

*Université de Sherbrooke
Département de génie électrique et informatique*

**Guide de configuration et d'utilisation des classes
et bibliothèques du livre « La Bible C++ »**

Guide rédigé par Pascal Durocher

*Révision 3
(12 septembre 2006)*

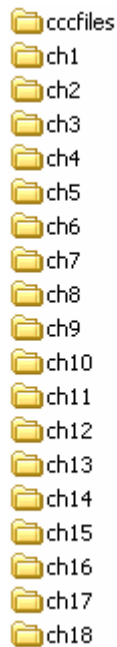
Installation des fichiers du livre « La Bible C++ ».....	4
La classe Time	5
La bibliothèque CCC_WIN	8
Récapitulation	13

Vous êtes bon, même très bon, car vous êtes rendu aux problèmes du chapitre 3. Vous devez donc utiliser la class `Time` et la bibliothèque graphique `CCC_WIN`. Voici comment faire...

Installation des fichiers du livre « La Bible C++ »

Dans un premier temps, vous devez télécharger les classes et bibliothèques proposées dans votre livre. Pour ce faire, dirigez-vous sur la page Web de l'APP2 et téléchargez le fichier `cccfiles.zip`.

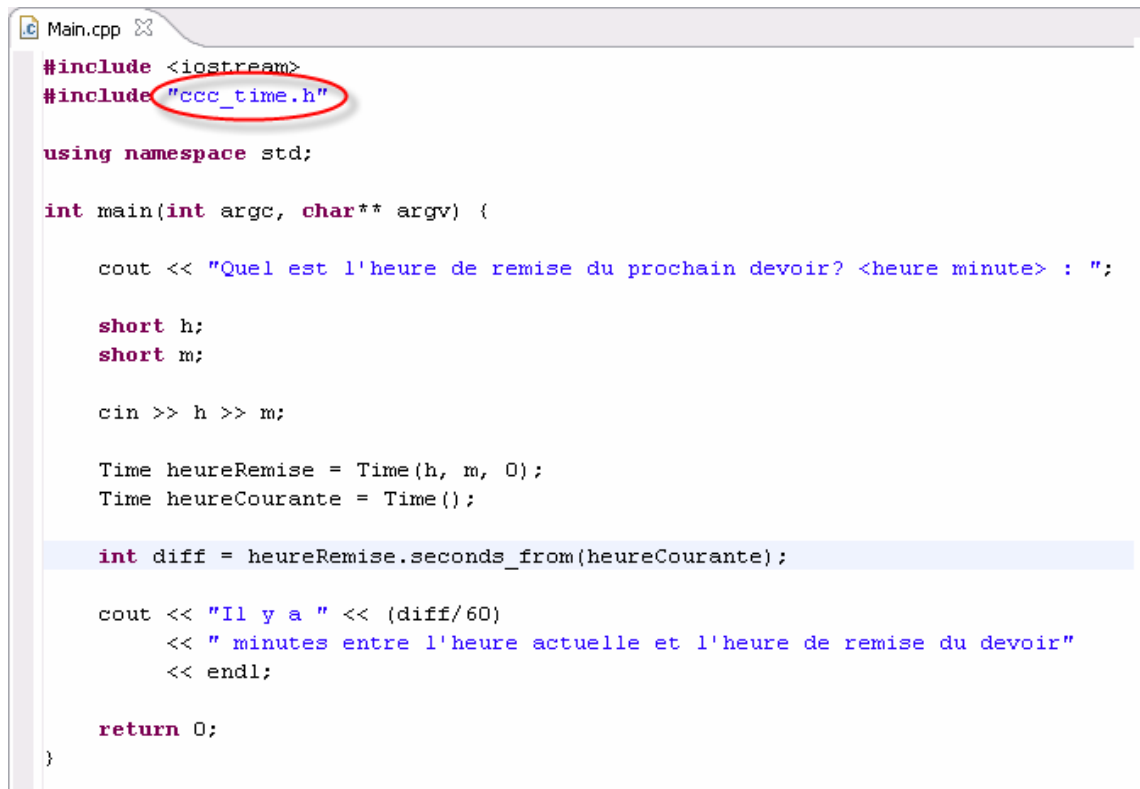
Décompressez ce fichier dans un répertoire qui ne fait pas partie de votre « workspace » Eclipse. Par exemple, vous pourriez décompresser le fichier `cccfiles.zip` dans le répertoire `C:\ccc3e`. Dans votre nouveau répertoire, vous devriez voir apparaître les répertoires suivants :



Pour le moment, on s'intéresse au répertoire `cccfiles`. Dans ce répertoire se trouve les classes utilisées dans l'ouvrage « La Bible C++ » ainsi que la bibliothèque graphique `CCC_WIN`. (Désolé pour la surutilisation du mot répertoire.)

