UMBB, Faculté des Sciences, Département d'informatique

MI, Première année, Semestre1 Matière : Algo et Structures de Données Statiques

# TP N° 2

# Création de projets sous Code::Blocks

**But du TP :**

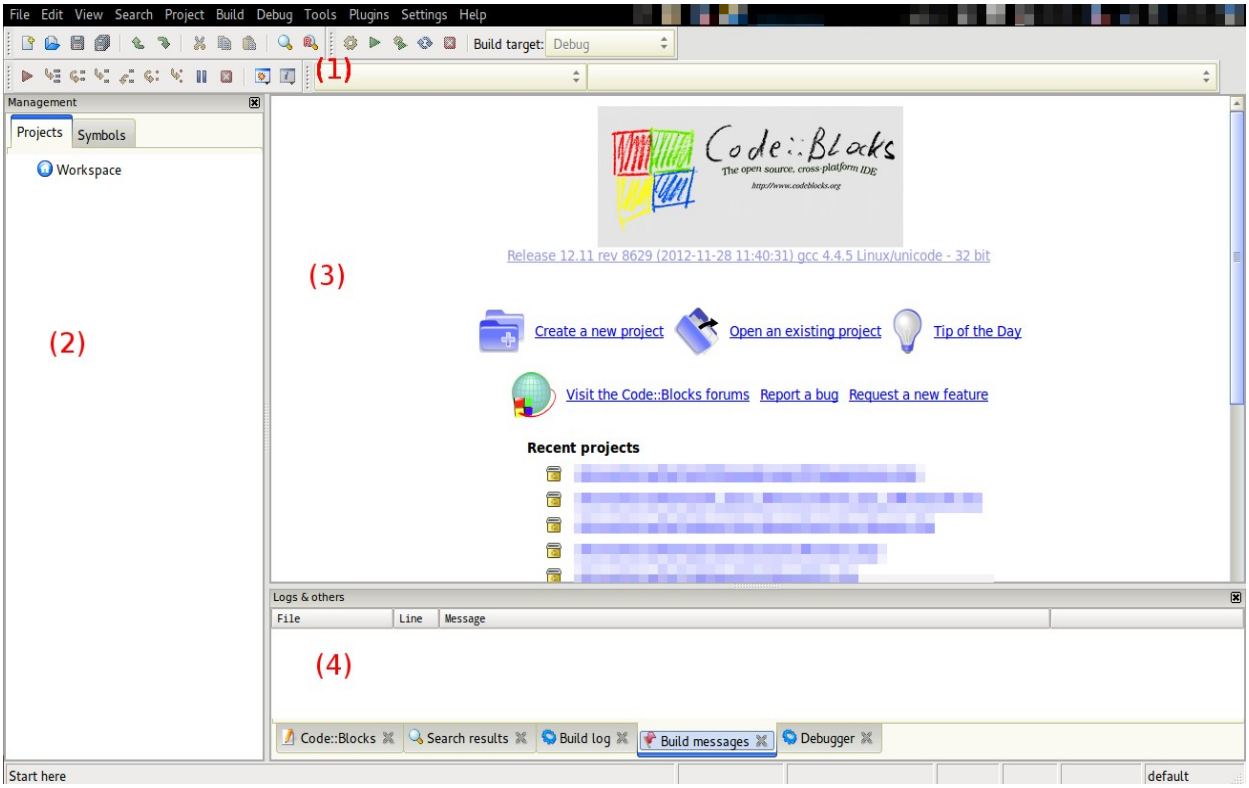
Apprendre quelques notions de base liées à la création de projets sous Codeblocks + Instructions alter.

**Énoncé :**

Codeblocks est un environnement de développement intégré i.e. il regroupe les outils de prise en charge des programmes utilisateur i.e. éditeur de texte, compilateur et éditeur de liens. Sous cet IDE, pour exécuter un programme, il faut créer un projet. Dans ce TP, il est demandé de reproduire les étapes suivantes (Pour Windows, télécharger puis installer codeblocks-20.03mingw-setup.exe ou une version plus ancienne depuis https://www.codeblocks.org/downloads/binaries/; Pour Ubuntu, suivre les directives du lien https://linuxhint.com/install-code-blocks-ubuntu/) :

## 1. COMPOSANTE DE LA FENETRE PRINCIPALE

Lancer Codeblocks. Une fenêtre va s'ouvrir où on distingue 4 grandes sections, numérotées sur la figure qui suit :



