais donc assigner lui une image de fond et définir la largeur et la hauteur correspondant à la dimension du graphique. Je vais également enlever les marges et paddings par défaut.

```
dl#csschart, dl#csschart dt, dl#csschart dd
{
    margin:0;
    padding:0;
}

dl#csschart
{
    background:url(bg_chart.gif) no-repeat 0 0;
    width:467px;
    height:385px;
}
```

La définition du titre n'a aucun intérêt visuel, je vais donc la cacher.

```
dl#csschart dt
{
    display:none;
}
```

Passons aux choses sérieuses. La définition des descriptions contient 2 éléments enfants: span et em. Cela me donne 3 éléments pour travailler.

Je vais mettre `<dd>` dans la bonne position en ajoutant les marges, en définissant la hauteur telle qu'elle corresponde à une valeur de 100% et la largeur telle qu'elle corresponde à une colonne de l'arrière-plan du graphique. Les `<dd>` sont flottants à gauche pour qu'ils s'alignent. Les `<dd>` ont une position relative de sorte que les éléments enfants puissent s'identifier à eux.

```
dl#csschart dd
{
    position:relative;
    float:left;
    display:inline;
    width:33px;
    height:330px;
    margin-top:22px;
}

dl#csschart dd.first
{
    margin-left:33px;
}
```

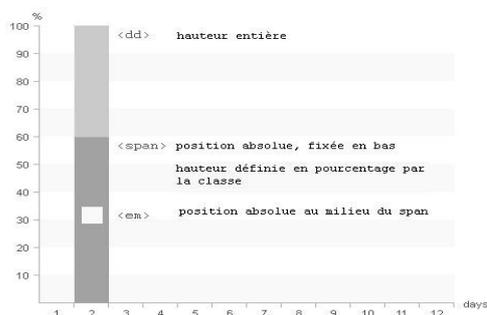
Les éléments Span représentent la colonne et les em sont placé au centre avec des valeurs. Les spans sont en position absolue, et placés en bas, afin de leur permettre de "grandir" de bas en haut sur la base de leur pourcentage.

```
dl#csschart span
{
    position:absolute;
    display:block;
    width:33px;
    bottom:0;
    left:0;
```

```
z-index:1;
color:#555;
text-decoration:none;
}

dl#csschart span em
{
    display:block;
    font-style:normal;
    float:left;
    line-height:200%;
    background:#fff;
    color:#555;
    border:1px solid #b1b1b1;
    position:absolute;
    top:50%;
    left:3px;
    text-align:center;
    width:23px;
}
```

Voici le schéma d'une barre graphique :



Alors, comment définissons-nous la hauteur exacte ?

Nous le faisons par réglage de la hauteur en pourcentage pour l'élément span. Vous avez remarqué dans la section ci-dessus que le span s'étend sur deux classes CSS. La première définit le type (type1 - type4). Dans mon cas les types sont utilisés pour présenter des couleurs différentes. La deuxième classe (p0 - P100) est celle qui définit la hauteur du span.

```
dl#csschart span.p0{height:0%;}
dl#csschart span.p1{height:1%;}
dl#csschart span.p2{height:2%;}
.
.
.
dl#csschart span.p100{height:100%;}
```

Ainsi, en plus de mettre la valeur en tant que texte, nous avons fourni une classe CSS qui correspond.

```
<dd><span class="type2 p80"><em>80</em></span></dd>
```

Cela fera l'affaire.

Bien sûr, le balisage peut être généré par un programme ou à la main, selon ce que vous préférez.

Voir un exemple en ligne ([Lien21](#)).

Retrouvez l'article d'Alen Grakalic traduit par Alban Lelasseux en ligne : [Lien22](#)

Position fixe en CSS

Ce tutoriel va vous expliquer comment positionner en CSS un élément de manière fixe, aussi bien sur Firefox ou les autres navigateurs alternatifs que sur Internet Explorer 6 et 7.

1. Préambule

S'il est facile de positionner de manière fixe des éléments sur les navigateurs comme Opera, Safari, Firefox ou IE7 qui implémentent en grande partie les propriétés CSS ([Lien23](#)) issues des spécifications CSS 2 ([Lien24](#)), ce n'est pas le cas sur IE6. Il existe cependant des méthodes pour simuler cette position fixe sur ce dernier, et nous allons voir l'une d'entre elles dans ce tutoriel.

Remarque : Cette méthode n'est fonctionnelle que si Javascript est activé sur le navigateur Internet Explorer 6 du visiteur.

De plus, on estime qu'un DOCTYPE complet et valide ([Lien25](#)) est utilisé dans le document dans lequel le code à venir sera placé. Sans cela, le code ne sera pas fonctionnel.

Dans les exemples suivants, on sous-entend que le code exclusif à IE6 est placé dans une feuille de style à part, incluse à l'aide de commentaires conditionnels ([Lien26](#)).

Aussi, on positionnera cet élément, les codes suivants ne venant que compléter celui-ci :

Elément utilisé tout au long du tutoriel

```
<div id="fixe">
  <p>Du texte</p>
</div>
```

Style correspondant

```
#fixe
{
  background      : #e99898;
  height          : 70px;
  border-bottom   : 2px solid #d07171;
  width           : 100%;
}

#fixe p
{
  margin          : 0;
}
```

Ce tutoriel est très inspiré d'un billet paru à cette adresse : Making Internet Explorer use position: fixed; ([Lien27](#))

2. Les différents types de positionnement

2.1. Position fixe en haut

Pour positionner notre élément en haut, il va falloir utiliser la propriété *top*, même si la valeur sera différente sur IE6.

Voici le code :

Position haut

```
#fixe
{
  position        : fixed;
  top             : 0;
}
```

Et voici le code pour IE6 :

Position haut IE6

```
#fixe
{
  position        : absolute;
  top             : expression( ignore =
document.documentElement.scrollTop + "px" );
}
```

Voir le résultat ([Lien28](#))

2.2. Position fixe en bas

Pour positionner notre élément en bas, il va falloir utiliser la propriété *bottom*. En revanche, on utilisera à nouveau *top* pour IE6.

Voici le code :

Position bas

```
#fixe
{
  position        : fixed;
  bottom          : 0;
}
```

Et voici le code pour IE6 :

Position bas IE6

```
#fixe
{
  position        : absolute;
  top             : expression( ignore = -
this.offsetHeight +
document.documentElement.clientHeight +
document.documentElement.scrollTop + "px" );
}
```

Voir le résultat ([Lien29](#))

2.3. Position fixe au centre

Pour positionner notre élément au centre, il va falloir utiliser la propriété *top*, et encore une fois, sa valeur sera différente sur IE6. La seule différence avec les deux exemples précédents est qu'il faut appliquer en plus une marge négative.

Voici le code :

Position centre

```
#fixe
{
  position        : fixed;
  top             : 50%;
  margin-top      : -36px; /* Correspond à moins la
hauteur totale de l'élément fixe (ici 70px + 2px de
bordure) divisée par 2 */
}
```

Et voici le code pour IE6 :

Position centre IE6

```
#fixe
{
    position      : absolute;
    top           : expression( ignore = -
this.offsetHeight +
document.documentElement.clientHeight / 2 +
document.documentElement.scrollTop + "px" );
    margin-top    : 36px;
}
```

Voir le résultat ([Lien30](#))

Même si cet exemple en lui-même n'est pas très usuel, on peut imaginer plein de cas d'utilisation, comme une fausse popup modale centrée.

3. Exemple de page avec header et footer fixes

Pour avoir un entête et un pied de page fixes, on va utiliser dans le même document les deux premières méthodes précédemment citées.

On ne peut évidemment pas leur donner le même nom, donc voici à quoi pourrait ressembler le code HTML :

Exemple de code HTML pour header et footer fixes

```
<div id="fixe-haut">
    <p>Un très joli bandeau que nous avons là !</p>
</div>

<div id="content">
    <p>Plein de contenu très intéressant ici.</p>
</div>

<div id="fixe-bas">
    <p>Ici le copyright</p>
</div>
```

En revanche, on va garnir un peu le code CSS pour que notre site ait un peu d'allure :

Code CSS correspondant

```
body
{
    margin      : 0;
    background  : url(fond-body.gif) fixed repeat-x
0 20px;
}

#fixe-haut
{
    background  : #6c123c url(fond-head.gif)
repeat-x;
    height      : 70px;
    position    : fixed;
    top         : 0;
    width       : 100%;
    left        : 0;
}

#fixe-bas
{
    background  : #f0f0f0;
    height      : 40px;
    border-top  : 2px solid #dedede;
    position    : fixed;
    bottom     : 0;
    width       : 100%;
}
```

```
left          : 0;
}

#fixe-haut p
{
    width       : 900px;
    margin      : 0 auto;
    color       : #ccc;
    font        : bold italic 25px times new roman,
serif;
    padding-top : 25px;
}

#fixe-bas p
{
    text-align  : center;
    color       : #a7a7a7;
    font        : normal 11px verdana, sans-serif;
    padding-top : 12px;
    margin      : 0;
}

#content
{
    padding     : 70px 0 42px 0;
    width       : 900px;
    margin      : auto;
    font        : normal 12px verdana, sans-serif;
}
```

Et les correctifs pour IE6 :

Code CSS correspondant

```
#fixe-haut
{
    position    : absolute;
    top         : expression( ignore1 =
document.documentElement.scrollTop + "px" );
}

#fixe-bas
{
    position    : absolute;
    top         : expression( ignore2 = -
this.offsetHeight +
document.documentElement.clientHeight +
document.documentElement.scrollTop + "px" );
}
```

Voir un exemple complet ([Lien31](#))

4. Aller plus loin...

- Sur IE6, lors du défilement de la page, vous aurez sans doute remarqué que le décalage n'est pas très fluide. Il existe un moyen très simple de corriger ce problème. Il suffit en effet de mettre un fond fixe au body :

Correction fond fixe

```
body
{
    background  : url(blank.gif) fixed;
}
```

- Dans les exemples précédents, notre élément prenait toute la largeur de l'écran. On peut imaginer avoir un élément fixe ne prenant qu'une partie de l'écran en largeur, en renseignant la propriété CSS *width* avec une autre valeur que 100%, disons par exemple 500px et

comme on ne le veut pas au bord, on renseignera également la propriété *left* avec une autre valeur que 0 :

Element fixe centré

```
#fixe
{
    width          : 500px;
```

```
left             : 50%;
height           : 300px;
top              : 0;
margin-left     : -250px;
background      : #ccc;
```

Retrouvez l'article de Josselin Willette en ligne : [Lien32](#)

Les dernières news

Qt 4.5 sous licence LGPL

Nokia vient d'annoncer que **Qt**, son outil de développement multi-plateformes d'interfaces utilisateur et d'applications de bureau et en environnement embarqué, va être disponible sous la licence Open-Source **LGPL 2.1** à partir de la version **4.5** de Qt, prévue pour **Mars 2009**. Auparavant, Qt était à disposition de la communauté Open-Source sous licence GPL. De plus, **Qt change de domaine**, passant de www.trolltech.com ([Lien33](#)) à www.qtsoftware.com ([Lien34](#)).

Le changement de licence au profit de la **LGPL** va être accessible aux **développeurs Open-Source et Commerciaux** avec plus de **souplesse** que la GPL et ainsi accroître la **flexibilité** des développements. De plus, les **dépôts du code source de Qt** vont être **accessibles publiquement** et les **contributions** de la part des communautés de développeurs d'applications de bureau et embarqué vont être **encouragées**. Avec ces changements, les développeurs pourront être plus **actifs** dans le développement et l'évolution de la plateforme Qt.

Qt 4.5 et ses futures versions seront aussi disponibles sous licence commerciale. Les licences des versions antérieures restent inchangées. De plus, le service offert par Qt sera étendu afin d'assurer que tout les projets de développement Qt puissent avoir accès au **même niveau de support**, quelque soit la licence choisie.

"Une plus large utilisation de Qt par les plus grandes entreprises vas permettre d'avoir de **meilleurs retours** et d'augmenter le nombre de contributions, permettant à **Qt** de continuer à être la **référence** en terme d'outil de développement d'interfaces graphiques et d'applications multi-plateformes. Cette accélération du développement de Qt va permettre aux développeurs, incluant Nokia, de **réaliser de meilleurs produits, réduire les temps de production et élargir le panel de diffusion de leurs solutions**", a déclaré Sebastian Nyström, Vice President de la division Qt Software chez Nokia ([Lien35](#)).

"Nokia apporte d'importantes **contributions aux communautés Open-Source** à travers le travail continu avec **Qt**, sa contribution au sein de **Symbian OS et S60** pour la **Symbian Foundation** et le développement sur la plateforme ouverte **Maemo**" déclare Kai Öistämö, Vice Président Exécutif de la division Matériel de chez Nokia. "L'utilisation de Qt, S60 sur Symbian OS, Maemo et les services OVI va permettre à chacune des plateformes de **profiter**

de la puissance de Qt, sans réécriture du code source. En ouvrant le système de licence de Qt et en encourageant les contributions, les utilisateurs de Qt vont avoir une **plus grande implication** dans le développement de Qt, ce qui va ainsi permettre une plus grande adoption de l'outil."

"La combinaison entre le framework applicatif multi-plateformes Qt et les logiciels **Freescale** offre aux développeurs OEM et d'applications une **liberté de choix unique** afin de sélectionner le chipset Freescale la plus adapté tout en permettant au développeur de **concevoir et maintenir un exemplaire basé sur un seul code de leur application**" explique Raja Tabet, Vice President of Solutions and Enablement Technology of Freescale "Le modèle LGPL est une **excellente et efficace option de licence** qui va permettre d'accélérer l'adoption et le déploiement de la plateforme combinant Freescale et Qt."

"Qt est utilisé intensivement dans les applications **Kubuntu et KDE**, et Canonical est très satisfait de voir ce fabuleux changement de modèle de licence" déclare Mark Shuttleworth, créateur du projet Ubuntu, "La nouvelle licence Qt va nous permettre de concevoir des applications encore plus 'souples' pour les utilisateurs. **L'investissement continu de Nokia** dans la **bibliothèque multi-plateformes Qt**, et la **plateforme Linux**, est un pilier majeur de l'innovation pour les **environnements de bureau et applications mobiles libres**."

"Nous félicitons Nokia pour avoir simplifié la licence de Qt" annonce Joe Miller, vice président de Platform & Technology Development, Linden Labs, "Nous avons trouvé Qt comme une **étonnante et novatrice technologie**, au delà de l'aspect licence, mais ce nouveau modèle a clairement simplifié notre décision d'utiliser **QtWebKit** pour son intégration dans **Second Life**."

"Le fait que Qt soit disponible sous les termes de la LGPL rationalise la conception d'applications utilisant les composants de KDE pour les applications basées sur Qt", déclare Sebastian Kügler, KDE e.V. Board Member, "Cette licence plus souple va clairement abaisser les barrières de l'adoption de Qt et des technologies KDE. **L'équipe KDE apprécie l'ouverture du processus de développement** et prévoit une **collaboration** plus intense entre **KDE et Qt Software**."

Commentez cette news en ligne : [Lien36](#)

Tutoriel Boost.Optional

Cet article présente la bibliothèque Boost.Optional de la famille de bibliothèques Boost.

1. Avant-propos

Boost se veut un ensemble de bibliothèques portables destinées à faciliter la vie du développeur C++. Boost se propose de construire du code de référence pouvant aussi, à terme, être incorporé au standard (STL). Pour plus d'informations, vous pouvez consulter :

- Les différents tutoriels Boost de [développez.com](#) : Tutoriels ([Lien37](#)).
- La F.A.Q Boost de [développez.com](#) : F.A.Q. ([Lien38](#)).
- Le site officiel de Boost : Boost ([Lien39](#)).
- Le site officiel de la bibliothèque : Boost.Optional ([Lien40](#)).

2. Introduction

2.1. Pré-requis

Avant toute chose, à l'instar de l'ensemble des bibliothèques Boost, le lecteur doit déjà être familier avec le langage C++ et plus particulièrement se sentir à l'aise avec les templates. La bibliothèque Boost.Optional utilise les bases de la programmation template. Il ne faut pas que celles-ci puissent être un obstacle à la compréhension.

L'utilisation de la bibliothèque Boost.Optional ne nécessite pas de connaissances particulières sur les autres bibliothèques Boost. Cependant, Boost.Optional s'interface avec la bibliothèque Boost.In Place Factory. Le tutoriel suivant présente cette bibliothèque : Tutoriel Boost.In Place Factory ([Lien41](#))

L'implémentation, quant à elle, s'appuie sur les bibliothèques suivantes :

- Boost.MPL
- Boost.Type Traits
- Boost.Utility.Assert

Sa compréhension nécessite donc d'avoir déjà côtoyé ces dernières.

L'implémentation utilise le concept de classes de traits et politiques ainsi que le concept de méta-programmation. Les tutoriels suivants présentent ces concepts :

- Présentation des classes de Traits et de Politiques en C++ ([Lien42](#)) ;
- La méta-programmation en C++ ([lien43](#)).

2.2. Objectifs

Deux idées maîtresses justifient Boost.Optional.

La première est liée au besoin d'avoir des variables à signification particulière représentant l'idée de valeur invalide. En formalisant le concept de valeur indéterminée, Boost.Optional permet de gérer les valeurs invalides sans avoir à étendre le champ des valeurs que peut prendre une donnée.

La seconde motivation tourne autour de la notion de variable non initialisée. La norme dit que l'utilisation d'une variable non initialisée est indéterminée. Or ce concept est en soi absent du langage. On ne peut pas faire de test pour savoir si une variable a

été initialisée ou non. Boost.Optional formalise le concept de variable non initialisée et permet de gérer cette indétermination.

2.3. Valeur invalide : une indétermination !

Considérons la fonction suivante (tirée de la documentation officielle) :

```
unsigned char get_async_input();
```

Cette fonction retourne l'octet disponible sur une interface, en mode asynchrone. Que faire si aucune donnée n'est disponible sur l'interface ? Plusieurs solutions sont couramment implémentées :

- l'utilisation d'une valeur spécifique du champ des valeurs possibles indiquant une valeur invalide :

```
static const unsigned char NO_DATA = (unsigned char)-1;
```

- l'extension du champ des valeurs possibles, les valeurs hors champ initial signifient l'invalidité :

```
unsigned int get_async_input();  
// Une valeur plus grande que 255 indique une  
valeur invalide.
```

- adjoindre un indicateur lié à l'état de validité de la valeur :

```
std::pair<bool,unsigned char> get_async_input();
```

Ces solutions souffrent d'inconvénients. La première solution n'est pas toujours possible : si toutes les valeurs du type sont significatives, ou si la fonction retourne un objet plus complexe dont on ne peut construire aisément une instance invalide. La seconde élargit le champ des valeurs en retour et rend moins immédiate la compréhension de ce résultat : difficile de se rendre compte que les seules valeurs valides sont les octets. La troisième apparaît un peu lourde.

Enfin, chacune de ces méthodes n'empêche pas l'utilisateur de traiter la valeur en retour comme une valeur valide. L'erreur peut passer inaperçue assez longtemps et devenir très difficile à diagnostiquer !

La norme précise que l'utilisation d'une variable non initialisée est indéterminée. C'est une notion très intéressante pour notre exemple. Avec les solutions proposées, la valeur en retour est initialisée, donc déterminée, mais elle revêt un caractère invalide par sa sémantique. En rendant notre valeur en retour non initialisée, de facto son utilisation est indéterminée, ce qui devient une erreur. La puissance conceptuelle de Boost.Optional est d'avoir marié cette notion d'indétermination avec celle de valeur invalide que peut prendre une variable ! Boost.Optional formalise la notion de variable non initialisée. Donc, avec Boost.Optional, la solution consiste à retourner une valeur invalide. Bien sûr, l'interface permet de diagnostiquer l'état de la variable pour une utilisation adéquate.

Une dernière parenthèse - toujours dans le fil de la documentation officielle - pour limiter notre champ d'application: le fait qu'une

valeur ne puisse être déterminée peut relever de l'erreur et doit être géré en tant que tel. Ainsi, `sqrt(-1)` est hors de notre propos car l'indétermination de la valeur en retour est directement liée au non respect de la pré-condition de la fonction : le paramètre n'est pas valide. Ce type d'erreur est habituellement géré selon les trois solutions suivantes :

- en disant que la valeur en retour de la fonction est indéterminée pour un paramètre incorrect (solution documentaire) ;
- par le biais d'une assertion en mode debug (diagnostique par test) ;
- par le biais d'une exception pour permettre au développeur de gérer cette erreur (gestion dans le programme).

Il est important de comprendre que les cas d'erreurs à l'origine d'une indétermination doivent être maintenus en tant que tels car ils restent conceptuellement la meilleure solution acceptable.