La classe Time

Pour utiliser la classe `Time` on doit inclure le fichier d'en-tête `ccc_time.h` à même notre programme. Afin d'illustrer le tout, voici un exemple de solution du problème 3.1 :



```

Main.cpp
#include <iostream>
#include "ccc_time.h"

using namespace std;

int main(int argc, char** argv) {

    cout << "Quel est l'heure de remise du prochain devoir? <heure minute> : ";

    short h;
    short m;

    cin >> h >> m;

    Time heureRemise = Time(h, m, 0);
    Time heureCourante = Time();

    int diff = heureRemise.seconds_from(heureCourante);

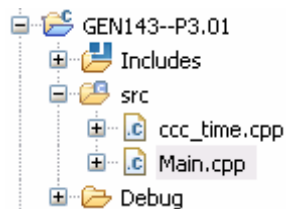
    cout << "Il y a " << (diff/60)
         << " minutes entre l'heure actuelle et l'heure de remise du devoir"
         << endl;

    return 0;
}

```

Notez l'inclusion du fichier d'en-tête `ccc_time.h`. Grâce à ce fichier, le compilateur comprend ce que vous voulez dire lorsque vous écrivez `Time`.

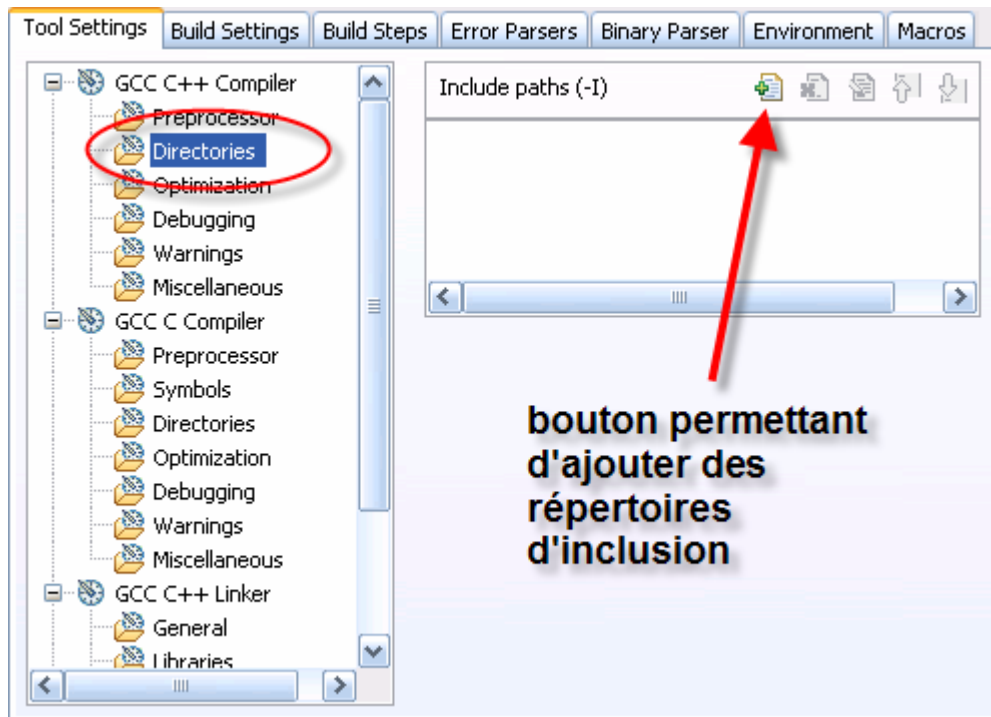
Il faut également inclure le fichier `ccc_time.cpp` dans le même dossier que votre programme. Par exemple, dans ce cas, le fichier de notre programme est `Main.cpp`; nous devons donc inclure le fichier `ccc_time.cpp` dans le même répertoire, tel qu'illustré :



