Introduction à `Code::Blocks`

1 Prise en main de l'environnement de travail `Code::Blocks`

1.1 Ajout d'un nouveau projet

Pour ajouter un nouveau projet à votre environnement de travail (*workspace*) suivez les étapes suivantes :

- Dans le menu de `Code::Blocks` choisissez File › New › Project (fig. 1).
- Sélectionnez le projet du type "application de console" (fig. 2).
- Choisissez C++, le langage du projet (fig. 3).
- Donnez un nom à votre projet et choisissez l'emplacement du répertoire (fig. 4).
- Dans les fenêtres de dialogue suivantes ne modifiez rien.
- Terminer en cliquant sur Finish (fig. 5).

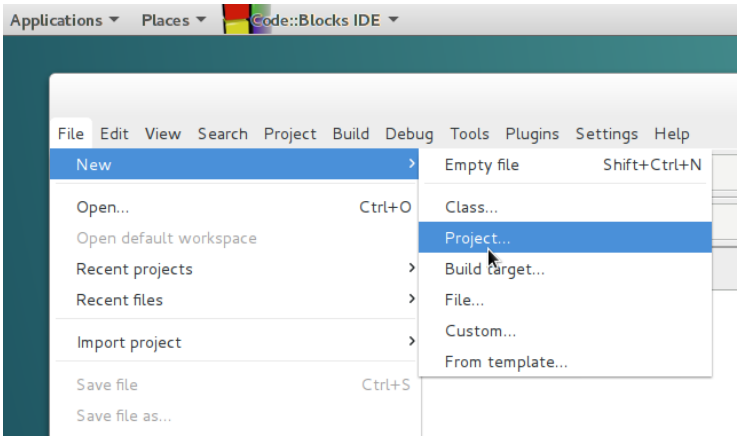

FIGURE 1 – Pour insérer le premier projet dans un *workspace* vide, choisissez File › New › Project.

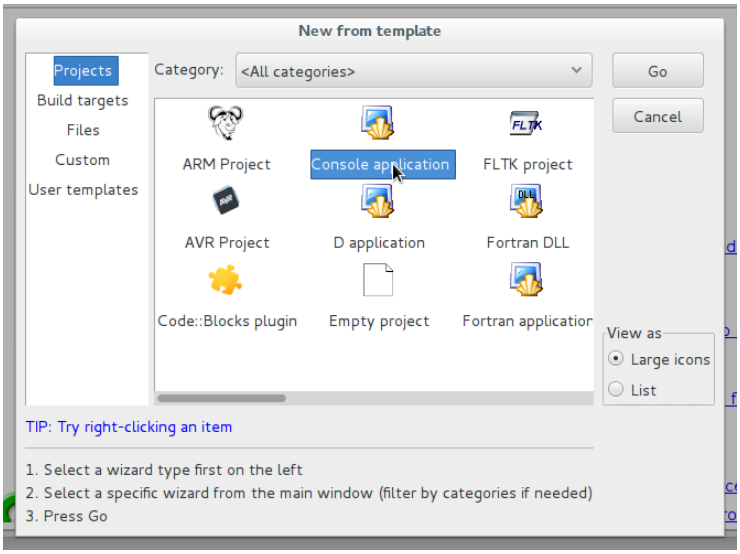

FIGURE 2 – Dans la liste des types de projets proposés, sélectionnez “Console application”.

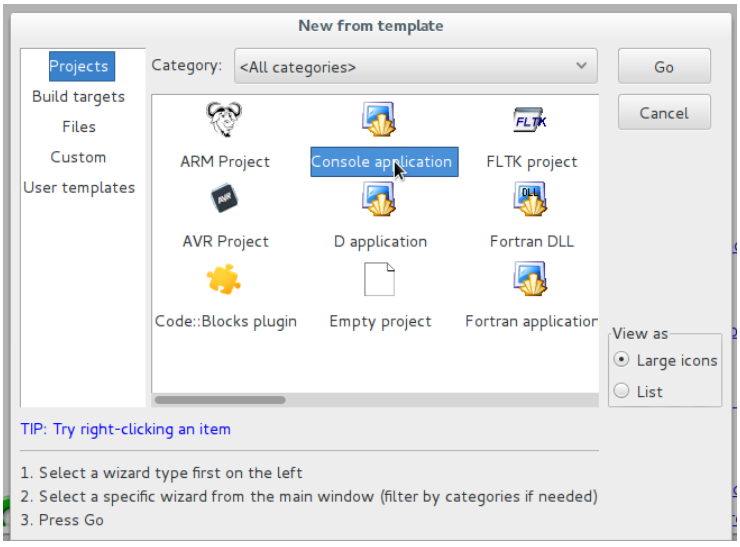

FIGURE 3 – Sélectionnez le langage du projet : C++ .