3. Premiers pas

3.1. Une variable `boost::optional`

La déclaration d'une variable se fait tout simplement comme suit :

```
boost::optional<Type> variable;
```

Cela donne par exemple :

```
boost::optional<int> opt_int;
boost::optional<std::string>opt_string;
boost::optional<CMaClasse> opt_ma_classe;
boost::optional<CMaClasse&> opt_ref_ma_classe;
```

3.2. Initialisation d'une variable `boost::optional`

La variable peut être initialisée à la construction :

```
boost::optional<int> opt_int(3); // opt_int est
initialisée à la valeur '3'
CMaClasse valeur(/*... paramètres du constructeur ...
*/);
boost::optional<CMaClasse> opt_ma_classe(valeur);
//opt_ma_classe est initialisée avec une copie de
'valeur'
```

Boost.In Place factory permet l'initialisation avec un constructeur adéquat :

```
class CMaClasse
{
public:
    CMaClasse(int vari_, std::string varstr_);
    // ... définition de CMaClasse
};
boost::optional<CMaClasse>
opt_ma_classe(boost::in_place(1,"chaîne"));
// Le constructeur CMaClasse(int vari_, std::string
varstr_) est utilisé pour
// initialiser la valeur
```

La variable peut aussi être explicitement initialisée à l'état invalide :

```
boost::optional<int> opt_int(boost::none);
```

Les différentes initialisations correspondent à des comportements différents :

```
boost::optional<CMaClasse> opt_ma_classe_1;
// Aucun constructeur de CMaClasse n'est appelé
// opt_ma_classe_1 est dans l'état invalide

boost::optional<CMaClasse> opt_ma_class_2(boost::none);
// Aucun constructeur de CMaClasse n'est appelé
// opt_ma_classe_2 est dans l'état invalide

boost::optional<CMaClasse>
opt_ma_class_3(CMaClasse(1,"chaîne"));
// Le constructeur par copie
CMaClasse::CMaClasse(CMaClasse const&) est appelé
// opt_ma_classe_3 contient une valeur valide

boost::optional<CMaClasse>
opt_ma_class_4(boost::in_place(1,"chaîne"));
// Le constructeur adéquat
CMaClasse::CMaClasse(int,std::string) est appelé
// opt_ma_classe_4 contient une valeur valide
```

3.3. Affecter une valeur

L'opérateur d'affectation permet de positionner la valeur d'une variable `boost::optional` :

```
boost::optional<int> opt_int;
opt_int = 3; // Initialisation avec une valeur

boost::optional<CMaClasse> opt_ma_classe;
opt_ma_classe = boost::in_place(1,"chaîne"); //
affectation avec un constructeur sur place

boost::optional<CMaClasse> opt_ma_classe_2;
opt_ma_classe_2 = CMaClasse(1,"chaîne"); // affectation
avec un constructeur par copie

boost::optional<CMaClasse> opt_ma_classe_3;
opt_ma_classe_3 = boost::none; // on force l'état
invalide
```

Le comportement de l'opérateur d'affectation diffère selon l'état de la variable :

```
boost::optional<CMaClasse> opt_ma_classe;
opt_ma_classe = CMaClasse(); // Appel du constructeur
par copie
// pour une variable dans l'état non initialisé
// ...
opt_ma_classe = CMaClasse(); // Appel de l'opérateur
'=' de CMaClasse
// pour une variable dans l'état initialisé

boost::optional<CMaClasse> opt_ma_classe2(CMaClasse());
opt_ma_classe2 = boost::none; // Appel du destructeur
de CMaClasse
// pour une variable initialisé.
// ...
opt_ma_classe2 = boost::none; // Aucun appel pour une
variable
// dans l'état non initialisé

boost::optional<CMaClasse> opt_ma_classe3;
opt_ma_classe3 = boost::in_place(1,"chaîne"); // Appel
du constructeur adéquat
// CMaClasse::CMaClasse(int, std::string) pour
une variable
// dans l'état non initialisé
// ...
opt_ma_classe3 = boost::in_place(42,"Autre
chaîne"); // Appel du destructeur
// puis de CMaClasse::CMaClasse(int,
std::string) pour une variable
// dans l'état initialisé
```

3.4. Vérifier l'état d'une variable

Une variable `boost::optional` peut être utilisée comme un booléen indiquant son état :

```
boost::optional<int> ma_variable(4);
if(ma_variable){
    std::cout<<"La variable est initialisée";
}
ma_variable = boost::none;
bool est_initialise = ma_variable;
if(!est_initialise){
    std::cout<<"La variable ne contient plus de valeur valide";
}
```

L'opérateur '!' donne la valeur inverse :

```
boost::optional<std::string> opt_string;
if(!opt_string){
    opt_string = "initialise";
}
if(!opt_string){
    std::cout<<"La variable n'est pas initialisée ?";
}
```

3.5. Récupérer la valeur

Une fois la variable `boost::optional` initialisée, la valeur est simplement récupérée avec l'opérateur '*':

```
boost::optional<int> opt_int(3);
if(opt_int){
    std::cout<<"La valeur est : "<< *opt_int;
}
```

Attention, l'accès à la valeur d'une variable dans l'état invalide aboutit à un comportement indéterminé :

```
boost::optional<int> opt_int;
std::cout<<"La valeur est : "<< *opt_int; // plantage assuré !
```

Il faut bien faire attention à ne pas oublier cette étoile magique sans quoi le compilateur cherche à évaluer la variable comme booléen :

```
boost::optional<int> opt_int;
std::cout<<opt_int<<std::endl; // Affiche 'faux' ou '0'
opt_int = 42;
std::cout<<opt_int<<std::endl; // Affiche 'vrai' ou '1'
std::cout<<*opt_int<<std::endl; // Affiche 42
```

Il faut bien distinguer les deux écritures différentes :

```
boost::optional<int> opt_int;
opt_int = 42; // on initialise la variable opt_int
*opt_int = 54; // on récupère l'entier associé et on en change sa valeur
```

La différence est plus flagrante avec une classe :

```
boost::optional<CMaClasse> opt_ma_classe;
opt_ma_classe = CMaClasse(); // Initialisation de
opt_ma_classe avec un constructeur par copie
*opt_ma_classe = CMaClasse(); // La valeur est
récupérée et l'opérateur d'affectation '='
// de CMaClasse est appelé
```

Une fois déréférencé, l'objet contenu s'utilise comme tous les autres objets du même type :

```
boost::optional<CMaClasse> opt_ma_classe(CMaClasse());
(*opt_ma_classe).MaMethode(); // Appel à
CMaClasse::MaMethode
```

La surcharge de l'opérateur '->' permet l'utilisation d'obtenir un pointeur sur l'objet contenu :

```
boost::optional<CMaClasse> opt_ma_classe(CMaClasse());
opt_ma_classe->MaMethode(); // Appel à
CMaClasse::MaMethode
```

Comme l'opérateur '*', l'appel à '->' sur un objet non initialisé provoque un comportement indéfini :

```
boost::optional<CMaClasse> opt_ma_classe;
opt_ma_classe->MaMethode(); // plantage assuré !
```

Pour terminer le tour de la question, signalons l'existence des fonctions équivalentes aux opérateurs décrits ci-dessus :

```
boost::optional<CMaClasse> opt_ma_classe(CMaClasse);
opt_ma_classe.get().MaMethode();
opt_ma_classe.get_ptr()->MaMethode();
```

`get_ptr` présente la particularité de ne pas planter si la variable n'a pas été initialisée mais retourne un pointeur NULL :

```
boost::optional<CMaClasse> opt_ma_classe;
if(opt_ma_classe.get_ptr()==NULL){
    opt_ma_classe = CMaClasse();
}
```

3.6. Comparer deux variables boost::optional

L'opérateur '==' permet de comparer deux variables `boost::optional` de même type :

```
boost::optional<int> opt_int1(1);
boost::optional<int> opt_int2(1);
if(opt_int1==opt_int2){
    ++*opt_int2;
}
```

La comparaison de deux variables `boost::optional` se fait sur leur état, et si besoin, sur leur valeur. Ainsi l'opérateur '==' du type de base est appelé si les deux opérandes sont dans l'état initialisé :

operator==	Etat indéterminé	Etat déterminé
Etat indéterminé	vrai	faux
Etat déterminé	faux	(*opt1)==(*opt2)

Notons que les opérateurs '!=', '<', '>', '<=' et '>=' sont aussi redéfinis. L'état indéterminé est considéré comme plus petit que l'état déterminé. Ensuite, si les deux opérandes sont dans l'état déterminé, c'est l'opérateur de la classe de base qui est invoqué :

operator <	Etat indéterminé	Etat déterminé
Etat indéterminé	faux (inférieur strict)	vrai
Etat déterminé	faux	(*opt1)<(*opt2)

3.7. Utilisation dans un conteneur

On peut utiliser `boost::optional` comme n'importe quel type :

```
std::vector<boost::optional<CMaClasse> >
vect_opt_ma_classe;
vect_opt_ma_classe.push_back(CMaClasse(1, "un"));
vect_opt_ma_classe.push_back(CMaClasse(2, "deux"));
vect_opt_ma_classe.push_back(CMaClasse(3, "trois"));
```

Attention, un élément du vecteur est de type `boost::optional` et non de type `CMaClasse`. Il faut déréférencer l'objet `boost::optional` pour retrouver l'élément contenu en veillant bien sûr à ce que l'objet soit dans l'état initialisé :

```
void Fonction(boost::optional<CMaClasse> const & opt_)
{
    if(opt_){
        opt_->MaMethode();
    }
}
int main()
{
    std::vector<boost::optional<CMaClasse> >
vect_opt_ma_classe;
    vect_opt_ma_classe.push_back(CMaClasse(1, "un"));
    vect_opt_ma_classe.push_back(CMaClasse(2, "deux"));
    vect_opt_ma_classe.push_back(CMaClasse(3, "trois"));

    std::for_each(vect_opt_ma_classe.begin(), vect_opt_ma
_classe.end(), Fonction);

    return 0;
}
```

4. Quelques exemples

4.1. Valeur en retour indéterminée

Commençons par un exemple de fonction s'appuyant sur `boost::optional` pour gérer une valeur en retour signifiant que le résultat n'a pu être déterminé :

```
boost::optional<Client>
CMaBase::TrouverExact(std::string const &nom_) const
{
    if(ExisteClient(nom_)){
        return boost::optional<Client>(LireClient(nom_));
    }
    // on retourne le client

    return boost::optional<Client>();// on retourne une
instance non initialisée !
}
bool CMaBase::ExisteClient(std::string const &nom_)
const
{
    // ...
}
Client CMaBase::LireClient(std::string const &nom_)
const
{
    // ...
}
```

Lorsqu'il existe une entrée dont le nom coïncide avec celui donné en paramètre, alors la valeur en retour est initialisée avec l'enregistrement lu. Sinon, la méthode retourne une instance de `boost::optional` dans l'état non initialisé.

L'utilisation est alors :

```
bool CVueClient::DemanderAffichageFiche()
{
    std::string nom = DemanderNom();
    boost::optional<Client>
le_client(CMaBase::TrouverExact(nom));
    bool b_trouve = le_client;// prend la valeur vrai si
la variable contient une valeur
    if(b_trouve){
        // Affichage de client
        AffichageFiche(*le_client); // on déréférence
pour obtenir la valeur
    }
    else{
        AvertirOperateur(nom, ERR_Client_Inconu);
    }
    return b_trouve;
}
void CVueClient::AffichageFiche(Client const
&le_client_)
{
    // procédure d'affichage...
}
```

4.2. Variable locale non initialisée

Une variable `boost::optional` peut simplifier l'écriture d'une méthode en tirant bénéfice de l'état non initialisé.

```
void DlgAvancement::OnEvtAvancement(Evenement evt_)
{
    boost::optional<bool> optAffichage;
    switch(evt_.type){
        case Evenement::E_Demarrage:
            SetRange(evt_.min, evt_.max);
            optAffichage = true;
            break;
        case Evenement::E_Avancement:
            StepIt();
            optAffichage = boost::none;
            break;
        case Evenement::E_Terme:
            SetPosition(evt_.max);
            optAffichage = false;
            break;
        default:
            ASSERT(false);
            break;
    }
    if(optAffichage){// on teste si la valeur est
initialisée.
        Show(*optAffichage);// on déréférence pour
obtenir la valeur booléenne.
        UpdateControles();
        Layout();
    }
}
```

Dans cet exemple, on veut effectuer un ensemble d'action pour deux cas précis. Pour les autres cas, la variable `boost::optional` utilise l'état non initialisé.

4.3. Membre d'une classe optionnel

Un membre peut ne pas avoir de valeur. `boost::optional` évite l'utilisation d'un pointeur avec tous les risques que cela peut comporter :

```
struct Adresse
{
    int numero;
    std::string nom_rue;
```

```

std::string code_postal;
std::string ville;
boost::optional<std::string> cedex;
};

```

L'état invalide signifie que l'adresse ne possède pas de cedex.

Une autre utilisation d'un membre `boost::optional` se présente dès qu'un membre a une durée de vie inférieure à celle de la classe le contenant. `boost::optional` gère la composition sans pointeur :

```

class wxMaVue
{
...
protected:
    boost::optional<wxTimer> m_timer;
};

wxMaVue::DemarrerTimer()
{
    m_timer = boost::in_place<wxTimer>();// si le membre
    était initialisé,
    // le destructeur est appelé arrêtant ainsi
    l'ancien timer
    // un nouveau timer est créé
    m_timer->Start(m_intervalle);
    ...
}

wxMaVue::ArreterTimer()
{
    m_timer = boost::none;// si m_timer est initialisé
    // le destructeur est appelé arrêtant le timer en
    cours
    ...
}

```

4.4. Réassigner une référence

L'utilisation d'une référence avec `boost::optional` est possible. Dès lors, la réinitialisation de la variable correspond à une réassignation de référence :

```

int main()
{
    int val_1 (5);
    int val_2 (42);

    boost::optional<int&> opt_ri;
    opt_ri = val_1;// opt_ri est liée à val_1
    std::cout<<val_1<<" "<<*opt_ri<<"
"<<val_2<<std::endl;
    *opt_ri = 12;// on change la valeur de val_1
    std::cout<<val_1<<" "<<*opt_ri<<"
"<<val_2<<std::endl;
    *opt_ri = val_2; // on change la valeur de val_1 qui
    vaut maintenant celle de val_2
    std::cout<<val_1<<" "<<*opt_ri<<"
"<<val_2<<std::endl;
    opt_ri = val_2;// on réassigne opt_ri : opt_ri est
    liée à val_2
    *opt_ri = 5; // on change la valeur de val_2
    std::cout<<val_1<<" "<<*opt_ri<<"
"<<val_2<<std::endl;

    return 0;
}

```

5. Description de la bibliothèque

5.1. En-tête

Commençons par l'essentiel : les fichiers d'en-têtes nécessaires !

- `boost/optional.hpp`
- `boost/optional/optional.hpp`
- `boost/none.hpp`

Concernant les deux premiers en-têtes, le principe de Boost veut qu'un en-tête global existe sous la racine de 'boost/' pour une bibliothèque. Les en-têtes présents dans un répertoire particulier permettent de n'utiliser qu'une partie spécifique de la bibliothèque. `Boost.Optional` ne fournissant qu'un seul service, l'en-tête à la racine se contente d'inclure le second en-tête.

L'utilisation du troisième en-tête est expliquée ci-dessous.

5.2. Espace de nommage

L'espace de nommage est :

```

namespace boost

```

5.3. Interface

5.3.1. La classe

Classe optional :

```

template<class T>
class optional
{
public :
    // Constructeurs :
    optional () ;
    optional ( boost::none_t ) ;
    optional ( T const& v ) ;
    optional ( bool condition, T const& v ) ;
    optional ( optional const& rhs ) ;
    template<class U> explicit optional ( optional<U>
const& rhs ) ;
    template<class InPlaceFactory> explicit optional
( InPlaceFactory const& f ) ;
    template<class TypedInPlaceFactory> explicit
optional ( TypedInPlaceFactory const& f ) ;

    // Opérateur d'affectation :
    optional& operator = ( boost::none_t ) ;
    optional& operator = ( T const& v ) ;
    optional& operator = ( optional const& rhs ) ;
    template<class U> optional& operator =
( optional<U> const& rhs ) ;
    template<class InPlaceFactory> optional& operator =
( InPlaceFactory const& f ) ;
    template<class TypedInPlaceFactory> optional&
operator = ( TypedInPlaceFactory const& f ) ;

    // Accesseur
    T const& get() const ;
    T& get() ;
    T const& get_value_or( T const& default ) const ;
    T const& operator *() const ;
    T& operator *() ;
    T const* get_ptr() const ;
    T* get_ptr() ;
    T const* operator ->() const ;
    T* operator ->() ;

    // Diagnostique de l'état
    operator unspecified-bool-type() const ;
    bool operator!() const ;
};

```

T peut être un type ou une référence :

Exemple de variable optional :

```
boost::optional<int> mon_entier;
boost::optional<CMaRessource&> ma_ressourc;
```

5.3.2. Convention

Certaines définitions sont légèrement différentes selon que la classe est instanciée avec une référence ou pas. Les définitions ci-dessous utilisent par convention la notation suivante :

- `optional<T>` : la définition s'applique sur le type (`optional<T>`) comme sur une référence (`optional<T&>`);
- `optional<non_ref(T)>` : la définition s'applique pour le type (`optional<T>`) mais pas sur une référence (`optional<T&>`);
- `optional<T&>` : la définition s'applique pour une référence (`optional<T&>`) mais pas pour le type (`optional<T>`).

5.3.3. Les constructeurs

Constructeur par défaut :

```
optional<T>::optional();
```

- **Objectif** : Constructeur par défaut.
- **Post-condition** : L'état est non-initialisé.
- **Exception(s)** : Aucune.
- **Exemple** :

```
optional<T> ma_variable
```

- **Notes** : Le constructeur de T *n'est pas appelé*.

Constructeur non-initialisé :

```
optional<T>::optional(boost::none_t);
```

- **Objectif** : Constructeur d'un objet non initialisé.
- **Post-condition** : L'état est non-initialisé.
- **Exception(s)** : Aucune.
- **Exemple** :

```
optional<T> ma_variable(boost::none);
```

- **Notes 1** : Le constructeur de T *n'est pas appelé*.
- **Notes 2** : `boost::none` de type `boost::none_t` est défini dans `boost/none.hpp`.

Constructeur initialisé :

```
optional<non_ref(T)>::optional(T const& _copie);
```

- **Objectif** : Constructeur d'un objet initialisé.
- **Post-condition** : L'état est initialisé et la valeur est une copie du paramètre.
- **Exception(s)** : Celle(s) que lève le constructeur par copie de `T::T(T const&)`;
- **Exemple** :

```
T mon_element;
```

```
optional<T> ma_variable(mon_element)
```

- **Notes** : Le constructeur `T::T(T const&)` est appelé.

Constructeur initialisé :

```
optional<T&>::optional(T & _reference);
```

- **Objectif** : Constructeur d'un objet initialisé.
- **Post-condition** : L'état est initialisé et la valeur est la référence fournie en paramètre. Il s'agit bien de la valeur de la référence et non une copie de l'objet.
- **Exception(s)** : Aucune.

• Exemple :

```
T mon_element;
T& ma_reference = mon_element;
optional<T&> ma_variable(ma_reference);
```

- **Notes** : Le constructeur de T *n'est pas appelé*.

Constructeurs conditionnels :

```
optional<non_ref(T)>::optional(bool _b_initialise, T const& _copie);
```

```
optional<T&>::optional(bool _b_initialise, T & _reference);
```

- **Objectif** : Constructeurs conditionnels.
- **Post-condition** : Si `_b_initialise` s'évalue à `false`, l'état est non-initialisé. Si `_b_initialise` s'évalue à `true`, l'état est initialisé ; la valeur est une copie du paramètre pour le premier cas, une copie de la référence dans le second.
- **Exception(s)** : cf. Note.
- **Exemple** :

```
T mon_element;
```

```
T& ma_reference = mon_element;
```

```
optional<T&> ma_variable(true,ma_reference);
```

```
optional<T> mon_autre_variable(false,mon_element);
```

- **Notes** : Si la condition s'évalue à `true`, alors ces constructeurs sont équivalents à `optional<non_ref(T)>::optional(T const& _copie)` respectivement `optional<T&>::optional(T & _reference)`. Si la condition vaut `false`, alors les constructeurs sont équivalents à `optional<T>::optional(boost::none_t)`.

Constructeur par copie :

```
optional<non_ref(T)>::optional(optional const& _copie);
```

- **Objectif** : Constructeur par copie.
- **Post-condition** : Si l'état de `_copie` est initialisé, alors l'état de `*this` est initialisé et la valeur est une copie de la valeur de `_copie`. Si l'état de `_copie` est non-initialisé, alors l'état de `*this` est non-initialisé.
- **Exception(s)** : Si l'état de `_copie` est initialisé, alors les exceptions levées sont celles de `T::T(T const &)`.
- **Exemple** :

```
T mon_element;
```

```
optional<T> ma_variable(mon_element);
```

```
optional<T> mon_autre_variable(ma_variable);
```

- **Notes** : Si l'état de `_copie` est initialisé, alors le constructeur de copie de T est appelé.

Constructeur par copie :

```
optional<T&>::optional(optional const& _copie);
```

- **Objectif** : Constructeur par copie.
- **Post-condition** : Si l'état de `_copie` est initialisée, alors l'état de `*this` est initialisé et la valeur est la même référence que `_copie`. Si l'état de `_copie` est non-initialisé, alors l'état de `*this` est non-initialisé.
- **Exception(s)** : Aucune.
- **Exemple** :

```
T mon_element;
```

```
T& ma_reference = mon_element;
```

```
optional<T&> ma_variable(ma_reference);
```

```
optional<T&> mon_autre_variable(ma_variable);
```

- **Notes** : Le constructeur de T *n'est pas appelé*.

Constructeur de conversion :

```
template<class U>
optional<non_ref(T)>::optional(optional<U> const&
_copie);
```

- **Objectif** : Constructeur de conversion.
- **Post-condition** : Si l'état de *_copie* est initialisé, alors l'état de **this* est initialisé et sa valeur est une copie de la valeur de *_copie* convertie en T : $T::T(U \text{ const } \&)$. Si l'état de *_copie* n'est pas initialisé, alors l'état de **this* n'est pas initialisé.
- **Exception(s)** : Si l'état de *_copie* est initialisé, alors les exceptions levées sont celles de $T::T(U \text{ const } \&)$.
- **Exemple** :

```
U mon_element;
optional<U> ma_variable(mon_element);
optional<T> mon_autre_variable(ma_variable);
```

- **Notes** : Si l'état de *_copie* est initialisé, alors le constructeur $T::T(U \text{ const } \&)$ est appelé.

Constructeur In Place Factory :

```
template<class InPlaceFactoryT>
    optional<non_ref(T)>::optional(InPlaceFactoryT
_usine);
template<class TypedInPlaceFactoryT>
    optional<non_ref(T)>::optional(TypedInPlaceFact
oryT _usine);
```

- **Objectif** : Constructeur avec un objet construit en place.
- **Post-condition** : l'état de **this* est initialisé et sa valeur est celle résultant de sa construction avec l'usine.
- **Exception(s)** : Celle(s) levée(s) par le constructeur appelé par l'usine.
- **Exemple** :

```
optional<T> ma_variable(boost::in_place(p0, p1, p2));
optional<T>
mon_autre_variable(boost::in_place<T>(p0,p1,p2);
```

- **Notes** : Pour plus d'informations sur la bibliothèque Boost.In Place factory, vous pouvez consulter : Tutoriel Boost.In Place Factory ([Lien41](#))

Retrouvez la suite de l'article de 3DArchi en ligne : [Lien44](#)

Les derniers tutoriels et articles

Curseurs de réglage de type "Facebook" avec Mootools et les feuilles de styles CSS

Cet article vous montre comment mettre en place facilement des curseurs de réglage de type "Facebook" en utilisant les technologies JavaScripts (Mootools) et les feuilles de style.

1. Curseurs de réglage de type "Facebook" avec Mootools et les feuilles de styles CSS

L'un des intérêts que trouve [David Walsh] à être un développeur utilisant Facebook est qu'il peut y trouver de bonnes idées pour rendre un site plus interactif. Facebook est composé de nombreuses fonctionnalités avancées AJAX et JavaScript : Chargement de photos par des flèches droites et gauches, des menus déroulants, fenêtres d'alertes et des curseurs de réglage. Un tel curseur permet à l'utilisateur de faire glisser un bouton sur un segment pour paramétrer une valeur en lieu et place d'un ennuyeux et fastidieux champ texte. [David Walsh a] créé un très simple système composé de trois curseurs qui permettent à l'utilisateur de modifier la hauteur, la largeur et la transparence d'une image. Voici ce que cela donne.



Le rendu final est une image précédée de trois curseurs de réglage

1.1. Première étape : Le script CSS

```
#opacity-area, #width-area, #height-area {
  background:url(horizontal.jpg) 0 8px no-repeat;
  height:23px;
  width:500px;
  margin:0 0 5px 0;
}
#opacity-slider, #width-slider, #height-slider {
  background:url(button-horizontal.jpg) no-repeat;
  width:33px;
  height:23px;
  cursor:pointer;
}
```

Il y a très peu de code CSS impliqué. [David Walsh] a utilisé deux images comme arrière-plan pour les curseurs:

L'image utilisée comme  curseur

L'image utilisée comme barre d'attache du curseur

1.2. Deuxième étape : Le code XHTML

```

<div id="opacity-area">
  <div id="opacity-slider"></div>
</div>
<div>
  <strong>Opacité:</strong> <span id="opacity-label"></span>%
</div>
<br /><br />
<div id="width-area">
  <div id="width-slider"></div>
</div>
<div>
  <strong>Largeur:</strong> <span id="width-label"></span> pixels
</div>
<br /><br />
<div id="height-area">
  <div id="height-slider"></div>
</div>
<div>
  <strong>Auteur:</strong> <span id="height-label"></span> pixels
</div>
<br /><br />
<p><i>La muse, comme toujours, est Christina Ricci.</i></p>
```

Une fois de plus, très peu de code XHTML est utilisé. Le code ne présente que des conteneurs [NDLT : sous forme de balise <div>] pour les différents éléments composant chaque curseur.

1.3. Troisième étape : Le code Javascript utilisant MooTools

```
window.addEvent('domready', function () {
  /* opacity slider */
  var mySlide = new Slider($('opacity-area'), $('opacity-slider'), {
    steps: 100,
    wheel: 1,
    onChange: function(step) {
      $('opacity-label').setHTML(step);
      $('muse').set('opacity', step / 100);
    }
  }).set(100);
  /* height slider */
```

```

var mySlide = new Slider($('height-area'), $
('height-slider'), {
    steps: 300,
    wheel: 1,
    onChange: function(step) {
        $('height-label').setHTML(step);
        $('muse').set('height', step);
    }
}).set(300);

/* width slider */
var mySlide = new Slider($('width-area'), $
('width-slider'), {
    steps: 300,
    wheel: 1,
    onChange: function(step) {
        $('width-label').setHTML(step);
        $('muse').set('width', step);
    }
}).set(300);
});

```

Ce code est très répétitif, ce qui, du coup, le rend très simple à comprendre. [David Walsh] a créé une nouvelle instance de la classe Slider pour chaque curseur qu'il voulait sur la page, et fournit [NDLT : comme paramètre au constructeur] les éléments "area" et "slider". Il a également fourni à la classe les valeurs bornes (valeur maximale - valeur minimale valant 0 par défaut), une valeur "roulette" qui permet de déplacer le curseur en utilisant la molette de la souris, et une fonction "onChange" qui modifie les valeurs de la feuille de style et ajuste la largeur, la hauteur ou l'opacité de l'image. Enfin, il utilise la fonction "set()" pour définir la valeur par défaut du curseur.