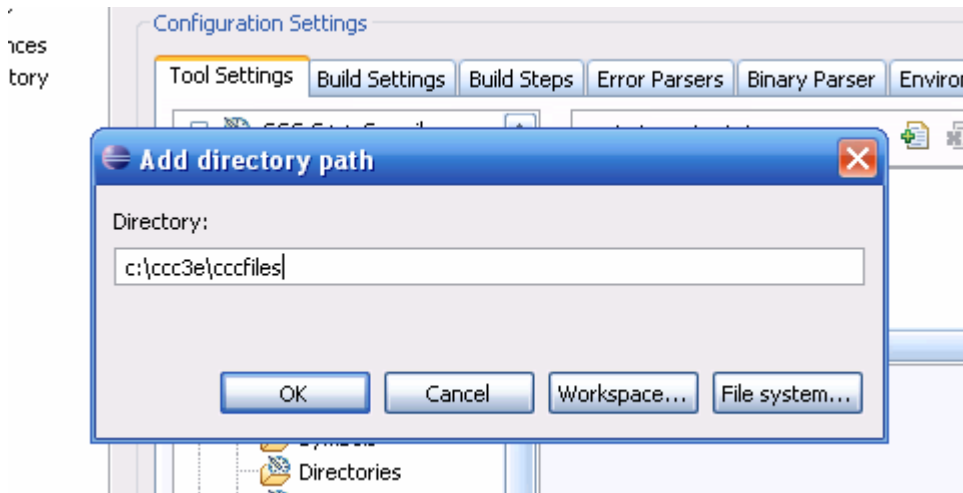
Se faisant, le compilateur va compiler `Main.cpp` ainsi que `ccc_time.cpp`. Le fichier d'en-tête `ccc_time.h` permet au compilateur de comprendre ce que vous entendez par

Time (le *quoi*). Le fichier code source `ccc_time.cpp` permet au compilateur d'accéder à une définition de Time (le *comment*).

Finalement, il faut indiquer au compilateur où trouver le fichier d'en-tête `ccc_time.h`. Vous savez qu'il se trouve, par exemple, dans `C:\ccc3e`. Le compilateur lui n'en sait rien. Pour l'indiquer au compilateur, vous devez ajouter un chemin d'inclusion de fichiers d'en-têtes. Voici comment faire : on clique sur le bouton droit de la souris sur le répertoire de notre projet. Ensuite, on sélectionne `Properties`. De là, on clique sur `C/C++ Build`. Dans la fenêtre de droite, on choisit `Tools Settings`. L'image qui suit parle d'elle-même.



On clique sur le bouton puis on entre le répertoire contenant les fichiers d'en-tête :



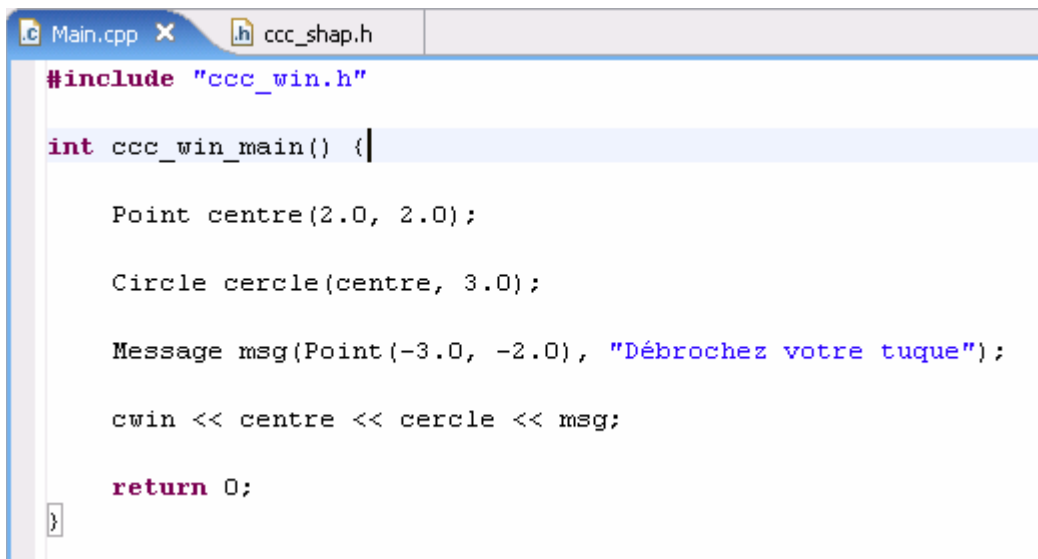
Voilà! Vous devriez être en mesure de compiler un programme qui fait usage de la classe `Time`.

