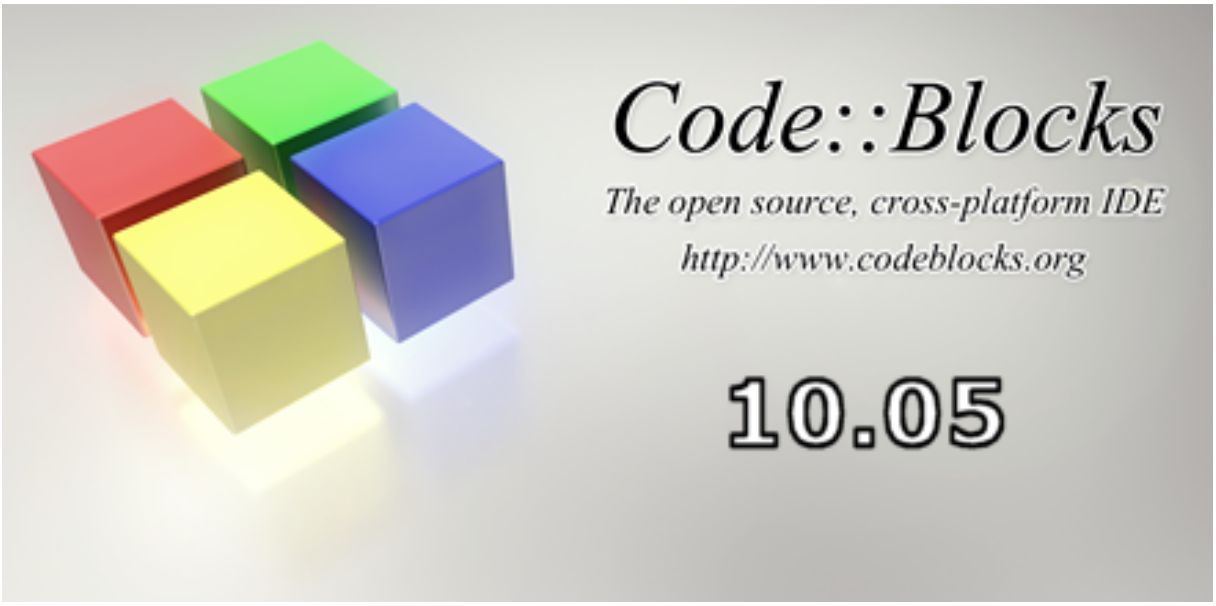


Code::Blocks

Manuel Utilisateur

Version 1.1

Merci à l'équipe CodeBlocks:

Anders F. Björklund (afb), Biplab Kumar Modak (biplab), Bartomiej wiecki (byo), Paul A. Jimenez (ceniza), Koa Chong Gee (cyberkoa), Daniel Orb (daniel2000), Lieven de Cock (killerbot), Yiannis Mandravellos (mandrav), Mispunt (mispunt), Martin Halle (morten-macfly), Jens Lody (jens), Jerome Antoine (dje), Damien Moore (dmoore), Pecan Heber (pecan), Ricardo Garcia (rickg22), Thomas Denk (thomasdenk), tiwag (tiwag)

Traduction de la version originale anglaise par Gérard Durand (gd_on).

1 Gestion de Projet CodeBlocks

Les instructions du chapitre 3 à la page 58 et du chapitre 4 à la page 68 sont les documentations officielles du site Wiki de CodeBlocks où elles ne sont disponibles qu'en anglais.

> **Note:**
>
> Remarque du traducteur : Les références aux menus sont traduites en français. Cela suppose donc que vous avez installé la francisation de l'interface de CodeBlocks que vous pouvez obtenir, notamment via le forum, dans la rubrique CodeBlocks Translation. Ne plus utiliser celle du site LaunchPad bien trop ancienne et largement dépassée. Les images ci-dessous sont celles de la documentation originale, en anglais

L'illustration ci-dessous montre l'apparence de la fenêtre de l'interface utilisateur de CodeBlocks.

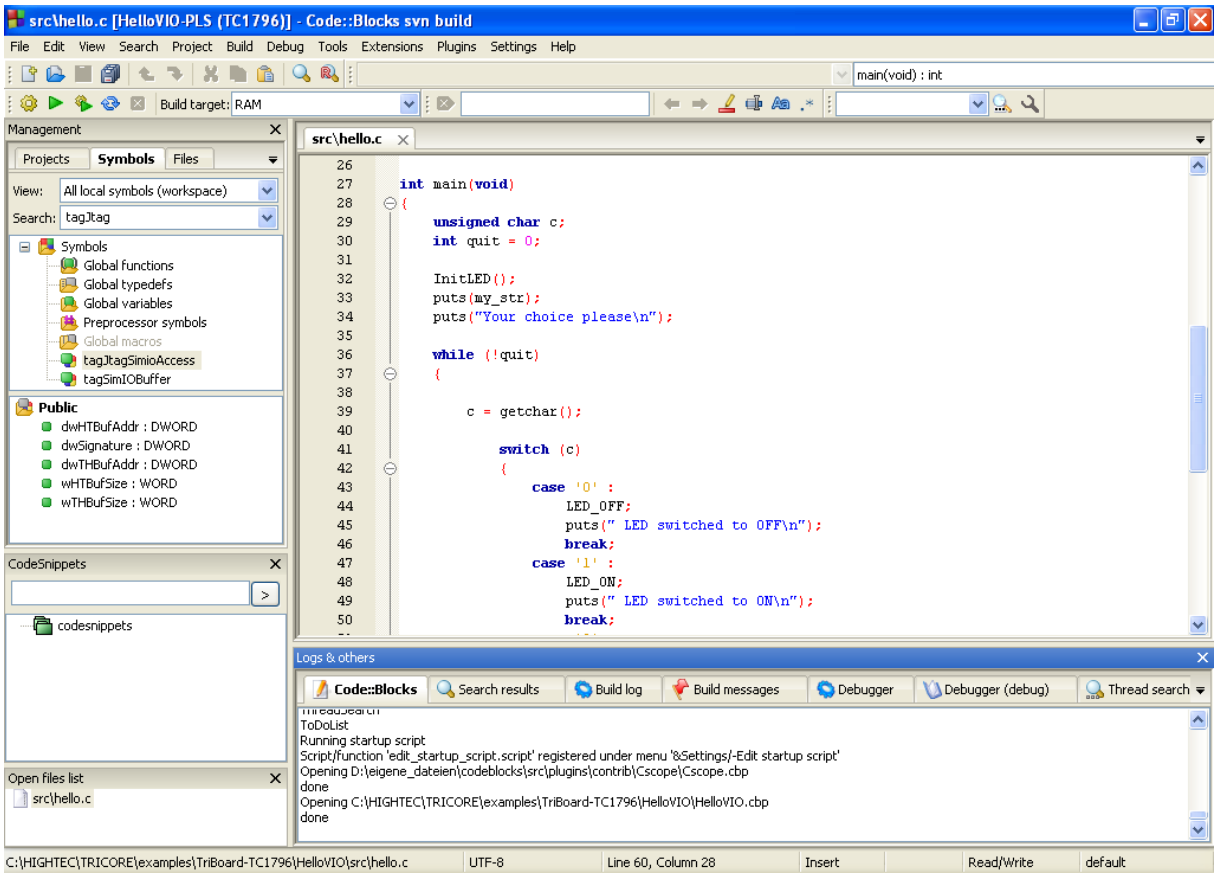


Figure 1.1: Environnement de développement Intégré (IDE) de CodeBlocks

Gestion Cette fenêtre contient l'interface 'Projets' qui dans le texte suivant sera référencée comme vue du projet. Cette vue affiche tous les projets ouverts dans CodeBlocks à

un instant donné. L'onglet 'Symboles' de la fenêtre Gestion affiche les symboles, les variables etc.

Éditeur Dans l'illustration ci-dessus, un fichier source nommé `hello.c` est ouvert avec colorisation de syntaxe dans l'éditeur.

Liste des fichiers ouverts affiche une liste de tous les fichiers ouverts dans l'éditeur, dans cet exemple : `hello.c`.

CodeSnippets peut être affiché via le menu 'Vue' →' CodeSnippets' . Ici vous pouvez gérer des modules de texte, des liens vers des fichiers et des liens vers des urls.

Journaux & autres . Cette fenêtre est utilisée pour sortir des résultats de recherche, des messages envoyés par un compilateur etc..

La barre d'état donne un aperçu des paramétrages suivants :

- Chemin absolu d'un fichier ouvert dans l'éditeur.
- L'éditeur utilise l'encodage par défaut de votre système d'exploitation. Cette configuration sera affichée par **default**.
- Numéros de ligne et de colonne de la position actuelle du curseur dans l'éditeur.
- Le mode de configuration du clavier pour insérer du texte (Insertion ou Remplacement).
- État actuel du fichier. Un fichier modifié sera marqué comme Modifie sinon cette case reste vide.
- Autorisation d'un fichier. Un fichier qui est en lecture seule sera affiché Lecture seule dans la barre d'état. Dans la fenêtre 'Ouvrir la liste de fichiers' ces fichiers seront identifiés par une icône de verrouillage superposée.

> **Note:**
>
> Dans l'éditeur courant, l'utilisateur peut choisir les propriétés du menu de contexte. Dans le dialogue apparaissant dans l'onglet 'Général' , l'option 'Le fichier est en lecture seule' peut être sélectionnée. Cette option marquera le fichier correspondant comme étant en lecture seule pour CodeBlocks, mais les attributs en lecture et écriture du fichier original ne seront pas modifiés dans le système de fichiers.

- Si vous démarrez CodeBlocks en ligne de commande avec `--personality=<profile>` la barre d'état affichera le profil utilisateur courant, sinon **default** sera affiché. Les paramètres de CodeBlocks sont enregistrés dans le fichier de configuration correspondant `<personality>.conf`.

CodeBlocks offre une gestion des projets très flexible et très compréhensible. Le texte suivant ne montre que quelques aspects de la gestion de projets.

1.1 Vue du projet

Dans CodeBlocks, les sources et les paramètres d'un processus de génération sont stockés dans un fichier projet <name>.cbp. Les sources en C/C++ et les fichiers d'entêtes correspondants (ou headers) sont les composants typiques d'un projet. La façon la plus simple de créer un projet est de passer par la commande 'Fichier' →' Projet' et de choisir un assistant. Vous pouvez alors ajouter des fichiers au projet via le menu de contexte 'Ajouter des fichiers' de la fenêtre de gestion.

CodeBlocks gère les fichiers de projets en catégories qui dépendent de l'extension des fichiers. Les catégories suivantes sont prédéfinies :

Sources contient les fichiers sources dont l'extension est `*.c;*.cpp;`.

ASM Sources contient les fichiers sources dont l'extension est `*.s;*.S;*.ss;*.asm`.

Headers contient, entre autres, les fichiers dont l'extension est `*.h;`.

Ressources contient les fichiers pour paramétrer l'aspect des fenêtres des wxWidgets avec les extensions `*.res;*.xrc;`. Ces types de fichiers sont affichés dans l'onglet 'Ressources' de la fenêtre de Gestion.

Les paramètres des types et catégories de fichiers peuvent être ajustés via le menu de contexte 'Arbre des projets' →' Editer les types et catégories de fichiers' . Ici, vous pouvez définir aussi des catégories personnalisées pour les extensions de votre choix. Par exemple, si vous souhaitez lister des scripts d'édition de liens avec l'extension `*.ld` dans une catégorie nommée `Linkerscript`, vous n'avez qu'à créer une nouvelle catégorie.

> **Note:**
>
> Si vous désactivez 'Arbre des projets' →' Catégoriser par type de fichiers' dans le menu de contexte, l'affichage par catégories sera masqué, et les fichiers seront listés comme ils sont stockés dans le système de fichiers.

1.2 Notes pour les Projets

Dans CodeBlocks, ce qu'on appelle des notes peuvent être stockées dans un projet. Ces notes peuvent contenir de brèves descriptions ou des points particuliers pour le projet correspondant. En affichant ces informations à l'ouverture d'un projet, les autres utilisateurs peuvent avoir un rapide aperçu de l'avancement du projet. L'affichage des notes peut être validé ou invalidé via l'onglet Notes des Propriétés d'un projet.

1.3 Modèle de Projet

CodeBlocks est fourni avec tout un ensemble de modèles de projets qui sont affichés quand on crée un nouveau projet. Cependant, vous pouvez aussi enregistrer des modèles personnalisés pour y sauvegarder vos propres spécifications d'options de compilation, les optimisations à utiliser, les options spécifiques aux machines etc. Ces modèles seront enregistrés

dans le répertoire `Documents and Settings\<user>\Application Data\codeblocks\UserTemplate`
Si les modèles doivent pouvoir être ouverts par tous les utilisateurs, ils devront être copiés dans un répertoire correspondant de l'installation de CodeBlocks. Ces modèles seront alors affichés lors du démarrage suivant de CodeBlocks dans 'Nouveau' →' Projet' →' Modèles utilisateur' .

> **Note:**
>
> Les modèles disponibles dans l'assistant Projet peuvent être édités en les sélectionnant via un clic droit.

1.4 Créer des Projets à partir de Cibles de Génération

Dans les projets, il est nécessaire d'avoir à disposition différentes variantes de projets. On appelle ces variantes Cibles de Génération. Elles diffèrent par leurs options de compilation, les informations de débogage et/ou le choix des fichiers. Une cible de génération peut aussi être externalisée dans un projet séparé. Pour ce faire, cliquer sur 'Projet' →' Propriétés' puis sélectionner la variante dans l'onglet 'Générer les cibles' et cliquer sur le bouton 'Créer un projet à partir d'une cible' (voir Figure 1.2 à la page 4).

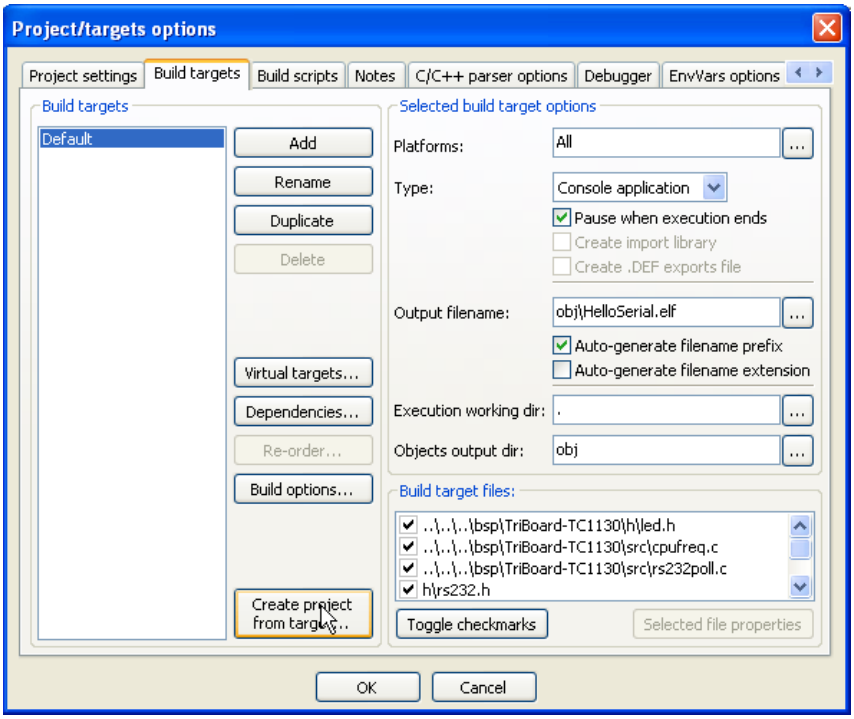


Figure 1.2: Cibles de Génération

1.5 Cibles Virtuelles

Les projets peuvent être également structurés dans CodeBlocks en ce qu'on appelle des cibles virtuelles. Une structure fréquemment utilisée de projet consiste en deux cibles de génération, la première cible 'Debug' qui contient des informations pour le débogage et la

seconde cible 'Release' sans ces informations. En ajoutant Cibles Virtuelles via 'Projet' →'Propriétés' →'Cibles de génération' on peut combiner des cibles de génération individuelles. Par exemple, une Cible Virtuelle 'All' peut créer les cibles Debug et Release simultanément. Les cibles virtuelles sont affichées dans la barre de symboles du compilateur dans Générer les cibles.

1.6 Étapes Pré- et Post Génération

Dans CodeBlocks on peut effectuer des opérations complémentaires avant et après la compilation d'un projet. Ces opérations sont appelées étapes de Pré génération ou Post génération. Des Post générations typiques sont :

- Création d'un format Intel Hexformat à partir un objet terminé
- Manipulation d'objets par `objcopy`
- Générer des fichiers de dump par `objdump`

Exemple

Créer le désassemblage d'un objet sous Windows. Le transfert vers un fichier nécessite l'appel à `cmd` avec l'option `/c`.

```
cmd /c objdump -D name.elf > name.dis
```

Un autre exemple de Post génération peut être l'archivage d'un projet. Pour cela, créez une cible de génération 'Archive' et incluez les instructions suivantes dans l'étape de post génération :

```
zip -j9 $(PROJECT_NAME)_$(TODAY).zip src h obj $(PROJECT_NAME).cbp
```

Avec cette commande, le projet actif et ses sources, entêtes et objets seront compressés en tant que fichier zip. En faisant ainsi, les variables intégrées $(PROJECT_NAME) et $(TODAY), le nom du projet et la date courante seront extraites (voir section 3.2 à la page 59). Après l'exécution de la cible 'Archive', le fichier compressé sera stocké dans le répertoire du projet.

Dans le répertoire `share/codeblocks/scripts` vous trouverez quelques exemples de scripts. Vous pouvez ajouter un script via le menu 'Paramètres' →'Edition de scripts' et l'enregistrer dans un menu. Si vous exécutez par exemple le script `make_dist` depuis le menu, alors tous les fichiers appartenant à un projet seront compressés dans une archive `<project>.tar.gz`.

1.7 Ajouter des Scripts à des Cibles de Génération

CodeBlocks offre la possibilité d'utiliser des actions de menus dans les scripts. Le script représente un autre degré de liberté pour contrôler la génération de votre projet.

> **Note:**
>
> Un script peut également être inclus dans une Cible de Génération.

1.8 Espace de travail et Dépendances de Projet

Des projets multiples peuvent être ouverts dans CodeBlocks. En enregistrant les projets ouverts via 'Fichier' →'Enregistrer l'espace de travail' vous pouvez les rassembler dans un seul espace de travail sous <name>.workspace. Si vous ouvrez <name>.workspace au démarrage suivant de CodeBlocks, tous les projets seront de nouveau affichés.

Les logiciels complexes sont un assemblage de composants qui sont gérés dans différents projets CodeBlocks. De plus, lors de la génération de tels logiciels, il y a souvent des dépendances entre ces projets.

Exemple

Un projet A contient des fonctions de base qui sont rendues disponibles aux autres projets sous forme d'une librairie. Maintenant, si les sources de ce projet sont modifiées, alors la librairie doit être re-générée. Afin de maintenir la consistance entre un projet B qui utilise ces fonctions et le projet A qui les implémente, le projet B doit dépendre du projet A. Les informations nécessaires aux dépendances des projets sont enregistrées dans l'espace de travail adéquat, ainsi chaque projet peut être généré séparément. L'utilisation des dépendances rend également possible le contrôle de l'ordre dans lequel sont générés les projets. Les dépendances de projets peuvent être configurées en sélectionnant le menu 'Projet' →'Propriétés' puis en cliquant sur le bouton 'Dépendances du projet'.

1.9 Inclure des Fichiers en Assembleur

Dans la fenêtre Gestion d'une vue de projet, les fichiers en Assembleur sont affichés dans la catégorie `ASM Sources`. L'utilisateur peut changer la liste des fichiers dans les catégories (voir section 1.1 à la page 3). Un clic droit sur un des fichiers assembleur listés ouvrira un menu de contexte. Sélectionner 'Propriétés' pour ouvrir une nouvelle fenêtre. Sélectionnez maintenant l'onglet 'Générer' et activez les deux champs 'Compiler le fichier' and 'Édition de liens du fichier'. Sélectionnez ensuite l'onglet 'Avancé' et exécutez les étapes suivantes :

1. Configurer 'Variable de compilation' à CC
2. Sélectionner le compilateur dans 'Pour ce compilateur'
3. Sélectionner 'Utiliser des commandes personnalisées pour générer ce fichier'
4. Dans la fenêtre, entrez :

```
$compiler $options $includes <asopts> -c $file -o $object
```

Les variables de CodeBlocks sont identifiées par un `$` (voir section 3.4 à la page 63). Elles sont automatiquement configurées, ainsi vous n'avez à remplacer que l'option de l'assembleur <asopt> par vos propres configurations.

1.10 Éditeur et Outils

1.10.1 Code par Défaut

Les règles de codage dans une compagnie imposent d'avoir un modèle standard. Avec CodeBlocks, il est possible d'inclure un contenu prédéfini automatiquement en début de fichier lors de la créations d'une nouvelle source C/C++ ou d'entêtes (headers). Le contenu prédéfini est dénommé code par défaut. Cette configuration peut être sélectionnée dans 'Paramètres' →'Éditeur' Code par Défaut. Si vous créez un nouveau fichier alors une expansion des variables macro, notamment celles de 'Paramètres' →'Variables Globales' , est effectuée. Un nouveau fichier peut être créé via le menu 'Fichier' →'Nouveau' →'Fichier' .

Exemple

```
/*************************************************************************
 *  Project: $(project)
 *  Function:
 *************************************************************************
 *  $Author: mario $
 *  $Name:  $
 *************************************************************************
 *
 *  Copyright 2007 by company name
 *
 ************************************************************************/
```

1.10.2 Abréviations

Pas mal de frappes au clavier peuvent être économisées dans CodeBlocks en définissant des abréviations. Ceci peut s'obtenir en sélectionnant 'Paramètres' →'Éditeur' et en définissant les abréviations par un nom <name>, qui peut alors être appelé par un raccourci clavier Ctrl-J (voir Figure 1.3 à la page 8).

On peut également les paramétrer en incluant des variables $(NAME) dans les abréviations.

```
#ifndef $(Guard token)
#define $(Guard token)
#endif // $(Guard token)
```

Quand on utilise l'abréviation <name> dans un texte source et qu'on utilise Ctrl-J, le contenu de la variable est récupéré puis inclus.

1.10.3 Personnalités

Les configurations de CodeBlocks sont enregistrées en tant que données d'application dans un fichier dénommé `<user>.conf` dans le répertoire de `codeblocks`. Ce fichier de configuration contient des informations telles que les derniers projets ouverts, le paramétrage de l'éditeur, l'affichage des barres de symboles etc. Par défaut, la personnalité 'default' est utilisée et sa configuration sauvegardée dans un fichier `default.conf`. Si CodeBlocks est lancé en ligne de commande avec le paramètre `--personality=myuser`, le paramétrage

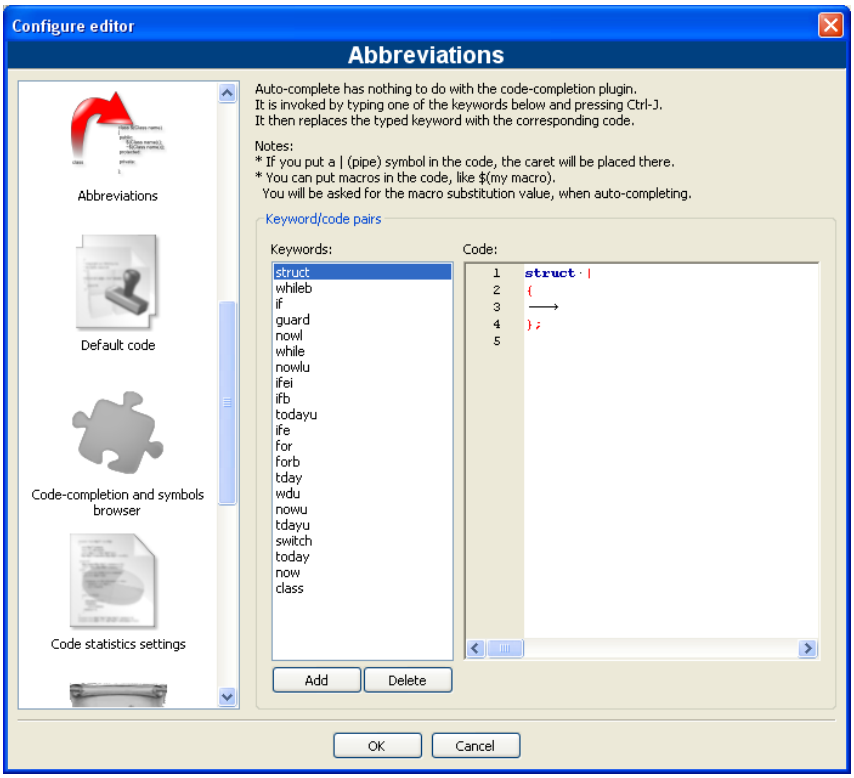


Figure 1.3: Définition des abréviations

sera enregistré dans un fichier `myuser.conf`. Si le profil n'existe pas déjà, il sera automatiquement créé. Cette procédure rend possible la création de différents profils pour différentes étapes de travail. Si vous lancez CodeBlocks en ligne de commande avec le paramètre additionnel `--personality=ask`, une boîte de sélection sera affichée avec tous les profils disponibles.

> **Note:**
>
> Le nom du profil/personnalité courant est affiché dans le coin à droite de la barre d'état.

1.10.4 Fichiers de Configuration

Les paramètres de CodeBlocks sont enregistrés dans le fichier de profil `default.conf` dans le répertoire `codeblocks` de votre Application Data. Quand vous utilisez des personnalités (ou profils) (voir sous-section 1.10.3 à la page 7), les détails de configuration sont enregistrés dans un fichier `<personality>.conf`.

L'outil `cb_share_conf`, qu'on trouve dans le répertoire d'installation de CodeBlocks, est utilisé pour gérer et enregistrer ces paramétrages.

