Developpez Le Mag

Edition de Octobre - Novembre 2009.
Numéro 24.
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

Sommaire

Java/Eclipse	Page 2
PHP	Page 8
Dev. Web	Page 14
JavaScript	Page 22
(X)HTML/CSS	Page 28
SGBD	Page 30
MS Office	Page 36
C/C++/GTK/Qt	Page 43
Mac	Page 49
Conception	Page 51
2D/3D/Jeux	Page 56
Liens	Page 64

Editorial

La température baisse, mais chez Developpez.com on s'active d'autant plus pour vous fournir notre sélection de meilleurs articles, critiques de livres, et questions/réponses sur diverses technologies.

Profitez-en bien !

La rédaction

Article PHP

Comment PHP a-t-il obtenu tant de succès ?

Découvrez une interview de Rasmus Lerdorf réalisée par le magazine américain *Linux Format*.

par **La rédaction PHP**
Page 8

Article Développement Web

Passez à l'UTF-8 sans manquer une étape

Ce tutoriel va vous expliquer comment encoder votre site intégralement en UTF-8 sans louper une étape qui pourrait faire apparaître des caractères disgracieux.

par **Josselin Willette**
Page 14

Projet Coin : Les modifications du langage pour Java 7

Il y a quelques mois de cela naissait le projet **Coin** (Lien1), qui avait pour objectif d'étudier les différentes propositions d'évolution du langage pour leurs éventuelles intégration dans la prochaine version de **Java**.
Le projet a eu un certain succès en recevant plus de 70 propositions, et c'est désormais l'heure du bilan : on connaît enfin la liste des changements acceptés pour l'inclusion dans le JDK**7** : Project Coin: The Final Five (Or So) (Lien2).

On retrouve donc 7 propositions (dont deux en regroupent plusieurs) proposant des modifications plus ou moins complexes sur 7 thèmes :

- La syntaxe en losange (Diamond syntax)
- Simplified Varargs Method Invocation
- Amélioration des valeurs numériques
- Support syntaxique des Collections
- Switch sur des chaînes de caractères
- Gestion automatique des ressources (ARM)
- Support de la JSR 292 dans le langage

Voyons ceci d'un peu plus près...

L'héritage de Java 5.0
Commençons donc par ce qui pourrait être considéré comme des correctifs suite à l'intégration des **Generics** dans **Java 5.0**. Les changements suivants sont en effet destinés à combler quelques lourdeurs héritées de l'intégration des **Generics** dans le langage :

La syntaxe en losange (Diamond syntax)
J'en ai déjà parlé (Lien3) il y a quelques jours : il s'agit tout simplement d'une syntaxe permettant d'éviter la duplication de la déclaration des types paramétrés, qui dans tous les cas doivent impérativement être identiques.
La syntaxe en losange consiste tout simplement à laisser le paramétrage vide (le compilateur se charge alors de le déduire du contexte).
Ainsi, ceci :

```
Map<Integer, List<String>> map = new
HashMap<Integer, List<String>>();
```

Pourra être remplacé par cela :

```
Map<Integer, List<String>> map = new HashMap<>();
```

Simplified Varargs Method Invocation
Soyons directs : cette modification risque de passer inaperçue pour 99% des développeurs !
En effet on retrouve ici un problème assez obscur qui interdit l'utilisation de tableaux de type paramétré : pour diverses raisons il est impossible de créer des tableaux de type Generics, notamment car la vérification du typage n'est pas effectuée au même moment (à la compilation pour les Generics, et à l'exécution pour les tableaux), ce qui peut entraîner des états totalement incohérents et difficiles à résoudre...
Toutefois, l'ellipse (également introduite dans Java 5) utilise des tableaux en interne. De ce fait lorsqu'on utilise les Generics avec l'ellipse, on peut se retrouver implicitement à créer des tableaux de types paramétrés :

```
  Comparator<String> c1 = ...
  Comparator<String> c2 = ...
  Comparator<String> c3 = ...

  List<Comparator<String>> list =
Arrays.asList(c1, c2, c3); // warning
```

Le compilateur produit alors un warning peu explicite pour prévenir du problème potentiel :

```
Main.java:14: warning: [unchecked] unchecked
generic array creation
of type java.util.Comparator<java.lang.String>[]
for varargs parameter
```

Mais le gros problème c'est qu'on ne peut rien faire contre ce warning (si ce n'est utiliser l'annotation `@SuppressWarning`), et que le problème potentiel tant décrié dépend uniquement du cas où l'implémentation de la méthode modifierait son tableau contenant les varargs... ce qui est fortement déconseillé !
Le warning a donc été déplacé sur la définition de la méthode et non pas lors de chacune de ses utilisations, afin d'éviter de multiplier les warnings inutilement.

Le sucre syntaxique
Passons désormais à ce que l'on pourrait appeler du sucre syntaxique, c'est-à-dire de nouvelles syntaxes qui se contentent de générer un code similaire mais plus verbeux. Mais cela n'en demeure pas pour autant inutile !
Note : il s'agit ici de regroupement de plusieurs propositions, et de ce fait les spécifications exactes ne sont pas pas encore détaillées. Je présente ici les fonctionnalités qui devraient normalement être prises en compte.

Amélioration des valeurs numériques
En plus de la forme décimale (par défaut), octale (avec le préfixe '`0`') ou hexadécimale (préfixe '`0x`'), il sera possible d'utiliser directement une forme binaire avec le préfixe '`0b`' :

```
int value = 0b10000000; // 128
```

L'intérêt de cela étant bien sûr de simplifier les opérations bit à bit sans avoir à convertir les valeurs numériques dans une autre base...
Il sera également possible d'utiliser le caractère underscore ('_') de manière totalement arbitraire entre deux chiffres de

n'importe quelle valeurs numériques :

```
double amount = 1_000_000_000.00;  // 1 milliard
int color = 0xff_ff_ff;       // white
int binary = 0b11_1110_1000;    // 1000 en
binaire
```

Ceci n'aura bien sûr aucune incidence sur la valeur générée à la compilation ! Le seul et unique but de cela est simplifier la lecture du nombre par le développeur.

Support syntaxique des Collections
L'intégration dans le langage se verra également améliorer pour les Collections, avec notamment un accès indexé pour les `Lists` et les `Maps` (respectivement avec un entier et un type compatible). Par exemple on pourra utiliser quelque chose comme cela :

```
List<String> list = ...
Map<String, Integer> map = ...

String firstValue = list[0];
list[0] = "new-value";

Integer i = map[firstValue];
map[list[0]] = map["x"];
```

```
List<String> list = ...
Map<String, Integer> map = ...

String firstValue = list.get(0);
list.set(0, "new-value");

map.get(firstValue);
map.put( list.get(0), map.get("x"));
```

A noter également la possibilité d'écrire des collections très simplement, à la manière des tableaux :

```
// Les Lists sont déclarés via des crochets :
List<Integer> integers = [ 1, 2, 3, 4, 5, 6 ];

// Les Sets sont déclarés via des accolades :
Set<Double> doubles = { 1.25, 2.50, 3.75, 5.00,
6.25, 7.50 };

// Les Map sont déclarés via des accolades, avec
deux-points séparant la clef de la valeur :
Map<String, String> strings = {
  "French" : "Français",
  "English" : "Anglais",
  "Spanish" : "Espagnol"
};
```

Les collections ainsi créées sont immuables, mais étant donné que toutes les collections peuvent s'initialiser à partir des données d'une autre collection, on pourra écrire quelque chose comme cela pour obtenir une instance modifiable à souhait :

```
List<Integer> integers = new
LinkedList<Integer>([ 1, 2, 3, 4, 5, 6 ]);

Set<Double> doubles = new HashSet<Double>({ 1.25,
2.50, 3.75, 5.00, 6.25, 7.50 });

Map<String, String> strings = new HashMap<String,
String>({
  "French" : "Français",
  "English" : "Anglais",
  "Spanish" : "Espagnol"
});
```

Les nouvelles structures de contrôle
On dénombre également deux (plus ou moins) nouvelles structures de contrôle, toujours dans un objectif de simplification du code.

Switch sur des chaînes de caractères
Tout aussi bête que son nom l'indique, cette structure permettra donc d'utiliser des chaînes de caractères dans un **switch** :

```
String value = ...

switch (value) {
case "azerty":
  doSomething();
  break;
case "qwerty":
  doAnotherThing();
  break;
case "":
  // do nothing
  break;
default:
  doDefaultAction();
  break;
}
```

Bref cela permet d'éviter les fastidieuses successions de **if/else**...

Gestion automatique des ressources (ARM)
Nous voici dans un domaine déjà bien plus intérressant : la gestion des ressources. Ceux qui ont l'habitude de suivre mes interventions sur les forums savent sûrement qu'on touche là à un sujet qui m'est important, et qui figure d'ailleurs dans la **FAQ Java** : Comment libérer proprement les ressources ? (Lien4)
Le problème vient donc des ressources "externes"(fichiers, sockets, resultset/statement/connexion JDBC, etc.), non-gérables par le garbage-collector, et qui doivent donc être libérées manuellement et explicitement via un appel à la méthode `close()`. Et pour faire cela efficacement on est obligé de passer par un bloc **try/finally** par ressource, ce qui alourdit considérablement le code.
L'objectif d'ARM est de simplifier tout cela en gérant automatiquement et proprement la fermeture du flux via une nouvelle syntaxe :

```
try ( /* declarations des ressources */ ) {
  /* traitements sur les ressources */
}
```

La déclarations des ressources s'effectue à la suite du `try`, et la fermeture est implicite dès la fin du bloc correspondant.
Pour reprendre l'exemple de la FAQ :

```
public static void writeToFile(URL url, File
file) throws IOException {
  // 1 - Création de la ressource (Fichier)
```

```
  FileOutputStream fos = new
FileOutputStream(file);
  try {
    // 2 - Utilisation de la ressource (Fichier)

    // 1 - Création de la ressource (URL)
    InputStream is = url.openStream();
    try {
      // 2 - Utilisation de la ressource (URL)
      byte[] buf = new byte[2048];
      int len;
      while ((len = is.read(buf)) > 0) {
        fos.write(buf, 0, len);
      }
    } finally {
      // 3 - Libération de la ressource (URL)
      is.close();
    }
  } finally {
    // 3 - Libération de la ressource (Fichier)
    fos.close();
  }
}
```

On pourra alors utiliser directement le code suivant, et donc se concentrer sur le traitements :

```
public static void writeToFile(URL url, File
file) throws IOException {
  // 1 - Création des ressource (Fichier &
URL) :
  try ( FileOutputStream fos = new
FileOutputStream(file);
    InputStream is = url.openStream() ) {
    // 2 Utilisation des ressources (Fichier &
URL) :
    byte[] buf = new byte[2048];
    int len;
    while ((len = is.read(buf)) > 0) {
      fos.write(buf, 0, len);
    }
  }
}
```

On obtient en résultat un joli gain de visibilité tout en gérant proprement ses ressources !
A noter que cela devrait également gérer les problèmes d'occultation d'exceptions qui peuvent survenir en cas d'erreurs multiples lors du traitement (une exception remontée par un bloc **try** peut être cachée par une autre dans un bloc **finally**), ce qui n'est quasiment jamais fait manuellement car le code en deviendrait alors carrément indescriptible !

Un petit côté dynamique...
Commençons par un petit rappel : les programmes **Java** sont compilés en **bytecode Java**, mais ce dernier est plus étendu que le langage du même nom. De même il est possible d'utiliser d'autres langages de programmation pour générer du bytecode, et même si dans les faits cela reste peu usité (contrairement à l'environnement **.NET**), cela prend de plus en plus d'essort en particulier avec les langages dynamiques (Python, Ruby, Perl, Javascript, Groovy, etc.).
Malheureusement, le bytecode comporte une grosse limitation pour ces langages : il ne gère pas l'invocation dynamique de méthode. Je m'explique : les différents modes d'invocations (**invokestatic**, **invokespecial**, **invokeinterface** ou **invokevirtual**) permettent uniquement d'appeler une méthode dont la signature est connue. Et même si la véritable méthode à exécuter peut varier selon le type réel de la classe, on se base toujours sur une définition de méthode existante dans le type déclaré lors de l'appel. On parle de langage fortement typé qui assure un certain nombre de vérifications dès la compilation.
A l'inverse, les langages dynamiques utilisent généralement des types dynamiques qui peuvent très bien ne pas être typés du tout à la compilation. On en revient donc à appeler une méthode totalement inconnue pour le compilateur, ce qui est impossible du point de vue du **bytecode Java**. Les implémentations de langages dynamiques se retrouvent donc à devoir jouer avec de la réflexion pour simuler ce comportement.
La JSR 292 est censée résoudre ce problème en proposant un nouveau mode d'invocation de méthode : **invokedynamic**, qui se chargerait de vérifier l'existence de la méthode seulement lors de son exécution.

Support de la JSR 292 dans le langage
A l'origine la **JSR 292** et **invokedynamic** ne devait donc impacter que le bytecode, mais pas le langage Java. Cette proposition consiste donc à intégrer cela au sein du langage, afin de faciliter un peu plus l'implémentation et l'intéraction des langages dynamiques, le tout via un nouveau package : `java.dyn`.
Note : je ne suis pas sûr de bien voir la séparation entre la **JSR 292** et la proposition du projet **Coin** tant les deux semblent avoir évolué en même temps et sont intimement liés...

MethodHandle
Tout d'abord, la classe `java.dyn.MethodHandle` permet de manipuler une référence vers le point d'entrée d'une méthode. C'est une sorte de pointeur vers une méthode qui permettra d'éviter toute la lourdeur de la réflection.
Cette classe comporte une méthode **type()** permettant d'obtenir des informations sur la méthode référencée (type de retour et nombre de paramètres), mais surtout une méthode **invoke()** un peu spéciale dans le sens où l'on peut l'utiliser avec n'importe quels paramètres, comme si la classe possédait un nombre infini de méthode **invoke()**...
Petit exemple :

```
JLabel label = new JLabel("Demo");
// On recherche la méthode d'instance 'setText()'
de la classe JLabel :
MethodHandle handle =
MethodHandles.lookup().findVirtual(JLabel.class,
"setText",
  MethodType.make(void.class, String.class)); //
type de retour + paramètres

// Les handles sur une méthode d'instance se voit
ajouter un paramètre initial,
// qui représente en fait la référence sur
laquelle on appliquera la méthode :
handle.invoke(label, "Hello"); // equivalent de
label.setText("Hello");
```

La classe **MethodHandles** propose un ensemble de méthodes utilitaires permettant de créer et manipuler les **MethodHandle**.

Cela peut sembler proche de l'API de réflexion, mais cela apporte un certain nombre de différences :

- Les vérifications d'accès sont uniquement effectuées lors de la création de l'objet **MethodHandle**, et non à chaque appel de la méthode **invoke()**.
- La JVM peut optimiser l'appel de la méthode **invoke()** en appelant directement la méthode cible comme si elle avait été appelée de manière normale.

Tout ceci implique des performances proches voire identiques à un appel de méthode standard, contrairement à la réflexion qui peut s'avérer assez "lourde"...
Le second avantage vient du fait que l'on peut manipuler la signature pointée par la **MethodHandle** pour jouer avec ses paramètres, par exemple en fixant la valeur d'un paramètre :

```
// On force la valeur du premier paramètre :
handle = handle.insertArgument(handle, 0,
label);

// Désormais notre handle est lié à une instance
précise :
handle.invoke("Hello"); // equivalent de
label.setText("Hello");
```

A noter que l'on pourrait directement faire ceci :

```
JLabel label = new JLabel("Demo");
MethodHandle handle =
MethodHandles.lookup().bind(label, "setText",
  MethodType.make(void.class, String.class)); //
type de retour + paramètres

handle.invoke("Hello"); // equivalent de
label.setText("Hello");
```

Ce mécanisme se retrouvera à la base de l'invocation dynamique.

InvokeDynamic
Le package **java.dyn** comporte une seconde classe tout aussi particulière : `InvokeDynamic`. Cette classe (final) ne comporte aucune méthode et peut donc sembler complètement inutile de prime abord. Il faut en réalité l'appréhender comme un marqueur syntaxique permettant d'effectuer des appels de méthodes dynamiques. En fait on peut considérer qu'elle possèdent un nombre infini de méthodes statiques :

```
InvokeDynamic.uneMethode();
InvokeDynamic.uneAutreMethode(1);
int x =
InvokeDynamic.<int>retourneUnInt("string", 2.0);
Object o =
InvokeDynamic.<Date>retourneUneDate();
// etc. (tout ce que l'on veut en fait)
```

Tous ces appels indiqueront au compilateur qu'il doit effectuer une invocation dynamique via **invokedynamic**, c'est-à-dire qu'ils ne seront pas vérifiés par le compilateur, mais que le langage devra mettre en place un mécanisme de recherche de la méthode à effectuer à l'exécution.
Mais comment ça marche ?
En fait il faut définir une méthode "bootstrap" qui sera chargée d'associer un **MethodHandle** à la signature de méthode d'une invocation dynamique. Lors de la première exécution d'un appel dynamique, la **JVM** passera toutes les informations utiles sur l'appel de méthode dynamique à la méthode "bootstrap". Cette dernière devra alors déterminer en effet de retourner une instance de la classe **callSite** qui fournira toutes ces informations, dont un **MethodHandle** qui sera alors exécuté par la JVM.
Par contre, l'appel au "bootstrap" ne sera plus fait pour les appels suivants sur la même définition (même nom, type de retour et paramètres). La JVM réutilisera alors les informations obtenues lors de l'appel précédent...
Petit exemple :

```
public class Main {

  // On enregistre la méthode de bootstrap pour
cette classe :
  static
{ Linkage.registerBootstrapMethod("bootstrap");
}

  private static void show(String methodName) {
    System.out.println("Hello " + methodName);
  }


  private static CallSite bootstrap(Class caller,
String name, MethodType type) {
    // On crée l'instance CallSite
    // (ici on force une méthode sans paramètre
et un retour de type void) :
    CallSite site = new CallSite(caller, name,
MethodType.make(void.class));

    // Création d'un MethodHandle (en fonction
des infos recus) :
    MethodHandle handle =
MethodHandles.lookup().findStatic(Main.class,
"show",
        MethodType.make(void.class,
String.class));
    // On force la valeur de l'argument (en
forcant le nom de la méthode) :
    handle = MethodHandles.insertArgument(handle,
0, name);

    // On défini la cible à exécuter :
    site.setTarget(handle);
    return site;
  }

  public static void main(String...args) {
    // Premier appel de 'method()' : exécute le
bootstrap
    InvokeDynamic.method();    // affiche : Hello
method
    InvokeDynamic.method();    // affiche : Hello
method
    // Premier appel de 'anotherMethod()' :
exécute le bootstrap
    InvokeDynamic.anotherMethod();  // affiche :
Hello anotherMethod
    InvokeDynamic.method();         // affiche :
Hello method
    InvokeDynamic.anotherMethod();  // affiche :
Hello anotherMethod
  }
}
```

Lors du premier appel dynamique sur **method()**, la méthode **bootstrap()** va être exécutée pour générer les informations utiles à la JVM, qui appelera alors la méthode **show()** avec un paramètre "**"method"**. Tous les autres appels sur la même méthode seront directement effectués sans réévaluer la méthode **bootstrap()**.
Bien sûr ici l'exemple est basique (et ne fonctionne qu'avec des méthodes sans paramètres), mais l'intérêt étant de déterminer dynamiquement une méthode à appeler par divers moyens...
Note : ceci peut déjà être testé de manière expérimentale avec les derniers builds du JDK7. Il faut toutefois utiliser les options **-XX:+EnableMethodHandles** *et* **-XX:+EnableInvokeDynamic** *respectivement pour activer le support des MethodHandles et de l'invocation dynamique au sein de la JVM. Ceci ne devrait plus être le cas lors de la version finale...*

cast vers InvokeDynamic
La classe **InvokeDynamic** cache un autre secret : il s'agira d'un type un peu particulier dont les références pourront accepter tout type d'objets, à l'instar de la classe **Object**. En clair il sera possible d'affecter n'importe quel objet à une variable de type **InvokeDynamic** :

```
  InvokeDynamic dynString = "I'm dynamic !";
  InvokeDynamic dynDate = new Date();
  InvokeDynamic dynLabel = new JLabel("I'm dynamic !");
```

L'intérêt vient du fait qu'on peut alors appeler une méthode de manière dynamique en la manipulant comme une référence :

```
  dynString.uneMethode();
  // est équivalent de :
  InvokeDynamic.uneMethode(dynString);
```

Les mêmes règles d'exécution seront ensuite respectées (appel du bootstrap pour déterminer la méthode à exécuter, etc.).
Note : A cette date, ceci ne semble pas être implémenté dans les derniers builds du JDK7.

Identifiant "exotique"
Comme je l'ai déjà dit, le bytecode Java est moins restrictif que le langage du même nom. En particulier il est possible d'utiliser une très large gamme de caractères dans tous les identifiants (nom de classes, de méthodes, d'attributs ou de variables). Ainsi un langage qui se compilerait en bytecode pourrait très bien utiliser des caractères "exotiques" pour le langage Java, ce qui rendrait alors ces éléments inaccessibles depuis un programme **Java**...
Pour pallier à cela on devrait avoir la possibilité de définir des identifiants "exotiques", en utilisant le même format que pour les chaînes de caractères, mais précédé d'un dièse (#).

```
  int uneVariableStandard = 5;
  int #"une Variable Exotique" = 10; // Le nom de la variable contient des espaces !

  System.out.println( uneVariableStandard * #"une Variable Exotique"); // affiche 50
```

Cela peut également permettre d'utiliser des noms qui sont normalement réservés en **Java** :

```
  int #"int" = 0; // La variable se nomme 'int' !
```

Bien sûr il est fortement déconseillé d'utiliser cela dans un programme en **Java** pur !
L'intérêt étant de permettre d'utiliser des éléments compilés en bytecode depuis un autre langage.
Par exemple Ruby utilise des caractères spéciaux dans le nom de ses méthodes (comme les méthodes de **JRuby** qui peuvent contenir des ?, = ou !), ce qui les rend inutilisables en **Java**. Les identifiants exotiques permettront de résoudre ce problème et autoriseront une interaction totale entre du code **Java** et d'autres langages, par exemple :

```
AnJRubyObject object = ...

  object.#"method?"(); // Appel de la méthode 'method?'
```

Révolution ou évolution ?
C'est sûr : on est loin d'une révolution, et même loin des modifications apportées par **Java 5.0**. Malgré l'apport de l'invocation dynamique cela reste malgré tout assez conservateur.
Personnellement, je suis très heureux de voir enfin arriver une gestion automatique des ressources (surtout en l'absence de closures qui aurait pu implémenter cela), mais je suis par contre étonné de l'absence des améliorations de la gestion des exceptions (rethrows (Lien5) et multi-catch (Lien6)) ! Cela me semblait pourtant acquis de longue date...
Pour le reste je ne suis pas d'avis particulièrement tranché
Et vous qu'en pensez-vous ? Participez au débat sur le forum ! (Lien7)

Retrouvez ce billet sur le blog d'adiGuba : Lien8

Struts 2

Ce livre sur Struts 2 s'adresse aux développeurs Java qui souhaitent disposer d'un ouvrage de référence, pour mettre en application le framework Java EE le plus répandu. L'ouvrage est décomposé en vingt-trois chapitres qui expliquent le fonctionnement du framewok, sa mise en place, chaque couche du modèle de conception MVC, l'utilisation des formulaires en passant par les plug-ins et le développement de résultats sans oublier la notion d'intercepteurs. Chaque concept est abordé de façon théorique et technique afin de permettre aux concepteurs disposant de connaissances en Java EE d'utiliser une API facilitant les développements d'applications web. Les applications utilisées dans les chapitres sont issues d'exemples concrets et sont téléchargeables sur le site de l'éditeur et sur la plate-forme de l'auteur (Lien9).

Critique du livre par Mike François

Comme le nom de l'ouvrage l'indique, Jérôme Lafosse nous livre au travers de plus de 400 pages, toutes les notions à connaître pour développer avec la nouvelle évolution du framework web Java Struts.

Dès les premières pages, nous avons un descriptif succinct des notions essentielles et de l'architecture MVC I et MVC II.
Cette attention particulière pour toute personne novice en la matière, lui permettrait de comprendre les explications données par la suite et pouvoir ainsi dès la fin du livre, développer ses propres applications web.

Les chapitres suivants détaillent chaque partie du framework allant du filtre à l'ajout et l'utilisation de plug-ins complémentaires à/ou déjà intégré dans le framework.

Enfin, Jérôme Lafosse conclut par une annexe recensant les différents intercepteurs.

Un apprentissage par l'exemple continu qui nous permet de voir l'évolution d'un code avec les différents acquis reçus au cours du chapitre, et qui démontre la valeur ajoutée de chaque partie tant sur le code que dans les exemples via les copies d'écran des résultats. Des explications claires et détaillées, qui nous permettent d'avoir deux utilisations du livre. La première serait pour l'apprentissage du langage et la seconde comme livre de référence pour le développement d'applications.
Il est très complet et couvre toutes les notions pour développer une application professionnelle en Struts 2 avec différentes librairies complémentaires comme JQuery pour l'utilisation d'Ajax, Velocity en tant que générateur de templates ...

Enfin, la police et la mise en page permettent une lecture aisée et fluide.

Les points négatifs que je pourrais apporter sont : Au niveau de l'édition, nous avons un caractère récurrent (qui est <+>) qui pourrait dérouter les plus novices.
Enfin, on aurait apprécié un surlignage des modifications ou des lignes du code qui étaient rajoutées dans les exemples suivant l'explication.

Retrouvez cette critique de livre sur la page livres Java : Lien10

Comment PHP a-t-il obtenu tant de succès ?

Rasmus Lerdorf n'est pas étranger au monde PHP vous le savez !

Le magazine américain *Linux Format* a réalisé une petite interview avec Rasmus Lerdorf dont voici les grandes lignes :

LF : Quel est votre engagement dans le développement de PHP aujourd'hui ?
Rasmus L : Ce qui est sûr c'est qu'il est moindre qu'il y a 10-15 ans, mais je lis les mailing-listes tous les jours et je discute de la suppression des bugs, de la sécurité et des questions de performances plus qu'autre chose.
Je suis aussi impliqué que je l'ai été depuis des années.

LF : Qui prend les décisions finales ?
RF : Nous. Nous possédons une mailing-liste accessible à tous, j'ai toujours insisté pour que le process soit transparent. N'importe qui peut consulter les archives de cette mailing liste et constater quelles décisions ont été prises et la position de chacun sur chacune d'entre elles.

La tendance est de donner la priorité au code, si 2 groupes se disputent une fonctionnalité et qu'un des deux a une implémentation de celle-ci et l'autre non, l'implémentation l'emporte. Aucune importance sur comment les groupes l'ont imaginée, si nous devons avoir cette fonctionnalité mais nous ne sommes pas d'accord sur l'implémentation, nous suivons ceux qui en feront une implémentation concrète.

LF : La pression est-elle le juge final des décisions ?
RL : Pour sûr, il y en a. J'essayais toujours de minimiser ce rôle, parce que je ne veux pas avoir le dernier mot dans beaucoup de ces choses. Je veux que le projet soit autosuffisant et je veux que cela s'auto-propulse. Si je suis dans la boucle comme le décideur, je prends chaque décision - qui ne pèse pas du tout.

Honnêtement, il y a une masse de choses que je ne connais pas assez. Je veux dire, prenez Sybase. Je n'ai jamais utilisé Sybase dans ma vie. Comment prendrais-je une décision intelligente contre une extension Sybase ? Pour la plupart du temps avec PHP, c'est comme cela. Il y a les groupes de personnes qui sont beaucoup mieux loties pour la prise de ces décisions.

LF : Quels langages vous ont inspirés pour le développement de PHP ?
RL : C et Perl. Ceux-là étaient les deux langues que j'utilisais à l'époque. Parce qu'à l'origine je n'essayais pas de construire un nouveau langage, j'ai simplement eu besoin d'une façon d'utilisation que je connaissais déjà dans les limites du serveur Web et de résoudre un problème. Je n'ai pas eu besoin de beaucoup de trucs qui était dans Perl et je n'ai pas voulu toute la gestion de la mémoire du C, donc j'ai eu besoin d'une version déshabillée de C, qui n'était pas tout à fait Perl.

Plus tard, C ++ et le Java étaient des langages que nous avons regardés pour comprendre ce que nous avons eu besoin de faire dans notre code orienté objet. Mais c'était d'autres personnes. Je n'ai jamais été un supporter énorme de l'OO - je l'utilise quand je pense que c'est approprié.

Je pense que la chose principale que je peux dire est : vous devez renoncer au contrôle. Si vous voulez construire un projet Open Source, vous ne pouvez pas laisser votre ego bloquer. Vous ne pouvez pas réécrire les patchs de tout le monde, vous devez donner un contrôle égal à tout un chacun.

LF : Gardez-vous un oeil sur d'autres langages comme Ruby ?
RF : Je pense que Rails arrive un peu tard, peut-être que la dernière version sera intéressante mais ils subsistent des problèmes de performances avec le scaffolding. Je n'ai jamais été un fan de la génération de code automatique.
Le scaffolding est bien mais se résume à la duplication d'applications existantes.
Le Ruby est un langage propre et agréable mais peu de monde le maîtrise.
Beaucoup de gens ont tentés Ruby à cause des screencasts qui martelaient : "Vous pouvez créer une application Web 100 fois plus vite" mais quand ils ont eu besoin de faire quelque chose de concret ils se sont dits : "Oh mon dieu, je ne connais pas assez Ruby ..."

Commentez cette news en ligne : Lien11

Les derniers tutoriels et articles

Les outils vraiment utiles pour les développeurs PHP

PHP est l'un des langages les plus largement utilisés pour créer des sites et des applications dynamiques.
Les Frameworks PHP comme Zend, CakePHP, CodeIgniter, etc et les classes et librairies PHP ont significativement simplifié nos vies.
Que vous soyez novice en PHP ou expert en développement : les outils que vous utilisez ont un impact direct sur votre productivité.
W3Avenue a compilé une liste d'outils et d'extensions vraiment utiles pour les développeurs PHP qui vont vous aider à accélérer vos développements et améliorer significativement la qualité complète de votre code.
L'article original (Lien12).

1. Les accélérateurs

- eAccelerator (Lien13)
  Accélérateur OpenSOurce PHP gratuit, optimiseur, et cache de contenu dynamique. Il augmente les performances des scripts PHP en les mettant en cache dans leur état compilé, éliminant ainsi complètement le surcoût de la compilation. Il optimise aussi les scripts pour accélérer leur exécution. eAccelerator réduit significativement la charge serveur et augmente la vitesse de votre code PHP entre 1 et 10 fois.
- IonCube PHP Accelerator (Lien14)
  L'accélérateur PHP ionCube est une extension PHP du moteur Zend installable facilement qui fournit un cache PHP, et qui est capable de délivrer une accélération substantielle des scripts PHP sans demander aucun changement de scripts, perte de contenu dynamique, ou autres compromis applicatifs.

2. Systèmes de builds

- Phing (Lien15)
  Un projet de système de build basé sur Apache Ant. Vous pouvez faire ce que vous voulez comme avec un système de build traditionnel comme GNU make, et son utilisation de fichiers de build simple en XML et de classes PHP extensibles de « taches » en fait un Framework de build facile à utiliser et extrêmement extensible. Les fonctionnalités incluent l'exécution de tests unitaires PHPUnit et SimpleTest, les transformations de fichiers, les opérations sur les fichiers systèmes, le support de build interactif, l'exécution SQL, les opérations CVS/SVN, les outils pour créer des packages PEAR, et bien plus encore.

3. Code

- BeautifyPHP (Lien16)
  BeautifyPHP est un service complètement gratuit qui permet aux visiteurs de correctement formater leur code PHP en accord avec les standards PEAR.
- PHP Beautifier (Lien17)
  Ce programme reformate et embellit les fichiers de code source PHP4 et PHP5 automatiquement. Le programme est Open Source et distribué sous les termes de la licence PHP. Il est écrit en PHP5 et possède un outil en ligne de commande.
- PHP Object Generator (POG) (Lien18)
  Un générateur de code source PHP Open Source qui génère automatiquement du code orienté objet propre et testé pour vos applications PHP4/PHP5. En générant des objets PHP avec des méthodes CRUD intégrées, POG vous donne une avance dans n'importe quel projet.
- UML2PHP5 (Lien19)
  UML2PHP5 est un plug-in destiné à se greffer sur l'application de diagramme DIA (Lien20). Il permet de générer automatiquement le squelette du code PHP des classes du diagramme. Le modèle objet de PHP5 se rapprochant de plus en plus de celui de java par exemple, il devenait urgent de fournir à la communauté un outil de design à la hauteur des outils disponibles pour d'autres langages.
- Instant SQL Formatter (Lien21)
  Instant SQL Formatter est un outil en ligne gratuit d'embellissement SQL. En plus d'embellir le code SQL, il peut traduire le code SQL en code C#, Java, PHP, DELPHI et d'autres langages de programmation. Il permet aussi de trouver tous les objets des bases de données comme les tables, les colonnes, les fonctions SQL en sélectionnant un format de sortie pour lister les objets des bases de données.
- PhpMyEdit (Lien22)
  Générateur de code PHP et éditeur de table MySQL. Les fonctions importantes fournies par phpMyEdit sont : la génération du code de manipulation des tables, ajout, modification, visualisation, copie et suppression d'enregistrements, pagination, tri et filtre des tables, recherche dans d'autres tables (relations 1:N), configuration des permissions, multiples styles de navigation possibles, contrôle du design de la sortie par CSS, logging utilisateurs, support du multilinguisme, possibilité d'étendre les classes des bases.
- PHP Obfuscator (Lien23)
  Encode et obfusque le code PHP afin de rendre le code de sortie difficile pour le « reverse

engineering ». L'application ne requiert aucune pré-modification sur votre code et pas de composants additionnels sur votre serveur. Le produit permet l'encodage des fonctions, des variables et la suppression des espaces.