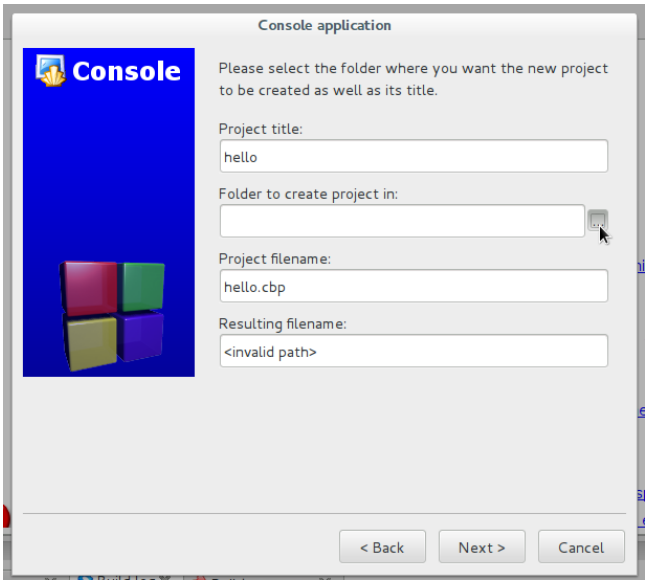

FIGURE 4 – Donner un nom à votre projet et sélectionnez son emplacement (*Folder to create project in*).

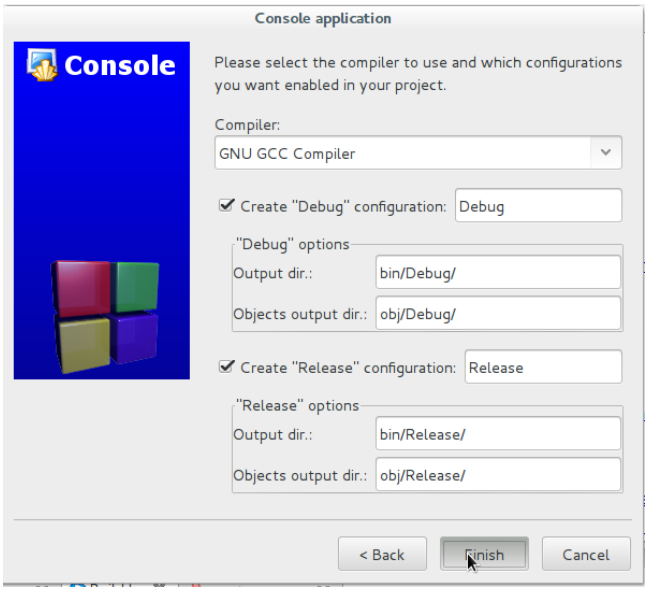

FIGURE 5 – Terminer la création d'un projet en cliquant sur Finish.

1.2 Ajout d'un projet déjà existant

Un espace de travail de `Code::Blocks` est juste une collection de projets, regroupée par un fichier XML avec extension `*.workspace`. Les projets peuvent se trouver n'importe où dans le système des fichiers, mais il est préférable de choisir un seul répertoire pour les projets de même catégorie. Pour ajouter un projet existant à un espace de travail, suivez les étapes suivantes :

— Copiez le répertoire du projet dans le répertoire qui regroupe les projets de l'espace de travail.

— Dans le menu de `Code::Blocks` choisissez File ⟩ Open et ouvrez le fichier de description du projet (c'est le fichier avec l'extension `*.cbp`), voir fig. 6.

