Tutoriel code::blocks

E. Lunéville

2006

Le logiciel **code::blocks** fait partie des logiciels de type EDI (Environnement de Développement Intégré, IDE en anglais) pour le langage C++. Il est multiplateforme en particulier Windows et Linux. Il propose dans une même fenêtre :

- la gestion d'un projet C++ avec le suivi des fichiers d'entête et d'implémentation
- la prise en compte de différents modèles de projet : console, graphique (winapi, qt, wxwidget)
- un éditeur avec mis en évidence de la syntaxe C++ , complétion de code et possibilité de le reconfigurer
- un outil de mise en forme du code C++
- un outil de visualisation et d'accès rapide aux différents composants du projet (fonction, classe, membre, enum, ...)
- l'appel intégré à différents compilateurs (par défaut le compilateur GNU) avec possibilité de modifier les options de compilation et d'exporter le makefile
- un debugger intégré permettant de tracer les variables
- un outil de profiling
- une zone pour saisir des actions à faire (to do list)
- des outils d'export du code (HTML, RTF, ODT)
- la possibilité de rajouter de nouvelles fonctionnalités plug-in

C'est un produit assez jeune sous licence GPL qui s'adresse plutôt au développeur individuel. Il n'y a pas pour le moment d'outil permettant de gérer les versions.

1 Installation

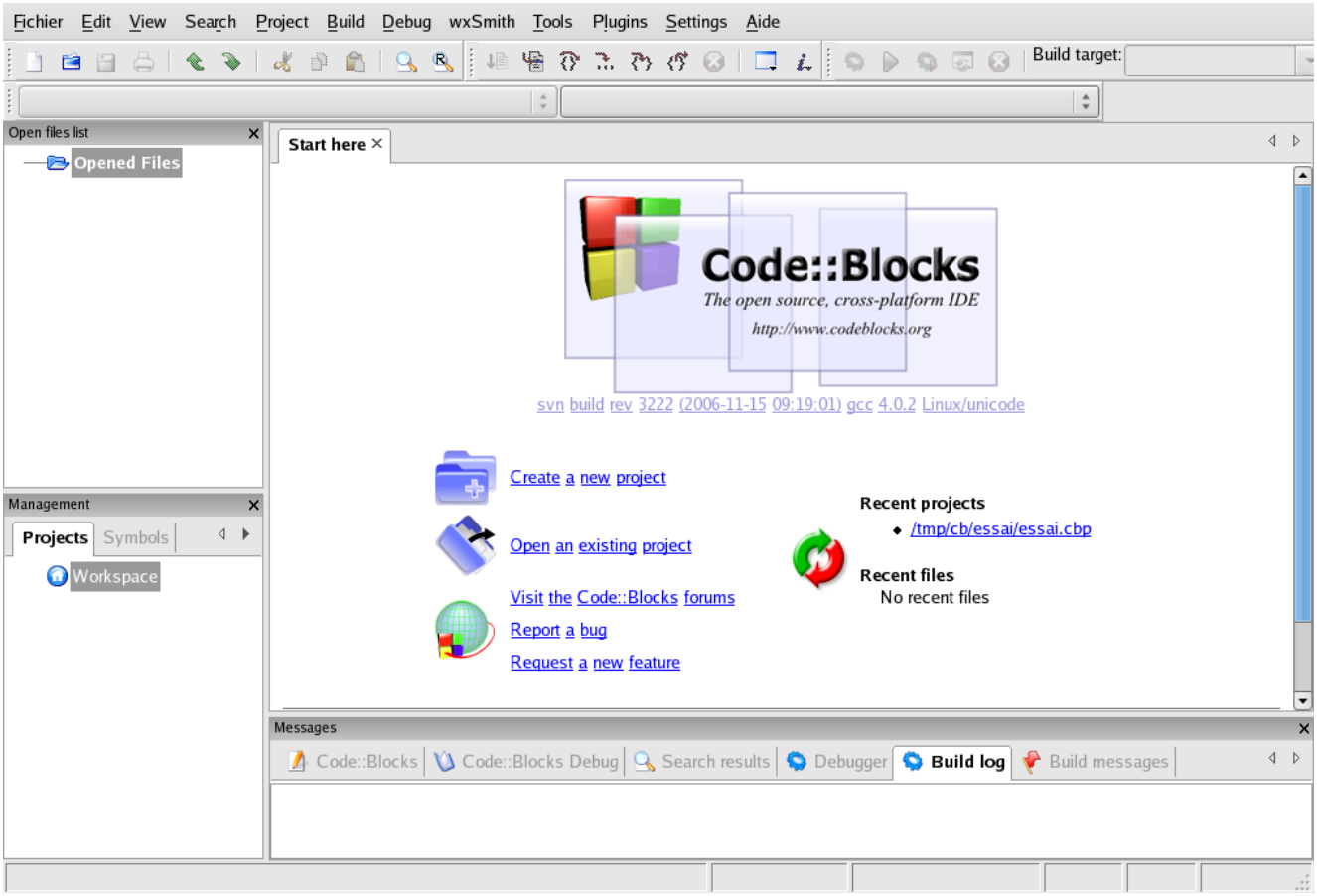
Suivant la plateforme et le type de distribution l'installation diffère. On trouve les différentes distributions de **code::blocks** à l'adresse suivante (dernière version stable en novembre 2006 : 1.0 RC2) :