- Code Eclipse (Lien24)
  Un obfuscateur PHP qui transforme le code normal et facile à lire en charabia avec très peu, voir aucune perte de vitesse et de compatibilité.

4. Base de données

- PHP Toolkit for ADO .NET Data Services (Lien25)
  Permet aux développeurs PHP d'accéder aux services de données créés en utilisant le Framework ADO.NET data services. Le but est de fournir le même service que la bibliothèque .NET pour lire et modifier les données et ses relations en utilisant des URI qui pointent sur des morceaux de données intégrées dans le web.
- Propel (Lien26)
  Propel est un Framework PHP5 d'ORM (Mapping d'objet relationnel). Il vous permet d'accéder à votre base de données en utilisant un jeu d'objets, en fournissant une API simple pour stocker et récupérer ses données.
- ADOdb (Lien27)
  Une bibliothèque orientée objet écrite en PHP qui abstrait les relations à la base de données pour la portabilité. Elle est modelée sur la bibliothèque ADO de Microsoft, mais a beaucoup d'améliorations qui la rendent unique comme les table, le support d'active records, la génération de code HTML pour la pagination de jeux de données avec les liens suivants/précédents, jeux de données en cache, la génération de menus HTML, etc.). Supporte un large nombre de bases de données incluant : MySQL, PostgreSQL, Interbase, Firebird, Informix, Oracle, MS SQL, Foxpro, Access, ADO, Sybase, FrontBase, DB2, SAP DB, SQLite, Netezza, LDAP, and generic ODBC, ODBTP.
- Doctrine (Lien28)
  Doctrine est un ORM (Mapping d'objet relationnel) pour PHP 5.2.3+ qui s'appuie sur l'une des plus hautes et puissantes couches d'abstraction de données (DBAL). Une de ses fonctions clef est de pouvoir écrire des requêtes dans un dialecte orienté objet propriétaire appelé Doctrine Query Language (DQL) inspiré du HQL Hibernate. Il fournit aux développeurs une puissante alternative au SQL qui maintient la flexibilité sans avoir recours à de la duplication de code inutile.

5. Débogage

- Xdebug, Outil de débogage et de profilage (Lien29)
  L'extension Xdebug vous aide à déboguer vos scripts en vous fournissant des informations sur vos variables. Les informations de débogage que fournit Xdebug comprennent : les traces de la pile et des fonctions de traçage dans les messages d'erreur, l'allocation mémoire et une protection contre les boucles infinies. Xdebug fournit aussi : des informations de profilage des scripts PHP, l'analyse de la couverture du code, la possibilité de déboguer interactivement vos scripts avec un client Xdebug. Vous devriez aussi regarder Webgrind (Lien30) Le profileur web PHP de Xdebug, MacGDBp (Lien31) l'application MacOS X qui débugue à distance les applications PHP contrôlées par Xdebug.
- FirePHP (Lien32)
  FirePHP est une suite idéale pour le développement AJAX où les requêtes JSON et XML propres sont requises. FirePHP vous permet de tout logger dans votre console FireBug en utilisant un simple appel à une méthode PHP. Toutes les données sont envoyées dans les headers de réponse et n'interfèreront pas avec le contenu de vos pages.
- DBG Outils PHP de débug et de profilage (Lien33)
  DBG est un débuggeur entièrement en PHP, un outil interactif qui vous aide à débuguer vos scripts. Il marche sur les serveurs de production et/ou de développement et vous permet de débuguer vos scripts localement ou à distance, depuis un éditeur ou une console.
- PHP Debug (Lien34)
  Fournit de l'assistance dans le débogage de code PHP, par les traces, affichage des variables, le chronométrage des process, les fichiers inclus, les requêtes exécutées. Ces informations sont rassemblées pendant l'exécution du script et sont affichées à la fin du script (dans une belle div flottante ou une table HTML) pour qu'elles soient lues et utilisées à n'importe quel moment.
- **Plus :** Debuglib (Lien35), Krumo (Lien36)

6. Développement

- PHP CodeSniffer (Lien37)
  PHP CodeSniffer est un script PHP5 qui segmente et "sniffe" le code PHP pour détecter les violations d'un standard de code défini. C'est un outil de développement essentiel qui assure la qualité de votre code. Il peut aussi aider à prévenir des erreurs sémantiques de code des développeurs.
- PhpDocumentor (Lien38)
  Similaire à Javadoc, écrit en PHP, phpDocumentor peut être utilisé depuis la ligne de commande ou via une interface web pour créer des documentations professionnelles à partir du code PHP. phpDocumentor contient un support pour la liaison à la documentation, incorporant des documents de niveau utilisateur comme les tutoriels la création de code source surligné avec des références croisées avec la documentation PHP.
- PHP Depend (Lien39)
  PHP Depend est un analyseur d'application et un outil de métrique qui aspire à fournir plein d'informations subtiles à propos d'un projet PHP spécifique. PHP Depend peut générer un large choix de métriques applicatives depuis une base

de code donnée, ces valeurs peuvent être utilisées pour mesurer la qualité d'un projet applicatif et aider à identifier les parties d'une application qui devraient être factorisées.

- PhpLangEditor (Lien40)
  Un plug-in Firefox qui vous permettra de traduire aisément vos fichiers et variables de langue dans vos scripts PHP.

7. IDE & Editeurs

- Aptana PHP Development Environment (Lien41)
  Aptana PHP est l'IDE robuste, gratuit, Open Source pour PHP incluant tout ce dont vous avez besoin pour démarrer rapidement et aller encore plus vite dans le développement, test, amélioration, et déploiement de vos applications PHP. Depuis les serveurs PHP pré-installés, l'auto complétion de code, les templates de code, la génération de code, le débogage, le refactoring, l'éditeur Smarty, les outils de base de données et bien plus, Aptana PHP vous donnent de bout en bout les outils dont vous avez besoin pour PHP, plus tout ce qu'Aptana Studio a à offrir.
- PHPEclipse (Lien42)
  PHPEclipse fonctionne sur toutes les plateformes majeures et possède les fonctionnalités suivantes : Coloration syntaxique, correspondance des accolades/parenthèses, repliage de code, autocomplétion du code, intégration du manuel PHP, templates de code, le support de Xdebug, de DBG et de CVS et SVN++.
- Zend Studio (Lien43)
  Un environnement de développement professionnel qui inclut l'édition de code PHP, débogage, profilage, tests unitaires, diagnostics et plus.
- PHPanywhere (Lien44)
  Un environnement de développement web gratuit pour le PHP, en d'autres mots c'est une application qui donne aux développeurs toutes les possibilités d'édition de code dont ils ont besoin pour développer en ligne. Il possède un vérificateur de syntaxe en temps réel, incluant le support de tout les formats web et un puissant client FTP.
- VS.Php For Visual Studio (Lien45)
  VS.php est un environnement de développement intégré basé sur Visual Studio 2008. Avec VS.php vous pouvez concevoir, développer, déboguer, et déployer vos applications PHP au sein de l'éditeur Visual Studio.
- Plus: NetBeans (Lien46), PhpED (Lien47), PHPEdit (Lien48), phpDesigner (Lien49), TextMate (Lien50), Komodo IDE (Lien51)

8. Sécurité

- PHP Intrusion Detection System (PHPIDS) (Lien52)
  Une couche de sécurité, simple à utiliser, bien structurée, rapide et à l'état de l'art pour les applications basées sur PHP. L'IDS ni ne déshabille, ni n'assainit, ni ne filtre une saisie malveillante, il reconnaît simplement quand un attaquant essaye de casser votre site et réagit exactement comme vous le souhaitez. Actuellement PHPIDS détecte toute les sortes de XSS, injections SQL, injections des headers, traversées de répertoires, RFE/LFI, attaques DOS et LDAP. Basé sur un jeu de filtres approuvé et lourdement évalué il juge et donne à n'importe quelle attaque une évaluation d'impact numérique qui facilite le choix de l'action à suivre après la tentative de piratage. Cela pourrait s'étendre de l'enregistrement simple à l'émission d'un mail de secours à l'équipe de développement, l'affichage d'un message d'alerte pour l'attaquant ou même la fin de la session de l'utilisateur.
- PhpSecInfo (Lien53)
  PhpSecInfo fournit un équivalent de la fonction phpinfo() qui montre les informations de sécurité à propos de l'environnement PHP, et offre des suggestions d'amélioration. Ce n'est pas un remplaçant aux techniques sécurisées de développement et ne fait aucun audit de code ou d'application, mais il peut être un excellent outil dans une approche de sécurité multi-niveaux.

9. Setup

- PHPConfig (Lien54)
  Une application graphique pour le fichier php.ini de PHP, où toutes les configurations de PHP prennent place. Le résultat final est que vous pouvez passer plus de temps sur votre code que sur la configuration du php.ini. Il supporte toutes les fonctionnalités standard de PHP, et possède un onglet spécial pour tous les plug-ins tierces.
- Lighty2Go (Lien55)
  Lighty2Go est un pack LightTPD, MySQL & PHP (LiMP) pour Windows. Prenez le avec vous sur votre clef USB et faites vous plaisir.
- PAMP (Lien56)
  Pack personnel AMP : Apache, MySQL et PHP pour mobile basé sur s60.
- WampServer (Lien57)
  Vous permet de configurer Apache, PHP et MySQL sur Windows. Il est aussi livré avec PHPMyAdmin pour gérer facilement vos bases de données. WampServer vous permet d'ajouter n'importe quelle version de Apache, PHP et MySQL.
- Server2Go (Lien58)
  Un serveur web qui marche sans installation et sur des medias protégés en écriture. Cela veut dire que les applications web basées sur Server2Go peuvent être utilisées directement depuis un CD, une clef USB, ou n'importe quel répertoire sur le disque sans se donner la peine de configurer Apache, PHP et MySQL.

10. Tests

- PHPUnit (Lien59)
  PHPUnit est un membre de la famille xUnit des Frameworks de test et fournit aussi bien un Framework qui permet l'écriture des tests facilement et l'exécution des fonctions de test que l'analyse de leurs résultats.

- SimpleTest (Lien60)
  SimpleTest est similaire à JUnit/PHPUnit. Il supporte les faux objets et peut être utilisé pour automatiser les tests de régression d'une application web avec un client http scriptable qui peut parser les pages HTML et simuler des événements comme le clic sur des liens ou les soumissions de formulaires.
- VfsStream (Lien61)
  vfsStream est un wrapper de flux pour un système de fichiers virtuels qui peut s'avérer utile dans les tests unitaires pour simuler le véritable système de fichiers. Il peut être utilisé avec n'importe quel Framework de tests comme PHPUnit ou SimpleTest.

11. Intégration continue

- phpUnderControl (Lien62)
  phpUnderControl est un add-on pour l'intégration continue dans CruiseControl, qui intègre quelques-uns des meilleurs outils de développement PHP. Ce projet aspire à faire vos premiers pas avec CruiseControl et PHP aussi facilement que possible. De plus phpUnderControl vient avec un outil en ligne de commande qui produit toutes les modifications à une installation CruiseControl existante.

12. Pense-bête

- PHP Cheat Sheet de AddedBytes (Lien63)
  Ce pense-bête PHP est une page de référence, listant les arguments de formatage de date, les expressions régulières, et toutes les fonctions communes.
- PHP 5 Online Cheat Sheet (Lien64)
  Couvre tous les types dans PHP ainsi que les méthodes de conversion, déclaration, manipulation, etc.
- The CheatSheet – CakePHP 1.2 (Lien65)
  Inclus une référence rapide sur les variables de configuration de Cake, les fonctions globales, les conventions, les chemins et le fichier index.php. Il inclut aussi des références aux propriétés, méthodes et fonctions de callback pour les modèles, les vues, les contrôleurs et les helpers.
- PHP $_SERVER Superglobal pour Apache & IIS (Lien66)
  Une table des clefs définies dans les superglobales $_SERVER de PHP qui tourne sur les serveurs Apache et IIS. Le but de cette table est de permettre aux développeurs une plongée dans ce à quoi ils doivent s'attendre si ils doivent migrer d'une plateforme à une autre.
- Smarty Cheat Sheet (Lien67)
  Tous ceux qui sont encore intéressés par Smarty peuvent consulter ce pense-bête. Il contient quelques astuces et des références pour les designers de templates Smarty.

13. Liens

Lisez aussi "Les classes et librairies vraiment utiles pour les développeurs PHP" (Lien68).
Vous pouvez aussi aller voir mes autres traductions (Lien69).

Retrouvez l'article de Saud Khan traduit par Joris Crozier en ligne : Lien70

Les classes et librairies vraiment utiles pour les développeurs PHP

Aujourd'hui des millions de sites et de serveurs web à travers le web utilisent PHP. Créé à l'origine par Rasmus Lerdorf (Lien71) en 1995 pour que tout le monde puisse créer facilement une page web personnelle (Personal Home Page), PHP a fait du chemin et est maintenant largement utilisé comme le langage approprié pour la majorité des projets de développement Web.
De nombreux Frameworks PHP (Lien72) ont vus le jour pour permettre un développement rapide avec PHP. Tandis qu'il existe un grand nombre de classes et de librairies PHP aussi importantes qui permettent d'en tirer les mêmes bénéfices. W3Avenue à compilé une liste de quelques classes et librairies vraiment utiles avec lesquelles tous les développeurs PHP devraient être familiarisés. Que vous préfériez utiliser un Framework PHP ou non, votre productivité devrait être accrue avec l'aide de ses classes et librairies.
L'article original (Lien73).

1. Base de données

- ADOdb (Lien74)
  Une librairie orientée objet écrite en PHP qui abstrait les relations à la base de données pour la portabilité. Elle est modelée sur la librairie ADO de Microsoft, mais a beaucoup d'améliorations qui la rendent unique comme les table pivots, le support d'active records, la génération de code HTML pour la pagination de jeux de données avec les liens suivants/précédents, jeux de données en cache, la génération de menus HTML, etc.). Supporte un large nombre de base de données incluant : MySQL, PostgreSQL, Interbase, Firebird, Informix, Oracle, MS SQL, Foxpro, Access, ADO, Sybase, FrontBase, DB2, SAP DB, SQLite, Netezza, LDAP, and generic ODBC, ODBTP.
- Doctrine (Lien75)
  Doctrine est un ORM (Mapping d'objet relationnel) pour PHP 5.2.3+ qui s'appuie sur l'une des plus hautes et puissantes couches

d'abstraction de données (DBAL). Une de ses fonctions clef est de pouvoir écrire des requêtes dans un dialecte orienté objet propriétaire appelé Doctrine Query Language (DQL) inspiré du HQL Hibernate. Il fournit aux développeurs une puissante alternative au SQL qui maintient la flexibilité sans avoir recours à de la duplication de code inutile.

- PHPLINQ (Lien76)
  Un jeu de classes PHP imitant les méthodes de l'extension LINQ de c#3 (Language Integrated Query). Les fonctions de PHPLinq : les opérateurs LINQ (select, take, skip, orderBy / orderByDescending, thenBy / thenByDescending), les expressions Lambda et les types anonymes.
- Mimesis (Lien77)
  Mimesis est une API bas niveau de base données par fichiers plats écrite en PHP et Open Source créée pour agir comme backend pour les scripts côté serveur qui demandent les fonctionnalités d'une base de données. Au lieu de parser des états SQL, Mimesis utilise des constructions orientées objet de PHP pour fournir une classe distincte avec différentes méthodes de manipulation de base de données.

2. Développement

- PHP CodeSniffer (Lien78)
  PHP CodeSniffer est un script PHP5 qui segmente et «sniff» le code PHP pour détecter les violations d'un standard de code défini. C'est un outil de développement essentiel qui assure la qualité de votre code. Il peut aussi aider à prévenir des erreurs sémantiques de code des développeurs.
- PhpDocumentor (Lien79)
  Similaire à Javadoc, écrit en PHP, phpDocumentor peut être utilisé depuis la ligne de commande ou via une interface web pour créer des documentations professionnelles à partir du code PHP. phpDocumentor contient un support pour la liaison à la documentation, incorporant des documents de niveau utilisateur comme les tutoriels la création de code source surligné avec des références croisées avec la documentation PHP.

3. Document

- TCPDF (Lien80)
  Une classe PHP Open Source pour générer des documents PDF. Ne requiert pas de librairie externe pour les fonctions basiques; supporte tous les formats de page ISO y compris UTF-8, Unicode, les langages RTL et HTML.
- PHPPowerPoint (Lien81)
  Basée sur le standard Microsoft OpenXML, la classe PHPPowerPoint vous permet de lire et écrire des documents PowerPoint. Les fonctions incluses : gérer les méta données de présentation (auteur, titre, description,.) , ajouter des slides, ajouter des images à votre présentation et plus encore.
- PHPExcel (Lien82)
  Basée sur le standard Microsoft OpenXML, la classe PHPExcel vous permet de lire et écrire des documents Excel. Les fonctions incluses sont : gérer les méta-données (auteur, titre, description, ...), feuilles multiples, différentes police et styles de police, les bordure de cellules, les remplissages, les gradients, ajouter des images à vos feuilles et beaucoup, beaucoup plus encore.
- PhpRtf Lite (Lien83)
  Le but de cette librairie est de créer des documents RTF avec PHP qui sont compatibles avec Microsoft Word et Open Office Writer. Les fonctions principales incluses : formatage des sections de documents : marges, taille du papier, les bordures et autres (les documents peuvent comporter plusieurs sections), le formatage des en-têtes et des pieds de page, le contrôle des paragraphes : arrières-plans, bordures, alignement et autres, le contrôle de la police : gras, italique, face, taille, couleur (il est possible d'utiliser le style de balise HTML), les images embarquées, (formats JPG et PNG), le formatage des tableaux : taille, bordures, arrières-plans, et l'alignement des cellules, le support UTF-8.
- PclZip (Lien84)
  Offre des fonctions de compression et d'extraction pour les archives au format ZIP (Winzip, PKZip). Elle vous donne la possibilité de créer des archives, de lister leur contenu et de les extraire dans le système de fichiers. PclZip définit une classe objet représentant une archive Zip. Cette classe manage les propriétés de l'archive et offre des méthodes d'accès et de manipulation de l'archive.

Retrouvez la suite de l'article de Saud Khan traduit par Joris Crozier en ligne : Lien85

Passez à l'UTF-8 sans manquer une étape

N'avez jamais vous pesté contre des caractères s'affichant mal, carrés, points d'interrogation ou caractères étranges à la place des accents ? Et ceci dès que vous essayiez d'utiliser un encodage en UTF-8 ?

Ce tutoriel va vous expliquer comment encoder votre site intégralement en UTF-8 sans louper une étape qui pourrait faire apparaître ces caractères disgracieux.

1. Introduction

Cet article est basé sur les technologies Apache, PHP et MySQL, donc aucun des codes suivants ne fonctionne sur un autre type d'environnement.

Selon votre environnement, le navigateur va utiliser différentes méthodes pour choisir quel encodage utiliser pour parser et afficher le document demandé. Dans le cas d'un fichier statique local, sans serveur (donc sans utiliser même en local des logiciels comme WAMP ou EasyPHP), le navigateur va utiliser la balise <meta> décrite plus loin, alors que dans le cas d'un serveur, le navigateur va se référer à l'en-tête renvoyé par celui-ci.

2. Au niveau du document HTML

L'encodage au niveau d'un document HTML se définit grâce à une balise <meta> :

Encodage du document
```
<meta http-equiv="content-type"
content="text/html; charset=utf-8" />
```

Cette balise doit être insérée dans l'élément head de votre document HTML et doit évidemment être unique.

3. Au niveau du fichier

Chaque fichier doit impérativement être enregistré en UTF-8 sans BOM (Byte Order Mark (Lien86)).

Chaque éditeur fonctionne de manière différente pour permettre l'enregistrement des fichiers en UTF-8. Voici quelques exemples de manipulation sur certains éditeurs :

- **Notepad++** : Aller dans Format > Encoder en UTF-8 (sans BOM).
- **Dreamweaver** : Aller dans Modifier > Propriétés de la page > Titre/Codage.
- **Aptana** : Aller dans Edit > Set Encoding.
- **Bloc notes Windows** : Aller dans Fichier > Enregistrer sous... > Sélectionner UTF-8 dans la liste Codage.

N'hésitez pas à me faire part des manipulations sur d'autres logiciels pour compléter la liste.

Il ne faut pas confondre document HTML et fichier. A savoir qu'un document HTML peut être formé de plusieurs fichiers, dans le cas d'include en PHP.

Donc l'ensemble des fichiers qui forment le document HTML doivent être enregistrés en UTF-8 sans BOM.

4. Au niveau du serveur

4.1. Méthode PHP

Il existe deux méthodes en PHP permettant d'afficher du texte en UTF-8. Après avoir bien sûr encodé correctement tous les fichiers selon la manière décrite juste au-dessus. L'une est radicale au niveau du fichier, l'autre se fait au cas par cas, sur chaque texte à afficher.

La méthode radicale consiste à mettre en première ligne de chaque fichier, un header qui va préciser au serveur de renvoyer de l'UTF-8 :

Header
```
header( 'content-type: text/html; charset=utf-
8' );
```

L'autre méthode consiste à utiliser une fonction PHP autour du texte que l'on veut afficher en UTF-8 :

Fonction utf8_decode()
```
echo utf8_decode( 'Ici mon texte en UTF-8' );
```

La différence entre les deux méthodes est flagrante. La seconde est d'une part plus contraignante parce qu'il faut l'utiliser sur chaque texte et d'autre part ne précise pas au serveur de renvoyer de l'UTF-8.

Faisons un petit test pour nous en convaincre. Nous allons créer deux fichiers distincts, un nommé test1.php et l'autre nommé test2.php. Ces deux fichiers seront évidemment encodés correctement en UTF-8 (cf III).
Dans test1.php nous mettons ce code :

test1.php
```
header( 'content-type: text/html; charset=utf-
8' );

echo 'Texte accentué.';
```

Dans test2.php nous mettons ce code :

test2.php
```
echo utf8_decode( 'Texte accentué.' );
```

Ouvrons-les dans le navigateur. Regardons l'encodage des pages dans Affichage > Encodage des caractères. Nous pouvons constater que test1.php est encodé en Unicode (UTF-8) alors que test2.php est encodé en Occidental (ISO-8859-1).

Mais pourquoi utf8_DEcode alors que l'on veut ENcoder en UTF-8 ? Ne serait-ce pas plutôt utf8_encode() qu'il faut utiliser ?
La réponse est non. En effet, notre fichier est **déjà** encodé en UTF-8 (cf III toujours). Donc les caractères accentués écrits dans ce fichier sont encodés en UTF-8. Le problème ? Vu que le serveur, lui, renvoie de l'ISO-8859-1 il faut donc décoder ces caractères accentués encodés en UTF-8 vers de l'ISO-8859-1.

Mais alors à quoi sert la fonction utf8_encode() ? A encoder des caractères en UTF-8 évidemment ! Seulement à encoder des caractères qui sont en ISO-8859-1, c'est-à-dire que le fichier est enregistré (cf III encore et toujours) non pas en UTF-8 (sans BOM) mais en ANSI et que le serveur renvoie de l'UTF-8 (ben oui, sinon aucun intérêt à encoder ces caractères en UTF-8).
Qui a dit que l'encodage était une partie de plaisir ?

4.2. Méthode Apache

Il existe aussi deux façons de préciser le charset au niveau Apache. Les deux méthodes reposent sur le même code qui est :

Encodage Apache
```
AddDefaultCharset utf-8
```

On peut mettre ce code à deux endroits différents selon que l'on travaille sur un serveur dédié sur lequel on a la main ou sur un serveur mutualisé.
Dans le cas d'un serveur dédié, on peut rajouter ce code dans le fichier de configuration Apache httpd.conf.
Dans le cas d'un serveur mutualisé, ce code peut être rajouté dans un fichier .htaccess placé à la racine du domaine. Et si par malchance votre serveur mutualisé interdit l'usage de .htaccess, il vous reste tout de même la méthode PHP.

5. Au niveau de la base de données

5.1. Aux champs

Avant de penser à ajouter la moindre information en base de données, il faut d'abord songer à modifier l'interclassement de tous les champs qui contiennent des données textes.
Pour cela, si vous avez phpMyAdmin, rien de plus simple, vous n'avez qu'à sélectionner un interclassement UTF-8 dans la liste déroulante de chaque champ au niveau de la structure des tables. Dans le cas contraire, il vous suffit d'exécuter une requête qui va bien pour l'ensemble de vos champs contenant des données texte :

Interclassement au niveau d'un champ
```
ALTER TABLE `ma_table` CHANGE `mon_champ`
`mon_champ` VARCHAR( 50 ) CHARACTER SET utf8
COLLATE utf8_unicode_ci NOT NULL;
```

Mieux encore, vous n'avez pas encore créé votre base de données et vous pouvez vous éviter cette manipulation en mettant un interclassement par défaut lors de sa création.
Sur phpMyAdmin, au moment de la créer, il suffit de sélectionner cet interclassement dans la liste déroulante. Sinon vous pouvez créer votre base de données de cette manière :

Interclassement au niveau de la base de données entière
```
CREATE DATABASE `ma_base` DEFAULT CHARACTER SET
utf8 COLLATE utf8_unicode_ci;
```

Devoir mettre de l'UTF-8 oui, mais lequel ? Que choisir entre utf8_unicode_ci et utf8_general_ci ?
Les différences entre ces deux interclassements sont minimes. Vous devez juste savoir que utf8_general_ci est plus rapide que utf8_unicode_ci. Mais en contrepartie, utf8_unicode_ci est plus précis que utf8_general_ci. Par exemple, lors d'une recherche, le caractère spécial allemand "**ß**" équivaut à "**s**" en utf8_general_ci alors qu'il vaut bien "**ss**" en utf8_unicode_ci.
En conclusion, si votre application ne gère pas le multilingue ou n'a pas besoin d'un niveau aussi précis de comparaison, il vaut mieux utiliser utf8_general_ci pour sa rapidité.

5.2. A la connexion

Avoir tous les champs correctement encodés, c'est bien beau, seulement il reste à définir en quel encodage les différentes couches de notre application vont communiquer. Ca ne se fait évidemment pas automatiquement. Maintenant que l'on a notre serveur en UTF-8 et donc nos scripts PHP, que l'on a également nos champs en base de données, il va falloir que ces deux technologies communiquent entre elles en UTF-8.
Pour cela, rien de plus simple. Juste après la connexion à la base de données et la sélection d'une base en PHP, il faut appeler la fonction mysql_set_charset() en lui précisant l'encodage :

Encodage à la connexion en PHP
```
mysql_set_charset( 'utf8' );
```

Seulement cette fonction n'est disponible que pour les versions 5 et supérieures de PHP. Donc pour les versions inférieures, l'appel de la fonction est à remplacer par l'exécution de cette requête :

Encodage à la connexion en SQL
```
SET NAMES "utf8";
```

Retrouvez l'article de Josselin Willette en ligne : Lien87

Adobe Illustrator CS4 : le dessin vectoriel par excellence

Leader dans le monde du design et du graphisme, la société Adobe a su se faire une place parmi le monde professionnel, grâce à la « créative suite » qui regroupe un certain nombre d'outils comme Illustrator, Photoshop, InDesign, etc. Chaque logiciel a son domaine de prédilection, on se retrouve donc avec Photoshop pour la retouche d'image, InDesign pour la publication, Illustrator pour l'illustration, etc.

Mais penchons-nous sur le logiciel Illustrator, et plus particulièrement sur la version présente dans la suite "Creative Suite 4 - CS4".

1. Les suites créatives proposées par la société ADOBE

Leader mondial dans le monde professionnel du web et du design, **la société ADOBE** nous propose différents types d'outils extrêmement puissants allant de l'édition d'images à la création d'animations sur des pages web. Tous ces outils sont réunis dans une série de collections ou « *Suite* » appelées **Creative Suite** qui ont l'avantage de proposer un set complet d'outils pour la création de vos projets. Voyons les différentes suites proposées par la société **ADOBE** :

- **Design Premium :** la suite Design Premium intègre tous les outils nécessaires pour le dessin, la création internet et applications de publication pour lesquelles on retrouve les outils suivants : Indesign CS4, Photoshop CS4 Extended, Illustrator CS4, Flash CS4 Professional, Dreamweaver CS4, Fireworks CS4 et Acrobat 9 Pro.
- **Web Premium :** la suite complète spécialement dédiée à la création internet pour laquelle on retrouve les outils suivants : Dreamweaver, Flash CS4 Professional, Photoshop CS4 Extended, Illustrator CS4, Fireworks CS4, Acrobat 9 Pro, Soundbooth CS4 et Contribute CS4 .
- **Production Premium :** la suite Production premium concerne uniquement le travail, le design et la composition de vidéo. On retrouve les outils suivant : After Effects CS4, Premiere Pro CS4, Flash CS4 Professional, Illustrator CS4, Soundbooth CS4, OnLocation CS4 et Encore CS4.
- **Master collection :** elle regroupe tous les outils disponibles dans les autres suites/collections.

Pour plus de détails sur ces fameuses suites : Lien88

Bien sûr, l'utilisateur peut prendre un outil à part sans prendre une suite complète. Mais le grand avantage de ces suites est surtout d'économiser. En effet, cela revient moins cher de prendre une suite que de prendre chaque application une par une.

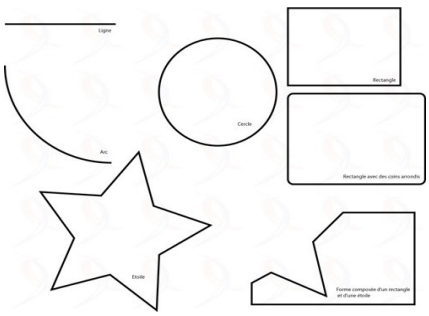
Pour la suite nous allons nous pencher sur le sujet original de cet article, à savoir la présentation de l'outil extrêmement puissant d'illustration, qui se nomme **illustrator CS4** - CS4 pour Creative Suite 4.

2. Le dessin vectoriel

2.1. Définition d'un dessin vectoriel

Répandu dans le monde professionnel et particulièrement dans le monde du graphisme, le dessin vectoriel est souvent utilisé pour améliorer le visuel de divers projets, dont les créations pour des sites web et pour de la publication (nous verrons au fil des articles sur **Illustrator CS4** des exemples pratiques).

Mais qu'est-ce que c'est exactement, le dessin vectoriel ? Un dessin vectoriel est constitué à la base de formes simples comme des points, des lignes, des arcs, des cercles, des rectangles, etc. Ces bases peuvent subir des transformations simples pour former des formes plus complexes comme le montre l'image ci-dessous.



Bien sûr, ce n'est qu'une infime partie des formes possibles que propose Illustrator, nous les verrons par la suite plus en détail.

À partir de là, vous allez sûrement me dire, mais quelles différences y a-t-il entre un dessin vectoriel et un dessin matriciel ? C'est très simple, une forme vectorielle est constituée de coordonnées très précises. Prenons un exemple simple, un rectangle. Imaginez-vous devant un rectangle, et que voyez-vous ? Il y a 2 choses qui nous sautent aux yeux : les traits – 4 pour un rectangle et 4 points à chaque extrémité des traits.

L'image ci-dessus montre une représentation d'un rectangle vue sous Illustrator. Bien sûr, les points ne sont pas aussi grands, mais c'est juste pour vous montrer à quoi

servent ces points.
Les points que l'on nomme **points d'ancrage**, comme pour les traits, sont calculés à partir de certaines données comme la position, la taille et/ou la rotation dans l'axe de travail, cela signifie que chaque point, chaque trait peut être modifié à volonté après la mise en place de la forme. En gros si vous voulez simplement modifier l'emplacement du point en bas à gauche de votre rectangle, il suffit de prendre l'outil de sélection directe et de glisser le point vers l'endroit voulu, bien sûr on va y revenir dessus plus tard dans l'article.

2.2. Avantage du dessin vectoriel par rapport au dessin matriciel

On a vu ensemble la définition d'un **dessin vectoriel** de base. Voyons maintenant les avantages qu'offrent le vectoriel par rapport au matriciel. Le **dessin matriciel** est une image formée de pixels, par exemple une image de 800x600 sera formée de 800 pixels de largeur et 600 pixels en hauteur. Ici, pas question de lignes, de points d'ancrage ou des formes (malgré cela certains logiciels comme **Photoshop** intègrent une partie vectorielle dans un plan de travail matriciel), une fois dessiné sur le plan de travail, il ne sera plus possible de le modifier sans l'effacer manuellement avec la gomme ou faire un **ctrl-Z** pour annuler la dernière commande.

À part l'avantage de la réédition d'objet (on appelle objet toute forme déposée sur le plan de travail) on peut aussi voir la différence avec le redimensionnement de l'image complète. Prenons un exemple : visualisez une image vectorielle et matricielle de 100x100 et de très bonne qualité chacune. Maintenant, redimensionnez-les à 600x600 et regardez le résultat – comme le montre l'image ci-dessous :

Dans une redimension en mode vectoriel, il y a généralement très peu de pertes, tandis qu'avec le dessin matriciel :

Comme vous pouvez le voir le résultat n'est pas vraiment ce que l'on pouvait espérer. On perd beaucoup au niveau de la résolution, l'image est floue et totalement pixélisée. Autant vous dire que ce type d'image est totalement hors course pour la présentation sur un environnement professionnel.

Bien sûr, nous n'allons pas utiliser du vectoriel pour de la retouche d'image. Chaque logiciel de la gamme **ADOBE** se distingue dans sa spécialité.

À noter que le dessin vectoriel perd également de la qualité, mais vraiment infime par rapport au matriciel, on parle là de très très haute résolution.

