**TUTORIEL WINDOWS**

# Introduction

Bonjour, ce tutoriel vous donne les différentes étapes à suivre pour faire fonctionner le code PFE sous Windows.

Les bibliothèques nécessaires sont :

1. Boost

2. Eigen3

3. Csparse

4. Tiff3

# Installation des différentes BIBLIOTHEQUes

1. Pour installer **Boost**, téléchargez l’installer de [**http://www.boostpro.com/download/**](http://www.boostpro.com/download/)

Installez-le.

2. **Eigen** n’a pas besoin d’être installé, il suffit de la copier dans le même dossier que les header files(.hpp). Dans notre cas, c’est le dossier Includes.

3. **Csparse** : Pour la compiler, il vous faut le programme MinGW. Après l’installation de ce dernier, naviguez dans le dossier Csparse et faites un « **make »**  puis entrez dans le dossier Lib (Csparse/Lib) et changez l’extension **libcsparse.a** en **libcsparse.lib.** C’est ce fichier dont on a besoin pour réussir à compiler notre code.

4. Pour installer Tiff, téléchargez l’installer à partir de l’URL qui suit :

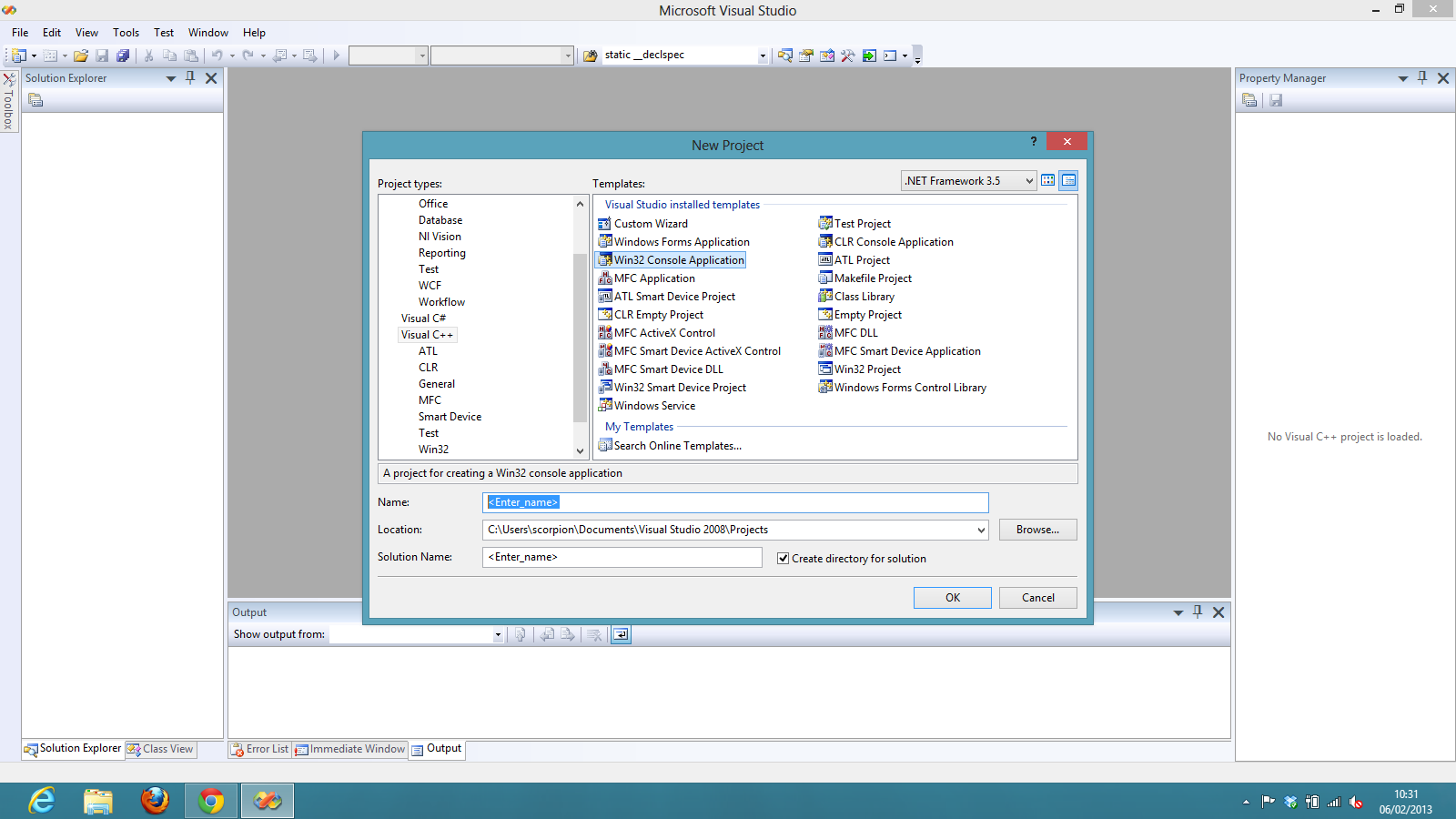
[**http://gnuwin32.sourceforge.net/packages/tiff.htm**](http://gnuwin32.sourceforge.net/packages/tiff.htm)