1.4. That's All Folks !

Ces curseurs peuvent être utilisés de bien des façons et [David Walsh] a choisi un tel exemple pour le rendre facilement utilisable [et pédagogique]. [David Walsh] propose, par exemple, d'utiliser les paramètres de chaque curseur afin de générer une nouvelle image à télécharger. Cependant c'est à vous de choisir comment utiliser de tels curseurs de réglages

Retrouvez l'article de David Walsh traduit par Alexandre Tranchant en ligne : [Lien45](#)

Lire un fichier XML avec jQuery

Encore une fois nous allons parler de jQuery et nous allons nous attarder à la lecture d'un fichier ou d'une source XML à partir de jQuery.

1. Introduction

Eh oui, encore une fois nous allons parler de jQuery, sans aucun doute une des meilleures bibliothèques JavaScript existantes. Bref, nous allons aujourd'hui nous attarder à la lecture d'un fichier ou d'une source XML à partir de jQuery. Alors qu'AJAX prend de plus en plus de place sur la toile voici comment vous aussi vous pouvez ajouter un petit plus à vos développements.

2. Étape préliminaire

Avant toute chose il faut construire notre page HTML et y inclure notre bibliothèque afin que le tout fonctionne. Voici le code du corps de notre page que nous utiliserons.

```

<body>
  <div id="Div_XML"></div>
</body>

```

3. La lecture du fichier XML

```

<sites>
  <site id="0">
    <url>http://www.pckult.net</url>
    <desc>
      <brief>Résumé</brief>
      <long>Description longue</long>
    </desc>
  </site>
  <site id="2">
    <url>http://www.microsoft.ca</url>
    <desc>
      <brief>Résumé</brief>
      <long>Description longue</long>
    </desc>
  </site>
  <site id="3">
    <url>http://www.intel.com</url>
    <desc>
      <brief>Résumé</brief>
      <long>Description longue</long>
    </desc>
  </site>
</sites>

```

```

</site>
</sites>

```

Comme toute programmation AJAX, on commence toujours par l'inclusion des tags suivants :

```

$(document).ready( function()
{
}
);

```

À l'intérieur même de ces tags, nous allons créer notre requête AJAX. La requête AJAX demande 4 paramètres distincts :

- Un type
- Une source (url)
- Un type de source
- L'opération à effectuer si la requête est valide

La partie la plus importante est bien sur la partie opération. C'est cette dernière qui va permettre de manipuler le fichier XML.

```

$.ajax( {
    type: "GET",
    url: "Fichier.xml",
    dataType: "xml",
    success: function(xml) { }
}
);

```

Une fois le fichier chargé, il nous faut maintenant lire son contenu. Nous allons donc utiliser la méthode **find()** afin de récupérer tous les éléments (dans le cas présent, les éléments « **site** ») de notre arbre XML.

```

$(xml).find('site').each( function(){ } );

```

Une fois que tous les éléments sont récupérés, il nous faut alors récupérer les données de chaque élément grâce à la fonction **attr()** et la fonction **find()**.

```

var id = $(this).attr('id');
var title = $(this).find('title').text();
var url = $(this).find('url').text();
$('<div class="items" id="link_' + id +
'"></div>').html('<a href="' + url + '">' + title + '</a>').appendTo('#Div_XML');

```

4. Code final

```

$(document).ready(
function()
{
$.ajax( {
type: "GET",
url: "sites.xml",
dataType: "xml",
success: function(xml)
{
$(xml).find('site').each(
function()
{
var id = $
(this).attr('id');
var title = $
(this).find('title').text();

```

```

var url = $
(this).find('url').text();
$('<div class="items"
id="link_' + id + '"></div>').html('<a href="' + url +
'">' + title + '</a>').appendTo('#Div_XML');
$(this).find('desc').each(
function()
{
var
brief = $(this).find('brief').text();
var
long = $(this).find('long').text();
$('<div
class="brief"></div>').html(brief).appendTo('#link_' + id
);
$('<div
class="long"></div>').html(long).appendTo('#link_' + id
);
});
});

```

Retrouvez l'article de Dave Lizotte en ligne : [Lien46](#)

Modifier l'opacité des éléments sélectionnés d'une liste avec Script.aculo.us

Cet article vous montrera comment modifier l'opacité d'un élément (X)HTML suite à une sélection grâce au Framework Script.aculo.us ([Lien47](#)).

1. Introduction

Script.aculo.us est un superbe framework permettant d'animer votre site web. Il est simple et en général le résultat est vraiment impressionnant. Voyons donc aujourd'hui comment modifier l'opacité d'un élément d'une liste lorsque celui-ci est sélectionné.



2. Inclure les librairies

Il faut tout d'abord commencer par inclure Script.aculo.us et Prototype. Insérez donc le code suivant entre vos balises <head></head>.

```

<script src="/scriptaculous/lib/prototype.js"
type="text/javascript"></script>
<script src="/scriptaculous/src/scriptaculous.js"
type="text/javascript"></script>

```

3. Le code HTML

Voici une liste de blocs <div> créée pour les besoins de l'exemple ci-dessous.

```

<div id="el1">

<span class="check">
<input type="checkbox" id="status1" value="0"
onclick="javascript:changeOpacity(1)">
</span>
<span class="title">Base de données: Terminer le
modèle de données</span>
<span class="desc">Compléter avec le DBA le modèle
de données</span>

```

```
</div>
```

Lorsque vous cliquez sur la case à cocher, l'événement **changeOpacity()** est déclenché. Celui-ci modifie l'opacité du bloc <div> de 100% à 30% et inversement. Vous pouvez ajouter un nombre infini de blocs <div>, pour autant que vous suiviez la même numérotation choisie pour vos *id* (el2, el3, el4, etc.). Par la suite, il vous suffira de modifier l'argument passé en paramètre à la fonction **changeOpacity(X)** en fonction du numéro du bloc <div> (el2 -> changeOpacity(2), el3 -> changeOpacity(3), etc.).

4. Le code Javascript

Maintenant, passons au code JavaScript. Il vous suffira d'inclure ce code entre les balises <head></head> de votre page.

```

<script type="text/javascript">
function changeOpacity(id)
{
$opacityStatus = $('<status' + id);
if($opacityStatus.value == 0)
{
// Si la case a été cochée, mettre à 30%
new Effect.Opacity('el' + id,
{duration:0.5, from:1.0, to:0.3});
$opacityStatus.value = 1;
}
else
{
// Si la case a été décocher, remettre à
100%
new Effect.Opacity('el' + id,
{duration:0.5, from:0.3, to:1.0});
$opacityStatus.value= 0;
}
}
</script>

```

Voir un exemple en ligne ([Lien48](#))

Retrouvez l'article de Dave Lizotte en ligne : [Lien49](#)

Les derniers tutoriels et articles

Intégrer la géolocalisation à Apache 2 sous Linux

Connaître la situation géographique des clients possède plus régulièrement un but statistique. Or, il peut s'avérer utile de pouvoir prendre en compte cet élément au niveau de la configuration du serveur même, ainsi que par les langages exécutés au niveau du serveur.

1. Présentation

Le module Apache GeoIP, développé et maintenu par la société MaxMind, vous permet non seulement de connaître la provenance de manière relativement fiable d'un visiteur mais de la faire intervenir dans vos développements ou encore dans la configuration du serveur.

La société MaxMind ne fournit pas seulement des modules pour les différentes versions du serveur web Apache mais aussi des bibliothèques pour divers langages (C - la base "commune", PHP, Perl, Java, etc). Cependant, une intégration au niveau du serveur lui-même présente quelques avantages et pré-requis.

L'unique condition préalable est de pouvoir accéder au serveur (donc de disposer des droits) pour pouvoir installer ce module et reconfigurer Apache.

Par contre, les enjeux sont multiples :

- Les informations relatives à la géolocalisation sont transmises sous forme de variable d'environnement au langage serveur employé. Leur récupération n'en est que plus aisée et ceci offre une certaine indépendance : il n'y a ainsi nul besoin d'installer une bibliothèque pour chaque langage que vous pourriez être amené à utiliser sur votre serveur ;
- Apache devient alors capable d'interagir en fonction de la localisation du client. Il peut ainsi directement effectuer un filtrage des clients, les rediriger, ...

2. Mise en place

Ici ne sera présentée que l'installation du module sous forme dynamique afin de pouvoir l'activer ou désactiver à volonté et pour une mise en place rapide (nous évitant une recompilation complète d'Apache).

2.1. A partir des sources

Il est nécessaire de télécharger trois choses avant de commencer :

- La bibliothèque C, dépendance requise au module que nous compilerons dans un deuxième temps :

```
cd ~
wget http://www.maxmind.com/download/geoip/api/c/GeoIP-1.4.5.tar.gz
```

Vérifier éventuellement qu'il n'existe pas de versions plus récentes que ce soit maintenant pour la bibliothèque C comme pour les sources du module que nous abordons ci-dessous.

- Les sources du module propres aux versions de la branche 2 d'Apache :

```
wget
http://www.maxmind.com/download/geoip/api/mod_geoip2/mod_geoip2_1.2.5.tar.gz
```

- Enfin, télécharger une base de données mise à disposition gratuitement (donc plus limitée). Il en existe deux : celle de base dont vous ne pourrez tirer que le pays du client et une seconde fournissant des informations plus précises puisque capable de vous donner la ville et la région de l'internaute.

```
# La base "pays" uniquement
wget
http://geolite.maxmind.com/download/geoip/database/GeoLiteCountry/GeoIP.dat.gz
```

```
# La base "villes allégée"
wget
http://www.maxmind.com/download/geoip/database/GeoLiteCity.dat.gz
```

La fiabilité des bases de données mises gratuitement à votre disposition est estimée à 98% pour celle des pays et à 69% en ce qui concerne celle des villes. Avec une perte supplémentaire d'environ 1,5% tous les mois si vous ne téléchargez pas sa nouvelle version paraissant de manière mensuelle.

Procédons à leur installation :

1. Comme mentionné plus haut, un certain ordre doit être respecté : il faut commencer par l'API C de GeoIP puisque le module repose sur celle-ci.
Décompressez l'archive :

```
tar xzf GeoIP-1.4.5.tar.gz -C /usr/local/src
```

Puis procédez à la compilation et installation (requérant les droits administrateur) :

```
cd /usr/local/src/GeoIP-1.4.5/
./configure --prefix=/usr/local/geoip
make
make install
```

Pour compiler cette bibliothèque vous devrez posséder sur votre système les fichiers d'entête de la bibliothèque zlib. Si ce n'est pas le cas, cherchez un paquet éponyme suffixé de *dev* ou *devel* puis installez-le.

2. Vient ensuite le tour du module Apache. Détarez les sources de celui-ci ...

```
tar xzf ~/mod_geoip2_1.2.4.tar.gz -C /usr/local/src
```

... puis passez à sa compilation.

```
cd /usr/local/src/mod_geoip2_1.2.4/  
apxs2 -i -a -L/usr/local/geoip/lib  
-I/usr/local/geoip/include -lGeoIP -c mod_geoip.c
```

Pour cette étape il vous sera nécessaire de posséder les fichiers d'entête d'Apache. Ceux qui ont donc installé leur serveur sous forme de paquet binaire auront sans doute à chercher un paquet nommé sur le modèle apache2(-threaded|mpm)-dev(el) ou encore httpd-dev(el).

Suivant votre système, la commande `apxs2` pourra être plus simplement appelée `apxs`.

La commande `apxs2`, telle qu'elle a été utilisée ci-dessus suppose que le répertoire où elle est située figure comme valeur de la variable d'environnement `PATH`. A défaut, il est possible d'en préciser son chemin.

3. Décompressez la base que vous avez récupérée plus tôt à l'aide de `gunzip` :

```
gunzip GeoIP.dat.gz  
# ou suivant la base choisie  
gunzip GeoLiteCity.dat.gz
```

Pour la déplacer vers un répertoire plus "approprié" :

```
cp ~/GeoIP.dat /usr/local/geoip/share/GeoIP/  
# ou suivant la base choisie  
cp ~/GeoLiteCity.dat /usr/local/geoip/share/GeoIP/
```

Pour la suite, assurez-vous que cette base est accessible en lecture de tous (surtout de l'utilisateur sous lequel fonctionne Apache).

Nous pouvons maintenant passer à la configuration d'Apache. Vérifiez tout d'abord la présence d'une ligne **LoadModule** ressemblant à celle ci-dessous (chemin à adapter) qui chargera notre nouveau module :

```
LoadModule geoip_module  
/usr/lib/apache2/modules/mod_geoip.so
```

Rajoutez-la à la suite des autres si elle se fait porter absente. Gardez le fichier ouvert en vue de placer une ligne `LoadFile` avant la précédente, nécessaire au bon fonctionnement du module, afin qu'Apache puisse auparavant charger la librairie `geoip` duquel dépend le module :

```
LoadFile /usr/local/geoip/lib/libGeoIP.so
```

Puis activez et configurez à présent notre module par la mention des lignes suivantes :

```
<IfModule mod_geoip.c>  
    GeoIPEnable On  
    GeoIPDBFile /usr/local/geoip/share/GeoIP/GeoIP.dat  
# ou /usr/local/geoip/share/GeoIP/GeoLiteCity.dat  
</IfModule>
```

A ce stade vous devriez relancer le serveur et vérifier le bon fonctionnement du tout en poursuivant directement à la partie intitulée Utilisation.

2.2. Exemple de Debian Etch pour une installation à partir des paquets binaires prévus

Dans la mesure où votre système prévoit des paquets pour le module GeoIP, vous avez tout intérêt à les privilégier : son installation n'en sera que plus rapide et plus facile.

Dans le cas de Debian, cela peut se résumer à la commande suivante :

```
aptitude install libapache2-mod-geoip
```

Debian vous installera uniquement la base de données "pays". Si vous prévoyez d'en utiliser une autre ou une version plus à jour, il vous faudra éventuellement décommenter la directive `GeoIPDBFile` du fichier `/etc/apache2/mods-available/geoip.conf` en retirant le caractère dièse (#) apparaissant en début de ligne et en modifier la valeur pour pointer sur la vôtre.

Normalement le module devrait être actif par défaut à l'issue de son installation. Assurez-vous en, en contrôlant la présence des liens symboliques `geoip.conf` et `geoip.load` dans le répertoire `/etc/apache2/mods-enabled` ou exécutez la commande `a2enmod`, qui le fera pour vous dans le cas contraire :

```
a2enmod geoip
```

Enfin, (re)démarrer le serveur web :

```
/etc/init.d/apache2 force-reload
```

3. Utilisation

Les variables d'environnement pour la base de données de type "pays" sont :

- `GEOIP_COUNTRY_CODE` : code international du pays (suivant la norme ISO 3166)
- `GEOIP_COUNTRY_NAME` : nom complet du pays (en anglais)

Si GeoIP est incapable de déterminer la provenance de l'utilisateur, les variables ne seront pas définies.

En ce qui concerne la base de données "allégée des villes", viennent s'ajouter à la liste précédente :

- `GEOIP_REGION` : un code indiquant la région (voir les valeurs plus bas ainsi que leur correspondance pour la France)
- `GEOIP_CITY` : la ville
- `GEOIP_DMA_CODE` : code DMA (Designated Market Area) attribué à une zone où les fréquences (télévision, radio, etc) sont identiques (concerne les Etats Unis, valeur par défaut : 0)
- `GEOIP_AREA_CODE` : indice téléphonique représentant une zone précise ([Lien50](#)) (concerne les Etats Unis, valeur par défaut : 0)
- `GEOIP_LATITUDE` : la latitude
- `GEOIP_LONGITUDE` : la longitude
- `GEOIP_POSTAL_CODE` : le code postal (concerne uniquement les Etats Unis, ne sera pas définie pour tout autre pays)

Les valeurs de la variable `GEOIP_REGION` pour les différentes régions françaises avec leur correspondance sont :

Région	Code
Alsace	C1
Aquitaine	97
Auvergne	98
Basse-Normandie	99
Bourgogne	A1
Bretagne	A2
Centre	A3
Champagne-Ardenne	A4
Corse	A5
Franche-Comté	A6
Haute-Normandie	A7
Île-de-France	A8
Languedoc-Roussillon	A9
Limousin	B1
Lorraine	B2
Midi-Pyrénées	B3
Nord-Pas-de-Calais	B4
Pays de la Loire	B5
Picardie	B6
Poitou-Charentes	B7
Provence-Alpes-Côte d'Azur	B8
Rhône-Alpes	B9

3.1. Par l'usage d'un langage de programmation côté serveur

La récupération des différentes "variables" créées par le module de géolocalisation dépendent de la manière dont ce dernier a été configuré. En effet, suivant la valeur de sa directive **GeoIPOutput**, elles ne seront pas accessibles de la même manière.

- Lorsque **GeoIPOutput** a pour valeur **Notes**, les variables relatives à la géolocalisation du client ne sont créées que dans la table interne d'Apache appelée *Notes*. Ce n'est pas, de ce fait, la méthode la plus facile pour ensuite les atteindre avec un langage serveur ;
- En revanche, avec la valeur **Env**, toutes ces variables sont exportées à la manière des variables d'environnement système. Cette forme facilite fortement leur usage, quel que soit le langage employé côté serveur, comme nous le verrons ci-dessous ;
- Enfin, la valeur **All** (par défaut), est à recommander puisque toutes ces variables seront définies deux fois, selon les méthodes des deux points précédents. Ainsi il n'y a plus lieu de se demander par quel moyen y avoir recourt.

3.1.1. PHP

Etant donné qu'il est plus courant d'utiliser PHP en tant que module Apache, vous disposez alors sans doute de l'extension apache ([Lien51](#)) qui fournit, entre autres, la fonction `apache_note` ([Lien52](#)). Ainsi que **GeoIPOutput** ait pour valeur *Notes* ou *All*, vous serez en mesure d'exploiter toutes les variables GEOIP.

Cependant, beaucoup trouveront certainement plus naturel de trouver ces informations dans le tableau superglobal `$_SERVER`. Chose possible dès lors que **GeoIPOutput** sera positionné à *All* ou *Env*.

Ci-dessous, un code PHP listant l'ensemble des variables GEOIP, dans l'hypothèse où elles vous sont mises à disposition comme des variables d'environnement :

```
foreach ($_SERVER as $k => $v) {
    if (strpos($k, 'GEOIP') !== FALSE) {
        printf('%s : %s<br />', $k, $v);
    }
}
```

3.1.2. Ruby

Par contre dans le cas de Ruby, il sera sans doute plus facile de trouver ces variables dans le hash `ENV`, recensant, comme son nom l'indique, toutes les variables d'environnement (ce qui implique que **GeoIPOutput** a pour valeur *Env* ou bien *All*) du fait de son utilisation CGI. Le code équivalent au script PHP présenté ci-dessus pourrait s'écrire de la sorte :

```
ENV.each do |k,v|
    puts "#{k} => #{v}" if k =~ /GEOIP/i
end
```

3.1.3. Perl

Pour des raisons autant de facilité que de portabilité, seule la méthode visant à exploiter l'exportation de ces variables comme variable d'environnement sera présentée (impliquant de nouveau **GeoIPOutput** à *Env* ou *All*). Vous les trouverez donc dans le hachage `%ENV` :

```
foreach (keys(%ENV)) {
    print "$_ => $ENV{$_}$/" if $_ =~ /GEOIP/i;
}
```

3.1.4. Autres langages

Nous vous invitons, pour ce qui est des autres langages, à consulter leurs documentations respectives ou à vous orienter sur les forums ([Lien53](#)).

3.2. Dans le cadre de la configuration du serveur seul

3.2.1. Première application : interdire l'accès à certains utilisateurs étrangers

Une première application, allant à l'encontre des principes d'accessibilité et de liberté d'Internet, pourrait être de bloquer des utilisateurs faisant partie de tel ou tel pays voir, pourquoi pas, de n'autoriser que ceux de pays prédéterminés.

En guise d'exemple, nous souhaitons bloquer les internautes russes et chinois. Se présentent deux possibilités :

- En introduisant une variable d'environnement :

```
SetEnvIf GEOIP_COUNTRY_CODE ^CN|RU$ FORBIDDEN=1
```

```
Deny from env=FORBIDDEN
```

Notre variable d'environnement *FORBIDDEN* ne sera créée que si l'on sait que le client est russe (code pays "RU") ou chinois (de code "CN") d'après le module GeoIP. Le fait que cette variable soit dans ces conditions définie, lui vaudra un blocage (le serveur retournera la page d'erreur 403 au lieu de la page attendue).

- A l'aide du module de réécriture :

```
RewriteEngine on
RewriteCond %{ENV:GEOIP_COUNTRY_CODE} ^CN|RU$
RewriteRule .* - [F]
```

Strictement la même chose, écrite avec des règles de réécriture. En réalité, le module a été quelque peu détourné de son but premier : le tiret fourni en deuxième paramètre de la directive RewriteRule indique qu'il ne faut pas effectuer de réécriture et c'est l'option F (pour forbidden, située entre les crochets, qui va retourner l'erreur 403 pour nos clients identifiés comme russes et chinois.

3.2.2. Deuxième application : redirection suivant le lieu de connexion

Pour ce second exemple d'application, l'objectif consiste à distinguer la provenance de l'utilisateur afin d'éventuellement rediriger celui-ci sur la page qui lui serait potentiellement la plus intéressante. Ici, nous supposons qu'il s'agit d'un site dédié à la région Corse (code associé : A5 - voir le tableau plus haut ; de domaine corse.fr).

Si l'utilisateur est reconnu comme étant local, nous le redirigerons sur la page *vie-pratique.html* fournissant diverses informations utiles et actualisées lorsqu'il demandera pour la première fois le document d'index (*index.html*). Ce premier passage sera marqué par un cookie, expirant après une heure, dans le but de ne pas entraver sa navigation.

On pourrait schématiser cette opération par l'algorithme suivant :

```
si GEOIP_REGION = "A5" et page_courante = "/index.html"
et non cookie_defini("local")
    alors
        créer_cookie("local", "1", ".corse.fr", 60,
"/")
/**
 * Paramètres :
 * - nom du cookie
 * - sa valeur
 * - le nom de domaine auquel il est associé
 * - durée en minutes avant expiration
 * - le chemin sur lequel il s'applique
 */
```

```
redirection("/vie-pratique.html")
```

```
fsi
```

Qui se traduirait à l'aide d'une règle de réécriture :

```
RewriteEngine on

RewriteCond %{ENV:GEOIP_REGION} =A5
# Règle n°1
RewriteCond %{HTTP_COOKIE} !\blocal\b
# Règle n°2
RewriteRule ^(:?index.html)?$ /vie-pratique.html
[L,CO=local:1:.corse.fr:60:] # Règle n°3
```

La première règle permet d'indiquer que seules les connexions établies depuis la région Corse, c'est-à-dire lorsque *GEOIP_REGION* est égal à *A5*, sont concernées.

La deuxième vérifie qu'un cookie nommé *local* n'a pas été créé. Si tel était le cas, on n'abandonnerait là la lecture de la règle, l'internaute ne serait pas redirigé.

Enfin l'opération de réécriture ou redirection, qui sera appliquée si les conditions précédentes ont été remplies et que le document demandé est la page d'index. A ce moment on en profite pour marquer la redirection en créant le cookie (option CO).

4. Epilogue

Bien qu'il existe de multiples solutions pour l'usage de l'"API" GeoIP, prenons pour preuve PHP qui se décline en une extension, en une bibliothèque écrite en PHP pur ou un paquet PEAR. Nous espérons, au travers de cet article, avoir montré tout l'intérêt de la déporter au sein du serveur web. D'autant plus, qu'un module est disponible pour chacune des branches d'Apache (l'installation du module pour une version 1.3 étant sensiblement identique).