Si vous souhaitez définir des paramètres standard pour plusieurs utilisateurs de l'ordinateur, le fichier de configuration `default.conf` doit être enregistré dans le répertoire `\Documents and Settings\Default User\Application Data\codeblocks`. Lors du premier démarrage,

CodeBlocks copiera les valeurs par défaut depuis 'Default User' vers le répertoire "Application data" de l'utilisateur courant.

Pour créer une version portable de CodeBlocks sur clé USB, procédez comme suit. Copiez le répertoire d'installation de CodeBlocks vers la clé USB et stockez le fichier de configuration `default.conf` dans ce répertoire. Cette configuration servira de paramétrage global. Faites attention au fait que ce fichier soit accessible en écriture, sinon les changements de configuration ne pourront y être enregistrés.

1.10.5 Navigation et Recherche

Dans CodeBlocks il y a plusieurs façons de naviguer rapidement entre les fichiers et les fonctions. Une procédure typique est la configuration de marques de recherche. Via le raccourci clavier Ctrl-B une marque est posée ou supprimée dans un fichier source. Via Alt-PgUp vous pouvez aller à la marque précédente, et via Alt-PgDn vous pouvez aller à la marque suivante.

Si vous sélectionnez l'espace de travail ou un projet particulier de l'espace de travail dans la vue du projet vous pouvez rechercher un fichier dans le projet. Sélectionnez tout simplement 'Rechercher le fichier' depuis le menu de contexte, puis tapez le nom du fichier et le fichier sera sélectionné. Si vous tapez sur la touche Entrée, ce fichier sera ouvert dans l'éditeur (voir Figure 1.4 à la page 9).

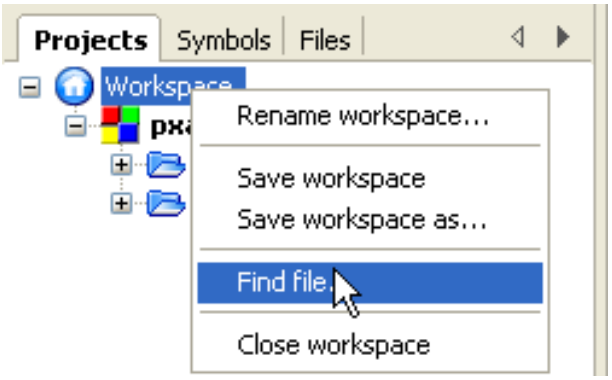


Figure 1.4: Recherche de fichiers

Dans CodeBlocks vous pouvez facilement naviguer entre les Entêtes/Sources en :

1. Positionnant le curseur à l'endroit où le fichier d'entête (header) est inclus puis ouvrir ce fichier via le menu de contexte 'Ouvrir le fichier inclus' (voir Figure 1.5 à la page 10)
2. Basculer du fichier d'entête au fichier source via le menu de contexte 'Basculer entête/source'
3. Sélectionner par exemple un define dans l'éditeur et choisir 'Trouver la déclaration' depuis le menu de contexte pour ouvrir le fichier contenant cette déclaration.

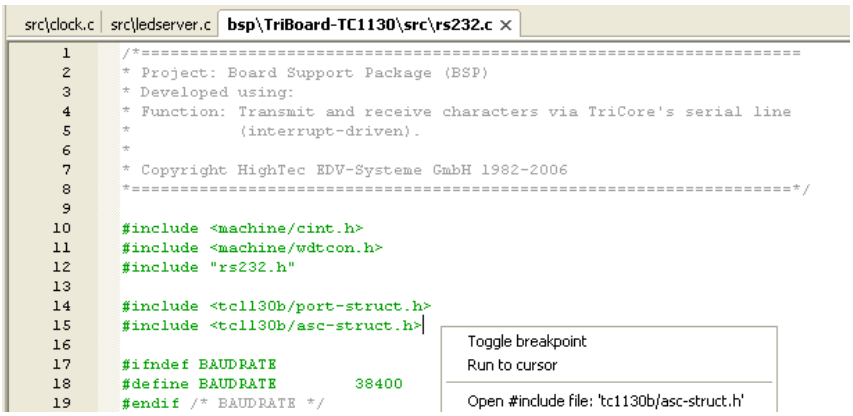


Figure 1.5: Ouverture d'un fichier d'en-têtes

CodeBlocks offre plusieurs possibilités de recherches dans un fichier ou un répertoire. La boîte de dialogue de recherche s'ouvre par 'Chercher' →'Rechercher' (Ctrl-F) ou 'Rechercher dans les fichiers' (Ctrl-Shift-F).

Alt-G et Ctrl-Alt-G sont d'autres fonctions utiles. Le dialogue qui s'ouvrira en utilisant ces raccourcis vous permet de choisir des fichiers/fonctions et aller vous positionner à l'implémentation de la fonction sélectionnée (voir Figure 1.6 à la page 10) ou bien ouvrir le fichier sélectionné dans l'éditeur. Vous pouvez utiliser dans le dialogue des jokers comme $*$ ou ? etc. pour y obtenir une recherche incrémentale.

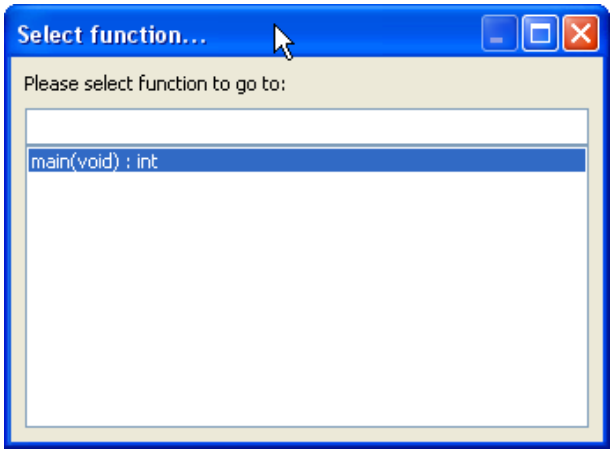


Figure 1.6: Recherche de fonctions

Note:

Avec le raccourci Ctrl-PgUp vous pouvez aller à la fonction précédente, et via Ctrl-PgDn vous pouvez aller à la fonction suivante.

Dans l'éditeur, vous pouvez ouvrir un nouveau dialogue Ouvrir des fichiers Ctrl-Tab et vous pouvez passer de l'un à l'autre via la liste affichée. Si vous appuyez sur la touche Ctrl, alors un fichier peut être sélectionné de différentes façons :

1. Si vous sélectionnez une entrée avec le bouton gauche de la souris, le fichier sélectionné sera ouvert.
2. Si vous appuyez sur la touche Tab vous passez de l'une à l'autre des entrées listées. En relâchant la touche Ctrl le fichier sélectionné sera ouvert.
3. Si vous déplacez la souris au-dessus des entrées listées, alors la sélection courante sera surlignée. En relâchant la touche Ctrl le fichier sélectionné sera ouvert..
4. Si le pointeur de souris est en dehors de la sélection surlignée, vous pouvez utiliser la molette de la souris pour basculer entre les entrées. En relâchant la touche Ctrl le fichier sélectionné sera ouvert.

Une façon commune de développer du logiciel est de jongler avec un ensemble de fonctions implémentées dans différents fichiers. L'extension "Browse Tracker" vous aidera à résoudre cette tâche en vous montrant dans quel ordre ont été sélectionnés les fichiers. Vous pouvez alors naviguer aisément entre les appels de fonctions (voir section 2.8 à la page 42).

L'affichage des numéros de lignes dans CodeBlocks peut s'activer via 'Paramètres' →'Éditeur' →'Paramètres généraux' à l'aide du champ 'Afficher les numéros de ligne'. Le raccourci Ctrl-G ou la commande de menu 'Rechercher' →'Aller à la ligne' vous aidera à atteindre la ligne désirée.

> **Note:**
>
> Si vous maintenez la touche Ctrl enfoncée en sélectionnant du texte dans l'éditeur de CodeBlocks vous pouvez lancer une recherche Internet, notamment avec Google, via le menu de contexte.

1.10.6 Vue des Symboles

La fenêtre Gestion de CodeBlocks offre une vue arborescente des symboles des sources en C/C++ pour naviguer dans les fonctions et les variables. Dans ce type de vue, vous pouvez travailler sur le fichier courant, le projet courant ou tout l'espace de travail.

> **Note:**
>
> Entrer un terme à chercher ou des noms de symboles dans le masque d'entrée 'Rechercher' du navigateur de Symboles permet d'obtenir une vue filtrée des symboles si concordance il y a.

Les catégories suivantes existent pour les symboles :

Fonctions Globales Liste l'implémentation des fonctions globales.

typedefs globales Liste l'utilisation des définitions **typedef**.

Variables globales Affiche les symboles de variables globales.

Symboles du pré-processeur Liste les directives du pré-processeur créées par **#define**.

Macros globales Liste les macros des directives du pré-processeur

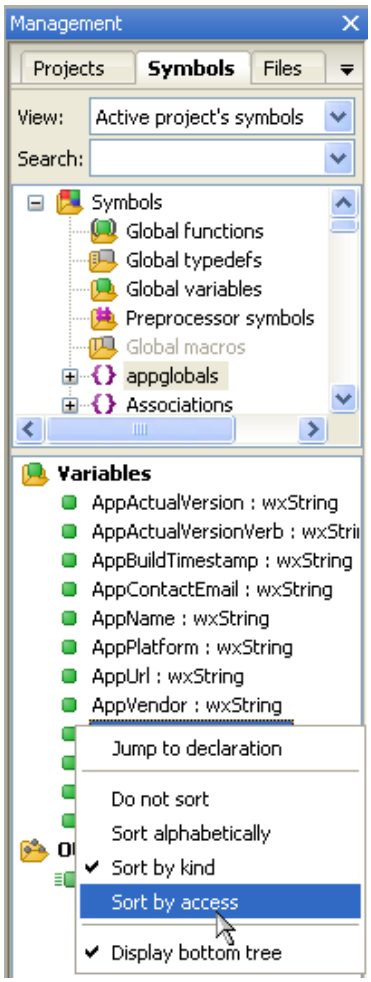


Figure 1.7: Vue des symboles

Les structures et classes sont affichées par le menu 'arbre du bas' et l'ordre de tri peut être modifié via le menu de contexte. Si une catégorie est sélectionnée à la souris, les symboles trouvés seront affichés dans la partie basse de la fenêtre (voir Figure 1.7 à la page 12). Double-cliquer sur un symbole ouvrira le fichier où il est défini ou bien la fonction où elle est implémentée, puis on se positionnera sur la ligne correspondante. Un rafraîchissement automatique du navigateur de symboles, sans avoir à sauvegarder de fichier, peut être activé par le menu 'Paramètres' →'Éditeur' →'Code Complétion' (voir Figure 1.8 à la page 13). Les performances de CodeBlocks seront affectées dans les projets comportant de nombreux symboles.

Note:

Dans l'éditeur, une liste de classes peut être affichée via les menus de contexte 'Insérer méthode de classe' ou 'Toutes méthodes de classes sans implémentation' .

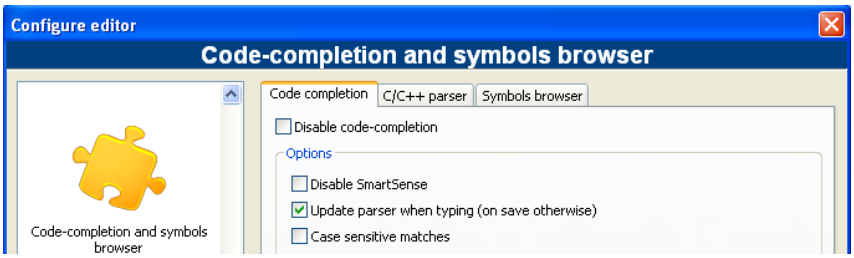


Figure 1.8: Activation de l'analyse en temps réel

1.10.7 Inclure des fichiers d'aide externes

L'environnement de développement CodeBlocks supporte l'inclusion de fichiers d'aide externes via le menu 'Paramètres' →'Environnement' . Insérez le manuel au format chm de votre choix dans la sélection 'Fichiers d'aide' , sélectionnez 'Ceci est le fichier d'Aide par défaut' (voir Figure 1.9 à la page 13). L'entrée $(keyword) est un substituant pour une sélection particulière dans votre éditeur. Vous pouvez alors sélectionner une fonction dans un fichier ouvert de CodeBlocks par un simple clic, et la documentation correspondante s'affichera lorsque vous appuierez sur F1.

Si vous avez inclus plusieurs fichiers d'aide, vous pouvez choisir un terme particulier dans l'éditeur, puis choisir le fichier d'aide dans le menu de contexte 'Chercher dans' pour que CodeBlocks l'y recherche.

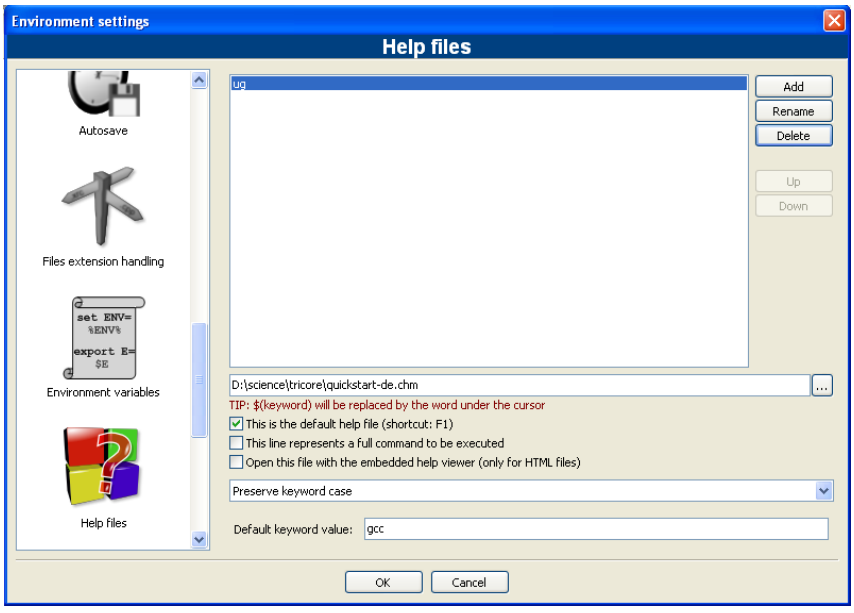


Figure 1.9: Configuration des fichiers d'aide

Dans CodeBlocks vous pouvez également ajouter un support de pages man. Ajouter seulement une entrée 'man' et spécifiez les chemins comme suit (NdT ici pour Linux!).

```
man:/usr/share/man
```

CodeBlocks fourni un 'Visualiseur HTML intégré', qui peut être utilisé pour afficher un simple fichier html et y rechercher des mots clés. Configurez simplement le chemin du

fichier html qui doit être analysé et cochez la case 'Ouvrir ce fichier avec le visualiseur d'aide intégré' via le menu 'Paramètres' →'Environnement' →'Fichiers d'aide' .

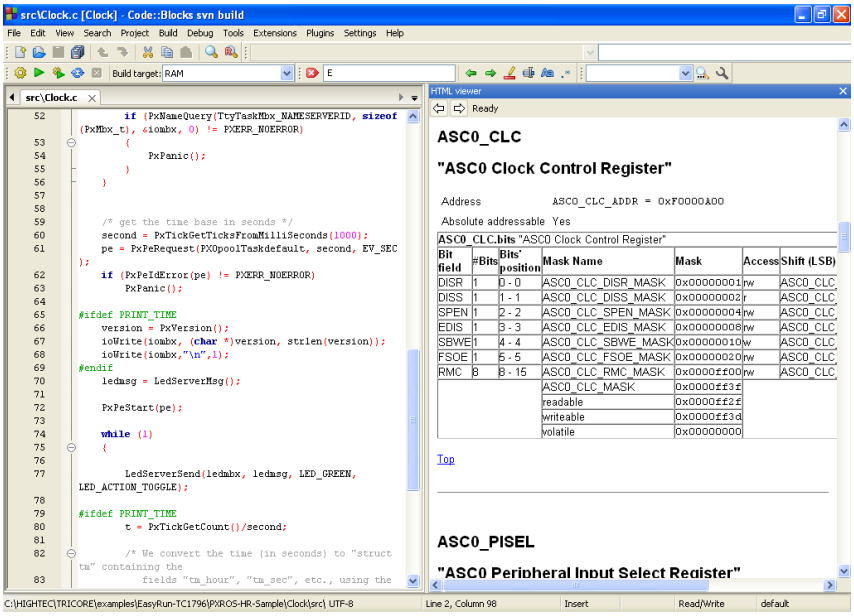


Figure 1.10: Visualiseur HTML intégré

Note:

Si vous sélectionnez un fichier html par double clic dans l'explorateur (voir section 2.7 à la page 38) alors le visualiseur html intégré sera démarré, du moins si aucune association vers les fichiers html n'est faite par le gestionnaire d'extensions de fichiers.

1.10.8 Inclure des outils externes

L'inclusion d'outils externes dans CodeBlocks est faisable via 'Outils' →'Configurer les outils' →'Ajouter' . Les variables internes (voir section 3.2 à la page 59) peuvent aussi être utilisées comme paramètres des outils. D'autre part, il y a plusieurs sortes d'options de lancement pour démarrer des applications externes. En fonction des options, les applications externes peuvent s'arrêter quand on quitte CodeBlocks. Si les applications doivent rester ouvertes après qu'on ait quitté CodeBlocks, l'option 'Lancer l'outil visible en mode détaché' doit être cochée.

1.11 Astuces pour travailler avec CodeBlocks

Dans ce chapitre nous présenterons quelques paramétrages utiles dans CodeBlocks.

1.11.1 Recherche de Modifications

CodeBlocks fourni une fonctionnalité pour pister les modifications effectuées dans un fichier source et affiche une barre dans la marge là où ont eût lieu les changements. Les

modifications sont marquées par une barre de changements jaune alors que celles qui ont déjà été enregistrées sont marquées par une barre de changements verte (voir Figure 1.11 à la page 15). Vous pouvez naviguer dans vos changements à l'aide du menu 'Rechercher' →'Aller à la ligne changée suivante' ou encore 'Rechercher' →'Aller à la ligne changée précédente' . La même fonctionnalité est accessible via les raccourcis clavier Ctrl-F3 et Ctrl-Shift-F3.

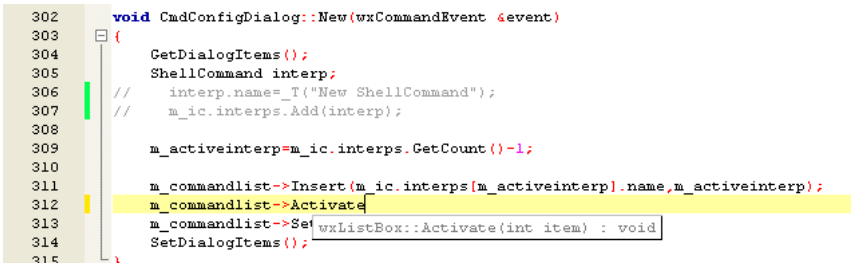


Figure 1.11: Recherche de modifications

Cette fonctionnalité peut être activée ou désactivée via la case à cocher 'Utiliser la barre de changements' dans le menu 'Paramètres' →'Éditeur' →'Marges et tirets' .

> **Note:**
>
> Si un fichier modifié est fermé, alors l'historique des changements tels que défaire/refaire ainsi que la barre de changements sont perdus. À l'aide du menu 'Édition' →'Effacer l'historique des changements' ou le menu de contexte correspondant vous pouvez effacer cet historique même si le fichier reste ouvert.

1.11.2 Échange de données avec d'autres applications

Les échanges de données entre CodeBlocks et d'autres applications sont possibles. Pour cela on utilise, avec Windows, le processus de communication inter processus DDE (Dynamic Data Exchange) et, avec les autres systèmes d'exploitation, une communication basée sur le protocole TCP.

Avec cette interface, différentes commandes peuvent être envoyées vers une instance de CodeBlocks en suivant la syntaxe suivante.

```
[<command>("<parameter>")]
```

Les commandes suivantes sont actuellement disponibles :

`Open` La commande

```
[Open("d:\temp\test.txt")]
```

utilise un paramètre, dans notre cas c'est le nom d'un fichier avec son chemin en absolu, et il s'ouvre dans une instance existante de CodeBlocks ou bien, si nécessaire, une première instance démarre.

`OpenLine` Cette commande ouvre un fichier dans une instance de CodeBlocks et se positionne sur la ligne dont le numéro est entré. Le numéro de ligne est spécifié par : ligne.

```
[OpenLine("d:\temp\test.txt:10")]
```

`Raise` Donne le "focus" à l'instance de CodeBlocks. Aucun paramètre ne doit être entré.

1.11.3 Configurer les variables d'environnement

La configuration d'un système d'exploitation se fait par ce qu'on appelle les variables d'environnement. Par exemple, la variable d'environnement PATH contient le chemin d'un compilateur installé. Le système d'exploitation analysera cette variable dans l'ordre d'écriture, c'est à dire que les entrées de la fin seront utilisées en dernier dans les recherches. Si plusieurs versions de compilateur ou d'autres applications sont installées, les situations suivantes peuvent se produire :

- On appelle une version incorrecte d'un logiciel
- Les logiciels installés s'appellent entre eux

Ainsi, on peut tomber sur le cas où différentes versions d'un compilateur ou d'un autre outil sont obligatoires pour différents projets. Lorsque cela arrive, une première solution est de changer les variables d'environnement dans le système d'exploitation pour chaque projet. Toutefois cette procédure est sujette à erreur et manque de flexibilité. Pour ce faire, CodeBlocks offre une solution élégante. Différentes configurations de variables peuvent être créées pour un usage uniquement en interne à CodeBlocks. De plus, vous pouvez passer de l'une à l'autre de ces configurations. La Figure 1.12 à la page 17 montre la boîte de dialogue que vous obtenez via 'Variables d'Environnement' dans 'Paramètres' →'Environnement' . On crée une configuration à l'aide du bouton 'Créer'.

L'accès et l'étendue des variables d'environnement ainsi créées sont limités à CodeBlocks. Vous pouvez étendre ces variables d'environnement comme toutes les autres variables dans CodeBlocks à l'aide de $(NAME).

> **Note:**
>
> La configuration d'une variable d'environnement pour chaque projet peut être sélectionnée dans le menu de contexte 'Propriétés' de l'onglet 'Options EnvVars'.

Exemple

Vous pouvez écrire dans un fichier <project>.env l'environnement utilisé dans une étape de post génération (voir section 1.6 à la page 5) puis l'archiver dans votre projet.

```
cmd /c echo \%PATH\%  > project.env
```

ou sous Linux

```
echo \$PATH > project.env
```

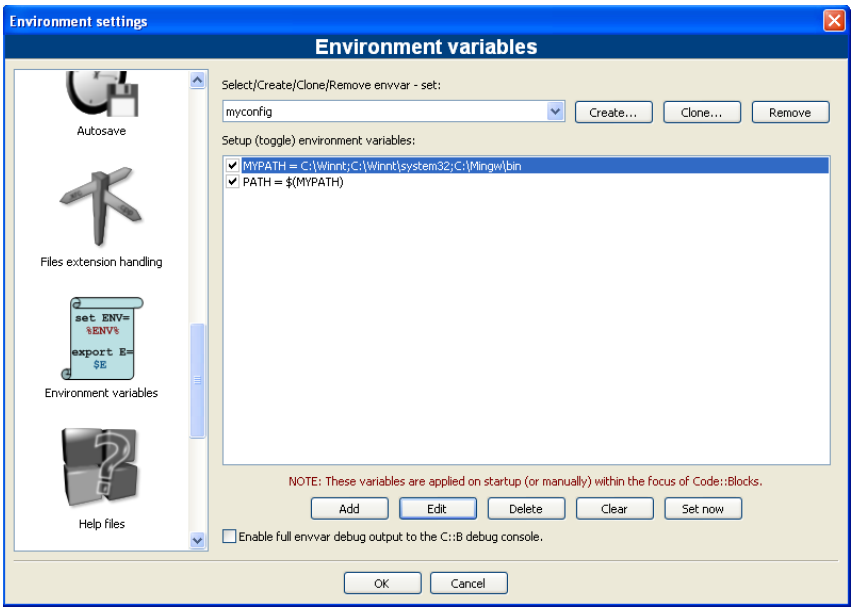


Figure 1.12: Variables d'environnement

1.11.4 Basculer entre diverses dispositions

En fonction des tâches à effectuer, il peut être utile d'avoir plusieurs configurations ou dispositions (ou présentations) différentes de CodeBlocks et de les sauvegarder. Par défaut, le paramétrage (notamment afficher/masquer les barres d'outils, aspect, etc.) est enregistré dans le fichier de configuration `default.conf`. En utilisant l'option en ligne de commande `--personality=ask` au démarrage de CodeBlocks, on peut choisir parmi plusieurs possibilités de paramétrages. En dehors de ces paramétrages globaux, il peut se produire une situation où vous souhaitez basculer entre différentes vues de fenêtres ou de barres de symboles pendant une session. L'édition de fichier et le débogage de projets sont deux exemples typiques de telles situations. CodeBlocks offre un mécanisme pour enregistrer et sélectionner différentes dispositions afin d'éviter à l'utilisateur d'avoir à fermer et ouvrir manuellement et fréquemment des fenêtres et des barres de symboles. Pour enregistrer une disposition, sélectionnez le menu 'Vue' →'Disposition' →'Enregistrer la disposition actuelle' et entrez un nom dans <nom>. La commande 'Paramètres' →'Éditeur' →'Raccourcis clavier' →'Vue' →'Dispositions' →'<name>' permet de définir un raccourci clavier pour ce processus. Il est ainsi possible de basculer entre les diverses dispositions simplement en utilisant ces raccourcis clavier.

> **Note:**
>
> Autre exemple : éditer un fichier en mode plein écran sans barre de symboles. Vous pouvez créer une disposition comme 'Plein Ecran' et lui assigner un raccourci spécifique.