La bibliothèque CCC_WIN

La bibliothèque graphique CCC_WIN vous permet de tracer toutes sortes de formes géométriques, d'afficher du texte et de saisir des entrées d'un utilisateur.

Afin de l'utiliser, on doit suivre certaines étapes.

Premièrement, on doit attacher sa tuque avec de la broche. Nous allons utiliser le petit programme bidon suivant comme plateforme à nos explications :



```
#include "ccc_win.h"

int ccc_win_main() {

    Point centre(2.0, 2.0);

    Circle cercle(centre, 3.0);

    Message msg(Point(-3.0, -2.0), "Débrochez votre tuque");

    cwin << centre << cercle << msg;

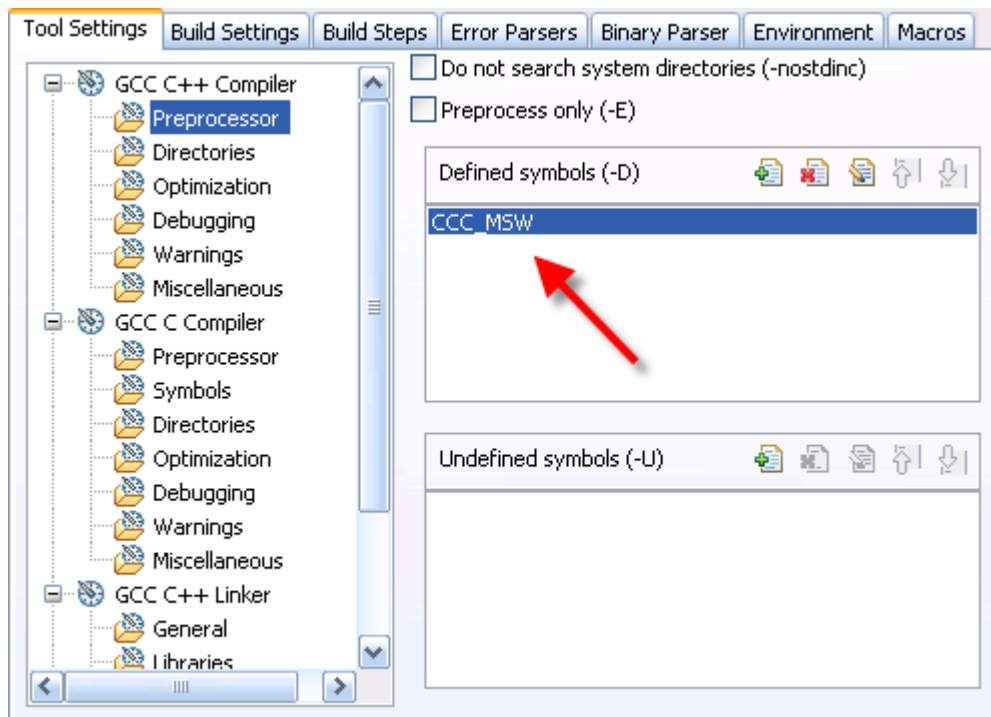
    return 0;
}
```

Deux choses très importantes à noter. Premièrement, on doit inclure le fichier d'en-tête `ccc_win.h`. Deuxièmement, on doit remplacer `main` par `ccc_win_main`.

