

# Compiler les libs wxWidgets sous Windows

(Révision 4 du 03/07/2008)

## Introduction

Bonjour à tous.

Nous allons voir, tout au long de ce tutoriel, comment compiler les libs wxWidgets sur votre ordinateur. Je vais essayer d'être le plus précis possible, mais il se peut que des oublis ou des erreurs se glissent dans mes explications. Si vous en détectez merci de me le faire savoir sur le site pour lequel ce tutoriel a été rédigé : [www.wxdev.fr](http://www.wxdev.fr)

## Ce qu'il vous faut avant de commencer

### **Quel que soit votre environnement de développement**

La première chose à obtenir pour pouvoir compiler les libs wxWidgets est l'archive contenant les fichiers sources.

Pour cela, rendez-vous sur le site officiel ([www.wxwidgets.org](http://www.wxwidgets.org)) dans la partie « Téléchargements » (Downloads), et rapatriez l'archive nommée wxAll.

Il vous faudra ensuite la décompresser : évitez, tant que faire se peut, de placer les fichiers dans un dossier dont le nom contient des espaces, car cela est très mal géré par certains compilateurs.

A partir de maintenant, je désignerais le répertoire obtenu par WXDIR. Afin d'être sûrs que l'on parle bien du même répertoire, il doit contenir les sous-dossiers « art, build, contrib, demos, distrib, docs, ..... ».

### **Pour la compilation avec MinGW**

Il vous faut bien entendu une installation fonctionnelle de MinGW. Pour vous en assurer, ouvrez une fenêtre de commandes (*Démarrer* → *Programmes* → *Accessoires* → *Invite de commandes*) et entrez la commande : `Mingw32-make.exe -version`

Vous devez obtenir un message vous indiquant le numéro de version de MinGW. Si ce n'est pas le cas, je vous laisse le soin de revoir votre installation, en vous aidant des informations que vous pourrez trouver sur le site officiel de MinGW : <http://mingw.org>.

### **Pour la compilation avec Visual C++ 2005**

Vous devez avoir installé Visual C++ 2005, que ce soit la version Express ou la version complète.

Il vous faut également le « Platform SDK » qui vous permettra d'avoir les fichiers headers nécessaires à la compilation. De plus, l'installation du « Platform SDK » a dû vous créer, dans le menu « Démarrer », les raccourcis permettant d'ouvrir une fenêtre « Dos » avec les variables d'environnement préréglées (*Démarrer* → *Tous les programmes* → *Microsoft Platform SDK* → *Open Build Environment Window*).

### **Pour la compilation avec Visual C++ 2008**

L'installation est beaucoup plus simple : un seul installateur pour la totalité du produit (Visual C++, le SDK, et même MSDN si ça vous chante).

Pour le raccourci permettant d'ouvrir la fenêtre « Dos » avec les variables d'environnement préréglées, il faut cliquer sur *Démarrer* → *Tous les programmes* → *Visual C++ 9.0 Express Edition* → *Visual Studio Tools* → *Invite de commandes de Visual Studio 2008*.

## Avant de commencer la compilation

Tout d'abord, il faut modifier le fichier servant à définir quelles parties des libs vous voulez compiler. Cela se fait par l'intermédiaire du fichier « setup.h ».

Il faut éditer le fichier WXDIR\include\wx\msw\setup.h. Je vous conseille d'en faire une copie au préalable, on ne sait jamais (ne serait-ce que pour révérifier plus tard quelles étaient les options par défaut).

Pour ma part, afin de vous fournir des libs les plus complètes possibles, voici les constantes que j'ai modifié dans ce fichier, ainsi que leurs valeur :

- #define wxUSE\_GLCANVAS 1
- #define wxUSE\_GRAPHICS\_CONTEXT 1 (uniquement avec Visual Studio)
- #define wxUSE\_ODBC 1

Vous pouvez, de votre côté, passer d'autres valeurs à « 0 » afin d'alléger la taille des libs, mais essayez d'abord de bien savoir ce que vous faites, avant de désactiver une option ou l'utilisation d'un contrôle. Lorsque votre fichier « setup.h » est prêt, passez au bon paragraphe, en fonction de votre environnement de développement.

## **Compiler les libs**

Nous allons utiliser une fenêtre « DOS » pour lancer la compilation.