Installez-la. Dans le dossier d’installation, il y a un autre dossier qui s’appelle lib. Dans ce dossier, c’est le fichier **libtiff.lib** qu’on va utiliser ensuite pour linker notre projet et utiliser cette bibliothèque.

# Création d’un nouveau Projet

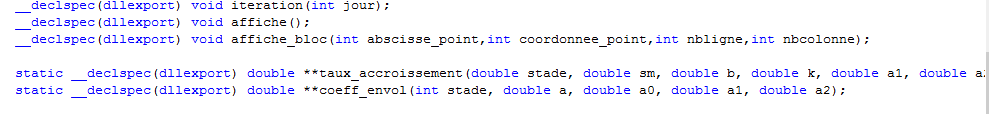
Pour générer la bibliothèque libpfe sous Windows, vous devez créer un nouveau projet de type **Application console Win32.**

1. Dans le menu **Fichier**, cliquez sur **Nouveau**, puis sur **Projet...**.
2. À partir du volet **Types de projets**, sous **Visual C++**, sélectionnez **Win32**.
3. À partir du volet **Modèles**, sélectionnez **Application console Win32**.
4. Choisissez un nom pour le projet, tel que **pfe**, et entrez-le dans le champ **Nom**. Choisissez un nom pour la solution, tel que **libpfe** et entrez-le dans le champ **Nom de solution**.
5. Cliquez sur **OK** pour démarrer l'Assistant Application Win32. À partir de la page **Vue d'ensemble** de la boîte de dialogue **Assistant Application Win32**, appuyez sur **Suivant**.
6. Dans la page **Paramètres de l'application** de l'**Assistant Application Win32**, sous **Type d'application**, sélectionnez **DLL** si la valeur est disponible ou **application Console** si **DLL** n'est pas disponible. Certaines versions de Visual Studio ne prennent pas en charge la création d'un projet de DLL à l'aide d'Assistants. Vous pouvez ultérieurement la modifier pour que votre projet soit compilé en DLL.
7. Dans la page **Paramètres de l'application** de l'**Assistant Application Win32**, sous **Options supplémentaires**, sélectionnez **Projet Vide**.
8. Cliquez sur **Terminer** pour créer le projet.