http://wiki.codeblocks.org/index.php?title=Compiled_packages_of_Code::Blocks

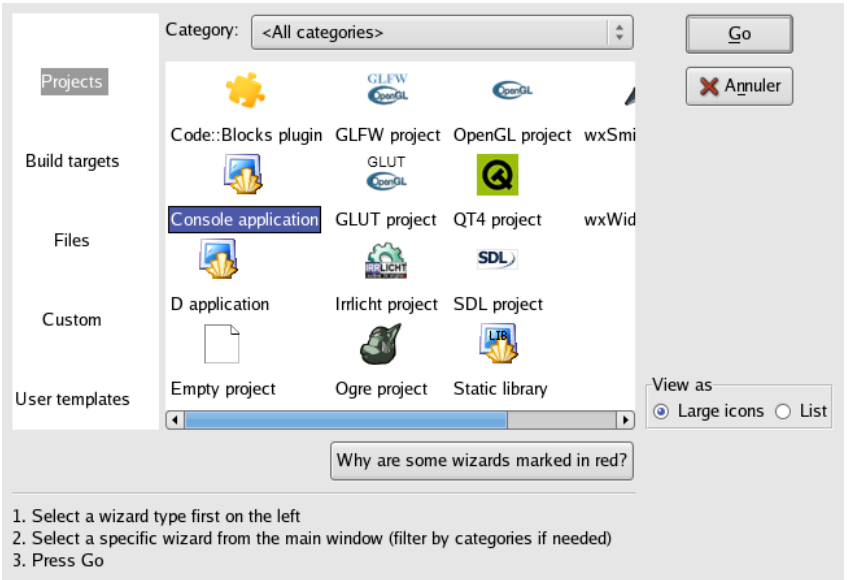
Ils existenr des versions dites SVN, qui sont les dernières versions. Depuis la version 1.0 RC2, de nombreuses améliorations ont été introduites et l'interface a beaucoup évolué. C'est pourquoi il est préférable d'utiliser ces versions en attendant la prochaine release candidate (RC3).
Il existe deux distributions pour Windows, la distribution sans compilateur et la distribution intégrant le compilateur GNU gcc (distribution minGW). Les distributions sont des exécutables Windows qui ne soulèvent aucune difficulté particulière d'installation; installer la version comprenant minGW si on ne dispose pad du compilateur GCC sous Windows.
Il existe des distributions de binaires pour différents linux : Gentoo, Fedora, Freebsd, Ubuntu, ... Attention, sous linux, l'interface graphique de **code::blocks** s'appuie sur les bibliothèques wxGTK qu'il faut donc se procurer par ailleurs : http://dag.wieers.com/packages/wxGTK/.

2 Prise en main de code::blocks

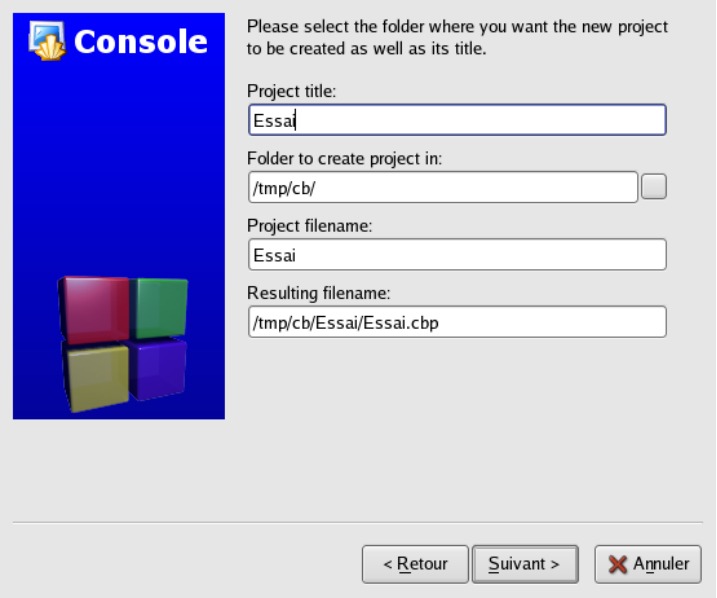
Nous décrivons dans ce qui suit un exemple d'utilisation de **Code::blocks**. La version utilisée est la SVN 3222 du 15 novembre 2006 sous Fedora core5 et Kde.
Une fois l'installation réussie de **code::blocks,** vous le lancez et devez obtenir une fenêtre de ce genre :



