Developpez
Le Mag

Edition de Décembre - Janvier 2009/2010.
Numéro 25.
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

Editorial

Ce mois-ci Google fait notre Une. Quoi de plus normal alors que la firme de Mountain View est de plus en plus présente dans nos vies.
Mais rassurez-vous, chez Developpez nous savons diversifier pour les goûts de chacun.

La rédaction

Article Développement Web

Introduction à la programmation d'une extension Google Chrome

Initiez-vous au développement d'une extension pour Google Chrome à l'aide d'un exemple simple.

par **Sylvain Dangin**
Page 16

Article Java/Eclipse

Introduction à Google App Engine

Cet article constitue une introduction à Google App Engine. Il s'agit de la traduction française de la présentation originale de Google App Engine.

par **Brocoli**
Page 2

Introduction à Google App Engine

Cet article constitue une introduction à Google App Engine. Il s'agit de la traduction française de la présentation originale de Google App Engine.

1. Qu'est-ce que Google App Engine ?

Google App Engine permet d'exécuter vos applications web sur l'infrastructure de Google. Les applications App Engine sont faciles à construire, faciles à maintenir et supportent facilement la montée en charge de votre trafic et de vos besoins croissants de stockage de données. Avec App Engine, il n'y a pas de serveurs à maintenir : vous chargez juste vos applications, et elles sont aussitôt disponibles pour vos utilisateurs.

Vous pouvez mettre à disposition vos applications depuis votre propre nom de domaine (tel que http://www.example.com/) en utilisant Google Apps (Lien1) ou utiliser directement le nom qui vous est fourni sur le domaine appspot.com. Vous pouvez partager votre application avec le monde, ou limiter l'accés aux membres de votre organisation.

Google App Engine supporte les applications écrites dans plusieurs langages de programmation. Avec l'environnement d'exécution Java™ App Engine (JRE), vous pouvez construire votre application en utilisant les technologies standard Java, incluant : la JVM, les servlets Java, et le langage de programmation Java (ou n'importe quel autre langage utilisant un compilateur ou un interpréteur basé sur une JVM, tels que le JavaScript ou le Ruby). App Engine comprend également un environnement d'exécution Python dédié, qui inclut un interpréteur Python rapide et une librairie Python standard. Les environnements Java et Python sont construits de sorte à assurer à vos applications un fonctionnement rapide, dans un espace sécurisé, et sans interférences avec les autres applications du système.

Avec App Engine, vous payez seulement ce que vous utilisez. Il n'y a, ni coût de mise en service, ni commissions récurrentes. Les ressources que vos applications utilisent, tel que le stockage et la bande passante, sont mesurées en giga-octets et facturées à des taux compétitifs. Vous contrôlez le niveau maximum de ressources que peut consommer votre application, de sorte que votre budget ne soit jamais dépassé.

App Engine ne coûte rien pour démarrer. Toutes les applications peuvent utiliser jusqu'à 500 Mo de stockage et assez de CPU et de bande passante pour supporter une demande d'environ 5 millions de pages vues par mois, absolument gratuitement. Quand vous activez la facturation pour votre application, vos limites gratuites sont augmentées et vous ne payez seulement que les ressources que vous consommez au-delà de ces limites.

2. L'environnement applicatif

Google App Engine rend facile la construction d'une application qui s'exécute de façon fiable, même sous une forte charge et avec d'importantes quantités de données. App Engine inclut les caractéristiques suivantes:

- Mise à disposition d'applications web dynamiques, avec support complet des technologies communes du web
- Stockage persistant avec recherches, tris et transactions
- Dimensionnement automatique selon la montée en charge et load-balancing
- API pour l'authentification des utilisateurs et envoi de courriers électroniques en utilisant les comptes Google
- un environnement de développement local complet qui simule Google App Engine sur votre ordinateur
- tâches planifiées pour déclencher des évènements à des horaires spécifiés et à des intervalles réguliers.

Votre application peut s'exécuter dans l'un des deux environnements suivant : Java ou Python. Chaque environnement fournit des protocoles standards et des technologies communes pour le développement d'applications web.

2.1. L'environnement sécurisé et indépendant ("Sandbox")

Les applications s'exécutent dans un environnement sécurisé qui fournit des accès limités au système d'exploitation. Ces limitations permettent à App Engine de distribuer les requêtes web pour l'application à travers de multiples serveurs, et de démarrer et arrêter les serveurs selon la charge. Votre application est isolée dans son propre environnement sécurisé et fiable qui est indépendant du matériel, du système d'exploitation et de la localisation physique du serveur web.

Quelques exemples des limitations de l'environnement sécurisé:

- Une application ne peut accéder aux autres ordinateurs sur Internet, que par les services de rapatriement d'URL et d'email fournis. Les autres ordinateurs ne peuvent se connecter à l'application qu'en faisant des requêtes HTTP (ou HTTPS) sur les ports standards.
- Une application ne peut pas écrire sur le système de fichiers. Une application peut lire des fichiers,

mais seulement les fichiers chargés avec le code de l'application. L'application doit utiliser la base de données App Engine ("datastore"), memcache ou d'autres services pour toutes les données qui persistent entre les requêtes.
- Le code de l'application s'exécute seulement en réponse à une requête web ou à une tâche dans la cron, et doit dans tous les cas retourner une réponse dans les 30 secondes. Un gestionnaire de requêtes ne peut pas lancer de process ou exécuter du code après que la réponse ait été fournie.

2.2. L'Environnement d'Exécution Java (JRE)

Vous pouvez développer votre application pour l'environnement d'exécution Java en utilisant les outils communs de développement web Java ainsi que les API standards. Votre application intéragit avec l'environnement en utilisant le standard des Servlet Java (Lien2) et peut utiliser des technologies courantes d'applications web telles que les JavaServer Pages (JSP) (Lien3).

Le JRE utilise Java 6. Le SDK Java App Engine supporte le développement d'applications utilisant le Java 5 ou 6.

L'environnement inclut la plateforme Java SE Runtime Environment (JRE) 6 (Lien4) et les librairies. Les restrictions de l'environnement indépendant sont implémentées dans la JVM. Une application peut utiliser n'importe quel propriété de code binaire ou de librairie, tant qu'elle n'outrepasse pas les restrictions de l'environnement. Par exemple, le code binaire qui essaye d'ouvrir une socket ou d'écrire un fichier provoquera une exception d'exécution.

Votre application accède à la plupart des services App Engine en utilisant les API Java standards. Pour le datastore App Engine, le SDK Java inclut une implémentation des Java Data Objects (JDO) (Lien5) et des interfaces Java Persistence API (JPA) (Lien6). Votre application peut utiliser l'API JavaMail (Lien7) pour envoyer des courriers électroniques avec le service de messagerie App Engine Mail. Les API HTTP java.net utilisent le service de rapatriement d'URL App Engine. App Engine inclut également des API bas-niveau pour ses services pour implémenter des adaptateurs additionnels, ou pour les utiliser directement depuis l'application. Consultez les documentations de ces services sur le datastore, memcache, URL fetch, mail, images et Google Accounts API.

Typiquement, les développeurs Java utilisent le langage de programmation Java et les API pour implémenter des applications web pour la JVM. Avec l'utilisation d'interpréteurs ou de compilateurs compatibles avec la JVM, vous pouvez aussi utiliser d'autres langages pour développer des applications web, tels que le JavaScript, le Ruby, ou Scala.

Pour plus d'informations à propos de l'environnement d'exécution Java, consultez L'Environnement d'Exécution Java (JRE) (Lien8).

2.3. L'Environnement d'Exécution Python

Avec l'environnement d'exécution Python App Engine, vous pouvez implémenter votre application en utilisant le langage de programmation Python, et l'exécuter dans un interpréteur Python optimisé. App Engine comprend des API riches et des outils pour le développement d'applications web en Python, incluant des API de modélisation de données riches, un framework d'application web simple d'utilisation et des outils pour manager et accéder aux données de votre application. Vous pouvez aussi profiter d'une grande variété de librairies matures et de frameworks pour le développement d'applications web en Python, tel que Django (Lien9).

L'environnement d'exécution Python utilise Python version 2.5.2. Le support additionnel de Python 3 est envisagé lors d'une prochaine mise à jour.

L'environnement Python inclut la librairie standard Python (Lien10). Bien sûr, toutes les caractéristiques de la librairie ne fonctionnent pas dans l'environnement sécurisé. Par exemple, un appel à une méthode qui essaye d'ouvrir une socket ou d'écrire un fichier générera une exception. Par commodité, plusieurs modules de la librairie standard dont les caractéristiques de base ne sont pas supportées par le système d'exploitation ont été désactivées et le code qui les importe provoquera une erreur.

Le code applicatif écrit pour l'environnement Python doit être écrit exclusivement en Python. Les extensions écrites en langage C ne sont pas supportées.

L'environnement Python fournit des API Python riches pour le datastore, Google Accounts, le rapatriement d'URL et les services de messagerie (non traduit). App Engine fournit aussi un framework simple d'applications web en Python nommé webapp afin de faciliter le démarrage de votre application.

Vous pouvez charger des librairies tiers dans votre application, tant qu'elles sont implémentées en Python pur et qu'elles ne nécessitent aucun module non supporté de la librairie standard.

Pour plus d'informations à propos de l'environnement d'exécution Python, consultez The Python Runtime Environment (Lien11).

2.4. La base de données ("Datastore")

App Engine fournit un service performant de stockage distribué de données qui comprend un moteur de recherche et la gestion des transactions. En même temps que les serveurs web distribués augmentent avec votre trafic, la base de donnée grossit avec vos données.

Le datastore App Engine n'est pas comme une base de données relationnelle traditionnelle. Les objets de données ou "entités" ont des propriétés. Les recherches peuvent retourner des entités selon un filtre donné et les trier selon les propriétés de leurs valeurs. Les valeurs de ces propriétés peuvent être l'un des types de valeur de propriété supportés.

Les entités Datastore n'ont pas de schéma. La structure de données des entités est fournie et mise en place par le code de votre application. Les interfaces JDO/JPA et l'interface

Python datastore incluent les dispositifs de mise en place de la structure à l'intérieur de votre application. Votre application peut aussi accèder au datastore directement pour mettre en place une structure répondant plus précisement à vos besoins.

Le datastore est fortement consistant et utilise le contrôle d'accès simultané optimiste. Une mise à jour de l'entité se déroule dans une transaction qui est réessayée un nombre de fois fixe si d'autres applications tentent de mettre à jour la même entité simultanément. Votre application peut exécuter plusieurs opérations concernant la base de données dans une seule transaction qui, soit réussit entièrement, soit ne réussit pas du tout, assurant l'intégrité de vos données.

Le datastore implémente les transactions à travers son réseau distribué en utilisant des "groupes d'entités". Une transaction manipule des entités à l'intérieur d'un simple groupe. Les entités du même groupe sont enregistrées ensemble par souci d'efficacité d'exécution des transactions. Votre application peut attribuer des entités à des groupes quand les entités sont créées.

2.5. Les comptes Google ("Google Accounts")

App Engine permet à une application d'intéragir avec l'authentification des comptes utilisateurs Google. Votre application peut autoriser un utilisateur à se connecter avec un compte Google, pour accéder à son email et au nom affichable associé au compte. En utilisant les comptes Google, l'utilisateur utilise votre application plus rapidement car il n'a pas besoin de créer un nouveau compte. Cela vous évite également les travaux d'implémentation d'un système d'authentification juste pour votre application.

Si votre application tourne sur Google Apps, elle peut utiliser les mêmes dispositifs avec les membres de votre organisation qu'avec les comptes Google Apps.

L'API des utilisateurs (Users API) peut aussi dire à l'application si l'utilisateur actuel est un administrateur déclaré pour cette application. Cela vous permet facilement de gérer une zone de votre site réservée aux administrateurs.

Pour plus d'informations à propos de l'intégration de Google Accounts, consultez "The users API reference" (Lien12).

2.6. Les services App Engine

App Engine fournit une variété de services qui vous permettent d'exécuter des opérations communes pour manager votre application. Les API suivantes sont fournies pour accéder à ces services :

2.6.1. Rappatriement d'URL ("URL Fetch")

Les applications peuvent accéder aux ressources sur Internet, tels que les services web ou d'autres données, en utilisant le service de rappatriement d'URL App Engine. Le service de rappatriement d'URL récupère des ressources web en utilisant la même infrastructure à haute-vitesse de Google que celle qui récupère les pages web pour beaucoup de produits Google.

2.6.2. Messagerie

Les applications peuvent envoyer des courriers électroniques en utilisant le service de messagerie App Engine. Le service de messagerie utilise l'infrastructure de Google pour envoyer des courriers électroniques.

2.6.3. Memcache

Le service Memcache fournit à votre application un cache mémoire de clés-valeurs à haute performance qui est accessible par plusieurs instances de votre application. Memcache est utile pour les données qui n'ont pas besoin de persistance et des caractéristiques transactionnelles du datastore, telles que les données temporaires ou les données copiées depuis le datastore vers le cache pour des accès très rapides.

2.6.4. Manipulation d'images

Le service Image permet à votre application de manipuler des images. Avec ses API, vous pouvez redimensionner, rogner, tourner et inverser des images aux formats JPEG et PNG.

2.6.5. Tâches planifiées

Le service Cron vous permet de planifier des tâches à exécuter à des intervalles réguliers. Pour plus d'informations, consultez les documentations Cron pour Python (Lien13) ou Java (Lien14).

3. Processus de développement

Les kits de développement logiciel App Engine (SDK) (Lien15) pour Java et Python incluent chacun un serveur d'applications web qui émule tous les services App Engine sur votre ordinateur local. Chaque SDK inclut toutes les API et librairies disponibles sur App Engine. Le serveur web simule également la sécurisation et l'indépendance de l'environnement, incluant les vérifications d'essais d'accès aux ressources du système non autorisées dans l'environnement d'exécution App Engine.

Chaque SDK inclut aussi un outil pour charger son application sur l'App Engine. Une fois que vous avez créé le code de votre application ainsi que les fichiers statiques et les fichiers de configuration, vous pouvez exécuter l'outil pour charger vos données. L'outil vous demande l'adresse email et le mot de passe de votre compte Google.

Quand vous effectuez une modification majeure à une application qui fonctionne déjà sur App Engine, vous pouvez mettre en ligne la modification en tant que nouvelle version. L'ancienne version continuera de servir les utilisateurs jusqu'à ce que vous passiez à la nouvelle version. Vous pouvez tester la nouvelle version sur App Engine tandis que l'ancienne version continue de fonctionner.

Le SDK Java fonctionne sur n'importe quelle plateforme avec Java 5 ou Java 6. Le SDK est disponible en fichier Zip. Si vous utilisez l'environnement de développement Eclipse, vous pouvez utiliser le plugin Google pour Eclipse pour créer, tester et charger vos applications App

Engine. Le SDK inclut également des outils en ligne de commande pour exécuter le serveur de développement et charger votre application sur App Engine.

Le SDK Python est implémenté en Python pur, et s'exécute sur n'importe quelle plateforme possédant Python 2.5, telles que Windows, Mac OS X et Linux. Le SDK est disponible en fichier Zip, et les programmes d'installation sont disponibles pour Windows et Mac OS X.

La Console d'administration est une interface web pour gérer vos applications tournant sur App Engine. Vous pouvez l'utiliser pour créer de nouvelles applications, configurer des noms de domaines, sélectionner quelle version de votre application doit fonctionner, examiner les accès et les logs d'erreurs, et parcourir vos données du datastore via l'utilisation de l'outil de visualisation de données.

4. Limites et quotas

Non seulement la création d'une application App Engine est facile, mais c'est également gratuit ! Vous pouvez créer un compte et publier votre application que les visiteurs peuvent utiliser de suite sans coût, et sans engagement. Une application d'un compte gratuit peut utiliser jusqu'à 500 Mo de stockage et servir 5 millions de pages vues par mois. Quand vous êtes prêt pour plus, vous pouvez activer la facturation, spécifier votre budget journalier maximum, et allouer votre budget pour chacune des ressources selon vos besoins.

Vous pouvez publier jusqu'à 10 applications par compte.

Chaque application se voit allouer des ressources avec des limites ou "quotas". Un quota détermine pour chacune des ressources une limite que l'application peut utiliser durant une journée calendaire Dans un futur proche, vous pourrez ajuster quelques-uns de ces quotas en achetant des ressources additionnelles.

Quelques caractéristiques imposent des limites indépendantes des quotas afin de protéger la stabilité du système. Par exemple, quand une application est appelée par une requête web, elle doit fournir une réponse dans les 30 secondes. Si l'application prend trop de temps, le traitement est terminé et le serveur retourne un code d'erreur à l'utilisateur. Le délai d'expiration est dynamique, et peut être raccourci si un gestionnaire de requête atteint fréquemment cette limite afin de préserver les ressources.

Un autre exemple de limite de service est le nombre de résultats renvoyé par une recherche dans la base de données. Une recherche peut retourner au plus 1000 résultats. Celles qui devraient retourner plus de résultats en retourneront seulement 1000. Dans ce cas, une requête qui exécute une telle recherche devrait prendre plus de 30s pour s'exécuter et donc dépasser le temps autorisé. Cependant, cette contrainte permet de limiter les ressources utilisé par le datastore.

Les tentatives de corruption ou d'abus de quotas, telle que l'utilisation d'applications sur plusieurs comptes qui fonctionnent ensemble, sont une violation des Conditions Générales du Service (Lien16), et peut résulter à la désactivation des applications ou à la fermeture du compte.

Pour obtenir la liste des quotas et des explications sur le système de quota, y compris les quotas pouvant être augmentés par l'activation de la facturation, consultez Quotas (Lien17).

5. Pour plus d'informations...

Pour plus d'informations à propos de Google App Engine:

- Visionner la vidéo Campfire One (Lien18) ou lisez le compte-rendu (Lien19).
- Visionner la vidéo de la démo de App Engine (Lien20).
- Télécharger le SDK (Lien21), ouvrez un compte Google (Lien22), puis lisez le Guide de Démarrage Java (Lien23) et réalisez le tutoriel.
- Parcourez la App Gallery (Lien24) pour des exemples d'applications construites avec App Engine.
- Explorez le reste de la documentation App Engine.

Retrouvez l'article de brocoli en ligne : Lien25

Interview avec Alexis Moussine Pouchkine

A l'heure de la sortie de GlassFish v3, Alexis Moussine Pouchkine, ambassadeur du projet GlassFish et membre actif des forums NetBeans et GlassFish sur Developpez.com, a accepté une interview exclusive autour du serveur d'application et des différents sujets sur lesquels la communauté se questionne.

1. Glassfish v3

*** Pourrais-tu expliquer ce que sont HK2 et OSGi ?**

Commençons par OSGi. Il s'agit d'un système de modules pour Java né dans l'embarqué, popularisé par Éclipse et désormais relativement populaire côté serveur. Pour faire simple, un module est un ensemble de fonctionnalités, une version, des dépendances vers d'autres modules et un API publique exposée (le reste étant de l'ordre du détail d'implémentation). HK2 est en quelque sorte le cœur de GlassFish v3 et la clé de son extensibilité. C'est aussi une

couche d'abstraction au dessus d'OSGi (les développeurs de GlassFish v3 n'utilisent pas directement OSGi).

*** Pourquoi avoir utilisé ces technologies pour GlassFish v3 ?**

OSGi assure les fonctionnalités de système de module, permet d'intégrer un patrimoine de code OSGi existant dans GlassFish et de bénéficier des outils qui viennent avec les implémentations (console par exemple). Et puis cela permet de ne pas réinventer la roue en matière de système de module. GlassFish v3 utilise Apache Felix par défaut. HK2 permet de garder GlassFish v3 cohérent et extensible. Chaque module contribue à un serveur qui n'a qu'un seul fichier de configuration (domain.xml), un seul outil en ligne de commande (asadmin) et une seule console d'administration web. Il n'est pas question, par exemple, qu'un nouveau conteneur, un mécanisme de persistance ou une pile web services vienne avec son propre fichier de configuration. Ce serait vite ingérable pour l'utilisateur. HK2 permet également de changer le système de module si nécessaire. GlassFish fonctionne également sur Equinox, mais également dans un mode statique (sans OSGi). Si un nouveau système (par exemple intégré au JDK) apparaît, l'adaptation nécessaire dans GlassFish sera minimale. Sur ces deux sujets, le blog de Jérôme Dochez, architecte de GlassFish, est une bonne source d'informations (Lien26) (question "à la carte" traitée dans la question suivante).

*** Qu'apporte la v3 au niveau des conteneurs ?**

Dans GlassFish v3, un conteneur est un module. Il peut être désactivé, voire même absent. Les conteneurs les plus connus fournis par défaut sont : servlet, ejb, jax-rs (jersey), jms, jsf, jax-ws (metro), jca, et jpa (eclipselink). Les autres conteneurs disponibles sont surtout liés aux environnements et frameworks d'autres langages pour la JVM : Groovy/Grails, JRuby/Rails, Jython/Django... L'existence même de ces modules bien définis permet de fournir GlassFish via deux distributions : web ou complet (profils définis dans Java EE 6), mais aussi de faire son serveur à la carte. On peut ainsi créer une distribution hybride répondant uniquement aux besoins de l'application exécutée (et, par exemple, proposer quelque chose de très petit). J'ai réalisé à ce titre quelques petites vidéos (Lien27)

*** Tu as fait de multiples présentations de GlassFish v3, comment ressens-tu la communauté vis-à-vis de GlassFish ?**

Les retours sont souvent très bons. Certains retiennent l'aspect modulaire, d'autres les fonctions de redéploiement à chaud avec préservation de la session, d'autres encore le support de Java EE 6. Disons que le commentaire le plus fréquent c'est quelque chose comme : "dommage que ce ne soit pas plus connu" ou "pourquoi ne faites-vous pas plus de pub ?".

*** Quelle est (ou sont) la nouveauté la plus intéressante de GlassFish v3 par rapport à v2 ?**

J'en retiendrais trois : - modularité (OSGi et HK2 comme décrit plus haut) - support complet de Java EE 6 (cela ne vaut pas les nouveautés de Java EE 5, mais cela crée beaucoup d'intérêt après que ce dernier ait permis de remettre Java EE sur les rails). - l'expérience utilisateur/développeur : démarrage et redéploiement rapide, préservation des sessions lors de redéploiements, occupation mémoire proportionnelle à l'usage des modules. GlassFish c'est une peu de légèreté dans ce monde de brutes des serveurs d'applications Java EE !

*** Quels sont les éléments différenciateurs de GlassFish par rapport à d'autres serveurs d'applications ?**

Par rapport aux offres commerciales, il est Open Source. C'est évident, mais cela vaut le coup de le rappeler. Cela contribue largement à son adoption par une large communauté d'utilisateurs, voire dans certains cas de contributeurs.

Je pense que l'architecture (HK2) pensée par Jérôme donne à GlassFish un nouveau souffle technologique tout en étant très pragmatique. Le support des derniers standards (Java EE 6, JAX-RS), mais aussi de technos comme Comet et plusieurs environnements dynamiques (Rails, Grails, Django) sont des points assez uniques. On peut aussi citer la partie gestion de paquetage de GlassFish (techno IPS/pkg) qui permet de gérer les modules du serveur comme on gère les paquetages d'une distribution Linux.

Enfin, je pense que c'est un des rares produits susceptible de plaire à la fois aux développeurs et à une population de production. Cette propriété a souvent été un critère de choix important en faveur de GlassFish.

*** Les évolutions à venir après la v3 ?**

V3 est assez similaire à ce qui a été fait avec GlassFish v1: support complet de Java EE (5 à l'époque) mais en mode mono-instance. Le partage de charge est de la responsabilité de l'utilisateur (mod apache par exemple). GlassFish v3.1 sera donc focalisé sur l'intégration de l'administration centralisée qui existe aujourd'hui dans la 2.x, ainsi que sur l'intégration des solutions de clustering et de réplication. Cette version 3.1, prévue courant 2010, sera à équivalence fonctionnelle avec la famille v2.x et rajoutera des fonctionnalités permettant de faciliter le déploiement de GlassFish dans des environnements virtualisés de type "Cloud".

*** Java EE 6 et GlassFish, pourquoi utiliser GlassFish ?**

Parce que Java EE 6 est une version de maturité par rapport à Java EE 5 avec un bon équilibre entre nouveautés et améliorations à la marge, mais aussi parce que GlassFish n'est plus (depuis un certain temps d'ailleurs) uniquement une implémentation de référence, mais bien un produit Open Source capable de rivaliser en fonctionnalités et performances avec tous les autres acteurs de ce marché.

2. Autres sujets

*** Une des difficultés à imposer GlassFish en ce moment vient de l'incertitude de son évolution après le rachat par Oracle.**

La FAQ publiée par Oracle fin octobre 2009 devrait rassurer très largement sur l'avenir de GlassFish (Lien28). GlassFish est open Source, implémente Java EE 6 aujourd'hui, et plait aux développeurs ainsi qu'à la production. GlassFish continue d'attirer de nombreux nouveaux clients, y compris dans cette période de transition. GlassFish v3 est une architecture élégante de modularité qui lui apporte une nouvelle jeunesse. Il existe déjà de nombreuses collaborations techniques entre GlassFish et Weblogic : Metro, EclipseLink, JSF, etc.

*** Quelle est ta réaction vis-à-vis du rachat par Oracle ?**

J'ai clairement approfondi ma connaissance d'Oracle depuis l'annonce du rachat. C'est une société qui a en commun avec Sun une culture de l'engineering. Contrairement à Sun, Oracle n'a pas pris le virage de l'Open Source de manière aussi radicale. Cela dit, tous les rachats de technologies open source effectués (InnoDB et Sleepycat/BerkleyDB par exemple) sont restés Open Source (difficile de faire autrement), avec la même licence (c'est déjà plus parlant) et avec des ressources de développement renforcées. Oracle a même mis TopLink (issu également d'une acquisition et désormais nommé EclipseLink) en Open Source il y a quelques années.
Il y a une très belle opportunité pour Oracle d'intégrer dans ses rangs une équipe d'ingénieurs innovants et de leur donner des moyens de réussir, ce qu'Oracle a su déjà très bien faire en élargissant régulièrement son périmètre d'action. Je suis donc optimiste.

*** Comment vis-tu le succès des aquariums, et que comptes-tu faire pour les faire évoluer ?**

Un des principes de fonctionnement de l'équipe GlassFish c'est "Think globally, act locally" soit littéralement "Penser globalement, agir localement". C'est pour cette raison que Sun essaye d'organiser des rencontres régulières sur ses technologies. Cela dit, il existe beaucoup d'autres initiatives similaires (les JUGs en font partie) et il me parait important de savoir travailler ensemble pour produire des évènements de qualité. Pour ce qui est des réunions Aquarium, je suis content d'avoir fait participer des utilisateurs (clients, partenaires) aux dernières réunions. On obtient souvent un bon équilibre de contenu en combinant technos et retours d'expérience. A noter que le 10 décembre, l'institut de formation Demos, partenaire de Sun, organise une soirée pour le lancement de Java EE 6 et de GlassFish v3 (détails à venir).

*** Que penses-tu de JavaFX, d'Android...**

C'est intéressant (et assez inhabituel) de citer ces deux là dans la même question. Je pense qu'ils ont beaucoup de similitudes notamment par leur utilisation d'une machine virtuelle, ce qui est assez différent de l'initiative Open Web Foundation (HTML5...) dont Google est pourtant à l'initiative. JavaFX propose une exécution multipériphérique sur poste de travail et sur téléviseur en plus des téléphones mobiles. Pour le reste, l'adoption d'Android semble réelle et régulière.

*** EDI Sun/Oracle**

Cf. le même document référencé plus haut (Lien29)

*** Comment expliques-tu l'explosion tardive des JUGs en France ?**

Très honnêtement, je ne me pose pas la question, je savoure la création de chacun d'entre eux. C'est à la fois assez naturel de voir des gens se regrouper autour d'une technologie aussi répandue, mais aussi très encourageant de voir des salles aussi combles tous les mois près de 15 ans après la création de Java. Je me disais qu'en faisant la somme de tous les événements des JUGs en France sur une année, on approche doit s'approcher de l'audience de JavaOne.

*** Qu'aimerais-tu voir sur developpez.com qui manque actuellement ?**

Peut-être des sessions de discussion ou des présentations en direct, des événements ponctuels en quelque sorte. Je pense qu'il y a beaucoup de francophones qui travaillent sur des produits et des technologies intéressantes (glassfish et jboss sont de bons exemples) qui sont susceptibles de venir présenter leur travail. Je pense que cela renforcerait la notion de communauté de développez. Le sujet à traiter pourrait venir directement de la communauté developpez.com.

Retrouvez l'article de Mike François en ligne : Lien30

Les livres Java

Android

Développer des applications mobiles pour les Google Phones.

Cet ouvrage s'adresse à tous ceux qui désirent se lancer ou se perfectionner dans le développement d'applications « mobiles » sous Android, le nouvel OS mobile open source lancé par Google. Il a pour but d'être le guide concret et indispensable pour développer une application, depuis le téléchargement du SDK (Software Development Kit) jusqu'au déploiement de l'application sur le téléphone. Il commence par décrire le « contexte » dans lequel Android a été créé par Google et ses partenaires de l'Open Handset Alliance. Il fournit ensuite l'essentiel de ce qu'il faut connaître de son architecture logicielle, avant de passer à la pratique du développement. La construction d'une interface graphique adaptée aux terminaux tactiles à taille réduite est expliquée en détail. Les quatre composants Activity, Service, BroadcastReceiver et ContentProvider, qui sont les piliers d'Android, sont décrits et mis en oeuvre avec des exemples. Un chapitre est consacré à la persistance des données et un autre aux communications réseau. Les derniers chapitres portent sur les écrans tactiles, les GPS, APN et autres accéléromètres qui sont «

embarqués » dans les smartphones modernes.

Critique du livre par Mike François

217 pages ! À la première description du livre, 217 pages sur un thème qui n'a jamais été aussi présent dans toutes les discussions, on se dit que certaines notions sont survolées.
Détrompez-vous et courez chercher ce livre qui a, sans nul doute, la plus détaillée et la plus explicite des informations pour s'initier et développer avec le SDK de Google.