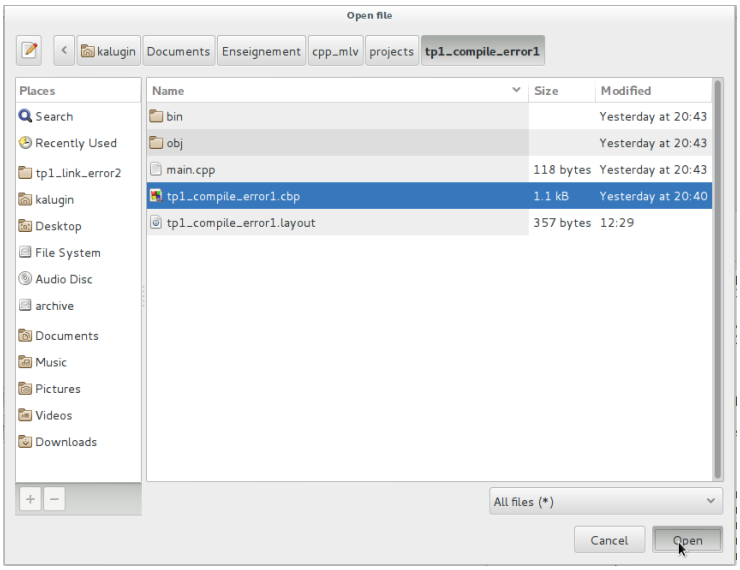

FIGURE 6 – Pour ajouter un projet existant à l'espace de travail, allez dans le menu File ⟩ Open et ouvrez le ficher de description du projet en cliquant sur Open.

1.3 Sélection d'un projet actif

S'il y a plusieurs projets dans un espace de travail, un seul projet est déclaré actif (son nom apparaît en gras dans la liste des projets, voir fig. 7). C'est ce projet qui est généré, exécuté ou débogué par défaut. Pour sélectionner un projet actif, il suffit de faire un double clic sur son nom dans la liste des projets.

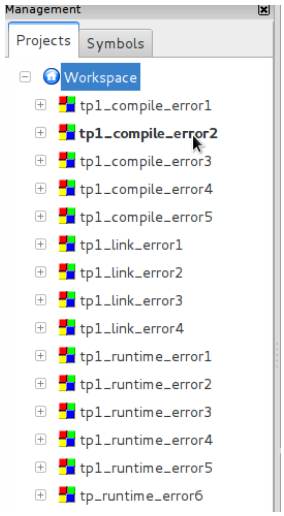

FIGURE 7 – Pour sélectionner un projet actif, faites un double clic sur son nom.

1.4 Sauvegarde de l'espace de travail

Un fichier de description de l'espace de travail peut être sauvegardé n'importe où, mais il est recommandé de le placer dans le répertoire qui contient les sous-répertoires de projets. Pour sauvegarder un espace de travail la première fois, allez dans le menu File ⟩ Save workspace as..., puis donnez un nom à votre espace de travail et appuyez sur Save (voir fig. 8)

1.5 Ouverture d'un espace de travail sauvegardé

Si vous avez un espace de travail ouvert, fermez-le (File ⟩ Close workspace). Puis avec File ⟩ Open ouvrez une fenêtre de dialogue et sélectionnez le fichier `*.workspace` voulu.

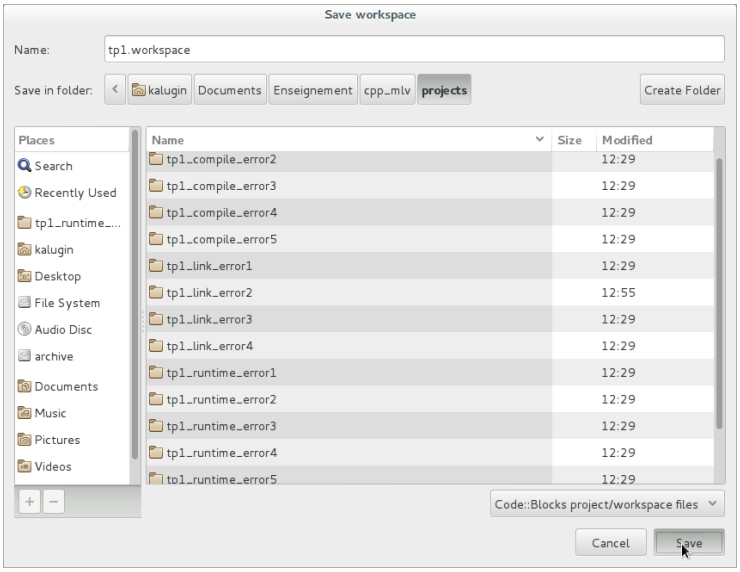

FIGURE 8 – Pour sauvegarder un espace de travail la première fois, allez dans le menu File » Save workspace as..., donnez un nom à l'espace de travail et cliquez sur Save.