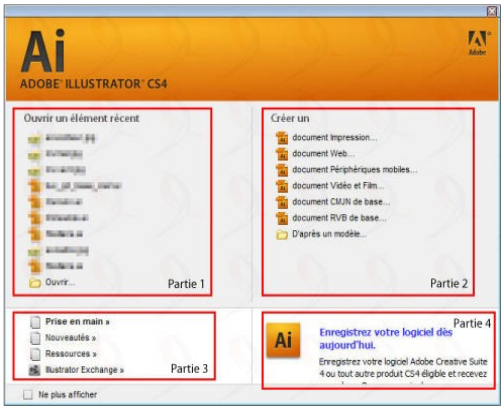
2.3. Domaine d'utilisation

Le domaine d'utilisation du dessin vectoriel est très vaste, on peut le retrouver dans le domaine de l'internet par exemple pour l'illustration d'un site web, pour la publication d'un journal ou de prospectus distribués au format papier ou au format électronique, pour la création de schémas techniques, etc.

3. Présentation du logiciel Illustrator CS4

3.1. Premier démarrage

Bon, il est grand temps de lancer l'application et de voir sa présentation en détail avec les explications.



Au lancement d'**Illustrator**, une fenêtre d'accueil se propose par défaut à nous, bien sûr il vous est possible de cocher "*Ne plus afficher*" si vous ne voulez plus que cet écran apparaisse au démarrage. Cependant, je vous conseille de le garder au démarrage pour une simple raison de facilité d'utilisation. Détaillons les 4 parties visibles.

- **Partie 1 - Ouvrir un élément récent :** comme le titre l'indique, dans cette partie vous verrez les derniers travaux effectués sur Illustrator CS4. Il y a tous les types de fichier, pas uniquement AI (Format de base pour Illustrator). Si jamais vous ne trouvez pas votre fichier, ce n'est pas grave, il vous suffira de cliquer simplement sur "Ouvrir..." en bas de la liste pour ouvrir explicitement votre fichier.
- **Partie 2 - Créer un :** cette partie est la partie la plus importante de cette fenêtre, car c'est elle qui va guider l'utilisateur vers son plan de travail adapté à son travail. Dans ce menu nous avons, *document Impression* qui sera plus adapté pour un rendu sur papier, *document Web* qui sera spécialement pour le Web, *Applicatif* concerne

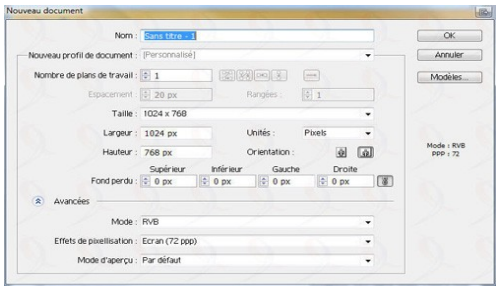
tout ce qui doit être sur l'écran, *document Périphériques mobiles* comme son nom l'indique pour la création de documents adaptés aux mobiles, *document Vidéo et Film* plus porté sur l'animation en tout genre et pour finir 2 choix sur les modes de couleurs de base.

- **Partie 3 :** cette partie vous donne de l'aide et vous accueille en présentant les nouveautés du logiciel.
- **Partie 4 :** cette partie vous montre des astuces actualisées.

Attention, Illustrator met en avant deux modes de couleur, le **RVB** et la **CMJN**. Ce n'est pas la même chose et nous le verrons dans la suite de l'article. Pour résumer on utilise **RVB** pour tous les travaux dont l'objectif final est un affichage à l'écran et **CMJN** pour tous les travaux dont l'objectif final est une impression sur papier.

3.2. Premier document Web

Dans la suite nous allons nous baser sur un travail de type Web, donc le choix "**Document Web**" est très bien pour commencer.



La fenêtre demande quelques précisions sur le plan de travail, et c'est à l'utilisateur de confirmer l'ensemble.

Pour commencer, nous pouvons voir dans un premier temps une entrée **Nom**. Là, rien de bien compliqué, il s'agit simplement du nom de votre travail. Il est important de donner dès le début un nom, car cela peut arriver qu'un jour vous arriviez à travailler avec plusieurs projets plus ou moins en même temps. Donner simplement un nom qui vous permettra de vous y retrouver facilement.

L'entrée suivante est **le profil de document**. Ce profil résume la plupart des options que l'on a vues sur l'écran de démarrage à savoir *Document Web, Document Impression, etc*. Sur l'image vous voyez "*personnalisé*", cela se met automatiquement dès que vous changez une option sur cette fenêtre.

Le nombre de plans de travail est une grande nouveauté de CS4, cela permet de définir un certain nombre de plans et de pouvoir les afficher selon la configuration indiquée sur les boutons à droite, l'espacement et le nombre de rangées. Le fonctionnement des plans est assez simple à comprendre, un plan est à la base un espace de travail supplémentaire, cela vous permet par exemple de créer un objet sous divers angles, ou avec une autre texture. Par défaut, on va laisser le nombre de plans de travail à 1.

La taille est la taille du plan de travail, par défaut la fenêtre nous propose 4 options : 640x480, 800x600, 1024x768 et Personnalisé. À noter, comme **les profils**, **Illustrator CS4** met automatiquement sur Personnalisé si on met une valeur dans les cases **Largeur** et **Hauteur** juste en dessous.

Les unités sont des unités de mesure pour le dessin. Pour le web, on prendra généralement l'unité pixel ou point et pour de l'impression les unités cm, mm, ou pied.

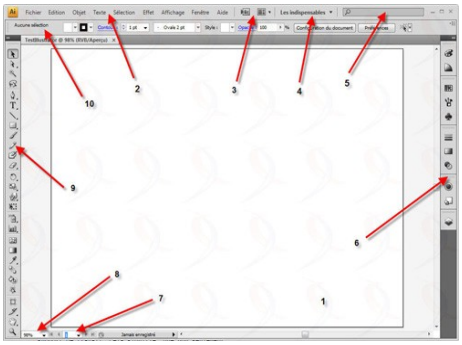
L'orientation est juste la position du plan du travail, vous pouvez choisir entre paysage (vu en largeur) ou portrait (vu en hauteur).

Le fond perdu est une zone qui sert de dépassement. Cette zone est surtout adaptée pour des travaux de type papier.

Le mode avancé propose quant à lui des options supplémentaires pour le plan de travail. Parmi ces options, on retrouve **le mode de couleur utilisé** : RVB ou CMJN, **l'effet de pixellisation** qui permet de spécifier la qualité de l'image pendant la création et **le mode d'aperçu** qui permet de choisir différentes vues de travail.

3.3. Le plan de travail

Le plan de travail est l'espace qui vous servira comme support de travail. Cela signifie, que non seulement vous disposez d'un espace de dessin, généralement sous forme d'une feuille à l'écran, mais aussi une panoplie d'outils et de fenêtres adaptés à la situation. Voyons en détails...



La partie 1 représente la partie la plus importante de l'interface. C'est ici que vous passerez le plus de temps pour la réalisation de vos dessins. Cette partie n'est autre que l'espace de dessin où vous disposez vos formes, vos calques ou vos modèles.

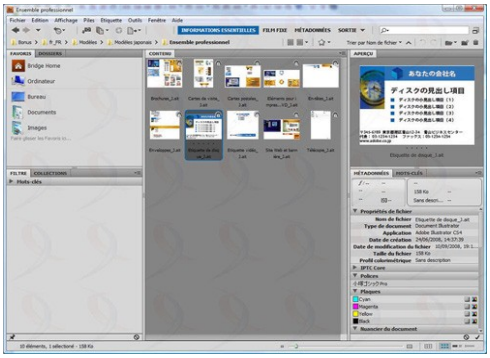
La partie 2 correspond aux menus d'Illustrator qui résument les différentes actions possibles dans l'interface, nous retrouvons bien sûr les commandes standards comme ouvrir ou enregistrer votre travail, mais aussi des commandes plus adaptées à **Illustrator**. Voyons en détail chaque menu :

- **Le menu fichier :** ce menu gère principalement tout ce qui est fichier et projet. Par exemple, vous pourrez enregistrer, importer ou exporter, imprimer ou scripter simplement votre projet.
- **Le menu Edition :** ce menu permet des opérations assez communes à d'autres applications, à savoir des commandes de copier/coller, de recherches de textes ou de la correction orthographique. À noter que c'est dans ce menu que vous trouvez les options pour

configurer le logiciel **Illustrator**.

- **Le menu Objet :** ce menu dispose des commandes spécialement dédiées aux objets (pour rappel, sous Illustrator un objet est une forme, composée ou non). Ainsi, on peut trouver des commandes de transformation, de disposition, de groupage, etc. Nous verrons sans aucun doute dans un autre article plus technique, l'utilisation de ces commandes en détail.
- **Le menu Texte :** ce menu comme le nom l'indique est dédié aux textes. Vous pouvez de ce fait, trouver des commandes liées comme le choix de la police, la taille, modifier la casse, ou simplement afficher ou non les caractères masqués.
- **Le menu Selection :** ce menu gère tout ce qui concerne la sélection des objets. Vous pouvez facilement grâce à ce menu, sélectionner ou dé-sélectionner en groupe ou individuellement.
- **Le menu Effet :** ce menu regroupe tous les filtres applicables sur vos objets. Un filtre est une opération que l'on applique directement sur un objet ou un groupe, par exemple un flou gaussien. Les filtres sont à la base très connus sous **Photoshop**.
- **Le menu Affichage :** ce menu permet de gérer l'affichage de votre plan de travail, de créer de nouvelles vues, d'afficher ou de masquer certains éléments de la fenêtre.
- **Le menu Fenêtre :** ce menu permet simplement de lancer les fenêtres flottantes nécessaires à votre travail.
- **Le menu Aide :** ce menu regroupe tout ce qu'il faut savoir pour l'aide d'**Illustrator**. Il permet aussi de s'enregistrer auprès de la société **ADOBE** pour recevoir les dernières nouveautés du logiciel (cependant, il vous a été demandé lors de l'installation, donc si vous l'avez déjà fait, il n'est pas nécessaire de le refaire).

La partie 3 est un raccourci vers **ADOBE Bridge** (image ci-dessous). Ce dernier est un organisateur d'image et de travaux qui les lie à toutes les applications de la suite **Creative Suite** et cela depuis la version CS2.



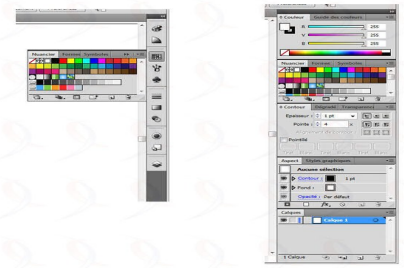
La partie 4 concerne la disposition de l'interface. En effet, **Illustrator** propose plusieurs types d'interface selon les besoins de l'utilisateur. Parmi lesquels on retrouve :

- Les indispensables (Lien89)
- Automatisation (Lien90)
- Comme FreeHand (Lien91)
- Comme InDesign (Lien92)
- Comme Photoshop (Lien93)
- Impression et épreuvage (Lien94)
- Peinture (Lien95)
- Typographie (Lien96)
- Web (Lien97)

Chaque interface est basée sur une disposition bien particulière et l'utilisateur pourra choisir selon son affinité. Il est possible également de faire soi-même sa propre disposition des fenêtres et de l'enregistrer dans un thème.

La partie 5 est une barre de recherche, il suffira à l'utilisateur de taper les mots clés de sa recherche pour lancer la recherche en ligne à l'adresse "*community.adobe.com*".

La partie 6 correspond aux options de dessin, que l'on peut voir sous 2 versions, une version réduite idéale pour les résolutions 1024x768 et une version plus standard pour des résolutions avec des fenêtres comme le montre l'image ci-dessous.



À gauche, nous avons la version réduite, comme dit précédemment, vraiment adaptée pour les résolutions faibles de l'ordre de 1024x768 et 1280x1024. Elle se présente sous la forme d'une barre d'outils verticale, pour laquelle chaque bouton est apparenté à une ou plusieurs fenêtres.

À droite, nous avons la version classique, celle que l'on retrouve dans chaque version d'Illustrator depuis le début. Il s'agit simplement d'un empilement de fenêtres des outils par défaut. Par la suite, l'utilisateur pourra sans soucis rajouter ou enlever selon ses besoins.

La partie 7 est un sélecteur de plan. Rappelons que les plans de travail – qu'on a expliqué au début de l'article - sont une des grandes nouveautés de la version **CS4**. Donc à partir de ce sélecteur vous pouvez choisir le plan où vous voulez travailler. Dans notre exemple, il n'y a que le choix d'un plan, car on a bien notifié 1 comme nombre de plans au début.

La partie 8 est le niveau de zoom du plan de travail en cours. Il est généralement conseillé de travailler avec les zooms pour créer les détails de vos illustrations.

La partie 9 est la barre d'outils d'**Illustrator CS4**, on peut y voir différents types d'outils comme des outils de sélections, des outils de créations (trait, courbe, forme, pinceau...), des outils de déformations ou simplement des outils de maillages.

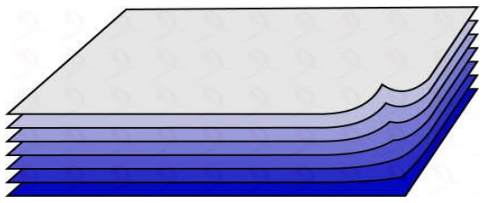
Nous ne verrons pas l'utilisation des outils en détail, car ce

serait trop long et surtout, ce n'est pas le but de cet article. Nous les verrons au fur et à mesure dans les prochains articles, plus techniques, **avec des exemples concrets**.

Et pour finir, **la partie 10** est une barre de configuration pour l'outil en cours. Elle permettra de régler certaines données de l'outil, par exemple, si vous prenez l'outil texte, vous pourrez changer la police, la taille ou la couleur dans cette barre.

3.4. Les calques ou "layers"

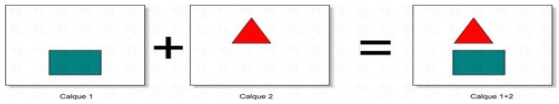
Quand on dessine en vectoriel, il est important de commencer avec les bonnes pratiques et cela, dans n'importe quel logiciel de vectorisation, pas seulement **Illustrator**. Les bonnes pratiques commencent déjà par l'utilisation **de calques ou "layers"** dans vos projets. Mais qu'est-ce que c'est exactement un calque ? Un calque peut se comparer à une couche transparente que l'on rajoute au-dessus d'un dessin. Si vous ne voyez pas, imaginez simplement des feuilles transparentes avec des images dessus se superposant les unes avec les autres comme le montre le schéma ci-dessous :



Les personnes qui ont déjà utilisé des logiciels comme **Photoshop, InDesign** ou plus particulièrement **AutoCAD**, connaissent les avantages d'utiliser les calques plutôt que de faire tout sur la même feuille.

Ces avantages sont nombreux, on peut citer le fait de pouvoir travailler des parties du dessin sans toucher à d'autres, de faire plusieurs versions de plans avec la possibilité de changer en un clic de souris, etc...

Voyons un petit schéma expliquant la fonction de base d'un calque :



Que voyons-nous sur le dessin ci-dessus ? On peut voir 2 calques qui contiennent chacun une forme : un rectangle vert sur le calque 1 et un triangle rouge sur le calque 2. Si on décide d'afficher les 2 calques ensemble, on peut voir le rectangle vert et le triangle rouge sur la même feuille.

À noter que si on décide de travailler sur le résultat des 2 calques, ce sera le calque placé au-dessus qui prendra les objets créés.

Sous **Illustrator**, il existe cependant plusieurs types de calque que l'on verra dans nos futurs articles basés sur les différentes techniques d'**Illustrator**.

3.5. RVB, CMJN ?

Pendant l'utilisation d'un logiciel de dessin, et plus particulièrement sous Illustrator CS4, vous avez sans doute remarqué le terme RVB et CMJN – nous l'avons vu lors de notre présentation de la création du plan de travail. Ce sont des modes de couleur couramment utilisés.

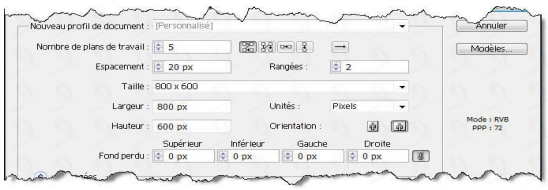
Comme dit plus haut, le mode de couleur **RVB** (Rouge, Vert et Bleu) est surtout utilisé pour des réalisations de type écran, comme des illustrations de sites web, d'illustrations pour de la PAO (Présentation Assistée par Ordinateur) ou simplement des images que l'on stockeraient. Comme vous pouvez le voir sur le dessin à gauche, la palette **RVB** dispose de trois barres de couleurs avec des valeurs de 0 à 255 chacune.

Le mode de couleurs **CMJN** est aussi connu sous le terme de **quadrichromie**. Les initiales **CMJN** signifient simplement : **C**yan, **M**agenta, **J**aune et **N**oir. Ce mode est surtout utilisé pour tout travail de type papier comme les illustrations sur journaux. Le mode **CMJN** dispose de 4 barres allant de 0 à 100%.

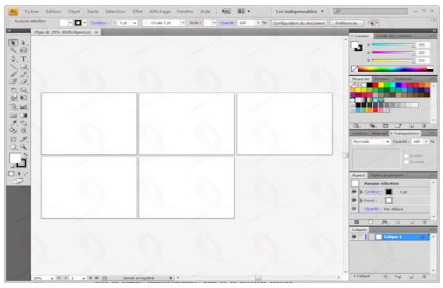
4. Nouveauté de la version CS4

Les nouveautés de la version **CS4** par rapport à la version **CS3** ne sont pas nombreuses. Cependant, elles apportent un nouvel intérêt non négligeable à la composition de dessin. Dans un premier temps, nous avons vu l'interface, un nouveau style de fenêtre modulaire comme on l'a vu juste avant sur un fond gris – ce qui permet de diminuer la fatigue. On a également de nouveaux outils comme l'outil « Forme de tâche », les masques d'écrêtage, la sélection par aspect, etc. Nous verrons plus en détail dans d'autres articles l'utilisation de tout cela par la pratique.

Mais, il serait intéressant de voir en détail les espaces (ou plans) de travail qui est une des grandes nouveautés d'**Illustrator CS4**. Comme dit plus haut dans cet article, **Illustrator CS4** permet de créer plusieurs plans de dessin pour un même projet. Prenons un exemple simple et concret : créons un projet avec 5 plans. Comme avant, Fichier->Nouveau...



Ce qui nous donne:



Comme vous pouvez le voir, il y a bien 5 plans dans l'interface d'**Illustrator CS4**. Le mieux est de vous montrer en action via la vidéo ci-dessous:

Dans la vidéo on peut voir que l'utilisateur peut utiliser les plans de travail pour dessiner plusieurs modèles de son projet dans le même fichier **.ai** (Fichier **Illustrator**). On peut voir également la mise en place de la disposition des plans grâce à l'outil " Plan de travail " de la barre d'outils.

5. Et après ?

Faire une présentation de toutes les possibilités qu'offre le logiciel **Illustrator** serait un travail titanesque, voire impossible dans un même et unique article. Il est donc logique, qu'on ne puisse pas tout présenter comme on le voudrait. Cependant, il y aura d'autres articles beaucoup plus techniques qui se rajouteront au fur et à mesure et qui vous montreront le fonctionnement de chaque outil dans des cas bien particuliers et bien sûr, le tout complètement illustré et animé par des vidéos.

6. L'avis d'un passionné de graphismes sur Illustrator CS4

Grand passionné de graphismes en tout genre (2D, 3D, vectoriel, ou matriciel), ce fut un grand honneur pour moi de tester **Illustrator CS4**. Habitué au monde du vectoriel avec le logiciel **Xara Xtreme Pro** et donc disposant déjà de quelques ressources dans le domaine, le test du logiciel Illustrator m'a permis de m'adapter sur des techniques différentes. Alors que dire finalement du logiciel **Illustrator** ?

Le logiciel est extrêmement puissant et professionnel, il propose de nombreux outils qui font qu'**Illustrator CS4** est unique et qui permettent de créer un dessin rapidement et avec une qualité hors du commun. Malheureusement, même si le logiciel dispose d'une interface plutôt agréable et vraiment adaptée, il convient de dire que **Illustrator** est vraiment dur à la prise en main pour des débutants. Par exemple, la création de dégradés transparents pour lesquels il faut créer un calque spécial, alors que d'autres applications le gèrent automatiquement et directement via un outil. Cependant, le logiciel est axé plutôt pour les professionnels que pour les personnes souhaitant apprendre le dessin, même si ça reste un must pour cela, et de ce fait, justifie complètement l'achat d'une licence.

Retrouvez l'article de Faith's Fall en ligne : Lien98

Outils pour construire un code jQuery évolutif.

Avec un JavaScript omniprésent et des codes de plus en plus complexes, le programmeur se doit de maîtriser les bases du langage, la structuration et la modularisation du code, il se doit d'écrire un code gérable, réutilisable et facilement évolutif.
Dans cet esprit, je vous propose de généraliser l'usage de la clôture jQuery (closure jQuery).
L'utilisation de la clôture implique celle de la fonction globale et d'un espace de noms.

1. La clôture jQuery

1.1. Introduction

Non, il ne s'agit pas de clôturer le compte bancaire de jQuery ! Il s'agit simplement de protéger votre code.

En anglais on parle de "closure" et ce mot peut se traduire par fermeture, mais aussi par clôture que je trouve plus parlant.

1.2. C'est quoi ?

Il s'agit d'une fonction anonyme function(){...}

entourée d'une clôture (function(){...})();

et dédiée à jQuery : (function($){...})(jQuery);

1.3. Un espace privé

La clôture (function(){...})(); est un espace privé. Elle s'exécutera automatiquement.

La clôture jQuery (function($){...})(jQuery); est un espace privé, dans lequel le symbole $ est clairement et exclusivement attribué à jQuery. Elle s'exécutera automatiquement.

1.4. Plus de conflits pour le $

Si vous utilisez une autre bibliothèque ("framework"), la clôture vous assure qu'il n'y aura pas de conflit pour la maîtrise du symbole $.

Exemple n°1 (Lien99)

```
(function($){
        //jQuery
        $("#conteneur")
            .css("backgroundColor","#FFCC66");
})(jQuery);

//Prototype
$('affiche').addClassName('active').show();

//jQuery
//erreur : $("#conteneur") is null !
$("#conteneur").css("color","red");
```

1.5. Objets, fonctions et variables locales

Tout ce qui est déclaré à l'intérieur de la clôture est inconnu en dehors à la seule condition que vous utilisiez le mot clé "var" comme il se doit.

L'utilisation de l'espace de noms de l'objet global "window" par votre code est source d'inexactitude, d'instabilité et d'insécurité.

Ce qui était tolérable à l'époque de scripts de quelques lignes est rigoureusement à proscrire à l'époque du "web 2.0" où nous mélangeons des centaines de lignes de scripts d'auteurs différents. Le risque de collisions dans l'espace de noms de l'objet global "window" devient une certitude.

La qualité de votre code est inversement proportionnelle aux nombres d'objets, de fonctions et de variables que vous avez introduits dans l'espace de noms de l'objet global "window".

Idéalement votre code ne doit dépendre que d'un seul objet global.

Exemple n°2 (Lien100)

```
(function($){
        a = "je suis a";
        var b = "je suis b";

        function mafunc(message){
                alert(message);
        }
})(jQuery);

alert(a);

//erreur : "b is not defined !"
alert(b);

//erreur : "mafunc() is not defined !"
mafunc("Hello world !");
```

1.6. L'opérateur this

Dès la clôture créée, le mot clé "this" représentant l'objet "window" sera disponible tant que vous ne modifierez pas sa valeur.

Je cite[2] :

"Le mot-clé this fait référence à l'objet de contexte (ou objet courant). En général, dans une méthode, this fait référence à l'objet appelant."

"Du fait du « passage » de this aux fonctions, this n'a pas de valeur fixe pour une fonction. Cela signifie qu'une fonction n'a pas de « propriétaire » ou de « parent », même s'il s'agit d'une méthode. En d'autres mots, une méthode n'est pas liée à l'objet dont elle est une méthode."

"La distinction entre méthodes et fonctions se fait uniquement au moment de l'appel : les appels de méthode passent l'« objet parent » comme valeur de this, tandis que les appels de fonctions passent l'objet global en tant que this."
Fin de la citation[2]. (Lien101)

Exemple n°3 (Lien102)

```
(function($){
	//équivalent à window.confirm()
	var r = this.confirm("Bienvenue dans mon cahier d'excercices sur jQuery & Co.");

	if (r == true){
		//équivalent à window.alert()
		this.alert("You pressed OK!")
	} else {
		this.alert("You pressed Cancel!")
	}

	//Une fonction anonyme entourée d'une clôture. Elle s'exécutera automatiquement.
	(function(){
		this.alert("La valeur de this n'ayant pas été modifiée, sa valeur est : " + this);
	})();

	//http://james.padolsey.com/javascript/jquery-delay-plugin/
	$.fn.delay = function(time, callback){
		jQuery.fx.step.delay = function(){};
		return this.animate({delay:1}, time, callback);
	}

	//On peut conserver la valeur actuelle de this en l'affectant à une variable, par exemple that
	var that = this;

	//équivalent à $(window).ready()
	$(this).ready(function(){

		alert("Dans $(window).ready(), that = " + that + ", et this = " + this);

		$("#conteneur h1").delay(2000, function(){
		    $(this)
			.css("color","#339900");

			alert("Dans cette fonction anonyme, that = " + that + ", et this = " + this);
		});
	});
})(jQuery);
```

Avec Internet Explorer 8 vous devrez actualiser la page pour obtenir l'affichage de toutes les boites de dialogue.

Avec Firefox, voici le contenu des trois dernières boites de dialogue :

- La valeur de this n'ayant pas été modifiée, sa valeur est : [object Window]
- Dans $(window).ready(), that = [object Window], et this = [object HTMLDocument]
- Dans cette fonction anonyme, that = [object Window], et this = [object HTMLHeadingElement]

1.7. Mais alors, comment ?

Je vous entends : "Si j'adopte son système, si je clôture tous mes programmes, comment appeler mon code ?"

Il faut conserver, au niveau global, un point d'entrée qui sera un objet ou une fonction. Mais pas n'importe comment.

2. La fonction globale

2.1. Exemple d'une "news box"

2.1.1. Description

Il s'agit de la version adaptée en jQuery d'une "news box" proposée par MARCHA (Lien103) en réponse à cette question (Lien104).

"News box" : diffusion verticale continue, de bas en haut, de messages (brefs, en principe) attirant l'attention sur de nouvelles informations. La courte pause avant la reprise de la diffusion indique au lecteur que la série est terminée.

Exemple n°4 : "News box" sans fonction globale (Lien105)

```
(function($){
	var speed = 1;
	var offset = 5;
	var interval = 60;
	var pos;
	var pos_initial;

	function anim() {
		$("#newslist").css({
			visibility:"visible",
			top:Math.floor(pos)
		});

		pos -= speed;

		if(pos < (-1 * $("#newslist")
				.height())) {
			pos = pos_initial;
		}
	}

	function startAnim() {
		pos_initial = $("#newsbox")
				.height() + offset;
```

```
                pos = pos_initial;
                setInterval("anim()", interval);
        }

        $(this).ready(function(){
                $("#newsbox").hover(
                        function(){
                                speed = 0;
                        },
                        function(){
                                speed = 1;
                        }
                );

                startAnim()
        });
})(jQuery);
```

2.1.2. Bug !

Avec Firefox et l'extension Firebug, le programme s'interrompt sur l'erreur : "anim is not defined", car la fonction anim() est inaccessible.

Alors qu'en absence de clôture le programme donne entière satisfaction.

Je vous entends : "On nous agace avec la clôture et dès le premier véritable exemple rien ne va plus !"

2.1.3. Pourquoi ?

La source de l'erreur est :

```
setInterval("anim()", interval)
```

Le problème vient de cette ligne de code qui demande à l'objet global "window" d'appeler la fonction anim() toutes les 60 millisecondes.

Or la fonction anim(), protégée par la clôture, est inaccessible pour l'objet global "window".

2.1.4. Ouvrez la porte !

Hé oui ! Nous avons besoin d'un point d'entrée dans la clôture.

Je cite[3] :
"Je vous rappelle qu'en JavaScript tout fonctionne en se fondant sur des objets. (...) Le langage JavaScript permet de créer simplement un objet en se fondant sur l'objet Object ou en utilisant une forme littérale dont la syntaxe est décrite par la notation JSON. (...) Il est possible d'ajouter, de modifier et de supprimer les entrées de l'objet tout au long de sa vie."

Extrait de la page 5 de la citation 3

```
var obj = new Object();

var obj = { ... };
// avec la notation JSON

obj["attribut"] = "valeur1";
// similaire à obj.attribut = "valeur1";

obj["methode"] = function(param1, param2) {
  ...
};
// similaire à obj.methode = function( ... ){ ...
};
```

Fin de la citation[3]. (Lien106)

2.2. Exemple d'une "news box" utilisant une fonction globale

2.2.1. Fonction globale et objet global

Nous pouvons donc écrire la fonction anim() de la manière suivante :

```
var anim = function() { ... };
```

Mais nous devons omettre le mot clé "var" pour rendre anim() accessible depuis l'extérieur de la clôture.

C'est cette construction :

```
anim = function() { ... };
```

que j'appelle une fonction globale.

Un fragment de code existant rarement seul, je me servirai de la fonction globale ou de l'objet global :

```
monObjet = { ... };
```

comme point d'entrée permettant de communiquer avec la clôture.

2.2.2. Exemple

Exemple n°5 : "news box" utilisant une fonction globale (Lien107)

```
(function($){
        var speed = 1;
        var offset = 5;
        var interval = 60;
        var pos;
        var pos_initial;

        anim = function() {
                $("#newslist").css({
                        visibility:"visible",
                        top:Math.floor(pos)
                });

                pos -= speed;

                if(pos < (-1 * $("#newslist")
                                .height())) {
                        pos = pos_initial;
                }
        };

        function startAnim() {
                pos_initial = $("#newsbox")
                              .height() + offset;
                pos = pos_initial;
                setInterval("anim()", interval);
        }
```

```
        $(this).ready(function(){
                $("#newsbox").hover(
                        function(){
                                speed = 0;
                        },
                        function(){
                                speed = 1;
                        }
                );

                startAnim()
        });
})(jQuery);
```

Attention, dans la fonction

```
window.setInterval(expression, millisecondes);
```

expression peut s'écrire sous trois formes :

1. un texte :

```
window.setInterval("anim()", 60)
```

2. le nom de la fonction :

```
window.setInterval(anim, 60);
```

3. une fonction anonyme :

```
window.setInterval(function(){anim();}, 60);
```

La forme 1 est la seule qui impose l'usage d'une fonction globale, car l'objet "window" convertit l'expression "anim()" en une recherche de la fonction window.anim() qu'il ne peut trouver lorsqu'elle est placée dans un espace privé.

Ce qui précède s'applique également à la fonction :

```
window.setTimeout(expression, millisecondes)
```

2.3. Pollution de l'espace de noms

En choisissant de promouvoir la clôture, espace privé, je souhaitais également libérer l'espace de noms "window", or chaque fonction globale ou objet global encombrera un peu plus cet espace de noms. De plus si comme il se doit, votre code est un composant réutilisable, le risque de collisions avec un autre composant du même nom est très élevé.

Il est clair que l'utilisation de la clôture implique l'usage d'un espace de noms privé.

3. Un espace de noms ("namespacing")

3.1. Rappel

Nous avons vu que la clôture offre un espace privé où le symbole $ est exclusivement attribué à jQuery.

Nous avons vu que la fonction globale permet d'obtenir un point d'entrée dans la clôture jQuery mais aux prix d'une pollution de l'espace de noms.

3.2. Introduction

Je cite[4] :
"Un espace de noms est une notion permettant de lever une ambiguïté sur des termes qui pourraient être homonymes sans cela. Il est matérialisé par un préfixe identifiant de manière unique la signification d'un terme. Au sein d'un même espace de noms, il n'y a pas d'homonymes. Les espaces de noms aident à la construction de programmes modulaires".
Fin de la citation[4] (Lien108)

Je vous invite à lire, au minimum, les pages 3 à 6 de l'excellent tutoriel de Nourdine Falola : Espaces de noms (ou namespace) en JavaScript (Lien109). Ces pages constituent l'introduction indispensable à la suite de cet article.

Idéalement, votre code ne doit être accessible qu'au travers d'un et d'un seul objet global.

3.3. Comment ?

Comme notre espace de noms contiendra du code jQuery, nous devons utiliser la clôture jQuery.

La première solution qui vient à l'esprit est de construire un objet global dans l'espace de noms window à l'intérieur d'une clôture jQuery.

On peut également construire un objet global dans l'espace de noms jQuery :

```
(fonction($){
        $.monObjet = { ... } ;
})(jQuery);
```

à l'intérieur d'une clôture jQuery.

3.4. Solution

Après avoir testé ces deux approches, une troisième solution a ma préférence.

Il s'agit toujours de construire un objet global dans l'espace de noms window, mais c'est la clôture jQuery qui génère l'objet grâce à l'écriture JSON.

Suivant votre degré d'expérience dans l'écriture de code JSON cette solution peut vous paraître complexe, mais il me semble qu'elle est d'un usage plus simple que les deux approches évoquées ci-dessus. De même, la gestion des exceptions et la création des sous-espaces de noms me semblent plus faciles et plus fiables.

JavaScript est très efficace pour résoudre une expression d'appel complexe, vous ne devez pas craindre la très légère diminution des performances qui en résulte, car elle est compensée largement par l'aspect auto-documentaire et la fiabilité de votre code.