Liens Développez en relation :

- Connaître le pays d'un visiteur en fonction de son adresse IP en PHP ([Lien54](#))

Liens externes :

- Le site de la société MaxMind à l'origine de ce module (entre autres) ([lien55](#))

Retrouvez l'article de Jean-Yves Lacour et julp en ligne : [Lien56](#)

Les derniers tutoriels et articles

Internationalisation des applications Flex

Cet article a pour but de vous présenter la gestion du multilinguisme dans une application Flex. L'application détecte automatiquement la langue de l'utilisateur pour en adapter, si la langue est gérée, son contenu. La langue par défaut est appliquée si l'application ne l'a pas détectée. Elle permet également le changement à la volée de la langue pour toute l'application en cours d'exécution.

1. Pré-requis

Avant de commencer, il faut tout d'abord, obtenir une copie locale du framework. En effet, le framework est proposé en anglais(En-US) il faut donc le copier pour créer un framework français (fr-FR).

Pour cela on ouvre une invite de commande et on se place dans le répertoire "bin" du SDK Flex (par défaut C:\Program Files\Adobe\Flex Builder 3\sdk\3.0.0\bin) :

déplacement vers le répertoire d'installation

```
cd C:\Program Files\Adobe\Flex Builder 3\sdk\3.0.0\bin
```

Puis, on copie le framework de base pour pouvoir l'utiliser dans une autre langue:

copie du framework

```
copylocale.exe en_US fr_FR
```

Si une erreur du type "unable to find JDK" ou "unable to find JVM" apparait, il faut ajouter une variable d'environnement JAVA_HOME (pour pointer vers le JDK):

- Panneau de configuration->Système, onglet Avancés, puis Variables d'environnements
- dans la zone "variables système" : cliquer sur Nouveau
- Nom de la variable : JAVA_HOME
- Valeur : répertoire vers le jdk (par exemple : C:\java\jdk1.4.2_17)
- si besoin redémarrer l'invite de commande et recommencer l'étape précédente.

2. Créer ses dictionnaires

Maintenant, nous pouvons commencer un nouveau projet que je nommerai localization. Dans les propriétés de ce projet, sur la partie Flex Compiler, nous rajoutons:

variables de compilation

```
-locale=fr_FR,en_US -source-path=./locale/{locale}
```

Ainsi, la langue par défaut est le français et à défaut de définition pour un composant du framework en français, il sera en anglais. De plus la seconde partie indique le chemin vers, ce que j'appelle, les dictionnaires. Il faut donc à présent créer ces dictionnaires. Nous créons donc un dossier "locale" à la racine du projet (au même niveau que src), puis dans ce dossier nouvellement créé on ajoute les dossiers en_US et fr_FR avec, à l'intérieur de chacun, un fichier i18n.properties (i18n étant l'abréviation de internationalisation, mais libre à vous de le nommer comme bon vous semble). Voici, le contenu de mon fichier i18n.properties (version FR) :

contenu d'un dictionnaire

```
#Commentaires bla bla
home_title=Bonjour!
title_window=Titre de la fenêtre
welcome_text=Bienvenue sur le composant 1
combo_text=C'est ici que vous pouvez changer la langue
button_text=Clique sur moi!
```

Vous l'avez remarqué, rien de compliqué, il suffit de copier-coller ce code dans le fichier anglais et de traduire... Pour les caractères non ASCII (é, è, ...), il faut sauvegarder le fichier français au format UTF-8, pour cela clic droit sur le fichier->propriétés et choisir Other dans Text field encoding, puis UTF-8.

3. Utilisation des dictionnaires dans un composant MXML

Maintenant, que les dictionnaires sont disponibles, comment y accéder ? C'est très simple, par défaut (référence à la variable de compilation) c'est le français qui est chargé, mais avec une ComboBox, par exemple, on peut passer d'une langue à l'autre pour cela, on utilise le resourceManager et sa propriété localeChain(Array) qui propage l'information à toute l'application. Sur l'évènement CHANGE de la ComboBox on a donc:

```
resourceManager.localeChain =
[myComboBox.selectedItem];
```

Il faut, pour que la modification soit effective, que les text et label de l'application récupère les valeurs depuis le resourceManager évidemment, voici comment opérer :

```
<!-- Dire au compilateur quelle ressource utiliser -->
<mx:Metadata>
    [ResourceBundle("i18n")]
</mx:Metadata>
```

```
<!-- Depuis un fichier MXML -->
<mx:Text text="{resourceManager.getString('i18n',
'welcome_text')}" />
```

```
//Depuis un composant AS (sous classe de UIComponent)
this.mybutton.label = resourceManager.getString('i18n',
'button_text');
```

```
//Code AS n'étant pas une sous classe de UIComponent
this.mybutton.label =
ResourceManager.getInstance().getString('i18n',
'button_text');
```

4. Utilisation des dictionnaires dans un composant AS

A présent, nous avons sans problème le changement à la volée pour les composants MXML mais pour les composants AS, il y a une petite subtilité. En effet, l'affectation du text ou du label doit

se faire dans la méthode `resourcesChanged` du `UIComponent`.
Voici un exemple de composant AS:

```
package
{
    import mx.containers.Canvas;
    import mx.controls.Button;

    public class myCompAS extends Canvas
    {
        public var mybutton:Button;

        public function myCompAS(){

        }
    }
}
```

```
        protected function init():void{
            this.resourcesChanged();
        }

        override protected function
resourcesChanged():void{
            if( this.mybutton != null ){
                this.mybutton.label =
resourceManager.getString('i18n', 'button_text');
            }
        }
    }
}
```

Une démonstration est disponible en ligne.

Retrouvez l'article de Jim Nastiq en ligne : [Lien57](#)


```

else
if @segment_name = 'indexsegment'
    select @v_device=@dbname+'_I01', @v_size='100M',
@v_type=''
else
if @segment_name = 'logsegment'
    select @v_device=@dbname+'_L01', @v_size='50M',
@v_type='LOG'
else
    begin
        raiserror 20001 'Type de segment non géré'
        return -1
    end

disk resize name = @v_device, size = @v_size

select @v_sql = 'alter database '+@dbname+' '+@v_type+'
ON '+@v_device+'='+@v_size+''''
print @v_sql
exec (@v_sql)
return 0
end
go

```

La procédure stockée suivante automatise la création d'une base de données selon les critères spécifiques et détermine le seuil critique.

```

user master
go
create procedure sp_auto_db_cree
( @v_dbname varchar ( 30 ))
as
begin

    declare @v_sql varchar ( 200 )
    declare @v_path varchar ( 200 )
    declare @v_file varchar ( 200 )
    declare @v_seg varchar ( 30 )
    declare @v_num_dev int
    declare @v_max_dev int

    /* Contrôle de la préexistence de la base */

    if exists (select name from sysdatabases where name
= @v_dbname)
        begin
            raiserror 20001 "La base existe déjà"
            return -1
        end

    /* Contrôle du nombre de devices configuré */

    select @v_max_dev=value from sysconfigures where
name = 'number of devices'
    select @v_num_dev=count(*) from sysdevices

    if @v_max_dev < @v_num_dev + 3 -- manque de
structures mémoire pour les devices
        begin
            select @v_num_dev=@v_max_dev + 3
            exec sp_configure 'number of devices',
@v_num_dev
        end

    /* Creation des devices spécifiques */

    select @v_path = left ( phynome , datalength
( phynome )- 10 ) from master..sysdevices where name =
'master'

```

```

    select @v_seg=@v_dbname+'_D01', @v_file=@v_path +
@v_dbname + '_DATA01.dat'
    disk init name= @v_seg, physname=@v_file ,
size='100M'

    select @v_seg=@v_dbname+'_I01', @v_file=@v_path +
@v_dbname + '_INDEX01.dat'
    disk init name= @v_seg, physname=@v_file ,
size='100M'

    select @v_seg=@v_dbname+'_L01', @v_file=@v_path +
@v_dbname + '_LOG01.dat'
    disk init name= @v_seg, physname=@v_file ,
size='50M'

    /* Creation de la base */
    select @v_sql = 'create database ' + @v_dbname + '
on ' + @v_dbname + '_D01='100M' log on ' + @v_dbname
+ '_L01='50M''''
    exec ( @v_sql )

    select @v_sql = 'alter database ' + @v_dbname + '
on ' + @v_dbname + '_I01='100M''''
    exec ( @v_sql )

    /* Attribution des segments */
    select @v_sql = @v_dbname+'..sp_addsegment
'indexsegment', '+@v_dbname+', '+@v_dbname+'_I01'
    exec (@v_sql)

    select @v_sql = @v_dbname+'..sp_dropsegment
'default', '+@v_dbname+', '+@v_dbname+'_I01'
    exec (@v_sql)

    select @v_sql = @v_dbname+'..sp_dropsegment
'system', '+@v_dbname+', '+@v_dbname+'_I01'
    exec (@v_sql)

    /* Délimitation des seuil d'alerte */

    select @v_sql = @v_dbname+'..sp_addthreshold
'+@v_dbname+', system, 5120, sp_auto_db_etend'
    exec (@v_sql)

    select @v_sql = @v_dbname+'..sp_addthreshold
'+@v_dbname+', default, 5120, sp_auto_db_etend'
    exec (@v_sql)

    select @v_sql = @v_dbname+'..sp_addthreshold
'+@v_dbname+', indexsegment, 5120, sp_auto_db_etend'
    exec (@v_sql)

    select @v_sql = @v_dbname+'..sp_addthreshold
'+@v_dbname+', logsegment, 5120, sp_auto_db_etend'
    exec (@v_sql)

return 0
end
go

```

Certains choix ont été fait afin de simplifier la démonstration :

- Localisation des devices dans un répertoire unique, à savoir celui hébergeant le device master : en terme de performances, il serait judicieux de choisir des disques distincts pour les divers segments
- Création arbitraire d'un device de 100M (50M pour les logs)
- Augmentation arbitraire d'un device de 100M (50M pour les logs)
- Trois segments distincts (data/system, log et index)

Il sera cependant aisément possible d'améliorer la procédure en "piochant" les informations modifiables via une table de

configuration.

Cette solution n'est pas optimum : elle a cependant le mérite de donner les pistes nécessaires à la résolution de cette problématique.

4. Références

1. Diagramme des tables systèmes de Sybase ASE ([Lien59](#))
2. Documentation de Sybase ASE ([Lien60](#))

Retrouvez l'article de facade en ligne : [Lien61](#)



Les livres Mac

Mac OS X Support Essentials v10.5

Complètement mis à jour pour Leopard, voici le curriculum officiel du cours de certification Mac OS X Support Essentials v10.5 d'Apple, qui constitue aussi une introduction parfaite pour quiconque a besoin de dépanner et d'optimiser des systèmes Mac OS X.

Conçu pour aider les techniciens, les spécialistes de l'assistance utilisateur et les utilisateurs Mac passionnés, ce guide complet expose de façon très détaillée, étape par étape, toutes les opérations depuis l'installation et la configuration du système d'exploitation jusqu'à la fourniture de services réseau et la gestion des périphériques, notamment les imprimantes.

Cet ouvrage contient :

- des explications d'experts sur les nouvelles fonctionnalités de Mac OS X 10.5 (nouvelle version du Finder, Time Machine, Boot Camp, etc.) et sur les technologies sous-jacentes, ainsi que la description des pratiques de sécurité recommandées et des procédures de dépannage;
- des leçons présentant les procédures étapes par étapes, illustrées avec des tâches réelles;
- de nombreux diagrammes et illustrations qui vous permettent de maîtriser rapidement les techniques;
- des objectifs pour chaque leçon et la durée estimée pour son accomplissement, afin de vous aider à planifier votre étude;
- le résumé de chaque chapitre et les révisions qui récapitulent ce que vous avez appris.

La collection Apple Training Series, conçue et rédigée par des experts Apple, est à la fois un outil d'autoformation et le curriculum officiel pour les programmes AIT (Apple IT Training) et ATT (AppleCare Technician Training). Une fois acquises, les notions abordées dans cet ouvrage, vous pourrez, si vous le souhaitez, vous présenter à l'examen Mac OS X Support Essentials pour obtenir la certification Apple Certified Support Professional.

À propos de l'auteur

Kevin M. White a été ingénieur système Apple avant de devenir consultant Apple indépendant. Formateur certifié Apple depuis 2000, il intervient aussi bien auprès de petits groupes professionnels que pour des auditoires plus larges, comme à Macworld Expo.

Critique du livre par Vincent Brabant

Ce livre répond aux objectifs qu'il s'est fixé, à savoir être *une introduction parfaite pour quiconque a besoin de dépanner et d'optimiser des systèmes Mac OS X*

Mais, je dois avouer que j'ai eu beaucoup de mal pour le lire car il est assez rébarbatif. Autant j'ai eu du plaisir à lire les livres de Guillaume Gete ou celui de Didier Sanz, autant ce fut une corvée pour terminer la lecture de ce livre.

Ce livre est sérieux, de la première phrase jusqu'à la dernière.

Un petit exemple pour illustrer mes propos : lorsque ce livre aborde Kerberos, il l'explique assez bien, mais sérieusement.

Or, Kerberos est également expliqué dans le livre Leopard Efficace de Guillaume Gete, de façon tout aussi claire, si même pas plus, mais en plus, de façon agréable et même humoristique.

Quoi de naturel que de retrouver ce sérieux et cette rigueur, me direz vous, puisque ce livre vous permet également de vous préparer pour la certification *Apple Certified Support Professional*.

C'est que vous n'avez jamais lu le livre Head First Java, par exemple, qui vous prépare à la certification Java.

Lorsqu'on a lu ces livres de la collection Head First qui, pour certains, vous prépare aux certifications Java, le choc est assez dur.

Même pas un petit gramme d'humour, qui vous décrocherait un sourire. Non, rien.

Et c'est vrai, qu'il vous prépare à cette certifications. Mais comment savoir ensuite si vous êtes vraiment prêt à passer cette certifications ? Pas de CD livré avec, avec un *Mock Exams*. Pas de possibilité de passer un Mock Exam en ligne.

Vous êtes quelque peu abandonné à vous-même. Dommage.

Mais les sujets abordés le sont correctement. Par exemple, j'ai particulièrement beaucoup aimé la leçon 10, qui décrit de façon assez détaillée le processus de démarrage de votre Mac.

J'ai même été surpris de voir, dans un livre de formation certifié Apple, je vous le rappelle, un lien vers le toolkit libre rEFIt ([Lien62](#)) qui permet facilement d'étendre les fonctionnalités EFI du Mac.

J'ai également fortement aimé la leçon 2, où est abordé les trousseaux de clé, FileVault, ainsi que la leçon 4, où est abordé les images disques, Time Machine

La partie sur le coupe-feu m'a semblée quelque peu légère, surtout si on doit dépanner quelqu'un qui peut accéder à internet depuis un programme mais pas depuis un autre.

Au final, malgré le fait que j'ai déjà lu pas mal de livres sur Mac

OS X, donc, "The Missing Manual" de Pogue, Le livre des secrets de Didier Sanz, Leopard efficace de Guillaume Gete, Big book of Apple hacks de Chris Seibold, et bien j'ai tout de même beaucoup appris encore à la lecture de celui-ci.

Si vous devez retenir quelque chose de ma critique, c'est ceci :

Vous apprendrez très certainement quelque chose. Ca c'est sûr. Mais vous ne saurez pas, une fois ce livre terminé, si vous êtes vraiment prêt à passer la certification.

Domage :-(

Retrouvez ces critiques de livre sur la page livres Mac : [Lien63](#)

Quels sont vos livres préférés pour Mac, iPhone ... ?

Nous vous invitons à indiquer ci-après vos livres préférés pour Mac, iPhone, iPod, ...

Qu'ils soient en Anglais, ou en Français, cela n'a pas d'importance.

Votez pour vos livres préférés sur Mac : [Lien64](#)

Les derniers tutoriels et articles

Éventail des différents outils de Fuzzing

Introduction au fuzzing tiré du magazine de Septembre 2008 de <http://www.hakin9.org>

1. Ou'est-ce le Fuzzing ?

Le fuzzing (dérivé du mot anglais fuzzy qui signifie flou) est une méthode permettant de traquer facilement des vulnérabilités dans des systèmes ou programmes. La technique est très simple : tout d'abord, il faut générer des données aléatoires à peu près conformes au protocole. Par exemple, ces données peuvent être de longues entrées comme 3425 x z, ou des indications de format tel que %a%a%a%, ou encore des valeurs d'entiers : 0,1,2, etc., ou tout simplement des caractères Unicode (de U+0000 à U+0FFF).

Cependant, ces données ne sont pas totalement aléatoires, car elles ressemblent aux spécifications du protocole. En effet, elles doivent obéir à la logique de programmation du système ciblé, notamment sur les limites dans les opérations de parsing.

Si le fuzzer n'envoie que des données totalement aléatoires au système ciblé, celui-ci les rejettera dès qu'elles ne correspondront plus à ce qui est attendu.

Il faudra donc envoyer des données dites semi-valides pour que le parser ne les refuse pas tout de suite, mais qui pourront cependant causer quelques problèmes.

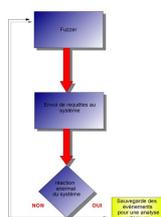
Il n'est pas rare que le système de parsing soit le maillon faible d'un système ciblé (ex: navigateur qui plante suite à une page web contenant des données aléatoires et complexes).

Dans certains cas, il se peut que le système ait soudainement un besoin de ressource en mémoire, ou que la charge du processeur tende violemment vers les 100 %

Ces réactions atypiques montrent que l'envoi de données semi-valides permet d'influencer le fonctionnement.

Dans chaque cas, les résultats sont sauvegardés pour une analyse ultérieure, dans d'autres cas, la même démarche peut être effectuée à nouveau, mais de façons différentes si le système n'a pas réagi.

Généralement, grâce à l'utilisation de cette technique, des erreurs dans la gestion de la mémoire ou dans d'autres parties du système ciblé sont découvertes.



Principe du fuzzing

Ces vulnérabilités peuvent alors être utilisées pour obtenir le total contrôle du système ainsi testé, si bien sûr cela est utilisé lors de tests de sécurité.

Évidemment, les faiblesses découvertes par le fuzzer ne sont pas toutes des vulnérabilités critiques, loin s'en faut, mais elles peuvent toutefois être utilisées par un individu malveillant.

Dans la suite de l'article, nous aborderons les différents types de fuzzers.

2. Les différents fuzzers

Il existe deux grandes catégories de Fuzzers.

Le premier type de fuzzer est composé de Framework qui reposent sur un API dans le but de créer des entrées et implanter des protocoles spécifiques.

Cette rubrique est composée de différents fuzzers.

2.1. Les Fuzzers de Frameworks

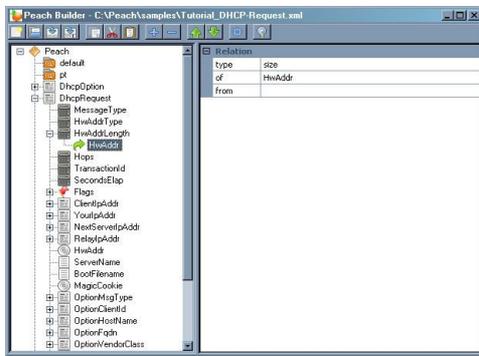
2.1.1. ANTIPARSER

ANTIPARSER est un fuzzer à toute épreuve, il permet l'envoi de données mal formées au niveau de l' API. Le but d'Antiparser est de fournir une API utilisable sur des modèles de protocoles de réseaux et sur différents formats de fichier. Une fois qu'un modèle a été créé, Antiparser utilise diverses méthodes pour créer des données aléatoires de façon à déclencher des bugs dans le logiciel ou le système ciblé. Le fuzzer inclut dans son fichier de téléchargement un tutoriel et un manuel d'utilisation très détaillé. Enfin, Antiparser fonctionne seulement avec la version supérieure de python 2.3.

Site officiel : ([Lien65](#))

2.1.2. PEACH

PEACH est un fuzzing écrit lui aussi en python. Le fuzzer a une facilité d'emploi remarquable et permet de gagner du temps de développement. Peach fuzze à peu près tout comme .NET, COM/ActiveX, le SQL, des dll, des applications de réseaux, ainsi que certaines structures web. À noter aussi que le site officiel de peach inclut un tutoriel d'utilisation bien détaillé.



```
# "<span class=\"vuln\"> %s </span>\n" \ "<span
class=\"url\"> http://%s%s%c%s
# </span>\n</p>\n\n"
# define MESSAGE (proto == POST) ? "POST" : "GET", \
message, T->host, ((proto ==
# POST) || action) ? action : "", \ qm, url
/**
** format is a string like
** <GET|POST> <possible vuln> <http://host/page?
query=bla> [error]
**/
char qm = ((proto == POST) ? '?' : '/');
fprintf( stdout, FMTSTDOUT, MESSAGE );
if ( T->opt->logtype == HTML )
freport( T->opt->logfile, FMTHTML, MESSAGE );
else
freport( T->opt->logfile, FMTTXT, MESSAGE );
if ( error ) {
if ( T->opt->logtype == HTML )
freport( T->opt->logfile, "<div class=\"error\">\n
%s\n</div>\n", error );
else
freport( T->opt->logfile, "--[ %s ]--\n", error );
fprintf( stdout, KYA RED "--[ %s ]--\n" DEF, error );
}
}
```

Site officiel : [Lien66](#)

2.1.3. SPIKE

SPIKE permet d'écrire facilement des API de protocoles et son utilisation est très simple. Le fuzzer étudie par ingénierie inverse les protocoles de réseaux. Il dispose de plusieurs exemples permettant de prendre facilement en main son utilisation. Il inclut une technique de brute force pour le serveur Web NTLM.

Site officiel : [Lien67](#)

2.1.4. SCRATCH

SCRATCH est un fuzzer de protocole écrit en python, qui permet dans les trois quarts du temps de trouver une grande variété de vulnérabilités à partir d'un simple paquet. Scratch fait une analyse heuristique des syntaxes de fichiers binaires pour déterminer les bugs exploitables. Scratch a une structure qui lui permet de fuzzer les protocoles binaires comme SSL et SMB. [Lien68](#)

2.2. Les Fuzzers pour le Web

Après les quatre principaux fuzzers d'API et de Frameworks, voyons maintenant la deuxième catégorie de fuzzers, celle des logiciels et des protocoles.

2.2.1. WAPITI

WAPITI est un fuzzer pour les applications Web écrites en python. Il utilise une technique qui lui est propre car il scanne toute l'application ciblée et il identifie toutes les saisies d'un utilisateur. Après le scan, il exécute alors une série d'essais sur chaque variable, comme l'injection de caractères spéciaux (et de ponctuations) cherchant ainsi des réactions atypiques de l'application. Wapiti s'utilise pour automatiser la découverte de faille de type SQL, des attaques d'injection de code, ainsi que les XSS.

Site officiel : [Lien69](#)

2.2.2. WEBFUZZER

WEBFUZZER est une application écrite en langage C par le groupe gunzip, le fuzzer se spécialise pour scanner les sites web, il teste à distance différentes vulnérabilités comme l'injection sql, cross site scripting, les failles php includes, l'exécution de code à distance, etc...

```
//extrait du code-source de web fuzzer
{
# define FMTSTDOUT GREEN KYA "(" DEF "%s" GREEN KYA ")" \
DEF GREEN ": %s " KYA "(" DEF RED "http://%s%s%c%s"
GREEN KYA ")" DEF "\n"
# define FMTTXT "(%s): %s (http://%s%s%c%s)\n"
# define FMTHTML "<p class=\"report\"> <span
class=\"proto\"> (%s) </span>\n" \
```

Site officiel : [Lien70](#).

2.3. Les Fuzzers pour le navigateur

Intéressons-nous maintenant aux différents fuzzers pour le navigateur. Il existe beaucoup d'applications disponibles sur le net, mais je n'en citerai que deux : AXMAN et HAMACHI.