2 Projet à plusieurs unités de compilation

Dans les langages C et C++ une unité de compilation est le synonyme d'un fichier source, c'est à dire un fichier ayant extension `*.c` ou `*.cpp` (ou plus précisément le résultat de *preprocessing* d'un tel fichier qui a comme effet l'inclusion du contenu des fichiers d'en-tête mentionnés avec la commande `#include`).

2.1 Ajout d'un nouveau fichier à un projet existant

Pour ajouter un nouveau fichier à un projet existant, sélectionnnez ce projet comme actif (voir la section 1.3), puis allez dans File › New › File... . Dans la fenêtre de dialogue choisissez le type approprié (soit "C/C++ header", soit "C/C++ source"), et cliquez sur Go. Pour un fichier source on vous demandera ensuite de préciser le langage (C ou C++). Ensuite, donnez un nom au nouveau fichier (avec le chemin d'accès complet, il est conseillé de placer les fichiers du projet dans le meme repertoire où se trouve le fichier `*.cbp`). Finalement, n'oubliez pas de cocher les cases "*Add file to active project in build target(s)*" et appuyez sur Save (voir fig. 9)

2.2 Ajout d'un fichier déjà existant

On commence par placer le fichier dans un endroit approprié dans l'arborescence des fichiers. Un projet `Code::Blocks` peut être composé des fichiers se trouvant dans les répertoires différents, mais dans la pratique il est conseillé de placer tous les fichiers du projet a l'endroit où se trouve son fichier de description (`*.cbp`). Ensuite, dans le menu contextuel du projet, sélectionnez Add files... (voir fig. 10), choisissez un fichier et appuyez sur Open.

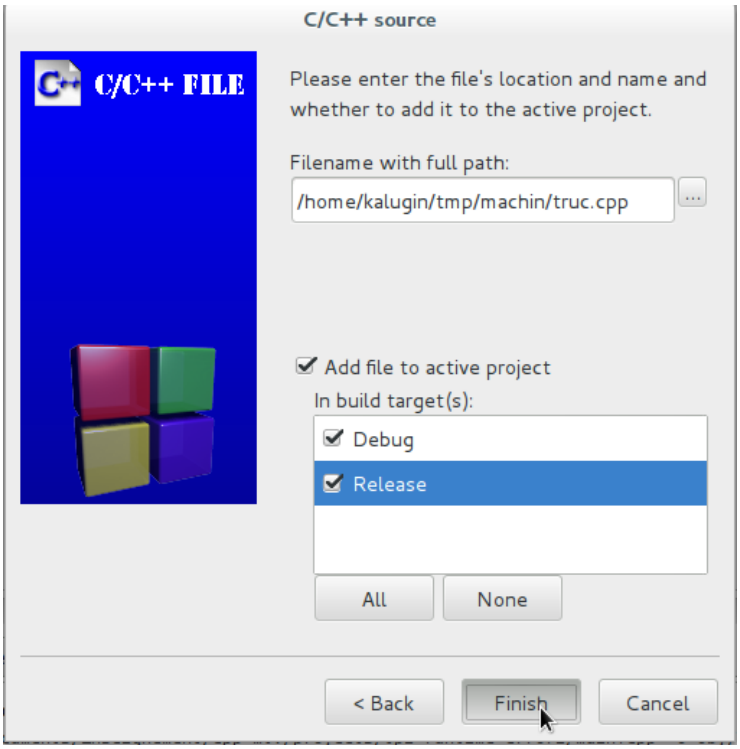

FIGURE 9 – Donnez un nom au nouveau fichier et précisez le chemin d'accès complet. S'il s'agit d'un fichier source ou d'en-tête, n'oubliez pas de l'inclure dans les dépendances du projet en cochant les cases correspondantes.

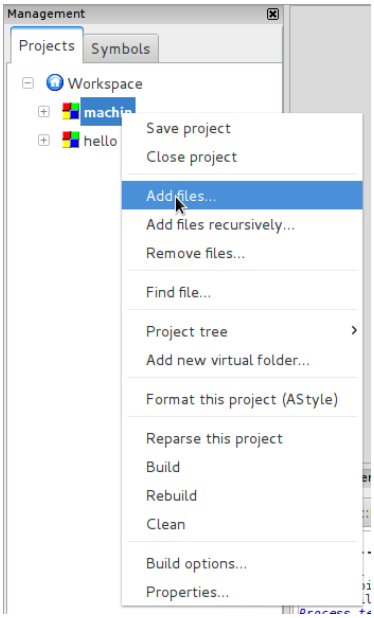

FIGURE 10 – Pour ajouter un fichier existant a un projet, sélectionnez Add files... dans son menu contextuel.