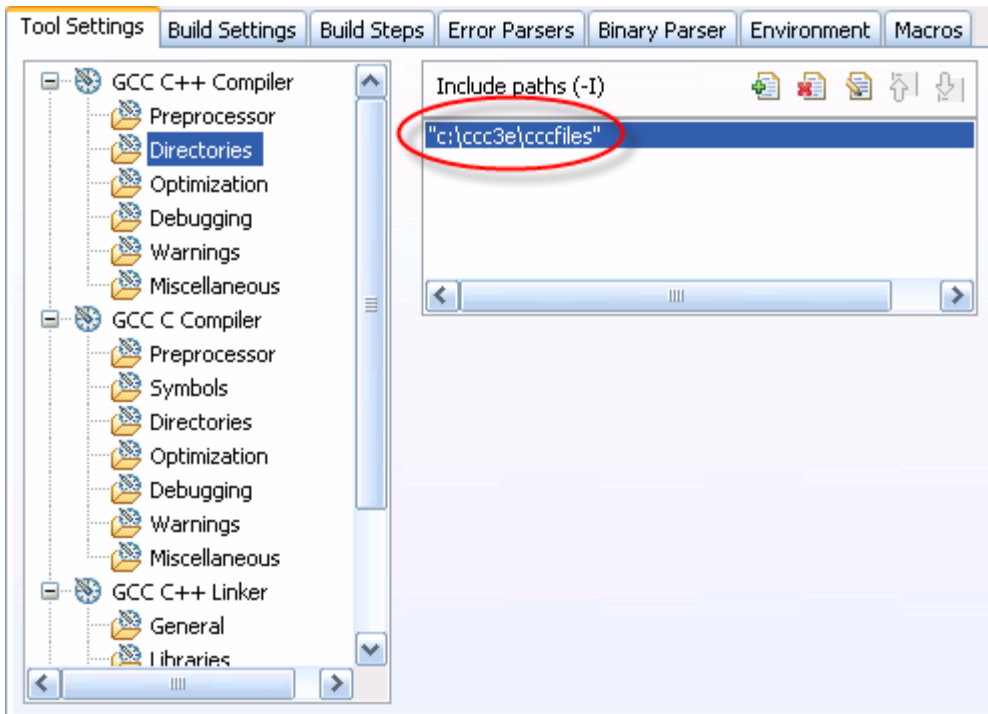
Ensuite, on doit configurer quelque peu le compilateur. On doit indiquer que l'on souhaite utiliser la version Win32 de la bibliothèque graphique CCC_WIN en définissant le symbole `CCC_MSW`. Ensuite, on doit demander au compilateur de lier (*linker* en bon français) la librairie `gdi32` avec notre programme. Cette librairie fournit les services graphiques utilisés par la version Win32 de la bibliothèque CCC_WIN. Finalement, il ne faut pas oublier d'ajouter le répertoire d'inclusion des fichiers d'en-têtes tout comme pour le cas de l'utilisation de la classe `Time`.

Comme une image vaut environ 1000 mots, voici quelque 4000 mots (oui, 4 images suivent).