1.11.5 Basculer entre projets

Si plusieurs projets ou fichiers sont ouverts en même temps, l'utilisateur a besoin d'un moyen pour passer rapidement de l'un à l'autre. CodeBlocks possède plusieurs raccourcis

pour ce faire.

Alt-F5 Active le projet précédent de la vue des projets.

Alt-F6 Active le projet suivant de la vue des projets.

F11 Dans l'éditeur, bascule entre un fichier source <`name`>`.cpp` et le fichier d'en-tête (header) correspondant <`name`>`.h`

1.11.6 Configurations étendue des compilateurs

Lors de la génération d'un projet, les messages du compilateur sont affichés dans l'onglet Messages de génération. Si vous souhaitez recevoir des informations détaillées, l'affichage peut être étendu. Pour cela, cliquez sur 'Paramètres' →'Compilateur et débogueur' puis sélectionnez l'onglet 'Autres options' dans le menu déroulant.

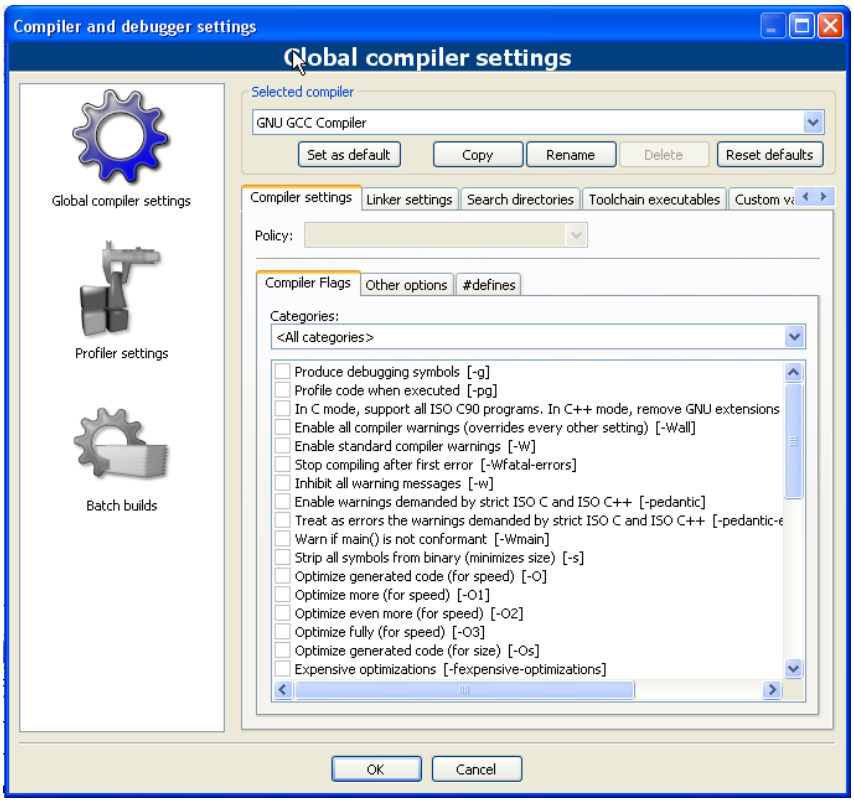


Figure 1.13: Configurer des informations détaillées

Assurez vous que le compilateur soit correctement sélectionné. L'option 'Ligne de commande complète' des Avertissements du compilateur permet de sortir des informations détaillées. De plus, ces sorties peuvent être redirigées vers un fichier HTML. Pour cela, sélectionnez 'Enregistrer le journal de génération dans un fichier HTML en fin de génération'. D'autre part, CodeBlocks peut afficher une barre d'avancement du processus de génération dans la fenêtre de génération qui peut être activée en cochant 'Afficher la barre de progression de génération'.

1.11.7 Zoomer dans l'éditeur

CodeBlocks possède un éditeur très puissant. Cet éditeur vous permet de changer la taille des caractères du texte affiché des fichiers ouverts. Si vous avez une souris avec une molette, vous n'avez qu'à appuyer sur la touche Ctrl tout en tournant la molette dans un sens ou l'autre pour agrandir ou réduire la taille du texte.

> **Note:**
>
> Avec le raccourci Ctrl-Numepad-/ ou à l'aide du menu 'Edition' →'Commandes spéciales' →'Zoom' →'Remise à 0' vous restaurez la taille originale du texte courant.

1.11.8 Mode de Repliement

Quand on édite des fichiers de texte, notamment des `*.txt`, dans CodeBlocks, il peut être utile d'avoir le texte replié, ce qui signifie que les lignes longues seront affichées sur plusieurs lignes à l'écran afin qu'elles puissent être correctement éditées. La fonction 'Repliement' peut être activée dans 'Paramètres' →'Éditeur' →'Autres Options' ou en cochant la case 'Activer le repliement' . Les touches "Home" et "Fin" positionnent respectivement le curseur en début et en fin de ligne repliée. Quand on choisi 'Paramètres' →'Éditeur' →'Autres Options' et 'La touche Home déplace toujours le curseur en première colonne' , le curseur sera positionné respectivement en début ou en fin de ligne si on appuie sur la touche "Home" ou "Fin". Si on désire placer le curseur au début de la première ligne du paragraphe en cours, il vous faut utiliser la combinaison de touches 'Alt-Home' . La même chose de façon analogue pour 'Alt-Fin' pour positionner le curseur en fin de la dernière ligne du paragraphe courant.

1.11.9 Sélection de modes dans l'éditeur

CodeBlocks supporte différents modes de sélection pour le couper-coller des chaînes de caractères.

1. Un texte de l'éditeur actif peut être sélectionné avec le bouton gauche de la souris, puis on relâche ce bouton. L'utilisateur peut se déplacer de haut en bas avec la molette de la souris. Si on appuie sur le bouton du milieu, le texte précédemment sélectionné sera inséré. Cet effet est disponible au niveau d'un fichier et peut être vu comme un presse papier de fichier.

2. Appuyer sur la touche 'ALT' active ce qu'on appelle la sélection en mode bloc et un rectangle de sélection s'affiche à l'aide du bouton gauche de la souris. Lorsqu'on relâche la touche Alt cette sélection peut être copiée ou collée. Cette option est utile si vous voulez sélectionner des colonnes, notamment dans un tableau et en copier-coller le contenu.

3. Dans le menu 'Paramètres' →'Éditeur' →'Marges et tirets' on peut activer ce qu'on appelle des 'Espaces Virtuels' . Ceci active la possibilité d'avoir une sélection en mode bloc qui peut commencer ou se terminer par une ligne vide.

4. Dans le menu 'Paramètres' →'Éditeur' →'Marges et tirets' on peut activer les 'Sélections Multiples' . En maintenant enfoncée la touche Ctrl l'utilisateur peut sélectionner diverses lignes dans l'éditeur actif avec le bouton gauche de la souris. Les sélections sont ajoutées dans le presse papier à l'aide des raccourcis Ctrl-C ou Ctrl-X. Ctrl-V en insèrera le contenu à la position courante du curseur. Une option complémentaire dénommée 'Active l'entrée clavier (et la suppression)' peut être activée pour les sélections multiples. Cette option est utile si vous voulez ajouter des directives de pré-processeur comme **#ifdef** sur plusieurs lignes de code source ou si vous voulez superposer ou remplacer du texte en plusieurs endroits.

> **Note:**
>
> La plupart des gestionnaires de fenêtres de Linux utilisent ALT-ClicGaucheDéplacer pour déplacer une fenêtre, aussi vous devrez désactiver cette fonctionnalité pour pouvoir sélectionner en mode bloc.

1.11.10 Repliement de code

CodeBlocks supporte ce qu'on appelle le repliement de code. Avec cette fonctionnalité vous pouvez replier notamment les fonctions dans l'éditeur de CodeBlocks. Un point de repliement est marqué dans la marge gauche de l'éditeur par un signe moins. Dans la marge, le début et la fin d'un point de repliement sont visibles à l'aide d'une ligne verticale. Si vous cliquez sur le signe moins avec le bouton gauche de la souris, la portion de code sera repliée ou dépliée. Via le menu 'Edition' →'Repliement' vous pouvez sélectionner le repliement. Dans l'éditeur, un code replié est vu comme une ligne horizontale continue.

> **Note:**
>
> Le style de repliement et la profondeur limite du repliement peuvent se configurer dans le menu 'Paramètres' →'Éditeur' →'Repliement' .

CodeBlocks fournit aussi la fonctionnalité de repliement pour les directives du pré-processeur. Pour l'activer, sélectionnez 'Replier les commandes du pré-processeur' dans l'entrée Repliement de 'Paramètres' →'Éditeur' .

Une autre façon de faire est de définir des points de repliement utilisateurs. Le point de départ du repliement s'entre comme un commentaire suivi d'une parenthèse ouvrante et la fin comme un commentaire suivi d'une parenthèse fermante.

```
//{
code avec repliement défini par l'utilisateur
//}
```

1.11.11 Auto complétion

Lorsque vous ouvrez un projet dans CodeBlocks les 'Répertoires de recherche' de votre compilateur et de votre projet, les fichiers sources et d'en-têtes de votre projet sont analysés. De plus les mots clés de l'analyseur syntaxique correspondant sont analysés

également. Les informations issues de l'analyse sont alors utilisées pour la fonctionnalité d'auto complétion dans CodeBlocks. Vérifiez s'il vous plait que cette fonctionnalité est bien activée dans l'éditeur. L'auto complétion est accessible au travers du raccourci Ctrl-Espace. Via le menu 'Paramètres' →'Éditeur' →'Colorisation syntaxique' vous pouvez ajouter des mots clés définis par l'utilisateur à votre analyseur syntaxique.

1.11.12 Recherche de fichiers cassés

Lorsqu'un fichier est supprimé du disque, mais est toujours inclus dans un fichier projet <project>.cbp, alors un 'fichier cassé' sera affiché avec un symbole "cassé" dans la vue du projet. Vous devriez utiliser 'Enlever ce fichier du projet' plutôt que de supprimer le fichier.

Dans de gros projets, avec de nombreux sous-répertoires, la recherche de fichiers cassés peut être une grande consommatrice de temps. Avec l'extension ThreadSearch (voir section 2.6 à la page 34) CodeBlocks apporte une solution simple à ce problème. Si vous entrez une expression de recherche dans ThreadSearch et sélectionnez l'option 'Fichiers du projet' ou 'Fichiers de l'espace de travail' , alors ThreadSearch analysera tous les fichiers qui sont inclus dans le projet ou l'espace de travail. Si un fichier cassé est trouvé, ThreadSearch génèrera une erreur sur le fichier absent.

1.11.13 Inclure des librairies

Dans les options de génération d'un projet, vous pouvez ajouter les librairies utilisées via le bouton 'Ajouter' dans l'entrée 'Librairies à lier' des 'Options de l'éditeur de liens'. Ce faisant, vous pouvez soit utiliser le chemin absolu de la librairie ou seulement donner son nom sans le préfixe `lib` ni l'extension du fichier.

Exemple

Pour une librairie nommée <path>\libs\lib<name>.a, écrire seulement <name>. L'éditeur de liens avec les chemins de recherche correspondants inclura alors correctement les librairies.

> **Note:**
>
> Une autre façon d'inclure des librairies est documentée dans la section 2.10 à la page 43.

1.11.14 Ordre d'édition de liens des fichiers objets

Lors de la compilation, les fichiers objets `name.o` sont créés à partir des sources `name.c/cpp`. L'éditeur de liens assemble les fichiers objets individuels pour en faire une application `name.exe` ou sur d'autre systèmes `name.elf`. Dans certains cas, il peut être préférable de prédéfinir l'ordre dans lequel seront liés les fichiers objets. Vous pouvez obtenir cela dans CodeBlocks en assignant des priorités. Dans le menu de contexte 'Propriétés' , vous pouvez définir les priorités d'un fichier dans l'onglet Générer. Une priorité faible fera que le fichier sera lié plus tôt.

1.11.15 Sauvegarde automatique

CodeBlocks offre la possibilité d'enregistrer automatiquement les projets et les fichiers sources, ou encore de créer des copies de sauvegarde. Cette fonctionnalité peut être activée dans le menu 'Paramètres' →'Environnement' →'Sauvegarde-auto' . Ce faisant, 'Enregistrer dans un fichier .save' doit être spécifié comme méthode de création de copie de sauvegarde.

1.11.16 Configuration des extensions de fichiers

Dans CodeBlocks, vous pouvez choisir entre plusieurs méthodes de traitement des extensions de fichiers. La boîte de dialogue de configuration s'ouvre par 'Paramètres' →'Gestion des extensions de fichiers' . Vous pouvez alors soit utiliser les applications assignées par Windows pour chaque extension de fichier (l'ouvrir avec l'application associée), ou changer la configuration pour chaque extension de telle façon que ce soit un programme défini par l'utilisateur qui soit lancé (lancer un programme externe), ou que ce soit CodeBlocks qui ouvre le fichier dans son éditeur (l'ouvrir dans l'éditeur de Code::Blocks).

> **Note:**
>
> Si un programme utilisateur est associé à une certaine extension de fichier, la configuration 'Désactiver Code::Blocks quand un programme externe est lancé' devrait être désactivée, sinon CodeBlocks sera fermé dès qu'un fichier qui possède cette extension est ouvert.

1.12 CodeBlocks en ligne de commande

L'Environnement de Développement Intégré (IDE) CodeBlocks peut être exécuté depuis une ligne de commande sans interface graphique. Dans ce cas, plusieurs options sont disponibles pour contrôler le processus de génération d'un projet. Comme CodeBlocks peut être piloté par des "scripts", la création d'exécutables peut être intégrée dans vos propres processus de travail.

```
codeblocks.exe /na /nd --no-splash-screen --built <name>.cbp --target='Release'
```

`<filename>` Spécifie le nom du fichier de projet `*.cbp` ou le nom de l'espace de travail `*.workspace`. Par exemple, <filename> peut être `project.cbp`. Placez cet argument en fin de ligne de commande, juste avant la redirection de la sortie, s'il y en a une.

`--file=<filename>[:ligne]`
Ouvrir un fichier dans Code::Blocks et, en option, se positionner sur une ligne particulière.

`/h, --help` Affiche un message d'aide concernant les arguments en ligne de commande.

`/na, --no-check-associations`
Ne faire aucun contrôle d'association de fichiers (Windows seulement).

`/nd, --no-dde` Ne pas lancer le serveur DDE (Windows seulement).

`/ni, --no-ipc` Ne pas lancer le serveur IPC (Linux et Mac seulement).

`/ns, --no-splash-screen`
Ne pas afficher l'écran de démarrage pendant le chargement de l'application.

`/d, --debug-log`
Afficher le journal de débogage de l'application.

`--prefix=<str>`
Configure le préfixe du répertoire de données partagées.

`/p, --personality=<str>, --profile=<str>`
Configure le profil (ou personnalité) à utiliser. Vous pouvez utiliser le paramètre ask pour afficher la liste de tous les profils disponibles.

`--rebuild` Nettoie et génère le projet ou l'espace de travail.

`--build` Génère le projet ou l'espace de travail.

`--target=<str>`
Configure la cible de génération. Par exemple `--target='Release'`.

`--no-batch-window-close`
Garde la fenêtre batch de journalisation visible après que la génération par batch soit terminée.

`--batch-build-notify`
Affiche un message une fois que la génération batch est terminée.

`--safe-mode` Désactive toutes les extensions (plugins) au démarrage.

`> <build log file>`
Placé en toute dernière position d'une ligne de commande, ceci permet à l'utilisateur de rediriger la sortie standard vers un fichier log. Ceci n'est pas à proprement parler une option de codeblocks, mais seulement une redirection standard des sorties des shells DOS/*nix.

1.13 Raccourcis

Même si une IDE comme CodeBlocks est surtout pilotée à la souris, les raccourcis claviers sont néanmoins moyen très pratique pour accélérer et simplifier le travail. Les tableaux ci-dessous regroupent quelques-uns des raccourcis claviers disponibles.

1.13.1 Éditeur

Fonction	Raccourci clavier
Défaire la dernière action	Ctrl-Z
Refaire la dernière action	Ctrl-Shift-Z
Permuter en-têtes / source	F11
Commenter le code surligné	Ctrl-Shift-C
Décommenter le code surligné	Ctrl-Shift-X
Auto-complète / Abréviations	Ctrl-Space/Ctrl-J
Bascule la marque	Ctrl-B
Aller à la marque précédente	Alt-PgUp
Aller à la marque suivante	Alt-PgDown

Ceci est une liste des raccourcis fournis pas le composant éditeur de CodeBlocks. Ces raccourcis ne peuvent pas être substitués.

Fonction	Raccourci clavier
Créer ou supprimer un signet	Ctrl-F2
Aller au signet suivant	F2
Sélectionner jusqu'au signet suivant	Alt-F2
Rechercher la sélection.	Ctrl-F3
Rechercher la sélection en arrière.	Ctrl-Shift-F3
Recherche des conditions concordantes du préprocesseur, passer les imbriquées	Ctrl-K

1.13.2 Files

Fonction	Raccourci clavier
Nouveau fichier ou projet	Ctrl-N
Ouvrir un fichier ou un projet existant	Ctrl-O
Enregistrer le fichier courant	Ctrl-S
Enregistrer tous les fichiers	Ctrl-Shift-S
Fermer le fichier courant	Ctrl-F4/Ctrl-W
Fermer tous les fichiers	Ctrl-Shift-F4/Ctrl-Shift-W

1.13.3 Vue

Fonction	Raccourci clavier
Afficher / masquer le panneau de Messages	F2
Afficher / masquer le panneau de Gestion	Shift-F2
Activer le précédent (dans l'arbre des projets)	Alt-F5
Activer le suivant (dans l'arbre des projets)	Alt-F6

1.13.4 Recherche

Fonction	Raccourci clavier
Rechercher	Ctrl-F
Rechercher le suivant	F3
Rechercher le précédent	Shift-F3
Rechercher dans les fichiers	Crtl-Shift-F
Remplacer	Ctrl-R
Remplacer dans les fichiers	Ctrl-Shift-R
Aller à la ligne	Ctrl-G
Aller à la ligne changée suivante	Ctrl-F3
Aller à la ligne changée précédente	Ctrl-Shift-F3
Aller au fichier	Alt-G
Aller à la fonction	Ctrl-Alt-G
Aller à la fonction précédente	Ctrl-PgUp
Aller à la fonction suivante	Ctrl-PgDn
Aller à la déclaration	Ctrl-Shift-.
Aller à l'implémentation	Ctrl-.
Ouvrir le fichier inclus	Ctrl-Alt-.

1.13.5 Générer

Fonction	Raccourci clavier
Générer	Ctrl-F9
Compiler le fichier courant	Ctrl-Shift-F9
Exécuter	Ctrl-F10
Générer et exécuter	F9
Re-générer	Ctrl-F11

2 Extensions

2.1 Astyle

Artistic Style est un indentateur de code source, un formateur de code source, et rend le code source des langages de programmation C, C++, C# plus beau. On peut l'utiliser pour sélectionner différents styles de règles de codage dans CodeBlocks.

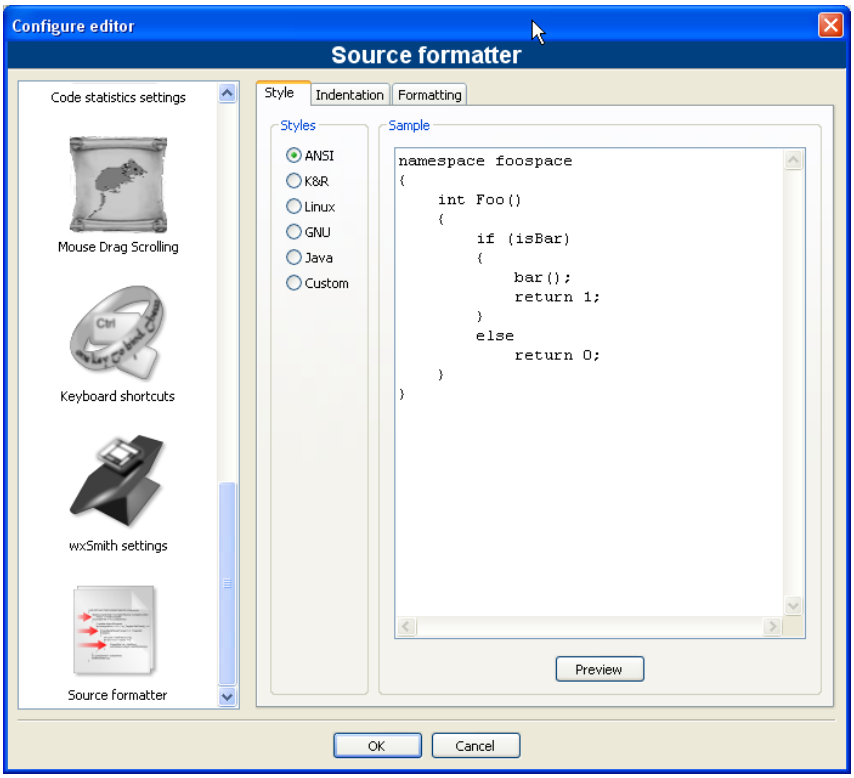


Figure 2.1: Formater votre code source

Quand on indente un code source, nous en tant que programmeurs avons tendance à utiliser à la fois des espaces et des caractères de tabulations pour créer l'indentation souhaitée. De plus, certains éditeurs insèrent par défaut des espaces à la place des tabulations quand on appuie sur la touche Tab, alors que d'autres éditeurs ont la faculté de rendre les lignes plus belles en ajoutant automatiquement des espaces en début de lignes, éventuellement en remplaçant dans ce code les tabulations utilisées jusqu'alors pour l'indentation par des espaces.

Comme le nombre de caractères affichés sur l'écran pour chaque caractère de tabulation change d'un éditeur à l'autre, un des problèmes courants auquel est confronté un programmeur qui passe d'un éditeur à un autre est qu'un code qui contient à la fois des espaces et des tabulations et qui était jusqu'à présent bien indenté, devient soudain difficile à regarder après le changement d'éditeur. Même si en tant que programmeur vous

faites attention à n'utiliser QUE des espaces ou QUE des tabulations, récupérer un code de quelqu'un d'autre peut malgré tout être problématique.

C'est pour résoudre ce problème qu'Artistic Style a été créé - un filtre écrit en C++ qui ré-indente et reformate automatiquement les fichiers sources en C / C++ / C#.

> **Note:**
>
> Quand vous copiez du code, par exemple depuis Internet ou d'un manuel, ce code sera automatiquement adapté aux règles de codage dans CodeBlocks.

2.2 CodeSnippets

L'extension CodeSnippets permet de structurer des modules de texte et des liens vers des fichiers en fonction de catégories dans une vue arborescente. Les modules sont utilisés pour stocker des fichiers fréquemment utilisés, des constructions de modules de texte, le tout géré depuis un endroit centralisé. Imaginez la situation suivante : Un certain nombre de fichiers source fréquemment utilisés sont stockés dans divers répertoires du système de fichiers. La fenêtre de CodeSnippets vous donne l'opportunité de créer des catégories et, à l'intérieur de ces catégories, des liens vers les fichiers requis. Avec cette fonctionnalité, vous pouvez contrôler l'accès aux fichiers indépendamment de l'endroit où ils sont stockés dans le système de fichiers, et vous pouvez rapidement naviguer entre ces fichiers sans avoir besoin de les chercher un peu partout dans le système.

> **Note:**
>
> Vous pouvez utiliser les variables CodeBlocks ou les variables d'environnement dans les liens vers les fichiers comme `$(VARNAME)/name.pdf` pour paramétrer un lien dans le navigateur de CodeSnippets.

La liste des modules de texte et des liens peut être enregistrée dans la fenêtre des CodeSnippets en cliquant sur le bouton droit de la souris et en sélectionnant 'Enregistrer l'index' depuis le menu de contexte. Le fichier `codesnippets.xml` qui est alors créé par cette procédure, se trouve dans le sous-répertoire `codeblocks` du répertoire `Documents and Settings\Application data`. Sous Linux, cette information est enregistrée dans le sous-répertoire `.codeblocks` de votre répertoire HOME. Les fichiers de configuration de CodeBlocks seront chargés au démarrage suivant. Si vous souhaitez enregistrer le contenu des CodeSnippets à un autre endroit, sélectionnez l'entrée 'Enregistrer l'index sous'. Pour charger ce fichier, sélectionnez 'Charger le fichier d'index' lors du démarrage suivant de CodeBlocks ou incluez le répertoire dans les 'Paramètres' du menu de contexte de 'Répertoire des Snippets'. Les paramétrages sont enregistrés dans le fichier correspondant `codesnippets.ini` dans votre application data.

Pour inclure une catégorie, utilisez le menu 'Ajouter une sous-catégorie'. Une catégorie peut contenir des Snippets (modules de texte) ou des Liens vers un fichier. Un module de texte est créé via la commande 'Ajouter un Snippet' depuis le menu de contexte. Le

contenu est intégré dans le module de texte comme un 'Nouveau snippet' en sélectionnant un passage de texte dans l'éditeur de CodeBlocks et en le glissant-déposant sur le module dont les propriétés s'affichent. En double cliquant sur la nouvelle entrée ou en sélectionnant 'Éditer le Texte' on en éditera le contenu.

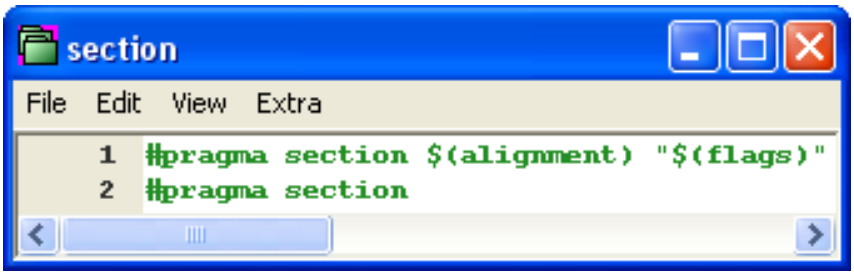


Figure 2.2: Édition d'un module de texte

La sortie d'un module de texte est gérée dans CodeBlocks via la commande 'Appliquer' du menu de contexte ou en faisant un glisser-déposer dans l'éditeur. Sous Windows, le contenu d'un Snippet peut également être glissé-déposé dans d'autres applications. Dans le navigateur de CodeSnippets vous pouvez copier une sélection par glisser-déposer vers une catégorie différente.