Des exemples, des captures d'écrans et des parties de code, nous aident à entrevoir les multiples notions abordées dans le sujet. Plus qu'un livre de chevet, un compagnon de voyage qui nous promet de ne pas voir le temps défiler tant les éléments et la description sont captivants. Venant de Java, que ce soit SE ou EE, ce livre est pour vous ! Bon nombre d'allusions ou de comparaisons aident les initiés de Duke à comprendre par analogie et méthodologie.

La structure globale est extrêmement agréable. Nous commençons chaque chapitre par un mini résumé des objectifs du chapitre et nous terminons par ce qu'il faut retenir.

Les titres de chaque partie résument avec exactitude le contenu. De l'explication détaillée d'Android à la vision de l'auteur sur le futur de la plateforme, chaque partie apporte des notions pertinentes et concrètes.

La première, voire la plus grande, motivation de l'achat de ce type de livre est de voir et comparer ce qu'apporte un système ou une technologie. L'auteur l'a bien compris et nous décrit, compare et résume les différents points qui font qu'Android pourrait devenir la première plateforme interopérable sur différents systèmes.

Pour le contenu du livre, comme toute lecture, nous attendons d'acquérir les connaissances nécessaires afin de pouvoir commencer, voire même développer en autonomie complète via les notions que nous venons d'acquérir.
Florent Garin a pensé aux différents cas de figure que nous pourrions rencontrer et il n'est pas rare de se dire "Et si je souhaite faire ça ?", et qu'en tournant la page, le cas soit évoqué. Il en va de même pour les différents cas auxquels nous ne pensons pas.

Enfin, comme on commence à en avoir l'habitude, nous pouvons disposer des bouts de code évoqués dans le livre. L'auteur a allégé le contenu en s'axant sur les parties essentielles et les points dont traite le chapitre. Cependant, vous pouvez accéder en ligne à toutes les sources pour tester, comparer et adopter la philosophie Android. Les différents tests des bouts de code m'amènent à dire qu'ils sont de très bonne qualité et on sent que l'auteur a testé différentes approches afin d'exposer celle qui correspond au mieux à la plateforme et qui apporte des éléments essentiels à cette dernière.

J'ai beaucoup apprécié le soin de l'auteur à expliquer les différentes notions comme le vocabulaire RPC qui permet, aux personnes les moins accoutumées au langage, de comprendre aisément les différents termes cités ou les différentes comparaisons. Aussi nous disposons d'une annexe fort utile décrivant les différentes commandes avec les différents attributs pour déployer en ligne de commande.

Au final, mon avis général est : sans nul doute... n'ayez pas de doute ! Si vous souhaitez développer sur cette plateforme, ce livre est le livre de référence que je vous recommande.

GWT

Créer des applications web interactives avec Google Web Toolkit (versions 1.7 et 2.0)

Cet ouvrage s'adresse à tous les développeurs Java qui souhaitent découvrir ce nouvel outil créé par Google pour le développement web. Il intéressera également tous ceux qui utilisent déjà AJAX et JavaScript et qui souhaitent enrichir leurs compétences. Le principe de GWT est simple : "coder en Java, compiler en JavaScript", et cette simplicité est aussi sa force. Ce livre est construit en deux parties. La première expose les bases qui permettent de comprendre GWT et de réaliser une application complète : vous y trouverez toutes les informations utiles sur des éléments tels que les widgets ou les mécanismes RPC de communication avec les serveurs. La deuxième explore les concepts avancés qui vous permettront d'élaborer des applications GWT plus sophistiquées, grâce aux possibilités d'internationalisation (i18n), d'envoi direct de requêtes HTTP, d'emploi de XML et JSON. Ce livre vous présente également les nouveautés et atouts de la version 2.0 de GWT : code splitting, OOPHM, UiBinder...

Critique du livre par benwit

Pour disposer d'un livre sur GWT dans leur langue, les développeurs francophones auront dû patienter trois ans. Cela peut paraître long mais cette attente s'avère finalement bénéfique. D'une part, Olivier Gérardin, l'auteur, a pu acquérir suffisamment d'expérience durant cette période sur divers projets réels pour maîtriser son sujet. D'autre part, GWT a gagné en maturité et c'est maintenant qu'il faut des ressources pour diffuser cette technologie au delà des early adopters. Le livre "GWT" d'Olivier Gérardin est en bonne voie pour être une de ces ressources.

Dans sa préface, Didier Girard note qu'il n'est jamais aisé de présenter une technologie car en général, on va soit trop loin soit pas assez. L'exercice est donc difficile mais il est vrai qu'Olivier a trouvé le bon équilibre.

Chapitre 1 : De HTML à GWT En exagérant à peine, Google fait croire que GWT va permettre à quiconque de développer une superbe application web 2.0. Il faut, avouons le, de bonnes bases et, trop souvent, je me rends bien compte dans les forums GWT qu'elles font défaut. Pour m'être essayé à l'exercice, il faut dire qu'il n'est pas facile de présenter toutes les notions indispensables sans craindre de perdre son lecteur. En choisissant l'approche historique, Olivier Gérardin y parvient brillamment.

Chapitre 2 : Hello, GWT Contrairement aux livres anglo-saxons qui souvent paraphrasent la doc d'installation de GWT par scripts de commande, celui-ci s'est mis au goût

du jour pour l'installation de GWT en utilisant le plugin de Google. Ce qui lui permet de passer plus de temps sur les différents fichiers d'un projet GWT et les liens qu'ils entretiennent entre eux. J'ai également bien aimé sa manière de créer un nouveau projet en privilégiant le standard "dynamic web project".

Chapitre 3 : Développer avec GWT Les différences entre le mode de développement et le mode de production sont bien expliquées ainsi que les contraintes sur le code Java. A recommander à tous ceux qui demandent pourquoi leur projet GWT ne compile pas !!!

Chapitre 4 : Widgets Une présentation des différents widgets GWT avec des exemples. J'ai bien aimé l'indication des styles css impliqués dans le rendu des composants; le débutant pourra regretter qu'il l'ait réservé aux widgets les plus complexes.

Chapitre 5 : Communiquer avec le serveur Un exemple concret avec toutes les étapes nécessaires. Un bon point également pour les chapitres sur la sérialisation et les exceptions.

Chapitre 6 : Internationalisation L'internationalisation "statique" n'aura plus de secret pour vous. On ressent bien la pratique de l'auteur car il y a plein de trucs et astuces. Pour ma part, je regrette que les solutions alternatives n'aient pas été développées (même si elles sont considérées comme moins propres par Google lui-même).

Chapitre 7 : Mécanismes avancés du compilateur Il est question ici de JSNI et de Deferred Binding, sujets très intéressants mais un peu court à mon goût. Ceci dit, il faut reconnaître que ces techniques ne concerneront qu'une minorité des cas d'utilisations de GWT.

Chapitre 8 : Le mécanisme d'historique de GWT Une fois ce chapitre lu, cela paraît tellement simple que j'aurais eu envie de l'utiliser un maximum si sa remarque de prudence (ô combien juste) n'était venue freiner mon enthousiasme.

Chapitre 9 : Envoyer des requêtes HTTP Un chapitre bien sympathique pour parler d'XML, de JSON et de la contrainte de sécurité "Same Origin Policy" et de son contournement.

Chapitre 10 : Créer ses propres Widgets Un panorama des différentes possibilités.

Chapitre 11 : Bibliothèques tierces Ce chapitre a l'intérêt de montrer qu'il existe également tout un écosystème GWT. Parce que pendant une période, elle fût le seul complément crédible et parce qu'elle met en perspective les deux autres librairies concurrentes, EXT-GWT et SmartGWT, il est compréhensible que GWT-EXT ait été abordée. Cependant, selon moi, l'auteur aurait dû en rester là et ne pas aborder son utilisation concrète à l'heure où elle est abandonnée et où des alternatives plus pérennes existent. C'est mon plus gros reproche, espérant que les débutants ne l'utiliseront pas. En revanche, j'ai apprécié le paragraphe sur les Google API en général et sur Google Gears en particulier. Maintenant que Chrome OS a été présenté, cette librairie gagne en intérêt.

Chapitre 12 : GWT 2.0 L'auteur n'oublie pas les principales modifications apportées par la dernière version, ce qui en fait à l'heure de cette critique le livre le plus à jour sur GWT ! Il se paye même le luxe d'expliquer comment récupérer la version 2.0, mais il y a plus simple puisque la Release Candidate est sortie le jour de la publication du livre.

En conclusion, si vous voulez découvrir GWT ou si certains des points précédents ont éveillé votre curiosité, je vous recommande ce livre.

Retrouvez ces critiques de livre sur la page livres Java : Lien31

Interopérabilité PHP / C# via WebService

Nous allons voir dans cet article ce qu'est un fichier WSDL, suivi par la création d'un WebService en PHP consommé par une application .NET en C# et l'inverse, la création d'un WebService C# (en WCF) consommé par une application PHP.

1. Introduction

Interopérabilité est peut-être un bien grand mot pour décrire ce qui va suivre. En effet, dès que l'on parle de WebService, il suffit d'avoir un fichier WSDL pour qu'on puisse consommer le WebService par une application d'un autre langage. Nous verrons donc dans un premier temps ce qu'est un fichier WSDL, puis la création d'un WebService en PHP consommé par une application .NET en C# et inversement.

2. Le fichier WSDL

Nous verrons juste une rapide présentation, pour plus d'informations, vous pouvez aller voir ici : Lien32.

Le fichier WSDL décrit une Interface (Lien33) publique d'accès à un Service Web (Lien34), notamment dans le cadre d'architectures de type SOA (Service Oriented Architecture) (Lien35). C'est une description fondée sur le XML (Lien36) qui indique « comment communiquer pour utiliser le service ». Il décrit notamment le Protocole de communication (SOAP RPC ou SOAP orienté message) (Lien37), le format de messages requis pour communiquer avec le service, les méthodes que le client peut invoquer ainsi que la localisation du service. Le fichier WSDL est un fichier XML qui commence par la balise <definitions> et qui contient les balises suivantes :

- <binding> : définit le protocole à utiliser pour invoquer le service web ;
- <port> : spécifie l'emplacement actuel du service ;
- <service> : décrit un ensemble de points finaux du réseau.

Les opérations possibles et les messages sont décrits de façon abstraite mais cette description renvoie à des protocoles réseaux et formats de messages concrets.

Il y a différents moyens de réaliser ce fichier, certains seront décrits dans cet article, mais étant donné que c'est du XML, un simple bloc note suffit !

3. WebService PHP consommé par une application C#

Pour créer un WebService facilement en PHP, il faut utiliser PHP5 avec l'extension SOAP activée. Pour cela, le mieux est de travailler en local avec Wamp Server (Lien38) (Wamp 5), qui permet d'activer cette extension.

Une fois le matériel prêt, on va commencer par ouvrir notre éditeur PHP préféré et écrire la classe qui va être utilisée par le WebService.

Dans notre exemple, on va juste définir une petite classe chargée de faire une opération et une soustraction :

PHP
```
class Math
{
    public function Add($a, $b)
    {
        return $a + $b;
    }

    public function Substract($a, $b)
    {
        return $a - $b;
    }
}
```

Voilà donc les méthodes qui seront exposées par le WebService.

Il ne reste plus qu'à créer le serveur et la partie PHP sera terminée. Pour cela, nous allons utiliser la classe SoapServer :

PHP
```
require 'Math.class.php';

// première étape : désactiver le cache wsdl lors de la phase de test
ini_set("soap.wsdl_cache_enabled", "0");

$wsdl= 'WebService.wsdl';
//Création du serveur SOAP avec le fichier WSDL
$server = new SoapServer($wsdl);
//Ajout de la classe Math dans les éléments proposés par le WebService
$server->setClass('Math');

//l'utilisation du WebService se fera toujours par la méthode POST
if ($_SERVER["REQUEST_METHOD"] == "POST")
{
    $server->handle();
}
else //dans le cas contraire, la page affichera les méthodes proposées par le WebService
{
    echo '<h4>Ce serveur SOAP peut gérer les fonctions suivantes : </h4><ul>';
    $functions = $server->getFunctions();
    foreach($functions as $func)
```

```
    {
        echo '<li>'.$func.'</li>';
    }

    echo '</ul>';
}
```

Vous avez sûrement remarqué qu'on fait appel au fichier WebService.wsdl alors qu'on ne l'a pas encore créé. On va donc s'en charger maintenant. Le mieux pour cela est d'utiliser un logiciel qui va générer le fichier wsdl final grâce à des informations qu'on lui aura apportées. Pour ma part j'utilise Altova XML Spy (Lien39) qui est disponible gratuitement en version d'essai. Voilà donc ce que ça donne en version graphique :

La partie serveur est à présent terminée, attaquons-nous au client .net en C#.

L'utilisation de Visual Studio 2008 est conseillée, il permet la génération automatique d'un proxy grâce à WCF qui va travailler à partir du fichier WSDL.

On va commencer par créer une application console. Ensuite, il suffit d'ajouter un Service Reference en passant comme adresse celle du fichier WSDL (http://localhost/WebServices/WebService.wsdl dans notre cas) et Math comme namespace pour la référence.

WCF a alors généré la classe MathServiceClient ainsi qu'un fichier de configuration qui vont nous servir à interagir avec le WebService.

On va tout d'abord instancier cette classe :

C#
```
MathServiceClient math = new MathServiceClient();
```

Puis on va simplement appeler nos deux méthodes :

C#
```
Console.WriteLine(math.Add(5, 5));
Console.WriteLine(math.Substract(10, 5));
```

Et bien sûr, il ne faut pas oublier de fermer le client :

C#
```
math.Close();
```

Et voilà, notre client .net consomme notre WebService PHP !

4. WebService C# en WCF consommé par une application PHP

Je ne vais pas détailler WCF ici, nous allons juste voir la mise en place d'un service via une application console. Si vous souhaitez en savoir plus sur WCF, voilà l'excellent article de Julien Corioland. (Lien40)

Pour faire rapide, pour un service WCF il est préférable d'avoir 3 projets différents : une bibliothèque de classes qui contiendra une interface qui servira de « contrat » pour le service, une bibliothèque de classe qui contiendra une classe qui implémentera le « contrat », et dans le cas présent, une application console qui va servir de serveur et qui va donc exposer le service.

Le service exposera les mêmes méthodes que dans le post précédent, c'est-à-dire une addition et une soustraction d'entiers.

Après avoir créé une première bibliothèque de classe nommée ServiceContract par exemple, il faut définir l'interface du service :

C#
```
[ServiceContract]
public interface IService
{
    [OperationContract]
    public int Add(int a, int b);

    [OperationContract]
    public int Substract(int a, int b);
}
```

L'attribut ServiceContract définit le contrat du service qui sera exposé. Il possède une propriété Namespace qui permet de changer le namespace du service (tempuri.org par défaut).

L'attribut OperationContract définit les méthodes visibles du service.

Ensuite, il faut créer la deuxième bibliothèque de classes qui va contenir l'implémentation du service :

C#
```
public class Math : IService
{
    #region IService Members

    public int Add(int a, int b)
    {
        return a + b;
    }

    public int Substract(int a, int b)
    {
        return a - b;
    }

    #endregion
}
```

Jusque là, rien de bien compliqué, il suffit d'implémenter l'interface définie plus haut. Passons maintenant au serveur, qui exposera le service WCF.

Le serveur sera une application console des plus classiques. La première chose à faire est de créer un fichier de configuration (App.config) qui va paramétrer le type, l'adresse et le protocole du service.

XML
```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <system.serviceModel>

    <services>
<!-- le type du service  -->
      <service name="ServiceImplementation.Math">
        <host>
          <baseAddresses>
<!-- l'adresse où le service sera accessible  -->
            <add
baseAddress="http://localhost:1664/MathService"/>
          </baseAddresses>
        </host>
        <!-- le protocole pour joindre le service
-->
        <endpoint address=""
        binding="basicHttpBinding"
        contract="ServiceContract.IMathService" /
>
      </service>
    </services>

  </system.serviceModel>
</configuration>
```

On crée ensuite un host en lui indiquant le type que l'on veut exposer en service :

C#
```
ServiceHost host = new
ServiceHost(typeof(ServiceImplementation.Math));
```

Il suffit ensuite de mettre le service en écoute :

C#
```
host.Open();

Console.WriteLine("Le service Math est à
l'écoute");

Console.WriteLine("Appuyez sur une touche pour
quitter");

Console.ReadLine();
host.Close();
```

Le service est maintenant prêt.

Mais le fichier WSDL n'a pas été créé, ce qui manque cruellement pour pouvoir consommer le service avec une application PHP. Heureusement, WCF permet de le générer automatiquement. Pour cela, il suffit de modifier le fichier de configuration.

On va tout d'abord rajouter un comportement au service. Dans la balise System.ServiceModel, on rajoute ceci :

XML
```
<behaviors>
    <serviceBehaviors>
        <behavior name="MathServiceBehaviors" >
            <serviceMetadata
httpGetEnabled="true" />
        </behavior>
    </serviceBehaviors>
</behaviors>
```

Il faut ensuite modifier la balise service comme suit :

XML
```
<service name="ServiceImplementation.Math"
behaviorConfiguration="MathServiceBehaviors">
```

Et voilà, notre service peut maintenant être consommé par n'importe quelle application via son fichier WSDL.

Il ne reste plus qu'à faire le client PHP. Pour cela, le mieux est de générer le client SOAP à partir du fichier WSDL. Je me suis personnellement servi de wsdl2php, disponible ici : Lien41, mais d'autres scripts font ça très bien aussi, comme WSDLInterpreter (Lien42).

Une fois le WSDL passé à la moulinette, on obtient donc une jolie classe Math qui hérite de SoapClient et qui possède deux fonctions : Add et Substract.

D'autres classes de deux types différents sont également générées :

- les classes qui vont servir à passer les paramètres : Add qui contient les deux entiers pour faire l'addition et Substract qui contient également deux entiers pour la soustraction.
- les classes qui vont contenir la réponse du WebService : AddResponse qui possède une propriété AddResult qui est le résultat de l'addition, et SubsctractResponse qui possède une propriété SubstractResult qui est le résultat de la soustraction.

Toutes ces classes sont générées dans un seul fichier au nom du service : Math.php.

Maintenant qu'on a de quoi consommer le service WCF, il ne reste plus qu'à créer le client PHP. On va tout d'abord créer le fichier client.php. Première chose à ajouter, un include sur le fichier généré par wsdl2php : Math.php.

Ensuite, on va instancier le client du service Math :

PHP
```
$client = new Math();
```

Il faut maintenant créer les paramètres pour la fonction Add :

PHP
```
$addParameters = new Add();
$addParameters->a = 2;
$addParameters->b = 2;
```

Vous remarquerez que a et b correspondent aux noms des paramètres de la fonction Add définis par le contrat du service.

On appelle la fonction Add :

PHP
```
$result = $client->Add($addParameters);
```

On peut ensuite utiliser la propriété AddResult de $result pour récupérer le résultat de l'addition et l'afficher par exemple :

PHP
```
echo $result->AddResult;
```

Et c'est fini, nous avons maintenant un client PHP qui consomme notre service WCF !

5. Conclusion

Vous savez maintenant comment créer et utiliser un client et un serveur en PHP et en WCF afin de faire communiquer vos applications.

Retrouvez l'article de Jérémie Bertrand en ligne : Lien43

Les dernières news

Configurer plusieurs environnements pour une application CakePHP

Parce qu'il est parfois utile de pouvoir développer son application à partir de son serveur de dev et de le balancer sur son serveur de prod en quelques clics sans avoir à faire attention que les logs de la base soient les bons, que les niveaux de debug, sécurité, cookies soient les bons.

Je vous propose ce que j'ai réussi à faire à partir du taff des autres, il sera toujours temps d'en discuter pour voir ce qui reste le plus optimal :

En premier lieu nous allons récupérer la classe de Rafael Bandeira (Lien44) qui va nous permettre de jouer avec des "environnements", la voila :

APP/Config/environment.php
```
/**
 * Singleton class to handle environment specific configurations.
 *
 * Auto-detect environment based on specific configured params and
 * allow per environment configuration and environment emulation.
 *
 * Environment. Smart Environment Handling.
 * Copyright 2008 Rafael Bandeira - rafaelbandeira3
 *
 * Licensed under The MIT License
 * Redistributions of files must retain the above copyright notice.
 */
class Environment {

  public $environments = array();

  protected $_configMap = array(
    'security' => 'Security.level'
  );

  protected $_paramMap = array(
    'server' => 'SERVER_NAME'
  );

  static function &getInstance() {
    static $instance = array();
    if (!isset($instance[0])) {
      $Environment = 'Environment';
      if (config('app_environment')) {
        $Environment = 'App' . $Environment;
      }
      $instance[0] = new $Environment();
      Configure::write('Environment.initialized', true);
    }
    return $instance[0];
  }

  static function configure($name, $params, $config = null) {
    $_this = Environment::getInstance();
    $_this->environments[$name] = compact('name', 'params', 'config');
  }

  static function start($environment = null) {
    $_this =& Environment::getInstance();
    $_this->setup($environment);
  }

  static function is($environment) {
    $_this =& Environment::getInstance();
    return ($_this->name === $environment);
  }

  public function __construct() {
    if (Configure::read('Environment.initialized')) {
      throw new Exception('Environment can only be initialized once');
      return false;
    }
  }

  public function setup($environment = null) {
    if (empty($environment)) {
      foreach ($this->environments as $environment => $config) {
        if ($this->_match($config['params'])) {
          break;
        }
      }
    }
    $config = array_merge(
      $this->environments[$environment]['config'],
      array('Environment.name' => $environment)
    );
    foreach ($config as $param => $value) {
      if (isset($this->_configMap[$param])) {
        $param = $this->_configMap[$param];
      }
      Configure::write($param, $value);
    }
  }

  protected function _match($params) {
    if ($params === true) {
      return true;
    }
    foreach ($params as $param => $value) {
      if (isset($this->_paramMap[$param])) {
        $param = $this->_paramMap[$param];
      }
```

```
      if (function_exists($param)) {
        $match = call_user_func($param, $value);
      } else {
        $match = (env($param) === $value);
      }
      if (!$match) {
        return false;
      }
    }
  return true;
  }
}
```

Cette classe va juste permettre de switcher les variables d'environnements, on pourrait allègrement faire plus léger mais pour l'instant on s'en contentera (on pourra l'alléger plus tard tant qu'à faire). Vous allez la mettre dans un fichier **environment.php** que vous collerez dans le répertoire **APP/Config**

Nous allons créer un répertoire "environments" dans le répertoire **APP/Config** et y créer un fichier que vous nommerez : **envs_list.php**

Dans ce fameux fichier vous allez définir les environnements dont vous avez besoin (oui car nous sommes organisés)

APP/Config/environments/envs_list.php

```
<?php
 Environment::configure(
   'development'
   ,array('server' => 'development')
   ,array('debug' => 2, 'security' => 'low')
  );
 Environment::configure(
   'production'
   ,array('server' => 'production')
   ,array('debug' => 0, 'security' => 'high')
  );
```

Jusque là rien de grave, mais il va falloir quand même appeler les fameux fichiers, donc dans le bootstrap.php nous allons ajouter les lignes suivantes :

APP/Config/Bootstrap.php

```
$configsDir    = APP . 'config' ;
$envsDir    = APP . 'config' . DS .
'environments';
$configFile = $configsDir . DS .
'environment.php';
$envsFile = $envsDir . DS . 'envs_list.php';


if ( file_exists( $configFile ) ){
  include_once( $configFile );
  if (file_exists( $envsFile )){
    include_once( $envsFile );
    Environment::start($_SERVER['APP_ENV']);
  }
}
```

à cette étape il vous manque encore un tout petit truc mais très gênant si on ne le configure pas : la variable APP_ENV, nous allons la définir dans le .htaccess de plus haut niveau (à vrai dire vous pouvez le définir dans n'importe quel .htaccess, seulement il faut le sortir le plus possible de la foule de répertoires afin de ne pas l'écraser lors des mises à jour des serveurs, sinon tout ceci perd de son intérêt)

.htaccess:

```
SetEnv  APP_ENV "development"
```

Voila vous pouvez faire quelques tests, tout devrait marcher, si vous avez page blanche, vérifiez bien les noms des environnements.

Maintenant sachez qu'avec cette classe, vous ne pouvez pas **switcher les configs de la base de données**.

Nous allons donc modifier quelque peu le fichier database.php en lui ajoutant un petit switch, je vous montre la classe complète, histoire que vous ne vous perdiez pas (j'ai piqué une partie de l'idée à Joel Moss (Lien45))

APP/Config/database.php

```
class DATABASE_CONFIG {

  var $development = array(
    'driver' => 'mysql',
    'persistent' => false,
    'host' => 'localhost',
    'login' => 'login',
    'password' => 'pass',
    'database' => 'localhost',
    'prefix' => '',
    'encoding'=>'utf8'
    );

  var $production = array(
    'driver' => 'mysql',
    'persistent' => false,
    'host' => 'localhost',
    'login' => 'login_user',
    'password' => 'pass_pass',
    'database' => '192.168.1.200', //Un autre
serveur quoi :)
    'prefix' => '',
    'encoding'=>'utf8'
    );

  var $default = array();

  function __construct()
  {
    if (isset($_SERVER['APP_ENV']))
      $this->default = $this-
>$_SERVER['APP_ENV'];
    else
      $this->default = $this->developpement;
//Metez en une par défaut au cas où
  }
  function DATABASE_CONFIG()
  {
    $this->__construct();
  }
}
```

Et voilà, tout devrait fonctionner au poil, je laisse le topic à dispo pour toute amélioration, je pourrais éventuellement en faire un article si ça interesse les gens .

Testé en 1.2.5 stable, attention toutefois, le fichier bootstrap.php est écrasé lors des mises à jour.

Commentez cette news en ligne : Lien46

ZendFramework 2.0 roadmap, que va-t-il se passer ?

La roadmap des spec de ZF2 est sortie.

Vous pouvez la consulter ici (Lien47). Elle est ouverte à suggestions (commentaires), ou sur la mailing list zf-contributors.

Petits rappels :

- ZF2.0 est prévue fin 2010 début 2011, la compatibilité sera cassée (parfois en profondeur) ;
- ZF2.0 tout comme Symfony2.0 (prévue fin 2010 aussi) sera *PHP5.3 only*. Il va donc falloir migrer, ce qui ne représente pas un problème majeur tant les cassures de compatibilités de PHP5.3 sont minimes au regard du gain en performance et en fonctionnalités qu'apporte ce langage que j'utilise officiellement maintenant depuis 1 mois.

Commentez cette news en ligne : Lien48

Introduction à la programmation d'une extension Google Chrome

Ce tutoriel a pour but d'initier au développement d'une extension pour Google Chrome à l'aide d'un exemple simple.

1. Introduction

Ce tutoriel a pour but d'avoir une version française et améliorée (si possible) de celui de Google (Lien49).

1.1. Connaissances nécessaires

Pour comprendre ce tutoriel, il est nécessaire de connaître :

- Les bases du langage HTML
- Les bases du langage Javascript

1.2. Préparer l'environnement de travail

Pour développer une extension pour Google Chrome vous aurez besoin de :

- Une version de Google Chrome permettant l'utilisation d'extensions (>= 4.0.429)
- Un répertoire sur votre disque dur
- Un éditeur de texte

2. Créer une extension

Le but de ce tutoriel va être de créer une extension pas-à-pas. La finalité sera un bouton ajouté à Google Chrome qui affichera un popup sous forme d'info-bulle contenant des liens vers 4 favoris sur lesquels on pourra cliquer pour accéder à la page.

2.1. Le commencement

Sur votre disque dur, créez un répertoire dans lequel vous allez éditer un fichier nommé **manifest.json**. Il va contenir les informations de l'extension.

- **name** : Son nom (indispensable)
- **version** : Son numéro de version (indispensable)
- **description** : Sa description (optionnelle)

Ces informations seront codées sous cette forme :

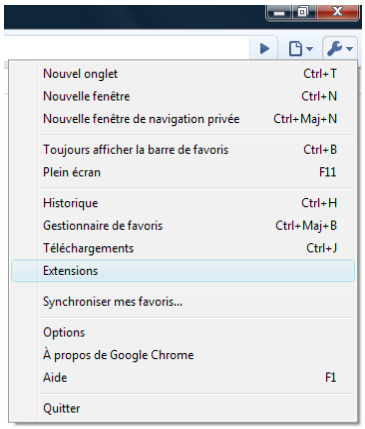
```
{
  "name": "Favoris Rapide",  // Nom
  "version": "1.0",  // Version
  "description": "Accéder à 4 favoris rapidement"
// Description
}
```

Pour réaliser des commentaires, il est nécessaire de mettre //, toute la suite de la ligne sera ignorée.

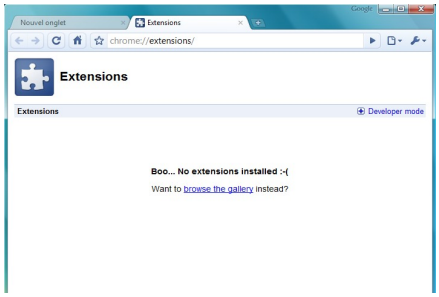
Comme on peut le constater, la syntaxe est très simple. Les paramètres comme les valeurs doivent être mis entre guillemets. Il faut faire attention à ne pas oublier les virgules entre chaque paramètre.

2.2. Installation de l'extension

Une fois le fichier enregistré, allez dans le panneau des extensions de Google Chrome :



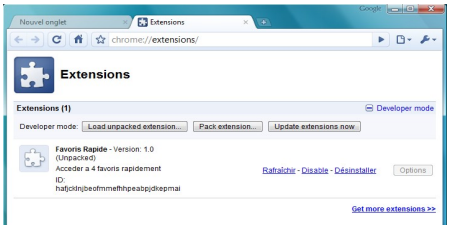
Ce qui va ouvrir un onglet avec le contenu suivant :



En cliquant sur **Developer mode**, vous accèderez à certaines fonctions aidant au développement d'extension :

- **Load unpacked extension** : charger une extension contenue dans un répertoire ;
- **Pack extension** : créer un package d'extension ;
- **Update extensions now** : mise à jour de toutes les extensions.