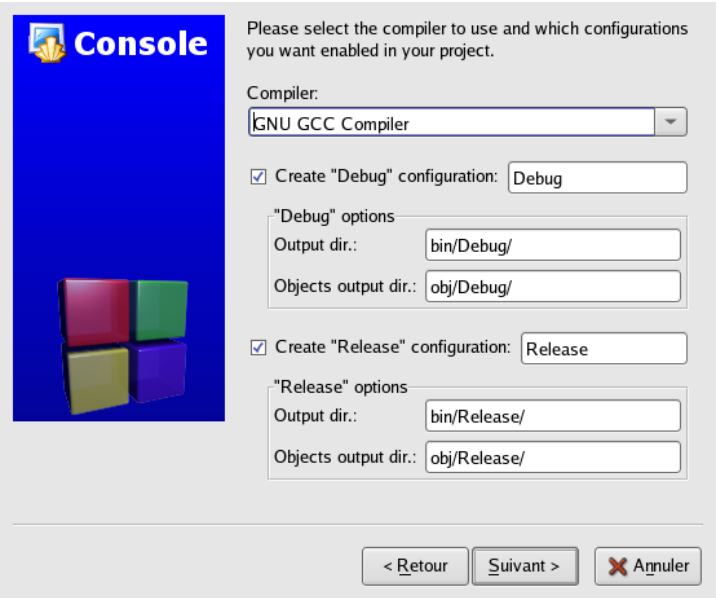
La première opération consiste à créer un nouveau projet : soit en cliquant sur **Create new project** soit en cliquant sur **New project** dans le menu **File** (raccourci clavier **Ctl-Shift-n).** Une fenêtre dialogue **New from template** s'ouvre et vous demande de choisir un modèle de projet :



Il existe des modèles prédéfinis de projets et on peut également choisir de créer seulement des fichier (.h, .cpp). On choisit *Application Console,* une fenêtre nommée **Application console** s'ouvre dans laquelle on spécifie le nom du projet et le dossier où il sera sauvé :