De plus, les modules de texte peuvent être paramétrés par des variables <name> qui peuvent être accédées via $(name) (voir Figure 2.2 à la page 28). Les valeurs des variables peuvent être récupérées dans un champ d'entrée si le module de texte est appelé via la commande du menu de contexte 'Appliquer'.

À côté des modules de texte, des liens vers des fichiers peuvent aussi être créés. Si, après avoir créé un module de texte, vous cliquez sur la commande 'Propriétés' du menu de contexte, vous pouvez alors sélectionner une cible de type lien en cliquant sur le bouton 'Lien cible'. Cette procédure convertira automatiquement le module de texte en un lien vers un fichier. Dans CodeSnippets, tous les modules de texte sont marqués par un symbole T, les liens vers un fichier par un symbole F et les urls par un symbole U. Si vous voulez ouvrir un fichier sélectionné (lien) dans la vue des codesnippets, sélectionnez tout simplement le menu de contexte 'Ouvrir le fichier' ou tout en maintenant enfoncée la touche 'Alt' effectuez un double clic sur le fichier.

Note:

Vous pouvez même ajouter une url (comme http://www.codeblocks.org) dans les modules de texte. L'url peut être ouverte en utilisant le menu de contexte 'Ouvrir l'Url' ou en utilisant un glisser-déposer vers votre navigateur favori.

Avec un tel paramétrage, si vous ouvrez un lien vers un fichier pdf depuis la vue des codesnippets, un visualiseur de fichiers pdf sera automatiquement démarré. Cette méthode rend possible à l'utilisateur l'accès à des fichiers répartis un peu partout sur le réseau, comme des données, mises en forme, documentations etc., à l'aide des applications communes, simplement par le biais d'un lien. Le contenu des codesnippets est enregistré dans le

fichier `codesnippets.xml`, la configuration est enregistrée dans le fichier `codesnippets.ini` de votre répertoire `application data`. Ce fichier ini contiendra, par exemple, le chemin du fichier `codesnippets.xml`.

CodeBlocks supporte l'utilisation de différents profils. Ces profils sont aussi nommés personnalités. En démarrant, avec l'option `--personality=<profile>`, CodeBlocks en ligne de commande vous créez ou utilisez un profil existant. Dans ce cas, le paramétrage ne sera pas enregistré dans le fichier `default.conf`, mais plutôt dans un `<personality>.conf` de votre répertoire `application data`. L'extension Codesnippets enregistrera alors ses paramètres dans un fichier `<personality>.codesnippets.ini`. Maintenant, si vous chargez un nouveau contenu `<name.xml>` dans les paramètres de codesnippets via 'Charger un fichier d'index', ce contenu sera enregistré dans le fichier ini correspondant. L'avantage de cette méthode tient dans le fait que dans le cas où il y a différents profils, on peut gérer plusieurs configurations de modules de textes et de liens.

L'extension offre une fonction de recherche complémentaire pour naviguer dans les catégories et les Snippets. La façon de rechercher dans les Snippets, catégories ou Snippets et catégories peut s'ajuster. En entrant l'expression de recherche requise, l'entrée correspondante est automatiquement sélectionnée dans la vue. La Figure 2.3 à la page 29 affiche une fenêtre CodeSnippets typique.

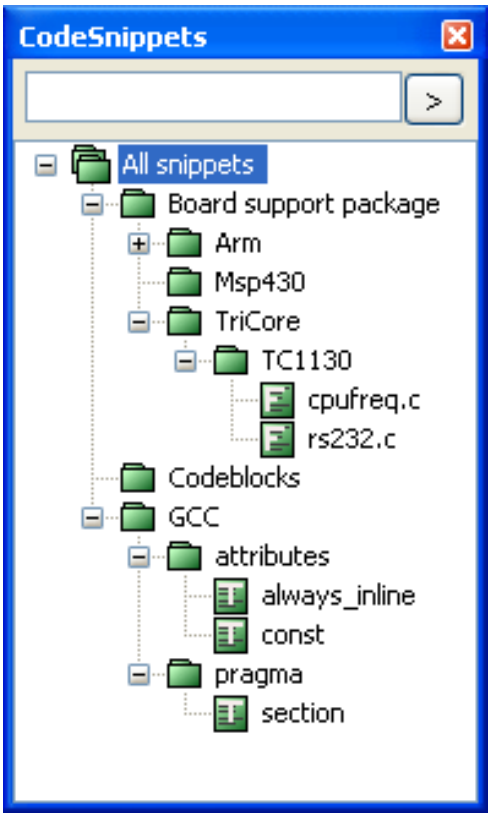
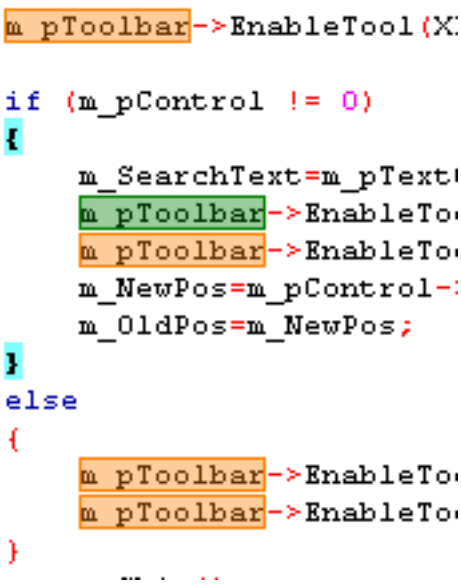


Figure 2.3: Vue des CodeSnippets

> **Note:**
>
> Quand on utilise de volumineux modules de texte, le contenu de ces modules devrait être enregistré sous forme de fichiers via 'Convertir en lien vers fichier' de façon à réduire l'utilisation mémoire du système. Si vous supprimez un codesnippet ou un lien vers un fichier il est en fait déplacé vers une catégorie `.trash`; si vous maintenez enfoncée la touche Maj cet élément sera réellement détruit.

2.3 Recherche Incrémentale

Pour obtenir une recherche efficace dans des fichiers ouverts, CodeBlocks fourni ce qu'on nomme une recherche incrémentale. Cette méthode de recherche s'initialise, pour un fichier ouvert, via le menu 'Rechercher' →'Recherche Incrémentale' ou par le raccourci clavier Ctrl-I. L'entrée active passe alors automatiquement à la configuration du masque de recherche dans la barre d'outil correspondante. Dès que vous commencez à entrer des termes de recherche, le fond du masque de recherche s'ajuste en fonction des occurrences des termes. Si un accord est trouvé dans l'éditeur actif, la position respective est marquée en couleur. Par défaut l'accord courant est surligné en vert. Cette configuration peut être changée dans 'Paramètres' →' Éditeur' →' Recherche Incrémentale' (voir Figure 2.4 à la page 31). En appuyant sur la touche Entrée la recherche saute à l'occurrence suivante de la chaîne de texte recherchée à l'intérieur du fichier. Avec Maj-Entrée, c'est l'occurrence précédente qui est sélectionnée. Cette fonctionnalité n'est pas supportée par Scintilla si la recherche incrémentale utilise des expressions régulières.



Si la chaîne de caractère recherchée ne peut pas être trouvée dans le fichier courant, afin d'indiquer que c'est ce qui se passe, le fond du masque de recherche est affiché en rouge.

ESC Quitte le module de Recherche Incrémentale.

ALT-Suppr Efface l'entrée du champ de recherche incrémentale.

Les icônes de la barre d'outil de Recherche Incrémentale ont les significations suivantes :

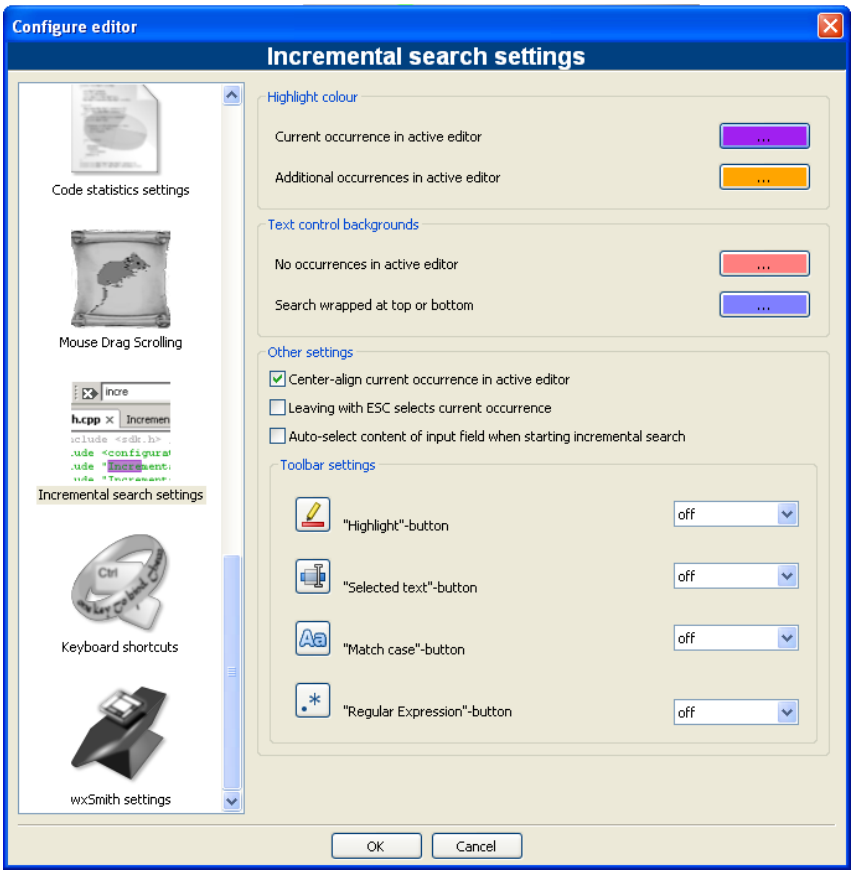


Figure 2.4: Paramètres pour la Recherche Incrémentale

 Suppression du texte dans le masque de recherche de la barre d'outils de Recherche Incrémentale.

 Navigation dans les occurrences de chaîne recherchée.

 En cliquant sur ce bouton ce sont toutes les occurrences de la chaîne recherchée qui sont surlignées en couleur, pas seulement la première.

 Activer cette option réduit le champ de recherche au passage de texte marqué dans l'éditeur.

 Cette option signifie que la recherche sera sensible à la casse (respect des majuscules et minuscules).

 Valider les expressions régulières dans le champ d'entrée de la recherche incrémentale.

Note:

Le paramétrage standard de cette barre d'outil peut être configuré dans 'Paramètres' →'Éditeur' →'Recherche Incrémentale' .

2.4 Liste des "à faire"

Dans des projets logiciels complexes, où différents développeurs sont impliqués, il est souvent nécessaire que différentes tâches soient effectuées par plusieurs utilisateurs. Pour cela, CodeBlocks possède une Liste des "à faire". Cette liste s'ouvre via 'Vue' →'Liste des "A faire"' , et contient les tâches à effectuer ensemble, avec leurs priorités, le type et le responsable de la tâche. On peut filtrer la liste par tâches, utilisateurs et/ou fichiers sources. Un tri par colonnes peut être effectué en cliquant sur le titre de la colonne correspondante.

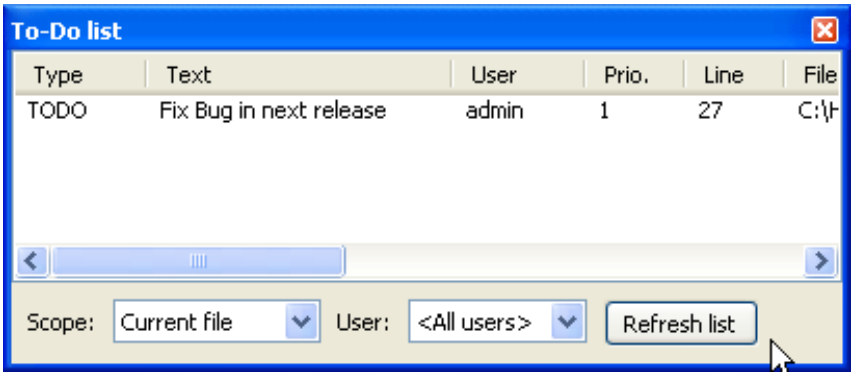


Figure 2.5: Affichage de la Liste des "A faire"

> **Note:**
>
> La liste des "à faire" peut être ajoutée à la console de messages. Sélectionnez l'option 'Inclure la liste des "A faire" dans le panneau de messages' à l'aide du menu 'Paramètres' →'Environnement' .

Si les fichiers sources sont ouverts dans CodeBlocks, un "à faire" peut être ajouté à la liste via la commande 'Ajouter un élement "à faire"' du menu de contexte. Un commentaire est ajouté dans le code sur la ligne sélectionnée.

```
// TODO (user#1#): ajouter un nouveau dialogue pour la prochaine release
```

Quand on ajoute un "à faire", une boîte de dialogue apparaît où les paramétrages suivants peuvent être faits (voir Figure 2.6 à la page 33).

Utilisateur Nom de l'utilisateur <user> pour le système d'exploitation. Les tâches pour d'autres utilisateurs peuvent également être créées ici. Pour cela, le nom de l'utilisateur correspondant doit être créé par Ajouter un nouvel utilisateur. L'assignation d'un "à faire" est alors faite via une sélection d'entrées pour cet utilisateur.

> **Note:**
>
> Notez que les Utilisateurs ici n'ont rien à voir avec les profils (ou personnalités) utilisés dans CodeBlocks.

Type Par défault, le type est TODO (à faire").

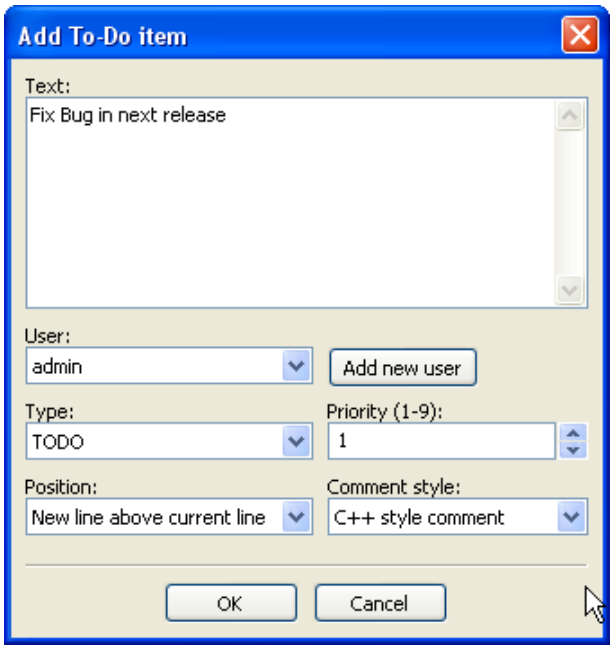


Figure 2.6: Dialogue pour ajouter un "à faire"

Priorité Dans CodeBlocks, l'importance de la tâche peut être exprimée par des priorités (1 - 9).

Position Ce paramètre spécifie si le commentaire doit être inclus avant, après ou bien à la position exacte du curseur.

Style de commentaire Une sélection de formats de commentaires (notamment doxygen).

2.5 Exporter du code Source

Il est souvent nécessaire de transférer du code source vers d'autres applications ou vers des e-mails. Si le texte est simplement copié, le formattage est perdu, ce qui rend le texte peu clair. La fonction exporter de CodeBlocks est une des solutions dans ce type de situations. Le format requis pour le fichier exporté peut être sélectionné via 'Fichier' →'Exporter' . Le programme adoptera alors le nom de fichier et le répertoire cible en fonction du fichier source ouvert et les proposera pour enregistrer le fichier à exporter. L'extension de fichier appropriée à chaque cas de figure sera déterminé par le type de l'exportation. Les formats suivants sont disponibles :

html Un format de type texte qui peut être affiché dans un navigateur web ou dans un traitement de texte.

rtf Le format Rich Text qui est un format basé sur du texte et qui peut être ouvert dans un traitement de texte comme Word ou OpenOffice.

odt Le format Open Document Text qui est un format standardisé spécifié par Sun et O'Reilly. Ce format peut être traité par Word, OpenOffice et d'autres traitements de texte.

pdf Le format Portable Document qui peut être ouvert par des applications comme Acrobat Reader.

2.6 Thread Search

Via le menu 'Rechercher' →'Thread Search' , cette extension peut être affichée ou masquée en tant qu'onglet dans la console de messages. Dans CodeBlocks, une prévisualisation de l'occurrence de la chaîne de caractères peut être affichée pour un fichier, un espace de travail ou un répertoire. Ce faisant, la liste des résultats de la recherche sera affichée sur la partie droite de la console ThreadSearch. En cliquant sur une entrée de la liste, une prévisualisation s'affiche sur la partie gauche. En double cliquant dans la liste, le fichier sélectionné est ouvert dans l'éditeur de CodeBlocks.

> **Note:**
>
> L'étendue des extensions de fichiers à inclure dans la recherche est préconfiguré et peut avoir besoin d'être ajusté.

2.6.1 Fonctionnalités

L'extension ThreadSearch offre les fonctionnalités suivantes :

- 'Recherche dans les fichiers' multi tâches (Multi-threaded).
- Éditeur interne en lecture seule pour voir les résultats
- Fichier ouvert dans l'éditeur de type notebook
- Menu contextuel 'Rechercher les occurrences' pour commencer une recherche dans les fichiers à partir du mot situé sous le curseur

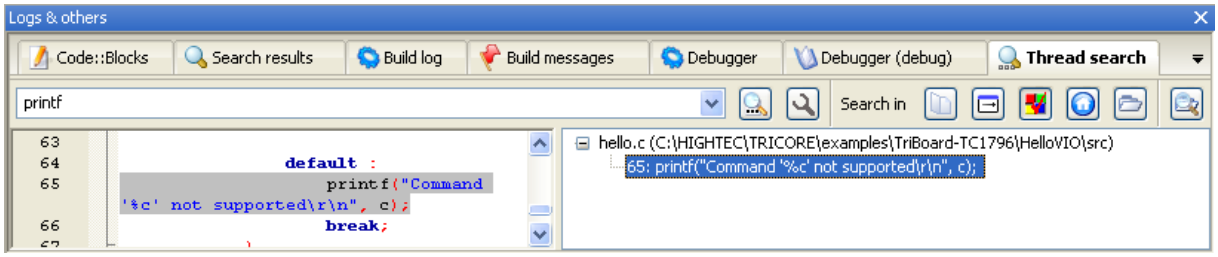


Figure 2.7: Panneau de Thread Search

2.6.2 Utilisation

1. Configurez vos préférences de recherche (voir Figure 2.8 à la page 35)

   Une fois l'extension installée, il y a 4 façons de conduire une recherche :

   a) Tapez/Sélectionnez un mot dans la boîte de recherche combinée et appuyez sur Entrée ou cliquez sur Rechercher dans le panneau de Thread search de la console de messages.

b) Tapez/Sélectionnez un mot dans la boîte de recherche combinée de la barre d'outil et appuyez sur Entrée ou cliquez sur le bouton Rechercher.

c) Clic droit sur n'importe quel 'mot' dans l'éditeur actif puis cliquez sur 'Rechercher les occurrences'.

d) Cliquez sur Rechercher/Thread search pour trouver le mot courant dans l'éditeur actif.

> **Note:**
>
> Les points 1, 2 et 3 peuvent ne pas être disponibles en fonction de la configuration courante.

2. Cliquez de nouveau sur le bouton de recherche pour arrêter la recherche en cours.
3. Un clic simple sur un élément résultat l'affiche dans la prévisualisation sur la droite.
4. Un double clic sur un élément résultat ouvre ou configure un éditeur sur la droite.

2.6.3 Configuration

Pour accéder au panneau de configuration de l'extension ThreadSearch cliquez sur (voir Figure 2.8 à la page 35) :

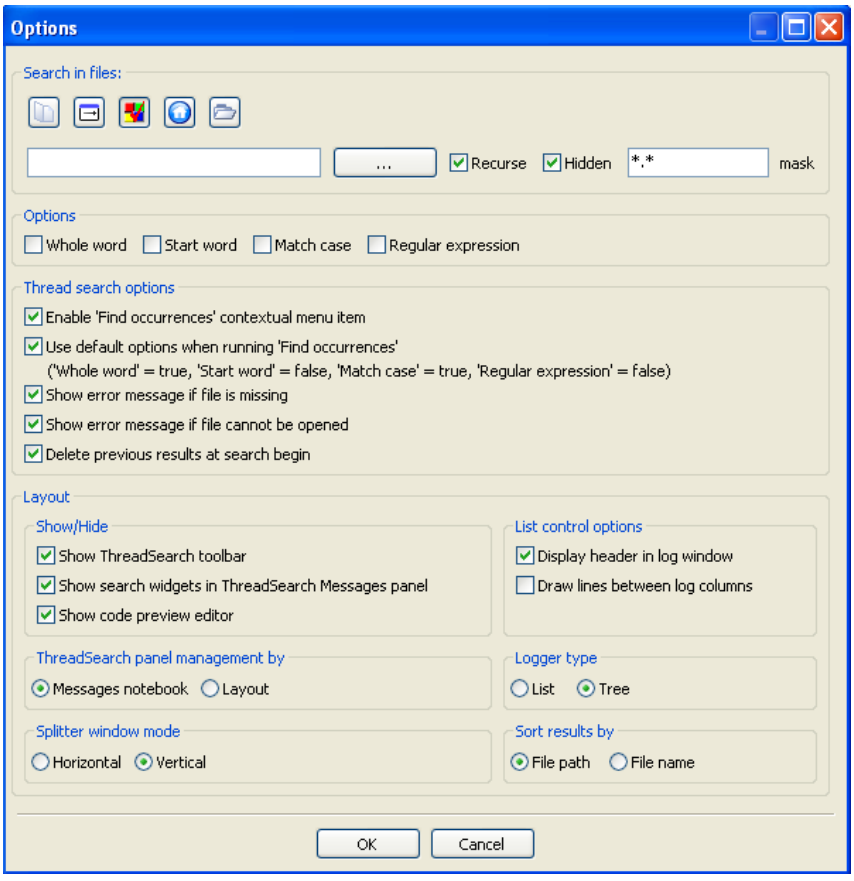


Figure 2.8: Configuration de Thread Search

1. Bouton des Options du panneau de Thread search dans la console des messages.
2. Bouton des Options dans la barre d'outils de Thread search.
3. Menu Paramètres/Environnement puis choisir l'élément Thread search dans la colonne de gauche.

> **Note:**
> Les points 1, 2 et 3 peuvent ne pas être disponibles en fonction de la configuration courante.

La recherche partielle défini l'ensemble de fichiers qui seront analysés.

- Les cases à cocher Projet et Espace de travail sont mutuellement exclusives.
- Le chemin du répertoire peut être édité ou configuré via le bouton Sélection.
- Masque est l'ensemble des spécifications de fichiers séparées par des ';'. Par exemple: `*.cpp;*.c;*.h`.

2.6.4 Options

Mot entier si coché, lignes contenant l'expression recherchée si l'expression recherchée est trouvée sans caractères alphanumériques +'_' avant et après.

Début de mot si coché, lignes contenant l'expression recherchée si l'expression recherchée est trouvée au début d'un mot sans caractères alphanumériques +'_' avant et après.

Respecter la casse si coché, la recherche est sensible à la casse (majuscules-minuscules).

Expression régulière l'expression recherchée est une expression régulière.

> **Note:**
> Si vous voulez chercher des expressions régulières comme `\n` vous devrez choisir l'option 'Utiliser des recherches RegEx avancées' via le menu 'Paramètres' →'Éditeur' →'Paramètres généraux' .

2.6.5 Options de Thread search (ou Tâche de Recherche)

Activer les éléments du menu contextuel 'Trouver les occurrences' Si coché, l'entrée Trouver les occurrences est ajoutée au menu contextuel de l'éditeur.

Utiliser les options par défaut du menu 'Trouver les occurrences' Si coché, un ensemble d'options par défaut est appliqué aux recherches lancées par 'Trouver les occurrences' du menu de contexte correspondant. Par défaut l'option 'Mot entier' et 'Respecter la casse' est activé.

Effacer les résultats précédents en début de recherche Si l'extension ThreadSearch est configurée en 'Vue arborescente' alors les résultats de la recherche sont listés dans l'ordre hiérarchique suivant,

- le premier noeud contient le terme cherché
- ensuite les fichiers qui contiennent ce terme sont listés
- dans cette liste les numéros des lignes et le contenu correspondant sont affichés

Si vous recherchez plusieurs termes, la liste deviendra confuse, aussi les résultats des recherches précédents peuvent être supprimés en utilisant cette option en début de recherche.

> **Note:**
>
> Dans la liste des occurrences les termes seuls ou tous les termes peuvent être supprimés via le menu de contexte 'Supprimer l'élément' ou 'Supprimer tous les éléments' .

2.6.6 Mise en page

Afficher l'entête dans la fenêtre de logs si coché, l'en-tête est affiché dans la liste des résultats de contrôle.

> **Note:**
>
> Si non coché, les colonnes ne sont plus redimensionnables mais on économise de la place.

Dessiner des lignes entre les colonnes Dessine des lignes entre les colonnes en mode Liste.

Afficher la barre d'outils de ThreadSearch Afficher la barre d'outils de l'extension ThreadSearch.

Afficher les widgets de recherche dans le panneau de messages de ThreadSearch Si coché, seuls les résultats de la liste de contrôle et l'éditeur de prévisualisation sont affichés. Les autres widgets de recherches sont masqués (économise de la place).