Cliquez sur **Load unpacked extension**, puis indiquez votre répertoire de travail pour charger votre extension, ce qui aura pour effet de la rajouter à la liste.



On voit donc que les informations sont bien affichées.

2.3. Mise en place de l'icône

Nous allons donc maintenant ajouter une icône à Google Chrome. Pour faire cela, il faudra ajouter un paramètre **browser_action** à notre fichier **manifest.json** en spécifiant certaines informations :

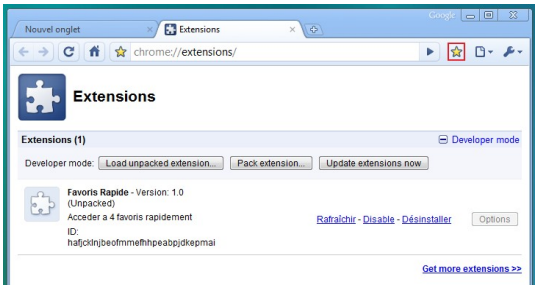
- **default_icon** : chemin de l'icône qui sera affiché (indispensable) ;
- **default_title** : titre affiché lors du survol de la souris sur l'icône (optionnel) ;
- **popup** : page qui pourra être ouverte à la façon d'une info-bulle lorsque l'utilisateur clique sur l'icône (optionnel).

Google conseille d'utiliser une icône de 19x19 pixels. Vous pouvez l'enregistrer sous différents formats. Pour ce tutoriel, il faudra créer une icône quelconque et la nommer icon.png.

En ajoutant ces informations au fichier **manifest.json**, nous obtenons :

```
{
  "name": "Favoris Rapide",
  "version": "1.0",
  "description": "Accéder à 4 favoris
rapidement",
  "browser_action": {
    "default_icon": "icon.png",  // Icone de
l'extension
    "default_title": "Favoris Rapide"  // Titre
affiche sur le titre
  }
}
```

Pour voir le changement, il sera nécessaire de retourner à la liste des extensions puis de cliquer sur **Rafraîchir**, ce qui aura donc pour effet d'ajouter l'icône à Google Chrome.



2.4. Le popup

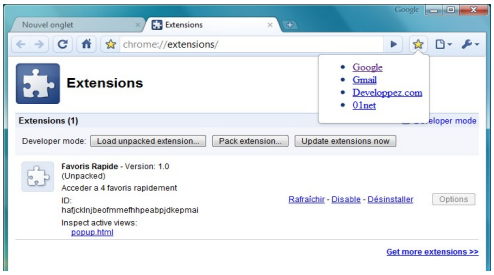
Maintenant que l'icône est en place, nous allons rajouter une page qui va s'ouvrir sous forme d'info-bulle sous le bouton. Cette page doit être écrite au format HTML. Ce fichier va se nommer, pour l'exemple, **popup.html** et contiendra pour le moment une simple liste avec des liens :

```
<html>
  <head>
    <meta http-equiv="Content-Type"
content="text/html; charset=UTF-8" />
  </head>
  <body>
    <ul>
      <li><a
href="http://www.google.fr">Google</a></li>
      <li><a
href="http://mail.google.fr">Gmail</a></li>
      <li><a
href="http://www.developpez.com">Developpez.com</
a></li>
      <li><a
href="http://www.01net.com">01net</a></li>
    </ul>
  </body>
</html>
```

Un banal fichier HTML respectant les standards. Cependant il faudra modifier le fichier **manifest.json** pour lier le bouton au fichier **popup.html** :

```
{
  "name": "Favoris Rapide",
  "version": "1.0",
  "description": "Accedér à 4 favoris
rapidement",
  "browser_action": {
    "default_icon": "icon.png",
    "default_title": "Favoris Rapide",
    "popup": "popup.html"
  }
}
```

Une fois de plus, cliquez sur **Rafraîchir** pour avoir le résultat suivant :



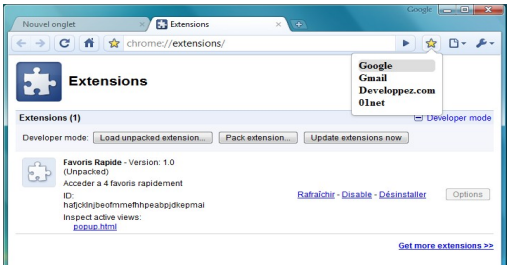
Il est très simple d'embellir ce "popup" en utilisant un peu de CSS. Pour ce tutoriel, il sera plus facile de le mettre dans le même fichier, il faudra donc insérer ce code entre les balises **<head>** :

```
<style>
  ul, li
  {
    list-style: none;
    padding: 0;
    margin: 0;
  }

  a
  {
    text-decoration: none;
    font-weight: bold;
    font-size: 16px;
    display: block;
    color: black;
  }

  a:hover
  {
    background-color: #DDDDDD;
    -webkit-border-radius: 5px;
  }
</style>
```

Enfin la possibilité d'utiliser les standards CSS sans avoir

à se soucier du résultat sous Internet Explorer

Ceci nous donne un résultat plus convenable :



2.5. Interaction avec Google Chrome

Le popup et les liens sont en place mais pourtant, le fait de cliquer sur les liens ne produit aucun effet. Il va falloir utiliser le javascript pour interagir avec Google Chrome. Nous allons donc modifier le fichier **popup.html** en ajoutant un évènement **onclick** sur tous les liens :

```
<li><a href=""
onclick="open_tab('http://www.google.fr')">Google
</a></li>
<li><a href=""
onclick="open_tab('http://mail.google.com')">Gmai
l</a></li>
<li><a href=""
onclick="open_tab('http://www.developpez.com')">D
eveloppez.com</a></li>
<li><a href=""
onclick="open_tab('http://www.01net.com')">01net<
/a></li>
```

Puis nous allons écrire la fonction **open_tab** :

```
function open_tab(url)
{
  chrome.tabs.create({"url": url});
}
```

Ici, on utilise la fonction **create** du module **chrome.tabs** permettant l'utilisation des onglets. Elle prend en paramètre un objet **createProperties** dont les attributs sont :

- **windowsId** : identifiant de la fenêtre ;
- **index** : position dans les onglets ;
- **url** : URL de la page internet à ouvrir ;
- **selected** : False si le nouvel onglet ne doit pas être sélectionné.

Les informations sur les différentes APIs de Google Chrome sont accessibles à la page suivante : Liens50

Tous ces paramètres sont optionnels. C'est pourquoi seule l'URL a été utilisée dans l'exemple. Cependant l'accès à ce module doit être indiqué dans le fichier **manifest.json :**

```
{
  "name": "Favoris Rapide",
  "version": "1.0",
  "description": "Accéder à 4 favoris
rapidement",
  "browser_action": {
    "default_icon": "icon.png",
    "default_title": "Favoris Rapide",
    "popup": "popup.html"
  },
  "permissions": ["tabs"] // Ajoute la permission
au module des onglets
}
```

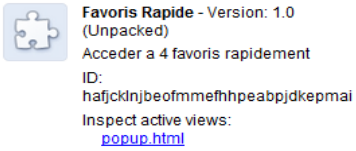
Voici le listing final de **popup.html** :

```
<html>
  <head>
    <meta http-equiv="Content-Type"
content="text/html; charset=UTF-8" />
    <style>
      ul, li
      {
        list-style: none;
        padding: 0;
        margin: 0;
      }

      a
      {
        text-decoration: none;
        font-weight: bold;
        font-size: 16px;
        display: block;
        color: black;
      }

      a:hover
      {
        background-color: #DDDDDD;
        -webkit-border-radius: 5px;
      }
    </style>
    <script type="text/javascript">
      function open_tab(url)
      {
        chrome.tabs.create({"url": url});
      }
    </script>
  </head>
  <body>
    <ul>
      <li><a href=""
onclick="open_tab('http://www.google.fr')">Google
</a></li>
      <li><a href=""
onclick="open_tab('http://mail.google.com')">Gmai
l</a></li>
      <li><a href=""
onclick="open_tab('http://www.developpez.com')">D
eveloppez.com</a></li>
      <li><a href=""
onclick="open_tab('http://www.01net.com')">01net<
/a></li>
    </ul>
  </body>
</html>
```

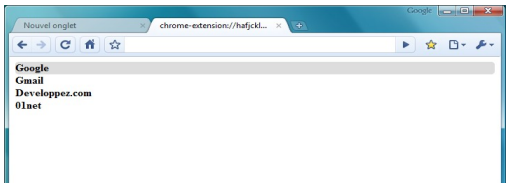
Voila, un dernier rafraîchissement et vous avez maintenant une extension opérationnelle que vous pouvez modifier à votre guise.

3. Amélioration et debuggage

Si vous désirez améliorer le fichier **popup.html** puis par la suite le debugger comme une page web, vous allez devoir l'ouvrir dans chrome en utilisant l'identifiant de l'extension que l'on trouve sur la page des extensions :



Dans cet exemple, l'identifiant est *hafjcklnjbeofmmefhhpeabpjdkepmai*, pour accéder à la page popup.html, il faudra se rendre à l'url : *chrome-extension://hafjcklnjbeofmmefhhpeabpjdkepmai/popup.html*.



Les modifications seront donc plus faciles à faire. Pour ouvrir le debugger, faites **Ctrl+Maj+I**. Pour la console javascript **Ctrl+Maj+J**.

4. Conclusion

Google Chrome permet de réaliser des extensions très facilement avec peu de connaissances. Au fil des versions, certaines informations pourront ne plus être valides mais si vous suivez ces changements, il sera aisé de mettre à jour vos extensions. N'hésitez pas à me transmettre vos suggestions pour enrichir et améliorer ce document notamment si vous trouvez des explications peu claires, des passages de code non optimisés, ou toute autre remarque. Elles seront les bienvenues.

Retrouvez l'article de Sylvain Dangin en ligne : Lien51

Quelle est la nouveauté la plus importante du HTML5 ?

Parmi la liste des 8 innovations majeures introduites par le nouveau standard

Le HTML5 introduit huit nouvelles fonctionnalités dont certaines risquent bien, si le standard est adopté par tous les navigateurs (Lien52), de révolutionner certaines applications Web.

On pense, bien évidemment, à la vidéo, avec les **nouvelles balises multimédias** (<audio> et <video>). Le support en natif des streamings risque de porter un coup fatal au Flash Player d'Adobe, déjà mis à mal par Silverlight de Microsoft. Cette nouvelle fonctionnalité permet par exemple de changer la taille d'une vidéo, de lui appliquer des filtres en temps réel, de naviguer beaucoup plus rapidement, d'y ajouter du texte, etc. Et ce, de manière étonnamment fluide.

Les utilisateurs de Chrome et de Safari peuvent d'ores et déjà visionner une démo de Youtube en HTML5 (Lien53).

La deuxième très grosse nouveauté, qui vient de pousser Google à abandonner Gears (Lien54), son outil de synchronisation de données en ligne et hors ligne, est la fonctionnalité de **stockage offline**.

Cette fonctionnalité repose sur ce que l'on pourrait appeler des "Super Cookies".

Ces cookies d'un nouveau type sont en fait un espace de stockage dans le navigateur pour les données de l'utilisateur mais aussi pour les applications en ligne type web-messsagerie, Google Docs (Lien55) ou Office 2010, une version très Cloud de la suite bureautique de Microsoft (Lien56). Le tout sans installer le moindre plug-in.

Plus ludique, **la balise <canvas>** permet de définir un espace sur une page pour y introduire un contenu (dessin, graphique, jeu, vidéo, etc.) avec lequel l'internaute peut interagir avec sa souris. Cette balise "tableau", vous paraît un peu floue : un exemple vaut mieux qu'un long discours (Lien57).

Là encore, le progrès tient énormément à l'absence d'installation de plug-in.

Moins voyant mais tout aussi important, le nouvel **attribut "contenteditable"** permet, comme son nom l'indique, l'édition de contenu, tout comme l'attribut "draggable" optimise **le glisser-déposer** si cher au Web 2.0.

Peu voyant également, mais très utile, une API qui permettra **l'accès à l'historique** et permettra aux pages Web de prévenir les problèmes de bouton "retour en arrière".

Les **outils de recherche** dans une page ne sont pas en reste non plus puisque les champs de saisie et toutes les applications qui en découlent bénéficieront d'une API qui permettra une meilleure interaction avec les autres éléments de la page, voire avec d'autres applications. Bref, pour faire très simple : des recherches plus efficaces pour l'utilisateur.

Enfin, la **géolocation** sera, elle aussi, grandement facilitée et répond à un besoin de plus en plus grand (Lien58), notamment en rapport avec les terminaux mobiles.

Dans la pratique, cela donne cette petite vidéo de présentation réalisée par Google, certainement le plus grand HTML5-évangéliste du moment (on le comprend, pour un fournisseur d'applications en ligne complexes, ce standard est une aubaine) : (Lien59)

Êtes-vous convaincus par le HTML5 ou pensez-vous qu'il ne s'imposera que très lentement, voire pas du tout ?

Et d'après vous, laquelle (ou lesquelles) de ces nouveautés vous paraît la plus importante ?

Source : HTML5 Demos and Examples (Lien60)

Commentez cette news en ligne : Lien61

Créer des coins arrondis à l'aide de Sprites CSS

Cet article est la traduction de : CSS Sprites + Rounded corners (Lien62). Retrouvez toutes les traductions de CSS Globe disponibles ici : Lien63.

Je sais qu'il y a des milliers de tutoriels concernant les coins arrondis réalisés uniquement en CSS. Je voudrais tout de même vous montrer cette méthode, en espérant qu'elle vous soit utile.

1. Introduction

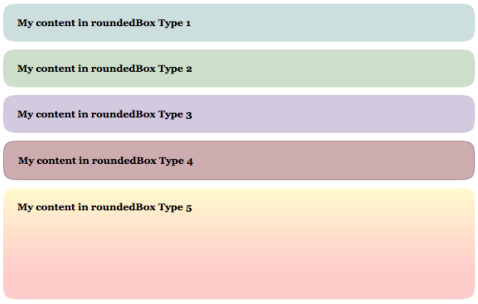
Je sais qu'il y a des milliers de tutoriels concernant **les coins arrondis réalisés uniquement en CSS**. Je voudrais tout de même vous montrer cette méthode, en espérant qu'elle vous soit utile.

Il est important de signaler que ce tutoriel présente une méthode avancée en ce qui concerne les CSS, mais je vais essayer de la rendre aussi simple que possible pour ceux qui débutent en CSS. **CSS 3** n'est pas encore abouti, donc jusqu'à ce que ce soit le cas, utilisons un code valide vis-à-vis du W3C.

Consultez la démonstration (Lien64) et téléchargez les sprites CSS et le code des coins arrondis (Lien65).

1.1. Qu'allons-nous faire ?

Ma version des coins arrondis consiste à créer un div conteneur, contenant 4 div à l'intérieur, positionnés de manière absolue, dans lesquels les images des coins arrondis sont constituées en utilisant un unique sprite CSS. Nous allons faire ceci :



1.2. Qu'est ce qui rend cette technique si cool ?

La possibilité de créer des éléments aux bordures arrondies avec une largeur et une hauteur fluide. Il n'y a pas de limites.

Cette technique, comme je l'ai indiqué plus tôt, est combinée avec les Sprites CSS (Lien66). Si vous ne savez pas ce que c'est ou comment les utiliser, lisez d'abord mon article précédent (Lien66). Vous avez compris les Sprites CSS ? Alors commençons !

2. Etape 1 : créons nos Sprites

1. Dans l'éditeur d'image de votre choix, créez un rectangle aux angles arrondis (je vais utiliser Fireworks pour cet exemple).

2. Découpez et exportez un des coins arrondis et sauvegardez-le dans un emplacement temporaire (nous allons le retourner plus tard)

3. Créez un nouveau document, importez votre coin, copiez le 3 fois et faites faire une rotation aux 3 copies pour obtenir les autres coins.

4. Créez une image avec les 4 coins, en les séparant d'un trait rouge d'un pixel d'épaisseur.

5. Exportez cette image et notre Sprite est terminé.

3. Etape 2 : le code HTML

Avant toute chose nous allons attribuer à notre div conteneur une classe avec comme valeur .roundedBox :

```
<div class="roundedBox"></div>
```

Nous devons ensuite ajouter 4 div qui vont constituer nos coins par la suite. Il faudra attribuer à chacun une classe avec comme valeur .corner, ainsi qu'une classe en fonction de leur position.

```
<div class="roundedBox">
  <strong>Mon contenu dans l'exemple 1 de boite
arrondie</strong>
  <div class="corner topLeft"></div>
  <div class="corner topRight"></div>

  <div class="corner bottomLeft"></div>
  <div class="corner bottomRight"></div>
</div>
```

Tout est fait ? Passons au code CSS.

4. Etape 3 : le code CSS

Comme vous le savez (sinon vous allez l'apprendre ici), les éléments positionnés de manière absolue sont placés en fonction du premier élément parent positionné de manière relative. Si aucun élément parent n'est positionné de manière relative, les éléments positionnés en absolute vont prendre l'élément body comme parent positionné de manière relative.

Hein ! Si vous ne comprenez pas cette phrase, ne vous inquiétez pas, vous allez la comprendre dans une seconde.

4.1. Déterminons d'abord le style de nos coins

Nous devons les positionner de manière absolue et définir leur largeur et leur hauteur (qui sera identique pour les 4 coins).

Mes coins ont une largeur et une hauteur de 17 pixels.


```
.corner {
        position : absolute;
        width : 17px;
        height : 17px;
}
```

Si vous exportez une image en forme de rectangle lorsque vous créez votre premier coin, la largeur et la hauteur ne seront pas identiques !

4.2. Déterminons maintenant le style de notre div conteneur

```
.roundedBox {
        position : relative;
}
```

Ce code permet à tout élément positionné de manière absolue **qui se trouve dans** un élément possédant la classe .roundedBox, de se positionner en fonction de cet élément, au lieu de se positionner en fonction de l'élément body.

Nous devons également lui attribuer un padding, car sinon, dans le cas contraire, les coins apparaîtraient au-dessus de notre texte et ce n'est pas ce que nous voulons.

Le padding haut et bas doit être égal et identique à la **hauteur** de l'image représentant le coin. Le padding gauche et droite doit être égal et équivalent à la **largeur** de l'image représentant le coin.

Comme vous le savez déjà, la largeur et la hauteur de mes coins sont identiques, donc le padding est le même pour les 4 cotés.

```
.roundedBox {
        position : relative;
        padding : 17px;
        margin : 10px 0;
}
```

J'ai également inclus une marge pour donner un peu d'espace à notre div.

4.3. Enfin déterminons le style de chaque coin individuellement

Nous allons définir la position absolue de chaque coin, ainsi que leur position d'arrière-plan (en fonction de notre Sprite).

```
.roundedBox {
        position : relative;
        padding : 17px;
        margin : 10px 0;
}

.corner {
       position : absolute;
       width : 17px;
       height : 17px;
}

.topLeft {
        top : 0;
        left : 0;
        background-position : -1px -1px;
}

.topRight {
        top : 0;
        right : 0;
        background-position : -19px -1px;
}

.bottomLeft {
        bottom : 0;
        left : 0;
        background-position : -1px -19px;
}

.bottomRight {
        bottom : 0;
        right : 0;
        background-position : -19px -19px;
}
```

Vous l'avez certainement noté mais notre feuille de style n'a pour l'instant pas chargé notre Sprite. Cela s'explique par le fait qu'on va utiliser différents Sprites, nous n'allons donc pas les définir de manière générique.

5. Exemple 1 de boite arrondie (Bleu)

5.1. Le code (X)HTML

```
<div class="roundedBox" id="type1">
    <strong>Mon contenu dans l'exemple 1 de boite
arrondie</strong>

    <div class="corner topLeft"></div>
    <div class="corner topRight"></div>
    <div class="corner bottomLeft"></div>
    <div class="corner bottomRight"></div>
</div>
```

Nous devons attribuer à notre div un id avec comme valeur #type1 pour appliquer un arrière-plan spécifique.

5.2. Le code CSS

Il faudra d'abord faire correspondre la couleur d'arrière-plan de l'id #type1 à la couleur de fond de notre coin dans le Sprite.

```
#type1 {
        background-color : #CCDEDE;
}
```

Puis, en utilisant la classe .corner, charger la Sprite CSS pour ce modèle de boite arrondie.

```
#type1 {
        background-color : #CCDEDE;
}

#type1 .corner {
        background-image : url(../images/corners-type1.gif);
}
```

Voilà, notre premier rectangle arrondi est terminé. Consultez l'exemple N° 1 de boite arrondie (bleu) (Lien67).

6. Exemple 2 de boite arrondie (Vert) / Exemple 3 de boite arrondie (Violet)

La seule différence entre l'exemple 1 et les exemples 2 et 3, c'est la couleur. Nous allons donc juste changer ce point.

6.1. Exemple 2 (Vert)

6.1.1. Code (X)HTML

```
<div class="roundedBox" id="type2">
    <strong>Mon contenu dans l'exemple 2 de boite arrondie</strong>

    <div class="corner topLeft"></div>
    <div class="corner topRight"></div>
    <div class="corner bottomLeft"></div>
    <div class="corner bottomRight"></div>
</div>
```

6.1.2. Code CSS

Changez simplement la couleur des Sprites et de l'arrière-plan.

```
#type2 {
        background-color : #CDDFCA;
}

#type2 .corner {
        background-image : url(../images/corners-type2.gif);
}
```

Consultez l'exemple N° 2 de boite arrondie (vert) (Lien68).

6.2. Exemple 3 (Violet)

6.2.1. Code (X)HTML

```
<div class="roundedBox" id="type3">
    <strong>Mon contenu dans l'exemple 3 de boite arrondie</strong>

    <div class="corner topLeft"></div>
    <div class="corner topRight"></div>
    <div class="corner bottomLeft"></div>
    <div class="corner bottomRight"></div>
</div>
```

6.2.2. Code CSS

Changez simplement la couleur des Sprites et de l'arrière-plan.

```
#type3 {
        background-color : #D3CADF;
}

#type3 .corner {
        background-image : url(../images/corners-type3.gif);
}
```

Consultez l'exemple N° 3 de boite arrondie (violet) (Lien69).

Vous avez compris le principe ? Alors allons maintenant un peu plus vite.

7. Exemple 4 (Rouge avec une bordure)

Quelle est la différence entre l'exemple 4 et les exemples 1, 2 et 3 ? La couleur et la bordure. Gérons ces éléments.

7.1. Le code (X)HTML

```
<div class="roundedBox" id="type4">
    <strong>Mon contenu dans l'exemple 4 de boite arrondie</strong>

    <div class="corner topLeft"></div>
    <div class="corner topRight"></div>
    <div class="corner bottomLeft"></div>
    <div class="corner bottomRight"></div>
</div>
```

7.2. Le code CSS

Ajoutez une bordure à vos coins dans le Sprite et faites correspondre l'arrière-plan et la bordure déclarée dans .roundedBox avec les coins du Sprite.


```
#type4 {
        background-color : #CCACAE;
        border :           1px solid #AD9396;
}

#type4 .corner {
        background-image : url(../images/corners-
type4.gif);
}
```

C'est là qu'il y a un hic. Nos coins ne vont pas recouvrir correctement notre bordure déclarée dans l'id #type4. Nous devons donc corriger leur position en réécrivant les précédents styles relatifs au positionnement. Faisont cela :

```
#type4 {
        background-color : #CCACAE;
        border : 1px solid #AD9396;
}

#type4 .corner {
        background-image : url(../images/corners-
type4.gif);
}

#type4 .topLeft {
        top : -1px;
        left : -1px;
}

#type4 .topRight {
        top : -1px;
        right : -1px;
}

#type4 .bottomLeft {
        bottom : -1px;
        left : -1px;
}

#type4 .bottomRight {
        bottom : -1px;
        right : -1px;
}
```

Nous en avons fini avec l'exemple 4. Consulter cet exemple de boite arrondie (rouge avec bordure) (Lien70).

Nous y sommes presque, ne partez pas maintenant.

8. Exemple 5 (avec un dégradé vertical)

L'exemple 5 nécessite un peu plus de travail. Nous devons faire les choses suivantes :

1. Changer la hauteur du coin supérieur ou inférieur (en fonction de notre dégradé) ;
2. Changer la position d'arrière-plan du coin supérieur ou inférieur (en fonction de notre dégradé) ;
3. Attribuer à notre div conteneur un background-repeat d'un pixel, pour créer la répétition de notre dégradé ;
4. Avoir une quantité significative de contenu ou déclarer un min-height pour notre div (en fonction de notre dégradé).

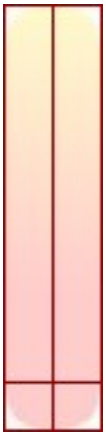
Au travail.

8.1. Le code (X)HTML (le même que précédemment)

```
<div class="roundedBox" id="type5">
    <strong>Mon contenu dans l'exemple 5 de boite
arrondie</strong>

    <div class="corner topLeft"></div>
    <div class="corner topRight"></div>
    <div class="corner bottomLeft"></div>
    <div class="corner bottomRight"></div>
</div>
```

8.2. Le code CSS

Mon dégradé est vertical, du haut vers le bas. Nous devons donc augmenter la hauteur des coins supérieurs et changer le background-position des coins inférieurs. Vous allez comprendre pourquoi je fais ça en voyant mon nouveau Sprite, qui est le suivant :



Et voici l'image d'arrière-plan de mon div :

L'image a une largeur de 1 pixel, mais elle est bien là !

Mes coins inférieurs ont une couleur définie et nous allons faire en sorte qu'elle corresponde avec la couleur d'arrière-plan du div.

Parlons moins et agissons plus. Commençons avec notre div conteneur.

```
#type5 {
	background : #FECBCA
url(../images/roundedbox-type5-bg.png) repeat-x 0
0;
	min-height : 110px;
}
```

Pour déterminer la couleur de l'arrière-plan, j'ai appliqué l'outil pipette sur l'arrière-plan des coins inférieurs. Puis j'ai ajouté l'image d'arrière-plan qui sera répétée dans l'abscisse x. Enfin j'ai ajouté un min-height, comme je vous l'ai dit plus tôt, pour que le dégradé ne soit pas interrompu. Nous pouvons maintenant ajouter les images des coins (j'ai changé le type d'image en .png pour améliorer la qualité du dégradé).

```
#type5 {
	background : #FECBCA
url(../images/roundedbox-type5-bg.png) repeat-x 0
0;
	min-height : 110px;
}

#type5 .corner
	background-image : url(../images/corners-
type5.png);
}
```

Il faut maintenant augmenter la hauteur des coins supérieurs (qui va dépendre du moment où le dégradé atteint la couleur fixe).

```
#type5 {
	background : #FECBCA
url(../images/roundedbox-type5-bg.png) repeat-x 0
0;
	min-height : 110px;
}

#type5 .corner
	background-image : url(../images/corners-
type5.png);
}

#type5 .topLeft,
#type5 .topRight {
	height : 140px;
}
```

Enfin, je corrige le background-position des coins inférieurs.

```
#type5 {
	background : #FECBCA
url(../images/roundedbox-type5-bg.png) repeat-x 0
0;
	min-height : 110px;
}

#type5 .corner
	background-image : url(../images/corners-
type5.png);
}

#type5 .topLeft,
#type5 .topRight {
	height : 140px;
}

#type5 .bottomLeft {
	background-position : -1px -142px;
}

#type5 .bottomRight {
	background-position : -19px -142px;
}
```

Tout est fait. Consultez l'exemple 5 de boite arrondie (avec un dégradé vertical (Lien71).

9.Internet Explorer 6

Ce mauvais navigateur a un problème avec cette technique. Vous devez attribuer au conteneur (.roundedBox, #type1, #type2, *etc.*) une largeur et une hauteur déterminées. Si vous ne les définissez pas, le conteneur aura l'air déformé.

Utilisez les commentaires conditionnels IE6 pour résoudre ce problème.

10 : Remarques finales

Vous pouvez inventer d'autres types de coins arrondis avec cette technique. Avec des dégradés horizontaux, des coins transparents. Faites juste un peu travailler vos neurones et vous parviendrez à obtenir ces styles.

J'espère que cette technique vous aura été utile et pas trop complexe.

Retrouvez l'article d'Ignacio Ricci traduit par Christophe F. en ligne : Lien72

Les librairies JavaScript vraiment utiles

Cet article est la traduction de :List of Really Useful JavaScript Libraries (Lien73).
Les frameworks JavaScript usuels apportent l'essentiel des fonctionnalités qu'un développeur peut espérer. De plus, leur capacité d'extension par les systèmes de plugin les rendent encore plus puissants. Malgré tout, il se peut que vous n'ayez pas envie d'utiliser de tels frameworks ou que celui que vous utilisez n'ait pas de fonctions suffisamment spécialisées pour vos besoins.
Dans ce cas, il existe de nombreuses librairies autonomes très utiles.
W3Avenue vous a compilé une liste de librairies JavaScript indépendantes qui devraient vous rendre service si votre framework (Prototype, jQuery, MooTools, Dojo, etc.) ne couvre pas vos besoins spécifiques ou si vous n'en utilisez pas.

1. Animation

- JSTweener (Lien74)

JSTweener est une librairie de gestion des transitions (*tween* en anglais) basée sur la classe Tweener (Lien75) utilisée dans le code *ActionScript* de Flash.

- Sfx() - JavaScript Animation Library (Lien76)

$fx() est une librairie JavaScript légère (moins de 4Ko) d'animation d'éléments HTML. Elle vous permet de modifier n'importe quelle propriété CSS progressivement avec un paramétrage simple. Vous pouvez aussi combiner les effets en les enchainant ou en les synchronisant. Enfin, de nombreux *callbacks* vous offrent beaucoup de liberté dans la gestion de vos effets.

- Facebook Animation (Lien77)

Cette librairie vous offre beaucoup de possibilités pour améliorer leur page Facebook avec juste une ou deux lignes de code. Toutes les animations sont basées sur les CSS, donc une bonne connaissance de CSS est utile.
A noter qu'il existe une version open-source fonctionnant en dehors de Facebook.