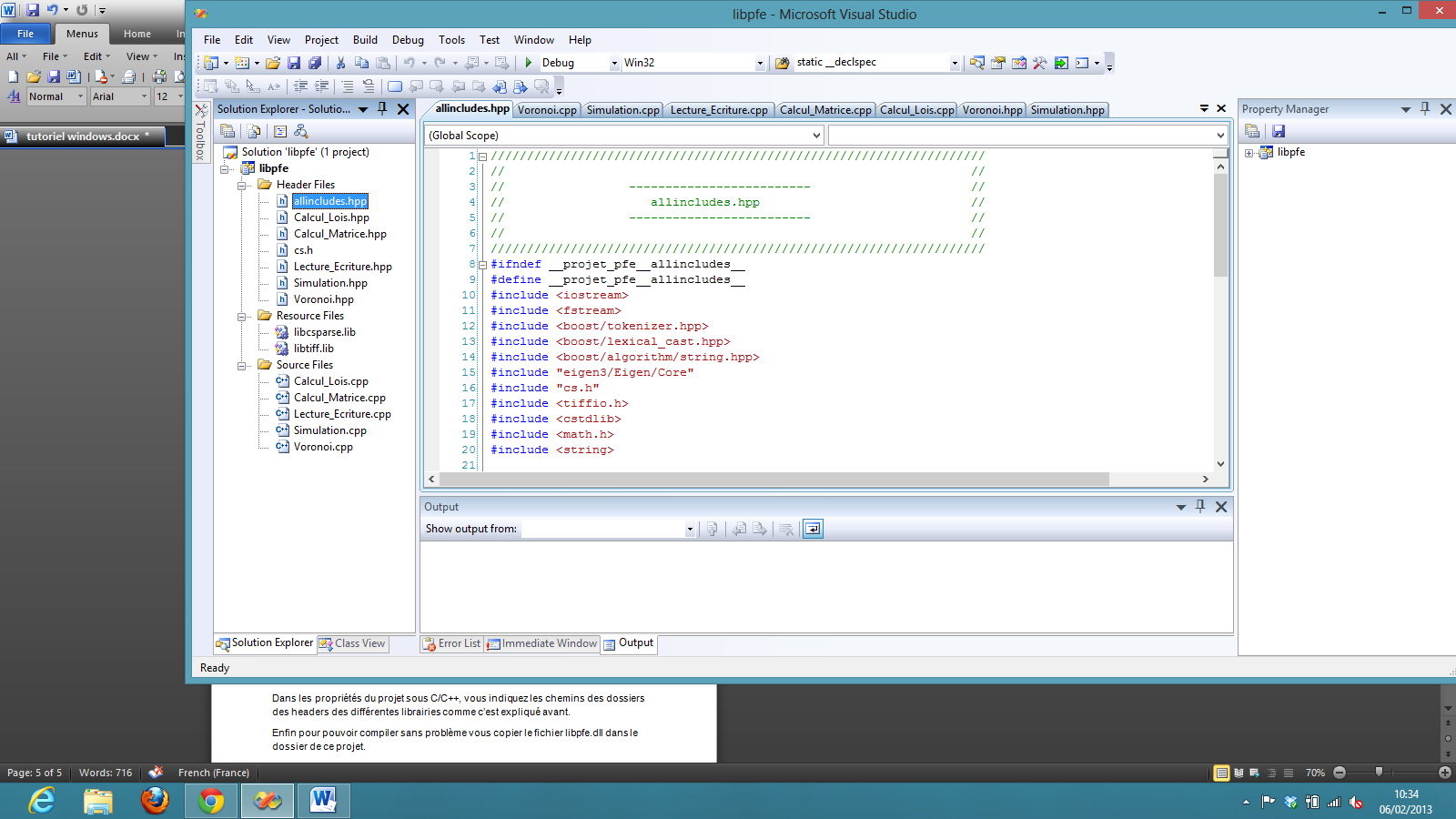
Dans le panel gauche du projet :

Ajoutez les fichiers .hpp dans **Header files**.

Pour pouvoir utiliser la bibliothèque qu’on veut générer, dans d’autres projets, il faut ajouter, avant chaque prototype (dans tous les fichiers hpp), **\_\_declspec(dllexport)** et **static \_\_declspec(dllexport)** si la méthode est statique.

Ajoutez les fichiers sources .cpp dans **Source files**.

Pour pouvoir utiliser la bibliothèque qu’on veut générer dans d’autres projets, il faut ajouter **#include <stdexcept>** dans tous les fichiers sources.