- Pour MinGW, ouvrez une fenêtre DOS classique (*Démarrer → Tous les programmes → Accessoires → Invite de commande*)
- Pour Visual C++, passez par le raccourci dont je vous ai parlé précédemment (*Démarrer → Tous les programmes → Microsoft Platform SDK → Open Build Environment Windows → Windows XP 32bits Build Environment → Set Windows XP 32-bit Build Environment (Retail)* pour la version 2005 et *Démarrer → Tous les programmes → Visual C++9 Express Edition → Visual Studio Tools → Invite de commande de Visual Studio 2008* pour la version 2008 ).

Placez-vous dans répertoire wxWidgets :

```
cd WXDIR
```

Et descendez jusqu'au répertoire contenant les « makefile » :

```
cd build
cd msw
```

Vous constaterez que ce répertoire contient les fichiers « makefile » pour chaque compilateur :

- pour MinGW, il s'agit du fichier « makefile.gcc »
- pour Visual C++, il s'agit du fichier « makefile.vc »

Il vous reste maintenant à savoir quel type de libs vous voulez créer :

- Release ou Debug
- Ansi ou Unicode
- Statiques ou Dynamiques
- Monolithiques ou Multi-libs

Vous pourrez, si vous le désirez, décider de compiler plusieurs versions (parmi les 16 combinaisons d'options ci-dessus) : elles pourront toutes cohabiter les unes avec les autres, tant que vous ne modifiez pas le fichier « setup.h » en cours de route.

Vous pouvez également définir, dans le cas des libs dynamiques, une chaîne de caractères qui apparaîtra dans le nom des fichiers dll. Il s'agit de l'option « Vendor ». Si vous ne la définissez pas, la chaîne « custom » sera utilisée.

Voici les commandes à entrer : je vous expliquerai ensuite les variantes en fonction des options ci-dessus :  
Pour MinGW :

```
mingw32-make.exe -f makefile.gcc BUILD=Release MONOLITHIC=0 UNICODE=1 SHARED=1
VENDOR=Xaviou USE_OPENGL=1 USE_ODBC=1 USE_QA=1
```

Pour Visual C++ :

```
nmake.exe -f makefile.vc BUILD=Release MONOLITHIC=0 UNICODE=1 SHARED=1
VENDOR=Xaviou USE_OPENGL=1 USE_ODBC=1 USE_QA=1 USE_GDIPLUS=1
```

Les explications :

- La première partie (mingw32-make.exe ou nmake.exe) est le nom de l'exécutable qui va se charger de gérer la compilation.
- Ensuite, on trouve l'option « -f », ainsi que le nom du fichier « makefile », qui est différent pour chaque compilateur.
- BUILD = Release (ou Debug, mais je doute que vous vouliez déboguer les libs wxWidgets).
- MONOLITHIC=0 (ou 1) Les libs « Monolithiques » ne génèrent qu'un seul fichier, et sont donc plus faciles à gérer, mais obligent à fournir un exécutable ou une dll plus grosse.
- UNICODE=1 (ou 0) Active ou non le support Unicode

- SHARED=1 (ou 0) pour créer les fichiers dll des libs (1) ou pour intégrer les libs à l'exécutable (0).
- VENDOR=xxxx : voir ci-dessus (n'est pris en compte qu'en mode SHARED=1).
- USE\_OPENGL=1 (ou 0) : pour créer la partie OpenGL des libs wxWidgets (ou non)
- USE\_ODBC=1 (ou 0) : comme ci-dessus, pour la partie « odbc ».
- USE\_QA=1 (ou 0) : idem, pour les libs « qa » .
- USE\_GDIPLUS=1 (ou 0) : pour l'utilisation des wxGraphicsContext (ne marche pas avec MinGW).

Voilà, c'est tout, il ne vous reste plus qu'à patienter, car suivant les options sélectionnées, et le compilateur utilisé, cela peut prendre plus ou moins de temps.

Il est à noter que la compilation en mode « SHARED » avec MinGW génère une multitude d'avertissements, mais il ne faut pas en tenir compte : c'est apparemment un problème connu sur les forums wxWidgets (pour info, je me suis amusé à rediriger ces avertissements vers un fichier texte, et il faisait au final près de 130 Mo).