Afficher l'éditeur de prévisualisation de code La prévisualisation du code peut être masquée soit par cette case à cocher soit par un double clic sur la bordure du séparateur en milieu de fenêtre. C'est ici qu'on peut le faire de nouveau s'afficher.

2.6.7 Panneau de Gestion

Vous pouvez choisir différents modes de gestion de la fenêtre de ThreadSearch. Avec le choix 'Panneau de Messages' la fenêtre ThreadSearch sera intégrée à la console de messages dans un des onglets. Si vous choisissez 'Mise en page' vous pourrez le détacher de la console et obtenir une fenêtre flottante que vous pourrez placer ailleurs.

2.6.8 Type de journal

La vue des résultats de recherche peut s'afficher de plusieurs façons. Le choix 'Liste' affiche toutes les occurrences sous forme d'une liste. L'autre mode 'Arborescence' assemble toutes les occurrences internes d'un fichier dans un noeud.

2.6.9 Mode de partage de fenêtre

L'utilisateur peut configurer la séparation de fenêtre de prévisualisation et de sortie des résultats de recherche horizontalement ou verticalement.

2.6.10 Tri des résultats de recherche

Les résultats de recherche peuvent être triés par le nom de chemin ou le nom de fichier.

2.7 Extensions FileManager et PowerShell

L'explorateur de fichiers Figure 2.9 à la page 39 est inclus dans l'extension FileManager, et se trouve dans l'onglet 'Fichiers'. L'aspect de File Explorer est montré à la Figure 2.9 à la page 39.

En haut vous trouverez le champ d'entrée du chemin. En cliquant sur le bouton à l'extrémité de ce champ, la flèche vers le bas listera un historique des entrées précédentes dans lesquelles on peut naviguer à l'aide d'une barre de défilement. La flèche vers le haut à droite du champ déplace d'un cran vers le haut dans la structure des répertoires.

Dans le champ 'Joker' vous pouvez entrer un filtre de visualisation pour l'affichage des fichiers. En laissant vide ce champ ou en y entrant $*$ vous afficherez tous les fichiers. En y entrant `*.c;*.h` par exemple, vous n'afficherez que les fichiers sources en C et les fichiers d'en-têtes (headers). Ouvrir la flèche du bas, affiche de nouveau la liste des dernières entrées.

Appuyer sur la touche Maj tout en cliquant, sélectionne un groupe de fichiers ou de répertoires, alors qu'appuyer sur la touche Ctrl tout en cliquant sélectionne des fichiers multiples ou des répertoires séparés.

Les opérations suivantes peuvent être obtenues via le menu de contexte si un ou plusieurs répertoires ont été sélectionnés dans l'Explorateur de Fichiers :

Make Root défini le répertoire courant comme répertoire de base.

Ajouter aux favoris configure un marqueur pour ce répertoire et l'enregistre dans les favoris. Cette fonction permet de naviguer rapidement entre des répertoires fréquemment utilisés ou encore sur des disques réseau.

Nouveau Fichier crée un nouveau fichier dans le répertoire sélectionné.

Nouveau Répertoire crée un nouveau sous répertoire dans le répertoire sélectionné.

Les opérations suivantes peuvent être obtenues via le menu de contexte si un ou plusieurs fichiers ou même un ou plusieurs répertoires ont été sélectionnés dans l'Explorateur de Fichiers :

Dupliquer copie un fichier/répertoire et le renomme.

Copier vers ouvre une boîte de dialogue pour entrer un répertoire cible dans lequel on copiera les fichiers/répertoires.

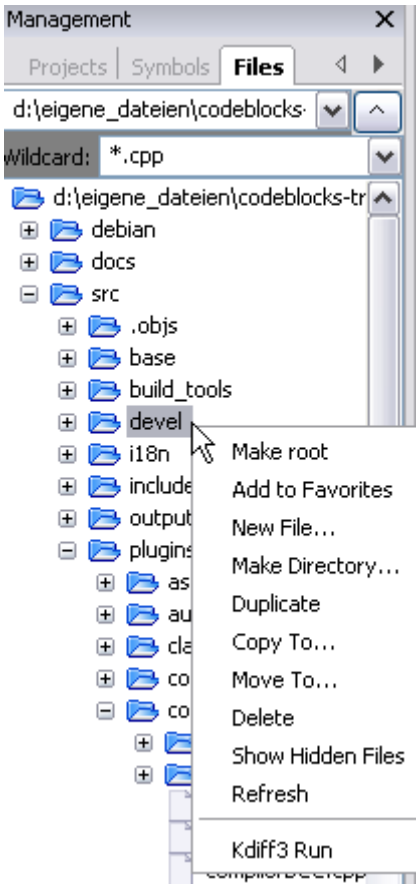


Figure 2.9: Le gestionnaire de fichiers

Déplacer vers déplace la sélection vers un autre endroit.

Supprimer supprime les fichiers/répertoires sélectionnés.

Afficher les fichiers masqués active/désactive l'affichage des fichiers systèmes masqués. Si activé, le menu est coché par un marqueur.

Actualiser actualise l'affichage de l'arborescence des répertoires.

Les opérations suivantes peuvent être obtenues via le menu de contexte si un ou plusieurs fichiers ont été sélectionnés dans l'Explorateur de Fichiers :

Ouvrir dans l'éditeur CB ouvre le fichier sélectionné dans l'éditeur de CodeBlocks.

Renommer renomme le fichier sélectionné.

Ajouter au projet actif ajoute le(s) fichier(s) au projet actif.

> **Note:**
>
> Les fichiers/répertoires sélectionnés dans l'explorateur de fichiers peuvent être accédés dans l'extension PowerShell à l'aide de la variable mpaths.

On peut spécifier via la commande de menu 'Paramètres' →'Environnement' →'PowerShell' des fonctions utilisateur. Dans le masque de PowerShell, une nouvelle fonction qui peut être nommée aléatoirement, est créée via le bouton 'Nouveau'. Dans le champ 'ShellCommand Executable', le programme exécutable est spécifié, et dans le champ en bas de la fenêtre, des paramètres additionnels peuvent être passés au programme. En cliquant sur la fonction dans le menu de contexte ou dans le menu de PowerShell, la fonction s'exécute et traite les fichiers/répertoires sélectionnés. La sortie est redirigée vers une fenêtre de shell séparée.

Par exemple une entrée de menu a été créée dans 'PowerShell' →'SVN' et dans le menu de contexte en tant que 'SVN'. Dans ce contexte `$file` signifie le fichier sélectionné dans l'explorateur de fichiers, `$mpath` les fichiers ou répertoires sélectionnés (voir section 3.2 à la page 59).

```
Ajouter;$interpreter add $mpaths;;;
```

Celle-ci et toutes les commandes suivantes créeront un sous-menu, dans ce cas 'Extensions' →'SVN' →'Ajouter' . Le menu de contexte est étendu de même. Cliquez sur la commande du menu de contexte pour faire exécuter la commande SVN `add` sur les fichiers/répertoires sélectionnés.

TortoiseSVN est un programme SVN très répandu qui s'intégre dans l'explorateur. Le programme `TortoiseProc.exe` de TortoiseSVN peut être démarré en ligne de commande et affiche une boîte de dialogue pour y entrer les données de l'utilisateur. Ainsi vous pouvez lancer des commandes, disponibles en tant que menus de contexte dans l'explorateur, également en ligne de commande. Vous pouvez donc l'intégrer en tant qu'extension du Shell dans CodeBlocks. Par exemple, la commande

```
TortoiseProc.exe /command:diff /path:$file
```

affichera les différences entre un fichier sélectionné dans l'explorateur de CodeBlocks et celui de la base de SVN. Voir Figure 2.10 à la page 41 comment intégrer cette commande.

> **Note:**
>
> Pour les fichiers qui sont sous le contrôle de SVN l'explorateur de fichier affiche des icônes superposées qui s'activent via le menu 'Vue' →'SVN Decorators' .

Exemple

Vous pouvez utiliser l'explorateur de fichiers pour afficher les différences sur des fichiers ou des répertoires. Suivez les étapes suivantes :

1. Ajoutez le nom via le menu 'Paramètres' →'Environnement' →'PowerShell' . C'est affiché comme une entrée par l'interpréteur de menu et le menu de contexte.
2. Sélectionnez le chemin absolu de l'exécutable Diff (notamment kdiff3). Le programme est accédé avec la variable `$interpreter` .
3. Ajoutez les paramètres de l'interpréteur

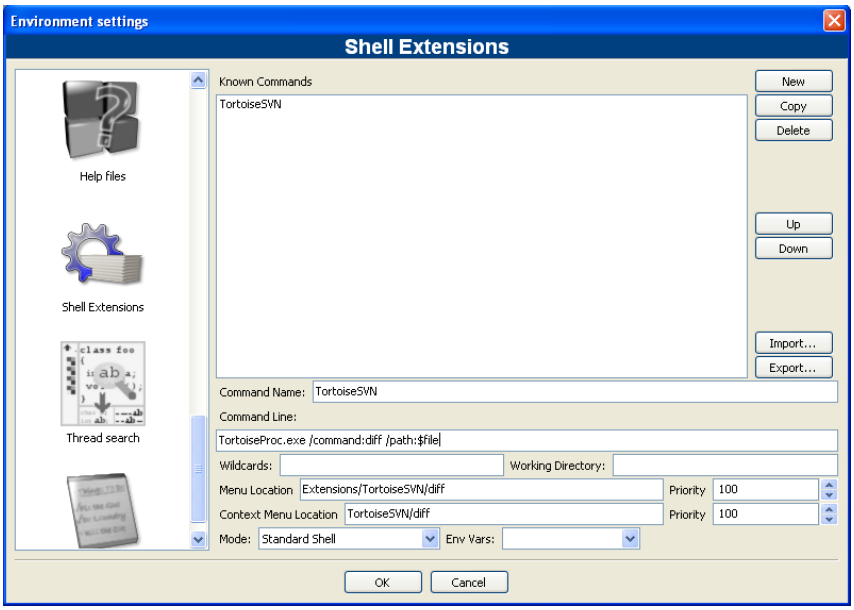


Figure 2.10: Ajout d'une extension Shell au menu de contexte

```
Diff;$interpreter $mpaths;;;
```

Cette commande sera exécutée en utilisant les fichiers ou répertoires sélectionnés en tant que paramètres. La sélection peut être accédée via la variable $mpaths. Ceci est une façon commode de différentier des fichiers ou des répertoires.

> **Note:**
>
> L'extension supporte l'utilisation des variables de CodeBlocks dans l'extension du Shell.

`$interpreter`	Appelle cet exécutable.
`$fname`	Nom du fichier sans son extension.
`$fext`	Extension du fichier sélectionné.
`$file`	Nom du fichier.
`$relfile`	Nom du fichier sans l'information de chemin.
`$dir`	Nom du répertoire sélectionné.
`$reldir`	Nom du répertoire sans l'information de chemin.
`$path`	Chemin absolu.
`$relpath`	Chemin relatif du fichier ou du répertoire
`$mpaths`	Liste des fichiers et répertoires sélectionnés actuellement
`$inputstr{<msg>}`	Chaîne de caractères qui est entrée dans une fenêtre de message.
`$parentdir`	Répertoire Parent (../).

> **Note:**
>
> Les entrées de l'extension Shell sont également disponibles en tant que menus de contexte dans l'éditeur de CodeBlocks.

2.8 Browse Tracker

Browse Tracker est une extension qui aide à naviguer parmi les fichiers récemment ouverts dans CodeBlocks. La liste des fichiers récents est sauvegardée dans un historique. Le menu 'Vue' →'Suivi de Navigation' →'Tout Effacer' permet d'effacer l'historique.

Dans les différents 'onglets' vous pouvez naviguer entre les divers éléments des fichiers récemment ouverts en utilisant l'entrée de menu 'Vue' →'Suivi de Navigation' →'Aller en arrière/Aller en avant' ou en utilisant les raccourcis claviers Alt-Gauche/Alt-Droit. Le menu de suivi de navigation est également accessible dans les menus de contexte. Les marqueurs sont enregistrés dans un fichier de mise en page `<projectName>.bmarks`

Quand on développe du logiciel, on passe souvent d'une fonction à une autre implémentée dans différents fichiers. L'extension de suivi de navigation vous aidera dans cette tâche en vous montrant l'ordre dans lequel ont été sélectionnés les fichiers. Vous pouvez alors naviguer confortablement dans les différents appels de fonctions.

L'extension permet même de naviguer entre les marqueurs de chaque fichier de l'éditeur de CodeBlocks. La position du curseur est mémorisée pour chacun des fichiers. Vous pouvez poser ces marqueurs en utilisant le menu 'Vue' →' Suivi de Navigation' →' Activer le marquage de navigation' ou en sélectionnant une ligne avec le bouton gauche de la souris. Une marque ... est alors posée dans la marge gauche. Avec les menus 'Vue' →'Suivi de Navigation' →'Marque précédente/Marque suivante' ou les raccourcis Alt-up/Alt-down vous pouvez naviguer entre les différents marques posées dans un fichier. Si vous voulez naviguer dans un fichier avec des marques triées en fonction du numéro de lignes, choisissez simplement le menu 'Vue' →'Suivi de Navigation' →'Trier les marques de navigation' .

En choisissant 'Effacer la marque de navigation' le marqueur de la ligne sélectionnée est supprimé. Si un marqueur est posé sur une ligne, le fait d'appuyer pendant 1/4 de seconde sur le bouton gauche de la souris tout en appuyant sur la touche Ctrl effacera le marqueur de cette ligne. Avec le menu 'Effacer toutes les marques de navigation' ou avec un Ctrl-clic gauche sur toute ligne non marquée, vous remettez à 0 tous les marqueurs d'un fichier.

Le paramétrage de l'extension peut être configuré via le menu 'Paramètres' →'Éditeur' →'Browse Tracker' .

> **Note:**
>
> NdT : certains menus ou affichages ne sont pas traduits car l'auteur de l'extension n'a pas marqué certaines chaînes comme étant traduisibles

Mark Style (Styles des marques) Les marques de navigation sont affichées par défaut comme des ... dans la marge. Avec le choix 'Book_Marks' elles seront affichées en tant que marque par une flèche bleue dans la marge. L'option "hide" supprime l'affichage des marques.

Toggle Browse Mark key Les marques peuvent être activées ou supprimées soit par un simple clic avec le bouton gauche de la souris soit avec un clic-gauche tout en maintenant la touche Ctrl enfoncée.

Toggle Delay Durée pendant laquelle le bouton gauche de la souris est enfoncé pour entrer dans le mode de marquage de navigation.

Clear All BrowseMarks (Effacer toutes les marques) tout en maintenant enfoncée la touche Ctrl soit par simple clic soit par double clic sur le bouton gauche de la souris.

La configuration de l'extension est enregistrée dans votre répertoire application data dans le fichier `default.conf`. Si vous utilisez la fonctionnalité des profils (ou personnalité) de CodeBlocks la configuration est alors lue dans votre fichier <`personality`>`.conf`.

2.9 Support de SVN

> **Note:**
>
> NdT : Cette extension est traduite ici, mais est obsolète. Vous avez donc de grandes chances de ne plus la trouver.

Le support du système de contrôle de version SVN est inclus dans l'extension CodeBlocks TortoiseSVN. Via le menu 'TortoiseSVN' →'Plugin settings' vous pouvez configurer les commandes svn accessibles dans l'onglet 'Integration' .

Menu integration Ajoute une entrée TortoiseSVN dans la barre de menu avec différents paramétrages.

Project manager Active les commandes TortoiseSVN du menu de contexte de la gestion de projet.

Editor Active les commandes TortoiseSVN du menu de contexte de l'éditeur.

Dans la configuration de l'extension vous pouvez choisir quelles sont les commandes svn qui sont accessibles dans le menu principal ou le menu de contexte. L'onglet intégration fournit une entrée 'Edit main menu' et 'Edit popup menu' pour paramétrer ces commandes.

> **Note:**
>
> L'Explorateur de fichiers dans CodeBlocks utilise différentes icônes superposées afin d'indiquer l'état de svn. Les commandes de TortoiseSVN sont incluses dans le menu de contexte.

2.10 LibFinder

Si vous voulez utilisez des librairies dans votre application, vous devez configurer votre projet pour cela. Un tel processus de configuration peut être difficile et ennuyeux car chaque librairie peut utiliser un schéma d'options particulier. Un autre problème est que

cette configuration diffère entre les plates-formes ce qui résulte en des incompatibilités entre des projets Unix et Windows.

LibFinder propose deux fonctionnalités majeures :

- Recherche des librairies installées sur votre système
- Inclure les librairies dans votre projet en seulement quelques clics en rendant le projet indépendant de la plate-forme

2.10.1 Recherche de librairies

La recherche des librairies est disponible via le menu 'Extensions' →'Library finder' . Son but est de détecter les librairies installées sur votre système et d'enregistrer les résultats dans la base de données de LibFinder (notez que ces résultats ne sont pas écrits dans les fichiers projets de CodeBlocks). La recherche commence par un dialogue où vous pouvez fournir un ensemble de répertoires où sont installées les librairies. LibFinder les analysera de façon récursive aussi, si vous ne savez pas trop où elles sont, vous pouvez sélectionner des répertoires génériques. Vous pouvez même entrer le disque complet – dans ce cas là, le processus de recherche prendra plus de temps mais il détectera davantage de librairies (voir Figure 2.11 à la page 44).

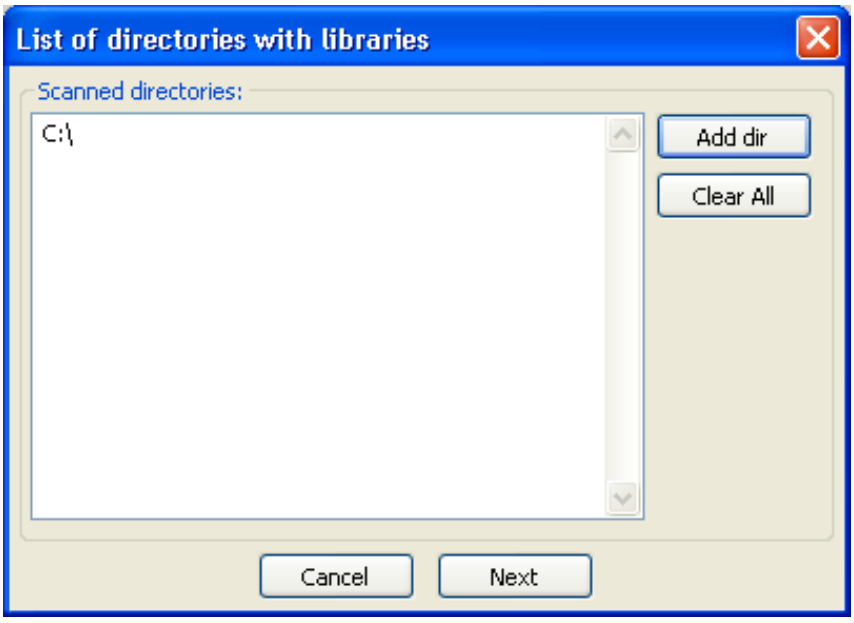


Figure 2.11: Liste de répertoires

Quand LibFinder est à la recherche de librairies, il utilise des règles spéciales pour détecter leur présence. Chaque ensemble de règle est situé dans un fichier xml. Actuellement LibFinder peut rechercher wxWidgets 2.6/2.8, CodeBlocks SDK et GLFW – la liste sera étendue dans le futur.

Note:

Pour obtenir davantage de détails sur comment ajouter un support de librairie dans LibFinder, lisez dans les sources de CodeBlocks `src/plugins/contrib/lib_finder/lib_finder/readme.txt`.

Après avoir terminé l'analyse, LibFinder affiche les résultats (voir Figure 2.12 à la page 45).

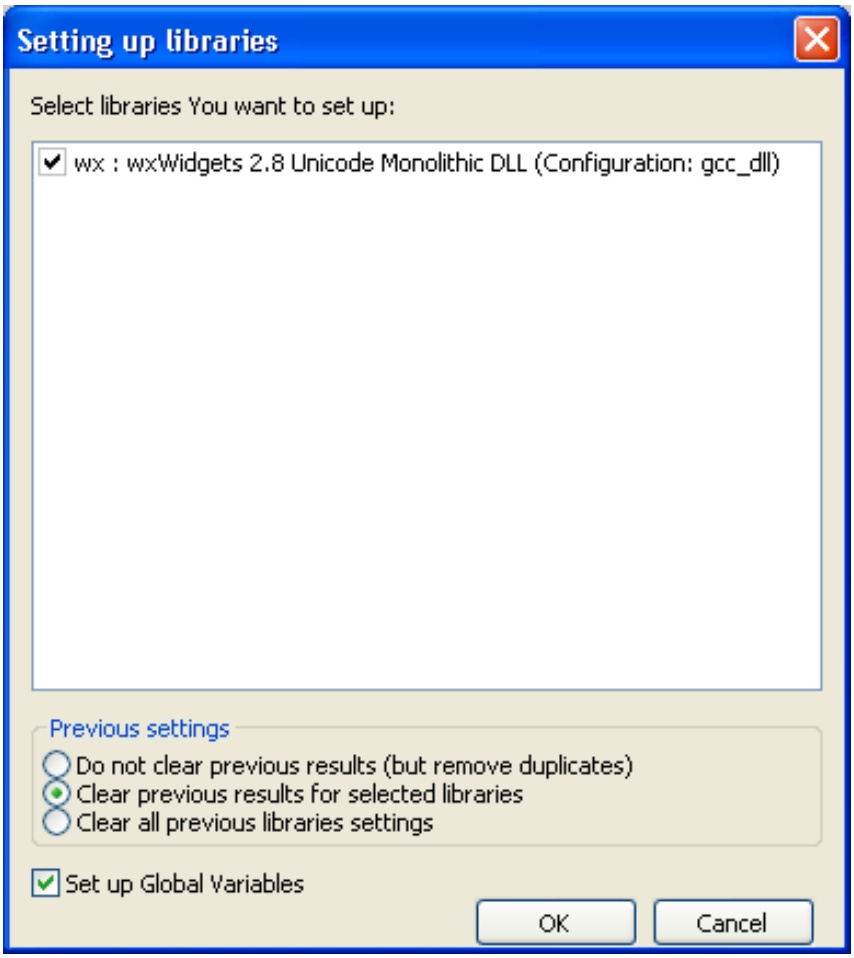


Figure 2.12: Résultats de recherche

Dans la liste, vous cochez les librairies qui doivent être enregistrées dans la base de données de LibFinder. Notez que chaque librairie peut avoir plus d'une configuration valide et les paramétrages ajoutés en premier sont plutôt destinés à être utilisés lors de la génération de projets.

Au-dessous de la liste, vous pouvez sélectionner ce qu'il faut faire avec les résultats des analyses précédentes :

Ne pas effacer les résultats précédents Cette option travaille comme une mise à jour des résultats existants – Cela ajoute les nouveaux et met à jour ceux qui existent déjà. Cette option n'est pas recommandée.

Seconde option (Effacer les résultats précédents des librairies sélectionnées) effacera

tous les résultats des recherches précédentes des librairies sélectionnées avant d'ajouter les nouveaux résultats. C'est l'option recommandée.

Effacer toutes les configurations précédentes des librairies quand vous sélectionnez cette option, la base de données de LibFinder sera effacée avant d'y ajouter les nouveaux résultats. C'est utile quand vous voulez nettoyer une base de données LibFinder contenant des résultats invalides.

Une autre option de ce dialogue est 'Configurer les Variables Globales' . Quand vous cochez cette option, LibFinder essaiera de configurer des Variables Globales qui sont aussi utilisées pour aider à traiter les librairies.

Si vous avez pkg-config d'installé sur votre système (C'est installé automatiquement sur la plupart des versions de systèmes linux) LibFinder proposera des librairies venant de cet outil. Il n'est pas nécessaire de faire une analyse spécifique pour celles-ci – elles seront automatiquement chargées au démarrage de CodeBlocks.

2.10.2 Inclure des librairies dans les projets

LibFinder ajoute un onglet supplémentaire dans les propriétés d'un projet 'Librairies' – Cet onglet montre les librairies utilisées dans le projet ainsi que celles connues de LibFinder. Pour ajouter une librairie dans votre projet, sélectionnez là dans le panneau de droite et cliquez sur le bouton <. Pour enlever une librairie d'un projet, sélectionnez la dans le panneau de gauche et cliquez sur le bouton > (voir Figure 2.13 à la page 46).

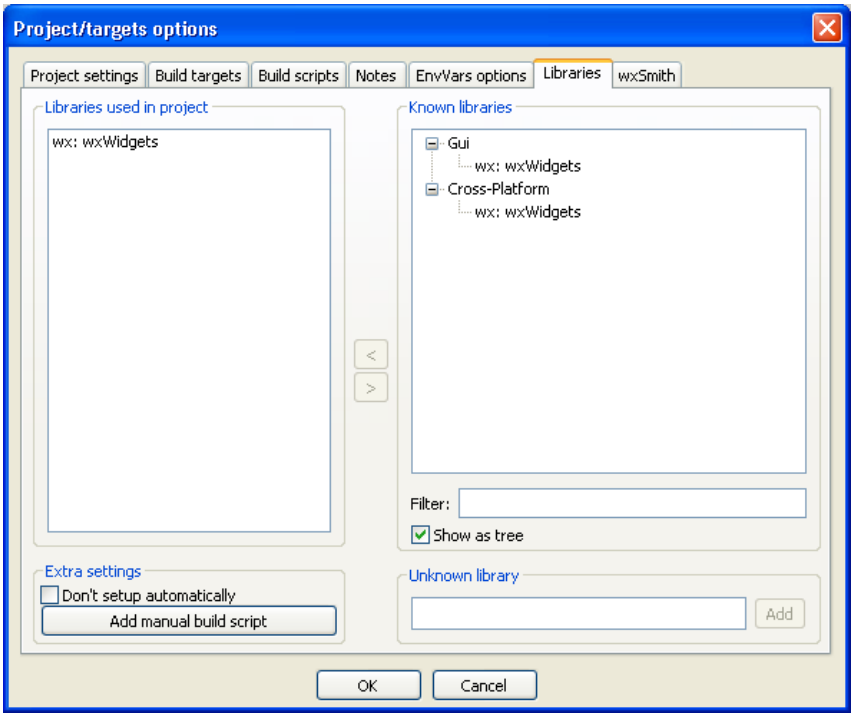


Figure 2.13: Configuration de projet

Vous pouvez filtrer les librairies connues de LibFinder en fournissant un filtre de recherche. La case à cocher 'Afficher comme un arbre' permet de basculer antre des vues sans catégories et des vues avec catégories.