3 Génération du projet

Pour générer un projet, utilisez le menu Build » Build ou bien la combinaison de touches Ctrl+F9 (Attention, si vous avez une fenêtre de terminal active avec un de vos programmes, ce choix sera inactif!).

4 Exécution et débogage

NB : si l'espace de travail contient plusieurs projets, les commandes d'exécution ou de débogage concernent uniquement le projet actif (voir la section 1.3).

4.1 Exécution normale

Pour lancer l'exécution ordinaire, utilisez le menu Build » Run ou la combinaison de touches Ctrl+F10.

4.2 Débogage

Le plus souvent, avant de lancer le débogage on a l'intérêt de définir un point d'arrêt à l'endroit suspect du programme. Pour le faire, cliquez avec le bouton droit sur la zone grise á gauche du texte du programme, et sélectionnez Add breakpoint dans le menu contextuel (voir fig. 11).

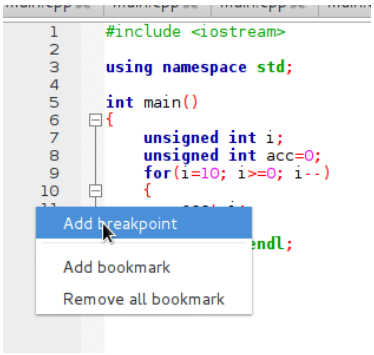

FIGURE 11 – Pour ajouter un point d'arrêt, cliquez avec le bouton droit dans la zone grise á gauche du texte du programme au niveau de la ligne visée, puis choisissez Add breakpoint dans le menu contextuel.

Pour lancer le débogage, allez vers Debug » Start/Continue ou bien utilisez la touche F8. Une fois que votre programme s'arrête sur un point d'arrêt, vous pouvez ouvrir des outils de débogage tel *Watches* ou *Call stack*.

4.2.1 Watches

Cet outil de débogage vous permet de visualiser l'état des variables de votre programme. Pour activer l'outil, utilisez le menu Debug ⟩ Debugging windows ⟩ Watches (voir fig. 12)

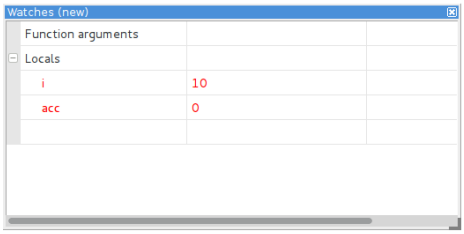

FIGURE 12 – La fenêtre *Watches* permet de visualiser les variables existantes dans le contexte du programme. Pour ouvrir cette fenêtre, utilisez le menu Debug ⟩ Debugging windows ⟩ Watches

4.2.2 Avancement pas-à-pas

Pour executer le programme dans le mode pas-à-pas, utilisez les boutons de la barre d'outils ou les raccourcis clavier correspondants (voir fig.). Le mode le plus fréquemment utilisé est celui de *next instruction*, correspondant à la combinaison de touches Alt + F7.

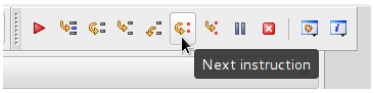

FIGURE 13 – Les boutons de la barre d'outils contrôlant l'exécution pas-à-pas.

4.2.3 Call stack

Cet outil permet de visualiser la pile d'appels et aussi de se placer dans le contexte d'exécution de chaque fonction appelée, pour visualiser les variables locales avec l'outil "*Watches*". Pour activer l'outil, utilisez le menu Debug ⟩ Debugging windows ⟩ Call Stack (voir fig. 14)

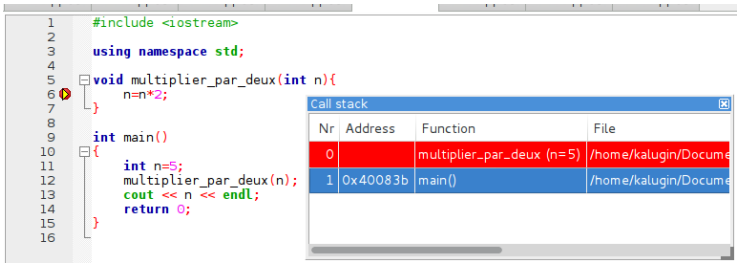

FIGURE 14 – La pile d'appels. On peut se placer le contexte d'exécution de chaque fonction dans la pile.