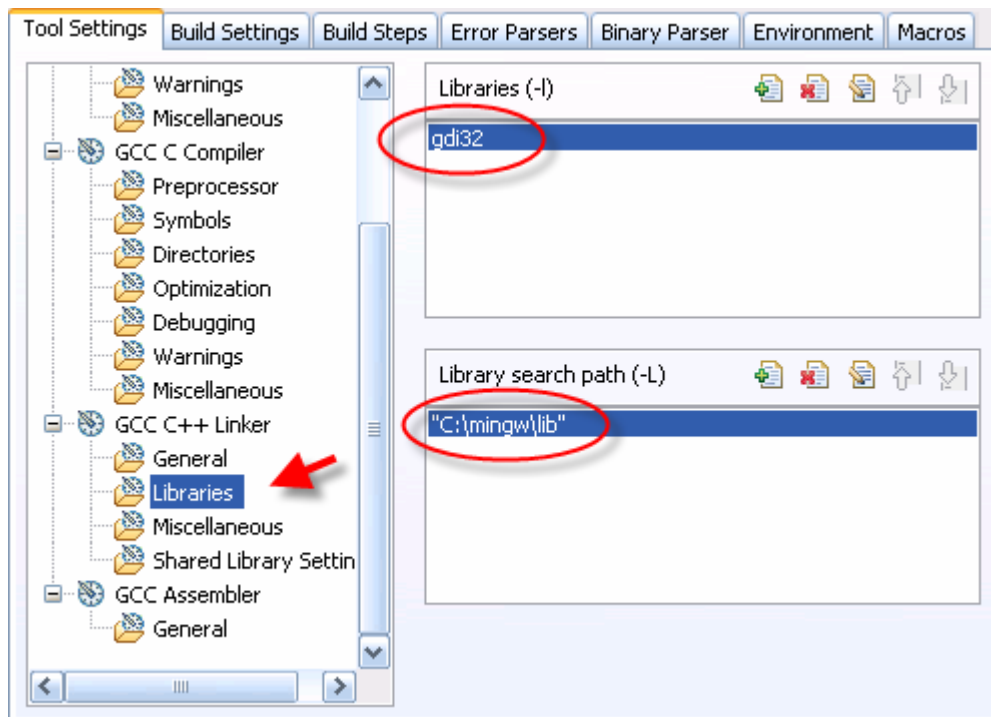
Définition du symbole CCC_MSW : on clique avec le bouton droit de la souris sur le dossier de notre projet. On sélectionne `Properties`. Dans la fenêtre de gauche, on clique sur `C/C++ Build`. Dans la fenêtre de droite se trouvent plusieurs onglets ; on choisit `Tools Settings`. L'image qui suit montre que dans la section `GCC C++ Compiler/Preprocessor`, l'on doit ajouter un symbole en cliquant sur le bouton avec un « plus » en vert et taper le nom de symbole `CCC_MSW`.



Ajout du répertoire d'inclusion des fichiers d'en-têtes : on clique avec le bouton droit de la souris sur le dossier de notre projet. On sélectionne `Properties`. Dans la fenêtre de gauche, on clique sur `C/C++ Build`. Dans la fenêtre de droite se trouvent plusieurs onglets ; on choisit `Tools Settings`. L'image qui suit montre que dans la section `GCC C++ Compiler/Directories`, l'on doit ajouter un chemin d'inclusion en cliquant sur le bouton avec un « plus » en vert et taper le chemin du répertoire où se trouve les fichiers d'en-têtes du livre.