Si vous voulez ajouter une librairie qui n'est pas disponible dans la base de données de LibFinder, vous pouvez utiliser le champ 'Librairie inconnue' . Notez que vous devriez entrer le "library's shortcode" (nom court, qui habituellement correspond au nom de variable globale) ou le nom de librairie dans pkg-config. Vous trouverez une liste de noms courts suggérés dans le Wiki de CodeBlocks dans Global Variables. L'usage de cette option n'est recommandé que lorsqu'on prépare un projet qui doit être généré sur d'autres machines où ce type de librairie existe et y est correctement détectée par LibFinder. Vous pouvez accéder à une variable globale dans CodeBlocks comme :

```
$(#GLOBAL_VAR_NAME.include)
```

Cocher l'option 'Ne pas configurer automatiquement' indiquera à LibFinder qu'il ne devrait pas ajouter automatiquement les librairies lors de la compilation. Dans ce cas, LibFinder peut s'invoquer depuis un script de génération. Un exemple d'un tel script est généré et ajouté au projet en appuyant sur 'Ajouter un script de génération manuel' .

2.10.3 Utilisation de LibFinder dans des projets générés par des assistants

Les assistants vont créer des projets qui n'utilisent pas LibFinder. Pour les intégrer avec cette extension, vous devrez mettre à jour manuellement les options de génération du projet. Ceci est facilement obtenu en enlevant tous les paramétrages spécifiques aux librairies et en ajoutant les librairies au travers de l'onglet 'Librairies' dans les propriétés du projet.

De tels projets deviennent indépendants des plates-formes. Tant que les librairies utilisées sont dans la base de données de LibFinder, les options de génération du projet seront automatiquement mises à jour pour coïncider avec les paramétrages de librairie propres aux plates-formes.

2.11 AutoVersioning

Une application de suivi de versions qui incrémente les numéros de version et de construction de votre application à chaque fois qu'un changement est effectué et l'enregistre dans un fichier `version.h` avec des déclarations de variables faciles à utiliser. Possède également une option pour proposer des changements dans un style à la SVN, un éditeur de schémas de versions, un générateur de journal des changements, et bien d'autres choses encore ...

2.11.1 Introduction

L'idée de développer l'extension AutoVersioning est venue lors du développement d'un logiciel en version pre-alpha qui exigeait des informations de version et d'état. Trop occupé par le codage, sans temps disponible pour maintenir la numérotation des versions, l'auteur a décidé de développer une extension qui puisse faire le travail avec aussi peu d'interventions que possible.

2.11.2 Fonctionnalités

Voici résumée la liste des fonctions couvertes par l'extension :

- Supporte C et C++.
- Génère et incrémente automatiquement des variables de versions.
- Éditeur de l'état du logiciel.
- Éditeur de schéma intégré pour changer le comportement de l'auto incrémentation des valeurs de versions.
- Déclaration des dates en mois, jour et année.
- Style de version Ubuntu.
- Contrôle des révisions Svn.
- Générateur de journal des changements.
- Fonctionne sous Windows et Linux.

2.11.3 Utilisation

Aller simplement dans le menu 'Projet' →'Autoversioning' . Une fenêtre popup comme celle-ci apparaîtra :

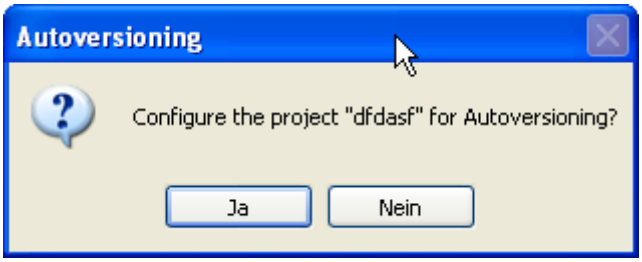


Figure 2.14: Configuration d'un projet pour Autoversioning

Quand on répond Oui au message de demande de configuration, la fenêtre principale de configuration d'AutoVersioning s'ouvre pour vous permettre de paramétrer les informations de version de votre projet.

Après avoir configuré votre projet pour le versionnage automatique, les paramètres entrés dans la boîte de dialogue de configuration sont enregistrées dans le fichier de projet et un fichier `version.h` est créé. Pour le moment, chaque fois que vous entrez dans le menu 'Projet' →'Autoversioning' , le dialogue de configuration qui apparaît vous permet d'éditer votre version de projet et les paramètres qui y sont liés, à moins que vous n'enregistriez pas les nouveaux changements effectués par l'extension dans le fichier de projet.

2.11.4 Onglets de la boîte de dialogue

2.11.4.1 Valeurs de Version

Ici vous entrez simplement les valeurs de version adéquates ou laissez l'extension Autoversioning le faire pour vous (see Figure 2.15 à la page 49).

Version Majeure Incrémenté de 1 quand le numéro mineur atteint son maximum

Version mineure Incrémenté de 1 quand le numéro de génération dépasse la barrière de nombre de générations, la valeur étant remise à 0 quand il atteint sa valeur maximale.

Numéro de génération (également équivalent à numéro de Release) - Incrémenté de 1 chaque fois que le numéro de révision est incrémenté.

Révision Incrémenté aléatoirement quand le projet a été modifié puis compilé.

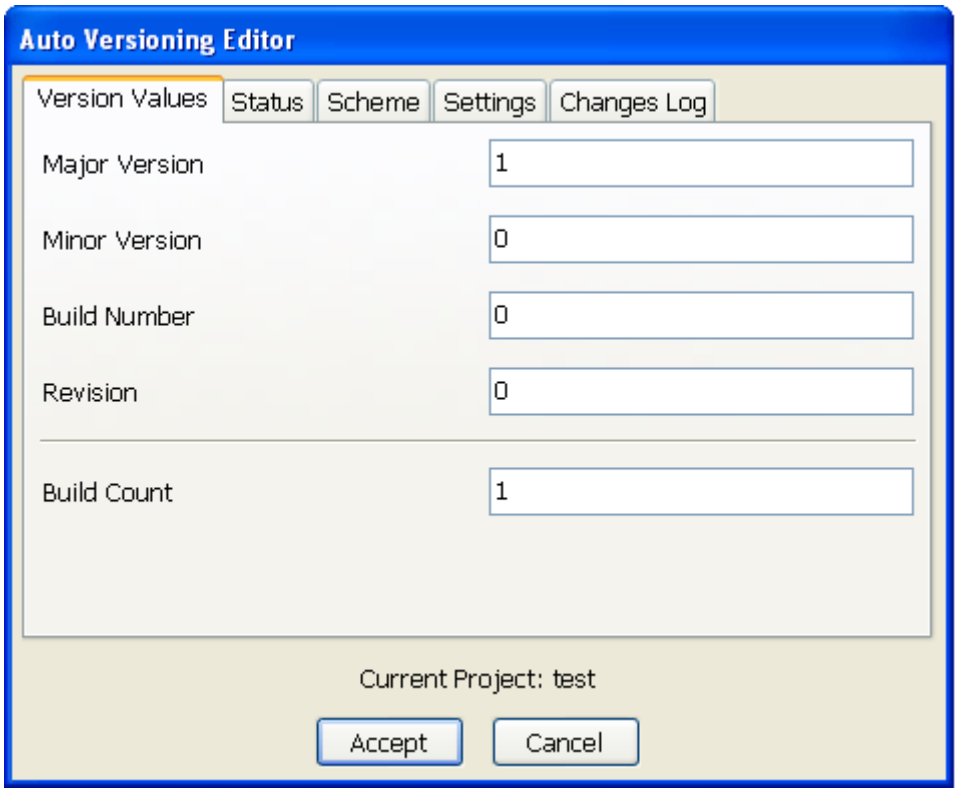


Figure 2.15: Configuration des Valeurs de Version

2.11.4.2 État

Quelques champs pour garder une trace de l'état de votre logiciel avec une liste de valeurs prédéfinies usuelles (voir Figure 2.16 à la page 50).

État du logiciel Un exemple typique pourrait être v1.0 Alpha

Abréviation Idem à l'état du logiciel mais comme ceci : v1.0a

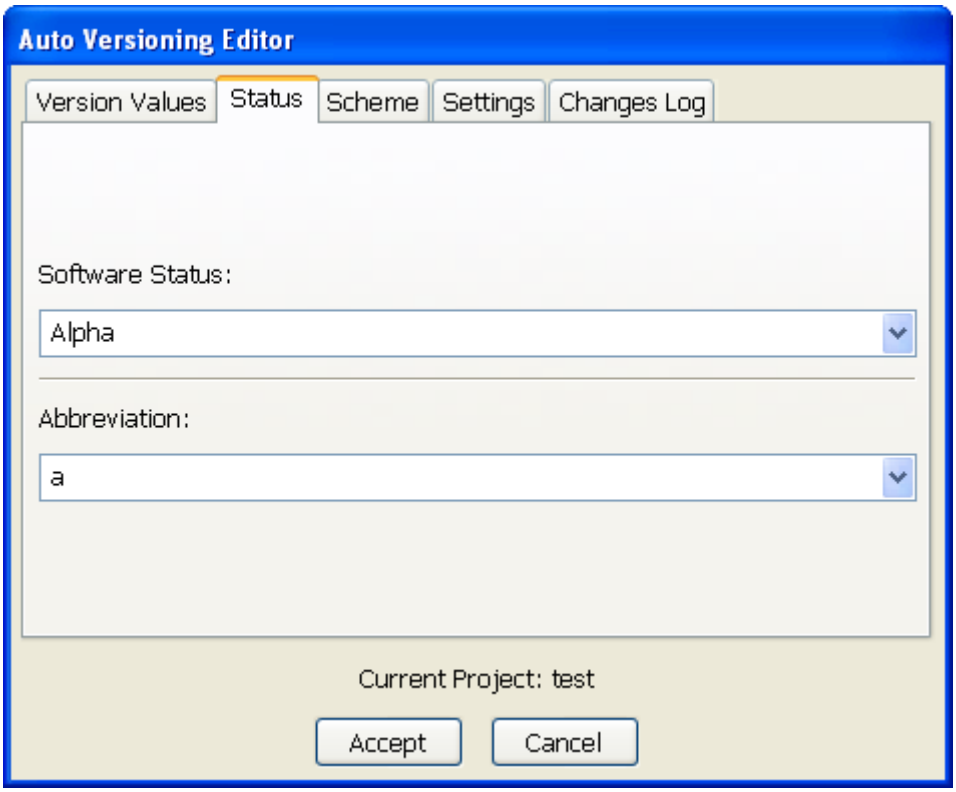


Figure 2.16: Configuration de l'État dans Autoversioning

2.11.4.3 Schéma

Vous permet d'éditer comment l'extension incrémentera les valeurs de version (voir Figure 2.17 à la page 51).

Valeur max pour numéro mineur Valeur maximale que peut atteindre la valeur mineure. Une fois cette valeur atteinte, le numéro Majeur est incrémenté de 1 et à la compilation suivante le numéro mineur sera remis à 0.

Nombre max de générations Quand cette valeur est atteinte, le compteur sera remis à 0 à la génération suivante. Mettre à 0 pour ne pas limiter.

Révision maximale Comme Nombre max de générations. Mettre à 0 pour ne pas limiter.

Révision aléatoire maximale Les révisions s'incrémentent par un nombre aléatoire que vous décidez. Si vous mettez 1, les révisions s'incrémenteront évidemment par 1.

Nombre de générations avant d'incrémenter Mineur Après des changements de code et des compilations avec succès, l'historique des générations s'incrémente, et quand cette valeur est atteinte alors la valeur Mineure s'incrémente.

2.11.4.4 Paramètres

Ici vous pouvez entrer certains paramètres du comportement d'Autoversioning (voir Figure 2.18 à la page 51).

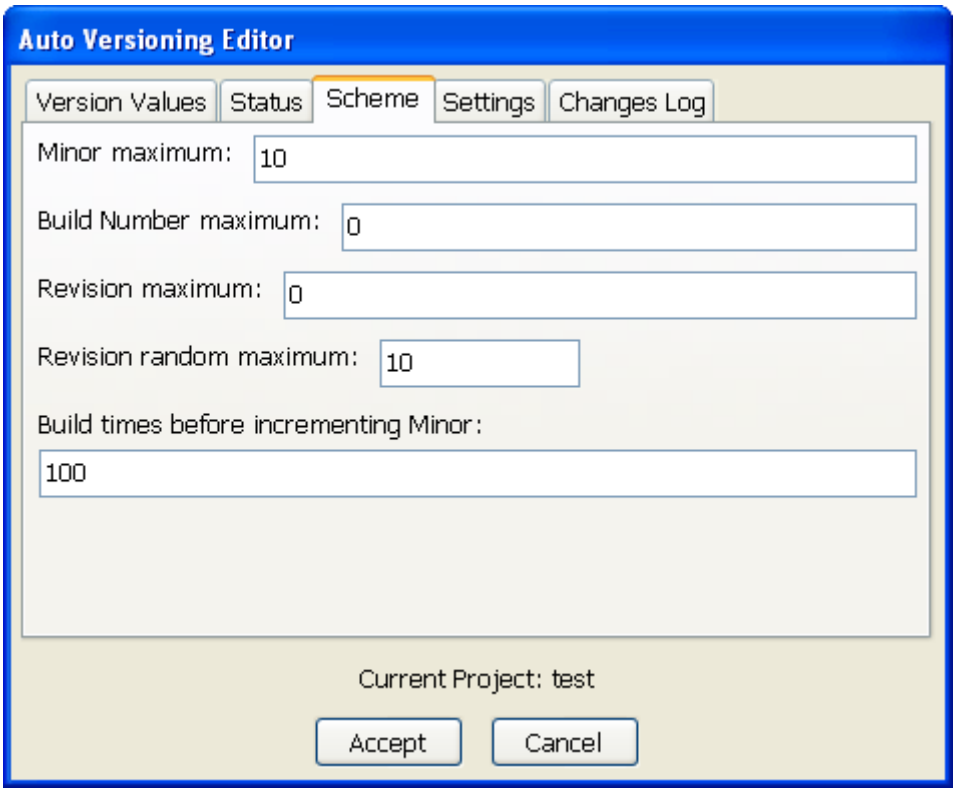


Figure 2.17: Schéma de fonctionnement d'Autoversioning

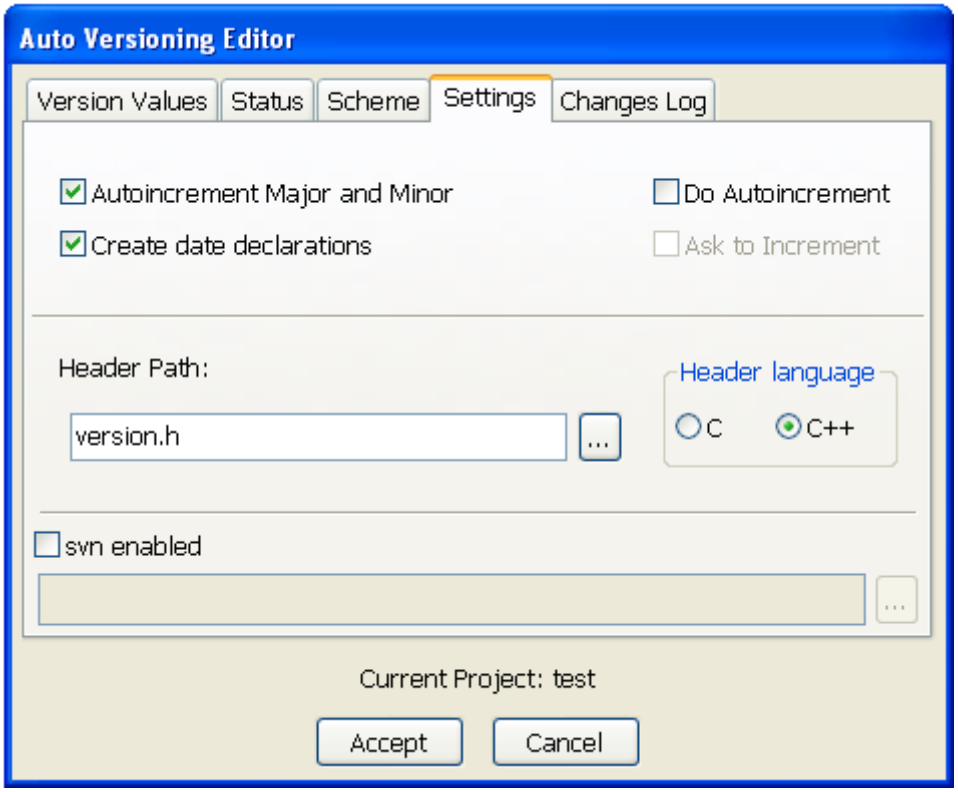


Figure 2.18: Paramètres d'Autoversioning

Auto-incrémente Majeur et Mineur Laisse l'extension incrémenter ces valeurs en utilisant le schéma. Si non coché, seuls les numéros de génération et de Révision s'incrémenteront.

Créer des déclarations de dates Crée des entrées dans le fichier `version.h` avec des dates et un style de version à la Ubuntu.

Incrémentation automatique Indique à l'extension d'incrémenter automatiquement dès qu'une modification est faite. Cette incrémentation interviendra avant la compilation.

Langage de l'en-tête Sélectionne le langage de sortie du fichier `version.h`

Interroger pour incrémenter Si Incrémentation automatique est coché, on vous interroge alors avant la compilation (si des changements ont été effectués) pour incrémenter les valeurs de version.

Svn activé Recherche dans le répertoire courant la révision Svn et la date puis génère les entrées correspondantes dans `version.h`

2.11.4.5 Journal des changements

Ceci vous permet d'entrer chaque changement effectué au projet afin de générer un fichier `ChangesLog.txt` (voir Figure 2.19 à la page 52).

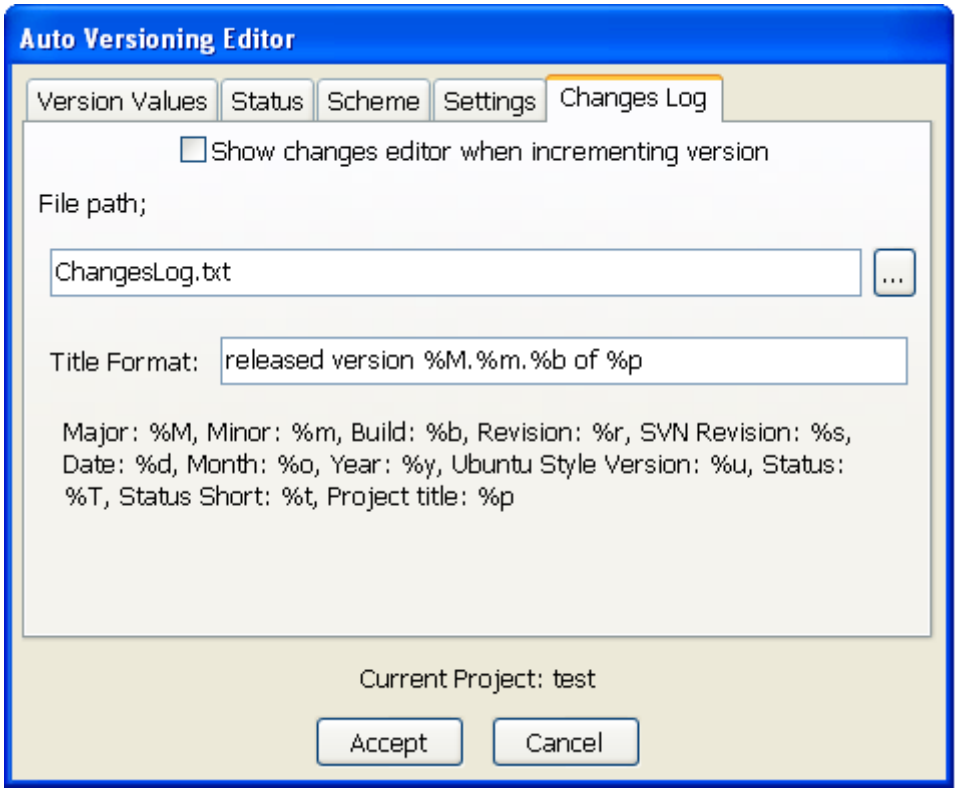


Figure 2.19: Journal des changements d'Autoversioning

Afficher les changements quand la version s'incrémente Affichera une fenêtre popup d'édition de journal quand la version est incrémentée.

Format du Titre Un titre formaté avec une liste de valeurs prédéfinies.

2.11.5 Inclusion dans votre code

Pour utiliser les variables générées par l'extension faire simplement **#include** <version.h>. Le code suivant est un exemple de ce qu'on peut faire :

```
#include <iostream>
#include "version.h"

void main(){
    std::cout<<AutoVersion::Major<<endl;
}
```

2.11.5.1 Sortie de version.h

Le fichier d'en-tête généré. Voici un exemple sur un fichier en mode c++ :

```
#ifndef VERSION_H
#define VERSION_H

namespace AutoVersion{

  //Date Version Types
  static const char DATE[] = "15";
  static const char MONTH[] = "09";
  static const char YEAR[] = "2007";
  static const double UBUNTU_VERSION_STYLE = 7.09;

  //Software Status
  static const char STATUS[] = "Pre-alpha";
  static const char STATUS_SHORT[] = "pa";

  //Standard Version Type
  static const long MAJOR = 0;
  static const long MINOR = 10;
  static const long BUILD = 1086;
  static const long REVISION = 6349;

  //Miscellaneous Version Types
  static const long BUILDS_COUNT = 1984;
  #define RC_FILEVERSION 0,10,1086,6349
  #define RC_FILEVERSION_STRING "0, 10, 1086, 6349\0"
  static const char FULLVERSION_STRING[] = "0.10.1086.6349";

}
#endif //VERSION_h
```

En mode C c'est la même chose qu'en C++ mais sans le namespace:

```
#ifndef VERSION_H
#define VERSION_H

  //Date Version Types
  static const char DATE[] = "15";
  static const char MONTH[] = "09";
  static const char YEAR[] = "2007";
```

```
  static const double UBUNTU_VERSION_STYLE = 7.09;

  //Software Status
  static const char STATUS[] = "Pre-alpha";
  static const char STATUS_SHORT[] = "pa";

  //Standard Version Type
  static const long MAJOR = 0;
  static const long MINOR = 10;
  static const long BUILD = 1086;
  static const long REVISION = 6349;

  //Miscellaneous Version Types
  static const long BUILDS_COUNT = 1984;
  #define RC_FILEVERSION 0,10,1086,6349
  #define RC_FILEVERSION_STRING "0, 10, 1086, 6349\0"
  static const char FULLVERSION_STRING[] = "0.10.1086.6349";

#endif //VERSION_h
```

2.11.6 Générateur de journal des changements

Cette boîte de dialogue est accessible à partir du menu 'Projet' →'Journal des changements' . Également si la case "Afficher l'éditeur des changements quand la version s'incrémente" est cochée, une fenêtre s'ouvrira pour vous permettre d'entrer la liste des changements après une modification des sources du projet ou un évènement d'incrémentation (voir Figure 2.20 à la page 55).