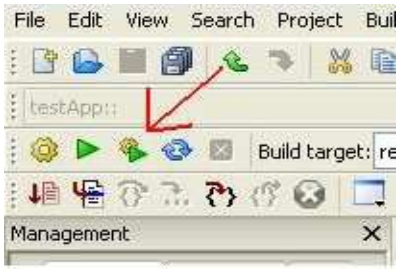

**(1) barre d'outils** : comprend des boutons qui nous permettrons de lancer certaines commandes sans passer par le menu;

**(2) liste des fichiers du projet** : la liste de tous les fichiers source de notre programme. Notez que sur cette capture aucun projet n'a été créé, donc la liste est vide ;

**(3) zone principale** : c'est là que nous écrirons notre code en langage C ;

**(4) zone de notification** : c'est ici que nous verrons les erreurs de compilation s'afficher si notre code comporte des erreurs.

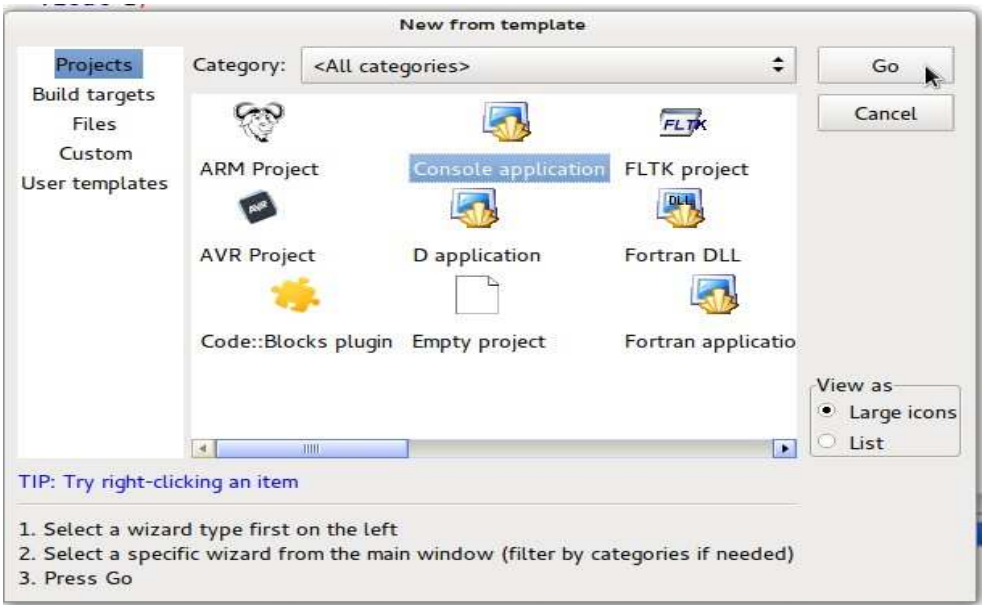
Intéressons-nous maintenant à une section particulière de la barre d'outils. Vous y trouverez les boutons suivants (dans l'ordre) : Build, Run, Build & Run que nous utiliserons régulièrement.