2.3.1. AXMAN

AXMAN est un fuzzer développé par le chercheur américain HD Moore, spécialisé dans le fuzzing d'ActiveX. Il permet de déterminer tous les objets COM proposés par Internet Explorer. Depuis le lancement de AxMan, de nombreuses failles ont été découvertes (et rendues publiques), notamment au niveau des navigateurs web, comme Internet Explorer, Firefox, Safari, Konqueror et Opera.

L'auteur a notamment dédié un blog à ces recherches [Lien71](#).

A noter aussi que le site officiel propose une démonstration en ligne : [Lien72](#).

2.3.2. HAMACHI

HAMACHI est un projet issu d'une communauté de développeurs, avec à la tête H D Moore et Aviv Raff, et ayant pour but de vérifier l'intégrité des navigateurs.

Hamachi est compilé en javascript et permet la recherche des DHTML (Dynamic HyperText Markup Language).

Le fuzzer bombarde le navigateur ciblé avec un maximum de valeurs dans le but de provoquer une réaction non prévue par les développeurs, amenant à la découverte d'une ou plusieurs vulnérabilités.

Site officiel : [Lien73](#)

2.4. Les Fuzzers de protocoles et de services

2.4.1. PROTOS

PROTOS, le projet Protos recherche différentes approches de tests au niveau du protocole en utilisant diverses techniques de type black-box. L'objectif est de soutenir activement l'élimination des failles de sécurité. Il existe d'autres fuzzers issus de ce projet, comme Protos HTTP-reply, Protos WAP, Protos SIP, etc...

L'adresse du site officiel : ([Lien74](#))

2.4.2. SNMPPFUZZER

SNMPPFUZZER est un dérivé du développement du fuzzer Protos, il contient un nouveau moteur de générateur de bugs écrit en Perl. Il fournit des méthodes efficaces pour déterminer le test à l'origine d'une réaction inattendue du système ciblé, il offre aussi aux utilisateurs une interface plus conviviale. ([Lien75](#))

2.4.3. FTPFUZZ

FTPFUZZ, est un fuzzer d'interface très simple pour évaluer les mises en oeuvre de serveur FTPD. Il permet à l'utilisateur de spécifier des commandes FTP par le biais des paramètres du fuzzer et de définir le cheminement à suivre pour chaque type de cas auquel l'utilisateur est confronté.

Son utilisation a mis à jour de nombreuses vulnérabilités dans de nombreux services FTP, notamment des vulnérabilités permettant l'exploitation à distance. ([Lien76](#))

2.4.4. Bluetooth Stack Smasher

BSS, ou Bluetooth Stack Smasher, est un fuzzer écrit et développé en langage C par Pierre Betouin dans le but d'effectuer différents tests sur les périphériques Bluetooth. Les codes-source du programme sont accompagnés d'une documentation réalisée par l'auteur lui-même. La documentation aide à mettre en pratique l'utilisation de BSS, sur différents réseaux bluetooth, comme ceux des téléphones portables... ([Lien77](#))

2.5. Les Fuzzers pour les couches de transport et de réseau

2.5.1. ISIC

ISIC est une suite d'outils de test de la stabilité d'une pile IP et de ses composantes (TCP, UDP, ICMP etc...). Il génère des piles de paquets pseudo-aléatoires sur le protocole ciblé. Dans la plupart des cas, les paquets ainsi créés sont acceptés. Ils sont ensuite envoyés à la machine ciblée pour contourner les règles de pare-feu ou trouver des bugs dans la pile IP. ISIC contient également un utilitaire servant à examiner le Hardware.

Site officiel : ([Lien78](#))

2.5.2. ip6sic

Ip6sic est un outil écrit en langage C qui permet de fuzzer la pile IPv6. Il fonctionne d'une manière très semblable à ISIC. Il a été élaboré pour fonctionner sur FreeBSD ainsi que sur OpenBSD et Linux.

Site officiel: ([Lien79](#))

3. À qui s'adresse ces outils ?

Quand un bug a été découvert, il est alors possible de réaliser d'autres tests pour savoir si la découverte est susceptible d'être exploitée. Si tel est le cas, un code d'exploitation est créé, le plus souvent à partir du framework Metasploit. De ce fait, le fuzzing se destine principalement aux développeurs ainsi qu'aux chercheurs en sécurité afin de corriger la (ou les) vulnérabilité(s), comme ce fut le cas durant l'été 2006. En effet, au cours de cette période, l'expert en sécurité HD Moore a rendu "publique" l'utilisation du fuzzing.

Toutefois, les crackers utilisent ce même procédé afin de tirer profit de la faiblesse du système ou de l'application (contrôle à distance, contournement de la sécurité, etc...)

4. Conclusion

Vous avez vu les principes fondamentaux du fuzzing, ainsi que les différents fuzzers. Ils permettent de découvrir plus rapidement des vulnérabilités parmi des milliers de lignes qui composent le code source d'un programme, et ils s'avèrent intéressants pour traquer les failles de type cross-site scripting (XSS) DoS (dénier de service), injection SQL, buffer overflow et format string. Il existe une grande variété de fuzzers issus notamment du monde Open Source, lequel développe de nombreux logiciels d'audits et de tests d'intrusions (voir la rubrique "liens"). Les outils de fuzzing sont de plus en plus présents dans le domaine de la sécurité, du fait d'une politique de sécurité applicative de plus en plus stricte. Néanmoins, le fuzzing n'est pas un procédé infaillible, car d'une part, certains critères d'un logiciel peuvent rendre son exploitation très complexe et d'autre part, un bug peut en cacher un autre.

5. Liens

D'autres fuzzers : ([Lien80](#))

Ouvrage traitant des fuzzers, disponibles chez l'éditeur O'reillys : ([Lien81](#))

Ouvrage traitant des fuzzers open source, disponible chez l'éditeur Syngress Media : ([Lien82](#))

Vidéo de la conférence de black-hat 2006 sur le fuzzing ([Lien83](#))

Article traitant des tests d'intrusion grâce aux Fuzzers, réalisé par Christophe AUFFRAY pour JDN Solutions : ([Lien84](#))

Retrouvez l'article de Pierre Therrode en ligne : [Lien85](#)

Les derniers tutoriels et articles

SharePoint et les technologies 3.5 - Part I

Le train technologique poursuivant sa course effrénée, voici un tutoriel qui se décline en plusieurs parties pour aborder l'utilisation de dotnet 3.5 au sein de SharePoint. La première partie se concentre sur la configuration de SharePoint et sur l'utilisation de l'AJAX dans des WebParts. La deuxième partie s'attardera sur Silverlight et la troisième abordera l'utilisation de WCF.

1. Introduction

Avant toute chose, les pré-requis pour pouvoir reproduire ce qui est décrit dans le tutoriel sont les suivants :

- Le framework 3.5 doit être installé sur le serveur
- Vous disposez de MOSS ou WSS avec le SP1 installé
- Vous disposez de Visual Studio 2005 ou 2008

La plate-forme dotnet évoluant sans cesse et SharePoint étant entièrement basé sur celle-ci, il est important de comprendre comment intégrer les dernières évolutions technologiques. Une des évolutions apportées par le framework 3.5 est l'intégration native d'ASP.NET AJAX dans les DLL du framework.

Avant, vous deviez télécharger les extensions AJAX et les installer en plus du framework .NET. Nous allons voir comment configurer et utiliser ces bibliothèques 3.5 au sein de composants SharePoint

2. Configuration de SharePoint

Pour commencer, il faut rendre SharePoint compatible avec le framework 3.5. Etant donné que Tobias Zimmergren a déjà entièrement décrit comment s'y prendre, je vous renvoie vers son blog ([Lien86](#)) qui explique en détail ce qui doit être ajouté dans le web.config de l'application SharePoint pour laquelle vous souhaitez utiliser 3.5.

Vous pouvez également télécharger la fonctionnalité ([Lien87](#)) développée par Renaud Comte permettant d'effectuer les modifications requises dans le web.config automatiquement. Elle s'intitule **3.5 Web.Config Feature (separate download)** et permet d'ajouter les sections nécessaires dans le fichier web.config de l'application SharePoint que vous souhaitez rendre compatible.

Enfin, à toutes fins utiles, je vous ai joint un exemple de web.config fonctionnel et compatible avec 3.5. Vous le trouverez dans la section téléchargement.

3. Le ScriptManager et les pages maîtres

L'objectif du tutoriel n'est pas de vous apprendre comment utiliser l'AJAX en ASP.NET mais bien d'aborder tant que faire se peut les spécificités liées à SharePoint. Une de ces spécificités réside notamment dans l'utilisation des **ScriptManager**.

Comme vous le savez sans doute, une instance de l'objet **ScriptManager** est impérative pour pouvoir travailler en AJAX. Généralement, c'est d'ailleurs le premier contrôle qui doit être ajouté à une page. Dans SharePoint, les composants web sont toujours exécutés dans un contexte où une page maître est

présente. En effet, qu'il s'agisse de WebParts, colonnes personnelles, web controls etc...tous ces composants sont encapsulés dans l'architecture SharePoint et bénéficient automatiquement de la présence de la page maître associée au site où ils sont exécutés.

Grâce ou à cause de cela, la déclaration du ScriptManager se fait généralement au sein de la page maître elle-même, comme ceci :

```
<asp:ScriptManager id="ScriptManager1"
EnablePartialRendering="TRUE" runat="server">
```

Au cas où vous ne bénéficiez pas d'une page maître avec un ScriptManager intégré (ce qui est le cas avec les pages maîtres standard), vous pouvez en déployer une vous-même. C'est ce que j'ai fait pour cet exemple, vous retrouverez une page maître qui sera ajoutée dans la galerie de pages maîtres lors de l'activation de la fonctionnalité liée au projet.

Vous pourrez donc vous inspirer de cela pour développer vos propres pages maîtres ou en modifier une existante avec SharePoint Designer par exemple.

4. Les techniques d'intégration de l'AJAX dans les composants SharePoint

4.1. Dans les WebParts, les contrôles web, les colonnes personnelles

Il existe globalement deux techniques majeures pour intégrer de l'AJAX dans les WebParts :

- **Dans le code-behind du contrôle lui-même** : cette technique a le désavantage de "mélanger" le code métier avec l'interface graphique si on n'y prend garde
- **Dans un contrôle utilisateur ASP.NET** : cette technique isole l'aspect graphique dans le contrôle utilisateur. Il faut ensuite charger celui-ci dans notre WebPart

Une déclinaison de la deuxième technique consiste à utiliser le composant de Jan Tielens, j'ai nommé le Return of SmartPart ([Lien88](#)) qui a été récemment mis à jour par son auteur.

4.2. Dans les pages applicatives

Les pages applicatives étant liées à la page maître **Application.master**, l'insertion du **ScriptManager** peut-être rendue plus complexe car il n'est pas évident de personnaliser cette page maître. Ceci dit, trois choix s'offrent néanmoins à vous :

- **Vous personnalisez Application.master** : en ajoutant la

déclaration du script manager mais en acceptant toutes les contraintes liées à la personnalisation de cette page (voir google).

- **Vous délivrez votre propre page maître applicative :** vous liez explicitement vos pages applicatives à votre page maître applicative personnelle
- **Vous générez le code AJAX "à la main" :** en fonction de l'environnement et de la complexité de la page, c'est une option envisageable. L'avantage d'ailleurs de cette technique est qu'elle ne nécessite aucune configuration spécifique au niveau du serveur.

4.3. Dans les workflows

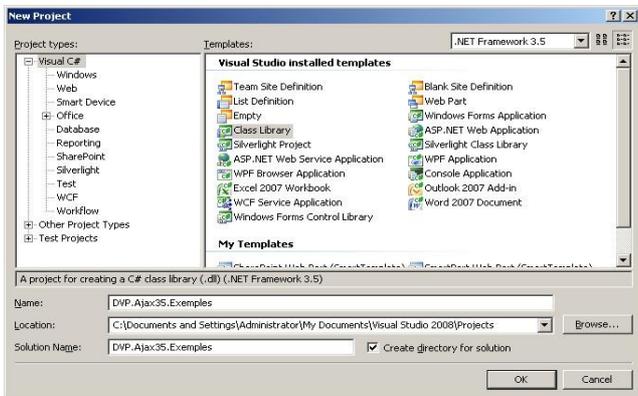
Les seuls composants nécessitant éventuellement de recourir à l'AJAX sont les formulaires d'association/initiation/modification des workflows. Si vous utilisez Forms Services, bonne nouvelle : l'AJAX y est intégré nativement sans que vous n'ayez rien à faire. Si vous utilisez des pages web classiques, vous pouvez soit utiliser le mode déclaratif d'ASP.NET, soit générer dynamiquement les contrôles dans le code-behind associé à la page.

5. Création d'un SPGridView avec de l'AJAX - Tout en Code-Behind

Dans la lignée de mon dernier article sur le SPGridView ([Lien89](#)), voici un WebPart qui utilise un SPGridView avec de la pagination bénéficiant de l'AJAX pour éviter un full page postback lorsque l'on passe d'une page à l'autre.

Voici pas à pas comment il faut s'y prendre avec Visual Studio 2008, notez que vous pouvez exécuter les mêmes étapes avec Visual Studio 2005

Créer un nouveau projet de type **Class Library**



Ajoutez ensuite les références suivantes :

1. System.Web => onglet dotnet
2. System.Web.Extensions => onglet dotnet
3. Microsoft.SharePoint => 12\ISAPI de SharePoint

Signez votre projet : Cliquez sur votre Projet => Propriétés => Onglet Signature => Signez le

Créez à présent une structure de répertoires et de fichiers comme illustré ci-dessous :



et renommez **class1.cs** en **SPGridViewAjax.cs**.

Voici à présent le contenu du fichier **feature.xml** :

```
<?xml version="1.0" encoding="utf-8"?>
<Feature Id="EE4B04A2-71A0-46E3-BA2A-783B0729DB3C"
  Title="SPGridView avec de l'AJAX"
  Description="SPGridView avec de l'AJAX"
  Version="1.0.0.0"
  Hidden="FALSE"
  Scope="Site"
  xmlns="http://schemas.microsoft.com/sharepoint/
t/">
  <ElementManifests>
    <ElementManifest Location="webpart.xml"/>
  </ElementManifests>
</Feature>
```

Fichier classique définissant une fonctionnalité de collection et pointant vers le fichier webpart.xml

Contenu du fichier **webpart.xml** :

```
<?xml version="1.0" encoding="utf-8" ?>
<Elements
  xmlns="http://schemas.microsoft.com/sharepoint/">
  <Module Name="WebParts"
    List="113"
    Url="_catalogs/wp"
    RootWebOnly="TRUE">
    <File Url="DVP.SPGridViewAJAX.webpart"
      Type="GhostableInLibrary" >
      <Property Name="Group" Value="DVP -
      SPGridView"></Property>
    </File>
  </Module>
</Elements>
```

Ce fichier indique qu'il faut déployer le WebPart dans la galerie de WebParts de la collection où la fonctionnalité est activée.

Contenu du fichier **DVP.SPGridViewAJAX.webpart** :

```
<?xml version="1.0" encoding="utf-8"?>
<webParts>
  <webPart xmlns="http://schemas.microsoft.com/WebPart/
v3">
    <metaData>
      <type name="signature" />
      <importErrorMessage>Cannot import this Web
Part.</importErrorMessage>
    </metaData>
    <data>
      <properties>
        <property name="Title"
type="string">DVP.SPGridViewAJAX.webpart</property>
      </properties>
    </data>
  </webPart>
</webParts>
```

Veillez à remplacer **signature** par votre signature que vous pouvez obtenir avec Reflector. Ce fichier indique simplement au système que le WebPart est lié au contrôle spécifié par la signature.

Reprenez à présent votre classe principale que vous avez nommé **SPGridViewAjax.cs** et faites-la dériver de **System.Web.UI.WebControls.WebParts.WebPart**. Ensuite, déclarez deux membres de classe comme suit :

```
SPGridView SimpleGrid = null;
UpdatePanel ControlPanel = null;
```

réécrivez à présent la méthode **CreateChildControls()**

```
protected override void CreateChildControls()
{
    ScriptManager ScriptMgr =
    ScriptManager.GetCurrent(Page);
    if (ScriptMgr == null)
    {
        throw new ApplicationException("Pas de script
manager!");
    }
    ControlPanel = new UpdatePanel();
    ControlPanel.UpdateMode =
UpdatePanelUpdateMode.Conditional;
    ControlPanel.ID = "ControlPanel";
    Controls.Add(ControlPanel);
    SimpleGrid = new SPGridView();
    SPDataSource Donnees = new SPDataSource();
    Donnees.List =
SPContext.Current.Web.Lists["Documents"];
    Donnees.Scope = SPViewScope.Recursive;
    SimpleGrid.AutoGenerateColumns = false;
    BoundField ColonneID = new BoundField();
    ColonneID.DataField = "ID";
    ColonneID.HeaderText = "ID";
    SimpleGrid.Columns.Add(ColonneID);
    BoundField ColonneTitre = new BoundField();
    ColonneTitre.DataField = "Title";
    ColonneTitre.HeaderText = "Title";
    SimpleGrid.Columns.Add(ColonneTitre);
    SimpleGrid.DataSource = Donnees;
    SimpleGrid.AllowPaging = true;
    SimpleGrid.PageSize = 1;
    SimpleGrid.PageIndexChanging += new
GridViewPageEventHandler(SimpleGrid_PageIndexChanging);
    ControlPanel.ContentTemplateContainer.Controls.Add(Si
mpleGrid);
    SimpleGrid.TemplateControl = null;
    SimpleGrid.PagerTemplate = null;
    ScriptMgr.RegisterAsyncPostBackControl(SimpleGrid);
    base.CreateChildControls();
}
```

Ce SPGridView est lié à la liste **Documents** du site courant et aux colonnes **ID** et **Title**. Vous devez éventuellement changer le nom de la liste pour que ça fonctionne chez vous et si vous êtes en français, remplacez **Title** par **Titre**...Tout ceci peut bien sûr être fait dynamiquement dans le code mais je préfère simplifier au maximum la lisibilité, c'est pourquoi il n'y a pas de gestion d'erreurs non plus.

Au niveau de l'AJAX, le code détecte d'abord si un ScriptManager est présent ou non dans le contexte courant et lance une exception si aucun ScriptManager n'est trouvé. Ensuite, un UpdatePanel est créé et le SPGridView lui est ajouté en tant que contrôle.

Le SPGridView étant un contrôle très sensible, il faut lui spécifier que nous ne développons pas de modèle de pagination via la ligne de code :

```
SimpleGrid.PagerTemplate = null;
```

sauf si bien sûr vous souhaitez en créer un :). Complétons à présent avec les deux méthodes manquantes :

```
void SimpleGrid_PageIndexChanging(object sender,
GridViewPageEventArgs e)
{
    SimpleGrid.PageIndex = e.NewPageIndex;
    SimpleGrid.DataBind();
    ControlPanel.Update();
}
protected override void OnPreRender(EventArgs e)
{
    SimpleGrid.DataBind();
    base.OnPreRender(e);
}
```

La méthode **SimpleGrid_PageIndexChanging** est appelée lors du click sur un numéro de page. Elle spécifie le nouvel index de page et rafraîchit l'UpdatePanel.

Le code complet est donc le suivant :

```
public class SPGridViewAjax :
System.Web.UI.WebControls.WebParts.WebPart
{
    SPGridView SimpleGrid = null;
    UpdatePanel ControlPanel = null;
    protected override void CreateChildControls()
    {
        ScriptManager ScriptMgr =
        ScriptManager.GetCurrent(Page);
        if (ScriptMgr == null)
        {
            throw new ApplicationException("Pas de script
manager!");
        }
        ControlPanel = new UpdatePanel();
        ControlPanel.UpdateMode =
UpdatePanelUpdateMode.Conditional;
        ControlPanel.ID = "ControlPanel";
        Controls.Add(ControlPanel);
        SimpleGrid = new SPGridView();
        SPDataSource Donnees = new SPDataSource();
        Donnees.List =
SPContext.Current.Web.Lists["Documents"];
        Donnees.Scope = SPViewScope.Recursive;
        SimpleGrid.AutoGenerateColumns = false;
        BoundField ColonneID = new BoundField();
        ColonneID.DataField = "ID";
        ColonneID.HeaderText = "ID";
        SimpleGrid.Columns.Add(ColonneID);
        BoundField ColonneTitre = new BoundField();
        ColonneTitre.DataField = "Title";
        ColonneTitre.HeaderText = "Title";
        SimpleGrid.Columns.Add(ColonneTitre);
        SimpleGrid.DataSource = Donnees;
        SimpleGrid.AllowPaging = true;
        SimpleGrid.PageSize = 1;
        SimpleGrid.PageIndexChanging += new
GridViewPageEventHandler(SimpleGrid_PageIndexChanging);
        ControlPanel.ContentTemplateContainer.Controls.Add(Si
mpleGrid);
        SimpleGrid.TemplateControl = null;
        SimpleGrid.PagerTemplate = null;
        ScriptMgr.RegisterAsyncPostBackControl(SimpleGrid);
        base.CreateChildControls();
    }
    void SimpleGrid_PageIndexChanging(object sender,
GridViewPageEventArgs e)
    {
        SimpleGrid.PageIndex = e.NewPageIndex;
        SimpleGrid.DataBind();
        ControlPanel.Update();
    }
}
```

```
protected override void OnPreRender (EventArgs e)
{
    SimpleGrid.DataBind ();
    base.OnPreRender (e);
}
}
```

Le résultat ressemble à ceci :

ID	Title
3	

1 2 3 4

avec une pagination en AJAX. Vous pouvez retrouver ce projet et une solution .wsp dans la section téléchargement.

Pour déployer le projet, je vous conseille d'utiliser l'add-in WSPBuilder ([Lien90](#)) pour Visual Studio

6. Utilisation de l'AjaxControlToolkit dans un contrôle utilisateur

Pour illustrer l'utilisation de l'AjaxControlToolkit, j'ai décidé de reproduire l'exemple de Microsoft disponible ici ([Lien91](#)). La seule différence est que cet exemple est dans un contexte web classique, je l'ai simplement remis dans le contexte de SharePoint.

Ajoutez ensuite les références suivantes :

1. System.Web => onglet dotnet
2. System.Web.Extensions => onglet dotnet
3. AjaxControlToolkit

La DLL **AjaxControlToolkit** est disponible sur le site de l'éditeur ([Lien92](#)).

Comme pour l'exemple précédent, créez un projet de type **Class Library** et nommez-le **DVP.AjaxToolkit.Exemples**. Créez ensuite la structure de répertoire et de fichier suivante :



Excluez toutefois les fichiers dvp.master et master.xml qui servent à déployer une page maître. Je les ai inclus dans le projet afin que vous puissiez plus facilement tester les WebParts. Ceci dit, je vous laisserai découvrir par vous même comment cela a été fait.

Les fichiers **feature.xml**, **webpart.xml** et **AjaxToolkitDvp.webpart** sont très similaires à ceux de l'exemple précédent. Il est donc inutile d'en réafficher le contenu. Concentrons nous plutôt sur ce qui change réellement, à savoir la création d'un contrôle utilisateur.

Dans la structure du projet et plus précisément dans le répertoire **CONTROLTEMPLATES/DvpAjaxToolkit**, il y a le contrôle utilisateur **dvpajaxtoolkit.ascx**. Celui-ci contient principalement le code suivant :

```
<%@ Control Language="C#"
    Inherits="DVP.AjaxToolkit35.Exemples.AjaxToolkitExe...
... "
%>
<% Register Tagprefix="cc1"
    Namespace="AjaxControlToolkit"
    Assembly="AjaxControlToolkit..." %>
```

A savoir les directives déclarant l'utilisation du toolkit et le fait que la page dérive de **AjaxToolkitExempleCodeBehind** (que nous implémenterons plus tard). J'ai volontairement abrégé la déclaration des directives, vous pourrez les retrouver au complet en téléchargeant l'exemple.