Ajoutez ensuite libtiff.lib et libcsparse.lib dans **Resource files.**

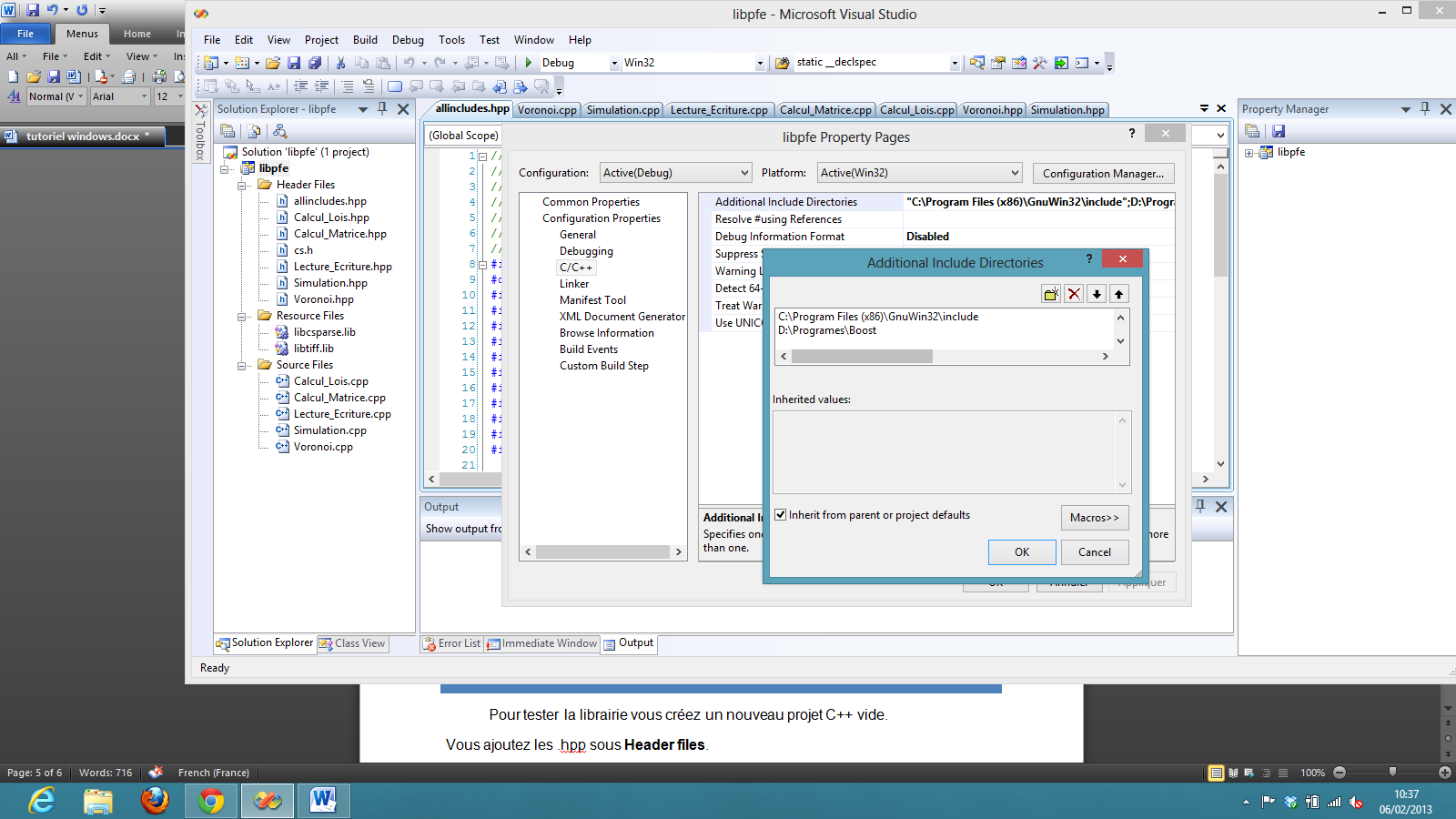
La dernière étape à réaliser est la spécification des chemins des différents dossiers qui contiennent les headers des bibliothèques boost et tiff.