3.5. Espace de noms dvjh

En notation JSON, l'objet global dvjh peut s'écrire comme suit :

```
try {
  var dvjh = {
    isdvjh: "dvjh/danielhagnoul",
    isMe: function(){
      return this.isdvjh;
    },
    infos: function(){
      return "Espace de noms dvjh.";
    }
  };
}
catch(err){
  alert(err);
}
```

Avec la génération par la clôture jQuery :

```
try {
  var dvjh = (function($) {
    //notre clôture peut toujours contenir
variables,
    //fonctions et objets privés.

    var isdvjh = "dvjh/danielhagnoul";

    return {
      isMe: function(){
        return isdvjh;
      },
      infos: function(){
        return "Espace de noms dvjh";
      }
    };
  })(jQuery);
}
catch(err){
  alert(err);
}
```

Et le sous-espace de noms dvjh.math :

```
try {
  dvjh.math = (function($) {
    var isdvjh = "dvjh_math/danielhagnoul";

    if (dvjh.isMe() != "dvjh/danielhagnoul/")
return null;

    return {
      isMe: function(){
        return isdvjh;
      },
      infos: function(){
        return "Espace de noms dvjh.math";
      }
    };
  })(jQuery);
}
catch(err){
  alert(err);
}
```

La gestion des exceptions et le test sur isMe nous assurent de l'existence d'un objet dvjh et que cet objet dvjh est bien le nôtre.

3.6. Composition de l'espace de noms dvjh le 9 août 2009

L'espace de noms dvjh doit être vu comme un "proof of concept" (une démonstration de faisabilité), il vous montre comment réaliser un espace de noms en jQuery, il ne s'agit pas d'un espace de noms de référence.

J'ai toutefois pris soin de le remplir soit avec des codes utiles pour jQuery soit avec des exemples de code intéressants.

Ci-dessous, le contenu de l'interface de l'espace de noms dvjh et de ses sous-espaces.

3.6.1. dvjh

- Exception type dvjh : dvjh.exception(message, args);
- alert(contenu de l'exception) : dvjh.displayException(exc);
- Séparateur de texte : dvjh.separateur(s);
- Interface de dvjh : dvjh.infos();

Le fichier dvjh.js contient également des fonctions utiles pour jQuery :

- jQuery delay plugin de James Padlosey (Lien110)
- Regex Selector for jQuery de James Padlosey (Lien111)

et le selecteur contains dans une clôture annexe.

Ainsi que Function.protoype.memoize de Keith Gaughan (Lien112) qui est utilisée dans dvjh.math et dvjh.math.statistic.

Je cite[9] :
"La mémoization est une technique d'optimisation de code consistant à réduire le temps d'exécution d'une fonction en mémorisant ses résultats d'une fois sur l'autre. Bien que liée à la notion de cache, la mémoization désigne une technique bien distincte de celles mises en oeuvre dans les algorithmes de gestion de la mémoire cache."

"Une fonction mémoizée stocke les résultats de ses appels précédents dans une table et, lorsqu'elle est appelée à nouveau avec les mêmes paramètres, renvoie la valeur stockée au lieu de la recalculer."

"Une fonction peut être mémoizée seulement si elle est référentiellement transparente, c'est-à-dire si sa valeur de retour ne dépend que de la valeur de ses arguments."

"La mémoization est une façon de diminuer le temps de calcul d'une fonction, au prix d'une occupation mémoire plus importante."
Fin de la citation[9] (Lien113)

3.6.2. dvjh.math

- Modulo informatique : dvjh.modulo(a, b);
- Nombre de Fibonacci : dvjh.fib(n);
- Nombre d'or à la puissance n : dvjh.phi(n);
- Interface de dvjh.math : dvjh.math.infos();

dvjh.math montre comment créer un sous-espace de noms.

En dehors du modulo informatique (Lien114), dvjh.math donne un exemple d'application de la mémoization pour le calcul du Nombre de Fibonacci.

3.6.3. dvjh.math.statistic

- Factorielle de n : dvjh.math.statistic.fact(n);
- Nombre d'arrangements à k éléments : dvjh.math.statistic.arran(n,k);
- Nombre de combinaisons à k éléments : dvjh.math.statistic.combi(n,k);
- Interface de dvjh.math.statistic : dvjh.math.statistic.infos();

dvjh.math.statistic montre comment créer un sous-sous-espace de noms et donne un exemple d'application de la mémoization pour le calcul de la Factorielle.

3.6.4. dvjh.calendar

- Date vers jour julien :
- dvjh.calendar.date_to_jj(an,mois,jours,heures,minutes,secondes);
- Jour julien vers date : dvjh.calendar.jj_to_date(jj);
- Jour de la semaine : dvjh.calendar.joursemaine(jj);
- Jour julien vers jour julien modifié : dvjh.calendar.jj_to_mjd(jj);
- Jour julien modifié vers jour julien : dvjh.calendar.mjd_to_jj(mjd);
- Interface de dvjh.calendar : dvjh.calendar.infos();

Les astronomes et les historiens amateurs ne se poseront pas la question de l'utilité des Jours Juliens. Pour les autres, je signale que la manipulation de dates par l'intermédiaire des JJ est beaucoup plus simple qu'il n'y paraît.

Je cite[10]:
"La période julienne est une numérotation continue et décimale des jours, indépendante du calendrier, de l'année, du mois, du jour de la semaine et du numéro du jour dans le mois."

"Elle a pour origine le 1er janvier de l'an 4713 avant notre ère (c'est-à-dire le 1er janvier - 4712 en notation des astronomes)."

"La continuité des jours permet de calculer un intervalle de temps sans risque d'erreur même s'il couvre à la fois les calendriers julien et grégorien. Les formules de calcul tiennent compte du changement de calendrier qui a eu lieu en 1582."

"Exemple : le 1er janvier 1001 (calendrier julien) à 0 h correspond au jour julien 2 086 673,5 ; le 1er janvier 2001 (1000 ans plus tard, mais en calendrier grégorien) à 0 h correspond au jour julien 2 451 910,5. Entre ces deux dates il s'est donc écoulé 365 237 jours. Remarquez qu'il ne s'est pas écoulé 365 250 comme on pourrait le déduire un peu rapidement. Il manque les 10 jours dûs à l'introduction du calendrier grégorien et les 3 jours bissextiles qui n'ont pas été ajoutés en 1700, 1800 et 1900."
Fin de la citation[10] . (Lien115)

3.6.5. dvjh.effects

- Animer en suivant l'ordre : dvjh.effects.byOrder(jQueryObjets, params, duration, easing, callback);
- Interface de dvjh.effects : dvjh.effects.infos();

```
dvjh.effects.byOrder($('.image_menu_on'),
{width:400, height:300}, 1000, 'easeInSine',
function(){alert('wow!')} );
```

La fonction byOrder() applique à tour de rôle la même animation à chaque objet jQuery contenu dans l'array jQueryObjets.

Le code de la fonction byOrder() n'est pas de moi, je sais l'avoir vu et copié sur l'internet mais j'ai perdu la référence. Mille excuses à l'auteur !

3.6.6. dvjh.utilities

- Regroupement de fonctions globales.
- Interface de dvjh.utilities : dvjh.utilities.infos();

Sert de point d'ancrage pour les fonctions globales (objets globaux).

3.6.7. dvjh.utilities.contentTable()

```
dvjh.utilities.contentTable('conteneur', 'tdm',
6, {tdmTitre:'Sommaire'});
```

Cette fonction globale (idée et code originaux (Lien116) de Giminik (Lien117), version jQuery de Daniel Hagnoul) permet de construire une table des matières.

3.7. Nota bene

Lorsque vous construirez votre propre espace de noms, pensez à lui donner un nom suffisamment long pour que le risque de collision avec un autre espace de noms soit réduit, il eût été préférable de nommer mon espace de noms danielhagnoul plutôt que dvjh, mais on se défait difficilement d'une mauvaise habitude.

Retrouvez l'article de Daniel Hagnoul en ligne : Lien118

Créer des bulles d'information avec aisance en CSS

Cet article est la traduction de : Create CSS pin balloons with ease (Lien119). Retrouvez toutes les traductions de Janko Jovanovic disponibles sur css.developpez.com (Lien120).
Au cours de cet article nous verrons une méthode pour créer uniquement avec du code CSS et (X)HTML des étiquettes ayant la forme de bulles d'information.

1. Introduction

La page d'accueil du site de l'Africa Tour 2008 (Lien121), créée par les mecs de Stylishmedia (Lien122) est un bon exemple montrant comment mettre en place facilement ce genre de procédé. Dans ce tutoriel, je tenterai de reproduire (par mes propres moyens) l'effet qu'utilise la carte présentée ci-dessous.

Comme je l'ai dit, l'effet est très simple puisque chaque continent est, par défaut, marqué par une étiquette en forme de bulle (je suis dans l'incapacité de donner un nom plus explicite) et à son survol elle s'agrandit. C'est le genre de choses que nous avons dû réaliser assez souvent, j'en suis sûr. Cependant, cet exemple a une astuce : la bulle grandira diagonalement en partant de son point d'ancrage (le point d'ancrage est représenté par le petit triangle placé à l'extrémité de la bulle).

Voici une petite démonstration en ligne (Lien123).

Nous avons trois tâches bien distinctes

- Ancrer proprement chaque bulle ;
- Positionner proprement chaque bulle ;
- Ajouter l'effet au survol.

2. Ancrage et positionnement

Chaque bulle doit être représentée par une division ( ), ayant une image en guise d'arrière-plan, positionnée de manière absolue (position: absolute) sur la carte du monde positionnée quant à elle de manière relative (position:relative). Voici la structure HTML à utiliser

```
<div id="map">
     <div id="america"></div>
     <div id="europe"></div>
     <div id="africa"></div>
     <div id="asia"></div>
     <div id="australia"></div>
     <div id="southAmerica"></div>
</div>
```

Et le code CSS associé

```
#map {
        width:669px;
        height:351px;
        background-image:url('map.jpg');
        position:relative;
}

#map div {
        width: 120px;
        height:60px;
        position:absolute;
        cursor:pointer;
        background-repeat:no-repeat;
}
```

Cette partie est très simple. Chacune des bulles doit avoir une position absolue et avoir des dimensions égales à 120px par 60px.

Maintenant, regardons l'image ci-dessous. Nous souhaitons ancrer la bulle au centre de l'Amérique du Nord à la position x = 150px et y = 120px. La méthode la plus simple pour positionner un élément est de fixer la valeur CSS du top et left qui lui sont associés. Pour ce faire, nous devons soustraire la largeur et la hauteur de l'image représentant la bulle à la valeur de l'abscisse x et de l'ordonnée y du point d'ancrage. Dans notre cas, la position de notre point est x = 30px et y = 60px.

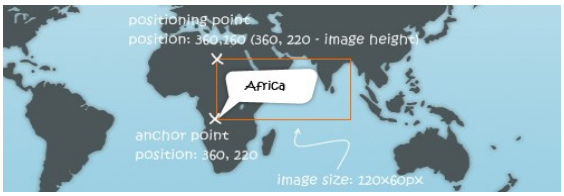
Mais ce n'est pas tout. Nous voulons que notre point d'ancrage reste inchangé pendant que la bulle s'élargira dans la direction opposée. Afin d'y parvenir, nous devons positionner l'image de fond correctement. La balise identifiable par #america devra avoir la propriété CSS background-position: right bottom qui nous assurera que le point d'ancrage ne bougera pas.

```
#america {
	background-
image:url('america_small.png');
	top:60px;
	left:30px;
	background-position:right bottom;
}
```

Le code qui suit nous permettra d'afficher aisément une autre image au survol :

```
#america:hover {
	background-image:url('america_big.png');
}
```

Regardons l'exemple suivant : la bulle associée à l'Afrique est symétrique (symétrie axiale (Lien124)) à celle de l'Amérique du Nord. Cela signifie que la position de l'image de fond sera différente.



L'image sera positionnée en bas et à gauche :

```
#africa {
	background-image:url('africa_small.png');
	top:160px;
	left:360px;
	background-position:left bottom;
}
```

Les autres bulles peuvent être placées en utilisant la même méthode. Voici le code CSS final :

```
#map {
	width:669px;
	height:351px;
	background-image:url('map.jpg');
	position:relative;
}
#map div {
	width: 120px;
	height:60px;
	position:absolute;
	cursor:pointer;
	background-repeat:no-repeat;
}

/* Bulles d'information fixées à droite */
#america {
	background-
image:url('america_small.png');
	top:60px;
	left:30px;
	background-position:right bottom;
}
#america:hover {
	background-image:url('america_big.png');
}
#europe {
	background-image:url('europe_small.png');
	top:50px;
	left:240px;
	background-position:right bottom;
}
#europe:hover {
	background-image:url('europe_big.png');
}
#southAmerica {
	background-
image:url('southamerica_small.png');
	top:190px;
	left:110px;
	background-position:right bottom;
}
#southAmerica:hover {
	background-
image:url('southamerica_big.png');
}

/* Bulles d'information fixées à gauche */
#africa {
	background-image:url('africa_small.png');
	top:160px;
	left:360px;
	background-position:left bottom;
}
#africa:hover {
	background-image:url('africa_big.png');
}
#asia {
	background-image:url('asia_small.png');
	top:60px;
	left:480px;
	background-position:left bottom;
}
#asia:hover {
	background-image:url('asia_big.png');
}
#australia {
	background-
image:url('australia_small.png');
	top:200px;
	left:540px;
	background-position:left bottom;
}
#australia:hover {
	background-
image:url('australia_big.png');
}
```

Comme je l'ai dit au début, ce n'était pas si compliqué. Maintenant, où pouvons-nous utiliser cet effet ? Premièrement sur les cartes (cartes du monde, des régions ou des villes). Vous pouvez l'utiliser pour indiquer n'importe quel endroit, la localisation du siège de votre entreprise, par exemple. Je ne vois pas d'autres possibilités, et vous ?

Retrouvez l'article de Janko Jovanovic traduit par Rodrigue Hunel en ligne : Lien125

Utilisation avancée des procédures stockées MySQL

Il y a quelques temps, j'ai écrit un article sur l'utilisation des procédures stockées dans MySQL et l'extension de MySQLi en PHP pour les exécuter. Je vais maintenant rapidement couvrir quelques techniques avancées que vous pouvez utiliser pour réduire le nombre de données transférées entre la base de données et votre application.

1. Récapitulatif des bases

D'abord un récapitulatif des bases pour ceux qui n'auraient pas lu mon article original. Les procédures stockées sont des bouts de code SQL stockés sur le serveur de base de données qui peuvent être appelés par leurs noms depuis les clients. Les procédures stockées peuvent traiter de multiples commandes SQL, prendre des paramètres et retourner des DataSets. Les procédures stockées sont présentes dans d'autres systèmes de bases de données depuis plusieurs années jusqu'à ce que récemment cette fonctionnalité soit accessible dans MySQL. Le format d'une requête SQL pour créer une procédure stockée ressemble à ceci :

```
DELIMITER $$

CREATE PROCEDURE `sdpExampleProcedure`
   (
      paramètre 1 VARCHAR(255),
      paramètre 2 INTEGER
   )
   BEGIN
      /*
      Requêtes SQL séparées par un point virgule
(;)
      Utilisant paramètre 1 et paramètre 2
*/
   END$$

DELIMITER ;
```

Notez la commande **DELIMITER** au début et à la fin qui change la manière de séparer les requêtes de MySQL. Ensuite pour appeler la procédure, vous pouvez utiliser la commande suivante :

```
CALL `sdpExampleProcedure`('testdata', '11');
```

2. Que puis-je faire avec les procédures stockées et pourquoi ?

Les procédures stockées peuvent être utilisées pour grouper les commandes SQL et ajouter une logique dans vos états SQL, tout en réduisant la quantité de données transférées depuis le serveur de base de données et votre application. Par exemple, en administrant des sessions dans une application PHP. Si vous voulez mettre à jour un enregistrement session quand une requête est faite pour une session existante, ou créer une nouvelle session en tout vous pourriez faire tout cela avec des contrôles dans une procédure stockée. Cela devrait normalement consister à utiliser une commande **SELECT** pour récupérer toutes les données voulues depuis le serveur de données en rapport avec l'id de session donné, ensuite en programmant votre code pour exécuter une commande **INSERT** ou **UPDATE** qui dépend du nombre de lignes retournées. Avec les procédures stockées vous pouvez inclure tout ceci dans une seule commande SQL simple qui retourne simplement vrai quand une session est créée, et faux lorsqu'une session est mise à jour.

3. Déclarer et configurer des variables dans vos procédures stockées

En addition des paramètres définis qui sont passés durant l'appel, il est possible de définir des variables sur lesquelles travailler durant l'exécution. Pour faire ça vous devez utiliser la commande **DECLARE**, faisant état du nom et du type de la variable. Dans l'exemple suivant, je définis 3 variables appelées "**session_count**", "**session_id**" et "**session_content**", de type **INTEGER**, **VARCHAR** (de taille 32) et **TEXT** respectivement.

```
DECLARE session_count INTEGER;
DECLARE session_id VARCHAR(32);
DECLARE session_content TEXT;
```

Maintenant que vous avez déclaré vos variables, vous pouvez configurer leur valeur en utilisant la commande **SET**. Une fois que ces variables ont une valeur, vous pouvez les utiliser plus tard dans votre procédure.

```
SET session_count = 1;
SET session_id =
'699d571326815e40b8e1ae99af04563c';
SET session_content = 'Du contenu de session
ici';
```

Si vous connaissez la valeur de vos variables au moment de les déclarer, vous pouvez utiliser la commande **DEFAULT** pour les configurer. Ci-dessous j'ai condensé les 6 commandes en 3 utilisant cette technique :

```
DECLARE session_count INTEGER DEFAULT 1;
DECLARE session_id VARCHAR(32) DEFAULT
'699d571326815e40b8e1ae99af04563c';
DECLARE session_content TEXT DEFAULT ''Du contenu
de session ici;
```

4. Donner des valeurs variables aux requêtes SELECT

Une utilisation excellente des variables dans vos procédures est d'utiliser le résultat d'une commande **SELECT** pour configurer la valeur d'une variable existante. Dans d'autres systèmes de bases de données,

ceci est vraiment simple, cependant dans MySQL bien que ce soit simple, la syntaxe est quelque peu inattendue.

Dans le but de transmettre une valeur depuis une requête **SELECT** vers une variable, il y a une ligne supplémentaire que nous devons ajouter à la requête **SELECT** juste avant la définition de la table dans laquelle les données seront extraites. La commande **INTO** met les données dans une variable définie. Voici un exemple pour illustrer son utilisation :

```
DECLARE session_count INTEGER;

SELECT COUNT(*)
INTO session_count
FROM `tblSessions`;
```

La variable session_count contient maintenant la valeur renvoyée par la requête **SELECT** qui compte le nombre de lignes dans la table des sessions. Il est aussi possible de mettre les valeurs dans de multiples variables comme dans l'exemple suivant :

```
DECLARE user VARCHAR(15);
DECLARE pass VARCHAR(32);

SELECT `strUserName`, `strPassword`
INTO user, pass
FROM `tblUsers`
LIMIT 0, 1;
```

5. Contrôler le flux de votre procédure stockée

Maintenant que nous avons configuré des variables et leurs avons assigné des valeurs, nous pouvons contrôler le flux de notre procédure pour décider de ce que nous allons faire par la suite. MySQL supporte une flopée de contrôleurs de flux que vous allez utiliser comme vous pourrez le voir dans d'autres langages de programmation.

5.1. La commande IF

La commande **IF** est bien sûr le contrôle de flux le plus connu et est vraiment similaire dans MySQL comme partout ailleurs. La commande commence par **IF**, suivi d'une opération logique qui retourne vrai ou faux et se termine par **THEN** pour la première ligne. Les actions qui ont besoin de s'exécuter après la condition vraie peuvent ensuite être définies et une commande **ELSE** peut être définie au besoin. La commande IF se termine par une commande **END IF ;** (prenez note du point virgule, source de beaucoup d'erreurs). La commande suivante vérifie si la variable "**session_count** " est supérieure à 0, si c'est le cas elle retourne vrai, sinon elle retourne faux.

```
IF session_count > 0 THEN
   SELECT 1;
ELSE
   SELECT 0;
END IF;
```

5.2. La commande CASE

MySQL inclut une implémentation de la commande switch case comme la plupart des langages de programmation usuels. Il ne peut y avoir aucun lien entre les cas comme il peut y en avoir en PHP mais il y a un cas par défaut au cas où aucune condition ne correspond. Voici un switch case qui essaie d'attraper une variable nommée "**catchme**" basée sur ses conditions. Le cas par défaut doit être appelé car "**catchme**" ne correspond à aucune des conditions définies dans le switch case.

```
DECLARE catchme INTEGER DEFAULT 10;

CASE catchme
   WHEN 2 THEN SET catchme = catchme + 2;
   WHEN 5 THEN SET catchme = catchme + 5;
   ELSE
      SET catchme = 50;
END CASE
```

5.3. La boucle WHILE

La boucle **WHILE**, outil commun hautement utilisé dans grand nombre de langages de programmation jamais faits par l'homme. MySQL n'y fait pas exception. La syntaxe est vraiment similaire aux autres la rendant facile à utiliser mais voici un résumé. Les mots clefs principaux que vous devez vous rappeler pour utiliser une boucle while dans MySQL sont **WHILE**, **DO** et **END WHILE;**. Quand vous définissez une boucle while vous pouvez lui donner un label qui peut être utilisé par plusieurs commandes pour contrôler la boucle. Le label vient avant la définition de la boucle while et après la fin de la boucle. L'exemple simple suivant crée une boucle WHILE appelée "**iterwhile**" qui s'itère tant que la variable "**iter**" est inférieure à 9.

```
DECLARE iter INTEGER DEFAULT 0;

iterwhile: WHILE iter < 9 DO
   SET iter = iter + 1;
END WHILE iterwhile;
```

5.4. La boucle LOOP

La commande **LOOP** ultra simple fait juste ça. Elle boucle indéfiniment jusqu'à ce que vous lui demandiez d'arrêter avec la commande **LEAVE**. Comme la boucle **WHILE**, elle peut être nommée grâce à un label dans sa définition afin d'être ciblée ensuite par des commandes pour la contrôler. Voici une commande **LOOP** appelée "**iterloop**" qui fait la même chose que l'exemple **WHILE** précèdent, mais utilise une commande **IF** et une commande **LEAVE** pour s'arrêter.

```
DECLARE iter INTEGER DEFAULT 0;

iterloop: LOOP
   IF iter < 9 THEN
      SET iter = iter + 1;
   ELSE
      LEAVE iterloop;
   END IF;
END LOOP iterloop;
```

5.5. La boucle REPEAT

La commande **REPEAT** est essentiellement une boucle **WHILE** qui vérifie si les conditions correspondent à la fin de chaque itération contrairement au **WHILE** qui vérifie en début. Cela veut dire que quand vous utilisez la boucle **REPEAT** elle sera exécutée au moins une fois. Voici l'équivalent **REPEAT** de **WHILE** et **LOOP** vus

précédemment :

```
DECLARE iter INTEGER DEFAULT 0;

iterrepeat: REPEAT
   SET iter = iter + 1;
UNTIL iter < 9
END REPEAT iterrepeat;
```

5.6. Les commandes LEAVE et ITERATE

Les commandes **LEAVE** et **ITERATE** sont similaires aux commandes **break** et **continue** utilisées dans d'autres langages de programmation. Elles peuvent être appliquées à n'importe quelle boucle **WHILE**, **LOOP** ou **REPEAT** qui sont actuellement actives en en faisant référence au label donné à la boucle. L'exemple suivant utilise les commandes **LEAVE** et **ITERATE** dans une **LOOP** appelée "**testloop**" :

```
DECLARE iter INTEGER DEFAULT 0;

testloop: LOOP
   IF iter = 10 THEN
      SET iter = iter + 12;
      ITERATE testloop;
   END IF;
   IF iter > 126 THEN
      LEAVE testloop;
   END IF;
   SET iter = iter + 1;
END LOOP testloop;
```

6. Conclusion

Un mix de ces commandes peut vraiment ajouter de la puissance à vos codes SQL. Pas seulement car elles peuvent réduire le trafic réseau mais parce que vous pouvez réduire le nombre d'appels à votre base de données significativement, rendant les choses un peu plus organisées. Si vous commencez seulement à utiliser ces techniques et avez des difficultés avec les erreurs de syntaxe SQL, veillez à bien vérifier vos commandes individuellement avant de les mettre ensemble dans une procédure. Gardez un oeil sur le point virgule aussi, il peut causer une belle somme de problèmes quand il n'est pas utilisé au bon endroit donc assurez vous de pouvoir différencier une commande d'une autre.

7. Liens

Vous pouvez aussi aller voir mes autres traductions (Lien126).

Retrouvez l'article de Luke Skelding traduit par Joris Crozier en ligne : Lien127

Blogs SGBD

SQL Server 2008, transtypage de date et optimiseur de requête

Avant SQL Server 2008, écrire une requête comportant un prédicat sur une date bien précise sans notion d'heures était plutôt fastidieux pour des requêtes sur des dates de transactions de stock par exemple où l'on désire un résultat à la journée . (Je prends l'exemple simple WHERE date = '20091001'). Dans ce cas là il fallait procéder à une transformation de la colonne de type DATETIME. Cependant, le nettoyage ou suppression des heures de cette colonne pouvait poser un inconvénient majeur car cela nécessitait un transtypage (je pense aux fonctions CAST et FLOOR qui donnent le résultat 20090101 00:00:00 par exemple). Hors le seul fait de transtyper une valeur de colonne empêche l'optimiseur de requêtes d'utiliser un index associé à cette même colonne. Cela obligeait par conséquent à écrire cette même requête avec un prédicat de date par intervalles (avec BETWEEN date AND date .).

Heureusement des nouveaux types de données DATE et TIME ont fait leur apparition avec SQL Server 2008 et il est maintenant possible de transtyper un type de données DATETIME en un autre type de données DATE en extrayant la valeur de date sans les heures. Ce nouveau transtypage est d'autant plus intéressant car dans les prédicats de requêtes où l'on doit exploiter seulement la partie date d'une colonne DATETIME l'optimiseur de requêtes peut maintenant profiter d'un index candidat associé à cette même colonne et ceci malgré la présence du transtypage.
Vérifions ceci par un exemple :
Une table de transactions de stock avec un type de mouvement (entrée, sortie), le code article concerné, la quantité et sa date de la transaction.

```
-- Création table des transactions de stocks
CREATE TABLE dbo.transaction_stocks
(
id INT IDENTITY(1,1) NOT NULL,
mvt CHAR(1) NOT NULL,
code_piece VARCHAR(10) NOT NULL,
qte INT NOT NULL,
date_tran DATETIME NOT NULL
);
-- Population de la table des transactions de
stocks
-- avec des données
DECLARE @i INT;
SET @i = 0;
WHILE @i < 10000
BEGIN
IF @i < 5000
  INSERT dbo.transaction_stocks
(mvt,code_piece,qte,date_tran) VALUES
('E','ME1001',CASE WHEN @i - RAND() * 1000 < 0
THEN 0 ELSE @i - RAND() * 1000 END,DATEADD(mm,-
FLOOR(RAND() * 100),GETDATE()));
ELSE
  INSERT dbo.transaction_stocks
(mvt,code_piece,qte,date_tran) VALUES
('S','ME1001',CASE WHEN RAND() * 1000 - @i > 0
THEN 0 ELSE RAND() * 1000 - @i END,DATEADD(mm,-
FLOOR(RAND() * 100),GETDATE()));
SET @i = @i + 1;
END;
SET @i = 0;
WHILE @i < 200
BEGIN
UPDATE dbo.transaction_stocks
SET date_tran = DATEADD(hh,- RAND() *
10,'20090101 23:59:00')
WHERE id = @i;
```

```
SET @i = @i + 1;
END;
-- Création d'un index cluster sur la date de
transaction
CREATE CLUSTERED INDEX IDX_date_tran
ON dbo.transaction_stocks
(
date_tran
);
```

Maintenant écrivons la requête qui permet de récupérer les transactions de stocks pour la journée du 1er janvier janvier 2009 (ce qui sous-entend toutes les transactions entre le 29 janvier 2009 00:00:00 et le 29 janvier 2009 23:59:00 inclus).

1ère solution : Suppression des heures de la colonne date_tran avec les fonctions CAST et FLOOR

```
SELECT * FROM dbo.transaction_stocks
WHERE CAST(FLOOR(CAST(date_tran AS
DECIMAL(15,4))) AS DATETIME) = '2009-01-01'
```

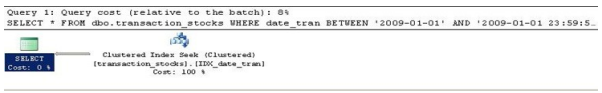
Le plan d'exécution est le suivant :

On voit que l'optimiseur de requêtes ne profite pas de l'index posé sur la table dbo.transaction_stocks du fait du transtypage présent dans le prédicat de la requête.

2ème solution : Ecriture du prédicat en se servant de la notion d'intervalle avec l'opérateur BETWEEN

```
SELECT * FROM dbo.transaction_stocks
WHERE date_tran BETWEEN '2009-01-01' AND '2009-
01-01 23:59:59'
```

Le plan d'exécution est le suivant :



Ici la réécriture de la requête avec l'opérateur BETWEEN permet à l'optimiseur de requêtes de profiter de l'index associé à la colonne date_tran.

Dernière solution : Avec le nouveau transtypage DATETIME en DATE

```
SELECT * FROM dbo.transaction_stocks
WHERE CAST(date_tran AS DATE) = '2009-01-01'
```

Le plan d'exécution est le suivant :

On voit ici que la requête utilise également une recherche d'index. On constate également que d'autres opérateurs de plan ont fait leur apparition (Opérateurs *Constant Scan* et *Compute Scalar*). Pour pouvoir utiliser l'index sur la colonne *date_tran,* SQL Server effectue d'abord une conversion de la valeur de notre argument *'2009-01-01'* en intervalle de date *'2009-01-01 00:00:00'* et *'2009-01-02 00:00:00'* et associe à cet intervalle une valeur de clé pour pouvoir faire une recherche dans l'index par la suite.
Le plan d'exécution en mode texte le confirme : On voit la définition des valeurs de l'intervalle avec l'opérateur computer scalar et la recherche par intervalle de dates avec ces valeurs au niveau de la recherche par index (Clustered Index seek ...)

```
|--Nested Loops(Inner Join, OUTER REFERENCES:
([Expr1007], [Expr1008], [Expr1006]))
|--Compute Scalar(DEFINE:(([Expr1007],[Expr1008],
[Expr1006])=GetRangeThroughConvert(CONVERT_IMPLIC
IT(date,[@1],0),CONVERT_IMPLICIT(date,[@1],0),
(62))))
|     |-Constant Scan
|--Clustered Index Seek(OBJECT:([TK432].[dbo].
[transaction_stocks].[IDX_date_tran]), SEEK:
([TK432].[dbo].[transaction_stocks].[date_tran] >
[Expr1007] AND [TK432].[dbo].
[transaction_stocks].[date_tran] < [Expr1008]),
WHERE:(CONVERT(date,[TK432].[dbo].
[transaction_stocks].
[date_tran],0)=CONVERT_IMPLICIT(date,[@1],0))
ORDERED FORWARD)
```

En d'autres termes cela revient à exécuter la requête suivante :

```
SELECT * FROM dbo.transaction_stocks
WHERE date_tran > '2009-01-01' AND date_tran <
'2009-01-02 00:00:00'
```

qui reste une requête par intervalle de dates.
Pour résumer, avec SQL Server 2008 vous pouvez rendre votre requête "sargable" même avec un transtypage d'un type DATETIME en DATE sur une recherche pour une date donnée, ce qui constitue une avancée au niveau de l'écriture de la requête même si on peut relativiser . :-)

Bonne recherche !!

Retrouvez ce billet sur le blog de mikedavem : Lien128

La FAQ SQL*Plus

La nouvelle FAQ SQL*Plus vient de sortir, découvrez dès maintenant une sélection de questions-réponses en provenance de cette FAQ

Quelles sont les variables d'environnement nécessaires pour lancer SQL*Plus ?

Les variables suivantes sont nécessaires :

- ORACLE_HOME : chemin vers l'installation Oracle
- ORACLE_SID : nom de l'instance qui sera utilisée par SQL*Plus
- PATH : chemin vers les binaires d'Oracle

D'où par exemple le résultat suivant (pour un serveur Unix) :

- $ORACLE_HOME : /oracle01/11gR1
- $ORACLE_SID : orcl
- $PATH : $PATH:/oracle01/11gR1/bin

Pour définir ces variables simplement, lancez une invite de commande et tapez les commandes suivantes

Pour un serveur Unix :

```
export ORACLE_HOME=/oracle01/11gR1
export ORACLE_SID=orcl
export PATH=$PATH:/oracle01/11gR1/bin
```

Pour un serveur Windows :

```
set %ORACLE_HOME%=c:\oracle01\11gR1
set %ORACLE_SID%=orcl
set %PATH%=%PATH%;C:\oracle01\11gR1\bin
```

L'initialisation de la variable $PATH sous Unix ou %path % sous Windows vous permettra d'accéder directement à SQL*Plus sans aller dans le dossier des binaires d'Oracle

Comment se connecter à SQL*Plus ?

Allez dans le dossier $ORACLE_HOME/bin ou vérifiez que ce répertoire fait partie de votre PATH (dans ce cas vous pourrez utiliser directement la commande sqlplus).

Ici se trouve l'exécutable sqlplus. Vous pouvez lancer directement sqlplus, on vous demandera un login et mot de passe pour vous connecter à votre base. Par contre cette méthode ne vous permettra pas de vous connecter à une autre base que celle par défaut ou avec un compte de type SYSDBA ou SYSOPER.

C'est pour cela qu'il convient de spécifier des paramètres de connexion.