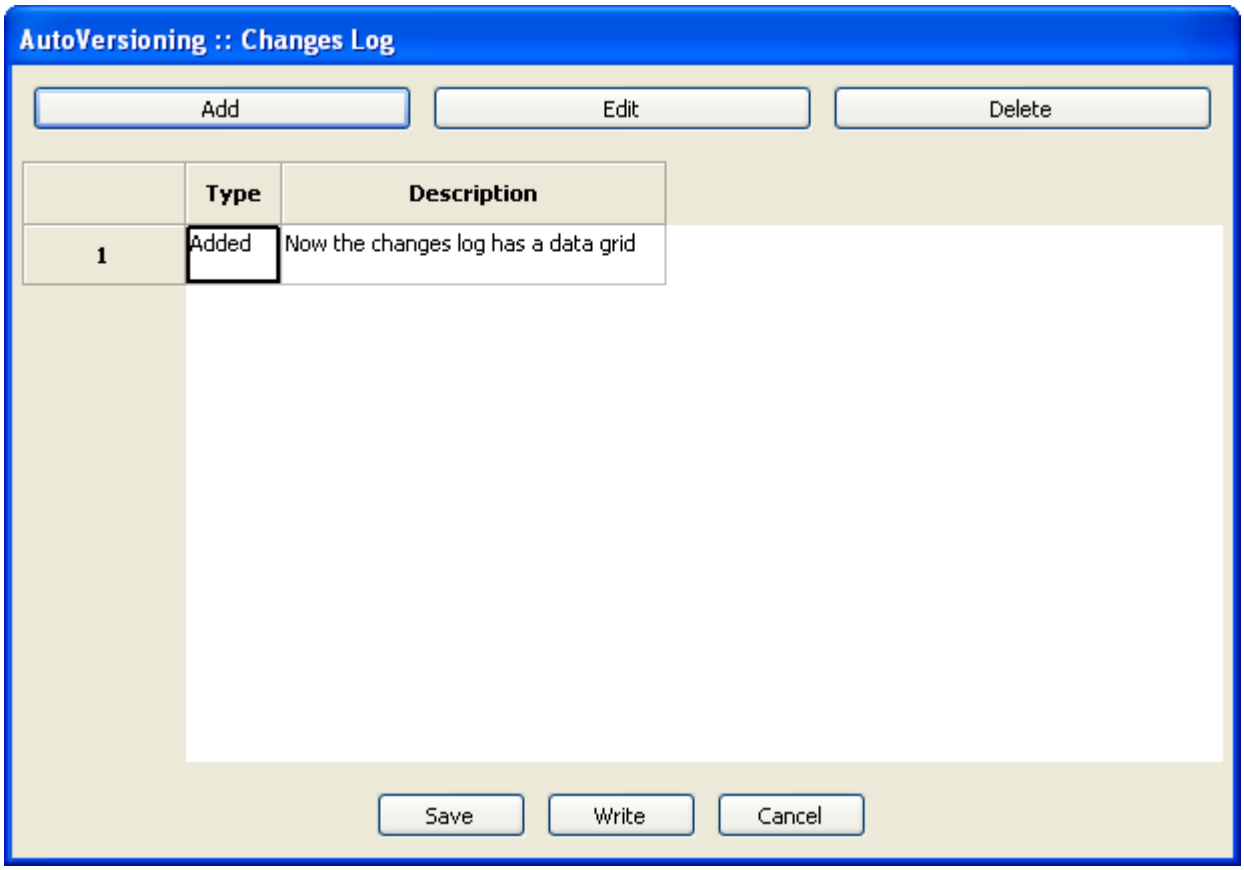


Figure 2.20: Changements dans un projet

2.11.6.1 Résumé des Boutons

Ajouter Ajoute une ligne à la grille de données

Éditer Active les modifications de la cellule sélectionnée

Supprimer Supprime la ligne courante de la grille de données

Enregistrer Enregistre dans un fichier temporaire (`changes.tmp`) les données actuelles pour pouvoir effectuer plus tard les entrées dans le journal des changements

Écrire Entre la grille de données dans le journal des changements

Annuler Ferme simplement la boîte de dialogue sans rien faire d'autre

Voici un exemple de sortie générée par l'extension dans le fichier `ChangesLog.txt` :

```
03 September 2007
   released version 0.7.34 of AutoVersioning-Linux

     Change log:
        -Fixed: pointer declaration
        -Bug: blah blah

02 September 2007
   released version 0.7.32 of AutoVersioning-Linux
```

```
     Change log:
        -Documented some areas of the code
        -Reorganized the code for readability

01 September 2007
   released version 0.7.30 of AutoVersioning-Linux

     Change log:
        -Edited the change log window
        -If the change log windows is leave blank no changes.txt is modified
```

2.12 Code statistics

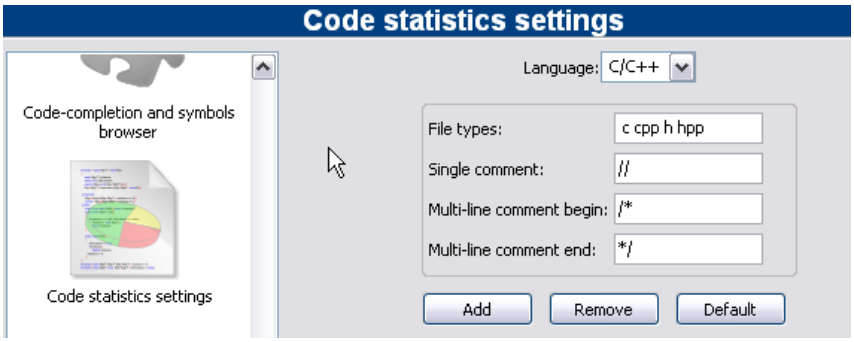


Figure 2.21: Configuration de Code Statistics

Basée sur les caractéristiques d'un masque de configuration, cette simple extension détecte les pourcentages de codes, commentaires et lignes blanches d'un projet. L'évaluation se lance via la commande de menu 'Extensions' →'Code statistics' .

2.13 Recherche de Code Source Disponible

Cette extension permet de sélectionner un terme dans l'éditeur et de rechercher ce terme à l'aide du menu de contexte 'Rechercher dans Koders' dans la base de données du site [↪Koders]. La boîte de dialogue permet d'ajouter la possibilité de filtrer le langage et le type de licence.

Cette recherche en base de données vous aidera à trouver du code source originaire du monde entier en provenance d'autres projets universitaires, de consortiums et d'organisations comme Apache, Mozilla, Novell Forge, SourceForge et bien d'autres, qui peuvent être utilisés sans avoir à réinventer la roue à chaque fois. SVP, bien observer la licence du code source dans chaque cas individuel.

2.14 Profilage de Code

Une interface graphique simple au Profileur GNU GProf.

2.15 Extension Symbol Table

Cette extension permet de rechercher des symboles dans des fichiers objets et dans des librairies. Les options et le chemin d'accès au programme nm en ligne de commande sont définis dans l'onglet des Options.

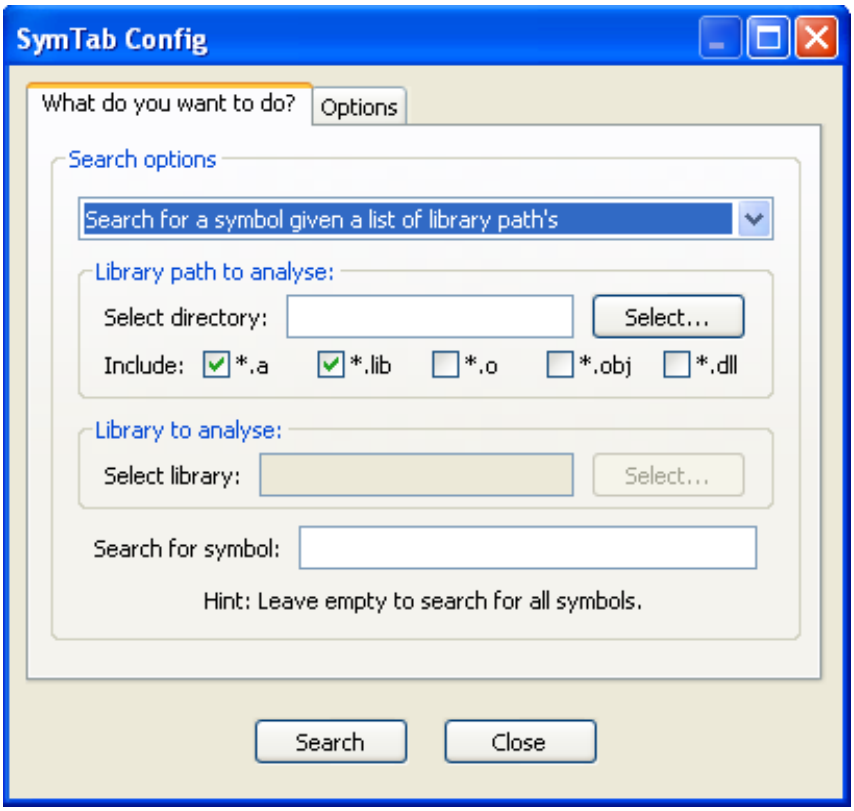


Figure 2.22: Configuration de Symbol Table

Cliquer sur 'Rechercher' démarre la recherche. Les résultats du programme NM sont alors affichés dans une fenêtre séparée nommée 'SymTabs Result'. Les noms des fichiers objets et des librairies contenant les symboles sont listés avec comme titre 'Sortie de NM'.

3 Expansion de Variables

CodeBlocks fait la différence entre plusieurs types de variables. Ces types peuvent servir à configurer l'environnement de création d'un programme, mais aussi à accroître la maintenabilité et la portabilité. L'accès aux variables de CodeBlocks s'obtient grâce à $<name>.

Variables d'Environnement sont configurées au démarrage de CodeBlocks. Elles peuvent modifier les variables d'environnement du système telles que PATH. Cela peut être utile dans les cas où une variable d'environnement spécifique est nécessaire à la création de projets. La configuration des variables d'environnement dans CodeBlocks se fait à l'aide de 'Paramètres' →'Environnement' →'Variables d'environnement' .

Variables internes sont prédéfinies dans CodeBlocks, et peuvent être accédées via leurs noms (voir les détails dans section 3.2 à la page 59).

Macros Commandes Ce type de variables est utilisé pour contrôler le processus de génération. Pour de plus amples informations se référer à section 3.4 à la page 63.

Variables Utilisateur sont des variables définies par l'utilisateur qui peuvent être spécifiées dans les options de génération d'un projet. Ici vous pouvez, par exemple définir votre type de processeur comme une variable MCU et lui assigner une valeur correspondante. Puis entrer dans les options de compilation `-mcpu=$(MCU)`, et CodeBlocks le remplacera automatiquement par le contenu. Par cette méthode, la configuration d'un projet peut être largement paramétrée.

Variables Globales sont surtout utilisées pour créer CodeBlocks à partir des sources ou pour le développement d'applications wxWidgets. Ces variables ont une signification bien particulière. Par rapport à toutes les autres, si vous configurez de telles variables et partagez votre fichier projet avec d'autres qui eux n'ont *pas* configuré ces variables glogales (ou GV), CodeBlocks demandera à l'utilisateur de les configurer. C'est un moyen pratique de d'assurer qu'un 'autre développeur' sait facilement ce qu'il doit configurer. CodeBlocks posera la question pour tous les chemins usuellement nécessaires.

3.1 Syntaxe

CodeBlocks traite de façon équivalente, en tant que variables, les séquences de caractères suivantes dans les étapes de pré-génération, post-génération ou génération :

- $VARIABLE
- $(VARIABLE)
- ${VARIABLE}
- %VARIABLE%

Les noms de variables doivent être composés de caractères alphanumériques et sont insensibles à la casse (minuscules-majuscules). Les variables commençant par un seul signe

dièse (#) sont interprétées comme des variables utilisateur globales (voir les détails dans la section 3.7 à la page 64). Les noms listés ci-dessous sont interprétés comme des types de variables internes.

Les variables qui ne sont ni de type utilisateur globales ni de type interne, seront remplacées par une valeur fournie dans le fichier projet, ou par une variable d'environnement si ce dernier échoue.

> **Note:**
>
> Les définitions par-cible sont prioritaires par rapport aux définitions par-projet.

3.2 Liste des variables internes

Les variables listées ci-dessous sont des variables internes à CodeBlocks. Elles ne peuvent pas être utilisés dans des fichiers sources.

3.2.1 Espace de travail CodeBlocks

`$(WORKSPACE_FILENAME)`, `$(WORKSPACE_FILE_NAME)`, `$(WORKSPACEFILE)`, `$(WORKSPACEFILENAME)`
: Le nom de fichier de l'espace de travail courant (.workspace).

`$(WORKSPACENAME)`, `$(WORKSPACE_NAME)`
: Le nom de l'espace de travail qui est affiché dans l'onglet Projets du panneau Gestion.

`$(WORKSPACE_DIR)`, `$(WORKSPACE_DIRECTORY)`, `$(WORKSPACEDIR)`, `$(WORKSPACEDIRECTORY)`
: Le répertoire où se trouve l'espace de travail.

3.2.2 Fichiers et répertoires

`$(PROJECT_FILENAME)`, `$(PROJECT_FILE_NAME)`, `$(PROJECT_FILE)`, `$(PROJECTFILE)`
: Le nom de fichier du projet en cours de compilation.

`$(PROJECT_NAME)`
: Le nom du projet en cours de compilation.

`$(PROJECT_DIR)`, `$(PROJECTDIR)`, `$(PROJECT_DIRECTORY)`
: Le répertoire commun de plus haut niveau du projet en cours de compilation.

`$(ACTIVE_EDITOR_FILENAME)`
: Le nom du fichier ouvert dans l'éditeur actif courant.

`$(ACTIVE_EDITOR_LINE)`
: Retourne le numéro de ligne courant dans l'éditeur actif.

`$(ACTIVE_EDITOR_COLUMN`
: Retourne le numéro de colonne courant dans l'éditeur actif.

`$(ACTIVE_EDITOR_DIRNAME)`
: le répertoire contenant le fichier actif courant (relatif au chemin de plus haut niveau).

`$(ACTIVE_EDITOR_STEM)`
: Le nom de base (sans extension) du fichier actif courant.

`$(ACTIVE_EDITOR_EXT)`
: L'extension du fichier actif courant.

`$(ALL_PROJECT_FILES)`
: Une chaîne contenant les noms de tous les fichiers du projet courant.

`$(MAKEFILE)`
: Le nom de fichier du makefile.

`$(CODEBLOCKS), $(APP_PATH), $(APPPATH), $(APP-PATH)`
: Le chemin de l'instance courante de CodeBlocks en cours d'exécution.

`$(DATAPATH), $(DATA_PATH), $(DATA-PATH)`
: Le répertoire 'partagé' de l'instance courante de CodeBlocks en cours d'exécution.

`$(PLUGINS)`
: Le répertoire des `plugins` (ou extensions) de l'instance courante de CodeBlocks en cours d'exécution.

`$(TARGET_COMPILER_DIR)`
: Le répertoire d'installation du compilateur appelé aussi chemin maître.

3.2.3 Cibles de génération

`$(FOOBAR_OUTPUT_FILE)`
: Le fichier de sortie d'une cible spécifique.

`$(FOOBAR_OUTPUT_DIR)`
: Le répertoire de sortie d'une cible spécifique.

`$(FOOBAR_OUTPUT_BASENAME)`
: Le nom de base du fichier de sortie (sans chemin, sans extension) d'une cible spécifique.

`$(TARGET_OUTPUT_DIR)`
: Le répertoire de sortie de la cible courante.

`$(TARGET_OBJECT_DIR)`
: Le répertoire objet de la cible courante.

`$(TARGET_NAME)`
: Le nom de la cible courante.

`$(TARGET_OUTPUT_FILE)`
: Le fichier de sortie de la cible courante.

`$(TARGET_OUTPUT_BASENAME)`
: Le nom de base du fichier de sortie (sans chemin, sans extension) de la cible courante.

`$(TARGET_CC), $(TARGET_CPP), $(TARGET_LD), $(TARGET_LIB)`
: L'outil de génération (compilateur, éditeur de liens, etc.) de la cible courante.

3.2.4 Langue et encodage

`$(LANGUAGE)`
: La langue du système en clair.

`$(ENCODING)` L'encodage du système en clair.

3.2.5 Heure et date

`$(TDAY)` Date courante sous la forme AAAAMMJJ (par exemple 20051228)

`$(TODAY)` Date courante sous la forme AAAA-MM-JJ (par exemple 2005-12-28)

`$(NOW)` Heure courante sous la forme AAAA-MM-JJ-hh.mm (par exemple 2005-12-28-07.15)

`$(NOW_L)` Heure courante sous la forme AAAA-MM-JJ-hh.mm.ss (par exemple 2005-12-28-07.15.45)

`$(WEEKDAY)` Nom du jour de la semaine en clair (par exemple 'Mercredi')

`$(TDAY_UTC)`, `$(TODAY_UTC)`, `$(NOW_UTC)`, `$(NOW_L_UTC)`, `$(WEEKDAY_UTC)`
Ces types sont identiques aux précédents mais exprimés en temps universel TU.

`$(DAYCOUNT)` Nombre de jours passés depuis une date arbitraire choisie comme origine (1er Janvier 2009). Utile comme dernier composant d'un numéro de version/génération.

3.2.6 Valeurs aléatoires

`$(COIN)` Cette variable simule un pile ou face (à chaque invocation) et retourne 0 ou 1.

`$(RANDOM)` Un nombre positif aléatoire sur 16 bits (0-65535)

3.2.7 Commandes du Système d'exploitation

La variable est remplacée par la commande effective du Système d'exploitation.

`$(CMD_CP)` Commande de Copie de fichiers.

`$(CMD_RM)` Commande de Suppression de fichiers.

`$(CMD_MV)` Commande de Déplacement de fichiers.

`$(CMD_MKDIR)` Commande de Création de répertoire.

`$(CMD_RMDIR)` Commande de Suppression de répertoire.

3.2.8 Évaluation Conditionnelle

```
$if(condition){clause si vraie}{clause si fausse}
```

L'évaluation Conditionnelle sera considérée comme vraie dans les cas suivants

- la condition est un caractère non vide autre que 0 ou false
- la condition est une variable non vide qui ne se résout pas en 0 ou false

- la condition est une variable qui est évaluée à true (implicite par une condition précédente)

L'évaluation Conditionnelle sera considérée comme fausse dans les cas suivants

- la condition est vide
- la condition est 0 ou false
- la condition est une variable qui est vide ou évaluée à 0 ou false

> **Note:**
>
> Notez SVP que les variantes de syntaxe de variable **%if** (...) ou **$(if)**(...) ne sont pas supportées dans ce type de construction.

Exemple

Par exemple : vous utilisez plusieurs plateformes et vous voulez configurer différents paramètres en fonction du système d'exploitation. Dans le code suivant, la commande de script [[ ]] est évaluée et la <commande> sera exécutée. Ce peut être utile dans une étape de post-génération.

```
[[ if (PLATFORM ==  PLATFORM_MSW) { print (_T("cmd /c")); } else { print (_T("sh ")); }
```

3.3 Expansion de script

Pour une flexibilité maximale, vous pouvez imbriquer les scripts en utilisant l'opérateur [[ ]] en tant que cas particulier d'expansion de variable. Les scripts imbriqués ont accès à toutes les fonctionnalités standard disponibles pour les scripts et se comportent comme des "backticks" (ou apostrophes inversées) de bash (à l'exception de l'accès au namespace de CodeBlocks). En tant que tels, les scripts ne sont pas limités à la production de sorties de type texte, mais peuvent aussi manipuler des états de CodeBlocks (projets, cibles, etc.).

> **Note:**
>
> La manipulation d'états de CodeBlocks devrait être implémentée dans des étapes de pré-générations plutôt que dans un script.

Exemple avec Backticks

```
objdump -D `find . -name *.elf` > name.dis
```

L'expression entre "backticks" (ou apostrophes inversées) retourne une liste de tous les exécutables `*.elf` des sous-répertoires. Le résultat de cette expression peut être utilisé directement par `objdump`. Au final, la sortie est redirigée vers un fichier nommé `name.dis`.

Ainsi, des processus peuvent être automatisés simplement sans avoir recours à aucune boucle.

Exemple utilisant un Script

Le texte du script est remplacé par toute sortie générée par votre script, ou ignoré en cas d'erreur de syntaxe.

Comme l'évaluation conditionnelle est exécutée avant l'expansion de scripts, l'évaluation conditionnelle peut être utilisée pour les fonctionnalités de type pré-processeur. Les variables internes (et les variables utilisateur) sont étendues en sortie de scripts, aussi on peut référencer des variables dans les sorties d'un script.

```
[[ print(GetProjectManager().GetActiveProject().GetTitle()); ]]
```

insère le titre du projet actif dans la ligne de commande.

3.4 Macros Commandes

`$compiler`	Accède au nom de l'exécutable du compilateur.
`$linker`	Accède au nom de l'exécutable de l'éditeur de liens.
`$options`	Flags du Compilateur
`$link_options`	Flags de l'éditeur de liens
`$includes`	Chemins des include du compilateur
`$c`	Chemins des include de l'éditeur de liens
`$libs`	Librairies de l'éditeur de liens
`$file`	Fichier source (nom complet)
`$file_dir`	Répertoire du fichier source sans le nom de fichier ni son extension.
`$file_name`	Nom du fichier source sans les informations de chemin ni l'extension.
`$exe_dir`	Répertoire du fichier exécutable sans le nom de fichier ni son extension.
`$exe_name`	Nom du fichier exécutable sans les informations de chemin ni l'extension.
`$exe_ext`	Extension de l'exécutable sans les informations de chemin ni le nom du fichier.
`$object`	Fichier objet
`$exe_output`	Fichier exécutable de sortie
`$objects_output_dir`	Répertoire de sortie des fichiers objets

3.5 Compilation d'un fichier unique

```
$compiler $options $includes -c $file -o $object
```

3.6 Édition de fichiers objets en exécutable

```
$linker $libdirs -o $exe_output $link_objects $link_resobjects $link_options $libs
```

3.7 Variables globales du compilateur

3.8 Synopsis

Travailler en tant que développeur sur un projet reposant sur des librairies tierces impose un certain nombre de tâches répétitives inutiles, comme configurer des variables de génération dépendantes du système de fichier local. Dans le cas de fichiers projets, une attention toute particulière doit être apportée afin d'éviter de diffuser une copie modifiée localement. Si on n'y prend pas garde, cela peut se produire facilement après avoir changé par exemple un flag de génération pour obtenir une version de type release.

Le concept de variable globale du compilateur est une nouvelle solution unique à CodeBlocks qui adresse ce problème. Les variables globales du compilateur vous permettent de configurer un projet une seule fois, avec n'importe quel nombre de développeurs utilisant n'importe quel système de fichiers pour compiler et développer ce projet. Aucune information locale ne nécessite d'être changée plus d'une fois.

3.9 Noms et Membres

Les variables globales du compilateur dans CodeBlocks se distinguent des variables par-projet par la présence d'un signe dièse en tête. Les variables globales du compilateur sont structurées ; chaque variable consiste en un nom et un membre optionnel. Les noms sont définissables librement, alors que les membres sont construits dans l'Environnement Intégré de Développement (IDE). Bien que vous puissiez choisir n'importe quoi comme nom en principe, il est recommandé de reproduire des identificateurs connus de packages communément utilisés. Ainsi, le nombre d'informations que l'utilisateur doit fournir est minimisé. L'équipe de CodeBlocks fourni une liste de variables recommandées pour divers packages connus.

Le membre base correspond à la même valeur que celle de la variable utilisée sans membre (alias).

Les membres include et lib sont par défaut des aliases pour base/include et base/lib, respectivement. Cependant, l'utilisateur peut les redéfinir si une autre configuration est souhaitée.

Il est généralement recommandé d'utiliser la syntaxe $(#variable.include) plutôt que son équivalent $(#variable)/include, car elle fournit une flexibilité accrue tout en étant fonctionnellement identique (voir sous-section 3.12.1 à la page 67 et Figure 3.1 à la page 65 pour plus de détails).

Les membres cflags et lflags sont vides par défaut et peuvent être utilisés pour fournir une possibilité de remplir un ensemble consistant unique de flags compilateur/éditeur de liens pour toutes les générations sur une machine donnée. CodeBlocks vous permet de définir des membres de variables utilisateur en complément de ceux prédéfinis.

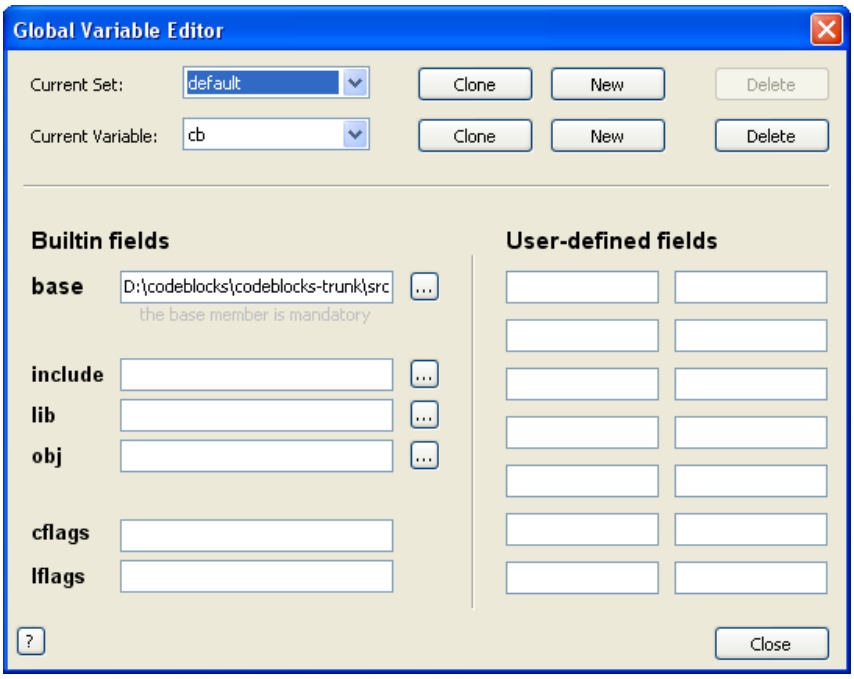


Figure 3.1: Variables Globales d'Environnement

3.10 Contraintes

- Les noms de variables de configuration ou de compilateur ne peuvent pas être vides, ne peuvent pas contenir de caractères blancs (ou espaces), doivent commencer par une lettre et ne contenir que des caractères alphanumériques. Les lettres Cyrilliques ou Chinoises ne sont pas des caractères alphanumériques. Si CodeBlocks rencontre une séquence de caractères non valides dans un nom, il peut la remplacer sans le demander.
- Toutes les variables nécessitent que leur base soit définie. Tout le reste est optionnel, mais la base est absolument obligatoire. Si vous ne définissez pas la base d'une variable, elle ne sera pas sauvegardée (et ce même si les autres champs ont été définis).
- Vous ne pouvez pas définir un nom de membre utilisateur qui a le même nom qu'un membre prédéfini. Actuellement, le membre utilisateur écrasera le membre prédéfini, mais en général, le comportement est indéfini dans ce cas.
- Les valeurs des variables et des membres peuvent contenir un nombre arbitraire de séquences de caractères, mais doivent respecter les contraintes suivantes :
  - Vous ne pouvez pas définir une variable par une valeur qui se référence à la même variable ou à n'importe lequel de ses membres
  - Vous ne pouvez pas définir un membre par une valeur qui se référence à ce même membre
  - Vous ne pouvez pas définir un membre ou une variable par une valeur qui se référence à la même variable ou membre par une dépendance cyclique.

CodeBlocks détectera les cas de définitions récursives les plus évidentes (ce qui peut arriver par accident), mais il ne fera pas d'analyse en profondeur de tous les cas possibles abusifs. Si vous entrez n'importe quoi, alors vous obtiendrez n'importe quoi; vous êtes avertis maintenant.

Exemples

Définir wx.include comme $(#wx)/include est redondant, mais parfaitement légal.

Définir wx.include comme $(#wx.include) est illégal et sera détecté par CodeBlocks.

Définir wx.include comme $(#cb.lib) qui est lui même défini comme $(#wx.include) créera une boucle infinie.