En plus des directives, le contrôle contient une déclaration de style contenant toutes les classes CSS utilisées par le contrôle de rating.

```
<style type="text/css">
    .accordionHeader
    {
        border: 1px solid #2F4F4F;
        color: white;
        background-color: #2E4d7B;
        font-family: Arial, Sans-Serif;
        font-size: 12px;
        font-weight: bold;
        padding: 5px;
        margin-top: 5px;
        cursor: pointer;
    }

    .accordionContent
    {
        background-color: #D3DEEF;
        border: 1px dashed #2F4F4F;
        border-top: none;
        padding: 5px;
        padding-top: 10px;
    }

    .ratingStar {
        font-size: 0pt;
        width: 13px;
        height: 12px;
        margin: 0px;
        padding: 0px;
        cursor: pointer;
        display: block;
        background-repeat: no-repeat;
    }

    .filledRatingStar {
        background-image:
        url (PublishingImages/FilledStar.png);
    }

    .emptyRatingStar {
        background-image:
        url (PublishingImages/EmptyStar.png);
    }

    .savedRatingStar {
        background-image:
        url (PublishingImages/SavedStar.png);
    }
</style>
```

Notez qu'il est préférable d'inclure les styles dans un fichier CSS à part que l'on déploiera ensuite dans une style library.

Ensuite vient le plus important, à savoir la déclaration du contrôle de rating :

```
<asp:UpdatePanel ID="UpdatePanel1" runat="server">
  <ContentTemplate>
    <div style="float: left; width:230;">How much do you
    like ASP.NET AJAX ?</div>
    <c1:Rating ID="LikeRating" runat="server"
    CurrentRating="3"
    MaxRating="5"
    StarCssClass="ratingStar"
    WaitingStarCssClass="savedRatingStar"
    FilledStarCssClass="filledRatingStar"
    EmptyStarCssClass="emptyRatingStar"
    OnChanged="LikeRating_Changed"
    style="float: left;">
  </c1:Rating>
  <br />
  <div style="clear:left;">
  <br />
  <asp:Button ID="ButtonSubmit" runat="server"
  Text="Submit"
  onclick="ButtonSubmit_Click" /><br /><br />
  <asp:Label ID="LabelResponse" runat="server" Text="[
  No response provided yet.]"></asp:Label>
  </div>
  </ContentTemplate>
</asp:UpdatePanel>
```

Ici on a simplement la déclaration de l'UpdatePanel et du contrôle de Rating. C'est un copier-coller pur et simple de l'exemple proposé par Microsoft. Ce contrôle fait appel à la méthode **LikeRating_Changed** et le bouton de soumission appelle quant à lui la méthode **ButtonSubmit_Click**. Ces deux méthodes sont implémentées dans le code-behind du contrôle utilisateur. C'est la classe **AjaxToolkitExempleCodeBehind** dont voici le contenu :

```
public class AjaxToolkitExempleCodeBehind :
System.Web.UI.UserControl
{
  protected Rating LikeRating;
  protected Label LabelResponse;

  protected void LikeRating_Changed(object sender,
  AjaxControlToolkit.RatingEventArgs e)
  {
    e.CallbackResult = "Upate done. Value = " + e.Value
    + " Tag = " + e.Tag;
  }
  protected void ButtonSubmit_Click(object sender,
  EventArgs e)
  {
    string howMuch = "[unknown]";

    switch (LikeRating.CurrentRating)
    {
      case 1:
        howMuch = "a bit.";
        break;
      case 2:
        howMuch = "some.";
        break;
      case 3:
        howMuch = "a fair bit.";
        break;
      case 4:
        howMuch = "a lot.";
        break;
```

```
      case 5:
        howMuch = "more than any thing.";
        break;
    }

    LabelResponse.Text = "You like ASP.NET AJAX " +
    howMuch + ".";
  }
}
```

Ici, la différence par rapport à l'exemple de Microsoft est que ce code est inclut dans une classe dérivant de **UserControl**. **LikeRating** et **LabelResponse** sont déclarés en **protected** et correspondent aux contrôles déclarés de manière déclarative...

Enfin, il ne reste plus à notre WebPart SharePoint qu'à charger ce contrôle utilisateur. C'est la classe **WebPartPrincipal.cs** qui se charge de cela :

```
public class WebPartPrincipal :
System.Web.UI.WebControls.WebParts.WebPart
{
  const string ControlFile =
  @"/_controltemplates/dvpajaxtoolkit/dvpajaxtoolkit.ascx";

  protected override void CreateChildControls()
  {
    ScriptManager ScriptMgr =
    ScriptManager.GetCurrent(Page);
    if (ScriptMgr == null)
    {
      throw new ApplicationException("Pas de script
      manager");
    }
    else if (!ScriptMgr.EnablePartialRendering)
    {
      ScriptMgr.EnablePartialRendering = true;
    }
    UserControl Ctrl = Page.LoadControl(ControlFile) as
    UserControl;
    Controls.Add(Ctrl);
    base.CreateChildControls();
  }
}
```

Le WebPart exécute le même test que pour l'exemple précédent pour détecter si un ScriptManager est bien présent ou non. Il fait également en sorte que la propriété **EnablePartialRendering** soit bien mise à TRUE sans quoi les changements ne seraient pas reflétés au sein du contrôle utilisateur.

Ensuite, le WebPart charge simplement le contrôle pointé par la constante **ControlFile**. Tout ceci donne le résultat suivant :



Pour déployer le projet, je vous conseille d'utiliser l'add-in WSPBuilder ([Lien90](#)) pour Visual Studio

7. Tester les projets présents en téléchargement

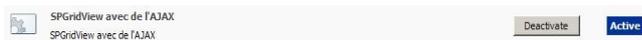
Pour tester les projets proposés en téléchargement, commencez par déployer la solution **DVP.AjaxToolKit35.Exemples.wsp** via stsadm. Ensuite, après activation de la fonctionnalité de collection :

Vous devriez retrouver une nouvelle page maître que vous pouvez associer à votre site :



Ensuite, il ne vous reste plus qu'à ajouter le WebPart à une page :

Ensuite, vous pouvez déployer l'autre solution **DVP.Ajax35.Exemples.wsp**. Après activation de la fonctionnalité :



et ajouter le WebPart contenant le SPGridView avec pagination.



Veillez à le déployer dans un site où il existe une bibliothèque de documents nommées **Documents** avec une colonne **Title**. Sinon changez simplement le code et mettez le nom de la bibliothèque que vous voulez.

Retrouvez l'article de Stephane Eyskens en ligne : [Lien93](#)

Formules mathématiques dynamiques MS ACCESS.

Tout le monde le sait, ACCESS n'a pas la réputation de flexibilité d'EXCEL pour tout ce qui touche aux calculs. En effet, rien n'est apparemment prévu pour saisir et résoudre des formules de calculs. Généralement, c'est à la charge du programmeur de prévoir, concevoir et faire procéder à la résolution de celles-ci. L'objectif de cet article est de démontrer que nous pouvons laisser l'utilisateur maître de la création de formules de calculs complexes.

1. Introduction

On reproche souvent à ACCESS de ne pas être souple en matière de formules mathématiques. En effet, le comparer à EXCEL est impossible, chaque produit a son utilité. C'est la raison pour laquelle l'automation permet des traitements qui sont difficiles, voire impossibles à reproduire avec ACCESS. Comme toute affirmation a son contraire, nous allons démontrer, avec certaines limites, que nous pouvons créer des formules mathématiques dynamiques avec ACCESS. Dynamique dans le sens où elles peuvent être entrées directement par l'utilisateur.

2. Un cas bien concret

Pour la démonstration, nous allons résoudre un problème souvent rencontré dans la gestion de contrats à moyen et long terme. Ce type de gestion est généralement assorti d'une clause de revalorisation contractuelle. En résumé, chaque année à date anniversaire du contrat, il est convenu d'une revalorisation indexée sur des indices officiels. Pour la France, vous pouvez consulter la majorité de ceux-ci sur le site INSEE. On utilise le ratio d'un indice constaté entre la date actuelle et la date année-1. Les contrats de location d'habitations en sont un parfait exemple.

2.1. Analyse rapide

Dans un premier temps, attachons-nous à analyser les différents composants de la solution.

Qu'avons-nous en entrée ?

Nous avons une ou plusieurs formules, les indices actuels et les indices précédents.

Que voulons-nous obtenir ?

Un coefficient de revalorisation ou directement un résultat.

2.2. Le pré-requis

Nous allons avoir besoin de plusieurs objets pour la démonstration. Commençons par les tables.

Table tblFormules Nom du champ	Type du champ - Taille	Description
N_Formule	NuméroAuto	identifie la formule
Formule	Texte - 255	stocke la formule de calcul

Dans un cas pratique, il peut être utile de créer un champ supplémentaire pour stocker une description de la formule.

Table tblIndices Nom du champ	Type du champ - Taille	Description
TypeIndice	Texte - 15	l'abréviation de l'indice dans la formule; souvent celui-ci correspond à sa dénomination officielle.
Mois	Numérique - Entier	Le numéro du mois (1 à 12).
Annee	Numérique - Entier	Le numéro de l'année sur 4 chiffres.
Indice	Réel	La valeur de l'indice avec 2 décimales maximum.

Dans TypeIndice, nous stockerons le nom de l'indice (BT48, CF, EBIQ...). Indice contiendra la valeur; le type réel simple est très largement suffisant puisque les indices comportent au maximum 2 décimales et la partie entière rarement 4 chiffres.

En règle générale, on utilise plusieurs indices qui peuvent être différents suivant la catégorie du métier. Il est évident que les métiers de la restauration n'ont que faire de l'indice du fil de cuivre ou de celui de la construction.

2.3. Les données

Voilà le décor planté; nous pouvons mettre quelques indices dans la table. Puis nous passerons au gros du travail, la création du moteur de calcul.

TypeIndice	Mois	Année	Indice
EBIQ	6	2007	100
EBIQ	6	2008	105.5
TCH	6	2007	114.4
TCH	6	2008	115.1
ICHTTS	6	2007	135
ICHTTS	6	2008	142

Notez que les indices EBIQ, TCH et ICHTTS sont des indices INSEE bien réels, seules les dates et les valeurs ne le sont pas. Par exemple TCH est l'un des indices des prix à la consommation, hors tabac et alcool.

Dans la table tblFormules, nous allons saisir quelques formules. Certaines ne feront pas du tout appel aux indices, rassurez-vous ! c'est volontaire !

N_Formule	Formule
1	(Sqr(4)*4)+1
2	_Montant*(0,15*(_EBIQ1/_EBIQ0)+0,2*(_TCH1/_TCH0)+0,65*(_ICHTTS1/_ICHTTS0))

Il n'y a pas de lettre 0 mais le chiffre zéro dans la dernière formule.

La formule numéro 1 contient Sqr qui est une fonction mathématique native de VBA. Elle renvoie la racine carré du nombre passé en argument. Si on la calcule de tête, voici ce que nous obtenons:

1. (sqr(4) *4)+1
2. (2 * 4) + 1
3. 8 + 1
4. 9

La deuxième formule est un peu plus complexe. Vous remarquerez que nous avons des opérateurs simples (+, *, /), des nombres décimaux (0,15 ; 0,2...) et également des éléments précédés d'un underscore (_). Ces derniers représentent les variables de la formule.

Admettons que la revalorisation se fasse en juin 2008 (06/2008) pour l'année précédente, soit juin 2007. Si nous recherchons les indices Année et Année-1, voici ce que nous obtiendrons :

1. _Montant*(0,15*(_EBIQ1/_EBIQ0)+0,2*(_TCH1/_TCH0)+0,65*(_ICHTTS1/_ICHTTS0))
2. _Montant*(0,15*(105,5/100)+0,2*(114,4/116)+0,65*(135/126.7))

Montant sera remplacé par le montant du contrat ou par 100 pour indiquer un pourcentage de revalorisation.

Vous savez tout sur la revalorisation du moins sur la partie calcul.

3. De la formule au résultat

Deux solutions s'offrent à nous, la première est de passer par

Excel, de lui laisser faire les calculs puis de réintégrer les résultats dans la base ACCESS; très simple quand il s'agit d'une formule avec une seule date. Mais que se passe-t-il lorsque vous avez 300 contrats, 10 formules différentes et presque autant de dates de révision ?

Tout faire passer dans une feuille Excel pré-programmée ? Un peu lourd et surtout long et sujet à problèmes.

3.1. MS ACCESS sur un air d'EXCEL ou comment créer son propre moteur de calcul

Ouvrez un module et créez une fonction publique que vous appellerez **Calcule()** (pas le nom mais le verbe conjugué) avec un paramètre String en entrée et un type Variant en sortie.

```
Public Function Calcule(cFormule as String) as Variant  
  
End Function
```

Nous avons choisi un Variant en sortie pour nous permettre de calculer n'importe quel type de données, du numérique au texte.

Dans un premier temps, nous allons renvoyer le résultat de la fonction mathématique **Sqr(4)**. Ecrivez

```
Calcule = Sqr(4)
```

Passer dans la fenêtre d'exécution puis écrivez.

```
? calcule("")
```

Le raccourci pour accéder à la fenêtre d'exécution dans VBE et **ctrl+G**.

Validez. Bravo ! Vous avez réussi à faire une fonction qui renvoie la racine carré de 4.

3.2. Alimentez le moteur de calcul

Pour bien comprendre comment va fonctionner notre fonction de calcul, nous allons passer par plusieurs étapes. Remplacez la ligne de la fonction **Calcule**

```
Calcule = sqr(4)
```

Par la ligne suivante.

```
Calcule = Eval("sqr(4)")
```

Passer dans la fenêtre d'exécution puis écrivez.

```
? Calcule("")
```

La fonction renvoie toujours 2 ? C'est que ça fonctionne.

3.3. Quelques mots sur la fonction EVAL

La fonction Eval() permet d'évaluer des expressions de type string (texte) valides. Inutile d'essayer d'exécuter tout et n'importe quoi, seules quelques fonctions natives et celles de l'utilisateur fonctionnent. Ce qui ne fonctionne pas :

```
Eval("msgbox ""bonjour""")
```

Ce qui fonctionne :

```
Eval("msgbox (""Bonjour"")")
```

Notez que le complément **GenMessenger** disponible gratuitement sur ce site utilise cette fonction pour afficher la boîte de message composée par l'utilisateur.

3.4. Concrétisez le moteur de calcul

Après cet intermède ludique et vulgarisateur, revenons à notre moteur de calcul. Remplacez la ligne :

```
Calculer = Eval("Sqr(4)")
```

Par celle-ci.

```
Calculer = Eval(cFormule)
```

Et testons cette modification dans la fenêtre d'exécution en tapant ceci.

```
? calculer("sqr(4)")
```

Mais non ! Ne croyez pas que notre fonction ne sait renvoyer que 2. Essayez avec une autre valeur ou d'autres fonctions et formules !

```
? calculer("sqr(8)")
? calculer("Year(Date)-1")
```

Voilà notre moteur est terminé.

4. Calcul des formules

Il existe plusieurs méthodes pour lire le contenu d'une table ; le `dlookup` ou le `recordset`. Utilisons le `recordset`. Allez dans le menu **Outils/Références** et sélectionnez **Microsoft DAO 3.6 Object Library**.

Créez une nouvelle fonction que nous appellerons **LireFormules()**.

```
Public Function LireFormules(idFormule As Integer) As String
    ' idFormule est le n° de la formule dans la table
    'Auteur Fabrice CONSTANS (loufab)
    http://loufab.developpez.com 2009
    Dim strCrit As String
    Dim rst As Recordset
    ' déclaration de l'objet
    Set rst =
CurrentDb.OpenRecordset("tblFormules", dbOpenSnapshot)
    ' initialisation du recordset
    With rst
        strCrit = .Fields(0).Name & " = " & idFormule
    ' composition du critère de recherche
        .FindFirst strCrit
    ' recherche
    If Not .NoMatch Then Debug.Print
Calculer(.Fields(1)) ' trouve et affiche le résultat
        .close
    ' fermeture
    End With
    Set rst = nothing
    ' libération
End Function
```

Dans la fenêtre d'exécution, testons cette nouvelle fonction. Tapez

```
? LireFormules(1)
? LireFormules(2)
```

Nous sommes maintenant capables d'utiliser une formule issue d'une table.

5. Fonction d'interprétation des formules.

Pour l'instant, nous évaluons une formule simple. Pour évaluer nos formules de revalorisation, il faut remplacer les indices par leur valeur respective. Nous avons besoin de 2 dates, la date de revalorisation et la date d'origine qui détermineront quels indices utiliser dans la période. L'intervalle étant fixé contractuellement, vous pourrez donc rencontrer des cas différents et de multiples formules.

On peut rencontrer plusieurs types d'indices mais toujours par paire, selon la formule suivante.

IndiceOrigine / indiceActuel

Voici la fonction :

```
Function Revalorise(Montant As Currency, DateOrigine As Date, DateActuelle As Date, iFormule As Integer) As Currency
    'Montant est le montant à revaloriser
    'DateOrigine date d'origine ou date la plus ancienne
    'DateActuelle date de revalorisation
    'iFormule est le N° de la formule
    'Auteur Fabrice CONSTANS (loufab)
    http://loufab.developpez.com 2009
```

```
On Error GoTo Err_Revalorise
```

```
Dim fml As String ' la formule
Dim FrstCar As Integer ' Numéro du caractère traité
Dim Indice As String ' abreviation de l'indice dans la formule
Dim valindice As Double '
```

```
fml = LireFormule(iFormule) ' on récupère la formule dans la table
```

```
fml = Replace(fml, "_Montant", Montant) ' on remplace le _Montant par sa valeur
```

```
Do
```

```
    FrstCar = 1
    While Not Mid(fml, FrstCar, 1) = "_" ' fait défiler les chiffres, signe jusqu'au "_"
        FrstCar = FrstCar + 1
    Wend
```

```
    FrstCar = FrstCar + 1 ' on se positionne sur le nom de l'indice
```

```
    While Mid(fml, FrstCar, 1) Like "[A-Z]" ' tant que c'est une lettre on traite le nom de l'indice
        Indice = Indice & Mid(fml, FrstCar, 1) ' on récupère l'indice
        FrstCar = FrstCar + 1
    Wend
```

```
    Indice = Indice & Mid(fml, FrstCar, 1) ' pour la terminaison
```

```
    ' 0 = indice ancien -> contrat
```

```
    ' 1 = indice actuel -> tbl_indice
```

```
    If Right(Indice, 1) = 1 Then ' chercher l'indice 1 étant le plus ancien
        valindice = DLookup("Indice", "tblIndices", "Mois = " & Month(DateActuelle) & " And Annee=" & Year(DateActuelle))
```

```
    Else ' Indice contrat
        valindice = DLookup("Indice", "tblIndices", "Mois = " & Month(DateOrigine) & " And Annee=" & Year(DateOrigine))
```

```

End If
' remplace dans la formule
fml = Replace(fml, "-" & Indice, valindice, , 1) '
remplace 1 seule fois l'indice et son caractère de
répérage.

Indice = ""

Loop While InStr(1, fml, "-") > 0 'Il n'y a plus
d'indice à traduire

fml = Replace(fml, ",", ".") 'remplace chaque virgule
par un point
Revalorise = Calcule(fml) ' retourne le résultat
de l'opération.

Exit Function

Err_Revalorise:
If Err = 94 Then
    Revalorise = 0
    Exit Function
ElseIf Err = 6 Then
    fml = Replace(fml, ",", ".") 'remplace chaque
virgule par un point
    Revalorise = Calcule(fml)
Else
    MsgBox Err & " -- " & Err.Description
    Revalorise = 0
    Exit Function
End If
End Function

```

Cas particulier ! L'abréviation de certains indices est composée de chiffres. C'est notamment le cas du **BT48** (indice de la construction). Vous devez remplacer les chiffres par autre chose; en effet, la fonction ci-dessus ne reconnaît pas les chiffres dans les abréviations.

5.1. Virgule ou point, le séparateur décimal

Le séparateur décimal pose un réel problème pour notre moteur de calcul. En effet, lorsque l'on saisit un nombre décimal dans ACCESS, c'est la virgule qui est utilisée, du moins pour ceux qui ont ce réglage dans la configuration régionale de Windows. Pour VBA, il en va autrement ! Lui n'accepte que le point, mais la virgule est utilisée pour la séparation de tout paramètre.

Ceci peut poser un problème lors de la composition de vos formules; dans ce cas utilisez les fonctions de conversions suivantes :

Fonctions	Syntaxe avec le moteur	Description
CCur()	Calcule("CCur("10,2")...")	Converti l'expression en format monétaire
CDate()	Calcule("cdate("10/2/2009")")	Converti la date en double
CDbl()	Calcule("cdb("10,25")")	Converti l'expression au format double

La règle est donc simple; il faut doubler la double-cote dans la chaîne. Comme pour toutes chaînes de caractères.

5.2. Quelques tests

Après cette importante parenthèse, nous allons procéder à des tests de notre formule. Dans la fenêtre d'exécution, essayez les

syntaxes suivantes.

```

? revalorise(100,"1/6/2007","1/6/2008",4)
105,5
? calcule("year(cdate("2/8/2008"))-1")
2007
? calcule("replace("c_est pas bon","_","",chr(149))")

```

Voilà nous pouvons évaluer à peu près toutes les fonctions, même celles que vous développerez.

Notez que si vous entrez une date avec les séparateurs de date #, celles-ci seront interprétées comme des dates au format anglo-saxon (mois / jour / année), ce qui est très gênant dans notre cas.

6. Exploitation

il n'y a pas de secret, une fonction reste une fonction et à ce titre elle peut évidemment être utilisée dans une requête. Ce qui pour une revalorisation en masse est un sérieux avantage.

Commençons par créer une petite table de "contrat" avec juste l'essentiel.

Table du champ	Type du champ Taille	Description
tblContrat		
TypeFormule	Numérique - Entier	Numéro de la formule utilisée pour ce contrat. Correspond au numéro identifiant la formule dans la table tblformule
Montant	Monétaire - Euro	Le montant du contrat.
DerniereReval	Date/Heure	La date de la dernière revalorisation.

Remplissez le contrat en utilisant des numéros de formules existants (2) et dont les dates correspondent à celles saisies dans les indices (1/6/2007).

Créez ensuite la requête comme suit.

```

SELECT tblContrat.typeFormule, tblContrat.Montant,
tblContrat.DernierReval,
Revalorise ([Montant], [DerniereReval], DateAdd("yyyy", 1,
[DerniereReval]), [TypeFormule])+tblContrat.Montant AS
NouveauTarif
FROM tblContrat;

```

Vous avez noté que nous utilisons la formule Revalorisation() directement dans la requête et que ses paramètres sont les champs de la table.

La seule chose que nous pouvons commenter c'est la fonction native DateAdd() que vous pouvez retrouver dans le tutoriel, Les fonctions Date / Heure de Maxence Hubiche ([Lien94](#)). Elle permet de rajouter 1 an à la date de dernière revalorisation. Pour calculer le nouveau prix, nous ajoutons l'ancien à la revalorisation. Vous n'avez plus qu'à transformer cette requête en requête Modification (Update) pour avoir une revalorisation automatisée.

Retrouvez la suite de l'article de Fabrice Constans en ligne : [Lien95](#)

Les derniers tutoriels et articles

Tutoriels Direct3D 10 : Bases – Partie 1 : Les bases de Win32

Une série de tutoriels a été créée afin d'apprendre à un programmeur les bases de la programmation de l'API Direct3D 10.

Ils couvrent la mise en place et les procédures basiques afin d'accomplir les tâches fondamentales, comme dessiner et transformer.

Découvrez le premier article de cette série : Les bases de Win32.

Les utilisateurs plus expérimentés migrant vers Direct3D 10 peuvent passer ces explications et aller voir les changements effectués dans cette itération de l'API.

Le SDK DirectX est téléchargeable ici ([Lien96](#))

1. Résumé

Dans ce tutoriel préliminaire, nous allons passer en revue les éléments nécessaires pour créer une application Win32. Nous allons préparer une fenêtre vide pour l'utilisation de Direct3D 10.

