Long64Lib

Version 1.0 – 24 Décembre 2002

Entiers longs 8 octets pour les développeurs 4D

Un plugin *gratuit* pour les bases de données 4te Dimension™

Thibaud Arguillere - targuillere@osmose.net

Ce plugin est gratuit et fait partie du projet *Math4D* (http://www.math4d.net), une librairie de code 4D contenant de nombreuses routines de traitement mathématique, sans intervention de plugin.

Long64Lib, en revanche, est un plugin qui est livré – comme Math4D – gratuitement, avec son code source. Ce plugin, comme son nom l'indique, permet aux développeurs 4D de manipuler des entiers longs codés sur 64 bits. Même si cela aurait pu être développé en code 4D « pur » (en utilisant des BLOBs), il était bien plus facile d'écrire un plugin, qui sera en plus probablement plus rapide à l'exécution.

Ainsi, vous pouvez librement utiliser Long64Lib, mais nous vous remerçions d'avance de bien vouloir lire cette partie d'abord :

C'est gratuit !

Cadeau de Noël d'Osmose Editeur pour les développeur s 4D

et les utilisateurs de bases de données 4D.

Ce plugin est gratuit et peut être utilisé sans payer quoi que ce soit à qui que ce soit. Comme toute chose gratuite, il n'est pas accompagné de support technique ni de garantie, …

VOUS L'UTILISEZ À VOS PROPRES RISQUES

Si vous le souhaitez, vous pouvez inclure le code source dans vos propres plugins ou applications, sans restriction. Si vous redistribuez le source, nous demandons qu'il soit clairement écrit qu'il a été fourni à l'origine par Osmose Éditeur.

Si vous modifiez le code, nous serons heureux de savoir ce que vous avez fait, alors n'hésitez pas à envoyer le code à targuillere@osmose.net. Notez cependant que :

-> Ce n'est absolument pas obligatoire. Vous pouvez modifier le code et conserver les changements pour votre propre usage, même pour un produit commercial.

-> En envoyant vos modifications, vous acceptez qu'elles soient publiées et que tout le monde puisse en profiter gratuitement, etc. Le code source indiquera que vous avez fait ces modifications, améliorations, etc.

Il est probable que le prochain chantier important consistera à implémenter la gestion des entiers longs 64 bits non signés, sauf si d'ici-là, 4D GoldFinger résoud tous les problèmes !

Finalement, tout le plugin consiste en un grand « wrapping » de l'implémentation native des long64 que les compilateurs offrent depuis longtemps.

Où et comment le plugin conserve-t-il les long64 ?

Notre décision a été de stocker les long64 dans des réels 4D. Un réel 4D n'est rien d'autre qu'un « double », toujours codé sur 8 octets. Ainsi, le développeur doit stocker les long64 dans des champs/variables explicitement de type « Réel ».

> *NOTE : à cause de cela (les réels stockent des long64), le développeur ne doit jamais manipuler ces réels lui-même (voir plus loin) et toujours utiliser la routine long64_New pour créer un long64.*

Créer un long64

Pour créer un long64, utiliser ***long64_New***. Cette fonction accepte 2 syntaxes:

-> Si vous passez une chaîne non vide dans $1, elle est utilisée pour créer le long64. La chaîne doit contenir uniquement des chiffres, et peut commencer par « - » ou « + ». Elle ne doit contenir rien d'autre : ni espace, ni rien :

```
$aLong64:=long64_New("10000000000000")  ` BON
$aLong64:=long64_New("10  000 000 000 000") ` PAS BON
```

-> Si vous passez une chaîne vide dans $1, alors vous *devez* passer *deux* entiers longs dans $2 et $3. $2 sera utilisé pour créer la partie « octets de poids faible» et $3 celle « octets de poids fort » du long64 :

```
$aLong:=long64_New("";0x11223344;0x55667788)
```

=> crée le long64 0x11223344556677

Utilisation des long64

Parce que les long64 sont stockés dans des réels, vous ne devez *jamais* utiliser les opérateurs usuels sur ces réels. Vous *devez*, en fait, utiliser les routines du plugin.

Par exemple, si l1 et l2 cont été déclarés C_REEL mais contiennent en fait des long64, vous ne pouvez pas utiliser d'appels tels que (ILLEGAL signifie « vous obtiendrez de mauvais résultats ») :

```
$result:=$l1 + $l2  ` ILLEGAL. Utiliser long64_Add
$result:=$l1 - $l2  ` ILLEGAL. Utiliser long64_Substract
$result:=$l1 * $l2  ` ILLEGAL. Utiliser long64_Multiply
```

```
$result:=$l1 / $l2  ` ILLEGAL. Utiliser long64_Divide
$result:=$l1 % $l2  ` ILLEGAL. Utiliser long64_Modulo

$result:=$l1 & $l2  ` ILLEGAL. Utiliser long64_AND
$result:=$l1 | $l2  ` ILLEGAL. Utiliser long64_InclusiveOR
$result:=$l1 ^| $l2  ` ILLEGAL. Utiliser long64_ExclusiveOR
` [$long64_NOT effectue le complément binaire]
$result:=$l1 >> 25 ` ILLEGAL. Utiliser long64_ShiftRight
$result:=$l1 << 25 ` ILLEGAL. Utiliser long64_ShiftLeft
```

Cette restriction dans l'utilisation des opérateurs s'applique également pour les comparaisons :

```
Si ($l1 > $l2)  ` ILLEGAL : Utiliser long64_Compare
```

Pour récupérer les 4 octets de bas poids ou les 4 octets de haut poids, utiliser les routines ***long64_LowLong*** et ***long64_HighLong***.