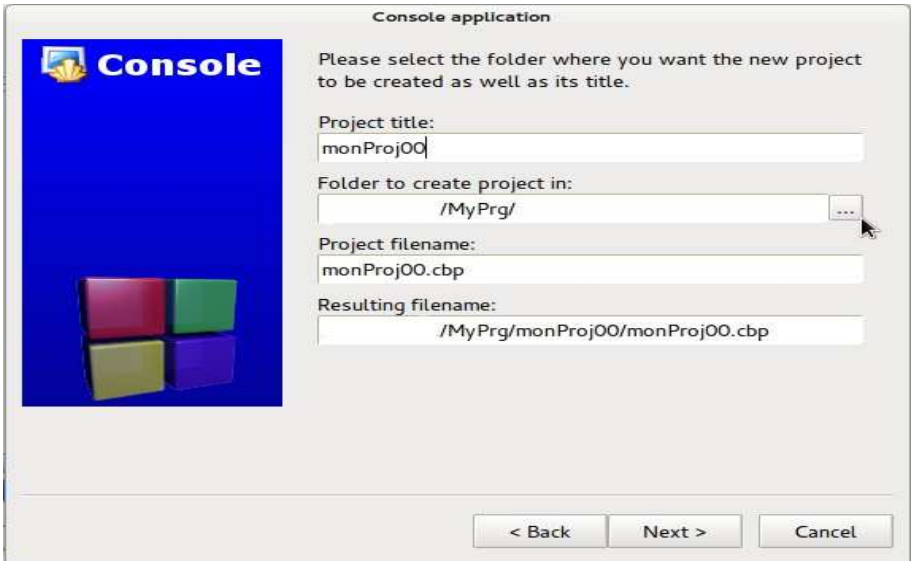

- **Build** : le code source est envoyé au compilateur qui va se charger de créer un exécutable (compilation+édition de liens). S'il y a des erreurs, l'exécutable ne sera pas créé et on vous indiquera les erreurs en bas de Code::Blocks ;
- **Run** : cette icône lance le dernier exécutable (attention il peut être différent de celui que vous voyez sur la fenêtre d'édition) que vous avez compilé. Cela vous permettra de tester votre programme et de voir ainsi ce qu'il donne. Dans l'ordre, si vous avez bien suivi, on doit d'abord compiler, puis exécuter pour tester ce que ça donne. On peut aussi utiliser le troisième bouton…
- **Build & Run** : c'est la combinaison des deux boutons précédents. C'est ce bouton que vous utiliserez le plus souvent (en première année MI). Notez que s'il y a des erreurs pendant la compilation (pendant la génération de l'exécutable), le programme ne sera pas exécuté car il faut les corriger toutes.
- **Rebuild** : quand vous faites `Build`, Code::Blocks ne recompile en fait que les fichiers que vous avez modifiés et non les autres. Parfois, vous aurez besoin de demander à Code::Blocks de vous recompiler tous les fichiers. Ce bouton ne nous sera pas utile en 1ière MI car il intervient dans les projets avec plusieurs fichiers.

## 2. CRÉER UN NOUVEAU PROJET

Créez un nouveau projet : Pour cela, allez dans le menu `File / New / Project`.
Dans la fenêtre qui s'ouvre, choisissez `Console application`.



Cliquez sur `Go` pour créer le projet. Un assistant s'ouvre.

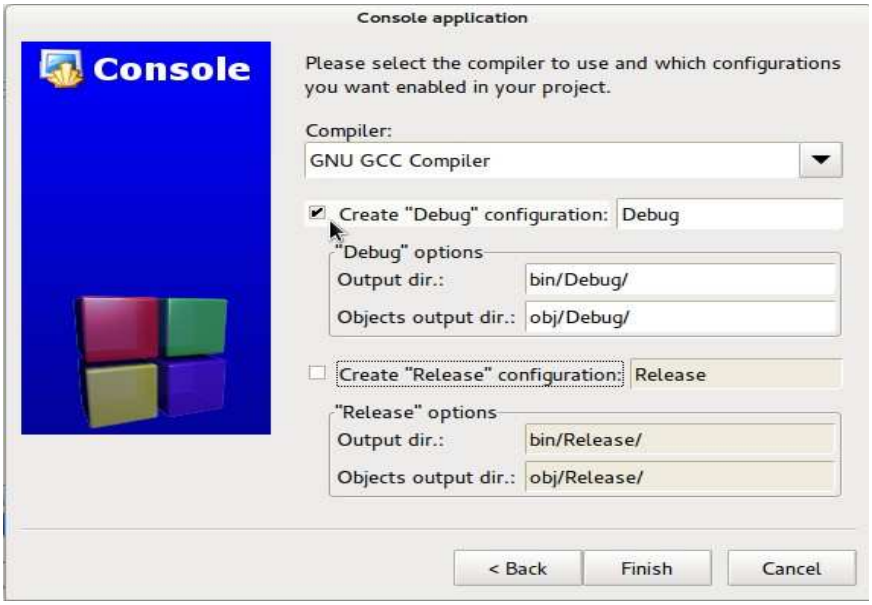



On vous demande ensuite si vous allez faire du C ou du C++ : répondez « C » (car C et C++ sont deux langages différents).

On vous demande le nom de votre projet et dans quel dossier les fichiers source seront enregistrés. Il faut *impérativement* choisir un nom de fichier (chemin inclus) sans espaces sinon CodeBlocks sera instable (il faut que vous ayez les droits d'accès à l'emplacement de votre fichier).


Après avoir choisi un répertoire pour y mettre les fichiers de votre projet, cliquez `Next`.