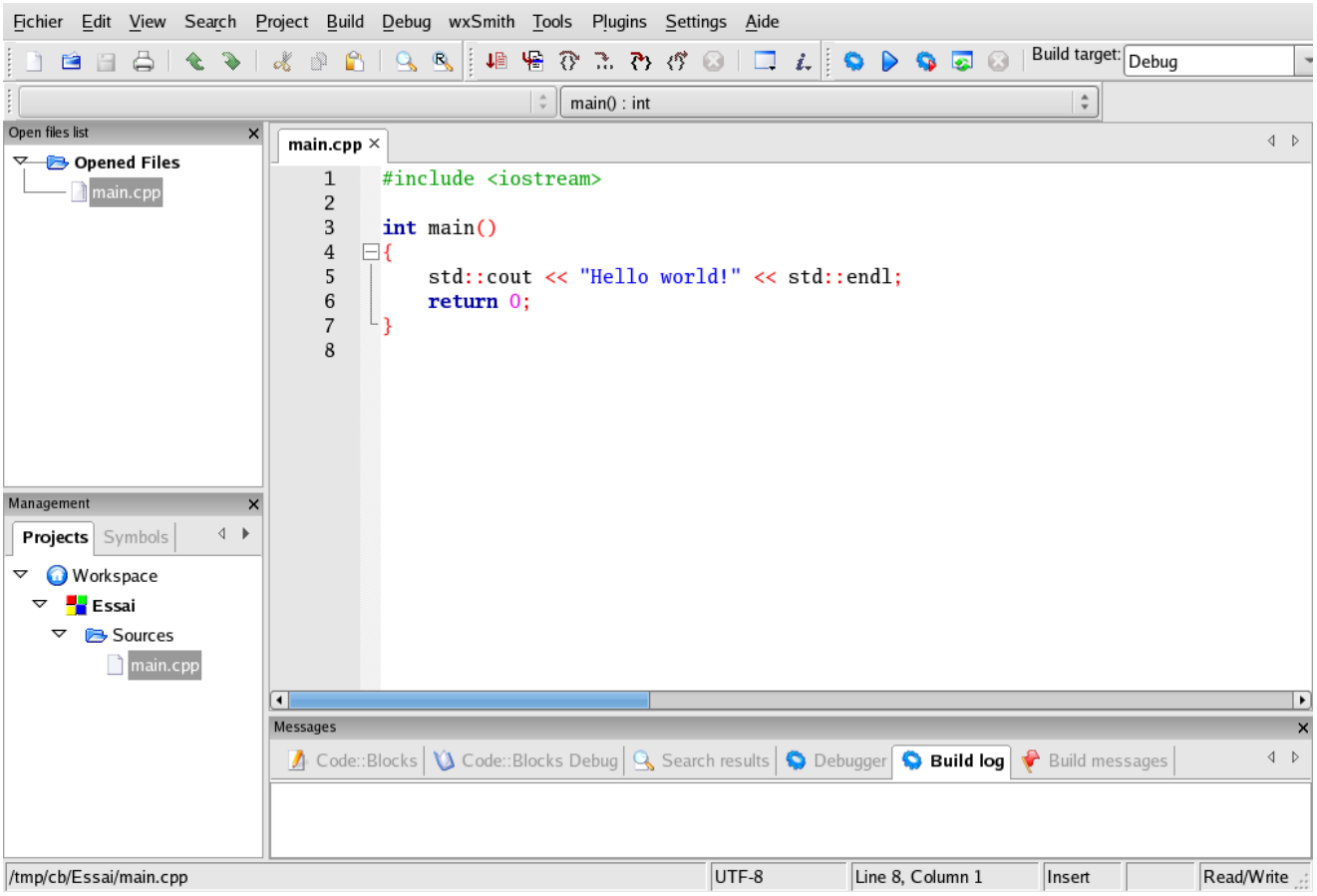


ensuite s'ouvre une fenêtre permettant de choisir le compilateur (par défaut GNU GCC) et les versions qui seront générées, par défaut Debug (permettant de faire du suivi d'exécution) et Release (version sabs option de débogage) :

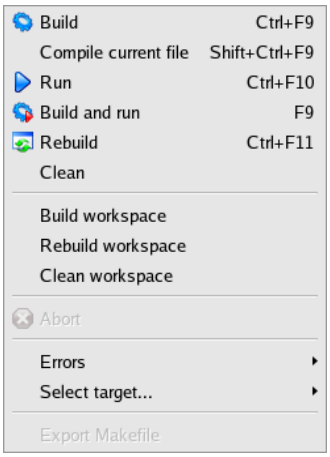


Enfin, on décide si l'on développe une application C ou C++ :

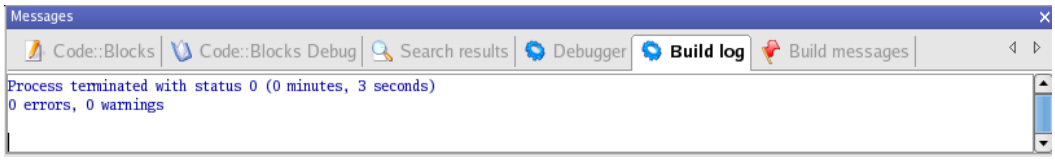
Par défaut, des fichiers sont crées (leur nombre dépendant du modèle de projet). Pour une *application console*, seul un fichier *main.cpp* minimaliste est créé (programme "Hello word!"). Dans le volet **Project** apparaît l'arborescence **Workspace** des projets et en particulier celui qui vient d'être créé. En cliquant sur le fichier source *main.cpp*, ce dernier est affiché dans une fenêtre centrale d'édition avec mis en évidence de la syntaxe C++ :



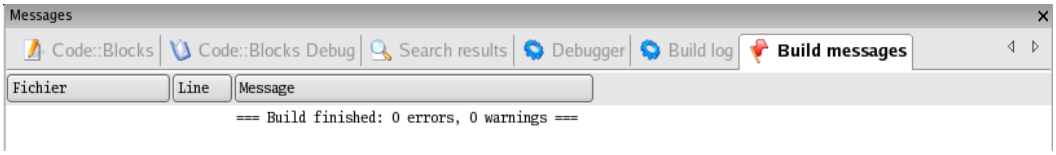
On peut modifier le code source de façon immédiate avec des actions standards d'un éditeur souris. A tout moment le projet peut être sauvé en cliquant sur l'icone représentant une disquette ou ouvrir le menu **File** qui offre plusieurs commandes de sauvegarde.
A ce stade, le projet peut déjà être compilé et exécuté. Pour le compiler, il suffit de cliquer sur la commande **Build** du menu **Build** (raccourci clavier **Ctl-F9**) qui va enchainer les opérations de compilation et d'édition de liens, générant ainsi un fichier exécutable (*essai.exe* dans notre exemple) ;