Ouvrez alors les propriétés de votre solution, et, dans le panel gauche, cliquez sur **C/C++** dans la ligne **Additional Include Directories.**

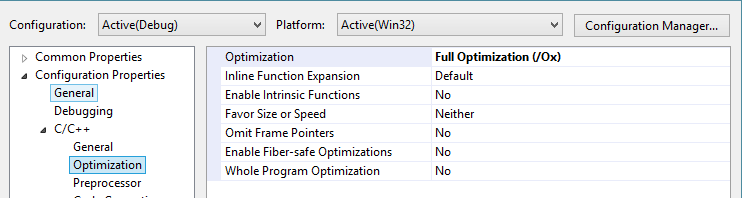
Ajoutez les chemins des dossiers d’installation des bibliothèquess boost et tiff.

Exemple : **C:\Program Files (x86)\GnuWin32\include**

et **C:\Program Files\boost\boost\_1\_51**

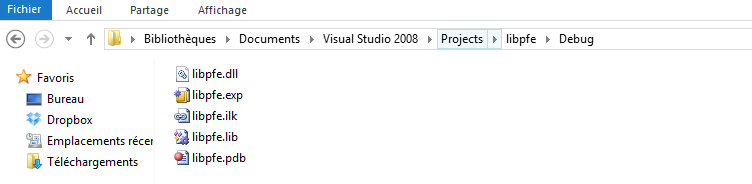
****

Pour activer l’optimisation automatique, dans le panel gauche, sous **C/C++**, cliquez sur **Optimization** et à la ligne **Optimization**, mettez **/O2** ou **/Ox**.



Vous pouvez maintenant compiler votre projet sans problème.

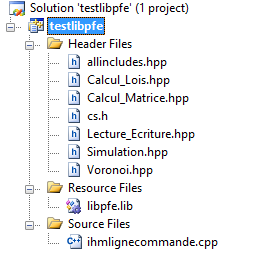
Dans le dossier **Debug** de votre projet, vous devez voir deux fichiers : **libpfe.dll** et **libpfe.lib**.



# Tester la bibliothèque en ligne de commande

Pour tester la bibliothèque, créez un nouveau projet C++ vide.

Ajoutez les .hpp dans **Header files**, ihmlignecommande.cpp dans **source files,** libpfe.lib sous **Resource files.**

****

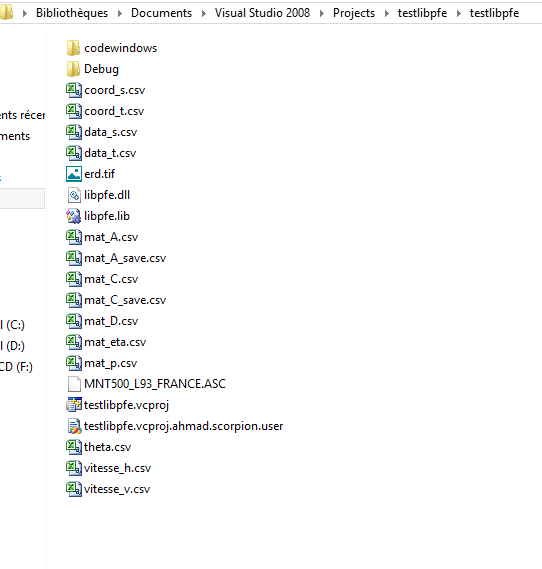
Dans les propriétés du projet, sous **C/C++,** indiquez les chemins des dossiers des headers des différentes bibliothèques comme expliqué précédemment.

Enfin, pour pouvoir compiler sans problème, copiez le fichier **libpfe.dll** dans le dossier de ce projet.