- Autres librairies : FX (Lien78), Animator.js (Lien79), jsAnim (Lien80)

2. Audio / Vidéo

- SoundManager (Lien81)

SoundManager importe et améliore l'API *Sound* de Flash et la rend disponible en JavaScript. La portion Flash est cachée et donc invisible, que ce soit pour le développeur ou l'utilisateur.

- Flowplayer JavaScript API (Lien82)

L'API JavaScript de *Flowplayer* vous permet de contrôler facilement et efficacement une ou plusieurs instances de Flowplayer (Lien83) dans une page HTML.
Flowplayer se compose de deux parties : l'objet *SWF* Flowplayer, inclus dans un objet Flash et une librairie JavaScript qui transforme les instructions simples de l'API en interaction plus complexe avec l'objet SWF (qui contrôle l'objet Flash).

3. Cookies

- Cookies (Lien84)

Cette librairie vous permet de récupérer et de manipuler les cookies HTTP dans le navigateur. Vous pouvez récupérer un cookie ou une liste, en créer de nouveaux, en supprimer, tester si le navigateur les accepte. Le framework jQuery n'est pas nécessaire pour utiliser la librairie, mais sa présence ajoute des fonctionnalités, comme créer un cookie à chaque remplissage d'un champ de formulaire ou le remplissage automatique d'un formulaire avec les valeurs des cookies.

- EasyCookie (Lien85)

Un script simple et facile d'utilisation permettant la gestion des cookies.

4. Cryptographie

- JavaScript MD5 (Lien86)

Une implémentation en JavaScript de l'algorithme MD5.

5. Bases de données

- Taffy DB (Lien87)

Taffy DB est une librairie AJAX libre. Compatible avec les principales librairies AJAX, ses principales fonctionnalités sont : une interface CRUD (Create, Read, Update, Delete), le tri, les boucles, la gestion de requêtes avancées, etc.

- ActiveRecord.js (Lien88)

AciveRecord.js est un outil de mapping objet-relationnel compatible avec tous les navigateurs. Son vocabulaire est proche de celui d'ActiveRecord pour Ruby, mais en utilisant la syntaxe et les bonnes pratiques JavaScript. Il peut être utilisé à partir d'un mappage chargé en mémoire ou avec un environnement SQL sur une plateforme Jaxer (SQLite et MySQL), AIR d'Adobe (SQLite) ou Google Gears (SQLite).

6. Date / Heure

- Date.js (Lien89)

Cette librairie open-source vous permet de manipuler les

dates et les heures facilement. Elle supporte plus de 150 formats et propose entre autres de fixer une date, de la parser, de la modifier, de faire des comparaisons, etc.

7. Débogage / Traçage

- Firebug Lite (Lien90)

Firebug est probablement l'extension de débogage la plus populaire de Firefox. Firebug Lite est l'alternative pour tester ses pages sur Internet Explorer, Opera et Safari. Il s'agit d'un script JavaScript à inclure pour obtenir certaines fonctionnalités de Firebug.

- Blackbird (Lien91)

Utilitaire de traçage open-source. Il vous permet, à travers une console assez réussie, d'afficher différents messages de débogage.

- NitobiBug (Lien92)

NitobiBug est un outil d'accès aux objets JavaScript basé sur le navigateur (similaire à Firebug). Il peut être utilisé dans différents navigateurs (IE6+, Safari, Opera, Firefox) et offre un outil stable et puissant pour développer des applications AJAX riches.

8. Polices / Texte / Typographie

- strokeText.js (Lien93)

strokeText.js est une librairie non intrusive fonctionnant avec les principaux navigateurs (Firefox 1.5 +, IE6 +, Opera 9 +, et Safari). La librairie propose une API d'écriture de texte pour *Canvas* et *VML*. La police intégrée sans serif est adaptée au *SVG* pour un rendu identique.

- typeface.js (Lien94)

Cette librairie vous permet d'intégrer vos polices afin d'éviter d'avoir à les afficher sous forme d'images. Au lieu d'utiliser une image ou du Flash pour afficher vos polices, utilisez typeface.js et écrivez votre code HTML / CSS comme si vos visiteurs avaient la police installée.

- Cufón (Lien95)

Remplacement rapide de texte à l'aide de *Canvas* et *VML*. Pas besoin de Flash ni d'images.

- Hyphenator.js (Lien96)

Coupe automatiquement les mots en fin de ligne sur votre site, ou sur tous les sites si vous l'utilisez en plugin.

- Autres librairies : sIFR (Lien97), Facelift Image Replacement (Lien98), FontJazz (Lien99).

9. Validation de formulaire

- LiveValidation (Lien100)

Cette librairie open-source légère vous permet d'effectuer des validations de champs de formulaire facilement et efficacement. Deux versions sont disponibles : une basée sur le *framework* Prototype et une autonome.

- wForms (Lien101)

Librairie non intrusive qui offre toutes les validations usuelles de formulaires sans avoir besoin de connaissances en programmation.

- Validanguage (Lien102)

Validanguage est une librairie open-source, non intrusive à gestion d'héritage, développée pour devenir le framework de validation de formulaires de référence.

- Autres librairies : Yav (Lien103), qForms JavaScript API (Lien104).

10. Flash

- SWFObject (Lien105)

SWFObject est une librairie facile d'emploi et respectant les standards pour intégrer du contenu Flash, elle utilise un script JavaScript très léger.

- AS3Wrapper (Lien106)

Une librairie compatible pour IE et Firefox qui rend disponible l'API Flash depuis vos scripts JavaScript. Du fait de la similarité des deux langages, coder du Flash en JavaScript est étonnamment facile. En plus de cela, les performances semblent être particulièrement satisfaisantes.

- Aflax (Lien107)

Une librairie qui permet d'utiliser pleinement le moteur d'exécution d'Adobe Flash 8.

11. Graphiques / Diagrammes

- PlotKit (Lien108)

Une librairie d'affichage de diagrammes. Elle supporte les *canvas* HTML, mais aussi le *SVG* avec *Adobe SVG Viewer* ou le support natif des navigateurs.

- JS Charts (Lien109)

Une librairie libre qui vous permet de créer des graphiques variés sous différentes formes, barres, camemberts, linéaires. Il vous suffit d'ajouter le fichier jscharts.js, d'intégrer vos données en XML ou dans un tableau JavaScript et ça s'affiche !

- Flot (nécessite jQuery) (Lien110)

Une librairie d'affichage de données pour jQuery. Affiche graphiquement vos données à la volée. Flot fonctionne avec IE6+, Firefox 2+, Safari 3+, Opera 9.5+ et Konqueror 4+.

Voir un tutoriel : Visualisation professionnelle de vos données avec Flot et jQuery (Lien111)

- Autre librairie : JavaScript Diagram Builder (Lien112)

12. Tables HTML

- SortTable (Lien113)

Cette librairie vous permet de rendre vos tables HTML triables. Elle est non intrusive (utilisation du DOM) et peut trier selon beaucoup de formats.

- DragTable (Lien114)

Vous permet de trier une table HTML par *drag and drop* (glisser / déplacer).

- KeyTable (Lien115)

Une librairie de gestion du clavier et de la délégation

d'événements pour vos tables HTML. Vous pouvez manier vos tables presque comme avec Excel pour les éditer par exemple.

13. Gestion des images / Visualisation / Dessin

- Processing.js (Lien116)

Processing.js utilise JavaScript pour dessiner des formes et manipuler des images grâce à l'élément canvas du HTML 5. Le code est léger et simple à utiliser. Un outil idéal pour visualiser des données, créer des interfaces utilisateur et développer des jeux pour le Web.

- Raphaël (Lien117)

Cette librairie va vous simplifier le travail sur les graphiques vectoriels pour le Web. Elle utilise les formats SVG et VML pour créer des graphiques. Elle est compatible pour Firefox 3.0+, Safari 3.0+, Opera 9.5+ et Internet Explorer 6.0+.

- Pixastic Image Processing Library (Lien118)

Cette librairie vous permet d'effectuer différentes actions sur les images avec un peu de JavaScript. Les effets disponibles comprennent : la désaturation/niveaux de gris, l'inversion, la rotation, l'ajustement de la luminosité et du contraste, le recadrage et bien d'autres.

- VectorGraphics Library (Lien119)

Une librairie de graphiques vectoriels très performante. Dessinez des lignes, des cercles, des ellipses, des lignes brisées, des polygones, des rectangles...

- Reflection.js (Lien120)

Ajoutez des reflets sur les images de vos pages. Du JavaScript non intrusif qui vous permet de garder votre code propre. Compatible avec les principaux navigateurs : Internet Explorer 5.5+, Firefox 1.5+, Safari, Chrome et Opera 9+. Sur les navigateurs plus anciens, le code utilise le principe de dégradation élégante, ainsi, vos visiteurs ne se rendront compte de rien. Tout cela en moins de 5 ko !

- CVI Libraries (Canvas Vml Image Effects) (Lien121)

Les librairies CVI utilisent du code non intrusif et sont compatibles avec les principaux navigateurs : Firefox 1.5+, Opera 9+, Internet Explorer 6+ et Safari. Les scripts sont fournis sous licence Netzgestade Software License Agreement. La licence permet l'utilisation pour des sites privés ou non commerciaux. Il existe aussi des licences commerciales. Les librairies incluses sont : bevel.js (Lien122), comer.js (Lien123), curl.js (Lien124), edge.js (Lien125), filmed.js (Lien126), glossy.js (Lien127), instant.js (Lien128), reflex.js (Lien129), shiftzoom.js (Lien130), snapfit.js (Lien131), slided.js (Lien132), sphere.js (Lien133).

- Drag & Drop for Images and Layers (Lien134)

Une librairie multi-navigateurs qui ajoute le *drag'n'drop* et d'autres fonctionnalités DHTML à vos calques et images.

- ExplorerCanvas (Lien135)

Les dernières versions des navigateurs Firefox, Safari, Chrome et Opera supportent la balise HTML5 <canvas> pour permettre l'utilisation de commandes de dessin 2D. ExplorerCanvas apporte les mêmes fonctionnalités pour Internet Explorer. Pour l'utiliser, vous n'avez qu'à placer un simple script sur vos pages existantes.

- Canvas 3D JS Library (C3DL) (Lien136)

Fournit aux développeurs un ensemble de classes pour rendre le développement 3D plus accessible sans pour autant avoir à étudier en profondeur les mathématiques 3D. Nécessite un navigateur supportant Canvas 3D (Firefox 3.5+).

- jsDraw2D (Lien137)

Librairie graphique open-source de dessin avancé. Dessinez des courbes de Bézier cubiques ou générales et des courbes ouvertes ou fermées en passant par des points donnés. Ne nécessite ni framework, ni plugin, ni SVG ni VML.

14. Clavier

- Shortcuts.js (Lien138)

Facilite l'ajout de raccourcis claviers à votre application JavaScript.

- Qfocuser (Lien139)

Une classe JavaScript de gestion de la navigation au clavier pour les widgets AJAX.

15. Cartes

- Mapstraction (Lien140)

Mapstraction est une librairie qui fourni une API commune pour différents services de cartographie, permettant de passer facilement de l'une à l'autre. Vous n'avez à programmer qu'une seule application, puis passer d'un prestataire à l'autre en fonction de vos besoins.

16. Maths

- Sylvester (Lien141)

Librairie JavaScript de calcul vectoriel et matriciel qui vous évite de multiplier les boucles et la gestion de tableaux complexes. Comprend des classes de gestion de vecteurs et matrices de dimensions quelconques et de modélisation de lignes et plans dans l'espace. Vous permet d'écrire du code orienté objet facile à lire et représentatif de l'objet mathématique qu'il utilise.

17. Expressions régulières

- XRegExp (Lien142)

Cette librairie ajoute des fonctionnalités aux expressions régulières natives, compatible tous navigateurs, incluant des fonctions de remplacement et une syntaxe améliorée. Diverses méthodes utiles et une puissante méthode de constructions récursives sont aussi présentes.

- textMonster (Lien143)

Cette librairie vous permet le chaînage des résultats d'expressions régulières vous permettant une recherche itérative.

18. URL

- JavaScript URL Library (Lien144)

Cette librairie facilite la construction et la décomposition des URL en tenant compte de leurs divers éléments. La

librairie crée un objet URL depuis une chaîne de caractères ou window.location. Toutes les parties sont manipulables et l'URL peut être facilement reconstruite.

- UED (URL Encoded Data) (Lien145)

Ce script prend en argument un tableau de paramètres et retourne la chaîne encodée au format UED. Cette chaîne peut alors être envoyée par POST ou GET.

19. Divers

- MoreCSS (Lien146)

Une librairie qui vous sera utile pour toutes les tâches habituelles en JavaScript : pop-up, onglets, infobulles, césure, présentation de listes etc. Tout ça sans avoir à apprendre JavaScript. Si vous connaissez déjà la syntaxe CSS, c'est l'outil qu'il vous faut.

- Modemazr (Lien147)

Une librairie très utile pour utiliser les dernières normes du Web (CSS 3, HTML 5) tout en prenant en compte les navigateurs plus anciens.

- IE7.js (Lien148)

Une librairie qui permet à IE7 de se comporter selon les standards. Elle règle les différences d'interprétation HTML et CSS et gère la transparence des images PNG pour IE6.

- Sizzle JavaScript Selector Library (Lien149)

Un moteur de sélection d'éléments selon la syntaxe CSS facilement intégrable dans un librairie tierce. Cette librairie supporte quasiment tous les sélecteurs CSS3, y compris des formes peu usuelles (comme ".foo\+bar" par exemple) et retourne le résultat selon l'ordre du document.

- DD_Roundies (Lien150)

Une librairie de création de coins arrondis, principalement utile pour IE.

- DD_BelatedPNG (Lien151)

Une librairie de gestion des images PNG pour IE6. Les PNG peuvent être intégrés soit dans une balise <img \> soit dans une propriété CSS background-image, dans ce cas, les propriétés additionnelles de position et répétition sont correctement prises en compte.

- SocialHistory.js (Lien152)

Cette librairie vous permet de déterminer quels réseaux sociaux sont utilisés par vos visiteurs.

- SyntaxHighlighter (Lien153)

Une librairie de coloration syntaxique pour les exemples de code sur vos sites.

- PHP.js (Lien154)

Une librairie qui porte en JavaScript certaines fonctions PHP usuelles. En incluant cette librairie, vous pourrez utiliser côté client vos fonctions PHP préférées. Très utile pour les développeurs PHP confrontés à l'émergence des besoins côté client.

20. Conclusion et remerciements

J'espère que ces librairies sauront vous aider et vous simplifier le développement JavaScript.
J'ai essayé de rendre cette liste la plus complète possible, cependant, si vous connaissez des librairies qui vous semblent manquer, n'hésitez pas à les proposer pour enrichir cet article.

Autres articles de la série :

- Les classes et librairies vraiment utiles pour les développeurs PHP (Lien155)
- Les outils vraiment utiles pour les développeurs JavaScript (Lien156)

Retrouvez l'article de Saud Khan traduit par Didier Mouronval en ligne : Lien157

Découvrez OpenGL 1.1 en VB6/VBA

Découvrez OpenGL 1.1 en VB6 ou en VBA.
Cet article est destiné aux développeurs VB6/VBA expérimentés qui souhaitent découvrir OpenGL 1.1.

1. Introduction

Au cours de ce tutoriel, nous allons découvrir la programmation de la 3D avec OpenGL (Lien158).
Cette découverte va se faire en langage VBA ou VB6.
Cependant, les fonctions et techniques utilisées sont les mêmes dans tous les langages.

Pour simplifier, lorsque j'emploie le terme **VB**, il réfère à **VB6** ou à **VBA**

2. Librairies utilisées

Tout d'abord OpenGL est installé avec Windows.
Il y a deux librairies pour OpenGL dans le répertoire système de Windows :
- **opengl32.dll** : c'est la librairie principale
- **glu32.dll** = **GL U**tility : c'est un complément qui contient quelques autres fonctions utiles

OpenGL ne gère pas le fenêtrage : c'est-à-dire qu'il faut fournir à OpenGL une fenêtre sur laquelle il va effectuer le rendu de l'image.
De plus il ne gère pas les entrées clavier, souris…

Pour gérer l'affichage des fenêtres et des événements divers (clavier, souris, minuterie...), il existe plusieurs librairies.
Glut (Lien159) est une librairie très largement utilisée.
Mais pour une utilisation en VB, je lui ai préféré freeglut (Lien160) qui expose des fonctions similaires, et quelques nouveautés qui nous seront utiles, voir indispensables en VBA.

Nous verrons qu'il est également possible de faire un rendu sur un formulaire sans freeglut.

Enfin pour appeler les fonctions de ces librairies, il faut déclarer chacune d'entre elles dans VB.
Nous allons donc préparer notre environnement de développement.

3. Le pack de module VB pour OpenGL

Tous les modules VB nécessaires à l'utilisation d'OpenGL sont regroupés dans une archive compressée au format zip.

Téléchargez cette archive :
Pack de module VB pour OpenGL (Lien161)

Pour utiliser **FreeGlut**, téléchargez également la librairie dll :
Librairie FreeGlut (Lien162)

4. Préparation de l'environnement de développement VB

Créez d'abord une nouvelle application *Microsoft Office* : **Access**, **Excel**, **Word**…
Ou créez un nouveau projet *Visual Basic 6*.

Afin de pouvoir utiliser les fonctions **OpengGL** en VB :

Importez le module standard **ModOpenGL_1_1**
Ce module contient les déclarations des fonctions et constantes de **OpenGL 1.1**.

Importez le module standard **ModOpenGLTools**
Ce module contient les déclarations des fonctions et constantes de **OpenGL 1.1** de **Glu**, ainsi que des fonctions spécifiques à Windows (wgl et gdi32).
Il contient également des fonctions supplémentaires utiles pour l'utisation d'openGL en VB.

Importez le module standard **ModOpenGLFreeGlut**
Ce module contient toutes les déclarations nécessaires à **freeglut**.

Pour utiliser **FreeGlut** (création de fenêtres) :

Placez la librarie freeglut.dll (Lien162) dans le même répertoire que l'application Office ou le projet VB6.

Cette librairie a été compilée avec *Visual C++ Express 2008*.

freeglut est sous licence : X-Consortium license (Lien163).

Avec Office, pour ouvrir l'éditeur VBA, tapez **ALT + F11**.
Pour importer un module, choisissez dans le menu : **Fichier** => **Importer un fichier...**

5. Première fenêtre avec freeglut

Créez un nouveau module standard (**Insertion** => **Module** dans Office).
Dans ce module nous ajoutons une fonction **FonctionOpenGL** dans laquelle nous écrirons le code de création de la fenêtre.

Fonction FonctionOpenGL
```
Function FonctionOpenGL()

End Function
```

Pour VB6, créez une procédure **Main** à la place de cette fonction.

Procédure d'entrée VB6
```
Sub Main()

End Sub
```

Notez qu'on a placé la librairie *freeglut.dll* dans le même répertoire que l'application.
Pour appeler les fonctions de cette librairie, il faut donc d'abord la charger.

Si on avait placé cette librairie dans le répertoire système de Windows, il n'aurait pas été nécessaire de la charger.

Pour charger la librairie, utilisons la fonction **LoadLibrary** (elle est déclarée dans le module **ModFreeGlut**).

Chargement de la librarie freeglut
```
' Chargement de freeglut
If LoadLibrary(CurrentProject.Path &
"\freeglut.dll") = 0 Then
    MsgBox "Impossible de charger la librairie
freeglut"
    Exit Function
End If
```

Si le programme n'arrive pas à charger la librairie, la fonction renvoie 0 et on affiche alors un message d'erreur.

Remplacez **CurrentProject** en fonction de l'application utilisée :
- **CurrentProject** pour **Access**.
- **ThisWorkBook** pour **Excel**.
- **ThisDocument** pour **Word**.
- **App** pour **VB6**.

On peut maintenant appeler les fonctions de *freeglut*.

Première étape, l'initialisation :

Initialisation de la librarie freeglut
```
' Initialisation de la librairie
glutInit 0&, ""
```

D'après la documentation de glutInit (Lien164), les paramètres de cette fonction d'initialisation sont utilisés pour les systèmes **X Window**.
Nous n'utilisons pas ces paramètres, on passe donc des valeurs vides.

Deuxième étape, l'initialisation du mode d'affichage :

Initialisation du mode d'affichage
```
' Initialisation du mode d'affichage
glutInitDisplayMode GLUT_RGBA Or GLUT_DOUBLE Or
GLUT_DEPTH
```

- **GLUT_RGBA** signifie que la fenêtre utilise le modèle de couleur RGB (rouge, vert, bleu) plus une composante alpha de transparence.
- **GLUT_DOUBLE** signifie que nous souhaitons utiliser un double buffer (nous le détaillerons plus tard).
- **GLUT_DEPTH** signifie que nous souhaitons utiliser un tampon de profondeur (nous l'utiliserons également plus tard).

Troisième étape, création d'une fenêtre :

Création d'une fenêtre
```
' Création d'une fenêtre
glutCreateWindow "Tutoriel fenêtre GLUT"
```

Le titre de la fenêtre est passé en paramètre.
Cette fonction renvoie un numéro de fenêtre (interne à freeglut) que l'on peut utiliser ensuite pour, par exemple, redimensionner, positionner ou détruire la fenêtre.
Il est également possible de créer plusieurs fenêtres et de passer de l'une à l'autre.
Nous n'utiliserons pas ce numéro.

Quatrième étape, exécution de la boucle principale :

Glut fonctionne avec une boucle principale qu'il est nécessaire d'exécuter.

Exécution de la boucle principale
```
' Boucle principale
glutMainLoop
```

Cette boucle est sans fin, les instructions situées après l'appel à cette fonction ne seront pas exécutées tout de suite.

En VBA avec Office :
Si on exécute la fonction **FonctionOpenGL** (touche *F5*), on voit bien la fenêtre s'afficher mais si on la ferme toute l'application se ferme brutalement.

Pour éviter ce comportement, on définit une option de *freeglut* très pratique :
ajoutez cette instruction juste après l'initialisation avec *glutInitDisplayMode*.

Définition de l'option de sortie de boucle
```
' Définition de l'option de sortie de boucle
glutSetOption GLUT_ACTION_ON_WINDOW_CLOSE,
GLUT_ACTION_GLUTMAINLOOP_RETURNS
```

Cette option demande à *freeglut* de continuer l'exécution après l'instruction *glutMainLoop* au lieu d'arrêter l'application.
On pourra alors si besoin fermer nous-mêmes l'application correctement.

Testons à nouveau notre programme, on peut fermer la fenêtre tout en conservant l'application ouverte.

Mais rien ne s'affiche dans la fenêtre, ou plutôt si : la fenêtre contient ce qui se situe dessous lors de son ouverture.
Pour l'instant c'est normal, on n'a pas encore géré l'affichage du contenu de notre fenêtre.

Cinquième étape, la fonction d'affichage :

On peut avec *freeglut* définir des fonctions de rappel (CallBack en anglais).
Ces fonctions sont exécutées depuis *freeglut* au cours de la boucle que nous avons exécutée avec **glutMainLoop**.
La liste des fonctions et de leur déclaration en VB est décrite dans le chapitre suivant.

La fonction de rappel indispensable est la fonction d'affichage.
Cette fonction est appelée à chaque fois qu'il est nécessaire d'afficher le contenu de la fenêtre.

Créez la fonction d'affichage dans le module VB :

Création de la fonction d'affichage
```
' Fonction d'affichage
Public Sub CallBackDraw()

End Sub
```

Puis définissez la fonction dans *freeglut*, **avant l'appel à glutMainLoop**.

Définition de la fonction d'affichage
```
' Fonction d'affichage
glutDisplayFunc AddressOf CallBackDraw
```

Dans la fonction d'affichage on va :
- Vider les buffers avec *glClear*.
- Réaliser les opérations de rendu (dessin d'obets, éclairage...), ce que nous ajouterons par la suite.
- Echanger les buffers pour afficher le résultat dans la fenêtre.

Contenu de la fonction d'affichage
```
' Fonction d'affichage
Public Sub CallBackDraw()
' Vide les buffers couleur et profondeur
glClear GL_COLOR_BUFFER_BIT Or
GL_DEPTH_BUFFER_BIT
' Echange les buffers
glutSwapBuffers
End Sub
```

Vous verrez sans doute dans diverses documentations un appel à **glFlush** qui demande à OpengGL d'exécuter les commandes en attente.
Nous n'appelons pas cette fonction ici, **glutSwapBuffers** le fait pour nous.

Notez également la fonction utilisée : **glutSwapBuffers** et non pas **SwapBuffers**.

Voici le rappel du code complet de notre première fenêtre GLUT :

Module de base pour création fenêtre GLUT
```
Option Explicit

Function FonctionOpenGL()
' Chargement de freeglut
If LoadLibrary(CurrentProject.Path &
"\freeglut.dll") = 0 Then
    MsgBox "Impossible de charger la librairie
freeglut"
    Exit Function
End If
' Initialisation de la librairie
glutInit 0&, ""
' Initialisation du mode d'affichage
glutInitDisplayMode GLUT_RGBA Or GLUT_DOUBLE Or
GLUT_DEPTH
' Création d'une fenêtre
glutCreateWindow "Tutoriel fenêtre GLUT"
' Définition de l'option de sortie de boucle
glutSetOption GLUT_ACTION_ON_WINDOW_CLOSE,
GLUT_ACTION_GLUTMAINLOOP_RETURNS
' Fonction d'affichage
glutDisplayFunc AddressOf CallBackDraw
' Boucle principale
glutMainLoop
End Function

' Fonction d'affichage
Public Sub CallBackDraw()
' Vide les buffers couleur et profondeur
glClear GL_COLOR_BUFFER_BIT Or
GL_DEPTH_BUFFER_BIT
' Echange les buffers
glutSwapBuffers
End Sub
```

Exécutez (*F5*) la fonction **FonctionOpenGL** pour afficher la fenêtre, qui est uniquement un fond noir pour l'instant.

- Notez que le buffer de couleurs est initialisé avec la couleur définie par glClearColor.
glClearColor 1, 0, 0, 1 par exemple définit une couleur de fond rouge.
- Le buffer de profondeur est vidé en donnant à chaque pixel une profondeur de 1.
Ceci peut être changé avec la fonction glClearDepth.

6. Les fonctions de rappel de freeglut

On peut avec *freeglut* définir des fonctions de rappel (CallBack en anglais).
Ces fonctions sont exécutées depuis *freeglut* au cours de la boucle que nous avons exécutée avec **glutMainLoop**.

Voir le tableau en ligne : Lien165

7. Découverte d'OpenGL

7.1. Version ciblée et extensions

Le module **ModOpenGL_1_1** contient les déclarations de la version 1.1 d'OpenGL.
Il existe des versions plus récentes d'OpenGL (jusqu'à la version 3.1 à ce jour) et également des extensions.
Nous nous contenterons, pour ce tutoriel de découverte, de la version 1.1 qui permet déjà de réaliser de nombreuses opérations 3D.

Les fonctions des versions supérieures à la version 1.1 sont considérées comme des extensions, nous ne les utiliserons pas au cours de ce tutoriel.

De nombreuses fonctions utilisées dans cet article sont dépréciées dans la version 3 d'OpenGL.
Il est cependant toujours possible de les utiliser, toutes les cartes graphiques n'offrent pas encore à ce jour de driver compatible OpengGL 3.
Vous pouvez consulter les spécifications OpenGL 3 (Lien166) pour une liste des fonctions dépréciées.

Pour l'utilisation des extensions, consultez le tutoriel suivant : « Les extensions OpenGL en VBA et VB6 » (Lien167)

7.2. La programmation par états

La programmation en OpenGL est basée sur des états.
C'est-à-dire que l'on définit un état (une couleur, une épaisseur de trait...), et il est alors utilisé par toutes les opérations de dessin qui suivent.
Par exemple si on définit la couleur rouge, tous les points dessinés ensuite seront de couleur rouge tant qu'on ne fait pas appel à une autre couleur.

7.3. Les procédures et constantes

Les fonctions et constantes *OpengGL* commencent par le préfixe **gl**.
Elles sont déclarées dans le module **ModOpenGL_1_1**.

Celles de *Glu* commencent par le préfixe **glu**.
Les fonctions préfixées par **wgl** sont spécifiques à **Windows**.
Ces fonctions sont déclarées dans le module **ModOpenGLTools**.

Les fonctions et constantes *freeglut* commencent par le préfixe **glut**.
Elles sont toutes déclarées dans le module **ModOpenGLFreeGlut**.

Quelques fonctions **gdi32** sont également utiles pour créer une fenêtre et un contexte openGL sans freeglut.
Ces fonctions sont déclarées dans le module **ModOpenGLTools**.

7.4. Les types de données

Le suffixe des procédures est fonction du type de données attendu.

suffixe	type de données	type de données VB
b	Byte	Byte
s	Short	Integer
i	Integer	Long
f	Float	Single
d	Double	Double
ub	Unassigned Byte	Byte
us	Unassigned Short	Integer
ui	Unassigned Integer	Long

On peut également trouver un suffixe **v** qui désigne un paramètre attendu sous forme de tableau.

Par exemple :
- **glNormal3d** attend 3 paramètres de type **Double**
- **glNormal3i** attend 3 paramètres de type **Long**
- **glNormal3iv** attend 1 paramètre de type **tableau de Long**

7.5. Les buffers

OpenGL utilise plusieurs **Tampons** (ou **Buffers**).
Ces buffers contiennent des informations relatives à la scène que nous dessinons.

7.5.1. Le tampon de couleurs

Egalement appelé tampon chromatique, il contient la couleur de chaque pixel de la scène.
Le paramètre **GLUT_RGBA** que nous avons utilisé pour initialiser *freeglut* demande l'utilisation d'un tampon de couleurs contenant 4 composantes par pixels : R pour rouge, G pour vert, B pour bleu, et A pour le canal Alpha qui détermine la transparence.

C'est sur ce tampon que nous dessinons.
Ce tampon est vidé en utilisant le paramètre **GL_COLOR_BUFFER_BIT** de la fonction **glClear**.

7.5.2. Le tampon de profondeur

Appelé également **Z-Buffer**, ce tampon contient la distance des pixels par rapport à l'observateur.