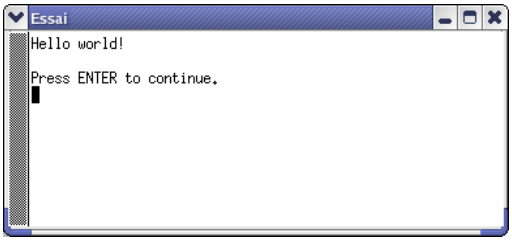
Les éventuels avertissements, messages d'erreur et l'état final du processus de compilation apparaissent dans l'onglet **Build log** situé en bas **:**



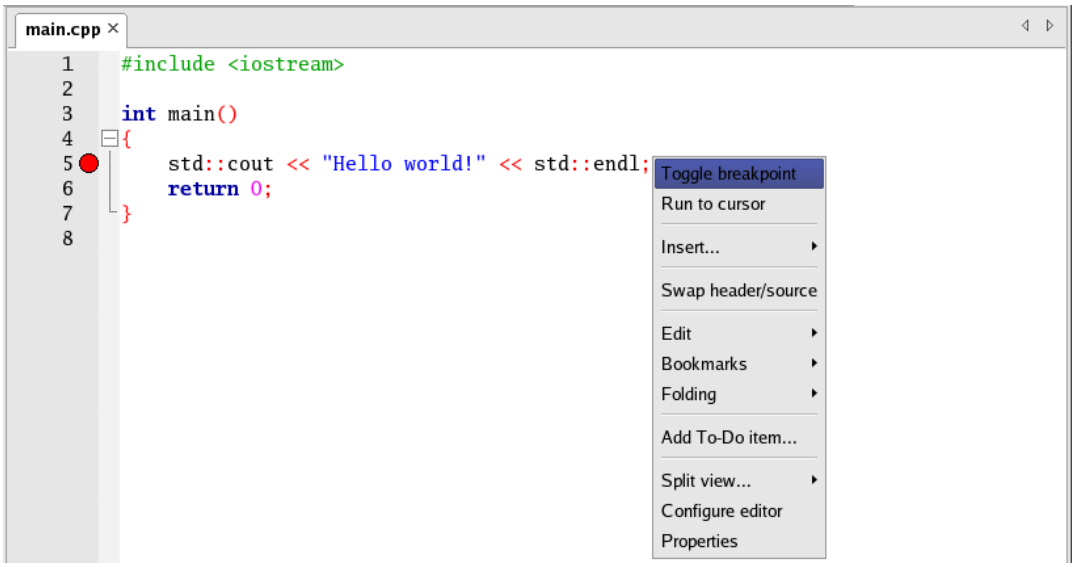
Lorsqu'il y a des erreurs, l'onglet **Build messages** indique les lignes en cause et en cliquant sur le message d'erreur on est positionné dans l'éditeur sur la ligne en cause :



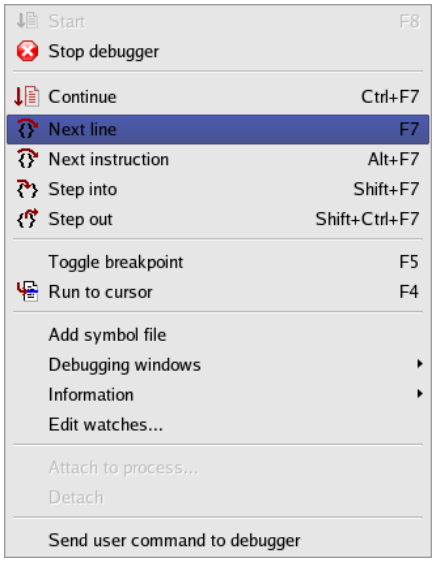
Pour exécuter le code, il suffit de cliquer sur la commande **Run** du menu **Build** (raccourci clavier **Ctl-F10**), une fenêtre d'éxécution s'ouvre alors (*application console*) :



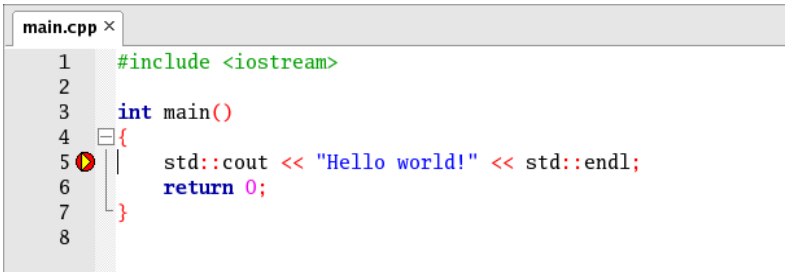
On peut enchainer la compilation et l'exécution en cliquant sur la commande **Build & Run** du menu **Build** (raccourci clavier **F5**).
Code::blocks offre des possibilités de débogage interactives. Il faut tout d'abord placer au moins un point d'arrêt sur une des lignes du code, soit en cliquant sur la colonne de gauche, soit en invoquant le menu contextuel (click bouton de droite) et en cliquant sur **Toggle breakpoint** au niveau de la ligne de code; un point rouge devrait apparaître :



