Création et utilisation des Dll sous Visual C++

Nous allons, dans ce court tutorial, voir quelques exemples de création de Dll sous Visual C++. Ces exemples seront séparés en deux catégories : Dans l'une nous verrons l'utilisation de la Dll crée en C++ et dans l'autre dans un programme en Visual Basic.

Mais qu'est ce que c'est qu'une Dll ?

Une Dll est un morceau de programme incapable de fonctionner seul. Mais alors quel est l'intérêt de mettre du code dans une Dll plutôt que dans le programme principal ?

- Premièrement, cela permet de ne pas se trouver avec un programme trop gros, un programme peut appeler autant de Dll qu'il en a besoin. Le programme est donc plus léger.
- Le fait de diviser le programme en Dll facilite le travail en équipe, chacun peut se consacrer à une seule Dll sans s'occuper des autres.
- Si une mise à jour du programme doit être effectuée, seule une seule Dll peut être modifiée, cela évite de redistribuer l'application au grand complet.
- Pour finir, plusieurs applications peuvent utiliser une même Dll simultanément, cela évite de devoir écrire les fonctions contenues dans cette Dll dans tous les programmes l'utilisant.

Premier exemple

Nous allons créer ici une Dll très simple, elle contiendra une fonction servant à calculer la factorielle d'un nombre.

Dans Visual C++, choisissez **New Project** et **Win32 Project**.
Choisissez ensuite **Dll** et laissez **Empty Project** et **Export symbols** vides.

Voici la fonction que nous voulons coder :

```
unsigned long int factorielle(int n)
{
  unsigned long int resultat = 1;

  if(n < 0)
    return -1;
  if(n == 0)
    return 1;
  for(; n > 0; n--)
    resultat *= n;

  return resultat;
}
```

Nous allons écrire notre fonction juste après

```
#include "stdafx.h"
BOOL APIENTRY DllMain( HANDLE hModule,
                       DWORD  ul_reason_for_call,
                       LPVOID lpReserved
                              )
{
    return TRUE;
}
```

Comme ceci :

```
extern "C" __declspec(dllexport) unsigned long int factorielle(int n)
{
  unsigned long int resultat = 1;

  if(n < 0)
    return -1;
  if(n == 0)
    return 1;
  for(; n > 0; n--)
    resultat *= n;

  return resultat;
}
```

`extern "C" __declspec(dllexport)` indique au compilateur que la fonction pourra être appelée depuis un programme extérieur. Toutes les fonctions qui ne serviront pas uniquement à l'intérieur de la Dll doivent être précédées de ceci.

Essayez de compiler. Vous devriez obtenir un fichier **.lib** ainsi qu'un **.dll**. Gardez les, vous en aurez besoin pour le programme client.

Créez le programme client, il peut s'agir d'un simple programme Console. Ajoutez cette ligne pour que la Dll soit incluse dans le projet (les fichiers **.lib** et **.dll** doivent se trouver dans le répertoire du projet).

```
#pragma comment (lib, "dll_factorielle.lib")
```

Ensuite, il suffit d'importer les fonctions comme ceci :

```
extern "C" __declspec(dllimport) unsigned long int factorielle(int n);
```

La fonction est ensuite utilisable comme une fonction « normale ».

Il suffit ensuite de distribuer le fichier **.dll** avec l'exécutable.

Deuxième exemple

Nous allons créer une petite Dll en C++ que nous utiliserons à partir d'un programme en Visual Basic. Elle sera tout simple puisqu'elle prendra en paramètre un tableau d'entiers en les triera dans l'ordre croissant. Le programme en Visual Basic affichera ensuite ces nombres.

Dans Visual C++, choisissez **New Project** et **Win32 Project**.
Choisissez ensuite **Dll** et cochez **Empty Project**.
J'ai choisi d'appeler ce projet **dll_classement**.

Il nous faut ajouter deux fichiers à ce projet :

- **dll_classement.cpp** qui contiendra le code
- **dll_classement.def** qui contiendra les informations d'exportation

Voici le contenu du fichier **dll_classement.cpp** :

```
void classement(int * nombres)
{
  int temp;

  for(int i = 1; i < nombres[0] - 1; i++)
  {
    for(int j = i + 1; j < nombres[0]; j++)
    {
      if(nombres[i] > nombres[j])
      {
        temp = nombres[i];
        nombres[i] = nombres[j];
        nombres[j] = temp;
      }
    }
  }
}
```

La fonction classement prend en paramètre un tableau de nombres, le premier nombre de ce tableau contient le nombre de nombres contenu dans le tableau…

Il s'agit du tri pas insertion, on parcourt le tableau et regardant un nombre et le suivant, si le premier est plus grand que le deuxième, on les inverse.

Voici le contenu du fichier **dll_classement.def** :

```
LIBRARY     dll_classement

EXPORTS
  classement   @1
```

La fonction classement est donc exportée.

Voici la procédure de déclaration de la fonction sous Visual Basic (dans un Module) :

```
Declare Sub classement Lib "dll_classement.dll" (nombres As Long)
```

Il faut remarquer que des **int** en C++ correspondent à des **Long** en Visual Basic.

Voici le code utilisé (j'ai placé un contrôle **ListBox** nommé **List1**)

```
' tableau de nombres
Dim nombres(0 To 10) As Long
' variable de boucle
Dim i As Integer

' nombre de nombre + 1 (en comptant l'indice 0)
nombres(0) = 11
' dès ici, nombres à classer
nombres(1) = 5
nombres(2) = 6
nombres(3) = 2
nombres(4) = 45
nombres(5) = 65
nombres(6) = 5
nombres(7) = 4
nombres(8) = 5
nombres(9) = 1
nombres(10) = 8

' appel de la fonction
classement nombres(0)

' affichage du résultat
For i = 1 To nombres(0) - 1
List1.AddItem (nombres(i))
Next i
```

La Dll est à distribuer avec l'exécutable. Il est possible que vous deviez créer l'exécutable pour tester le programme.

Voici la fin de ce cours tutorial… Si vous avez des critiques, des conseils d'améliorations des problèmes, des encouragements, n'hésitez pas à me mailer à raf_guglielmetti@yahoo.fr