2. Source

(SDK root)\Samples\C++\Direct3D10\Tutorials\Tutorial00

3. Préparation de la fenêtre

Chaque application Windows requiert au moins un objet fenêtre. Avant même de voir les spécificités de Direct3D 10, notre application doit avoir un objet fenêtre fonctionnel. 3 choses sont nécessaires :

1. Enregistrer la classe de la fenêtre

```
//  
// Enregistrement de la classe de la fenêtre  
//  
WNDCLASSEX wcx;  
wcx.cbSize = sizeof(WNDCLASSEX);  
wcx.style = CS_HREDRAW | CS_VREDRAW;  
wcx.lpfnWndProc = WndProc;  
wcx.cbClsExtra = 0;  
wcx.cbWndExtra = 0;  
wcx.hInstance = hInstance;  
wcx.hIcon = LoadIcon(hInstance,  
(LPCTSTR)IDI_TUTORIAL1);  
wcx.hCursor = LoadCursor(NULL, IDC_ARROW);  
wcx.hbrBackground = (HBRUSH)(COLOR_WINDOW+1);  
wcx.lpszMenuName = NULL;  
wcx.lpszClassName = szWindowClass;  
wcx.hIconSm = LoadIcon(wcx.hInstance,  
(LPCTSTR)IDI_TUTORIAL1);  
if( !RegisterClassEx(&wcx) )  
return FALSE;
```

2. Créer un objet fenêtre

```
//  
// Création de la fenêtre  
//  
g_hInst = hInstance; // Stock l'identifiant de  
l'instance de notre variable globale  
RECT rc = { 0, 0, 640, 480 };  
AdjustWindowRect( &rc, WS_OVERLAPPEDWINDOW, FALSE );  
g_hWnd = CreateWindow( szWindowClass, L"Direct3D 10  
Tutoriel 0 : Preparation de la fenetre",  
WS_OVERLAPPEDWINDOW,  
CW_USEDEFAULT, CW_USEDEFAULT,  
rc.right - rc.left, rc.bottom - rc.top, NULL, NULL,
```

```
hInstance, NULL);
```

```
if( !g_hWnd )  
return FALSE;
```

```
ShowWindow( g_hWnd, nCmdShow );
```

3. Récupérer et distribuer les messages de cette fenêtre

```
//  
// Boucle principale des messages  
//  
MSG msg = {0};  
while( GetMessage( &msg, NULL, 0, 0 ) )  
{  
TranslateMessage( &msg );  
DispatchMessage( &msg );  
}  
  
LRESULT CALLBACK WndProc( HWND hWnd, UINT message,  
WPARAM wParam, LPARAM lParam )  
{  
PAINTSTRUCT ps;  
HDC hdc;  
  
switch (message)  
{  
case WM_PAINT:  
hdc = BeginPaint(hWnd, &ps);  
EndPaint(hWnd, &ps);  
break;  
  
case WM_DESTROY:  
PostQuitMessage(0);  
break;  
  
default:  
return DefWindowProc(hWnd, message, wParam,  
lParam);  
}  
  
return 0;  
}
```

Celles-ci sont les étapes nécessaires pour configurer un objet fenêtre qui est nécessaire pour chaque application Windows. Si nous compilons et démarrons ce code, nous verrons une fenêtre avec un fond blanc.

Retrouvez cette série d'article de Romain Perruchon, Cyril Doillon, et Arnaud Feltz en ligne : [Lien97](#)

Cel-shading : le rendu cartoon

Le cel-shading est une technique de rendu 3D de plus en plus utilisée dans les jeux et les applications 3D. Le principe est de se rapprocher de plus en plus des dessins animés ou des bandes dessinées. Ce tutoriel va vous donner les moyens d'arriver à de tels résultats par différentes techniques.

1. Présentation du cel-shading

1.1. Historique

A l'image de l'évolution de la peinture artistique, la réalisation de scènes 3D virtuelles a voulu se rapprocher de plus en plus de la réalité avec des environnements de plus en plus détaillés. Mais aux alentours des années 2000, un nouveau courant est arrivé qui voulait se détacher du réalisme pour se rapprocher d'un visuel beaucoup plus "dessins animés". L'optique de cela est de donner l'impression de se retrouver projeté dans l'univers de son enfance, dans lequel on pouvait maintenant entièrement interagir, comme si nous étions le "héros" de ce dessin animé.



Jet Set Radio (2000)

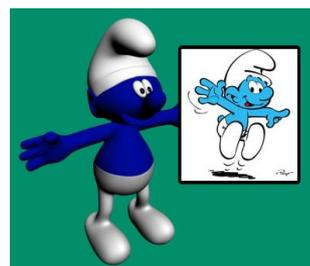
C'est ainsi que le "cel-shading" est né, littéralement "ombrage de celluloïd". Cette technique, aussi appelée "toon-shading", permet d'afficher tous types d'objets 3D, réalistes ou non, sous la forme de dessins illustrés comme les bandes dessinées. Cette technique est apparue pour la première fois au grand public par le jeu "Jet Set Radio" de Sega en 2000. Elle s'est ensuite popularisée par le succès du jeu "XIII" qui est une adaptation très réussie de la série de bandes dessinées du même nom. Dans "XIII", les développeurs avaient même ajouté des effets supplémentaires comme des bulles, des onomatopées, des cases en surimpressions, ... pour donner encore plus l'impression d'être plongé directement dans l'histoire à travers le héros. Il est très courant de voir actuellement de nouvelles productions avec ce type de rendu notamment pour des jeux qui veulent faire oublier le réel aux joueurs.



XIII (2003)

1.2. Principe

L'objectif du cel-shading est de modifier notre rendu pour s'approcher au maximum du dessin des bandes dessinées par exemple. Il faut donc trouver visuellement ce qui caractérise un dessin cartoon par rapport à tous autres dessins ou peintures.



Objet 3D VS Bande dessinée

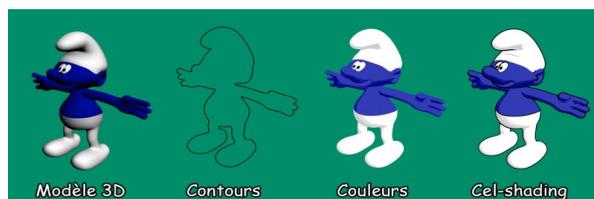
Tout d'abord, la première caractéristique qui saute aux yeux est le contour noir de chaque objet ou personnage du dessin. Ce trait noir, généralement à l'encre, correspond au trait du dessinateur pour dessiner ces objets. Qui n'a jamais griffonné de simples traits noirs pour représenter un petit personnage sur un coin de feuille ? Ce simple contour permet de se détacher du réel car, tout simplement, il n'existe pas dans la vie réelle. Cet élément est donc une des bases principales pour obtenir un rendu cartoon. Nous devons donc trouver une méthode pour créer cette bordure sur nos objets 3D.

Le trait noir était aussi utile pour se détacher de la nécessité de couleur dans les bandes dessinées surtout lorsque les planches apparaissaient dans des journaux en noirs et blanc.

Ceci est déjà intéressant pour afficher des scènes 3D sous forme de dessins cartoons, mais un autre élément entre en jeu. Les dessins cartoons sont souvent produits plus rapidement que les dessins réalistes. Est-il possible de passer une semaine sur un dessin d'une planche de bande dessinée quand l'intégralité de l'album est à réaliser ? Ainsi, les couleurs sont beaucoup moins travaillées, notamment au niveau de la qualité des jeux de lumières de l'environnement. Plutôt que de réaliser de beaux dégradés de couleurs, l'artiste va souvent se contenter de n'utiliser que quelques nuances de couleurs pour les différentes luminosités des objets. Il faut donc essayer de reproduire ce dégradé simplifié sur nos objets 3D.

Pour les anciennes bandes dessinées, le nombre de couleurs utilisables était souvent limité par l'imprimeur donc l'artiste devait obligatoirement se limiter.

Le cel-shading possède deux composantes bien distinctes : des dégradés de lumière simplifiés et des contours. Nous allons donc voir les deux effets indépendamment car aucun lien n'existe entre eux à part la volonté d'afficher un rendu cartoon. De plus, les deux techniques peuvent être utilisées l'une sans l'autre sans problème en fonction des besoins de votre application 3D ou de votre jeu.



Composition du cel-shading

2. Les techniques d'ombrage

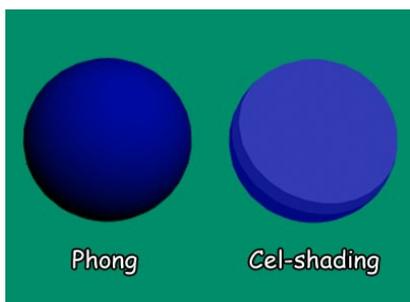
2.1. Rappel sur la lumière

Pour réaliser notre rendu cel-shading, nous allons commencer par la partie "ombrage simplifié" de la technique. Pour cela, nous allons tout d'abord se rappeler comment est calculée la lumière dans les modèles classiques de rendu. Le modèle le plus souvent utilisé est "l'ombrage de Phong". Celui-ci se compose de trois éléments de lumières : ambiant, diffuse et spéculaire.

- La lumière ambiante correspond à la couleur apparaissant dans la scène lorsque toutes les lumières sont éteintes.
- La couleur diffuse correspond à la réaction de l'objet à la lumière.
- La couleur spéculaire correspond à la lumière réfléchie sur l'objet.

Pour simplifier, nous n'allons pas prendre en compte cette lumière "spéculaire". La couleur résultante sur un point de l'objet s'obtient par un calcul très simple. Il suffit d'ajouter la partie ambiante et la partie diffuse, où cette dernière est modulée en fonction de l'angle formé entre la normale au point et la direction de la lumière.

L'ombrage résultant par cette technique sur une sphère est un dégradé de couleur allant d'une zone claire (zone éclairée directement) à une zone foncée (zone non éclairée). Ce dégradé provient de l'évolution de la normale de la surface de l'objet. Cela donne un réalisme et une profondeur à l'objet. Or, pour un rendu cartoon, nous voulons "casser" ce réalisme et donc ce dégradé en influant sur cette composante "diffuse".



Simplification de la lumière

2.2. Avec une texture 1D

Il est fortement conseillé de se renseigner sur le **Multitexturing** avant de lire cette partie.

La première approche pour créer un effet de lumière "simplifié" est l'utilisation d'une Texture 1D pour la lumière. La technique est assez simple mais propose une approche originale. Plutôt que de calculer explicitement la lumière en chaque point de l'objet, nous allons le simuler. Pour cela, nous allons décomposer la couleur diffuse de notre objet en deux éléments : couleur et lumière. La partie "couleur" correspond à la couleur de l'objet lorsqu'il n'est pas éclairé par une lumière. Celle-ci peut être une simple couleur pour la surface complète ou une texture 2D classique.

Pour la composante "lumière" de la couleur, nous allons utiliser une texture à une dimension qui va simuler notre dégradé de lumière. Plus la texture aura une grande résolution, plus la précision du dégradé sera grande. Comme vous l'avez sûrement compris, l'objectif pour nous est de réduire la qualité de la texture pour "simplifier" la lumière.

L'utilisation de cette texture nécessite souvent que la carte graphique accepte le multitexturage car nous avons besoin de deux textures par objet, une pour la couleur et une pour la lumière. La modulation de ces textures va se faire par

multiplication. Ainsi, les zones utilisant la partie foncée de la lumière apparaîtront foncées, et inversement pour la partie claire. Au niveau des coordonnées de texture, la partie "couleur" utilise les coordonnées classiques de notre objet et la partie "lumière" sera plus particulière. Pour chaque sommet, nous avons besoin d'une coordonnée entre 0 et 1 sur notre texture. Cette valeur sera calculée grâce à l'angle entre la direction de la lumière L et la normale au sommet N. Plutôt que de calculer l'angle directement, nous allons directement utiliser le produit scalaire des deux vecteurs. Pour un produit scalaire égal à 1, la lumière est maximale donc la coordonnée doit être à 1. Pour un produit scalaire inférieur ou égal à 0, la lumière est minimale donc la coordonnée doit être à 0. Ainsi :

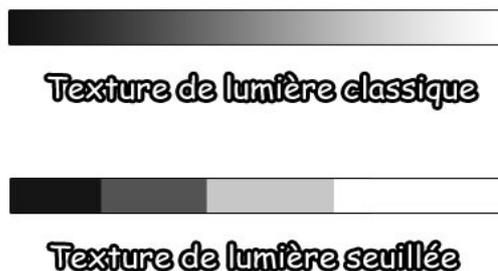
$$U = L.N$$

Si $U < 0$ alors $U = 0$

- Avec L, direction normalisée de la lumière
- Avec N, normale de la surface au point
- Avec U, coordonnée résultante sur la texture de "lumière"

Pour éviter le dernier test pour un produit scalaire négatif, nous pouvons définir la texture pour qu'elle se "prolonge" pour des valeurs sortant du domaine [0,1].

Par cette technique, il nous suffit donc de "seuiller" la texture de lumière pour obtenir notre effet "cartoon". Pour une texture à 16 niveaux de gris par exemple, nous pouvons la transformer pour n'utiliser que 4 niveaux et ainsi "simplifier" le modèle de lumière.



Texture 1D de lumière

2.3. Pixel Shader

Il est fortement conseillé de se renseigner sur l'utilisation des **Pixel Shader** (ou **Fragment Shader**) avant de lire cette partie.

Avec les évolutions des cartes graphiques, le rendu d'objet 3D réaliste a été grandement amélioré. L'apparition des "pixel shader", qui permettent un calcul défini par l'application pour chaque pixel de l'écran, fût l'une des améliorations qui augmenta les capacités des affichages. Grâce à eux, il est maintenant possible de calculer directement la couleur d'un pixel de façons très variées. On peut notamment calculer la lumière en un point de l'écran.

Le calcul utilisé dans les pixels shaders est identique aux méthodes classiques à savoir avec des parties ambiantes, diffuses et spéculaires. Nous n'avons ainsi plus besoin d'une texture supplémentaire dédiée à la lumière pour dessiner nos objets 3D car le calcul se fait directement sur la couleur du pixel. Grâce aux informations de normales et de lumière, il est possible de calculer notre produit scalaire, vu précédemment, pour obtenir notre "pourcentage" de noir entre 0 et 1 en ce point. Celui-ci pourra

ainsi être modulé par la couleur de l'objet, provenant d'une texture ou d'une simple couleur diffuse.

Notre objectif est toujours le même, "simplifier" les nuances de couleurs de la lumière. Nous pouvons donc agir directement sur notre valeur de lumière au sein du pixel shader. Il suffit donc de "seuiller" la valeur avec des valeurs prédéfinies. Par exemple :

```
U = L.N
Si U < 0.2 alors U = 0
Si U >= 0.2 et U < 0.4 alors U = 0.2
Si U >= 0.4 et U < 0.5 alors U = 0.4
Si U >= 0.5 alors U = 1
```

Nous obtenons ainsi une nuance de quatre couleurs répartie non-uniformément sur la plage de valeur [0,1]. Notre résultat est donc identique à la version précédente si toutefois, la texture 1D correspond au même seuillage que celui du pixel shader. Cette technique nécessite que la carte graphique supporte les pixels shader mais permet de gagner un emplacement de texture pour chaque objet 3D à dessiner.

3. Les contours

3.1. Multi-passes

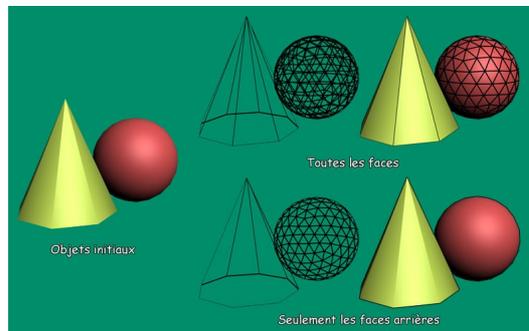
Il est fortement conseillé de se renseigner sur le **Backface Culling** avant de lire cette partie.

La deuxième grosse composante du "cel-shading" se situe dans l'affichage des contours de chaque objet. Contrairement à la partie précédente où nous modifions uniquement le dessin de l'objet, ici, nous voulons rajouter un élément supplémentaire. Comment dessiner des pixels noirs supplémentaires et qui suivent les contours de l'objet ?

La première approche que nous allons aborder va essayer d'utiliser la même géométrie que notre objet 3D. De base, chaque objet est dessiné par une composition de nombreux triangles. La particularité liée à l'utilisation de cette forme est que, quelle que soit l'orientation de l'objet, le contour de l'objet correspond toujours à une ou plusieurs arêtes de triangles. Nous allons donc utiliser cette propriété pour faire apparaître les contours de la forme.

Nous allons donc dessiner le modèle de l'objet une seconde fois mais en se contentant de dessiner les arêtes des faces. Si nous dessinons comme cela, nous n'aurons pas un résultat acceptable car l'objet sera dessiné normalement et chacun de ses triangles sera entouré d'un trait noir. Nous allons d'abord filtrer pour ne dessiner que les faces n'étant pas dirigées vers la caméra. Cela conserve effectivement les contours de l'objet car chaque bordure de celui-ci est composée au minimum d'un triangle vers la caméra et d'un autre dans le sens opposé. Nous évitons ainsi de dessiner les arêtes qui se situent au premier plan sur l'objet et nous conservons des surfaces colorées uniformes.

Le principe est donc de dessiner une première fois l'objet 3D texturé et coloré normalement et de le redessiner une seconde fois en mode filaire, en noir et uniquement les faces "arrières". Pour modifier l'aspect du contour, il est possible de changer la couleur du trait ainsi que son épaisseur pour forcer l'effet cartoon. Cette technique possède un désavantage assez important car, quelle que soit la distance de l'objet à la caméra, l'épaisseur du trait sera la même. Ainsi, plus l'objet s'éloignera, plus le contour sera prononcé sur celui-ci car l'épaisseur correspond à une taille fixe à l'écran.



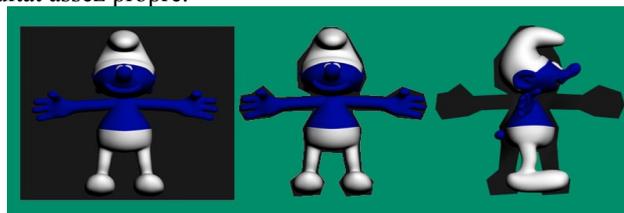
Création du contour d'objets 3D

3.2. Enveloppe

Il est fortement conseillé de se renseigner sur le **Backface Culling** avant de lire cette partie.

Nous nous sommes rendus compte que l'utilisation du même modèle pour dessiner ses contours posait des inconvénients car il oblige à dessiner deux fois chacun des objets et pouvait avoir des effets non désirés sur les objets petits ou éloignés. Pour éviter ce dernier effet, nous allons essayer d'ajouter un nouveau modèle pour chaque objet devant recevoir des contours. Mais comment doit être ce nouveau modèle ?

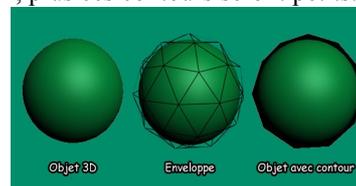
Le principe est simple. Prenons par exemple un simple rectangle noir et plaçons le derrière notre modèle du point de vue de la caméra. Ce rectangle va dépasser de notre modèle à certains endroits pour former un contour noir mais ne suivant pas exactement le modèle initial. Améliorons le rectangle pour qu'il suive assez fidèlement le contour du modèle, nous arrivons à un résultat assez propre.



Rectangle / Adaptation de la forme / Changement d'angle du modèle

Il nous reste cependant un problème, si la caméra bouge, le profil du modèle ne sera plus le même et il faudrait donc modifier notre forme de contour. Pour résoudre ce problème nous allons utiliser un mesh complètement en 3D qui va entourer globalement l'ensemble du mesh initial. Nous nous retrouvons ainsi avec des faces à l'avant du modèle et d'autres à l'arrière. Seules ces dernières nous intéressent donc nous allons dessiner l'enveloppe unique des faces dirigées vers la caméra et vers le centre de l'enveloppe. Ainsi n'apparaîtront que les faces formant le contour de l'objet.

Cette technique augmente significativement la quantité de triangles affichés pour un seul objet et cela peut poser des gros problèmes pour des scènes très complexes. Par contre, elle offre une plus grande souplesse sur la forme des contours qui ne sont plus limités à suivre exactement la forme de l'objet mais peuvent utiliser des angles beaucoup plus marqués. De même, le problème des tailles fixes des contours disparaît car les contours sont intégralement intégrés à la scène et aux objets 3D. Ainsi, plus l'objet sera loin, plus ces contours seront petits.



3.3. Filtre sur la profondeur

Avec les améliorations des cartes graphiques, les techniques de rendu ont aussi évolué. Une nouvelle technique appelée "deferred shading" propose une nouvelle approche du dessin de scène 3D. Cette technique a comme principal intérêt pour nous de calculer l'image finale par pixel de l'écran. Nous pouvons ainsi utiliser des techniques de traitement d'images pour créer les contours de nos objets 3D.

Le "deferred shading" a pour principe de dessiner chaque composante (couleur, normal, profondeur, ...) dans des textures différentes et ensuite de calculer la couleur de chaque pixel grâce à un calcul de lumière classique. L'intérêt de la technique est de pouvoir ajouter un grand nombre de lumières dynamiques sans impacter les performances.

Les techniques de traitement d'image utilisent les convolutions qui calculent un pixel donné à partir d'une zone de pixels centrée sur celui-ci. Des filtres usuels comme Sobel ou Prewitt permettent la détection des contours d'une image. Le principe de ces filtres est simple, si une différence (de couleur généralement) existe entre deux pixels consécutifs cela crée un contour plus ou moins prononcé sur ces pixels. Soit, pour un pixel, plus la différence entre sa valeur et les valeurs de pixels l'entourant est grande, plus le contour sera marqué. Généralement, nous n'utilisons que les pixels directement consécutifs soit un calcul sur 9 pixels (filtre 3x3), mais il peut être utile de prendre des niveaux supplémentaires sur 25 pixels par exemple (filtre 5x5).

Vous pourrez trouver plus d'informations sur le traitement d'images dans les tutoriels : Introduction au traitement numérique d'image ([Lien98](#)) de Florent Humbert et Les filtres usuels en traitements d'images ([Lien99](#)) de Xavier Philippeau.

Dans notre cas, nous avons plus d'informations que sur les simples photographies car nous possédons les éléments de profondeur de l'image. Effectivement, un contour sur une image cartoon correspond à une différence de profondeur assez grande entre l'objet à entourer et ce qui se trouve derrière lui. Plutôt que d'appliquer notre filtre sur la couleur de l'image, nous allons directement l'appliquer sur notre image de profondeur. Ainsi, les grandes différences de profondeurs seront marquées par un contour.

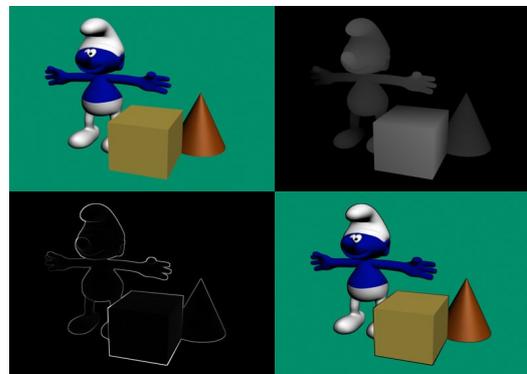


Image initiale / Tampon de profondeur / Détection de contours / Image finale

Cette technique a comme principal intérêt de réduire la quantité de géométrie utilisée pour chaque objet car dans les deux techniques précédentes, soit l'objet était dessiné deux fois, soit un modèle supplémentaire intervenait. Ici, le calcul se fait directement en 2D sur l'image écran de la scène 3D grâce aux informations de profondeur de chaque pixel.

4. Conclusion

Nous avons pu voir dans ce tutoriel les différentes techniques utilisées pour dessiner des scènes 3D à la façon des bandes dessinées ou des dessins animés. La technique de "cel-shading" n'est donc pas une technique mais un ensemble de deux composantes : les couleurs simplifiées et les contours. Dans vos applications 3D ou vos jeux, il est tout à fait possible d'utiliser l'une sans l'autre en fonction de l'aspect que vous voulez donner à vos modèles 3D. Il est important de faire des compromis, notamment pour les contours, entre la quantité de géométrie, la difficulté de création des modèles et les performances pour trouver la méthode qui vous correspond le mieux. Le "cel-shading" continue son évolution au fil du temps par l'apparition de graphisme plus ou moins subtil comme le dernier Prince of Percia qui utilise des textures assez détaillées et fines ainsi que des contours pour un rendu à mi-chemin entre le cartoon et le réalisme.