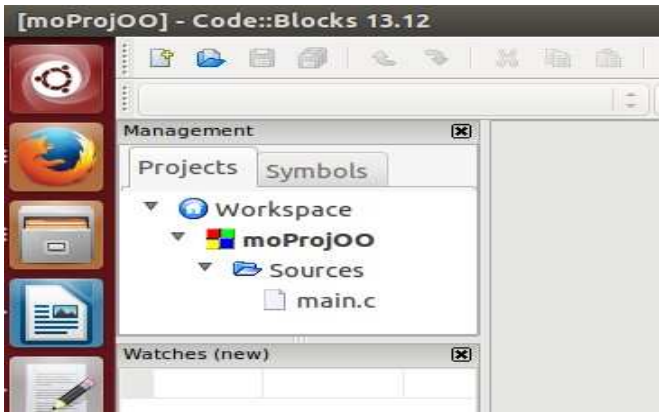
Enfin, la dernière page vous permet de choisir de quelle façon le programme doit être compilé. Vous pouvez laisser les options par défaut, mais veillez à ce que la case `Debug` au moins soit cochée.



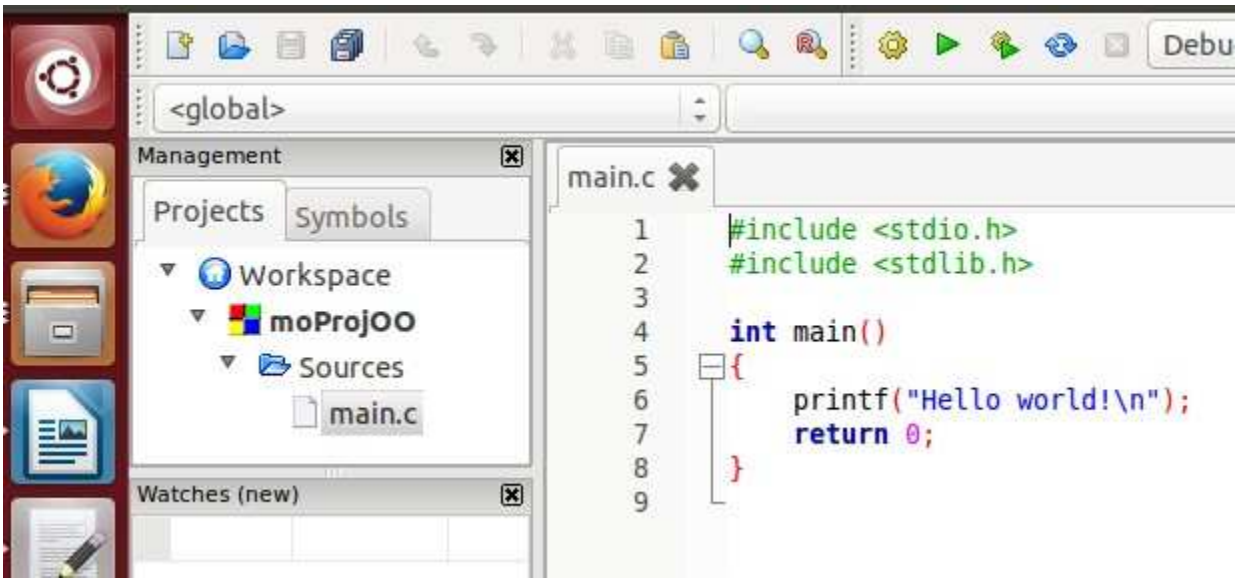
En cliquant sur `Finish`, Code::Blocks créera un premier projet avec un tout petit peu de code source dedans.

### 3. EXECUTION DE NOTRE PROGRAMME

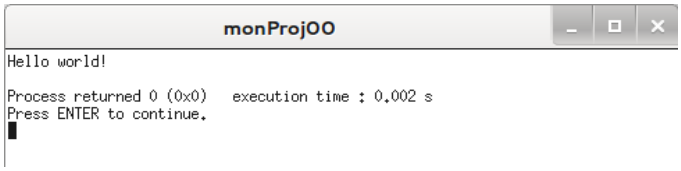
Dans le cadre de gauche « Projects », développez l'arborescence en cliquant sur le petit « + » (ou le triangle ça dépend des versions) pour afficher la liste des fichiers du projet.


Vous devriez avoir au moins un `main.c` (dans Sources) que vous pourrez ouvrir en double-cliquant dessus (si au lieu de main.c vous avez main.cpp ceci veut dire que vous n'avez pas bien suivi les étapes. Alors ...).