## **Compiler les libs additionnelles**

La compilation s'est bien passée, et vous êtes maintenant en possession de vos libs prêtes à être utilisées. Il y a cependant des parties additionnelles des libs que vous pouvez également compiler, si vous en avez besoin. Elles se trouvent dans le répertoire « WXDIR\contrib » :

- « fl » (pour Frame Layout) : ajoute des facilités pour gérer l'organisations de différents « panels » à l'intérieur d'une fenêtre (déplacement par glisser/déplacer, accrochage, ...).
- « foldbar » : gestion d'un menu structuré en sous-parties comme dans la partie gauche de l'explorateur windows (la zone bleue par défaut que l'on obtient quand la liste des dossiers n'est pas affichée).
- « gizmos » : collection de différentes classes qui peuvent s'avérer très utiles dans certains cas : wxDynamicShashWindow, wxEditableListBox, wxLEDNumberCtrl, wxMultiCellCanvas, wxRemotelyScrolledTreeCtrl, ...
- « mmedia » : ajout de fonctionnalités multimédia diverses
- « net » : ajout de classes concernant les réseaux (intranet, internet, email, ...)
- « ogl » (pour Object Graphics Library) : API pour les applications ayant besoin d'afficher des objets connectés par lignes, ...
- « plot » : classes destinées au dessin de courbes
- « stc » (Styled Text Control) : Contrôle utilisant Scintilla (un éditeur avec coloration syntaxique).
- « svg » : classes pour la gestion de fichiers svg (Simple Vector Graphics).

Pour les compiler le processus est le même que pour les libs principales (la seule différence étant qu'il n'y a pas besoin de modifier le fichier « setup.h »).

Ainsi, pour compiler « fl », il vous suffit de vous placer dans le répertoire « WXDIR\contrib\build\fl » et de ressaisir la commande ayant servi à compiler les libs principales.

Il faut répéter cette opération pour chaque « contrib » que vous désirez compiler

## **Compiler les exemples**

Lorsque les libs (et éventuellement les libs additionnelles) sont compilées, vous pouvez, si vous le désirez, compiler quelques exemples présents dans le dossier « WXDIR\samples » (ainsi que dans le dossier « WXDIR\contrib\samples »).

La méthode est toujours la même : placez-vous dans le dossier de l'exemple concerné et entrez la commande de compilation que nous avons vue plus haut.

## **Quelle configuration choisir ?**

Ça, c'est à vous de voir, en fonction de vos habitudes en programmation, de ce que vous voulez faire, ...

Il faut simplement savoir que, par exemple, les libs « Unicode » poseront problème pour fonctionner sur Windows 95 ou 98 qui ne prennent pas en charge l'unicode.

Par contre, les systèmes Windows 2000 et supérieurs (je ne sais pas ce qu'il en est pour Windows Me) utilisent nativement l'Unicode. Les applications « Ansi » seront en quelque sorte « traduites » lors de leur exécution.

Ensuite, à vous de voir si vous voulez fournir un simple exécutable, ou cet exécutable accompagné d'une ou plusieurs dll (par exemple, si vous avez plusieurs exécutables à fournir, une seule version des dll suffira).

Si vous pensez qu'à ce stade, il est encore trop tôt pour faire un choix, sachez que vous pouvez tout à fait faire cohabiter plusieurs configurations des libs dans le même dossier :

- unicode ou ansi
- dynamiques ou statiques
- monolithiques ou multilibs

Il vous suffit simplement de relancer le processus de compilation (sans toucher au fichier « setup.h ») en activant ou désactivant les paramètres correspondants.

## **C'est fini pour l'instant**

Voilà, vous devez normalement avoir une installation fonctionnelle des libs wxWidgets.

Pour simplifier un peu, je vous conseille de créer un répertoire « bin » dans lequel vous placerez toutes les dll éventuellement obtenues, et ajoutez ce répertoire à la variable système « PATH » afin que Windows sache où les trouver. Elles se trouvent normalement dans les dossiers « WXDIR\lib\gcc\_dll » et « WXDIR\lib\vc\_dll », en fonction des options sélectionnées lors de la compilation.

Ce tutoriel est maintenant terminé.

Je vous invite à me signaler les éventuels problèmes que vous auriez pu rencontrer lors de sa lecture, afin de le faire évoluer en fonction de vos besoins.

En attendant, merci de votre attention, bonne prog et @+.

Xav'

## **Révisions :**

Rev.	Date	Changements
01	27/12/2007	Création du tutoriel
02	05/03/2008	Ajout des différences pour Ms Visual C++ 2008
03	09/04/2008	Corrections diverses + Ajout options GDI+ et wxGraphicsContext
04	03/07/2008	Corrections diverses + Compilation des contribs et des exemples