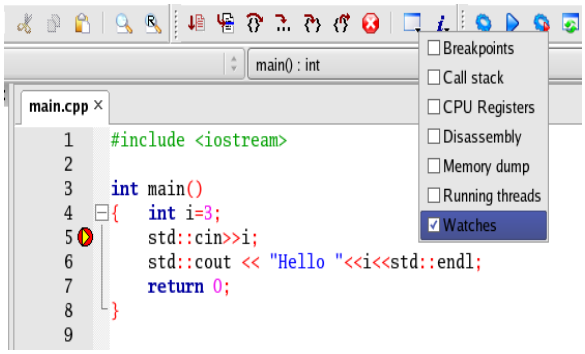
Ensuite on lance l'exécution du programme à l'aide de la commande **Start** du menu **Debug :**



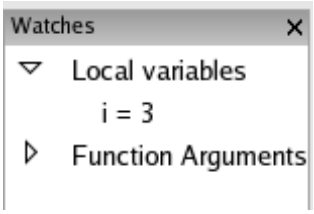
le programme s'arrête sur la première ligne rencontrée qui possède un point d'arrêt (un curseur apparaît) :



A ce stade, on peut consulter le contenu des variables actives dans la fenêtre **Watches** que l'on fait apparaître en cochant la case Watches dans le menu **Debugging windows** :



La fenêtre **Watches** fournit par défaut le contenu des variables actives et on peut en rajouter d'autres à l'aide du menu contextuel de l'éditeur :

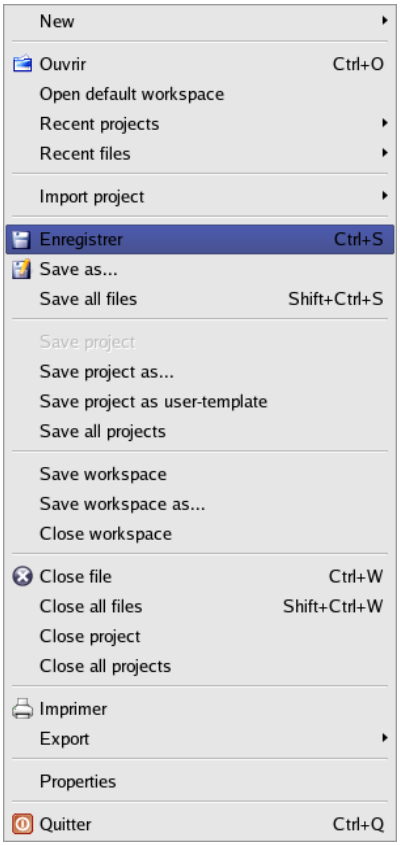


On peut dérouler l'exécution suivant différents modes : commandes **Step over (F7), Step into** (**Shift-F7**) ou **Step out (Ctl-Shift-F7) et Abort** du menu **Debug** ou via les boutons d'accès rapide **:**

3 Principales fonctionnalités

Nous ne décrivons que les grandes lignes et invitons le lecteur à les explorer en détail par lui même.

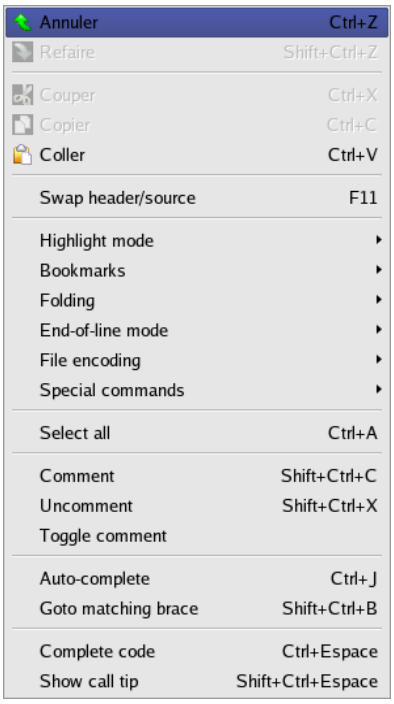
3.1 Menu fichier



Le menu **File** propose les actions usuelles relatives aux fichiers ainsi que le actions d'impression et d'export de projet (HTML, RTF,ODT et PDF).

3.2 Menu Edit

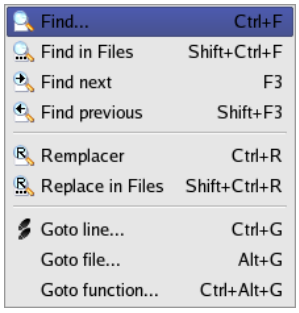
Le menu **Edit** propose les actions attachées à l'éditeur du code :



auto-complétions de code, passer du fichier d'implémentation au fichier entête, poser des marqueurs, (dé)commenter, ...