La couleur utilisée dans le tampon de couleur est la couleur qui a la plus faible valeur dans le tampon de profondeur.
En cas de transparence, c'est un peu plus complexe : la couleur affichée est définie par le mode de mélange de couleurs choisi.

Ceci est géré par *OpenGL* :
- si on a défini l'utilisation de ce tampon par l'utilisation de **GLUT_DEPTH** lors de l'initialisation du mode d'affichage de *freeglut*.
- et si on active les tests de profondeur avec l'appel à la fonction : **glEnable GL_DEPTH_TEST**.

Ce tampon est vidé en utilisant le paramètre **GL_DEPTH_BUFFER_BIT** de la fonction **glClear**.

7.5.3. Le tampon d'accumulation

Ce tampon peut cumuler plusieurs images.
On ne dessine pas directement sur ce tampon.
On transfère les pixels depuis ou vers le tampon de couleurs.

Une utilisation courante de ce tampon est l'anti-aliasing (lissage), obtenu en cumulant plusieurs images légèrement décalées.

Pour utiliser ce tampon il faut, à l'initialisation du mode d'affichage de *freeglut*, préciser le paramètre **GLUT_ACCUM**.

Ce tampon est vidé en utilisant le paramètre **GL_ACCUM_BUFFER_BIT** de la fonction **glClear**.

7.5.4. Le tampon pochoir

Plus souvent appelé tampon **stencil**.
Ce tampon permet de restreindre l'affichage à une zone réduite de l'image.

Les utilisations courantes de ce tampon sont :
- les effets d'ombre
- les effets mirroir
- l'affichage à travers un trou (serrure, fenêtre…)

Pour utiliser ce tampon il faut, à l'initialisation du mode d'affichage de *freeglut*, préciser le paramètre **GLUT_STENCIL**.

Ce tampon est vidé en utilisant le paramètre **GL_STENCIL_BUFFER_BIT** de la fonction **glClear**.

7.5.5. Le double tampon

Vous avez sans doute remarqué l'utilisation du mode d'affichage **GL_DOUBLE** pour notre première fenêtre.

Ce paramètre définit l'utilisation du double tampon (ou double buffer).
C'est le tampon de couleurs qui est doublé.

C'est un paramètre très important car cela permet de réduire les scintillements à l'affichage.
En effet, au lieu de directement dessiner un par un les éléments sur la fenêtre, on va dessiner sur un tampon caché puis intervertir les deux tampons une fois tout le dessin réalisé.

Le changement de tampon se fait à l'aide de la fonction **gluSwapBuffers**.
L'affichage de la fenêtre est mis à jour au moment de l'appel à cette fonction.

7.6. Les unités

L'unité n'est pas le cm, le mètre ou autre...
Tout est relatif, à vous de définir l'unité de votre scène.
Si par exemple vos créez une scène sur laquelle vous souhaitez déplacer un personnage, vous pouvez arbitrairement décider qu'une valeur de 1 correspondra à 1 mètre.
Ainsi se déplacer de 0,5 correspondra à un déplacement de 50 cm.
Pour créer un cube de 2 mètres de côté, vous créerez un cube de 2 de côté.
Toute la scène sera alors à l'échelle.

Il suffit de choisir une unité en fonction de la scène dessinée pour se simplifier les calculs.
Si vous créez un monde à l'échelle d'un insecte, vous prendrez peut-être plutôt 1cm pour une unité openGL.

7.7. Les vertices

Un **vertice** est un **point**. Vous lirez également souvent le mot anglais **vertex**.

Les vertices sont la base du dessin en 3D avec OpenGL

Ces vertices sont regroupés en primitives géométriques (point, triangle, rectangle...)

7.8. Les primitives géométriques

Pour dessiner une primitive géométrique, il faut d'abord appeler la fonction **glBegin** en précisant la primitive souhaitée.

Voir tableau en ligne : Lien168

Chaque vertice est défini par ses coordonnées à l'aide d'une fonction **glVertex***.

La primitive doit se terminer par un appel à **glEnd**.

On peut, entre **glBegin** et **glEnd**, cumuler plusieurs primitives géométriques de même type.

Testons l'affichage d'un triangle :

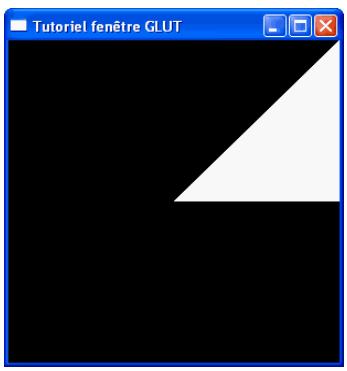
Après le vidage des buffers (*glClear*) et avant l'échange de buffer (*glutSwapBuffers*), ajoutons le code de dessin d'un triangle.
Pour bien structurer le programme, nous allons ajouter une procédure **Render** dans laquelle nous déplaçons le code de vidage des tampons.

Appel de la fonction de rendu
```
Public Sub CallBackDraw()
' Appel de la fonction de rendu
Call Render
' Echange les buffers
glutSwapBuffers
End Sub
```

Tout le code de dessin est regroupé dans la fonction **Render**, qui pourra ainsi être appelée de divers endroits du code si besoin.

Dessin d'un triangle
```
Public Sub Render()
' Vide les buffers couleur et profondeur
glClear GL_COLOR_BUFFER_BIT Or
GL_DEPTH_BUFFER_BIT
' Début de la primitive
glBegin GL_TRIANGLES
    ' Ajoute les trois sommets
    glVertex2d 0, 0
    glVertex2d 1, 0
    glVertex2d 1, 1
' Fin de la primitive
glEnd
End Sub
```

Exécutez la fonction **FonctionOpenGL** pour visualiser le triangle.



Vous l'avez sans doute remarqué, nous avons utilisé des coordonnées à deux dimensions pour dessiner notre triangle.
Malgré tout, on a bien un affichage en trois dimensions, le triangle a été placé avec une profondeur de 0 sur l'axe Z.
glVertex2d 1, 1 est équivalent à *glVertex3d 1, 1,0*.

7.9. Les couleurs des vertices

Le triangle ainsi dessiné est blanc. C'est la couleur par défaut.
On peut affecter une couleur différente à chaque point du triangle.

Il suffit d'appeler une fonction **glColor***.

On ne gère pas de transparence, on peut donc utiliser une fonction à 3 paramètres.

Ajoutez juste après l'appel à **glBegin** la fonction de changement de couleur :

Color le triangle en jaune
```
' Début de la primitive
glBegin GL_TRIANGLES
    ' Couleur jaune
    glColor3d 1, 1, 0
[...]
```

Un seul appel à la fonction **glColor*** suffit à colorer tous les vertices.
En fait la couleur jaune reste la couleur de tous les vertices qui seront ensuite dessinés par le programme.
Si on souhaite réinitialiser la couleur, il faut redéfinir la valeur par défaut (couleur blanche) :

Réinitialise la couleur
```
    ' Couleur blanche par défaut
    glColor4d 1, 1, 1,1
```

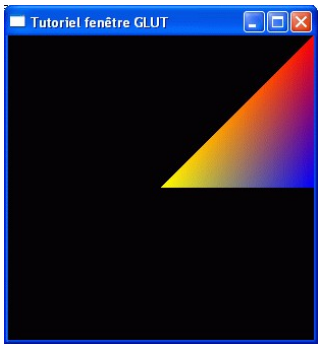
En VB on utilise souvent un entier long pour définir la couleur, notamment les constantes vbWhite, vbRed, vbGreen…

Pour simplifier le choix des couleurs, j'ai ajouté au module **ModOpengGLTools** les fonctions **glColor3VB** et **glColor4VB**.
On peut leur donner en argument une couleur de type entier long.

Utilisons ces fonctions pour affecter à chaque point une couleur différente :

Affecter à chaque point une couleur différente
```
Public Sub Render()
' Vide les buffers couleur et profondeur
glClear GL_COLOR_BUFFER_BIT Or
GL_DEPTH_BUFFER_BIT
' Début de la primitive
glBegin GL_TRIANGLES
    ' Ajoute les trois sommets
    glColor3VB vbYellow ' Jaune
    glVertex2d 0, 0
    glColor3VB vbBlue ' Bleu
    glVertex2d 1, 0
    glColor3VB vbRed ' Rouge
    glVertex2d 1, 1
' Fin de la primitive
glEnd
End Sub
```

Nous remarquons que les couleurs s'appliquent en dégradé :



Retrouvez la suite de l'article de Thierry Gasperment en ligne : Lien169

Témoignages Office 2010

Découvrez les témoignages de nos spécialistes autour d'Office 2010.

1. Fabrice Constans

Si Microsoft avait mis le paquet sur l'interface lors de la sortie d'Office 2007, l'équipe de Seattle ne s'est pas reposée sur ses lauriers pour sortir cette nouvelle version 2010. En effet de nombreuses évolutions ont vu le jour notamment pour Microsoft ACCESS, son produit de gestion de base de données.
Les essais de cette nouvelle version confirment la volonté de Microsoft de ne pas lâcher de terrain sur le secteur de la base de données grand public. Sa prise en main est rapide grâce à la réorganisation de certains menus et les nouvelles fonctionnalités sont conséquentes. Microsoft ne tient pas à abandonner le moteur de base de données qui a fait le succès d'ACCESS et le prouve en adjoignant un puissant système de triggers. Les néophytes ne sont pas en reste puisque l'éditeur de macros a été remanié rendant encore plus accessibles les nombreuses fonctions d'ACCESS.
Le nouveau look ne s'arrête pas à l'interface de développement : les boutons de commandes des formulaires sont dotés de nouveaux effets et de superbes possibilités graphiques. De quoi mettre un terme à l'austérité légendaire des applications.
Outre la disparition de la sécurité utilisateur depuis la version 2007 on peut regretter l'absence d'évolutions de l'éditeur VBA ; on aurait aimé pouvoir modifier le code des bibliothèques personnelles sans devoir tout fermer.
Bref les avantages sont un sérieux motif pour évoluer vers cette superbe version surtout que, comme son prédécesseur il utilise le format accdb, ce qui garantit une migration sans surprise de vos applications.

Fabrice C.

2. Morgan Billy

J'ai eu l'occasion de tester le Technical Preview de la version 2010 de la suite Office. Cet opus a reçu des nouveautés qui permettent d'offrir un bond en avant à Access. Mes préférences parmi les nouvelles fonctionnalités sont :

- L'éditeur de macros qui a été complètement repensé. Il se veut plus accessible aux débutants et donne accès à de nouvelles fonctions. Pour les plus expérimentés, il évitera quelques lignes de code. La présentation se veut plus orientée développeur dans sa disposition mais très conviviale dans l'ergonomie (mon article (Lien170)).
- Le générateur d'expression qui a été complètement revu au niveau du design (enfin) et affiche les formules comme Excel afin de se repérer avec les séparateurs.
- La mise en forme conditionnelle qui a reçu de nouvelles conditions (critères) ainsi qu'une interface épurée. Ma préférence va à l'intégration de barres de progression dans l'affichage (très ressemblant à Excel).
- Une partie design sur les contrôles enfin améliorée, avec le changement de couleur sur le survol d'un bouton.

Je pense que cette nouvelle version se veut accessible aux débutants mais amène des fonctionnalités, pour les plus expérimentés, non négligeables. Lorsque je compare les nouveautés qu'avait subi Access 2007 et maintenant 2010, j'ai vraiment l'impression que celles-ci constituent de belles avancées.
Il y a quelques mois, des bruits couraient sur l'abandon d'Access par Microsoft. Avec mes tests de la version 2010, je peux vous affirmer que ce n'est pas pour demain, je dirais même qu'Access se rapproche des autres SGDB.
Access n'a pas sonné ses dernières heures.

Morgan B.

3. Christophe Warin

L'ouverture du programme bêta au grand public dans les semaines à venir marque avant tout la fin de la Technical Preview. L'heure sera alors essentiellement à la vérification de la localisation, les fonctionnalités quant à elles ne devraient pas subir de grands bouleversements. Comme cela a été fait pour Access 2007, il est temps de réaliser un premier bilan des nouveautés et surtout de déterminer si, oui ou non, cette nouvelle mouture se veut indispensable.
Si, de prime abord, j'avais été littéralement conquis par Access 2007, je me suis finalement un peu ravisé, constatant que la plupart des nouveautés n'étaient pas vraiment orientées vers le développement professionnel : les développeurs envisageant la montée en puissance de leur application ont en effet tendance à fuir des fonctions aussi spécifiques que les champs à valeurs multiples ou les champs pièces-jointes. Il ne restait guère plus que le ruban comme lot de consolation bien que celui-ci ne soit pas paramétrable dans l'environnement de développement.

De ce fait, c'est avec un peu de retenue que j'ai entamé la Technical Preview 2010.
Et j'ai été agréablement surpris.
D'une part, au niveau de l'interface, l'ergonomie du ruban reste de mise et est renforcée par l'amélioration des nombreux assistants et plus particulièrement l'éditeur de macros. Celui-ci se voit complètement métamorphosé, améliorant la clarté des actions en comparaison à l'obsolète tableau illisible. C'est aussi le cas de la mise en forme conditionnelle (dont le nombre d'expressions a augmenté), du générateur d'expressions...
D'autre part, sur le plan technique, il faut noter une réelle volonté de rapprocher Access des plus grands SGBD tels que SQL Server. Tout d'abord, les champs calculés viennent apporter une simplification de la présentation des données permettant d'afficher des calculs simples sans avoir recours systématiquement à des requêtes. Ensuite, les événements de tables combinés aux datamacros qui, bien que l'on puisse regretter qu'ils ne soient pas développés en SQL mais à base de langage macro, se révèlent particulièrement efficaces. Ils permettent de confier au moteur de base de données des règles de gestion fondamentales qui, par le passé, étaient traitées en VBA avec un risque d'incohérence en cas d'attaque depuis un programme tiers. Enfin, la table USysApplicationLog, véritable journal répertoriant les erreurs des macros de données à l'instar des observateurs d'événements des plus gros systèmes.
Pour conclure, en quelques mots, Access 2010 semble marquer un véritable tournant dans le développement d'applications Access professionnelles garantissant encore un peu plus l'intégrité et la cohérence des données sans pour autant négliger l'aspect graphique des productions (look des nouveaux contrôles) ni le confort du développeur. En bref, un produit à ne pas manquer pour tous ceux qui désirent des applications plus belles, plus puissantes et plus stables.

Christophe W.

Retrouvez ces témoignages en ligne : Lien171

Les bonnes pratiques de la programmation des macros de données

Toujours à propos des évènements de table d'Access 2010, je vous propose un nouveau document consacré cette fois-ci à la méthodologie de cette nouvelle approche.
Il est destiné à un public d'un niveau intermédiaire connaissant déjà les concepts mis en jeu dans une base de données et dont la curiosité l'amène à rechercher les solutions les plus optimales. Vous y découvrirez les quelques pièges à éviter et les optimisations possibles, tout en gardant à l'esprit qu'il s'agit là d'une nouvelle fonctionnalité peu documentée puisqu'issue d'une version en cours de développement (Access 2010 Beta)

1. Introduction

A plusieurs reprises, nous avons illustré les bienfaits des évènements de table au sein d'une base de données Access. Pourtant, leur utilisation excessive ou inadaptée peut conduire à l'apparition de problèmes graves mettant en jeu la cohérence des données. Etant donné qu'il serait fastidieux de lister un à un les exemples où les évènements de table peuvent être utilisés, je vous propose un guide résumant les points essentiels devant attirer toute votre attention.

2. La structure du code

Avant de rentrer dans le vif du sujet et de se concentrer sur les données de l'application, arrêtons-nous quelques instants sur l'exemple de code VBA suivant :

```
Private Sub btnBlue_Click()
    txtNom.ForeColor = vbBlue
    txtPrenom.ForeColor = vbBlue
    txtAdresse.ForeColor = vbBlue
    txtVille.ForeColor = vbBlue
    txtCP.ForeColor = vbBlue
End Sub

Private Sub btnRed_Click()
    txtNom.ForeColor = vbRed
    txtPrenom.ForeColor = vbRed
    txtAdresse.ForeColor = vbRed
    txtVille.ForeColor = vbRed
    txtCP.ForeColor = vbRed
End Sub

Private Sub btnGreen_Click()
    txtNom.ForeColor = vbGreen
    txtPrenom.ForeColor = vbGreen
    txtAdresse.ForeColor = vbGreen
    txtVille.ForeColor = vbGreen
    txtCP.ForeColor = vbGreen
End Sub
```

Qui n'a jamais hurlé devant un tel code lorsqu'il était nécessaire de rajouter une nouvelle zone de texte ? Il aurait pourtant été si facile et préférable d'écrire :

```
Private Sub btnBlue_Click()
    Call sub_txtColor(vbBlue)
End Sub

Private Sub btnRed_Click()
    Call sub_txtColor(vbRed)
End Sub

Private Sub btnGreen_Click()
    Call sub_txtColor(vbGreen)
End Sub


Sub sub_txtColor(vColor As Integer)
    txtNom.ForeColor = vColor
    txtPrenom.ForeColor = vColor
    txtAdresse.ForeColor = vColor
    txtVille.ForeColor = vColor
    txtCP.ForeColor = vColor
End Sub
```

Le même principe peut être appliqué aux macros de données. Ainsi, si les évènements **Après Insertion** et **Après MAJ** doivent produire les mêmes traitements, inutile de pratiquer du copier/coller. Créez simplement une macro de données nommée **AprèsInsertetMaj** et lancez la dans chacun des évènements grâce à la méthode **ExécuterMacroDonnées**. De plus, comme sous VBA, il est possible de définir des paramètres pour les sous-macros.

3. Modélisation

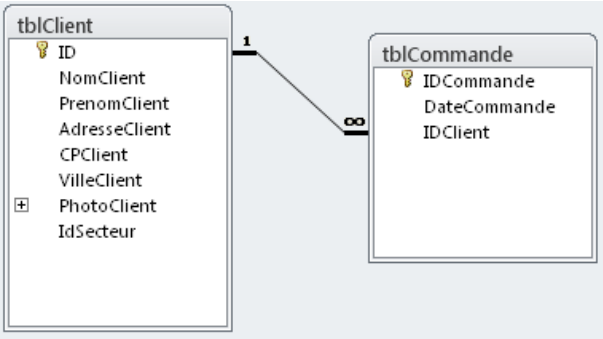
Les évènements de table introduits avec Microsoft Access 2010 permettent de mettre en place la quasi-totalité des traitements qu'il est possible de faire subir à un jeu de données. Ils peuvent être utilisés pour valider la saisie de l'utilisateur (nous verrons cette partie au chapitre suivant) ou bien pour transposer la saisie dans une autre structure en y appliquant des règles de gestion jusque-là confiées à VBA via un formulaire. Comme pour chaque nouveauté Access, le public utilisateur peut être divisé en 3 catégories :

- Les récalcitrants : ils n'utiliseront jamais les évènements de table, persuadés qu'ils sont peu fiables ou inutiles bien qu'ils fassent partie de ceux qui les ont réclamés pendant longtemps.
- Les utilisateurs avertis : ils connaissent la fonctionnalité et savent en tirer profit.
- Les novices : ils viennent de découvrir la nouveauté et ont besoin de parfaire leurs connaissances avant de la mettre en place dans un cadre professionnel.

C'est justement à cette troisième catégorie que ce document, et plus particulièrement ce chapitre, est adressé. En effet, si tous les autres chapitres tiennent plus de l'optimisation et du perfectionnement d'une base, celui-ci et dans une moindre mesure le suivant, seront les garants d'une application stable, robuste, évolutive et maintenable.

Le principal risque réside dans une dénormalisation de la structure de la base. Celle-ci pourra être « volontaire » (le concepteur a sciemment ignoré les règles) ou « involontaire » (les maintenances successives et le recours systématique aux évènements de table ont conduit à des duplications de structure faisant du projet une véritable usine à gaz)

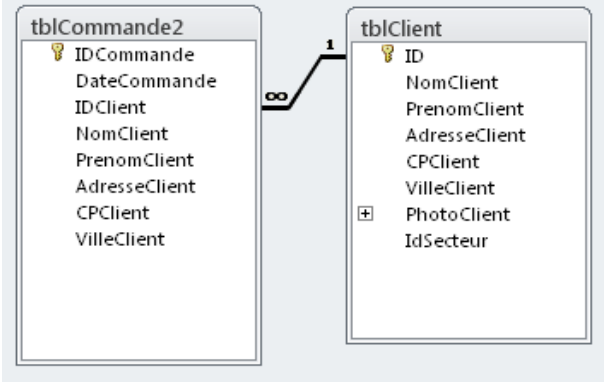
Prenons l'exemple d'une gestion de commandes :



Cet exemple est le cas d'école le plus courant : des commandes, des clients, les informations du client sont disponibles via la relation des deux tables, etc. Oui mais...

Cet exemple n'est pas juste, que va-t-il se passer lorsque le client va changer d'adresse ? **Réponse** : la nouvelle adresse va être utilisée par toutes les commandes y compris celles datant de l'ancienne domiciliation. **Résultat** : des statistiques géographiques fausses, des documents comptables faux, etc.

Partant de ce constat, par simplicité, le rapatriement des données du client dans la table des commandes est réalisé aveuglément et mène au schéma et à la macro de données ci-dessous :




```
/* Recherche des informations du client
  Rechercher un enregistrement dans    tblClient
                    Condition Where
ID=[tblCommande2].[IDClient]
                                Alias    recClient
    DefinirVarLocale    (vNom;[NomClient])
    DefinirVarLocale    (vPrenom;[PrenomClient])
    DefinirVarLocale    (vAdresse;[AdresseClient])
    DefinirVarLocale    (vVille;[VilleClient])
    DefinirVarLocale    (vCP;[CPClient])

/* Edition de la commande
  ModifierEnregistrement
                                Alias
    DefinirChamp    (NomClient;vNom)
    DefinirChamp    (PrenomClient;vPrenom)
    DefinirChamp    (AdresseClient;vAdresse)
    DefinirChamp    (VilleClient;vVille)
    DefinirChamp    (CPClient;vCP)
  Terminer ModifierEnregistrement
```

D'une part, la redondance des champs **NomClient** et **PrenomClient** consomme inutilement de l'espace de stockage et d'autre part, un œil extérieur pourrait rapidement conclure que la table **tblClient** est finalement inutile, menant un jour à sa suppression puis un peu plus tard à sa restauration mais avec une conception inversée : les commandes sont saisies dans la structure **tblCommande**, puis si le client n'existe pas, il sera ajouté à la table **tblClient**.

```
/*  Vérifie si le client existe dans la table
tblClient

  DefinirVarLocale    (vTrouve;False)
  Rechercher un enregistrement dans    tblClient
                    Condition Where
[NomClient]=[tblCommande3].[NomClient]
                                Alias    recClient

  DefinirVarLocale    (vTrouve;Not
IsNull([NomClient])
```

```
  Si Not [vTrouve]  Alors

     Creer un enregistrement dans     tblClient
                                Alias
recNewClient

          DefinirChamp  (NomClient;
[tblCommande3].[NomClient])
          DefinirChamp  (PrenomClient;
[tblCommande3].[PrenomClient])
          DefinirChamp  (AdresseClient;
[tblCommande3].[AdresseClient])
          DefinirChamp  (VilleClient;
[tblCommande3].[VilleClient])
          DefinirChamp  (CPClient;[tblCommande3].
[CPClient])
  Fin Si
```

Comme indiqué ci-dessus, quelques lignes seulement de code très basiques ont permis de dénormaliser complètement la structure de la base de données. A de rares expressions près, il est important de considérer que les évènements de table doivent intervenir au sein d'un modèle de données normalisé. Les autres cas consisteront en une solution de remplacement de règles de gestion complexes. Un exemple parmi tant d'autres : la gestion d'un stock FIFO (premier entré, premier sorti). Il est plus facile de décrémenter à chaque sortie les entrées correspondantes plutôt que d'établir une formule récursive lorsque l'utilisateur souhaite valoriser son stock.

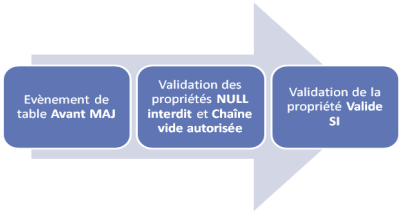
4. Validation des données

Les évènements de table **Before *** (**Avant Suppression** et **Avant Modification**) permettent de vérifier que les nouvelles données insérées dans la table respectent des critères établis. Leur architecture se déroule autour d'une gestion d'erreur : dès qu'une règle de gestion n'est pas respectée, l'action **DéclencheErreur** permet d'avertir l'utilisateur et de stopper la mise à jour des données.

Avant toute utilisation des évènements de table afin de valider des données, il est primordial d'une part de connaître l'ensemble des propriétés de contrôles déjà disponibles et d'autre part de comprendre comment est effectuée la validation lors de mise à jour en masse.

L'ordre des différents tests effectués par le moteur de base de données est important afin de déterminer si certains d'entre eux sont superflus. Par exemple, si la vérification par évènement de table intervient après la vérification de la règle **Null Interdit**, il sera possible de considérer dans toute la macro que la valeur du champ est renseignée et donc, par conséquent, qu'un recours à **Nz** ou **IsNull** sera inutile.

Malheureusement, ce n'est pas le cas, les évènements de table interviennent en amont de tout autre traitement. Dans le cas d'une mise à jour, le processus de vérification peut être schématisé de la façon suivante :



A la grande question : "Faut-il gérer les règles **Valide Si**, **Null interdit** et **Chaîne vide autorisée** dans la macro d'évènement de table ?", difficile de répondre. Au regard du peu de ressources nécessaires, j'aurais tendance à "doublonner" les vérifications. (Par exemple : écarter les **NULL** dans la macro tout en fixant la propriété **Null Interdit** à **Oui**). Toutefois, quelle que soit votre décision, il est important que celle-ci reste la même au sein de tout le projet et que ces règles de validation soient suffisamment détaillées dans la documentation afin d'éviter deux versions différentes de tests au sein de deux propriétés d'une même table.

Passé cette question, il est nécessaire de garder à l'esprit, qu'entreront dans le processus de validation, des données que l'on aurait pu croire refoulées par les autres propriétés (qui interviennent hélas trop tard).

Autre point important : la portée de l'erreur. Celle-ci n'a pas vraiment d'incidence lorsque les mises à jour sont réalisées ligne à ligne. Toutefois, lorsque c'est une requête **action** qui est à l'origine du processus, cette notion devient capitale. Pour mettre en évidence nos propos, prenons l'exemple d'une table **tblHistorique** dont le contenu ne doit pas dépasser 10 lignes.

Structure de la table **tblHistorique** :

Nom du champ	Type	Extra
IDLigne	Numéro Auto	Clé primaire
IDEvenement	Numérique	
DateEvenement	Date	

Code de la macro de données **Avant Modification** :

```
Si [Insertion] Alors
  Rechercher un enregistrement dans
qryCountHistorique
                    Condition Where
                                Alias  recHisto
  Si [NB]>=10 Alors
        DeclencherErreur
                  Numero de l'erreur  10001
          Description de l'erreur  Taille
maximale atteinte
  Fin Si
Fin Si
```

Requête qryCountHistorique :

```
SELECT Count(*) AS NB
FROM tblHistorique;
```

Constituons un jeu d'essai :

IDLigne	IDEvenement	DateEvenement
1	1	21/11/2009 16:14:40
2	3	21/11/2009 16:14:44
3	2	21/11/2009 16:14:46
4	1	21/11/2009 16:14:47
5	1	21/11/2009 16:14:40
6	3	21/11/2009 16:14:44
7	2	21/11/2009 16:14:46
8	1	21/11/2009 16:14:47
9	3	22/11/2009 16:14:00
10	3	22/11/2009 16:14:04

L'insertion d'une nouvelle ligne directement depuis la table provoque l'affichage du message suivant :

Supprimons maintenant les deux derniers enregistrements et tentons d'exécuter la requête ci-dessous :

```
INSERT INTO tblHistorique ( IDEvenement,
DateEvenement )
SELECT IDEvenement, DateEvenement
FROM tblHistorique;
```

Il s'agit simplement de dupliquer les enregistrements de la table **tblHistorique** déjà présents. Trois hypothèses peuvent être envisagées :

- Seulement deux enregistrements vont être copiés, les autres ne seront pas validés, la table aura atteint sa taille maximale.
- Tous les enregistrements seront copiés, la règle de validation n'ayant lieu qu'au début de la transaction.
- Aucun enregistrement ne sera copié, la règle de validation ayant lieu au début de la transaction.

C'est cette troisième hypothèse qui est utilisée par le moteur de base de données. On parle alors de **contrainte au niveau de la transaction**. Si la règle n'est pas respectée par un seul des échantillons de la transaction, l'ensemble des actions de celle-ci est invalidé. Ajoutons que si l'ordre d'exécution est lancé depuis la méthode **Execute** d'un objet DAO, l'erreur levée ne possède pas le numéro qui a été défini dans la macro mais un numéro standard pour toutes les erreurs de validation : **3939**. (La description de l'erreur est quant à elle conservée).

Difficile de parler de véritable limitation. Cela en devient véritablement une si le développeur méconnaît ces mécanismes mais s'il adapte ses projets en conséquence, il ne devrait pas rencontrer de problèmes majeurs.

5. Identification des données

Dans un précédent article (Création d'une numérotation personnalisée (Lien172)), nous avons vu comment tirer avantage des méthodes des macros de données afin de réaliser une séquence pour la numérotation de factures. La question principale que se posent plusieurs lecteurs : que se passe-t-il en cas d'accès concurrent ?

Pour illustrer les mécanismes mis en jeu, plaçons-nous dans un environnement multi-utilisateurs avec les éléments suivants :

- Deux utilisateurs : A et B
- Une table tblFacture(NumFacture,IDClient,DateFacture)
- NumFacture est clé primaire et calculé par un évènement de table **Après Insertion** que nous ne développerons pas ici.