Voici la syntaxe pour ce connecter à SQL*Plus :

```
sqlplus (utilisateur[/mot_de_passe]
[@identifiant_BDD] | /) [AS SYSDBA | AS SYSOPER]
| /NOLOG
```

Exemple pour se connecter avec l'utilisateur scott :

```
sqlplus scott/tiger
```

Exemple pour se connecter avec un compte SYSDBA

```
sqlplus sys/change_on_install AS SYSDBA
```

Pour se connecter avec le compte SYSTEM :

```
sqlplus /
```

Vous pouvez également taper la commande suivante et spécifier les paramètres de connexion directement dans le prompte SQL>

```
sqlplus /nolog
```

Ensuite pour se connecter à un compte on utilise la même syntaxe que précédemment mais avec la commande CONNECT

```
SQL> connect scott/tigger
```

ou

```
SQL> connect sys/change_on_install AS SYSDBA
```

Toutes ces chaînes de connexion vont se connecter sur le ORACLE_SID défini par défaut à moins de redéfinir cette variable d'environnement.

Comment récupérer sa version de SQL*Plus ?

Pour vérifier ou récupérer sa version de SQL*Plus, tapez la commande suivante :

```
sqlplus -V
```

Comment afficher les propriétés liées à SQL*Plus ?

Il suffit de taper la commande suivante :

```
SQL> define
```

Le résultat sera le suivant (mais peut différer par rapport à votre version de SQL*Plus) :

```
DEFINE _DATE            = "30/03/09" (CHAR)
DEFINE _CONNECT_IDENTIFIER = "XE" (CHAR)
DEFINE _USER            = "HR" (CHAR)
DEFINE _PRIVILEGE       = "" (CHAR)
DEFINE _SQLPLUS_RELEASE = "1002000100" (CHAR)
DEFINE _EDITOR          = "Notepad" (CHAR)
DEFINE _O_VERSION       = "Oracle Database 10g
```

```
Express Edition Release 10.2.0.1.0 - Production" (CHAR)
DEFINE _O_RELEASE       = "1002000100" (CHAR)
```

- _DATE : date courante
- _CONNECT_IDENTIFIER : identifiant de la base de données
- _USER : utilisateur avec lequel vous êtes connecté
- _PRIVILEGE : privilège utilisé pour se connecter (AS SYSDBA, AS SYSOPER)
- _SQLPLUS_RELEASE : numéro de version
- _EDITOR : éditeur qui sera utilisé pour modifier les requêtes
- _O_VERSION : nom de version de la base de données
- _O_RELEASE : numéro de version de la base de données

Vous pouvez évidement afficher séparément toutes ces valeurs en utilisant toujours la commande define et en spécifiant à la suite, la variable à afficher. Comme par exemple :

```
SQL> define _CONNECT_IDENTIFIER
```

Dans le même ordre d'idée, pour définir une nouvelle valeur de variable tapez la commande

```
SQL> define variableName=newValue
```

Exemple :

```
SQL> define _EDITOR="c:/Notepad++/Notepad++.exe"
```

Certaines variables sont en lecture seule.

Comment afficher le schéma et le nom de l'instance dans le prompt ?

Uniquement valide à partir de la version 10G d'Oracle.

Modifiez le fichier $ORACLE_HOME/sqlplus/admin/glogin.sql (sous Unix) ou %oracle_home%/sqlplus/admin/glogin.sql (sous Windows) et ajoutez :

```
SQL> SET sqlprompt
"&_USER@&_CONNECT_IDENTIFIER> "
```

Comment changer le nom du prompt SQL> ?

On peut parfois vouloir changer le nom du prompt SQL> pour une nouvelle valeur.

Pour ce faire, tapez la commande suivante :

```
SQL> SET sqlprompt "nouvelleValeur"
```

Exemple :

```
SQL> SET sqlprompt "jsd03 SQL> "
```

On peut aussi combiner cette commande pour afficher un prompt du genre "username"@"SID". Par exemple :

```
SQL> SET sqlprompt &_USER@&_CONNECT_IDENTIFIER>
```

Notez que si vous changez d'utilisateur, le prompt restera comme vous l'avez défini précédemment.

Comment afficher l'heure dans le prompt SQL*Plus ?

Cette fonctionnalité permet de suivre en temps réel l'exécution d'un script.

Pour afficher l'heure en début de prompt, il suffit de taper la commande suivante ou de la mettre en début de script sql :

```
SQL> set time on
```

Résultat :

```
20:36:20 SQL>
```

Comment savoir avec quel compte je suis connecté à SQL*Plus ?

Utiliser la commande suivante :

```
SQL> SHOW USER
```

Résultat (si mon nom d'utilisateur est JSD03) :

```
USER IS "JSD03"
```

Comment activer / utiliser l'aide SQL*Plus ?

Pour utiliser l'aide sur une commande, il suffit de taper HELP suivi de la commande. Exemple :

```
SQL> help SHOW
```

ou encore

```
SQL> help host
```

Il se peut néanmoins que cette aide ne soit pas installée. Dans ce cas, allez dans le dossier $ORACLE_HOME/sqlplus/admin/help (sous Unix) ou %oracle_home%/sqlplus/admin/help (sous Windows) et lancez les commandes suivantes avec un compte SYS ou SYSTEM :

```
sqlplus SYS/password @helpdrop.sql --supprime les tables d'aide si elles existent déjà
sqlplus SYS/password @helpbld.sql --créée les tables d'aide
sqlplus SYS/password @helpus.sql --charge les données dans les tables d'aide
```

*Retrouvez la FAQ SQL*Plus en ligne : Lien129*

Excel et Fichiers Batch : Passage de Paramètres

L'objectif de ces quelques lignes est de mettre en lumière les possibilités de passage de paramètres à Excel depuis Batch.

1. Introduction

Cet article fait suite à un précédent concernant le passage de paramètres depuis Batch vers Access (Lien130). Pour aller plus loin dans la récupération de paramètres depuis batch, voici une suite proposée pour l'application Excel. Le but de cet article est de partir des exemples donnés avec la fonction **/cmd** pour multiplier les comportements à l'ouverture du classeur selon les paramètres qu'on lui passera. Elle permettra aux utilisateurs de gérer eux-mêmes des lancements automatiques totalement autonomes, sans une présence obligatoire devant la machine.

2. Avertissements

Pour cet article il est bon de définir une convention d'écriture. Le caractère " doit être scrupuleusement reproduit. L'utilisation de la touche F1 est vivement conseillée à tous les stades de l'utilisation d'EXCEL. L'amélioration constante de l'aide en fait un partenaire de choix dans l'apprentissage permanent d'EXCEL. Personnellement je ne peux m'en passer, ne serait-ce que pour mémoire.

3. Pré-requis

Bien qu'elle ne soit pas obligatoire, je recommande la lecture de l'article précédemment cité. Les fonctions VBA évoquées sont documentées dans l'aide en ligne ainsi que dans la FAQ à votre disposition ici : Lien131.

4. Exemple d'un besoin utilisateur

Plaçons-nous dans le cas d'un utilisateur travaillant dans la finance.
Il travaille avec des plates-formes boursières, qui lui délivrent des données tout au long de la journée.
Une journée type de ses activités avec une application Excel pourrait se dérouler de la façon suivante :
Il souhaite effectuer un traitement à **9h00**, une récupération des prix de la veille, à la fermeture de la bourse de Paris. Il effectue sur des données des calculs puis les met à disposition des autres équipes.
A **11h00**, il doit lancer d'autres traitements, pour la vérification des notations des entreprises, avec pour conséquence d'imposer des suspensions de validations, si certains critères ne sont pas respectés.
A **14h00**, un export sous Outlook doit être généré avec les données de la matinée.
A **17h30**, la valorisation des portefeuilles des gérants avant la fermeture de la bourse.
A **18h00**, les données doivent être stockées sur le serveur des équipes " **risques** " pour une contre valorisation.

Bref, plutôt que d'imposer une présence continue et un lancement manuel de chacune des procédures en cours de journée, à heures précises, on veut pouvoir automatiser ces lancements avec des outils simples à mettre en place.

5. Utilisation des tâches planifiées

Les exemples de créations de tâches planifiées sont déjà expliqués dans cette partie d'article : Lien132

6. Utilisation des fichiers batch

La fonction de base d'un fichier batch (qui s'exécute manuellement ou depuis un planificateur de tâches) est d'exécuter une suite de commandes DOS. L'article de Loufab indique les principales syntaxes auxquelles les utilisateurs ont affaire. Une ligne DOS se décompose basiquement en 2 parties :

- le logiciel A avec lequel on souhaite ouvrir un fichier B
- le répertoire du fichier B à exécuter avec le logiciel A

Le fichier batch porte l'extension .bat et s'exécute avec un double-clic sur le fichier.

6.1. Comment créer un fichier batch

Un fichier batch est un simple fichier texte. Il peut être créé et rempli depuis n'importe quel éditeur de texte. Le cours de Victor Laurie à ce sujet (Lien133) est très complet.

6.2. Outils de développement

Plusieurs outils sont possibles. Ma préférence va à NotePad++ : Lien134
D'autres outils sont téléchargeables à l'adresse suivante : Lien135

7. Ouverture d'Excel par un batch

Il est possible de lancer Excel en exécutant un fichier batch. La ligne de commande DOS basique est la suivante (ici pour Office 2003 - version 11):

```
"C:\Program Files\Microsoft
Office\Office11\EXCEL.EXE"
"C:\temp\classeur1.xls"
```

Cela lance Excel, et Excel se charge d'ouvrir le fichier classeur1.xls, situé dans le répertoire **C:\temp** de la machine. Il est possible d'ouvrir Excel en spécifiant de nombreux éléments, nommés commutateurs. La liste de ces commutateurs est disponible ici : Lien136

Exemple : si l'on souhaite ouvrir le fichier Classeur1 en lecture seule la ligne de commande sera la suivante :

```
"C:\Program Files\Microsoft
Office\Office11\EXCEL.EXE" /r
"C:\temp\classeur1.xls"
```

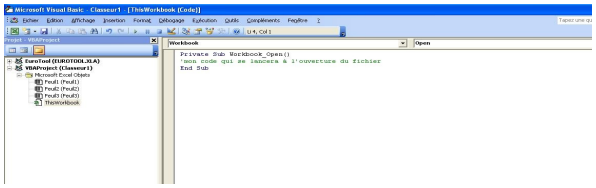
NB : Dans le cas où les noms de fichiers ou macros sont composés (existence d'espaces entre les mots), il est obligatoire de mettre les noms entre guillemets.
De plus, on constatera que les commutateurs sont placés avant le path du fichier que l'on souhaite ouvrir.

```
"C:\Program Files\Microsoft
Office\Office11\EXCEL.EXE"
"C:\temp\classeur1.xls" /e
```

Cette ligne permettant de lancer le fichier Classeur1 sans afficher l'écran d'affichage Excel. C'est la ligne de commande par défaut qui est lancée lorsqu'on double clique sur un fichier Excel.
Il n'existe pas de commutateur dédié au passage de paramètres. Toutefois, nous utiliserons un de ces commutateurs valides pour faire passer les informations que l'on souhaite à Excel.

8. Exécution automatique dans Excel

Il est possible de lancer automatiquement une fonction, procédure ou encore un formulaire à l'ouverture, en utilisant l'évènement **Open** du classeur.



Si l'on souhaite ouvrir un formulaire, il suffit de rédiger la ligne suivante.

```
Private Sub Workbook_Open()
NomduFormulaire.Show
End Sub
```

De la même façon, il est possible de lancer des macros à l'ouverture, en les lançant depuis l'évènement Workbook_Open. Tout un processus alliant code VBA/génération de graphique pourra être possible. L'avantage de ce système est de pouvoir effectuer à chaque ouverture la même série de processus sans qu'une intervention de l'utilisateur soit forcément nécessaire. L'inconvénient d'une manipulation automatique à l'ouverture est qu'un processus peut avoir lieu à un moment critique, modifiant des données qui ne devraient pas l'être, à cause d'un lancement trop hâtif ou trop tardif.

9. Passage de paramètres

On cherche donc à faire passer des informations à Excel afin de paramétrer son comportement. On verra dans le chapitre 8 les codes batch et VBA à mettre en œuvre pour influer sur le comportement du fichier à l'ouverture. En dehors d'une information stockée dans un fichier, une feuille ou dans le système, il est possible de faire passer directement les informations depuis la ligne de commande batch.
Cette information peut donc se résumer à une chaîne de caractères.

La fonction qui permet de récupérer le contenu de la ligne de commande Batch est basée sur la fonction **GetCommandLine**. Les détails à propos de cette fonction sont disponibles ici : Lien137.

```
Private Declare Function GetCommandLine Lib
"kernel32" Alias "GetCommandLineA" () As Long
```

Il existe deux versions principales de la fonction, une version Unicode (Lien138) et une version ANSI (Lien139) (principes de codage des caractères différents (Lien140)). Ici il s'agit de la version ANSI.
Cette fonction ne retourne qu'une variable de type Long. On passe donc par une fonction qui convertit cette variable en tableau de paramètres :

```
Private Declare Function lstrlen Lib "kernel32"
Alias "lstrlenA" (lpString As Any) As Long

Private Declare Function lstrcpy Lib "kernel32"
Alias "lstrcpyA" (lpString1 As Any, lpString2 As
Any) As Long

'fonction proposée par Tony Proctor sur le forum
public de Microsoft : microsoft.public.vb.winapi
Private Function GetCmd() As String
   Dim lpCmd As Long
   lpCmd = GetCommandLine()
   GetCmd = Space$(lstrlen(ByVal lpCmd))
   lstrcpy ByVal GetCmd, ByVal lpCmd
End Function
```

La fonction de départ comportera donc le code suivant

```
Private Sub Workbook_Open()
Dim monparam As Variant 'déclare une variable
    monparam = GetCmd 'affecte la valeur de la
ligne de commande
    If Not IsNull(monparam) Then 'si la variable
est nulle
        If Len(monparam) > 0 Then 'on s'assure
qu'il y a eu une ligne de commande passée
            'code de traitement
        End If
    End If
End Sub
```

La nouvelle étape est donc pour le développeur de mettre en place une nomenclature d'écriture, qui soit efficace et logique pour être naturelle, mais suffisamment complexe pour passer un grand nombre d'informations si nécessaire. L'information peut se limiter à un nom de formulaire à ouvrir, à une macro à lancer, à une procédure à exécuter… mais si d'un coup on veut passer les trois à la suite ? Il faut donc mettre en place un séparateur, qui délimitera chaque élément qu'on souhaitera faire passer en tant que paramètre.

10. Plusieurs comportements ?

Le comportement d'un fichier, quel drôle d'idée ! Et pourtant on peut trouver de bons exemples de ce qu'on pourrait définir comme comportement d'un fichier. Son autonomie, par exemple, où un fichier effectue une série

de processus seul, sans autre intervention de l'utilisateur que son ouverture initiale. Des menus qui apparaissent ou non selon le compte utilisateur, selon son niveau d'accréditation. De la même façon, on peut imaginer le grossissement des polices d'écriture pour un utilisateur qu'on aura identifié comme déficient visuel.

Il est possible de paramétrer toutes ces propriétés dans Excel. Le développeur doit donc préparer la gestion en amont. Cette gestion des individus passe souvent par une feuille des utilisateurs. Mais il serait pratique qu'en cas d'absence d'administrateur, un simple fichier puisse contenir toutes ces informations et qu'Excel puisse y lire ces informations (qui je suis ? ce que je vais faire ? quels sont mes droits ?).
Bien que non présentes nativement dans les fonctions Excel, l'auteur se propose d'utiliser les propriétés de traitement de chaînes de caractères pour imposer un comportement au fichier Excel dès son ouverture par le fichier batch.

11. Codes VBA et Batch

11.1. Idée générale

Il est possible de passer des paramètres depuis le fichier batch à la suite de la fonction **/cmd**.

Option	/cmd
Ligne de commande	"C:\Program Files\Microsoft Office\Office11\EXCEL.EXE" /cmd/parametres "C:\temp\classeur1.xls"
Description	Ouvre le fichier en passant le paramètre spécifié.

Ces paramètres sont ensuite récupérés par la fonction **GetCmd()** dans une fonction VBA, lancée à l'ouverture du fichier Excel.

11.2. Fonction initiale

La fonction qui permet de récupérer les paramètres passés dans le batch est de la forme :

```
Private Sub Workbook_Open()
Dim macmdline As Variant
Dim monparam As Variant 'déclare une variable

    macmdline = GetCmd 'affecte la valeur de la ligne de commande
    If Not IsNull(macmdline) Then 'si la variable est nulle
        If Len(macmdline) > 0 Then 'on s'assure qu'il y a eu une ligne de commande passée
            If InStr(macmdline, "/cmd") > 0 Then
                macmdline = Replace(macmdline, ThisWorkbook.FullName, "", , , vbTextCompare)
                monparam = Split(macmdline, "/cmd")
                Debug.Print Mid(monparam(1), 2, Len(monparam(1)) - 3)
            End If
        End If
    End If
End Sub
```

Les chapitres suivants montreront les exemples d'exécution de différents objets dans Excel. Y seront successivement indiqués les cas d'utilisation de ces codes, leur traitement dans l'évènement **WorkBook_Open()** et les résultats attendus.

11.3. Exemples

11.3.1. Activation d'une feuille

Ici, on souhaite activer une feuille du classeur, en passant le nom de la feuille dans le fichier batch.

La ligne de commande batch serait du type

```
"C:\Program Files\Microsoft Office\Office11\EXCEL.EXE" /cmd/feuil3 "C:\temp\classeur1.xls"
```

Le code VBA sera donc

```
Private Sub Workbook_Open()
Dim macmdline As Variant
Dim monparam As Variant 'déclare une variable

    macmdline = GetCmd 'affecte la valeur de la ligne de commande
    If Not IsNull(macmdline) Then 'si la variable est nulle
        If Len(macmdline) > 0 Then 'on s'assure qu'il y a eu une ligne de commande passée
            If InStr(macmdline, "/cmd") > 0 Then
                macmdline = Replace(macmdline, ThisWorkbook.FullName, "", , , vbTextCompare)
                monparam = Split(macmdline, "/cmd")
                ThisWorkbook.Worksheets(Mid(monparam(1), 2, Len(monparam(1)) - 3)).Activate
            End If
        End If
    End If
End Sub
```

On estimera que le nom de la feuille est bon. Une gestion d'erreur peut être intégrée dans l'évènement **Workbook_Open()**.

11.3.2. Lancement d'une macro

Ici, on souhaite lancer une macro dont on passera le nom en paramètre. La ligne de commande batch serait du type

```
"C:\Program Files\Microsoft Office\Office11\EXCEL.EXE" /cmd/MaFonction "C:\temp\classeur1.xls"
```

Le code VBA sera donc :

```
Private Sub Workbook_Open()
Dim macmdline As Variant
Dim monparam As Variant 'déclare une variable

    macmdline = GetCmd 'affecte la valeur de la ligne de commande
    If Not IsNull(macmdline) Then 'si la variable est nulle
        If Len(macmdline) > 0 Then 'on s'assure qu'il y a eu une ligne de commande passée
            If InStr(macmdline, "/cmd") > 0 Then
```

```
                macmdline = Replace(macmdline,
ThisWorkbook.FullName, "", , , vbTextCompare)
                monparam = Split(macmdline,
"/cmd")
                Application.Run Mid(monparam(1),
2, Len(monparam(1)) - 3)
            End If
        End If
    End If
End Sub
```

A noter qu'il est nécessaire que la fonction soit dans un **module** et pas dans une page de code liée à une feuille ou au classeur.

11.3.3. Lancement d'un formulaire

Ici, on souhaite ouvrir un formulaire dont on passera le nom en paramètre. La ligne de commande batch sera

```
"C:\Program Files\Microsoft
Office\Office11\EXCEL.EXE" /cmd/MonForm
"C:\temp\classeur1.xls"
```

Le code VBA sera donc :

```
Private Sub Workbook_Open()
Dim macmdline As Variant
Dim monparam As Variant 'déclare une variable

    macmdline = GetCmd 'affecte la valeur de la
ligne de commande
    If Not IsNull(macmdline) Then 'si la variable
est nulle
        If Len(macmdline) > 0 Then 'on s'assure
qu'il y a eu une ligne de commande passée
            If InStr(macmdline, "/cmd") > 0 Then
                macmdline = Replace(macmdline,
ThisWorkbook.FullName, "", , , vbTextCompare)
                monparam = Split(macmdline,
"/cmd")
                VBA.UserForms.Add(Mid(monparam(1)
, 2, Len(monparam(1)) - 3)).Show
            End If
        End If
    End If
End Sub
```

A noter qu'une autre manière d'ouvrir un formulaire au démarrage est de le spécifier.
On a donc pu voir les principales exécutions issues des paramètres.
Le paragraphe suivant a pour but de complexifier le raisonnement en passant plusieurs paramètres.

11.4. Combinaison de plusieurs paramètres de type d'exécution identique

L'idée est de séparer chacun des paramètres par un slash " / ", et de traiter tous les paramètres les uns à la suite des autres. La méthode employée par l'auteur ici n'est pas la plus usitée, le lecteur pourra bien évidemment trouver d'autres façons d'aborder le sujet.

11.4.1. Exemple : Affichage des paramètres passés à l'utilisateur

Ligne de commande

```
"C:\Program Files\Microsoft
Office\Office11\EXCEL.EXE" /cmd/A1/Jpcheck
"C:\temp\classeur1.xls"
```

On utilise la fonction **Split()** pour récupérer les différents paramètres.

```
Private Sub Workbook_Open()
Dim macmdline As Variant
Dim monparam As Variant 'déclare une variable
Dim ListeParam As Variant
Dim i As Integer
macmdline = GetCmd 'affecte la valeur de la ligne
de commande
If Not IsNull(macmdline) Then 'si la variable est
nulle
    If Len(macmdline) > 0 Then 'on s'assure qu'il
y a eu une ligne de commande passée
        If InStr(macmdline, "/cmd") > 0 Then
            macmdline = Replace(macmdline,
ThisWorkbook.FullName, "", , , vbTextCompare)
            monparam = Split(macmdline, "/cmd")
            ListeParam = Split(Mid(monparam(1),
2, Len(monparam(1)) - 3),"/")
            For i = 0 to UBound(ListeParam)
               MsgBox "Paramètre n°" & i+1 & " :
" & ListeParam(i)
            Next i
        End If
    End If
End If
End Sub
```

11.4.2. Exemple : exécution de plusieurs macros

Ligne de commande

```
"C:\Program Files\Microsoft
Office\Office11\EXCEL.EXE" /cmd/Macro1/Macro2
"C:\temp\classeur1.xls"
```

On utilise la fonction **Split()** pour récupérer les différents paramètres.

```
Private Sub Workbook_Open()
Dim macmdline As Variant
Dim monparam As Variant 'déclare une variable
Dim ListeParam As Variant
Dim i As Integer

macmdline = GetCmd 'affecte la valeur de la ligne
de commande
If Not IsNull(macmdline) Then 'si la variable est
nulle
    If Len(macmdline) > 0 Then 'on s'assure qu'il
y a eu une ligne de commande passée
        If InStr(macmdline, "/cmd") > 0 Then
            macmdline = Replace(macmdline,
ThisWorkbook.FullName, "", , , vbTextCompare)
            monparam = Split(macmdline, "/cmd")
            ListeParam = Split(Mid(monparam(1),
2, Len(monparam(1)) - 3),"/")
            For i = 0 to UBound(ListeParam)
               Application.Run ListeParam(i)
            Next i
        End If
    End If
End If
End Sub
```

11.5. Plus loin dans la combinaison des paramètres

11.5.1. Paramètres de types d'exécution différents

En partant de l'exemple donné plus haut, on peut pousser le raisonnement de l'exécution des paramètres selon leur type. On appliquera donc une nomenclature pour se retrouver. Chaque paramètre sera précédé d'un identifiant mono-caractère alphanumérique.
Par souci de compréhension, on prendra comme exemples :

Identifiant alphanumérique	Objet associé	Fonction de lancement VBA	Exemple
F	Formulaire (Form)	VBA.UserForms.Add().Show	FMonForm
M	Macro	Application.Run	MMaMacro
S	Feuille (Sheet)	Worksheets().Activate	SMaFeuille
Etc.			

On lancera donc la ligne de commande suivante :

```
"C:\Program Files\Microsoft Office\Office11\EXCEL.EXE" /cmd/mMacro1 "C:\temp\classeur1.xls"
```

Le code VBA sera le suivant :

```
Private Sub Workbook_Open()
Dim macmdline As Variant
Dim monparam As Variant 'déclare une variable
Dim ListeParam As Variant
Dim i As Integer

macmdline = GetCmd 'affecte la valeur de la ligne de commande
If Not IsNull(macmdline) Then 'si la variable est nulle
    If Len(macmdline) > 0 Then 'on s'assure qu'il y a eu une ligne de commande passée
        If InStr(macmdline, "/cmd") > 0 Then
            macmdline = Replace(macmdline, ThisWorkbook.FullName, "", , , vbTextCompare)
            monparam = Split(macmdline, "/cmd")
            Select Case UCase(Left(Mid(monparam(1), 2, Len(monparam(1)) - 3),1))
                Case "F":
                    VBA.UserForms.Add(Mid(monparam(1), 2, Len(monparam(1)) - 3)).Show
                Case "M":
                    Application.Run Mid(monparam(1), 2, Len(monparam(1)) - 3)
                Case "S":
                    Worksheets(Mid(monparam(1), 2, Len(monparam(1)) - 3)).Activate
            End Select
        End If
    End If
End If
End Sub
```

11.5.2. Exécution de plusieurs paramètres de types différents

De la même façon que pour la gestion de plusieurs paramètres de même type, on adapte les derniers exemples pour obtenir cette ligne de commande :

```
"C:\Program Files\Microsoft Office\Office11\EXCEL.EXE" /cmd/mMacro1/fMonForm "C:\temp\classeur1.xls"
```

On souhaite donc d'abord lancer la macro MaMacro puis ouvrir le formulaire MonForm. Le code VBA sera donc le suivant :

```
Private Sub Workbook_Open()
Dim macmdline As Variant
Dim monparam As Variant 'déclare une variable
Dim ListeParam As Variant
Dim i As Integer

macmdline = GetCmd 'affecte la valeur de la ligne de commande
If Not IsNull(macmdline) Then 'si la variable est nulle
    If Len(macmdline) > 0 Then 'on s'assure qu'il y a eu une ligne de commande passée
        If InStr(macmdline, "/cmd") > 0 Then
            macmdline = Replace(macmdline, ThisWorkbook.FullName, "", , , vbTextCompare)
            monparam = Split(macmdline, "/cmd")
            ListeParam = Split(Mid(monparam(1), 2, Len(monparam(1)) - 3),"/")
            For i = 0 to UBound(ListeParam)
                Select Case UCase(Left(ListeParam(i),1))
                    Case "F":
                        VBA.UserForms.Add(Mid(ListeParam(i),2)).Show
                    Case "M":
                        Application.Run Mid(ListeParam(i),2)
                    Case "S":
                        Worksheets(Mid(ListeParam(i),2)).Activate
                End Select
            Next i
        End If
    End If
End If
End Sub
```

12. Conclusion

Il est donc possible pour un utilisateur de générer des fichiers batch qui passent des paramètres dans Excel, et que ceux-ci soient lus et compris par l'application. La nomenclature de programmation proposée dans cet article n'est qu'un exemple, libre à chacun d'adapter les idées de traitement des paramètres passés par le fichier batch.

Retrouvez l'article de Jean-Philippe André en ligne : Lien141

Gestion des images dans le ruban Access

Découvrir les différentes possibilités de gestion des images du ruban Access.
Intégrer les images à votre application.
Utiliser des images transparentes.

1. Introduction

Dans la **version 2007** d'Office, les barres de menus ont été remplacées par **le ruban**.
La programmation de ce ruban se fait en **XML**.

Les images affichées sur les contrôles du ruban peuvent être soit :
- des images standards intégrées à Office.
- des images personnalisées.

Nous allons voir les différentes techniques pour gérer ces images.

Pour créer plus facilement le XML de vos rubans, utilisez l'assistant ruban (Lien142).

Pour mieux comprendre la programmation des rubans, consultez ces tutoriels :
- Programmez et personnalisez le ruban de vos applications Access 2007 (Lien143).
- La personnalisation du ruban sous Excel 2007 (Lien144).
- Personnalisation du ruban : Les fonctions d'appel CallBacks (Lien145).
- Comment personnaliser le Ruban de Word 2007 (Lien146).

2. Contrôles de ruban utilisant des images

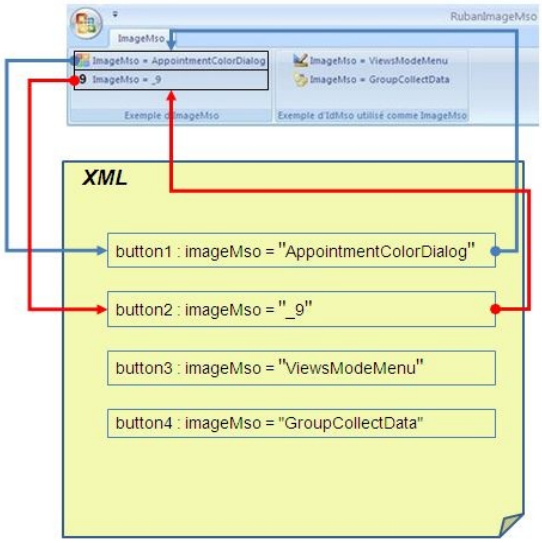
Les types de contrôles du ruban qui affichent une image sont :

Type de contrôle	Description
button	Bouton.
comboBox	Liste déroulante avec une zone de texte d'édition.
dropDown	Liste déroulante sans zone de texte d'édition.
dynamicMenu	Menu dynamique
editBox	Zone de texte d'édition.
gallery	Galerie de contrôles.
group	Groupe contenant les contrôles.
toggleButton	Bouton bascule.

3. Utilisation d'une image standard : attribut ImageMso

De nombreuses images standards sont disponibles : elles sont installées avec Office.

Pour utiliser une de ces images, définissez tout simplement l'attribut XML **imageMso** du contrôle.



La liste des valeurs possibles pour **imageMso** est disponible dans un fichier Excel sur le site de microsoft : Liste des imageMso (Lien147).
Notez qu'il est également possible d'utiliser une des valeurs de l'attribut **idMso** proposées dans la Liste des idMso (Lien148).

Voici un exemple de code XML utilisant l'attribut **imageMso** :

Utilisation de imageMso

```
<customUI
xmlns="http://schemas.microsoft.com/office/2006/0
1/customui">
    <ribbon startFromScratch="true">
        <tabs>
            <tab id="tab1" label="ImageMso">
                <group id="group1"
label="Exemple d'ImageMso">
                    <button id="button1"
imageMso="AppointmentColorDialog" label="ImageMso
= AppointmentColorDialog"/>
                    <button id="button2"
imageMso="_9" label="ImageMso = _9"/>
                </group>
                <group id="group2"
label="Exemple d'IdMso utilisé comme ImageMso">
                    <button label="ImageMso =
AppointmentColorDialog" imageMso="ViewsModeMenu"
id="button3"/>
                    <button id="button4"
label="ImageMso = GroupCollectData"
imageMso="GroupCollectData"/>
                </group>
            </tab>
        </tabs>
    </ribbon>
</customUI>
```

Ce code XML suffit à afficher les images.
Aucun code VBA n'est nécessaire.

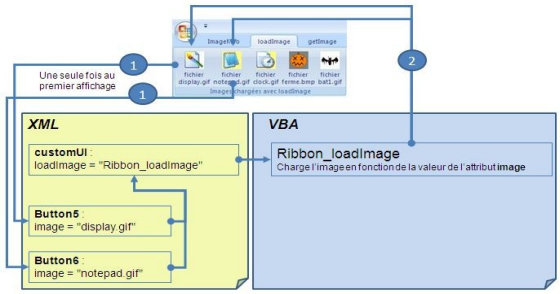
Voici le ruban obtenu :

Utilisez l'assistant ruban (Lien142) pour trouver plus facilement l'*imageMso* dont vous avez besoin.

4. Chargement initial : callback loadImage

Un callback est un appel à une procédure VBA.
Si vous ne savez pas comment fonctionnent les callback, je vous invite à consulter les tutoriels cités en introduction.
loadImage est un callback associé à l'élément **CustomUI**.

La procédure ainsi définie sera **exécutée une seule fois** pour chaque contrôle dont l'attribut **image** est renseigné.
Elle sera ignorée en cours d'utilisation de l'application, même après un appel à **Invalidate** ou **InvalidateControl**.



Cette attribut **loadImage** ne peut donc pas être utilisé pour des images dynamiques qui doivent changer en fonction d'événements dans la base de données.

Voici le début du code XML d'un ruban utilisant l'attribut **loadImage** :

Attribut loadImage
```
<customUI
xmlns="http://schemas.microsoft.com/office/2006/0
1/customui" loadImage="Ribbon_loadImage">
```

La procédure VBA **Ribbon_loadImage** sera exécutée au premier affichage de chaque contrôle de ruban dont l'attribut **image** est renseigné.

Voici comment l'écrire :

Procédure Ribbon_loadImage
```
Sub Ribbon_loadImage(imageId As String, ByRef
image)


End Sub
```

La procédure nous envoie un paramètre **imageId** : c'est en fait la valeur de l'attribut **image** de l'élément dont l'image doit être chargée.
En retour, on doit modifier le paramètre **image** qui attend un objet de type **IpictureDisp**.

Un objet de type **IPictureDisp** peut être simplement créé avec la fonction VBA **LoadPicture**.

Voici un exemple de code XML utilisant l'attribut **loadImage** :

Utilisation de loadImage
```
<customUI
xmlns="http://schemas.microsoft.com/office/2006/0
1/customui" loadImage="Ribbon_loadImage">
    <ribbon startFromScratch="true">
        <tabs>
            <tab id="tab2" label="loadImage">
                <group id="group3"
label="Images chargées avec loadImage">
                    <button id="button5"
label="fichier display.gif" image="display.gif"
size="large"/>
                    <button id="button6"
label="fichier notepad.gif" image="notepad.gif"
size="large"/>
                    <button id="button7"
label="fichier clock.gif" image="clock.gif"
size="large"/>
                    <button id="button8"
label="fichier ferme.bmp" image="ferme.bmp"
size="large"/>
                    <button id="button9"
label="fichier bat1.gif" image="bat1.gif"
size="large"/>
                </group>
            </tab>
        </tabs>
    </ribbon>
</customUI>
```

Avec, dans un module VBA, le code de la fonction de chargement des images :

Procédure de chargement des images
```
Sub Ribbon_loadImage(imageId As String, ByRef
image)
Set image = LoadPicture(CurrentProject.Path &
"\images\" & imageId)
End Sub
```

Notez que dans cet exemple, les images sont des fichiers stockés dans un sous-répertoire nommé **images**.