Pour ajouter/soustraire/Multiplier/Diviser… une constante numérique à un long64, vous devez construire un long64 à partir de la constante en appelant la routine long64_New :

```
  ` Ajouter 1 à $aLong64 :
$aLong64:=long64_Add($aLong64;  long64_New("1"))
```

Utilitaires

long64_MainDispacth permet au développeur de manipuler plus facilement les long64 dans un EXECUTER.

long64_MAX renvoie la valeur maximale d'un long64 : 0x7FFFFFFFFFFFFFFF.

long64_MIN renvoie la valeur maximale d'un long64 :(-0x7FFFFFFFFFFFFFFF - 1).

long64_ByteSwap réalise le byte-swapping d'un long64.

Syntaxe

OPERATIONS ARITHMETIQUES

long64_Add

long64_Substract

long64_Multiply

long64_Divide

long64_Modulo

Ces routines ont toutes la même syntaxe :

result:=***routine***(l1;l2)

Parametre	Type 4D
l1	Réel (contenant un long64)
l2	Réel (contenant un long64)
result	Réel (contenant un long64)

Ces routines attendent 2 paramètres : un réel 4D préalablement remplit via ***long64_New*** (ou résultat d'une opération long64_nn précédente). Elles retournent le résultat de l'opération. Les noms des routines sont, nous pensons, suffisament explicites pour se passer d'explications supplémentaires.

result:=***long64_Add***(aLong64;anOtherOne)

MANIPULATION DE BITS

long64_AND

long64_InclusiveOR

long64_ExclusiveOR

long64_NOT

long64_ShiftRight

long64_ShiftLeft

long64_HighLong

long64_LowLong

long64_AND, ***long64_InclusiveOR*** et ***long64_ExclusiveOR*** acceptent la syntaxe suivante :

result:=***routine***(l1;l2)

Paramètre	Type 4D
l1	Réel (contenant un long64)
l2	Réel (contenant un long64)
result	Réel (contenant un long64)

l1 et l2 must doivent avoir été créés via ***long64_New*** ou sont le résultat d'une opération long64_nn précédente.

long64_NOT attend un seul paramètre (type 4D réel) et renvoie son complément binaire :

```
result:=long64_NOT(aLong64)
```

long64_ShiftRight and long64_ShiftLeft font ce qu'elles sont supposées faire :

Paramètre	Type 4D
l1	Réel (contenant un long64)
l2	Entier ou entier long
result	Réel (contenant un long64)

IMPORTANT : le plugin ne vérifie pas si l2 est bien >=0 et <=63.

long64_HighLong et ***long64_lowLong*** renvoient respectivement les 4 octets de poids fort et les 4 octets de poids faible du long64. Elles retournent une valeur de type entier long.

long64_Compare

```
result:=long64_Compare(l64_1;l64_2)
```

Les long64 étant stockés dans des réels 4D, il n'est pas possible de les comparer directement. IL faut utiliser cette fonction qui attend deux réels 4D stockant des long64, et qui renvoie un entier dont la valeur peut être :

0 : l64_1 = l64_2

-1 : l64_1 < l64_2

1 : l64_1 > l64_2

UTILITAIRES et DIVERS

long64_New crée un long64 et le renvoie sous la forme d'un réel 4D. Voir le début de cette documentation.

long64_ToString

```
$l64AsString:=long64_ToString(aLong64)
```

aLong64 est un réel4D construit avec ***long64_New*** ou résultat d'une opération précédente.

$l64AsString est le Long64 converti en chaîne. Aucun formatage n'est réalisé :

```
$strMax:=long64_ToString  long64_MAX)
```

...renvoie "9223372036854775807" dans $strMax.

long64_MainDispatch

$result:=***long64_MainDispatch***(selector;l1;l2)

Paramètre	Type 4D
selector	Alpha (longueur = 1)
l1	Réel (contenant un long64)
l2	Réel (contenant un long64)
result	Réel (contenant un long64)

Le paramètre selector est une chaîne/texte de longueur 1. Il doit contenir le symbole de l'opération à effectuer :

"+", "-", "*", "/", "%"

"1", "|", "^", "~"

Pour le décallage de bits à droite ou à gauche, utiliser un seul signe de décallage au lieu de 2 : « > » au lieu de « >> » et « < » au lieu de « << ».

long64_MAX n'atttend pas de paramètre et renvoie la valeur maximale que peut prendre un long64 :

0x7FFFFFFFFFFFFFFF

9 223 372 036 854 775 807

long64_MIN n'atttend pas de paramètre et renvoie la valeur minimale que peut prendre un long64 :

0x7FFFFFFFFFFFFFFF - 1

-9 223 372 036 854 775 808

long64_ByteSwap attend un réel 4D stockant un long64 et renvoie le même long64 byte-swappé. Cela peut être utile lors des échanges de valeurs entre Mac et PC.