3.11 Utilisation des Variables Globales du Compilateur

Tout ce que vous avez à faire pour utiliser des variables globales de compilateur c'est de les mettre dans votre projet! Oui, c'est aussi simple que cela.

Quand l'Environnement Intégré de Développement (IDE) détecte la présence d'une variable globale inconnue, il vous demande d'entrer sa valeur. La valeur sera sauvegardée dans vos paramètres, ainsi vous n'aurez jamais besoin d'entrer deux fois l'information.

Si vous avez besoin de modifier ou de supprimer une variable plus tard, vous pourrez le faire depuis le menu des paramètres.

Exemple

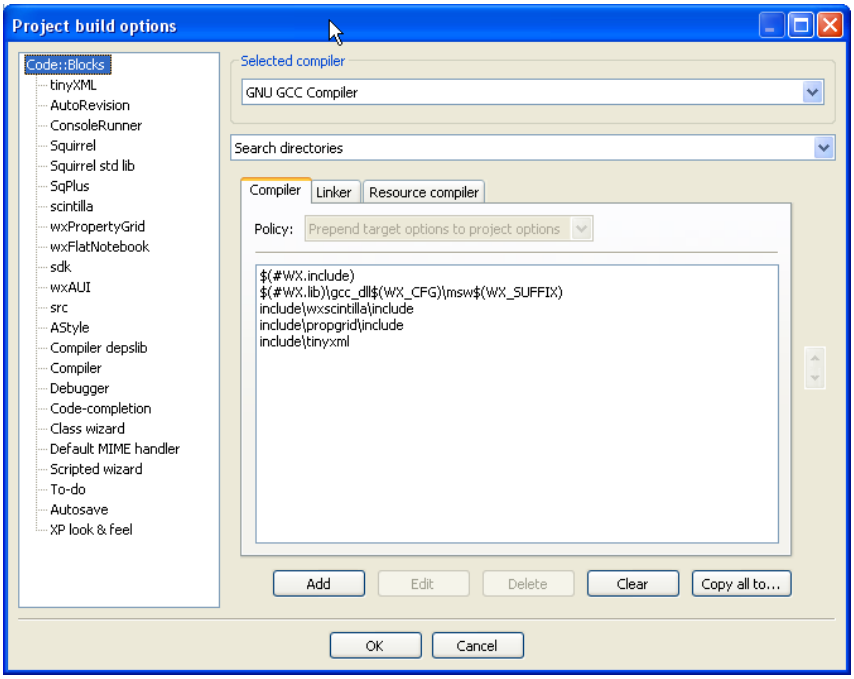


Figure 3.2: Variables Globales

L'image ci-contre montre à la fois les variables par-projet et globales. WX_SUFFIX est défini dans le projet, mais WX est une variable utilisateur globale.

3.12 Ensembles de Variables

Parfois, vous voulez utiliser différentes versions d'une même librairie, ou vous développez deux branches d'un même programme. Bien qu'il soit possible de gérer cela avec une variable globale de compilateur, cela peut devenir fastidieux. Dans ce cas, CodeBlocks supporte des ensembles de variables. Un ensemble de variables est une collection indépendante de variables, identifiée par un nom (les noms d'ensemble ont les mêmes contraintes que les noms de variables).

Si vous souhaitez basculer vers un autre ensemble de variables, vous sélectionnez tout simplement un ensemble différent depuis le menu. Des ensembles différents n'ont pas obligatoirement les mêmes variables, et des variables identiques dans différents ensembles n'ont pas forcément les mêmes valeurs, ni même des membres utilisateurs identiques.

Un autre point positif à propos des ensembles est que si vous avez une douzaine de variables et que vous voulez obtenir un nouvel ensemble avec une de ces variables pointant vers un endroit différent, vous n'êtes pas obligés de ré-entrer toutes les données à nouveau. Vous pouvez simplement créer un clone de l'ensemble courant, ce qui dupliquera toutes vos variables.

Supprimer un ensemble supprimera également toutes les variables de cet ensemble (mais pas celles d'une autre ensemble). L'ensemble `default` est toujours présent et ne peut pas être supprimé.

3.12.1 Mini-Tutoriel de membres utilisateur

Comme décrit auparavant, écrire $(#var.include) et $(#var)/include revient à la même chose par défaut. Aussi pourquoi donc écrire quelque chose d'aussi non intuitif que $(#var.include)?

Prenons l'exemple d'une installation standard de Boost sous Windows. Généralement, vous vous attendriez à ce qu'un package fictif ACME ait ses fichiers include dans ACME/include et ses librairies dans ACME/lib. Optionnellement, il pourrait mettre ses entêtes (headers) dans un autre sous répertoire appelé acme. Ainsi après avoir ajouté les chemins corrects dans les options du compilateur et de l'éditeur de liens, vous pouvez vous attendre à **#include** <acme/acme.h> et éditer les liens avec `libacme.a` (ou quelque chose de ce genre).

Boost, cependant, installe les entêtes dans `C:\Boost\include\boost-1_33_1\boost` et ses librairies dans `C:\Boost\lib` par défaut. Il semble impossible d'obtenir ceci simplement sans devoir tout ajuster sur chaque nouveau PC, particulièrement si vous devez travailler sous Linux mais aussi avec un autre OS.

C'est là que la véritable puissance des variables globales utilisateur se révèle. Quand vous définissez la valeur de la variable #boost, vous allez juste un cran plus loin que d'habitude. Vous définissez le membre include comme `C:\Boost\include\boost-1_33_1\boost` et le membre lib comme `C:\Boost\lib`, respectivement. Votre projet utilisant $(#boost.include) et $(#boost.lib) travaillera comme par magie correctement sur tout PC sans aucune modifications. Vous n'avez pas besoin de savoir pourquoi, vous ne voulez pas savoir pourquoi.

4 Générer CodeBlocks à partir des Sources

4.1 Introduction

Cet article décrit le processus de création des nightly builds (générations nocturnes !), et peut être utilisé comme un guide si vous voulez générer CodeBlocks vous-même. La description suivante est une séquence d'actions.

Afin de procéder à notre tâche de génération, nous aurons besoin de plusieurs outils. Créons d'abord une liste d'ingrédients pour notre recette de cuisine.

- un compilateur
- un système de génération initial (une version précédente déjà fonctionnelle)
- les sources de CodeBlocks
- un programme zip
- svn (système de contrôle de versions)
- wxWidgets

4.1.1 WIN32

Comme les développeurs de CodeBlocks génèrent CodeBlocks en utilisant GCC, nous ferons de même sous Windows. Le portage le plus facile et le plus propre est MinGW. C'est le compilateur distribué avec CodeBlocks quand vous téléchargez le package officiel. Ici, on s'en tiendra à la version 3.4.5, qui fonctionne bien.

D'abord, une brève explication des composants de MinGW :

gcc-core le coeur de la suite GCC

gcc-g++ le compilateur c++

mingw Runtime l'implémentation des librairies "run time"

mingw utils plusieurs utilitaires (implémentation de petits programmes que GCC utilise lui-même)

win32Api l'API (Interface de Programmation d'Application) pour créer des programmes Windows

binutils plusieurs utilitaires utilisés dans l'environnement de génération

make le programme make de Gnu, ainsi vous pouvez générer à partir de fichiers make

GDB le débogueur Gnu

Je vous suggère d'extraire (et d'installer pour GDB) le tout dans un répertoire `C:\MinGW`. Le reste de cet article supposera que c'est là que vous l'avez mis. Si vous avez déjà une installation de CodeBlocks qui a été fournie avec MinGW, je vous recommande malgré tout d'installer MinGW comme décrit ici. Un compilateur n'a pas a être dans l'arborescence d'un Environnement de Développement Intégré (IDE); ce sont deux choses bien distinctes. CodeBlocks le fournit avec les versions officielles afin que l'utilisateur standard n'ait pas à se préoccuper de ce genre de choses.

Vous pouvez avoir besoin d'ajouter le répertoire `bin` de votre installation MinGW à votre variable path. Un moyen simple de faire cela est d'entrer la commande suivante dans une fenêtre DOS :

```
set path=%PATH%;C:\MinGW\bin;C:\MinGW\mingw32\bin;
```

4.1.2 Système de génération initial

Sur [↪CODEBLOCKS] est disponible un fichier de description de projet `CodeBlocks.cbp`. Si vous chargez ce fichier dans CodeBlocks alors vous êtes en mesure de générer CodeBlocks à partir des sources. Tout ce dont vous avez besoin c'est d'une version de CodeBlocks déjà pré-générée.

Premièrement, téléchargez une version nightly. Vous pouvez faire votre sélection à partir de là ([↪FORUM] rubrique Nightly Builds). Les versions nightly sont des versions Unicode, contenant le coeur et les plugins contributifs.

Ensuite, décompressez le fichier 7-zip dans n'importe quel répertoire de votre choix. Si vous n'avez pas 7-zip, vous pouvez le télécharger gratuitement depuis [↪7Z].

Maintenant, CodeBlocks nécessite une `dll` supplémentaire pour travailler correctement: la dll wxWidgets. Vous pouvez aussi la télécharger depuis le forum des nightly builds. Dézippez là simplement dans le même répertoire que celui où vous avez décompressé la version nightly de CodeBlocks. Il vous faut aussi la dll `mingwm10.dll`. Elle est normalement dans le répertoire bin de votre installation de MinGW. C'est pourquoi il est important de s'assurer que le sous répertoire bin de votre installation MinGW est bien dans votre variable path.

Enfin, démarrer cette nouvelle génération d'une nightly de CodeBlocks. Elle devrait trouver le compilateur MinGW qui vient d'être installé.

4.1.3 Système de Contrôle de Versions

Afin de pouvoir récupérer les dernières sources de CodeBlocks sources, nous avons besoin d'installer un système de contrôle de versions.

Les développeurs de CodeBlocks fournissent leurs sources par le biais du système de contrôle de versions [↪Subversion]. Aussi, nous avons besoin d'un client pour accéder à leur dépôt svn des sources. [↪TortoiseSVN] est un bon client pour Windows, facile d'utilisation, et qui est disponible gratuitement. Téléchargez le et installez le, en gardant tous les paramètres suggérés.

Maintenant, créez un répertoire où vous voulez, par exemple `D:\projets\CodeBlocks`. Faire un clic droit dans ce répertoire et choisir dans le menu popup : svn-checkout (ou SVN Extraire si vous avez installé la francisation de SVN). Dans la boîte de dialogue qui apparaît, entrez l'information suivante dans Url of Repository (URL du référentiel) :

`svn://svn.berlios.de/codeblocks/trunk`

et laissez les autres paramètres comme ils sont.

Maintenant, soyez patient pendant que TortoiseSVN récupère les sources les plus récentes du dépôt de CodeBlocks dans votre répertoire local. Oui ; toutes ces sources de CodeBlocks viennent chez vous !

Pour plus d'informations sur le paramétrage de SVN, voir info dans SVN settings. Si vous n'aimez pas l'intégration dans l'explorateur ou cherchez une solution inter-plateforme vous pouvez jeter un oeil sur RapidSVN..

4.1.4 wxWidgets

[↪Wxwidgets] est une abstraction de plateforme qui fourni une API supportant de nombreuses choses comme une Interface Graphique Utilisateur (GUI), des sockets, fichiers, fonctionnalités de registres. En utilisant cette API, vous pouvez créer des programmes indépendants des plateformes.

CodeBlocks est une application wxWidgets (soit après : wx), ce qui signifie que pour exécuter CodeBlocks vous avez besoin des fonctionnalités wx. Cela peut se faire de deux façons. Soit par une `.dll` soit par une librairie statique. CodeBlocks utilise wx en tant que dll et cette dll peut aussi se télécharger depuis la section nightly builds du forum.

Néanmoins, si nous voulons générer une application wx, nous devons inclure les headers (ou en-têtes) des sources wx. Elles donnent au compilateur les informations sur les fonctionnalités de wx. En plus de ces fichiers de headers, notre application a besoin d'être liée aux librairies d'importation wx. Bien, voyons cela pas à pas.

Wx est fourni sous forme de fichiers sources dans un fichier zip, aussi, nous devons le générer par nous-mêmes. Nous avons déjà le compilateur MinGW, donc nous avons tous les outils nécessaires sous la main.

Dézippez maintenant les sources wx dans `C:\Projets` et ainsi nous aurons au final un répertoire wx de base comme celui-ci : `C:\Projets\wxWidgets-2.8.9`. Ensuite, dézippez les patchs (s'il y en a !) dans le même sous répertoire afin de remplacer les fichiers modifiés. Notez que nous allons référencer le répertoire wx de base à partir de maintenant comme <wxDir>

Maintenant, nous allons générer les wxWidgets. Voici comment faire :

Premièrement, assurez-vous que `C:\MingGW\bin` est dans votre variable path, car durant la génération quelques programmes résidant dans le répertoire MinGW\bin seront appelés. De plus, Make doit être en version 3.80 ou supérieure.

Il est temps maintenant de compiler les wxWidgets. Ouvrir une fenêtre de commande DOS et se placer dans le répertoire des wxWidgets :

```
cd <wxDir>\build\msw
```

Nous sommes maintenant au bon endroit. Nous allons d'abord nettoyer les sources :

```
mingw32-make -f makefile.gcc SHARED=1 MONOLITHIC=1 BUILD=release UNICODE=1 clean
```

Maintenant que tout est propre, nous pouvons compiler les wxWidgets :

```
mingw32-make -f makefile.gcc SHARED=1 MONOLITHIC=1 BUILD=release UNICODE=1
```

Cela prend un certain temps.

Pour générer une version debug, suivez ces étapes :

- Nettoyer les précédentes compilations par

  ```
  mingw32-make -f makefile.gcc SHARED=1 MONOLITHIC=1 BUILD=debug UNICODE=1 clean
  ```

- Compiler par

  ```
  mingw32-make -f makefile.gcc SHARED=1 MONOLITHIC=1 BUILD=debug UNICODE=1
  ```

Bien, regardons maintenant dans le répertoire (`<wxDir>\lib\gcc_dll`). Les librairies d'importation et les dll y sont visibles et il doit aussi y avoir un sous répertoire `mswu\wx` à cet endroit, contenant `setup.h`.

Bravo! Vous venez de générer les wxWidgets!

Faisons encore quelques tâches préliminaires complémentaires avant d'attaquer la compilation de CodeBlocks.

4.1.5 Zip

Durant la génération de CodeBlocks, plusieurs ressources seront compressées dans des fichiers zip. Aussi, le processus de génération doit pouvoir accéder à un zip.exe. Nous devons télécharger ce zip.exe et le mettre quelque part dans notre path. `MingW\bin` est un bon endroit pour cela.

Vous pouvez depuis ce site (http://www.info-zip.org/pub/infozip/Zip.html) télécharger zip.exe gratuitement et ceci (http://switch.dl.sourceforge.net/sourceforge/infozip/zip232.zip) est un lien direct(32bit) vers la plus récente version au moment de la rédaction de cet article.

Une fois téléchargé, extraire simplement zip.exe vers l'endroit approprié.

4.1.6 Générer Codeblocks

win32 Lancer `update_revision.bat` (NdT : n'est plus dans les versions récentes !)

linux Lancer `update_revision.sh`

Avec cette fonction la révision SVN de la génération Nightly est mise à jour dans les sources. On peut trouver ce fichier dans le répertoire principal des sources de CodeBlocks.

4.1.7 WIN32

Maintenant, ouvrez le projet `CodeBlocks.cbp` dans CodeBlocks. Générez CodeBlocks en lançant le processus de génération. Après la création de CodeBlocks, les fichiers générés avec les informations de débogage se trouvent dans le sous-répertoire `devel`. En appelant le fichier batch `update.bat` depuis le répertoire source, les fichiers sont copiés dans le sous-répertoire `output` et les informations de débogage sont retirées.

4.1.8 LINUX

Quand on génère sous Linux, les étapes suivantes sont nécessaires. Dans cet exemple nous supposons que vous êtes dans le répertoire source de CodeBlocks. Sous Linux, la variable d'environnement PKG_CONFIG_PATH doit être configurée. Le répertoire <prefix> doit contenir le fichier `codeblocks.pc`. (NdT : Cette section ne semble pas à jour avec les dernières versions)

```
PKG_CONFIG_PATH=$PKG_CONFIG_PATH:<prefix>
```

```
sh update_revsion.sh
./bootstrap
./configure --with-contrib=[all | noms des plugins séparés par des virgules]
--prefix=<install-dir>
make
make install (root)
```

4.1.9 Générer des plugins

Ensuite, configurer les variables globales via 'Paramètres' →'Variables Globales' .

Variable cb

Pour la variable cb, entrer dans base le répertoire source de CodeBlocks.

```
<prefix>/codeblocks/src
```

Variable wx

Pour la variable wx, entrer dans base le répertoire source de wx (par ex.

```
C:\Programme\wxWidgets-2.8.9
```

Dans le projet CodeBlocks, la variable projet WX_SUFFIX est configurée à `u`. Cela signifie que pendant la génération de CodeBlocks l'édition de liens se fera avec la librairie `*u_gcc_custom.dll`. Les générations officielles des nightly de CodeBlocks seront liées avec `gcc_cb.dll`. Pour ce faire, il faut faire comme suit.

```
gcc_<VENDOR>.dll
```

La variable <VENDOR> est donnée dans le fichier de configuration `compiler.gcc`. Pour s'assurer qu'une distinction soit possible entre une génération officielle de CodeBlocks et celles effectuée par vous-mêmes, la configuration par défaut VENDOR=custom ne devrait pas être changée.

Après, créez l'espace de travail `ContribPlugins.cbp` via 'Projet' →'Générer l'espace de travail' . Puis exécutez une fois de plus `update.bat`.

4.1.10 Linux

Configurez la variable wx à l'aide des variables globales.

base /usr

include /usr/include/wx-2.8

lib /usr/lib

déboguer CodeBlocks. Démarrez CodeBlocks dans le répertoire output et chargez `CodeBlocks.cbp` en tant que projet. Entrer le point d'arrêt et démarrer par 'Debug and Run' (f8).

4.1.11 Plugins Contributifs

Ceci nous amène à la dernière tâche préliminaire. Le code de CodeBlocks peut se diviser en 2 parties principales : le coeur avec les plugins internes et les plugins contributifs. Vous devez toujours d'abord générer la partie coeur/plugins internes avant de générer la partie contributive.

Pour générer la partie interne, Vous pouvez utiliser le fichier de projet de CodeBlocks que vous trouverez dans : <`cbDir`>\`src`\`CodeBlocks.cbp`. Notre répertoire maître de CodeBlocks sera maintenant mentionné comme <cbDir>. Un espace de travail est quelque chose qui regroupe plusieurs projets entre eux. La génération des plugins contributifs se trouve dans

```
<cbDir>\src\ContribPlugins.workspace
```

Mais, créons un espace de travail contenant l'ensemble. Plaçons cet espace de travail dans le répertoire maître <cbDir> (NdT : là où il y a déjà CodeBlocks.cbp et ContribPlugins.workspace). Utiliser simplement un éditeur de texte normal et créer un fichier dont le nom est `CbProjects.workspace` puis le remplir avec le contenu suivant (NdT : le fichier ci-dessous est différent de la version originale anglaise, car mis à jour au moment de la traduction. En fait, il s'agit tout simplement du fichier ContribPlugins.workspace auquel on a changé le titre du Workspace (3ème ligne), ajouté une ligne, la 4ème, pour générer en tout premier le projet CodeBlocks.cbp et déplacé l'option active="1") :

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<CodeBlocks_workspace_file>
  <Workspace title="Workspace">
    <Project filename="CodeBlocks.cbp" active="1" />
    <Project filename="plugins\contrib\AutoVersioning\AutoVersioning.cbp" />
    <Project filename="plugins\contrib\BrowseTracker\BrowseTracker.cbp" />
    <Project filename="plugins\contrib\Cccc\Cccc.cbp" />
    <Project filename="plugins\contrib\CppCheck\CppCheck.cbp" />
    <Project filename="plugins\contrib\codesnippets\codesnippets.cbp" />
    <Project filename="plugins\contrib\codestat\codestat.cbp" />
    <Project filename="plugins\contrib\copystrings\copystrings.cbp" />
    <Project filename="plugins\contrib\Cscope\Cscope.cbp" />
    <Project filename="plugins\contrib\devpak_plugin\DevPakPlugin.cbp" />
    <Project filename="plugins\contrib\DoxyBlocks\DoxyBlocks.cbp" />
    <Project filename="plugins\contrib\dragscroll\dragscroll.cbp" />
    <Project filename="plugins\contrib\EditorTweaks\EditorTweaks.cbp" />
    <Project filename="plugins\contrib\envvars\envvars.cbp" />
```

```
    <Project filename="plugins\contrib\source_exporter\Exporter.cbp" />
    <Project filename="plugins\contrib\byogames\byogames.cbp" />
    <Project filename="plugins\contrib\headerfixup\headerfixup.cbp" />
    <Project filename="plugins\contrib\help_plugin\help-plugin.cbp" />
    <Project filename="plugins\contrib\HexEditor\HexEditor-win.cbp" />
    <Project filename="plugins\contrib\IncrementalSearch\IncrementalSearch.cbp" />
    <Project filename="plugins\contrib\keybinder\keybinder.cbp" />
    <Project filename="plugins\contrib\cb_koders\cb_koders.cbp" />
    <Project filename="plugins\contrib\lib_finder\lib_finder.cbp">
      <Depends filename="plugins\contrib\wxSmithContribItems\wxSmithContribItems.cbp" />
    </Project>
    <Project filename="plugins\contrib\MouseSap\MouseSap.cbp" />
    <Project filename="plugins\contrib\NassiShneiderman\NassiShneiderman.cbp" />
    <Project filename="plugins\contrib\profiler\cbprofiler.cbp" />
    <Project filename="plugins\contrib\regex_testbed\RegExTestbed.cbp" />
    <Project filename="plugins\contrib\symtab\symtab.cbp" />
    <Project filename="plugins\contrib\ThreadSearch\ThreadSearch.cbp">
      <Depends filename="plugins\contrib\wxSmithContribItems\wxSmithContribItems.cbp" />
    </Project>
    <Project filename="plugins\contrib\wxSmithContribItems\wxSmithContribItems.cbp">
      <Depends filename="plugins\contrib\wxSmith\wxSmith.cbp" />
    </Project>
    <Project filename="plugins\contrib\wxSmith\wxSmith.cbp" />
    <Project filename="plugins\contrib\wxSmithAui\wxSmithAui.cbp">
      <Depends filename="plugins\contrib\wxSmith\wxSmith.cbp" />
    </Project>
    <Project filename="tools\cb_share_config\cb_share_config.cbp" />
  </Workspace>
</CodeBlocks_workspace_file>
```

Nous utiliserons cet espace de travail pour générer l'ensemble de CodeBlocks.

4.1.12 Générer l'ensemble de CodeBlocks

Nous sommes arrivés à l'étape finale ; notre but ultime. Lancer l'exécutable de CodeBlocks depuis notre téléchargement de génération nightly. Choisir Ouvrir dans le menu Fichier et rechercher l'espace de travail que nous venons de créer puis l'ouvrir. Soyez un peu patient pendant que CodeBlocks analyse le tout, puis CodeBlocks vous demandera d'entrer 2 variables globales, ces variables globales indiqueront à la version nightly de CodeBlocks où trouver les wxWidgets (rappelez-vous : fichiers header et librairies) et où trouver .... CodeBlocks, ce qui est nécessaire aux plugins contributifs qui ont besoin de savoir (ainsi que pour tout plugin créé par un utilisateur) où est le sdk (fichiers d'entêtes (headers) de CodeBlocks) are. Dans notre cas, ces valeurs sont :

wx <wxDir> répertoire de base des wxWidgets.

cb <cbDir>/src répertoire contenant les sources de CodeBlocks.

Aller maintenant dans le Menu Projet et choisissez (re)générer l'espace de travail, et allez faire un tour. Regardez comment CodeBlocks est en train de générer CodeBlocks.

Une fois la génération terminée, ouvrez une fenêtre console dans <cbDir>/src et lancez la commande update.bat. Cela transfèrera tout ce qui est utile depuis <cbDir>/src/devel

vers <cbDir>/src/output. En plus, cela supprimera tous les symboles de déboguage. Cette étape est très importante - ne l'oubliez jamais.

Vous pouvez maintenant copier la dll wx à la fois dans ce répertoire output et dans devel.

Vous pouvez alors fermer CodeBlocks. Rappelez-vous, nous étions avec la version nightly téléchargée ?

Il est temps de tester la nouvelle. Dans le répertoire output, lancez CodeBlocks.exe. Si tout s'est bien passé, vous avez généré votre propre nightly de CodeBlocks faite maison.

URL catalog

[↪7Z] 7z zip homepage.
http://www.7-zip.org

[↪ARM] ARM homepage.
http://www.arm.com/

[↪BERLIOS] Codeblocks at berlios.
http://developer.berlios.de/projects/codeblocks/

[↪FORUM] Codeblocks forum.
http://forums.codeblocks.org/

[↪WIKI] Codeblocks wiki.
http://wiki.codeblocks.org/index.php?title=Main_Page/

[↪CODEBLOCKS] Codeblocks homepage.
http://www.codeblocks.org/

[↪GCC] GCC home page.
http://gcc.gnu.org/

[↪HIGHTEC] HighTec homepage.
http://www.hightec-rt.com/

[↪Koders] Koders homepage.
http://www.koders.com/

[↪MSP430] MSP430 homepage.
http://www.ti.com/sc/msp430

[↪PowerPC] Motorola homepage.
http://e-www.motorola.com/

[↪TriCore] TriCore homepage.
http://www.infineon.com/tricore/

[↪TortoiseSVN] TriCore homepage.
http://tortoisesvn.net/

[↪Subversion] TriCore homepage.
http://subversion.tigris.org/

[↪Wxwidgets] WxWidgets homepage.
http://www.wxwidgets.org/

[↪Wxcode] WxCode homepage.
http://wxcode.sourceforge.net/