**Exemple** : J’ai créé un projet qui s’appelle testlibpfe. Je copie alors le .dll dans C:\Users\scorpion\Documents\Visual Studio 2008\Projects\testlibpfe\testlibpfe

Le projet doit alors compiler sans problème.

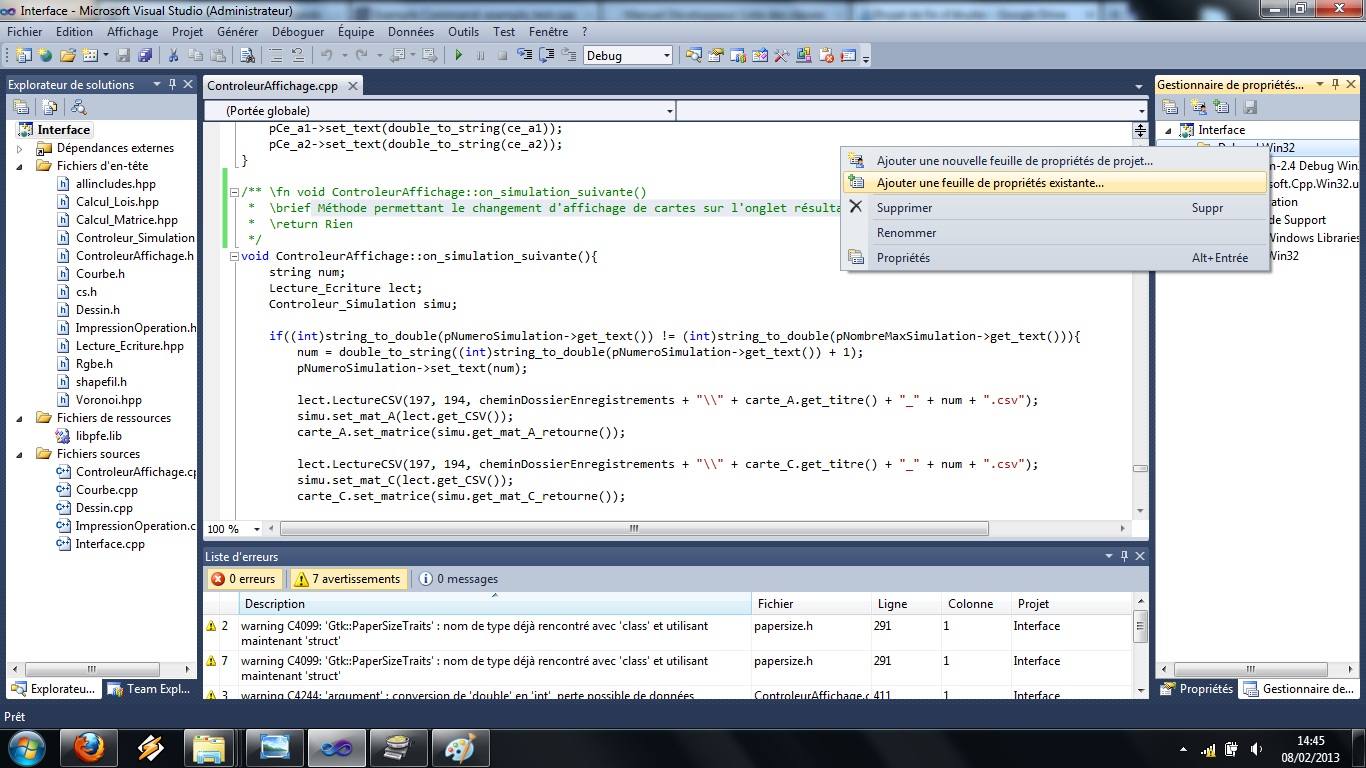
**Remarque** : Vous devez copier tous les .csv dans le dossier du projet.



# Tester la bibliothèque avec interface graphique

Pour tester la bibliothèque, créez un nouveau projet C++ vide.