Voici le contenu de la table **tblFacture** à t=0

NumFacture	IDClient	DateFacture
FA20091001	1	12/10/2009
FA20091002	2	13/10/2009

A t=1, les deux utilisateurs créent une nouvelle facture. En partant du principe que l'utilisateur A est à l'origine du document daté du 14/10/2009 et l'utilisateur B à l'origine du document daté du 15/10/2009, voici le contenu de la table **tblFacture** disponible pour chaque utilisateur,

Utilisateur A :

NumFacture	IDClient	DateFacture
FA20091001	1	12/10/2009
FA20091002	2	13/10/2009
A calculer	*2*	*14/10/2009*

Utilisateur B :

NumFacture	IDClient	DateFacture
FA20091001	1	12/10/2009
FA20091002	2	13/10/2009
A calculer	*1*	*15/10/2009*

A cet instant, chacun des deux utilisateurs pense créer la facture FA20091003. Cette erreur présente-elle un problème ? Oui, si le champ **NumFacture** n'est pas indexé sans doublon car deux factures portant le même identifiant comptable seront créées, ce qui, je ne vous le cache pas est un cauchemar pour les comptables qui devront justifier de cet incident. En revanche, si le champ est correctement indexé, une erreur sera levée pour l'utilisateur appliquant ses modifications en dernier. Bien évidemment, le risque est minime mais il doit toutefois alerter sur les dangers du temps de traitement des évènements de table. Plus celui-ci augmente pour un utilisateur, plus le risque que d'autres exécutent des actions sur le même jeu de données est important.

6. La Récursivité

La récursivité est la notion qui intervient lorsqu'un évènement de table exécute une action conduisant à un nouveau déclenchement de cet évènement, que celui-ci s'exécute sur le même enregistrement ou sur un autre.

6.1. Récursivité non désirée

Pour illustrer ce terme, prenons l'exemple d'un professeur paranoïaque souhaitant stocker la date de mise à jour de ses notes dans sa base de données afin de détecter la moindre tentative de fraude.

Voici la structure de sa table **tblResultat** :

Nom du champ	Type	Extra
IDNote	Numéro Auto	Clé primaire
IDEleve	Numérique	Relié à la table tblElève
IDDevoir	Numérique	Relié à la table tblDevoir
Note	Numérique	
DateMaj	Date	

La valeur du champ **DateMaj** est modifiée via une macro de données **MacroMaj** exécutée sur les évènements **Après Insertion** et **Après MAJ.**

```
ModifierEnregistrement
  DefinirChamp(DateMaj;Now())
Terminer ModifierEnregistrement
```

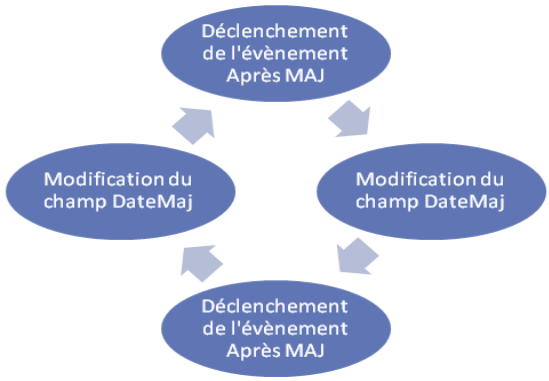
A l'utilisation, notre professeur ne constate aucun dysfonctionnement. La table est mise à jour correctement à chaque insertion et à chaque mise à jour.

IDNote	IDEleve	IDDevoir	Note	DateMAJ
1	12	1	10	19/11/2009 14:11:50
2	13	1	11	19/11/2009 14:12:13
3	14	1	0	19/11/2009 14:12:24
4	15	1	20	19/11/2009 14:12:27

Cependant, la visualisation de la table **USysApplicationLog** permet de mettre en évidence un défaut majeur :

Category	Context	Created	Description	Error Number
Execution	EditRecord	19/11/2009 14:12:14	Une limite de ressources a été atteinte pour les macros de données.	-20341
Execution	EditRecord	19/11/2009 14:12:24	Une limite de ressources a été atteinte pour les macros de données.	-20341
Execution	EditRecord	19/11/2009 14:12:28	Une limite de ressources a été atteinte pour les macros de données.	-20341

Ceci est dû à une récursivité dans l'évènement **Après MAJ :** la modification d'un enregistrement entraîne la modification d'un de ses champs (DateMAJ). Ceci constitue une nouvelle modification de l'enregistrement, ce qui entraine une nouvelle modification d'un de ses champs (DateMAJ), ce qui...



Cette erreur a deux répercussions néfastes :

- La table **USysApplicationLog** grossit inutilement
- Les opérations d'écritures sont plus lentes

Bien entendu, il est possible d'inhiber la boucle ainsi générée. Pour cela, il suffit de réaliser un test sur l'évènement **Après MAJ** afin de savoir si le champ modifié nécessite ou non de d'exécuter le reste de la macro. Le code devient alors :

```
Si Updated("Note")
  ExécuterMacroDonnées(tblResultat.MacroMaj)
Fin Si
```

La différence en termes de délai d'écriture est assez impressionnante puisque la récursivité illustrée plus haut est deux fois plus longue que l'opération « sécurisée ».

Pour information, voici la moyenne des résultats obtenus pour 1000 insertions via un **recordset** :

- Avec récursivité inutile : 0,84 secondes
- Sans récursivité : 0,43 secondes

Retrouvez la suite de l'article de Christophe Warin en ligne : Lien173

Extraction d'informations depuis un fichier de données en C

Dans certains projets, il peut être nécessaire d'extraire des données depuis un fichier. Que ce soit une liste de contacts, des données pour remplir des tableaux ou pour des tableurs, des options utilisateurs, etc... Ce tutoriel va vous expliquer les bases pour lire différents formats et en extraire les données.

1. Introduction

Dans certains projets, il peut être nécessaire d'extraire des données depuis un fichier. Que ce soit une liste de contacts, des données pour remplir des tableaux ou pour des tableurs, des options utilisateurs, etc. Ce tutoriel va vous expliquer les bases pour lire différents formats et en extraire les données.

Dans un premier temps *(Chapitre III)*, nous allons voir comment créer un fichier avec une structure appropriée au stockage d'une liste de contacts pour par exemple, un programme de gestion de contacts. Dans le Chapitre IV nous verrons comment lire des fichiers de type INI et en extraire les données suivant leur type (entiers, chaînes de caractères). Nous finirons par le format CSV au Chapitre V.

Le format XML (eXtended Markup Langage) (Lien174) ne sera pas traité ici, un tutoriel lui est entièrement consacré dans le tutoriel suivant : « Utilisation de la bibliothèque libxml2 en C » (Lien175)

2. Pré-requis

Quelques pré-requis sont indispensables pour les chapitres à suivre. Des fonctions que tout le monde devrait avoir constamment sous le coude, voici celles indispensables pour ce tutoriel et qui traitent des chaînes de caractères :

Fonction : str_dup

```
static char * str_dup (const char * str)
{
   char * dup = NULL;

   if (str != NULL)
   {
      size_t size = strlen (str) + 1;
      dup = malloc (size);

      if (dup != NULL)
      {
         memcpy (dup, str, size);
      }
   }

   return dup;
}
```

Cette fonction permet de copier simplement une chaîne de caractères. Le nouveau pointeur retourné par la fonction doit explicitement être libéré avec la fonction standard free (Lien176).

La seconde fonction indispensable dans ce tutoriel permet de terminer une chaîne de caractères avec un zéro de fin de chaîne, ce qui n'est pas toujours le cas surtout dans des chaînes extraites d'un fichier :

Fonction : str_finalize

```
static void str_finalize (char * str)
{
   char * p = strchr (str, '\n');

   if (p != NULL)
   {
      *p = 0;
   }
}
```

Cette fonction recherche un éventuel saut de ligne et le remplace par un zéro terminal. Les sauts de ligne sont surtout présents lorsqu'on récupère des lignes de texte à partir d'un fichier mais les fonctions standards attendent un zéro de fin de chaîne.

3. Traitement d'un fichier avec une structure personnalisée

Dans ce premier chapitre, nous allons voir comment récupérer des informations dans un fichier avec une structure de données personnalisée soit, la structure suivante :

```
{
   name:Franck.H
   age:32
   address:16 rue du General De Gaulle, 67000
Strasbourg
   phone:0618325015
}
```

Dans ce type de structure, les accolades délimitent un bloc de données, ce qui nous permet donc de déterminer à quel moment dans le code, il faut commencer à tester et récupérer les informations. Le caractère **':'** sert de séparateur entre une donnée et son nom de champ dans le fichier, on pourrait aussi mettre par exemple le caractère **'='** qui est plus souvent utilisé dans ce cas de figure.

3.1. Etude algorithmique

Il faut pour commencer, déterminer un algorithme et donc un cheminement permettant de partir de l'ouverture d'un bloc de données jusqu'à, dans notre cas, la recherche du nom d'un contact puis de récupérer ses données personnelles. Cela vaut pour tout autre programme. Voici un algorithme possible :

```
Ouvrir fichier
   Tant que ligne <> NIL lire ligne fichier
      Si caractère 0 de ligne égal à '{'
         alors ouvrir bloc
         sinon si bloc ouvert
            Si recherche label égal à "name"
               alors
                  Récupérer l'age
                  Récupérer l'adresse
                  Récupérer le téléphone
            Fsi
      Fsi
   Fin tant que
Fermer fichier
```

Voici une brève explication du cheminement :

1. On ouvre le fichier à lire
2. On lit le fichier ligne par ligne par le biais d'une boucle **Tant que** *(while)* tant qu'on ne se trouve pas à la fin du fichier.
3. On teste sur chaque ligne et tant qu'un bloc n'a pas été ouvert, si le premier caractère de la chaîne est un caractère **'{'**
4. Si un bloc n'a pas encore été ouvert et que le premier caractère est **'{'**, on ouvre le bloc en initialisant une variable d'état (généralement de type booléen mais un type entier convient très bien aussi). On saute ensuite à la prochaine ligne.
5. Si un bloc a déjà été ouvert, on recherche le nom du contact à retrouver en commençant par rechercher l'enregistrement portant l'identifiant **"name"**. Si cet identifiant a été trouvé, on compare le nom qui le suit avec le nom à retrouver.
6. Si le contact a été retrouvé, on récupère une à une ses informations.
7. Fermeture du fichier

3.2. Implémentation

Attelons-nous maintenant à l'écriture du code en C. Il nous faut pour commencer, inclure les fichiers d'en-tête adéquats :

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

Nous devons maintenant préparer diverses parties pour pouvoir développer notre futur code. Nous allons plutôt utiliser des directives du pré-processeur pour définir quelques constantes symboliques ce qui permettra une plus grande facilité dans la maintenance future du code *(il faut toujours penser à la maintenance du code)* et aussi, rendra le code plus lisible.

```
#define BEGIN        '{'
#define SEP          ':'
#define BUF_SIZE     128
#define FILENAME     "perso.txt"

#define TAG_NAME     "name"
#define TAG_AGE      "age"
#define TAG_ADRESS   "address"
#define TAG_PHONE    "phone"
```

Les constantes commençant par **"TAG_"** sont en fait les différents champs qu'on retrouve dans un bloc de données du fichier. La constante **BUF_SIZE** permet de définir la limite d'un tableau de caractères donc le nombre maximum de caractères que peut contenir une chaîne après la lecture.

3.2.1. Fonction : get_infos

```
contact_infos * get_info (const char *
contact_name);
```

Voici la fonction principale qui va faire presque tout le travail, elle retourne une structure de type **contact_infos** dont voici la définition :

```
typedef struct
{
   char *   name;
   char *   adress;
   char *   phone;
   int      age;
}
contact_infos;
```

et prend comme unique paramètre le nom du contact à retrouver. Voici l'implémentation de la fonction :

```
contact_infos * get_info (const char *
contact_name)
{
   contact_infos  *  ret;
   FILE           *  file     = NULL;
   char              buff     [BUF_SIZE];
   int               opened   = 0;


   file = fopen (FILENAME, "r");
/* 1.- */

   if (file != NULL)
   {
      while ((fgets (buff, BUF_SIZE, file) !=
NULL))   /* 2.- */
      {
         if (buff[0] == BEGIN && !opened)
/* 3.- */
         {
            opened = 1;
/* 4.- */
         }
         else if (opened)
         {
            char * s = strstr (buff, TAG_NAME);
/* 5.- */

            if (s != NULL)
            {
               s += strlen (TAG_NAME) + 1;
               str_finalize (s);

               if (strcmp (s, contact_name) == 0)
/* 5.- */
               {
                  ret = malloc (sizeof (* ret));

                  if (ret != NULL)
                  {
                     ret->name = str_dup (s);
```

```
                    get (ret, file, AGE);
/* 6.- */
                    get (ret, file, ADDRESS);
/* 6.- */
                    get (ret, file, PHONE);
/* 6.- */
                 }

                 break;
              }
              else
              {
                 opened = 0;
              }
           }
        }
     }

     fclose (file);
/* 7.- */
  }


  return ret;
}
```

On peut revoir grâce à la numérotation en commentaire, les principales actions de l'explication du cheminement du chapitre sur l'étude algorithmique. J'ai parlé durant cette étude d'un entier qui peut servir de drapeau booléen pour déterminer l'ouverture d'un bloc de données, c'est ce que fait la variable **opened**. On peut voir dans la partie **5** du code, la recherche du tag **name**. Juste après, dans la condition **if**, on décale le pointeur sur la longueur du tag plus **1** caractère pour passer après le séparateur de champs **':'** donc au début du nom du contact.

S'en suit la comparaison entre le nom du contact retrouvé et celui passé en argument à la fonction. Si c'est la bonne personne, on alloue dynamiquement un espace pour la structure que doit renvoyer la fonction et on la remplit par le biais d'une seconde fonction nommée **get** que nous allons voir juste après.

On peut apercevoir l'utilisation de constantes en troisième argument de cette seconde fonction, voici l'énumération correspondante qui servira à récupérer la suite des informations du contact :

```
typedef enum
{
   NAME,
   AGE,
   ADDRESS,
   PHONE,

   NB_TAG
}
contact_tags;
```

3.2.2. Fonction : get

```
void get (contact_infos * p, FILE * file,
contact_tags tag);
```

Cette fonction permet de terminer la récupération des informations restantes d'un contact. Ses trois arguments sont dans l'ordre :

- Un pointeur valide sur la structure que doit renvoyer la fonction **get_infos**.
- Un pointeur valide sur le fichier en cours de lecture.
- Le tag de type *contact_tags* correspondant à l'information que doit récupérer la fonction.

Voici l'implémentation de la fonction:

```
get (contact_infos * p, FILE * file, contact_tags
tag)
{
   char     buff  [BUF_SIZE];
   char *   s     = NULL;


   if ((fgets (buff, BUF_SIZE, file)) != NULL)
/* 1.- */
   {
      s = strchr (buff, SEP);
/* 2.- */

      if (s != NULL)
      {
         s++;
/* 3.- */
         str_finalize (s);
/* 4.- */

         switch (tag)
/* 5.- */
         {
            case AGE:
            {
               p->age = strtol (s, NULL, 10);
            }
            break;

            case ADDRESS:
            {
               p->adress = str_dup (s);
            }
            break;

            case PHONE:
            {
               p->phone = str_dup (s);
            }
            break;

            default:
               break;
         }
      }
   }
}
```

A chaque appel de la fonction, on lit la ligne suivante *(1)* du fichier. A partir de la ligne récupérée, on recherche *(2)* la position du séparateur de champs. Si le séparateur a été trouvé, on décale le pointeur de 1 caractère *(3)* pour se retrouver au début de la donnée à récupérer. On finalise enfin la chaîne *(4)* afin d'y ajouter un caractère de fin de chaîne en lieu et place du saut de ligne. Nous pouvons maintenant récupérer la donnée courante *(5)* suivant le tag passé en argument.

Vous pouvez remarquer la conversion de l'âge en entier

lors de la récupération de cette donnée. Ici l'ordre de récupération des données est immuable mais il est tout à fait possible de créer une fonction plus souple mais qui demande davantage de temps et de lignes de code.

Retrouvez la suite de l'article de Franck Hecht en ligne : Lien177

Programmation par contrat, application en C++

Ce tutoriel vise à présenter de manière relativement concise les objectifs de la conception et de la programmation par contrat, ainsi que les techniques de mise en œuvre dans le langage C++. Le lecteur est supposé connaître les bases de la programmation, de l'approche orientée objet et de la généricité. Ce tutoriel s'adresse donc à des développeurs de niveau moyen à expérimenté.

1. Présentation générale, historique, contexte et objectifs

La programmation par contrat est une technique de conception (en anglais, on utilise le terme plus heureux de *design by contract*) inventée par Bertrand Meyer, qui l'a intégrée dans le langage Eiffel.

L'objectif de cette méthode est d'arriver à écrire du code plus robuste et plus facilement réutilisable. Partant du principe que la première qualité d'un code est la correction (le code fait ce pour quoi il a été conçu), les objectifs de la programmation par contrat sont :

- Réduire le temps consacré au débogage du code, en détectant les erreurs au plus tôt dans le cycle de développement.
- Faciliter la détection des erreurs.
- Augmenter la possibilité de réutilisation du code, en s'assurant qu'un code correct dans un environnement le sera aussi dans un nouvel environnement.

2. Approche fonctionnelle, définition du contrat

Un contrat, tel qu'on l'entend en programmation par contrat, est avant tout une relation de confiance entre deux parties. Dans l'approche fonctionnelle, les deux parties sont l'appelant et l'appelé. Ces deux parties s'engagent, chacune, à respecter leur part du contrat.

La terminologie la plus couramment employée utilise une analogie avec le monde réel. On parle ainsi de **fournisseur** (l'appelé *fournit* un service) et de **client** (l'appelant *fait appel* au service).

2.1. Précondition

Les préconditions, ce sont les conditions nécessaires et suffisantes à la bonne exécution de la fonction. Par exemple :

```
double racine_carree(x)
```

a une précondition assez évidente, qui est x >= 0.

En programmation classique, cette précondition n'est jamais exprimée. Elle est implicite. Aussi, le programmeur doit deviner, à partir de ce que fait la fonction, quelles sont ses préconditions. L'expérience montre que le programmeur a souvent tendance à croire que la fonction qu'il appelle en fait plus que ce qu'elle ne fait réellement. Aussi, en programmation par contrat, sont incluses, dès la conception, les conditions nécessaires et suffisantes à l'appel de la fonction. À charge de l'appelant de s'assurer qu'il les respecte.

La fonction sera donc déclarée de la sorte (en pseudo-code) :

```
double racine_carree(x) require x >= 0;
```

La notation utilisée ici n'est là que pour illustrer, et gagner en clarté. Ce n'est donc pas du code C++ valide, inutile d'essayer de le compiler. Une présentation de ce qui est possible en C++ sera donnée plus loin.

La différence peut paraître mineure, mais elle est fondamentale. En exprimant ces préconditions, on garantit à l'appelant que, s'il les respecte, on aura un comportement défini.

Qu'est-ce qu'une bonne précondition, comment dois-je définir mes préconditions ?

Une bonne précondition répond aux critères suivants :

- Elle est vérifiable par l'appelant : nous verrons pourquoi plus loin.
- Elle porte généralement sur les paramètres de la fonction.
- Je n'ai pas défini de comportement, si elle n'est pas respectée.
- Dès qu'on respecte mes préconditions, j'ai un comportement défini.

Une fonction peut bien entendu avoir plusieurs préconditions. Comme une précondition est une expression booléenne, ces préconditions peuvent se combiner au moyen des opérateurs logiques OU et ET. Sans autre précision, le défaut est l'opérateur ET. La fonction strlen, par exemple, pourrait s'exprimer comme ceci :

```
int strlen(char * s)
  require s != NULL
  require "s terminée par un \0"
```

On remarque que la deuxième précondition n'est pas réellement exprimable dans le langage. Ce n'est pas un problème, le contrat s'adressant avant tout au développeur qui, lui, est tout à fait capable de le comprendre.

2.2. Postcondition

La postcondition, c'est une propriété qui est vérifiée lorsque la fonction est terminée. Là encore, en programmation classique, elle n'est jamais exprimée. En programmation par contrat, elle sera exprimée dans la déclaration de la fonction, par exemple :

```
int carre(x) ensure return >= 0
```

Ce que cette postcondition dit, c'est que la valeur de retour de « carre » est positive. Ce qui était implicite, que le développeur devait deviner, est désormais exprimé, noir sur blanc, et garanti. Cela fait désormais partie du contrat

Comment définir une bonne postcondition ?

Les postconditions ont un gros problème par rapport aux préconditions. On ne sait pas exactement comment s'arrêter, car l'ensemble des choses que l'on peut garantir est virtuellement infini. Ainsi, on peut très bien écrire :

```
int carre(x)
  ensure return >=0
  ensure "la réponse à la question sur la vie,
l'univers et le reste est 42"
```

Aussi, pour qu'une postcondition soit utile, il faut qu'elle réponde au moins à l'une des caractéristiques suivantes :

- Elle porte sur la valeur de retour de la fonction.
- Elle porte sur l'un des paramètres de la fonction (passé par référence, par exemple).
- Elle apporte une information utile.

Le dernier point est le plus sujet à discussion. En reprenant nos deux exemples, et seulement en lisant les contrats, on sait qu'on a le droit d'écrire :

```
double x = racine_carree(carre(-3.4));
```

En effet, la postcondition de « carre » vérifie la précondition de « racine_carree », aussi, grâce à mes contrats, j'ai la garantie que mon code va fonctionner. On peut donc dire que ma postcondition m'est utile. De la même manière, la méthode push_back de la classe std::vector se déclare de la sorte :

```
std::vector::push_back(T) ensure size() ==
pre.size() + 1
```

Ou, autrement dit, on garantit qu'après push_back, la taille de mon vecteur a augmenté de 1, et donc, qu'on peut appeler pop_back() sans vérifier que la taille est supérieure à zéro.

2.2.1. Postconditions et exceptions

Définir des postconditions impose de les garantir. Ce n'est parfois pas possible. Par exemple, une méthode « open » sur un objet fichier a comme postcondition que le fichier est ouvert. Toutefois, cela serait présupposer que l'opération réussit toujours, or, elle peut échouer, pour de multiples raisons. Doit-on renoncer au contrat pour autant ?

La réponse est bien évidemment non. Les langages modernes disposent d'un mécanisme d'exception, qui peut être utilisé pour notifier à l'appelant que l'appelé n'a pas été en mesure de respecter son contrat. On ne vérifiera donc pas les postconditions en cas d'exception.

Cela donne quelque part un « guide » sur « quand utiliser des exceptions ». Si une fonction n'est pas en mesure de garantir ses postconditions, alors elle doit lever une exception.

2.3. Responsabilité du contrat

Maintenant que nous savons définir un contrat, reste à déterminer à qui échoit quoi. De manière assez logique, le respect des postconditions ne peut être que le fait de l'appelé, l'appelant n'y ayant pas accès.

De manière moins évidente, c'est l'**appelant** qui est responsable des préconditions. En effet, les préconditions sont les *conditions nécessaires et suffisantes pour appeler la fonction*. De fait, elles doivent être respectées avant même de rentrer dans le corps de l'appelé. C'est donc à l'appelant qu'elles échoient.

Le contrat est donc bien un engagement bipartite, entre l'appelant et l'appelé. L'appelant s'engage sur les préconditions, et l'appelé s'engage sur les postconditions. Si les préconditions ne sont pas respectées, il faut corriger le code de l'appelant. Si les postconditions ne sont pas respectées, il faut corriger le code de l'appelé.

2.4. Rupture du contrat

Rapidement se pose la question de la rupture du contrat. Que se passe-t-il si l'une des deux parties ne respecte pas ses engagements ? À cela, la programmation par contrat n'impose rien. Dès lors que les deux parties n'ont pas fait leur part du contrat, le programme est faux, et il est inutile d'essayer de se "*rattraper aux branches*" (sauf, bien sûr, si des vies humaines sont en jeu). Il faut avant tout corriger le programme.

Ceci s'oppose complètement à la programmation défensive, qui elle, vise à rejeter tout paramètre invalide, et à renvoyer l'appelant dans les cordes au moyen d'une exception (dont, souvent, il ne saura pas quoi faire).

Le comportement lors de la rupture de contrat est un choix, généralement fait à la compilation. Il faut donc le considérer comme un comportement indéfini.

2.5. Neutralité du contrat

Maintenant que vous savez qu'un contrat non vérifié est un comportement indéfini, et qu'un programme correct respecte toujours ses contrats, se pose la question de savoir s'il faut tester la validité du contrat.

La réponse n'est pas si évidente qu'elle en a l'air. Tester le contrat est utile, car cela permet de changer un comportement indéfini en un comportement défini (par exemple, lever une exception), qui permettra de diagnostiquer rapidement l'erreur. Mais tester le contrat a un coût, qu'il n'est pas nécessaire de payer si le programme est correct.

Il est très important de bien faire la différence entre contrat et validation de données utilisateur. Un contrat ne sert pas à vérifier que les données saisies sont correctes, il sert à exprimer le fait qu'une fonction nécessite des données correctes en entrée. La validation de données saisies est une étape normale du traitement logiciel, et en aucun cas la programmation par contrat ne vise à supprimer cette étape. En revanche, bien utilisée, elle permettra d'identifier plus facilement un manquement dans cette étape.

De fait, ce qui est souvent fait est de vérifier le respect du contrat lors de la phase de *debug*, et de ne plus le faire lors de la phase de *release*. Ce qu'il est indispensable de vérifier, c'est que le comportement du programme est le même, que cette vérification soit activée ou pas. On parle de neutralité du contrat, car celui-ci est totalement neutre du point de vue de l'exécution du programme.

À ce titre, les préconditions et postconditions ne peuvent avoir d'effet de bord.

3. Approche objet

Dans le monde objet, le contrat s'applique aussi. Sur les fonctions membres, il s'applique de la même manière que sur les fonctions classiques, mais il faut tenir compte du paramètre supplémentaire qu'est le "this" sur lequel est appelée la routine.

Les préconditions vont donc pouvoir s'appliquer à l'état de l'objet. Par exemple, supposons une classe FileStream, une précondition pour appeler la méthode "Read" est que le FileStream soit dans l'état "Ouvert". De même pour les postconditions.

Attention lors de la définition de préconditions sur des variables membres, il ne faut pas oublier que les préconditions doivent être vérifiables par le client. Aussi, il ne faut pas imposer de préconditions sur un état non exposé de l'objet. Pour les postconditions, ce problème n'existe pas, mais on peut s'interroger sur l'intérêt d'une telle postcondition pour le client.

La notion "d'état exposé" est assez subjective. En effet, une postcondition de "open()" étant que le fichier est ouvert, cet état est exposé même si la classe ne comporte pas de méthode "is_open()". Personnellement, je préfère m'en tenir à état exposé = vérifiable par code, mais ce n'est en aucun cas faux d'être plus permissif sur cette définition, l'essentiel étant que l'appelant soit en mesure de respecter sa part du contrat.

3.1. Invariants de classe

L'objet ayant un état, on peut définir des propriétés sur cet état, qui sont toujours vérifiées. Une sorte de pré/postcondition universelle, respectée par l'ensemble des routines publiques (y compris protégées) de l'objet. Ces propriétés sont appelées **invariants de classe**.

Les invariants sont valides en entrée et en sortie de chaque routine, toutefois, il est tout à fait possible de rompre ces invariants, le temps d'un traitement. Ce n'est pas un problème du moment que, une fois sorti de la routine, l'invariant est de nouveau respecté.

Un corollaire intéressant de cela est qu'en *multithread*, si une routine doit violer un invariant, alors elle ne peut le faire qu'au sein d'une section protégée, et tous les autres appels de fonction doivent attendre que cet invariant soit à nouveau respecté pour, soit commencer, soit terminer.

Pour prendre l'exemple de la classe std::vector, un invariant de cette classe est que la taille du vecteur est toujours supérieure ou égale à zéro.

L'invariant n'est pas nécessairement une propriété visible depuis l'extérieur, en effet, c'est une garantie, le client n'a pas d'influence dessus. Toutefois, comme pour les postconditions, l'invariant doit donner une information *utile* au client.

3.1.1. Invariants et exceptions

On l'a vu, en cas d'exception, les postconditions ne sont pas respectées. La question est plus délicate à régler pour les invariants. En effet, la méthode at() sur std::vector est susceptible de nous renvoyer une exception, mais l'objet reste utilisable, et ses invariants sont respectés. À contrario, un socket dont l'invariant est "Ouvert" et qui se retrouverait soudainement fermé après un send() du fait d'une erreur réseau n'est plus utilisable, et ses invariants sont rompus.

La meilleure approche me semble donc de laisser le choix, et d'informer le client. On aura donc des exceptions qui cassent les invariants de l'objet, d'autres non. Ceci peut être indiqué au client en typant les exceptions qui cassent les invariants, en les faisant par exemple dériver d'une même classe breaking_exception.

Ainsi, si une opération casse l'invariant, l'utilisateur en est informé, et l'erreur de programmation (violation d'invariant) se produira si l'utilisateur réutilise l'objet (appelle une routine de l'objet). Il faudra toutefois prendre en compte que la violation de l'invariant est due non à un *bug* du code de l'objet, mais à l'utilisation d'un objet "cassé".

Bien que l'invariant devrait être respecté en entrée du destructeur, dans l'implémentation, nous ne le vérifierons pas. En effet, le destructeur est tout de même appelé pour des objets "cassés", et nous ne voulons certainement pas que ce soit considéré comme une erreur de programmation (même si l'objet est cassé, on sait le plus souvent dans quelle mesure et comment le libérer proprement).

Une autre approche consiste à dire que les objets doivent toujours respecter leurs invariants, et que s'ils ne sont pas en mesure de le faire, c'est que la conception est mauvaise. Cette approche a l'avantage que toute violation d'invariant est due à un *bug* dans la classe elle-même, ce qui facilite le diagnostic. Cet avantage compense la perte de flexibilité dans la définition des invariants.

3.2. Héritage du contrat

Avant d'aller plus loin, il faut définir de quel héritage on parle. La programmation par contrat s'intéresse avant tout à l'héritage polymorphique. Pour faire un programme correct et réutilisable, il est essentiel d'avoir un certain nombre de garanties. Dans le cas de l'héritage

polymorphique, la garantie que l'on veut est que si on crée une classe B qui dérive de A, alors, tout le code déjà écrit, qui utilise des A, est valide pour un B. Cette propriété est aussi connue sous le nom de *Principe de Substitution de Liskov* (LSP en anglais).