Voici le ruban obtenu :

La transparence des images n'a pas été conservée (les images gif utilisées sont des images avec transparence).
Nous verrons plus loin dans cet article comment conserver la transparence des images chargées.

Retrouvez la suite de l'article de Thierry Gasperment en ligne : Lien149

IRT : un raytracer interactif - Partie 4

Quatrième partie de la série de tutoriels dédiée au raytracing (voir ici pour la table des matières : Lien150). Suréchantillonnage et bounding box.

1. Objectifs

Maintenant qu'il est possible de créer une scène simple, il est nécessaire d'améliorer sa qualité et de l'accélérer. Pour cela, il va falloir :

- lancer plusieurs rayons pour un seul pixel de l'image,
- ne lancer un rayon que sur la zone utile (définie par la BoundingBox de la scène)

2. Utilisation d'une Bounding Box

Chaque objet occupe une certaine place dans la scène. En les conjuguant, il est possible d'obtenir la taille globale de la scène. Pour cela, on va ajouter aux primitives de retourner leur BoundingBox :

bounding_box.h
```
struct BoundingBox
{
  /// coin droit
  Point3df corner1;
  /// coin gauche
  Point3df corner2;
};
```

sphere.cpp
```
BoundingBox Sphere::getBoundingBox() const
{
  BoundingBox bb;

  bb.corner1 = center - radius;
  bb.corner2 = center + radius;

  return bb;
}
```

Maintenant, il est possible de modifier le code d'ajout d'une primitive, avec **bb** la BoundingBox propre à la scène :

simple_scene.cpp
```
SimpleScene::SimpleScene()
  :primitives(), lights()
{
  bb.corner1 =
std::numeric_limits<float>::max();
  bb.corner2 =
std::numeric_limits<float>::min();
}

unsigned long
SimpleScene::addPrimitive(Primitive* primitive)
{
  if(std::find(primitives.begin(),
primitives.end(), primitive) != primitives.end())
    throw std::out_of_range("Primitive already
added");

  BoundingBox primitive_bb = primitive-
>getBoundingBox();
  min(bb.corner1, primitive_bb.corner1);
  max(bb.corner2, primitive_bb.corner2);

  primitives.push_back(primitive);
  return primitives.size() - 1;
}
```

Dans le cas présent, les fonctions min et max ont été modifiées pour permettre de calculer le vecteur minimum ou maximum de deux vecteurs.

On ajoute aussi la possibilité de recalculer la BoundingBox (même si cela ne sera pas forcément utilisé tout de suite).

simple_scene.cpp
```
const BoundingBox&
SimpleScene::getBoundingBox() const
{
  return bb;
}

void SimpleScene::computeBoundingBox()
{
  std::vector<Primitive*>::iterator it =
primitives.begin();
  bb = (*it)->getBoundingBox();

  for(++it; it != primitives.end(); ++it)
  {
    BoundingBox bb_bis = (*it)-
>getBoundingBox();
    min(bb.corner1, bb_bis.corner1);
    max(bb.corner2, bb_bis.corner2);
  }

  return;
}
```

Maintenant, cette BoundingBox permet de savoir si un rayon doit être tiré ou non. Pour cela, une fonction spécifique **mustShoot()** va être développée, et elle appellera une nouvelle méthode de la BoundngBox.

raytracer.cpp
```
bool Raytracer::mustShoot(const Ray& ray, const
BoundingBox& bb) const
{
      for ( int i = 0; i < 3; ++i )
```

```
    {
      if (ray.direction()(i) < 0)
           {
                  if (ray.origin()(i) <
bb.corner1(i))
        {
          return false;
        }
      }
        else if (ray.origin()(i) > bb.corner2(i))
      {
        return false;
      }
    }
  }
```

bounding_box.h

```
    bool getEntryExitDistances(const Ray& ray,
DataType& tnear, DataType& tfar) const
    {
      tfar = std::numeric_limits<float>::max();
      tnear =
std::numeric_limits<float>::epsilon();
      for(int i = 0; i < 3; ++i)
      {
        float pos = ray.origin()(i) + tfar *
ray.direction()(i);
        float pos_near = ray.origin()(i) + tnear
* ray.direction()(i);
        if(ray.direction()(i) < 0)
        {
          if(pos < corner1(i))
          {
            tfar = (corner1(i) - ray.origin()(i))
/ ray.direction()(i);
          }
          else if(pos > corner2(i))
          {
            tfar =
std::numeric_limits<float>::epsilon();
          }
          if(pos_near > corner2(i))
          {
            tnear = (corner2(i) - ray.origin()
(i)) / ray.direction()(i);
          }
          else if(pos_near < corner1(i))
          {
            tnear =
std::numeric_limits<float>::max();
          }
        }
        else if (ray.direction()(i) > 0)
        {
          if(pos > corner2(i))
          {
            tfar = (corner2(i) - ray.origin()(i))
/ ray.direction()(i);
          }
          else if(pos < corner1(i))
          {
            tfar =
std::numeric_limits<float>::epsilon();
          }
          if(pos_near < corner1(i))
          {
            tnear = (corner1(i) - ray.origin()
(i)) / ray.direction()(i);
          }
          else if(pos_near > corner2(i))
          {
            tnear =
std::numeric_limits<float>::max();
          }
        }

        if (tnear > tfar)
        {
          return false;
        }
      }

      return true;
    }
```

Avec cette BoundingBox, il va être possible d'implémenter une primitive de type parallélépipède droit, aligné sur les axes. Cela va permettre d'ajouter un peu de diversité dans les scènes.

3. Suréchantillonnage

Un rayon unique échantillonne l'image en un point. En revanche, l'oeil humain ne fonctionne pas comme ça. Il moyenne l'information reçue sur une surface en un seul "pixel" qui sera envoyé au cerveau. L'objectif du suréchantillonnage est de permettre d'approcher ce moyennage sur la surface, en approchant l'intégrale de tous les rayons possibles par la somme de quelques-uns.

L'impression donnée pour un seul rayon est un crênelage, aussi appelé aliasing. En fait, cet effet est dû à un échantillonnage trop faible par rapport à la fréquence maximale de l'image, qui est infinie dans le cas d'une image réelle. L'effet est d'autant plus visible sur une image avec beaucoup de détails.

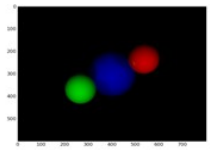
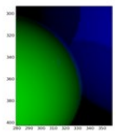


Image originale



Zoom sur l'image, permettant de visualiser le crênelage

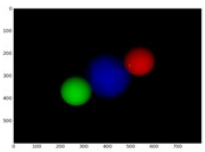
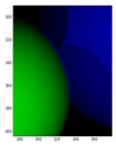


Image suréchantillonnée (2,2)



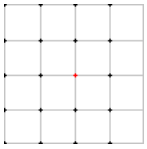
Zoom sur l'image

3.1. Echantillonnage uniforme

L'échantillonnage uniforme est la première méthode présentée. Au lieu de travailler sur une image de taille (w, h), on travaille sur une image de taille plus importante (m*w, n*h), puis chaque bloc de taille (m, n) sera moyenné en un seul pixel.

Logiquement, la performance est divisée par m*n, en revanche l'image devient plus naturelle.

Au lieu de n'envoyer qu'un seul rayon sur une maille, on va en envoyer plusieurs comme indiqué dans la figure suivante :



Echantillonnage uniforme (4,4)

Les résultats pour un facteur 2 et un facteur 4 sont les suivants :

Oversampling (2,2)

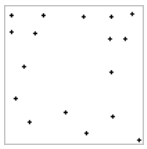
Oversampling (4,4)

Un oversampling faible améliore le résultat, mais le crênelage reste visible. Avec un oversampling plus important, l'effet disparaît.

3.2. Echantillonnage aléatoire

L'oversampling uniforme a l'avantage de la facilité, mais cela n'empêche qu'il reste un aliasing. On va maintenant envisager d'autres méthodes d'échantillonnage, donc l'échantillonnage aléatoire.

Le premier example sera de choisir un certain nombre de points au hasard autour du rayon central.



Echantillonnage aléatoire (4,4)

Les résultats pour un facteur 2 et un facteur 4 sont les suivants :

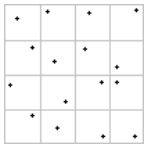
Oversampling (2,2)

Oversampling (4,4)

L'inconvénient de cette solution est qu'il est possible que les points tirés aléatoirement laissent de grosses surfaces vides, comme c'est le cas dans l'image précédente, ce qui veut dire que des détails peuvent être manqués.

3.3. Echantillonnage jittered

Pour pallier cet inconvénient, on va repartir sur l'échantillonnage uniforme, mais cette fois-ci, on va tirer un point dans chaque sous-case.



Echantillonnage jittered (4,4)

Les résultats pour un facteur 2 et un facteur 4 sont les suivants :

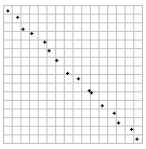
Oversampling (2,2)

Oversampling (4,4)

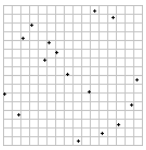
3.4. Echantillonnage n-rooks

L'échantillonnage précédent est très performant. En revanche, si on regarde ce qui se passe sur une dimension, on se rend compte que la répartition n'est pas idéale. En effet, on peut obtenir des trous, tout comme c'était le cas en 2D.

Pour résoudre ce problème, on va découper les deux dimensions en n morceaux et on va placer un échantillon au hasard dans les cases diagonales. Par la suite, on effectue une permutation des lignes. On obtiendra toujours sur chaque ligne et colonne un échantillon par case. En revanche, on va retrouver des trous en 2D.



Placement d'un échantillon dans les cases diagonales



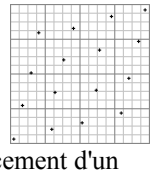
Permutation selon ligne ou colonne

Oversampling (2,2)

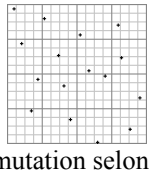
Oversampling (4,4)

3.5. Echantillonnage multi-jittered

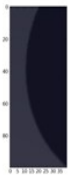
Ce mode d'échantillonnage reprend les avantages des deux méthodes précédentes. Un échantillon par ligne ou colonne, mais aussi un échantillon par case en 2D.



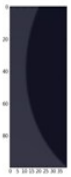
Placement d'un échantillon par sous-case



Permutation selon les lignes et les colonnes



Oversampling (2,2)



Oversampling (4,4)

Le résultat, tout comme pour le cas précédent, ne peut pas se différencier dans ce cas précis d'un échantillonnage jittered. Leurs avantages sont surtout importants s'il est nécessaire d'utiliser un échantillonneur de même type sur des lumières ou des ombres, ce qui n'est pas fait pour le moment.

3.6. Comparaison

Voici deux images regroupant les comparaisons. De gauche à droite, on a l'échantillonnage uniforme, puis aléatoire, jittered, n-rooks et multi-jittered.

Comparaison des différentes techniques pour un oversampling (2,2)

Comparaison des différentes techniques pour un oversampling (4,4)

3.7. Implémentation

L'implémentation a été effectuée à l'aide de templates C++. Le raytracer est devenu une structure template où le paramètre est le type de sampler à utiliser.

Dans la méthode **draw()**, au lieu d'appeler directement **computeColor()**, on passe par le sampler (instancié comme une variable de la structure) qui possède lui aussi cette méthode. Dans cette méthode, on parcourt les différents échantillons vus précédemment pour créer plusieurs rayons, et la couleur résultante sera la couleur moyenne de ces différents rayons.

Pour les détails de chaque sampler, je vous invite à regarder le code source sur Launchpad (Lien151).

D'autres méthodes d'échantillonnage existent, comme la méthode Quasi Monte Carlo (Lien152).

4. Résultats

Comme prévu, le suréchantillonnage ralentit la construction de l'image. On constate aussi qu'à partir d'un échantillonnage jittered, les résultats semblent visuellement meilleurs. Le surcoût d'utilisation d'un système plus complexe que l'échantillonnage uniforme est faible et donc ces échantillonnages peuvent être envisagés. En revanche, pour chaque pixel, on utilise la même répartition. Il est possible de complexifier les échantillonneurs pour générer à chaque appel de nouveaux échantillons.

En ce qui concerne la BoundingBox, les résultats sont flagrants dans le cas test. Il est clair que si l'oeil se situe dans la scène, ces tests auront un impact négatif. Dans le cas de la scène des précédents tutoriels, avec 3 sphères et 3 lumières, on gagne 10% de temps de calcul sous Windows (.5 s à .45 s).

5. Conclusion

Maintenant que le concept de BoundingBox a été introduit, et vu la perte de vitesse, il est intéressant de tester des structures accélératrices.

Il est aussi à noter que selon les cas (*i.e.* selon le découpage des fonctions, ou simplement le fait de retourner une variable par référence constante, même si cela ne se produit qu'une seule fois pendant un calcul), le compilateur de Microsoft peut s'emmêler les pinceaux. Dans les meilleurs cas, la vitesse d'exécution passe sous la seconde. Les raisons exactes de ce comportement me sont inconnues...

Retrouvez l'article de Matthieu Brucher en ligne : Lien153

Etude de cas : dp state en c++

Le pattern state, ou patron état, est l'un des patterns les plus utilisés. Dans sa définition initiale, il est très simple, mais comme beaucoup de design pattern, son implémentation concrète peut varier beaucoup selon le contexte. Je vous propose ici d'analyser plusieurs implémentations possibles.

1. Introduction

1.1. Généralités

Le *pattern State* (patron Etat) est un *behavioural design pattern* (patron de conception de type comportemental).

Les design patterns fournissent des modèles théoriques qui permettent de résoudre des problèmes récurrents. Ils ont été pensés de façon à répondre à un maximum de problèmes, de façon à ce qu'ils puissent être appliqués avec succès dans les contextes les plus différents possibles. De fait, ils sont utilisés régulièrement par les développeurs qui font de l'objet, et on les retrouve donc dans de nombreux logiciels.

Concernant les design pattern, je pense qu'il est inutile de les apprendre par coeur. D'autant plus qu'à force de les utiliser on finit par les connaître sans avoir à faire l'effort de les apprendre. En revanche je crois qu'il est important

de les connaître, juste pour savoir qu'ils existent et savoir où trouver des informations lorsque nous voulons les appliquer. La compréhension de ces design pattern est également un excellent exercice qui permet de comprendre un certain nombre de problématiques relatives au paradigme objet et d'en appréhender des solutions efficaces, générales et élégantes.

L'UML propose tout un tas de diagrammes, qui permettent de représenter différentes étapes de la réalisation d'un logiciel, et de différentes façons. Pour comprendre cet article, vous aurez juste besoin de savoir lire un diagramme de classe et savoir ce qu'est un diagramme d'état.
Ici je n'utiliserai que des diagrammes très simplifiés, nettoyés de tout ce qui n'a pas de rapport direct avec le sujet dont traite le diagramme.

1.2. Définition du pattern state

L'idée de ce pattern est d'obtenir une classe, que j'appellerai *contexte* (context), qui aura un comportement circonstancié à un *état* courant, c'est-à-dire un comportement différent selon son *état*.
. Nous allons donc créer une classe abstraite qui définit l'interface publique des états. En gros, nous allons y mettre les fonctions du *contexte* dont le comportement peut varier. C'est l'*état abstrait* (abstract state).
. Ensuite il faut implémenter les *états concrets*, qui hériteront de l'*état abstrait*
. . Après on crée un pointeur *état courant*, en variable membre du *contexte*.
. Lorsqu'on souhaite que le *contexte* change d'état, il suffit de modifier l'*état courant*.

Une implémentation d'un pattern state est considérée comme meilleure si elle respecte les critères suivants : . Le LSP (Lien154) est respecté (entre l'*état abstrait* et les *états concrets*). Cela facilite la manipulation des états et peut éviter des situations complexes et/ou ambiguës.
. Les *états concrets* ne possèdent pas de données. Les données doivent se trouver dans le *contexte* ou dans l'*état abstrait*.

1.3. Le diagramme d'état

Le diagramme d'état est généralement moins connu que le diagramme de classes, je vais donc en dire un mot. Il sert à représenter un automate d'état fini (ou graphe). C'est extrêmement simple en fait : il y a des états - les nœuds du graphe - et des évènements (ou transitions)- les arcs du graphe - qui permettent de passer d'un état à l'autre.

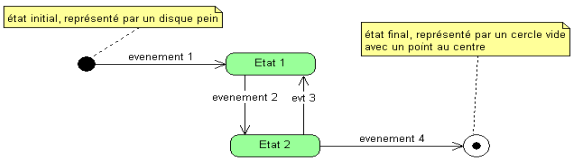


diagramme d'état

Prenons un exemple didactique simplissime et sans rapport avec le développement logiciel (l'UML est conçu pour représenter tous types de problèmes, pas uniquement de programmation). Prenons donc l'exemple d'un graveur de CD. Il pourra se trouver dans trois états différents : **stand-by** (il ne fait rien), **lecture**, et **enregistrement**. Pour passer de l'un à l'autre de ces états, on a 3 événements : **play**, **stop**, et **record**, qui correspondent par exemple à l'activation par l'utilisateur de la touche de la télécommande correspondante.

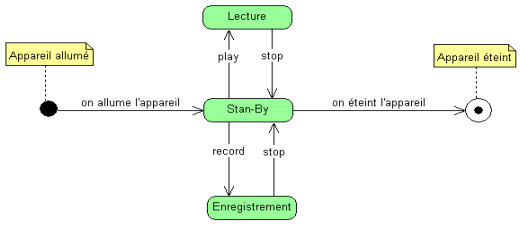


diagramme d'état

Retrouvez la suite de l'article de r0d en ligne : Lien155

Les dernières news

Des binaires Qt à disposition !

Bonjour,

Vous avez déjà essayé de compiler Qt. Beaucoup y arrivent, mais cette opération prend un certain temps. D'autres n'y arrivent pas, et abandonnent le framework.

Aujourd'hui, nous vous présentons une solution : les binaires prêts à l'emploi. Il vous suffit de les télécharger, d'y appliquer un patch, et c'est prêt !
À vous les joies de Qt !

Actuellement, voici les quelques binaires disponibles : la version 4.5.2, la dernière stable, pour Visual Studio 2008, et la toute dernière Technology Preview de la version 4.6, pour Visual Studio 2008 et GCC 4.4.

Tous les paquets incluent la documentation, les démonstrations, les exemples, et les sources : pas besoin de télécharger d'autres fichiers, tout ce dont vous pourriez avoir besoin est là !

Vous pouvez les télécharger par FTP, par HTTP ou par Torrent, au choix, sur la page dédiée : Les binaires Qt (Lien156).

Si vous avez des questions, des problèmes avec ces binaires, postez directement sur le forum, en précisant le tag [Bin] dans le titre de votre message : nous répondrons à vos questions.

Vous avez des réflexions sur ces binaires ?
Des commentaires ?
Un plug-in dont vous avez besoin n'est pas disponible ?
Vous aimeriez voir votre plateforme supportée ?

Commentez cette news en ligne : Lien157

Declarative UI : le futur du développement d'IHM

Salut,

Je pense que la plupart d'entre vous ont entendu parler de QML ou de *declarative UI*, mais sans vraiment trop savoir de quoi ça parle.

Je vais essayer de résumer.

L'architecture "widget" est une architecture robuste et structurée qui a fait ses preuves. Mais cette architecture est peu flexible, peu adaptée aux composants non rectangulaires et aux animations et qui fournit donc des IHM très statiques.

Seulement, le besoin change. Et à quoi devrait ressembler une IHM dans le futur pour vous ? Sûrement à des IHM vivantes, avec une plus grande interaction avec l'utilisateur, avec des petits effets visuels. Les meilleurs exemples sont les téléphones mobiles et leurs interfaces de plus en plus attractives.

Il suffit de regarder l'interface de l'iPhone, Androïd, HTC et compagnie. Rien à voir avec les logiciels d'aujourd'hui. Tout est en mouvement : on zoome, on fait des rotations... On exploite le stylet, le doigt, la luminosité ambiante, l'orientation de l'appareil...
L'application de visualisation de photo de l'iPhone est un très bon exemple.
Et bien sûr, la mode commence à s'étendre sur les PC.

En gros voilà ***declarative UI*** qui est un projet R&D de Nokia sur le développement de ces IHM, et QML, qui est un langage pour exprimer de manière lisible (par un humain) ces IHM.

De plus, Developpez.com met des binaires précompilés de Qt à votre disposition. Dans lesquels vous trouverez une version compilée pour visual 2008 SP1 de la branche *kinetic-declarative-ui* et donc voir à quoi cela va ressembler et bien sûr de jouer avec.

Des binaires Qt à disposition ! (Lien158)

Et vous que pensez-vous de tout cela ? Vous êtes pour ? Contre ? Des remarques positives ou négatives?

Commentez cette news en ligne : Lien159

Des nouvelles sur Qt 4.6, suite à la sortie de la première beta !

Bonjour,

Les premières versions beta de Qt 4.6 et de Qt Creator 1.3 viennent de sortir, chez Nokia ! Elles sont d'ores et déjà disponibles au téléchargement (Lien160) sur leur site.

Citation de **Lars Knoll, directeur R&D Qt chez Nokia**

```
Community developers working alongside us have
provided extensive technical feedback to support
the enhancement and fine-tuning of both Qt and Qt
Creator. Since we opened Qt for contributions in
May 2009, there have been over 400 community
submissions to Qt and Qt Creator. This is very
rewarding and endorses our decision to work in a
more transparent, open way.
```

Ainsi, cette nouvelle version de Qt se basera encore plus sur les contributions du public. Elle s'axera sur plusieurs sujets-clé :

- Le support de nouvelles plateformes : Windows 7, Mac OS X 10.6, mais aussi Symbian (Qt s'intègre d'ailleurs au framework de la plateforme S60). En plus du support dans Qt Creator de la cible Symbian, des optimisations pour Maemo 6 (basée sur Linux/X11), et l'annonce du port de Qt pour Maemo6, ceci fait de Qt un standard, non seulement pour le bureau, mais aussi pour le mobile.

- L'amélioration de l'expérience utilisateur, avec un nouveau framework d'animation, des effets graphiques avancés, ainsi que le support des gestes et du multi-touch.

- De meilleures performances pour des modules-clé de Qt, comme un nouveau backend pour QtScript basé sur JavaScriptCore (le moteur JavaScript de Webkit), et le support d'OpenVG pour l'affichage vectoriel en deux dimensions. Cette version est donc la plus performante jusqu'à aujourd'hui !

La version finale de Qt 4.6 est attendue pour le quatrième trimestre de cette année, c'est-à-dire dans très peu de temps.

En même temps, un nouveau blog dédié à Qt a été ouvert au début de ce mois : The Qt Blog (Lien161). Son contenu sera principalement les nouveautés attendues pour les prochaines versions, des détails sur l'utilisation de Qt, des informations sur les événements Qt, ainsi que d'autres sujets susceptibles d'intéresser la communauté Qt.

Source : communiqué de presse.

Qu'attendez-vous de cette nouvelle version ? Les améliorations proposées vont-elles vous servir ? Par exemple, Qt n'est-il pas assez performant actuellement ?

Participez-vous au développement de Qt par le biais de Gitorious ? Est-ce utile pour Qt ? Les développeurs ont-ils plus de retours sur le framework de cette manière ?

Commentez cette news en ligne : Lien162

Petit résumé du C4, rendez-vous des développeurs Mac et iPhone

La 4ième édition du C4 est maintenant terminée.

Nous allons tenter ici de résumer ce qui y a été abordé

Tout d'abord, Jonathan 'Wolf' Rentzsch l'organisateur du C4 a ouvert la conférence par ce qui pourrait être perçu comme une provocation pour certains.
Il a proposé à son auditoire d'enterrer AppleScript et d'utiliser à la place JSTalk.

On vous avait déjà présenté dans le passé JSCocoa (Lien163)

Et bien sachez que JSTalk se repose sur JSCocoa,

Mais voilà. Tout le problème est là : certains développeurs sont quelque peu allergiques à la syntaxe de JavaScript et trouvent bien plus lisible un script AppleScript.

Plus tard, dans la conférence, Jonathan Rentzsch a présenté un invité surprise, Marshall Culpepper.

Marshall Culpepper est le développeur en chef (lead developer) d'un produit qui a fait couler beaucoup d'encre lors de sa présentation au C4 : Appcelerator Titanium

Ce produit vous promet quelque part monts et merveilles : Vous codez votre application en Javascript, et elle se retrouve convertie en application pour Mac, Windows, Linux. Et même iPhone et Android.

Suite à cette annonce on a vraiment vu deux clans s'affronter : les développeurs Web qui étaient heureux de pouvoir capitaliser sur les langages et technologies qu'ils connaissaient déjà pour développer des applications natives sous Mac, iPhone et même Windows, Linux et Android.
Et puis les développeurs desktop, ceux qui utilisent Cocoa et Objective-C pour le développement de leur Application, et AppleScript pour les scripts, qui étaient horrifiés à l'idée de voir ainsi JavaScript venir marcher sur leur platebande.

Dave Dribin a fait une présentation sur les tests unitaires, expliquant comment s'y prendre sous Mac, les outils qui sont à notre disposition, etc.

Louis Gerbarg a proposé un Blitz Talk, une présentation de 5 minutes, où il expliquait pourquoi il vous faut vous intéresser à être un ingénieur en compilateur. Et d'expliquer pourquoi, comment et combien il est important, et facile, de contribuer aux projets que sont Clang et LLVM.

Vous pouvez retrouver les slides de sa présentation ici : Lien164

Ken Aspeslagh, de Ecamm Network, a donné 14 conseils pour réussir, dont en
voici quelques uns :

pour réussir votre projet, il vous faut

- travailler à temps plein dessus
- Automatiser le plus possible les tâches
- engager une personne pour le support
- Diversifier vos produits
- Eviter les mauvais projets
- ne pas copier du code trouvé sur internet

Et mon préféré : Ignorer les conseils

Les conseils de Ken pour réussir vos projets : Lien165

Encore un autre sujet fort intéressant : Why Hack your Mac ? (Lien166)

où l'auteur explique comment comprendre les rapports de crash, comment en tirer parti, utiliser les outils tels que Instruments.app (basé sur DTrace, un projet de Sun) pour terminer sur comment tirer profit de vos exploits.

Kevin Mitchell a également tenu une courte présentation de PyObjC, un projet qui permet de faire le lien entre Python et Objective-C. (Lien167)

Il y a également eu une présentation de Coda
http://www.panic.com/coda/ (Lien168) que l'on peut décrire comme une fenêtre de développement d'application web. Le mieux est encore d'aller voir sur le site de Coda. Il vous expliqueront bien mieux que moi ce que c'est.

Un petit bijou a également été présenté lors de ce C4 : CorePlot

Qu'est-ce que CorePlot ?
Un framework opensource pour réaliser des graphes. Framework valable aussi bien pour Mac que pour iPhone. C'est mon coup de coeur. (Lien169)

Et vous, que pensez-vous de l'utilisation de JavaScript au lieu d'AppleScript ?

Ou de développer des applications en JavaScript pour les rendre ensuite natives Mac et iPhone ? Est-ce que cela vous paraît être une bonne idée ?

Sources :
Le site consacré au C4 : Lien170
AppCelerator Itanium : Lien171
JSTalk : Lien172

Interview Patrick Geiller, créateur de JSCocoa : Lien173
PyObjC : Lien174
Le blog de Dribin : Lien175
Le blog de Louis Gerbarg : Lien176
Clang : Lien177
LLVM : Lien178

Commentez cette news en ligne : Lien179

Les métriques (McCabe, Halstead) et l'index de maintenabilité

La complexité de code a une influence directe sur la qualité et le coût d'un logiciel. Elle impacte sur la durée de vie et l'exploitation d'un logiciel, et plus particulièrement sur son taux de défauts, sa testabilité et sa maintenabilité. Une bonne compréhension et maîtrise de la complexité d'un code permet de développer un logiciel de meilleure qualité.

Dans cet article, Klaus LAMBERTZ, introduit différentes métriques (mesures) permettant d'évaluer la qualité du code, ainsi que la notion d'Index de Maintenabilité relative au coût de maintenance d'un logiciel.
Dans un second temps, il nous présente quelques outils permettant de mettre en pratique les notions introduites et d'établir une évaluation qualitative d'un code source / logiciel.

La logique veut que plus un document est complexe, plus il sera difficile à comprendre et à analyser, et donc à corriger. En terme de développement de logiciel, la complexité d'une application a un impact direct sur le pourcentage d'erreurs et la robustesse du code, puisque la complexité du logiciel se reflète sur sa difficulté à être testé.
Pour garantir un logiciel de qualité tout en assurant des coûts de test et de maintenance faibles, la complexité d'un logiciel devrait être mesurée dès le début du codage. Ainsi lorsque les valeurs recommandées sont dépassées, le développeur peut intervenir rapidement.
Pour quantifier la complexité d'un logiciel, on se sert de métriques.
Les métriques peuvent être classées en trois catégories :

- celles mesurant le processus de développement ;
- celles mesurant des ressources ;
- et celles de l'évaluation du produit logiciel.

Les métriques de produits mesurent les qualités du logiciel. Parmi ces métriques, on distingue les métriques traditionnelles et les métriques orientées objet.
Les métriques orientées objet prennent en considération les relations entre éléments de programme (classes, méthodes). Les métriques traditionnelles se divisent en deux groupes : les métriques mesurant la taille et la complexité, et les métriques mesurant la structure du logiciel. Les métriques mesurant taille et complexité les plus connues sont les métriques de lignes de code ainsi que les métriques de Halstead. Les métriques mesurant la structure d'un logiciel comme la complexité cyclomatique de McCabe se basent sur des organigrammes de traitement ou des structures de classe.
Cet article décrit les métriques traditionnelles (les métriques des lignes de code, le nombre cyclomatique de McCabe et les métriques de Halstead) ainsi que l'index de maintenabilité.

Les métriques des Lignes de code

Pour quantifier la complexité d'un logiciel, les mesures les plus utilisées sont les lignes de code (LOC acronyme de « lines of code ») puisqu'elles sont simples, faciles à comprendre et à compter. Cependant ces mesures ne prennent pas en compte le contenu d'intelligence et la disposition du code.
On peut distinguer les types de métriques de lignes de code suivants :

- LOCphy: nombre de lignes physiques (total des lignes des fichiers source) ;
- LOCpro: nombre de lignes de programme (déclarations, définitions, directives, et code ;
- LOCcom: nombre de lignes de commentaire ;
- LOCbl: nombre de lignes vides (en anglais number of blank lines).

Quelles sont les limites acceptables ?
La longueur des fonctions devrait être de 4 à 40 lignes de programme. Une définition de fonction contient au moins un prototype, une ligne de code, et une paire d'accolades, qui font 4 lignes.
En règle générale, une fonction plus grande que 40 lignes de programme doit pouvoir s'écrire en plusieurs fonctions plus simples. A chaque règles son exception, les fonctions contenant un état de sélection avec beaucoup de branches ne peuvent pas être décomposées en fonctions plus petites sans réduire la lisibilité.
La longueur d'un fichier devrait contenir entre 4 et 400 lignes de programme, ce qui équivaut déjà à un fichier possédant entre 10 et 40 fonctions. Un fichier de 4 lignes de programme correspond à une seule fonction de 4 lignes, c'est la plus petite entité qui peut raisonnablement occuper un fichier source complet.
Un fichier de plus de 400 lignes de programme est généralement trop long pour être compris en totalité.
Pour aider à sa compréhension, on estime qu'au minimum 30 % et maximum 75 % d'un fichier devrait être commenté.
Si moins d'un tiers du fichier est commenté, le fichier est soit très trivial, soit pauvrement expliqué. Si plus de 75% du fichier est commenté, le fichier n'est plus un programme, mais un document.
Seul un fichier « header » déroge à cette règle car lorsqu'il est correctement commenté, le pourcentage de commentaires peut parfois dépasser 75%.

Le nombre cyclomatique de Mc Cabe : v(G)

La complexité Cyclomatique (complexité de McCabe), introduite par Thomas McCabe en 1976, est le calcul le

plus largement répandu des métriques statiques. Conçue dans le but d'être indépendante du langage, la métrique de McCabe indique le nombre de chemins linéaires indépendants dans un module de programme et représente finalement la complexité des flux de données.
Il correspond au nombre de branches conditionnelles dans l'organigramme d'un programme.
Le nombre cyclomatique évalue le nombre de chemins d'exécution dans la fonction et ainsi donne une indication sur l'effort nécessaire pour les tests du logiciel.
Pour un programme qui consiste en seulement des états séquentiels, la valeur pour v(G) est 1.
Plus le nombre cyclomatique est grand, plus il y aura de chemins d'exécution dans la fonction, et plus elle sera difficile à comprendre et à tester. Du fait que le nombre cyclomatique décrive la complexité du flux de contrôle, il est évident que les modules et les fonctions ayant un nombre cyclomatique élevé auront besoin de plus de cas de tests que celles avec une complexité de McCabe plus basse. Le principe de base est que chaque fonction devrait avoir un nombre de cas de tests au moins égal au nombre cyclomatique, pour que tous les chemins soient couverts au moins une fois.
Les constructions de langage suivantes incrémentent le nombre cyclomatique : *if (...), for (...), while (...), case ...:, catch (...), &&, ||, ?, #if, #ifdef, #ifndef, #elif*. Le calcul commence toujours avec la valeur 1. A ceci on ajoute le nombre de nouvelles branches.
Quelles sont les limites acceptables pour le nombre cyclomatique v(G) ?
Une fonction devrait avoir une nombre cyclomatique inférieur à 15. Si une fonction a plus que 15 chemins d'exécution elle est difficile à comprendre et à tester. Pour un fichier le nombre cyclomatique ne devrait pas dépasser 100.