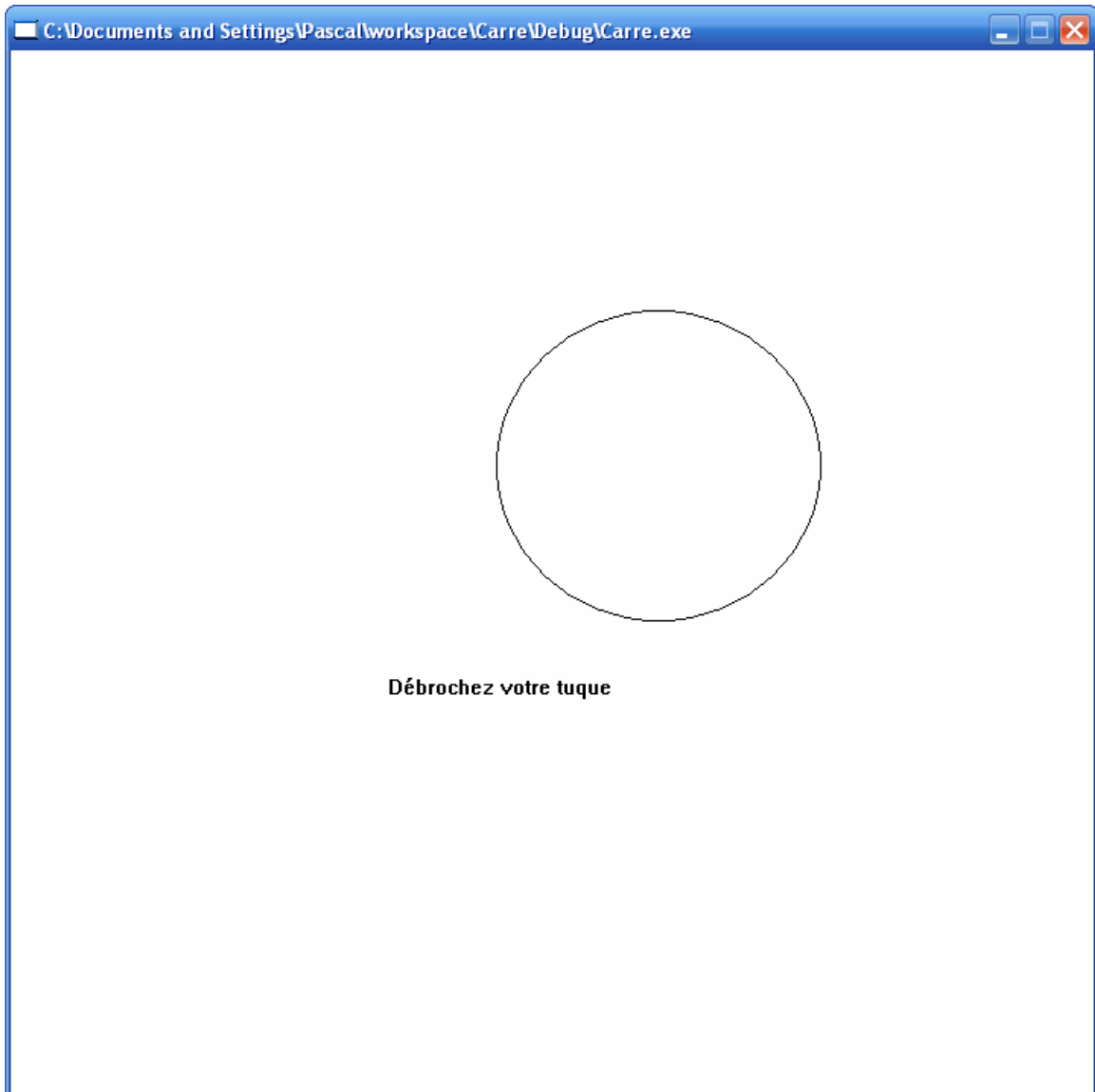


Indication au compilateur du besoin de se lier à la librairie `gdi32` : on clique avec le bouton droit de la souris sur le dossier de notre projet. On sélectionne `Properties`. Dans la fenêtre de gauche, on clique sur `C/C++ Build`. Dans la fenêtre de droite se trouvent plusieurs onglets ; on choisit `Tools Settings`. L'image qui suit montre que dans la section `GCC C++ Linker/Libraries`, l'on doit ajouter le nom de la librairie avec laquelle on doit lier notre programme ainsi que le chemin où se situe cette librairie. Non, nous n'allons pas, pour une troisième fois, répéter de quelle manière procéder pour ajouter ces valeurs.



Il ne faut pas oublier de copier les fichiers `ccc_msw.cpp` et `ccc_shap.cpp` dans le répertoire de votre programme (pour les mêmes raisons que vous aviez copié le fichier `ccc_time.cpp` dans le répertoire de votre programme faisant utilisation de la classe `Time`).

Si tout se passe bien, l'exécution de votre programme devrait générer une fenêtre avec le contenu qui suit :



Récapitulation

Pour utiliser une classe du livre, il faut ajouter le fichier de définition de la classe (`.cpp`) dans le même répertoire que votre programme et s'assurer que son fichier d'en-tête correspondant (`.h`) est bel et bien dans un répertoire d'inclusion. Il faut également penser d'ajouter la directive `#include` au haut de votre programme.

Afin d'utiliser la bibliothèque graphique `CCC_WIN`, il faut indiquer au compilateur quelle version de la bibliothèque l'on désire utiliser en définissant le symbole approprié (p.ex. : `CCC_MSW`). On doit ajouter les fichiers `ccc_msw.cpp` (si on a utilisé le symbole `CCC_MSW`) et `ccc_shap.cpp` dans le répertoire de notre programme. Les fichiers `ccc_msw.h` (si on a utilisé le symbole `CCC_MSW`) et `ccc_shap.h` doivent se trouver dans un répertoire d'inclusion. On doit ajouter la directive `#include` appropriée. Finalement, il faut indiquer au compilateur que l'on désire lier notre programme à la librairie `gdi32`.

Sans trop vous en rendre compte, vous commencez à écrire des programmes plus élaborés. N'oubliez pas que c'est en programmant que l'on apprend, alors n'hésitez pas à faire TOUS les exercices proposés.