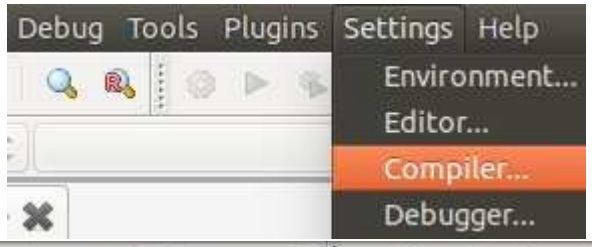

Il suffit de l'exécuter pour avoir une fenêtre en mode console vous présentant un message Hello world !. Tappez 'entrer' et vous allez retourner a votre fenêtre pour éditer/modifier votre code source.

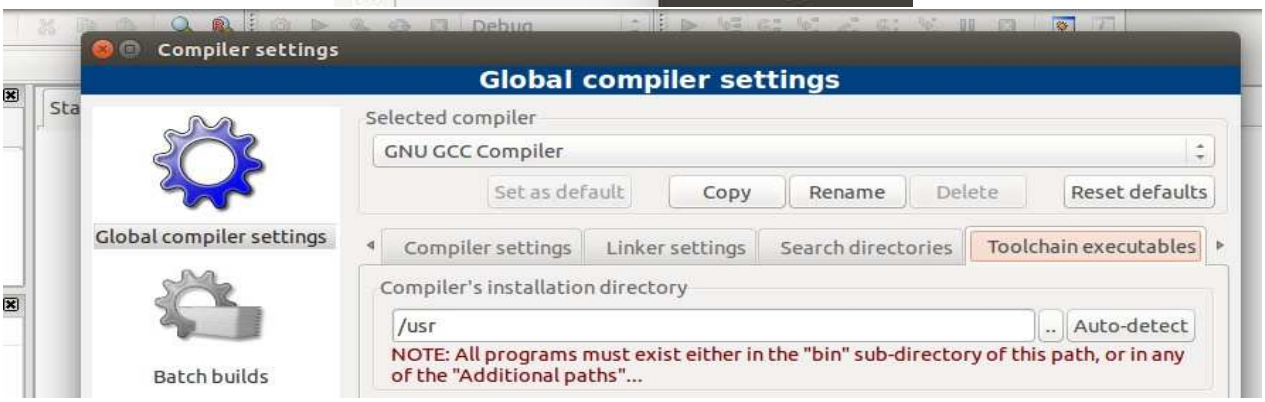


Si au lieu de ça vous avez des messages comme :


Ceci veut dire que Codeblocks ne parvient pas à trouver le compilateur C. Il faut donc :

- s'assurer que vous avez bien téléchargé la version de Codeblocks incluant un compilateur C (pour Windows, c'est celle qui contient **mingw**). Et que
- le chemin de ce compilateur est mentionné dans la fenêtre Global compiler settings qui peut être affichée depuis le menu **Settings → Compiler...** ensuite l'onglet **Toolchain executables** :




Si malgré cette explication vous êtes tjrs bloqués, googlez « codeblocks missing compiler », il y a des vidéos qui vous expliquerons comment résoudre ce problème.

**4. QUELQUES PROGRAMMES À RÉALISER :**

- Écrire un programme qui demande à l'utilisateur un entier puis lui affiche son carré.
- Écrire un programme qui permet de résoudre une équation de la forme : a $x$ + b = 0 pour tous les cas possibles des réels a et b.
- Écrire un programme qui permet de résoudre une équation du second degré.
- En utilisant **switch**, écrire un programme qui demande à l'utilisateur deux nombres réels puis affiche un menu proposant de réaliser l'une des 4 opérations arithmétiques de base. Il affichera par la suite le résultat de l'opération choisie. Pour la division par zéro se programme doit afficher un message d'erreur au lieu de réaliser l'opération.
- Pour bien comprendre le comportement de switch, analyser le programme suivant, déduire ce qu'il fait et pourquoi il le fait convenablement :

```c
#include <stdio.h>
int mois;
int main(){
    printf("Bonjour, donnez un numéro de mois "
           " je vais vous dire combien de jours il contient:");
    scanf("%d",&mois);
    switch(mois){
        case 1: case 3: case 5: case 7: case 8: case 10: case 12:
                printf("Votre mois contient 31 jours"); break;
        case 4: case 6: case 9: case 11:
                printf("Votre mois contient 30 jours"); break;
        case 2: printf("Votre mois contient 28 ou 29 jours"); break;
        default: printf("Le nombre %d n'est pas un numéro de mois valide!",mois);
    }
    return 0;
}
```

Commenter les **break** (i.e. //break ; voir la capture en bas) un par un puis tous ensemble, chaque fois sauvegarder et ré-executer le programme avec différentes entrées. Déduire.