Pour se convaincre de l'utilité du LSP, on peut raisonner par l'absurde. Si le LSP n'est pas respecté, alors, des fonctions que j'ai écrites pour A, ne fonctionneront pas pour B. C'est-à-dire que je vais devoir réécrire du code spécifique pour gérer ces B. De plus, je vais devoir connaître le type dynamique pour différencier les cas où j'ai B des cas où j'ai A. J'ai donc perdu tout le bénéfice de mon héritage polymorphique.

Afin d'avoir cette garantie, il existe un outil formidable qui est le contrat. Si B respecte le même contrat que A, alors, B est substituable à A. Mais on perd beaucoup en flexibilité. Heureusement, c'est un peu plus souple :

- Les préconditions peuvent en effet être étendues. Le minimum est que la routine de B fonctionne pour l'ensemble des valeurs admises par celle de A, mais rien ne l'empêche de faire plus.
- Les postconditions peuvent être renforcées. Là aussi, le minimum est de garantir la même chose que ce que faisait A, mais il est toujours possible de garantir plus.
- Les invariants doivent être respectés, mais il est possible d'en définir de nouveaux, *à condition de s'assurer que les routines définies dans A, susceptibles d'être appelées, ne les violent pas*. En général, on définira de nouveaux invariants sur des membres n'existant pas dans A.

Les plus perspicaces d'entre vous se diront qu'il y a un problème à cette approche, à savoir que l'héritage polymorphique impose une restriction supplémentaire, "this est un B", plus restrictive que "this est un A". Nous verrons plus loin pourquoi ce n'est pas un problème.

Retrouvez la suite de l'article de Julien Blanc en ligne : Lien178

Un updater avec Qt

Comment approfondir mes connaissances dans Qt ? Comment créer un updater avec Qt ? Quelques questions auxquelles cette série espère répondre.

1. Objectifs

L'objectif de cette série sera de créer un updater, un programme qui permet de mettre à jour une application depuis une source située en général sur Internet. Évidemment, il a aussi pour objectif de réduire au maximum les données échangées, pour soulager le serveur et le client, qui doit alors moins attendre.

D'abord, nous allons commencer par un updater tout simple. Une simple fenêtre avec un bouton pour lancer la mise à jour, qui s'effectuera tout aussi simplement en remplaçant les fichiers en local par la dernière version disponible sur le serveur, sans vérifier qu'elle est bien nécessaire. Les fichiers seront téléchargés par le protocole HTTP (Lien179).

2. Contenu

Ensuite, nous améliorerons le programme : il pourra vérifier sur le serveur que la mise à jour est nécessaire en comparant le poids et les dates, ces informations seront transmises sous forme de tableau par un script sur le serveur.

Puis, le programme téléchargera un XML et le parsera, afin de vérifier la présence de mises à jour. Il vérifiera aussi un hash MD5 (Lien180) des fichiers, pour vérifier que les versions locales ne sont pas corrompues.

Postérieurement, nous ajouterons une base de données en local qui contiendra ces données, et qui servira à la comparaison. L'utilisateur aura la possibilité de vérifier l'intégrité des fichiers par rapport à la dernière mise à jour.

Subséquemment, les données transmises seront compressées par LZMA, pour économiser la bande passante.

Puis, nous intégrerons cela dans un assistant, avec votre logo, et l'icône du bouclier sous Vista lorsqu'une élévation des privilèges est requise.

Ensuite, nous autoriserons la traduction de l'application, pour que tous puissent l'avoir dans leur langue, du chinois à l'anglais, en passant bien sûr par le français et le russe.

C'est alors que nous nous intéresserons à la création de mises à jour : avant, tout se passait sur le FTP. Maintenant, vous aurez une autre application qui vous permettra de créer des mises à jour et de les mettre sur le serveur, toujours par le protocole FTP (Lien181).

Alors, nous enverrons un mail à un serveur, qui se chargera de l'envoyer à tous les inscrits à votre newsletter pour les informer de l'existence de la nouvelle mise à jour.

Mais votre application sera de plus en plus utilisée, et vos serveurs tomberont de plus en plus vite à genoux. Il vous faudra y remédier : la meilleure solution est de répartir la charge de travail sur plusieurs serveurs, et mêmes clients. Nous utiliserons les *torrents* et *Metalinks* pour ce faire.

Il deviendra alors intéressant d'avoir des statistiques sur le téléchargement de vos mises à jour : l'updater pingera votre serveur, celui-ci stockera ces statistiques, et votre créateur de mise à jour pourra récupérer ces statistiques et en faire un graphique.

Cependant, vous n'aimeriez pas que tous puissent contrôler le contenu de ce ping ? Ou de toute autre transaction ? C'est pour cela que le SSL (Lien182) peut être utilisé avec le protocole HTPP (Lien179) : le résultat est le HTTPS

(Lien183), protocole que Qt gère aussi simplement que le HTTP (Lien179).

Finalement, certaines de vos mises à jour sont réservées à ceux qui ont payé votre produit : vous n'aimeriez pas que n'importe quel petit malin puisse les utiliser sans payer ? Nous crypterons donc ces fichiers dès leur envoi depuis votre PC, et les décrypterons sur le PC du client.

Voici le programme de cette série. Qt, même s'il permet beaucoup de facilités, n'en propose pas à tous les niveaux : par exemple, il n'est pas trivial de construire un graphique. C'est pourquoi je vous présenterai deux autres librairies, prévues pour fonctionner avec Qt : Qwt et Qxt.

La première permet de gérer les graphiques, justement. La seconde, tout ce qui est envoi de mail, en ce qui concerne notre exemple.

3. Présentation

Chacun des articles présentés sera divisé en deux parties : une partie théorique et une partie pratique.

La théorie sera faite pour pouvoir être lue indépendamment de la série, afin que chacun puisse s'en servir comme référence future, sans être perturbé par la présence d'éléments se référant à l'article précédent, obligeant sa relecture. Ceci ne signifie pas que les exemples ne seront pas de la partie : ils ne se rapporteront pas directement à notre updater.

Ensuite viendra la partie pratique : elle utilisera cette théorie, qui deviendra un prérequis pour comprendre le code. Certaines parties sont ardues, et difficilement explicables hors d'un exemple, cette partie permettra de faire le point dessus, de comprendre toutes les subtilités des techniques développées.

Le code de l'updater sera disponible à tous sous licence GPL, sur le SVN de Développez. Le code sera arrêté à chaque article et fourni dans une archive.

4. Prérequis

Cette série ne commencera pas à un niveau nul : pour la suivre, vous serez censés avoir lu, compris et intégré « Débuter dans la création d'interfaces graphiques avec Qt4 » (Lien184). Vous serez ainsi familiarisé avec la conception de GUI sous Qt, avec ses principes fondateurs et avec la manière de l'utiliser.

Vous devrez évidemment disposer d'un environnement de développement qui vous convient : il s'agit d'un éditeur de texte, ainsi que d'un compilateur récent. Aussi, vous aurez installé Qt en version 4.5, Qwt en version 5.2, Qxt en version 0.5 et QCA en version 2.0. Cela signifie que vous aurez les sources et les binaires compilés, prêts à être utilisés. Vous devrez aussi avoir un plugin pour QtSql compilé : celui pour QSLite 3, livré avec Qt, sera amplement suffisant.

Qwt, Qxt et QCA se compilent comme tout autre projet Qt. Référez-vous à leur documentation pour plus d'informations.

- Installer Qt avec MinGW : Lien185
- Compiler Qt sur toutes les plateformes (et cross-compiler Qt : Lien186
- Installer Qt avec support de MySQL, de FireBird et d'OpenSSL : Lien187
- Qt : Lien188
- Qwt : Lien189
- Qxt : Lien190
- QCA : Lien191

Retrouvez l'article de Thibaut Cuvelier en ligne : Lien192

Les livres C++

The Art of Concurrency

A Thread Monkey's Guide to Writing Parallel Applications

Looking to take full advantage of multi-core processors with concurrent programming ? As one of the few resources to focus on implementing algorithms in the shared-memory model of multi-core processors, rather than just on theoretical models or distributed-memory architectures, The Art of Concurrency provides the knowledge and hands-on experience you need. You'll get detailed explanations and usable samples to help you transform algorithms from serial to parallel code, along with advice and analysis to steer you clear of mistakes.

Critique du livre par Matthieu Brucher

Free lunch is over (Lien193), c'est le moment de paralléliser. The Art of Concurrency comble le manque d'une méthodologie pour développer des applications parallèles.

Basé principalement sur les applications multithreadées, le livre couvre pthread, les Windows threads, OpenMP et Intel Threading Building Blocks. On y parle aussi des applications multi-processus, lorsqu'il y a des indications spécifiques et différentes des applications multithreads.

Le livre commence, après deux chapitres sur les actions à mener avant de paralléliser son application, ainsi que sur ce qui pourra être parallélisé et ce qui ne pourra jamais l'être. Avant les algorithmes usuels parallélisables, l'auteur utilise trois autres chapitres pour décrire sa méthodologie permettant d'atteindre notre objectif. L'assurance de la correction (i.e. vérifier si le comportement reste identique) est une tâche difficile, donc 8 règles sont proposées pour aider, et enfin les bibliothèques support (pthread, Windows thread, OpenMP et TBB) sont exposées.

La plus grande partie du livre est consacrée, comme je l'ai suggéré, à des algorithmes somme toute simples, mais qui peuvent être parallélisés : sommes et "scans", mapreduce, tri, recherche et algorithmes de graphes. Chaque fois,

plusieurs algorithmes sont tout d'abord codés en sériel, puis parallélisés avec potentiellement plusieurs bibliothèques support. Enfin, à chaque fois l'efficacité, la simplicité, la portabilité et la scalabilité sont étudiées. Cela sert à prendre du recul sur l'opération effectuée.

Le dernier chapitre est une revue rapide des outils additionnels qu'on peut utiliser (*i.e.*, non obligatoires). Il s'agit principalement d'outils Intel, surtout car Intel propose de nombreux outils parmi les meilleurs.

Même si l'auteur travaille pour Intel, les outils de celui-ci ne sont pas plus mis en avant que les autres. Le ton global du livre est adapté, pas trop sérieux, pas trop copain-copain, juste ce qu'il faut.

En résumé, un très bon livre pour qui veut paralléliser son application.

Retrouvez cette critique de livre sur la page livres C++ : Lien194

Les dernières news

Sortie de Qt 4.6 et de Qt Creator 1.3

Bonjour,

Il est désormais beaucoup plus facile de créer des applications de qualité pour Symbian, Maemo, et beaucoup d'autres plateformes !

En effet, Qt 4.6 vient de sortir, et supporte ces deux nouvelles plateformes. Parmi les nouveautés, on compte aussi de nouvelles possibilités graphiques, le support du multi-touch et des gestes. Qt 4.6 rend le développement d'applications avancées plus facile.

Citation de **Sebastian Nyström, Vice President, Application Services and Frameworks chez Nokia**

```
Qt 4.6 marks an exciting time for developers,
regardless of their arget form factor or platform

Developers can easily create visually appealing
and web-connected applications for desktops or
devices, including targeting the hundreds of
millions of Symbian and Maemo-based devices

The community will enjoy using Qt's intuitive
programming interface to quickly create powerful,
appealing applications
```

De nouvelles plateformes

Qt 4.6 supporte Symbian, ainsi que Windows 7, Mac OS X 10.6 (Snow Leopard) et le futur Maemo 6. Grâce à la communauté, QNX et VxWorks sont aussi supportés

Grâce au support de Symbian et de Maemo, les développeurs peuvent viser de nombreuses nouvelles plateformes avec une seule et unique base de code.

Aussi, les API du projet Mobility sont désormais fournies : Qt peut donc localiser, gérer des contacts, envoyer des messages, et bien d'autres choses encore !

Le renouveau du visuel

Qt 4.6 continue sur la lancée de Qt 4.5, et ajoute le framework Animation, qui inclut la machine à états. Ceci permet d'avoir de nouveaux effets graphiques avancés : opacité, ombres, filtres...

Plus de doigts

Une des fonctionnalités les plus attendues de Qt 4.6 est la gestion du multi-touch. Avec le support des gestes, Qt dispose de deux nouvelles méthodes d'entrée. Qt peut donc servir à créer des applications dynamiques et tactiles pour les utilisateurs, qui peuvent ainsi interagir plus facilement avec l'application.

Plus de puissances

Les développeurs de Qt ont remarqué que, généralement, quand on utilise une IHM aussi poussée, les performances en pâtissent : ils ont décidé d'aller à l'encontre de ce principe, en optimisant encore plus leur framework. En conséquence : l'effet *waouw !*

Ceci a été permis par les algorithmes, repensés et fortement optimisés, à la base de Qt Graphics View, mais aussi par un nouveau module de dessin OpenGL, le support des fonctionnalités d'OpenVG pour les périphériques, et le tout nouveau support de DirectFB.

De nouveaux outils

Comme précédemment, cette nouvelle version de Qt est assortie d'une nouvelle version de Qt Creator.

Grande nouveauté pour les francophones : cette version 1.3 de Qt Creator est désormais disponible en français !

Qu'y a-t-il d'autre comme nouveautés ?

Le support préliminaire de Symbian, un *refactoring* plus aisé, la compilation sur plusieurs cœurs avec Microsoft Visual C++, et l'inclusion de la dernière version stable de MinGW : la 4.4.

Source : communiqué de presse.

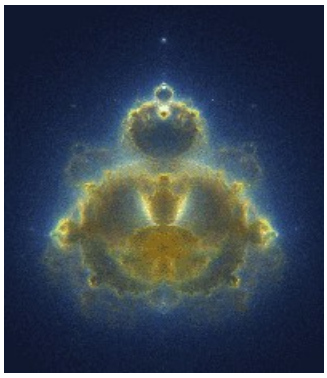
Commentez cette news en ligne : Lien195

Le premier défi Qt !

Bonjour,

Depuis longtemps, vous languissez à l'attente de leurs lancement. C'est maintenant chose faite. Mais ne traînons pas plus en vains bavardages, passons aux choses sérieuses.

Quel est le sujet de ce premier défi ? Une petite image pour vous mettre sur la piste ?



Version plus grande : Lien196
Version haute qualité : Lien197

Reconnaissez-vous ce chef-d'œuvre ? Il s'agit du Buddhabrot. Non, il ne s'agira pas de devoir l'implémenter : tel n'est pas le but. Vous devrez simplement l'afficher (ceci n'est que la base du défi : libre à vous d'ensuite continuer sur cette piste !), nous vous en fournissons une implémentation simpliste : voici la deuxième partie du défi, utiliser toute la puissance de Qt (multithreading, par exemple) pour l'optimiser. Seuls Qt et le C++ sont autorisés.

Le défi s'ouvre ce lundi 30 novembre 2009 et s'achèvera le dimanche 28 février 2010 à minuit.

Un nouveau forum est à votre disposition pour présenter votre solution au défi : les défis Qt (Lien198).

La page d'accueil des défis Qt : Lien199
Les règles : Lien200
Dépôt des projets : Lien201
Le forum du défi : Lien202

Le défi du Buddhabrot : Lien203

N'hésitez pas à poster ci-après toute question relative aux défis, à leurs règles, etc. ! Pour des questions plus techniques, non relatives directement au défi, utilisez le forum Qt (Lien204) ou le forum multithreading avec Qt (Lien205), selon la nature de votre question.

Envisagez-vous de participer à ce défi ? Connaîtriez-vous des gens qui seraient intéressés par ce défi ?

Commentez cette news en ligne : Lien206

Bullet physics - tutoriel

Ce tutoriel est consacré à ceux qui veulent découvrir la magnifique librairie Bullet physics. Vous apprendrez les types d'objets rigides ainsi que l'intégration dans votre application.

La version de Bullet physics utilisée dans ce tutoriel est la 2.75.

1. Introduction

Bullet est une librairie de détection de collisions gérant des objets dynamiques comme des boites, sphères, capsules et cetera pour rendre les scènes dynamiquement plus réalistes. Elle est utilisable dans des applications temps réel, par exemple des jeux.

La librairie est open-source (Lien207), gratuite pour utilisation commerciale, sous licence zlib (Lien208).

Il y a un forum pour la communauté autour de cette librairie traitant de détections de collisions et physiques à l'adresse suivante : Lien209

2. Téléchargement et compilation

2.1. Pré requis

Vous devez avoir de l'expérience avec C++ et OpenGL ainsi qu'une librairie pour créer un contexte OpenGL comme SFML (Lien210), SDL (Lien211) ou encore GLUT (Lien212).

2.2. Téléchargement

Vous pouvez télécharger le code source de bullet physics depuis Google Code : Lien213

2.3. Compilation

Bullet est livré avec des fichiers de projets auto-générés pour Microsoft Visual Studio 6, 7, 7.1 et 8. Le fichier de projet principal se trouve dans **Bullet/msvc/8/wksbullet.sln** (remplacez **8** avec votre version de M. Visual Studio).

Sur d'autres plateformes, comme Linux ou Mac OS-X, Bullet peut être compilé utilisant **make**, **cmake** (Lien214), ou **jam** (Lien215).
cmake peut auto-générer des fichiers de configuration pour Xcode, KDevelop, MSVC et d'autres systèmes de compilation.

Tapez juste **cmake** dans une console, étant dans le dossier principal de Bullet.

Si vous n'utilisez pas Microsoft Visual Studio ou cmake, vous pouvez lancer **./autogen.sh ./configure** pour créer des fichiers Makefile et Jam ensuite tapez **make** ou **jam**.

Jam est un système de construction qui peut compiler la librairie, les démos, mais aussi auto-générer des fichiers de projet Microsoft Visual Studio. Si jam n'est pas installé, vous pouvez le télécharger depuis ce lien : Lien216

2.4. Tester les démos

Essayez d'exécuter et expérimentez avec l'exécutable **BasicDemo** comme point de départ. Bullet peut être utilisé de plusieurs façons : en tant que simulation de corps rigides, librairie de détection de collisions ou bas niveau comme « GJK calculabilité du plus proche point ».

3. Types de données basiques et la librairie mathématique

Les types de données basiques, gestion de mémoire et conteneurs sont localisés dans **Bullet/src/LinearMath**.

btScalar : un nombre à virgule flottante.
De façon à pouvoir compiler en simple précision en virgule flottante ou double précision, Bullet utilise le type de donnée **btScalar**. Par défaut, **btScalar** est un typedef de float. Il peut être double en définissant **BT_USE_DOUBLE_PRECISION** en haut du fichier **Bullet/src/LinearMath/btScalar.h**.

btVector3 : les positions 3D et les vecteurs peuvent être représentés en utilisant **btVector3** qui contient 3 composants scalaires x, y, z.
Plusieurs opérations peuvent être effectuées sur un **btVector3**, par exemple, addition, soustraction et longueur d'un vecteur.

btQuaternion et **btMatrix3x3**
Les orientations 3D et rotations peuvent être représentées en utilisant **btQuaternion** ou **btMatrix3x3**.

btTransform est une combinaison d'une position et d'une orientation.
Il peut être utilisé pour transformer des points et vecteurs d'un espace de coordonnées dans un autre. Aucune mise à l'échelle ou cisaillement n'est autorisé.

Bullet utilise un système de coordonnées droitier :

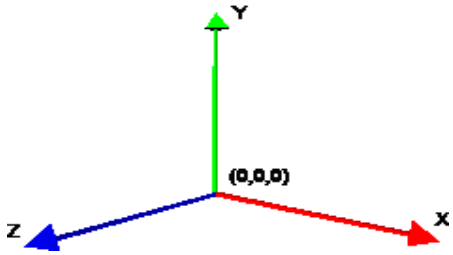


Figure 1 : Système de coordonnées droitier

btTransformUtil et **btAabbUtil** fournissent des fonctions de transformations et des AABB (Aligned Axis Bounding Box).
Aligned Axis Bounding Box ou AABB signifient : boite englobante qui reste toujours orientée sur les axes X, Y et Z.
Pour augmenter les performances un calcul est avant tout fait sur les boites englobantes des objets pour savoir lesquelles se chevauchent. Pour les AABB qui ne se chevauchent pas, leurs objets ne nécessitent pas de calculs de collisions par exemple.

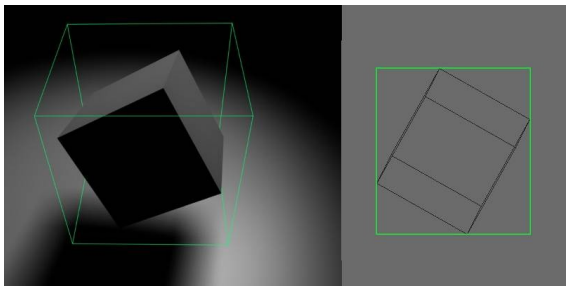


Figure 2 : Représentation d'une boite englobante ou AABB ainsi que sa représentation en 2D à droite

4. Les types d'objets rigides

4.1. Les primitives convexes

btBoxShape : boite définie par la moitié de la longueur des côtés.
*Très utile pour représenter des objets cubiques, voir **Figure 3**.*

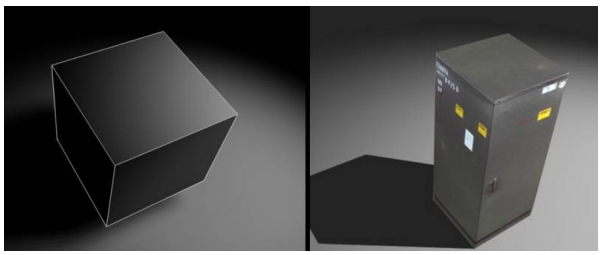


Figure 3 : Objet dynamique en forme de boite à gauche et un objet qu'on peut représenter par une boite dans un jeu, à droite

btSphereShape : sphère définie par son rayon.
*Peut représenter les objets sphériques comme les ballons de foot, billes de billard et cetera, voir **Figure 4**.*

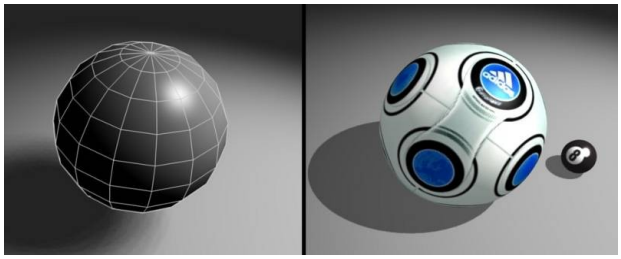


Figure 4 : Objet dynamique sphérique à gauche et des objets qu'on peut représenter par une sphère, à droite

btCapsuleShape: capsule autour de l'axe Y.
btCapsuleShapeX/Z peuvent être utilisés pour spécifier autour de l'axe **X** ou **Z**.

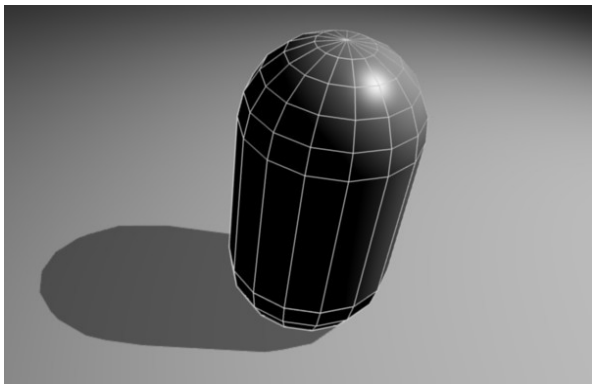


Figure 5 : Objet dynamique en forme de capsule

btCylinderShape : cylindre autour de l'axe Y.
btCylinderShapeX/Z peuvent être utilisés pour spécifier autour de l'axe **X** ou **Z**.
*Forme cylindrique, utile pour représenter des bidons et cetera, voir **Figure 6**.*

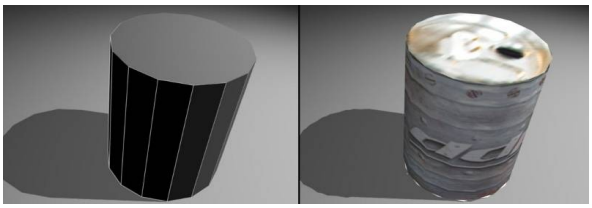


Figure 6 : Objet dynamique en forme de cylindre à gauche, avec on peut représenter un bidon, à droite

btConeShape : cône autour de l'axe Y.
btConeShapeX/Z peuvent être utilisés pour spécifier autour de l'axe **X** ou **Z**.

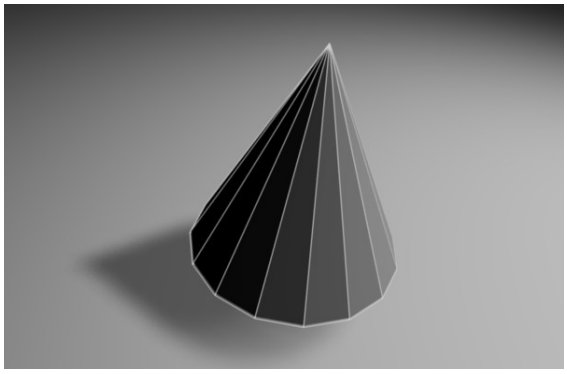


Figure 7 : Objet dynamique en forme de cône

btMultiSphereShape : pour créer un convex hull à partir de plusieurs sphères qui peuvent par exemple être utilisées

pour créer une capsule.
*Objet dynamique composé de plusieurs sphères pour représenter certaines formes. Les sphères peuvent être animées également. Voir **Figure 8**.*

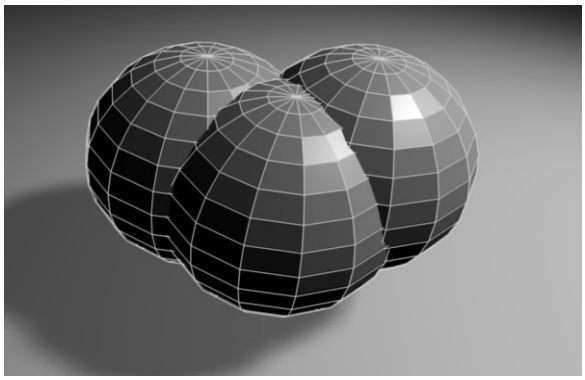


Figure 8 : Objet dynamique composé de plusieurs sphères. Ceci est un seul objet.

4.2. Formes composées (Compound shapes)

BtCompoundShape
De multiples formes convexes peuvent être combinées dans un compound shape. Ceci devient une forme concave composée de multiples sous-formes convexes, appelées des formes enfant. Chaque forme enfant a sa propre transformation locale, relative à **btCompoundShape**.
*Objet composé de plusieurs formes pour représenter des modèles 3D sans passer par des tableaux de sommets. Voir **Figure 9**.*

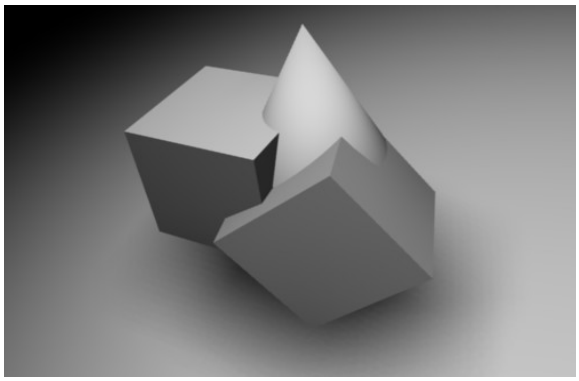


Figure 9 : Exemple de compound shape ou forme composée. Ceci est un seul objet.

4.3. Formes convexes à partir de modèles 3D (Convex Hull Shapes)

BtConvexHullShape
Bullet supporte plusieurs façons de représenter des formes convexes à partir de meshes triangulés. La façon la plus simple et facile est de créer un **btConvexHullShape** et de passer un tableau de sommets.

4.4. Formes concaves statiques à partir de modèles 3D (Concave Triangle Meshes)

Pour des objets statiques de l'environnement, un moyen très efficace pour représenter les maillages de triangles est d'utiliser un **btBvhTriangleMeshShape**.

4.5. Décomposition convexe (Convex decomposition)

Idéalement, des modèles 3D concaves devraient être utilisés seulement pour représenter du décor statique. Autrement, son enveloppe convexe doit être utilisée en faisant passer le maillage à **btConvexHullShape**. Si une forme convexe unique n'est pas assez détaillée, des parties convexes multiples peuvent être combinées en un objet composite appelé **btCompoundShape**.

La décomposition convexe peut être utilisée pour décomposer le maillage concave en plusieurs parties convexes. Regarder **Demos/ConvexDecompositionDemo** pour une façon automatique de faire une décomposition convexe.

4.6. Hauteur du champ (Height field)

Bullet fournit un soutien pour le cas particulier d'un terrain 2D concave par l'intermédiaire du **btHeightfieldTerrainShape**.
Regardez **Demos/TerrainDemo** pour son utilisation. Voir **Figure 10**.

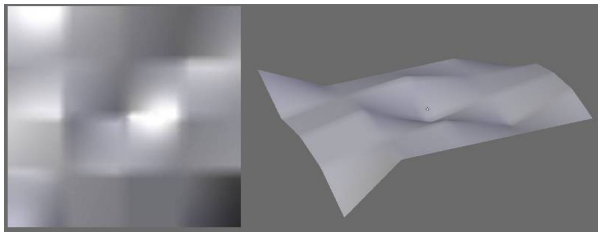


Figure 10 : Exemple d'un hauteur de champ (texture 2D) à gauche et sa transformation en 3D à droite

4.7. Plan infini

btStaticPlaneShape : comme son nom l'indique btStaticPlaneShape représente un plan infini ou de taille définie. Idéal pour faire un sol pour ses tests et expérimentations rapides. Cette forme se doit d'être statique.