Les Métriques de Halstead

Les métriques de complexité de Halstead qui procurent une mesure quantitative de complexité ont été introduites par l'américain Maurice Halstead. Elles sont basées sur l'interprétation du code comme une séquence de marqueurs, classifiés comme un opérateur ou une opérande.
Toutes les métriques de Halstead sont dérivées du nombre d'opérateurs et d'opérandes :

- nombre total des opérateurs uniques (n1)
- nombre total des opérateurs (N1)
- nombre total des opérandes uniques (n2)
- nombre total des opérandes (N2)

Sur la base de ces chiffres on calcule :

- La *Longueur du programme* (N) : N = N1 + N2.
- La *Taille du vocabulaire* (n) : n = n1 + n2

A partir de là, on obtient le *Volume du Programme* (V) en multipliant la taille du vocabulaire par le logarithme 2 : V = N * log2(n)
Le volume d'une fonction devrait être compris entre 20 et 1000. Le volume d'une fonction, d'une ligne et sans paramètre, qui n'est pas vide est d'environ 20. Une fonction avec un volume supérieur à 1000 comporte probablement trop de choses. Le volume d'un fichier devrait être au minimum à 100 et au maximum à 8000.
Le Niveau de difficulté (D) ou propension d'erreurs du programme est proportionnel au nombre d'opérateurs uniques (n1) dans le programme et dépend également du nombre total d'opérandes (N2) et du nombre d'opérandes uniques (n2). Si les mêmes opérandes sont utilisés plusieurs fois dans le programme, il est plus enclin aux erreurs.
D = (n1 / 2) * (N2 / n2)
Le *Niveau de programme* (L) est l'inverse du Niveau de difficulté. Un programme de bas niveau est plus enclin aux erreurs qu'un programme de haut niveau.
L = 1 / D
L'Effort à l'implémentation (E) est proportionnel au volume (V) et au niveau de difficulté (D). Cette métrique est obtenu par la formule suivante :
E = V * D
Halstead a découvert que diviser l'effort par 18 donne une approximation pour le *Temps pour implémenter* (T) un programme en secondes.
T = E / 18
Il est même possible d'obtenir le « nombre de bugs fournis » (B) qui est une estimation du nombre d'erreurs dans le programme. Cette valeur donne une indication pour le nombre d'erreurs qui devraient être trouvées lors du test du logiciel. Le « nombre de bugs fournis » est calculé selon la formule suivante:
B = (E (2/3)) / 3000
Des expériences ont montré qu'un fichier source écrit en C ou C++ contient malheureusement très souvent plus d'erreurs que B ne suggère.

L'Index de Maintenabilité (MI)

L'index de maintenabilité permet d'évaluer lorsque le coût de la correction du logiciel est plus élevé que sa réécriture. Il est conseillé de réécrire des parties du logiciel avec une mauvaise maintenabilité afin d'économiser du temps et donc de l'argent lors de la maintenance.
L'index de maintenabilité est calculé à partir des résultats de mesures de lignes de code, des métriques de Mc Cabe et des métriques de Halstead.
Il y a trois variantes de l'index de maintenabilité :
* la maintenabilité calculée sans les commentaires (MIwoc, Maintainability Index without comments):

```
MIwoc = 171 - 5.2 * ln(aveV) -0.23 *
aveG -16.2 * ln(aveLOC)
```

Sachant que:
aveV = valeur moyenne du volume d'Halstead Volume (V) par module
aveG = valeur moyenne de la complexité cyclomatique v(G) par module
aveLOC = nombre moyen de lignes de code (LOCphy) par module
* la maintenabilité concernant des commentaires (MIcw, Maintainability Index comment weight) :

```
MIcw = 50 * sin(sqrt(2.4 * perCM))
```

* l'Index de maintenabilité (MI, Maintainability Index) est la somme de deux précédents :

```
MI = MIwoc + MIcw
```

La valeur du MI indique la difficulté (ou facilité) de

maintenir une application.
Si la valeur est 85 ou plus la maintenabilité est bonne. Une valeur entre 65 et 85 indique une maintenabilité modérée. Si l´index de maintenabilité est inférieur à 65, la maintenance est difficile. Il est donc préférable de réécrire des parties « mauvaises » du code.

Conclusions

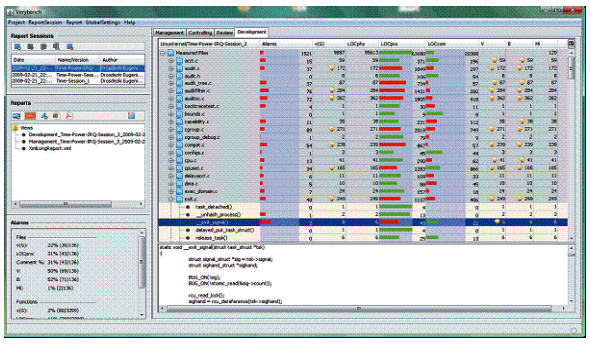
Les métriques de lignes de code, le nombre cyclomatique de McCabe, les métriques de Halstead et l'index de maintenabilité sont des moyens efficaces pour mesurer la complexité, la qualité et la maintenabilité d'un logiciel. Ces métriques servent également à localiser les modules difficiles à tester et à maintenir. Des actions correctives peuvent alors être enclenchées pour corriger une complexité trop élevée plutôt que de garder des modules susceptibles d'être chers en maintenance.

Mesure de complexité avec Testwell CMT++ et Testwell CMTJava

Les outils Testwell CMT++ et Testwell /CMTJava permettent de mesurer la complexité du code. Grâce à l'analyse des fichiers non-préprocessés les outils Testwell sont extrêmement rapides. Même des grands projets peuvent être analysés en quelques minutes.
L'interface « Verybench » de Testwell CMT++/CMTJava permet d'obtenir différentes présentations de métriques pour développeurs, réviseurs, testeurs et le management.
En plus des sorties textes, HTML, XML et CSV il est possible d'obtenir des rapports en PDF avec une présentation très claire qui permet de voir rapidement les défaillances de qualité.

Vue « développement »

Les développeurs ont une connaissance détaillée de leurs codes sources et veulent mesurer la qualité de leur travail pour y ajouter des améliorations au niveau des codes.
Ainsi, la vue de développement dispose d'une représentation en arbre qui montre tous les fichiers analysés et les fonctions d'une session de report ainsi que les résultats adéquats des mesures.



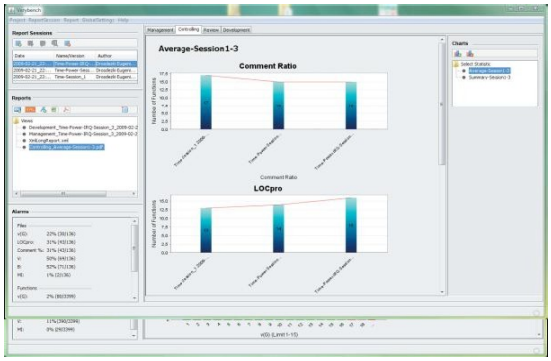
Dans la représentation en arbre de cette vue, toutes les valeurs mesurées qui ne se trouvent pas à l'intérieur des limites prétextées, sont caractérisées par des petites ampoules lumineuses. La colonne d´alerte et les métriques LOCpro et LOCcom se visualisent par des barres d'avancement.
Lors de l'ouverture de la vue développement, la représentation en arbre s´affiche pour le niveau « fichier ». En cliquant sur un fichier précis les fonctions de ce fichier sont présentées avec plus de détails. Au choix d'un fichier ou fonction, le code source adéquat est également affiché.

Vue « revue »

La vue „revue" aide à identifier les parties «critiques » ayant besoin d´un contrôle et d'une attention particulière.
Cette vue montre seulement les fichiers qui se trouvent « à l'extérieur » des limites d'alertes.

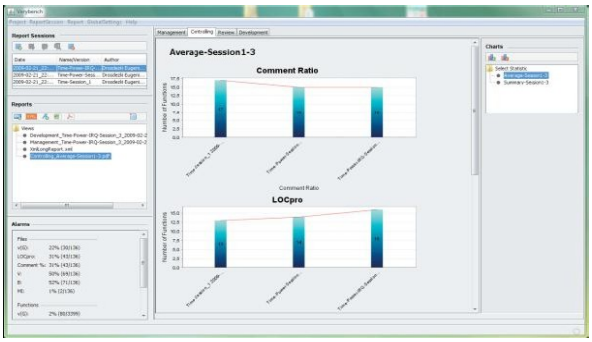
Vue « management »

La vue management donne un aperçu rapide indiquant le stade de développement et la qualité du projet sans détails gênants.
Les moyennes des métriques de tous les fichiers sources d'une session sont présentées dans un diagramme de Kiviat.
De plus des diagrammes-barres sont établis. Ce diagramme permet de voir, pour la métrique choisie, le nombre de fonctions comprises pour chaque valeur.



Vue « audit »

La vue audit permet de produire des statistiques sur plusieurs sessions et de poursuivre le développement temporel de la complexité d'un projet.
On peut obtenir des statistiques pour des valeurs sommaires et moyennes d'une session ainsi que des valeurs d'une fonction déterminée.



Retrouvez l'article de Klaus Lambertz en ligne : Lien180

Mise en place du caddying

Nous mettons en place une nouvelle étape de notre adaptation à Scrum. Adaptation c'est bien l'un des mots-clés de Scrum, avec inspection et transparence. Eh bien l'inspection de notre manière de travailler nous indique que, depuis le début du passage à Scrum, nous négligeons le développement des compétences techniques, et nous ne valorisons pas les gens qui d'eux-mêmes deviennent très compétents sur certains points.
Sans aller dans le détail, ce manque de développement de certaines compétences techniques se traduit par divers *gaspillages* (des réécritures de code, du code trop compliqué, du réapprentissage d'erreurs déjà connues). Tous ces points pourraient certainement s'améliorer si nous prenions plus le temps de développer nos compétences.

Dans un domaine moins technique, on retrouve ce genre de problème dans le développement des compétences des ScrumMasters et des Product Owners. Une fois leur certification obtenue, ils sont "lâchés dans la nature", sans accompagnement particulier. Peu s'auto-formeront ensuite, ne serait-ce qu'en lisant quelques blogs.
Pourquoi cet état de choses ? D'une part il y a une certaine pression pour produire le plus de points possibles à chaque sprint, en raison de l'urgence de sortir de nouvelles versions de logiciels. Cette pression ne laisse guère de place à autre chose que du travail à court terme. D'autre part il y a peut-être une sorte de culture d'entreprise, où le développement de compétences semble limité à quelques passionnés qui parviendront à se perfectionner même dans l'environnement le moins favorable ! La même culture favorise et valorise les rôles de héros et de pompiers au détriment des rôles d'enseignant, formateur, mentor...
Pour les managers qui progressent dans la compréhension du Lean et de l'agilité, le développement des compétences devient par contre une préoccupation essentielle, et ils devraient créer des conditions qui lui sont favorables. C'est ce que nous essayons de faire depuis peu. Une première piste est l'organisation de journées "labo" comme je le mentionnais dans un billet précédent (Lien181). Nous expérimentons depuis la rentrée une autre piste, qui est l'organisation d'un programme de caddying (Lien182). Les journées "labo" s'intercalent entre les sprints, tandis que le caddying se déroulera en parallèle des sprints. Nos journées "labo" visent surtout à favoriser le travail collectif et casser les silos, et ont pour vocation de produire des résultats à l'échelle de la journée. Le caddying vise à obtenir des résultats à l'échelle de plusieurs mois.
Pourquoi caddying ? Par analogie avec le caddie qui accompagne le joueur de golf. Le caddie n'est pas un simple porteur de clubs, c'est un joueur expérimenté qui a déjà joué sur le terrain en question, et qui donne des conseils pertinents au joueur (même aux joueurs de très haut niveau). Toutefois, ce n'est pas le caddie qui tape dans la balle... C'est cette analogie que nous utilisons pour expliquer le programme : les caddies sont des gens expérimentés qui donnent à leurs collègues des moyens pour les faire progresser, sous la forme d'entretiens, questions-réponses, ateliers, conseils de lecture, pair-programming, co-présentations... Vu toutes ces possibilités de mise en place concrète, je préfère éviter le terme coaching (Lien183) au profit de caddying.
Voici comment nous procédons concrètement pour démarrer cette nouvelle activité :
- nous avons identifié des thèmes dans lesquels nous devons développer nos compétences, ainsi que des caddies potentiels. Les caddies pourront libérer 1 jour par semaine pour leur activité (travail avec les joueurs, avec les autres caddies, veille technologique dans leur domaine). Nous aurons entre cinq et huit caddies. Chaque caddie aura environ trois joueurs.
- nous avons fait appel au volontariat pour trouver des joueurs. Les joueurs consacreront environ 2h par semaine au travail avec leur caddie (discussion en tête-à-tête, pair-programming, ateliers...). Les joueurs sont volontaires et choisissent librement leur caddie.
- nous avons également recueilli les demandes de joueurs potentiels, afin d'identifier de nouveaux thèmes et de nouveaux caddies. Etant donné la part importante du volontariat dans ce programme, il faut en effet arriver à faire coïncider l'offre de services des caddies et les demandes des joueurs, tout en restant à peu près dans le périmètre des thèmes importants pour l'entreprise.
- l'organisation sera flexible : on peut cesser d'être caddie (si on n'a plus envie, si on est trop pris par un projet...), un caddie peut être le joueur d'un autre caddie, un joueur devenu expérimenté peut devenir caddie.
- il y a un responsable du programme (en l'occurrence votre serviteur) qui coordonne l'action des caddies, identifie les thèmes (en se basant sur des dysfonctionnements connus, ainsi que sur des orientations stratégiques)...
- les caddies seront encouragés à publier des retours d'expérience sur leur action, ou encore sur leur domaine de prédilection... L'idée étant que leur action devrait servir à générer de la connaissance (thème que j'explorais dans ce précédent billet (Lien184)).
- les caddies ne sont pas supposés répondre instantanément à tous les problèmes soumis par des joueurs. Au contraire ils doivent si possible conduire leur joueur à élaborer lui-même une solution. Pour cela ils devront être créatifs et pédagogues. Par exemple ils pourront imaginer des ateliers, des défis pour leurs joueurs : il me semble en effet intéressant de proposer d'autres activités que la simple lecture de livres ou d'articles.
Tout cela est un point de départ, rien n'est gravé dans le marbre, et nous nous adapterons au fur et à mesure du déroulement de cette expérience.
Les paires (caddy, joueur) commencent leurs travaux début octobre. J'espère pouvoir publier des retours d'expérience prochainement !

Retrouvez ce billet sur le blog de Bruno Orsier : Lien185

Gestion de projet : Vers les méthodes agiles

Cet ouvrage rassemble plus de dix années d'expérience en gestion de projet informatique, et compare les méthodologies traditionnelles - qui définissent à l'avance les besoins et organisent les activités à réaliser, leur séquencement, les rôles et les livrables à produire - aux méthodes agiles.

Ces dernières prennent le contre-pied des méthodes prédictives en évitant une définition trop précoce et figée des besoins ; elles ont montré une surprenante efficacité en pariant sur la souplesse des équipes. Ce guide aidera les chefs de projet, déjà familiarisés avec les méthodes traditionnelles de conduite de projet et attirés par les méthodes dites " agiles " ou débutant dans le métier, à évaluer et améliorer leurs compétences en gestion de projet. Il guidera également architectes, analystes, développeurs ou testeurs dans la conduite de leurs projets, ainsi que tous les clients ou experts métier non informaticiens souhaitant appréhender rapidement les enjeux et la répartition des rôles au sein d'un projet.

Critique du livre par Arnaud Lemercier

"Gestion de projet : vers les méthodes agiles" est un livre décrivant les rôles du chef de projet (au sens large) et les différentes façons d'appréhender les problématiques liées aux projets informatiques.
Il montre l'apport des méthodes agiles (Scrum, XP, DSDM, Crystal...) par rapport aux méthodes classiques et compare leurs avantages et leurs faiblesses.

Ce livre est illustré par de nombreux témoignages de spécialistes qui répondent à des problématiques courantes dans la vie d'un projet. Un véritable retour d'expérience qui ne manquera pas de vous faire évoluer.
Véronique Messager Rota nous amène à réfléchir aux sources d'échecs d'un projet et définit des bonnes pratiques organisationnelles, humaines et techniques.
Elle détaille également les différentes étapes d'un projet et identifie leur but et les problèmes récurrents.
Pourquoi le client n'arrive t-il pas à exprimer son besoin ?
Pourquoi seulement 30% de l'application sont réellement utilisés ?
En quoi le pilotage par les tests est un facteur de réduction des coûts ?
Comment estimer la charge d'un projet ?
Quelle doit-être la charge allouée à l'estimation d'un projet ?

Bien que ce livre soit tourné vers la découverte des méthodes agiles, certaines notions de pilotage de projet sont parfois complexes pour un débutant.

Pour conclure, ce livre est un très bon outil.
Il répond à beaucoup de questions que se posent les chefs de projet informatique et constitue un recueil d'expériences très enrichissant. Attention tout de même à ne pas tomber dans l'apologie aveugle des méthodes agiles.
Ayez toujours en tête que ce ne sont que des bonnes pratiques, que vous devez les utiliser si vous en sentez l'utilité, et non parce que vous pensez que c'est à la mode.

Retrouvez cette critique de livre sur la page livres Conception : Lien186

CUDA approfondi

Dans l'article précédent, vous avez pu vous familiariser avec CUDA. Nous n'avons que survolé la matière, afin de vous donner goût au calcul par GPU. Cette fois-ci, nous allons un peu plus approfondir certains points, tout en restant dans le runtime, sans nous aventurer dans le driver (pour le moment).
Notez que je partirai du fait que vous avez lu et compris l'article précédent !
Introduction à CUDA (Lien187)

1. Introduction

Nous avons étudié quelques principes généraux de l'architecture de CUDA, afin de bien le comprendre. Nous avons ensuite étudié ses différentes briques. Nous nous étions arrêtés sur le *runtime*, composant que nous allons continuer dans cet article. Ce composant définit quelques variables et types, qui seront bientôt vus. Nous n'avons qu'entr'aperçu la manière d'écrire en mémoire : vous verrez qu'il y a aussi moyen de définir des tableaux en 2D et en 3D, entre autres. Quand vous avez appris à appeler un *kernel*, je ne vous ai pas dévoilé la consistance du quatrième paramètre de configuration à l'exécution : les flux. Cela sera réglé.

Pour faciliter le développement, NVIDIA met à disposition quelques librairies : CuBLAS et CuFFT. Vous apprendrez à les utiliser, uniquement en C. La communauté GPGPU (Lien188) met à notre disposition une autre librairie, CUDPP, que nous allons aussi aborder. Cette librairie en utilise une autre, CUTIL. Thrust, une dernière librairie, est l'équivalent de la STL pour CUDA.

Mais ces librairies sont de gros mastodontes, parfois, et vous devrez peut-être effectuer de plus petites opérations mathématiques ou atomiques, qui s'adaptent à plus de situations. Cependant, ces calculs ne sont pas synchrones : il faut donc les synchroniser avec votre application.

Et n'oublions pas que les GPU sont avant tout prévus pour la 3D : CUDA peut interopérer avec les API majeures du marché, OpenGL, ainsi que DirectX 9 et 10.

De nos jours, les entreprises ont besoin d'applications fiables, qui ne plantent pas à la moindre erreur : CUDA permet aussi de vérifier les retours des fonctions.

Aussi, les systèmes multi-GPU sont de plus en plus fréquents, et CUDA ne permet de n'en utiliser qu'un, en mode « pilote automatique ». Nous verrons qu'il y a moyen de le choisir. Vous pouvez même utiliser plusieurs GPU en même temps !

2. Variables intégrées

Ces variables sont disponibles dans tous les *kernels*, et indiquent les options de configuration lors du lancement du *kernel* ainsi que la position du thread dans la grille.

Nom de la variable	Type	Utilité
gridDim	dim3	Dimensions de la grille
blockIdx	uint3	Index du bloc dans la grille
blockDim	dim3	Dimensions du bloc
threadIdx	uint3	Index du thread dans le bloc
warpSize	int	Taille du warp

Cependant, il y a quelques restrictions.

- On ne peut demander l'adresse de ces variables ;
- On ne peut leur affecter une nouvelle valeur.

Voici un exemple d'utilisation.

```
printf("Dimensions de la grille : ( x = %d ; y = %d ; z= %d )", gridDim.x, gridDim.y, gridDim.z);
printf("Dimensions du bloc : ( x = %d ; y = %d ; z= %d )", blockDim.x, blockDim.y, blockDim.z);
printf("Identifiant de ce bloc : ( x = %d ; y = %d ; z = %d )", blockIdx.x, blockIdx.y, blockIdx.z);
printf("Identifiant de ce thread : ( x = %d ; y = %d ; z = %d )", threadIdx.x, threadIdx.y, threadIdx.z);
printf("Taille du warp : %d", warpSize);
```

3. Types de vecteurs intégrés

Voici la liste exhaustive de ces types.

- char1
- uchar1
- char2
- uchar2
- char3
- uchar3
- char4
- uchar4
- short1
- ushort1
- short2
- ushort2
- short3
- ushort3

- short4
- ushort4
- int1
- uint1
- int2
- uint2
- int3
- uint3
- int4
- uint4,
- long1
- ulong1
- long2
- ulong2
- long3
- ulong3
- long4
- ulong4
- float1
- float2
- float3
- float4
- double2

Ces types sont dérivés des types du C char, unsigned char, short, unsigned short, int, unsigned int, long, unsigned long, float et double, comme indiqué par leur nom.

Il s'agit de structures. Ils représentent des vecteurs de *une* à *quatre* dimensions, accessibles via X, Y, Z et W, dans l'ordre.

On les initialise à l'aide des fonctions TYPE make_TYPE (TYPE x, TYPE y, TYPE z, TYPE w), comme le montre ce court exemple.

```
float1 floatVectorOne    =    make_float1(1.0);
float2 floatVectorTwo    =    make_float2(1.0,
4.2);
float3 floatVectorThree  =    make_float3(1.0,
4.2, 1475.41742);
float4 floatVectorFour   =    make_float4(1.0,
4.2, 1475.41742, 0.000000000004105);
```

On peut accéder aux composantes de ces vecteurs très simplement.

```
floatVectorOne.x;

floatVectorTwo.x;
floatVectorTwo.y;

floatVectorThree.x;
floatVectorThree.y;
floatVectorThree.z;

floatVectorFour.x;
floatVectorFour.y;
floatVectorFour.z;
floatVectorFour.w;
```

Il existe aussi le type dim3, qui sert à spécifier les dimensions (d'une grille, d'un bloc ...). Il s'agit d'un vecteur à trois dimensions d'entiers. Toutes les composantes non initialisées valent 1.

Nous avons appris dans l'article précédent à les utiliser et à les déclarer. Ce petit bout de code vous rappellera les méthodes à employer.

```
dim3 dim     (5);
dim3 dimgrid (1, 1);
dim3 dimblock(5, 24, 240);
```

4. Textures

Les textures ne sont pas des types semblables aux autres. Elles se déclarent à la manière d'un *kernel*, mais, avant d'entrer dans ces subtilités, il faut déjà savoir ce qu'est une texture, dans le langage CUDA.

CUDA supporte une partie du matériel de *texturing* des GPU, utilisé pour les opérations sur la mémoire des textures. Cette mémoire est lue par le *kernel* quand une fonction de *texture fetching* (littéralement : aller chercher une texture) est appelée.

Dans cette section, quand je parlerai de tableaux, il s'agira de tableaux CUDA. Vous verrez l'importance de ceci à la prochaine section, concernant la mémoire.

4.1. Références de texture

Le premier paramètre passé à une telle fonction s'appelle une *référence de texture*. Une référence de texture précise quels endroits de la mémoire seront utilisés pour cette texture. On doit la lier à une région de la mémoire, la *texture*, avant de pouvoir l'utiliser. Plusieurs références peuvent pointer sur une même texture, ou sur des textures qui se superposent.

Une référence possède plusieurs attributs. Parmi ceux-ci, la *dimensionnalité*, qui spécifie la manière d'accéder à la texture : *via* un tableau à une dimension (avec une seule *coordonnée de texture*) ; à deux dimensions (avec deux coordonnées de texture) ; ou à trois dimensions (avec trois coordonnées de texture). Les éléments d'une texture sont appelés *texels*, des éléments de texture (***tex**ture **el**ements*).

D'autres attributs sont les types des données d'entrée et de sortie, la manière d'interpréter les coordonnées d'entrée, entre autres.

4.2. Déclaration & attributs à la compilation

Quelques attributs doivent être connus à la compilation, et ne peuvent donc pas être changés plus tard. Ces attributs sont définis dès la déclaration de la référence de texture.

```
texture <Type, Dim, ReadMode> refDeTex;
```

Type y représente le type de données retourné lors du *fetch*. Cet attribut est limité aux entiers, aux flottants de simple précision et à tous les types de vecteurs, décrits plus haut.

Dim est la dimensionnalité, le nombre de dimensions, de la texture, qui peut valoir 1, 2 ou 3. Ce paramètre est optionnel, et vaut par défaut 1.

ReadMode ne peut prendre que deux valeurs : cudaReadModeNormalizedFloat ou cudaReadModeElementType.

Si *Type* est un entier de 16 ou de 8 bits, et que *ReadMode* vaut *cudaReadModeNormalizedFloat*, alors la valeur est, en vérité, retournée en tant que flottant. Toute la plage de valeurs de l'entier est reportée dans l'intervalle [0.0 ; 1.0] pour un non-signé, dans [- 1.0 ; + 1.0] pour un signé. Par exemple, 0xFF sera lu comme 1.

Si *ReadMode* vaut *cudaReadModeElementType*, aucune conversion n'est effectuée.

Il s'agit d'un paramètre optionnel, dont la valeur par défaut est *cudaReadModeElementType*.

4.3. Référence des attributs déclarés à l'exécution

Les autres attributs d'une référence sont *mutables*, et peuvent sans problème être changés à l'exécution. Ils spécifient si les coordonnées de la texture sont normalisées ou non, le mode d'adressage, et le filtrage.

Par défaut, les textures sont référencées avec des coordonnées flottantes dans l'intervalle [0 ; N], où *N* est la taille de la texture dans la dimension correspondant à la coordonnée.

Par exemple, une texture de taille 64 x 32 sera référencée avec des coordonnées dans les intervalles [0 ; 63] et [0 ; 31] pour les coordonnées *x* et *y*.

Des coordonnées normalisées reportent ces intervalles dans l'intervalle [0.0 ; 1.0]. Cette normalisation convient parfaitement à certaines applications, s'il est nécessaire que les coordonnées soient indépendantes de la taille de la texture, par exemple.

Le mode d'adressage définit ce qui arrive lorsqu'un élément hors dimensions est demandé. Les valeurs en dessous de 0 sont mises à 0, et les variables plus grandes que N sont mises à N-1, dans le cas de coordonnées non normalisées. Dans le cas de coordonnées normalisées, les coordonnées sont ramenées dans l'intervalle [0.0 ; 1.0]. Dans ce dernier cas, il existe aussi un mode *wrap*, qui n'utilise que la partie fractionnaire de la coordonnée. 1.25 devient 0.25 ; -1.25 devient 0.75.

Le filtrage de textures linéaires ne peut être effectué que sur des textures qui retournent des flottants. Il effectue des interpolations de basse précision entre les texels proches. Quand ce mode est activé, les texels proches de la cellule recherchée sont lus et la valeur retournée est interpolée, sur base de l'espace entre la valeur recherchée et les données. Une interpolation simple est effectuée pour des textures à une dimension, une interpolation bilinéaire est effectuée pour des textures à deux dimensions.

4.4. Comparaison des mémoires des textures

4.4.1. Mémoire linéaire et tableaux

Une texture peut être stockée en mémoire linéaire ou sous forme de tableaux. La première méthode possède quelques désavantages.

- Dimensionnalité forcée à 1 ;
- Pas de support de filtrage ;
- Adressage uniquement non normalisé ;
- Pas de mode d'adressage : toute valeur hors intervalle est ramenée à 0.

Le matériel met en vigueur une politique d'alignement sur les adresses des textures. Pour rendre ceci plus transparent pour les programmeurs, les fonctions de *bind* des références renvoient un *offset* à appliquer aux recherches pour lire la mémoire désirée. Les pointeurs de base retournés par les fonctions d'allocation de CUDA se conforment à cette contrainte, ce qui fait qu'il n'est pas obligatoire de passer *l'offset* lorsqu'on les utilise.

4.4.2. Mémoire constante et mémoire globale

La lecture sur le périphérique avec le principe des textures présente bien des avantages en comparaison des mémoires globale ou constante.

- Elles sont en cache (ce qui améliore fortement les performances si les données sont disponibles) ;
- Elles ne sont pas sujettes aux contraintes sur l'accès à la mémoire que les deux autres doivent respecter pour de bonnes performances ;
- La latence est mieux cachée, améliorant les performances des applications qui lisent aléatoirement la mémoire ;
- Les données d'entrée en entiers sur 8 ou 16 bits peuvent être converties en flottants sur 32 bits de l'intervalle [0.0 ; 1.0] ou [-1.0 ; 1.0].

Si, en plus, la texture est stockée sous forme de tableau, le matériel fournit d'autres capacités, qui peuvent être utiles pour certaines applications, spécialement dans le domaine du traitement d'images.

Capacité	Utilité	Problème
Filtrage	Interpolation rapide, mais peu précise, de texels	Valide uniquement si la référence de texture renvoie un flottant
Textures à coordonnées normalisées	Codage indépendant de la résolution	
Modes d'adressage	Gestion automatique des cas de bordure	Ne peut être utilisé qu'avec des coordonnées normalisées

Cependant, dans le même *kernel*, le cache des textures n'est pas gardé cohérent en fonction des écritures sur la mémoire globale. Ainsi, toute recherche sur une adresse qui a subi une écriture globale dans le même appel de *kernel* retourne des données indéfinies.

En d'autres termes, un thread peut lire en toute quiétude un endroit de la mémoire si et seulement si cet endroit a été

mis à jour par un autre *kernel* ou une copie mémoire, mais pas si elle l'a été par le même thread ou un autre thread du même appel de *kernel*.

Ceci n'a d'utilité que lors de la recherche depuis la mémoire linéaire, comme un *kernel* ne peut pas écrire dans un tableau.

4.5. Fonctions de fetching

Dans le cas de la mémoire linéaire, nous utilisons les fonctions de la famille tex1Dfetch().

```
float tex1Dfetch
(
    texture < unsigned char, 1,
cudaReadModeNormalizedFloat > refDeTex,
    int x
);
```

x est l'ordonnée du flottant dont on recherche la valeur dans la texture dont une référence est *refDeTex*.

Le filtrage et les modes d'adressage ne sont pas supportés, contrairement aux couples et aux quadruplets.

```
float4 tex1Dfetch
(
    texture < uchar4, 1,
cudaReadModeNormalizedFloat > texRef,
    int x
);
```

Par contre, dans le cas de tableaux, nous utiliserons les familles tex1D(), tex2D() et tex3D().

```
float tex1D
(
    texture < float, 1,
cudaReadModeNormalizedFloat > texRef,
    float x
);

float tex2D
(
    texture < float, 2,
cudaReadModeNormalizedFloat > texRef,
    float x, float y
);

float tex3D
(
    texture < float, 3,
cudaReadModeNormalizedFloat > texRef,
    float x, float y, float z
);
```

4.6. Fonctions de configuration

Le type texture, tel que défini par l'API haut niveau, est un dérivé du type textureReference, lui défini par l'API bas niveau.

```
struct textureReference
{
  int                          normalized     ;
  enum    cudaTextureFilterMode  filterMode     ;
  enum    cudaTextureAddressMode addressMode[3] ;
  struct  cudaChannelFormatDesc  channelDesc    ;
}
```

normalized non-nul signifie que la texture est normalisée ; toute autre valeur signifie la non-normalisation.

filterMode spécifie le mode de filtrage de la texture. S'il s'agit de cudaFilterModePoint, le texel le plus proche sera retourné ; s'il s'agit de cudaFilterModeLinear, une interpolation sera retournée.

addressMode précise le mode d'adressage : cudaAddressModeWrap impose que les valeurs incorrectes soient remises dans l'ensemble des correctes en n'utilisant que la partie fractionnaire ; cudaAddressModeClamp utilise l'autre mode.
Chacun des éléments du tableau correspond à une dimension à laquelle est appliqué le mode d'adressage.

channelDesc désigne le format de la valeur retournée à la lecture de la texture.

```
struct cudaChannelFormatDesc
{
    int x;
    int y;
    int z;
    int w;
    enum cudaChannelFormatKind f;
};
```

Chacun des entiers donne le nombre de bits de chaque composante retournée, **f**, leur type.

- **cudaChannelFormatKindSigned** : entier signé ;
- **cudaChannelFormatKindUnsigned** : entier non signé ;
- **cudaChannelFormatKindFloat** : flottant.

Tous ces paramètres peuvent donc être spécifiés à l'exécution depuis l'hôte. Ils ne s'appliquent, comme dit précédemment, qu'aux textures liées à un tableau.

Avant qu'un *kernel* ne puisse utiliser une référence de texture pour la lire, la référence doit être liée à une texture, avec les fonctions cudaBindTexture() ou cudaBindTextureToArray().