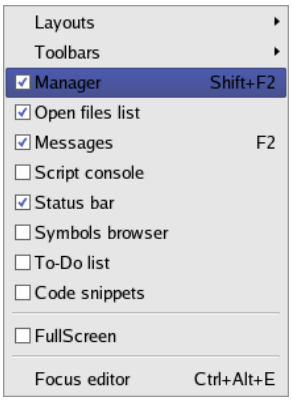
3.3 Menu Search

Le menu **Search** permet d'effectuer des opérations de recherche, de déplacement rapide et de remplacement :



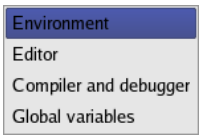
3.4 Menu View

Ce menu permet de gérer les différentes fenêtres actives de l'environnement :



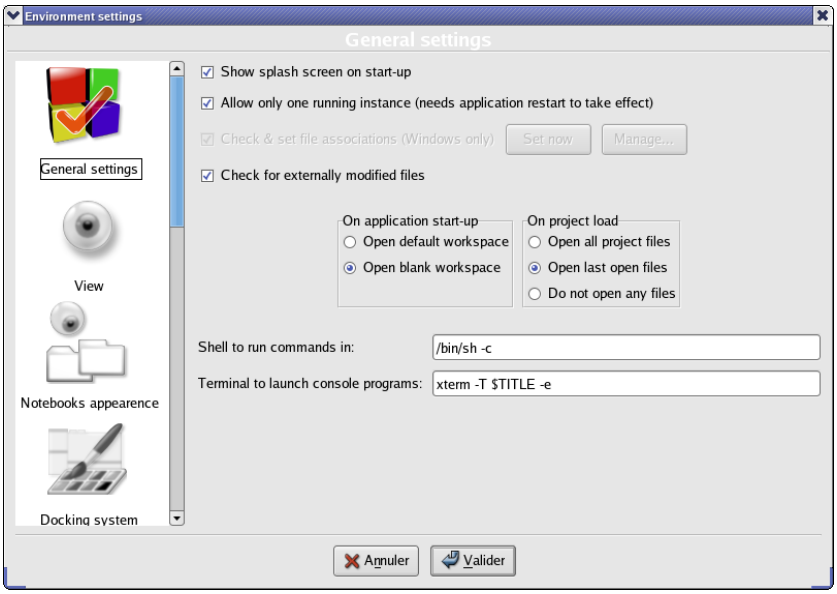
3.5 Menu Settings

Le menu **Settings** permet de régler différents comportements de l'environnement. Il se divise en 4 catégories :



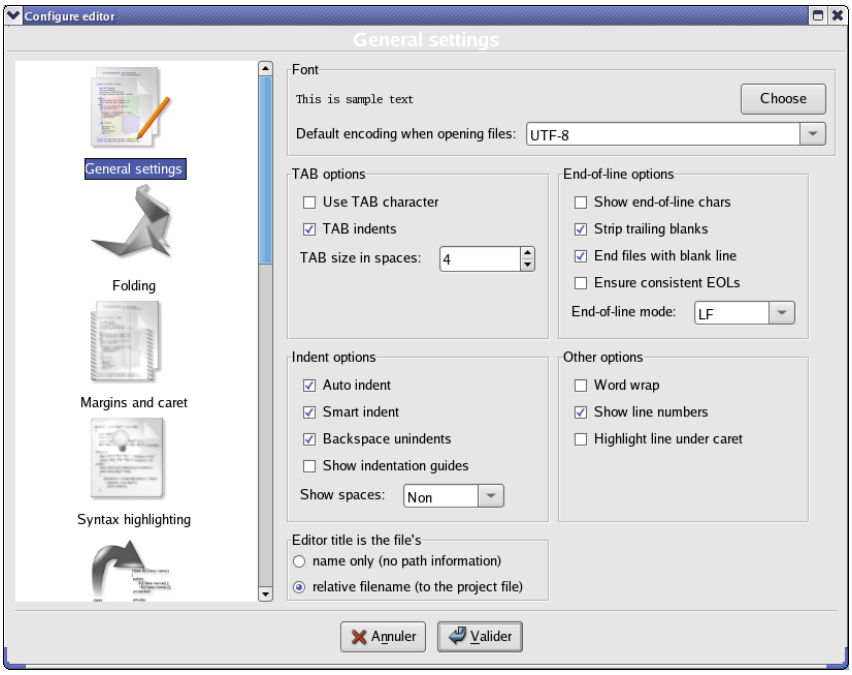
3.5.1 Menu Environnement

Ce menu ouvre une fenêtre proposant les règlages généraux de l'environnement qui de décompose en divers règlages, dont le choix des polices des fénêtres de message, la taille des icones, l'activation des sauvegardes automatiques, ... :