Prince of Percia (2008)

Le modèle 3D du schtroumpf provient du site 3DValley : ([Lien100](#))

Retrouvez l'article de Cyril Doillon en ligne : [Lien101](#)

Reportage sur l'Ankama Convention #3

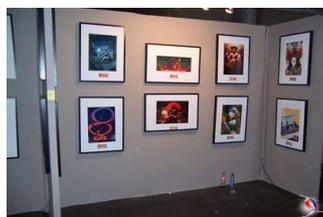
Les 20 et 21 décembre 2008 dernier, à Lille, a eu lieu la troisième édition de l'Ankama Convention organisée par le groupe Ankama notamment connu pour son MMORPG en partie gratuit Dofus. Cet événement sera notamment l'occasion de découvrir la nouvelle version de Dofus mais également de mettre les mains sur leur nouvelle production, Wakfu.

1. La convention

C'est en cette veille des fêtes de fin d'année, le weekend du 20 et 21 décembre 2008, qu'Ankama a décidé d'organiser la troisième édition de sa convention qui lui est entièrement dédiée, l'Ankama Convention #3. Ankama, principalement connu pour son original MMORPG partiellement gratuit Dofus, est devenu en quelques années l'un des leaders du genre. Intégrant maintenant de nombreuses filiales spécialisées dans divers domaines comme l'édition ou le jeux-vidéo, Ankama a décidé de rassembler tous ces domaines d'activités lors de cette convention pour montrer ces dernières nouveautés et permettre aux joueurs et aux fans de l'univers, de rencontrer les équipes de productions.



Organisé en différents stands, l'Ankama Convention #3 est composée de tous les représentants du groupe Ankama. On trouvera bien évidemment toute une partie consacrée aux jeux-vidéo d'Ankama dont Dofus mais également la future production : Wakfu. Ces productions ont été l'occasion pour Ankama de sortir de nombreux produits dérivés comme des magazines de presse, des bandes dessinées et des dessins animés. La convention est donc l'occasion de rassembler tous ces produits et de les faire découvrir ou redécouvrir au public de lillois rassemblé dans le Grand Palais.



Côté jeu vidéo, Dofus et Wakfu ont pu se montrer sous de nouvelles facettes aux fans de ces séries. De nombreux PC étaient à disposition pour essayer l'une des premières versions jouables de Wakfu. On pouvait également s'essayer à la nouvelle version de Dofus 2.0 qui présentait des graphismes remodelés et de nouvelles fonctionnalités de jeu. Dofus Arena et Dofus : Serveur Héroïque étaient également disponibles à l'essai pour découvrir de tout nouveaux concepts de jeux et de gameplay basés sur l'univers Dofus.



De nombreuses animations étaient présentes parmi les 4000m² d'espace du Grand Palais de Lille. Ainsi, en permanence étaient projetés des épisodes de la série animée Wakfu, tirés du jeu-vidéo

éponyme. On pouvait également retrouver des conférences permettant aux fans de poser leurs questions aux développeurs de Dofus et Wakfu. Des séances de "Art Live" étaient également proposées qui permettaient de montrer la réalisation d'illustrations de A à Z en direct. La chaîne télévisée "NoLife" spécialisée en Jeux vidéo et culture japonaise était de la partie avec des émissions enregistrées lors du salon.



2. Dofus

2.1. Dofus 2.0

Lors de la convention, Ankama a mis à disposition plus de 150 PC dédiés à la nouvelle version de son MMORPG phare : Dofus 2.0. Cette organisation a permis, aussi bien aux jeunes fans de l'univers original qu'aux susceptibles nouveaux joueurs de ce MMORPG, de découvrir une partie des nouvelles fonctionnalités de cette nouvelle bouture.

Dofus 2.0 consiste en de nombreuses améliorations. La partie "client" du jeu a été complètement repensée et réécrite. L'objectif de cela est d'optimiser les connexions avec les différents serveurs du jeu ainsi que d'améliorer ses performances globales. Les équipes d'Ankama ont également pensé au futur par l'ajout de possibilités d'extensions simplifiées.

Mais la principale évolution de cette nouvelle version réside en la refonte de l'intégralité des graphismes du jeu. Chaque composant du jeu a été redessiné que ce soit des armes, en passant par l'interface, jusqu'aux éléments de base des décors. Ankama promet qu'avec cette nouvelle version, aucun compte ne sera réinitialisé ou remis à zéro et que les joueurs conserveront leurs personnages actuels ainsi que leurs équipements.



2.2. Dofus : Serveur Héroïque

Un serveur de Dofus très spécial était également présent sur la convention : le serveur Héroïque. Celui-ci propose une nouvelle expérience de jeu aux fans de l'univers du MMORPG d'Ankama. Avec le même jeu, il est possible de découvrir une nouvelle façon de jouer car le serveur héroïque n'autorise pas l'échec car toute mort des personnages des joueurs est définitive.

Un nouveau challenge et de nouvelles stratégies, c'est ainsi que l'on pourrait définir ce nouveau mode de jeu original. De nombreux ajustements gameplay ont été fait notamment sur la vitesse d'évolution de son personnage car l'XP ainsi que le taux d'objet récupérés ont été triplé. De plus, lorsqu'un personnage meurt, l'intégralité des ses objets revient au joueur ou au monstre vainqueur. Pour accéder à ce serveur, il vous suffit de vous connecter directement avec votre compte Dofus classique sur le

serveur nommé "Oto Mustam"



2.3. Dofus Arena

50 PC, 600 canettes Dofus Arena Max, 100 tee-shirts collector, voilà en quelques chiffres ce que représentait le stand Dofus Arena de l'Ankama Convention. Pour rappel, Dofus Arena est un jeu à part entière dans l'univers de Dofus. Jouable par l'ensemble des joueurs du MMO phare d'Ankama, ce jeu se définit comme un jeu de combat stratégique en ligne tour par tour permettant d'affronter n'importe quel autre joueur. La convention a été l'occasion de découvrir ou redécouvrir ce mode de jeu mais a également marqué la fin de la Dofus Arena League de 2008.

32 PC étaient donc réservés aux matchs officiels de la league Dofus Arena. Ce concours se terminait par la finale qui fut l'occasion de récompenser le champion 2008.



2.4. Dofus Pocket et Dofus Arena Pocket

Ankama a voulu montrer sa polyvalence au sein de la convention en montrant deux alternatives portables de leurs jeux en ligne. C'est ainsi que les joueurs pouvaient s'essayer à Dofus Pocket et Dofus Arena Pocket.

Le principe de Dofus Pocket est d'accéder, via son téléphone portable, à une grosse partie de l'univers Dofus. On peut, par exemple, consulter l'encyclopédie Dofus qui contient toutes les informations utiles sur ce monde original. Que ce soit les monstres, les objets, les drops, tout est disponible. Vous pourrez également consulter toutes les informations de vos personnages, vos informations de santé, nourrir vos familiers, ... Dofus Pocket vous permet également de gérer votre magasin si vous êtes resté en mode "marchand" en vous déconnectant, et cela, sans être à proximité d'un ordinateur.

Contrairement au précédent, Dofus Arena Pocket est entièrement dissocié du jeu Dofus Arena. Cette version propose une version allégée du jeu initial permettant de faire des parties simples et rapides entre amis par connexion Bluetooth. Cette version mobile donne un gameplay simple et compréhensible rapidement par n'importe quel joueur passionné de l'univers Dofus. C'est donc l'occasion de s'essayer à cet univers sans avoir à rester des heures devant l'écran de son ordinateur.



3. Wakfu

3.1. Le jeu

L'Ankama Convention #3 était l'occasion de découvrir la nouvelle production d'Ankama en matière de jeux en ligne : Wakfu. Wakfu se présente comme un nouveau jeu de rôle en ligne massivement multijoueur (MMORPG) à l'image de son antécédent qu'est Dofus. L'univers se compose de combats, de magies et d'humour dans un thème original. Le joueur incarnera un personnage dont il aura choisi la classe et combattra monstres et autres joueurs à partir d'un système de combats simple et intuitif.

Wakfu propose une nouvelle expérience de jeu sur le même thème que Dofus. Ankama nous annonce un "univers persistant avec un écosystème autonome où les monstres ont une IA propre [...], ils évoluent au milieu de 25000 joueurs qui progressent en simultanée". Chaque personnage que le joueur aura créé et entièrement personnalisé, que ce soit des armes aux vêtements ou de la peau à la coiffure, sera unique. Il pourra se déplacer grâce à un "scrolling multidirectionnel dans un univers en 3D isométrique" contrairement à Dofus où le déplacement se faisait par zone de la taille d'un écran.



3.2. Concept cross-média

Wakfu pourrait être un nouveau MMORPG comme les autres mais Ankama a d'autres ambitions. Ils veulent créer un univers particulier qui sera présent sous de nombreux formats et des nombreux médias différents. Wakfu sera bien présent au travers du jeu vidéo mais se complètera par des éditions consoles DS et Xbox360 par exemple, par des séries animées diffusées sur France 3 et des applications pour téléphones portables. On trouvera également Wakfu sur internet et sous formats papiers (manga, presse, ...) mais également sous forme d'un Jeu de Carte à Collectionner qui a été présenté lors de la convention.

Ankama nous promet de nombreuses interactions entre les différents médias comme des annonces dans le jeu pour des épisodes de la série ou inversement, des quêtes annexes de la série animée poursuivies et synchronisées dans le jeu vidéo. Ceci est un concept très novateur qui peut fournir une nouvelle expérience vidéo ludique qui commence à s'infiltrer dans le réel. Ce projet devient concret par une version Beta de Wakfu déjà jouable, des épisodes animés Wakfu présentés à la convention, un jeu de carte testable par les joueurs. Ankama a récemment signé avec Microsoft pour un projet "Island of Wakfu" pour le Xbox Live Arcade dans la lignée de ce concept "cross-média".



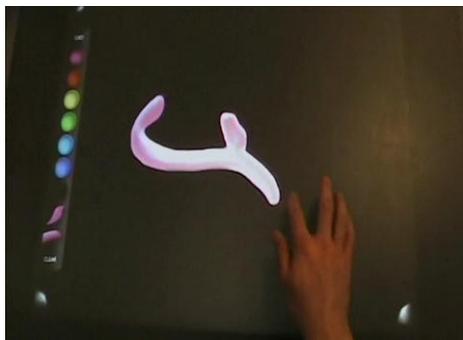
Retrouvez l'article de Cyril Doillon en ligne : [Lien102](#)

La table Surface de Microsoft aux prochaines TechDays 2009

Microsoft a annoncé, il y a peu de temps, la table tactile multitouch de Microsoft, baptisé *Microsoft Surface* sera présente aux TechDays 2009 qui se dérouleront le 10 février prochain au Palais des Congrès de Paris.

Peut être avez vous déjà vu cette table en action où l'on voit les utilisateurs grossir et tourner des photos, ... Qui n'a jamais rêver de dessiner directement avec un doigt ou plusieurs doigts sans se salir ? Le principe est simple, on connaît déjà les écran tactile à un point voir deux points supportés en même temps, ici, toute surface ou ensemble de points est détecté.

Au niveau multimédia et jeux, on peut très vite imaginer tout ce qui peut résulter comme application de ce genre de technologie. Imaginez un jeu de stratégie en face à face où l'on dirige ses troupes du bout des doigts, pensez à un jeu de carte solitaire où l'on peut déplacer plusieurs carte en même temps directement avec ses mains, ...



Cela peut vraiment apporter beaucoup au domaine du high-tech et plus particulièrement au multimédia.

Vous pouvez découvrir cette table en action ici ([Lien103](#))

Avez vous une opinion sur cette table ? des idées d'utilisations ?

Les chiffres des meilleures ventes de consoles et jeux

En ce début d'année 2009, il est temps de faire un bilan des ventes mondiales dans les jeux vidéos sur consoles.

Pour les consoles de salons, on peut voir que la Nintendo Wii surpasse ses concurrentes avec plus de 46 millions de consoles vendues depuis son lancement dont plus de 3 millions en France et 26 millions pour cette seule année 2008. La Xbox360 de Microsoft ainsi que la Playstation3 de Sony comptent respectivement 27 et 19 millions de ventes soit l'équivalent de la Wii a elles deux.

Coté consoles portables, la Nintendo DS dépasse largement sa rivale, la PlaystationPortable (PSP) de Sony avec 97 millions de ventes dont plus de 6 millions sur le territoire français. Un tiers de ces ventes se sont cumulées en 2008.

	Total (2008)	France (2008)	Total	France
CONSOLES DE SALON				
Wii	26 457 179	1 833 114	46 061 772	3 189 022
PS3	10 606 714	775 721	19 705 525	1 332 637
X360	11 524 711	576 960	27 651 986	1 198 314
CONSOLES PORTABLES				
PSP	13 099 298	543 486	43 594 223	1 996 926
DS	33 630 869	2 604 916	97 537 446	6 884 593

Pour les jeux vidéos, Wii Sport de Nintendo sur Wii vient de dépasser le record mondial de Super Mario Bros sur la Nintendo NES avec plus de 41 millions de ventes depuis sont lancement. On peut voir dans le top 10 des meilleurs ventes de jeux, la prédominance de Nintendo. Le premier jeu d'un autre éditeur vient de Rockstar Games qui place Grand Theft Auto: San Andreas à la 12e place avec 17,67 millions de ventes. Les Sims de Electronic Arts se place à la 16e place avec 16,08 millions de ventes.

TOP 10 des ventes mondiales			
Support	Titre	Editeur	Ventes (millions)
Wii	Wii Sport	Nintendo	41.78
NES	Super Mario Bros	Nintendo	40.24
Gameboy	Pokemon Rouge/Vert/Bleu	Nintendo	31.38
Gameboy	Tetris	Nintendo	30.26
NES	DuckHunt	Nintendo	28.31
Gameboy	Pokemon Or/Argent	Nintendo	23.11
DS	Nintendogs	Nintendo	21.68
SuperNES	Super Mario World	Nintendo	20.61
Wii	Wii Play	Nintendo	20.50
DS	New Super Mario Bros	Nintendo	18.55

Retrouvez ces billets sur le blog de la rubrique 2D/3D/Jeux : [Lien104](#)

Liens

- Lien1 : <http://www.jboss.org/tools/>
Lien2 : <http://download.jboss.org/jbosstools/updates/stable>
Lien3 : <http://download.jboss.org/jbosstools/updates/development>
Lien4 : <http://download.jboss.org/jbosstools/builds/nightly/latestBuild.html>
Lien5 : <http://lmellouk.developpez.com/tutoriels/jsf/jbosstools/>
Lien6 : <http://wicketstuff.org/confluence/display/STUFFWIKI/Wiki>
Lien7 : <http://www.amazon.fr/exec/obidos/ASIN/1932394982/wwwdeveloppec-21>
Lien8 : <http://www.tomsquest.com/blog/critique-du-livre-wicket-in-action/>
Lien9 : <http://java.developpez.com/livres/>
Lien10 : <http://www.wampserver.com/>
Lien11 : <http://notepad-plus.sourceforge.net/fr/site.htm>
Lien12 : <http://www.mamp.info/en/mamp.html>
Lien13 : <http://tuppis.com/smultron/>
Lien14 : <http://doc.ubuntu-fr.org/lamp>
Lien15 : <http://bluefish.openoffice.nl/download.html>
Lien16 : http://fr.wikipedia.org/wiki/Indice_de_masse_corporelle
Lien17 : <http://sylvie-vauthier.developpez.com/tutoriels/php/grand-debutant/>
Lien18 : <http://disruptive-innovations.com/zoo/cssvariables/>
Lien19 : <http://www.journaldunet.com/developpeur/tutoriel/php/050218-php-css-dynamique.shtml>
Lien20 : <http://css4design.developpez.com/tutoriels/css/feuille-de-style-css-dynamique-avec-php/>
Lien21 : <http://cssglobe.developpez.com/tutoriels/css/histogramme/fichiers/>
Lien22 : <http://cssglobe.developpez.com/tutoriels/css/histogramme/>
Lien23 : <http://dico.developpez.com/html/1693-Internet-CSS-Cascading-Style-Sheets.php>
Lien24 : <http://www.yoyodesign.org/doc/w3c/css2/cover.html>
Lien25 : http://xhtml.developpez.com/faq/?page=html_generalites#html_doctypes
Lien26 : http://www.blog-and-blues.org/articles/Les_syntaxes_de_commentaires_conditionnels_pour_IE_Windows
Lien27 : <http://www.howtocreate.co.uk/fixePosition.html>
Lien28 : <http://j-willette.developpez.com/tutoriels/css/position-fixed/exemple/position-haut.html>
Lien29 : <http://j-willette.developpez.com/tutoriels/css/position-fixed/exemple/position-bas.html>
Lien30 : <http://j-willette.developpez.com/tutoriels/css/position-fixed/exemple/position-centre.html>
Lien31 : <http://j-willette.developpez.com/tutoriels/css/position-fixed/exemple/exemple-page.html>
Lien32 : <http://j-willette.developpez.com/tutoriels/css/position-fixed/>
Lien33 : <http://www.trolltech.com/>
Lien34 : <http://www.qtsoftware.com/>
Lien35 : [http://adserver.adtech.de/?adlink\[3.0\]224\[1429628\]\[1\]16\[AdId%3D1974983%3BBnId%3D8%3DBlink%3Dhttp%3A%2F%2Fqtsoftware.com](http://adserver.adtech.de/?adlink[3.0]224[1429628][1]16[AdId%3D1974983%3BBnId%3D8%3DBlink%3Dhttp%3A%2F%2Fqtsoftware.com)
Lien36 : <http://www.developpez.net/forums/d672056/c-cpp/bibliotheques/qt/qt-4-5-sous-licence-lgpl/>
Lien37 : <http://cpp.developpez.com/cours/?page=bibliotheques#tutoriels-boost>
Lien38 : <http://cpp.developpez.com/faq/cpp/?page=boost>
Lien39 : <http://www.boost.org/>
Lien40 : http://www.boost.org/doc/libs/1_37_0/libs/optional/doc/html/index.html
Lien41 : http://apais.developpez.com/tutoriels/c++/boost/in_place_factory/
Lien42 : <http://alp.developpez.com/tutoriels/traitspolicies/>
Lien43 : <http://loulou.developpez.com/tutoriels/cpp/metaprog/>
Lien44 : <http://apais.developpez.com/tutoriels/c++/boost/optional/>
Lien45 : <http://atranchant.developpez.com/web/curseur-mootools/>
Lien46 : <http://pckult.developpez.com/tutoriels/javascript/frameworks/jquery/lecture-fichier-xml/>
Lien47 : <http://script.aculo.us/>
Lien48 : <http://pckult.developpez.com/tutoriels/javascript/frameworks/scriptaculous/opacite-liste/fichiers/index.htm>
Lien49 : <http://pckult.developpez.com/tutoriels/javascript/frameworks/scriptaculous/opacite-liste/>
Lien50 : <http://www.whitepages.com/maps/>
Lien51 : <http://fr.php.net/apache>
Lien52 : http://fr.php.net/apache_note
Lien53 : <http://www.developpez.net/forums/>
Lien54 : <http://cyberzoide.developpez.com/php4/ip-to-country/>
Lien55 : <http://www.maxmind.com/>
Lien56 : <http://apache.developpez.com/cours/geolocalisation/>
Lien57 : <http://jm-mace.developpez.com/tutoriel/flex/internationalisation/>
Lien58 : <http://fadace.developpez.com/ase/reduction>
Lien59 : http://infocenter.sybase.com/help/topic/com.sybase.dc70204_1500/pdf/ase15pst.pdf
Lien60 : <http://sybooks.sybase.com/nav/summary.do?prod=9938&lang=en&Submit.x=22&Submit.y=12&Submit=Submit&prodName=Adaptive+Server+Enterprise&archive=0>
Lien61 : <http://fadace.developpez.com/ase/autoextension/>
Lien62 : <http://refit.sourceforge.net/screen.html>
Lien63 : <http://mac.developpez.com/livres/>
Lien64 : <http://www.developpez.net/forums/d344819/hardware-systemes-logiciels/autres-systemes/mac/livre-prefere-mac/>
Lien65 : <http://antiparser.sourceforge.net/>
Lien66 : <http://peachfuzz.sourceforge.net/>
Lien67 : <http://www.immunitysec.com/resources-freesoftware.shtml>
Lien68 : <http://packetstormsecurity.org/UNIX/misc/scratch.rar>
Lien69 : <http://wapiti.sourceforge.net/>
Lien70 : <http://gunzip.altervista.org/>
Lien71 : <http://browserfun.blogspot.com/>
Lien72 : <http://www.metasploit.com/users/hdm/tools/axman>
Lien73 : <http://metasploit.com/users/hdm/tools/hamachi/hamachi.html>
Lien74 : <http://www.ee.oulu.fi/research/ouspg/protos/>
Lien75 : <http://packetstormsecurity.org/UNIX/misc/snmp-fuzzer-0.1.1.tar.bz2>
Lien76 : <http://www.infigo.hr/files/ftp/fuzz.zip>

Lien77 : <http://www.secuobs.com/bss-0.6.tar.gz>
Lien78 : <http://www.packetfactory.net/Projects/ISIC>
Lien79 : <http://ip6sic.sourceforge.net/>
Lien80 : <http://packetstormsecurity.org/fuzzer>
Lien81 : <http://safari.oreilly.com/9780321446114>
Lien82 : <http://www.amazon.fr/Open-Source-Fuzzing-Tools-Rathaus/dp/1597491950>
Lien83 : http://media.blackhat.com/bh-usa-06/video/2006_BlackHat_Vegas-V18-Sparks_and_Others-Sidewinder.mp4
Lien84 : <http://www.journaldunet.com/solutions/securite/analyses/07/1105-logiciels-test-intrusion.shtml>
Lien85 : <http://thpierre.developpez.com/articles/fuzzing/>
Lien86 : <http://www.zimmergren.net/archive/2008/09/24/how-to-use-linq-and-the-net-3-5-framework-with-sharepoint-to-retrieve-sharepoint-list-items.aspx>
Lien87 : <http://www.codeplex.com/features/Release/ProjectReleases.aspx?ReleaseId=2502>
Lien88 : <http://www.codeplex.com/smartpart/Release/ProjectReleases.aspx?ReleaseId=10697>
Lien89 : <http://stephaneey.developpez.com/tutoriel/sharepoint/spgridview/>
Lien90 : <http://www.codeplex.com/wspbuilder>
Lien91 : <http://www.asp.net/AJAX/AjaxControlToolkit/Samples/Rating/Rating.aspx>
Lien92 : <http://www.asp.net/ajax/ajaxcontroltoolkit/samples/>
Lien93 : <http://stephaneey.developpez.com/tutoriel/sharepoint/sp35/>
Lien94 : <http://mhubiche.developpez.com/vba/fiches/comprendre/fonctions/datetime/>
Lien95 : <http://loufab.developpez.com/tutoriels/access/formulesmathematiques/>
Lien96 : [http://msdn.microsoft.com/fr-fr/directx/aa937788\(en-us\).aspx](http://msdn.microsoft.com/fr-fr/directx/aa937788(en-us).aspx)
Lien97 : <http://msdn.developpez.com/direct3d/10/tutoriels/base/>
Lien98 : <http://humbert-florent.developpez.com/algorithmique/traitement/introduction/>
Lien99 : <http://xphilipp.developpez.com/articles/filtres/>
Lien100 : http://www.3dvalley.com/models/models_characters.shtml
Lien101 : <http://raptor.developpez.com/tutorial/rendering/celshading/>
Lien102 : <http://raptor.developpez.com/reportage/ankama/>
Lien103 : <http://blogs.technet.com/mstechdays/archive/2009/01/04/venez-d-couvrir-la-table-surface-aux-microsoft-techdays-2009.aspx>
Lien104 : <http://blog.developpez.com/jeux/>