Ajoutez les .hpp dans **Header files**, ihmlignecommande.cpp dans **source files,** libpfe.lib sous **Resource files.**

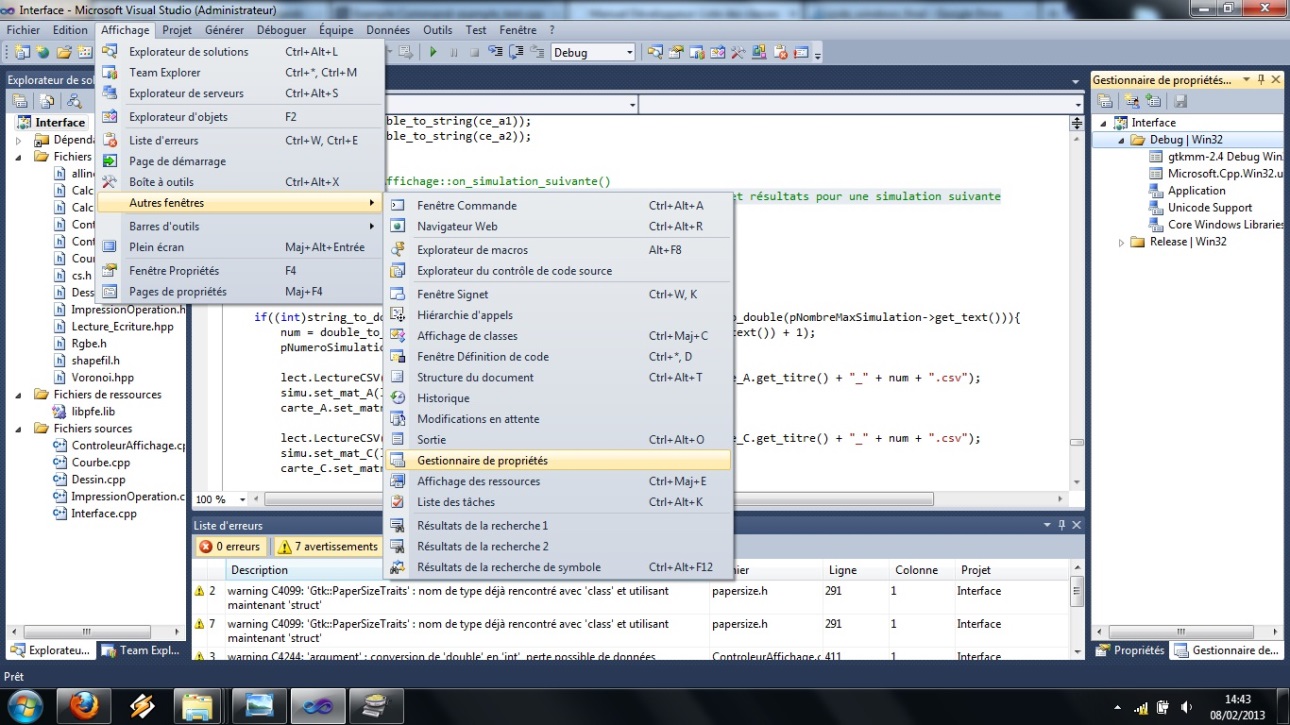
****

Dans les propriétés du projet, sous **C/C++,** indiquez les chemins des dossiers des headers des différentes bibliothèques comme expliqué précédemment.

Enfin, pour pouvoir compiler sans problème, copiez le fichier **libpfe.dll** dans le dossier de ce projet.