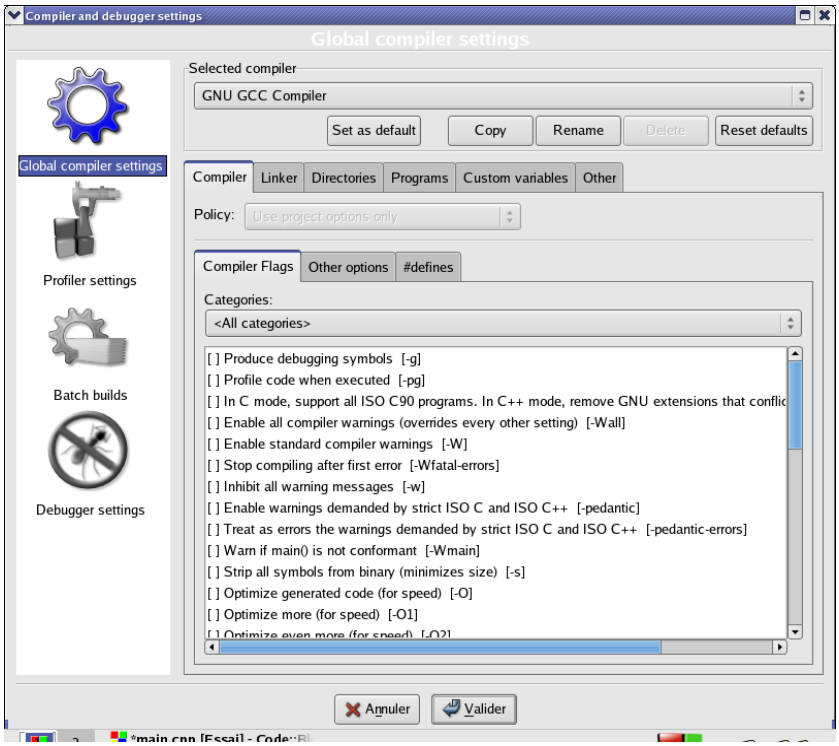
3.5.2 Menu Editor

Le menu Editor permet de règler divers paramètres de l'éditeur de code C++, avec entre autres : le choix de la police de l'éditeur, les couleurs des syntaxes C++, la définition des abréviations, les règles de complétion de codes, les raccourcis clavier, le choix de la mise en forme automatique du code, ...



3.5.3 Menu Compiler and Debugger

Ce menu permet de préciser les paramètres du compilateur et du debogueur qui seront appliqués par défaut. On y trouve la plupart des options du compilateur et la possibilité d'en activer d'autres, la possibilitéd'indiquer des librairies externes,... La fenêtre liée au debogueur permet d'activer la possibilité de voir le contenu d'une variable du code lors du survol à la souris en mode debug et d'afficher la fenêtre ou apparaissent tous les messages de debogueur.



3.5.4 Menu Global Variable

Ce menu permet de définir des variables internes à Code::blocks pouvant être utilées par le compilateur.

3.6 Barre d'accès rapide

La barre d'icones propose sous forme de boutons les actions les plus utilisées :

dont la signification s'affiche lorsque la souris les survole : Nouveau projet, Nouveau fichier, Ouvrir un projet ou un fichier, Sauver les modifications, Défaire, Refaire, Couper, Copier, Coller, Rechercher dans un ou plusieurs fichiers, Remplacer dans le fichier courant, Construire (compiler le projet), Exécutere, Stopper la construction ou l'exécution, Un champ indiquant la cible de construction, Continuer (*mode debug*), Exécuter jusqu'au curseur (*mode debug*), Avancer sans rentrer dans les fonctions(*step over du mode debug*), Avancer en rentrant dans les fonctions (*step in du mode debug*), Sortir d'une fonction (*step out du mode debug*), Stopper le mode debug.

4 Fichiers d'aide

Par défaut, **code::blocks** n'offre pas de documentation C++ en ligne. Dans le menu **Settings -> Environment -> Help files** il est possible d'en ajouter, par exemple celle de BorlandC++ sous windows si on l'a ! Sinon, on peut toujours utiliser une aide sur internet, par exemple celle, très complète, de C. Casteyde : *http://casteyde.christian.free.fr/cpp/cours/online/book1.html.* Il est possible de l'appeler directement depuis **code::blocks** depuis le menu **Tools** en ajoutant un nouvel outil (commande **Configure tools ...** dans le menu **Tools**).
On trouvera des informations relatives à Code::blocks à l'adresse suivante :

http://wiki.codeblocks.org/index.php?title=Special:Allpages.