Ce code lie une référence à un espace en mémoire linéaire, pointée par devPtr.

```
//Avec l'API bas niveau
texture <float, 1, cudaReadModeElementType>
texRef;
textureReference * texRefPtr;
cudaGetTextureReference(&texRefPtr, "texRef");
cudaChannelFormatDesc channelDesc =
cudaCreateChannelDesc<float>();
cudaBindTexture(0, texRefPtr, devPtr,
&channelDesc, size);

//Avec l'API haut niveau
texture<float, 1, cudaReadModeElementType>
texRef;
cudaBindTexture(0, texRef, devPtr, size);
```

Ce code lie une référence à un tableau, cuArray.

```
//Avec l'API bas niveau
texture <float, 2, cudaReadModeElementType>
texRef;
textureReference * texRefPtr;
cudaGetTextureReference(& texRefPtr, "texRef");
cudaChannelFormatDesc channelDesc;
cudaGetChannelDesc(& channelDesc, cuArray);
cudaBindTextureToArray(texRef, cuArray, &
channelDesc);

//Avec l'API haut niveau
texture <float, 2, cudaReadModeElementType>
texRef;
cudaBindTextureToArray(texRef, cuArray);
```

Le format précisé lors de la liaison d'une texture à une référence doit correspondre aux paramètres de la déclaration de référence. Sinon, les résultats sont indéfinis.

La fonction cudaUnbindTexture() sert à délier une référence d'une texture.

4.7. Remarque

Vous ne pouvez pas déclarer une texture dans un *.cu* et l'utiliser dans un autre *.cu*, sous peine de problèmes à l'édition des liens.

Cependant, ceci peut être contourné. Il vous suffit d'utiliser des fonctions comme celles-ci pour les récupérer.

```
// Declaration d'une texture 2D
texture<float, 2, cudaReadModeElementType>
dataTest_tex;

// Accesseur depuis une fonction __host__
texture<float, 2, cudaReadModeElementType>
& getTexture()
{
    return dataTest_tex;
}

// Accesseur depuis une fonction __device__ ou
__global__
static __inline__ __device__
texture<float, 2, cudaReadModeElementType>
& getDeviceTexture()
{
    return dataTest_tex;
}
```

5. Les flux

5.1. Définition

Un flux est une source de données, comme un fichier, la mémoire ou le réseau. Ceci est une définition très générale, qui s'applique à tous les domaines. Dans CUDA, il ne s'agit pas exactement de la même chose.

Pour faciliter les exécutions concurrentes entre hôte et périphérique, certaines fonctions sont asynchrones, dans le runtime (la partie de CUDA que nous apprenons) : l'application récupère le contrôle avant que les calculs soient entièrement effectués. Ainsi, il arrive fréquemment que plusieurs de ces fonctions soient en cours d'exécution en même temps : elles sont concurrentes.

Ces fonctions sont les suivantes.

- cuLaunchGrid() ;
- cuLaunchGridAsync() ;
- Les fonctions de copie suffixées Async ;
- Les copies du périphérique vers le périphérique ;
- Les appels de *kernels* avec des fonctions __global__.

Les applications s'occupent de la concurrence *via* des *flux*. Un flux est une séquence d'instructions, qui doivent s'exécuter dans un certain ordre. D'un autre côté, des flux peuvent arrêter leur exécution pour un autre flux.

5.2. Création

Un flux est défini en créant un objet flux, et en le spécifiant à un appel de *kernel*, ou à une copie de mémoire. Par exemple, ce code crée deux flux.

```
cudaStream_t stream[2];
for (int i = 0 ; i < 2 ; ++i)
    cudaStreamCreate(& stream[i] );
```

Chacun de ces flux est défini, par ce code, comme une séquence d'une copie de l'hôte au périphérique, d'un appel de *kernel*, et d'une copie du périphérique vers l'hôte.

```
for (int i = 0 ; i < 2 ; ++i)
    cudaMemcpyAsync(inputDevPtr + i * size,
hostPtr + i * size, size, cudaMemcpyHostToDevice,
stream[i]);
for (int i = 0 ; i < 2 ; ++i)
    mykernel <<< 100, 512, 0, stream[i] >>>
(outputDevPtr + i * size, inputDevPtr + i * size,
size);
for (int i = 0 ; i < 2 ; ++i)
    cudaMemcpyAsync(hostPtr + i * size,
outputDevPtr + i * size, size,
cudaMemcpyDeviceToHost, stream[i]);
cudaThreadSynchronize();
```

Chaque flux copie sa portion du tableau hostPtr vers inputDevPtr dans la mémoire du périphérique, traite ce dernier tableau, et copie le résultat outputDevPtr sur l'hôte, dans la partie correspondante de hostPtr.

cudaThreadSynchronize() est appelée à la fin pour s'assurer que tous les flux ont fini de s'exécuter avant d'essayer d'utiliser leur résultat.

cudaStreamSynchronize() peut être appelée pour synchroniser un flux précis, avec tous les autres flux qui continuent leur exécution normale.

cudaStreamDestroy() est utilisée pour la destruction d'un flux.

cudaStreamQuery() permet de vérifier que toutes les opérations du flux déjà envoyées soient bien effectuées.

Tout appel de *kernel* ou opération de mémoire sans le paramètre de flux ne commence qu'à la fin des autres opérations. Ces opérations sont affectées au flux 0.

5.3. Événements

Les événements permettent de vérifier l'état d'avancement du flux.

Voici la création de deux événements.

```
cudaEvent_t start, stop;
cudaEventCreate(&start);
cudaEventCreate(&stop);
```

Et voici une autre version du code précédent, qui permet de vérifier son état d'avancement, grâce aux événements.

```
cudaEventRecord(start, 0);

for (int i = 0; i < 2; ++i)
    cudaMemcpyAsync(inputDev + i * size,
inputHost + i * size, size,
cudaMemcpyHostToDevice, stream[i]);
for (int i = 0; i < 2; ++i)
    mykernel <<< 100, 512, 0, stream[i] >>>
(outputDev + i * size, inputDev + i * size,
size);
for (int i = 0; i < 2; ++i)
    cudaMemcpyAsync(outputHost + i * size,
outputDev + i * size, size,
cudaMemcpyDeviceToHost, stream[i]);

cudaEventRecord(stop, 0);
cudaEventSynchronize(stop);
float elapsedTime;
cudaEventElapsedTime(&elapsedTime, start, stop);
cudaEventDestroy(start);
cudaEventDestroy(stop);
```

Ce code est assez explicite : on précise qu'un événement a lieu (ligne 1), puis on exécute les instructions (lignes 3 à 8), on enregistre qu'un autre événement a lieu (ligne 10), on attend qu'il soit émis (ligne 11), on calcule le temps écoulé entre les deux événements (lignes 12 et 13), puis on supprime les événements (lignes 14 et 15).

6. Gestion des erreurs

Aucune fonction de CUDA ne renvoie directement ses erreurs. Il faut faire appel à une fonction spéciale, cudaGetLastError(), qui renvoie le code d'erreur, sous forme de cudaError_t.

Toutes les fonctions du runtime renvoient donc un code d'erreur, mais dans le cas d'appels asynchrones, comme le *kernel* retourne immédiatement, on ne peut pas savoir si une erreur a eu lieu. Vous devez donc utiliser une fonction de synchronisation afin de pouvoir récupérer l'erreur.

Chaque erreur est effacée par la suivante, et une réussite renvoie aussi un code d'erreur. Récupérer une erreur remet le compteur à la réussite.

Un *kernel* ne peut pas retourner de code d'erreur, on doit donc vérifier s'il peut s'exécuter sans problème avant de le lancer.

Il existe une seconde fonction pour gérer les erreurs : cudaGetErrorString(), qui prend en paramètre un code d'erreur, tel que retourné par cudaGetLastError(). Elle retourne, sous la forme d'un const char *, la description de l'erreur.

Code d'erreur	Signification
cudaSuccess	Pas d'erreur.
cudaErrorMissingConfiguration	Configuration manquante.
cudaErrorMemoryAllocation	Erreur lors de l'allocation de la mémoire.
cudaErrorInitializationError	Erreur lors de l'initialisation.
cudaErrorLaunchFailure	Erreur au lancement.
cudaErrorPriorLaunchFailure	Erreur lors d'un lancement précédent.
cudaErrorLaunchTimeout	Temps d'exécution trop long.
cudaErrorLaunchOutOfResources	Plus de ressources disponibles pour l'exécution.
cudaErrorInvalidDeviceFunction	Fonction invalide pour le périphérique.
cudaErrorInvalidConfiguration	Configuration invalide.
cudaErrorInvalidDevice	Périphérique invalide.
cudaErrorInvalidValue	Valeur invalide.
cudaErrorInvalidPitchValue	Valeur de pas invalide.
cudaErrorInvalidSymbol	Symbole invalide.
cudaErrorMapBufferObjectFailed	Erreur lors de l'accès au buffer.
cudaErrorUnmapBufferObjectFailed	Erreur lors de l'accès au buffer.
cudaErrorInvalidHostPointer	Pointeur sur l'hôte invalide.
cudaErrorInvalidDevicePointer	Pointeur sur le périphérique invalide.
cudaErrorInvalidTexture	Texture invalide.
cudaErrorInvalidTextureBinding	Binding de texture invalide.
cudaErrorInvalidChannelDescriptor	Descripteur de chaîne invalide.
cudaErrorInvalidMemcpyDirection	Direction invalide pour memcpy.
cudaErrorAddressOfConstant	Erreur dans l'adresse d'une constante.
cudaErrorTextureFetchFailed	Erreur lors de l'accès à une texture.
cudaErrorTextureNotBound	Texture non liée.
cudaErrorSynchronizationError	Erreur de synchronisation.
cudaErrorInvalidFilterSetting	Paramètre de filtre invalide.
cudaErrorInvalidNormSetting	Paramètre de norme invalide.
cudaErrorMixedDeviceE	Exécution mixte sur le

xecution	périphérique.
cudaErrorCudartUnloading	CUDA runtime en cours de déchargement.
cudaErrorUnknown	Erreur inconnue.
cudaErrorNotYetImplemented	Fonction pas encore implémentée.
cudaErrorMemoryValueTooLarge	Valeur mémoire trop grande.
cudaErrorInvalidResourceHandle	Handle de ressource invalide.
cudaErrorNotReady	Pas encore prêt.
cudaErrorInsufficientDriver	CUDA runtime plus récent que le driver.
cudaErrorSetOnActiveProcess	Set on active process error.
cudaErrorNoDevice	Pas de périphérique disponible.
cudaErrorStartupFailure	Échec du lancement.
cudaErrorApiFailureBase	Erreur dans l'API.

7. Gestion des périphériques

cudaGetDeviceCount() et cudaGetDeviceProperties() permettent d'énumérer les périphériques, les cartes graphiques, et leurs capacités.

```
int deviceCount;
cudaGetDeviceCount(& deviceCount);
int device;

for (device = 0 ; device < deviceCount ; ++device)
{
    cudaDeviceProp deviceProp;
    cudaGetDeviceProperties(&deviceProp, device);
}
```

Seuls les périphériques avec un indice de capacité de calcul supérieur ou égal à 1.0 sont retournés, les autres ne pouvant pas être exploités par CUDA.

cudaSetDevice() sert à sélectionner le périphérique qui exécutera les calculs du thread CPU.

```
cudaSetDevice(device);
```

Un périphérique doit être choisi **avant** d'appeler le moindre *kernel*, soit explicitement avec cudaSetDevice(), soit implicitement (le périphérique 0 est alors choisi). Tout appel ultérieur à cudaSetDevice() échouera.

La fonction cudaGetDeviceProperties() renvoie une structure ainsi définie.

```
struct cudaDeviceProp
{
    char name[256];
    size_t totalGlobalMem;
    size_t sharedMemPerBlock;
    int regsPerBlock;
    int warpSize;
    size_t memPitch;
    int maxThreadsPerBlock;
    int maxThreadsDim[3];
    int maxGridSize[3];
    size_t totalConstMem;
    int major;
    int minor;
    int clockRate;
    size_t textureAlignment;
    int deviceOverlap;
    int multiProcessorCount;
    int kernelExecTimeoutEnabled;
}
```

- **name** : chaîne ASCII identifiant le périphérique ;
- **totalGlobalMem** : mémoire globale totale disponible sur le périphérique en bytes ;
- **sharedMemPerBlock** : mémoire partagée disponible pour un bloc en bytes (cette mémoire est partagée par tous les blocs de threads résidant simultanément sur un multiprocesseur) ;
- **regsPerBlock** : nombre de registres 32 bits disponibles à un bloc de threads ;
- **warpSize** : taille d'un warp en threads ;
- **memPitch** : pas maximal en bytes permis par la copie de mémoire qui implique des régions de mémoire allouées par cudaMallocPitch(); ;
- **maxThreadsPerBlock** : nombre maximal de threads par bloc ;
- **maxThreadsDim[3]** : dimensions maximales d'un bloc ;
- **maxGridSize[3]** : dimensions maximales d'une grille ;
- **totalConstMem** : total de mémoire constante disponible sur le périphérique ;
- **major** : numéro de révision majeur de la capacité de calcul ;
- **minor** : numéro de révision mineur de la capacité de calcul ;
- **clockRate** : fréquence de l'horloge en kilohertz ;
- **textureAlignment** : alignement des textures requis, les adresses de base des textures qui sont alignées sur ceci n'ont pas besoin d'appliquer un offset lors des recherches ;
- **deviceOverlap** : 1 si le périphérique peut copier de la mémoire entre l'hôte et le périphérique en même temps qu'exécuter un *kernel*, 0 sinon ;
- **multiProcessorCount** : nombre de multiprocesseurs sur le périphérique ;
- ***kernel*ExecTimeoutEnabled** : 1 s'il y a une limite de temps pour les *kernels*, 0 sinon.

Vous pouvez définir vous-même une structure de ce genre, et demander à CUDA de choisir le périphérique qui y correspond le mieux, grâce à la fonction cudaChooseDevice(int * dev, const struct cudaDeviceProp * prop) (elle retourne le périphérique dans dev).

Retrouvez la suite de l'article de Thibaut Cuvelier en ligne : Lien189

Les chiffres impressionnants des effectifs de Blizzard travaillant sur World Of Warcraft, le jeu massivement multijoueur en ligne

Nombreux sont ceux qui voudraient s'essayer à développer un MMORPG.
Petit ou grand, il s'agit d'un projet conséquent qui ne peut être abordé seul.
Non seulement l'équipe doit être importante, mais également le côté technique avec l'infrastructure (bande passante, serveurs, etc.).

Pour convaincre quelques sceptiques, voici quelques chiffres que des responsables de chez Blizzard ont laissé filtrer :

Citation des **équipes chargées de World Of Warcraft se subdivisent en pas moins de 30 départements**

- Celui chargé de la **programmation** du jeu massivement multijoueur est composé de **32 personnes** ayant réalisé pas moins de **5,5 millions de lignes de code**
- Pour le **design**, il est question de **37 personnes** à l'heure actuelle. Des employés chargés notamment de l'organisation des événements internes au jeu, mais qui ont également à leur actif pas moins de 70.000 compétences et quelques 40.000 personnages non-joueurs.
- Le département en charge du « **service qualité** » n'est évidemment pas en reste et avec **245 personnes** employées, il **a déjà identifié plus de 180.000 bugs**.
- Plus proche des joueurs, l'équipe en charge du suivi et de ce que l'on appelle le « **support** » est constituée de **2056 gametesters**
- Auxquels sont associées 340 personnes chargées du support commercial et 66 personnes qui s'occupent de la relation entre Blizzard et les joueurs.
- Disponibilité du jeu en 10 langues
- Chaque **nouvelle langue** revient à gérer plus de **360.000 lignes de texte**, soit environ 2 millions de mots !
- L'équipe en charge des différents services Internet gère plus de 900.000 fichiers.
- Le jeu fait travailler 4.600 personnes
- Sur le plan technique ce sont plus de 20.000 ordinateurs, 13.250 serveurs, 75.000 microprocesseurs, 112,5 teraoctets de mémoire vive et 1,3 petaoctet de données...

Si après ça vous avez toujours envie de vous lancer dans le développement d'un MMO, vous ne pourrez pas dire qu'on ne vous a pas prévenus

Que pensez vous de ces chiffres ?

Commentez cette news en ligne : Lien190

Les processeurs graphiques sont-ils voués à disparaître ?

Selon Tim Sweeny, CEO fondateur de Epics Game (moteurs Unreal Tournament), le processeur graphique serait à inscrire sur la liste des espèces menacées.

La technologie OpenCL (Open Computing Language) est le résultat du mariage entre une API et un langage de programmation dérivé du C. Elle a été crée pour faciliter la programmation de l'intersection entre les CPU (de plus en plus parallèles) et les GPU (de plus en plus programmables).

Mais alors que la technologie Grand Central (qui simplifie le développement multithread) semble être bien absorbée, la technologie OpenCL pose, elle, de très gros problèmes de par l'éloignement géographique entre le processeur graphique (relégué à la terminaison d'un port PCI Express) et la mémoire centrale. Dans ce cas, le transfert mémoire des données nécessaires au calcul coûte énormément de temps et d'argent, et il faudra alors évaluer si le processeur est plus interessant en termes financiers par rapport à la parallélisation des calculs.

En effet, bien qu'il soit extrêmement puissant pour les calculs parallèles, le processeur graphique est en général assez mauvais pour les tests. Comment vaincre ce problème, d'autant plus lorsque l'on ne connaît pas la complexité des calculs qui seront à effectuer (ce qui est fréquent pour les projets de haute performance) ?

Si l'on se penche pour prendre le pouls de ce marché, on s'aperçoit que NVIDIA y est leader, talonné à distance respectueuse par Intel (dont la puissance des processeurs graphiques ne rivalise pas encore avec ceux du "maître" mais des innovations sont en route - implémentation de traitements graphiques et d'architectures superscalaires).

NVIDIA serait donc logiquement le premier à pâtir de cette situation qui pourrait malmener ce marché dans sa globalité.

Source : L'étude de Tim Sweeny (Lien191)

NVIDIA réussira-t-il à se sortir de cette mauvaise passe simplement en incorporant des processeurs graphiques dans des ARM ?

ATI se plaçant juste après NVIDIA en terme de performances, et appartenant à AMD qui serait en mesure d'instaurer un traitement graphique dans ses processeurs, pourrait-il doubler NVIDIA et lui prendre sa place de leader ?

Commentez cette news en ligne : Lien192

Liens

Lien1 : http://openjdk.java.net/projects/coin/

Lien2 : http://blogs.sun.com/darcy/entry/project_coin_final_five
Lien3 : http://blog.developpez.com/adiguba/p7992/java/7-dolphin/diamond-syntax/
Lien4 : http://java.developpez.com/faq/java/?page=langage_fichiers#libererRessources
Lien5 : http://www.developpez.net/forums/d460138/java/communaute-java/debats/jdk-7-proposition-8-rethrow-exceptions/
Lien6 : http://www.developpez.net/forums/d460135/java/communaute-java/debats/jdk-7-proposition-7-pouvoir-catcher-plusieurs-exceptions/
Lien7 : http://www.developpez.net/forums/d803232/java/communaute-java/debats/projet-coin-modifications-langage-java-7-a/
Lien8 : http://blog.developpez.com/adiguba/p8012/java/7-dolphin/modif-du-project-coin/
Lien9 : http://www.gdawj.com/
Lien10 : http://java.developpez.com/livres/
Lien11 : http://www.developpez.net/forums/d810007/php/langage/php-t-obtenu-tant-succes/
Lien12 : http://www.w3avenue.com/2009/08/26/really-useful-tools-for-php-developers/
Lien13 : http://eaccelerator.net/
Lien14 : http://www.php-accelerator.co.uk/
Lien15 : http://phing.info/
Lien16 : http://www.beautifyphp.com/
Lien17 : http://pear.php.net/package/PHP_Beautifier
Lien18 : http://www.phpobjectgenerator.com/
Lien19 : http://uml2php5.zpmag.com/en/index.php
Lien20 : http://live.gnome.org/Dia
Lien21 : http://www.dpriver.com/
Lien22 : http://www.phpmyedit.org/
Lien23 : http://www.raizlabs.com/software/phpobfuscator/
Lien24 : http://www.codeeclipse.com/
Lien25 : http://phpdataservices.codeplex.com/
Lien26 : http://propel.phpdb.org/
Lien27 : http://adodb.sourceforge.net/
Lien28 : http://www.doctrine-project.org/
Lien29 : http://www.xdebug.org/
Lien30 : http://code.google.com/p/webgrind/
Lien31 : http://www.bluestatic.org/software/macgdbp/
Lien32 : http://www.firephp.org/
Lien33 : http://www.php-debugger.com/
Lien34 : http://www.php-debug.com/
Lien35 : http://phpdebuglib.de/
Lien36 : http://krumo.sourceforge.net/
Lien37 : http://matrix.squiz.net/developer/tools/php_cs
Lien38 : http://manual.phpdoc.org/
Lien39 : http://www.pdepend.org/
Lien40 : http://phplangeditor.mozdev.org/
Lien41 : http://www.aptana.com/php
Lien42 : http://www.phpeclipse.com/
Lien43 : http://www.zend.com/products/studio/
Lien44 : http://phpanywhere.net/
Lien45 : http://www.jcxsoftware.com/vs.php
Lien46 : http://www.netbeans.org/features/php/index.html
Lien47 : http://www.nusphere.com/products/phped.htm
Lien48 : http://www.phpedit.com/en
Lien49 : http://www.mpsoftware.dk/phpdesigner.php
Lien50 : http://macromates.com/
Lien51 : http://www.activestate.com/komodo/
Lien52 : http://php-ids.org/
Lien53 : http://phpsec.org/projects/phpsecinfo/index.html
Lien54 : http://www.analogx.com/contents/download/network/phpconf.htm
Lien55 : http://www.lighty2go.com/
Lien56 : http://wiki.opensource.nokia.com/projects/PAMP
Lien57 : http://www.wampserver.com/
Lien58 : http://www.server2go-web.de/
Lien59 : http://www.phpunit.de/
Lien60 : http://simpletest.sourceforge.net/
Lien61 : http://code.google.com/p/bovigo/wiki/vfsStream
Lien62 : http://phpundercontrol.org/
Lien63 : http://www.addedbytes.com/cheat-sheets/php-cheat-sheet/
Lien64 : http://www.serversidemagazine.com/cheat-sheets/PHP5/
Lien65 : http://cakephp.org/downloads/Resources
Lien66 : http://koivi.com/apache-iis-php-server-array.php
Lien67 : http://hasin.wordpress.com/2006/06/10/smarty-cheat-sheet-version-20/
Lien68 : http://jcrozier.developpez.com/tutoriels/web/php/classes-et-librairies-utiles-developpeurs/
Lien69 : http://jcrozier.developpez.com/
Lien70 : http://jcrozier.developpez.com/tutoriels/web/php/outils-utiles-developpeurs/
Lien71 : http://en.wikipedia.org/wiki/Rasmus_Lerdorf
Lien72 : http://www.phpframeworks.com/
Lien73 : http://www.w3avenue.com/2009/08/11/really-useful-classes-and-libraries-for-php-developers/
Lien74 : http://adodb.sourceforge.net/
Lien75 : http://www.doctrine-project.org/

Lien76 : http://phplinq.codeplex.com/
Lien77 : http://mimesis.110mb.com/
Lien78 : http://matrix.squiz.net/developer/tools/php_cs
Lien79 : http://manual.phpdoc.org/
Lien80 : http://www.tcpdf.org/
Lien81 : http://www.phppowerpoint.net/
Lien82 : http://www.phpexcel.net/
Lien83 : http://sourceforge.net/projects/phprtf/
Lien84 : http://www.phpconcept.net/pclzip/index.en.php
Lien85 : http://jcrozier.developpez.com/tutoriels/web/php/classes-et-librairies-utiles-developpeurs/
Lien86 : http://fr.wikipedia.org/wiki/Byte_Order_Mark
Lien87 : http://j-willette.developpez.com/tutoriels/web/encoder-son-site-en-utf8/
Lien88 : http://www.adobe.com/fr/products/creativesuite/
Lien89 : http://bwp-necromance.developpez.com/tutoriel/adobe/illustrator/cs4/images/mod_indispensable.jpg
Lien90 : http://bwp-necromance.developpez.com/tutoriel/adobe/illustrator/cs4/images/mod_automatisation.jpg
Lien91 : http://bwp-necromance.developpez.com/tutoriel/adobe/illustrator/cs4/images/mod_commefreehand.jpg
Lien92 : http://bwp-necromance.developpez.com/tutoriel/adobe/illustrator/cs4/images/mod_commeindesign.jpg
Lien93 : http://bwp-necromance.developpez.com/tutoriel/adobe/illustrator/cs4/images/mod_commephotoshop.jpg
Lien94 : http://bwp-necromance.developpez.com/tutoriel/adobe/illustrator/cs4/images/mod_impression.jpg
Lien95 : http://bwp-necromance.developpez.com/tutoriel/adobe/illustrator/cs4/images/mod_peinture.jpg
Lien96 : http://bwp-necromance.developpez.com/tutoriel/adobe/illustrator/cs4/images/mod_typographie.jpg
Lien97 : http://bwp-necromance.developpez.com/tutoriel/adobe/illustrator/cs4/images/mod_web.jpg
Lien98 : http://bwp-necromance.developpez.com/tutoriel/adobe/illustrator/cs4/
Lien99 : http://danielhagnoul.developpez.com/fondations/cloturer/exemple1.html
Lien100 : http://danielhagnoul.developpez.com/fondations/cloturer/exemple2.html
Lien101 : https://developer.mozilla.org/fr/R%C3%A9f%C3%A9rence_de_JavaScript_1.5_Core/Op%C3%A9rateurs/Op%C3%A9rateurs_sp%C3%A9ciaux/l%27op%C3%A9rateur_this#Liaison_de_m.C3.A9thode
Lien102 : http://danielhagnoul.developpez.com/fondations/cloturer/exemple3.html
Lien103 : http://www.developpez.net/forums/u28015/marcha/
Lien104 : http://www.developpez.net/forums/d674593/webmasters-developpement-web/javascript/bibliotheques-frameworks/jquery/script-defilement-vertical/#post3947789
Lien105 : http://danielhagnoul.developpez.com/fondations/fonctionGlobale/exemple1.html
Lien106 : http://t-templier.developpez.com/tutoriel/javascript/javascript-poo1/
Lien107 : http://danielhagnoul.developpez.com/fondations/fonctionGlobale/exemple2.html
Lien108 : http://fr.wikipedia.org/wiki/NameSpace#Langages_de_programmation
Lien109 : http://falola.developpez.com/tutoriels/javascript/namespace/
Lien110 : http://james.padolsey.com/javascript/jquery-delay-plugin/
Lien111 : http://james.padolsey.com/javascript/jquery-delay-plugin/
Lien112 : http://talideon.com/weblog/2005/07/javascript-memoization.cfm
Lien113 : http://fr.wikipedia.org/wiki/Memoization
Lien114 : http://fr.wikipedia.org/wiki/Modulo_(informatique)
Lien115 : http://pagesperso-orange.fr/jean-paul.cornec/MJD.htm
Lien116 : http://giminik.developpez.com/articles/javascript-dom/table-des-matieres/
Lien117 : http://www.developpez.net/forums/u308/giminik/
Lien118 : http://danielhagnoul.developpez.com/tutoriels/javascript/outils-pour-construire-code-jquery-evolutif/
Lien119 : http://www.jankoatwarpspeed.com/post/2009/04/19/Create-CSS-pin-balloons-with-ease.aspx
Lien120 : http://css.developpez.com/
Lien121 : http://www.stopchildlabour.eu/africatour2008/
Lien122 : http://www.stylishmedia.com/
Lien123 : http://r-hunel.developpez.com/tutoriels/css/bulle-informations/fichiers/
Lien124 : http://fr.wikipedia.org/wiki/Symétrie_axiale
Lien125 : http://r-hunel.developpez.com/tutoriels/css/bulle-informations/
Lien126 : http://jcrozier.developpez.com/
Lien127 : http://jcrozier.developpez.com/tutoriels/web/php/utilisation-avancee-procedures-stockees-mysql/
Lien128 : http://blog.developpez.com/mikedavem/p8137/sql-server-2008/sql/sql-server-2008-transtypage-de-date-et-o/
Lien129 : http://oracle.developpez.com/faq/faq-sqlplus/
Lien130 : http://jpcheck.developpez.com/tutoriels/access/access-et-fichiers-batch-passage-parametres/
Lien131 : http://excel.developpez.com/faq
Lien132 : http://jpcheck.developpez.com/tutoriels/access/access-et-fichiers-batch-passage-parametres/#LV
Lien133 : http://windows.developpez.com/cours/ligne-commande/
Lien134 : http://notepad-plus.sourceforge.net/fr/site.htm
Lien135 : http://astase.com/produits/
Lien136 : http://support.microsoft.com/kb/291288/fr
Lien137 : http://msdn.microsoft.com/en-us/library/ms683156(VS.85).aspx
Lien138 : http://fr.wikipedia.org/wiki/Table_des_caractères_Unicode
Lien139 : http://fr.wikipedia.org/wiki/American_National_Standards_Institute
Lien140 : http://windows.developpez.com/faq/win32/?page=concepts#CONCEPTS_ansi_unicode
Lien141 : http://jpcheck.developpez.com/tutoriels/office/excel-et-fichiers-batch-passage-parametres/
Lien142 : http://arkham46.developpez.com/articles/office/assistant-ruban/
Lien143 : http://warin.developpez.com/access/ruban/
Lien144 : http://silkyroad.developpez.com/excel/ruban/
Lien145 : http://silkyroad.developpez.com/excel/callbacks/
Lien146 : http://heureuxoli.developpez.com/office/word/ruban/
Lien147 : http://www.microsoft.com/downloads/details.aspx?FamilyID=12b99325-93e8-4ed4-8385-74d0f7661318&displaylang=en
Lien148 : http://www.microsoft.com/downloads/details.aspx?familyid=4329D9E9-4D11-46A5-898D-23E4F331E9AE&displaylang=en
Lien149 : http://arkham46.developpez.com/articles/access/rubanimages/
Lien150 : http://matthieu-brucher.developpez.com/tutoriels/3D/raytracer/
Lien151 : https://code.launchpad.net/irt
Lien152 : http://www.uni-kl.de/AG-Heinrich/EMS.pdf
Lien153 : http://matthieu-brucher.developpez.com/tutoriels/3D/raytracer/04-oversampling-boundingbox/
Lien154 : http://cpp.developpez.com/faq/cpp/?page=heritage#heritage_lsp
Lien155 : http://r0d.developpez.com/tutoriels/cpp/variations-sur-dp-state/

Lien156 : http://qt.developpez.com/binaires/fr/
Lien157 : http://www.developpez.net/forums/d807522/c-cpp/bibliotheques/qt/binaires-qt-disposition/
Lien158 : http://www.developpez.net/forums/m4657309-10/
Lien159 : http://www.developpez.net/forums/d811427/c-cpp/bibliotheques/qt/declarative-ui-futur-developpement-dihm/
Lien160 : http://qt.nokia.com/downloads
Lien161 : http://blog.qt.nokia.com/
Lien162 : http://www.developpez.net/forums/d822180/c-cpp/bibliotheques/qt/nouvelles-qt-4-6-suite-sortie-premiere-beta/
Lien163 : http://mac.developpez.com/interviews/patrick-geiller/
Lien164 : https://devwhy.s3.amazonaws.com/file..._5_minutes.pdf
Lien165 : http://macdaddyworld.com/Ken_Aspeslagh_BlitzTalk.pdf
Lien166 : http://geekable.com/blitz/c4_3/jcz_c...talk_sep22.pdf
Lien167 : http://www.dashingfalcon.com/webfm_send/10
Lien168 : http://www.panic.com/coda/
Lien169 : http://www.sunsetlakesoftware.com/si...ePlotIntro.pdf
Lien170 : http://rentzsch.com/
Lien171 : http://www.appcelerator.com/
Lien172 : http://flyingmeat.com/jstalk/
Lien173 : http://mac.developpez.com/interviews/patrick-geiller/
Lien174 : http://pyobjc.sourceforge.net/
Lien175 : http://www.dribin.org/dave/blog/arch.../c4_book_list/
Lien176 : http://devwhy.blogspot.com/2009/09/c43.html
Lien177 : http://clang.llvm.org/
Lien178 : http://www.llvm.org/
Lien179 : http://www.developpez.net/forums/d814609/systemes/autres-systemes/mac/petit-resume-c4-rendez-developpeurs-mac-iphone/
Lien180 : http://verifysoft.developpez.com/articles/qualite/metriques-index-maintenabilite/
Lien181 : http://blog.developpez.com/bruno-orsier/p8008/developpement-agile/scrum-et-les-journees-labo/
Lien182 : http://pyxis-tech.com/blog/2009/05/05/nouveau-processus-caddy-a-pyxis-technologies/
Lien183 : http://blog.developpez.com/bruno-orsier/p6916/developpement-agile/coaching-pour-managers-agiles/
Lien184 : http://blog.developpez.com/bruno-orsier/p7330/developpement-agile/essayons-de-generer-de-la-connaissance/
Lien185 : http://blog.developpez.com/bruno-orsier/p8092/developpement-agile/mise-en-place-du-caddying/
Lien186 : http://conception.developpez.com/livres/
Lien187 : http://tcuvelier.developpez.com/gpgpu/cuda/introduction/
Lien188 : http://www.gpgpu.org/
Lien189 : http://tcuvelier.developpez.com/gpgpu/cuda/approfondi/
Lien190 : http://www.developpez.net/forums/d811100/applications/developpement-2d-3d-jeux/chiffres-impressionnants-effectifs-blizzard-travaillant-world-of-warcraft/
Lien191 : http://graphics.cs.williams.edu/archive/SweeneyHPG2009/TimHPG2009.pdf
Lien192 : http://www.developpez.net/forums/d807440/club-professionnels-informatique/actualites/processeurs-graphiques-voues-disparaitre/