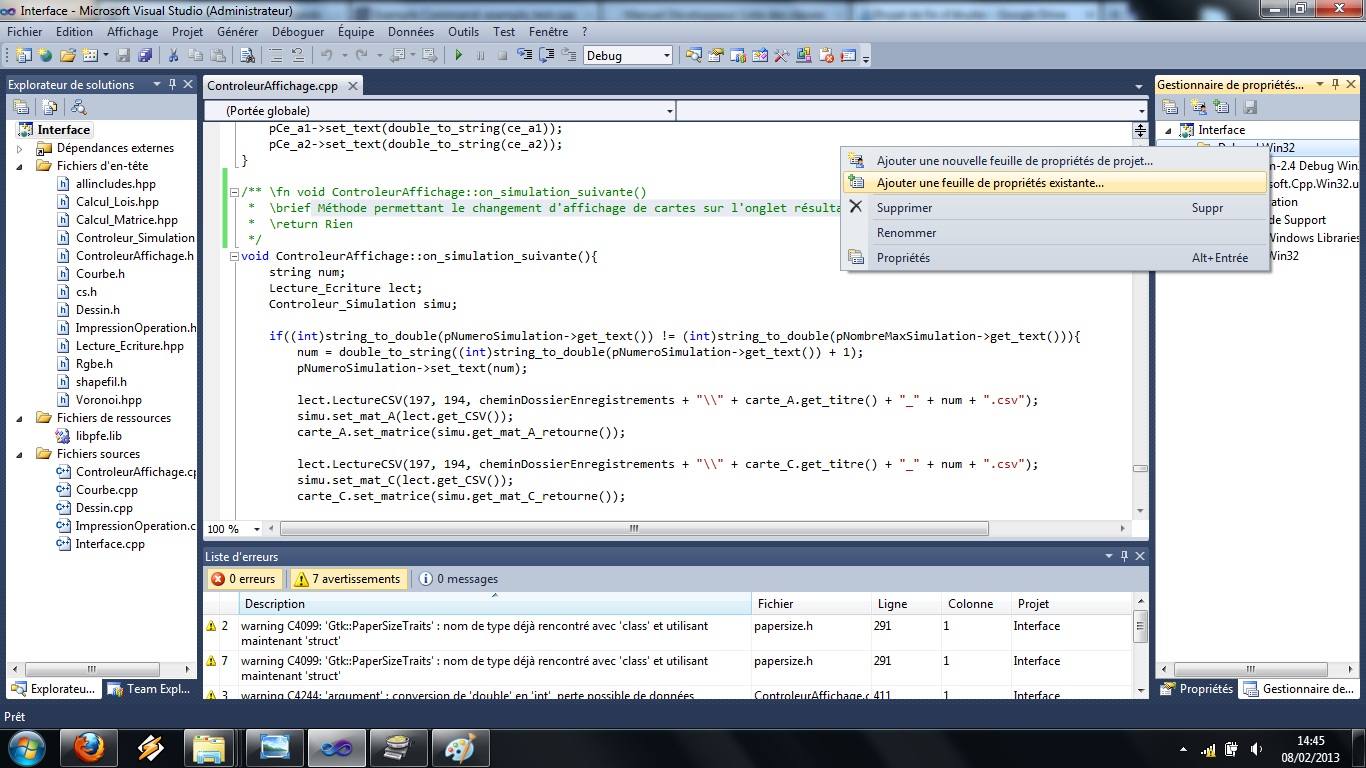
**Exemple** : J’ai créé un projet qui s’appelle testlibpfe. Je copie alors le .dll dans C:\Users\scorpion\Documents\Visual Studio 2008\Projects\testlibpfe\testlibpfe

Vous devez ajouter les propriétés de GTK pour pouvoir fonctionner sous VS 2008. Pour ce faire, vous affichez le **panneau de configuration**



Sur cette image ci-dessus, on peut visualiser le panneau de configuration sur la droite.

Ensuite, une fois après avoir installé GTKmm vous trouverez dans le chemin d’installation un dossier **MSVC** contenant un fichier s’intitulant **gtkmm-2.4d.vsprops.** Celui-ci doit être ajouté dans le dossier **Debug du panel de configuration** en faisant un clic droit sur le dossier debug et sélectionner **Ajouter une feuille de propriétés existante.** Vous naviguez vers **gtkmm-2.4d.vsprops**. L’image suivante illustre ces propos :



Le projet doit alors compiler sans problème.

**Remarque** : Vous devez copier tous les .csv dans le dossier du projet.