5. Intégration dans notre application

5.1. Le header Bullet

On commence par inclure Bullet dans notre application :

Header Bullet
```
#include "btBulletDynamicsCommon.h"
```

5.2. Déclarations

Ensuite on déclare le nom du monde physique :

Monde physique
```
btDiscreteDynamicsWorld *myWorld;
```

La classe **btBroadphaseInterface** fournit une interface pour détecter les objets où leurs AABB se chevauchent.

Broadphase
```
btBroadphaseInterface *myBroadphase;
```

btCollisionDispatcher supporte des algorithmes qui peuvent gérer des paires de collisions ConvexConvex et ConvexConcave, le temps de l'impact, le point le plus proche et la pénétration de profondeur.

Collision Dispatcher
```
btCollisionDispatcher *myDispatcher;
```

btCollisionConfiguration permet de configurer les allocataires de mémoire.

Default Collision Configuraiton
```
btDefaultCollisionConfiguration
*myCollisionConfiguration;
```

btSequentialImpulseConstraintSolver est une implémentation SIMD rapide de la méthode Projected Gauss Seidel (iterative LCP).

Sequential Impulse Constraint Solver
```
btSequentialImpulseConstraintSolver
*mySequentialImpulseConstraintSolver;
```

Position, orientation.

btTransform
```
btTransform myTransform;
```

btDefaultMotionState fournit une implémentation pour synchroniser les transformations.

Motionstate
```
btDefaultMotionState *myMotionState,
                     *myMotionState_Sol;
```

Une matrice OpenGL, pour récupérer la position, la rotation d'un objet.

Matrice OpenGL
```
btScalar matrix[16];
```

Le corps d'une boite et de notre sol.
btRigidBody est la classe principale des objets rigides

Rigid body
```
btRigidBody *body,
            *body_sol;
```

5.3. Une boite OpenGL

Une fonction pour afficher une boite à l'aide d'OpenGL

Boite OpenGL
```
void myBox(LDEfloat x, LDEfloat y, LDEfloat z)
{
    glPushMatrix();
    glScalef(x,y,z);
    glBegin (GL_QUADS);
    //face 1
    glNormal3i(-1, 1,-1);
    glVertex3i(-1, 1,-1); glVertex3i( 1, 1,-1);
    glVertex3i( 1,-1,-1); glVertex3i(-1,-1,-1);
    //face 2
    glNormal3i(-1,-1,-1);
    glVertex3i(-1,-1,-1); glVertex3i( 1,-1,-1);
    glVertex3i( 1,-1, 1); glVertex3i(-1,-1, 1);
    // face 3
    glNormal3i( 1,-1, 1);
    glVertex3i( 1,-1, 1); glVertex3i( 1,-1,-1);
    glVertex3i( 1, 1,-1); glVertex3i( 1, 1, 1);
    //face 4
    glNormal3i( 1, 1,-1);
    glVertex3i( 1, 1,-1); glVertex3i(-1, 1,-1);
    glVertex3i(-1, 1, 1); glVertex3i( 1, 1, 1);
    //face 5
    glNormal3i(-1, 1, 1);
    glVertex3i(-1, 1, 1); glVertex3i(-1, 1,-1);
    glVertex3i(-1,-1,-1); glVertex3i(-1,-1, 1);
    //face 6
    glNormal3i( 1,-1, 1);
    glVertex3i( 1,-1, 1); glVertex3i( 1, 1, 1);
    glVertex3i(-1, 1, 1); glVertex3i(-1,-1, 1);
    glEnd();
    glPopMatrix();
}
```

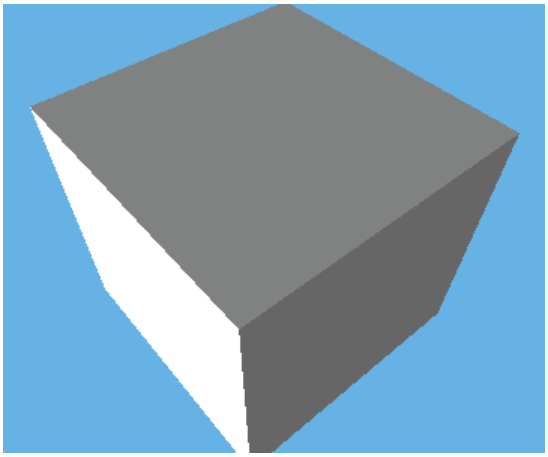


Figure 11 : Le code donne cette magnifique boite

5.4. Initialisation

Contient la configuration par défaut pour la mémoire, la collision

Configuration
```
myCollisionConfiguration = new
btDefaultCollisionConfiguration();
```

Le collision dispatcher par défaut. Pour du processus parallèle utilisez un dispatcher différent. Regardez **Extras/BulletMultiThreaded**.

Dispatcher
```
myDispatcher = new
btCollisionDispatcher(myCollisionConfiguration);
```

Initialisation du broadphase (détecteur des objets où leurs AABB se chevauchent)

Breadphase
```
myBroadphase = new btDbvtBroadphase();
```

Constraint solver par défaut. Pour du processus parallèle utilisez un constrint solver différent. Regardez **Extras/BulletMultiThreaded**.

Sequential Impulse Constraint Solver
```
mySequentialImpulseConstraintSolver = new
btSequentialImpulseConstraintSolver;
```

On initialise le monde physique Bullet.

Initialisation
```
myWorld = new
btDiscreteDynamicsWorld(myDispatcher,myBroadphase
,mySequentialImpulseConstraintSolver,myCollisionC
onfiguration);
```

On définit la gravité, de façon à ce que les objets tombent vers le bas (**-Y**).

Gravité
```
myWorld->setGravity( btVector3(0,-10,0) );
```

5.5. Création de la boite dynamique

On déclare une forme et on l'initialise en tant que boite de la taille 1,1,1 (x, y, z)

Initialisation en forme de boite (x, y, z)
```
btCollisionShape* shape = new
btBoxShape( btVector3(1,1,1) );
```

On initialise notre btTransform et on lui affecte une position (la position de la boite)

Position de la boite
```
myTransform.setIdentity();
myTransform.setOrigin( btVector3(10,5,0) );
```

L'inertie de la boite

L'inertie
```
btVector3 localInertia(0,0,0);
```

Le poids de la boite

Le poids
```
btScalar mass = 0.5f;
```

On calcule l'inertie suivant le poids.
Note : inutile de calculer l'inertie si la boite n'a pas de poids puisqu'elle devient statique.

Calcul de l'inertie suivant le poids
```
if ( mass )
    shape->calculateLocalInertia( mass,
localInertia );
```

Il est conseillé d'utiliser motionState car il fournit des capacités d'interpolation et synchronise seulement les objets "actifs".

Motionstate
```
myMotionState = new
btDefaultMotionState(myTransform);
```

On regroupe les informations de la boite à partir de la masse, l'inertie et cetera

Informations sur ce corps
```
btRigidBody::btRigidBodyConstructionInfo
myBoxRigidBodyConstructionInfo( mass,
myMotionState, shape, localInertia );
```

On construit le corps de la boite à partir de l'information regroupée.

Initialisation du corps
```
body = new
btRigidBody(myBoxRigidBodyConstructionInfo);
```

Et enfin on ajoute notre boite dans le monde Bullet

Ajout de la boite dans le monde
```
myWorld->addRigidBody(body);
```

5.6. Création du sol

La création du sol consiste à créer une boite comme indiqué ci-dessus mais en la mettant statique.

sol
```
// Forme en tant que boite
btCollisionShape* shape_sol = new
btBoxShape( btVector3(10,1,10) );

myTransform.setIdentity();

// Position du sol
myTransform.setOrigin( btVector3(0,0,0) );
btVector3 localInertiaSol(0,0,0);

mass = 0; // Le fait que le poids de cet objet
soit zéro le rends statique

myMotionState_Sol = new
btDefaultMotionState(myTransform);

btRigidBody::btRigidBodyConstructionInfo
sol_info( mass, myMotionState_Sol, shape_sol,
localInertiaSol );

body_sol = new btRigidBody(sol_info);

// On ajoute le sol dans le monde Bullet
myWorld->addRigidBody(body_sol);
```

6. La boucle d'affichage de l'application/jeu

6.1. Mise à jour des transformations

Dans la boucle d'affichage on devra appeler la fonction **stepSimulation** pour calculer et mettre à jour les transformations des objets dynamiques.
0.001 doit être le frametime de votre application/jeu. C'est-à-dire, le temps que prend une frame pour tout exécuter/afficher.

sol
```
if ( myWorld )
        myWorld->stepSimulation( 0.001 );
```

6.2. Récupération des transformations et affichage des objets

On recupère la matrice OpenGL de la boite. C'est grâce à cette matrice qu'on appliquera la transformation effectuée par la librairie Bullet à l'objet qu'on affichera avec OpenGL.

sol
```
myMotionState-
>m_graphicsWorldTrans.getOpenGLMatrix( matrix );
```

On affiche la boite à l'aide d'OpenGL

sol
```
/* grâce à glPushMatrix (ouverture) et
glPopMatrix (fermeture) on retiens les
transformations seulement appliquées à cette
boite */
glPushMatrix();
/* On applique les transformations à la boite */
glMultMatrixf( matrix );
/* Ensuite, on affiche la boite */
myBox(1,1,1);
glPopMatrix();
```

On affiche le sol (toujours une boite)

sol
```
 /* On affiche le sol */
myBox(10,1,10);
```

7. Conclusion

7.1. Vidéo de ce qu'on peut arriver à faire avec un peu de motivation

Vidéo sur Youtube : Lien217

Il est facile d'ajouter d'autres types d'objets dynamiques en initialisant **btCollisionShape** avec **btSphereShape** par exemple. Vous avez les autres formes indiquées plus haut.

7.2. Conclusion

Bullet physics est multi-plateformes, largement utilisé, gratuit pour utilisation commerciale, open-source et a une belle communauté, un excellent choix.

8. Code source

Voici le code source complet. La gestion du fenêtrage est faite à l'aide de la SFML (Lien210) : Lien218

Retrouvez l'article de Dorin Grigore en ligne : Lien219

Les dernières news

L'Unreal Development Kit disponible gratuitement

Aujourd'hui, Epic vient de livrer une version gratuite de son kit de développement de jeu basé sur l'Unreal Engine 3 : l'Unreal Development Kit (UDK).

Bien que ce kit contienne toutes les possibilités offertes par l'UE3, il reste néanmoins limité. En effet, la licence interdit son utilisation pour une production commerciale ou générant des revenus indirects. Il peut par contre être utilisé pour un produit entièrement gratuit, ou à des fins d'éducation.

Cerise sur le gâteau, l'UDK est fourni avec tous les outils nécessaires à la création d'un jeu, ainsi qu'avec des démos, des ressources et toute la documentation.

A n'en pas douter, ce kit de développement va faire le régal des petites équipes de développement qui vont pouvoir développer leurs jeux sur un outils professionnel de qualité.

Unreal Developpement Kit (Lien220)

La page de Téléchargement direct (Lien221)

La documentation : Lien222

La licence d'utilisation du kit : Lien223

Ne risque-t-on pas de voir ce mastodonte prendre la place des petits moteurs sur le créneaux des jeux indépendants gratuits ?
Livrer une version gratuite d'un tel moteur : une technique pour lier les étudiants et les petites entreprises à leur technologie ?
Face au Cry Engine 3 et au futur ID Tech 5, l'Unreal Engine fait-il encore le poids pour les jeux commerciaux ?

Commentez cette news en ligne : Lien224

Un programme révolutionnaire permet de transformer une webcam bon marché en scanner 3D

Son nom est long, ennuyant et compliqué, et pourtant, il est tout le contraire : Probabilistic Feature-based On-line Rapid Model Acquisition (ou ProFORMA pour les intimes) est un programme révolutionnaire écrit par une équipe d'étudiants dirigée par Qui Pan (étudiant au Département d'Ingénierie de l'Université de Cambridge en Angleterre).

Leur software permet de transformer n'importe quelle webcam bon marché en un scanner 3D.

Normallement, scanner en 3D nécéssite beaucoup de temps, de même que l'emploi de matériel spécifique. ProFORMA annule tout cela : il suffit de faire faire une rotation à un objet devant l'objectif de la webcam qui le scanne en temps réel, créant ainsi un modèle 3D en aussi peu de temps qu'il en faut pour retourner l'objet !

Encore plus immpressionnant : le scan terminé, la caméra continue de suivre l'objet dans l'espace et rend immédiatement ses mouvements sur l'écran.

Le système fonctionne grâce à des formules mathématiques très poussées, permettant même d'ignorer les occultations éventuelles du modèle par la main le tenant. Puis, un processus appelé tetrahedralisation de Delaunay permet de transformer la surface 2D en un modèle en 3D.

Toutes les applications en sont ensuite possibles : créer un clône virtuel de votre jouet préféré, ou bien même votre propre avatar. Il seraient ensuite interessant de pouvoir inclure ces modèles dans vos jeux vidéos préférés, mais je m'emballe peut-être un peu vite, là...

Lorsque le projet sera terminé, le programme sera disponible sous Linux puis sous Windows. Inscrivez-vous ici pour en savoir plus : Lien225

Que vous inspire l'arrivée d'une telle technologie pour le grand public ?

Source : Démonstration vidéo des capacités de ProFORMA : Lien226

Commentez cette news en ligne : Lien227

Liens

Lien1 : http://www.google.com/a/
Lien2 : http://java.sun.com/products/servlet/
Lien3 : http://java.sun.com/products/jsp/
Lien4 : http://java.sun.com/javase/technologies/index.jsp
Lien5 : http://java.sun.com/jdo/index.jsp
Lien6 : http://java.sun.com/developer/technicalArticles/J2EE/jpa/
Lien7 : http://java.sun.com/products/javamail/
Lien8 : http://brocoli.developpez.com/
Lien9 : http://www.djangoproject.com/
Lien10 : http://docs.python.org/lib/lib.html
Lien11 : http://code.google.com/appengine/docs/python/runtime.html
Lien12 : http://code.google.com/appengine/docs/python/users/
Lien13 : http://code.google.com/intl/fr/appengine/docs/python/config/cron.html
Lien14 : http://code.google.com/intl/fr/appengine/docs/java/config/cron.html
Lien15 : http://code.google.com/appengine/downloads.html
Lien16 : http://code.google.com/appengine/terms.html
Lien17 : http://code.google.com/intl/fr/appengine/docs/quotas.html
Lien18 : http://www.youtube.com/watch?v=3Ztr-HhWX1c
Lien19 : http://code.google.com/appengine/articles/cf1-text.html
Lien20 : http://www.youtube.com/watch?v=bfgO-LXGpTM
Lien21 : http://code.google.com/appengine/downloads.html
Lien22 : http://appengine.google.com/
Lien23 : http://brocoli.developpez.com/
Lien24 : http://appgallery.appspot.com/
Lien25 : http://brocoli.developpez.com/articles/presentation/google-app-engine/
Lien26 : http://blogs.sun.com/dochez
Lien27 : http://blogs.sun.com/alexismp/tags/alacarte
Lien28 : http://www.oracle.com/us/sun/038563.pdf
Lien29 : http://www.oracle.com/us/sun/038563.pdf
Lien30 : http://x-plode.developpez.com/articles/interviewAlexisMP/
Lien31 : http://java.developpez.com/livres/
Lien32 : http://www.w3.org/TR/wsdl
Lien33 : http://fr.wikipedia.org/wiki/Interface
Lien34 : http://fr.wikipedia.org/wiki/Service_Web
Lien35 : http://fr.wikipedia.org/wiki/Service_Oriented_Architecture
Lien36 : http://fr.wikipedia.org/wiki/Extensible_Markup_Language
Lien37 : http://fr.wikipedia.org/wiki/Protocole_de_communication
Lien38 : http://www.wampserver.com/
Lien39 : http://www.altova.com/xmlspy.html
Lien40 : http://blogs.dotnet-france.com/julienc/post/Introduction-a-Windows-Communication-Foundation.aspx
Lien41 : http://www.urdalen.no/wsdl2php/
Lien42 : http://code.google.com/p/wsdl2php-interpreter/
Lien43 : http://laedit.developpez.com/PHP/CSharp/Interoperabilite/
Lien44 : http://blog.rafaelbandeira3.com/2008/12/05/handling-multiple-enviroments-on-cakephp/
Lien45 : http://bakery.cakephp.org/articles/view/easy-peasy-database-config
Lien46 : http://www.developpez.net/forums/d834850/php/bibliotheques-frameworks/cakephp/configurer-plusieurs-environnements-application-cakephp/
Lien47 : http://framework.zend.com/wiki/display/ZFDEV2/Zend+Framework+2.0+Roadmap
Lien48 : http://www.developpez.net/forums/d836595/php/outils/zend/zend-framework/zendframework-2-0-roadmap-va-t-se-passer/
Lien49 : http://code.google.com/chrome/extensions/getstarted.html
Lien50 : http://code.google.com/chrome/extensions/api_index.html
Lien51 : http://sylvain-dangin.developpez.com/tutoriels/general/introduction-programmation-extension-google-chrome/
Lien52 : http://www.developpez.net/forums/d810817/club-professionnels-informatique/actualites/google-remercie-microsoft-contribution-html5/
Lien53 : http://www.youtube.com/html5
Lien54 : http://www.developpez.net/forums/d810817/club-professionnels-informatique/actualites/html-5-google-remercie-microsoft-contribution-arrete-gears/#post4829759
Lien55 : http://www.developpez.net/forums/d822044/club-professionnels-informatique/actualites/google-docs-senrichit-fonctions-collaboratives-partage-dossier/
Lien56 : http://www.developpez.net/forums/d776805/club-professionnels-informatique/actualites/microsoft-office-2010-activez-fonctionnalites-cloud-beta/
Lien57 : http://9elements.com/io/projects/html5/canvas/
Lien58 : http://www.developpez.net/forums/d818041/club-professionnels-informatique/actualites/leurope-lance-egnos-nouveau-systeme-gps-gratuit-plus-precis/#post4694428
Lien59 : http://www.youtube.com/watch?v=W4FbF8GKChk
Lien60 : http://html5demos.com/
Lien61 : http://www.developpez.net/forums/d845233/club-professionnels-informatique/actualites/nouveaute-plus-importante-apportee-html5/
Lien62 : http://cssglobe.com/post/3714/css-sprites-rounded-corners
Lien63 : http://cssglobe.developpez.com/
Lien64 : http://christophe-f.developpez.com/traductions/css/coins-arrondis/fichiers/rounded-corners.html
Lien65 : ftp://ftp-developpez.com/christophe-f/traductions/css/coins-arrondis/fichiers/demo-rounded-corners.zip
Lien66 : http://cssglobe.developpez.com/tutoriels/css/ameliorer-temps-chargement-images/
Lien67 : http://christophe-f.developpez.com/traductions/css/coins-arrondis/fichiers/exemple1.html
Lien68 : http://christophe-f.developpez.com/traductions/css/coins-arrondis/fichiers/exemple2.html
Lien69 : http://christophe-f.developpez.com/traductions/css/coins-arrondis/fichiers/exemple3.html

Lien70 : http://christophe-f.developpez.com/traductions/css/coins-arrondis/fichiers/exemple4.html
Lien71 : http://christophe-f.developpez.com/traductions/css/coins-arrondis/fichiers/exemple5.html
Lien72 : http://christophe-f.developpez.com/traductions/css/coins-arrondis/
Lien73 : http://www.w3avenue.com/2009/05/25/list-of-really-useful-javascript-libraries/
Lien74 : http://coderepos.org/share/wiki/JSTweener
Lien75 : http://code.google.com/p/tweener/
Lien76 : http://fx.inetcat.com/
Lien77 : http://developers.facebook.com/animation/
Lien78 : http://ryanmorr.com/archives/fx-lightweight-and-standalone
Lien79 : http://berniecode.com/writing/animator.html
Lien80 : http://jsanim.com/
Lien81 : http://www.schillmania.com/projects/soundmanager2/
Lien82 : http://flowplayer.org/documentation/api/index.html
Lien83 : http://flowplayer.org/
Lien84 : http://code.google.com/p/cookies/
Lien85 : http://pablotron.org/software/easy_cookie
Lien86 : http://pajhome.org.uk/crypt/md5/index.html
Lien87 : http://taffydb.com/
Lien88 : http://www.activerecordjs.org/record.html
Lien89 : http://www.datejs.com/
Lien90 : http://getfirebug.com/lite.html
Lien91 : http://www.gscottolson.com/blackbirdjs/
Lien92 : http://www.nitobibug.com/
Lien93 : http://www.netzgesta.de/dev/text/
Lien94 : http://typeface.neocracy.org/
Lien95 : http://cufon.shoqolate.com/generate/
Lien96 : http://code.google.com/p/hyphenator/
Lien97 : http://wiki.novemberborn.net/sifr3/
Lien98 : http://facelift.mawhorter.net/
Lien99 : http://fontjazz.com/
Lien100 : http://www.livevalidation.com/
Lien101 : http://www.formassembly.com/wForms/
Lien102 : http://www.drlongghost.com/validanguage.php
Lien103 : http://yav.sourceforge.net/en/index.html
Lien104 : http://www.pengoworks.com/index.cfm?action=get:qforms
Lien105 : http://code.google.com/p/swfobject/
Lien106 : http://www.effectgenerator.com/AS3Wrapper/
Lien107 : http://www.aflax.org/
Lien108 : http://www.liquidx.net/plotkit/
Lien109 : http://www.jscharts.com/
Lien110 : http://code.google.com/p/flot/
Lien111 : http://pckult.developpez.com/tutoriels/javascript/frameworks/jquery/visualisation-donnees-flot/
Lien112 : http://www.lutanho.net/diagram/
Lien113 : http://www.kryogenix.org/code/browser/sorttable/
Lien114 : http://www.danvk.org/wp/dragtable/
Lien115 : http://www.sprymedia.co.uk/article/KeyTable
Lien116 : http://ejohn.org/blog/processingjs/
Lien117 : http://raphaeljs.com/
Lien118 : http://www.pixastic.com/lib/
Lien119 : http://www.walterzorn.com/jsgraphics/jsgraphics_e.htm
Lien120 : http://cow.neondragon.net/stuff/reflection/
Lien121 : http://www.netzgesta.de/lab/
Lien122 : http://www.netzgesta.de/bevel/
Lien123 : http://www.netzgesta.de/corner/
Lien124 : http://www.netzgesta.de/curl/
Lien125 : http://www.netzgesta.de/edge/
Lien126 : http://www.netzgesta.de/filmed/
Lien127 : http://www.netzgesta.de/glossy/
Lien128 : http://www.netzgesta.de/instant/
Lien129 : http://www.netzgesta.de/reflex/
Lien130 : http://www.netzgesta.de/shiftzoom/
Lien131 : http://www.netzgesta.de/snapfit/
Lien132 : http://www.netzgesta.de/slided/
Lien133 : http://www.netzgesta.de/sphere/
Lien134 : http://www.walterzorn.com/dragdrop/dragdrop_e.htm
Lien135 : http://code.google.com/p/explorercanvas/
Lien136 : http://www.c3dl.org/
Lien137 : http://jsdraw2d.jsfiction.com/
Lien138 : http://www.openjs.com/scripts/events/keyboard_shortcuts/
Lien139 : http://code.google.com/p/qfocuser/
Lien140 : http://mapstraction.com/
Lien141 : http://sylvester.jcoglan.com/
Lien142 : http://stevenlevithan.com/regex/xregexp/
Lien143 : http://markchristian.org/projects/textmonster/
Lien144 : http://www.fliquidstudios.com/projects/javascript-url-library/
Lien145 : http://www.openjs.com/scripts/data/ued_url_encoded_data/
Lien146 : http://yellowgreen.de/morecss/
Lien147 : http://www.modernizr.com/
Lien148 : http://code.google.com/p/ie7-js/
Lien149 : http://sizzlejs.com/
Lien150 : http://dillerdesign.com/experiment/DD_roundies/
Lien151 : http://www.dillerdesign.com/experiment/DD_belatedPNG/

Lien152 : http://www.azarask.in/blog/post/socialhistoryjs/
Lien153 : http://alexgorbatchev.com/wiki/SyntaxHighlighter
Lien154 : http://phpjs.org/
Lien155 : http://jcrozier.developpez.com/tutoriels/web/php/classes-et-librairies-utiles-developpeurs/
Lien156 : http://javascript.developpez.com/cours/outils-vraiment-utiles-pour-developpeurs-javascript/
Lien157 : http://javascript.developpez.com/cours/librairies-javascript-vraiment-utiles/
Lien158 : http://dico.developpez.com/html/1078-Langages-OpenGL-Open-Graphics-Library.php
Lien159 : http://opengl.org/resources/libraries/glut.html
Lien160 : http://freeglut.sourceforge.net/
Lien161 : ftp://ftp-developpez.com/arkham46/articles/office/vbaopengl/fichiers/packopenglvb.zip
Lien162 : ftp://ftp-developpez.com/arkham46/articles/office/vbaopengl/fichiers/freeglutdll.zip
Lien163 : http://www.opensource.org/licenses/mit-license.php
Lien164 : http://www.opengl.org/resources/libraries/glut/spec3/node10.html
Lien165 : http://arkham46.developpez.com/articles/office/vbaopengl/?page=page_1#LVI
Lien166 : http://www.khronos.org/opengl/
Lien167 : http://arkham46.developpez.com/articles/office/vbaopenglext/
Lien168 : http://arkham46.developpez.com/articles/office/vbaopengl/?page=page_2#LVII-H
Lien169 : http://arkham46.developpez.com/articles/office/vbaopengl/
Lien170 : http://dolphy35.developpez.com/article/access2010/editeurmacros/
Lien171 : http://access.developpez.com/access2010/temoignages/index.php
Lien172 : http://warin.developpez.com/tutoriels/access/numauto2010/
Lien173 : http://warin.developpez.com/tutoriels/access/access2010/bonnes-pratiques-evenements-table/
Lien174 : http://dico.developpez.com/html/37-Internet-XML-eXtended-Markup-Langage.php
Lien175 : http://julp.developpez.com/c/libxml2/?page=sommaire
Lien176 : http://man.developpez.com/man3/free.3.php
Lien177 : http://franckh.developpez.com/articles/c-ansi/parsing/
Lien178 : http://julien-blanc.developpez.com/articles/cpp/Programmation_par_contrat_cplusplus/
Lien179 : http://dico.developpez.com/html/219-Internet-HTTP-HyperText-Transmission-Protocol.php
Lien180 : http://dico.developpez.com/html/878-Securite-MD5-Message-Digest-5.php
Lien181 : http://dico.developpez.com/html/1230-Telecom-FTP-File-Transfert-Protocol.php
Lien182 : http://dico.developpez.com/html/1589-Securite-SSL-Secure-Sockets-Layer.php
Lien183 : http://dico.developpez.com/html/1591-Internet-HTTPS-HTTPS-Secured.php
Lien184 : http://qt.developpez.com/tutoriels/introduction-qt/
Lien185 : http://qt.developpez.com/tutoriels/katanaenmousse/
Lien186 : http://tcuvelier.developpez.com/cross-gcc/qt4/
Lien187 : http://giminik.developpez.com/articles/C++-Qt4/installation-mysql-firebird-openssl/
Lien188 : http://www.qtsoftware.com/
Lien189 : http://qwt.sourceforge.net/
Lien190 : http://www.libqxt.org/
Lien191 : http://delta.affinix.com/qca/
Lien192 : http://tcuvelier.developpez.com/qt/updater/01-introduction/
Lien193 : http://www.gotw.ca/publications/concurrency-ddj.htm
Lien194 : http://cpp.developpez.com/livres/?page=livresBibliotheques#L9780596521530
Lien195 : http://www.developpez.net/forums/d844607/c-cpp/bibliotheques/qt/sortie-qt-4-6-qt-creator-1-3-beaucoup-nouveautes-multi-touch-support-gestes/
Lien196 : http://qt.developpez.com/defis/01-buddhabrot/images/buddha.png
Lien197 : http://qt.developpez.com/defis/01-buddhabrot/images/buddha_hq.jpg
Lien198 : http://www.developpez.net/forums/f1378/c-cpp/bibliotheques/qt/defis-qt/
Lien199 : http://qt.developpez.com/defis/
Lien200 : http://qt.developpez.com/defis/regles/
Lien201 : http://qt.developpez.com/defis/upload/
Lien202 : http://www.developpez.net/forums/f1378/c-cpp/bibliotheques/qt/defis-qt/
Lien203 : http://qt.developpez.com/defis/01-buddhabrot/
Lien204 : http://www.developpez.net/forums/f376/c-cpp/bibliotheques/qt/
Lien205 : http://www.developpez.net/forums/f1025/c-cpp/bibliotheques/qt/multithreading/
Lien206 : http://www.developpez.net/forums/d840430/c-cpp/bibliotheques/qt/defis-qt/premier-defi-qt/
Lien207 : http://fr.wikipedia.org/wiki/Open_source
Lien208 : http://opensource.org/licenses/zlib-license.php
Lien209 : http://www.bulletphysics.com/Bullet/phpBB3/
Lien210 : http://www.sfml-dev.org/
Lien211 : http://www.libsdl.org/
Lien212 : http://fr.wikipedia.org/wiki/OpenGL_utility_toolkit
Lien213 : http://code.google.com/p/bullet/downloads/list
Lien214 : http://www.cmake.org/
Lien215 : http://www.perforce.com/jam/jam.html
Lien216 : ftp://ftp.perforce.com/jam
Lien217 : http://www.youtube.com/watch?v=3gpcCWKyrGA
Lien218 : http://happy.developpez.com/tutoriels/bullet-physics/fichiers/main.cpp.rar
Lien219 : http://happy.developpez.com/tutoriels/bullet-physics/
Lien220 : http://udk.com/index.html
Lien221 : http://udk.com/download.html
Lien222 : http://udk.com/documentation.html
Lien223 : http://udk.com/licensing.html
Lien224 : http://www.developpez.net/forums/d832889/applications/developpement-2d-3d-jeux/lunreal-development-kit-disponible-gratuitement/
Lien225 : http://mi.eng.cam.ac.uk/~qp202/
Lien226 : http://www.youtube.com/watch?v=vEOmzjImsVc
Lien227 : http://www.developpez.net/forums/d842748/club-professionnels-informatique/actualites/programme-revolutionnaire-permet-transformer-webcam-marche-scanner-3d/